Lab #1 - Gapminder Dataset

Econ 224

August 28th, 2018

Installing Required Packages

Welcome to the first lab of Econ 224! Today we'll be giving you a crash course in two R packages that we'll be using throughout the semester: `dplyr` and `ggplot2`. Before we can get started, you'll need to install both of these packages. A quick way to install both of them at once, along with several other packages that may come in handy later, is `install.packages('tidyverse')`. Note that you only need to do this *once*. The dataset we'll work with today is also available as an R package called `gapminder`. Make sure that you have both `tidyverse` and `gapminder` installed before continuing.

The Gapminder Dataset

Our next step is to load both `tidyverse`, which contains `dplyr` and `ggplot2`, and `gapminder`, which contains the data we'll be analyzing today:

```r
library(tidyverse)
library(gapminder)
```

Exercise #1

Now that you've loaded `gapminder`, use the command `?gapminder` to view the R help file for this dataset and read the documentation you find there and answer the following questions:

- What information does this dataset contain?
- How may rows and columns does it have?
- What are the names of each of the columns, and what information does each contain?
- What is the source of the dataset?

Solution to Exercise # 1

Answer Goes Here

What is a tibble?

Let's see what happens if we display the `gapminder` dataset:

```r
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

If you're used to working with dataframes in R, this may surprise you. Rather than filling up the screen with lots of useless information, R shows us a helpful summary of the information contained in `gapminder`. This is because `gapminder` is *not* a dataframe; it's a *tibble*. For the moment, all you need to know about tibbles is that they are souped up versions of R dataframes that are designed to work seamlessly with `dplyr`. (If you want to learn more, see Chapter 7 of *R for Data Science*) But what is `dplyr` in the first place?

What is `dplyr`?

The `dplyr` package provides a number of powerful but easy-to-use tools for data manipulation in R. We'll be making heavy use of this package throughout the semester. Rather than trying to explain everything in advance, let's just dive straight in.

Filter Rows with `filter`

Let's run the following command in R and see what happens:

```r
gapminder %>% filter(year == 2007)
```

```
## # A tibble: 142 x 6
##    country     continent  year lifeExp       pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Afghanistan Asia       2007    43.8  31889923      975.
##  2 Albania     Europe     2007    76.4   3600523     5937.
##  3 Algeria     Africa     2007    72.3  33333216     6223.
##  4 Angola      Africa     2007    42.7  12420476     4797.
##  5 Argentina   Americas   2007    75.3  40301927    12779.
##  6 Australia   Oceania    2007    81.2  20434176    34435.
##  7 Austria     Europe     2007    79.8   8199783    36126.
##  8 Bahrain     Asia       2007    75.6    708573    29796.
##  9 Bangladesh  Asia       2007    64.1 150448339     1391.
## 10 Belgium     Europe     2007    79.4  10392226    33693.
## # ... with 132 more rows
```

Compare the results of running this command to what we got when we typed `gapminder` into the console above. Rather than displaying the whole dataset, now R is only showing us the 142 rows for which the column `year` has a value of `2007`.

So how does this work? The `%>%` symbol is called a *pipe*. Pipes play very nicely with `dplyr` and make our code very easy to understand. The tibble `gapminder` is being piped into the function `filter()`. The argument `year == 2007` tells `filter()` that it should find all the rows such that the logical condition `year == 2007` is `TRUE`.

Oh no! Have we accidentally deleted all of the other rows of `gapminder`? Nope: we haven't made any changes to `gapminder` at all. If you don't believe me try entering `gapminder` at the console. All that this command does is *display* a subset of `gapminder`. If we wanted to store the result of running this command, we'd need to assign it to a variable, for example

