ISLR2, must be downloaded the first time they are used. This can be done directly from within R. For example, on a Windows system, select the Install package option under the Packages tab. After you select any mirror site, a list of available packages will appear. Simply select the package you wish to install and R will automatically download the package. Alternatively, this can be done at the R command line via install.packages("ISLR2"). This installation only needs to be done the first time you use a package. However, the library() function must be called within each R session.

3.6.2 Simple Linear Regression

The ISLR2 library contains the Boston data set, which records medv (median house value) for 506 census tracts in Boston. We will seek to predict medv using 12 predictors such as rm (average number of rooms per house), age (average age of houses), and lstat (percent of households with low socioeconomic status).

```
> head(Boston)
     crim zn indus chas   nox    rm  age    dis rad tax
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242
4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222
5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222
6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222
  ptratio lstat medv
1    15.3  4.98 24.0
2    17.8  9.14 21.6
3    17.8  4.03 34.7
4    18.7  2.94 33.4
5    18.7  5.33 36.2
6    18.7  5.21 28.7
```

To find out more about the data set, we can type ?Boston.

We will start by using the lm() function to fit a simple linear regression model, with medv as the response and lstat as the predictor. The basic syntax is lm(y ∼ x, data), where y is the response, x is the predictor, and data is the data set in which these two variables are kept. lm()

```
> lm.fit <- lm(medv ∼ lstat)
Error in eval(expr, envir, enclos) : Object "medv" not found
```

The command causes an error because R does not know where to find the variables medv and lstat. The next line tells R that the variables are in Boston. If we attach Boston, the first line works fine because R now recognizes the variables.

```
> lm.fit <- lm(medv ∼ lstat, data = Boston)
> attach(Boston)
> lm.fit <- lm(medv ∼ lstat)
```

If we type `lm.fit`, some basic information about the model is output. For more detailed information, we use `summary(lm.fit)`. This gives us p-values and standard errors for the coefficients, as well as the R^2 statistic and F-statistic for the model.

```
> lm.fit

Call:
lm(formula = medv ~ lstat)

Coefficients:
(Intercept)        lstat
      34.55        -0.95

> summary(lm.fit)

Call:
lm(formula = medv ~ lstat)

Residuals:
   Min     1Q Median     3Q    Max
-15.17  -3.99  -1.32   2.03  24.50

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  34.5538     0.5626    61.4   <2e-16 ***
lstat        -0.9500     0.0387   -24.5   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 6.22 on 504 degrees of freedom
Multiple R-squared: 0.544,      Adjusted R-squared: 0.543
F-statistic:  602 on 1 and 504 DF,  p-value: < 2e-16
```

We can use the `names()` function in order to find out what other pieces of information are stored in `lm.fit`. Although we can extract these quantities by name—e.g. `lm.fit$coefficients`—it is safer to use the extractor functions like `coef()` to access them.

names()

coef()

```
> names(lm.fit)
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "xlevels"
[10] "call"          "terms"         "model"
> coef(lm.fit)
(Intercept)       lstat
      34.55       -0.95
```

In order to obtain a confidence interval for the coefficient estimates, we can use the `confint()` command.

confint()

```
> confint(lm.fit)
            2.5 % 97.5 %
(Intercept) 33.45 35.659
```

```
lstat        -1.03 -0.874
```

The `predict()` function can be used to produce confidence intervals and prediction intervals for the prediction of `medv` for a given value of `lstat`.

predict()

```
> predict(lm.fit, data.frame(lstat = (c(5, 10, 15))),
    interval = "confidence")
    fit   lwr   upr
1 29.80 29.01 30.60
2 25.05 24.47 25.63
3 20.30 19.73 20.87
> predict(lm.fit, data.frame(lstat = (c(5, 10, 15))),
    interval = "prediction")
    fit    lwr   upr
1 29.80 17.566 42.04
2 25.05 12.828 37.28
3 20.30  8.078 32.53
```

For instance, the 95 % confidence interval associated with a `lstat` value of 10 is $(24.47, 25.63)$, and the 95 % prediction interval is $(12.828, 37.28)$. As expected, the confidence and prediction intervals are centered around the same point (a predicted value of 25.05 for `medv` when `lstat` equals 10), but the latter are substantially wider.

