HW7.R

xboxv

2020-03-29

```
library("gdata")
```

```
## gdata: Unable to locate valid perl interpreter
## gdata:
## gdata: read.xls() will be unable to read Excel XLS and XLSX files
## gdata: unless the 'perl=' argument is used to specify the location of a
## gdata: valid perl intrpreter.
## gdata:
## gdata: (To avoid display of this message in the future, please ensure
## gdata: perl is installed and available on the executable search path.)
```

```
## gdata: Unable to load perl libaries needed by read.xls()
## gdata: to support 'XLX' (Excel 97-2004) files.
```

```
##
```

```
## gdata: Unable to load perl libaries needed by read.xls()
## gdata: to support 'XLSX' (Excel 2007+) files.
```

```
##
```

```
## gdata: Run the function 'installXLSXsupport()'
## gdata: to automatically download and install the perl
## gdata: libaries needed to support Excel XLS and XLSX formats.
```

```
##
## Attaching package: 'gdata'
```

```
## The following object is masked from 'package:stats':
##
##     nobs
```

```
## The following object is masked from 'package:utils':
##
##     object.size
```

```
## The following object is masked from 'package:base':
##
##     startsWith
```

```
library("zipcode")
library("maps")
```

```
## Warning: package 'maps' was built under R version 3.6.3
```

```r
library("ggplot2")
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```r
library(readxl)
```

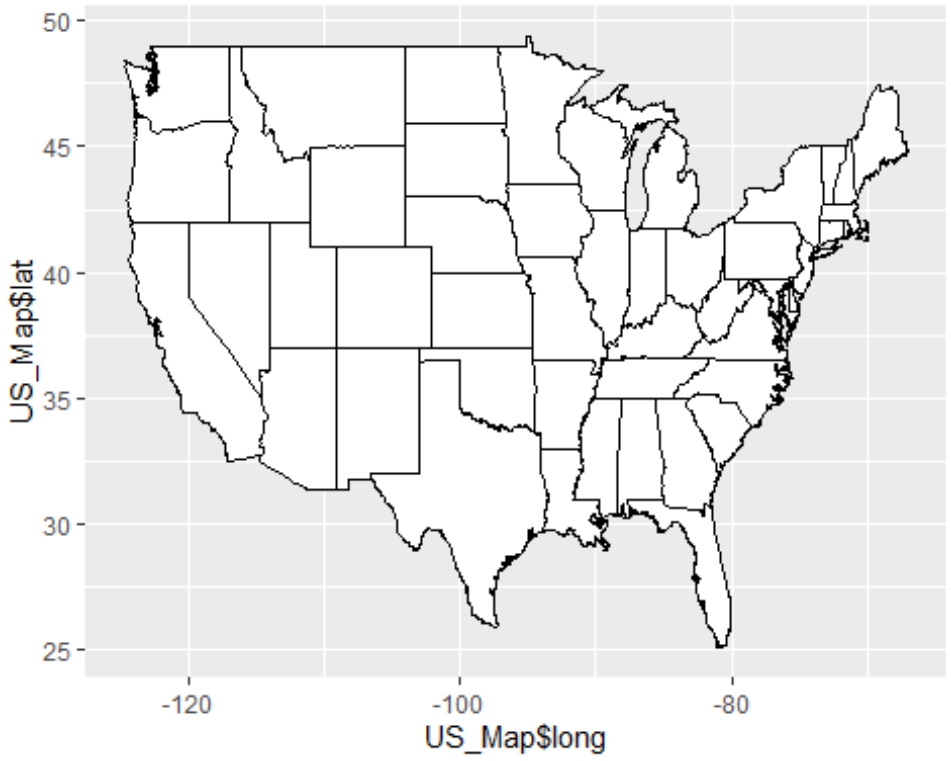
```
## Warning: package 'readxl' was built under R version 3.6.3
```

```r
#step 1
#read the dataa

HW7_Data <- read_excel("C:/Users/xboxv/Downloads/HW7-Data.xlsx")
```

