Twitter

Francisco Pérez Hernández. Cristina Zuheros Montes, .

7/4/2017

Variable para indicar el número de tweets a sacar

```
total_tweets_a_sacar = 10000
```

Crear Credenciales

Lo primero será ir al siguiente enlace https://apps.twitter.com y registrarnos para obtener nuestras credenciales quedando un fichero llamado "credenciales.R" con la siguiente estructura:

```
#Cargamos las librerías
library("ROAuth")
library("base64enc");
library("twitteR");
library("streamR");

#Cargar parámetros de configuración
reqURL <- "https://api.twitter.com/oauth/request_token"
accessURL <- "https://api.twitter.com/oauth/access_token"
authURL <- "https://api.twitter.com/oauth/authorize"
options(httr_oauth_cache=T)

#Cargar las credenciales obtenidas del paso anterior
consumer_key <- "pegar aquÃ? la credencial"
consumer_secret <-"pegar aquÃ? la credencial"
access_token <-"pegar aquÃ? la credencial"
access_secret <-"pegar aquÃ? la credencial"

#Ejecutar la autenticación de TwitteR
setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)

#streamR authentication
credentials_file <- "my_oauth.Rdata"
if (file.exists(credentials_file)){
  load(credentials_file)
} else {
  cred <- OAuthFactory$new(consumerKey = consumer_key, consumerSecret =
                             consumer_secret, requestURL = reqURL, accessURL = accessURL, authURL = aut
  cred$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
  save(cred, file = credentials_file)
}
```

Obtener datos de twitter

```r
# Cargar la librería específica de TwitterR
library(twitteR);

# Leer el fichero de credenciales creado anteriormente, Â¡cuidado con la ruta del fichero!.
source('credenciales.R')
```

```
## Loading required package: RCurl
```

```
## Loading required package: bitops
```

```
## Loading required package: rjson
```

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

```r
# FunciÃ³n que permite buscar: #hastag, @usuarios, palabras
tweets <- searchTwitter("#brexit", n=100, lang="en")

# Quedarse solo con el primer tweet para datos concretos del mismo
tweet <- tweets[[1]];
# Mostrar la estructura del tweet
#str(tweet)
# Obtener el texto del tweet:
tweet$getText()
```

```
## [1] "RT @Paul1Singh: Labours position on #brexit is almost identical to that of the Tories. #Generall
```

```r
# Obtener información acerca del usuario:
usuario <- getUser(tweet$getScreenName());
# Mostrar la estructura del usuario
#str(usuario)
# Obtener el nombre del usuario
usuario$getName()
```

```
## [1] "Bradders"
```

Instalación de paquetes necesarios

```r
# Instalar el paquete Sentiment
require('pacman')
```

```
## Loading required package: pacman
```

```r
#if (!require('pacman')) install.packages('pacman&')
#pacman::p_load(devtools, installr)
#install.Rtools()
#install_url('http://cran.r-project.org/src/contrib/Archive/Rstem/Rstem_0.4-1.tar.gz')
#install_url('http://cran.r-project.org/src/contrib/Archive/sentiment/sentiment_0.2.tar.gz')
if (!require('pacman')) install.packages('pacman')
pacman::p_load(twitteR, sentiment, plyr, ggplot2, wordcloud, RColorBrewer, httpuv, RCurl, base64enc)
options(RCurlOptions = list(cainfo = system.file('CurlSSL', 'cacert.pem', package = 'RCurl')))
#setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)
#setup_twitter_oauth(api_key,api_secret)
```

AnÃ¡lisis de sentimientos

Sacar tweets

Lo primero que vamos a hacer será sacar twetts sobre el brexit y sacar de ellos su texto

```
tweets <- searchTwitter("#brexit", n=total_tweets_a_sacar, lang="en")
texto_tweets = sapply(tweets, function(x) x$getText())
```

