Tarea BAIN 17/03/2022

Gonzalo Fernández Suárez

20/3/2022

Es buena práctica cargar al inicio las librerías necesarias para ejecutar el resto del markdown

```
library(data.table)
library(stringr)
library(quanteda)
library(quanteda.textplots)
library(quanteda.textstats)
library(tidyverse)
library(tidytext)
```

Objetivos de este documento RMarkdown y de la tarea

El objetivo de este documento es proporcionar instrucciones sobre la tarea evaluada #1 de la asignatura BAIN, y a la vez, proporcionar un esquema básico de análisis -al menos de los primeros pasos.

Es *crítico* que dispongas de los objetos R contenidos en los diferentes ficheros .rda subidos al Blackboard. (O que verifiques que tus objetos guardados a partir de las clases contienen la misma información).

Tarea 1

Se trata de que realices un análisis comparativo de los tweets **sobre un tema de tu elección**. Como es evidente, será más interesante si se trata de un tema frecuente ("popular") en el conjunto total de tweets, y quizás también polémico. Estos temas los puedes encontrar en los bi- y tri-gramas obtenidos en clases anteriores, y que puedes replicar a partir del código en BAIN_22_resumen_preproceso_para_tarea_1.rmd.

El proceso de trabajo sugerido (hay infinidad de maneras de abordarlo) es:

1. Extrae del dataframe con todos los tweets aquel subconjunto que contiene el tema (o temas) que te interesen.
2. Crea un corpus con ese subconjunto (y limpia la memoria para evitar colapsos) y añade los metadatos (docvars).
3. Limpia de stopwords y en general de aquellos tokens que no te añaden información o significado.
4. Genera bi- o tri-gramas a partir de los tokens limpiados y muestra la frecuencia relativa de los mismos.

Hasta aquí el resultado supone 5 puntos en el trabajo.

5. Genera los tokens por separado para distintas categorías de tweets (campo Category que se ha introducido como metadato), o por fechas (a partir de la fecha introducida como metadato) y haz una comparación entre ambos a partir de objetos bi- o tri-gramas.
6. Genera wordclouds comparativos y extrae conclusiones.

Estos dos puntos suponen otros 2 puntos adicionales al trabajo.

7. Realiza un análisis estadístico de corpus o genera tópicos (topicmining) como veremos en clase.
8. Realiza visualizaciones que permitan extraer conclusiones significativas a partir del proceso.

Estos dos puntos suponen otros 2 puntos (hasta 9 sobre 10) del trabajo.

Y un último punto lo otorgaré por la calidad general del documento entregado.

POR FAVOR LA ENTREGA DEBE SER UN RMARKDOWN *O* COMO ÚNICA ALTERNATIVA, UN SCRIPT .R CON UN .DOC O .PDF O .PPT ASOCIADO

Analisis sociocultural sobre el impacto de atentados terroristas entre 2012-2018 en twitter

Bibliografía: www.since911.com

1 Creación de dataframe con los tweets de interés

Cargamos el objeto creado en clase con los **tweets ya filtrados y limpios** de expresiones regulares y puntuación. Para comprobar que es correcto nos aseguramos de que tenga 1849909 filas (tweets) y 14 columnas.

```
load('tweets_filtrados_data_frame_base_antes_de_corpus.rda')
dim(nuevo_objeto2)
```

```
## [1] 1849909      14
```

Comprobamos también que **no existen URLs ni caracteres no-ascii** (Como emoticonos).

```
grep('noascii', iconv(nuevo_objeto2$content,
                      from = 'UTF-8',
                      to = 'ASCII',
                      sub = 'noascii'))
```

```
## integer(0)
```

```
length(grep('http', nuevo_objeto2$content))/nrow(nuevo_objeto2) #% de tui
ts con http

## [1] 0
```

Ahora mismo la columna “publish_date” contiene información de tipo character (string). Necesitamos pasar **esos datos a tipo fecha** (Date).

```
nuevo_objeto2$publish_date <- as.Date(nuevo_objeto2$publish_date,
                                      format = "%m/%d/%Y")
summary(nuevo_objeto2)

##     author            content             region            language
##  Length:1849909     Length:1849909     Length:1849909     Length:1849909
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   publish_date        harvested_date       following       followers
##  Min.   :2012-02-06   Length:1849909     Min.   :    0   Min.   :     0
##  1st Qu.:2016-01-07   Class :character   1st Qu.:  631   1st Qu.:   715
##  Median :2016-09-04   Mode  :character   Median : 2120   Median :  2168
##  Mean   :2016-07-17                      Mean   : 4529   Mean   :  7965
##  3rd Qu.:2017-01-28                      3rd Qu.: 6479   3rd Qu.: 13250
##  Max.   :2018-05-23                      Max.   :76210   Max.   :145244
##
##     updates             post_type           account_type    retweet
##  Min.   :     1              :1037383   Right     :490249   0:1037383
##  1st Qu.:  2101   QUOTE_TWEET:  28133   local     :450349   1: 812526
##  Median :  5475   RETWEET    : 784393   Left      :410130
##  Mean   : 12492                         Hashtager :229627
##  3rd Qu.: 15822                         news      :137807
##  Max.   :166113                         Commercial:111375
##                                         (Other)   : 20372
##      account_category  new_june_2018
##  NewsFeed    :588156   0:1602167
##  RightTroll  :490249   1: 247742
##  LeftTroll   :410130
##  HashtagGamer:229627
##  Commercial  :111375
##  Fearmonger  : 10773
##  (Other)     :  9599
```

Las fechas de los tweets recolectados se encuentran entre **2012-02-06 y 2018-05-23**.

Extraemos un subconjunto del DataFrame con la información característica para este estudio.

Si nos paramos a mirar el dataframe que tenemos (nuevo_objeto2). Observaremos que las columnas mas relevantes a la hora de **escoger un subconjunto característico** para un estudio son:

- region
- language
- publish_date

```r
unique(nuevo_objeto2$region)
```

```
## [1] "United States"  "United Kingdom"
```

```r
unique(nuevo_objeto2$language)
```

```
## [1] "English"
```

```r
length(unique(nuevo_objeto2$publish_date))
```

```
## [1] 1364
```

Debido a filtrados anteriores, la información almacenada en nuestro Dataframe proviene de EEUU y UK. Además todos los tweets se encuentran en Inglés. Por otro lado tenemos tweets de **1364 días distintos**, por lo que escogeré mis subconjuntos de tweets característicos en función de **rangos de tiempo**.

