Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

Context

I have managed a website named NovoCiv (https://forums.novociv.org) with discussion forums since August 2010. Over the past 8 years the forums have amassed over 880,000 posts. Each post was either created by one of the 1,100 users, or by a server-side script. The posts made by server-side scripts will be referred to as "Bot" posts. Each post made by a user has an IP address attached to it. Posts made by Bots have a null value in the IP address field.

Objective

An IP address can be used to roughly estimate a geological location. It is easy to take any single IP address and put it into a free website such as https://whatismyipaddress.com/ip-lookup to find out an estimate of the geographical location. These tools will show latitude and longitude, a city name, a web service provider and other interesting information.

However, what if I would like to know the locations of 880,000 IP addresses? Even if I could search one IP per second, it would take around 250 hours to get the locations of all the IP addresses. The objective of my project is to use the tools provided by Base R and R packages in order to geolocate all 880,000 IP addresses, and then use this information to visualize the frequency of posting locations on a world map.

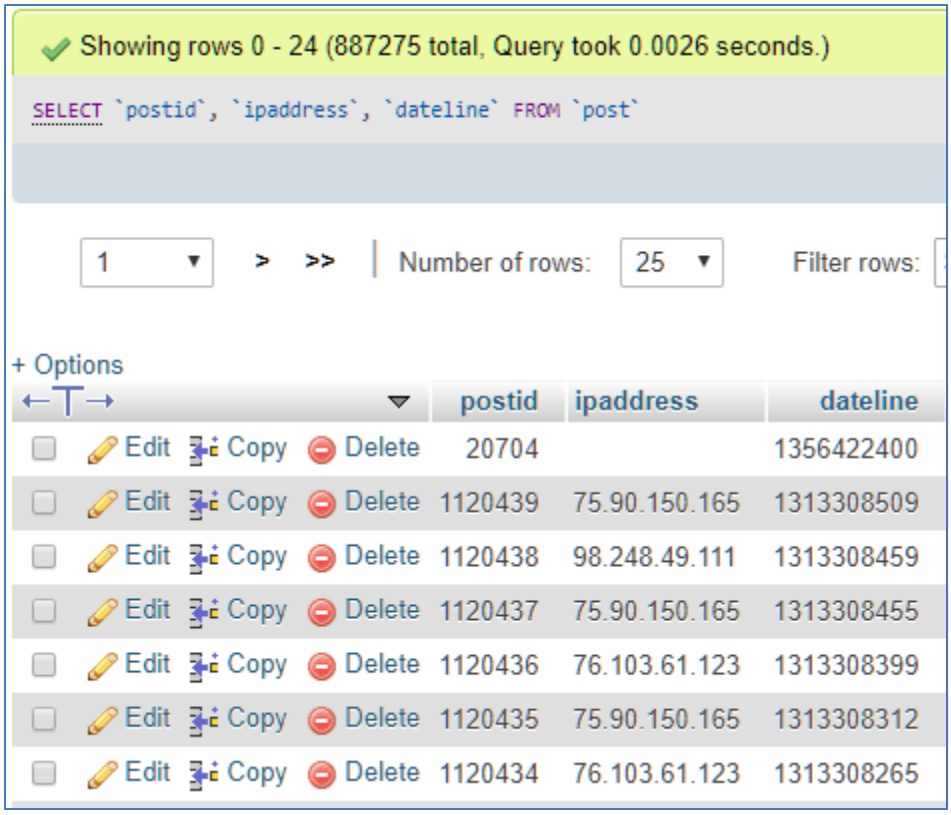
Data Management

I have to prepare my data before I can visualize it. In order to easily share my dataset, so that readers of this project can replicate my charts, I must first collect and organize the data. My starting point is a `post` table in a MYSQL database.

To begin, I have to export the relevant columns of my `post` table in the website database to a .csv file.

