Post02 - Geocode, Leaflet and Map Visualization

Katherine Zhou

November 29, 2017

Introduction

In this post, I will introduce a function geocode() and a new package leaflet, and combine them with Shiny app to develop a simple app: “Hello World”. Inspired by work we did in last lab and homework, I want to investigate a new method of map visualization to create a more user-friendly and interactive environment.

Motivation

Personally, I love travel. My goal is to travel all around the world. It would be great if there is an app that keeps track of all the places I have traveled and marks them on a world map. As we did in lab, there are several ways to achieve it, such as basic plot() or ggmap(). However, it would be tedious to find the longtitude and latitude of each country manually and the map is not dynamic enough to view. Thus, I want to introduce a new function geocode and a new package leaflet.

First of all, make sure you download this package and install it to your RStudio. Library it to get ready!

```
# install.packages("leaflet")
library(leaflet)
library(ggplot2)
library(ggmap)
# geocode() is a function from ggmap()
```

Data Preparation with Geocode

Instead of downloading data online, I will prepare a small dataset myself including all the countries I have been to and the dates when I visited them.

```
# data cleaning
visit <- c('china',
           'hong kong',
           'taiwan',
           'japan',
           'singapore',
           'malaysia',
           'sewden',
           'norway',
           'denmark',
           'united kingdom',
           'germany',
           'austria',
           'czech republic',
           'australia',
           'new zealand',
           'united states',
           'mexico'
           )
date <- c(1993,
          2011,
          2013,
          2006,
          2013,
          2013,
          2007,
          2007,
          2007,
          2008,
          2015,
          2013,
          2013,
          2012,
          2012,
          2012,
          2016)
# you can try to create your own data set
```

Geocode:

This function takes names of places and returns their longitudes and latitudes. Here, visit_lon and visit_lat are two numeric vectors that contain the corresponding longitudes and latitudes.

```
visit_ll <- geocode(visit)
```

```
## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "united
## kingdom"
```

```
## Warning: geocode failed with status OVER_QUERY_LIMIT, location = "czech
## republic"
```

```
visit_lon <- visit_ll$lon
visit_lat <- visit_ll$lat

dat <- data.frame(
  country=visit,
  lon=visit_lon,
  lat=visit_lat,
  date=date
)
print.data.frame(dat,right=F)
```

```
##    country        lon         lat        date
## 1  china           104.195397  35.861660 1993
## 2  hong kong       114.109497  22.396428 2011
## 3  taiwan          120.960515  23.697810 2013
## 4  japan           138.252924  36.204824 2006
## 5  singapore       103.819836   1.352083 2013
## 6  malaysia        101.975766   4.210484 2013
## 7  sewden           18.643501  60.128161 2007
## 8  norway            8.468946  60.472024 2007
## 9  denmark           9.501785  56.263920 2007
## 10 united kingdom          NA         NA 2008
## 11 germany          10.451526  51.165691 2015
## 12 austria          14.550072  47.516231 2013
## 13 czech republic          NA         NA 2013
## 14 australia       133.775136 -25.274398 2012
## 15 new zealand     174.885971 -40.900557 2012
## 16 united states   -95.712891  37.090240 2012
## 17 mexico         -102.552784  23.634501 2016
```

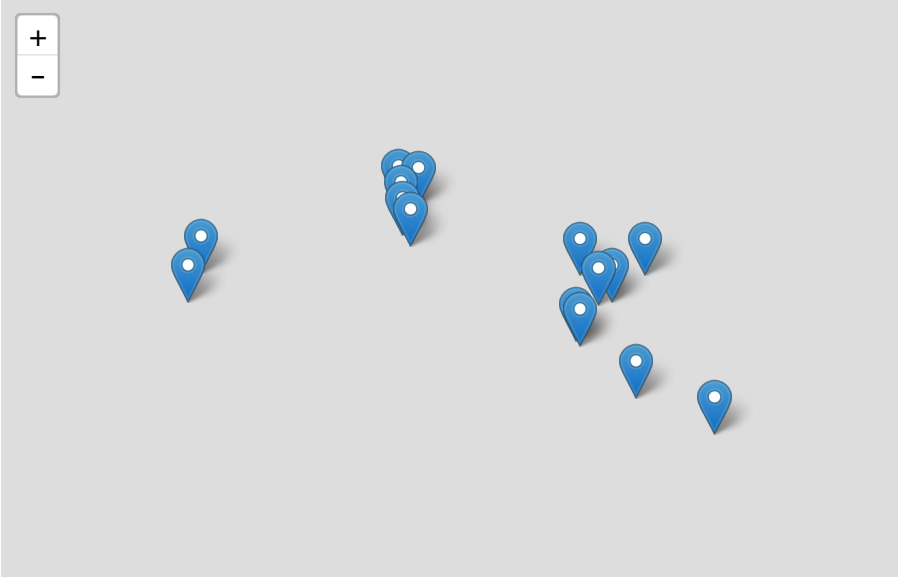
```
# export data to a CSV file for storage and future reference
write.csv(dat,'data/country_kz.csv')
```