Obtenemos subconjuntos de tweets (Con mas de 4000 tweets) desde 5 días antes de cada atentado hasta 25 días después. Perdiodos de 30 días. La información de cada atentado ha sido obtenida de www.since911.com

```r
charlie_hebdo_jan_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-01-02" &
                                   publish_date < "2015-02-02")
dim(charlie_hebdo_jan_2015)

hyper_cacher_jan_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-01-04" &
                                   publish_date < "2015-02-04")
dim(hyper_cacher_jan_2015)

sousse_june_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-06-23" &
                                   publish_date < "2015-07-23")
dim(sousse_june_2015)

bangkok_aug_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-08-13" &
                                   publish_date < "2015-09-13")
dim(bangkok_aug_2015)

ankara_oct_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-10-05" &
                                   publish_date < "2015-11-05")
dim(ankara_oct_2015)

metrojet_flight_oct_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-10-26" &
                                   publish_date < "2015-11-26")
dim(metrojet_flight_oct_2015)

beirut_nov_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-11-07" &
                                   publish_date < "2015-12-07")
dim(beirut_nov_2015)

paris_nov_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-11-08" &
                                   publish_date < "2015-12-08")
dim(paris_nov_2015)

san_berdardino_dec_2015 <-  subset (nuevo_objeto2,
                                   publish_date > "2015-12-17" &
```

```
                                        publish_date < "2016-01-17")
dim(san_berdardino_dec_2015)

brussels_march_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-03-17" &
                                        publish_date < "2016-04-17")
dim(brussels_march_2016)

orlando_june_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-06-07" &
                                        publish_date < "2016-07-07")
dim(orlando_june_2016)

jo_cox_june_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-06-11" &
                                        publish_date < "2016-07-11")
dim(jo_cox_june_2016)

dhaka_july_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-06-26" &
                                        publish_date < "2016-07-26")
dim(dhaka_july_2016)

saudi_arabia_july_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-07-01" &
                                        publish_date < "2016-08-01")
dim(saudi_arabia_july_2016)

nice_france_july_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-07-09" &
                                        publish_date < "2016-08-09")
dim(nice_france_july_2016)

berlin_dec_2016 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-12-14" &
                                        publish_date < "2017-01-14")
dim(berlin_dec_2016)

istanbul_jan_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2016-12-26" &
                                        publish_date < "2017-01-26")
dim(istanbul_jan_2017)

westminster_march_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2017-03-17" &
                                        publish_date < "2017-04-17")
dim(westminster_march_2017)

manchester_may_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2017-05-17" &
                                        publish_date < "2017-06-17")
dim(manchester_may_2017)

london_bridge_june_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2017-05-28" &
                                        publish_date < "2017-06-28")
dim(london_bridge_june_2017)

finsbury_june_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2017-06-14" &
                                        publish_date < "2017-07-14")
dim(finsbury_june_2017)

barcelona_cambrils_aug_2017 <-  subset (nuevo_objeto2,
                                        publish_date > "2017-08-12" &
                                        publish_date < "2017-09-12")
dim(barcelona_cambrils_aug_2017)

parsons_green_sept_2017 <-  subset (nuevo_objeto2,
```

```
                                    publish_date > "2017-09-10" &
                                    publish_date < "2017-10-10")
dim(parsons_green_sept_2017)

marseille_oct_2017 <-  subset (nuevo_objeto2,
                                    publish_date > "2017-09-26" &
                                    publish_date < "2017-10-26")
dim(marseille_oct_2017)

mogadishu_oct_2017 <-  subset (nuevo_objeto2,
                                    publish_date > "2017-10-09" &
                                    publish_date < "2017-11-09")
dim(mogadishu_oct_2017)

new_york_oct_2017 <-  subset (nuevo_objeto2,
                                    publish_date > "2017-10-26" &
                                    publish_date < "2017-11-26")
dim(new_york_oct_2017)

sinai_nov_2017 <-  subset (nuevo_objeto2,
                                    publish_date > "2017-11-19" &
                                    publish_date < "2017-12-19")
dim(sinai_nov_2017)

carcasonne_trebes_march_2018 <-  subset (nuevo_objeto2,
                                    publish_date > "2017-03-18" &
                                    publish_date < "2017-04-18")
dim(carcasonne_trebes_march_2018)
```

```
## [1] 13910    14
## [1] 14544    14
## [1] 96348    14
## [1] 43102    14
## [1] 25052    14
## [1] 25416    14
## [1] 37679    14
## [1] 39355    14
## [1] 52186    14
## [1] 53676    14
## [1] 56646    14
## [1] 53526    14
## [1] 39443    14
## [1] 43724    14
## [1] 49034    14
## [1] 98620    14
## [1] 94371    14
## [1] 77099    14
## [1] 29072    14
## [1] 26461    14
## [1] 30567    14
## [1] 23753    14
## [1] 8020   14
## [1] 6671   14
## [1] 4218   14
## [1] 13179    14
## [1] 16194    14
## [1] 78398    14
```

Como resultado tenemos **28 dataframes** representando los **tweets enviados días antes y después de distintos atentados** en todo el mundo.

Ahora generamos un dataframe que contenga **todos estos subconjuntos** y lo llamaremos *terrorism_attacks*.

```
dataframes = list(ankara_oct_2015,bangkok_aug_2015,barcelona_cambrils_aug
_2017,beirut_nov_2015,berlin_dec_2016,brussels_march_2016,carcasonne_treb
es_march_2018,charlie_hebdo_jan_2015,dhaka_july_2016,finsbury_june_2017,h
yper_cacher_jan_2015,istanbul_jan_2017,jo_cox_june_2016,london_bridge_jun
e_2017,manchester_may_2017,marseille_oct_2017,metrojet_flight_oct_2015,mo
gadishu_oct_2017,new_york_oct_2017,nice_france_july_2016,orlando_june_201
6,paris_nov_2015,parsons_green_sept_2017,san_berdardino_dec_2015,saudi_ar
abia_july_2016,sinai_nov_2017,sousse_june_2015,westminster_march_2017)

terrorism_attacks_tweets <- Reduce( union_all , dataframes)

dim(terrorism_attacks_tweets)

## [1] 1150264      14
```

Guardamos el objeto (Dataframe previo al corpus QUANTEDA y stop_words).

```
save(terrorism_attacks_tweets,stop_words, file = "terrorism_attacks_tweet
s.rda")
```

Limpiamos el entorno de trabajo para no ocupar demasiada memoria.

```
rm(list=ls())
```

