Final Project: Data Mining Twitter in R

Danielle Yoseloff

12/12/2016

Introduction

My topic of interest is spelling on social media in the US. I want to see how frequently Americans misspell the four words: their, a lot, received, and separate with their common misspellings(thier, alot, recieved, seperate). I want to draw some conclusion about different regions of the US so I used geolocated tweets from Boston, Los Angeles, Chicago, Houston, and Atlanta. To get a true random sample I searched the 100 most recent tweets which contained either the correct spelling or the most frequent misspelling per city for the four most commonly misspelled words. When looking at our results we must keep in mind that twitter itself does not have a form of auto correct however, most smartphones and devices people tweet from do. The output will give sentence outputs of the findings.

Note: The most frequently misspelled words and their most common misspelling list was taken from a study this article refers to http://www.grammar.net/misspelledwords

Another note: In this project I chose to use entirly new topics which I had yet to explore in previous assignmnets. (function use, API connection, wordcloud, and mapping)

```
library(twitteR)
library(tm)
```

```
## Loading required package: NLP
```

```
library(SnowballC)
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
library(ggmap)
library(maps)

consumer_key <- "rXe2OXtCADNyIFzaNGxGhexnM"
consumer_secret <- "zl2R9x42I2QVcS7OJ67zqcDKqyCoBHpjybaJjuLVDC2R3TYspo"
access_token <- "2985033126-vj0tyMKeZWS4zfMlj7ClwJEjvlcBDjhXBD4xqry"
access_secret <- "lO6GshHPc5uIFe1FJjnJtiWhsphBPRySAV8fOCCCZw5EZ"
setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
```

```
## [1] "Using direct authentication"
```

```
correct <- c("their", "a lot", "received", "separate")
misspelled <- c("thier", "alot", "recieved", "seperate")

cities <- data.frame(matrix(ncol = 2, nrow = 0))
colnames(cities) <- c("name", "geocode")
cities <- rbind(cities, data.frame(name="Boston", geocode="42.3601,-71.0589,30mi"))
cities <- rbind(cities, data.frame(name="Los Angeles", geocode="34.0522,-118.2437,30mi"))
cities <- rbind(cities, data.frame(name="Chicago", geocode="41.8781,-87.6298,30mi"))
cities <- rbind(cities, data.frame(name="Houston", geocode="29.7604,-95.3698,30mi"))
cities <- rbind(cities, data.frame(name="Atlanta", geocode="33.7490,-84.3880,30mi"))

count <- 100
data_txt <- list()
results <- list()

apply(cities, 1, function(city) {
  for(wordIndex in 1:length(correct)) {
    searchTerm <- paste(paste(correct[wordIndex], "OR", misspelled[wordIndex], sep=' '),
                        '-filter:retweets', sep=' ')
    tweets <- searchTwitter(searchTerm, n=count, geocode=as.character(city['geocode']),
                        resultType="recent")
    sapply(tweets, function(x) {
      index <- length(data_txt) + 1
      data_txt[[index]] <<- x$text
    })
    incorrectPercentage <- 100 - length(Filter(function(x) grepl(correct[wordIndex],
                        tolower(iconv(x$text,to="utf-8-mac")),fixed=TRUE), tweets))/count*100
    results[[length(results) + 1]] <<- c(city = city['name'], percent = incorrectPercentage)
    print(paste(city['name'], "misspells", correct[wordIndex], "with",
                misspelled[wordIndex], incorrectPercentage, "percent of the time"))
  }
})
```

```
## [1] "Boston misspells their with thier 2 percent of the time"
## [1] "Boston misspells a lot with alot 7 percent of the time"
## [1] "Boston misspells received with recieved 5 percent of the time"
## [1] "Boston misspells separate with seperate 14 percent of the time"
## [1] "Los Angeles misspells their with thier 2 percent of the time"
## [1] "Los Angeles misspells a lot with alot 9 percent of the time"
## [1] "Los Angeles misspells received with recieved 5 percent of the time"
## [1] "Los Angeles misspells separate with seperate 15 percent of the time"
## [1] "Chicago misspells their with thier 4 percent of the time"
## [1] "Chicago misspells a lot with alot 2 percent of the time"
## [1] "Chicago misspells received with recieved 5 percent of the time"
## [1] "Chicago misspells separate with seperate 5 percent of the time"
## [1] "Houston misspells their with thier 1 percent of the time"
## [1] "Houston misspells a lot with alot 6 percent of the time"
## [1] "Houston misspells received with recieved 4 percent of the time"
## [1] "Houston misspells separate with seperate 4 percent of the time"
## [1] "Atlanta misspells their with thier 0 percent of the time"
## [1] "Atlanta misspells a lot with alot 6 percent of the time"
## [1] "Atlanta misspells received with recieved 3 percent of the time"
```

```
## [1] "Atlanta misspells separate with seperate 8 percent of the time"
```

```
## NULL
```

```
data_txt <- unlist(data_txt)
```

