Gapminder / Github Homework

March 4, 2020

This homework will combine many of the skills you have learned up to this point. It will take you through many steps one at a time to build up a nice analysis.

References:

* Book chapter on many models: <https://r4ds.had.co.nz/many-models.html>
* Gapminder youtube video: <https://www.youtube.com/watch?v=jbkSRLYSojo>
* Gapminder website: <https://www.gapminder.org>
* Gapminder data: <https://www.gapminder.org/data/>

You will download data for average systolic blood pressure for women and the GDP per capita (both from World bank and the alternative GCP per capita) and any others you are interested in and build a new gapminder tibble. Note that the output shown below is not complete. We will fix that.

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

**Part 1: Creating our own gapminder animation**

1. Read data in and put into a single tibble.

Always determine if the current working directory has a data folder.

If not, make one.

if(!file.exists("./data")) {dir.create("./data")}

Go to the gapminder website and find the URLs we discussed in class for alt\_GDP, women’s blood pressure, GDP per capita, life expectancy, population and any others you want to look at. For alt GDP, use the entry from PWT6.2 titled Alternative GDP/capita (PPP$, inflation-adjusted) from PWT

fileUrls <- c("https://docs.google.com/spreadsheet/pub?key=0AkBd6lyS3EmpdHo5S0J6ekhVOF9QaVhod05QSGV4T3c&output=xlsx",

"https://docs.google.com/spreadsheet/pub?key=phAwcNAVuyj2tPLxKvvnNPA&output=xlsx”, "https://docs.google.com/spreadsheet/pub?key=tSUr\_yZVbM6a3AGJEq\_Z2Pw&output=xlsx", "https://docs.google.com/spreadsheet/pub?key=0ArfEDsV3bBwCdHBzUVVSMDlTX1ZCUnNJQ3ZFdkFXVFE&output=xlsx",

“<https://docs.google.com/spreadsheet/pub?key=phAwcNAVuyj0XOoBL_n5tAQ&output=xlsx>”,

"<https://docs.google.com/spreadsheet/pub?key=0AkBd6lyS3EmpdHo5S0J6ekhVOF9QaVhod05QSGV4T3c&output=xlsx>”)

var\_names <- c("GDP","life\_expectancy", "alt\_GDP", "blood press", "population")

*Fill in the function get\_clean to download and read in the excel file from the url provided and then put the data in a column with the variable name specified in var\_name*

get\_clean <- function(url\_in, var\_name){  
# you need to fill in this part – 1/2 point

}  
# testing line – try it out with the first entry in fileUrls and var\_names to get:

out1 <- get\_clean(fileUrls[1],var\_names[1])  
head(out1)

You should get something that looks similar to this:

## # A tibble: 6 x 3

## country year GDP

## <chr> <dbl> <dbl>

## 1 Albania 1980 1061.

## 2 Albania 1981 1100.

## 3 Albania 1982 1111.

## 4 Albania 1983 1101.

## 5 Albania 1984 1065.

## 6 Albania 1985 1060.

# Now use map2 to apply this function to the entire list of urls and names.

all\_data <- map2(Fill in this part – 1/2 point)  
head(all\_data)