2 Creación de Corpus de Quanteda y agregación de metadatos.

Cargamos el dataframe deseado.

```
load('terrorism_attacks_tweets.rda')
dim(terrorism_attacks_tweets)

## [1] 1150264      14

summary(terrorism_attacks_tweets)

##     author            content             region            language
##  Length:1150264     Length:1150264     Length:1150264     Length:1150264
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   publish_date        harvested_date       following       followers
##  Min.   :2015-01-03   Length:1150264     Min.   :    0   Min.   :     0
##  1st Qu.:2015-12-04   Class :character   1st Qu.:  676   1st Qu.:   722
##  Median :2016-07-19   Mode  :character   Median : 2547   Median :  2482
##  Mean   :2016-08-02                      Mean   : 5130   Mean   :  8665
##  3rd Qu.:2017-03-23                      3rd Qu.: 7750   3rd Qu.: 13588
##  Max.   :2017-12-18                      Max.   :76204   Max.   :145244
```

```
##
##    updates             post_type          account_type    retweet
##  Min.   :     1              :689837   Right     :296626   0:689837
##  1st Qu.:  2113   QUOTE_TWEET: 17692   local     :289100   1:460427
##  Median :  6105   RETWEET    :442735   Left      :231280
##  Mean   : 14690                       Hashtager :134809
##  3rd Qu.: 17004                       news      :101821
##  Max.   :166113                       Commercial: 74158
##                                       (Other)   : 22470
##      account_category  new_june_2018
##  NewsFeed    :390921   0:966618
##  RightTroll  :296626   1:183646
##  LeftTroll   :231280
##  HashtagGamer:134809
##  Commercial  : 74158
##  Fearmonger  : 16008
##  (Other)     :  6462
```

Creamos el **objeto Corpus** a partir del dataframe cargado.

```
terrorism_attacks_corpus <- corpus(terrorism_attacks_tweets$content)
```

Añadimos los metadatos al Corpus

Los metadatos serán:

- account_category: el tipo de cuenta que envió el tweet.
- retweet: si el tweet es un retweet o no.
- publish_date: la fecha de publicación.

```
docvars(terrorism_attacks_corpus, "Category") <- terrorism_attacks_tweets
$account_category
docvars(terrorism_attacks_corpus, "Retweet") <- terrorism_attacks_tweets$
retweet
docvars(terrorism_attacks_corpus, "Date_published") <- terrorism_attacks_
tweets$publish_date
```

Creamos *tokens_t* para **separar por espacios** las palabras contenidas en en el corpus.

```
tokens_t <- tokens(terrorism_attacks_corpus)
```

3 Limpieza de stopwords y tokens no significativos

Stop words recogidas de:

- www.blog.hubspot.com
- www.countwordsfree.com
- www.gist.github.com/larsyencken
- www.sites.google.com
- www.algs4.cs.princeton.edu
- www.github.com/stopwords-iso

Con ayuda de Visual Studio Code y www.textcompare.org para la correcta identación y eliminado de las palabras duplicadas.

```
mystopwords <- c( stopwords("english"), as.character(seq.int(from = 0, to
= 9)),
"im","t","r","rt","|","@","the","in","of","a","for","is","on","ABOUT","AC
TUALLY","ALMOST","ALSO","ALTHOUGH","ALWAYS","AM","AN","AND","ANY","ARE","
AT","BE","BECAME","BECOME","BUT","BY",…

…,news","video","time","NA NA")

length(mystopwords)

## [1] 1437
```

Tenemos **1434 stopwords** del inglés.

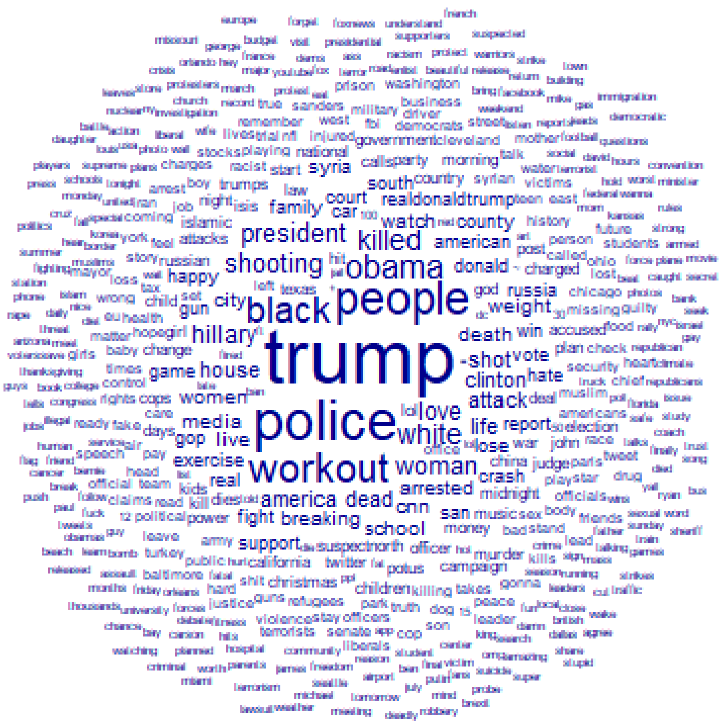
Generamos ahora una **matriz** con las **stopwords ya filtradas**.

```
tokens_t_nostop <- tokens_select(tokens_t,
                                 pattern = mystopwords,
                                 selection = "remove")


matriz <- dfm(tokens_t_nostop)
topfeatures(matriz,
            100)
```

```
##          trump         police         people        workout          black
##          51465          37338          30143          25339          21562
##          obama          white         killed      president       shooting
##          19695          14848          14550          14318          13998
##          woman           love        hillary          house        clinton
##          13843          12229          11382          10876          10718
##        america           dead           life         attack           shot
##          10492          10358          10238          10218           9773
##          death           city          women          watch          media
##           9454           9081           8767           8741           8734
##            cnn       breaking         school           live         weight
##           8667           8660           8354           8096           8087
##       arrested         donald realdonaldtrump       american           lose
##           7927           7907           7896           7876           7824
##          court         family       exercise            car            gun
##           7719           7711           7678           7603           7580
##            san         report          crash           vote         county
##           7253           7121           7110           7055           6946
##           real         russia           game            win           hate
##           6913           6871           6870           6841           6820
##          south          syria          fight          happy            gop
##           6791           6511           6498           6452           6409
##        support        suspect          north           deal        country
##           6400           6219           6183           6182           6166
##          texas            law        officer          money            god
##           5991           5971           5965           5942           5775
##            war           girl       midnight        charged        twitter
##           5747           5679           5604           5604           5571
##          china          party          isis        attacks           dies
##           5505           5464           5399           5393           5392
##          calls           post          start         trumps        russian
##           5359           5325           5273           5237           5194
##          music         change            hit       national        islamic
##           5087           5062           5059           5038           5022
##           cops          potus          child     california      christmas
##           4964           4963           4933           4900           4869
##           kids           plan           days            lol           kill
##           4852           4844           4837           4814           4806
##       children     government          judge        accused           john
##           4802           4793           4786           4784           4783
```

Generamos un wordcloud a partir de esta matriz

```
suppressWarnings({

textplot_wordcloud(matriz,
                   rotation = 0)
})
```




```
# Generamos un archivo png con el wordcloud
suppressWarnings({

  png(filename = "wordcloud_tokens_1_palabra.png",
      height = 3000,
      width = 3000)

  textplot_wordcloud(matriz,
                     rotation = 0,
                     max_words = 1000)
})

dev.off()

## png
##   2
```

Guardamos el **corpus** y los **tokens**.

```
save(  matriz,
       mystopwords,
       tokens_t,
       tokens_t_nostop,
       terrorism_attacks_corpus,
     file = "terrorism_corpus_and_tokens_gram.rda")
```

4 Generación de bi-gramas, tri-gramas y cálculo de frecuencias.

Mostramos la **frecuencia** de los tokens almacenados en *terrorism_attacks_tweets*.

