Assignment 7

Assignment 7

For this assignment, I want you to use the Census ACS API to download and plot data. Complete the following steps:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidycensus)
```

1. Download data for all of the counties in Georgia on education levels.

```
# Get your own key and save as my_acs_key.txt
#my_acs_key<-readLines("my_acs_key.txt",warn = FALSE)
#acs_key<-my_acs_key

acs_key<-"a0f3f8cc65205f8040f93b4e9168f0f09a4cfdbb"

census_api_key(acs_key,install=FALSE,overwrite =TRUE)
```

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

```
# OR just paste it here.
```

2. Compute the proportion of the population in each county that has a bachelor's degree or above by county.

```
## Educ Characteristics by County for Texas

educ_vars<-get_acs(geography = "county",state="GA",
                   table="B15003",geometry = TRUE)
```

```
## Getting data from the 2015-2019 5-year ACS

## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions

## Loading ACS5 variables for 2019 from table B15003. To cache this dataset for faster access to ACS tab

##   |                                                                              |
```

```
## Spread, so that each level of education gets its own column
educ_vars<-educ_vars%>%
  select(GEOID,NAME,variable,estimate)%>%
  spread(key=variable,value = estimate)

## rename to be all lower case
names(educ_vars)<-str_to_lower(names(educ_vars))

## Calculate prop with at least bachelor's for every county
educ_vars<-educ_vars%>%
  mutate(prop_bach=(b15003_022+
                    b15003_023+
                    b15003_024+
                    b15003_025)/b15003_001)

## simplify to just proportion
educ_vars<-educ_vars%>%
  select(geoid,name,prop_bach)
```

3. Download data for all of counties in Georgia for family income.
4. Compute the proportion of the population in each county that has family income above 75,000.

```
## Income by County for Georgia

income_vars<-get_acs(geography = "county",state="GA",
                     table="B19001",
                     geometry=TRUE)
```

```
## Getting data from the 2015-2019 5-year ACS

## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions

## Loading ACS5 variables for 2019 from table B19001. To cache this dataset for faster access to ACS tab
```

```
## Spread, so that each level of education gets its own column
income_vars<-income_vars%>%
  select(GEOID,NAME,variable,estimate)%>%
  spread(key=variable,value = estimate)

## rename to be all lower case
names(income_vars)<-str_to_lower(names(income_vars))
```

```
## Calculate prop with at least bachelor's for every county
income_vars<-income_vars%>%
  mutate(prop_75p=(b19001_013+
                     b19001_014+
                     b19001_015+
                     b19001_016+
                     b19001_017)/b19001_001)

## simplify to just proportion
income_vars<-income_vars%>%
  select(geoid,name,prop_75p)
```

5. Download data for all of the counties in Georgia on health insurance coverage status.
6. Calculate the proportion of the population in each county that does not have health insurance.

```
#v18 <- load_variables(2018, "acs5", cache = TRUE)
```

```
insurance_vars<-get_acs(geography = "county",state="GA",
                     table="B27010",geometry = TRUE)
```

