DSFTSW-Exam3

Alex Rigney

7/9/2020

Here I’m loading a lot of libraries, so I don’t have to do as much later.

#Library time!  
library(rio)

## Warning: package 'rio' was built under R version 4.0.2

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.0.2

## -- Attaching packages ----------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.1 v dplyr 1.0.0  
## v tidyr 1.1.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## Warning: package 'ggplot2' was built under R version 4.0.2

## Warning: package 'tidyr' was built under R version 4.0.2

## Warning: package 'readr' was built under R version 4.0.2

## Warning: package 'stringr' was built under R version 4.0.2

## -- Conflicts -------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(WDI)  
library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

library(labelled)

## Warning: package 'labelled' was built under R version 4.0.2

library(labelled)  
library(data.table)  
library(varhandle)  
library(ggrepel)

## Warning: package 'ggrepel' was built under R version 4.0.2

library(geosphere)

## Warning: package 'geosphere' was built under R version 4.0.2

library(rgeos)

## Warning: package 'rgeos' was built under R version 4.0.2

## Loading required package: sp

## Warning: package 'sp' was built under R version 4.0.2

## rgeos version: 0.5-3, (SVN revision 634)  
## GEOS runtime version: 3.8.0-CAPI-1.13.1   
## Linking to sp version: 1.4-2   
## Polygon checking: TRUE

library(viridis)

## Warning: package 'viridis' was built under R version 4.0.2

## Loading required package: viridisLite

library(mapview)

## Warning: package 'mapview' was built under R version 4.0.2

library(rnaturalearth)

## Warning: package 'rnaturalearth' was built under R version 4.0.2

library(rnaturalearthdata)

## Warning: package 'rnaturalearthdata' was built under R version 4.0.2

library(devtools)

## Warning: package 'devtools' was built under R version 4.0.2

## Loading required package: usethis

library(remotes)

## Warning: package 'remotes' was built under R version 4.0.2

##   
## Attaching package: 'remotes'

## The following objects are masked from 'package:devtools':  
##   
## dev\_package\_deps, install\_bioc, install\_bitbucket, install\_cran,  
## install\_deps, install\_dev, install\_git, install\_github,  
## install\_gitlab, install\_local, install\_svn, install\_url,  
## install\_version, update\_packages

## The following object is masked from 'package:usethis':  
##   
## git\_credentials

library(raster)

## Warning: package 'raster' was built under R version 4.0.2

##   
## Attaching package: 'raster'

## The following object is masked from 'package:data.table':  
##   
## shift

## The following object is masked from 'package:dplyr':  
##   
## select

## The following object is masked from 'package:tidyr':  
##   
## extract

library(sp)  
library(sf)

## Warning: package 'sf' was built under R version 4.0.2

## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1

library(Imap)

##   
## Attaching package: 'Imap'

## The following object is masked from 'package:purrr':  
##   
## imap

library(countrycode)

## Warning: package 'countrycode' was built under R version 4.0.2

library(shiny)

## Warning: package 'shiny' was built under R version 4.0.2

##   
## Attaching package: 'shiny'

## The following object is masked from 'package:geosphere':  
##   
## span

library(leaflet)

## Warning: package 'leaflet' was built under R version 4.0.2

library(shinydashboard)

## Warning: package 'shinydashboard' was built under R version 4.0.2

##   
## Attaching package: 'shinydashboard'

## The following object is masked from 'package:graphics':  
##   
## box

library(V8)

## Warning: package 'V8' was built under R version 4.0.2

## Using V8 engine 6.2.414.50

##   
## Attaching package: 'V8'

## The following object is masked from 'package:leaflet':  
##   
## JS

library(shinyjs)

## Warning: package 'shinyjs' was built under R version 4.0.2

##   
## Attaching package: 'shinyjs'

## The following object is masked from 'package:shiny':  
##   
## runExample

## The following object is masked from 'package:raster':  
##   
## click

## The following object is masked from 'package:rgeos':  
##   
## show

## The following object is masked from 'package:sp':  
##   
## show

## The following objects are masked from 'package:methods':  
##   
## removeClass, show

library(RColorBrewer)  
library(readr)  
library(tidyr)  
library(leaflet.extras)

