Proyecto 2

Fernando Stein Vallarta 165455 Pablo López Landeros 178863
Manuel García Garduño 162136

Introducción

Durante los últimos 10 años, las redes sociales han crecido a un ritmo descomunal y parece que cada vez toman mayor relevancia en varios aspectos de nuestras vidas. Esta tendencia ha dado pie a estudiar a través de la estadística, todo lo que fluye a través de estas redes, esto con el objetivo de poder transformar los datos en información, la información en conocimiento y el conocimiento en *insights*. Es por esto que ya hoy en día existen gran cantidad de APIs y código "prefabricado" que nos permite facilmente minar los datos que se encuentran en los servidores de Twitter, Facebook y/o Instagram.

Para un primer acercamiento a estas herramientas se plantea este proyecto. Consiste en utilizar los conceptos de muestreo visto en clase y el paquete de R *rtweet* para estudiar la estructura de Twitter. Concretamente, enfocamos el estudio en extraer y analizar el contenido de tweets que involucren o discutan al presidente de los Estados Unidos: **Donald Trump**.

```
#Importamos librerías:
rm(list= ls())

#Importamos librerias

# 0. Importamos Librerias
library(cowplot)
library(dplyr)
library(glue)
library(ggplot2)
library(ggmap)
library(ggcorrplot)
library(jsonlite)
library(kableExtra)
library(knitr)
library(ks)
library(leaflet)
library(lubridate)
library(mapsapi)
library(moments)
library(rtweet)
library(rjson)
library(RCurl)
library(stringr)
library(tidytext)
library(tidyverse)
library(tmaptools)
```

Extracción de los Tweets muestra

```
#Parte 1: Importamos los tweets y los convertimos en una base de datos.
#SegÃºn el link https://github.com/ropensci/rtweet/issues/356 existe un error en la configuraciÃ³n
#de rTweet cuando usamos files de tipo json, luego entonces utilizamos el fix proporcionado en el
#siguiente link: https://gist.github.com/JBGruber/dee4c44e7d38d537426f57ba1e4f84ab

#El fix consiste en la función recover_stream, la cual se construye en las siguientes líneas:

#' Recovers Twitter damaged stream data (JSON file) into parsed data frame.
#'
#' @param path Character, name of JSON file with data collected by
#'   \code{\link{stream_tweets}}.
#' @param dir Character, name of a directory where intermediate files are
#'   stored.
#' @param verbose Logical, should progress be displayed?
#'
#' @family stream tweets
recover_stream <- function(path, dir = NULL, verbose = TRUE) {

  # read file and split to tweets
  lines <- readChar(path, file.info(path)$size, useBytes = TRUE)
  tweets <- stringi::stri_split_fixed(lines, "\n{")[[1]]
  tweets[-1] <- paste0("{", tweets[-1])
  tweets <- tweets[!(tweets == "" | tweets == "{")]

  # remove misbehaving characters
  tweets <- gsub("\r", "", tweets, fixed = TRUE)
  tweets <- gsub("\n", "", tweets, fixed = TRUE)

  # write tweets to disk and try to read them in individually
  if (is.null(dir)) {
    dir <- paste0(tempdir(), "/tweets/")
    dir.create(dir, showWarnings = FALSE)
  }

  if (verbose) {
    pb <- progress::progress_bar$new(
      format = "Processing tweets [:bar] :percent, :eta remaining",
      total = length(tweets), clear = FALSE
    )
    pb$tick(0)
  }

  tweets_l <- lapply(tweets, function(t) {
    pb$tick()
    id <- unlist(stringi::stri_extract_first_regex(t, "(?<=id\":)\\d+(?=,)"))[1]
    f <- paste0(dir, id, ".json")
    writeLines(t, f, useBytes = TRUE)
    out <- tryCatch(rtweet::parse_stream(f),
                    error = function(e) {})
    if ("tbl_df" %in% class(out)) {
      return(out)
```

```
    } else {
      return(id)
    }
  })

  # test which ones failed
  test <- vapply(tweets_l, is.character, FUN.VALUE = logical(1L))
  bad_files <- unlist(tweets_l[test])

  # Let user decide what to do
  if (length(bad_files) > 0) {
    message("There were ", length(bad_files),
            " tweets with problems. Should they be copied to your working directory?")
    sel <- menu(c("no", "yes", "copy a list with status_ids"))
    if (sel == 2) {
      dir.create(paste0(getwd(), "/broken_tweets/"), showWarnings = FALSE)
      file.copy(
        from = paste0(dir, bad_files, ".json"),
        to = paste0(getwd(), "/broken_tweets/", bad_files, ".json")
      )
    } else if (sel == 3) {
      writeLines(bad_files, "broken_tweets.txt")
    }
  }

  # clean up
  unlink(dir, recursive = TRUE)

  # return good tweets
  return(dplyr::bind_rows(tweets_l[!test]))
}

# 1. Minería de Datos
# Después de emplear el fix, comenzamos con la minería de datos/tweets
# Notemos que tomaremos 2 muestras, pues esto nos servirá para realizar captura y recaptura
#Mineria de Datos
file.name1 <- "data_tw.json"
file.name2 <- "data_tw2.json"
time.ellapsed<- 600

#MUESTRA 1
stream_tweets(      "Trump",
                    parse=FALSE,
                    file_name=file.name1,
                    language = "en",
                    timeout = time.ellapsed
)

data_1<-recover_stream(file.name1)
data_1<- as.data.frame(data_1)

#MUESTRA 2
stream_tweets(      "Trump",
```

```
                    parse=FALSE,
                    file_name=file.name2,
                    language = "en",
                    timeout = time.ellapsed
)

data_2<-recover_stream(file.name2)
data_2 <- as.data.frame(data_2)

#Para ambos data.frames nos quedamos con las variables de relevancia
data1<- data_1 %>% select(user_id, status_id, screen_name, text,verified)
data2<-data_2 %>% select(user_id, status_id, screen_name, text, verified)

#Para realizar captura y recaptura juntaremos las dos bases de datos y buscaremos los textos
# únicos para obtener nuestra "k"
r<-rbind(data1,data2)
r.len <- r%>%count()
unicos <- r %>% distinct(text)%>%tally()%>%as.numeric()
repetidos<- r.len - unicos
```

```
data_1 <- as.data.frame(readRDS(file = 'data_1.rds'))
data_2 <- as.data.frame(readRDS(file = 'data_2.rds'))

#Para ambos data.frames nos quedamos con las variables de relevancia
data1<- data_1 %>% select(user_id, status_id, screen_name, text,verified)
data2<-data_2 %>% select(user_id, status_id, screen_name, text, verified)

#Para realizar captura y recaptura juntaremos las dos bases de datos y buscaremos los textos
# únicos para obtener nuestra "k"
r<-rbind(data1,data2)
r.len <- r%>%count()
unicos <- r %>% distinct(text)%>%tally()%>%as.numeric()
repetidos<- r.len - unicos
```