Limpiado del texto

Vamos a ver un ejemplo de los primeros tweets encontrados de como vamos limpiando el texto

```
head(texto_tweets)
```

```
## [1] "RT @Paul1Singh: Labours position on #brexit is almost identical to that of the Tories. #Generall
## [2] "When does @theresa_may  even have time to organise\\campaign a #GE2017? Shouldn<U+0092>t she be
## [3] "RT @beingrichard: With only #Murdoch's money to go on, @thetimes assumes May victory. But #Brexi
## [4] "RT @projectremain: 8 Labour MPs Who've Quit Parliament After General Election Was Called, brexit
## [5] "Retweeted John Van Reenen (@johnvanreenen):\n\nGreat takedown of Minford's #Brexit economic delu
## [6] "#BREXIT is the beginning of getting back to the core of english-speaking commonwealth including
```

```
cat("\nEliminamos retweet\n")
```

```
##
## Eliminamos retweet
```

```
texto_tweets = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', texto_tweets)
head(texto_tweets)
```

```
## [1] ": Labours position on #brexit is almost identical to that of the Tories. #GeneralElection2017 ht
## [2] "When does @theresa_may  even have time to organise\\campaign a #GE2017? Shouldn<U+0092>t she be
## [3] ": With only #Murdoch's money to go on, @thetimes assumes May victory. But #Brexit voters trust #
## [4] ": 8 Labour MPs Who've Quit Parliament After General Election Was Called, brexit supporter Stuart
## [5] "Retweeted John Van Reenen (@johnvanreenen):\n\nGreat takedown of Minford's #Brexit economic delu
## [6] "#BREXIT is the beginning of getting back to the core of english-speaking commonwealth including
```

```
cat("\nEliminar usuarios\n")
```

```
##
## Eliminar usuarios
```

```
texto_tweets = gsub('@\\w+', '', texto_tweets)
head(texto_tweets)
```

```
## [1] ": Labours position on #brexit is almost identical to that of the Tories. #GeneralElection2017 ht
## [2] "When does   even have time to organise\\campaign a #GE2017? Shouldn<U+0092>t she be focussing on
## [3] ": With only #Murdoch's money to go on,  assumes May victory. But #Brexit voters trust #Corbyn to
## [4] ": 8 Labour MPs Who've Quit Parliament After General Election Was Called, brexit supporter Stuart
## [5] "Retweeted John Van Reenen ():\n\nGreat takedown of Minford's #Brexit economic delusions... https
## [6] "#BREXIT is the beginning of getting back to the core of english-speaking commonwealth including
```

```
cat("\nEliminamos puntuaciÃ³n\n")
```

```
##
## Eliminamos puntuaciÃ³n
```

```
texto_tweets = gsub('[[:punct:]]', '', texto_tweets)
head(texto_tweets)
```

```
## [1] " Labours position on brexit is almost identical to that of the Tories GeneralElection2017 https
## [2] "When does   even have time to organisecampaign a GE2017 Shouldn<U+0092>t she be focussing on br
## [3] " With only Murdochs money to go on  assumes May victory But Brexit voters trust Corbyn too amp
## [4] " 8 Labour MPs Whove Quit Parliament After General Election Was Called brexit supporter Stuart a
## [5] "Retweeted John Van Reenen \n\nGreat takedown of Minfords Brexit economic delusions httpstcoMTkT
## [6] "BREXIT is the beginning of getting back to the core of englishspeaking commonwealth including a
```

```
cat("\nEliminamos nÃºmeros\n")
```

```
##
## Eliminamos nÃºmeros
```

```
texto_tweets = gsub('[[:digit:]]', '', texto_tweets)
head(texto_tweets)
```

```
## [1] " Labours position on brexit is almost identical to that of the Tories GeneralElection httpstcoK
## [2] "When does   even have time to organisecampaign a GE Shouldn<U+0092>t she be focussing on brexit
## [3] " With only Murdochs money to go on  assumes May victory But Brexit voters trust Corbyn too amp
## [4] "  Labour MPs Whove Quit Parliament After General Election Was Called brexit supporter Stuart am
## [5] "Retweeted John Van Reenen \n\nGreat takedown of Minfords Brexit economic delusions httpstcoMTkT
## [6] "BREXIT is the beginning of getting back to the core of englishspeaking commonwealth including a
```

```
cat("\nEliminamos enlaces html\n")
```

```
##
## Eliminamos enlaces html
```

```
texto_tweets = gsub('http\\w+', '', texto_tweets)
head(texto_tweets)
```

```
## [1] " Labours position on brexit is almost identical to that of the Tories GeneralElection "
## [2] "When does   even have time to organisecampaign a GE Shouldn<U+0092>t she be focussing on brexit
## [3] " With only Murdochs money to go on  assumes May victory But Brexit voters trust Corbyn too amp
## [4] "  Labour MPs Whove Quit Parliament After General Election Was Called brexit supporter Stuart am
## [5] "Retweeted John Van Reenen \n\nGreat takedown of Minfords Brexit economic delusions "
## [6] "BREXIT is the beginning of getting back to the core of englishspeaking commonwealth including a
```

```
cat("\nEliminamos espacios innecesarios\n")
```

```
##
## Eliminamos espacios innecesarios
```

```
texto_tweets = gsub('[ \t]{2,}', '', texto_tweets)
texto_tweets = gsub('^\\s+|\\s+$', '', texto_tweets)
head(texto_tweets)
```

```
## [1] "Labours position on brexit is almost identical to that of the Tories GeneralElection"
## [2] "When doeseven have time to organisecampaign a GE Shouldn<U+0092>t she be focussing on brexit Se
## [3] "With only Murdochs money to go onassumes May victory But Brexit voters trust Corbyn too amp may
## [4] "Labour MPs Whove Quit Parliament After General Election Was Called brexit supporter Stuart amon
## [5] "Retweeted John Van Reenen \n\nGreat takedown of Minfords Brexit economic delusions"
## [6] "BREXIT is the beginning of getting back to the core of englishspeaking commonwealth including a
```

```
#Función para eliminar posibles errores al pasar a minúscula
try.error = function(x){
  # creamos un missing value
```

```
  y = NA
  # tryCatch error
  try_error = tryCatch(tolower(x), error=function(e) e)
  # if not un error
  if (!inherits(try_error, 'error'))
    y = tolower(x)
  return(y)
}
cat("\nPasamos a minúscula si no hay error\n")
```

```
##
## Pasamos a minúscula si no hay error
```

```
texto_tweets = sapply(texto_tweets, try.error)
head(texto_tweets)
```

```
##                                  Labours position on brexit is almost identical to that of the Tories
##                                "labours position on brexit is almost identical to that of the tories g
##     When doeseven have time to organisecampaign a GE Shouldn<U+0092>t she be focussing on brexit Seem
##   "when doeseven have time to organisecampaign a ge shouldn<U+0092>t she be focussing on brexit seems
##            With only Murdochs money to go onassumes May victory But Brexit voters trust Corbyn too an
##          "with only murdochs money to go onassumes may victory but brexit voters trust corbyn too amp
##   Labour MPs Whove Quit Parliament After General Election Was Called brexit supporter Stuart among th
## "labour mps whove quit parliament after general election was called brexit supporter stuart among the
##                                    Retweeted John Van Reenen \n\nGreat takedown of Minfords Brexit eco
##                                  "retweeted john van reenen \n\ngreat takedown of minfords brexit econ
##                BREXIT is the beginning of getting back to the core of englishspeaking commonwealth in
##              "brexit is the beginning of getting back to the core of englishspeaking commonwealth inc
```

```
cat("\nEliminamos NAs en el texto\n")
```

```
##
## Eliminamos NAs en el texto
```

```
texto_tweets = texto_tweets[!is.na(texto_tweets)]
names(texto_tweets) = NULL
head(texto_tweets)
```

```
## [1] "labours position on brexit is almost identical to that of the tories generalelection"
## [2] "when doeseven have time to organisecampaign a ge shouldn<U+0092>t she be focussing on brexit see
## [3] "with only murdochs money to go onassumes may victory but brexit voters trust corbyn too amp may
## [4] "labour mps whove quit parliament after general election was called brexit supporter stuart among
## [5] "retweeted john van reenen \n\ngreat takedown of minfords brexit economic delusions"
## [6] "brexit is the beginning of getting back to the core of englishspeaking commonwealth including an
```