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project



Showing rows 0 - 24 (887275 total, Query took 0.0026 seconds.)

```
SELECT `postid`, `ipaddress`, `dateline` FROM `post`
```

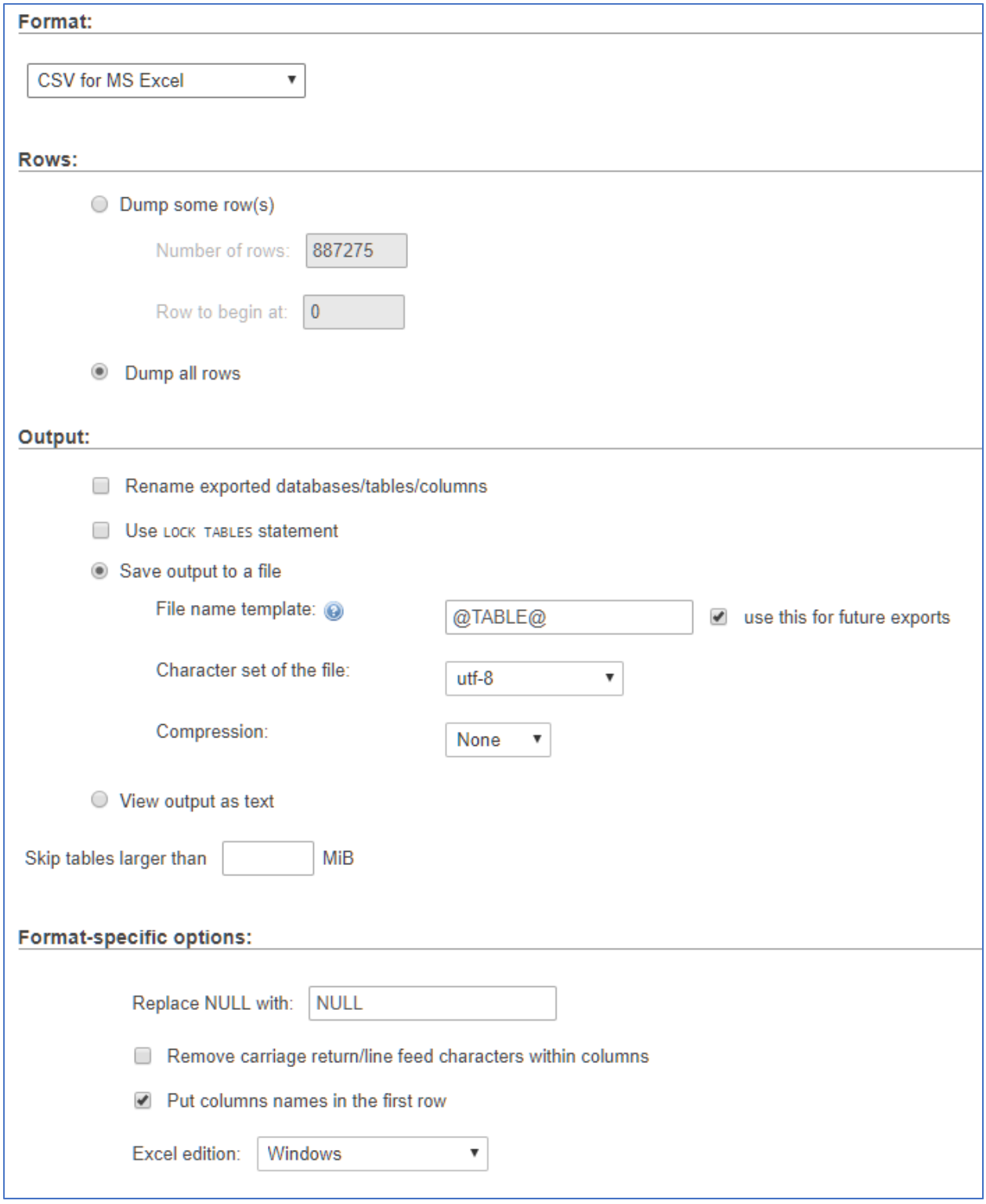
1 ▾ > >> | Number of rows: 25 ▾ Filter rows:

+ Options

	postid	ipaddress	dateline
Edit Copy Delete	20704		1356422400
Edit Copy Delete	1120439	75.90.150.165	1313308509
Edit Copy Delete	1120438	98.248.49.111	1313308459
Edit Copy Delete	1120437	75.90.150.165	1313308455
Edit Copy Delete	1120436	76.103.61.123	1313308399
Edit Copy Delete	1120435	75.90.150.165	1313308312
Edit Copy Delete	1120434	76.103.61.123	1313308265

