Linear Regression Exercises

Simone Collier

Simple Linear Regression

Start by loading the packages that have the data we need. If you need to install the packages first then run `install.packages("PACKAGENAME")` in your console before running the code chunk.

```
library(ISLR2)
```

Take a look at the Boston dataset using the table generating function `kable()` available in RMarkdown.

```
knitr::kable(head(Boston))
```

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

If we want to learn more about the variables in this dataset we can use `?`.

```
?Boston
```

Let's try fitting a simple linear regression to our `Boston` data

- Response variable Y: `mdev`, the median value of owner-occupied homes in
- Predictor variable X: `rm`, average number of rooms per dwelling.

The `lm()` function from the `stats` package performs the fitting of our linear model using the general syntax `lm(y ~ x, data)`.

```
lm.medv.rm <- lm(medv ~ rm, data = Boston)
lm.medv.rm
```

```
##
## Call:
## lm(formula = medv ~ rm, data = Boston)
##
## Coefficients:
## (Intercept)           rm
##     -34.671        9.102
```

The output tells us that $\hat{\beta}_0 = -34.671$ and $\hat{\beta}_1 = 9102$.

Instead of including `data = Boston` in `lm()`, we can use `attach()` to make the variables associated with `Boston` available.

```
attach(Boston)

lm.medv.rm <- lm(medv ~ rm)
lm.medv.rm
```

```
##
## Call:
## lm(formula = medv ~ rm)
##
## Coefficients:
## (Intercept)           rm
##     -34.671        9.102
```

Use the `summary()` function to look at the results of out fit.

```
lm.medv.rm_summary <- summary(lm.medv.rm)
```

We can call `?summary.lm` to see what each of the outputs mean. We can extract individual elements from the summary such as a information about the linear regression coefficients.

```
?summary.lm

lm.medv.rm_summary$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) -34.670621  2.6498030 -13.08423 6.950229e-34
## rm            9.102109  0.4190266  21.72203 2.487229e-74
```

Make it neater using `kable()`.

```
knitr::kable(lm.medv.rm_summary$coefficients)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-34.670621	2.6498030	-13.08423	0
rm	9.102109	0.4190266	21.72203	0

So we have the coefficient estimates, their associated standard errors, and the t-statistic and p-value associated with the hypothesis test $H_0 : \beta_1 = 0$. The p-value for this test is significant so we can conclude there is a relationship between `medv` and `rm`.

To compute the 95% confidence intervals for the regression coefficient estimates based on the standard errors:

```
confint(lm.medv.rm)
```

```
##                  2.5 %     97.5 %
## (Intercept) -39.876641 -29.464601
## rm            8.278855   9.925363
```

We can also find the RSE and R^2 statistic in the summary of the linear regression model.

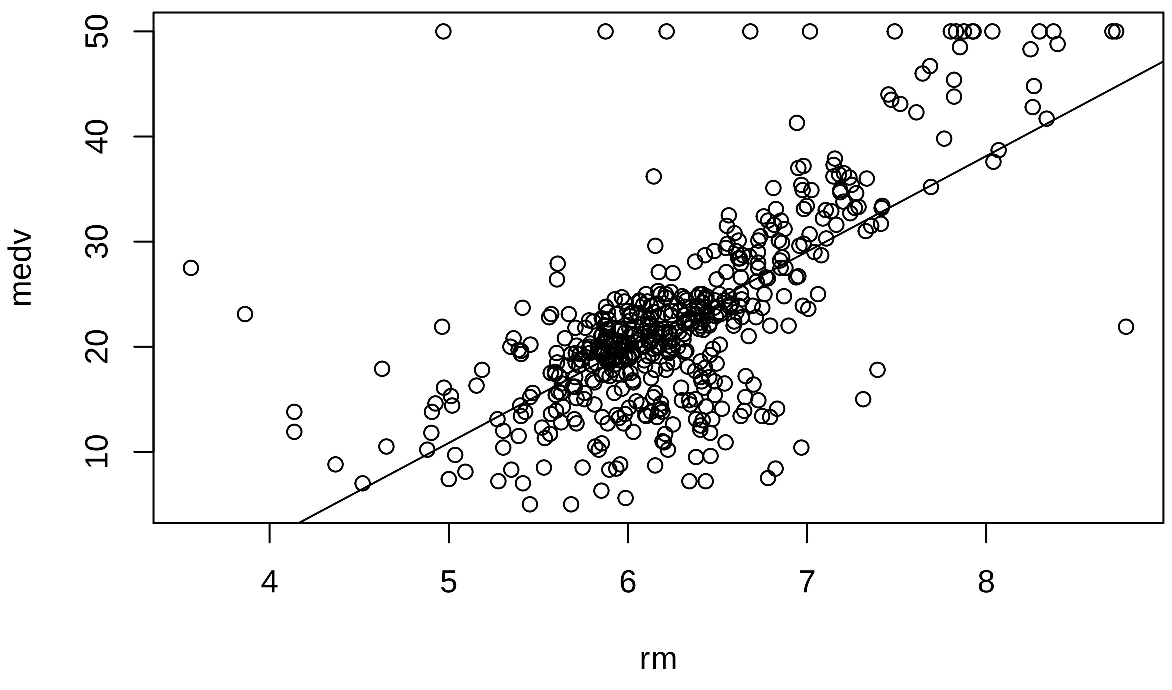
```
lm.medv.rm_summary
```

