IDS575 HW1

Scott Brewer 676075252

Ono Gantsog 672164839

ESLII Ex. 2.8

Compare the classification performance of linear regression and k- nearest neighbor classification on the zipcode data. In particular, consider only the 2's and 3's, and k = 1, 3, 5, 7 and 15. Show both the training and test error for each choice. The zipcode data are available from the book website www-stat.stanford.edu/ElemStatLearn.

Import train and test zipcode data, rename target variable as Y, and check for NAs:

```
setwd("/Users/cymorene/Documents/UIC/2018Spring/IDS575/HW1")
train <- read.table('zip.train')
test <- read.table('zip.test')
names(train)[names(train)=="V1"] <- "Y"
names(test)[names(test)=="V1"] <- "Y"
sum(is.na(train))

## [1] 0

sum(is.na(test))

## [1] 0
```

Subset train and test data to only include Y = 2 or 3 to reduce to a binary classification problem. Assign Y variables for later use:

```
train23 <- subset(train, (Y == 2) | (Y == 3))
test23 <- subset(test, (Y == 2) | (Y == 3))
actual_train23 <- train23$Y
actual_test23 <- test23$Y
```

Develop linear regression model on subset of training data and use model to predict Y using the same data:

```
zip.lm <- lm(Y ~ ., data = train23)
pred_train23 <- predict(zip.lm,train23)
```

Adjust predicted values to factorized values of 2 or 3.

```
pred_train23[pred_train23 < 2.5] <- 2
pred_train23[pred_train23 >= 2.5] <- 3
```

Calculate confusion matrix for linear model using predictions on train data, then calculate error rate:

```
table(pred_train23, actual_train23)

##              actual_train23
## pred_train23   2   3
##             2 728   5
##             3   3 653
```

```
1 - mean(pred_train23 == actual_train23) # Train error rate

## [1] 0.005759539
```

Use linear model from above to predict on test set

```
pred_test23 <- predict(zip.lm, test23)
```

Adjust predicted values to factorized values of 2 or 3.

```
pred_test23[pred_test23 < 2.5] <- 2
pred_test23[pred_test23 >= 2.5] <- 3
```

Calculate confusion matrix for linear model using predictions on train data, then calculate error rate:

```
table(pred_test23, actual_test23)

##             actual_test23
## pred_test23   2   3
##           2 191   8
##           3   7 158

1 - mean(pred_test23 == actual_test23) # Test error rate

## [1] 0.04120879
```

Comparing the lm train error and test error, we see the expected result of very low error on the train data and higher error on the test data. That said, the test error is quite low at ~4.1%.

Develop KNN models for K=1,3,5,7,15 and calculate error for each on training and test data. Compare to error from linear model

```
library(class) # Required for knn function
```

Train Data KNN

Compute knn models for each k value using training data for model source and predictions:

```
knn1_train <- knn(train = train23[,-1], test = train23[,-1], k=1, cl=as.factor(train23$Y)
)
knn3_train <- knn(train = train23[,-1], test = train23[,-1], k=3, cl=as.factor(train23$Y)
)
knn5_train <- knn(train = train23[,-1], test = train23[,-1], k=5, cl=as.factor(train23$Y)
)
knn7_train <- knn(train = train23[,-1], test = train23[,-1], k=7, cl=as.factor(train23$Y)
)
knn15_train <- knn(train = train23[,-1], test = train23[,-1], k=15, cl=as.factor(train23$
Y))
```

Compute confusion matrices and error rates for each k value model:

```
table(knn1_train, actual_train23)

##            actual_train23
## knn1_train   2   3
##          2 731   0
##          3   0 658

1 - mean(knn1_train == actual_train23) # Train error rate
```

```
## [1] 0

table(knn3_train, actual_train23)

##            actual_train23
## knn3_train   2   3
##          2 728   4
##          3   3 654

1 - mean(knn3_train == actual_train23) # Train error rate

## [1] 0.005039597

table(knn5_train, actual_train23)

##            actual_train23
## knn5_train   2   3
##          2 726   3
##          3   5 655

1 - mean(knn5_train == actual_train23) # Train error rate

## [1] 0.005759539

table(knn7_train, actual_train23)

##            actual_train23
## knn7_train   2   3
##          2 725   3
##          3   6 655

1 - mean(knn7_train == actual_train23) # Train error rate

## [1] 0.006479482

table(knn15_train, actual_train23)

##             actual_train23
## knn15_train   2   3
##           2 721   3
##           3  10 655

1 - mean(knn15_train == actual_train23) # Train error rate

## [1] 0.009359251
```

As expected, error rates for K=1 on the train data are 0 as each point essentially predicts itself. Then, error on the training data rises from K=1 to K=15 as expected as the model goes from overfitting to more general.

Test Data KNN

Compute knn models for each k value using training data for model source and predictions:

```
knn1_test <- knn(train = train23[,-1], test = test23[,-1], k=1, cl=as.factor(train23$Y))
knn3_test <- knn(train = train23[,-1], test = test23[,-1], k=3, cl=as.factor(train23$Y))
knn5_test <- knn(train = train23[,-1], test = test23[,-1], k=5, cl=as.factor(train23$Y))
knn7_test <- knn(train = train23[,-1], test = test23[,-1], k=7, cl=as.factor(train23$Y))
knn15_test <- knn(train = train23[,-1], test = test23[,-1], k=15, cl=as.factor(train23$Y)
)
```

Compute confusion matrices and error rates for each k value model:

```
table(knn1_test, actual_test23)

##          actual_test23
## knn1_test   2   3
##         2 192   3
##         3   6 163

1 - mean(knn1_test == actual_test23)

## [1] 0.02472527

table(knn3_test, actual_test23)

##          actual_test23
## knn3_test   2   3
##         2 191   4
##         3   7 162

1 - mean(knn3_test == actual_test23)

## [1] 0.03021978

table(knn5_test, actual_test23)

##          actual_test23
## knn5_test   2   3
##         2 191   4
##         3   7 162

1 - mean(knn5_test == actual_test23)

## [1] 0.03021978

table(knn7_test, actual_test23)

##          actual_test23
## knn7_test   2   3
##         2 189   3
##         3   9 163

1 - mean(knn7_test == actual_test23)

## [1] 0.03296703

table(knn15_test, actual_test23)

##           actual_test23
## knn15_test   2   3
##          2 187   3
##          3  11 163

1 - mean(knn15_test == actual_test23)

## [1] 0.03846154
```

On the test data, the error rates are worse than the same train knn as expected. That said, the knn test error rates are all under 4% beating the earlier linear model error of 4.1%. In this case for 2 and 3 digits, the range of k values tested indicates that a lower k value should be used for the best error rates as K=1 had and error of 2.5% and K=15 had and error of 3.8% with the error values rising with K.

ISLR CH 2. Ex 5

What are the advantages and disadvantages of a very flexible (versus a less flexible) approach for regression or classification? Under what circumstances might a more flexible approach be preferred to a less flexible approach? When might a less flexible approach be preferred?

Part a:

A less flexible approach such as linear model is accurate when the underlying relationships are linear, but linear models naturally have high bias and low variance which may not be desired. A more flexible approach has an advantage because it can closely estimate an non-linear relationship but it will naturally have a high variance. Often, a less flexible approach is simpler and more interpretable, but at the loss of overall accuracy. Alternatively, more flexible models can be less interpretable, but more accurate. Additionally, the training error declines while the test error doesn't always decline.

Part b:

A more flexible approach such as knn is preferred when there is modeling for prediction (ie not trying to determine a relationship). When there is a need for accurate predictions for response versus actual values and an interpretable result is not as important, flexible approaches are better than less flexible models such as linear regression.

Linear models (inflexible) methods are mostly good at explaining how certain predictors are affecting the target variable, which is modeling for inference. When interpretability is very important and accuracy can suffer as a result of better interpretability, then a less-flexible approach may be more appropriate.

ISLR Ch 2. Ex 7

The table below provides a training data set containing six observations, three predictors, and one qualitative response variable. Obs. X1 X2 X3 Y 1 0 3 0 Red 2 2 0 0 Red 3 0 1 3 Red 4 0 1 2 Green 5 -1 0 1 Green 6 1 1 1 Red Suppose we wish to use this data set to make a prediction for Y when X1 = X2 = X3 = 0 using K-nearest neighbors. (a) Compute the Euclidean distance between each observation and thetestpoint,X1 =X2 =X3 =0.
(b) What is our prediction with K = 1? Why?
(c) What is our prediction with K = 3? Why?
(d) If the Bayes decision boundary in this problem is highly non- linear, then would we expect the best value for K to be large or small? Why?

Create data object

```
X <- c(0,2,0,0,-1,1,3,0,1,1,0,1,0,0,3,2,1,1)
Y <- c('Red','Red','Red','Green','Green','Red')
data <- matrix(c(X,Y), nrow = 6, ncol = 4)
data <- as.data.frame(data, stringsAsFactors=FALSE)
data$V1 <- as.numeric(data$V1)
data$V2 <- as.numeric(data$V2)
data$V3 <- as.numeric(data$V3)
```

Part a: Calculate Euclidean distance from (0,0,0) to each observation:

```
d <- c()
for (i in 1:nrow(data)){
  a <- data[i,1:3]
  b <- c(0,0,0)
  d[i] <- sqrt(sum((a-b)^2))
}
d

## [1] 3.000000 2.000000 3.162278 2.236068 1.414214 1.732051
```

Part b: Calculate and explain the prediction for (0,0,0) at K=1:

From part a, observation 5 is nearest to (0,0,0) with a distance of 1.414 thus, the prediction for K=1 is Green.

Part c: Calculate and explain the prediction for (0,0,0) at K=3.

From part a, observations 2, 5, and 6 are nearest to (0,0,0) with distances of 2.000, 1.414, and 1.732 respectively. The classifications of the these points are Red, Green, and Red, therefore for K=3, the majority class is Red, which will be the prediction for (0,0,0) at K=3.

Part d: If the Bayes decision boundary is highly non-linear, would the best value for K be large or small? Why?

Smaller values of K tend to yield more non-linear decision boundaries, with many boundaries around localized pockets at very low values (ie overfitting training data) while larger values of K tend toward more and more linear boundaries, therefore to mimic a non-linear Bayes decision boundary, the expected value for K would be small.

