BAIN_22_Tarea_2_GonzaloFernandezSuarez

Gonzalo Fernandez Suarez

14/4/2022

Objetivo de este estudio

Analizar el texto extraído de las intervenciones en el Congreso de los Diputados de:

- Pedro Sánchez Pérez-Castejón
- Santiago Abascal Conde
- Pablo Casado Blanco

Enfocándonos en técnicas de **Text Mining y Sentiment Analysis** para compararlos y sacar conclusiones.

```
# Librerias utilizadas
library("pdftools")
library("tidyverse")
library("tidytext")
library("ggplot2")
library("ggpubr")
library("tidyr")
library("quanteda")
```

Extracción del texto de PDFs y filtrado del mismo

Gracias al código de web scraping realizado en Python, tenemos un conjunto de PDFs que contienen el **texto completo de los últimos plenos en los que han intervenido** estos oradores.

Procedimiento

Generamos una lista con los nombres de los PDFs

```
files <- list.files(recursive = TRUE, pattern = "PDF$")
length(files)
```

```
## [1] 62
```

Ahora extraemos el texto de cada elemento de files y lo guardamos en textoPlenos.

```
# Lista con el texto de cada PDF
textoPlenos <- lapply(files, pdf_text)
```

Tenemos guardado en **textoPlenos** el texto de 62 PDFs. Sin embargo, este objeto es una lista de listas de vectores character, por lo que no nos vale aun para filtrar el texto de impurezas y analizarlo. Necesitamos transformar este objeto a un único vector de caracteres.

```
# Lista de textos a un unico vector de tipo character
vectorTextoPlenos <- unlist(textoPlenos)
class(vectorTextoPlenos)
```

```
## [1] "character"
```

```
# Limpieza general de saltos de linea y espacios innecesarios
vectorTextoPlenos <- gsub("\\n", "", vectorTextoPlenos)
vectorTextoPlenos <- gsub("\\s+", " ", vectorTextoPlenos)

# Guardamos el texto de los PDFs
save(textoPlenos, file = "textoPlenos.rda")
# Guardamos el vector del texto de los PDFs
save(vectorTextoPlenos, file = "vectorTextoPlenos.rda")
```

Limpiamos memoria y trabajamos con el vector generado.

```
# Limpiamos memoria y cargamos el vector
rm(list = ls())
load("vectorTextoPlenos.rda")
```

Ahora ya podemos filtrar el texto para quedarnos solo con lo que nos interesa. A la hora de capturar las intervenciones de los políticos podemos seguir el siguiente patrón.

1. Capturar el texto que se encuentre entre “El señor :” (Cuando empieza a hablar) y “La señora PRESIDENTA:” (Cuando terminan su discurso y la presidenta del congreso ejerce sus funciones de moderadora)
2. Eliminamos cualquier cosa que esté entre paréntesis por que aluden a la realidad y no son palabras del orador (Ej: (Aplausos), (Rumores), ...). También otras anotaciones no relevantes que se encuentran entre guiones.
3. Por ultimo, colapsamos todas las intervenciones de cada político en un único vector con el que trabajaremos.

Intervenciones de Pablo Casado

```
# Buscamos las intervenciones
intervencionesCasado <- regmatches(vectorTextoPlenos, regexec("El señor CASADO BLANCO:\\s*(.*?)\\s*La señora PRESIDENTA:", vectorTextoPlenos))
# Eliminamos las filas vacias => character(0)
intervencionesCasado <- intervencionesCasado[!sapply(intervencionesCasado, identical, character(0))]
# Nos quedamos con el segundo objeto de cada lista, porque contiene el texto de dentro sin los patrones de busqueda
intervencionesCasado <- lapply(intervencionesCasado , function(x) unlist(x)[2] )

# Filtramos el texto
intervencionesCasado <- str_remove_all(intervencionesCasado, regex("\\([^()]+\\)")) # Eliminacion de parentesis y su contenido
intervencionesCasado <- str_remove_all(intervencionesCasado, regex("\\— DEL DIPUTADO[^()]+\\?.")) # Eliminacion de otro tipo de anotación no relevante
intervencionesCasado <- trimws(intervencionesCasado) # Eliminamos espacios al principio y al final

sprintf("Nº de intervenciones de Casado: %d ", length(intervencionesCasado) )
```

```
## [1] "Nº de intervenciones de Casado: 93 "
```

```
intervencionesCasado_Sep <- intervencionesCasado
intervencionesCasado[1]
```

```
## [1] "Señor Sánchez, sus recetas económicas son tan creíbles como sus promesas
electorales, y encima proponen las mismas recetas fracasadas que nos llevaron a la peor
crisis económica de nuestra historia: más despilfarro, más déficit y más impuestos. Pero
el Partido Popular es un partido de Estado y también de Gobierno, aunque estemos
temporalmente en la oposición. Por eso el lunes le ofrecí pactar los Presupuestos
Generales si rompe con los independentistas, una oferta, por cierto, a la que usted no ha
contestado. Hoy tiene oportunidad de hacerlo, porque ya no tiene excusa. Usted eligió
libremente a sus socios para la investidura, pero si ahora elige otra vez a los mismos
para los presupuestos es porque quiere. ¿O cree que al señor Torra le importan los
parados en España, o al señor Otegi los agricultores y ganaderos, o al señor Junqueras
los pensionistas o los autónomos? Ya lo dijeron aquí, les importa España un comino. Y
parece que a usted también si acepta una mesa de autodeterminación con un inhabilitado, o
reformar el Código Penal para indultos por la puerta de atrás, o la politización de la
Fiscalía General a cambio de unos presupuestos para seguir en la Moncloa. Ahora todo el
mundo sabe que usted tiene otra alternativa, así que si quiere abandonar el extremismo,
si quiere volver a la centralidad, anuncie hoy mismo que rompe con los independentistas y
siéntese con nosotros a pactar esos acuerdos de Estado que le he ofrecido. Deje de
reservar el diálogo como premio a la minoría radical y siéntese a dialogar con la mayoría
moderada, que es la que le conviene a España. . Aún está a tiempo de rectificar, y si lo
hace tendrá nuestra mano tendida. Es la única que puede evitar que usted caiga al vacío.
Créame, que yo no miento. ."

intervencionesCasado <- paste(intervencionesCasado, collapse = "") # Colapsamos la lista
en un unico vector
```

