Advanced Data Science I Project

Ben Barrett

October 11, 2017

R Markdown

Introduction

ADD A FULL INTRODUCTION - ANY EVIDENCE OF TWEETS AND DISASTERS; USING TWEETS IN ANALYSES (OR OTHER SOCIAL MEDIA), WITH REFERENCES

These methods are designed to pull Twitter data to identify times and areas hardest hit by Hurricane Harvey. Geocoding the tweets primarily relies on the user location (entered once when a user first starts Twitter), rather than the location associated with an individual tweet, as most users turn the tweet location data off. In this case, however, these methods should be appropriate - as people who fled the impacted area before the storm hit will still have their user location linked to the impacted area. This mode of analysis relies on the assumption that the number of tweets about Hurricane Harvey coming out of an area correlates with the level of destruction in that area.

Data and Methods

Data

The Python library 'Tweepy' was used to connect to the Twitter Streaming API and download relevant tweets. The Python program, twitter_streaming.py (reproduced below), was adapted from code provided by Mikael Brunila (1), and used to live stream tweets. The stream was set to search for the hashtags 'HurricaneHarvey' and 'HurricaneHarveyRelief', and was started at 9:10AM on 9/1/2017. The live tweet stream was stopped at 9:56AM on 9/4/2017, which resulted in a program run time of 36 hours, 46 minutes and a total of 3,289,336 KB (3.290 GB) of Twitter data collected. This corresponds to 1,491.086 KB of data per minute, on average. All of the output was saved as a JSON file, twitter_data.json, available in my Dropbox.

```
#twitter_streaming.py

#Import the necessary methods from tweepy library
import tweepy
from tweepy import Stream
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
import json

#Variables that contains the user credentials to access Twitter API
access_token = "Access_Token"
access_token_secret = "Access_Token_Secret"
consumer_key = "Consumer_Key"
consumer_secret = "Consumer_Secret"

auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)

api = tweepy.API(auth)
@classmethod
def parse(cls, api, raw):
    status = cls.first_parse(api, raw)
    setattr(status, 'json', json.dumps(raw))
```

```
    return status

# Status() is the data model for a tweet
tweepy.models.Status.first_parse = tweepy.models.Status.parse
tweepy.models.Status.parse = parse
class MyListener(StreamListener):

     def on_data(self, data):
        try:
            with open('twitter_data.json', 'a') as f:
                f.write(data)
                return True
        except BaseException as e:
            print("Error on_data: %s" % str(e))
        return True

     def on_error(self, status):
        print(status)
        return True

#Set the hashtag to be searched
twitter_stream = Stream(auth, MyListener())
twitter_stream.filter(track=['HurricaneHarvey', 'HurricaneHarveyRelief'])
```

Methods

Because of the large data file size, the twitter_data.json file was first streamed into R using a handler to randomly sample 25% of each page of JSON lines (total run time = 1.5 hours, code reproduced below). This code found 420,294 records, each corresponding to a tweet collected during the livestream. The random sample yielded a study dataset of 22,689 tweets. The reduced file serves as the basis for reproducibility.

```
install.packages("jsonlite")
install.packages("rdrop2")
install.packages("tibble")
library(jsonlite)
library(rdrop2)
library(tibble)

# Setting up the Dropbox authentication
token <- drop_auth()
saveRDS(token, "~/Advanced Data Science I/AdvDataSci_Project/.gitignore.httr-oauth.RDS")
drop_auth(rdstoken= "~/Advanced Data Science I/AdvDataSci_Project/.gitignore.httr-oauth.RDS")
drop_share("twitter_data.json")

# Streamining in Twitter data
con_in <- url("https://www.dropbox.com/s/0rubxdgwvt9od66/twitter_data.json?dl=1")
con_out <- file(tmp <- tempfile(), open = "wb")
set.seed(4)
stream_in(con_in, handler= function(randsam){
  randsam <- randsam[sample(1:nrow(randsam), round(0.25*length(randsam))),]
  stream_out(randsam, con_out, pagesize=1000)
}, pagesize=500, verbose=TRUE)
close(con_out)

tweets <- stream_in(file(tmp))
```

```
nrow(tweets)
unlink(tmp)

# Flattening and saving the streamed in data
tweets_flat <- flatten(tweets)
tweets_tbl <- as_data_frame(tweets_flat)
saveRDS(tweets_tbl, "tweets_tbl.rds")
saveRDS(tweets_flat, "tweets_flat.rds")
```