## Warning: package 'leaflet.extras' was built under R version 4.0.2

library(htmltools)

## Warning: package 'htmltools' was built under R version 4.0.2

##   
## Attaching package: 'htmltools'

## The following object is masked from 'package:geosphere':  
##   
## span

library(shinyWidgets)

## Warning: package 'shinyWidgets' was built under R version 4.0.2

##   
## Attaching package: 'shinyWidgets'

## The following object is masked from 'package:shinyjs':  
##   
## alert

library(ggplot2)  
library(tidycensus)

## Warning: package 'tidycensus' was built under R version 4.0.2

library(rvest)

## Loading required package: xml2

##   
## Attaching package: 'rvest'

## The following object is masked from 'package:shinyjs':  
##   
## html

## The following object is masked from 'package:purrr':  
##   
## pluck

## The following object is masked from 'package:readr':  
##   
## guess\_encoding

library(XML)

## Warning: package 'XML' was built under R version 4.0.2

##   
## Attaching package: 'XML'

## The following object is masked from 'package:rvest':  
##   
## xml

library(tidytext)

## Warning: package 'tidytext' was built under R version 4.0.2

library(stringr)  
library(tidyverse)  
  
census\_api\_key("54ca5907890a0d821d64fa09074c575b35ee63e1")

## To install your API key for use in future sessions, run this function with `install = TRUE`.

1. Clear the environment. [5 points]

rm(list=ls(all=TRUE)) # clear environment

1. Use the tidycensus package to
2. find the inequality Gini index variable explained on the last exam,

#Load the data set!  
v15 <- load\_variables(year = 2015,  
"acs5")  
  
View(v15)  
  
#The Gini variable is: B19083\_001  
  
  
v10 <- load\_variables(year = 2010,  
"acs5")  
  
View(v10)  
  
#Gini for 2010: B19083\_001, which is the same

1. import in the state-level inequality Gini estimates for 2010 and 2015 in the five-year American Community Survey as a single panel dataset;

library(tidyverse)  
#Import the 2015, 2010 gini estimates dataset  
inequality\_panel15 <- get\_acs(geography = "state",  
variables = c("2015" = c("B19083\_001")),  
year = 2015)

## Getting data from the 2011-2015 5-year ACS

inequality\_panel15

## # A tibble: 52 x 5  
## GEOID NAME variable estimate moe  
## <chr> <chr> <chr> <dbl> <dbl>  
## 1 01 Alabama 2015 0.475 0.0023  
## 2 02 Alaska 2015 0.418 0.0062  
## 3 04 Arizona 2015 0.465 0.0016  
## 4 05 Arkansas 2015 0.470 0.0025  
## 5 06 California 2015 0.486 0.0008  
## 6 08 Colorado 2015 0.459 0.0018  
## 7 09 Connecticut 2015 0.493 0.0019  
## 8 10 Delaware 2015 0.446 0.004   
## 9 11 District of Columbia 2015 0.532 0.0042  
## 10 12 Florida 2015 0.484 0.0014  
## # ... with 42 more rows

inequality\_panel10 <- get\_acs(geography = "state",  
variables = c("2010" = c("B19083\_001")),  
year = 2010)

## Getting data from the 2006-2010 5-year ACS

inequality\_panel10

## # A tibble: 52 x 5  
## GEOID NAME variable estimate moe  
## <chr> <chr> <chr> <dbl> <dbl>  
## 1 01 Alabama 2010 0.47 0.003  
## 2 02 Alaska 2010 0.412 0.006  
## 3 04 Arizona 2010 0.453 0.002  
## 4 05 Arkansas 2010 0.459 0.003  
## 5 06 California 2010 0.469 0.001  
## 6 08 Colorado 2010 0.455 0.003  
## 7 09 Connecticut 2010 0.482 0.003  
## 8 10 Delaware 2010 0.436 0.005  
## 9 11 District of Columbia 2010 0.535 0.005  
## 10 12 Florida 2010 0.471 0.002  
## # ... with 42 more rows

inequality\_panel <- bind\_rows(inequality\_panel10, inequality\_panel15)  
inequality\_panel

