Chapter 3 - Linear Regression

Lab Solution

1 Problem 9

First we will read the “Auto” data. Note that most datasets referred to in the text are in the R package the authors developed. So we just need to use the data() function to extract it into R’s memory.

```r
library(ISLR)
data("Auto")
```

Each of the 392 rows of this dataset corresponds to a make/model of car, and columns represent various characteristics of that make/model.

```r
names(Auto)
## [1] "mpg"    "cylinders"  "displacement" "horsepower"  "weight"  
## [6] "acceleration" "year"  "origin"  "name"

dim(Auto)
## [1] 392  9

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
```

Eventually, we will try to predict the fuel efficiency of a car given its various characteristics.

(a). Produce a scatterplot matrix which includes all of the variables in the data set.

A scatterplot matrix can be constructed with the pairs() function.

```
pairs(Auto)
```
Note some of the trends in the data. For example, as the number of cylinders increases it appears that fuel efficiency increases and decreases. There are numerous curvilinear relationships with mpg as well.

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

As the problem mentions, we need to remove the name column because it is not numeric. Recall the output of names(Auto) above, and note that "name" is the 9th entry in that output, so cor(Auto[, -9]) would do the job for us. However it would be inefficient to lookup this number each time, so instead we can use the which() function to find the column for us to remove in the same way. Additionally, the subset() function can be used to remove the names of columns. So that

\[
\text{cor(Auto[, -9])} \\
\text{cor(Auto[, -which(names(Auto) %in% c("name"))])} \\
\text{cor(subset(Auto, select=-name))}
\]

will all produce

<table>
<thead>
<tr>
<th></th>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>origin</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>mpg</td>
<td>1.000000</td>
<td>-0.7776175</td>
<td>-0.8051269</td>
<td>-0.7784268</td>
<td>-0.8322442</td>
<td>0.4233285</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cylinders</td>
<td>-0.7776175</td>
<td>1.0000000</td>
<td>0.9508233</td>
<td>0.8429834</td>
<td>0.8975273</td>
<td>-0.5046834</td>
<td></td>
<td></td>
</tr>
<tr>
<td>displacement</td>
<td>-0.8051269</td>
<td>0.9508233</td>
<td>1.0000000</td>
<td>0.8972570</td>
<td>0.9329944</td>
<td>-0.5438005</td>
<td></td>
<td></td>
</tr>
<tr>
<td>horsepower</td>
<td>-0.7784268</td>
<td>0.8429834</td>
<td>0.8972570</td>
<td>1.0000000</td>
<td>0.8645377</td>
<td>-0.6891955</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weight</td>
<td>-0.8322442</td>
<td>0.8975273</td>
<td>0.9329944</td>
<td>0.8645377</td>
<td>1.0000000</td>
<td>-0.4168392</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acceleration</td>
<td>0.4233285</td>
<td>-0.5046834</td>
<td>-0.5438005</td>
<td>-0.6891955</td>
<td>-0.4168392</td>
<td>1.0000000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>year</td>
<td>0.5805410</td>
<td>-0.3456474</td>
<td>-0.3698552</td>
<td>-0.4163615</td>
<td>-0.3091199</td>
<td>0.2903161</td>
<td></td>
<td></td>
</tr>
<tr>
<td>origin</td>
<td>0.5652088</td>
<td>-0.5689316</td>
<td>-0.6145351</td>
<td>-0.4551715</td>
<td>-0.5850054</td>
<td>0.2127458</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Note that the signs and magnitudes more or less match the directions and strengths of the relationships we observe in the scatterplot matrix, though because of the non-linear nature of some relationships these results may be dubious!

(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.

We can tell the lm function to use all other variables in a dataset by using a "." after the ~ in formula specification. To remove a variable we can subtract it after we place the ".". For example in this case:

```r
my_lm = lm(mpg ~ cylinders+displacement+horsepower+weight+acceleration+year+origin, data=Auto)
my_lm = lm(mpg ~ . - name, data=Auto)
```

Both give us the same result

```r
summary(my_lm)
```

### Call:
lm(formula = mpg ~ . - name, data = Auto)

### 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: 
'***' 0.001 '**' 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

Comment on the output. For instance:

i. **Is there a relationship between the predictors and the response?**

   The global F test has a very small p-value, indicating there is some relationship between this set of predictors and the response.

ii. **Which predictors appear to have a statistically significant relationship to the response?**

   Examination of the p-values in the coefficient table suggests that displacement, weight, year, and origin have a statistically significant relationship with mpg, while cylinders, horsepower, and acceleration do not.
iii. What does the coefficient for the year variable suggest?