```
##
## Call:
## lm(formula = medv ~ rm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.346  -2.547   0.090   2.986  39.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -34.671      2.650  -13.08   <2e-16 ***
## rm             9.102      0.419   21.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.616 on 504 degrees of freedom
## Multiple R-squared:  0.4835, Adjusted R-squared:  0.4825
## F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
```

From the R^2 statistic we can see that some of the variation in `medv` is explained by `rm` but a lot of it is not. This might be an indication that there are other variables in the data set that are affecting the response.

We can plot our data and the linear regression model we fit.

```
plot(medv ~ rm)
abline(lm.medv.rm)
```



Multiple Linear Regression

We will choose several variables from the `Boston` data to be our predictors while retaining `medv` as our response.

- X_1: `rm`, average number of rooms per dwelling.
- X_2: `nox`, nitrogen oxides concentration (parts per 10 million)

Now we can use our `lm()` function to fit our model. With multiple predictors, the function input becomes `lm(Y ~ X_1 + X_2 + ...)`.

```
mlm.fit <- lm(medv ~ rm + nox)
mlm.fit
```

```
##
## Call:
## lm(formula = medv ~ rm + nox)
##
## Coefficients:
## (Intercept)           rm          nox
##     -18.206        8.157      -18.971
```

So we have $\hat{\beta}_0 = -18.206$, $\hat{\beta}_1 = 8.157$, and $\hat{\beta}_2 = -18.971$.

```
summary(mlm.fit)
```

```
##
## Call:
## lm(formula = medv ~ rm + nox)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.889  -3.287  -0.636   2.518  39.638
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -18.2059     3.3393  -5.452 7.82e-08 ***
## rm            8.1567     0.4173  19.546  < 2e-16 ***
## nox         -18.9706     2.5304  -7.497 2.97e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.281 on 503 degrees of freedom
## Multiple R-squared:  0.5354, Adjusted R-squared:  0.5336
## F-statistic: 289.9 on 2 and 503 DF,  p-value: < 2.2e-16
```

- The p-value at the end of the summary indicates that there is a relationship between at least one of the predictors and the response.
- The RSE and R^2 have improves compared to our simple linear regression fit. But, are the improvements the result of a better predictive model or overfitting?

Prediction intervals incorporate both the reducible and irreducible error. They can be computed using the `predict()` function. So for `rm = 6` and `nox = 0.54` we can predict Y and the prediction interval.