We will now plot `medv` and `lstat` along with the least squares regression line using the `plot()` and `abline()` functions.

abline()

```
> plot(lstat, medv)
> abline(lm.fit)
```

There is some evidence for non-linearity in the relationship between `lstat` and `medv`. We will explore this issue later in this lab.

The `abline()` function can be used to draw any line, not just the least squares regression line. To draw a line with intercept `a` and slope `b`, we type `abline(a, b)`. Below we experiment with some additional settings for plotting lines and points. The `lwd = 3` command causes the width of the regression line to be increased by a factor of 3; this works for the `plot()` and `lines()` functions also. We can also use the `pch` option to create different plotting symbols.

```
> abline(lm.fit, lwd = 3)
> abline(lm.fit, lwd = 3, col = "red")
> plot(lstat, medv, col = "red")
> plot(lstat, medv, pch = 20)
> plot(lstat, medv, pch = "+")
> plot(1:20, 1:20, pch = 1:20)
```

Next we examine some diagnostic plots, several of which were discussed in Section 3.3.3. Four diagnostic plots are automatically produced by applying the `plot()` function directly to the output from `lm()`. In general, this command will produce one plot at a time, and hitting *Enter* will generate the next plot. However, it is often convenient to view all four plots together. We can achieve this by using the `par()` and `mfrow()` functions, which tell `R`

par()
mfrow()

to split the display screen into separate panels so that multiple plots can be viewed simultaneously. For example, `par(mfrow = c(2, 2))` divides the plotting region into a 2×2 grid of panels.

```
> par(mfrow = c(2, 2))
> plot(lm.fit)
```

Alternatively, we can compute the residuals from a linear regression fit using the `residuals()` function. The function `rstudent()` will return the studentized residuals, and we can use this function to plot the residuals against the fitted values.

`residuals()`
`rstudent()`

```
> plot(predict(lm.fit), residuals(lm.fit))
> plot(predict(lm.fit), rstudent(lm.fit))
```

On the basis of the residual plots, there is some evidence of non-linearity. Leverage statistics can be computed for any number of predictors using the `hatvalues()` function.

`hatvalues()`

```
> plot(hatvalues(lm.fit))
> which.max(hatvalues(lm.fit))
375
```

The `which.max()` function identifies the index of the largest element of a vector. In this case, it tells us which observation has the largest leverage statistic.

`which.max()`

3.6.3 Multiple Linear Regression

In order to fit a multiple linear regression model using least squares, we again use the `lm()` function. The syntax `lm(y ~ x1 + x2 + x3)` is used to fit a model with three predictors, `x1`, `x2`, and `x3`. The `summary()` function now outputs the regression coefficients for all the predictors.

```
> lm.fit <- lm(medv ~ lstat + age, data = Boston)
> summary(lm.fit)

Call:
lm(formula = medv ~ lstat + age, data = Boston)

Residuals:
   Min     1Q Median     3Q    Max
-15.98  -3.98  -1.28   1.97  23.16

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  33.2228     0.7308   45.46   <2e-16 ***
lstat        -1.0321     0.0482  -21.42   <2e-16 ***
age           0.0345     0.0122    2.83   0.0049 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 6.17 on 503 degrees of freedom
```

```
Multiple R-squared: 0.551,      Adjusted R-squared: 0.549
F-statistic:  309 on 2 and 503 DF,  p-value: < 2e-16
```