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C7057 / R7057C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C26133 / R26133C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C26134 / R26134C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C26135 / R26135C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C26202 / R26202C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C29646 / R29646C3: got '.'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
sheet, :
## Expecting numeric in C29981 / R29981C3: got '.'
```

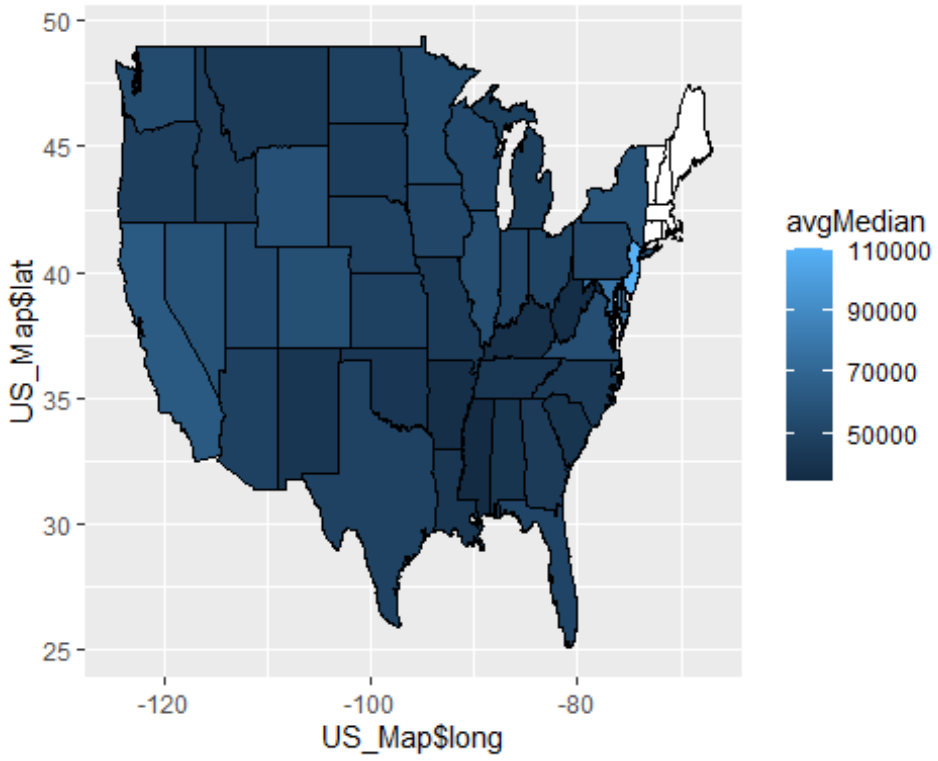
```r
View(HW7_Data)

#renaming the colmnsu
colnames(HW7_Data) <- c("zip","median","mean","population")
View(HW7_Data)

#importing the zipcode package
data("zipcode")

#remvoing Alaska and Hawai from the data
latestZip <- subset(zipcode,zipcode$state != "AK")
```

```r
newZip <- subset(latestZip,latestZip$state != "HI")

#step 2

#combining the two dataframes
mergeData <- merge(x=HW7_Data,y=newZip,by="zip")

#in mergeData, this will sort the state abbreviations
stateAbr <- sort(unique(mergeData$state))

#get the average and the population
avgMedian <- tapply(as.numeric(mergeData$median),mergeData$state,mean)
newPop <- tapply(as.numeric(mergeData$population),mergeData$state,sum)

#a simple data frame with just the average median income and the population
for each state.
simpleData <- data.frame(avgMedian,newPop,stateAbr)

#adding the states name to the SimpleData dataframe
simpleData$states <- state.name[match(simpleData$stateAbr,state.abb)]

#get the map data ofr US
US_Map <- map_data("state")

#creating a simple map
map.simple  <- ggplot()
map.simple  <- map.simple +
geom_map(data=US_Map,aes(x=US_Map$long,y=US_Map$lat,map_id=region),map=US_Map
,fill="white", color="black")