```
predict(mlm.fit, data.frame(rm = 6, nox = 0.54), interval = "prediction")
```

```
##        fit      lwr      upr
## 1 20.48992 8.134498 32.84533
```

$\hat{Y} = 20.48992$ is the prediction and $(8.134498, 32.84533)$ is the 95% prediction interval. We interpret this as 95% of intervals of this form will contain the true value of Y for the corresponding suburb.

Qualitative Predictors

We can examine the relationship between `medv` and `chas`, where

$$\text{chas} = \begin{cases} 1 & \text{if tract bounds Charles River} \\ 0 & \text{if not} \end{cases}$$

We can perform the regression as usual:

```
lm(medv ~ chas)
```

```
##
## Call:
## lm(formula = medv ~ chas)
##
## Coefficients:
## (Intercept)         chas
##      22.094        6.346
```

- $\hat{\beta}_0 = 22.094$: the average median house value for suburbs that are not bound by the Charles river.
- $\hat{\beta}_1 = 6.346$: the difference in the average median house value for suburbs that are bound by the Charles River versus those that are not.

Interaction Term

Let's look at the relationship between the response `medv` and the predictors `lstat` (the percent of households with low socioeconomic status) and `age` (the percent of homes built prior to 1940). We can also include the interaction term between `lstat` and `age`.

The syntax used to implement this is `lm(y ~ x1 + x2 + x1:x2, data)` or `lm(y ~ x1 * x2, data)` for shorthand.

```
summary(lm(medv ~ lstat * age))
```

```
##
## Call:
## lm(formula = medv ~ lstat * age)
##
## Residuals:
```

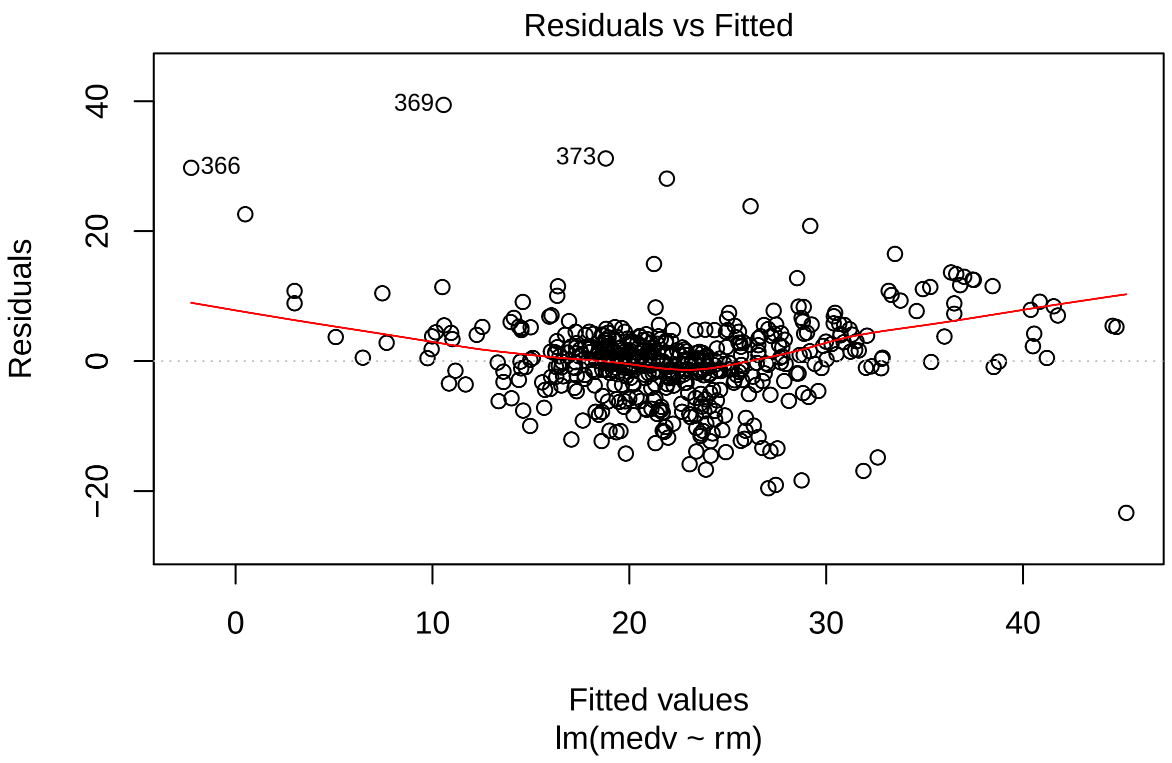
```
##     Min      1Q  Median      3Q     Max
## -15.806  -4.045  -1.333   2.085  27.552
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.0885359  1.4698355  24.553  < 2e-16 ***
## lstat       -1.3921168  0.1674555  -8.313 8.78e-16 ***
## age         -0.0007209  0.0198792  -0.036   0.9711
## lstat:age    0.0041560  0.0018518   2.244   0.0252 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.149 on 502 degrees of freedom
## Multiple R-squared:  0.5557, Adjusted R-squared:  0.5531
## F-statistic: 209.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