Lectura de los RDS

Estimación del total de cuentas verificadas

En Twitter, una cuenta verificada aparece con una palomita azul a un lado del nombre de usuario. Esto permite a la gente saber que esa cuenta de interés público es auténtica. Por lo tanto, esto no solo es una mejora estética sino una manera de darle un poco más respaldo a lo que se escriba en la red social. Es por eso que nos interesamos en estimar un total de cuentas verificadas que se involucran en hablar sobre Donald Trump.

Para estimar el total y sus intervalos de confianza, necesitamos primero la N. La estimaremos adaptando el método de captura recaptura propuesto por Lohr en el capítulo 13 del libro *Sampling: Design and Analysis*. Utilizaremos la fórmula:

$$\hat{N} = \frac{n_1 n_2}{m}$$

Para adaptar el método, capturamos tweets durante 5 mins dos veces. Obteniendo así n_1 com el tamaño de la primer captura. n_2 el tamaño de la segunda captura y m como la cantidad de usuarios que aparecían en ambas capturas.

Una vez teniendo un estimador de N. utilizaremos el estimador para el total visto en clase:

$$\hat{t} = N\bar{x_s} = N\frac{1}{n_1}\sum_{i=1}^{n_1} x_i$$

Con intervalos de confianza:

$$\hat{t} \pm Z_{1-\frac{\alpha}{2}}\sqrt{\hat{V(t)}}$$

donde:

$$\hat{V(t)} = N^2\frac{1-\frac{n_1}{N}}{n_1}S^2_{x,s}$$

y a su vez:

$$S^2_{x,s} = \frac{1}{n_1-1}\sum_{k=1}^{n_1}(x_k - \bar{x_s})^2$$

Valor esperado y Varianza del estimador:

$$E[\hat{t}] = E[\sum_{k=1}^{N}\frac{x_k}{\pi_k}\mathbb{I}_S] = \sum_{k=1}^{N}\frac{x_k}{\pi_k}E[\mathbb{I}_S(x_k)] = \sum_{k=1}^{N}\frac{x_k}{\pi_k}\pi_k = t$$

Es decir, nuestro estimador $\hat{t}$ es insesgado. Obtengamos entonces un estimador del total con su respectivo intervalo de confianza al 99%:

```
#Obtenemos nuestra N
n1 <- nrow(data1)
n2 <- nrow(data2)
m <- as.numeric(repetidos)
N <- (n1 * n2)/m #Utilizando la formula de captura y recaptura
N <- as.numeric(N) *100 #Notamos que es el 1% de la muestra

# 2. Buscamos encontrar el número de cuentas verificadas que hablan acerca de Trump

verified <- data_1 %>% select(user_id,verified)
trus <- length (verified$verified[verified$verified==TRUE])

cols <- sapply(verified, is.logical)
verified[,cols] <- lapply(verified[,cols], as.numeric)
```

```
## Warning in `[<-.data.frame`(`*tmp*`, , cols, value = list(0, 0, 0, 0, 0, :
## provided 282 variables to replace 1 variables
```