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project



The resulting .csv file is in a semicolon delimited format. I have to use some excel functions to format the file into a proper .xlsx format, before importing to R.

```
> library(readxl)
> post <- read_excel("C:/Users/jared_000/Desktop/SGH/1st Semester/Data Visualization/post.xlsx")
```

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

Now that my data is in R as a dataframe, I can use the R package `rgeolocate` to gather information from the free maxmind database (https://dev.maxmind.com/geoip/geoip2/geolite2/).

```
> #Take the IP column
> ips <- unlist(post[,3], use.names=FALSE)
> #Geolocate them
> system.time(
+ rgeolocate::maxmind(
+ ips, "~/R/GeoLite2-City.mmdb", c("city_name","longitude",
"latitude")
+ ) -> xdf
+ )
   user  system elapsed
  13.64    0.16   14.26
```

The resulting dataframe `xdf` has 3 columns: city_name, latitude and longitude. My next step is to add on the columns IP address and dateline and do some sorting. One of the posts has a faulty value in dateline, so I have to change it to a value within the scope of my project.

```
> test <- xdf
> test[,4] <- post$dateline
> test[,5] <- post$ipaddress
> colnames(test)[colnames(test)=="V4"] <- "dateline"
> colnames(test)[colnames(test)=="V5"] <- "ipaddress"
> test2 <- test[order(test$dateline),]
> test2[1,4] <- 1281496979
```

Next, I want to convert my dateline column into an R friendly date format. My website database stores the dateline values as unix timestamps. Using the library `anytime`, I can quickly convert them to proper dates.

```
> library(anytime)
> system.time(
+ test2[,6] <- anytime(test2[,4])
+ )
   user  system elapsed
      0       0.02    0.02
> colnames(test2)[colnames(test2)=="V6"] <- "date"
> setwd("C:/Users/jared_000/Desktop/SGH/1st Semester/Data
Visualization/Map-1")
> saveRDS(test2, file="posts2.rds")
```

This file `posts2.rds` is the first dataset which is provided with the project report. Next, I will prepare a second dataset that geolocates based on Country, rather than City.

```
> system.time(
+   rgeolocate::maxmind(
+     ips, "~/R/GeoLite2-City.mmdb", c("country_name")
+   ) -> countries
+ )
   user  system elapsed
   5.45    0.11    5.72
```

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

The resulting dataset `countries` has a single column of country names. Next, I will use the functions in the `tidyverse` library in order to consolidate this data.

```
> countries %>%
+   count("country_name") -> cts
> View(cts)
> cts[55,1] <- "Bots"
> saveRDS(cts, file="countries.rds")
```

The file `countries.rds` is the second dataset that is provided with this project.