Intervenciones de Pedro Sanchez

```
# Buscamos las intervenciones
intervencionesSanchez <- regmatches(vectorTextoPlenos, regexec("El señor PRESIDENTE DEL
GOBIERNO \\(Sánchez Pérez-Castejón\\):\\s*(.*?)\\s*La señora PRESIDENTA:",
vectorTextoPlenos))
# Eliminamos las filas vacias => character(0)
intervencionesSanchez <- intervencionesSanchez[!sapply(intervencionesSanchez, identical,
character(0))]
# Nos quedamos con el segundo objeto de cada lista, porque contiene el texto de dentro
sin los patrones de busqueda
intervencionesSanchez <- lapply(intervencionesSanchez , function(x) unlist(x)[2] )

# Filtramos el texto
intervencionesSanchez <- str_remove_all(intervencionesSanchez, regex("\\([^()]+\\).")) #
Eliminacion de parentesis y su contenido
intervencionesSanchez <- str_remove_all(intervencionesSanchez, regex("\\— DEL
DIPUTADO[^()]+\\?.")) # Eliminacion de otro tipo de anotación no relevante
intervencionesSanchez <- trimws(intervencionesSanchez) # Eliminamos espacios al principio
y al final

sprintf("Nº de intervenciones de Sanchez: %d ", length(intervencionesSanchez) )

## [1] "Nº de intervenciones de Sanchez: 176 "

intervencionesSanchez_Sep <- intervencionesSanchez
intervencionesSanchez[1]

## [1] "Muchas gracias, señora presidenta. Claro que no, no hemos renunciado, señor
Casado, ni a crear empleo ni a reducir el paro ni tampoco a eliminar, por ejemplo, el
despido por baja médica que recogía la reforma laboral del Partido Popular y que ayer
precisamente derogamos en el Consejo de Ministros."
```

```
intervencionesSanchez <- paste(intervencionesSanchez, collapse = "") # Colapsamos la lista en un unico vector
```

Intervenciones de Santiago Abascal

```
# Buscamos las intervenciones
intervencionesAbascal <- regmatches(vectorTextoPlenos, regexec("El señor ABASCAL CONDE:\\s*(.*?)\\s*La señora PRESIDENTA:", vectorTextoPlenos))
# Eliminamos las filas vacias => character(0)
intervencionesAbascal <- intervencionesAbascal[!sapply(intervencionesAbascal, identical, character(0))]
# Nos quedamos con el segundo objeto de cada lista, porque contiene el texto de dentro sin los patrones de busqueda
intervencionesAbascal <- lapply(intervencionesAbascal , function(x) unlist(x)[2] )

# Filtramos el texto
intervencionesAbascal <- str_remove_all(intervencionesAbascal, regex("\\([^()]+\\)")) # Eliminacion de contenido entre parentesis
intervencionesAbascal <- str_remove_all(intervencionesAbascal, regex("\\— DEL DIPUTADO[^()]+\\?.")) # Eliminacion de otro tipo de anotación no relevante
intervencionesAbascal <- trimws(intervencionesAbascal) # Eliminamos espacios al principio y al final

sprintf("Nº de intervenciones de Abascal: %d ", length(intervencionesAbascal) )

## [1] "Nº de intervenciones de Abascal: 22 "

intervencionesAbascal_Sep <- intervencionesAbascal
intervencionesAbascal[1]

## [1] "Señor Sánchez, ¿cómo se atreve usted a hablarme de monólogos si siempre trae las respuestas escritas, si usted nunca contesta a mis preguntas? Conteste por lo menos hoy. ¿Qué va a hacer usted para impedir que VOX siga cruzando las líneas que dice usted que cruzamos? Conteste también lo que no me ha contestado durante toda esta legislatura: ¿por qué mintió a los españoles prometiéndoles que no pactaría con estos, con esos y con aquellos?  Contésteme a eso y entonces no habrá monólogos, habrá diálogo político, algo que nunca ha habido en esta Cámara por su culpa."

intervencionesAbascal <- paste(intervencionesAbascal, collapse = "") # Colapsamos la lista en un unico vector
```

Extraccion y procesamiento de las intervenciones completado

```
# Guardamos las intervenciones unificadas en un vector
save(intervencionesAbascal, intervencionesCasado, intervencionesSanchez, file = "intervenciones.rda")
# Guardamos las intervenciones separadas por intervencion (Lista de textos de las distintas intervenciones)
save(intervencionesAbascal_Sep, intervencionesSanchez_Sep, intervencionesCasado_Sep, file = "intervencionesSep.rda")

# Limpiamos memoria
rm(list = ls())
```

Analisis comparativo de las intervenciones

```
load("intervenciones.rda")
```

Hora de estructurar el texto a analizar. Para esta primera parte del análisis utilizaremos como unidad elemental de significado cada palabra y las analizaremos en conjunto. Debido a que hay muchas palabras que no tienen relevancia las filtraremos usando las stopwords del español de quanteda, añadiendo otras más obtenidas de gitHub y otras pocas añadidas por mí manualmente al ir haciendo este estudio.

```
extra_stopwords <- read.table("spanish.txt", encoding = "UTF-8") # Leo las stopwords descargadas de gitHub (En este archivo e incluido más manualmente)
colnames(extra_stopwords) <- c('word') # Renombro la columna a "word"
es_stopwords <- quanteda::stopwords("spanish") # Obtenemos las stopwords de quanteda
es_stopwords <- data.frame( "word" = es_stopwords) # Generamos un dataframe con las stopwords
es_stopwords <- union_all(es_stopwords, extra_stopwords) # Unimos los conjuntos de stopwords
sprintf("Nº de stopwords en español: %d", nrow(es_stopwords))
```

```
## [1] "Nº de stopwords en español: 930"
```

```
save(es_stopwords, file = "es_stopwords.rda") # Guardamos las stopwords
```

Generemos un dataframe con los tokens de las intervenciones para empezar a analizarlas.

