Data Visualization on Maps using Leaflet in R

Mee Kyoung Seo

October 27, 2017

Introduction

Hey there, how are you?

My name is Mee Kyoung and I will be guiding you through this post!

This post will be on data visualization, specifically on **maps**. We will be exploring the various visual features of maps in R and then we will be learning how to analyze data through maps.

In this post, I will assume that you as a reader *do not* have experience on drawing maps in R, and so will guide you from the very basics.

Why maps?

1. My first reason is that I always have been fascinated with maps, and wanted to explore how extensively terrains, roads, and regions can be mapped in R.

2. My main reason, however, is that employing maps as the data visualization technique would make data sets *easily readable*.

So which data sets are you examining through Maps?

Among the 16,700 data sets on data.gov, I found two data sets particularly interesting.

- The first data set is on Heart Disease Mortality Rates.
    - This csv file is on the death rate of US adults from heart cardiovascular disease, and limited the age to greater than thirty five.
- The second data set is on Youth Tobacco Rates.
    - This csv file is on how frequent do middle and high school students smoke tobacco. This survey uses representative samples of middle and high school students..

Mission goal of this post?

1. Learn about how to visualize maps in R.
2. Learn about how to analyze data using maps as the main visualization technique.

If there are no more further questions, **let's get started!**

Exploring Maps in R

Installing and Loading Packages

Before we start, let's install and load packages so we can visualize data sets using maps. I will explain what each package is used for as we move along the post.

```
library("ggmap")
```

```
## Loading required package: ggplot2
```

```
library("mapproj")
```

```
## Loading required package: maps
```

```
library("leaflet")
library("dplyr")
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## 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
```

Basic Format of Maps

Let's find basic maps for various locations by applying the code from r-bloggers. For this section, we will use packages **ggmap** and **mapproj**.

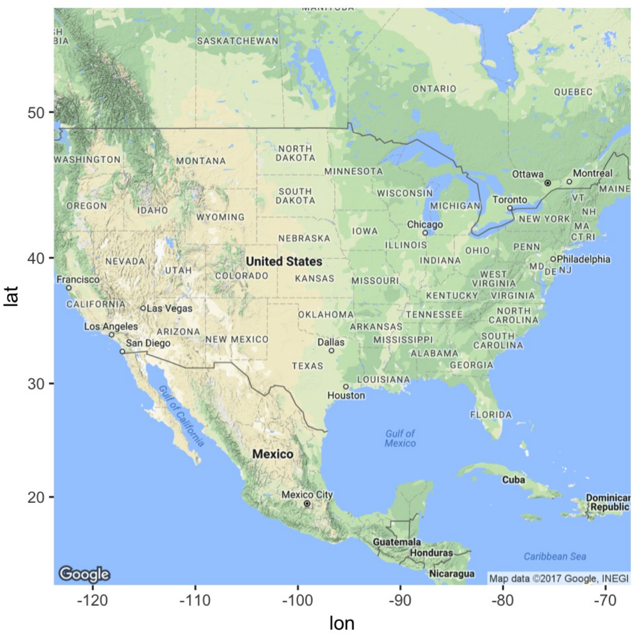
Let's find the map of the United States. Learning from the tutorial link above, I have first assigned **Map_of_United_States** as **get_map()**, and specified the location as United States, and used the zoom feature to zoom into the map. Then, I applied **ggmap** to Map_of_United_States to get the map image.

```
Map_of_United_States <- get_map(location = 'United States', zoom=4)
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=United+States&zoom=4&size=640x640&scale=2&m
aptype=terrain&language=en-EN&sensor=false
```

```
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=United%20States&sensor=false
```

```
ggmap(Map_of_United_States)
```