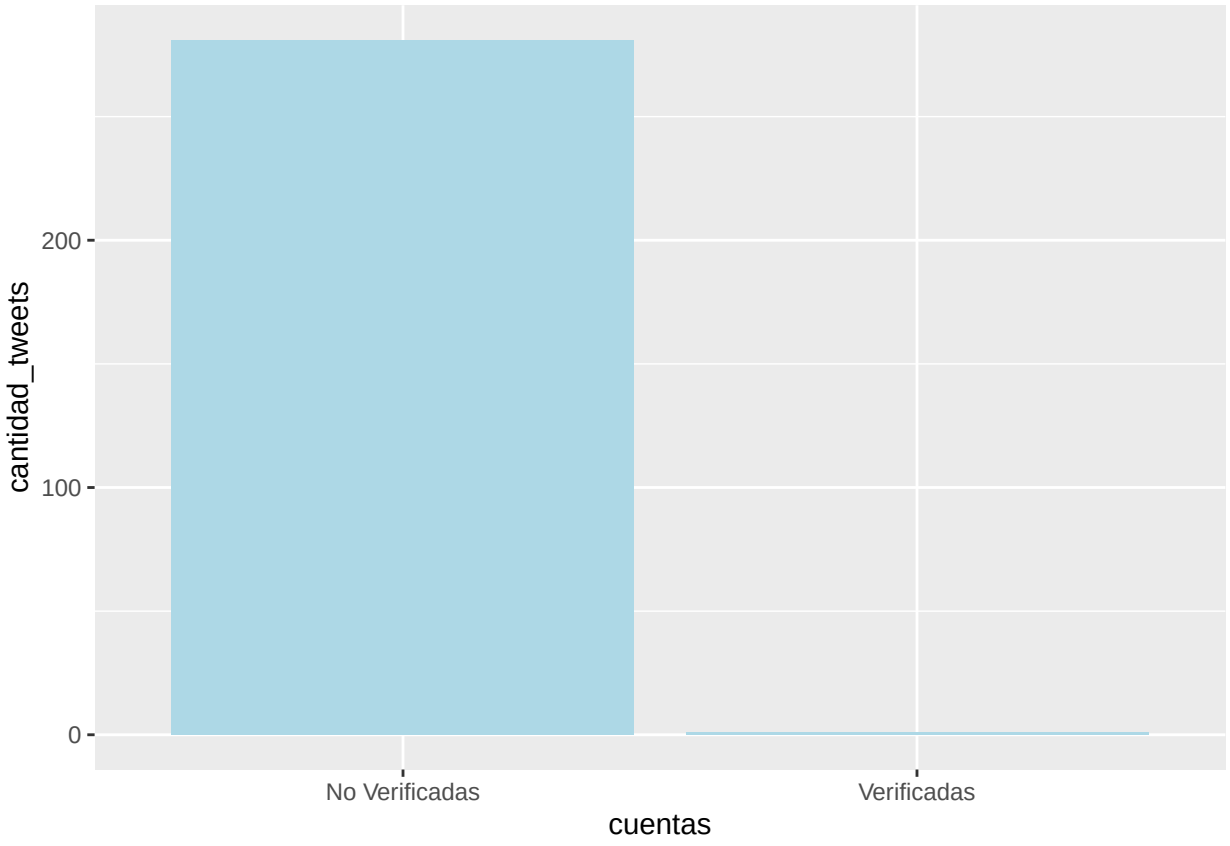
```
x_s <- mean(verified$verified)
t_hat <- N*(1/n1)*(sum(verified$verified))*100
alpha <- .01
z <- qnorm(1-alpha/2)
term <- rep(0,n1)
#estimar sx
for (i in 1:n1) {

  term[i]=(verified$verified[i]-x_s)^2
}

s_x <- ((1/(n1-1))*(sum(term))*100)
```

```
V_hat <- N^2 * ((1-n1/N)/n1 )* s_x

#Intervalo de confianza al 99% del total de cuentas verificadas que hablan de Trump

IC_ver <- c(t_hat-z*sqrt(V_hat),t_hat + z*sqrt(V_hat))
ciLow <- round(IC_ver[1])
ciHigh <- round(IC_ver[2])
paste0("Se estiman ", round(t_hat) ," cuentas verificadas tweeteando sobre Trump con intervalo de confi
       ciLow, ",", ciHigh,"] del 99%")
```

```
## [1] "Se estiman 0 cuentas verificadas tweeteando sobre Trump con intervalo de confianza [0,0] del 99
```

Para darnos una idea de la proporción de cuentas verificadas, hagamos una gráfica de barras que muestre el total de cuentas verificadas y no verificadas en la muestra.

```
#Realizamos una gráfica
df <- data.frame(cuentas=c("Verificadas", "No Verificadas"),
                 cantidad_tweets=c(trus,n1-trus))
p<-ggplot(data=df, aes(x=cuentas, y=cantidad_tweets)) +
  geom_bar(stat="identity",fill="lightblue")
p
```



Estimar la diferencia promedio entre seguidos y seguidores de los usarios que hablan acerca de Trump

(i.e. Followers - Following). La solución a este problema se reduce a la estimación de una media poblacional, puesto que cada uno de los usuarios de twitter representa un número, a saber, el cantidad dada por $Followers - Following$.
Como hemos visto en clase, la media de una muestra $\mathcal{S}$ es un estimador insesgado de la media poblacional puesto que

$$\mathbb{E}[\bar{X}] = \mathbb{E}[\frac{1}{n}\sum_{i=1}^{N} x_i \mathbb{I}_{\mathcal{S}}(x_i)] = \frac{1}{n}\sum_{i=1}^{N} \mathbb{E}[x_i \mathbb{I}_{\mathcal{S}}(xi)] = \frac{1}{n}\sum_{i=1}^{N} x_i \frac{n}{N} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