ISLR Ch 2. Ex. 10

This exercise involves the Boston housing data set.
(a) To begin, load in the Boston data set. The Boston data set is part of the MASS library in R. How many rows are in this data set? How many columns? What do the rows and columns represent?
(b) Make some pairwise scatterplots of the predictors (columns) in this data set. Describe your findings.
(c) Are any of the predictors associated with per capita crime rate? If so, explain the relationship.
(d) Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.
(e) How many of the suburbs in this data set bound the Charles river?
(f) What is the median pupil-teacher ratio among the towns in this data set?
(g) Which suburb of Boston has lowest median value of owner- occupied homes? What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.
(h) In this data set, how many of the suburbs average more than seven rooms per dwelling? More than eight rooms per dwelling? Comment on the suburbs that average more than eight rooms per dwelling.

Part a. How many rows are in this data set? How many columns?

Load data:

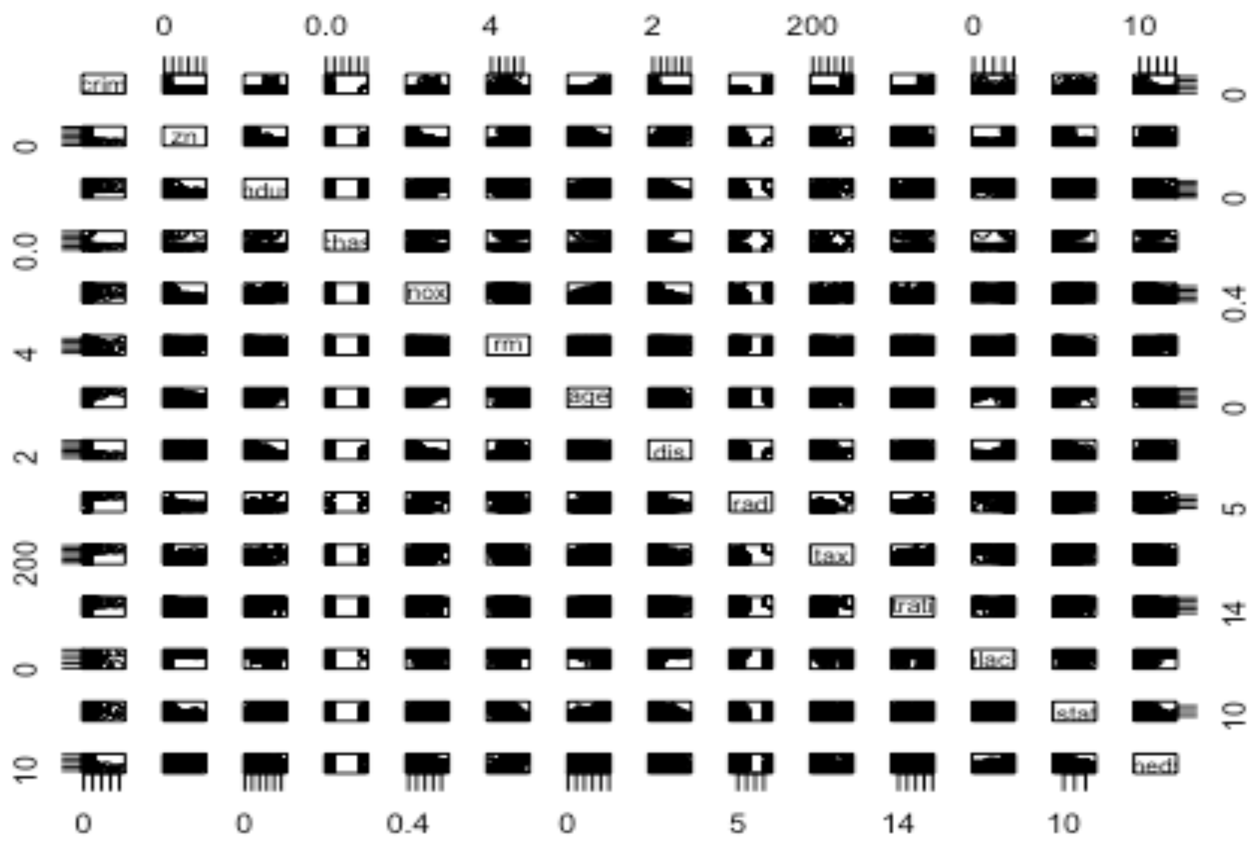
```r
library(MASS)
Boston <- Boston
```

Part a: What do the rows and columns represent?

The Boston data frame has 506 rows and 14 columns. Each column is a feature, each row is a town crim - per capita crime rate by town. zn - proportion of residential land zoned for lots over 25,000 sq.ft. indus - proportion of non-retail business acres per town. chas - Charles River dummy variable (= 1 if tract bounds river; 0 otherwise). nox - nitrogen oxides concentration (parts per 10 million). rm - average number of rooms per dwelling. age - proportion of owner-occupied units built prior to 1940. dis-weighted mean of distances to five Boston employment centres. rad - index of accessibility to radial highways. tax - full-value property-tax rate per $10,000. ptratio - pupil-teacher ratio by town. black - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town. lstat - lower status of the population (percent). medv - median value of owner-occupied homes in $1000s.

Part b: Make some pairwise scatterplots of the predictors (columns) in this data set. Describe your findings.

```r
pairs(Boston)
```



From the plots above, we can see the following relationships: nox and age are positively correlated. nox and dis are negatively correlated. rm and lstat are negatively correlated rm and medv are positively correlated. lstat and medv are negatively correlated.

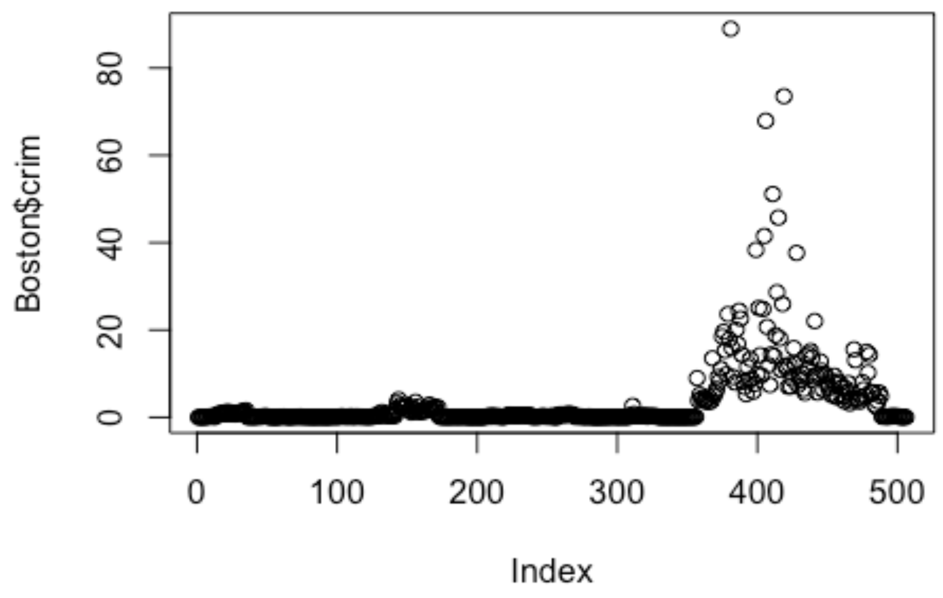
Part c Are any of the predictors associated with per capita crime rate? If so, explain the relationship.

```
cormat <- cor(Boston)
cormat

##                crim          zn       indus         chas         nox
## crim     1.00000000 -0.20046922  0.40658341 -0.055891582  0.42097171
## zn      -0.20046922  1.00000000 -0.53382819 -0.042696719 -0.51660371
## indus    0.40658341 -0.53382819  1.00000000  0.062938027  0.76365145
## chas    -0.05589158 -0.04269672  0.06293803  1.000000000  0.09120281
## nox      0.42097171 -0.51660371  0.76365145  0.091202807  1.00000000
## rm      -0.21924670  0.31199059 -0.39167585  0.091251225 -0.30218819
## age      0.35273425 -0.56953734  0.64477851  0.086517774  0.73147010
## dis     -0.37967009  0.66440822 -0.70802699 -0.099175780 -0.76923011
## rad      0.62550515 -0.31194783  0.59512927 -0.007368241  0.61144056
## tax      0.58276431 -0.31456332  0.72076018 -0.035586518  0.66802320
## ptratio  0.28994558 -0.39167855  0.38324756 -0.121515174  0.18893268
## black   -0.38506394  0.17552032 -0.35697654  0.048788485 -0.38005064
## lstat    0.45562148 -0.41299457  0.60379972 -0.053929298  0.59087892
## medv    -0.38830461  0.36044534 -0.48372516  0.175260177 -0.42732077
##                  rm         age         dis          rad         tax
## crim    -0.21924670  0.35273425 -0.37967009  0.625505145  0.58276431
## zn       0.31199059 -0.56953734  0.66440822 -0.311947826 -0.31456332
## indus   -0.39167585  0.64477851 -0.70802699  0.595129275  0.72076018
## chas     0.09125123  0.08651777 -0.09917578 -0.007368241 -0.03558652
## nox     -0.30218819  0.73147010 -0.76923011  0.611440563  0.66802320
## rm       1.00000000 -0.24026493  0.20524621 -0.209846668 -0.29204783
## age     -0.24026493  1.00000000 -0.74788054  0.456022452  0.50645559
## dis      0.20524621 -0.74788054  1.00000000 -0.494587930 -0.53443158
## rad     -0.20984667  0.45602245 -0.49458793  1.000000000  0.91022819
## tax     -0.29204783  0.50645559 -0.53443158  0.910228189  1.00000000
## ptratio -0.35550149  0.26151501 -0.23247054  0.464741179  0.46085304
## black    0.12806864 -0.27353398  0.29151167 -0.444412816 -0.44180801
## lstat   -0.61380827  0.60233853 -0.49699583  0.488676335  0.54399341
## medv     0.69535995 -0.37695457  0.24992873 -0.381626231 -0.46853593
##            ptratio       black      lstat       medv
## crim     0.2899456 -0.38506394  0.4556215 -0.3883046
## zn      -0.3916785  0.17552032 -0.4129946  0.3604453
## indus    0.3832476 -0.35697654  0.6037997 -0.4837252
## chas    -0.1215152  0.04878848 -0.0539293  0.1752602
## nox      0.1889327 -0.38005064  0.5908789 -0.4273208
## rm      -0.3555015  0.12806864 -0.6138083  0.6953599
## age      0.2615150 -0.27353398  0.6023385 -0.3769546
## dis     -0.2324705  0.29151167 -0.4969958  0.2499287
## rad      0.4647412 -0.44441282  0.4886763 -0.3816262
## tax      0.4608530 -0.44180801  0.5439934 -0.4685359
## ptratio  1.0000000 -0.17738330  0.3740443 -0.5077867
## black   -0.1773833  1.00000000 -0.3660869  0.3334608
## lstat    0.3740443 -0.36608690  1.0000000 -0.7376627
## medv    -0.5077867  0.33346082 -0.7376627  1.0000000
```

