# R Workshop

Lecture 5: More advanced functionality in R and R Studio, Part II

May 17, 2024

## 1 Regular expressions

Often, when you are working with large amounts of data, and are trying to subset based on some aspects of your data, you have restrictons on strings that you want to implement. For example, you want to subset to all words/sentences that begin with a "p", or all words/sentences that have the sequence "can" or all words/sentences that have exactly 4 letters. This is where regular expressions help. Now, I can't teach you everything about regular expressions in a few minutes, but I want to give you a flavour of what's possible.

As with all things in R, there are multiple ways to do this in R, and I will teach you one way using the `grepl()` function. I'll use a built in data set — **iris**, which gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.

With the following, we can filter to all the rows where the column *Species* has an "e" in them.

```
iris %>%
  filter(grepl(pattern="e",x=Species))
```

If you want to rows where the *Species* start with a "v" (for example), you need to let the `grepl()` function know that the sequence you are looking for is at the beginning using a carat "^".

With the following, we can filter to all the rows where the column *Species* has an "e" in them.

```
iris %>%
  filter(grepl(pattern="^v",x=Species))
```

What if you wanted any word that starts with a "vi" or an "se"?

```
iris %>%
  filter(grepl(pattern="^(vi|se)",x=Species))
```

What if you want all the words with a certain number of letters? The "." marks a letter that you don't know. Below, I will filter to all the words with 6 letters. You can use the beginning ("^") and ending ("$") symbols along with the wild-card (".") character.

```
iris %>%
  filter(grepl(pattern="^......$",x=Species))
```

There is so much more...See here.

## 2 Converting R output to LaTeX code with `stargazer()`

**stargazer** is a great package. If you use the main function `stargazer()` with a `data.frame`, then it will spit out a summary of that `data.frame`.

I will use another built in data set — **mtcars**, which has fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973–74 models).

```
#To get
# Stargazer can output any data.frame in LaTeX code
mtcars %>%
  stargazer()
```

The output of the relevant LATEX code is the following. Note, the caption is empty, but you can fill it with what you want.

Table 1:

| Statistic | N | Mean | St. Dev. | Min | Max |
|---|---|---|---|---|---|
| mpg | 32 | 20.091 | 6.027 | 10.400 | 33.900 |
| cyl | 32 | 6.188 | 1.786 | 4 | 8 |
| disp | 32 | 230.722 | 123.939 | 71.100 | 472.000 |
| hp | 32 | 146.688 | 68.563 | 52 | 335 |
| drat | 32 | 3.597 | 0.535 | 2.760 | 4.930 |
| wt | 32 | 3.217 | 0.978 | 1.513 | 5.424 |
| qsec | 32 | 17.849 | 1.787 | 14.500 | 22.900 |
| vs | 32 | 0.438 | 0.504 | 0 | 1 |
| am | 32 | 0.406 | 0.499 | 0 | 1 |
| gear | 32 | 3.688 | 0.738 | 3 | 5 |
| carb | 32 | 2.812 | 1.615 | 1 | 8 |

If you want the original `data.frame`, then set the *summary* argument to F. You can also control other arguments like *digits* — please look at the Help for all the arguments.

```
#To get
# Stargazer can output any data.frame in LaTeX code
mtcars %>%
  stargazer(summary=F,digits=1)
```

The output of the above code is the following:

Table 2:

| | mpg | cyl | disp | hp | drat | wt | qsec | vs | am | gear | carb |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Mazda RX4 | 21 | 6 | 160 | 110 | 3.9 | 2.6 | 16.5 | 0 | 1 | 4 | 4 |
| Mazda RX4 Wag | 21 | 6 | 160 | 110 | 3.9 | 2.9 | 17.0 | 0 | 1 | 4 | 4 |
| Datsun 710 | 22.8 | 4 | 108 | 93 | 3.9 | 2.3 | 18.6 | 1 | 1 | 4 | 1 |
| Hornet 4 Drive | 21.4 | 6 | 258 | 110 | 3.1 | 3.2 | 19.4 | 1 | 0 | 3 | 1 |
| Hornet Sportabout | 18.7 | 8 | 360 | 175 | 3.1 | 3.4 | 17.0 | 0 | 0 | 3 | 2 |
| Valiant | 18.1 | 6 | 225 | 105 | 2.8 | 3.5 | 20.2 | 1 | 0 | 3 | 1 |
| Duster 360 | 14.3 | 8 | 360 | 245 | 3.2 | 3.6 | 15.8 | 0 | 0 | 3 | 4 |
| Merc 240D | 24.4 | 4 | 146.7 | 62 | 3.7 | 3.2 | 20 | 1 | 0 | 4 | 2 |
| Merc 230 | 22.8 | 4 | 140.8 | 95 | 3.9 | 3.1 | 22.9 | 1 | 0 | 4 | 2 |
| Merc 280 | 19.2 | 6 | 167.6 | 123 | 3.9 | 3.4 | 18.3 | 1 | 0 | 4 | 4 |
| Merc 280C | 17.8 | 6 | 167.6 | 123 | 3.9 | 3.4 | 18.9 | 1 | 0 | 4 | 4 |
| Merc 450SE | 16.4 | 8 | 275.8 | 180 | 3.1 | 4.1 | 17.4 | 0 | 0 | 3 | 3 |
| Merc 450SL | 17.3 | 8 | 275.8 | 180 | 3.1 | 3.7 | 17.6 | 0 | 0 | 3 | 3 |
| Merc 450SLC | 15.2 | 8 | 275.8 | 180 | 3.1 | 3.8 | 18 | 0 | 0 | 3 | 3 |
| Cadillac Fleetwood | 10.4 | 8 | 472 | 205 | 2.9 | 5.2 | 18.0 | 0 | 0 | 3 | 4 |
| Lincoln Continental | 10.4 | 8 | 460 | 215 | 3 | 5.4 | 17.8 | 0 | 0 | 3 | 4 |
| Chrysler Imperial | 14.7 | 8 | 440 | 230 | 3.2 | 5.3 | 17.4 | 0 | 0 | 3 | 4 |
| Fiat 128 | 32.4 | 4 | 78.7 | 66 | 4.1 | 2.2 | 19.5 | 1 | 1 | 4 | 1 |
| Honda Civic | 30.4 | 4 | 75.7 | 52 | 4.9 | 1.6 | 18.5 | 1 | 1 | 4 | 2 |
| Toyota Corolla | 33.9 | 4 | 71.1 | 65 | 4.2 | 1.8 | 19.9 | 1 | 1 | 4 | 1 |
| Toyota Corona | 21.5 | 4 | 120.1 | 97 | 3.7 | 2.5 | 20.0 | 1 | 0 | 3 | 1 |
| Dodge Challenger | 15.5 | 8 | 318 | 150 | 2.8 | 3.5 | 16.9 | 0 | 0 | 3 | 2 |
| AMC Javelin | 15.2 | 8 | 304 | 150 | 3.1 | 3.4 | 17.3 | 0 | 0 | 3 | 2 |
| Camaro Z28 | 13.3 | 8 | 350 | 245 | 3.7 | 3.8 | 15.4 | 0 | 0 | 3 | 4 |
| Pontiac Firebird | 19.2 | 8 | 400 | 175 | 3.1 | 3.8 | 17.0 | 0 | 0 | 3 | 2 |
| Fiat X1-9 | 27.3 | 4 | 79 | 66 | 4.1 | 1.9 | 18.9 | 1 | 1 | 4 | 1 |
| Porsche 914-2 | 26 | 4 | 120.3 | 91 | 4.4 | 2.1 | 16.7 | 0 | 1 | 5 | 2 |
| Lotus Europa | 30.4 | 4 | 95.1 | 113 | 3.8 | 1.5 | 16.9 | 1 | 1 | 5 | 2 |
| Ford Pantera L | 15.8 | 8 | 351 | 264 | 4.2 | 3.2 | 14.5 | 0 | 1 | 5 | 4 |
| Ferrari Dino | 19.7 | 6 | 145 | 175 | 3.6 | 2.8 | 15.5 | 0 | 1 | 5 | 6 |
| Maserati Bora | 15 | 8 | 301 | 335 | 3.5 | 3.6 | 14.6 | 0 | 1 | 5 | 8 |
| Volvo 142E | 21.4 | 4 | 121 | 109 | 4.1 | 2.8 | 18.6 | 1 | 1 | 4 | 2 |

```r
#First, get the output of a model
model = mtcars %>%
  lmer(formula=mpg~gear*wt + (1|am))

#Now, we have to extrac the model output
#Then convert that into a data.frame
#Then, we can use the stargazer() function
summary(model)$coefficients %>%
  data.frame() %>%
  stargazer(summary=F,digits=1)
```

The output of the above code is the following:

Table 3:

|              | Estimate | Std..Error | t.value |
| ------------ | -------- | ---------- | ------- |
| (Intercept)  | 17.4     | 11.1       | 1.6     |
| gear         | 5.6      | 2.9        | 1.9     |
| wt           | 1.4      | 3.3        | 0.4     |
| gear:wt      | -2.0     | 0.9        | -2.2    |