## # A tibble: 104 x 5  
## GEOID NAME variable estimate moe  
## <chr> <chr> <chr> <dbl> <dbl>  
## 1 01 Alabama 2010 0.47 0.003  
## 2 02 Alaska 2010 0.412 0.006  
## 3 04 Arizona 2010 0.453 0.002  
## 4 05 Arkansas 2010 0.459 0.003  
## 5 06 California 2010 0.469 0.001  
## 6 08 Colorado 2010 0.455 0.003  
## 7 09 Connecticut 2010 0.482 0.003  
## 8 10 Delaware 2010 0.436 0.005  
## 9 11 District of Columbia 2010 0.535 0.005  
## 10 12 Florida 2010 0.471 0.002  
## # ... with 94 more rows

1. rename estimate as gini in your final data frame, which you should call inequality\_panel;

setnames(inequality\_panel, "estimate", "gini")

1. rename NAME to state as well;

setnames(inequality\_panel, "NAME", "state")

1. ensure that inequality\_panel has a year variable so we can distinguish between the 2010 and 2015 gini index data; and

setnames(inequality\_panel, "variable", "year")

1. as a final step, run the head() command so we can get a quick peak at inequality\_panel (Hint: you may need to import each year separately and then append the two data frames together.) [15 points]

head(inequality\_panel)

## # A tibble: 6 x 5  
## GEOID state year gini moe  
## <chr> <chr> <chr> <dbl> <dbl>  
## 1 01 Alabama 2010 0.47 0.003  
## 2 02 Alaska 2010 0.412 0.006  
## 3 04 Arizona 2010 0.453 0.002  
## 4 05 Arkansas 2010 0.459 0.003  
## 5 06 California 2010 0.469 0.001  
## 6 08 Colorado 2010 0.455 0.003

1. Reshape the inequality\_panel wide, such that the gini values for 2010 and 2015 have their own columns. Also, please keep both the state and GEOID variables. Call the resulting data frame inequality\_wide. After you are done with the reshape, run the head() command so we can get a quick peak at the data. [5 points]

inequality\_wide <-   
 inequality\_panel %>%  
pivot\_wider(id\_cols = c("GEOID", "state", "year"), # unique IDs  
names\_from = "year", # names for new wide vars  
values\_from = "gini", # data to put in new wide vars  
names\_prefix = "gini\_" ) # prefix to add before years  
  
head(inequality\_wide)

## # A tibble: 6 x 4  
## GEOID state gini\_2010 gini\_2015  
## <chr> <chr> <dbl> <dbl>  
## 1 01 Alabama 0.47 0.475  
## 2 02 Alaska 0.412 0.418  
## 3 04 Arizona 0.453 0.465  
## 4 05 Arkansas 0.459 0.470  
## 5 06 California 0.469 0.486  
## 6 08 Colorado 0.455 0.459

1. Reshape inequality\_wide to long format. Once you are done, run the head() command so we can get a quick peak at the data. [5 points]

inequality\_long <-  
 inequality\_wide %>%  
 pivot\_longer(cols = starts\_with("gini"), # use columns starting with "year"  
 names\_to ="year", # name of new column  
 names\_prefix = "gini\_", # part of string to drop  
 values\_to = "gini", # where to put numeric values  
 values\_drop\_na = FALSE)  
  
head(inequality\_long)

## # A tibble: 6 x 4  
## GEOID state year gini  
## <chr> <chr> <chr> <dbl>  
## 1 01 Alabama 2010 0.47   
## 2 01 Alabama 2015 0.475  
## 3 02 Alaska 2010 0.412  
## 4 02 Alaska 2015 0.418  
## 5 04 Arizona 2010 0.453  
## 6 04 Arizona 2015 0.465

1. Show with some R code that inequality\_panel and inequality\_long have the same number of observations. [5 points]

nrow(inequality\_long)==nrow(inequality\_panel)

## [1] TRUE

1. Collapse the inequality\_long data frame by state, such that you obtain a single mean gini score for each state for the years 2010 and 2015. When collapsing, also keep both the GEOID and state variables. Call your resulting data frame inequality\_collapsed. [5 points]

inequality\_collapsed <-  
 inequality\_long %>%  
 group\_by(GEOID, state) %>% # tell R the unique IDs  
 summarize(across(where(is.numeric), mean)) # summarize numeric vars by mean

