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.

1. 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.
2. Genera wordclouds comparativos y extrae conclusiones.

Estos dos puntos suponen otros 2 puntos adicionales al trabajo.

1. Realiza un análisis estadístico de corpus o genera tópicos (topicmining) como veremos en clase.
2. 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](https://since911.com/explore/terrorism-timeline)

## 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 tuits 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

unique(nuevo\_objeto2$region)

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

unique(nuevo\_objeto2$language)

## [1] "English"

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](https://since911.com/explore/terrorism-timeline)

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\_trebes\_march\_2018,charlie\_hebdo\_jan\_2015,dhaka\_july\_2016,finsbury\_june\_2017,hyper\_cacher\_jan\_2015,istanbul\_jan\_2017,jo\_cox\_june\_2016,london\_bridge\_june\_2017,manchester\_may\_2017,marseille\_oct\_2017,metrojet\_flight\_oct\_2015,mogadishu\_oct\_2017,new\_york\_oct\_2017,nice\_france\_july\_2016,orlando\_june\_2016,paris\_nov\_2015,parsons\_green\_sept\_2017,san\_berdardino\_dec\_2015,saudi\_arabia\_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\_tweets.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](https://blog.hubspot.com/marketing/stop-words-seo#:~:text=The%20most%20common%20SEO%20stop,your%2C%20our%2C%20and%20their.)
* [www.countwordsfree.com](https://countwordsfree.com/stopwords)
* [www.gist.github.com/larsyencken](https://gist.github.com/larsyencken/1440509)
* [www.sites.google.com](https://sites.google.com/site/kevinbouge/stopwords-lists)
* [www.algs4.cs.princeton.edu](https://algs4.cs.princeton.edu/35applications/stopwords.txt)
* [www.github.com/stopwords-iso](https://github.com/stopwords-iso/stopwords-en/blob/master/stopwords-en.json)

Con ayuda de Visual Studio Code y [www.textcompare.org](https://www.textcompare.org/text/remove-duplicate-words/) 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","ACTUALLY","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.

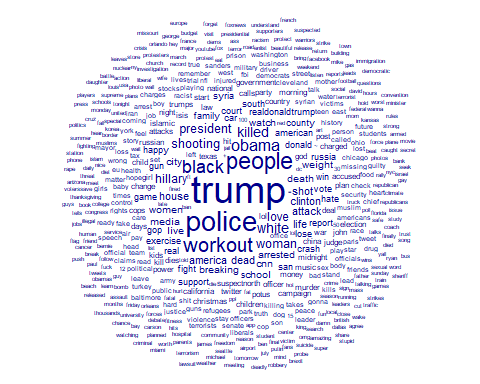
Generamos ahora una **matriz** con las **stopwords ya filtradas**.

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 5679 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.

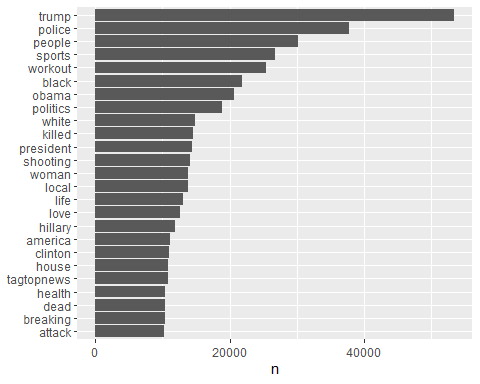
terrorism\_tibble <- as\_tibble(terrorism\_attacks\_tweets)  
  
tokens\_t\_word <- terrorism\_tibble %>%  
 unnest\_tokens(word,  
 content,  
 to\_lower = TRUE)  
  
tokens\_t\_word\_nostop <- tokens\_t\_word %>%  
 anti\_join(stop\_words)

## 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.

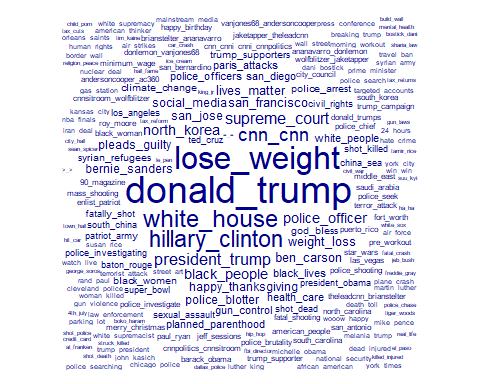
save(tokens\_t\_nostop, file = "tokens\_t\_nostop.rda")  
save(stop\_words, file = "stop\_words.rda")

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

rm(list=ls())

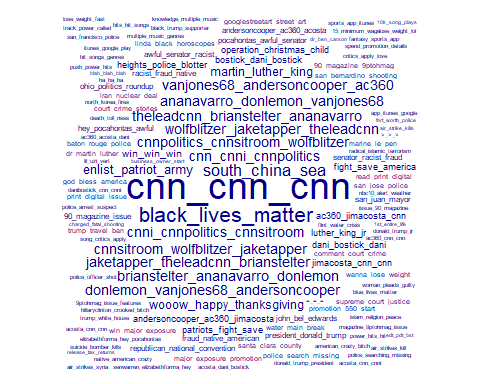
Construimos un bi-grama a partir de *tokens\_t\_nostop*. Con el una *matriz\_bi* para obetener su wordcloud.