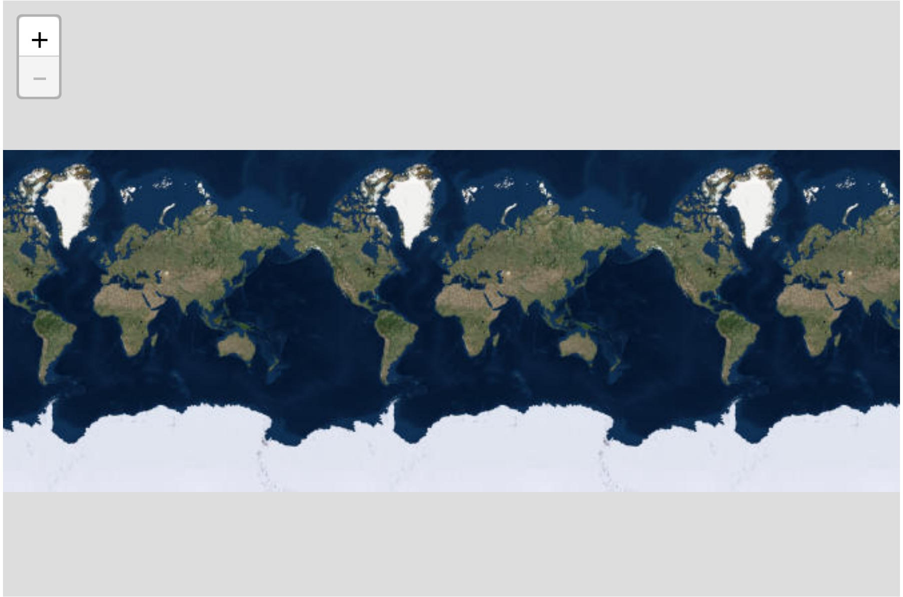


Zooming In and Out

Now that we can find the basic maps of any location by specifying the location, let’s use leaflet to see if we can zoom in and out of the map freely in R. From this section and onwards, we will be using codes from **leaflet**.

For this code below, after learning from the tutorial link above, I first assigned leaflet_map to **leaflet()** and added the tiles (background). **See how far you can zoom in and out!**

```
leaflet_map=leaflet() %>% addTiles() %>% addProviderTiles("Esri.WorldImagery")
leaflet_map
```



Leaflet | © OpenStreetMap contributors, CC-BY-SA, Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

Visualization Features in Leaflet

Doesn’t the previous map seem a little boring? In leaflet, we are able to add in various different background styles for maps. Let’s explore what backgrounds are available.

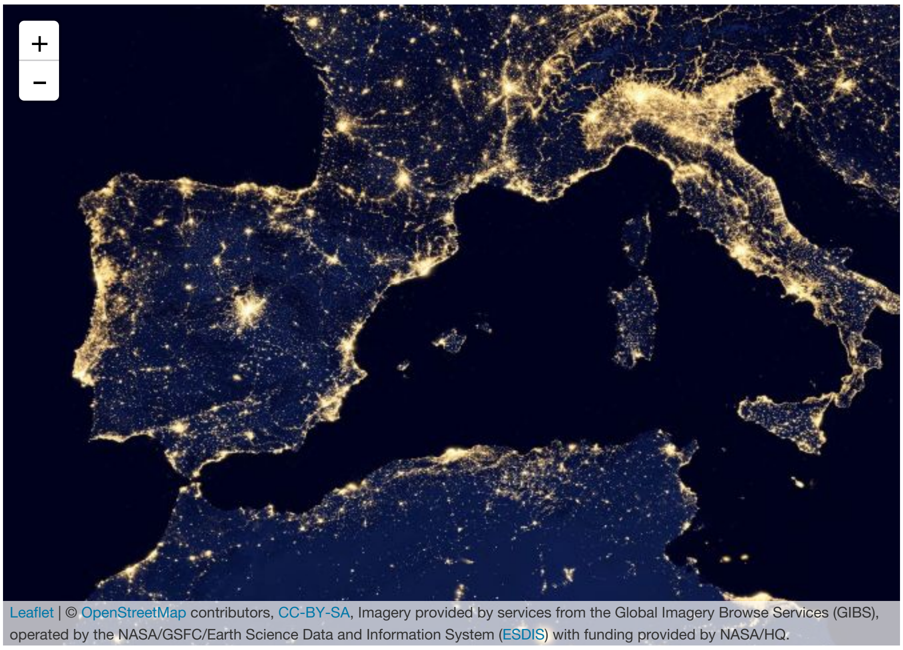
If you wish to check out the available backgrounds for maps further, go to List of tiles.

First Map Tile

This map tile is from List of tiles, using code taken from the tutorial R Gallery, shows the NASA image Earth at Night.