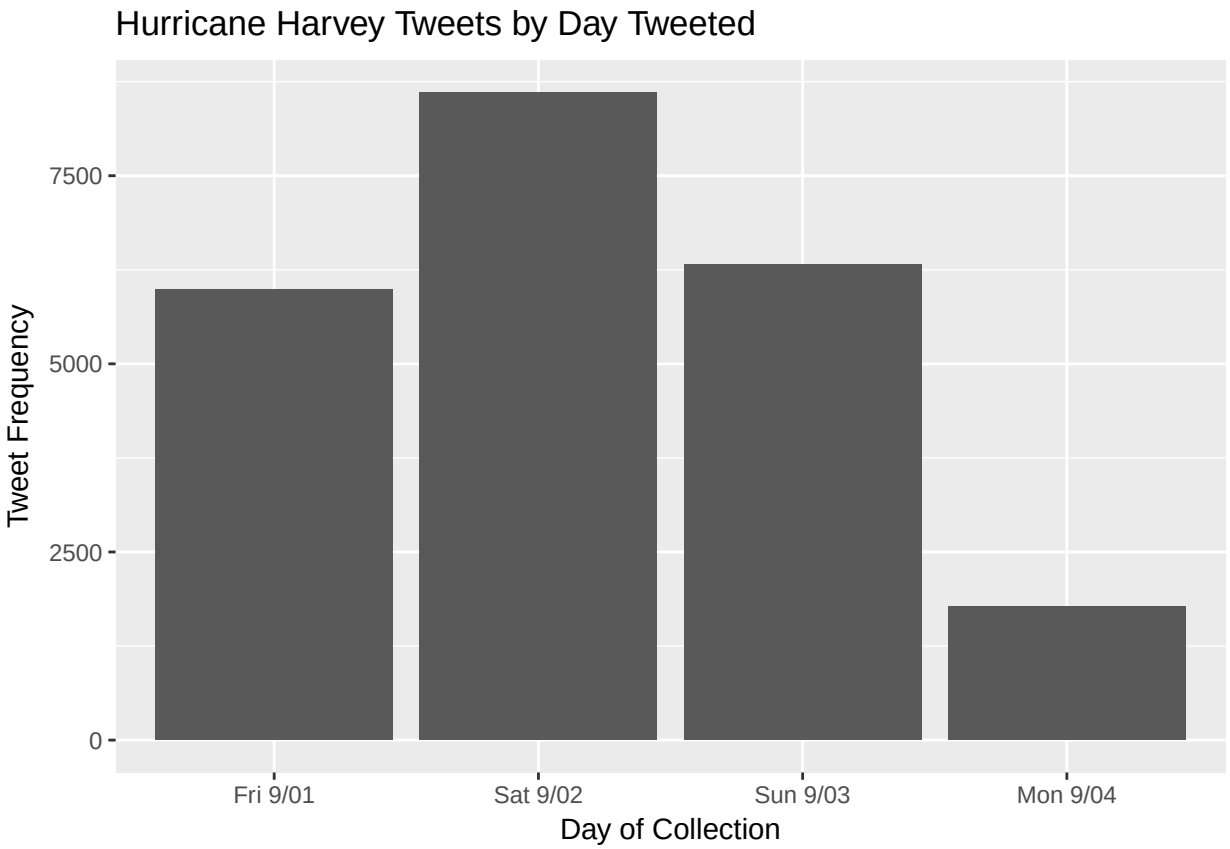
Tweets were restricted to only those that had a user location recorded, which gave a sample of 15,535 tweets. Then, tweets were further restricted to those either with geographic coordinates already saved, those made in Texas or Louisiana (tweet location), those with a user location set within Texas or Louisiana, or those with a user location of geographic coordinates. This left 1,842 tweets, 21 of which had geographic coordinates already saved, and 80 of which had a location of tweet creation recorded (state and city). The user locations set as geographic coordinates were then extracted, and set as the tweet coordinates if the tweet did not already have geographic coordinates assigned to it. Google Maps was used to assess whether these new coordinate values fell within Texas or Louisiana and, if they did not, the observations were deleted. Finally, tweet locations and user locations that were only assigned a state (Texas or Louisiana) and not a city, and were not already assigned geographic coordinates, were deleted. Tweet locations and user locations that were assigned a state and city had their geographic coordinates imputed by assigning a random latitude and random longitude, bound between the respective city's most extreme border points, as identified by Google Maps. If a tweet had an entry for tweet location and user location, the tweet location was used preferentially. This process left a final sample size of XXXX, with each tweet having geographic coordinates associated with it.

NOTE: THE ABOVE STEPS HAVEN'T BEEN FULLY COMPLETED YET - I'M STILL WORKING ON PULLING THE USER LOCATION CITY FROM THE ONE THAT CAME LINKED TO THE TWEET - USERS ARE ALLOWED TO ENTER WHATEVER THEY WANT AS A USER LOCATION, SO THERE IS A TON OF VARIABILITY IN TEXT FORMAT AND CONTENT, SO PROCESS IS TAKING AWHILE. WOULD BE OPEN TO SUGGESTIONS ON HOW TO EXPEDITE THIS, AND FOR THOUGHTS REGARDING THE IMPUTATION OF GEOGRAPHIC COORDINATES.

Results

PLOT THE TWEETS ON A SHAPEFILE OF TEXAS AND LOUISIANA USING GGPLOT2. THEN, USE SPATIAL STATISTICS TO ASSESS SPACIAL INTENSITY AND CLUSTERING (CODE FOR THIS IS ALREADY CREATED, JUST NEED TO APPLY ONCE FULLY CLEANED DATASET IS FINISHED).

Figure to help visualize on what days the tweets were collected from, and how many tweets came from each day. WILL MOST LIKELY DELETE THIS ONCE THE TWEET COORDINATES ARE OVERLAID ON A SHAPEFILE.



References 1: Brunila, M. (2017). Scraping, extracting and mapping geodata from Twitter. http://www.mikaelbrunila.fi/2017/03/27/scraping-extracting-mapping-geodata-twitter/. [accessed September 1, 2017].