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"

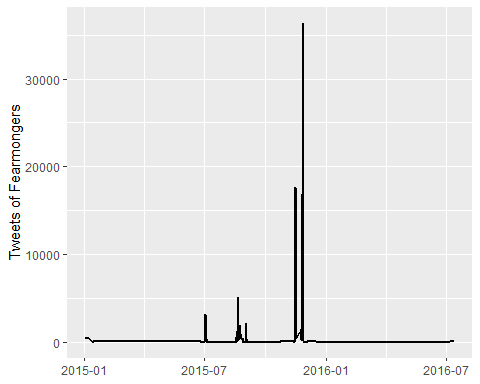
Hacemos una cuenta de los tweets por día y los mostramos de forma descendente.

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

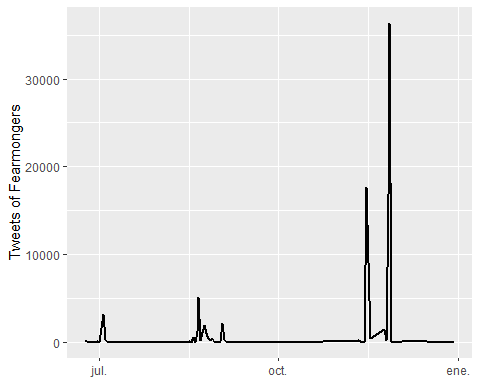
Veamos a lo largo del tiempo como se distribuyen los tweets de los “Fearmongers”.

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")



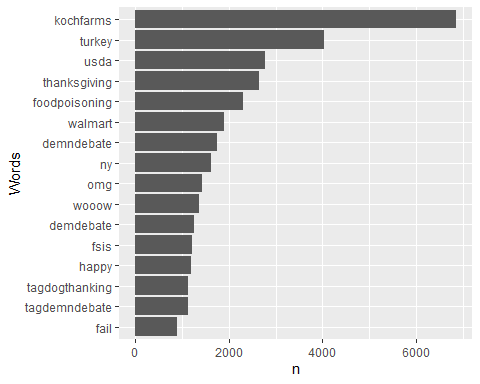
Observamos que solo hay tweets entre Junio de 2015 y Enero de 2016. **Acotemos** el periodo temporal para visualizar mejor la **zona con actividad**.

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.

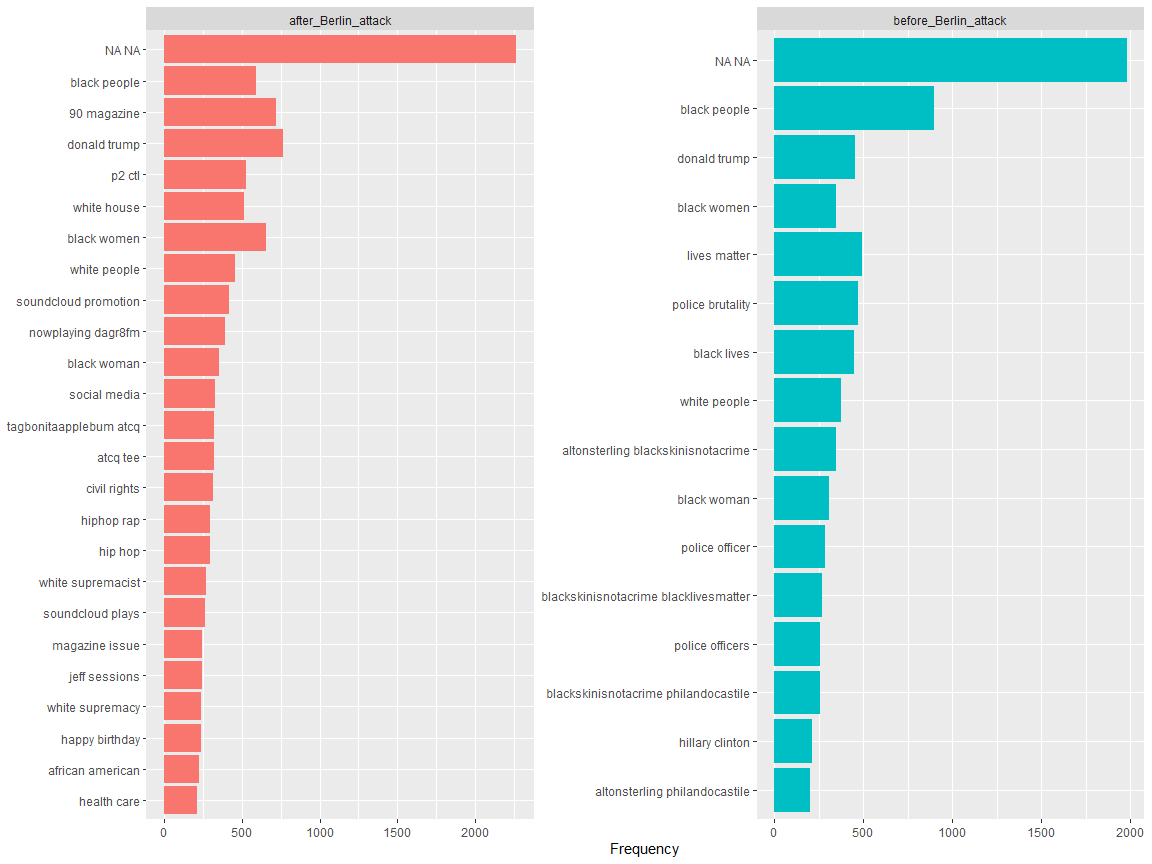
Vamos a centrarnos en la frecuencia inversa por documento del conjunto **LeftTroll**, **RightTroll** y **NewsFeed**.