For this code, after learning from the tutorial link above, I first assigned the Night_Image to leaflet and added the basic background, and then setView with parameters lat and lng for latitude and longitude, and zoomed in. Lastly, I added in the **addProviderTiles** with the name of the tile, which can be accessed in List of tiles.

```
Night_Image=leaflet()
Night_Image=addTiles(Night_Image) %>% setView(lat = 40, lng = 3, zoom = 5) %>% addProviderTiles("NASAGIBS.ViirsEarthAtNight2012")
Night_Image
```



Leaflet | © OpenStreetMap contributors, CC-BY-SA, Imagery provided by services from the Global Imagery Browse Services (GIBS), operated by the NASA/GSFC/Earth Science Data and Information System (ESDIS) with funding provided by NASA/HQ.

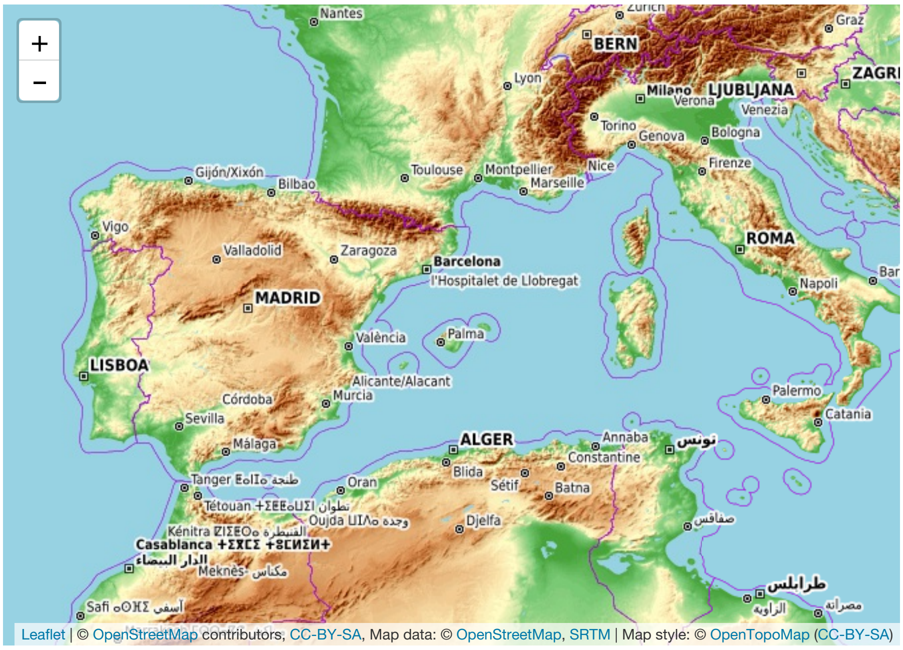
Second Map Tile

This map tile is from List of tiles, using code also taken from the tutorial R Gallery, shows the topography of world map.

For this code, I did the same exact thing as number 1, except I added in the topography tile instead of the night view tile.

```
Topography_Image=leaflet()
Topography_Image=addTiles(Topography_Image) %>% setView(lat = 40, lng = 3, zoom = 5) %>% addProviderTiles("OpenTopoMap")

Topography_Image
```



Leaflet | © OpenStreetMap contributors, CC-BY-SA, Map data: © OpenStreetMap, SRTM | Map style: © OpenTopoMap (CC-BY-SA)

Using Markers in Leaflet

Below, I have the table **States_Lat_Long** that shows each state with its corresponding Latitude and Longitude. I have collected this data from inkplant. I have also made the "location" column into character format, and then showed the top ten rows of this table by doing head(States_Lat_Long, 10).

```
States_Lat_Long <- read.csv("../data/states_lat_log.csv")
States_Lat_Long$Location <- as.character(States_Lat_Long$Location)
head(States_Lat_Long, 10)
```