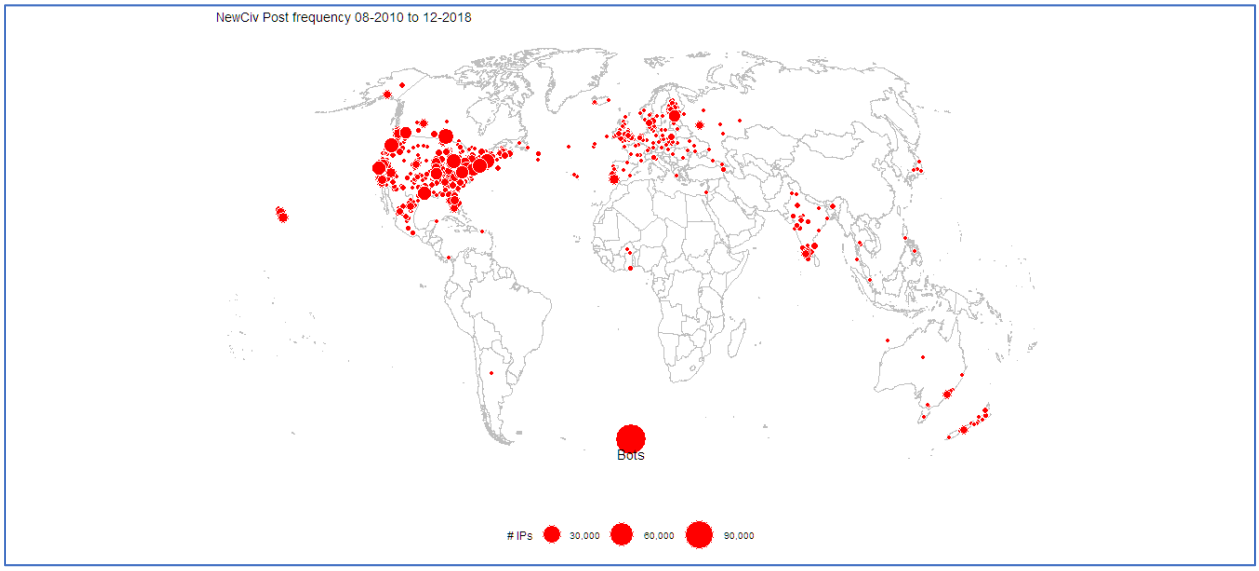
Data Visualization

At this point, the file `project.R` can be used. From here on, I will reference the line numbers, rather than posting the specific code.

After loading the required libraries and setting my working directory, I load the posts2.rds file. The file is located in the Map-1 folder because it is used with the Shiny app, which is explained later.

Using the `plyr` library, on lines 14-17 I count the frequency of each city and take the mean of the latitude and longitude of each city. The data is ordered smallest to largest, so that the largest circles will be plotted last on the map.

Lines 19-20 store the world map, and then lines 22-37 plot the post frequency on the world map with ggplot2. The map is shown below, but higher resolution versions can be viewed at https://novociv.org/map.png and https://novociv.org/maphd.png.

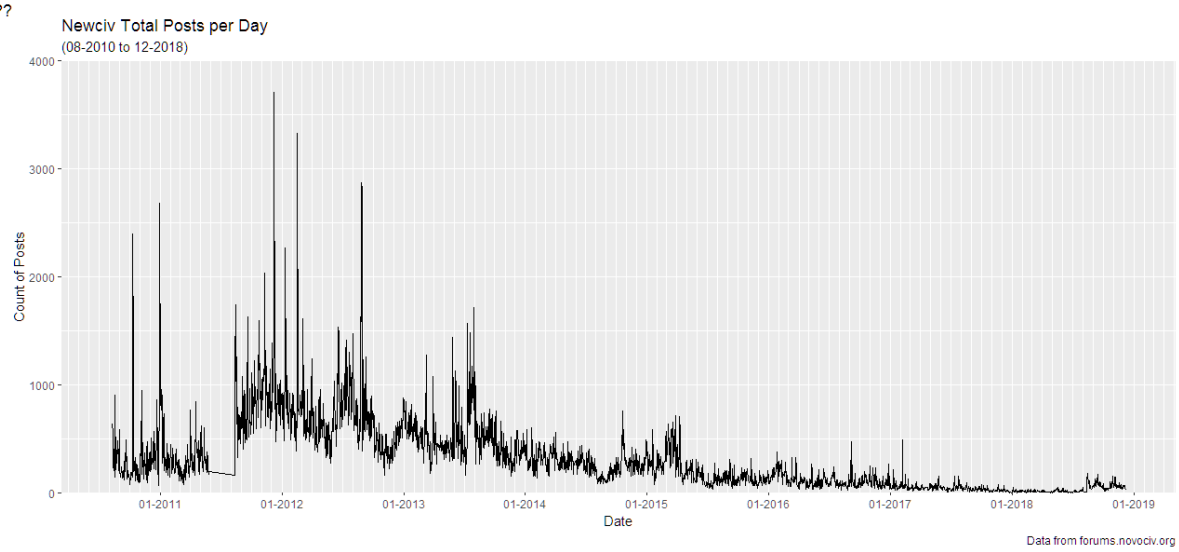


Next, lines 39-42 count the number of posts per day, and lines 44-54 are used to create a line graph in ggplot2 showing the overall post frequency over time.

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

Figure ??