A partir de este resultado y utilizando la modificación del Teorema Central del Límite para muestras aleatorias finitas podemos construir un intervalo de confianza a nivel 99% a partir del intervalo aleatorio dado por:

$$\bar{X}_{\mathcal{S}} \pm z_{\alpha/2}\sqrt{Var(\bar{X}_{\mathcal{S}})}$$

Bajo este marco teórico, los pasos a seguir para resolver el problema son los siguientes:

- Obtener una muestra aleatoria de usuarios de twitter
  - Usar la función `stream_tweets()`para obtener una muestra aleatoria de tweets
  - Obtener el id del usuario que hizo el tweet medinate la función `users_data()`
- Para cada uno de los usuarios de la muestra, obtener la diferencia $Followers - Following$
  - Creamos una base de datos que contenga el numero de usuario, número de seguidores, número de seguidos y el valor de la diferencia.
- Calculamos la media de los datos que obtuvimos en la muestra

*Calculamos el intervalo de confianza
A continucación se muestra el código:

```
# 3. Estimar la diferencia promedio entre seguidos y seguidores de los usarios que hablan acerca de Tru
#(i.e. Followers - Following).
muestra.usuarios  <- users_data(data_1)

#Usamos una base de datos más sencilla que nos sirva
user_data <- data.frame("user_id" = muestra.usuarios$user_id,
                        "Followers" = muestra.usuarios$followers_count,
                        "Following" = muestra.usuarios$friends_count,
                        "Diferencia" = muestra.usuarios$followers_count-muestra.usuarios$friends_count)
```

Sea Theta el promedio de la cantidad Followers-Following de todos los nodos en el universo de twitter. Entonces el estimador de máxima verosimilitud, insesgado y consistente del parámetro Theta está dado por el promedio de las observaciones en nuestra muestra.

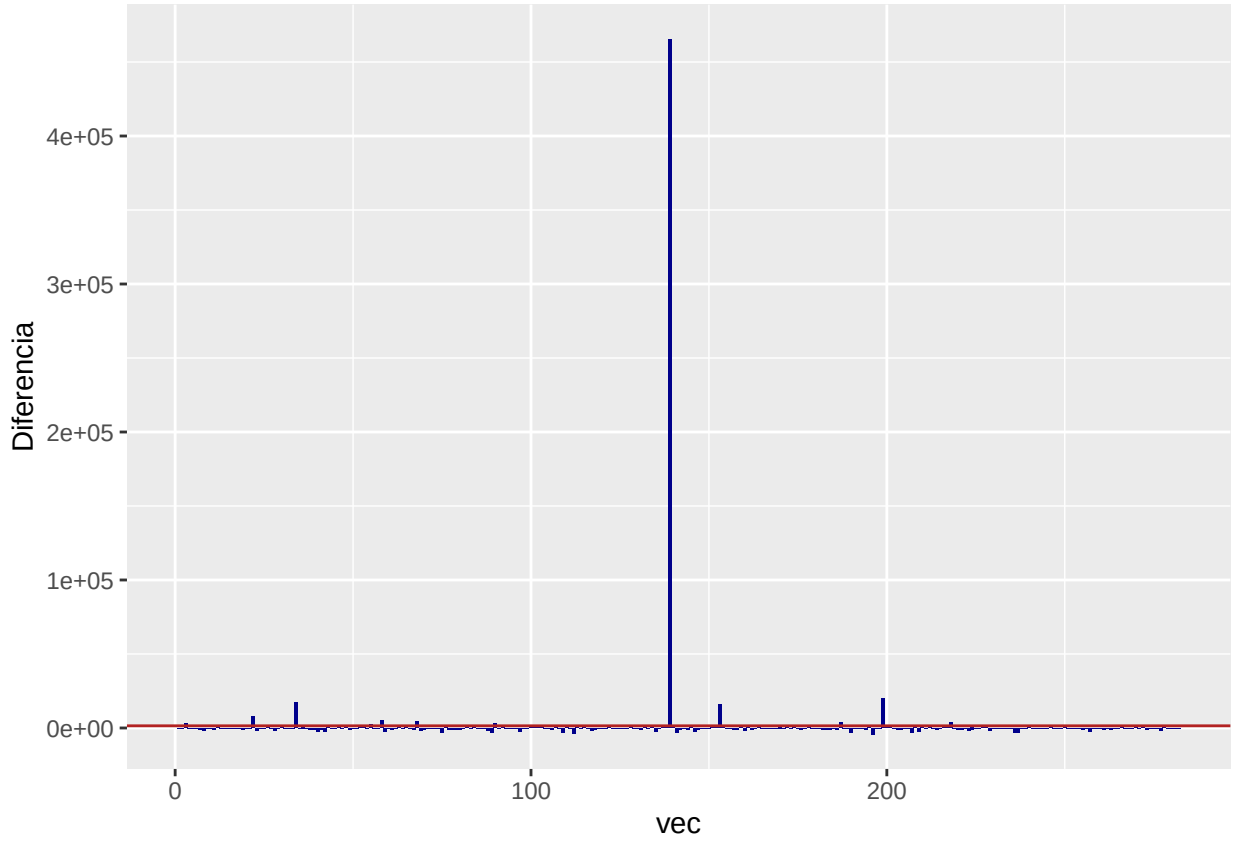
```
x.barra <- mean(user_data$Diferencia)
```