The regression coefficient for year is 0.750773, which suggests that for every one year increase of a make/model car, mpg increases by 0.750773. i.e. newer cars are more fuel efficient.

(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?

In R, plot() is what is called a 'generic' function. This means that you can pass it many different types of objects and it will do different things depending on what is passed. In this problem, my_lm is an lm object. So when we call plot(my_lm), R will immediately call the subroutine plot.lm(), which is the plotting function for lm objects.

```
par(mfrow=c(2,2))  # creates a 2 x 2 panel display
plot(my_lm)
```

![Diagnostic Plots](image)

From the plots, it seems:

- The residual plot has a strong curve, indicating issues with the fit.
• The assumption of normality does not seem to be violated.
• Observation 14 has high leverage.

(e). Use the * and : symbols to fit linear regression models with interaction effects. Do any interactions appear to be statistically significant?

Note that when specifying formulas, the * operator gives the multiplicative term as well as the first order terms, and the : operator just gives the multiplicative term. This means that

\[
\text{lm}(Y \sim a*b) \\
\text{lm}(Y \sim a + b + a:b)
\]

would give the same result.

There are many choices for possible interaction effects. Typically, one needs to have knowledge of the problem in order to choose meaningful combinations of predictor variables. For example, in this problem, it seems logical that the effect of weight and acceleration may have some kind of joint impact on mpg, hence we can modify the model in part (c) to add this in.

\[
\text{my\_lm\_e} = \text{lm}(\text{mpg} \sim . - \text{name} + \text{weight:acceleration}, \text{data}=\text{Auto})
\]

\[
\text{summary(my\_lm\_e)}
\]

|            | Estimate | Std. Error | t value | Pr(>|t|) |
|------------|----------|------------|---------|----------|
| (Intercept)| -4.364e+01 | 5.811e+00 | -7.511 | 4.18e-13 *** |
| cylinders  | -2.141e-01 | 3.078e-01 | -0.696 | 0.487117 |
| displacement| 3.138e-03 | 7.495e-03 | 0.419 | 0.675622 |
| horsepower | 1.348e-02 | 1.636e-03 | -3.071 | 0.002287 * |
| weight     | 4.027e-03 | 1.363e-03 | 2.462 | 0.014268 * |
| acceleration| 1.613e+00 | 2.422e-01 | 6.726 | 6.36e-11 *** |
| year       | 7.821e-01 | 8.408e-05 | 16.184 | < 2e-16 *** |
| origin     | 3.846e-01 | 1.033e+00 | 3.636 | 0.000141 *** |
| weight:acceleration | -5.826e-04 | 8.408e-05 | -6.928 | 1.81e-11 *** |

Note that our interaction effect is significant and the \( R^2 \) value increased, indicating a better fit. There are many different possibilities here, so your answers will likely vary.

(f). Try a few different transformations of the variables, such as \( \log(X) \), \( \sqrt{X} \), \( X^2 \), etc. Comment on your findings.

\[
\text{my\_lm\_f} = \text{lm}(\text{mpg} \sim . - \text{name} + \text{log(weight)} + \text{sqrt(horsepower)} + \text{I(displacement}^2) + \text{I(cylinders}^2), \text{data}=\text{Auto})
\]

\[
\text{summary(my\_lm\_f)}
\]
Problem 13

(a). Using the `rnorm()` function, create a vector, \( x \), containing 100 observations drawn from a \( N(0,1) \) distribution. This represents a feature, \( X \).

```r
set.seed(1)
x = rnorm(100)
```

(b). Using the `rnorm()` function, create a vector, \( \varepsilon \), containing 100 observations drawn from a \( N(0, 0.25) \) distribution i.e. a normal distribution with mean zero and variance 0.25.

```r
eps = rnorm(100, 0, sqrt(0.25))
```

(c). Using \( x \) and \( \varepsilon \), generate a vector \( y \) according to the model

\[
Y = -1 + 0.5X + \varepsilon
\]

What is the length of the vector \( y \)? What are the values of \( \beta_0 \) and \( \beta_1 \) in this linear model?

```r
y = -1 + 0.5*x + eps
```

\( y \) is comprised of 100 observations, and the values of the intercept \( \beta_0 \) and slope \( \beta_1 \) are -1 and 0.5, respectively.

