Assessing the impacts of COVID-19 pandemic on public opinion concerning policing using Twitter data - A demonstration using 'Opitools' package

Author:
Adepeju, M.
Big Data Centre, Manchester Metropolitan University, Manchester, M15 6BH

Date:
2021-02-24

Abstract

The lack of tools for analysing cross-impact of opinions expressed amongst multiple subjects, within a text document, facilitate the development of 'opitool' package. For instance, given a collection of tweets on a specific subject A, a researcher may want to assess whether the opinion expressed on subject A in relation to another (secondary) subject B has significantly impacted the overall opinion from the text document. For a real life example, we may want to examine if public opinion expressed concerning policing (as subject A) has been significantly impacted by the public concerns for COVID-19 pandemic (as subject B) (see Adepeju and Jimoh, 2021). This document describes how the aforementioned analysis can be completed using the opitools package.

Introduction

The opitools is an opinion analytical toolset designed for assessing cross-impacts of opinion expressed on multiple subjects in an opinion-based text documents (OTD) from a social media platform. An OTD (input as textdoc) should composed of individual text records on a specific subject (A). An example of an OTD is a collection of Twitter posts concerning a specific topic or hashtag. Any other subjects referenced in relation to the primary subject A can be referred to as secondary subjects, and they can be identified through the keywords used in the text records.In the article (Adepeju and Jimoh 2021), we described how to deploy opitools in order to answer a real-life research question, such as 'what are the impacts of COVID-19 pandemic (secondary subject) on the public opinion concerning policing (primary subject) across England and Wales?' The opitools may be used to answer similar questions relating to many public organisations in order to unravelling important issues that may be driving confidence and trust in relation to their services.

Downloading Twitter data

The rtweet package (Kearney 2019) is one of the R packages that provide access to the Twitter API for data download. The code section below can be used to download tweets for a pre-defined geographical coverage (lat:'53.805,long:-4.242,radius: 350mi') within the last seven days (free download). We will be downloading tweets relating to 'policing'. Thus, I define the search words as {"police", "policing", "law enforcement"} Note: A user need to first secure access to Twitter developer platform (from here), then follow instructions on this page on how to obtain a set of tokens (keys) required connect to the Twitter API.

Working directory

Set a local directory:

```
WORKING_DIR <- 'C:/R/Github/JGIS_Policing_COVID-19'

#setting working directory
setwd(WORKING_DIR)
```

Installing libraries

```
library(opitools) #for impact analysis
library(rtweet) #for data download
library(twitteR) #for setting up Twitter authorization
#> Warning: package 'twitteR' was built under R version 4.0.3
```

Running essential function and define tokens

Free Twitter developer accounts have a restriction of 18,000 tweets per 15 minutes, otherwise a user may loose access (temporarily) to download data. Therefore, it is important to wait for 15 minutes after every 18,000 tweets download. Run the `waitFun` function to help surpass the restriction as the data is being downloaded using different search words.

```
#Run function
waitFun <- function(x){
  p1 <- proc.time()
  Sys.sleep(x)
  proc.time() - p1
}

#specify tokens and authorize
#Note: replace asterisk with real keys

consumer_key <- '*******************************'
consumer_secret <- '*******************************'
access_token <- '*******************************'
access_secret <- '*******************************'

setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)

token <- create_token(
  app = "AppName", #App name
  consumer_key = consumer_key,
  consumer_secret = consumer_secret)
```

Start download

```
#Define the keywords for subject A
keywords <- c("police", "policing", "law enforcement")

#tweets holder
all_Tweets <- NULL

#Loop through each keyword and wait for 15 minutes
#and row-bind the results
for(i in seq_len(length(keywords))){
```

```
  tweets_g1 <- NULL

  tweets_g1 <- search_tweets(q=keywords[i],  n=17500, type="recent", include_rts=TRUE,
                             token = token, lang="en",geocode='53.805,-4.242,350mi')

  if(nrow(tweets_g1)!=0){
    tweets_g1 <- tweets_g1 %>% dplyr::mutate(class=keywords[i])
    all_Tweets <- rbind(all_Tweets, tweets_g1)
  }

  flush.console()
  print(paste(nrow(tweets_g1), nrow(tweets_g1), sep="||"))
  print("waiting for 15.5 minutes")
  waitFun(960)
}

#save the output
write_as_csv(all_Tweets, "tweets.csv", na="NA", fileEncoding = "UTF-8")
```