## Warning: Ignoring unknown aesthetics: x, y

map.simple

## Warning: Use of `US_Map$long` is discouraged. Use `long` instead.

## Warning: Use of `US_Map$lat` is discouraged. Use `lat` instead.
```



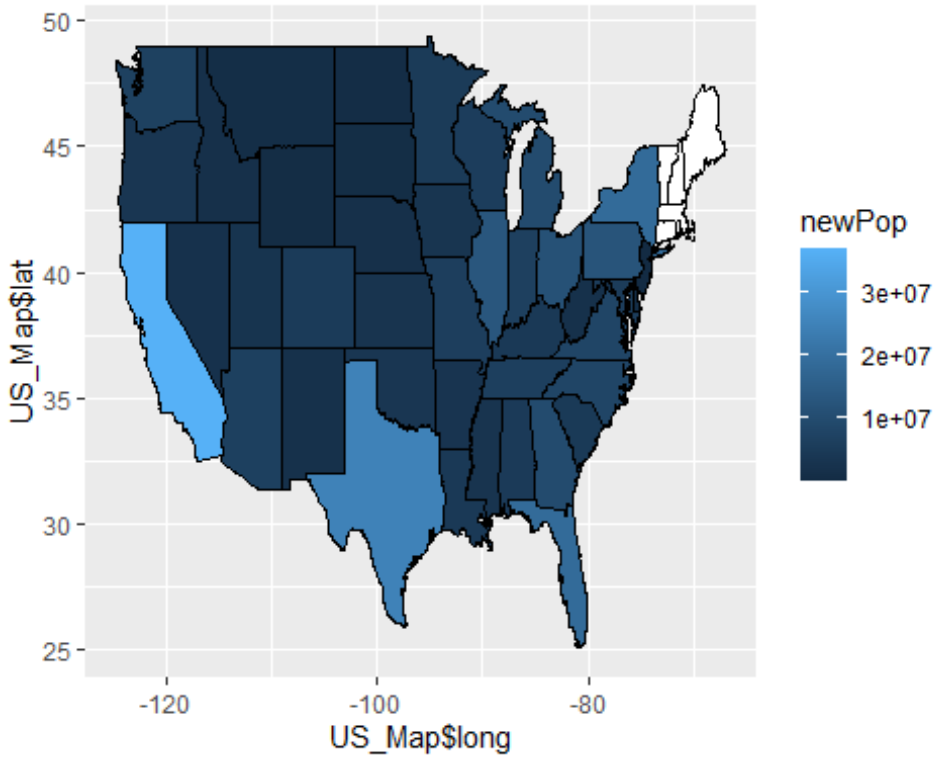

```
#Map representing the color with the average median income of that state
simpleData$states <- tolower(simpleData$states)
Income_map  <- map.simple +
geom_map(data=simpleData,map=US_Map,aes(fill=avgMedian,map_id=states),color="
black",na.rm=TRUE)
Income_map
```

```
## Warning: Use of `US_Map$long` is discouraged. Use `long` instead.

