Linear regression in R

Cheatsheet

2024-09-06

Assumed knowledge

- You know how to install and load packages in R.
- You know how to import data into R.
- You recognise data frames and vectors.

Data structure

The data should be in a **long format** (also known as tidy data), where each row is an observation and each column is a variable (Figure 1). If your data is not already structured this way, reshape it manually in a spreadsheet program or in R using the `pivot_longer()` function from the `tidyr` package.

Sex	BW
F	2.15
M	2.55
F	2.95
F	2.70
M	2.20
F	1.85
M	2.55
M	2.60

F	M
2.15	2.55
2.95	2.20
2.70	2.55
1.85	2.60

Figure 1: Data should be in long format (left) where each row is an observation and each column is a variable. This is the preferred format for most statistical software. Wide format (right) is also common, but may require additional steps to analyse or visualise in some instances.

! Data

For this cheatsheet we will use data from the penguins dataset from the `palmerpenguins` package. You may need to install this package:

```
install.packages("palmerpenguins")
data(penguins)
```

About

Regression analysis is the most commonly used statistical technique for modelling the relationship between variables that can be continuous, categorical or a mix of both. In fact, other techniques such as the *t*-test, ANOVA, ANCOVA and even non-parametric tests can be considered as special cases of regression analysis. In this cheatsheet, we will focus on **linear regression**.

R packages used

Implementing linear models

Simple linear regression

```r
fit01 <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
```

Multiple linear regression

```r
fit02 <- lm(body_mass_g ~ flipper_length_mm + bill_length_mm,
    data = penguins
)
```

Interactions

```r
fit03 <- lm(body_mass_g ~ flipper_length_mm * bill_length_mm,
    data = penguins
)
```

Regression involving categorical variables

```r
fit04 <- lm(body_mass_g ~ species + sex, data = penguins)
```

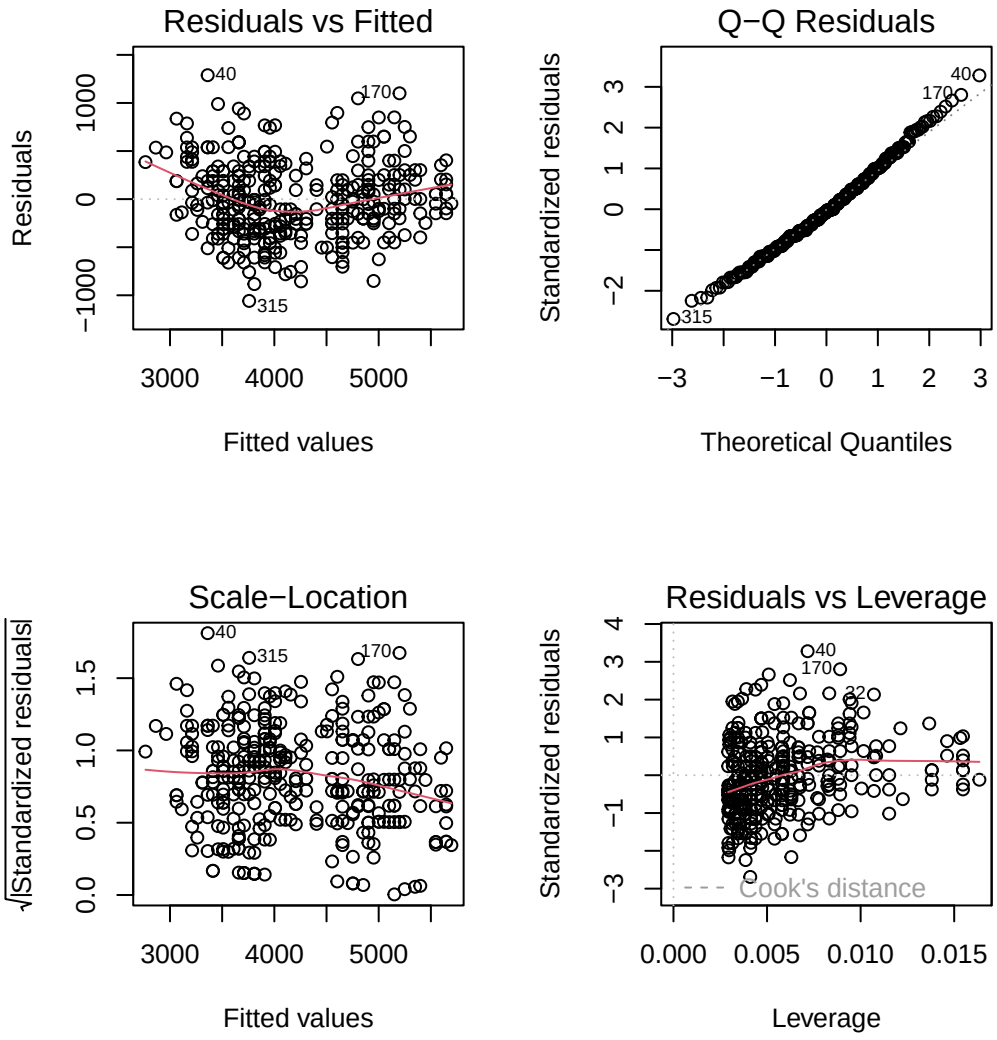
Regression involving a mix of continuous and categorical variables

```r
fit04 <- lm(body_mass_g ~ species + flipper_length_mm,
    data = penguins
)
```

Assumptions

Use the `plot()` function on the linear mode object to check the assumptions of the linear regression model.

```r
par(mfrow = c(2, 2)) # Set up a 2x2 grid of plots
plot(fit01)
```

