

## Twitter Analysis

|                    |   |
|--------------------|---|
| <b>Course:</b>     | Introduction to Data Analysis                     |
| <b>University:</b> | UCSC Silicon Valley extension                     |
| <b>Instructor:</b> | Pramod Gupta                                      |
| <b>Students:</b>   | Renato Roschel Ribeiro<br>Christian Vargas Castro |
| <b>Assignment:</b> | Project proposal                                  |

### Introduction:

With the advancement in technology, communication grew exponentially in the last two decades. It is now easier to communicate and connect with people across the world. As technology grows users to get connected to different parts of the web and one of those branches is social networks.

Social networks are a powerful tool that not only has shown can help people to express their ideas, they also can being used as an advertisement platform for companies from any economic sector in the world.

With millions of available data in social networks, nowadays we can get enormous information about how people behaves, for instance, why we don't better use this data to analyze consumers behaviors and interactions with companies.

### Objective:

To analyze Twitter data in order to provide information for marketers. The project consists in to connect to the Twitter API using R and Rstudio. Once connected, the script is set up to pulling out data from @samsungus, @lgus, @moto and @sonyxperia from tweeter timeline and the search engine will be stored in CSV files.

### Companies:

- Samsung - <https://twitter.com/samsungmobileus>
- LG - <https://twitter.com/LGUS>
- Moto - <https://twitter.com/moto>
- Sony - <https://twitter.com/sonyxperia>

It was chosen only mobile accounts from the USA.

## Packages used:

|  |  |   |
|--|--|---|
| <ul style="list-style-type: none"><li>• twitterR</li><li>• Sentimentr</li><li>• Maps</li><li>• sentR</li><li>• Httpuv</li><li>• Mapproj</li><li>• data.table</li></ul> | <ul style="list-style-type: none"><li>• RSQLite</li><li>• Htttr</li><li>• ggedit</li><li>• RJSONIO</li><li>• Tidytext</li><li>• Broom</li><li>• Scales</li></ul> | <ul style="list-style-type: none"><li>• Stringr</li><li>• Devtools</li><li>• Rjson</li><li>• Ggpubr</li><li>• Ggplot2</li><li>• Ggmap</li><li>• Stringi</li></ul> |
|--|--|---|

## Connecting to Twitter API:

Twitter app: <https://apps.twitter.com>

Create a project and get the access:

<https://apps.twitter.com/app/13782765/show>


iCloud FGTS Itaú Personalité UOL Despesas Cacula Adm Facebook Outlook.com Gmail globo.com Bradesco Saúde

Twitter Application Management

# ProjectR

Test OAuth

Details Settings Keys and Access Tokens Permissions

 My project in R for UCSC  
<http://www.roschel.net>

## Organization

Information about the organization or company associated with your application. This information is optional.

|                      |      |
|----------------------|------|
| Organization         | None |
| Organization website | None |

## Application Settings

Your application's Consumer Key and Secret are used to [authenticate](#) requests to the Twitter Platform.

|                        |   |
|------------------------|---|
| Access level           | Read and write ( <a href="#">modify app permissions</a> )         |
| Consumer Key (API Key) | <a href="#">tokens</a> ( <a href="#">manage keys and access</a> ) |
| Callback URL           | <a href="http://127.0.0.1:1410">http://127.0.0.1:1410</a>         |

```
options(httr_oauth_cache=T);

api_key <- "get it from app.twitter.com.br";

api_secret <- " get it from app.twitter.com.br ";

access_token <- "get it from app.twitter.com.br ";

access_token_secret <- " get it from app.twitter.com.br ";

setup_twitter_oauth(api_key,api_secret);
```

### Data Source:

```
# Get information by account:
getUser("samsungmobileus");

# Get n timeline posts:
userTimeline('samsungmobileus', n=n_timelines);

# Get n company mentions by users:
searchTwitter("@samsungmobileus", n=n_timelines);

# We will use n=500 for the demonstration. But, in the real world, we could get more
data and stored the result.
```