The Boston data set contains 12 variables, and so it would be cumbersome to have to type all of these in order to perform a regression using all of the predictors. Instead, we can use the following short-hand:

```
> lm.fit <- lm(medv ~ ., data = Boston)
> summary(lm.fit)

Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.130  -2.767  -0.581   1.941  26.253

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  41.61727    4.93604    8.43  3.8e-16 ***
crim         -0.12139    0.03300   -3.68  0.00026 ***
zn            0.04696    0.01388    3.38  0.00077 ***
indus         0.01347    0.06214    0.22  0.82852
chas          2.83999    0.87001    3.26  0.00117 **
nox         -18.75802    3.85135   -4.87  1.5e-06 ***
rm            3.65812    0.42025    8.70  < 2e-16 ***
age           0.00361    0.01333    0.27  0.78659
dis          -1.49075    0.20162   -7.39  6.2e-13 ***
rad           0.28940    0.06691    4.33  1.8e-05 ***
tax          -0.01268    0.00380   -3.34  0.00091 ***
ptratio      -0.93753    0.13221   -7.09  4.6e-12 ***
lstat        -0.55202    0.05066  -10.90  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 4.8 on 493 degrees of freedom
Multiple R-squared:  0.734,     Adjusted R-squared:  0.728
F-statistic:  114 on 12 and 493 DF,  p-value: < 2e-16
```

We can access the individual components of a summary object by name (type `?summary.lm` to see what is available). Hence `summary(lm.fit)$r.sq` gives us the R^2, and `summary(lm.fit)$sigma` gives us the RSE. The `vif()` function, part of the `car` package, can be used to compute variance inflation factors. Most VIF's are low to moderate for this data. The `car` package is not part of the base R installation so it must be downloaded the first time you use it via the `install.packages()` function in R.

vif()

```
> library(car)
> vif(lm.fit)
   crim      zn   indus    chas     nox      rm     age     dis
   1.77    2.30    3.99    1.07    4.37    1.91    3.09    3.95
    rad     tax ptratio   lstat
   7.45    9.00    1.80    2.87
```

What if we would like to perform a regression using all of the variables but one? For example, in the above regression output, age has a high p-value. So we may wish to run a regression excluding this predictor. The following syntax results in a regression using all predictors except age.

```
> lm.fit1 <- lm(medv ∼ . - age, data = Boston)
> summary(lm.fit1)
...
```

Alternatively, the update() function can be used. update()

```
> lm.fit1 <- update(lm.fit, ∼ . - age)
```

3.6.4 Interaction Terms

It is easy to include interaction terms in a linear model using the lm() function. The syntax lstat:black tells R to include an interaction term between lstat and black. The syntax lstat * age simultaneously includes lstat, age, and the interaction term lstat×age as predictors; it is a shorthand for lstat + age + lstat:age.

```
> summary(lm(medv ∼ lstat * age, data = Boston))

Call:
lm(formula = medv ∼ lstat * age, data = Boston)

Residuals:
   Min     1Q Median     3Q    Max
-15.81  -4.04  -1.33   2.08  27.55

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 36.088536   1.469835   24.55  < 2e-16 ***
lstat       -1.392117   0.167456   -8.31  8.8e-16 ***
age         -0.000721   0.019879   -0.04    0.971
lstat:age    0.004156   0.001852    2.24    0.025 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 6.15 on 502 degrees of freedom
Multiple R-squared: 0.556,  Adjusted R-squared: 0.553
F-statistic:  209 on 3 and 502 DF,  p-value: < 2e-16
```