Para un **correcto filtrado** añadimos *mystopwords* al objeto *stop_words*.

```
dim(stop_words)
```

```
## [1] 1149    2
```

```
stop_words_add <- data.frame( "word" = mystopwords,
                              "lexicon" = "GFS")
stop_words <- union_all(stop_words, stop_words_add)
dim(stop_words)
```

```
## [1] 2586    2
```

Pasamos de tener 1149 stop words a 2583.

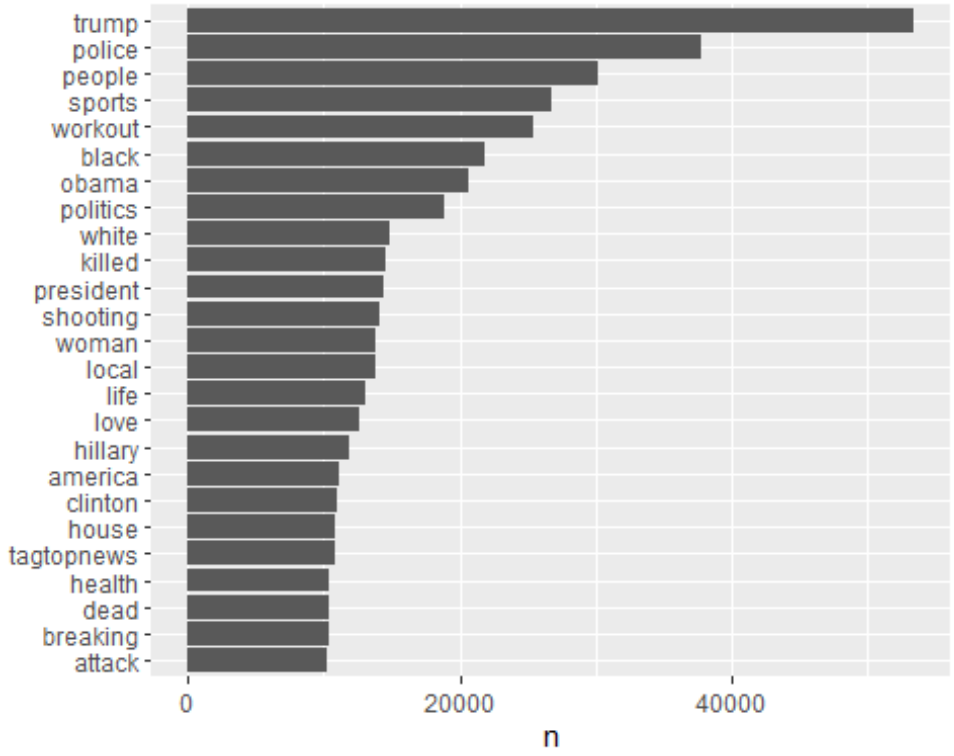
```
terrorism_tibble <- as_tibble(terrorism_attacks_tweets)

tokens_t_word <- terrorism_tibble %>%
  unnest_tokens(word,
                content,
                to_lower = TRUE)

tokens_t_word_nostop <- tokens_t_word %>%
  anti_join(stop_words)
```

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

```
tokens_t_word_nostop %>%
  count(word ,sort = TRUE)
```

```
## # A tibble: 549,980 x 2
##    word        n
##    <chr>    <int>
##  1 trump    53442
##  2 police   37843
##  3 people   30164
##  4 sports   26717
##  5 workout  25365
##  6 black    21839
##  7 obama    20644
##  8 politics 18879
##  9 white    14878
## 10 killed   14564
## # ... with 549,970 more rows
```

```
suppressWarnings({

  tokens_t_word_nostop %>%
    count(word, sort = TRUE) %>%
    filter(n > 10000) %>%
    mutate(word = reorder(word, n)) %>%
    ggplot(aes(n, word)) +
    geom_col() +
    labs(y = NULL)
})
```



Guardamos *tokens_t_nostop* (Tiene las stopwords ya filtradas) y *stop_words* con nuestras stopwords ya añadidas.

```
save(tokens_t_nostop, file = "tokens_t_nostop.rda")
save(stop_words, file = "stop_words.rda")
```

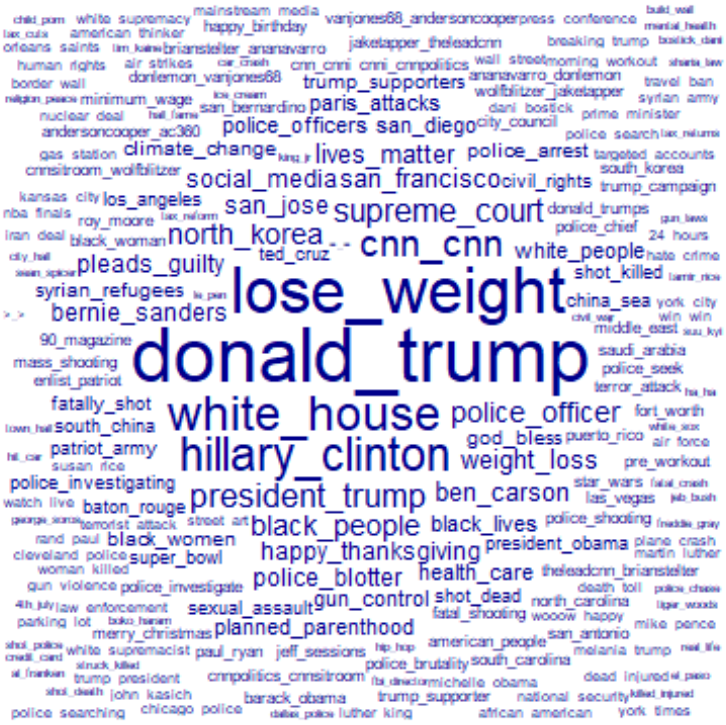
Liberamos memoria debido a que los bi-tri-gramas consumen mucha RAM.

```
rm(list=ls())
```

Construimos un bi-grama a partir de *tokens_t_nostop*. Con el una *matriz_bi* para obetener su wordcloud.

```
load("tokens_t_nostop.rda")

bigramas_terrorism <- tokens_ngrams(tokens_t_nostop,
2)
```

```
matriz_bi1 <- dfm(bigramas_terrorism)

suppressWarnings({

  textplot_wordcloud(matriz_bi1,
                     rotation = 0)

  rm(bigramas_terrorism,matriz_bi1)
  })
```



Construimos un tri-grama. También su *matriz_tri* y generamos su wordcloud.

```
trigramas_terrorism <- tokens_ngrams(tokens_t_nostop,
                                     3)
matriz_tri1 <- dfm(trigramas_terrorism)
suppressWarnings({

  textplot_wordcloud(matriz_tri1,
                     rotation = 0)

  rm(trigramas_terrorism,matriz_tri1)
  })
```

5 Generamos tokens por separado para distintas fechas (a partir de la fecha introducida como metadato) y los comparamos a partir de objetos bi- o tri-gramas.