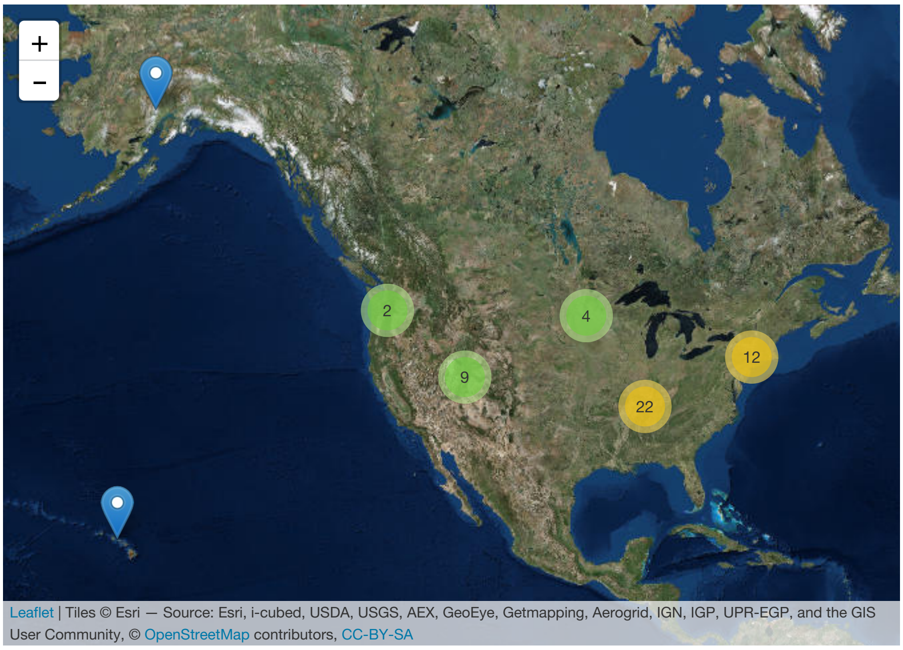
```
##                Location Latitude  Longitude
## 1               Alabama 32.80667  -86.79113
## 2                Alaska 61.37072 -152.40442
## 3               Arizona 33.72976 -111.43122
## 4              Arkansas 34.96970  -92.37312
## 5            California 36.11620 -119.68156
## 6              Colorado 39.05981 -105.31110
## 7           Connecticut 41.59778  -72.75537
## 8              Delaware 39.31852  -75.50714
## 9  District of Columbia 38.89744  -77.02682
## 10              Florida 27.76628  -81.68678
```

Since we have the latitudes and longitudes for each state, we can mark this on the map using **addMarkers** from R Studio. The code will assume that columns "Latitude" and "Longitude" will show Latitude and Longitude values of each location. If you zoom in and out, you can see the markers are functioning as clusters, and this was achieved by using **clusterOptions=markerClusterOptions()**. **Try zooming in and out as much as you can!** This map will show the markers for each state.

```
Marker_Map = leaflet(States_Lat_Long) %>% addProviderTiles("Esri.WorldImagery")
Marker_Map = addTiles(Marker_Map) %>% addMarkers( clusterOptions = markerClusterOptions() )
```

```
## Assuming 'Longitude' and 'Latitude' are longitude and latitude, respectively
```

```
Marker_Map
```



Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community, © OpenStreetMap contributors, CC-BY-SA

Data Cleaning

Now, let's see if we can map our data onto the leaflet package. In order to do that, we must first clean out our data. This is where I will be using the **dplyr** package.

Let's load the first data set, Heart Disease Mortality Rate, as **data1**. On the second line of the code, I have selected two columns, the location and the heart mortality rates by each state. On the third line, I have summed the rates of heart mortality rate by state and then renamed my columns.

```
dat1 <- read.csv("../data/Heart_Disease_Mortality_Data_Among_US_Adults__35___by_State_Territory_and_County.csv")
data1 <- select(dat1[dat1$GeographicLevel=="State", ], c("LocationDesc", "Data_Value") )
data1 <- summarise(group_by(data1[!is.na(data1$Data_Value), ], "Location"=LocationDesc), "Heart_Mortality_Rates" =
sum(Data_Value))
head(data1, 10)
```

```
## # A tibble: 10 x 2
##                Location Heart_Mortality_Rates
##                  <fctr>                 <dbl>
##  1              Alabama                5254.2
##  2               Alaska                4551.4
##  3       American Samoa                1030.6
##  4              Arizona                4548.1
##  5             Arkansas                5357.6
##  6           California                5303.5
##  7             Colorado                4409.6
##  8          Connecticut                3747.6
##  9             Delaware                3890.3
## 10 District of Columbia                4160.1
```