Partimos del estimador puntual que previamente hemos definido para crear un intervalo de confianza al nivel 99% para el parámetro Theta:

```
n            <- length(user_data$Diferencia) #Tamaño de la muestra
N            <- 330000000                    #Cantidad de usuarios activos en twitter en el último mes
                                             #Fuente https://www.statista.com/statistics/282087/number-of-monti
a            <- qnorm(1-alpha/2)
desv.xbarra  <- sqrt((1-n/N)*var(user_data$Diferencia)/n)
```

Realizamos un esquema para visualizar bien los datos:

```
n   <- length(user_data$Diferencia)
vec <- seq(1:n)
ggplot(user_data, aes(x = vec, y = Diferencia))+
  geom_bar(stat = 'identity', fill = 'blue4')+
  geom_hline(yintercept = x.barra, color = 'firebrick')
```



Ahora tenemos los elementos suficientes para construir nuestro intervalo de confianza al nivel 99%:

```
limite.inferior <- x.barra - a*desv.xbarra
limite.superior <- x.barra + a*desv.xbarra
paste0( "IC: [",
       limite.inferior, ",", limite.superior,"]")
```

```
## [1] "IC: [-2811.43793140724,5718.13296686823]"
```

Estimación de la longitud promedio de un post

Una de las preguntas asignadas se refiere a el tamaño promedio de los tweets. De donde en primera instancia debemos de obtener la longitud de los tweets.

```
len<-nchar(data1$text, type = "chars", allowNA = FALSE, keepNA = NA)
data1$len<-len
```

Ahora bien, nosotros tomamos una muestra de tweets utilizando el paquete de rtweet. Sin embargo, nos interesa conocer el tamaño promedio de tweet de la población. De ahí que necesitamos $\bar{x}_U$, la media poblacional. Se propone la media muestral como estimador de la muestra poblacional.

Se demostró que la media muestral es un estimador insesgado de la media poblacional.

Ahora bien, si quisieramos un intervalo de confianza de la media debemos notar que no conocemos σ^2 o la varianza poblacional.

De la teoría de probabilidad sabemos que si $Z \sim N(0,1)$ and $W \sim \chi^2(s)$, entonces la variable $Y = \frac{Z}{\sqrt{W/\nu}}$ sigue una distribución t con s grados de libertad.

Luego entonces definimos a Z = $\frac{\sqrt{n}(\bar{x_s} - \bar{x_u})}{\sigma}$ la cual se distribuye normal estándar. De igual forma, definimos $W = \frac{(n-1)S^2}{\sigma^2}$ sigue una distribución $\chi^2(s)$ con $n-1$ grados de libertad. De ahí que $Y = \frac{\bar{x}_S - \bar{x}_U}{\frac{S}{\sqrt{n}}}$ sigue una distribución t con $n-1$ grados de libertad.

De ahí que un intervalo de confianza para $\bar{x}_U$ esta dado por $\bar{x}_S \pm t_{(n-1,1-\frac{\alpha}{2})}\frac{\sqrt{n}}{S}$

Utilizando lo anteriormente encontrado, tenemos los siguientes resultados al insertar el código en R.

```
len.bar<- mean(data1$len)
n<-length(data1$len)
s<-sd(len)
```

Con su respectivo intervalo de confianza dado por:

```
talpha<-qt(1-alpha/2,n-1)
ic <- round(c(len.bar - talpha*s/sqrt(n), len.bar + talpha*s/sqrt(n)), 2)
paste0("IC: [",
       ic[1], ",", ic[2],"]")
```

```
## [1] "IC: [149.6,176.48]"
```

Sentiment Analysis

Para el siguiente punto, buscamos explicar las características relacionadas con los tweets que se escriben acerca de Donald Trump. En concreto, queremos saber si los posts que se escriben hablan de forma positiva, neutral o negativa acerca de Donald Trump. Para esto necesitamos realizar algo llamado "Sentiment Analysis" o Análisis de Sentimientos en español. El análisis de sentimientos es una técnica de aprendizaje de maquina que detecta la polaridad dentro de un texto, es decir, trata de describir las emociones que el texto presenta. Para esto, utilizaremos los paquetes tidyverse y tidytext junto con el diccionario "bing" realizado por Bing Liu que clasifica las palabras en positivas y negativas.

```
dropwords <- c("trump") # La palabra "Trump" tiene una conotación en el lexicon

sentiment<-data1 %>%
  unnest_tokens(word, text, token = 'words') %>%
  left_join(get_sentiments("bing") %>% filter(!word %in% dropwords)) %>% # Obtiene las palabras de sent
  mutate(sentiment = if_else(sentiment == 'positive', 1, -1, missing = 0)) %>% # Asigna valores a las p
  group_by(user_id) %>% #Agrupamos las palabras por clave
  summarise(sentiment = sum(sentiment))
```

```
## Joining, by = "word"
```