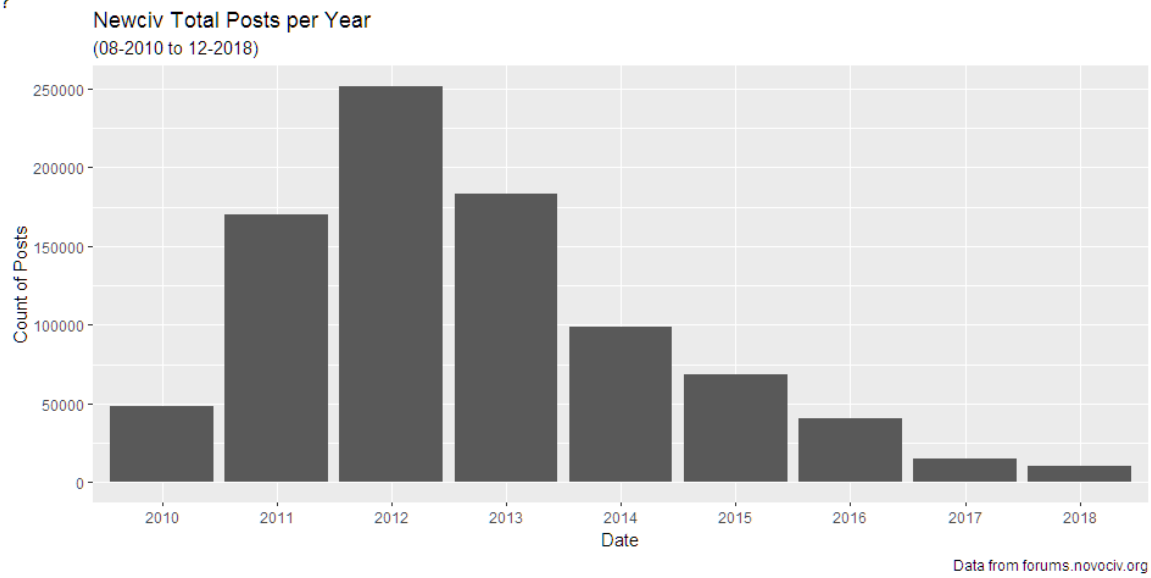


Unfortunately, this graph shows that the activity level of my website has fallen to almost nothing. In my experience, forums as a means of community discussion have fallen in popularity over the past few years. Many existing members now prefer to keep in touch using WhatsApp or Discord, and younger people no longer go to forums. The result is a declining userbase, which inevitably leads to declining activity.

An interesting thing to note is the straight line showing in the middle of 2011. During this time, the website was hacked by a member of a rival community. The hacker deleted the entire database, and my newest backup was from several months prior. The result was that several months of posts were lost.

Lines 56-57 further consolidate the data into years, and lines 59-66 create a bar chart showing the activity level over the past 8 years.

Figure ??