3.6.5 Non-linear Transformations of the Predictors

The lm() function can also accommodate non-linear transformations of the predictors. For instance, given a predictor X, we can create a predictor X^2 using I(X^2). The function I() is needed since the ^ has a special meaning in a formula object; wrapping as we do allows the standard usage in R, which is to raise X to the power 2. We now perform a regression of medv onto lstat and lstat^2. I()

```
> lm.fit2 <- lm(medv ~ lstat + I(lstat^2))
> summary(lm.fit2)

Call:
lm(formula = medv ~ lstat + I(lstat^2))

Residuals:
   Min     1Q Median     3Q    Max
-15.28  -3.83  -0.53   2.31  25.41

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 42.86201    0.87208    49.1   <2e-16 ***
lstat       -2.33282    0.12380   -18.8   <2e-16 ***
I(lstat^2)   0.04355    0.00375    11.6   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 5.52 on 503 degrees of freedom
Multiple R-squared: 0.641,  Adjusted R-squared: 0.639
F-statistic:  449 on 2 and 503 DF,  p-value: < 2e-16
```

The near-zero p-value associated with the quadratic term suggests that it leads to an improved model. We use the `anova()` function to further quantify the extent to which the quadratic fit is superior to the linear fit.

anova()

```
> lm.fit <- lm(medv ~ lstat)
> anova(lm.fit, lm.fit2)
Analysis of Variance Table

Model 1: medv ~ lstat
Model 2: medv ~ lstat + I(lstat^2)
  Res.Df   RSS Df Sum of Sq   F Pr(>F)
1    504 19472
2    503 15347  1      4125 135 <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

Here Model 1 represents the linear submodel containing only one predictor, `lstat`, while Model 2 corresponds to the larger quadratic model that has two predictors, `lstat` and `lstat`^{2}. The `anova()` function performs a hypothesis test comparing the two models. The null hypothesis is that the two models fit the data equally well, and the alternative hypothesis is that the full model is superior. Here the F-statistic is 135 and the associated p-value is virtually zero. This provides very clear evidence that the model containing the predictors `lstat` and `lstat`^{2} is far superior to the model that only contains the predictor `lstat`. This is not surprising, since earlier we saw evidence for non-linearity in the relationship between `medv` and `lstat`. If we type

```
> par(mfrow = c(2, 2))
> plot(lm.fit2)
```

then we see that when the `lstat`^{2} term is included in the model, there is little discernible pattern in the residuals.

In order to create a cubic fit, we can include a predictor of the form `I(X^3)`. However, this approach can start to get cumbersome for higher-order polynomials. A better approach involves using the `poly()` function to create the polynomial within `lm()`. For example, the following command produces a fifth-order polynomial fit:

`poly()`

```
> lm.fit5 <- lm(medv ~ poly(lstat, 5))
> summary(lm.fit5)

Call:
lm(formula = medv ~ poly(lstat, 5))

Residuals:
    Min      1Q  Median      3Q     Max
-13.543  -3.104  -0.705   2.084  27.115

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)       22.533      0.232   97.20  < 2e-16 ***
poly(lstat, 5)1 -152.460      5.215  -29.24  < 2e-16 ***
poly(lstat, 5)2   64.227      5.215   12.32  < 2e-16 ***
poly(lstat, 5)3  -27.051      5.215   -5.19  3.1e-07 ***
poly(lstat, 5)4   25.452      5.215    4.88  1.4e-06 ***
poly(lstat, 5)5  -19.252      5.215   -3.69  0.00025 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 5.21 on 500 degrees of freedom
Multiple R-squared: 0.682,  Adjusted R-squared: 0.679
F-statistic:  214 on 5 and 500 DF,  p-value: < 2e-16
```

This suggests that including additional polynomial terms, up to fifth order, leads to an improvement in the model fit! However, further investigation of the data reveals that no polynomial terms beyond fifth order have significant p-values in a regression fit.

By default, the `poly()` function orthogonalizes the predictors: this means that the features output by this function are not simply a sequence of powers of the argument. However, a linear model applied to the output of the `poly()` function will have the same fitted values as a linear model applied to the raw polynomials (although the coefficient estimates, standard errors, and p-values will differ). In order to obtain the raw polynomials from the `poly()` function, the argument `raw = TRUE` must be used.

Of course, we are in no way restricted to using polynomial transformations of the predictors. Here we try a log transformation.