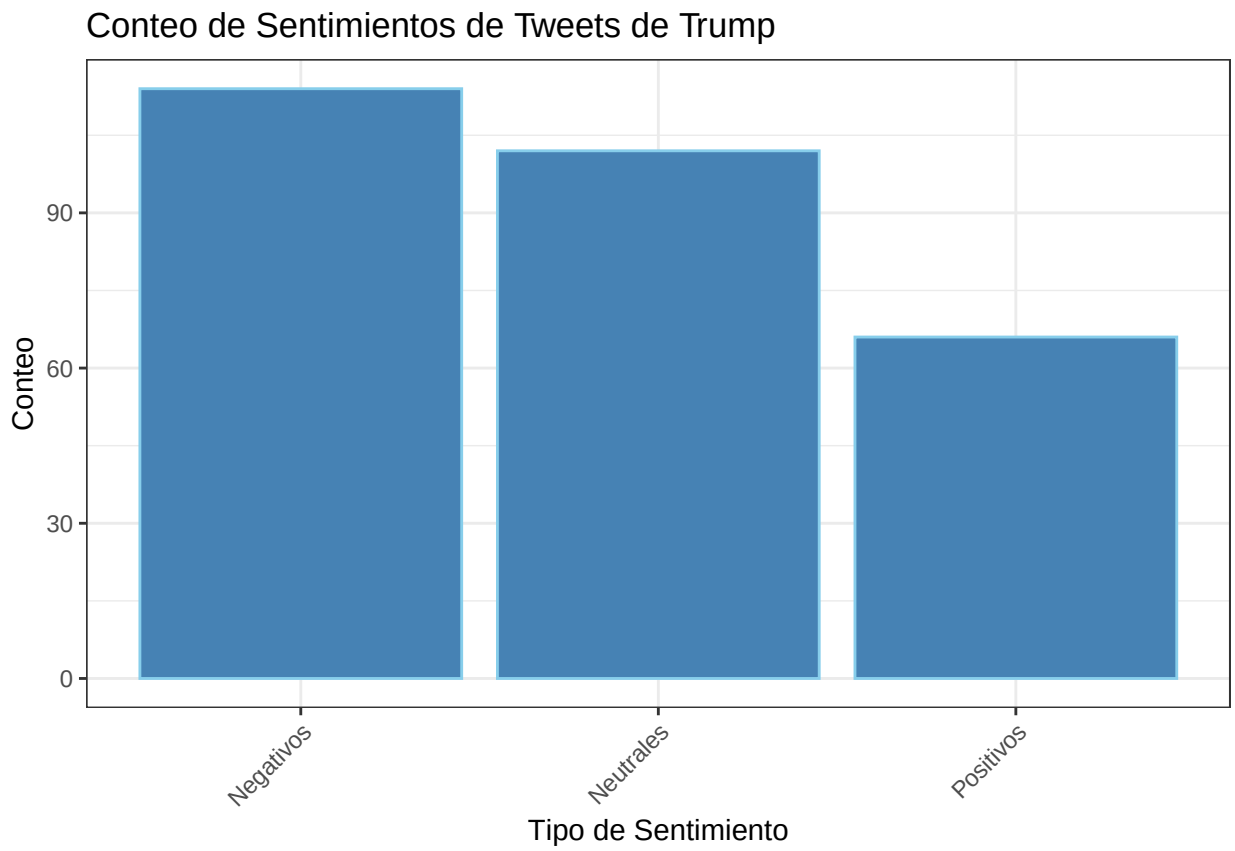
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#Agrupamos por tipo de sentimiento del tweet

n.positivo<-subset(sentiment,sentiment>0)%>%count()
n.positivo$class<-c("Positivos")
n.neutral<-subset(sentiment,sentiment==0)%>%count()
n.neutral$class<-c("Neutrales")
n.negativo<-subset(sentiment,sentiment<0)%>%count()
```

```r
n.negativo$class<-c("Negativos")

n<- as.numeric(n.positivo$n + n.neutral$n + n.negativo$n)

count.sentiment<-rbind(n.positivo,n.neutral,n.negativo)

#Realizamos una tabla y un plot
kable(count.sentiment, booktabs = T) %>% kable_styling(latex_options = "striped")
```

n	class
66	Positivos
102	Neutrales
114	Negativos

```r
ggplot(count.sentiment)+
  geom_col(aes(x = class, y = n), color='skyblue', fill='steelblue') +
  theme_bw() +
  labs(
    title = "Conteo de Sentimientos de Tweets de Trump",
    x = "Tipo de Sentimiento",
    y = "Conteo"
  ) +
  theme(axis.text.x=element_text(angle=45, hjust=1))
```



Utilizando lo descrito anteriormente obtenemos los siguientes resultados:

Ahora bien, notemos que únicamente tenemos una muestra. Pero notemos que se trata de una muestra estratificada, donde cada estrato es la postura de cada tweet. Luego entonces, utilizamos la N obtenida previamente mediante captura y recaptura. Ahora bien, para obtener el total poblacional utilizamos la fórmula obtenida en clases dada por $n_h = n\frac{N_h}{N}$. Aquí buscamos N_h, de ahí que realizamos el despeje dado por: $N_h = N\frac{n_h}{n}$. Con lo anterior obtenemos los siguientes resultados.