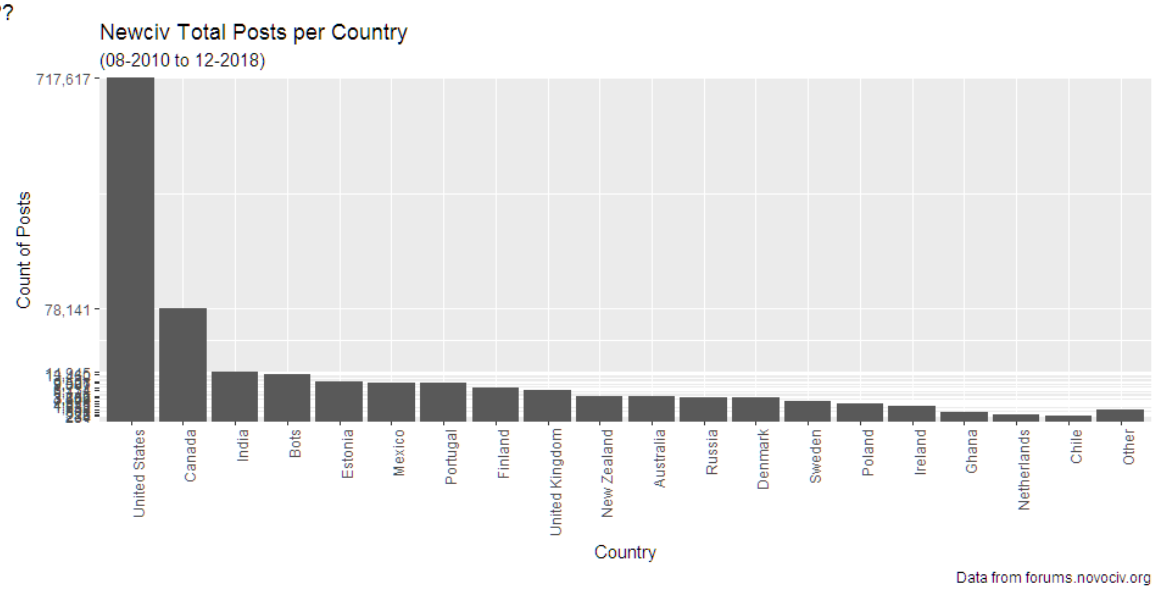


Geolocating and Mapping Posts from 08-2010 to 12-2018

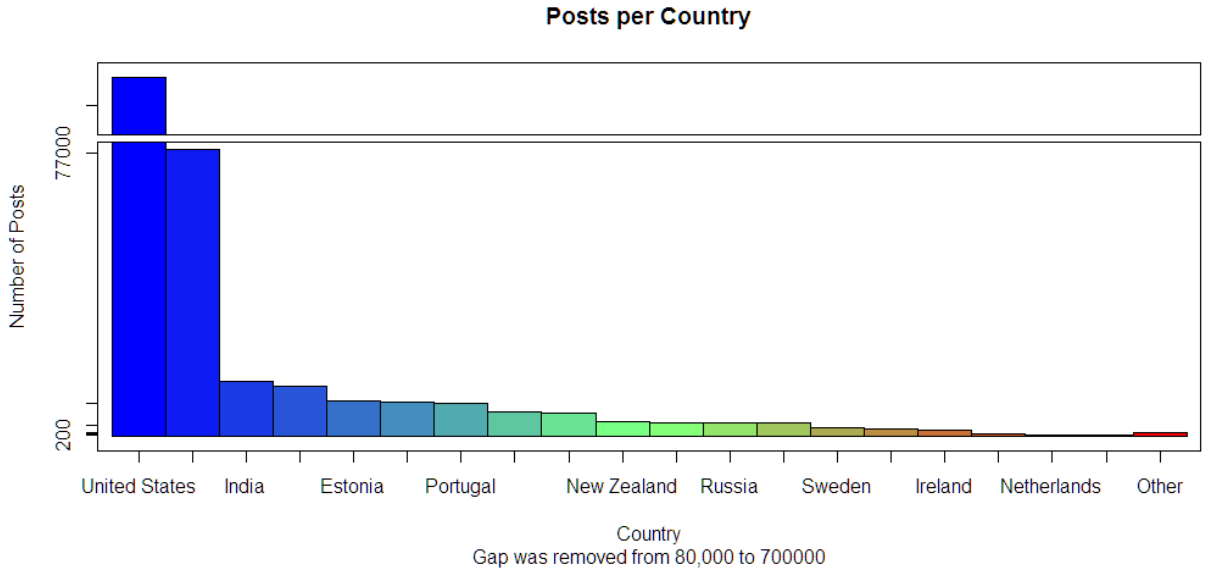
Data Visualization in R – Jared Crivello 83472 Final Project

Lines 68-72 read the `countries.rds` file and create a list of the top 19 countries by post frequency, plus all other countries as 'Other'. Lines 74-83 Attempt to create a barchart to present this data.

Figure ??



Due to the extreme difference between the United States and the rest of the world, the chart is effectively useless. I attempted to graph with a log2 scale and a log10 scale, but ultimately the sqrt scale was the best option. Even so, the chart is largely unreadable.



In lines 85-91, I attempted to use the `plotrix` library to make a barchart with gap. Unfortunately, plotrix has less formatting options than ggplot2, and even with the gap the chart is largely unreadable.

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

	country_name	freq
54	United States	717617
9	Canada	78141
24	India	14945
55	Bots	13460
14	Estonia	9526
32	Mexico	9307
39	Portugal	9007
15	Finland	6734
53	United Kingdom	6174
34	New Zealand	3860
3	Australia	3706
44	Russia	3553
13	Denmark	3444
49	Sweden	2371
38	Poland	1946
25	Ireland	1532
19	Ghana	523
33	Netherlands	286
10	Chile	234
27	Other	909

Ultimately, it turned out that the best way to present the data was using a simple table.

Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

Interactive chart with shiny:

I wanted to take my world map and split it into months using shiny. The shiny app is contained in the Map-1 folder. In addition to preparing the app which can be run locally, I have hosted the app at https://novociv.shinyapps.io/map-1/. This website will let anyone interact with my shiny app from their browser, without the need to run any code in R.