## Warning: Use of `US_Map$lat` is discouraged. Use `lat` instead.
```



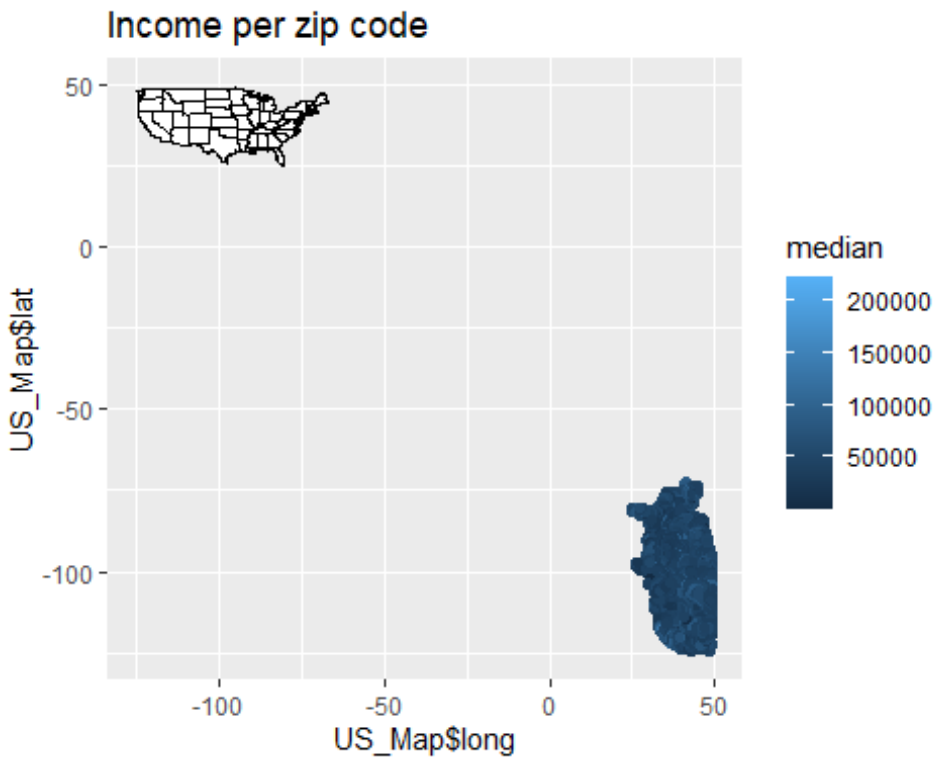

```
#second map with color representing the population of the state.
Popu_map <- map.simple +
geom_map(data=simpleData,map=US_Map,aes(fill=newPop,map_id=states),color="bla
ck",na.rm=TRUE)
Popu_map
```

```
## Warning: Use of `US_Map$long` is discouraged. Use `long` instead.

