Data Visualization on Location Values using Plotly and Leaflet in R

Mee Kyoung Seo

November 27, 2017

Introduction

Hey there, how is it going?

This is Mee Kyoung, and I will be guiding you through this post. This post will be on data visualization, especially applying the packages **Plotly** and **Leaflet**.

In this post, I will assume you as a reader *do not* have any experience on working with 3D scatterplots or maps, and so will guide you from the very basics.

Why Plotly and Leaflet?

I chose to use Plotly and Leaflet because:

1. Latitude and longitude values can be well mapped to these packages
2. Plotly and Leaflet provide **interactive** plots

Therefore, for each scatterplot and maps shown in this post, make sure to hover your mouse over the points and zoom in and out to view awesome effects!

What data will you be using?

In this post, I will be analyzing **Rainfall Rates in India**. Though I am not from India, I found this dataset located at Kaggle interesting and chose to analyze this dataset. I will also be using the data that provides longitude and latitude values for each state in India at Latlong.

Mission goal of this post?

1. learn about how to visualize 3D Scatterplots in R
2. learn about how to analyze data using Leaflets in R

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

Data Cleaning

Installing and loading all necessary packages

For this first section, I will be using **dplyr** for data cleaning. I will also be using **plotly** and **leaflet** for data visualization section, which will come up later in the post.

```
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(plotly)
```

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

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

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

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(leaflet)
```

First, I downloaded the data set on India's rainfall rates from Kaggle, and assigned the dataframe as **Rainfall_Raw**. Below shows the first five rows of Rainfall.

```
Rainfall_Raw <- read.csv("India_rainfall.csv")
head(Rainfall_Raw, 5)
```

```
##                 STATE_UT_NAME      DISTRICT   JAN  FEB   MAR   APR   MAY
## 1 ANDAMAN And NICOBAR ISLANDS       NICOBAR 107.3 57.9  65.2 117.0 358.5
## 2 ANDAMAN And NICOBAR ISLANDS SOUTH ANDAMAN  43.7 26.0  18.6  90.5 374.4
## 3 ANDAMAN And NICOBAR ISLANDS N & M ANDAMAN  32.7 15.9   8.6  53.4 343.6
## 4           ARUNACHAL PRADESH          LOHIT  42.2 80.8 176.4 358.5 306.4
## 5           ARUNACHAL PRADESH    EAST SIANG  33.3 79.5 105.9 216.5 323.0
##     JUN   JUL   AUG   SEP   OCT   NOV   DEC ANNUAL Jan.Feb Mar.May Jun.Sep
## 1 295.5 285.0 271.9 354.8 326.0 315.2 250.9 2805.2   165.2   540.7  1207.2
## 2 457.2 421.3 423.1 455.6 301.2 275.8 128.3 3015.7    69.7   483.5  1757.2
## 3 503.3 465.4 460.9 454.8 276.1 198.6 100.0 2913.3    48.6   405.6  1884.4
## 4 447.0 660.1 427.8 313.6 167.1  34.1  29.8 3043.8   123.0   841.3  1848.5
## 5 738.3 990.9 711.2 568.0 206.9  29.5  31.7 4034.7   112.8   645.4  3008.4
##   Oct.Dec
## 1   892.1
## 2   705.3
## 3   574.7
## 4   231.0
## 5   268.1
```

Then, I selected two columns, state names and the corresponding annual rain rates, and called the new dataframe as **Rainfall**. After that, I grouped the annual rain rates by state, and summed up the annual rain fall rates by state. I called this new data frame as **Rainfall_Group**.

Below shows the first five rows of Rainfall_Group.

```
Rainfall <- Rainfall_Raw[, c("STATE_UT_NAME", "ANNUAL")]
Rainfall$STATE_UT_NAME <- as.character(Rainfall$STATE_UT_NAME)

Rainfall_Group <- summarise(
  group_by(Rainfall, STATE_UT_NAME),
  "ANNUAL" = sum(ANNUAL)
)
Rainfall_Group<- select(Rainfall_Group, c(State = STATE_UT_NAME, ANNUAL))
head(Rainfall_Group, 5)
```

```
## # A tibble: 5 x 2
##                          State  ANNUAL
##                          <chr>   <dbl>
## 1 ANDAMAN And NICOBAR ISLANDS  8734.2
## 2               ANDHRA PRADESH 21736.7
## 3            ARUNACHAL PRADESH 46838.0
## 4                        ASSAM 66267.7
## 5                        BIHAR 45621.1
```

Next, I assigned the **Lat_Long** dataframe as the data I downloaded from latlong.net that provides Latitude and Longitude values for each state.

```
Lat_Long <- read.csv("India_Lat_Long.csv")
Lat_Long<- select(Lat_Long, c(State = PLACENAME, Latitude, Longitude))
Lat_Long$State <- as.character(Lat_Long$State)
head(Lat_Long, 5)
```

```
##            State Latitude Longitude
## 1      TELANGANA 17.12318  79.20882
## 2 MADHYA PRADESH 23.47332  77.94800
## 3        HARYANA 29.23848  76.43188
## 4   CHHATTISGARH 21.29513  81.82823
## 5        HARYANA 29.06577  76.04050
```

After that, I merged the Rainfall_Group with the Lat_Long data frame. For this merged data, we will be looking at 16 states in India.