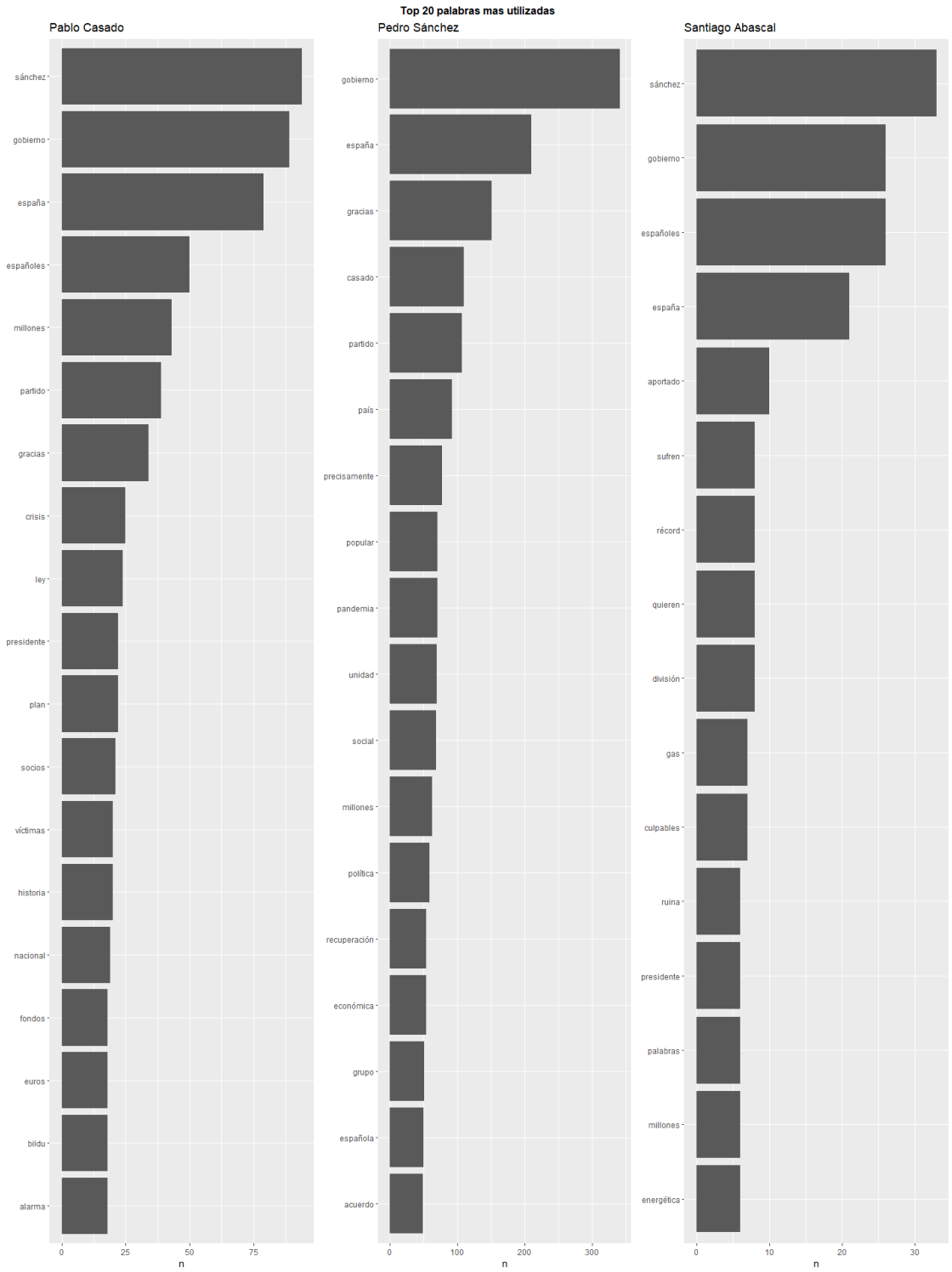
```
# Pablo Casado
Casado_df <- tibble( text = intervencionesCasado) # Convertimos el vector en un dataframe
tokens_Casado  <-   Casado_df %>%
                    unnest_tokens(word, text) %>% # Generamos las tokens del dataframe
                    anti_join(es_stopwords) # Eliminamos las stopwords

# Pedro Sanchez
Sanchez_df <- tibble( text = intervencionesSanchez)
tokens_Sanchez  <-  Sanchez_df %>%
                    unnest_tokens(word, text) %>%
                    anti_join(es_stopwords)

# Santiago Abascal
Abascal_df <- tibble( text = intervencionesAbascal)
tokens_Abascal  <-  Abascal_df %>%
                    unnest_tokens(word, text) %>%
                    anti_join(es_stopwords)
```