Clasificador de sentimientos

Ahora vamos a clasificar por emociones y obtener la mejor de ellas

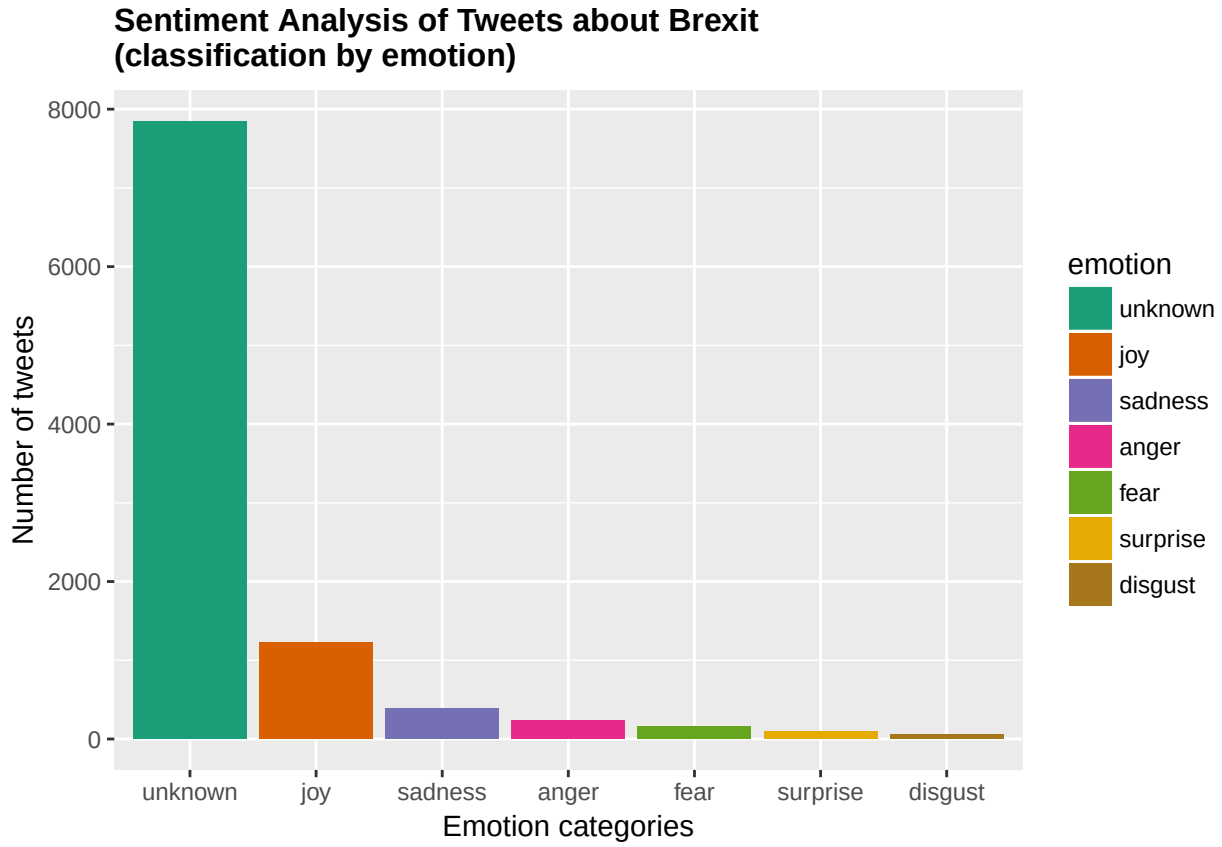
```
clasificacion_emociones = classify_emotion(texto_tweets, algorithm='bayes', prior=1.0)
emociones = clasificacion_emociones[,7]
# sustituimos NA's por 'unknown'
emociones[is.na(emociones)] = 'unknown'
```

Clasificamos por popularidad los tweets y montamos el dataframe

```
clasificacion_popularidad = classify_polarity(texto_tweets, algorithm='bayes')
popularidad = clasificacion_popularidad[,4]
data = data.frame(text=texto_tweets, emotion=emociones, polarity=popularidad, stringsAsFactors=FALSE)
data = within(data, emotion <- factor(emotion, levels=names(sort(table(emotion), decreasing=TRUE))))
```