My entire shiny app is contained in the file app.R in the Map-1 folder. I begin by including the necessary libraries. Next, I have a function which takes a slider date input and formats it to look nicer. The next step is to load the posts2.rds dataset. Line 12 is the reason why the posts2.rds file was saved to the Map-1 folder.

After importing the dataset, I user a tidyverse function to add a column of year-month. Then I split the dataframe into a list of dataframes, grouped by the month. Finally, with lines 18-19 I load the world map.

With the initial data prepared, I then create the ui. Line 23 plots the map which is generated in the server section, but it includes `click="map_hover"``. This will enable me to click on the map in order to see details about the specific data points. Lines 25-30 are used to make the date slider look nicer.

Next, I create the server. Lines 34-44 just deal with the slider, with the end result of `shiny()` outputting whatever month the slider is set to. Then lines 46-54 grab the dataframe associated with the month output by `shiny()`, summarize the data and prepare it for plotting, and outputs the result when `dat()` is called.

Lines 56-67 generate and output the city names and post frequencies for any points nearby when a user clicks on the map. Due to the fact that the earth (a globe) is plotted on a 2D map, this functionality is not always accurate. Therefore, I added a generously high threshold.

Lines 69-90 then store the ggplot with the variable values of `shiny()` and `dat()`. It is important to note that line 80 is distinctly different than in the ggplot for the "total posts" world map.

```
scale_size(name = "# IPs", label=scales::comma, range = c(2,25),
limits = c(1,5000))
```

By setting a range of 2-25, I tell the map the minimum and maximum possible size of points. Then with limits of 1-5000, I tell the map to consider a frequency of 1 as size 2, and a frequency of 5000 as size 25. This is important, because it creates a standard that is held regardless of the overall activity level for a month. This is shown in the screenshots of the shiny app below.

With the shiny app programmed, running it is simple.

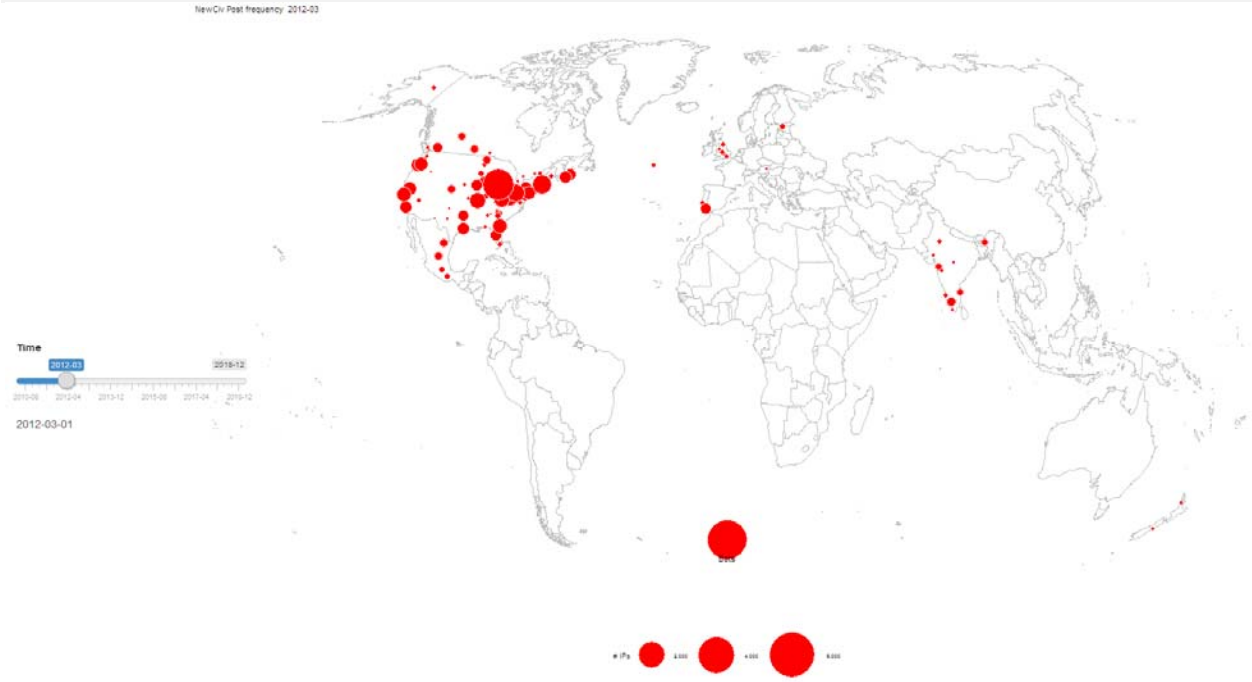
```
> setwd("C:/Users/jared_000/Desktop/SGH/1st Semester/Data
Visualization")/Data Visualization/")
> library(shiny)
> runApp("Map-1")
```