Exploration of a text document

Following the data download, a user may wish to explore the inherent characteristics of the data. For example, "What is the nature of word usage in a Twitter text document compareed to typical natural language document?" This question can be answered by comparing the log frequency plot of the document with the Zipf's distribution (Zipf 1936) – the famous frequency distribution expected of a natural language document. By Zipf's distribution, we expect the frequency of words contained in the document to be inversely proportional to its rank in a frequency table. The `word_distrb` function can be used to generate a plot (e.g Figure) that shows the comparison.

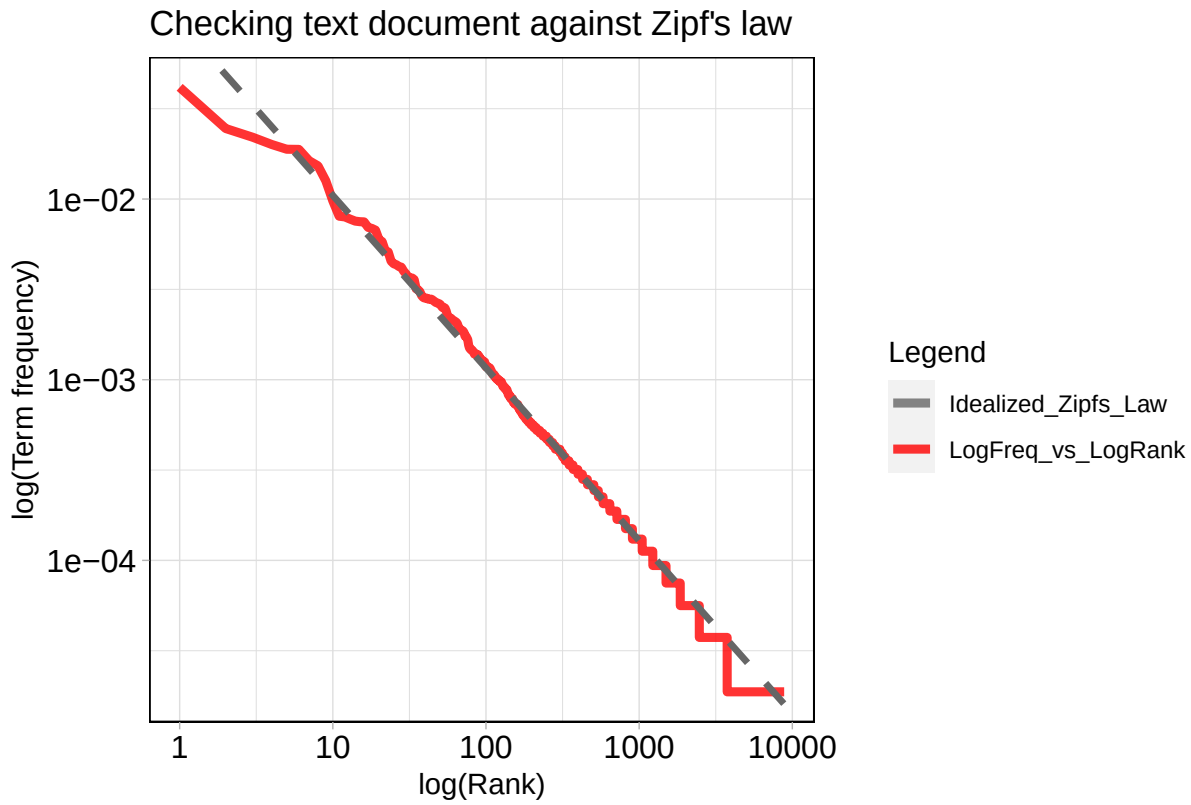
```
#using a randomised Twitter data from 'opitools'

#data(tweets)

tweets_dat <- as.data.frame(tweets[,1])

plt = word_distrib(textdoc = tweets_dat)

#to plot
plt
```