# your all\_data will be a list of 5 variables

## [[1]]

## # A tibble: 7,988 x 3

## country year GDP

## <chr> <dbl> <dbl>

## 1 Albania 1980 1061.

## 2 Albania 1981 1100.

## 3 Albania 1982 1111.

## 4 Albania 1983 1101.

## 5 Albania 1984 1065.

## 6 Albania 1985 1060.

## 7 Albania 1986 1092.

## 8 Albania 1987 1054.

## 9 Albania 1988 1014.

## 10 Albania 1989 1092.

## # ... with 7,978 more rows

##

## [[2]]

## # A tibble: 43,857 x 3

## country year life\_expectancy

## <chr> <dbl> <dbl>

## 1 Afghanistan 1800 28.2

## 2 Afghanistan 1801 28.2

## 3 Afghanistan 1802 28.2

## 4 Afghanistan 1803 28.2

## 5 Afghanistan 1804 28.2

## 6 Afghanistan 1805 28.2

## 7 Afghanistan 1806 28.2

## 8 Afghanistan 1807 28.1

## 9 Afghanistan 1808 28.1

## 10 Afghanistan 1809 28.1

## # ... with 43,847 more rows

##

## [[3]]

## # A tibble: 7,334 x 3

## country year alt\_GDP

## <chr> <dbl> <dbl>

## 1 Afghanistan 1970 1731.

## 2 Afghanistan 1971 1748.

## 3 Afghanistan 1972 2120.

## 4 Afghanistan 1973 2119.

## 5 Afghanistan 1974 2148.

## 6 Afghanistan 1975 2263.

## 7 Afghanistan 1976 2270.

## 8 Afghanistan 1977 2121.

## 9 Afghanistan 1978 2205.

## 10 Afghanistan 1979 2121.

## # ... with 7,324 more rows

##

## [[4]]

## # A tibble: 5,771 x 3

## country year `blood press`

## <chr> <dbl> <dbl>

## 1 Afghanistan 1980 122.

## 2 Afghanistan 1981 122.

## 3 Afghanistan 1982 122.

## 4 Afghanistan 1983 123.

## 5 Afghanistan 1984 123.

## 6 Afghanistan 1985 123.

## 7 Afghanistan 1986 123.

## 8 Afghanistan 1987 123.

## 9 Afghanistan 1988 124.

## 10 Afghanistan 1989 124.

## # ... with 5,761 more rows

##

## [[5]]

## # A tibble: 20,176 x 3

## country year population

## <chr> <dbl> <dbl>

## 1 Afghanistan 1800 3280000

## 2 Afghanistan 1810 3280000

## 3 Afghanistan 1820 3323519

## 4 Afghanistan 1830 3448982

## 5 Afghanistan 1840 3625022

## 6 Afghanistan 1850 3810047

## 7 Afghanistan 1860 3973968

## 8 Afghanistan 1870 4169690

## 9 Afghanistan 1880 4419695

## 10 Afghanistan 1890 4710171

## # ... with 20,166 more rows

1. Join the outputs into one tibble with a column for each variable (hint – perhaps use one of your purrr functions)

joined\_data <-   
# you need to fill in this part (1 pt) –

# ½ pt for not losing years, 1/2 pt for getting into correct format

1. Next we will add the continent data. Use the countrycode package to add a column called continent to your dataset (or use gapminder to add them). Either way, identify any countries without a continent showing and remedy this. Save your result to new\_gapminder.

new\_gapminder <- # you need to fill in this part (1 pts)

# 1/2 pt for getting continents added without losing years in your data,

# ½ point for proving that you have all continents filled in

new\_gapminder %>%  
 arrange(country, year)

## # A tibble: 48,270 x 8

## country year GDP life\_expectancy alt\_GDP `blood press` population

## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>

## 1 Afghan~ 1800 NA 28.2 NA NA 3280000

## 2 Afghan~ 1801 NA 28.2 NA NA NA

## 3 Afghan~ 1802 NA 28.2 NA NA NA

## 4 Afghan~ 1803 NA 28.2 NA NA NA

## 5 Afghan~ 1804 NA 28.2 NA NA NA

## 6 Afghan~ 1805 NA 28.2 NA NA NA

## 7 Afghan~ 1806 NA 28.2 NA NA NA

## 8 Afghan~ 1807 NA 28.1 NA NA NA

## 9 Afghan~ 1808 NA 28.1 NA NA NA

## 10 Afghan~ 1809 NA 28.1 NA NA NA

## # ... with 48,260 more rows, and 1 more variable: continent <chr>

1. Produce an animated plot across time life expectancy as a function of GDP with dot size by population of the country & colored by continent. (1 point). First keep only the variables you will need for plotting in a new tibble called plot\_data. Fill in any missing population data (look back at tidyr function fill) and leave out any rows that have an NA left in them. Feel free to do additional plots to analyze other variables if you want to. Note: the animation will not knit into anything except an html which will not upload to blackboard. So, run the animation for your own enjoyment and knit the document to either .pdf or .doc. We will grade your result based on the non-animated graph that will show when you knit it.

**Part 2: Analyze life expectancy as function of GDP**

1. Now, model life expectancy for each country as a function of GDP? (hint – use plot\_data from your plot above) (1/2 pt)
2. Graph your residuals by country and facet by continent. (1/2 pt)
3. Are there countries and continents for which this is a particularly bad model? Use broom::glance to make this determination and provide an explanation for your conclusions. (1 point)

Part 3:

For the gapminder data of life expectancy and year, a linear trend seems to be slightly too simple for the overall trend.

* 1. Transform year so that it has a mean of 0 (1/2 pt)
  2. Model with a quadratic polynomial. How can you interpret the coefficients of the quadratic for Belgium? (1 pts)
  3. Use glance() to identify all countries that do not fit the model well and plot the residuals for the countries that do not fit the model well. (1/2 pts)

**Part 4:**

Github (2 pts)

1. Create a github account and start a repo called myfirstrepo.
2. Put any files you would like in the repo (please not your homework solutions).
3. create a branch off the repo
4. change a file in the branch and update the master with the changed files
5. Create another repo that is a fork of someone else’s repo on github
6. Post a link to your github account where you submit your blackboard assignment