Demystifying R Part 2

You might see a warning message just above this file. Something like… "R Markdown requires the knitr package (version 1.2 or higher)" Don't worry about this for now. We'll address it at the end of this file.

1. Run the following command to see what it does.

```
summary(mtcars)
```

```
##       mpg             cyl             disp             hp       
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0  
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7  
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0  
##       drat             wt             qsec             vs        
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000  
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000  
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000  
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375  
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000  
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000  
##        am              gear            carb      
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000  
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
##  Median :0.0000   Median :4.000   Median :2.000  
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812  
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000  
```

If you know about quantiles, then the output should look familiar. If not, you probably recognize the min (minimum), median, mean, and max (maximum). We'll go over quantiles in Lesson 3 so don't worry if the output seems overwhelming.

The str() and summary() functions are helpful commands when working with a new data set. The str() function gives us the variable names and their types. The summary() function gives us an idea of the values a variable can take on.

2. In 2013, the average mpg (miles per gallon) for a car was 23 mpg. The car models in the mtcars data set come from the year 1973-1974. Subset the data so that you create a new data frame that contains cars that get 23 or more mpg (miles per gallon). Save it to a new data frame called efficient.
3. How many cars get more than 23 mpg? Use one of the commands you learned in the demystifying.R to answer this question.

```
subset(mtcars, mpg > 23)
```

```
##                 mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Merc 240D      24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
```

4. We can also use logical operators to find out which car(s) get greater than 30 miles per gallon (mpg) and have more than 100 raw horsepower.

```
subset(mtcars, mpg > 30 & hp > 100)
```

```
##               mpg cyl disp  hp drat    wt qsec vs am gear carb
## Lotus Europa 30.4   4 95.1 113 3.77 1.513 16.9  1  1    5    2
```

There's only one car that gets more than 30 mpg and 100 hp.

5. What do you think this code does? Scroll down for the answer.

```
subset(mtcars, mpg < 14 | disp > 390)
```

```
##                      mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Cadillac Fleetwood  10.4   8  472 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8  460 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8  440 230 3.23 5.345 17.42  0  0    3    4
## Camaro Z28          13.3   8  350 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8  400 175 3.08 3.845 17.05  0  0    3    2
```

Note: You may be familiar with the || operator in Java. R uses one single & for the logical operator AND. It also uses one | for the logical operator OR.

The command above creates a data frame of cars that have mpg less than 14 OR a displacement of more than 390. Only one of the conditions for a car needs to be satisfied so that the car makes it into the subset. Any of the cars that fit the criteria are printed to the console.

Now you try some.

6. Print the cars that have a 1/4 mile time (qsec) less than or equal to 16.90 seconds to the console.

```
subset(mtcars, qsec == 1/4 |qsec <= 16.90)
```

```
##                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4        21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Duster 360       14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Dodge Challenger 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## Camaro Z28       13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Porsche 914-2    26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa     30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L   15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino     19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora    15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
```

7. Save the subset of cars that weigh under 2000 pounds (weight is measured in lb/1000) to a variable called lightCars. Print the numbers of cars and the subset to the console.

```r
lightCars <- subset(mtcars, wt < 2)
lightCars
```

```
##                 mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Honda Civic    30.4   4 75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4 71.1  65 4.22 1.835 19.90  1  1    4    1
## Fiat X1-9      27.3   4 79.0  66 4.08 1.935 18.90  1  1    4    1
## Lotus Europa   30.4   4 95.1 113 3.77 1.513 16.90  1  1    5    2
```

8. You can also create new variables in a data frame. Let's say you wanted to have the year of each car's model. We can create the variable mtcars$year. Here we'll assume that all of the models were from 1974. Run the code below.

```r
mtcars$year <- 1974
mtcars$year
```

```
##  [1] 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974
## [16] 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974 1974
## [31] 1974 1974
```

Notice how the number of variables changed in the work space. You can also see the result by double clicking on mtcars in the workspace and examining the data in a table.

To drop a variable, subset the data frame and select the variable you want to drop with a negative sign in front of it.

```r
mtcars <- subset(mtcars, select = -year)
mtcars
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

Notice, we are back to 11 variables in the data frame.

9. What do you think this code does? Run it to find out.