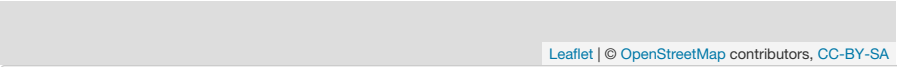
Leaflet

Basic Usage

```
leaflet() %>%
  addTiles() %>%
# the easiest way to add tiles is by default when the OpenStreetMap is used
# you could also use other maps from online resources
  addMarkers(dat$lon, dat$lat,
             popup='',label= dat$country)
```

```
## Warning in validateCoords(lng, lat, funcName): Data contains 2 rows with
## either missing or invalid lat/lon values and will be ignored
```






```
# the first input is longitude and the second is latitude
# label can be displayed on mouse over
# here, the label is country name
```

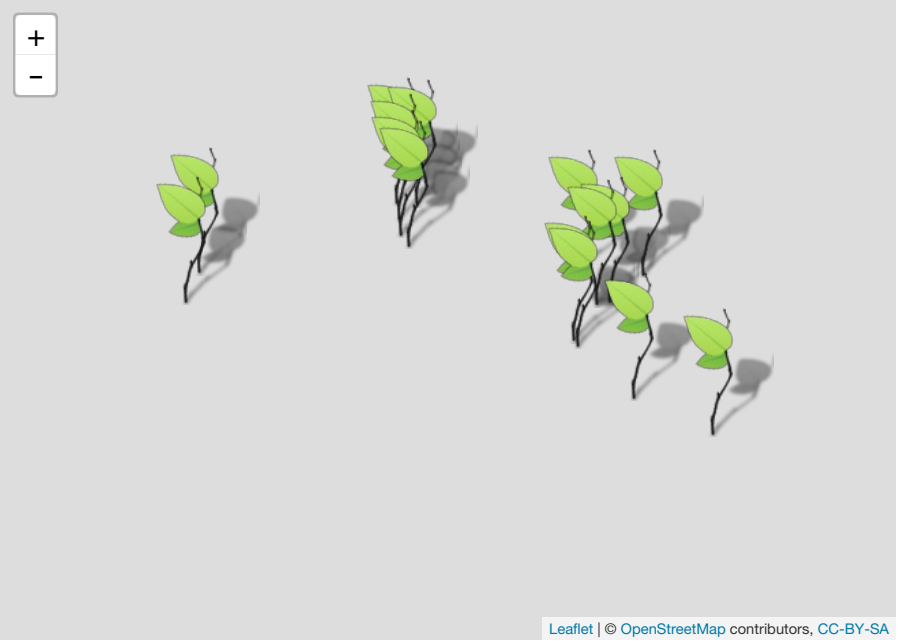
It's great that you could zoom in and out using "+" and "-" without extra coding. It's a built-in feature of leaflet.

Icon Makers

Leaflet is a highly customizable package. Here, I will show you how to change the icon shape. As a default, the icon marker is a blue dropped pin, as the one shown above. We can simply use makeIcon and choose other icons from online resources.

```
# creating an icon using makeIcon function.
greenLeafIcon <- makeIcon(
  iconUrl = "http://leafletjs.com/examples/custom-icons/leaf-green.png",
# shape of icon
  iconWidth = 38, iconHeight = 95,
# these parameters define the dimensions of icon displayed
  iconAnchorX = 22, iconAnchorY = 94,
# point of the icon which corresponds to marker's location
  shadowUrl = "http://leafletjs.com/examples/custom-icons/leaf-shadow.png",
# shape of shadow
  shadowWidth = 50, shadowHeight = 64,
  shadowAnchorX = 4, shadowAnchorY = 62
# these parameters define the dimensions of shadow displayed
)

# using greenLeafIcon as marker
leaflet() %>% addTiles() %>%
  addMarkers(dat$lon, dat$lat, icon = greenLeafIcon)
```

```
## Warning in validateCoords(lng, lat, funcName): Data contains 2 rows with
## either missing or invalid lat/lon values and will be ignored
```