Limpiamos memoria para evitar colapsos.

```
rm(list=ls())
```

Cargamos nuestro objeto *terrorism_attacks_tweets* que contiene los tweets que queremos ya listos para ser procesados. También cargamos *stop_words* actualizado.

```
load("terrorism_attacks_tweets.rda")
load("stop_words.rda")
```

Veamos rápidamente que categorías de cuentas tenemos.

```
table(terrorism_attacks_tweets$account_category)
```

```
##
##   Commercial   Fearmonger HashtagGamer    LeftTroll     NewsFeed   NonEnglish
##        74158        16008       134809       231280       390921         2885
##   RightTroll      Unknown
##       296626         3577
```

Vamos a generar un dataframe solo con los tweets de cuentas categorizadas como “Fearmonger”.

```
terrorism_fearmongers <- as_tibble(terrorism_attacks_tweets) %>%
  filter(account_category == "Fearmonger" )
dim(terrorism_fearmongers)

## [1] 16008    14
```

Tenemos 16008 "Fearmongers" en nuestro conjunto de tweets. Generamos los **tokens** y **quitamos las stop words**.

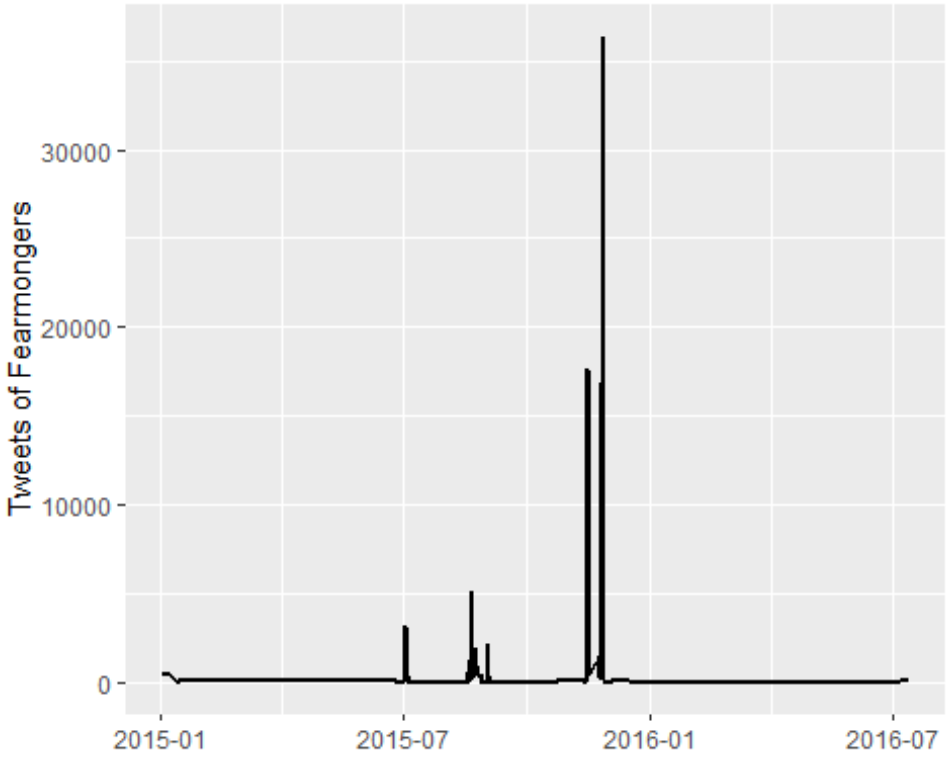
```
tokens_t_fearmongers <- terrorism_fearmongers %>%
  unnest_tokens(word,
                content,
                to_lower = TRUE)

tokens_t_fearmongers_nostop <- tokens_t_fearmongers %>%
  anti_join(stop_words)

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

Hacemos una cuenta de los tweets por día y los mostramos de forma descendente.

```
tokens_t_fearmongers_nostop %>%
  group_by(publish_date) %>%
  count() %>%
  arrange(desc(n))

## # A tibble: 75 x 2
## # Groups:   publish_date [75]
##    publish_date     n
##    <date>       <int>
##  1 2015-11-27   36362
##  2 2015-11-15   17604
##  3 2015-11-26   16798
##  4 2015-08-21    5151
##  5 2015-07-03    3129
##  6 2015-09-02    2169
##  7 2015-08-24    1950
##  8 2015-11-24    1401
##  9 2015-08-20    1188
## 10 2015-08-25     940
## # ... with 65 more rows
```

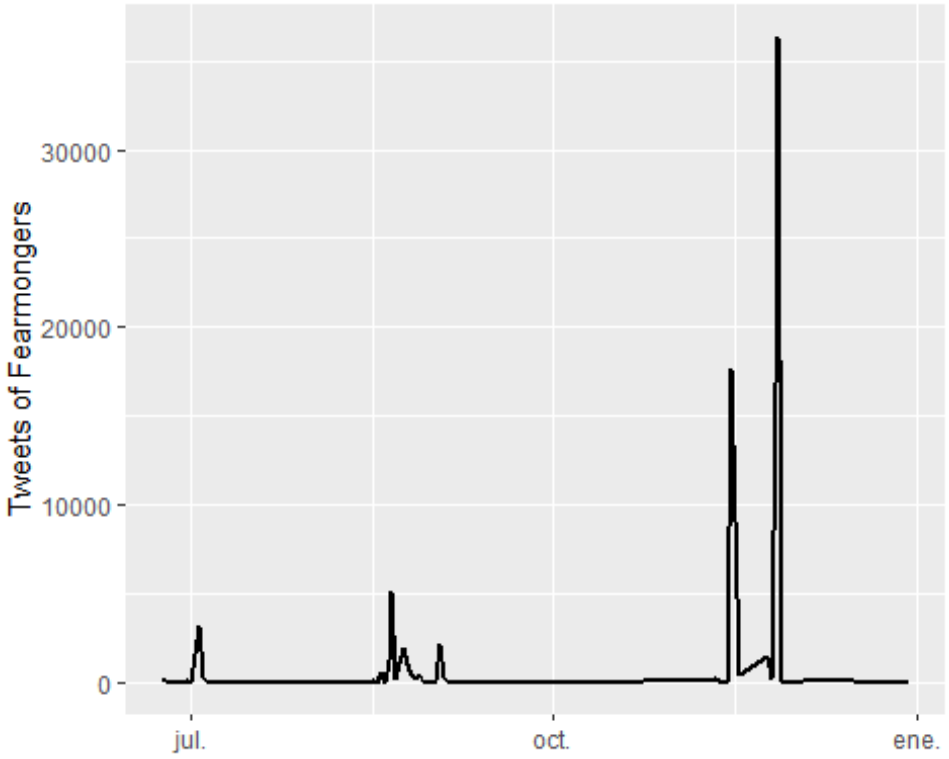
Veamos a lo largo del tiempo como se distribuyen los tweets de los "Fearmongers".