Análisis de las palabras más utilizadas

Primero hacemos un conteo de las palabras más utilizadas de cada orador para guardar el número de veces que se repite la 20º palabra. Así podremos fijar en ese número de repeticiones el gráfico y mostrar las 20 palabras más repetidas por orador.

```
nMin_Casado  <-   tokens_Casado %>%
        count(word, sort = TRUE)
nMin_Casado <- nMin_Casado[20,][2]
nMin_Sanchez  <-   tokens_Sanchez %>%
        count(word, sort = TRUE)
nMin_Sanchez <- nMin_Sanchez[20,][2]
nMin_Abascal  <-   tokens_Abascal %>%
        count(word, sort = TRUE)
nMin_Abascal <- nMin_Abascal[20,][2]
```

```r
C  <-   tokens_Casado %>%
        count(word, sort = TRUE) %>%
        filter(n > nMin_Casado$n) %>%
        mutate(word = reorder(word, n)) %>%
        ggplot(aes(n, word)) +
        geom_col() +
        labs(y = NULL, title = "Pablo Casado")

S  <-   tokens_Sanchez %>%
        count(word, sort = TRUE) %>%
        filter(n > nMin_Sanchez$n) %>%
        mutate(word = reorder(word, n)) %>%
        ggplot(aes(n, word)) +
        geom_col() +
        labs(y = NULL, title = "Pedro Sánchez")

A  <-   tokens_Abascal %>%
        count(word, sort = TRUE) %>%
        filter(n > nMin_Abascal$n) %>%
        mutate(word = reorder(word, n)) %>%
        ggplot(aes(n, word)) +
        geom_col() +
        labs(y = NULL, title = "Santiago Abascal")

palabras_mas_utilizadas <-  ggarrange(C, S, A,
                            ncol = 3, nrow = 1)

annotate_figure(
  palabras_mas_utilizadas,
  top = text_grob("Top 20 palabras mas utilizadas",
                  color = "black", face = "bold")
  )
```



Guardamos las tokens generadas de cada orador

```
save(tokens_Abascal,tokens_Casado,tokens_Sanchez, file = "tokens.rda")

# Limpiamos memoria
rm(list = ls())
```

Análisis sentimental del discurso de cada orador

Cargamos el diccionario en español para el 'Sentiment Analysis' y los tokens generados a partir de las intervenciones.

```
load("dictionary_kaggle_es.rda")
load("tokens.rda")
```

Construimos un dataframe para las palabras positivas y otro para las negativas.

```
positive_df <- data.frame( word = dict_kaggle_es$positive$V1, sentiment = 'positive')
negative_df <- data.frame( word = dict_kaggle_es$negative$V1, sentiment = 'negative')
```

Cruzamos los datos para saber cuales son las palabras positivas y negativas más utilizadas. También generamos gráficos para compararlas.

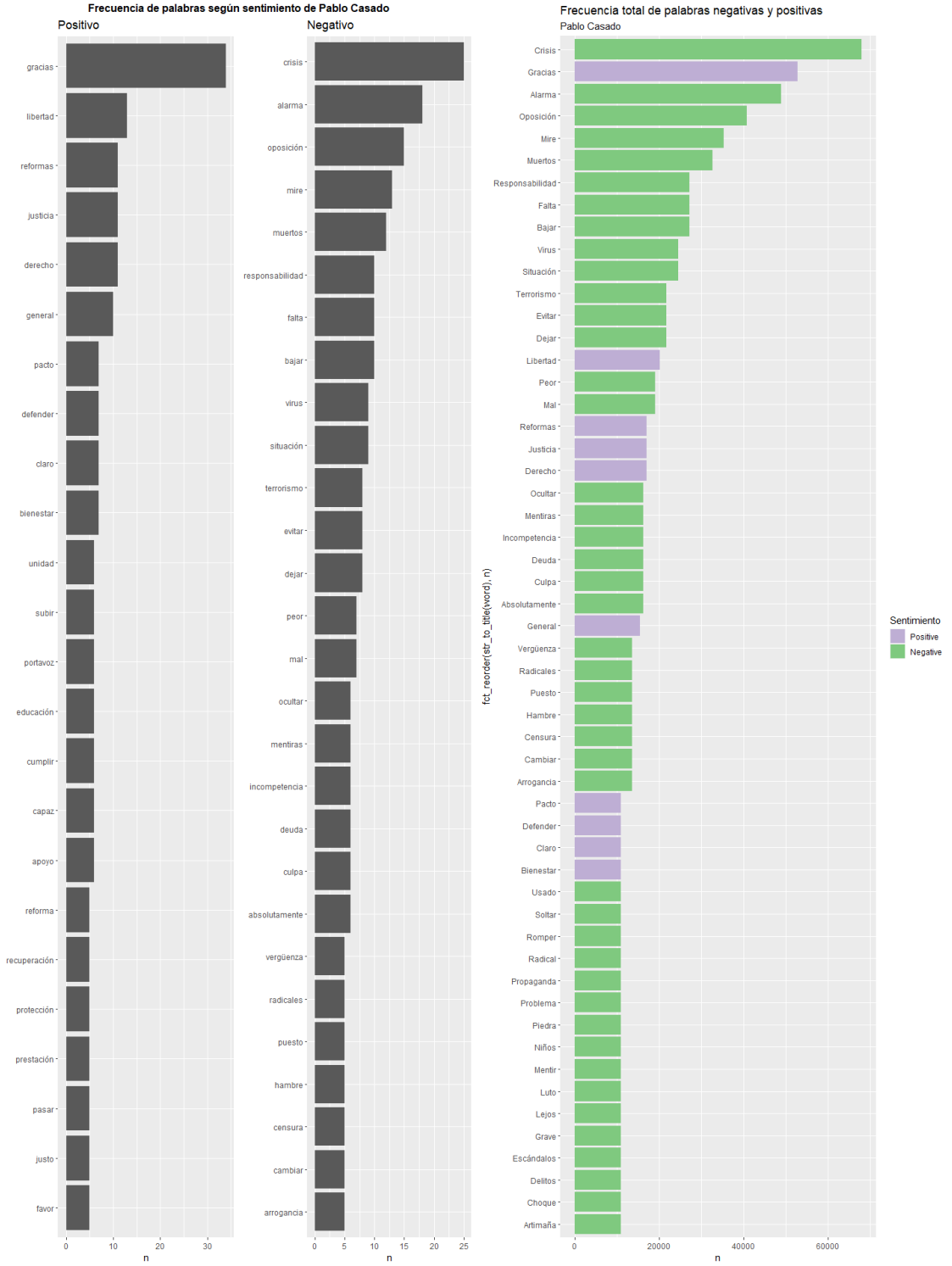
Pablo Casado

```
Casado_count_pos <-  tokens_Casado %>%          # Contamos las palabras positvas
                      semi_join(positive_df) %>%
                      count(word, sort = TRUE)
Casado_count_neg <-  tokens_Casado %>%          # Contamos las palabras negativas
                      semi_join(negative_df) %>%
                      count(word, sort = TRUE)

# Grafica de palabras positivas con mayor frecuencia
posCasado <-   Casado_count_pos %>%
                filter(n > Casado_count_pos[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Positivo")
# Grafica de palabras negativas con mayor frecuencia
negCasado <-   Casado_count_neg %>%
                filter(n > Casado_count_neg[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Negativo")

# Generamos grafica de positivas y negativas por separado
sent_Casado <-  ggarrange(posCasado, negCasado,
                          ncol = 2, nrow = 1)
a1 <- annotate_figure(
  sent_Casado,
  top = text_grob("Frecuencia de palabras según sentimiento de Pablo Casado",
                  color = "black", face = "bold")
  )

# Contamos palabras positivas y negativas conjuntamente
sent_dict <- union_all(positive_df, negative_df)
Casado_count_sent <- tokens_Casado %>%
                      inner_join(sent_dict) %>%
                      count(word, sentiment, sort = TRUE)

# Grafica de palabras positivas y negativas de mayor a menor frecuencia
b1 <- Casado_count_sent %>%
      filter(n > Casado_count_sent[60,]$n) %>%
      mutate(n = if_else(sentiment == "negative", n, n)) %>%
      ggplot(aes(fct_reorder(str_to_title(word), n), n, fill = str_to_title(sentiment)))
+
```

```
      geom_col() +
      coord_flip() +
      scale_fill_brewer(type = "qual") +
      guides(fill = guide_legend(reverse = TRUE)) +
      labs( title = "Frecuencia total de palabras negativas y positivas",
            subtitle = "Pablo Casado",
            y = "n",
            fill = "Sentimiento")
```