```
mergeddata <- merge(Lat_Long, Rainfall_Group, by="State")
mergeddata
```

```
##             State Latitude Longitude  ANNUAL
## 1           ASSAM 26.24416  92.53784 66267.7
## 2         GUJARAT 22.30943  72.13623 24032.9
## 3         HARYANA 29.23848  76.43188 12905.7
## 4         HARYANA 29.06577  76.04050 12905.7
## 5       KARNATAKA 15.31728  75.71389 35838.5
## 6       KARNATAKA 15.31728  75.71389 35838.5
## 7          KERALA 10.85052  76.27108 41123.5
## 8  MADHYA PRADESH 23.47332  77.94800 51615.5
## 9  MADHYA PRADESH 25.79403  78.11653 51615.5
## 10    MAHARASHTRA 19.60119  75.55298 44750.6
## 11    MAHARASHTRA 19.66328  75.30029 44750.6
## 12      RAJASTHAN 27.39128  73.43262 19192.7
## 13     TAMIL NADU 11.12712  78.65689 30720.2
## 14        TRIPURA 23.74513  91.74683  9916.5
## 15  UTTAR PRADESH 28.20761  79.82666 67836.6
## 16    WEST BENGAL 22.97862  87.74780 34398.1
```

Analyzing data through plotly

After cleaning the data, now I will plot the data points using plotly. I have referenced codes from plotly examples.

Hover your mouse over each point in the scatterplot to see the specifics! You can zoom in and out, and also drag around the scatterplot!

I first set plot_ly to the mergeddata, and set x-axis, y-axis, and z-axis as Latitude, Longitude, and Annual Rainfall rates respectively.

I then added the points using **add_markers()**, and then added in the labels using **layout**. The x, y, z axis are called as Latitude, Longitude, and Annual Rainfall.

```
basic_plot <- plot_ly(mergeddata, x = ~Latitude, y = ~Longitude, z = ~ANNUAL) %>%
  add_markers() %>%
  layout(scene = list(xaxis = list(title = 'Latitude'),
                      yaxis = list(title = 'Longitude'),
                      zaxis = list(title = 'Annual Rainfall')))
basic_plot
```

Adding Visuals to the 3D Scatterplots

We can also show this plot by state, by using **color = ~State**. This will show 16 unique points with 16 unique colors, each point showing the annual rainfall rate for the state.

```
colored_plot <- plot_ly(mergeddata, x = ~Latitude, y = ~Longitude, z = ~ANNUAL, color = ~State) %>%
  add_markers() %>%
  layout(scene = list(xaxis = list(title = 'Latitude'),
                      yaxis = list(title = 'Longitude'),
                      zaxis = list(title = 'Annual Rainfall')))
colored_plot
```

Mapping Data by Leaflet

Now since we have the 3D scatterplots, let's map our data on to the map using Leaflet.

Leaflet is a package that displays the data frame to an actual map. I have referenced the code from R Graph Gallery.

I assigned Map_of_India as the map, setting data as the mergeddata. I then added the markers by setting x and y values as Latitude and Longitude values, and setted the popup as the name of each state. I also added the addTiles() for the background of the map.

Below, you can see the states marked from our dataframe.

```
Map_of_India =leaflet(data = mergeddata) %>% addTiles() %>% addProviderTiles("Esri.WorldImagery") %>% addMarkers(~
Longitude, ~Latitude, popup = ~as.character(State))
Map_of_India
```

On the map above, we can see that the states marked do have rainfalls. However, we cannot see how much rainfall each state receives just from the previous map.

Therefore, we will map the data so that the rainfall rates can be visualized easily on the map. I have referenced the codes below through R Graph Gallery.

After adding the addTiles, which is the background, I setted the markers as circles by using the code addCircleMarkers.

Thus, the rainfall rates are proportional to the circle sizes shown below.

If the rainfall rate is greater than 40k, I colored the circle blue, and if it is less than 40k, I colored the circle yellow.

Here are the results.

```
colored_map=leaflet(data = mergeddata) %>% addTiles() %>%  addProviderTiles("Esri.WorldImagery") %>%

  addCircleMarkers(~Longitude, ~Latitude, radius=~ANNUAL/3500,
  color=~ifelse(ANNUAL>40000 , "red", "blue") )

colored_map
```

Conclusion and Take Home Message

To summarize, for this the post, we explored many different ways to analyze data sets that have location values such as latitude and longitude values. We visualized two specific methods, using 3D scatterplot and leaflet. 3D Scatterplot was helpful to see the correlation between the three variables, and see how each states were correlated with one another. The leaflet mapping was also helpful to see the different rainfall rates for each state, and we could easily visualize the numbers.

Through this post, I hope you were able to learn about how to visualize 3D Scatterplots in R using this data set, and further learn about how to analyze data using Leaflets in R, mapping various data sets.

Thank you for reading my blog post!

Reference

1. Kaggle for India's Rainfall Rates
2. Latlong.net for Latitude and longitude values of India
3. Plotly examples on Scatterplots
4. Leaflet Circle Mapping on R-Graph Gallery
5. Leaflet Markers on R-Graph Gallery
6. Various color charts and Parameters for Plotting
7. STHDA for basic formatting of 3D Scatterplot
8. 133 Github Source for Working with dplyr Packages