### Cleaning up data for map projection – part 1 – before Google Maps API:

```
# Step 1 - Removing users without location:
total.mentions_c1<-subset(total.mentions_c1, location!="");

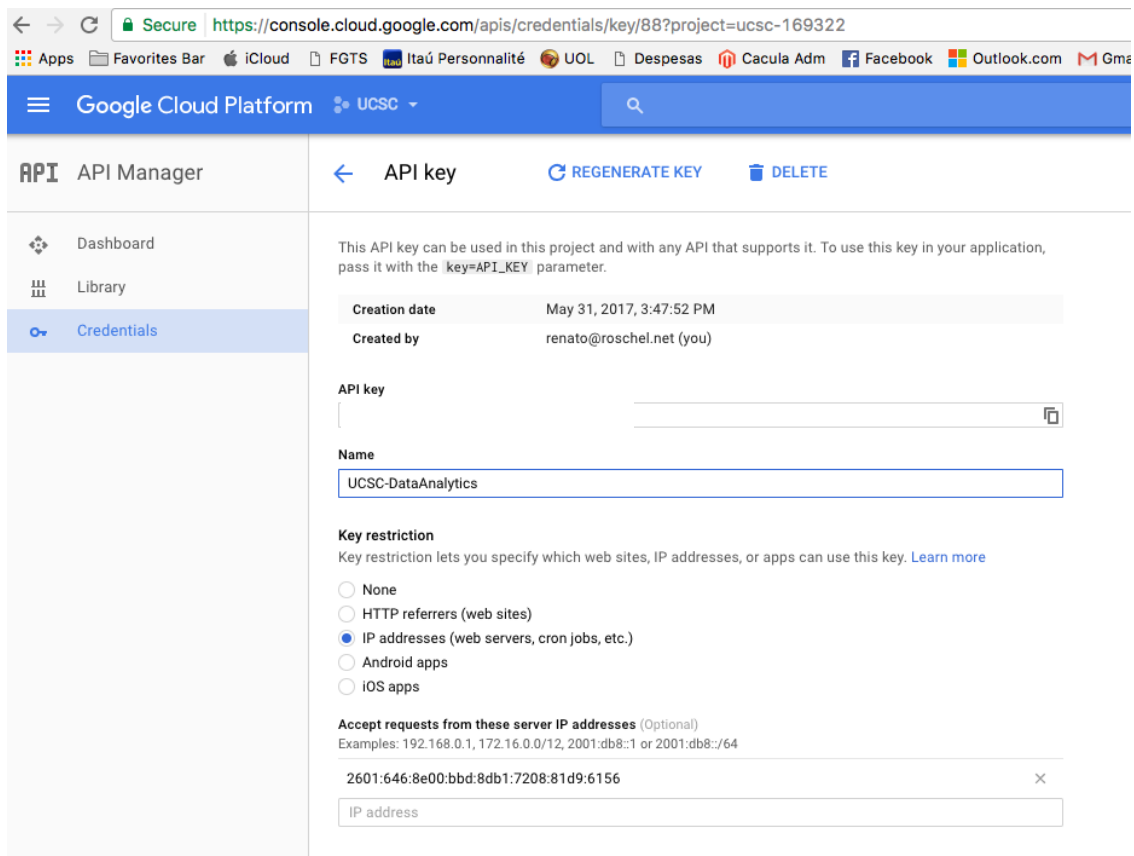
# Step 2 - Removing special characters (Our own function):
# this function was built in order to
remove_special_chars("410 ☘️ East Baltimore "); eliminate special characters

total.mentions_location <- sapply(total.mentions_c2$location,remove_special_chars,simplify = F);
```

### Geocoding using Google Maps API:

<https://console.cloud.google.com/apis/credentials?project=ucsc-169322>

It was created a project called UCSC for this project. Since the project is created you need to start the service Google Maps Geocoding API. Every follower location will be submitted to the API in order to get longitude and latitude.



Warning 1) Be aware. Google IPI provides only 2,500 search times per day for free.

Warning 2) In this project, we geocoded 200 addresses per company, 600 total.

Warning 3) In order to use the function `geocode()`, we needed to modify the code and apply our API key on it. This will allow us to use our quote of 2,500 geocoding times per day. Therefore, the code was downloaded and edited line 174

Warning 4) `geocode_results <- sapply(mentions$location, geocode_apply, simplify = F)`