```
## Getting data from the 2015-2019 5-year ACS

## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions

## Loading ACS5 variables for 2019 from table B27010. To cache this dataset for faster access to ACS tab
```

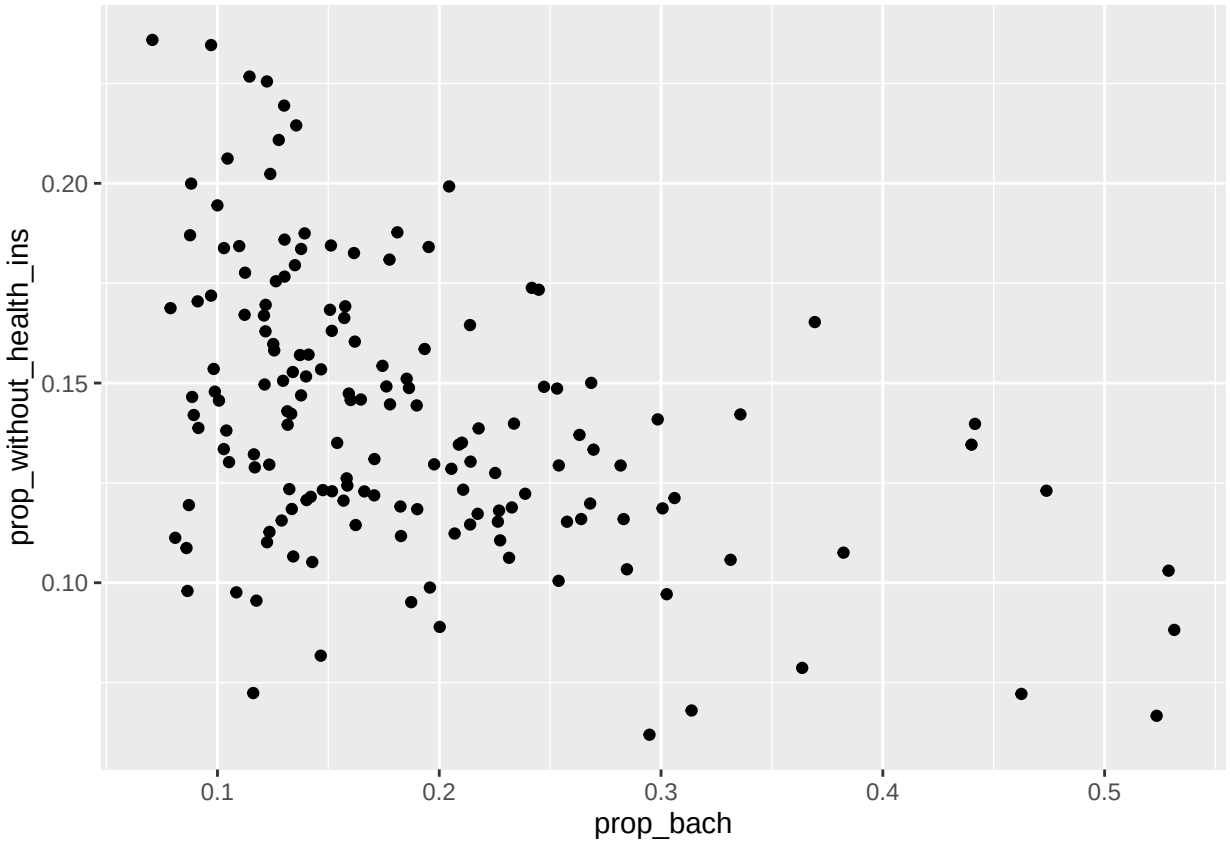
```
## Spread, so that each level of education gets its own column
insurance_vars<-insurance_vars%>%
  select(GEOID,NAME,variable,estimate)%>%
  spread(key=variable,value = estimate)

## Calculate prop with at least bachelor's for every county
insurance_vars<-insurance_vars%>%
  mutate(prop_without_health_ins=((B27010_017+
                                   B27010_033+
                                   B27010_050+
                                   B27010_066)/B27010_001))

