Linear Additive Model

Introduction

The linear additive model may seem a very esoteric concept in data science, but it is critical to understand how we make predictions from statistical models, and how we test whether those models significantly explain the variance in our observations. It also underscores an important concept in statistical modelling: the assumption that the different effects that combine to explain an observed value are *additive*.

Case Study: Barley Yield

```
barley = read.csv("data/barley.csv")
head(barley)
```

```
##   block gen yield
## 1     1 new 291.1
## 2     1 old 223.9
## 3     2 new 321.4
## 4     2 old 249.9
## 5     3 new 399.3
## 6     3 old 330.8
```

How would we go about constructing the linear model for this dataset? First, we need to calculate the population mean for yield. Remember, we use "barley$yield" to tell R to calculate the mean for the yield column of the barley dataset.

```
barley_mu = mean(barley$yield)
barley_mu
```

```
## [1] 295.2525
```

Now we need to calculate the means for the new and old genotypes. We need to calculate these separately, so we will need to create subsets for the old and new genotypes. We can do this most intuitively by using the *tidyverse* package and the *filter()* function. Remember the pipe ("%>%") command feeds the results of one line of code to the next line. The first line of the code below creates a new dataset, "barley_old", that is initially equal to the barley dataset.

In the second line of code, however, we use the *filter()* function to subset the data. We tell R to filter the data to those rows where the value in the "gen" column is equal to "old". Whenever we use the filter function, we need to use two equal signs – believe me, it is easy to forget and just use one!

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
barley_old = barley %>%
  filter(gen=="old")
```

Now we can calculate the mean across all observations of the old barley genotype.

```
barley_old_mean = mean(barley_old$yield)
barley_old_mean
```

```
## [1] 271.82
```

Finally, we can subtract the population mean from the mean yield of the old barley genotype to calculate the old genotype's effect.

```
barley_old_effect = barley_old_mean - barley_mu
barley_old_effect
```

```
## [1] -23.4325
```