```
N.positivo <- as.numeric(n.positivo$n)/n * N
N.negativo <- as.numeric(n.negativo$n)/n * N
N.neutral  <- as.numeric(n.neutral$n)/n * N
```

Estimar seguidores de Trump fuera de EU

En este apartado se pretende conocer la reelevancia cultural de Donald Trump en el extranjero. Entiéndase por extranjero, por supuesto, a la población que reside fuera de Estados Unidos. La hipótesis es simple: tanta más reelevancia cultural tendrá Donald Trump, cuanto mayor sea la proporción de usuarios extranjeros que lo siguen. Así pues, nuestra tarea se traduce en la construcción de un intervalo de confianza para la proporción p de seguidores de Donald Trump que residen fuera de los Estados Unidos.
El problema de estimar una proporción dada una población finita ha sido abordado en el salón de clases, con lo cual, a continuación se resumen los aspectos más importantes del estimador que hemos elegiddo para la proporción p. Dada una población finita U de N elementos y un subconjunto A de esa población, si lo que pretendemos es estimar la proporción $p = \frac{|A|}{|U|}$ entonces un buen estimador dada una muestra de tamaño n, en el sentido de que es insesgado y consistente, está dado por:

$$\hat{p} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_A(x_i)$$

El insesgamiento se sigue del hecho de que:

$$\mathbb{E}[\hat{p}] = \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_A(x_i)] = \frac{1}{n}\mathbb{E}[\sum_{i=1}^{n}\mathbb{I}_A(x_i)] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\mathbb{I}_A(x_i)] = \frac{1}{n}np = p$$

Además, la varianza del estimador está dada por:

$$Var(\hat{p}) = \frac{p(1-p)}{n}\frac{N-n}{N-1}$$

La consistencia del estimador es evidente, en tanto que su error cuadrático medio coincide, en este caso, con su varianza. Así pues, tomamos el límite cuando el tamaño de nuestra muestra tiende al tamaño de nuestra población para probar la consistencia:

$$\lim_{n\to N} Var(\hat{p}) = 0$$

Es claro que si aquello que pretendemos estimar es el valor de p, no podremos calcular el valor real de la varianza del estimador $\hat{p}$. Este hecho es crucial en virtud de que necesitaremos conocer el valor de la varianza para poder construir un intervalo de confianza. Una forma de arreglar este problema es, por muy repetitivo que suene, estimar la varianza del estimador de la siguiente forma:

$$\widehat{Var(\hat{p})} = \frac{\hat{p}(1-\hat{p})}{n}\frac{N-n}{N-1}$$

De esta forma, si aplicamos el Teorema Central del Límite para construir un intervalo de confianza nivel $(1-\alpha)$x100% para la proporción p obtenemos el siguientes limites de nuestro intervalo aleatorio:

$$\hat{p} \pm z_{\alpha/2}\sqrt{\widehat{Var(\hat{p})}}$$

Este intervalo aleatorio captura el verdadero valor del parámetro p con probabilidad $1-\alpha$. Siguiendo este esquema teórico, ahora podemos aplicarlo para responder a la pregunta que nos atañe en esta sección, a saber, ¿qué proporción de seguidores de Donald Trump reside fuera de los Estados Unidos? A continucación se presenta el código utilizado. Obtenemos una muestra aleatoria de 5,000 followers de Trump:

```
#
muestra_aux       <- get_followers("realDonaldTrump", n = 5000)
muestra_aux1      <- lookup_users(muestra_aux$user_id)
muestra_followers <- users_data(muestra_aux1)
saveRDS(muestra_followers,file = "muestraTrump.rds")
```

```
muestra_followers <- as.data.frame(readRDS('muestraTrump.rds'))

#Eliminamos de nuestra base de datos aquellas entradas que no tienen especificada
#una ubicación
muestra_followers <- muestra_followers%>%filter(location != "")

#Obtenemos el tamaño real de nuestra muestra
n_muestra         <- length(muestra_followers$user_id)

#Ahora nos quedamos con los usuarios que tweetean desde Estados Unidos
muestra_followers <- muestra_followers%>%filter(str_detect(location, "USA")|
                                                  str_detect(location, "United States")|
                                                  str_detect(location, ", AL")|
                                                  str_detect(location, ", AK")|
                                                  str_detect(location, ", AZ")|
                                                  str_detect(location, ", AR")|
                                                  str_detect(location, ", CA")|
                                                  str_detect(location, ", CO")|
                                                  str_detect(location, ", CT")|
                                                  str_detect(location, ", DE")|
                                                  str_detect(location, ", FL")|
                                                  str_detect(location, ", GA")|
                                                  str_detect(location, ", HI")|
                                                  str_detect(location, ", ID")|
                                                  str_detect(location, ", IL")|
                                                  str_detect(location, ", IN")|
                                                  str_detect(location, ", IA")|
                                                  str_detect(location, ", KS")|
                                                  str_detect(location, ", KY")|
                                                  str_detect(location, ", LA")|
                                                  str_detect(location, ", ME")|
                                                  str_detect(location, ", MD")|
                                                  str_detect(location, ", MA")|
                                                  str_detect(location, ", MI")|
                                                  str_detect(location, ", MN")|
                                                  str_detect(location, ", MS")|
                                                  str_detect(location, ", MO")|
                                                  str_detect(location, ", MT")|
                                                  str_detect(location, ", NE")|
                                                  str_detect(location, ", NV")|
                                                  str_detect(location, ", NH")|
                                                  str_detect(location, ", NJ")|
                                                  str_detect(location, ", NM")|
                                                  str_detect(location, ", NY")|
```

