Exam 3 RMarkdown

Cynthia Sun

7/8/2020

## Part One: Tidy Census Gini

#clear environment  
rm(list = ls(all=TRUE))  
  
# load tidycensus   
library(tidycensus)  
census\_api\_key("0979f51018f9aad11498289530215de0e9dd4c4f", install = TRUE,  
 overwrite = TRUE)  
  
gini\_2010 <- load\_variables(year = 2010, "acs5")  
gini\_2015 <- load\_variables(year = 2015, "acs5")  
  
View(gini\_2010)  
View(gini\_2015)  
  
# filter gini inequality  
gini\_2010 <- get\_acs(geography = "county",   
 variables = c(gini = c("B19083\_001")),   
 year = 2010)  
gini\_2015 <- get\_acs(geography = "county",   
 variables = c(gini = c("B19083\_001")),   
 year = 2010)  
# add years  
gini\_2010$year = 2010  
gini\_2015$year = 2015  
  
# combine the datasets into a panel data  
library(data.table)  
library(dplyr)  
inequality\_panel = left\_join(x = gini\_2010,  
 y = gini\_2015,  
 by = "NAME")  
  
# rename NAME variable to `state`  
setnames(inequality\_panel, "NAME", "state")  
  
# rename other variables too, make the irrelevant ones null  
inequality\_panel <- subset(inequality\_panel, select = -c(variable.y))  
inequality\_panel <- subset(inequality\_panel, select = -c(GEOID.y))  
inequality\_panel$year.x <- NULL  
inequality\_panel$year.y <- NULL  
  
# rename variables to specify the year  
setnames(inequality\_panel, "moe.x", "moe 2010")  
setnames(inequality\_panel, "moe.y", "moe 2015")  
setnames(inequality\_panel, "variable.x", "variable")  
setnames(inequality\_panel, "estimate.x", "estimate 2010")  
setnames(inequality\_panel, "estimate.y", "estimate 2015")  
setnames(inequality\_panel, "GEOID.x", "GEOID")  
  
#peek at head of data  
head(inequality\_panel)

Reshape to wide format.

# reshape panel to wide format  
library(tidyverse)  
inequality\_wide <-  
 inequality\_panel %>%  
 pivot\_wider(id\_cols = c("state", "GEOID", "estimate 2010", "estimate 2015"),  
 names\_from = c("state", "GEOID"),  
 values\_from = c("estimate 2010", "estimate 2015"))  
head(inequality\_wide)

Reshape to long format.

# reshape panel to long format  
inequality\_long <- # reshape panel to wide format  
 inequality\_wide %>%  
 pivot\_longer(cols = starts\_with('estimate'),  
 names\_to = c("state", "GEOID"),  
 names\_sep = " ",  
 values\_to = c("estimate 2010", "estimate 2015"),  
 values\_drop\_na = FALSE)  
  
head(inequality\_long)

Map of United States

# state polygons  
#load all the packages  
library(rio)  
library(tidyverse)  
library(googlesheets4)  
library(labelled)  
library(data.table)  
library(varhandle)  
library(ggrepel)  
library(geosphere)  
library(rgeos)  
library(viridis)  
library(mapview)  
library(rnaturalearth)  
library(rnaturalearthdata)  
library(devtools)  
library(rnaturalearthhires)  
library(raster)  
library(sp)  
library(sf) #primary package to map stuff, looks like dataframe unlike raster  
library(ggsflabel)  
library(Imap) #nice mapping/color functions  
  
USmap <- ne\_countries(continent = "North America",   
 scale = "large",   
 returnclass = "sf")  
  
#mean of gini  
mean <- mean(inequality\_panel$`estimate 2015`)  
US\_only = subset(inequality\_panel$continent == "united states of america")  
  
us\_final\_map = ggplot() +  
 geom\_sf(data = us\_map) +  
 geom\_sf(data = inequality\_collapsed, aes(fill = `Log Value`)) +  
 scale\_fill\_viridis(option = "viridis") +  
 ggtitle("US gini inequality 2006 and 2007") +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme\_void()

## Part Two: WDI GDP

library(WDI)  
gdp\_current = WDI(country = "all", indicator = "NY.GDP.MKTP.CD",  
 start = 2006, end = 2007, extra = FALSE, cache = NULL)  
  
#deflate data with USD 2015  
deflator\_data = WDI(country = "all", indicator = "NY.GDP.DEFL.ZS",  
 start = 2006, end = 2007, extra = FALSE, cache = NULL)  
  
# rename deflator variable  
library(data.table)  
setnames(deflator\_data, "NY.GDP.DEFL.ZS", "deflator")  
  
#subset to get data frame with only USD, remove deflator table  
usd\_deflator <- subset(deflator\_data, country == "United States")  
rm(deflator\_data)  
  
#drop unnecessary variables  
usd\_deflator$iso2c <- NULL  
usd\_deflator$country <- NULL  
  
#merge to match years + deflation rates by USD  
gdp\_deflated = left\_join(x = gdp\_current,  
 y = usd\_deflator,  
 by = "year")  
setnames(gdp\_deflated, "NY.GDP.MKTP.CD", "GDP")  
  
# ACTUALLY FUNCTION OF DEFLATING IT  
gdp\_deflated$deflated\_amount = gdp\_deflated$GDP/  
 (gdp\_deflated$deflator/100)  
  
# peek at data  
head(gdp\_deflated)

* I picked constant 2015 USD because it is more accurate.

### Question 10

The three main components are: The User Interface, the Server and the shinyApp(ui, server) functions. The User Interface consists of Input ID and Output ID. The Server also consists of the Input ID and Output ID.

## PART THREE: USAID

## READ PDF FROM WEBSITE  
# first check libraries  
library(pdftools)

## Using poppler version 0.73.0

library(tidyr)   
library(tidytext)   
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(stringr)   
library(ggplot2)  
  
# read pdf from the website  
armeniatext =pdf\_text(pdf =  
 "https://pdf.usaid.gov/pdf\_docs/PA00TNMG.pdf")  
  
# convert to df  
armeniatext <- as.data.frame(armeniatext, stringAsFactors = FALSE)   
armeniatext$page=c(1:65)   
colnames(armeniatext)[which(names(armeniatext) == "armeniatext")] <- "text"  
  
# TOKENIZE NOW  
tidy\_armenia <- armeniatext %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

# Frequency  
armeniafreq <- tidy\_armenia %>%  
 count(word, sort = TRUE)  
head(armeniafreq, n = 5)

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

## Billboard

### Billboard  
library(rvest)  
hot100page <- "https://www.billboard.com/charts/hot-100"  
hot100exam <- read\_html(hot100page)  
  
# check nodes of hot 100  
body\_nodes <- hot100exam %>%  
 html\_node("body") %>%  
 html\_children()  
  
#scrape data for rank, artist, title, and last week  
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()  
  
last\_week <- hot100exam %>%  
 rvest::html\_nodes('body') %>%  
 xml2::xml\_find\_all("//div[contains(@class,  
 'chart-element\_\_meta text')]") %>%  
 rvest::html\_text()

### Github Repo

[Cynthia Sun’s Github Repo](https://github.com/cynthiamsun/DataSciExam3)