I have done the same for my second data set, Youth Tobacco Survey, as **data2**.

```
dat2 <- read.csv("../data/Youth_Tobacco_Survey__YTS__Data.csv")
data2 <- select(dat2, c("LocationDesc", "Data_Value", "Data_Value_Unit"))
a <- data2[!is.na(data2$Data_Value), ]
data2 <- summarise(group_by(a, "Location" = LocationDesc), "Youth_Tobacco_Rates" = sum(Data_Value))
head(data2, 10)
```

```
## # A tibble: 10 x 2
##                Location Youth_Tobacco_Rates
##                  <fctr>               <dbl>
##  1              Alabama              9935.2
##  2              Arizona              3936.8
##  3             Arkansas              5443.2
##  4           California              2059.4
##  5             Colorado              1136.3
##  6          Connecticut              5837.9
##  7             Delaware              6177.6
##  8 District of Columbia               852.6
##  9              Florida              2786.5
## 10              Georgia              5625.1
```

However, since the two data sets need latitude and longitude values to be able to map on leaflet, I will merge data1 and data2 with States_Lat_Long.

```
data1 <- merge(x=States_Lat_Long, y=data1, by="Location")
head(data1, 10)
```

```
##                Location Latitude  Longitude Heart_Mortality_Rates
## 1               Alabama 32.80667  -86.79113                5254.2
## 2                Alaska 61.37072 -152.40442                4551.4
## 3               Arizona 33.72976 -111.43122                4548.1
## 4              Arkansas 34.96970  -92.37312                5357.6
## 5            California 36.11620 -119.68156                5303.5
## 6              Colorado 39.05981 -105.31110                4409.6
## 7           Connecticut 41.59778  -72.75537                3747.6
## 8              Delaware 39.31852  -75.50714                3890.3
## 9  District of Columbia 38.89744  -77.02682                4160.1
## 10              Florida 27.76628  -81.68678                4437.5
```

```
data2 <- merge(x=States_Lat_Long, y=data2, by="Location")
head(data2, 10)
```

```
##                Location Latitude  Longitude Youth_Tobacco_Rates
## 1               Alabama 32.80667  -86.79113              9935.2
## 2               Arizona 33.72976 -111.43122              3936.8
## 3              Arkansas 34.96970  -92.37312              5443.2
## 4            California 36.11620 -119.68156              2059.4
## 5              Colorado 39.05981 -105.31110              1136.3
## 6           Connecticut 41.59778  -72.75537              5837.9
## 7              Delaware 39.31852  -75.50714              6177.6
## 8  District of Columbia 38.89744  -77.02682               852.6
## 9               Florida 27.76628  -81.68678              2786.5
## 10              Georgia 33.04062  -83.64307              5625.1
```

Map data on Leaflet

Now that we have finished data cleaning, we can move on to our final section, mapping data onto leaflet.

In this section, we will be adding circle markers, using the tutorial from R graph gallery.

The two maps coded below will show:

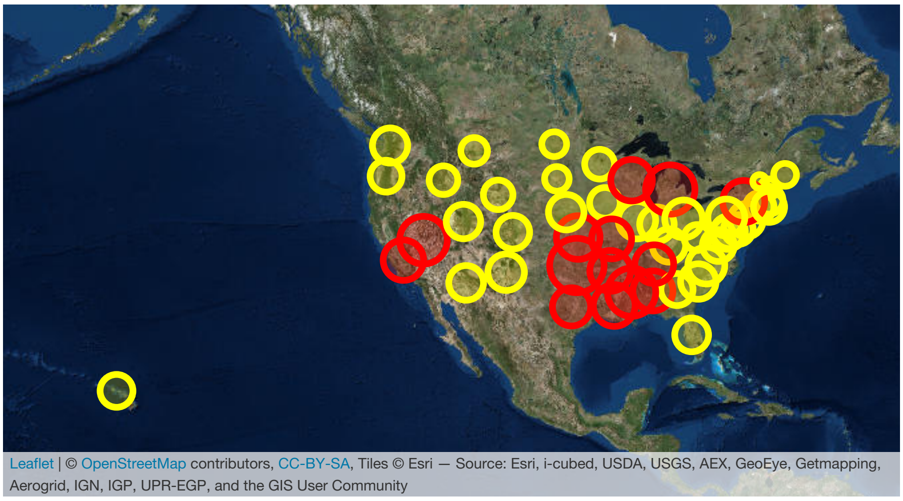
- regions that show high heart mortality rates and high teenage tobacco rates
- different colors of circles that show high vs low rates