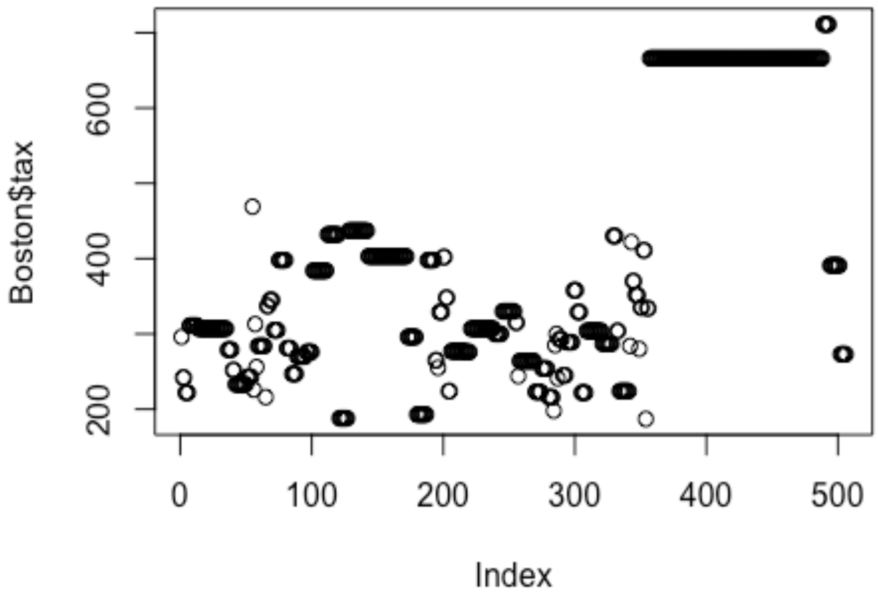
From the correlation matrix, we see that crim and rad are positively correlated (0.62) and crim and nox are inversely correlated (0.76923011).

Part d Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.

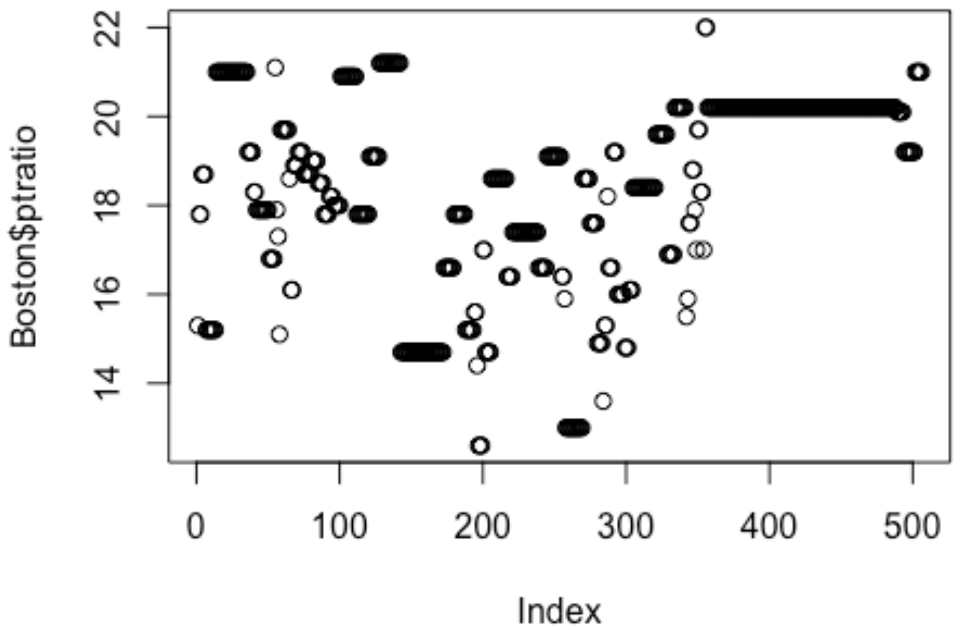
```
library(psych)
max(Boston$crim) #[1] 88.9762

## [1] 88.9762

range(Boston$crim)

## [1]  0.00632 88.97620

describe(Boston$crim)

##    vars   n mean  sd median trimmed  mad  min   max range skew kurtosis
## X1    1 506 3.61 8.6   0.26    1.68 0.33 0.01 88.98 88.97 5.19     36.6
##      se
## X1 0.38

plot(Boston$crim)
```



```
max(Boston$tax) #[1] 711

## [1] 711

range(Boston$tax)

## [1] 187 711

describe(Boston$tax)

##    vars   n   mean     sd median trimmed    mad min max range skew
## X1    1 506 408.24 168.54    330  400.04 108.23 187 711   524 0.67
##    kurtosis   se
## X1    -1.15 7.49

plot(Boston$tax)
```



```
max(Boston$ptratio)

## [1] 22

range(Boston$ptratio)

## [1] 12.6 22.0

describe(Boston$ptratio)

##    vars   n  mean   sd median trimmed mad  min max range skew kurtosis  se
## X1    1 506 18.46 2.16  19.05   18.66 1.7 12.6  22   9.4 -0.8     -0.3 0.1

plot(Boston$ptratio)
```



From numbers of describe function and plot, we see that crim rate of 88,98 is outlier in the data. Median value for crime rate is 0.26. There are only 54 suburbs that has crime rate higher than 10. Range for crime rate is 0 - 89.

Tax rate is also skewed data, median is at 330 and max value is at 711. 137 suburbs have higher tax rate than 600. But tax rate predictor is less skewed than the crime rate predictor. Range for tax rate is 187 - 711.

Pupil-teacher max ratio, 22, is actually closer to its median value (19) and its range is 12.6 - 22. It is skewed to the left, majority of the suburbs have ratio more than 20.

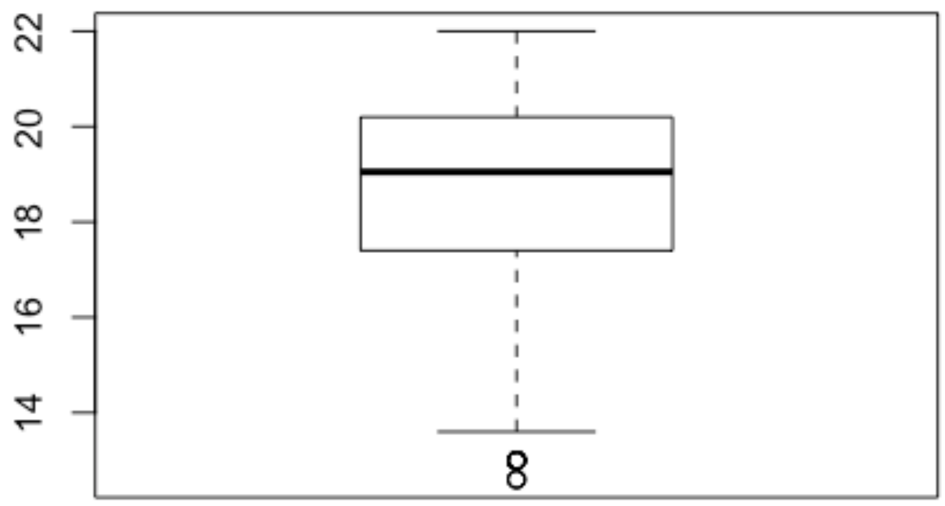
10e How many of the suburbs in this data set bound the Charles river?

```
sum(Boston$chas)

## [1] 35
```

35 of the suburbs are bound to Charles river.

10f What is the median pupil-teacher ratio among the towns in this data set?

```
boxplot(Boston$ptratio)
```



```
median(Boston$ptratio)

## [1] 19.05
```

Median pupil-teacher ratio amound the towns is 19.05.

10g Which suburb of Boston has lowest median value of owner- occupied homes? What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

```
min(Boston$medv)

## [1] 5

Boston[which.min(Boston$medv),]

##        crim zn indus chas   nox    rm age    dis rad tax ptratio black
## 399 38.3518  0  18.1    0 0.693 5.453 100 1.4896  24 666    20.2 396.9
##     lstat medv
## 399 30.59    5

for(i in 1:ncol(Boston))
{ predictor <- colnames(Boston)[i]
print(paste(predictor, ": ", Boston[which.min(Boston$medv),][predictor], ", ", range(Bost
on[[predictor]])[1] , " - " , range(Boston[[predictor]])[2]))
}

## [1] "crim :  38.3518 ,  0.00632  -  88.9762"
## [1] "zn :  0 ,  0  -  100"
## [1] "indus :  18.1 ,  0.46  -  27.74"
## [1] "chas :  0 ,  0  -  1"
## [1] "nox :  0.693 ,  0.385  -  0.871"
## [1] "rm :  5.453 ,  3.561  -  8.78"
## [1] "age :  100 ,  2.9  -  100"
## [1] "dis :  1.4896 ,  1.1296  -  12.1265"
## [1] "rad :  24 ,  1  -  24"
## [1] "tax :  666 ,  187  -  711"
## [1] "ptratio :  20.2 ,  12.6  -  22"
## [1] "black :  396.9 ,  0.32  -  396.9"
## [1] "lstat :  30.59 ,  1.73  -  37.97"
## [1] "medv :  5 ,  5  -  50"
```

The suburb with index 399 has lowest median value of 5.

In the result below, we displayed each predictor name, value of the predictor of the suburb that has min median value and the range of the predictor. From the result, we see that 'Accessability index to highway' and 'Black proportion' predictors have max values.