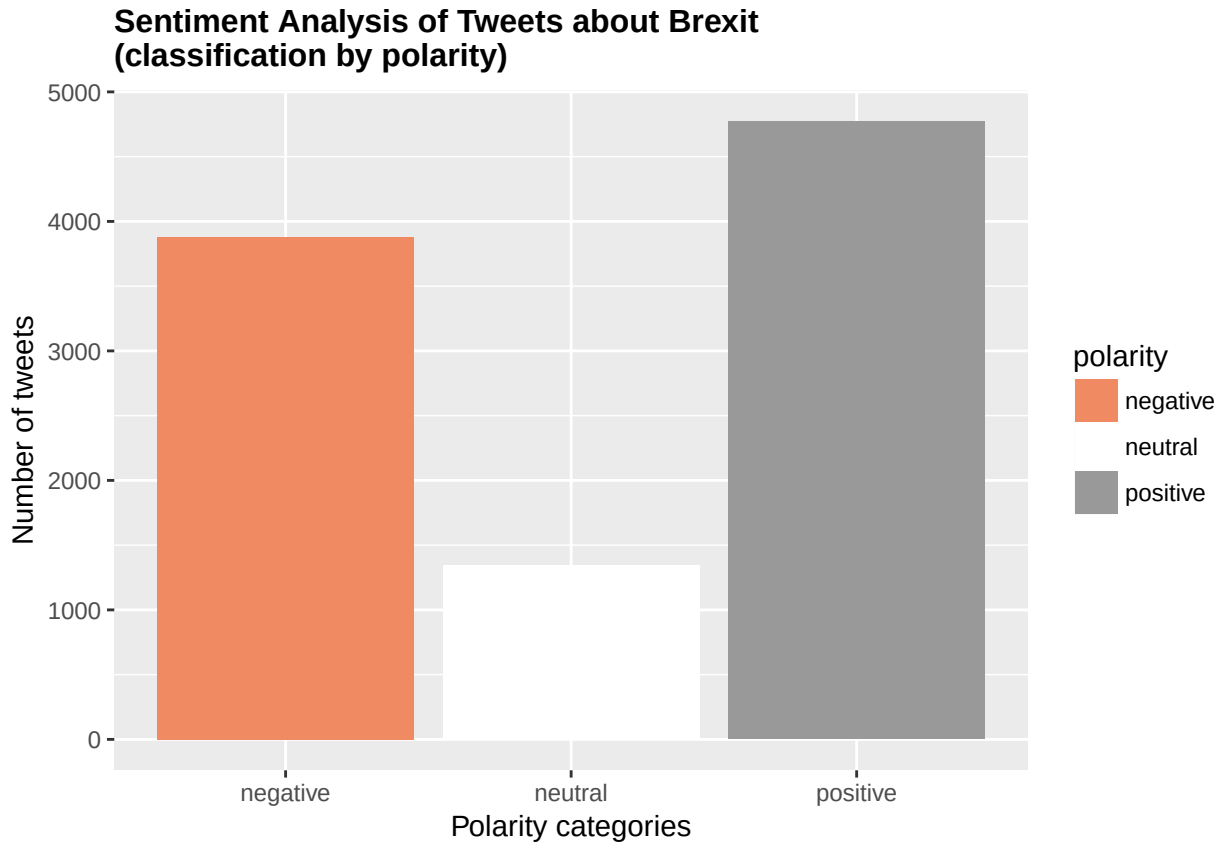
Vamos a mostrar una gráfica en función de la distribución de las emociones sacadas.

```
ggplot(data, aes(x=emotion)) +
geom_bar(aes(y=..count.., fill=emotion)) +
scale_fill_brewer(palette='Dark2') +
labs(x='Emotion categories', y='Number of tweets') +
ggtitle('Sentiment Analysis of Tweets about Brexit\n(classification by emotion)') +
theme(plot.title = element_text(size=12, face='bold'))
```



Vamos a mostrar una gráfica en función de la distribución de la popularidad.

```
ggplot(data, aes(x=polarity)) +
geom_bar(aes(y=..count.., fill=polarity)) +
scale_fill_brewer(palette='RdGy') +
labs(x='Polarity categories', y='Number of tweets') +
ggtitle('Sentiment Analysis of Tweets about Brexit\n(classification by polarity)') +
theme(plot.title = element_text(size=12, face='bold'))
```



Ahora vamos a crear una nube de palabras para ver que es lo que más se usa. Lo primero será separar el texto por emociones y visualizar las palabras

```
emociones_data = levels(factor(data$emotion))
tama_emociones_data = length(emociones_data)
documento_emociones = rep('', tama_emociones_data)
for (i in 1:tama_emociones_data)
{
tmp = texto_tweets[emociones == emociones_data[i]]
documento_emociones[i] = paste(tmp, collapse=' ')
}

# eliminamos palabras vacias como or,as,off...
documento_emociones = removeWords(documento_emociones, stopwords('english'))
# Creamos el corpus
corpus = Corpus(VectorSource(documento_emociones))
termdocumentmatrix = TermDocumentMatrix(corpus)
termdocumentmatrix = as.matrix(termdocumentmatrix)
colnames(termdocumentmatrix) = emociones_data

# comparison word cloud
comparison.cloud(termdocumentmatrix, colors = brewer.pal(tama_emociones_data, 'Dark2'),
scale = c(3,.5), random.order = FALSE, title.size = 1.5)
```

Análisis Android, iPhone y iPad.

Cargamos las librerías necesarias para dicho análisis.

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:twitteR':
##
##     id, location

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(purrr)
```

```
##
## Attaching package: 'purrr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     contains, order_by
```

```
## The following object is masked from 'package:plyr':
##
##     compact
```

```
library(twitteR)
```

Vamos a trabajar con los mismos tweets y los almacenamos en una estructura de dataframe.