Word cloud

A word cloud is a fun way to look at data to see trends in topic. I created the style of the word could to emulate the topic of spelling with an elemntart font and playful yet readable colors. I put the data for all tweets (all words for all cities) into the word cloud. As expected there does not appear to be any trending word themes.

```
data_txt <- iconv(data_txt,to="utf-8-mac")
data_corpus <- Corpus(VectorSource(data_txt))
data_clean <- tm_map(data_corpus, removePunctuation, lazy=TRUE)
data_clean <- tm_map(data_clean, content_transformer(tolower), lazy=TRUE)
data_clean <- tm_map(data_clean, removeWords, stopwords("english"), lazy=TRUE)
data_clean <- tm_map(data_clean, removeNumbers, lazy=TRUE)
data_clean <- tm_map(data_clean, stripWhitespace, lazy=TRUE)
wordcloud(data_clean, scale = c(4,1), max.words = 200, random.order = F,
          colors=brewer.pal(10,"Paired"), vfont=c("script", "plain"))
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : postcrossing could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : recognition could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : another could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : anyone could not be fit on page. It will not be
## plotted.
```

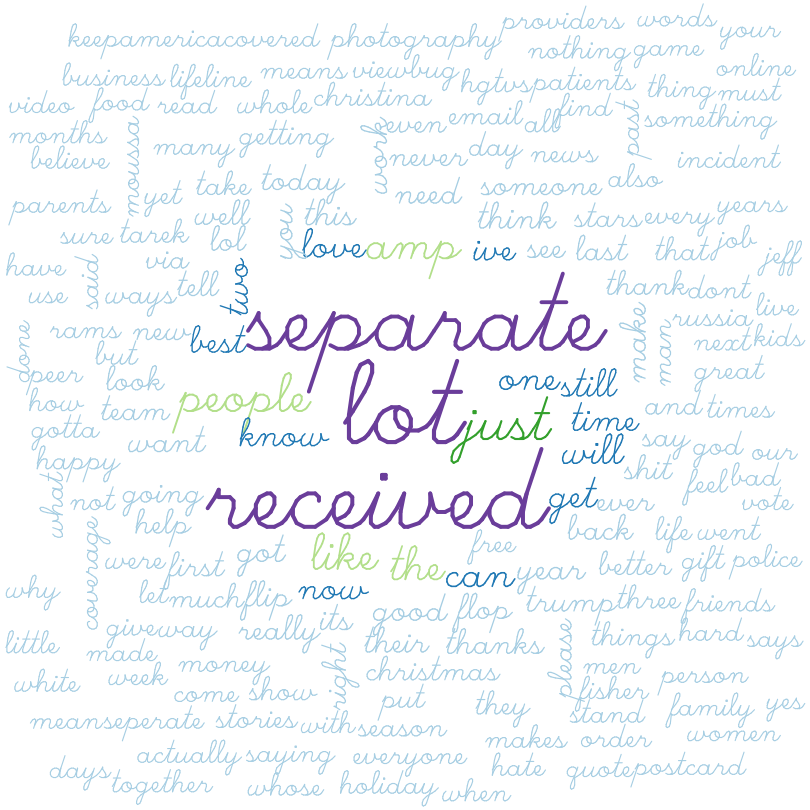
```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : customer could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : guys could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : head could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : mind could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : play could not be fit on page. It will not be plotted.

## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : students could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : there could not be fit on page. It will not be plotted.

## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : though could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : beautiful could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(data_clean, scale = c(4, 1), max.words = 200,
## random.order = F, : these could not be fit on page. It will not be plotted.
```



Mapping

This map is a visial representation of the information I found while doing this project. The cities pop out in orange on a visially pleasing google maps theme. The size of the dots is a precice calculation using the data

such that the size corrosponds to the total percent of the time that city spells common words incorrectly. In other words; the larger the dot, the more frequently that city misspells common words on twitter.

Note: Disregard the output with nulls. To show the function is working correctly the results variable is printed after the function.