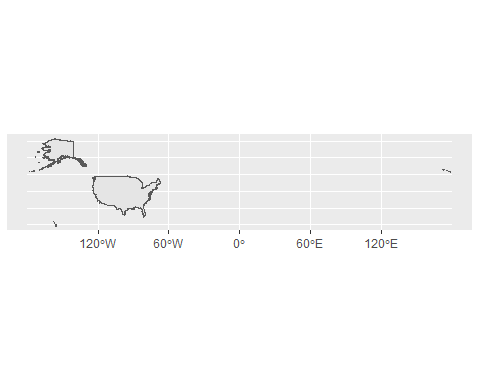
## `summarise()` regrouping output by 'GEOID' (override with `.groups` argument)

1. Produce a map of the United States that colors in the state polygons by their mean gini scores from inequality\_collapsed, using the WGS84 coordinate system. When 2 doing so, use the viridis color scheme. (Note: there are a few different ways to produce the map. We will accept any one of these ways, and the answer key will detail 3 ways. If you want to choose the method with the shape file, you can get the state-level shape file on the Census page). [10 points]

library(rnaturalearthhires)  
us <- ne\_countries(country = 'United States of America', scale = "large", returnclass = "sf")  
  
# us map(basic)  
us\_basic <- ggplot() +  
 geom\_sf(data = us)   
 geom\_sf(data = inequality\_collapsed, aes(color=gini))

## [[1]]  
## mapping: colour = ~gini   
## geom\_sf: na.rm = FALSE  
## stat\_sf: na.rm = FALSE  
## position\_identity   
##   
## [[2]]  
## <ggproto object: Class CoordSf, CoordCartesian, Coord, gg>  
## aspect: function  
## backtransform\_range: function  
## clip: on  
## crs: NULL  
## datum: crs  
## default: TRUE  
## distance: function  
## expand: TRUE  
## fixup\_graticule\_labels: function  
## is\_free: function  
## is\_linear: function  
## label\_axes: list  
## label\_graticule:   
## labels: function  
## limits: list  
## modify\_scales: function  
## ndiscr: 100  
## range: function  
## render\_axis\_h: function  
## render\_axis\_v: function  
## render\_bg: function  
## render\_fg: function  
## setup\_data: function  
## setup\_layout: function  
## setup\_panel\_guides: function  
## setup\_panel\_params: function  
## setup\_params: function  
## train\_panel\_guides: function  
## transform: function  
## super: <ggproto object: Class CoordSf, CoordCartesian, Coord, gg>

print(us\_basic)



1. Use the WDI package to import in data on Gross Domestic Product (GDP) in current US dollars. When doing so, include all countries and only the years 2006 and 2007. Rename your GDP variable to gdp\_current. [5 points]

library(WDI)

1. Deflate gdp\_current to constant 2010 or 2015 US dollars, and call the new variable gdp\_deflated. In words, also tell us the base year that you picked and why. At the end, run a head() command to prove that everything works. [5 points]

gdp\_current = WDI(country = "all", indicator = c("NY.GDP.DEFL.ZS"),  
 start = 2006,   
 end = 2007,   
 extra = FALSE, cache = NULL)

1. In a Shiny app, what are the three main components and their subcomponents? [5 points]

The main components are the UI, Server, and Execution The UI also has input and output components and the server holds the actual code that determines what happens to inputs to produce the outputs.

1. Pull this .pdf file from Mike Denly’s webpage. It is a report on governance in Armenia that Mike Denly and Mike Findley prepared for the US Agency for International Development (USAID). [5 points]

library(pdftools)

## Warning: package 'pdftools' was built under R version 4.0.2

## Using poppler version 0.73.0

mytext=pdf\_text(pdf = "https://pdf.usaid.gov/pdf\_docs/PA00TNMG.pdf")

1. Convert the text pulled from this .pdf file to a data frame, using the , stringsAsFactors=FALSE option. Call the data frame armeniatext. [5 points]

mytext <- as.data.frame(mytext)  
mytext$page=c(1:65)  
colnames(mytext)[which(names(mytext) == "mytext")] <- "text" #change column name  
armeniatext <- mytext

