dplyr tutorial

For Lab 3

1-27-2025

In Lab 3, there are many possible ways to extract what you need from a dataframe in order to perform ANOVA from scratch in Part 3, but one approach would be to utilize `dplyr` commands. This will be the easiest way in my opinion, but if you get a different approach to work, that would be totally fine as well.

Since we have varying levels of familiarity with `dplyr` among the students in this class (and even those with familiarity may need a refresher), I would like the TAs to start discussion sections by running through this document on general help using `dplyr`. We will demonstrate this on the `PlantGrowth` dataset, with the purpose of showing how to get particular quantities that you need for ANOVA.

As a sidenote, Chapter 3 of the ModernDive textbook mentioned in class will be a useful reference for all of this (moderndive.com/v2).

The pipe operator %>%

One of the cornerstones of the `dplyr` package is the pipe operator: `%>%`

The way that this works is that it takes what precedes it, and passes it to the next statement. Let's load the `PlantGrowth` dataset and see it in action. The `PlantGrowth` dataset is one that is built-in and included with R; it is experimental data with three treatment groups and an outcome variable of the weight of the resulting plant.

You may either put the following into an R Script file (and run it interactively in the Console), or an R Markdown file (and knit to see your output).

```
library(dplyr)
data(PlantGrowth)

# A simple example with the pipe operator
example <- PlantGrowth %>%
  summarize(y_bar = mean(weight))

print(example)
```

```
##   y_bar
## 1 5.073
```

Here, the dataset called `PlantGrowth` is passed to the `summarize` function in the next line, which then finds the mean of the `weight` variable.

A couple notes:

- There is a keyboard shortcut for the pipe operator: on a PC, it is ctrl-shift-M; on a Mac, it is Cmd-shift-M.

- In the code above, `mean` calculates the mean and then it is stored into `y_bar` within the `example` object

So, we can also specifically refer to the mean value with:

```
example$y_bar
```

```
## [1] 5.073
```

The group_by function

When calculating the necessary quantities for ANOVA, you will need to calculate the means, variances and sample sizes for each group that are in the input dataframe. The `group_by` function makes this very straightforward to do. We can see this again on the `PlantGrowth` dataset.

In the `PlantGrowth` dataset, there is a treatment variable called `group`. We can find the mean, variance and sample size of the `weight` variable within each group as follows:

```
summary <- PlantGrowth %>%
  group_by(group) %>%
  summarize(y_bar_i = mean(weight),
            S2_i = var(weight),
            n_i = n())

print(summary)
```

```
## # A tibble: 3 x 4
##   group y_bar_i  S2_i   n_i
##   <fct>   <dbl> <dbl> <int>
## 1 ctrl     5.03 0.340    10
## 2 trt1     4.66 0.630    10
## 3 trt2     5.53 0.196    10
```

You will need to similarly get the means, variances and sample size within each group for the ANOVA quantities, and you can now follow the above code to do so.

In the above code, similar to the previous example, I have given names of `y_bar_i`, `S2_i` and `n_i` to the means, variances and sample sizes respectively. So for example, I can specifically call out all of the sample variances at the same time with:

```
summary$S2_i
```

```
## [1] 0.3399956 0.6299211 0.1958711
```

Calculating SSE

Recall from Lecture 5 Slides 12-13 that we could calculate SS_E as:

$$SS_E = \sum_{i=1}^{a}(n-1)S_i^2$$

where S_i^2 is the sample variance of the i^{th} group. R can do this very easily for us. As seen just above, the variance of each group can be called as a vector with `summary$S2_i`. We can then easily multiply each of these by their respective sample sizes minus 1, elementwise:

```
(summary$n_i - 1) * summary$S2_i
```

```
## [1] 3.05996 5.66929 1.76284
```

This gives each value inside the summation in the equation above; then to sum them, all we need to do is use the `sum` function:

```
sum((summary$n_i - 1) * summary$S2_i)
```

```
## [1] 10.49209
```

This is the SS_E for the `weight` variable in the `PlantGrowth` dataset. You thus need to do a similar thing for any other dataset to get SS_E on it.

Get Df

We need some way to get the needed values to find the numerator Df and denominator Df. Recall from Lecture 5 that the numerator df is $a-1$ where a is the number of groups. In the `PlantGrowth` dataset, there are 3 groups as mentioned previously. One easy way to get this is to look at the size of the `summary` object:

```
print(summary)
```

```
## # A tibble: 3 x 4
##   group y_bar_i  S2_i   n_i
##   <fct>   <dbl> <dbl> <int>
## 1 ctrl     5.03 0.340    10
## 2 trt1     4.66 0.630    10
## 3 trt2     5.53 0.196    10
```

This has 3 rows, where each row corresponds to a group. You can thus of course hardcode $a = 3$, but if you wanted to have R figure that out on its own, we can use the `dim` function:

```
dim(summary)
```

```
## [1] 3 4
```

The first number is the number of rows; the second number is the number of columns. So,

```
dim(summary)[1]
```

```
## [1] 3
```

gives the number of rows, which is again the number of groups in the dataset. You could then just store this as the variable `a` for your df calculations (but again you are welcome to simply hardcode it if you desire).

Recall also that the denominator df is $N-a$ where N is the total sample size. We can then just use the `dim` function on the entire dataframe:

```r
dim(PlantGrowth)[1]
```

```
## [1] 30
```

So this would be `N` and could be stored as such for your calculations.

Calculating the ANOVA quantities on any other dataset is then just a matter of putting all of these pieces together with modifications as appropriate, and a few other calculations that should either follow from what was demonstrated above or otherwise be relatively straightforward.