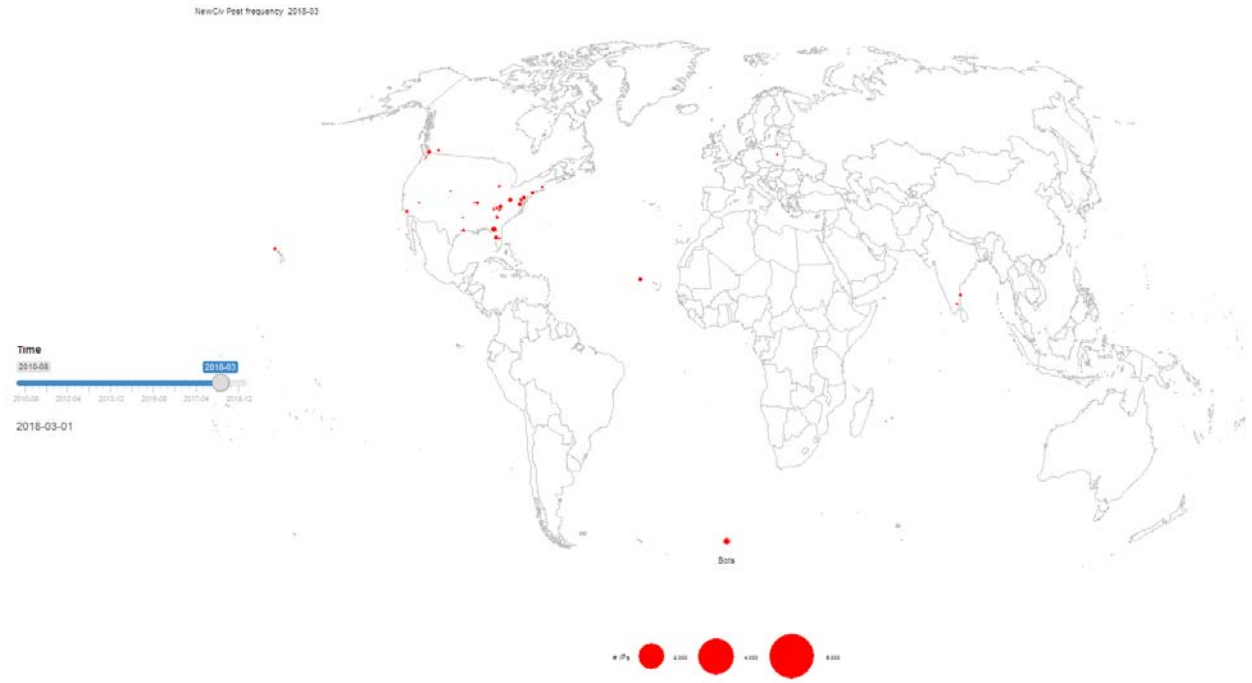
Geolocating and Mapping Posts from 08-2010 to 12-2018

Data Visualization in R – Jared Crivello 83472 Final Project

To host my shiny app online, I made a free account with shinyapps.io. Then I installed the `rsconnect` package in R, and authorized my account with my secret key. Once my account was authorized, I simply had to run a few lines of code to upload the app.

```
> library(rsconnect)
> rsconnect::deployApp('Map-1')
```





Looking at the Total Posts per Day graph, it is clear that March 2012 had thousands of posts per day. On the other hand, March 2018 had sometimes less than 100 posts per day. By setting a strict scale on line 80 of app.R, the viewer can easily see the difference in activity between these months while looking at the shiny map.