10h In this data set, how many of the suburbs average more than seven rooms per dwelling? More than eight rooms per dwelling? Comment on the suburbs that average more than eight rooms per dwelling.

```
nrow(Boston[Boston$rm>7,])

## [1] 64

nrow(Boston[Boston$rm>8,])

## [1] 13

Boston[Boston$rm>8,]

##        crim zn indus chas    nox    rm  age    dis rad tax ptratio  black
## 98  0.12083  0  2.89    0 0.4450 8.069 76.0 3.4952   2 276    18.0 396.90
## 164 1.51902  0 19.58    1 0.6050 8.375 93.9 2.1620   5 403    14.7 388.45
## 205 0.02009 95  2.68    0 0.4161 8.034 31.9 5.1180   4 224    14.7 390.55
## 225 0.31533  0  6.20    0 0.5040 8.266 78.3 2.8944   8 307    17.4 385.05
## 226 0.52693  0  6.20    0 0.5040 8.725 83.0 2.8944   8 307    17.4 382.00
## 227 0.38214  0  6.20    0 0.5040 8.040 86.5 3.2157   8 307    17.4 387.38
## 233 0.57529  0  6.20    0 0.5070 8.337 73.3 3.8384   8 307    17.4 385.91
## 234 0.33147  0  6.20    0 0.5070 8.247 70.4 3.6519   8 307    17.4 378.95
## 254 0.36894 22  5.86    0 0.4310 8.259  8.4 8.9067   7 330    19.1 396.90
## 258 0.61154 20  3.97    0 0.6470 8.704 86.9 1.8010   5 264    13.0 389.70
## 263 0.52014 20  3.97    0 0.6470 8.398 91.5 2.2885   5 264    13.0 386.86
## 268 0.57834 20  3.97    0 0.5750 8.297 67.0 2.4216   5 264    13.0 384.54
## 365 3.47428  0 18.10    1 0.7180 8.780 82.9 1.9047  24 666    20.2 354.55
##     lstat medv
## 98   4.21 38.7
## 164  3.32 50.0
## 205  2.88 50.0
## 225  4.14 44.8
## 226  4.63 50.0
## 227  3.13 37.6
## 233  2.47 41.7
## 234  3.95 48.3
## 254  3.54 42.8
## 258  5.12 50.0
## 263  5.91 48.8
## 268  7.44 50.0
## 365  5.29 21.9
```

64 suburbs have average higher than seven rooms per dwelling and 13 suburbs have average higher than eight rooms per dwelling. From the result, we see that most of the suburbs have low values in ‘Accessability index’, ‘Pupil-teacher ratio’, ‘Tax rate’ and ‘Distance to employement center’

ISLR Ch. 3 Ex. 9

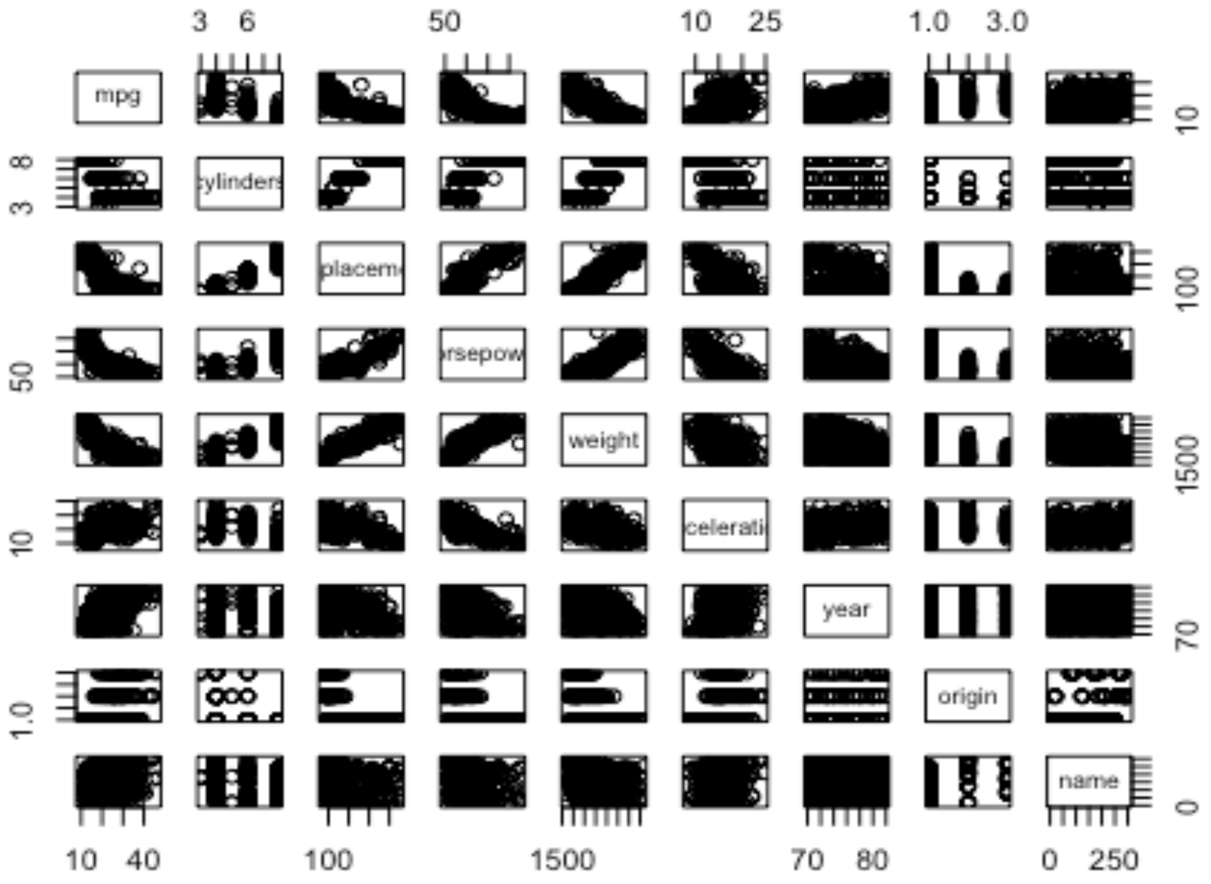
This question involves the use of multiple linear regression on the Auto data set. (a) Produce a scatterplot matrix which includes all of the variables in the data set.
(b) Compute the matrix of correlations between the variables using the function cor(). You will need to exclude the name variable, which is qualitative.
(c) Use the lm() function to perform a multiple linear regression with mpg as the response and all other variables except name as the predictors. Use the summary() function to print the results. Comment on the output. For instance:
i. Is there a relationship between the predictors and the response?
ii. Which predictors appear to have a statistically significant relationship to the response?
iii. What does the coefficient for the year variable suggest?
(d) Use the plot() function to produce diagnostic plots of the linear regression fit. Comment on any problems you see with the fit. Do the residual plots suggest any unusually large outliers? Does the leverage plot identify any observations with unusually high leverage?
(e) Use the * and : symbols to fit linear regression models with interaction effects. Do any interactions appear to be statistically significant?
(f) Try a few different transformations of the variables, such as log(X), sqrt(X), X2. Comment on your findings.

Load data and check for NAs:

```
library(ISLR)
data <- Auto
sum(is.na(data))

## [1] 0
```

Part a: Produce a scatterplot matrix for all variables in the data set

```
pairs(Auto)
```



Part b: Compute the matrix of correlations between the variables using the function cor(). You will need to exclude the name variable, which is qualitative.

```
corMatrix <- cor(data[,-9])
corMatrix

##                     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
##              acceleration       year     origin
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

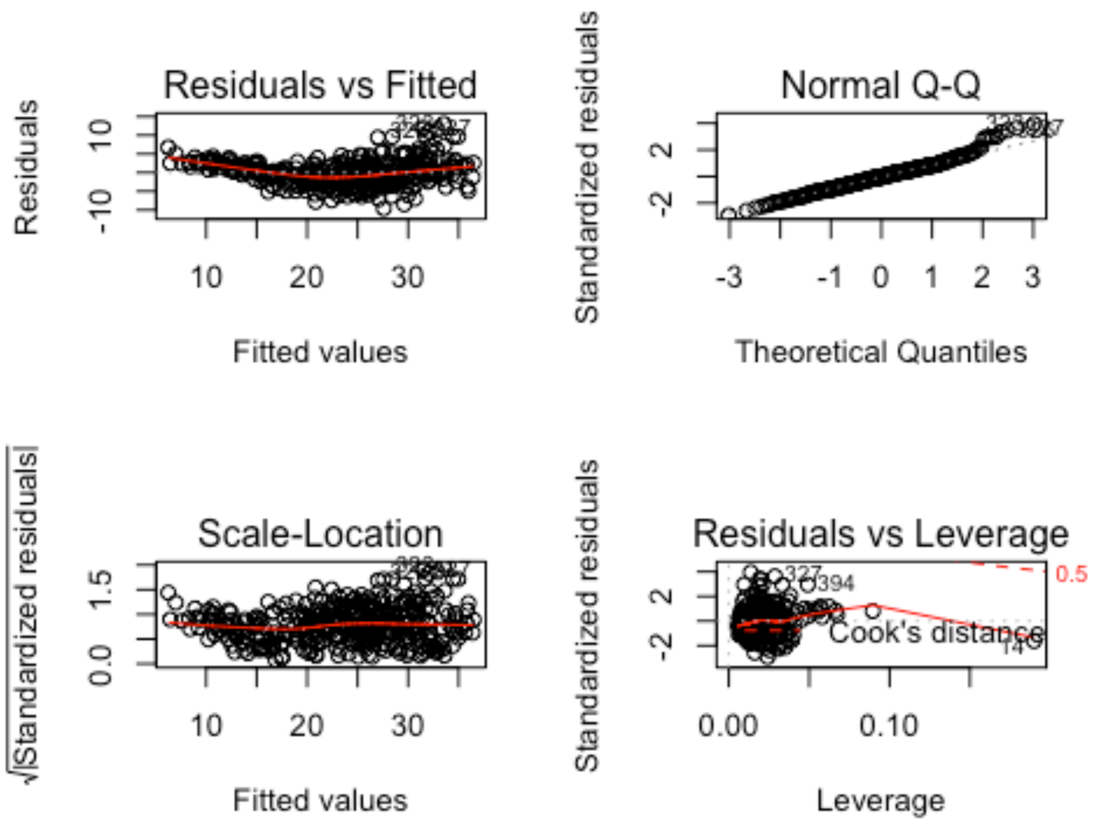
Part c: Perform multiple linear regression with mpg as response and all others (except name) as predictors

```r
lmfit <- lm(mpg ~ . -name, data)
summary(lmfit)

##
## Call:
## lm(formula = mpg ~ . - name, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

From the summary we see that there are statistically significant relationships between several variables and mpg: displacement, weight, year, and origin. Other variables were notstatistically significant: cylinders, horsepower, acceleration. The calculated coefficients indicate how much one unit of change in the variable would change the mpg, for instance a year increase will increase mpg by 0.750773 while 1 lb increase in weight will decrease mpg by .006474. Overall, the lm fit is pretty good, with an R^2 of 0.8215, as being closer to 1 indicates a better fit.

Part d: Plot diagnostic graphs for lmfit and comment

```r
par(mfrow=c(2,2))
plot(lmfit)
```



The residual plot is heteroschedastic with definite U shape. Several outliers are also present in the upper range for observations 323, 327, and 387. Overall, the Q-Q plot indicates normality, except at the lower and upper ends where the previously noted outliers deviate from normality. The leverage chart looks good with the exception of some of the same outliers from before, but also a new one: Observation 14 has an expectionally high leverage.