#### Comparación LeftTrolls antes y después del 19 Dec 2016

terrorism\_left\_trolls <- as\_tibble(terrorism\_attacks\_tweets) %>%  
 filter(account\_category == "LeftTroll")  
  
dim(terrorism\_left\_trolls)

## [1] 231280 14

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")  
  
})



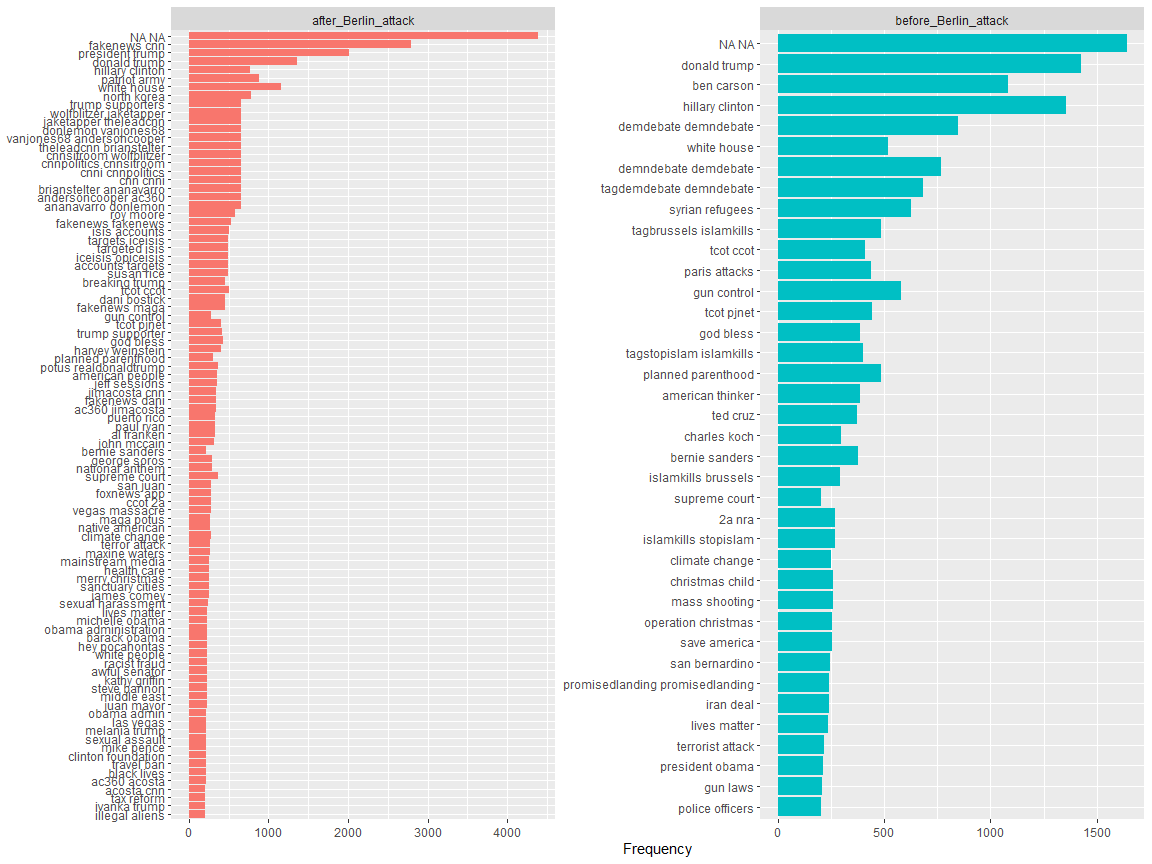
#### Comparación RightTrolls antes y después del 19 Dec 2016

# 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 %>%  
 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 %>%  
 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")  
  
})