```
tokens_t_fearmongers_nostop %>%
  group_by(publish_date) %>%
  count() %>%
  ggplot(aes(publish_date, n)) +
  geom_line(size = 1) +
  labs(x = NULL, y = "Tweets of Fearmongers")
```



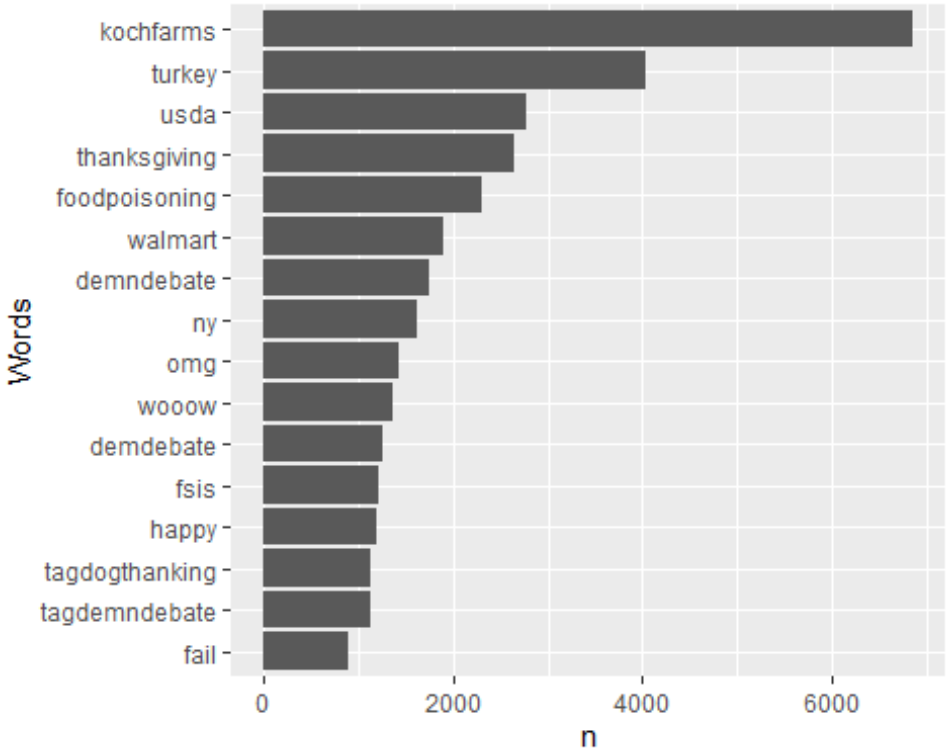
Observamos que solo hay tweets entre Junio de 2015 y Enero de 2016. **Acotemos** el periodo temporal para visualizar mejor la **zona con actividad**.

```
tokens_t_fearmongers_nostop %>%
  filter(  publish_date < "2016-01-01" &
           publish_date > "2015-05-01") %>%
  group_by(publish_date) %>%
  count() %>%
  ggplot(aes(publish_date, n)) +
  geom_line(size = 1) +
  labs(x = NULL, y = "Tweets of Fearmongers")
```



Veamos las **tf-idf** de los términos usados por los "Fearmongers" (n > 800).

```
tokens_t_fearmongers_nostop %>%
  count(word, sort = TRUE) %>%
  filter(n > 800) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = "Words", x = "n")
```



Comparación de tf-idf de LeftTrolls, RightTrolls y NewsFeed antes/después de 19 Dec 2016

Eliminamos espacio de trabajo para liberar RAM. Y cargamos los objetos necesarios.

```
rm(list=ls())
load("terrorism_attacks_tweets.rda")
load("stop_words.rda")
```

Observemos la diferencia entre las palabras con mayor frecuencia inversa por documento **antes y después del atentado con mas tweets** próximos a su fecha. Como hemos podido observar al principio del estudio, cuando hemos sacado subconjuntos por atentados, el atentado con más tweets cercanos a su fecha es el del (**19 de Diciembre de 2016 en Berlín**)[https://since911.com/explore/terrorism-timeline#jump_time_item_412].

Para ver esta diferencia dejaremos de lado los "Fearmongers", que no tienen una actividad significativa en este periodo.

Vamos a centrarnos en la frecuencia inversa por documento del conjunto **LeftTroll**, **RightTroll** y **NewsFeed**.

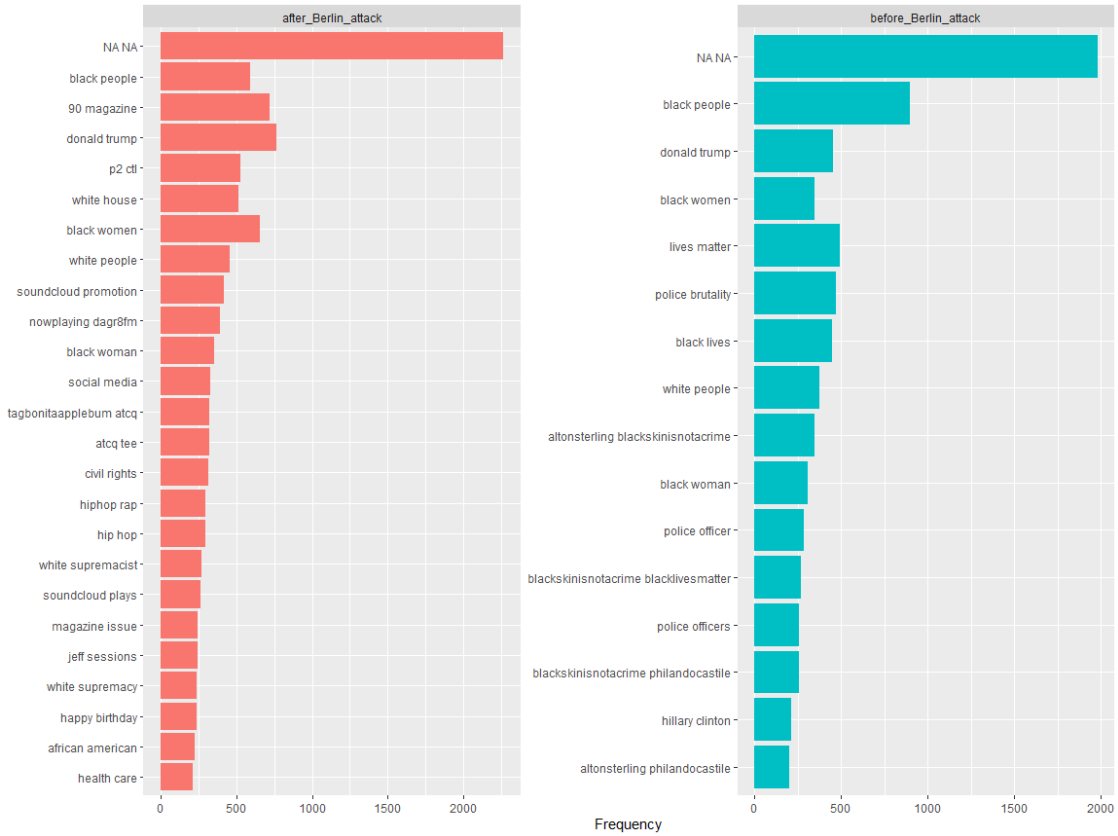
Comparación LeftTrolls antes y después del 19 Dec 2016