```
> summary(lm(medv ~ log(rm), data = Boston))
...
```

3.6.6 Qualitative Predictors

We will now examine the `Carseats` data, which is part of the `ISLR2` library. We will attempt to predict `Sales` (child car seat sales) in 400 locations based on a number of predictors.

```
> head(Carseats)
  Sales CompPrice Income Advertising Population Price
1  9.50       138     73          11        276   120
2 11.22       111     48          16        260    83
3 10.06       113     35          10        269    80
4  7.40       117    100           4        466    97
5  4.15       141     64           3        340   128
6 10.81       124    113          13        501    72
  ShelveLoc Age Education Urban  US
1       Bad  42        17   Yes Yes
2      Good  65        10   Yes Yes
3    Medium  59        12   Yes Yes
4    Medium  55        14   Yes Yes
5       Bad  38        13   Yes  No
6       Bad  78        16    No Yes
```

The `Carseats` data includes qualitative predictors such as `Shelveloc`, an indicator of the quality of the shelving location—that is, the space within a store in which the car seat is displayed—at each location. The predictor `Shelveloc` takes on three possible values: *Bad*, *Medium*, and *Good*. Given a qualitative variable such as `Shelveloc`, `R` generates dummy variables automatically. Below we fit a multiple regression model that includes some interaction terms.

```
> lm.fit <- lm(Sales ~ . + Income:Advertising + Price:Age,
    data = Carseats)
> summary(lm.fit)

Call:
lm(formula = Sales ~ . + Income:Advertising + Price:Age, data =
Carseats)

Residuals:
   Min     1Q Median     3Q    Max
-2.921 -0.750  0.018  0.675  3.341

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         6.575565   1.008747    6.52  2.2e-10 ***
CompPrice           0.092937   0.004118   22.57  < 2e-16 ***
Income              0.010894   0.002604    4.18  3.6e-05 ***
Advertising         0.070246   0.022609    3.11  0.00203 **
Population          0.000159   0.000368    0.43  0.66533
Price              -0.100806   0.007440  -13.55  < 2e-16 ***
ShelveLocGood       4.848676   0.152838   31.72  < 2e-16 ***
ShelveLocMedium     1.953262   0.125768   15.53  < 2e-16 ***
Age                -0.057947   0.015951   -3.63  0.00032 ***
```

```
Education          -0.020852   0.019613   -1.06  0.28836
UrbanYes            0.140160   0.112402    1.25  0.21317
USYes              -0.157557   0.148923   -1.06  0.29073
Income:Advertising  0.000751   0.000278    2.70  0.00729 **
Price:Age           0.000107   0.000133    0.80  0.42381
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 1.01 on 386 degrees of freedom
Multiple R-squared: 0.876,      Adjusted R-squared: 0.872
F-statistic:  210 on 13 and 386 DF,  p-value: < 2e-16
```

The `contrasts()` function returns the coding that `R` uses for the dummy variables. `contrasts()`

```
> attach(Carseats)
> contrasts(ShelveLoc)
       Good Medium
Bad       0      0
Good      1      0
Medium    0      1
```

Use `?contrasts` to learn about other contrasts, and how to set them.

`R` has created a `ShelveLocGood` dummy variable that takes on a value of 1 if the shelving location is good, and 0 otherwise. It has also created a `ShelveLocMedium` dummy variable that equals 1 if the shelving location is medium, and 0 otherwise. A bad shelving location corresponds to a zero for each of the two dummy variables. The fact that the coefficient for `ShelveLocGood` in the regression output is positive indicates that a good shelving location is associated with high sales (relative to a bad location). And `ShelveLocMedium` has a smaller positive coefficient, indicating that a medium shelving location is associated with higher sales than a bad shelving location but lower sales than a good shelving location.

3.6.7 Writing Functions

As we have seen, `R` comes with many useful functions, and still more functions are available by way of `R` libraries. However, we will often be interested in performing an operation for which no function is available. In this setting, we may want to write our own function. For instance, below we provide a simple function that reads in the `ISLR2` and `MASS` libraries, called `LoadLibraries()`. Before we have created the function, `R` returns an error if we try to call it.