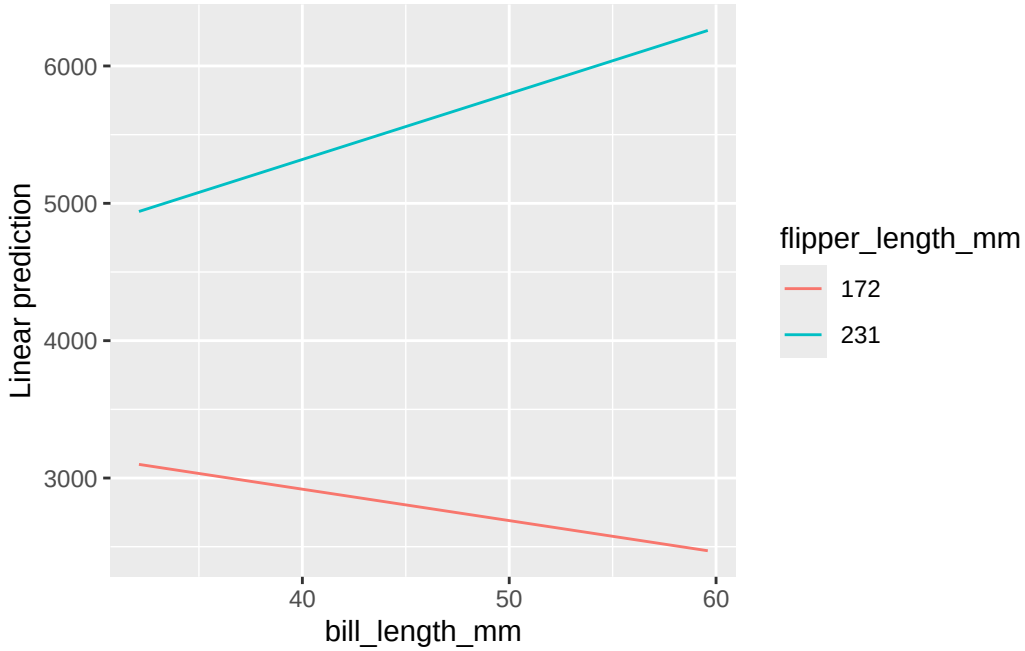



```
par(mfrow = c(1, 1)) # Reset the plot layout
```

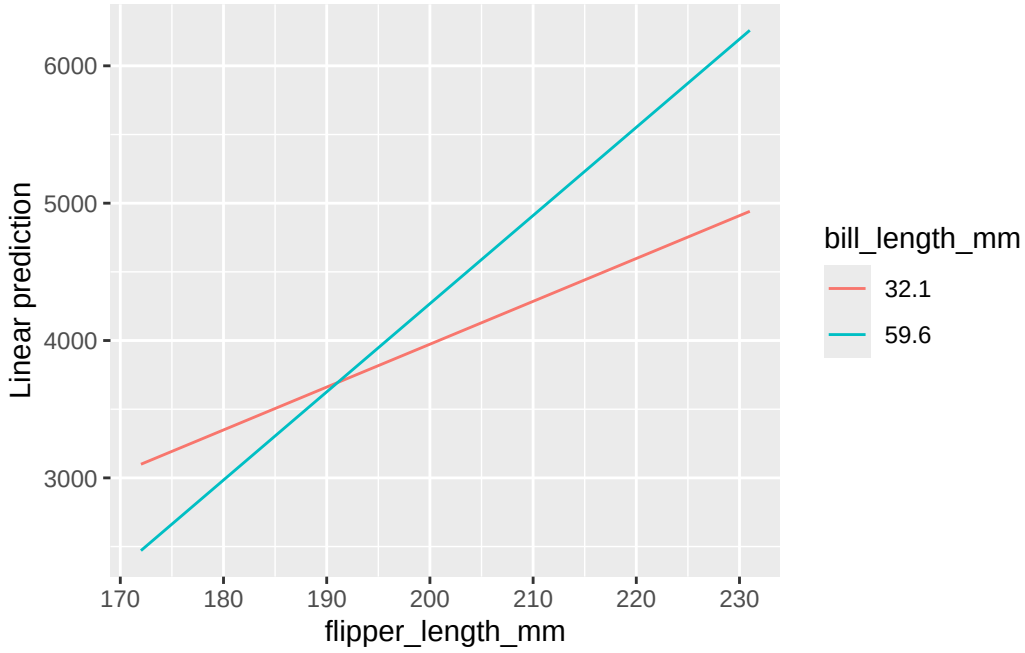
Viewing interactions

Use the `emmeans()` function to interpret interactions in a linear model. For continuous variables, you need to specify the range of the covariate with the `cov.reduce` argument – set to `range` to avoid the default of using the mean.

```
emmip(fit03, flipper_length_mm ~ bill_length_mm, cov.reduce = range)
```




```
emmip(fit03, bill_length_mm ~ flipper_length_mm, cov.reduce = range)
```



Other resources

- It might be worthwhile to use the `performance` package to assess model fit (including assumptions using `check_model()`).
- I use this a lot: the `interactions` package for visualising interactions in GLM models. However it is very technical and not for beginners – use if you are comfortable with R.
- The `gtsummary` package is great for summarising regression models using `tbl_regression()`, but you may need to tweak it further to get the output you want. Another package that can do something similar is the `sjPlot` package, using `tab_model()`. **Alternatively, you can manually create the table (sometimes it can be easier to copy numbers depending on your level of expertise).**