## Warning: Use of `US_Map$lat` is discouraged. Use `lat` instead.
```



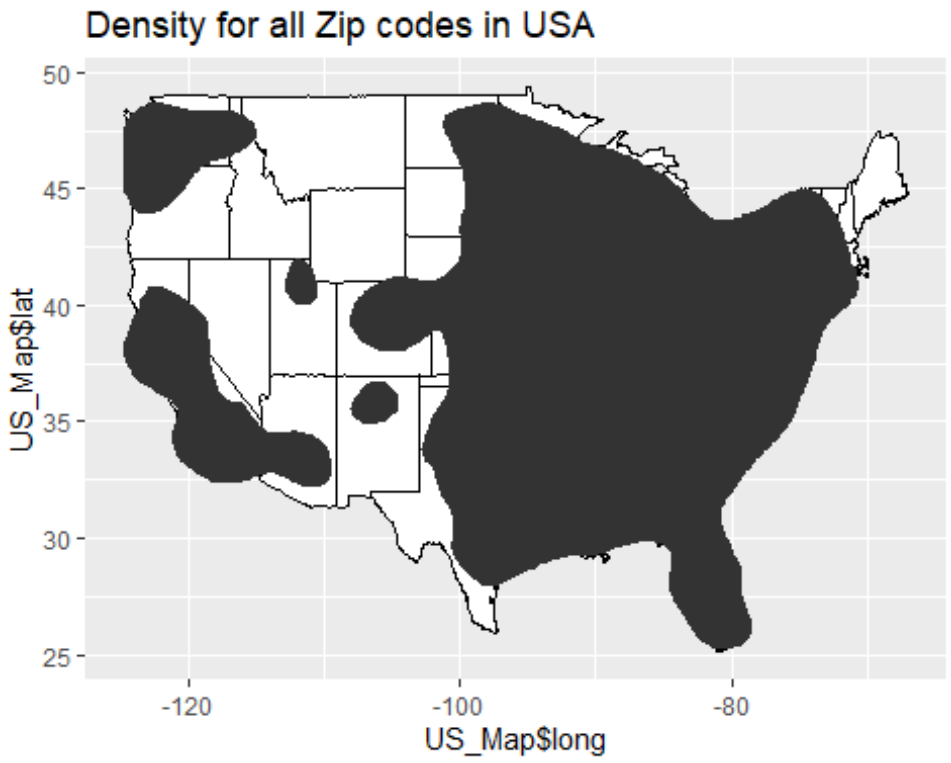

```
#Step 3

#Show the income per zip code
Inc_zip <- map.simple +
geom_point(data=mergeData,aes(x=mergeData$latitude,y=mergeData$longitude,colo
ur=median),na.rm=TRUE)
Inc_zip <- Inc_zip + ggtitle("Income per zip code")
Inc_zip
```

```
## Warning: Use of `US_Map$long` is discouraged. Use `long` instead.

## Warning: Use of `US_Map$lat` is discouraged. Use `lat` instead.

## Warning: Use of `mergeData$latitude` is discouraged. Use `latitude`
instead.

## Warning: Use of `mergeData$longitude` is discouraged. Use `longitude`
instead.
```




```
#step 4
#Using the Stat.density2D fucntion to represent the zip code density
map.density <- map.simple + stat_density2d(aes(x=mergeData$longitude,
y=mergeData$latitude), data=mergeData, geom="polygon") +
  scale_fill_gradient(low="black",high="green")+
  scale_alpha(range=c(0.00,0.25))+
  ggtitle("Density for all Zip codes in USA")

map.density
```

```
## Warning: Use of `US_Map$long` is discouraged. Use `long` instead.

## Warning: Use of `US_Map$lat` is discouraged. Use `lat` instead.

## Warning: Use of `mergeData$longitude` is discouraged. Use `longitude`
instead.

## Warning: Use of `mergeData$latitude` is discouraged. Use `latitude`
instead.
```

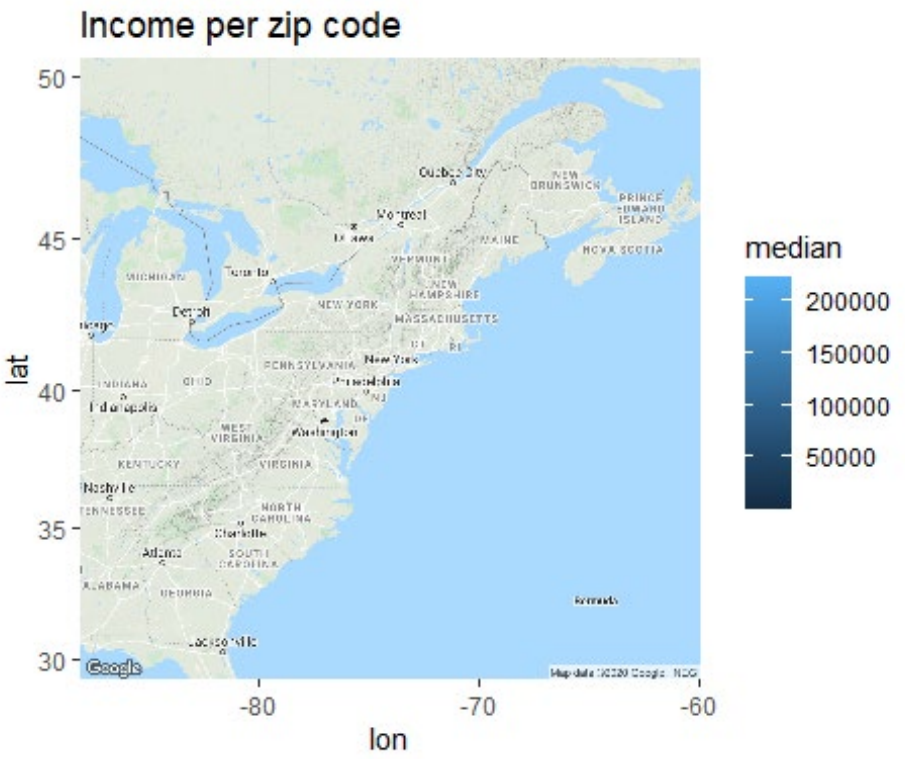



```
#step 5

library(ggmap)

## Warning: package 'ggmap' was built under R version 3.6.3

## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.