Another note: While there is a max limit on the word cloud there may be some error if words cannot fit on the page.

```
boston <- 0
houston <- 0
los_angeles <- 0
chicago <- 0
atlanta <- 0

sapply(results, function(element) {
  if(as.character(element['city.name']) == "Boston") {
    boston <<- boston + as.numeric(element['percent'])
  }
  if(as.character(element['city.name']) == "Houston") {
    houston <<- houston + as.numeric(element['percent'])
  }
  if(as.character(element['city.name']) == "Los Angeles") {
    los_angeles <<- los_angeles + as.numeric(element['percent'])
  }
  if(as.character(element['city.name']) == "Chicago") {
    chicago <<- chicago + as.numeric(element['percent'])
  }
  if(as.character(element['city.name']) == "Atlanta") {
    atlanta <<- atlanta + as.numeric(element['percent'])
  }
})
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
##
## [[8]]
```

```
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## [1] 0
##
## [[18]]
## [1] 6
##
## [[19]]
## [1] 9
##
## [[20]]
## [1] 17
```

```
results
```

```
## [[1]]
## city.name   percent
##  "Boston"       "2"
##
## [[2]]
## city.name   percent
##  "Boston"       "7"
##
## [[3]]
## city.name   percent
##  "Boston"       "5"
##
## [[4]]
## city.name   percent
```

```
##  "Boston"      "14"
##
## [[5]]
##     city.name       percent
## "Los Angeles"           "2"
##
## [[6]]
##     city.name       percent
## "Los Angeles"           "9"
##
## [[7]]
##     city.name       percent
## "Los Angeles"           "5"
##
## [[8]]
##     city.name       percent
## "Los Angeles"          "15"
##
## [[9]]
## city.name   percent
## "Chicago"       "4"
##
## [[10]]
## city.name   percent
## "Chicago"       "2"
##
## [[11]]
## city.name   percent
## "Chicago"       "5"
##
## [[12]]
## city.name   percent
## "Chicago"       "5"
##
## [[13]]
## city.name   percent
## "Houston"       "1"
##
## [[14]]
## city.name   percent
## "Houston"       "6"
##
## [[15]]
## city.name   percent
## "Houston"       "4"
##
## [[16]]
## city.name   percent
## "Houston"       "4"
##
## [[17]]
## city.name   percent
## "Atlanta"       "0"
##
```

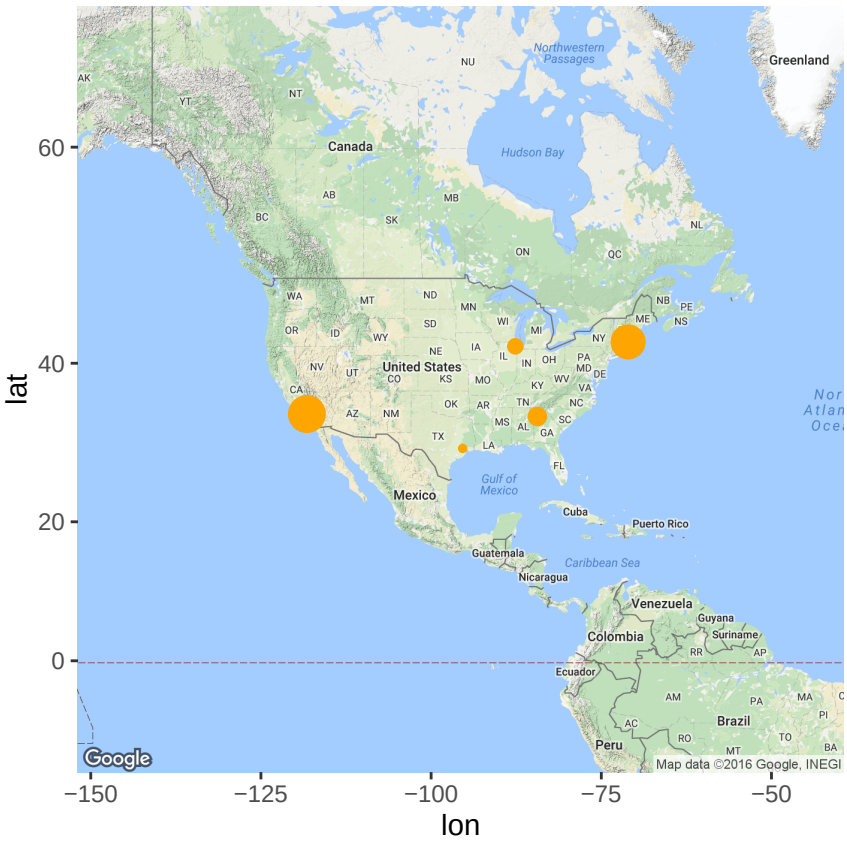
```
## [[18]]
## city.name   percent
## "Atlanta"       "6"
##
## [[19]]
## city.name   percent
## "Atlanta"       "3"
##
## [[20]]
## city.name   percent
## "Atlanta"       "8"
```

```
percents <- c(boston, los_angeles, chicago, houston, atlanta)
max_percents <- max(percents)
percents <- sapply(percents, function(x) x/max_percents*6)

map <- cbind(geocode(as.character(cities$name)), cities)
```

```
## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Boston&sensor=false

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Los%20Angeles&sensor=

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Chicago&sensor=false

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Houston&sensor=false

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=Atlanta&sensor=false
```

```
ggmap(get_map(location = 'usa', zoom = 3)) + geom_point(data=map, aes(x=lon, y=lat, size=percents),show
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=usa&zoom=3&size=640x640&scale=2&m

## Information from URL : http://maps.googleapis.com/maps/api/geocode/json?address=usa&sensor=false
```


lat
60
40
20
0
−150
−125
−100
−75
−50
lon
Northwestern Passages
Greenland
Hudson Bay
Canada
United States
Mexico
Gulf of Mexico
Cuba
Puerto Rico
Caribbean Sea
Guatemala
Nicaragua
Venezuela
Guyana
Suriname
Colombia
Ecuador
Brazil
Peru
Google
Map data ©2016 Google, INEGI