Part e: Use * and : operators in the fit to investigate for interactions

First, we use vif() and the scatter matrix to identify potential interactions

```
library(car)

##
## Attaching package: 'car'

## The following object is masked from 'package:psych':
##
##     logit

vif(lmfit)

##    cylinders displacement   horsepower       weight acceleration
##    10.737535    21.836792     9.943693    10.831260     2.625806
##         year       origin
##     1.244952     1.772386
```

From vif, we see that cylinders, displacement, horsepower, and weight all have likely collinearity. The scatterplot from earlier confirms that these variables show clear patterns among each other. Additionally, general car knowledge supports a relationship among these variables

```
lmfitint1 <- lm(mpg ~ horsepower*cylinders*displacement*weight + acceleration + year + or
igin, data)
summary(lmfitint1)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower * cylinders * displacement * weight +
##     acceleration + year + origin, data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.112 -1.543 -0.125  1.382 13.069
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                               -8.089e+01  5.876e+01  -1.377
## horsepower                                 5.010e-01  5.933e-01   0.844
## cylinders                                  1.689e+01  1.106e+01   1.527
## displacement                              -3.377e-01  4.311e-01  -0.783
## weight                                     4.547e-02  1.982e-02   2.294
## acceleration                              -2.050e-01  9.953e-02  -2.060
## year                                       7.423e-01  4.534e-02  16.372
## origin                                     7.726e-01  2.666e-01   2.898
## horsepower:cylinders                      -1.370e-01  1.004e-01  -1.365
## horsepower:displacement                    4.712e-03  4.731e-03   0.996
## cylinders:displacement                     1.707e-02  5.956e-02   0.287
## horsepower:weight                         -4.866e-04  2.029e-04  -2.398
## cylinders:weight                          -8.292e-03  3.443e-03  -2.408
## displacement:weight                       -5.558e-05  1.348e-04  -0.412
## horsepower:cylinders:displacement         -4.120e-04  6.219e-04  -0.663
## horsepower:cylinders:weight                7.695e-05  3.132e-05   2.457
## horsepower:displacement:weight             2.351e-07  1.450e-06   0.162
## cylinders:displacement:weight              1.274e-05  1.831e-05   0.696
## horsepower:cylinders:displacement:weight -6.842e-08  1.898e-07  -0.361
##                                           Pr(>|t|)
## (Intercept)                                0.16949
## horsepower                                 0.39898
## cylinders                                  0.12758
## displacement                               0.43385
## weight                                     0.02234 *
## acceleration                               0.04014 *
## year                                       < 2e-16 ***
## origin                                     0.00397 **
## horsepower:cylinders                       0.17323
## horsepower:displacement                    0.31993
## cylinders:displacement                     0.77460
## horsepower:weight                          0.01698 *
## cylinders:weight                           0.01652 *
## displacement:weight                        0.68028
## horsepower:cylinders:displacement          0.50802
## horsepower:cylinders:weight                0.01446 *
## horsepower:displacement:weight             0.87126
## cylinders:displacement:weight              0.48691
## horsepower:cylinders:displacement:weight   0.71866
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.826 on 373 degrees of freedom
## Multiple R-squared:  0.8749, Adjusted R-squared:  0.8689
## F-statistic: 144.9 on 18 and 373 DF,  p-value: < 2.2e-16
```

Based on fitting with interactions with high vif variables, re-fit with most significant interactions. The main effect for interacting variables remains included:

```
lmfitint2 <- lm(mpg ~ horsepower:cylinders:weight +
                  horsepower:weight +
                  cylinders:weight +
                  horsepower + cylinders + weight + displacement + acceleration + year +
origin, data)
summary(lmfitint2)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower:cylinders:weight + horsepower:weight +
##     cylinders:weight + horsepower + cylinders + weight + displacement +
##     acceleration + year + origin, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0027 -1.6059 -0.0449  1.4575 11.7696
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -4.736e+00  5.669e+00  -0.835 0.404053
## horsepower                  -1.331e-01  4.390e-02  -3.031 0.002607 **
## cylinders                   -3.516e-01  1.013e+00  -0.347 0.728834
## weight                      -2.611e-03  3.269e-03  -0.799 0.425006
## displacement                 5.892e-04  7.089e-03   0.083 0.933804
## acceleration                -1.300e-01  9.456e-02  -1.375 0.169879
## year                         7.577e-01  4.481e-02  16.906  < 2e-16 ***
## origin                       8.386e-01  2.514e-01   3.336 0.000932 ***
## horsepower:weight           -3.154e-05  2.741e-05  -1.150 0.250699
## cylinders:weight            -6.900e-04  4.228e-04  -1.632 0.103503
## horsepower:cylinders:weight  8.070e-06  2.592e-06   3.114 0.001988 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.9 on 381 degrees of freedom
## Multiple R-squared:  0.8655, Adjusted R-squared:  0.862
## F-statistic: 245.2 on 10 and 381 DF,  p-value: < 2.2e-16
```

There’s a much improved F stat over interacting all collinear variables, but some aren’t significant now

```
lmfitint3 <- lm(mpg ~ horsepower:cylinders:weight +
                  horsepower + cylinders + weight + displacement + acceleration + year +
origin, data)
summary(lmfitint3)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower:cylinders:weight + horsepower +
##     cylinders + weight + displacement + acceleration + year +
##     origin, data = data)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.3075 -1.6061 -0.1048  1.5049 11.9046
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    1.342e+00  4.387e+00   0.306  0.75986
## horsepower                    -1.462e-01  1.679e-02  -8.706  < 2e-16 ***
## cylinders                     -1.377e+00  2.932e-01  -4.696  3.7e-06 ***
## weight                        -7.931e-03  5.839e-04 -13.582  < 2e-16 ***
## displacement                   4.488e-03  6.703e-03   0.670  0.50351
## acceleration                  -7.705e-02  8.740e-02  -0.882  0.37858
## year                           7.674e-01  4.449e-02  17.249  < 2e-16 ***
## origin                         7.876e-01  2.494e-01   3.157  0.00172 **
## horsepower:cylinders:weight    4.257e-06  3.860e-07  11.028  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.903 on 383 degrees of freedom
## Multiple R-squared:  0.8645, Adjusted R-squared:  0.8617
## F-statistic: 305.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

```
vif(lmfitint3)
```

```
##                  horsepower                   cylinders
##                   19.385130                   11.603783
##                      weight                displacement
##                   11.415789                   22.828539
##                acceleration                        year
##                    2.697954                    1.246386
##                      origin horsepower:cylinders:weight
##                    1.873325                   28.336605
```

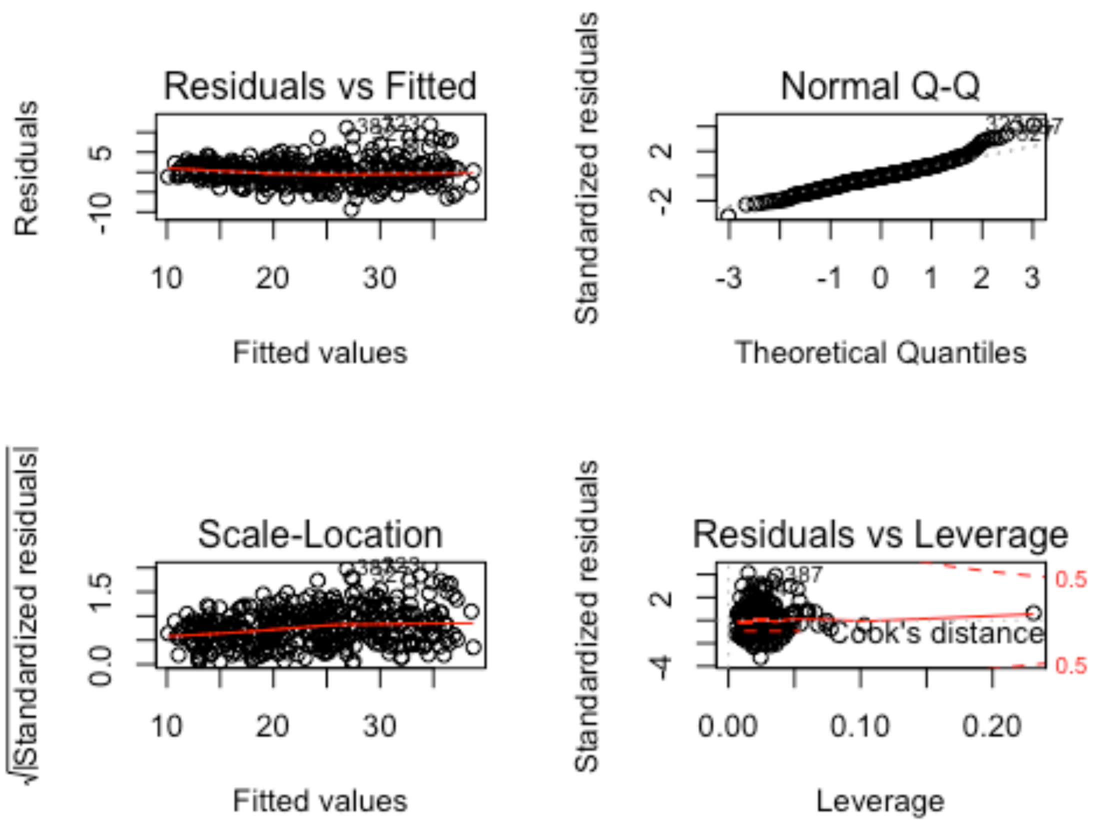
Here we see our highest F stat yet, but let’s try some adding a few more interactions based on scatterplot:

```
lmfitint4 <- lm(mpg ~ horsepower:cylinders:weight +
                  acceleration:horsepower +
                  horsepower + cylinders + weight + displacement + acceleration + year +
origin, data)
summary(lmfitint4)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower:cylinders:weight + acceleration:horsepower +
##     horsepower + cylinders + weight + displacement + acceleration +
##     year + origin, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0881 -1.6271 -0.0618  1.3659 11.7906
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -8.073e+00  5.277e+00  -1.530  0.12690
## horsepower                    -6.331e-02  3.124e-02  -2.026  0.04343 *
## cylinders                     -9.977e-01  3.141e-01  -3.176  0.00161 **
## weight                       -6.596e-03  7.177e-04  -9.190  < 2e-16 ***
## displacement                 -6.170e-03  7.450e-03  -0.828  0.40805
## acceleration                  3.579e-01  1.636e-01   2.188  0.02927 *
## year                          7.676e-01  4.399e-02  17.451  < 2e-16 ***
## origin                        6.908e-01  2.485e-01   2.779  0.00572 **
## horsepower:acceleration      -5.574e-03  1.780e-03  -3.132  0.00187 **
## horsepower:cylinders:weight   3.707e-06  4.201e-07   8.824  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.87 on 382 degrees of freedom
## Multiple R-squared:  0.8679, Adjusted R-squared:  0.8648
## F-statistic: 278.9 on 9 and 382 DF,  p-value: < 2.2e-16
```

After trying a few more interactions, lmfitint3 is currently the highest F stat model