```
tweets_df <- tbl_df(map_df(tweets, as.data.frame))
```

Creamos otra estructura de dataframe con aquellos tweets que han sido publicados desde iPhone, desde Android o desde iPad.

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:RCurl':
##
##     complete
```

```
tweets_iph_an_ipa <- tweets_df %>%
  select(id, statusSource, text, created) %>%
  extract(statusSource, "source", "Twitter for (.*?)<") %>%
  filter(source %in% c("iPhone", "Android", "iPad"))
```

Podemos ver el número de tweets que se han publicado desde iPhone, Androir y iPad, siendo este último medio el menos utilizado.

```
num_iphone <- nrow(tweets_df %>%
  select(id, statusSource, text, created) %>%
  extract(statusSource, "source", "Twitter for (.*?)<") %>%
  filter(source %in% "iPhone"))
num_android <- nrow(tweets_df %>%
  select(id, statusSource, text, created) %>%
  extract(statusSource, "source", "Twitter for (.*?)<") %>%
  filter(source %in% "Android"))
num_ipad <- nrow(tweets_iph_an_ipa) - num_iphone - num_android
num_iphone
```

```
## [1] 2511
```

```
num_android
```

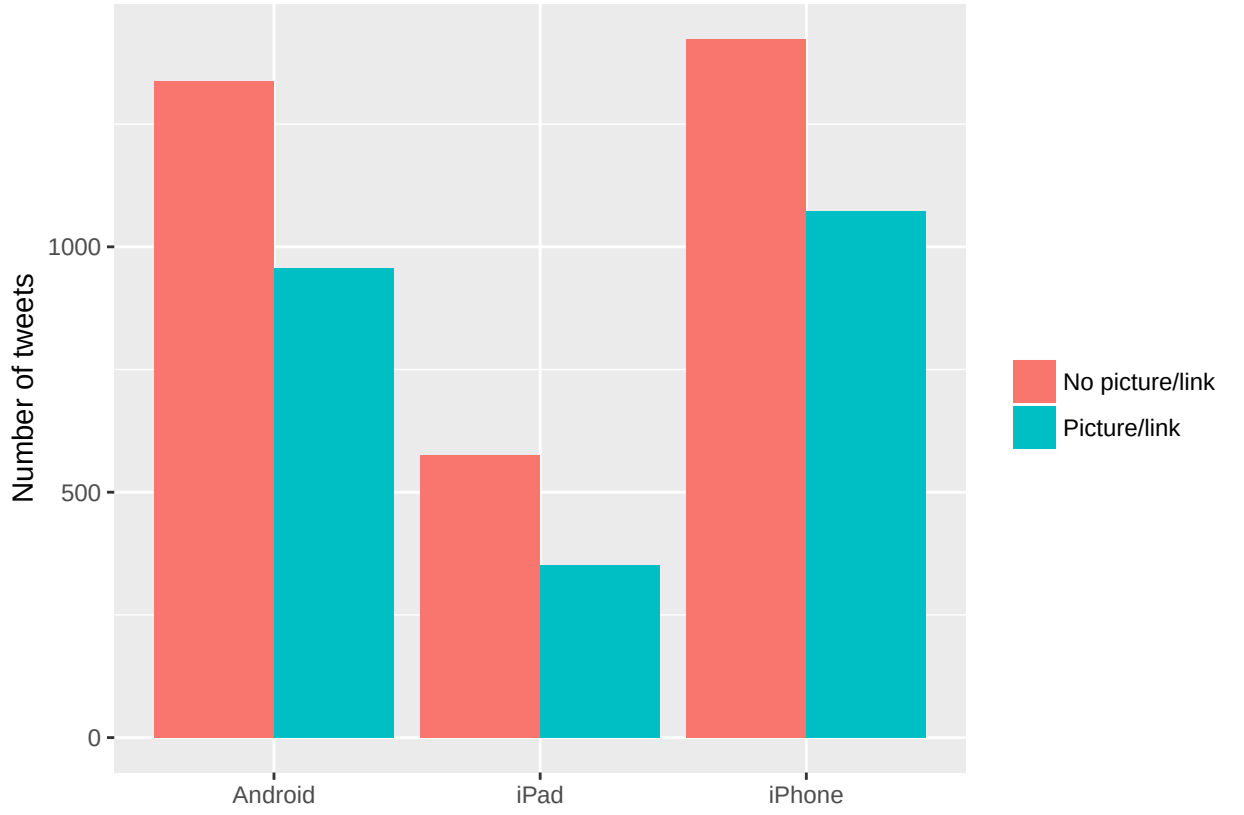
```
## [1] 2298
```

```
num_ipad
```

```
## [1] 931
```

Podemos comparar la cantidad de tweets que se publican con y sin imágenes o links. Hay tendencia en los tres casos a que los tweets no contengan ni imágenes ni links.

```
library(stringr)
tweet_picture_counts <- tweets_iph_an_ipa %>%
  filter(!str_detect(text, '^"')) %>%
  count(source,
        picture = ifelse(str_detect(text, "t.co"),
                         "Picture/link", "No picture/link"))

ggplot(tweet_picture_counts, aes(source, n, fill = picture)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "", y = "Number of tweets", fill = "")
```



Ahora creamos un nuevo dataset donde cada línea muestra una palabra relevante de un tweets. De este modo, para un mismo tweet podremos tener varias filas en tweet_words.