## Cleaning up data for map projection – part 2 – after Google Maps API:

```
# Step 1 - Eliminating results that returns status != OK:
condition_c1 <- sapply(samsung.geocode_results, function(x) x["status"]=="OK");
geocode_results <- samsung.geocode_results[condition_c1];

# Step 2 - Eliminating results that we don't know exactly where is:
condition_c2 <- lapply(samsung.geocode_results, lapply, length);
condition_c2a <- sapply(condition_c2, function(x) x["results"]=="1");
geocode_results <- samsung.geocode_results[condition_c2a];
```

```

# Step 3 - Script to cleaning up geocoding (user function in R):
geocode_results <- clean_geocoding_results(geocode_results);

# Step 4 - Removing users from outside of the USA:
american_results<-subset(results_f, grepl(",", USA", results_f$Location)==TRUE);

# Step 5 – Removing results with one comma or less:
american_results$commas<-sapply(american_results$Location, function(x)
length(as.numeric(gregexpr(",", as.character(x))[[1]])));

# leave just address with two commas
american_results<-subset(american_results, commas==2);

#Drop the "commas" column:
american_results<-subset(american_results, select=-commas);

```

## Geomapping:

```

# Generate a blank map: loading a map of the United States with an Albers projection

map_projection <- map("state", proj="albers", param=c(39, 45),
                      col="#999999", fill=TRUE, bg=NA, lwd=0.2, add=FALSE, resolution=1)

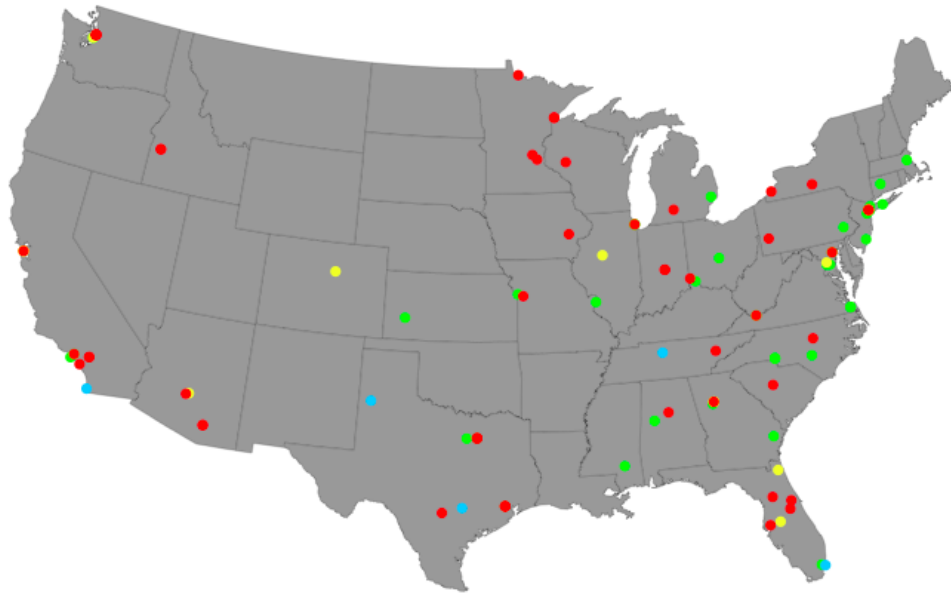
# Add points to it:
points(mapproject(samsung.american_results$lng, samsung.american_results$lat), col=NA,
bg="#2EFE2E", pch=21, cex=1.0)
points(mapproject(moto.american_results$lng, moto.american_results$lat), col=NA, bg="#F4FA58",
pch=21, cex=1.0)
points(mapproject(lg.american_results$lng, lg.american_results$lat), col=NA, bg="#FE2E2E", pch=21,
cex=1.0)
points(mapproject(sony.american_results$lng, sony.american_results$lat), col=NA, bg="#2ECCFA",
pch=21, cex=1.0)

# Add a title
mtext("The Geomapping of followers that mentioned Samsung, Moto, LG and Sony", side = 3, line = -
3.5, outer = T, cex=1.5, font=3)

# Add bottom titles
mtext("Samsung", side = 1, line = -2, outer = T, cex=1.5, font=3, col="#2EFE2E")
mtext("Moto", side = 1, line = -4, outer = T, cex=1.5, font=3, col="#F4FA58")
mtext("LG", side = 1, line = -2, outer = T, cex=1.5, font=3, col="#FE2E2E")
mtext("Sony", side = 1, line = -4, outer = T, cex=1.5, font=3,
col="#2ECCFA")