## Please cite ggmap if you use it! See citation("ggmap") for details.

# Get api key using google cloud platform, did a lot of rersearch on it to
getthis zoominng step right.
register_google(key = "AIzaSyCniRebA4EVvTRnWlA0iw4KnDdFeuPFUN0")

# Get the center of New York by using get_googlemap
NyMap <- get_googlemap("New York", zoom = 5) %>% ggmap()

## Source :
https://maps.googleapis.com/maps/api/staticmap?center=New%20York&zoom=5&size=
640x640&scale=2&maptype=terrain&key=xxx

## Source :
https://maps.googleapis.com/maps/api/geocode/json?address=New+York&key=xxx

#step 3 again using hte new map
Inc_zip <- NyMap +
geom_point(data=mergeData,aes(x=mergeData$latitude,y=mergeData$longitude,colo
ur=median),na.rm=TRUE)
```

```
Inc_zip <- Inc_zip + ggtitle("Income per zip code")
Inc_zip
```

```
## Warning: Use of `mergeData$latitude` is discouraged. Use `latitude`
instead.
```

```
## Warning: Use of `mergeData$longitude` is discouraged. Use `longitude`
instead.
```

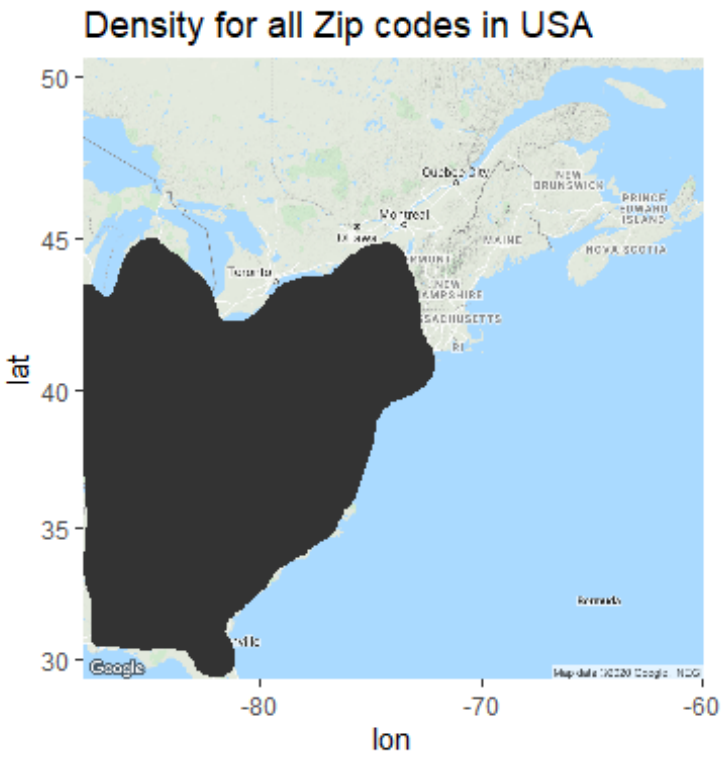



```
#step 4 again
map.density <- NyMap + stat_density2d(aes(x=mergeData$longitude,
y=mergeData$latitude), data=mergeData, geom="polygon") +
  scale_fill_gradient(low="black",high="green")+
  scale_alpha(range=c(0.00,0.25))+
  ggtitle("Density for all Zip codes in USA")

map.density
```

```
## Warning: Use of `mergeData$longitude` is discouraged. Use `longitude`
instead.
```

```
## Warning: Use of `mergeData$latitude` is discouraged. Use `latitude`
instead.
```

```
## Warning: Removed 16988 rows containing non-finite values (stat_density2d).
```


Density for all Zip codes in USA
50
45
40
35
30
lat
-80
-70
-60
lon
Google