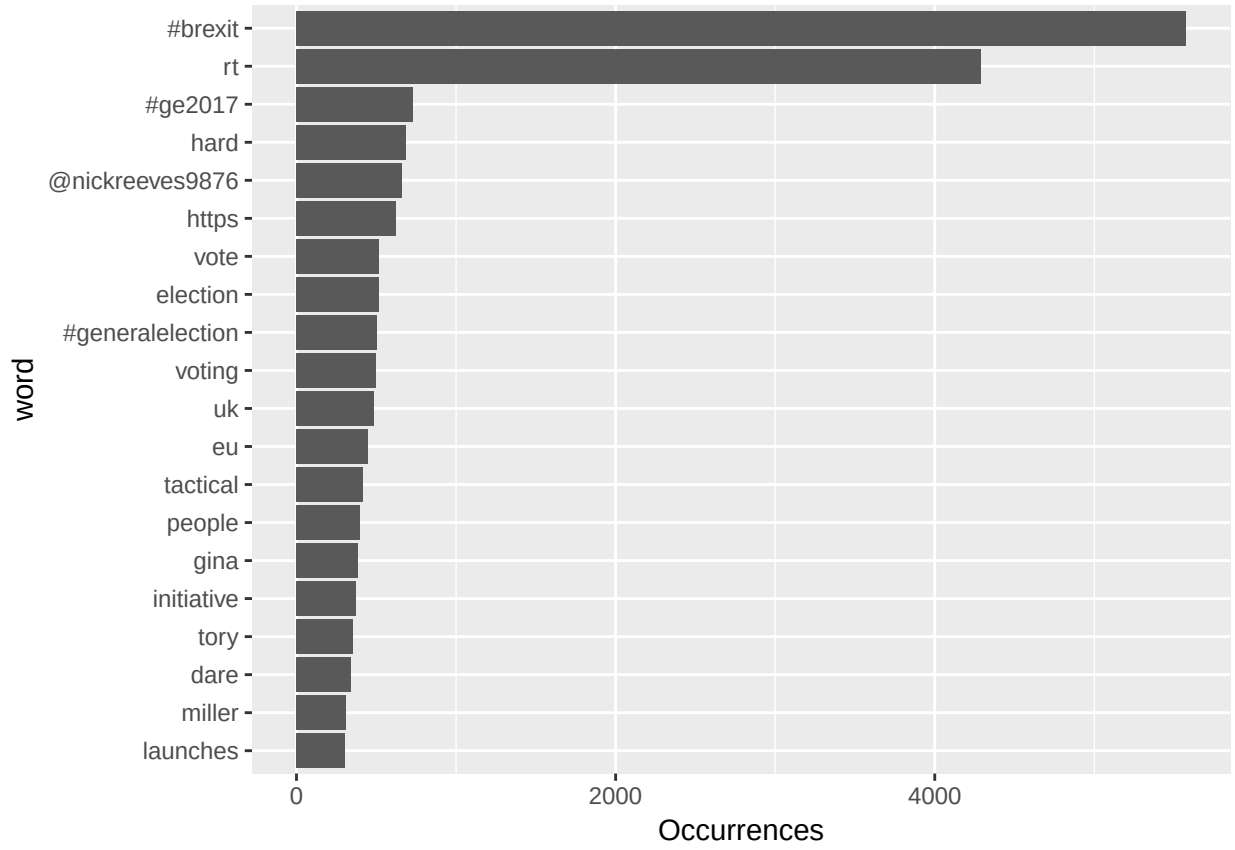
```
library(tidytext)
reg <- "([^A-Za-z\\d#@']|'(?![A-Za-z\\d#@]))"
tweet_words <- tweets_iph_an_ipa %>%
  filter(!str_detect(text, '^"')) %>%
  mutate(text = str_replace_all(text, "https://t.co/[A-Za-z\\d]+|&amp;", "")) %>%
  unnest_tokens(word, text, token = "regex", pattern = reg) %>%
  filter(!word %in% stop_words$word,
         str_detect(word, "[a-z]"))
tweet_words
```

```
## # A tibble: 59,678 × 4
```

```
##                    id  source             created    word
##                 <chr>   <chr>              <dttm>   <chr>
## 1  854780571179986944 Android 2017-04-19 19:35:43    @cer
## 2  854780571179986944 Android 2017-04-19 19:35:43   grant
## 3  854780571179986944 Android 2017-04-19 19:35:43 germany
## 4  854780571179986944 Android 2017-04-19 19:35:43  doesnt
## 5  854780571179986944 Android 2017-04-19 19:35:43    easy
## 6  854780571179986944 Android 2017-04-19 19:35:43 #brexit
## 7  854780571179986944 Android 2017-04-19 19:35:43      uk
## 8  854780571179986944 Android 2017-04-19 19:35:43 glamour
## 9  854780571179986944 Android 2017-04-19 19:35:43 missing
## 10 854780571179986944 Android 2017-04-19 19:35:43   links
## # ... with 59,668 more rows
```

Podemos mostrar en una gráfica cuáles son las 20 palabras más relevantes de todas.