Gráfica de Frecuencias según sentimiento + Gráfica de palabras con más frecuencias y su sentimiento

```
ggarrange(a1, b1,
          ncol = 2, nrow = 1)
```



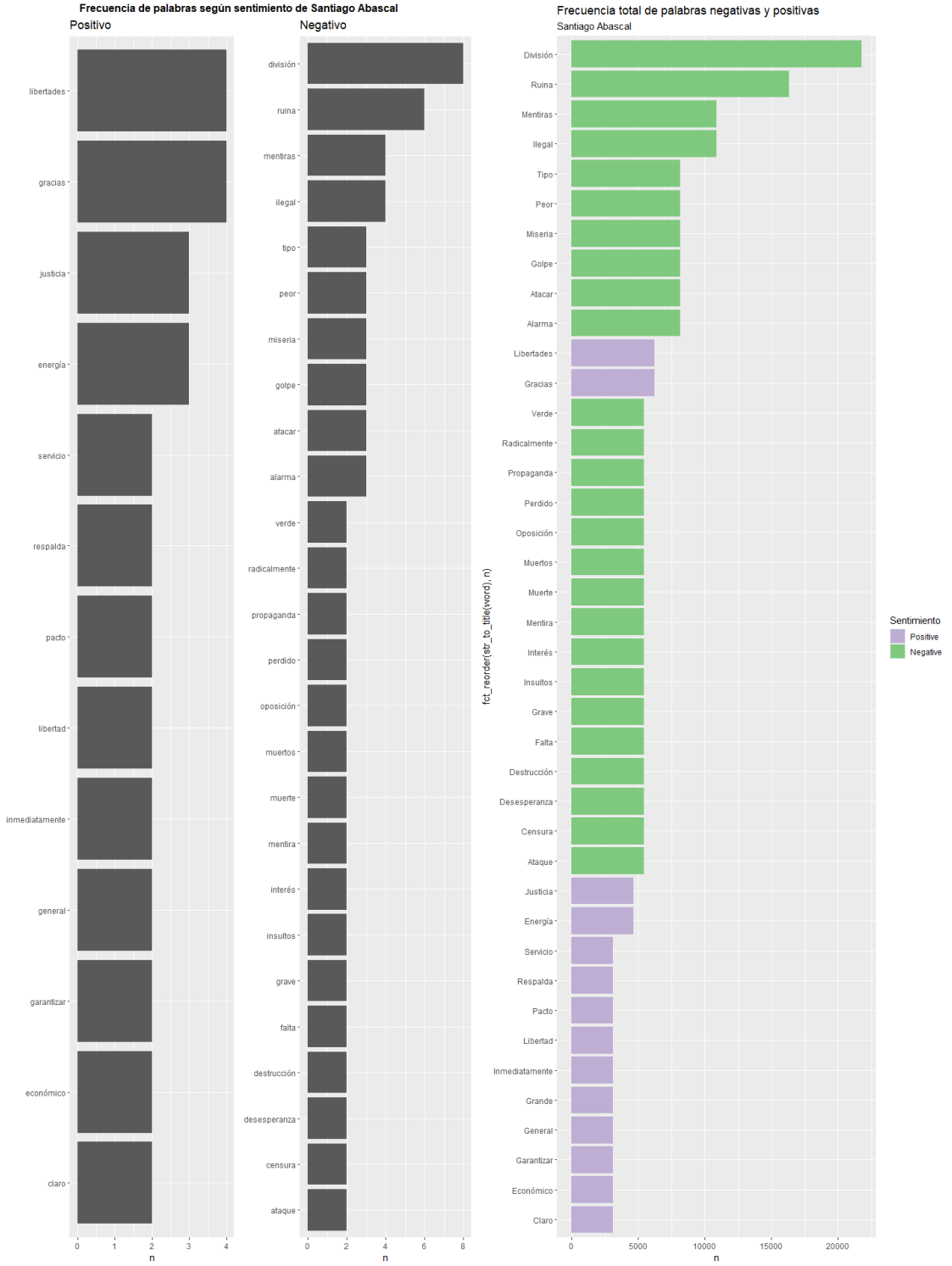
Santiago Abascal

```
Abascal_count_pos <-  tokens_Abascal %>%          # Contamos las palabras positvas
                      semi_join(positive_df) %>%
                      count(word, sort = TRUE)
Abascal_count_neg <-  tokens_Abascal %>%          # Contamos las palabras negativas
                      semi_join(negative_df) %>%
                      count(word, sort = TRUE)

# Grafica de palabras positivas con mayor frecuencia
posAbascal <-   Abascal_count_pos %>%
```

```
                filter(n > Abascal_count_pos[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Positivo")
# Grafica de palabras negativas con mayor frecuencia
negAbascal <-   Abascal_count_neg %>%
                filter(n > Abascal_count_neg[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Negativo")

# Generamos grafica de positivas y negativas por separado
sent_Abascal <-  ggarrange(posAbascal, negAbascal,
                           ncol = 2, nrow = 1)
a3 <- annotate_figure(
  sent_Abascal,
  top = text_grob("Frecuencia de palabras según sentimiento de Santiago Abascal",
                  color = "black", face = "bold")
  )

# Contamos palabras positivas y negativas conjuntamente
sent_dict <- union_all(positive_df, negative_df)
Abascal_count_sent <- tokens_Abascal %>%
                      inner_join(sent_dict) %>%
                      count(word, sentiment, sort = TRUE)

# Grafica de palabras positivas y negativas de mayor a menor frecuencia
b3 <- Abascal_count_sent %>%
      filter(n > Abascal_count_sent[60,]$n) %>%
      mutate(n = if_else(sentiment == "negative", n, n)) %>%
      ggplot(aes(fct_reorder(str_to_title(word), n), n, fill = str_to_title(sentiment)))
+
      geom_col() +
      coord_flip() +
      scale_fill_brewer(type = "qual") +
      guides(fill = guide_legend(reverse = TRUE)) +
      labs( title = "Frecuencia total de palabras negativas y positivas",
            subtitle = "Santiago Abascal",
            y = "n",
            fill = "Sentimiento")
```