```
gapminder2007 <- gapminder %>% filter(year == 2007)
gapminder2007
```

```
## # A tibble: 142 x 6
##    country     continent  year lifeExp       pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Afghanistan Asia       2007    43.8  31889923      975.
##  2 Albania     Europe     2007    76.4   3600523     5937.
##  3 Algeria     Africa     2007    72.3  33333216     6223.
##  4 Angola      Africa     2007    42.7  12420476     4797.
##  5 Argentina   Americas   2007    75.3  40301927    12779.
##  6 Australia   Oceania    2007    81.2  20434176    34435.
##  7 Austria     Europe     2007    79.8   8199783    36126.
##  8 Bahrain     Asia       2007    75.6    708573    29796.
##  9 Bangladesh  Asia       2007    64.1 150448339     1391.
## 10 Belgium     Europe     2007    79.4  10392226    33693.
## # ... with 132 more rows
```

Exercise #2

1. Explain the difference between `x = 3` and `x == 3` in R.
2. Use `filter` to choose the subset of `gapminder` for which `year` is 2002.
3. If you instead try to choose the subset with `year` equal to 2005, something will go wrong. Try it and explain what happens and why.
4. Store the data for Asian countries in a tibble called `gapminder_asia`. Display this tibble.

Solution to Exercise #2

1. The first assigns the value `3` to the variable `x`; the second tests whether `x` is equal to `3` and returns either `TRUE` or `FALSE`.
2. Use the following code:

```
gapminder %>% filter(year == 2002)
```

```
## # A tibble: 142 x 6
##    country     continent  year lifeExp       pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>     <int>     <dbl>
##  1 Afghanistan Asia       2002    42.1  25268405      727.
##  2 Albania     Europe     2002    75.7   3508512     4604.
##  3 Algeria     Africa     2002    71.0  31287142     5288.
##  4 Angola      Africa     2002    41.0  10866106     2773.
##  5 Argentina   Americas   2002    74.3  38331121     8798.
##  6 Australia   Oceania    2002    80.4  19546792    30688.
##  7 Austria     Europe     2002    79.0   8148312    32418.
##  8 Bahrain     Asia       2002    74.8    656397    23404.
##  9 Bangladesh  Asia       2002    62.0 135656790     1136.
## 10 Belgium     Europe     2002    78.3  10311970    30486.
## # ... with 132 more rows
```

3. If you go back to the help file for `gapminder` you'll see that it only contains data for every fifth year. The year 2005 isn't in our dataset so `dplyr` will display an empty tibble:

```
gapminder %>% filter(year == 2005)
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: country <fct>, continent <fct>, year <int>,
## #   lifeExp <dbl>, pop <int>, gdpPercap <dbl>
```

4. Use the following code:

```
gapminder_asia <- gapminder %>% filter(continent == 'Asia')
gapminder_asia
```

```
## # A tibble: 396 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 386 more rows
```

Filtering two variables

We can use `filter` to subset on two or more variables. For example, here we display data for the US in 2007:

```
gapminder %>% filter(year == 2007, country == 'United States')
```

```
## # A tibble: 1 x 6
##   country       continent  year lifeExp       pop gdpPercap
##   <fct>         <fct>     <int>   <dbl>     <int>     <dbl>
## 1 United States Americas   2007    78.2 301139947    42952.
```

Exercise # 3

1. When I displayed data for the US in 2007, I put quotes around `United States` but not around `year`. Explain why.
2. Which country had the higher life expectancy in 1977: Ireland or Brazil? Which had the higher GDP per capita?

Solution to Exercise # 3

1. This is because `year` contains numeric data while `country` contains character data, aka string data.
2. From the results of the following code, we see that Ireland had both a higher life expectancy and GDP per capita.

```
gapminder %>% filter(year == 1977, country == 'Ireland')
```

```
## # A tibble: 1 x 6
##   country continent  year lifeExp     pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Ireland Europe     1977    72.0 3271900    11151.
```

```
gapminder %>% filter(year == 1977, country == 'Brazil')
```

```
## # A tibble: 1 x 6
##   country continent  year lifeExp       pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>     <int>     <dbl>
## 1 Brazil  Americas   1977    61.5 114313951     6660.
```