```r
par(mfrow=c(2,2))
plot(lmfitint3)
```



Residuals are much more heteroschedastic and the lower end of Q-Q shows more normality, while top end is still skewed

Part f: Try a few transformations.

Based on curved scatterplots for mpg vs displacement, horsepower, weight, and acceleration let's start by transforming those variables:

```r
lmfit.xf <- lm(mpg ~ I(1/horsepower) + I(1/weight) + I(1/displacement) + I(log(accelerati
on)) +
                  cylinders + year + origin, data)
summary(lmfit.xf)

##
## Call:
## lm(formula = mpg ~ I(1/horsepower) + I(1/weight) + I(1/displacement) +
##     I(log(acceleration)) + cylinders + year + origin, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5006 -1.5841 -0.1152  1.4502 12.4492
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -3.962e+01  5.787e+00  -6.846 3.01e-11 ***
## I(1/horsepower)       7.226e+02  1.237e+02   5.840 1.12e-08 ***
## I(1/weight)           3.152e+04  5.453e+03   5.781 1.54e-08 ***
## I(1/displacement)     6.025e+01  1.653e+02   0.365 0.715614
## I(log(acceleration)) -5.083e+00  1.389e+00  -3.660 0.000287 ***
## cylinders            -2.445e-01  2.047e-01  -1.194 0.233056
## year                  7.575e-01  4.425e-02  17.120  < 2e-16 ***
## origin                6.850e-01  2.668e-01   2.568 0.010619 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.926 on 384 degrees of freedom
## Multiple R-squared:  0.862,  Adjusted R-squared:  0.8595
## F-statistic: 342.6 on 7 and 384 DF,  p-value: < 2.2e-16

vif(lmfit.xf)

##      I(1/horsepower)          I(1/weight)    I(1/displacement)
##             8.646734            13.443993            13.480909
## I(log(acceleration))            cylinders                 year
##             2.884374             5.571261             1.213455
##               origin
##             2.109545
```

After trying various transforms, like X^2, sqrt, and log, inspection of the scatterplots suggested an inverse relationship for hp, wt, and disp, with a log relationship for accelaration. This combo yields the highest F stat yet. Now let’s use the highest vif transformed values for potential interactions:

```
lmfit.int.xf <- lm(mpg ~ I(1/horsepower):I(1/weight):I(1/displacement) +
                  I(1/horsepower) + I(1/weight) + I(log(acceleration)) +
                    cylinders+ displacement + year + origin, data)
summary(lmfit.int.xf)

##
## Call:
## lm(formula = mpg ~ I(1/horsepower):I(1/weight):I(1/displacement) +
##     I(1/horsepower) + I(1/weight) + I(log(acceleration)) + cylinders +
##     displacement + year + origin, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.2463 -1.5222 -0.1758  1.4208 12.4990
##
```

```
## Coefficients:
##                                              Estimate Std. Error
## (Intercept)                                -4.229e+01  6.761e+00
## I(1/horsepower)                             7.241e+02  1.497e+02
## I(1/weight)                                 3.528e+04  6.517e+03
## I(log(acceleration))                       -4.569e+00  1.498e+00
## cylinders                                  -4.458e-01  2.857e-01
## displacement                                5.164e-03  6.066e-03
## year                                        7.621e-01  4.517e-02
## origin                                      7.863e-01  2.628e-01
## I(1/horsepower):I(1/weight):I(1/displacement) -3.243e+06  1.720e+07
##                                            t value Pr(>|t|)
## (Intercept)                                 -6.255 1.06e-09 ***
## I(1/horsepower)                              4.836 1.92e-06 ***
## I(1/weight)                                  5.413 1.09e-07 ***
## I(log(acceleration))                        -3.050  0.00244 **
## cylinders                                   -1.560  0.11958
## displacement                                 0.851  0.39516
## year                                        16.870  < 2e-16 ***
## origin                                       2.991  0.00296 **
## I(1/horsepower):I(1/weight):I(1/displacement)  -0.188  0.85059
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.927 on 383 degrees of freedom
## Multiple R-squared:  0.8622, Adjusted R-squared:  0.8593
## F-statistic: 299.6 on 8 and 383 DF,  p-value: < 2.2e-16
```

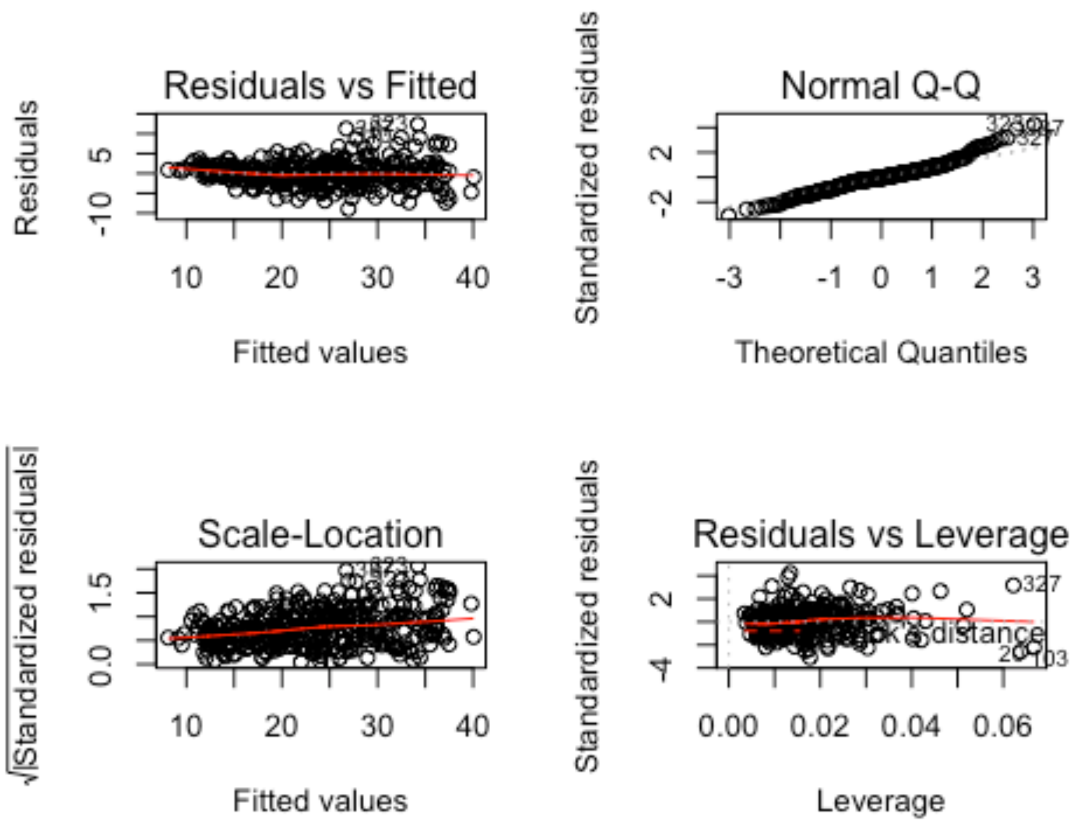
That didn't add value. At this point, let's try removing the highest p value variables cylinders and displacement

```
lmfit.xf.select <- lm(mpg ~ I(1/horsepower) + I(1/weight) + I(log(acceleration)) +
                  + year + origin, data)
summary(lmfit.xf.select)
```

```
##
## Call:
## lm(formula = mpg ~ I(1/horsepower) + I(1/weight) + I(log(acceleration)) +
##     +year + origin, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9579 -1.6025 -0.1189  1.4858 12.3661
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -4.507e+01  4.450e+00 -10.129  < 2e-16 ***
## I(1/horsepower)       7.052e+02  1.230e+02   5.734 1.99e-08 ***
## I(1/weight)           3.645e+04  3.759e+03   9.696  < 2e-16 ***
## I(log(acceleration)) -4.360e+00  1.308e+00  -3.334 0.000937 ***
## year                  7.684e-01  4.342e-02  17.697  < 2e-16 ***
## origin                7.644e-01  2.372e-01   3.222 0.001379 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.927 on 386 degrees of freedom
## Multiple R-squared:  0.8611, Adjusted R-squared:  0.8593
## F-statistic: 478.8 on 5 and 386 DF,  p-value: < 2.2e-16
```

The best model yet by F stat and R2, let's check its plots:

```
par(mfrow=c(2,2))
plot(lmfit.xf.select)
```

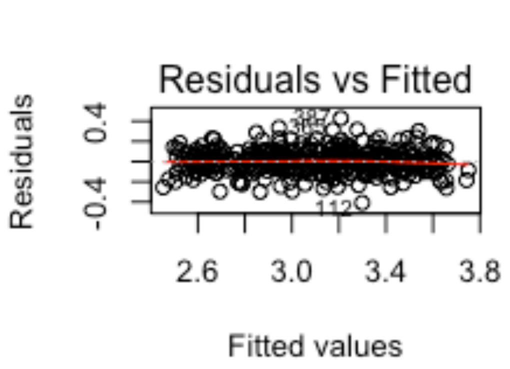


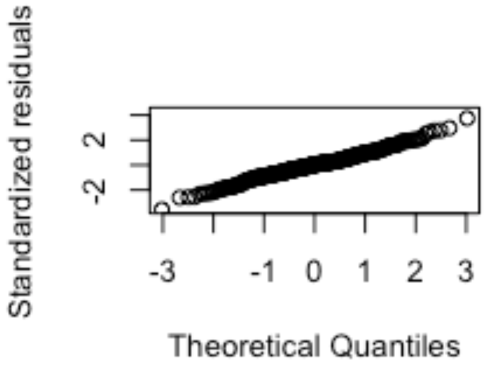
Here we see a funnel shape in the residuals, per ISLR that shape can be addressed with a log or sqrt transform on Y, so let's try transforming mpg via sqrt and log:

```
lmfit.xf.select1 <- lm(I(log(mpg)) ~ I(1/horsepower) + I(1/weight) +
                          acceleration + cylinders + displacement + year + origin, data)
summary(lmfit.xf.select1)