```

*The Geomapping of followers that mentioned Samsung, Moto, LG and Sony*



Moto Sony  
Samsung LG

## Sentimental analysis

### Getting word library:

```
# Getting words with positive conotation:
positive <- get_sentiments("bing");
positive <- subset(positive,sentiment=="positive");
positive <- positive$word;

# Getting words with negative conotation:
negative <- get_sentiments("bing");
negative <- subset(negative,sentiment=="negative");
negative <- negative$word;
```

### Reading data from CSV mentions:

```
# read csv file with the storage mentions:
mydata = read.csv("Project/twitter-data-mentions.csv");

# Analyzing data that was not retweet, eliminating RT's:
mydata <- subset(mydata,mydata$isRetweet==FALSE);

# Assign only text column, tweets:
test <- mydata$text;
```

### Analyzing and storing data in the twitter-data-sentimental.csv:

```
# 1. Simple Summation:
out.aggregate <- classify.aggregate(test, positive, negative);
# 2. Naive Bayes:
out.naivebayes <- classify.naivebayes(test);

out.naivebayesmydata$POS <- out.naivebayes[,1];
mydata$NEG <- out.naivebayes[,2];
mydata$POS_NEG <- out.naivebayes[,3];
mydata$SENT <- out.naivebayes[,4];

path.csv <- "twitter-data-sentimental.csv";
write.csv(mydata , file = path.csv);
```

## Twitter-data-sentimental.csv – examples:

| text  | SENT     |
|---|----------|
| @SamsungMobileUS Love this camera / phone. <a href="https://t.co/tTBtFKIiGG">https://t.co/tTBtFKIiGG</a>                      | positive |
| I'm looking forward to the release of the @SamsungMobileUS #Note8   | positive |
| @SamsungMobileUS It's been a fantastic 2 months i ½. I'm probably keeping this phone for years                                | positive |
| @SamsungMobileUS its not just good,its the best phone ive ever had  | positive |
| @SamsungMobileUS your phones are amazing  | positive |
| @SamsungMobileUS thank your much for your great customer service !!!! i love my new gear 2!!!                                 | positive |
| text  | SENT     |
| @SamsungMobileUS Ship as promised? Instead 2-3wks, tell me after i pay. Cancel? No can, already on the mail pony. Jâ€!        | negative |
| @SamsungSupport We ell i responded 2 days ago and still no service #FAIL @SamsungMobileUS @SamsungUS @SamsungMobile           | negative |
| @SamsungMobileUS Your new phones are useless without an SD Card slot and Removable batteries! When the phone freezeâ€!        | negative |
| damn @SamsungMobileUS can we get a variety on family emoji's... <a href="https://t.co/1B6ir2XnEq">https://t.co/1B6ir2XnEq</a> | negative |
| @bmac0823 @SamsungMobileUS That and they falsely advertised the " full screen " for movies, pics, videos ect... nonâ€!        | negative |
| No, @SamsungMobileUS despite your new commercials, I'm not getting my hand burned again!!                                     | negative |

## Outputs:

- SENTIMENTAL ANALYSIS ON TWEETS  
# twitter-data-sentimental.csv
- ALL DATA ABOUT MENTIONS  
# twitter-data-mentions.csv
- ALL DATA ABOUT TIMELINES  
# twitter-data-timeline.csv
- ALL HEAD DATA SUCH AS likes, followers, following, tweets  
# twitter-data.csv

## Plots base on CSV files that were created:

### Exploratory Analysis

Once we got the .csv files with the Twitter information of the companies that we chose.

We will proceed to load this csv files as data frames on R, and do the basic exploratory analysis to get familiarized with the data frame and their variables

```
# real csv file from the directory
df.twitter.temp <- read.csv(
  file="C:/Users/PC/Dropbox/Final Project/twitter-data.csv", header=TRUE, sep=",")

str(df.twitter.temp) #check temporary file
df.twitter <- df.twitter.temp[2:5]

#Check if we have NA in the df.twitter that was loaded
is.na.data.frame(df.twitter)

# take a quick look to the dataframe
```



```
tbl_df(df.twitter)

#check structure of the df.twitter
str(df.twitter)

#summary with main statistics of the df.twitter
summary(df.twitter)
```

## Cleaning up data

3 some Data Types were loaded with default data type format by R, so we proceed to re-format some features.