## rename to be all lower case
names(insurance_vars)<-str_to_lower(names(insurance_vars))

## simplify to just proportion
insurance_vars<-insurance_vars%>%
  select(geoid,name,prop_without_health_ins)
```

7. Plot the proportion uninsured as a function of education, and then as a function of income.

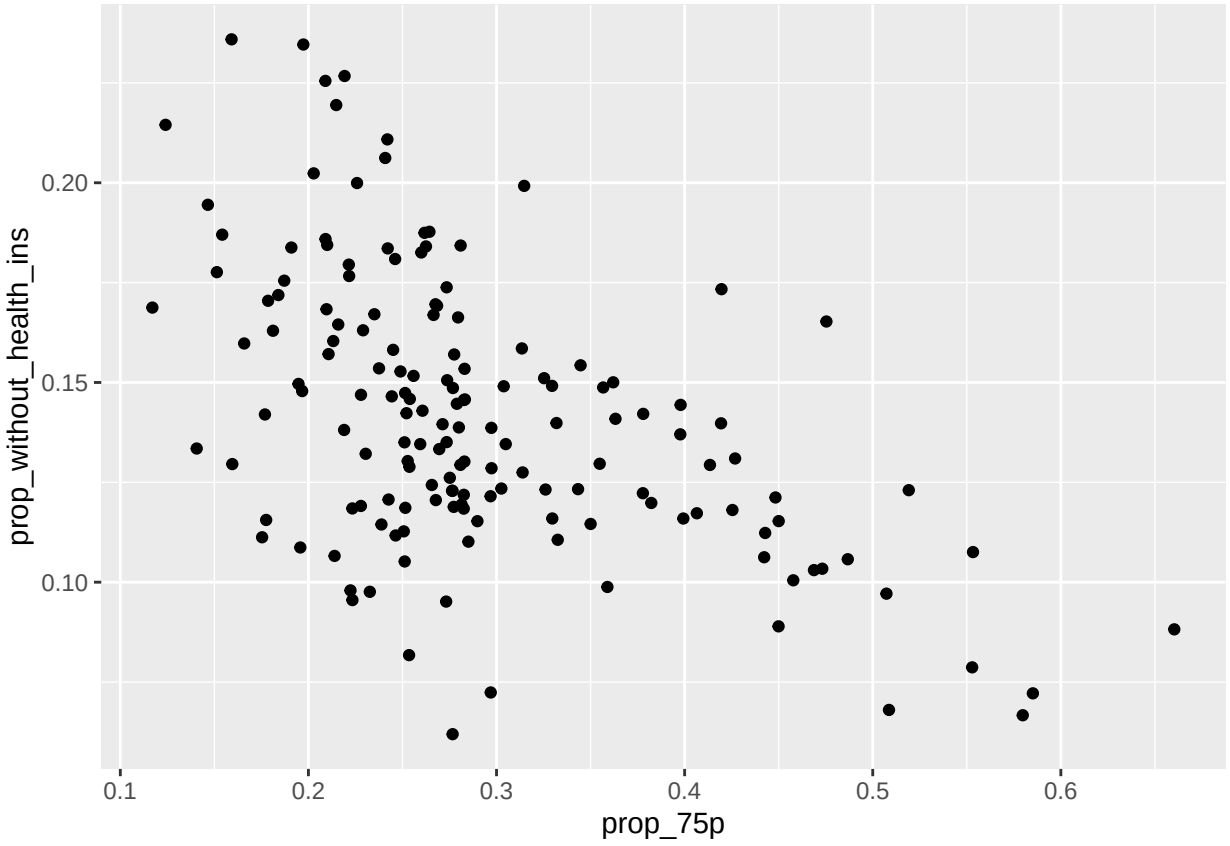
```
educ_vars_2<-educ_vars%>%as_tibble()%>%select(geoid,name,prop_bach)

income_vars_2<-income_vars%>%as_tibble()%>%select(geoid,name,prop_75p)
```

```
insurance_vars_2<-insurance_vars%>%as_tibble()%>%select(geoid,name,prop_without_health_ins)

combined<-left_join(educ_vars_2,income_vars_2,by=c("geoid","name"))

combined<-left_join(combined,insurance_vars_2,by=c("geoid","name"))
```

```
combined%>%
  ggplot(aes(x=prop_bach,y=prop_without_health_ins))+
  geom_point()
```




```
combined%>%
  ggplot(aes(x=prop_75p,y=prop_without_health_ins))+
  geom_point()
```



8. Model the proportion uninsured as a function of education and income.

```
mod1<-lm(prop_without_health_ins~prop_bach+prop_75p,data=combined)
summary(mod1)
```

```
##
## Call:
## lm(formula = prop_without_health_ins ~ prop_bach + prop_75p,
##     data = combined)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.080145 -0.018971 -0.002199  0.016900  0.075749
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.19427    0.00744  26.112  < 2e-16 ***
## prop_bach   -0.01596    0.04313  -0.370    0.712
## prop_75p    -0.17160    0.04008  -4.282 3.23e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0297 on 156 degrees of freedom
## Multiple R-squared:  0.2789, Adjusted R-squared:  0.2697
## F-statistic: 30.17 on 2 and 156 DF,  p-value: 8.383e-12
```