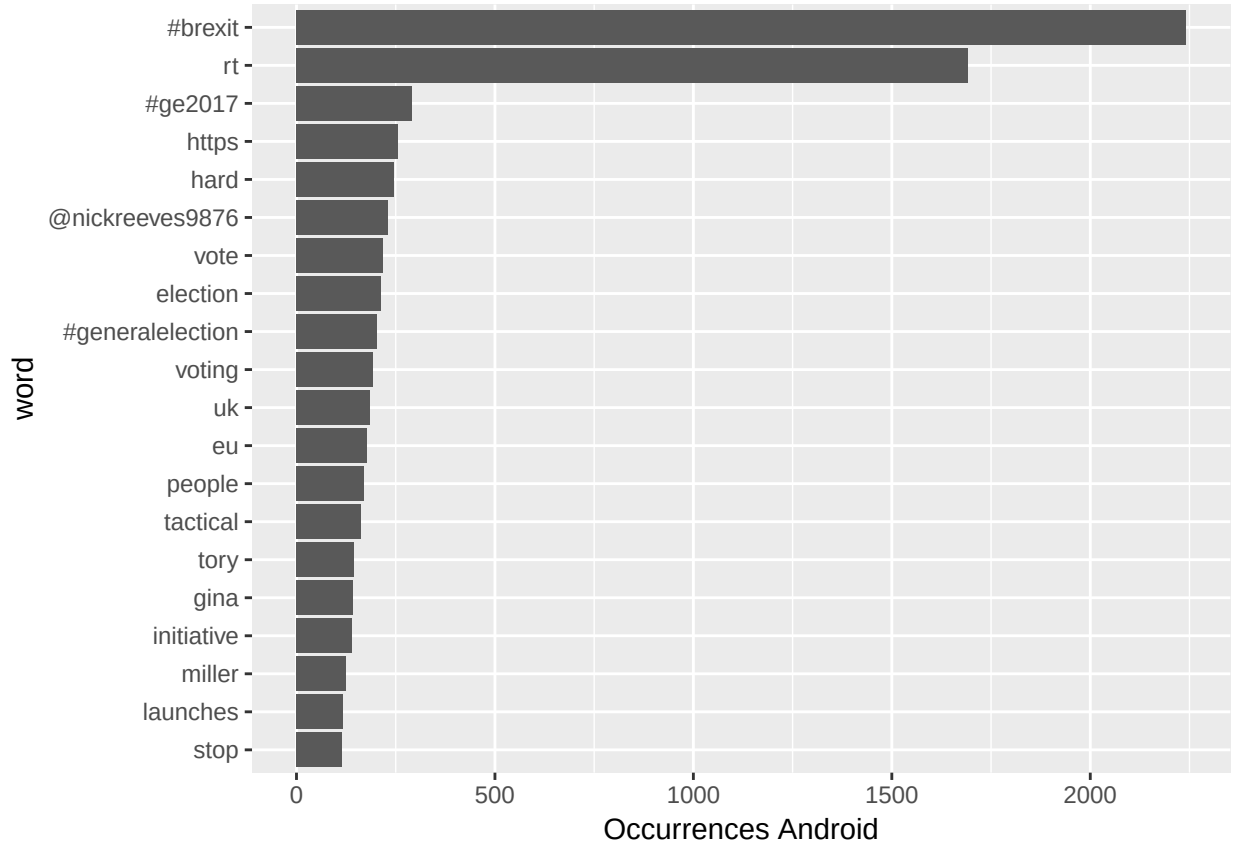
```
tweet_words %>%
  count(word, sort = TRUE) %>%
  head(20) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  ylab("Occurrences") +
  coord_flip()
```



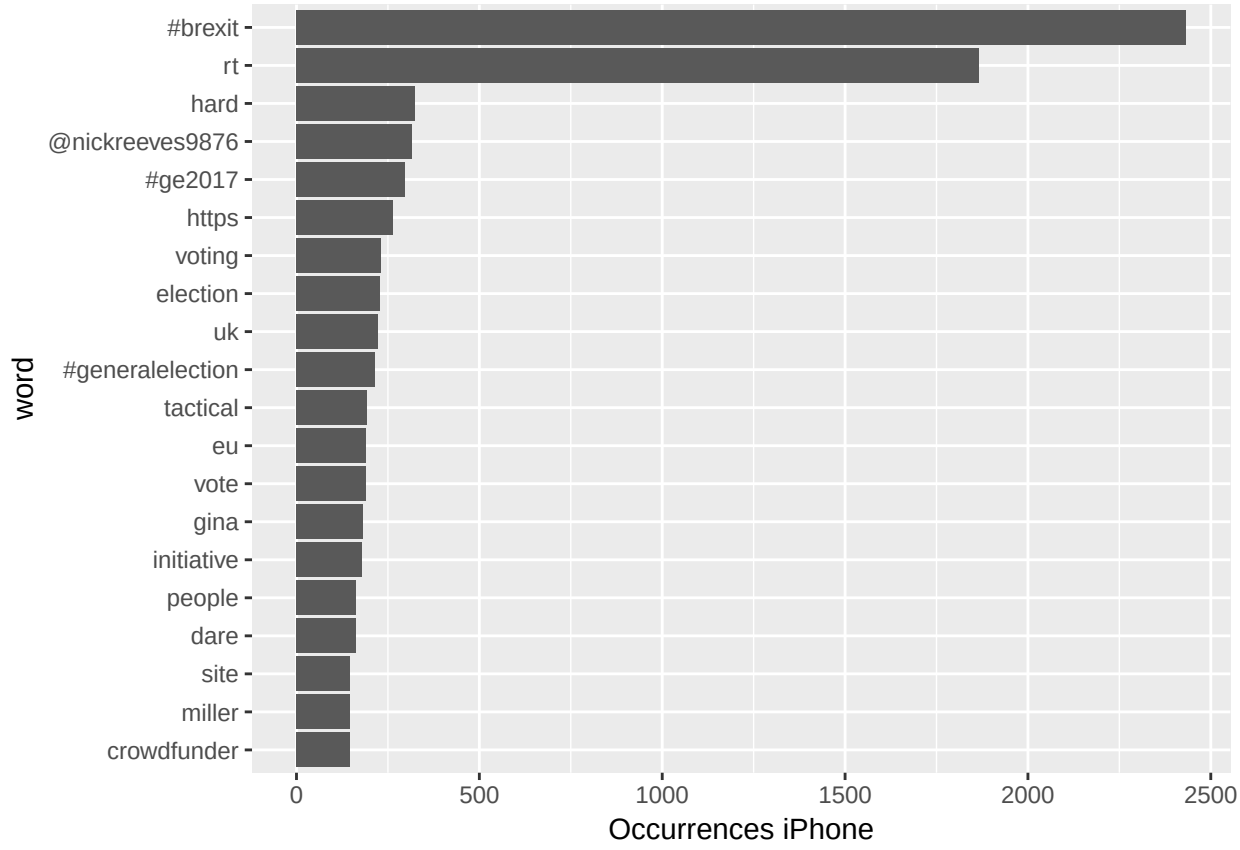
Finalmente podemos mostrar tres gráficas comparativas que representan las palabras más relevantes para Android, iPhone y iPad. Vemos que no hay grandes diferencias significativas y llama la atención que en los tres casos una de las palabras más usadas es rt para que la gente de difusión al propio tweet que se está