##
## Call:
## lm(formula = I(log(mpg)) ~ I(1/horsepower) + I(1/weight) + acceleration +
##     cylinders + displacement + year + origin, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40869 -0.06995  0.00188  0.06410  0.42937
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.813e-01  1.897e-01   2.010   0.0451 *
## I(1/horsepower)  2.240e+01  5.163e+00   4.339 1.83e-05 ***
## I(1/weight)      1.340e+03  2.026e+02   6.613 1.27e-10 ***
## acceleration    -9.537e-03  3.873e-03  -2.462   0.0142 *
## cylinders       -2.762e-02  1.115e-02  -2.477   0.0137 *
## displacement    -2.643e-04  2.285e-04  -1.156   0.2482
## year             3.052e-02  1.782e-03  17.133  < 2e-16 ***
## origin           1.342e-02  9.609e-03   1.396   0.1634
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1159 on 384 degrees of freedom
## Multiple R-squared:  0.886,  Adjusted R-squared:  0.8839
## F-statistic: 426.2 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lmfit.xf.select1)
```

```
## Error: $ operator is invalid for atomic vectors
```





Here we see no shape to residuals and the QQ chart indicates normality. Due to the transform of mpg, cylinders and displacement were re-added. Overall this model has a lower F stat than untransformed mpg, but better residuals and QQ normality measures, thus transformed mpg has a better linear relationship with the variables and should be used.

ISLR Ch 3. Ex. 15

This problem involves the Boston data set, which we saw in the lab for this chapter. We will now try to predict per capita crime rate using the other variables in this data set. In other words, per capita crime rate is the response, and the other variables are the predictors.
(a) 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.
(b) 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 $H_0 : \beta_j = 0$?
(c) How do your results from (a) compare to your results from (b)? Create a plot displaying the univariate regression coefficients from (a) on the x-axis, and the multiple regression coefficients from (b) on the y-axis. That is, each predictor is displayed as a single point in the plot. Its coefficient in a simple linear regression model is shown on the x-axis, and its coefficient estimate in the multiple linear regression model is shown on the y-axis.

(d) Is there evidence of non-linear association between any of the predictors and the response? To answer this question, for each predictor X, fit a model of the form:
$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$.

Part a: For each predictor, fit a simple linear regression model

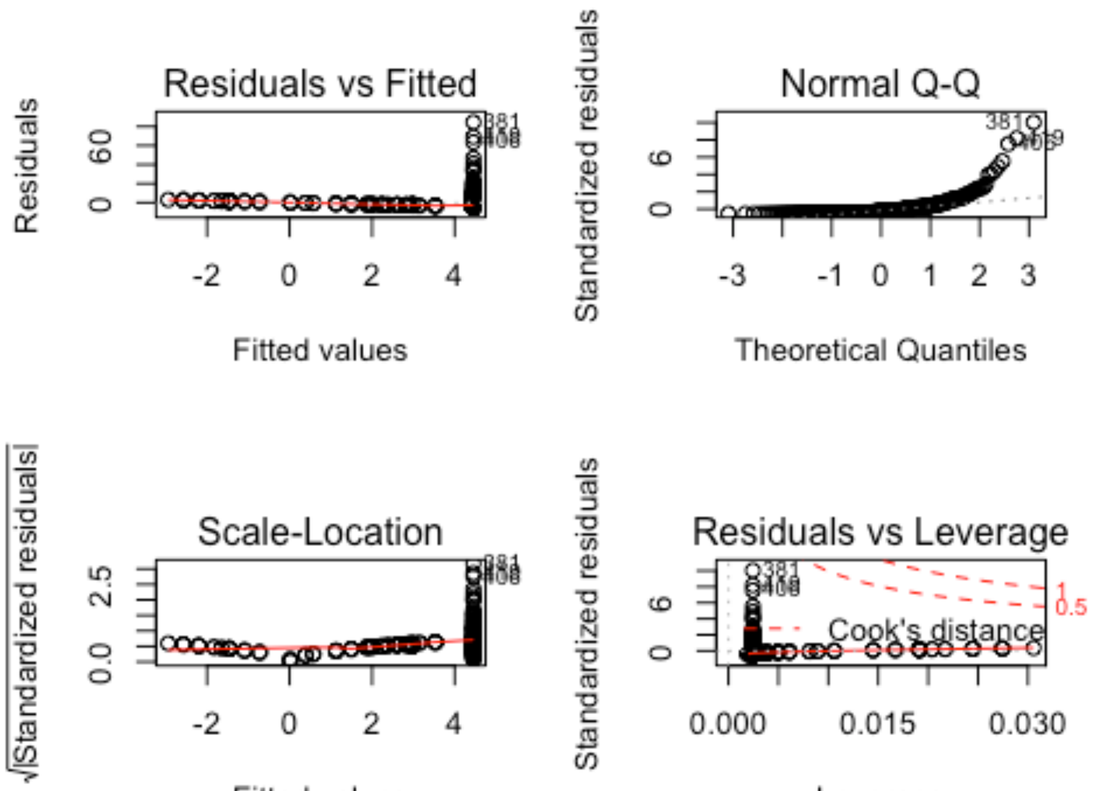
```r
prData <-Boston
n <- ncol(prData)-1
townPredictor <-rep(NA, n)
townCorrelation <-rep(NA,n)
townCoef <- rep(NA, n)
townPvalue <- rep(NA, n)
for(i in 2:ncol(prData))
{ predictor <- colnames(prData)[i]
  fit <- lm(crim ~ prData[[predictor]], data= prData)
  townPredictor[i-1] <- predictor
  townCorrelation[i-1] <-summary(fit)$adj.r.squared
  townCoef[i-1] <- fit$coefficients[2]
  townPvalue[i-1] <- summary(fit)$coefficients[2,4]
}
corCrim<-data.frame(townPredictor, townCorrelation, townCoef, townPvalue)
corCrim

##    townPredictor townCorrelation    townCoef   townPvalue
## 1             zn      0.03828352 -0.07393498 5.506472e-06
## 2          indus      0.16365394  0.50977633 1.450349e-21
## 3           chas      0.00114594 -1.89277655 2.094345e-01
## 4            nox      0.17558468 31.24853120 3.751739e-23
## 5             rm      0.04618036 -2.68405122 6.346703e-07
## 6            age      0.12268419  0.10778623 2.854869e-16
## 7            dis      0.14245126 -1.55090168 8.519949e-19
## 8            rad      0.39004886  0.61791093 2.693844e-56
## 9            tax      0.33830395  0.02974225 2.357127e-47
## 10       ptratio      0.08225111  1.15198279 2.942922e-11
## 11         black      0.14658431 -0.03627964 2.487274e-19
## 12         lstat      0.20601869  0.54880478 2.654277e-27
## 13          medv      0.14909551 -0.36315992 1.173987e-19
```

From the matrix that consists of each predictor, correlation with crime rate, coefficient of the predictor and p-value of simple linear regression, we see that only the predictor “chas” is not associanted statistically significant to crim. The other predictors all have p-value less than 5%, which means there is a statistically significant association with the response.

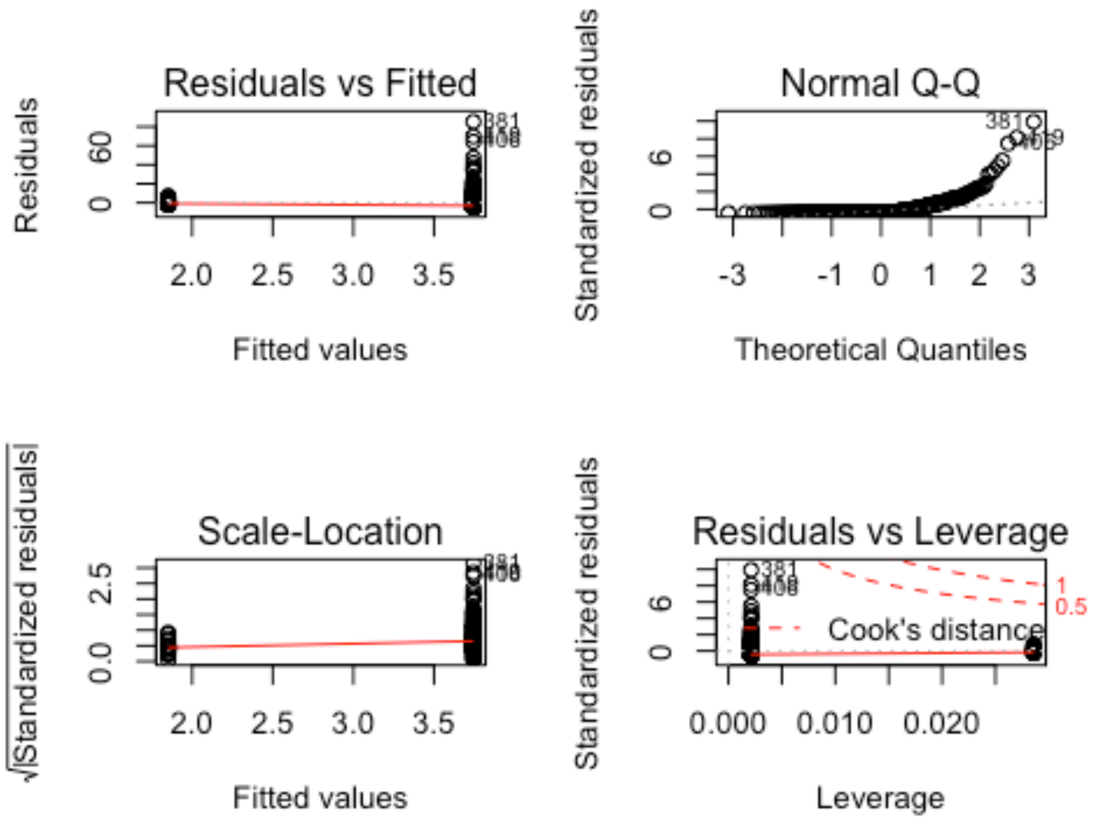
```
par(mfrow=c(2,2))
zn_fit <- lm(crim ~ zn, data= Boston)
summary(zn_fit)
```

```
##
## Call:
## lm(formula = crim ~ zn, data = Boston)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.429 -4.222 -2.620  1.250 84.523
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.45369    0.41722  10.675  < 2e-16 ***
## zn          -0.07393    0.01609  -4.594 5.51e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.435 on 504 degrees of freedom
## Multiple R-squared:  0.04019,    Adjusted R-squared:  0.03828
## F-statistic:  21.1 on 1 and 504 DF,  p-value: 5.506e-06
```

```
plot(zn_fit)
```



```
chas_fit <- lm(crim ~ chas, data= Boston)
summary(chas_fit)