For a natural language text, the relationship between the rank and the frequency should have a negative slope with all points falling on a straight line. Any deviation from the straight line can be considered an element of imperfection of the text document. An example of such element is the presence of abnormal terms in the document. From Figure 1 we divide the graph into the three sections: the upper, the middle and the lower section. By fitting a regression line (to represent an ideal Zipf's distribution), we can see what the slope of the upper section is quite different from the rest of the graph. The deviation at the high rank is quite unusual because a corpus of English language often contains enough of common words, such as 'the', 'of', and 'at', in order to obey the Zipf's law. This deviation only suggests a significant use of a wide range of abbreviation instead of these common words. The remaining part of the graph suggests that the documents . Apart from the small deviation at the upper section of the graph, we can state that the law holds within most parts of the Twitter text document.

Impact Analysis

The randomized Twitter data (above) relates to the tweets containing law enforcement-related search words (e.g. policing, law enforcement, etc), and are downloaded during the era of COVID-19 pandemic. We will like to assess the impacts of COVID-19 pandemic (as a secondary subject B) on the original theme of the data, i.e. `policing during pandemic`. First, we need to identify keywords that have been used to reference COVID-19 pandemic in the text data. A user can employ any relevant analytical approach (e.g. frequency analysis) in order to identify such keywords. Alternatively, a user can input those keywords manually. Thus, for COVID-19 pandemic, the defined keywords can be accessed by typing `covid_keys` in the console:

```
> covid_keys

#          keys
#1     pandemic
```