Gráfica de Frecuencias según sentimiento + Gráfica de palabras con más frecuencias y su sentimiento

```
ggarrange(a3, b3,
          ncol = 2, nrow = 1)
```



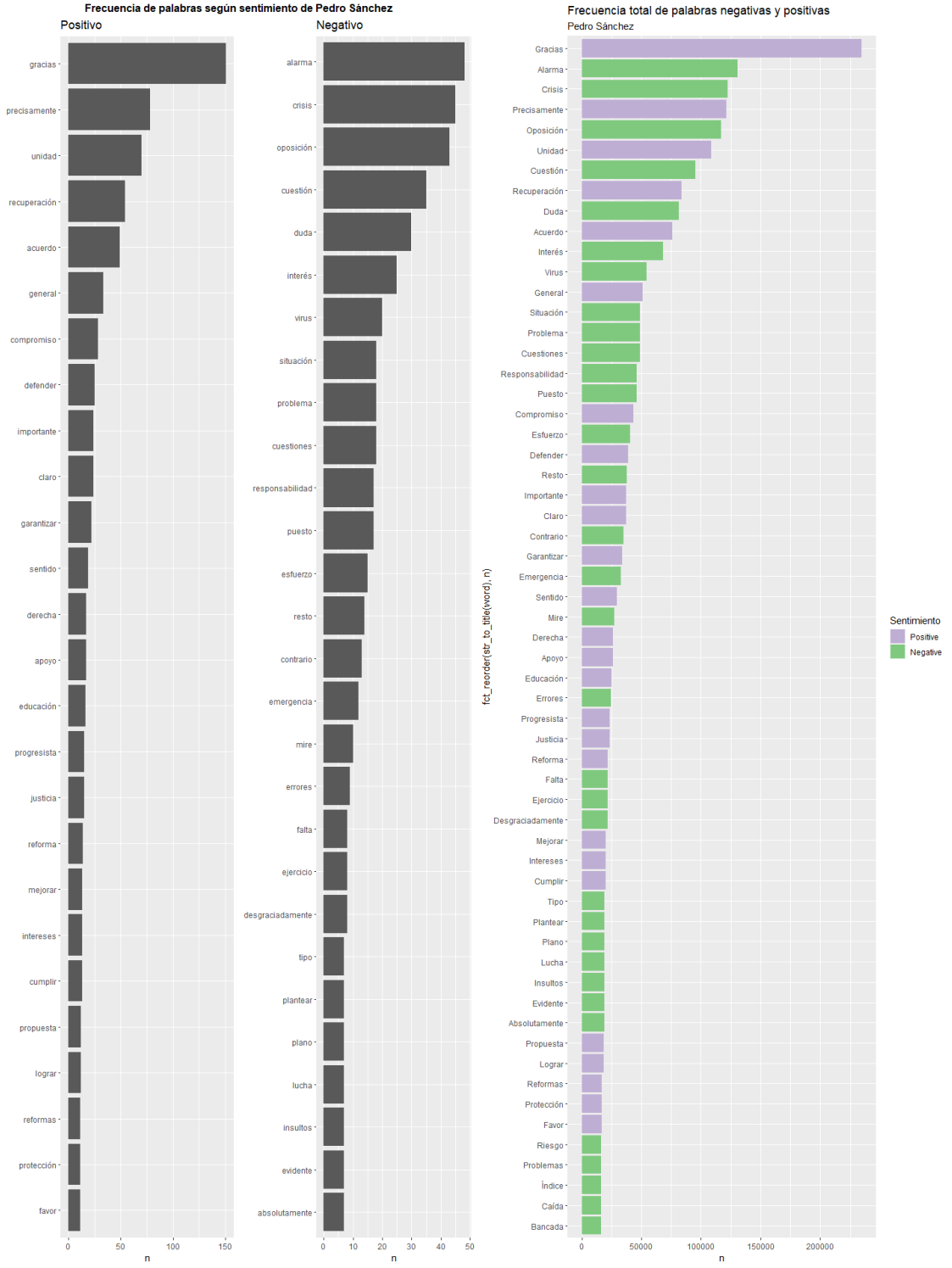
Pedro Sanchez

```
Sanchez_count_pos <-  tokens_Sanchez %>%          # Contamos las palabras positvas
                      semi_join(positive_df) %>%
                      count(word, sort = TRUE)
Sanchez_count_neg <-  tokens_Sanchez %>%          # Contamos las palabras negativas
                      semi_join(negative_df) %>%
                      count(word, sort = TRUE)

# Grafica de palabras positivas con mayor frecuencia
posSanchez <-    Sanchez_count_pos %>%
```

```
                filter(n > Sanchez_count_pos[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Positivo")
# Grafica de palabras negativas con mayor frecuencia
negSanchez <-   Sanchez_count_neg %>%
                filter(n > Sanchez_count_neg[30,]$n) %>%
                mutate(word = reorder(word, n)) %>%
                ggplot(aes(n, word)) +
                geom_col() +
                labs(y = NULL, title = "Negativo")

# Generamos grafica de positivas y negativas por separado
sent_Sanchez <-  ggarrange(posSanchez, negSanchez,
                           ncol = 2, nrow = 1)
a2 <- annotate_figure(
  sent_Sanchez,
  top = text_grob("Frecuencia de palabras según sentimiento de Pedro Sánchez",
                  color = "black", face = "bold")
  )

# Contamos palabras positivas y negativas conjuntamente
sent_dict <- union_all(positive_df, negative_df)
Sanchez_count_sent <- tokens_Sanchez %>%
                      inner_join(sent_dict) %>%
                      count(word, sentiment, sort = TRUE)

# Grafica de palabras positivas y negativas de mayor a menor frecuencia
b2 <- Sanchez_count_sent %>%
      filter(n > Sanchez_count_sent[60,]$n) %>%
      mutate(n = if_else(sentiment == "negative", n, n)) %>%
      ggplot(aes(fct_reorder(str_to_title(word), n), n, fill = str_to_title(sentiment)))
+
      geom_col() +
      coord_flip() +
      scale_fill_brewer(type = "qual") +
      guides(fill = guide_legend(reverse = TRUE)) +
      labs( title = "Frecuencia total de palabras negativas y positivas",
            subtitle = "Pedro Sánchez",
            y = "n",
            fill = "Sentimiento")
```