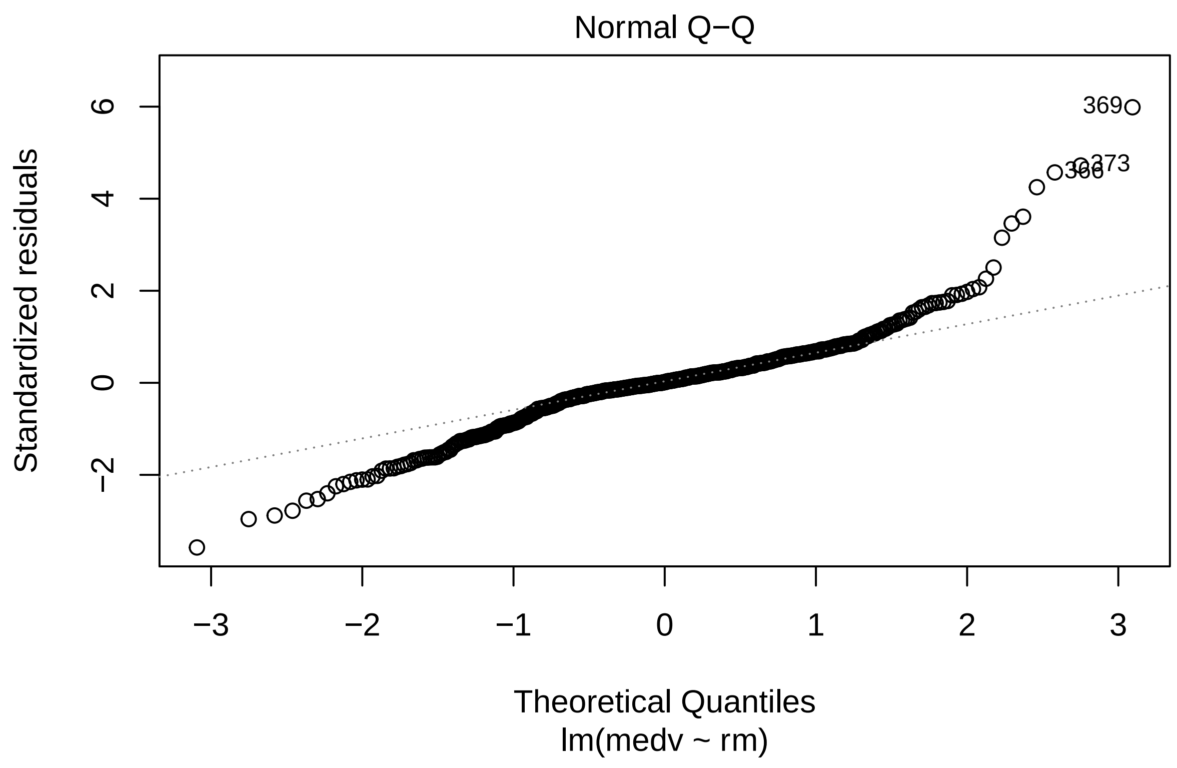
The interaction term has a *p*-value of 0.0252. Even though the *p*-value for `age` is not significant, we will still include it in our model due to the hierarchical principal.