EasyButton and Shiny Integration

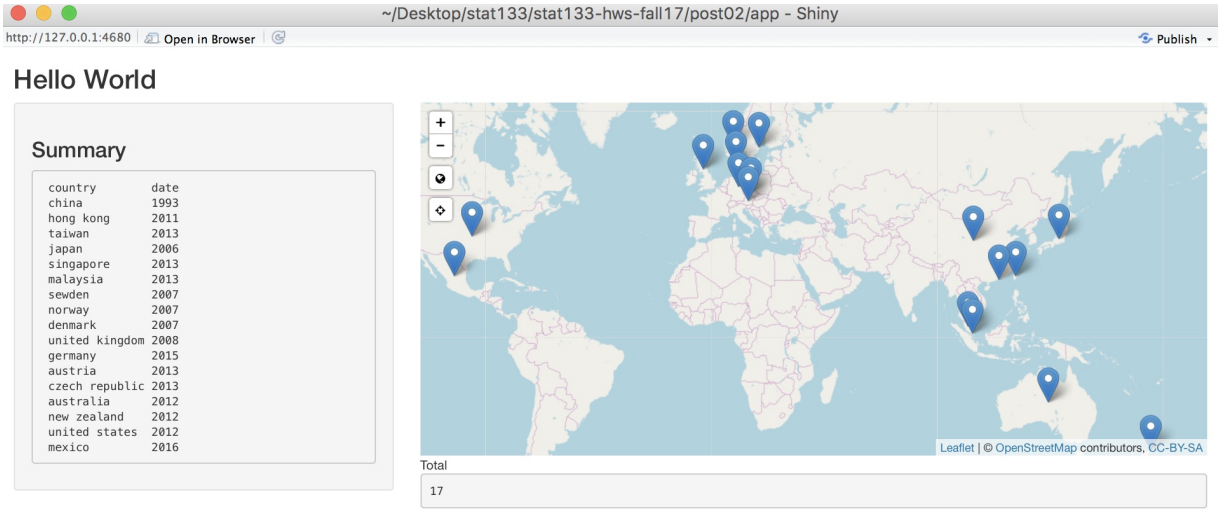
EasyButton() and addEasyButton(), two functions in package leaflet, can be used to add buttons to the map, in addition to the existing buttons, zoom in and out.

Leaflet can be incorporated into Shiny using a leafletOutput() and assigning a renderLeaflet call to the output. It is the same idea as we did in the homework, just different functions.

```
library(shiny)
# shiny app
ui <- fluidPage(
  titlePanel("Hello World"),
  sidebarLayout(

    sidebarPanel(
    h3('Summary'),
    verbatimTextOutput('sum')
    ),

    mainPanel(
    leafletOutput('mymap'),
    print('Total'),
    verbatimTextOutput('total')
    )
)
)

server <- function(input, output) {

  output$mymap <- renderLeaflet({
    leaflet() %>% addTiles() %>%
      addMarkers(dat$lon, dat$lat,
                 popup='',label= dat$country)%>%
# easyButton can create customized buttons and addEasyButton adds them to the map
# the first button returns zoom level to the default and the second button marks your current location
      addEasyButton(easyButton(
        icon="fa-globe", title="Zoom to Level 1",
        onClick=JS("function(btn, map){ map.setZoom(1); }"))) %>%
      addEasyButton(easyButton(
        icon='fa-crosshairs', title='Locate Me',
        onClick=JS("function(btn, map){ map.locate({setView: true});}")))
    })

  output$sum <- renderPrint({
  print.data.frame(sum,row.names = F, right = F)
   })

  output$total <- renderText({
    length(dat$country)
   })

}

shinyApp(ui = ui, server = server)
```

Screenshoot of this Shiny app

```
library(knitr)
include_graphics('images/helloworld.png',auto_pdf = TRUE)
```




```
# this figure gives us a good demonstration on how it works
```

Discussion and Conclusion

As what I have demonstrated above, function geocode() and package leaflet offer users a new way to develop maps and they can even be incorporated into Shiny app for a more dynamic environment. The app, “Hello World”, is just a simple example of how to utilize them to meet user’s need.

I hope that RStudio is not just a subject learnt and being tested in Stat 133. It can be extended to our daily life as an interesting and useful tool.

Reference

Leaflet (https://rstudio.github.io/leaflet/)
Geocode (https://www.rdocumentation.org/packages/ggmap/versions/2.6.1/topics/geocode)
Basemaps (https://rstudio.github.io/leaflet/basemaps.html)
Icon Markers (https://rstudio.github.io/leaflet/markers.html)
Shiny Integration (https://rstudio.github.io/leaflet/shiny.html)
Additional Features (https://rstudio.github.io/leaflet/morefeatures.html)
easyButtonState (https://www.rdocumentation.org/packages/leaflet/versions/1.1.0/topics/easyButtonState)