```r
mtcars$year <- c(1973, 1974)
mtcars$year
```

```
##  [1] 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973
## [16] 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974 1973 1974
## [31] 1973 1974
```

Open the table of values to see what values year takes on.

Drop the year variable from the data set.

10. Now you are going to get a preview of ifelse(). For those new to programming this example may be confusing. See if you can understand the code by running the commands one line at a time. Read the output and make sense of what the code is doing at each step.

If you are having trouble don't worry, we will review the ifelse statement at the end of Lesson 3. You won't be quizzed on it, and it's not essential to keep going in this course. We just want you to try to get familiar with more code.

```r
mtcars$wt
```

```
##  [1] 2.620 2.875 2.320 3.215 3.440 3.460 3.570 3.190 3.150 3.440 3.440 4.070
## [13] 3.730 3.780 5.250 5.424 5.345 2.200 1.615 1.835 2.465 3.520 3.435 3.840
## [25] 3.845 1.935 2.140 1.513 3.170 2.770 3.570 2.780
```

```r
cond <- mtcars$wt < 3
cond
```

```
##  [1]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [13] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE
## [25] FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE
```

```r
mtcars$weight_class <- ifelse(cond, 'light', 'average')
mtcars$weight_class
```

```
##  [1] "light"   "light"   "light"   "average" "average" "average" "average"
##  [8] "average" "average" "average" "average" "average" "average" "average"
## [15] "average" "average" "average" "light"   "light"   "light"   "light"
## [22] "average" "average" "average" "average" "light"   "light"   "light"
## [29] "average" "light"   "average" "light"
```

```
cond <- mtcars$wt > 3.5
mtcars$weight_class <- ifelse(cond, 'heavy', mtcars$weight_class)
mtcars$weight_class
```

```
##  [1] "light"   "light"   "light"   "average" "average" "average" "heavy"
##  [8] "average" "average" "average" "average" "heavy"   "heavy"   "heavy"
## [15] "heavy"   "heavy"   "heavy"   "light"   "light"   "light"   "light"
## [22] "heavy"   "average" "heavy"   "heavy"   "light"   "light"   "light"
## [29] "average" "light"   "heavy"   "light"
```

You have some variables in your workspace or environment like 'cond' and efficient. You want to be careful that you don't bring in too much data into R at once since R will hold all the data in working memory. We have nothing to worry about here, but let's delete those variables from the work space.

```
rm(cond)
rm(efficient)
```

```
## Warning in rm(efficient): object 'efficient' not found
```

Save this file if you haven't done so yet.

You'll have the opportunity to create one Rmd file for the final project in this class and submit the Rmd file and knitted output (or HTML file). You'll need the knitr package to do that so let's install that now. **Uncomment** the following two lines of code and run them.

```
# install.packages('knitr', dependencies = T)
# library(knitr)
```

Once you've installed knitr, **comment** out the two lines of code above. When you click the **Knit HTML** button a web page will be generated that includes both content (text and text formatting from Markdown) as well as the output of any embedded R code chunks within the document.

You've reached the end of the file so now it's time to write some code to answer a question to continue on in Lesson 2.

Which car(s) have an mpg (miles per gallon) greater than or equal to 30 OR hp (horsepower) less than 60? Create an R chunk of code to answer the question.

```
subset(mtcars, mpg >= 30)
```

```
##                 mpg cyl disp  hp drat    wt  qsec vs am gear carb year
## Fiat 128       32.4   4 78.7  66 4.08 2.200 19.47  1  1    4    1 1974
## Honda Civic    30.4   4 75.7  52 4.93 1.615 18.52  1  1    4    2 1973
## Toyota Corolla 33.9   4 71.1  65 4.22 1.835 19.90  1  1    4    1 1974
## Lotus Europa   30.4   4 95.1 113 3.77 1.513 16.90  1  1    5    2 1974
##                weight_class
## Fiat 128              light
## Honda Civic           light
## Toyota Corolla        light
## Lotus Europa          light
```

```
subset(mtcars, hp < 60)
```

```
##              mpg cyl disp hp drat    wt  qsec vs am gear carb year weight_class
## Honda Civic 30.4   4 75.7 52 4.93 1.615 18.52  1  1    4    2 1973        light
```

Once you have the answer, go the Udacity website to continue with Lesson 2.

Note: You use brackets around text followed by two parentheses to create a link. There must be no spaces between the brackets and the parentheses. Paste or type the link into the parentheses. This also works on the discussions!

And if you want to see all of your HARD WORK from this file, click the **KNIT HTML** button now. (You may or may not need to restart R).

CONGRATULATIONS

You'll be exploring data soon with your new knowledge of R.