Let's set map1 as leaflet(), with data as **data1**, the table on Heart Mortality Rates. Next, we will add the tiles (background), and then use the feature **addCircleMarkers**. Inside, I have the longitude, latitude, and setted the circle radius as ratios of the Heart Mortality Rates. The big circles had the color red, while the small circles had the color yellow.

```
map1=leaflet(data = data1) %>% addTiles() %>%  addProviderTiles("Esri.WorldImagery") %>%

addCircleMarkers(~Longitude, ~Latitude, radius=~Heart_Mortality_Rates/350,
  color=~ifelse(data1$Heart_Mortality_Rates>5000 , "red", "yellow") )

map1
```



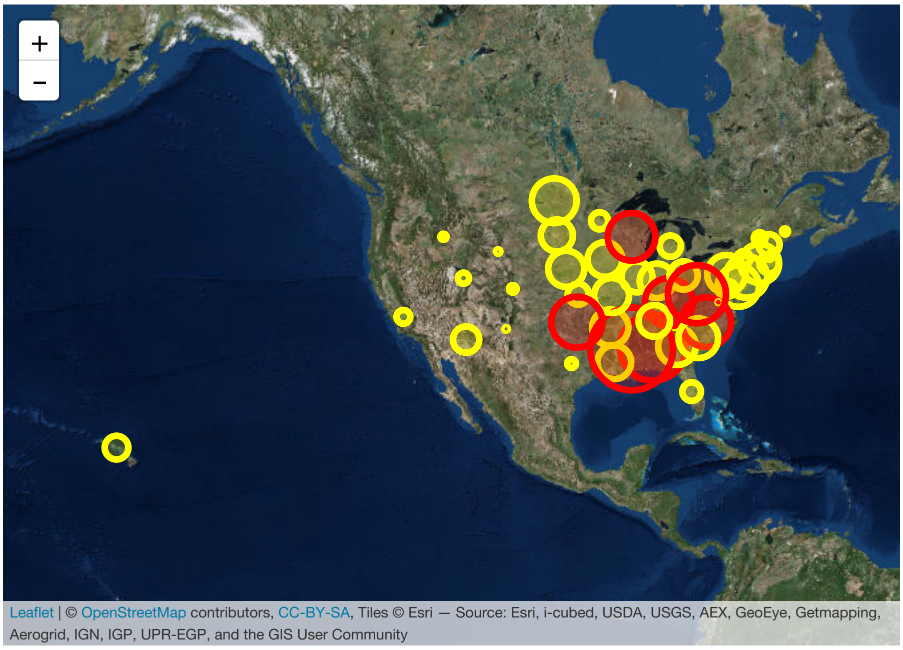
Leaflet | © OpenStreetMap contributors, CC-BY-SA, Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

The map2 for data 2 is the same code as map1, except the data set is now on Youth Tobacco Rates instead of Heart Mortality Rates.

```
map2=leaflet(data = data2) %>% addTiles() %>%  addProviderTiles("Esri.WorldImagery") %>%

  addCircleMarkers(~Longitude, ~Latitude, radius=~Youth_Tobacco_Rates/400,
  color=~ifelse(Youth_Tobacco_Rates>7000 , "red", "yellow") )

map2
```



Leaflet | © OpenStreetMap contributors, CC-BY-SA, Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

Conclusion

From the two map images above, we can conclude that there *seems to be an association* between *Youth Tobacco Rates* and *Heart Mortality Rates*. At the regions teenagers smoke more, there are more people who die from cardiovascular disease. We can see this by checking the red circles overlapping from map 1 and map 2, and how they are located in similar locations. Drawing maps is amazingly fascinating to work with because of the visual effects it has. To further read about visualizing maps in R, go to R Graph Gallery or Leaflet.

Reference

1. data.gov
2. Heart Disease Mortality Rates
3. Youth Tobacco Rates
4. r-bloggers
5. leaflet
6. List of tiles
7. R Gallery
8. R Studio