```
terrorism_left_trolls <- as_tibble(terrorism_attacks_tweets) %>%
  filter(account_category == "LeftTroll")
```

```r
dim(terrorism_left_trolls)
```

```
## [1] 231280     14
```

```r
bigrams_tokens_left_trolls <- terrorism_left_trolls %>%
  unnest_tokens(bigram,
                content,
                to_lower = TRUE,
                token = "ngrams",
                n = 2)

bigrams_separated <- bigrams_tokens_left_trolls %>%
  separate(bigram, c("word1", "word2"),
           sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

bigrams_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep = " ")

bigrams <- bigrams_united[ , c(4, 11, 14)]
bigrams <- bigrams %>%
  mutate(publish_date,
         attack = ifelse(publish_date < "2016-12-19",
                         "before_Berlin_attack",
                         "after_Berlin_attack"))
suppressWarnings({

  bigrams %>%
    group_by(bigram, attack) %>%
    count() %>%
    filter(n > 200) %>%
    ungroup() %>%
    mutate(bigram = reorder(bigram, n)) %>%
    ggplot(aes(n, bigram, fill = attack)) +
    geom_col(show.legend = FALSE) +
    labs(x = "Frequency", y = NULL) +
    facet_wrap(~attack, ncol = 2, scales = "free")

})
```

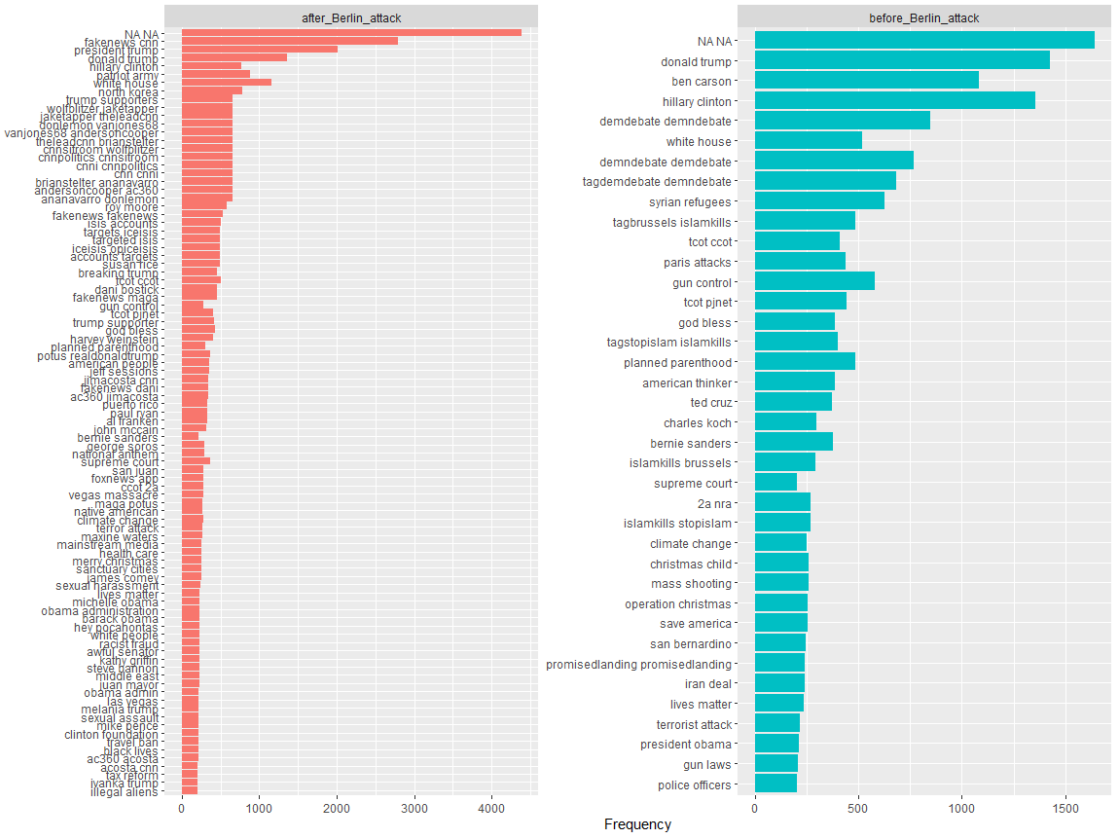


Comparación RightTrolls antes y después del 19 Dec 2016

```
# Liberamos memoria y cargamos
rm(list=ls())
load("terrorism_attacks_tweets.rda")
load("stop_words.rda")
```

```
terrorism_right_trolls <- as_tibble(terrorism_attacks_tweets) %>%
  filter( account_category == "RightTroll" )

dim(terrorism_right_trolls)