Gráfica de Frecuencias según sentimiento + Gráfica de palabras con más frecuencias y su sentimiento

```
ggarrange(a2, b2,
          ncol = 2, nrow = 1)
```




```
# Limpiamos memoria
rm(list = ls())
```

Analisis de la Frecuencia Inversa por Documento

En esta parte trataremos de calcular cuales son las palabras con más importancia en términos generales. Hasta ahora teníamos en cuenta cuantas veces se repetía una misma palabra a lo largo de todas las intervenciones, si esta se repetía mucho salía en una posición superior del gráfico. Pero en realidad, una palabra verdaderamente importante en un discurso no tiene por que repetirse demasiado, de hecho las palabras con verdadero poder no suelen hacerlo.

Por este motivo nos disponemos a calcular la frecuencia inversa de cada palabra por documento (tf-idf):

$$idf(term) = ln(\frac{n_{documents}}{n_{documentsContainingTerm}})$$

Esta fórmula nos ayudará a entender cuales son las palabras más significativas del discurso de cada orador.Utilizaremos como "documento" las distintas intervenciones.

Proceso

Cargamos la lista de textos de las distintas intervenciones

```
load("intervencionesSep.rda")
load("es_stopwords.rda")

length(intervencionesSanchez_Sep)
```

```
## [1] 176
```

Ahora calculamos las tf-idfs.

Pedro Sánchez

```
intervencionesSanchez_df <- data.frame(texto = intervencionesSanchez_Sep) # Gneramos un data frame con las intervenciones
intervencion <- c(1:nrow(intervencionesSanchez_df)) # Generamos una columna con los identificadores de cada intervencion
intervencionesSanchez_df <- cbind(intervencion, intervencionesSanchez_df) # Añadimos la columna intervenciones al df

Sanchez_words <- intervencionesSanchez_df %>%
  unnest_tokens(word, texto) %>% # Generamos los tokens por palabras de cada intervencion
  anti_join(es_stopwords) %>% # Eliminamos stop words
  count(intervencion, word, sort = TRUE) # Ordenamos

# Agrupamos por intervencion
total_Sanchez_words <- Sanchez_words %>%
  group_by(intervencion) %>%
  summarize(total = sum(n))

Sanchez_words <- left_join(Sanchez_words, total_Sanchez_words)

Sanchez_tf_idf <- Sanchez_words %>%
  bind_tf_idf(word, intervencion, n) %>% # Calculamos las tf-idf de cada token (palabra)
  select(-total) %>%
  arrange(desc(tf_idf)) # Ordenamos de mayor a menor tf-idf

# Generamos la grafica
Sanchez_tf_idf_graf <-    Sanchez_tf_idf %>%
                          filter(tf_idf > Sanchez_tf_idf[30,]$tf_idf) %>%
                          mutate(word = reorder(word, tf_idf)) %>%
                          ungroup() %>%
                          ggplot(aes(tf_idf, word)) +
                          geom_col() +
                          labs(y = NULL, title = "Pedro Sánchez")
```