1. Tokenize the data by word and then remove stop words. [5 points]

armeniatext=armeniatext %>%  
unnest\_tokens(word, text)  
data(stop\_words)  
armeniatext <- armeniatext %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

1. Figure out the top 5 most used word in the report. [5 points]

wordfreq <- armeniatext %>%  
count(word, sort = TRUE)  
head(wordfreq, 5)

## word n  
## 1 armenia 252  
## 2 political 207  
## 3 corruption 186  
## 4 governance 185  
## 5 democracy 132

1. Load the Billboard Hot 100 webpage, which we explored in the course modules. Name the list object: hot100exam [5 points]

hot100page <- "https://www.billboard.com/charts/hot-100"  
hot100exam <- read\_html(hot100page)

1. Use rvest to obtain identify all of the nodes in the webpage. [5 points]

body\_nodes <- hot100exam %>%  
html\_node("body") %>%  
html\_children()  
body\_nodes

## {xml\_nodeset (36)}  
## [1] <div class="header-wrapper ">\n<header id="site-header" class="site-head ...  
## [2] <div class="site-header\_\_placeholder"></div>  
## [3] <script>\n var PGM = window.PGM || {};\n PGM.config = PGM. ...  
## [4] <div class="chart-piano-overlay\_\_attachment-point"></div>  
## [5] <main id="main" class="page-content"><div id="charts" data-page-title="T ...  
## [6] <div class="ad\_desktop dfp-ad dfp-ad-promo " data-position="promo" data- ...  
## [7] <div class="ad-container footerboard footerboard--bottom">\n <div cla ...  
## [8] <footer id="site-footer" class="site-footer"><div class="container foote ...  
## [9] <div class="biz-modal">\n <div class="biz-modal\_\_content">\n < ...  
## [10] <script>\n window.CLARITY = window.CLARITY || [];\n</script>  
## [11] <div class="ad\_clarity" data-out-of-page="true" style="display: none;">< ...  
## [12] <script>\n var darkMatterCMD = function() {\n this.darkMatterC ...  
## [13] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/art ...  
## [14] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/clo ...  
## [15] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/rea ...  
## [16] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/rea ...  
## [17] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/rea ...  
## [18] <script src="https://www.billboard.com/assets/1593527595/js/vendors\_/rea ...  
## [19] <script src="https://www.billboard.com/assets/1593527595/js/default\_/art ...  
## [20] <script src="https://www.billboard.com/assets/1593527595/js/default\_/rea ...  
## ...

1. Use Google Chrome developer to identify the necessary tags and pull the data on Rank, Artist, Title, and Last Week. HINT 1: In class we showed you how to get the first three of these. You simply need to add the Last Week ranking. HINT 2: You can navigate two ways. Hovering to find what you need or by doing Cmd+F / Ctrl+F and using actual data to find the location. HINT 3: You’re looking to update the code based on the way the information is in referenced. Try out some different options and see what shows up in the environment. Keep trying until you see that you have a chr [1:100] with values that correspond to what is in the web page. [5 points]

rank <- hot100exam %>%  
rvest::html\_nodes('body') %>%  
xml2::xml\_find\_all("//span[contains(@class,  
'chart-element\_\_rank\_\_number')]") %>%  
rvest::html\_text()  
  
artist <- hot100exam %>%  
rvest::html\_nodes('body') %>%  
xml2::xml\_find\_all("//span[contains(@class,  
'chart-element\_\_information\_\_artist')]") %>%  
rvest::html\_text()  
  
title <- hot100exam %>%  
rvest::html\_nodes('body') %>%  
xml2::xml\_find\_all("//span[contains(@class,  
'chart-element\_\_information\_\_song')]") %>%  
rvest::html\_text()  
  
lastweek <- hot100exam %>%  
rvest::html\_nodes('body') %>%  
xml2::xml\_find\_all("//span[contains(@class,  
'chart-element\_\_meta text--center color--secondary text--last')]") %>%  
rvest::html\_text()  
  
#<div class="chart-element\_\_meta text--center color--secondary text--last">1</div>

Final question. Save all of the files (i.e. .Rmd, .dta, .pdf/Word Doc), push them to your GitHub repo, and provide us with the link to that repo. [no points; 15-point penalty for lack of submission (see above)]