```
                                    str_detect(location, ", NC")|
                                    str_detect(location, ", ND")|
                                    str_detect(location, ", OH")|
                                    str_detect(location, ", OK")|
                                    str_detect(location, ", OR")|
                                    str_detect(location, ", PA")|
                                    str_detect(location, ", RI")|
                                    str_detect(location, ", SC")|
                                    str_detect(location, ", SD")|
                                    str_detect(location, ", TN")|
                                    str_detect(location, ", TX")|
                                    str_detect(location, ", UT")|
                                    str_detect(location, ", VT")|
                                    str_detect(location, ", VA")|
                                    str_detect(location, ", WA")|
                                    str_detect(location, ", WV")|
                                    str_detect(location, ", WI")|
                                    str_detect(location, ", WY")|
                                    str_detect(location, ", DC")|
                                    str_detect(location, "Alabama")|
                                    str_detect(location, "Alaska")|
                                    str_detect(location, "Arizona")|
                                    str_detect(location, "Arkansas")|
                                    str_detect(location, "California")|
                                    str_detect(location, "Colorado")|
                                    str_detect(location, "Connecticut")|
                                    str_detect(location, "Delaware")|
                                    str_detect(location, "Florida")|
                                    str_detect(location, "Georgia")|
                                    str_detect(location, "Hawaii")|
                                    str_detect(location, "Idaho")|
                                    str_detect(location, "Illinois")|
                                    str_detect(location, "Indiana")|
                                    str_detect(location, "Iowa")|
                                    str_detect(location, "Kansas")|
                                    str_detect(location, "Kentucky")|
                                    str_detect(location, "Louisiana")|
                                    str_detect(location, "Maine")|
                                    str_detect(location, "Maryland")|
                                    str_detect(location, "Massachusetts")|
                                    str_detect(location, "Michigan")|
                                    str_detect(location, "Minnesota")|
                                    str_detect(location, "Mississippi")|
                                    str_detect(location, "Missouri")|
                                    str_detect(location, "Montana")|
                                    str_detect(location, "Nebraska")|
                                    str_detect(location, "Nevada")|
                                    str_detect(location, "New Hampshire")|
                                    str_detect(location, "New Jersey")|
                                    str_detect(location, "New Mexico")|
                                    str_detect(location, "New York")|
                                    str_detect(location, "North Carolina")|
                                    str_detect(location, "North Dakota")|
```

```
                                                    str_detect(location, "Ohio")|
                                                    str_detect(location, "Oklahoma")|
                                                    str_detect(location, "Oregon")|
                                                    str_detect(location, "Pennsylvania")|
                                                    str_detect(location, "Rhode Island")|
                                                    str_detect(location, "South Carolina")|
                                                    str_detect(location, "South Dakota")|
                                                    str_detect(location, "Tennessee")|
                                                    str_detect(location, "Texas")|
                                                    str_detect(location, "Utah")|
                                                    str_detect(location, "Vermont")|
                                                    str_detect(location, "Virginia")|
                                                    str_detect(location, "Washington")|
                                                    str_detect(location, "West Virginia")|
                                                    str_detect(location, "Wisconsin")|
                                                    str_detect(location, "Wyoming")
)

#Obtenemos la cantidad de usuarios de nuestra muestra que no tweetean desde Estados Unidos
n_extranjeros <- n_muestra-length(muestra_followers$user_id)


#Estimamos la proporción de de tweets extranjeros
estim         <- n_extranjeros/n_muestra

#A partir de esta estimación puntual de la proporción de followers extranjeros de Trump
#crearemos un intervalo de confianza a nivel 99%
N                 <- 83650000 #Numero total de followers de Trump
alpha             <- 0.01
Z                 <- qnorm(1 - alpha/2)
varianza.gorro    <- estim*(1-estim)*(N - n_muestra)/n_muestra/(N-1)
limite.inferior   <- estim - Z*sqrt(varianza.gorro)
limite.superior   <- estim + Z*sqrt(varianza.gorro)
paste0("IC: [",
       limite.inferior, ",", limite.superior,"]")
```

```
## [1] "IC: [0.656980980887364,0.727770792162282]"
```

Conclusión

La opinión de la gente es algo sumamente relevante en algunos temas y simplemente es difícil de ignorar. A través del análisis de sentimiento y la minería de datos en redes se puede construir una idea bastante fidedigna de la percepción que está teniendo la gente sobre una marca, producto, negocio, tema polémico o persona. Esto no es poca cosa ya que el tener esta posibilidad, podemos utilizar la información para tomar mejores decisiones sobre estrategias políticas o de negocio.

Referencias

1. Repositorio de JBGruber ** recover_stream.R**. Link: https://gist.github.com/JBGruber/dee4c44e7d38d537426f57ba1e4f84ab
2. Lohr, S. L. (2019). Sampling design and analysis. Boca Raton, FL: CRC Press.