```
> LoadLibraries
Error: object ‘LoadLibraries’ not found
> LoadLibraries()
Error: could not find function "LoadLibraries"
```

We now create the function. Note that the `+` symbols are printed by `R` and should not be typed in. The `{` symbol informs `R` that multiple commands are about to be input. Hitting *Enter* after typing { will cause R to print the + symbol. We can then input as many commands as we wish, hitting *Enter* after each one. Finally the } symbol informs R that no further commands will be entered.

```
> LoadLibraries <- function() {
+   library(ISLR2)
+   library(MASS)
+   print("The libraries have been loaded.")
+ }
```

Now if we type in LoadLibraries, R will tell us what is in the function.

```
> LoadLibraries
function() {
  library(ISLR2)
  library(MASS)
  print("The libraries have been loaded.")
}
```

If we call the function, the libraries are loaded in and the print statement is output.

```
> LoadLibraries()
[1] "The libraries have been loaded."
```

3.7 Exercises

Conceptual

1. Describe the null hypotheses to which the p-values given in Table 3.4 correspond. Explain what conclusions you can draw based on these p-values. Your explanation should be phrased in terms of sales, TV, radio, and newspaper, rather than in terms of the coefficients of the linear model.

2. Carefully explain the differences between the KNN classifier and KNN regression methods.

3. Suppose we have a data set with five predictors, X_1 = GPA, X_2 = IQ, X_3 = Level (1 for College and 0 for High School), X_4 = Interaction between GPA and IQ, and X_5 = Interaction between GPA and Level. The response is starting salary after graduation (in thousands of dollars). Suppose we use least squares to fit the model, and get $\hat{\beta}_0 = 50, \hat{\beta}_1 = 20, \hat{\beta}_2 = 0.07, \hat{\beta}_3 = 35, \hat{\beta}_4 = 0.01, \hat{\beta}_5 = -10$.

   (a) Which answer is correct, and why?

      i. For a fixed value of IQ and GPA, high school graduates earn more, on average, than college graduates.

ii. For a fixed value of IQ and GPA, college graduates earn more, on average, than high school graduates.

iii. For a fixed value of IQ and GPA, high school graduates earn more, on average, than college graduates provided that the GPA is high enough.

iv. For a fixed value of IQ and GPA, college graduates earn more, on average, than high school graduates provided that the GPA is high enough.

(b) Predict the salary of a college graduate with IQ of 110 and a GPA of 4.0.

(c) True or false: Since the coefficient for the GPA/IQ interaction term is very small, there is very little evidence of an interaction effect. Justify your answer.

4. I collect a set of data ($n = 100$ observations) containing a single predictor and a quantitative response. I then fit a linear regression model to the data, as well as a separate cubic regression, i.e. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$.

(a) Suppose that the true relationship between X and Y is linear, i.e. $Y = \beta_0 + \beta_1 X + \epsilon$. Consider the training residual sum of squares (RSS) for the linear regression, and also the training RSS for the cubic regression. Would we expect one to be lower than the other, would we expect them to be the same, or is there not enough information to tell? Justify your answer.

(b) Answer (a) using test rather than training RSS.

(c) Suppose that the true relationship between X and Y is not linear, but we don't know how far it is from linear. Consider the training RSS for the linear regression, and also the training RSS for the cubic regression. Would we expect one to be lower than the other, would we expect them to be the same, or is there not enough information to tell? Justify your answer.

(d) Answer (c) using test rather than training RSS.

5. Consider the fitted values that result from performing linear regression without an intercept. In this setting, the ith fitted value takes the form

$$\hat{y}_i = x_i \hat{\beta},$$

where

$$\hat{\beta} = \left(\sum_{i=1}^{n} x_i y_i \right) / \left(\sum_{i'=1}^{n} x_{i'}^2 \right). \tag{3.38}$$

Show that we can write

$$\hat{y}_i = \sum_{i'=1}^{n} a_{i'} y_{i'}.$$

What is $a_{i'}$?

Note: We interpret this result by saying that the fitted values from linear regression are linear combinations *of the response values.*

6. Using (3.4), argue that in the case of simple linear regression, the least squares line always passes through the point $(\bar{x}, \bar{y})$.

7. It is claimed in the text that in the case of simple linear regression of Y onto X, the R^2 statistic (3.17) is equal to the square of the correlation between X and Y (3.18). Prove that this is the case. For simplicity, you may assume that $\bar{x} = \bar{y} = 0$.

Applied

8. This question involves the use of simple linear regression on the `Auto` data set.

   (a) Use the `lm()` function to perform a simple linear regression with `mpg` as the response and `horsepower` as the predictor. Use the `summary()` function to print the results. Comment on the output. For example:

      i. Is there a relationship between the predictor and the response?

      ii. How strong is the relationship between the predictor and the response?

      iii. Is the relationship between the predictor and the response positive or negative?

      iv. What is the predicted `mpg` associated with a `horsepower` of 98? What are the associated 95 % confidence and prediction intervals?

   (b) Plot the response and the predictor. Use the `abline()` function to display the least squares regression line.

   (c) Use the `plot()` function to produce diagnostic plots of the least squares regression fit. Comment on any problems you see with the fit.

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. `cor()`

(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 $\log(X)$, $\sqrt{X}$, X^2. Comment on your findings.

10. This question should be answered using the `Carseats` data set.

(a) Fit a multiple regression model to predict `Sales` using `Price`, `Urban`, and `US`.

(b) Provide an interpretation of each coefficient in the model. Be careful—some of the variables in the model are qualitative!

(c) Write out the model in equation form, being careful to handle the qualitative variables properly.

(d) For which of the predictors can you reject the null hypothesis $H_0 : \beta_j = 0$?

(e) On the basis of your response to the previous question, fit a smaller model that only uses the predictors for which there is evidence of association with the outcome.

(f) How well do the models in (a) and (e) fit the data?

(g) Using the model from (e), obtain 95 % confidence intervals for the coefficient(s).

(h) Is there evidence of outliers or high leverage observations in the model from (e)?

11. In this problem we will investigate the t-statistic for the null hypothesis $H_0 : \beta = 0$ in simple linear regression without an intercept. To begin, we generate a predictor `x` and a response `y` as follows.