Helpful plots

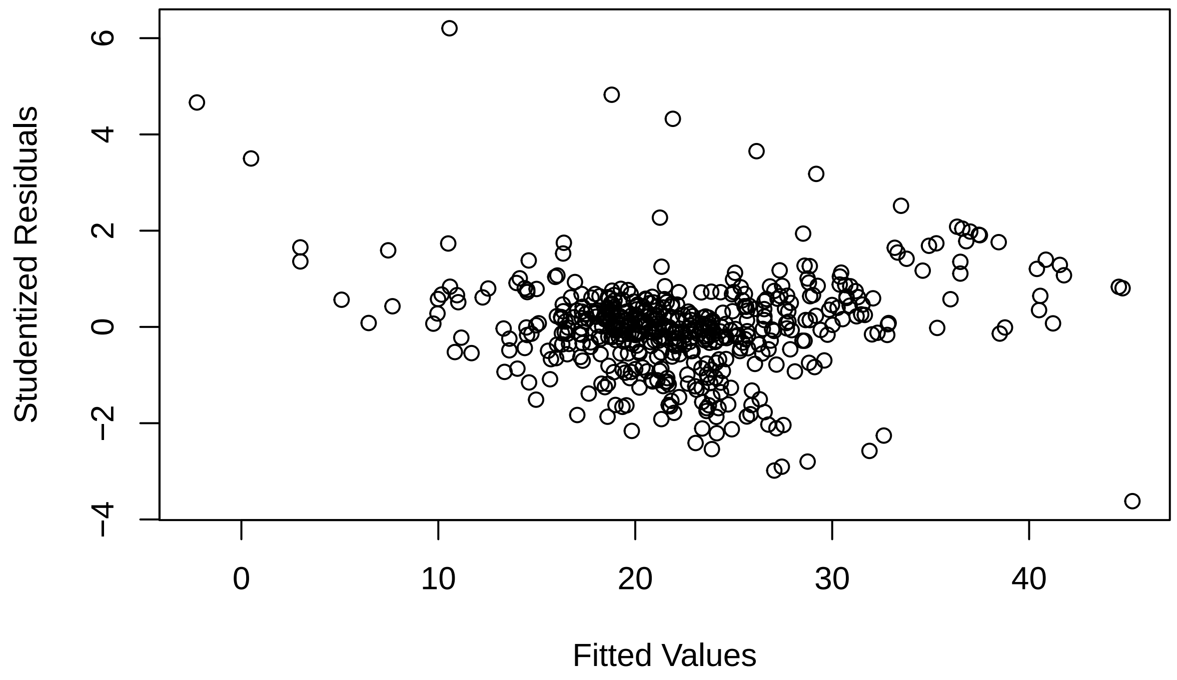
There are a few plots that we discussed that can help to identify problems with our data or with our fit. If we use the `plot()` function there are 4 plots that are automatically generated. We are particularly interested in the first 2 so we use the argument `which = c(1, 2)`. We will also look at the studentized resultuals which we can plot using `rstudent()`.

```
plot(lm.medv.rm, which = c(1, 2))
```





```r
plot(predict(lm.medv.rm), rstudent(lm.medv.rm),
     xlab = "Fitted Values", ylab = "Studentized Residuals")
```



What information about our fitted model can you gather from these plots? Are there any outliers or high leverage points?

Fit a linear regression model on the `Boston` ***data set including all the predictors. The shorthand for this is*** `lm(medv ~ .)`***. Interpret the summary including the hypothesis tests for the***

coefficients and the RSE and $R^2 values. Make plots of the fit including confidence intervals for the fitted line. Recreate and interpret the three plots we have just made using your new fit.

Ask for help if you get stuck!

These exercises were adapted from : James, Gareth, et al. An Introduction to Statistical Learning: with Applications in R, 2nd ed., Springer, 2021.