##
## Call:
## lm(formula = crim ~ chas, data = Boston)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.738 -3.661 -3.435  0.018 85.232
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.7444     0.3961   9.453   <2e-16 ***
## chas         -1.8928     1.5061  -1.257    0.209
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.597 on 504 degrees of freedom
## Multiple R-squared:  0.003124,   Adjusted R-squared:  0.001146
## F-statistic: 1.579 on 1 and 504 DF,  p-value: 0.2094

plot(chas_fit)
```



For comparison, we put two linear regression model plots. One is of predictor zn (example of statistically significant association) and the other is predictor chas (the one predictor that does not have association).

Part b: Fit a multiple regression model to predict the response

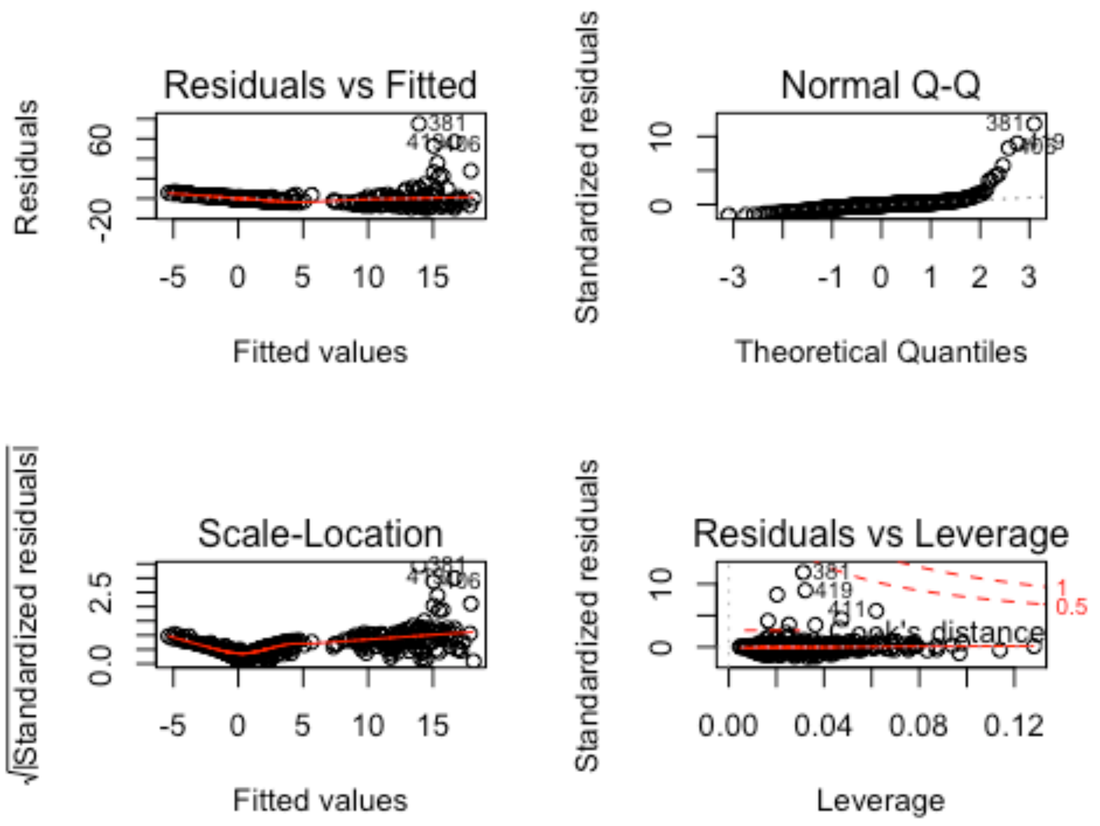
```
mfit <-lm(crim ~ ., data = prData)
summary(mfit)
```

```
##
## Call:
## lm(formula = crim ~ ., data = prData)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.924 -2.120 -0.353  1.019 75.051
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.033228   7.234903   2.354 0.018949 *
## zn            0.044855   0.018734   2.394 0.017025 *
## indus        -0.063855   0.083407  -0.766 0.444294
## chas         -0.749134   1.180147  -0.635 0.525867
## nox         -10.313535   5.275536  -1.955 0.051152 .
## rm            0.430131   0.612830   0.702 0.483089
## age           0.001452   0.017925   0.081 0.935488
## dis          -0.987176   0.281817  -3.503 0.000502 ***
## rad           0.588209   0.088049   6.680 6.46e-11 ***
## tax          -0.003780   0.005156  -0.733 0.463793
## ptratio      -0.271081   0.186450  -1.454 0.146611
## black        -0.007538   0.003673  -2.052 0.040702 *
## lstat         0.126211   0.075725   1.667 0.096208 .
## medv         -0.198887   0.060516  -3.287 0.001087 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.439 on 492 degrees of freedom
## Multiple R-squared:  0.454,  Adjusted R-squared:  0.4396
## F-statistic: 31.47 on 13 and 492 DF,  p-value: < 2.2e-16
```

```
mfit$coefficients
```

```
##   (Intercept)            zn         indus          chas           nox
##  17.033227523   0.044855215  -0.063854824  -0.749133611 -10.313534912
##            rm           age           dis           rad           tax
##   0.430130506   0.001451643  -0.987175726   0.588208591  -0.003780016
##       ptratio         black         lstat          medv
##  -0.271080558  -0.007537505   0.126211376  -0.198886821
```

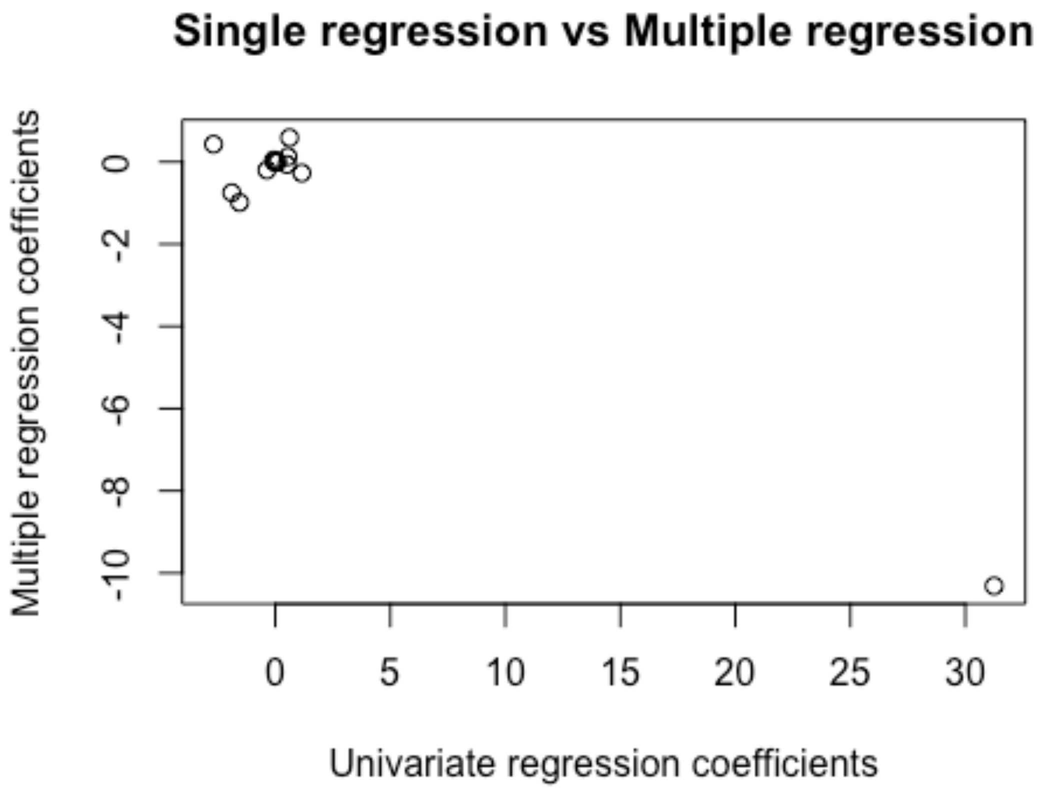
```
par(mfrow=c(2,2))
plot(mfit)
```



Since p-value is less than 5%, we can reject Null hypothesis for the predictors zn, dis, rad, black and medv. Adjusted r-square of the model is 0.44 and p-value is less than 2.2e-16.

Part c: How do your results from (a) compare to your results from (b)?

```
#dev.off()
plot(townCoef, mfit$coefficients[-1], main = "Single regression vs Multiple regression",
xlab = "Univariate regression coefficients", ylab = "Multiple regression coefficients")
```



An interesting observation from the plot is that coefficients of the predictors are clustered around 0 both in simple and multiple linear models except for predictor nox. That means one unit change of nox affects response much higher than the other predictors.

Part d: Is there evidence of non-linear association between any of the predictors and the response?

```
nlPred <-rep(NA, n)
nlCorr <-rep(NA,n)
nlPvalue<-rep(NA,n)

for(i in 2:ncol(prData))
{
  predictor <- colnames(prData)[i]
  nlfit <- lm(crim ~ prData[[predictor]] + prData[[predictor]]^2 + prData[[predictor]]^3,
data= prData)
  nlPred[i-1] <- predictor
  nlCorr[i-1] <-summary(nlfit)$adj.r.squared
  nlPvalue[i-1] <- summary(nlfit)$coefficients[2,4]
}

townPvalue

##  [1] 5.506472e-06 1.450349e-21 2.094345e-01 3.751739e-23 6.346703e-07
##  [6] 2.854869e-16 8.519949e-19 2.693844e-56 2.357127e-47 2.942922e-11
## [11] 2.487274e-19 2.654277e-27 1.173987e-19

nlPvalue
```

```
##  [1] 5.506472e-06 1.450349e-21 2.094345e-01 3.751739e-23 6.346703e-07
##  [6] 2.854869e-16 8.519949e-19 2.693844e-56 2.357127e-47 2.942922e-11
## [11] 2.487274e-19 2.654277e-27 1.173987e-19
```

```
townCorrelation
```

```
##  [1] 0.03828352 0.16365394 0.00114594 0.17558468 0.04618036 0.12268419
##  [7] 0.14245126 0.39004886 0.33830395 0.08225111 0.14658431 0.20601869
## [13] 0.14909551
```

```
nlCorr
```

```
##  [1] 0.03828352 0.16365394 0.00114594 0.17558468 0.04618036 0.12268419
##  [7] 0.14245126 0.39004886 0.33830395 0.08225111 0.14658431 0.20601869
## [13] 0.14909551
```

In the comparison of pvalue of single and multiple linear models, there is not change in the numbers. Same is true for correlation numbers. There is no better non-linear association with the response.