- The column “created” was loaded as a factor, and we want date format.
- Giving right format to data frame using as.Date,

```
## modify the datatype that came from the raw data
## the column "created" will be re-formatted to have a date datatype,
## to proceed with plot frequencies with date format

myDate <- as.Date(df.twitter.data.timeline$created)
tbl_df(myDate)

#replace string or integer creating a new column with the right date format
#Then we will replace the new data was created into the dataframe

df.twitter.data.timeline[["created"]] <- myDate
str(df.twitter.data.timeline)

# We will Check if we have NA in the df.twitter that was loaded,
# this function gave us the name of the column where we got NA values

colnames(df.twitter.data.timeline)[colSums(is.na(df.twitter.data.timeline)) > 0]

# NUMBER OF NA in my DATA, this function gave us the number of Na per column
```

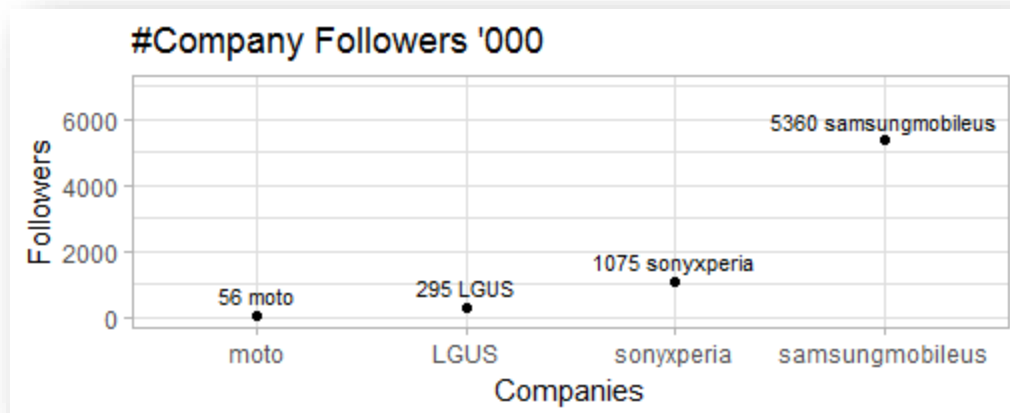
Once we got comfortable with the dataset and after the cleaning up process or preparation of the data, we will proceed to plot the information, using the package ggplot2

```
##### FOLLOWERS DOTS #####

dots.chart.xCompanies.yFollowers <- ggplot(df.twitter, aes(x = reorder(t.account,
+t.followers), y =(df.twitter$t.followers/1000),
```

```
label = paste(round(df.twitter$t.followers/1000, digits = 0), df.twitter$t.account))) +
geom_point() + geom_text(size=3, hjust = +0.5, vjust = -0.6) +
labs( x = "Companies", y = "Followers", title = "#Company Followers '000") +
ylim(0,7000)+
theme_light()
```

```
dots.chart.xCompanies.yFollowers
```



## Conclusions:

1. Most of the data that came from Twitter is strings, so we have to prepare and get rid of multiples characters of values that sometimes are not useful for analysis and plotting.
2. Not always is a good practice to ignore or get rid of NA values, in the cleaning up process in this specific scenario the data frame had NA values in the column replyToSN that corresponded to the exact number of posts of each company,

Having the number of post per company and the text information that a company wrote, allow to build and create statistics, KPI's, and graphic charts to see and understand how a company is using social networks, and understand how is the way that they use this platform.

3. In the analysis process of the timeline data, we could classify and separate the information in brand posts, customer replies, and company retweets.

The brand posts refer to all those that a company wrote, without reply or retweet some else account. This means that a company is promoting their products and trying to engagement customers with special offers or with fidelity programs.

One of the interesting findings was that we could analyze peak points during any posts date range. During this analysis, we found that the peak points in the case of Samsung corresponded to the Samsung S8 released, that was held in April 2017.

“Twitt: Pre-order the Galaxy S8 or S8+ by 4/20, and get the new Gear VR with Controller for free. <https://t.co/toQkBi06VD>”

These findings lead us to think we can build algorithms aiming to predict future company product releases and trends, for instance, most of the companies have certain patterns regarding product releasing.

4. Unfortunately, in the past companies cannot get access to what customers thought about them. But this is not a problem anymore with the mentions file we got from those 4 companies we could analyze how was customers behave and most important thing how they interacted writing own tweets mentioning the company Twitter account. During this process, we identify that we got two main interactions forms from the users: the first one refers to customer engagement with the brand, and the second one refers to customer service platform, in which customers express their unconformities with products or with customer service.