## [1] 296626     14

bigrams_tokens_right_trolls <- terrorism_right_trolls %>%
  unnest_tokens(bigram,
                content,
                to_lower = TRUE,
                token = "ngrams",
                n = 2)

bigrams_separated <- bigrams_tokens_right_trolls %>%
  separate(bigram, c("word1", "word2"),
           sep = " ")

bigrams_filtered <- bigrams_separated %>%
```

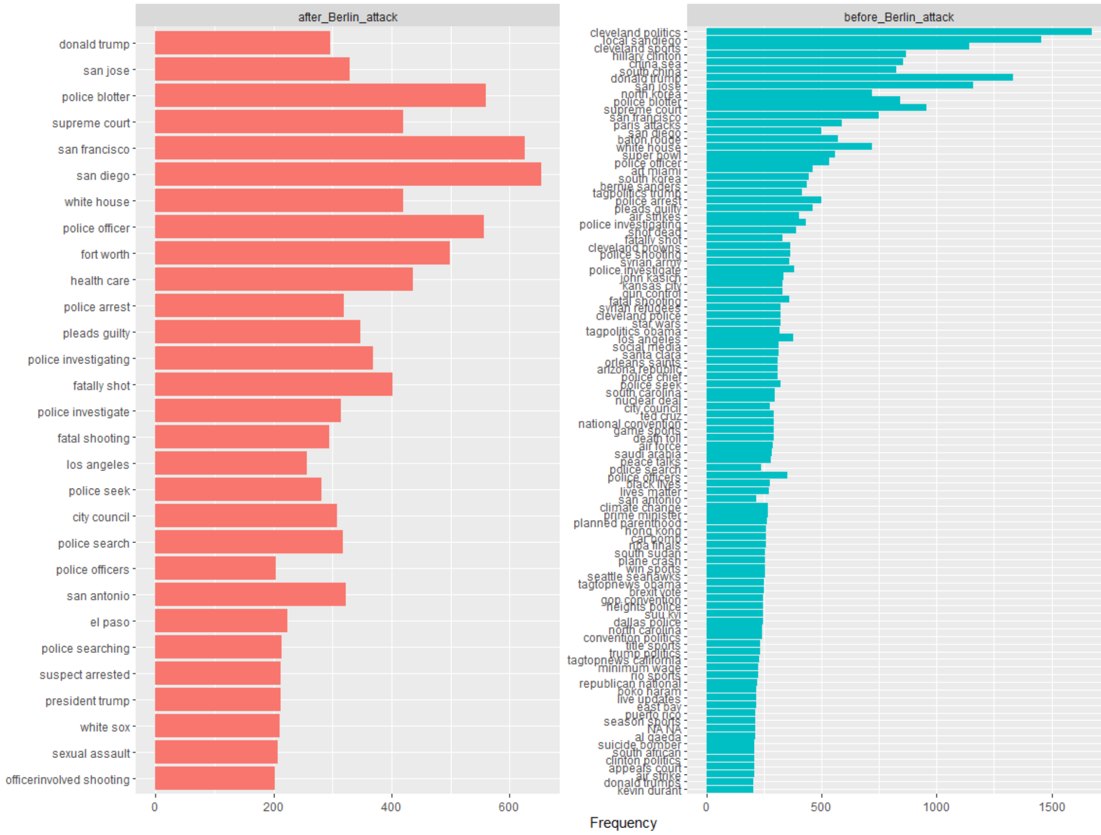
```r
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

bigrams_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep = " ")

bigrams <- bigrams_united[ , c(4, 11, 14)]
bigrams <- bigrams %>%
  mutate(publish_date,
         attack = ifelse(publish_date < "2016-12-19",
                         "before_Berlin_attack",
                         "after_Berlin_attack"))
suppressWarnings({

  bigrams %>%
    group_by(bigram, attack) %>%
    count() %>%
    filter(n > 200) %>%
    ungroup() %>%
    mutate(bigram = reorder(bigram, n)) %>%
    ggplot(aes(n, bigram, fill = attack)) +
    geom_col(show.legend = FALSE) +
    labs(x = "Frequency", y = NULL) +
    facet_wrap(~attack, ncol = 2, scales = "free")

})
```



Comparación NewsFeed antes y después del 19 Dec 2016

```
# Liberamos memoria y cargamos
rm(list=ls())
load("terrorism_attacks_tweets.rda")
load("stop_words.rda")

terrorism_news_feed <- as_tibble(terrorism_attacks_tweets) %>%
  filter( account_category == "NewsFeed")

dim(terrorism_news_feed)

## [1] 390921     14

bigrams_tokens_news_feed <- terrorism_news_feed %>%
  unnest_tokens(bigram,
                content,
                to_lower = TRUE,
                token = "ngrams",
                n = 2)

bigrams_separated <- bigrams_tokens_news_feed %>%
  separate(bigram, c("word1", "word2"),
           sep = " ")

bigrams_filtered <- bigrams_separated %>%
```

```r
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

bigrams_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep = " ")

bigrams <- bigrams_united[ , c(4, 11, 14)]
bigrams <- bigrams %>%
  mutate(publish_date,
         attack = ifelse(publish_date < "2016-12-19",
                         "before_Berlin_attack",
                         "after_Berlin_attack"))
suppressWarnings({

  bigrams %>%
    group_by(bigram, attack) %>%
    count() %>%
    filter(n > 200) %>%
    ungroup() %>%
    mutate(bigram = reorder(bigram, n)) %>%
    ggplot(aes(n, bigram, fill = attack)) +
    geom_col(show.legend = FALSE) +
    labs(x = "Frequency", y = NULL) +
    facet_wrap(~attack, ncol = 2, scales = "free")

})
```


after_Berlin_attack
before_Berlin_attack
donald trump
san jose
police blotter
supreme court
san francisco
san diego
white house
police officer
fort worth
health care
police arrest
pleads guilty
police investigating
fatally shot
police investigate
fatal shooting
los angeles
police seek
city council
police search
police officers
san antonio
el paso
police searching
suspect arrested
president trump
white sox
sexual assault
officerinvolved shooting
0
200
400
600
cleveland politics
local sandiego
cleveland sports
hillary clinton
china sea
south china
donald trump
san jose
north korea
police blotter
supreme court
san francisco
paris attacks
san diego
baton rouge
white house
super bowl
police officer
art miami
south korea
bernie sanders
tagpolitics trump
police arrest
pleads guilty
air strikes
police investigating
shot dead
fatally shot
cleveland browns
police shooting
syrian army
police investigate
john kasich
kansas city
gun control
fatal shooting
syrian refugees
cleveland police
star wars
tagpolitics obama
los angeles
social media
santa clara
orleans saints
arizona republic
police chief
police seek
south carolina
nuclear deal
city council
ted cruz
national convention
game sports
death toll
air force
saudi arabia
peace talks
police search
police officers
black lives
lives matter
san antonio
climate change
prime minister
planned parenthood
hong kong
car bomb
nba finals
south sudan
plane crash
win sports
seattle seahawks
tagtopnews obama
brexit vote
gop convention
heights police
suu kyi
dallas police
north carolina
convention politics
title sports
trump politics
tagtopnews california
minimum wage
rio sports
republican national
boko haram
live updates
east bay
puerto rico
season sports
nfl nfl
al qaeda
suicide bomber
south african
clinton politics
appeals court
air strike
donald trumps
kevin durant
0
500
1000
1500
Frequency