publicando.

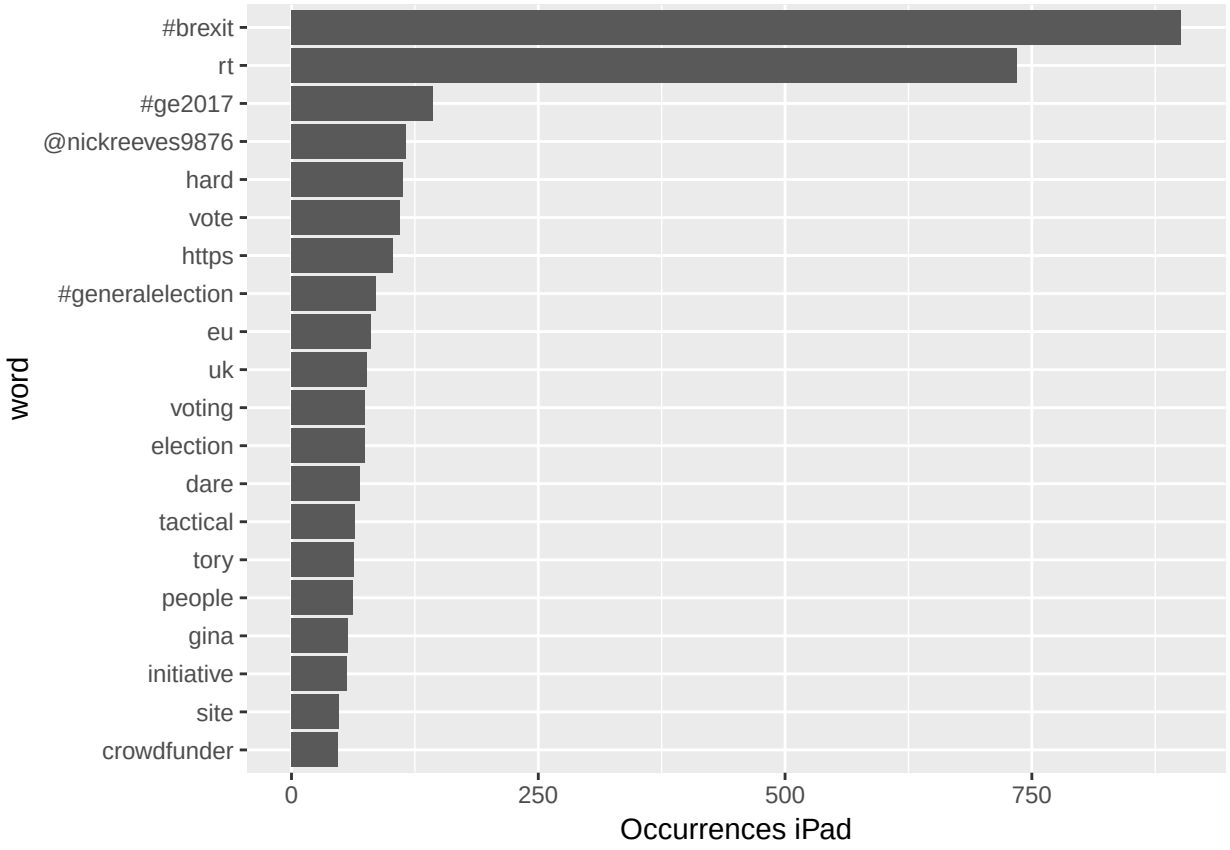
```
t_words_android <- tweet_words[which(tweet_words$source=="Android"),]
t_words_iphone <- tweet_words[which(tweet_words$source=="iPhone"),]
t_words_ipad <- tweet_words[which(tweet_words$source=="iPad"),]

t_words_android %>%
  count(word, sort = TRUE) %>%
  head(20) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  ylab("Occurrences Android") +
  coord_flip()
```




```
t_words_iphone %>%
  count(word, sort = TRUE) %>%
  head(20) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  ylab("Occurrences iPhone") +
  coord_flip()
```




```
t_words_ipad %>%
  count(word, sort = TRUE) %>%
  head(20) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  ylab("Occurrences iPad") +
  coord_flip()
```


#brexit
rt
#ge2017
@nickreeves9876
hard
vote
https
#generalelection
eu
uk
voting
election
dare
tactical
tory
people
gina
initiative
site
crowdfunder
word
0
250
500
750
Occurrences iPad