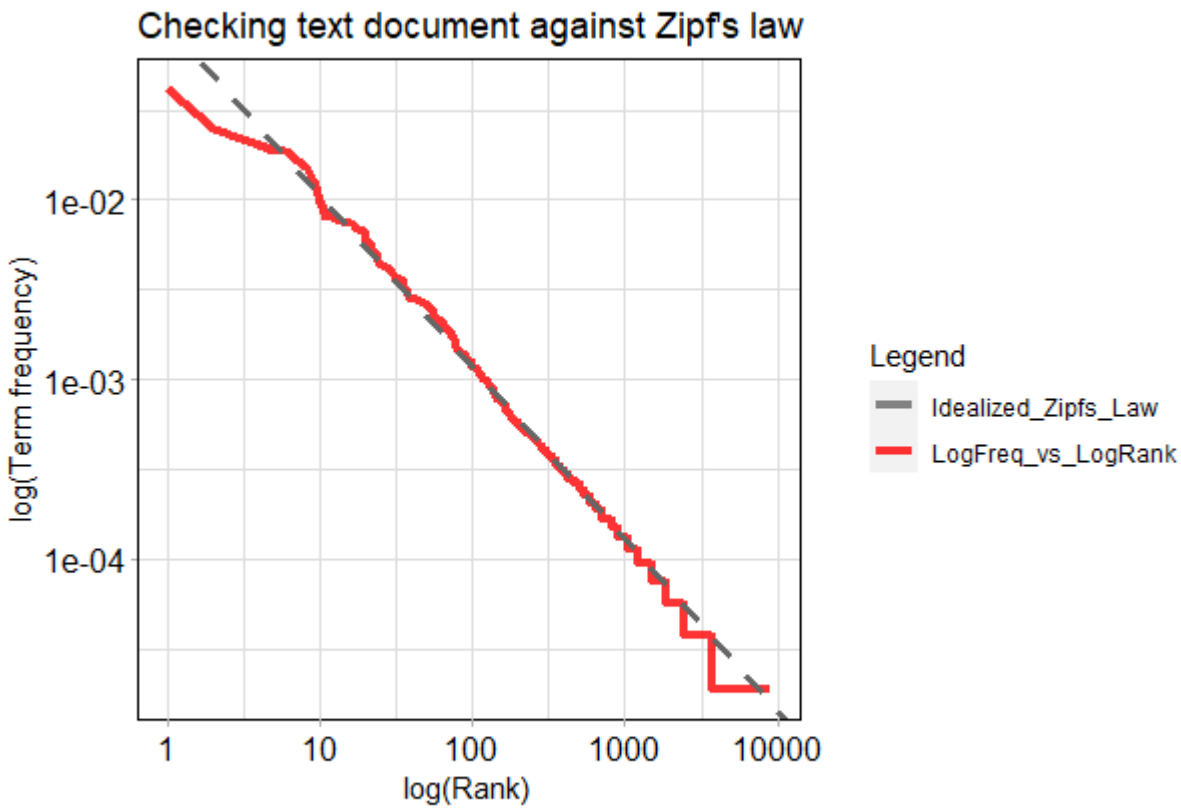


Figure 1: Figure 1: Data freq. plot vs. Zipf's distribution

```
#2    pandemics
#3     lockdown
#4    lockdowns
#5       corona
#6  coronavirus
#7        covid
#8      covid19
#9     covid-19
#10       virus
#11     viruses
#12  quarantine
#13      infect
#14     infects
#15   infecting
#16    infected
```

The main impact analysis can be executed by running the following codes:

```
# Get an n x 1 text document
tweets_dat <- as.data.frame(tweets[,1])

# Run the analysis

results <- opi_impact(tweets_dat = tweets_dat, sec_keywords=covid_keys, metric = 1,
                      fun = NULL, nsim = 99, alternative="two.sided",
                      pplot = TRUE, quiet=TRUE)

print(results)
```

To preview all the output variables of `results` object, type:

```
names(results)
```

- The description of these variables are as follow:
  - `test` - title of the analysis
  - `criterion` - criterion for determining the significance value
  - `exp_summary` - summary of expected opinion scores
  - `p_table` - details of Statistical Significance
  - `p_key` - keys for interpreting the statistical significance value
  - `p_formula` - function of opinion score employed

For example, to preview the `p_table`, type:

```
results$p_table
```

The definition of opinion function may vary according to the domain of interest, therefore, a user is allowed to specify a function for computing the opinion scores. Let's say we define an opinion score as `score = (P + O - N)/(P + O + N)`, where `P`, `O`, and `N`, represent the amount of positive, neutral and negative, text records (tweets), respectively. Thus, the analysis can be re-run with the user-defined opinion score function, as follows:

```
#define opinion score function
```

```
myfun <- function(P, N, O){
   score <- (P + O - N)/(P + O + N)
   return(score)
}
```

Re-run analysis

```
results <- opi_impact(tweets_dat = tweets_dat, sec_keywords=covid_keys, metric = 5,
                      fun = myfun, nsim = 99, alternative="two.sided",
                      pplot = TRUE, quiet=TRUE)

print(results)
```

Conclusion

The `akmedoids` package has been developed in order to aid the replication of a place-based crime inequality investigation conducted in Adepeju et al. (2019). Meanwhile, the utility of the functions in this package are not limited to criminology, but rather can be applicable to longitudinal datasets more generally. This package is being updated on a regular basis to add more functionalities to the existing `functions` and add new functions to carry out other longitudinal data analysis.

We encourage users to report any bugs encountered while using the package so that they can be fixed immediately. Welcome contributions to this package which will be acknowledged accordingly.

References

Adepeju, M., and F. Jimoh. 2021. "An Analytical Framework for Measuring Inequality in the Public Opinions on Policing – Assessing the Impacts of Covid-19 Pandemic Using Twitter Data." *Journal of Geographical Information System (in Press)*.

Kearney, MW. 2019. "Rtweet: Collecting and Analyzing Twitter Data." *Journal of Open Source Software* 4(42): 1829.

Zipf, G. 1936. *The Psychobiology of Language*. Routledge.