```
> set.seed(1)
> x <- rnorm(100)
> y <- 2 * x + rnorm(100)
```

(a) Perform a simple linear regression of `y` onto `x`, *without* an intercept. Report the coefficient estimate $\hat{\beta}$, the standard error of this coefficient estimate, and the t-statistic and p-value associated with the null hypothesis $H_0 : \beta = 0$. Comment on these results. (You can perform regression without an intercept using the command `lm(y∼x+0)`.)

(b) Now perform a simple linear regression of `x` onto `y` without an intercept, and report the coefficient estimate, its standard error, and the corresponding t-statistic and p-values associated with the null hypothesis $H_0 : \beta = 0$. Comment on these results.

(c) What is the relationship between the results obtained in (a) and (b)?

(d) For the regression of Y onto X without an intercept, the t-statistic for $H_0 : \beta = 0$ takes the form $\hat{\beta}/\text{SE}(\hat{\beta})$, where $\hat{\beta}$ is given by (3.38), and where

$$\text{SE}(\hat{\beta}) = \sqrt{\frac{\sum_{i=1}^{n}(y_i - x_i\hat{\beta})^2}{(n-1)\sum_{i'=1}^{n} x_{i'}^2}}.$$

(These formulas are slightly different from those given in Sections 3.1.1 and 3.1.2, since here we are performing regression without an intercept.) Show algebraically, and confirm numerically in `R`, that the t-statistic can be written as

$$\frac{(\sqrt{n-1})\sum_{i=1}^{n} x_i y_i}{\sqrt{(\sum_{i=1}^{n} x_i^2)(\sum_{i'=1}^{n} y_{i'}^2) - (\sum_{i'=1}^{n} x_{i'} y_{i'})^2}}.$$

(e) Using the results from (d), argue that the t-statistic for the regression of `y` onto `x` is the same as the t-statistic for the regression of `x` onto `y`.

(f) In `R`, show that when regression is performed *with* an intercept, the t-statistic for $H_0 : \beta_1 = 0$ is the same for the regression of `y` onto `x` as it is for the regression of `x` onto `y`.

12. This problem involves simple linear regression without an intercept.

    (a) Recall that the coefficient estimate $\hat{\beta}$ for the linear regression of Y onto X without an intercept is given by (3.38). Under what circumstance is the coefficient estimate for the regression of X onto Y the same as the coefficient estimate for the regression of Y onto X?

    (b) Generate an example in `R` with $n = 100$ observations in which the coefficient estimate for the regression of X onto Y is *different from* the coefficient estimate for the regression of Y onto X.

    (c) Generate an example in `R` with $n = 100$ observations in which the coefficient estimate for the regression of X onto Y is *the same as* the coefficient estimate for the regression of Y onto X.

13. In this exercise you will create some simulated data and will fit simple linear regression models to it. Make sure to use `set.seed(1)` prior to starting part (a) to ensure consistent results.

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

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

    (c) Using `x` and `eps`, generate a vector `y` according to the model

    $$Y = -1 + 0.5X + \epsilon. \tag{3.39}$$

    What is the length of the vector `y`? What are the values of β_0 and β_1 in this linear model?

    (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 β_0 and β_1?

    (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.

    (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.

(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 ϵ in (b). Describe your results.

(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 ϵ in (b). Describe your results.

(j) What are the confidence intervals for β_0 and β_1 based on the original data set, the noisier data set, and the less noisy data set? Comment on your results.

14. This problem focuses on the *collinearity* problem.

(a) Perform the following commands in R:

```
> set.seed(1)
> x1 <- runif(100)
> x2 <- 0.5 * x1 + rnorm(100) / 10
> y <- 2 + 2 * x1 + 0.3 * x2 + rnorm(100)
```

The last line corresponds to creating a linear model in which y is a function of x1 and x2. Write out the form of the linear model. What are the regression coefficients?

(b) What is the correlation between x1 and x2? Create a scatterplot displaying the relationship between the variables.

(c) Using this data, fit a least squares regression to predict y using x1 and x2. Describe the results obtained. What are $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\beta}_2$? How do these relate to the true β_0, β_1, and β_2? Can you reject the null hypothesis $H_0 : \beta_1 = 0$? How about the null hypothesis $H_0 : \beta_2 = 0$?

(d) Now fit a least squares regression to predict y using only x1. Comment on your results. Can you reject the null hypothesis $H_0 : \beta_1 = 0$?

(e) Now fit a least squares regression to predict y using only x2. Comment on your results. Can you reject the null hypothesis $H_0 : \beta_1 = 0$?

(f) Do the results obtained in (c)–(e) contradict each other? Explain your answer.

(g) Now suppose we obtain one additional observation, which was unfortunately mismeasured.

```
> x1 <- c(x1, 0.1)
> x2 <- c(x2, 0.8)
> y <- c(y, 6)
```

Re-fit the linear models from (c) to (e) using this new data. What effect does this new observation have on the each of the models? In each model, is this observation an outlier? A high-leverage point? Both? Explain your answers.

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

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon.$$