(d). Create a scatterplot displaying the relationship between \( x \) and \( y \). Comment on what you observe.
(e) Fit a least squares linear model to predict \( y \) using \( x \). Comment on the model obtained. How do \( \hat{\beta}_0 \) and \( \hat{\beta}_1 \) compare to \( \beta_0 \) and \( \beta_1 \)?

```r
my_lm = lm(y~x)
summary(my_lm)
```

```
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.93842 -0.30688 -0.06975  0.26970  1.17309
```

## Coefficients:
| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) -1.01885 | 0.04849 | -21.010 | < 2e-16 *** |
| x 0.49947 | 0.05386 | 9.273 | 4.58e-15 *** |

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4814 on 98 degrees of freedom
Multiple R-squared: 0.4674, Adjusted R-squared: 0.4619
F-statistic: 85.99 on 1 and 98 DF, p-value: 4.583e-15

(f). Display the least squares line on the scatterplot obtained in (d). Draw the population regression line on the plot, in a different color. Use the legend() command to create an appropriate legend.

```r
plot(x, y)
abline(my_lm, lwd=3, col=2)
abline(-1, 0.5, lwd=3, col=3)
legend("bottomright", legend = c("model fit", "pop. regression"), col=2:3, lwd=3)
```

(g). Now fit a polynomial regression model that predicts y using x and x^2. Is there evidence that the quadratic term improves the model fit? Explain your answer.

```r
lm_fit_sq = lm(y ~ x + I(x^2))
summary(lm_fit_sq)
```
Call: lm(formula = y ~ x + I(x^2))

Residuals:
   Min     1Q Median     3Q    Max
-0.98252 -0.31270 -0.06441  0.29014  1.13500

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.97164  0.05883  -16.517  < 2e-16 ***
x           0.50858  0.05399    9.420  2.4e-15 ***
I(x^2)      -0.05946  0.04238   -1.403  0.164
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.479 on 97 degrees of freedom
Multiple R-squared: 0.4779, Adjusted R-squared: 0.4672
F-statistic: 44.4 on 2 and 97 DF, p-value: 2.038e-14

(h). Repeat (a)-(f) after modifying the data generation process in such a way that there is less noise in the data. The model (3.39) should remain the same. You can do this by decreasing the variance of the normal distribution used to generate the error term in (b). Describe your results.

set.seed(1)
x = rnorm(100)
eps = rnorm(100, 0, 0.125)
y = -1 + 0.5*x + eps
lm.fit = lm(y~x)
summary(lm.fit)

Call: lm(formula = y ~ x)
Residuals:
   Min     1Q Median     3Q    Max
-0.23461 -0.07672 -0.01744  0.06742  0.29327
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.00471  0.01212  -82.87  < 2e-16 ***
x           0.49987  0.01347   37.12  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.1203 on 98 degrees of freedom
Multiple R-squared: 0.9336, Adjusted R-squared: 0.9329
F-statistic: 1378 on 1 and 98 DF, p-value: < 2.2e-16

plot(x, y)
abline(lm.fit, lwd=3, col=2)
abline(-1, 0.5, lwd=3, col=3)
legend("bottomright", legend = c("model fit", "pop. regression"), col=2:3, lwd=3)
(i). Repeat (a)-(f) after modifying the data generation process in such a way that there is more noise in the data. The model (3.39) should remain the same. You can do this by increasing the variance of the normal distribution used to generate the error term in (b). Describe your results.

```r
set.seed(1)
x = rnorm(100)
eps = rnorm(100, 0,.75)
y = -1 + 0.5*x + eps
lm.fit = lm(y~x)
summary(lm.fit)
```

##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##-2.5105 -0.8507  0.0262  0.8462  2.4904
##
## Coefficients:
##             Estimate Std. Error t value
## (Intercept) -0.9861     0.1092  -8.984
## x             0.5306     0.0735  7.192
##
## Residual standard error: 1.048 on 98 degrees of freedom
## Multiple R-squared:  0.5445, Adjusted R-squared:  0.5362
## F-statistic: 52.09 on 1 and 98 DF,  p-value: 4.979e-13
```
## Residuals:
## Min 1Q Median 3Q Max
## -1.4076 -0.4603 -0.1046 0.4046 1.7596
##
## Coefficients:
## Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.02827 0.07274 -14.136 < 2e-16 ***
## x 0.49920 0.08080 6.179 1.48e-08 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7221 on 98 degrees of freedom
## Multiple R-squared: 0.2803, Adjusted R-squared: 0.273
## F-statistic: 38.18 on 1 and 98 DF, p-value: 1.479e-08

plot(x, y)
abline(lm.fit, lwd=3, col=2)
abline(-1, 0.5, lwd=3, col=3)
legend("bottomright", legend = c("model fit", "pop. regression"), col=2:3, lwd=3)