Pablo Casado

```
intervencionesCasado_df <- data.frame(texto = intervencionesCasado_Sep) # Gneramos un
data frame con las intervenciones
intervencion <- c(1:nrow(intervencionesCasado_df)) # Generamos una columna con los
identificadores de cada intervencion
intervencionesCasado_df <- cbind(intervencion, intervencionesCasado_df) # Añadimos la
columna intervenciones al df

Casado_words <- intervencionesCasado_df %>%
  unnest_tokens(word, texto) %>% # Generamos los tokens por palabras de cada intervencion
  anti_join(es_stopwords) %>% # Eliminamos stop words
  count(intervencion, word, sort = TRUE) # Ordenamos

# Agrupamos por intervencion
total_Casado_words <- Casado_words %>%
  group_by(intervencion) %>%
  summarize(total = sum(n))

Casado_words <- left_join(Casado_words, total_Casado_words)

Casado_tf_idf <- Casado_words %>%
  bind_tf_idf(word, intervencion, n) %>% # Calculamos las tf-idf de cada token (palabra)
  select(-total) %>%
  arrange(desc(tf_idf)) # Ordenamos de mayor a menor tf-idf

# Generamos la grafica
Casado_tf_idf_graf <-   Casado_tf_idf %>%
                        filter(tf_idf > Casado_tf_idf[30,]$tf_idf) %>%
                        mutate(word = reorder(word, tf_idf)) %>%
                        ungroup() %>%
                        ggplot(aes(tf_idf, word)) +
                        geom_col() +
                        labs(y = NULL, title = "Pablo Casado")
```

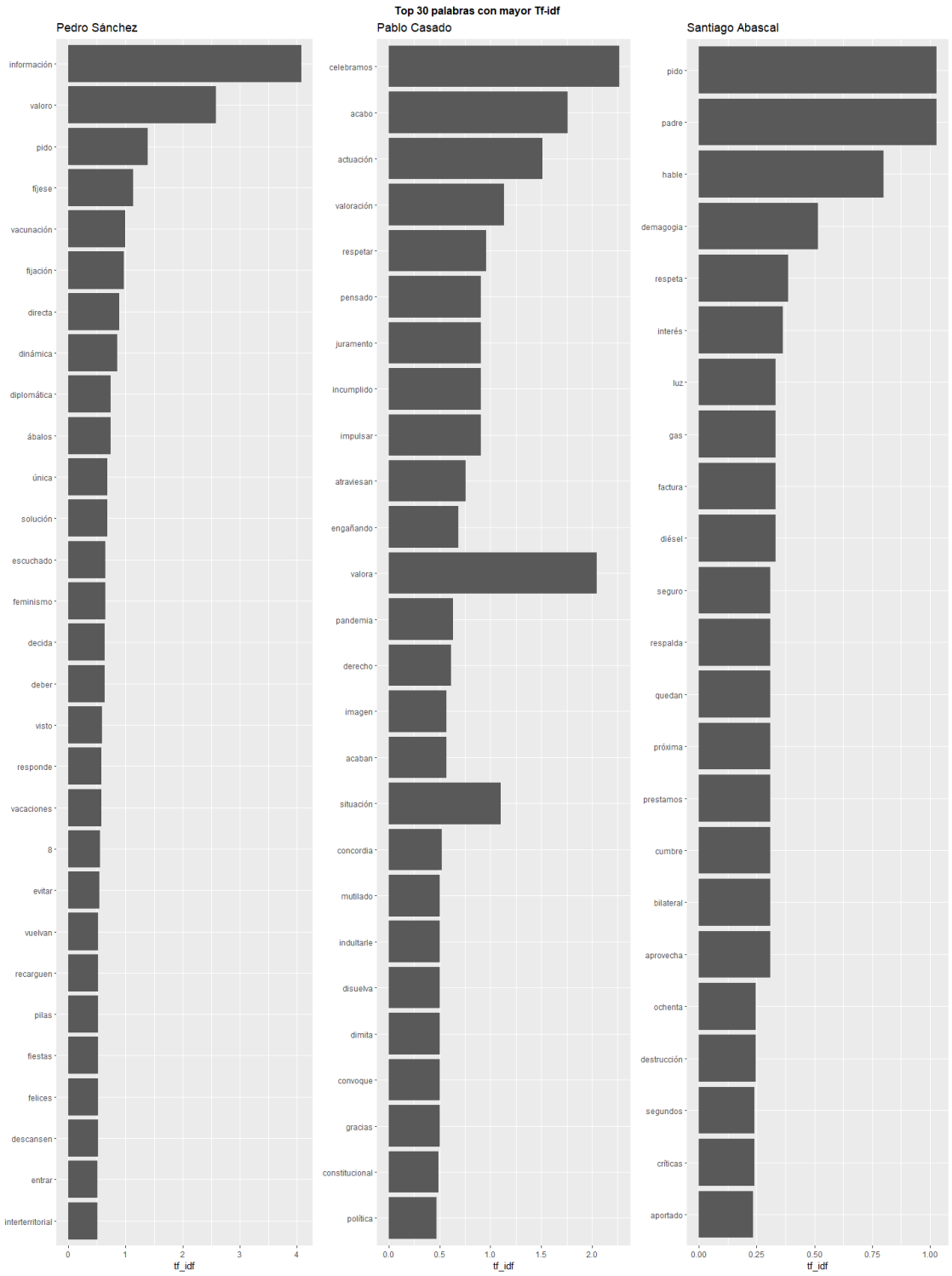
Santiago Abascal

```
intervencionesAbascal_df <- data.frame(texto = intervencionesAbascal_Sep) # Gneramos un
data frame con las intervenciones
intervencion <- c(1:nrow(intervencionesAbascal_df)) # Generamos una columna con los
identificadores de cada intervencion
intervencionesAbascal_df <- cbind(intervencion, intervencionesAbascal_df) # Añadimos la
columna intervenciones al df

Abascal_words <- intervencionesAbascal_df %>%
  unnest_tokens(word, texto) %>% # Generamos los tokens por palabras de cada intervencion
  anti_join(es_stopwords) %>% # Eliminamos stop words
  count(intervencion, word, sort = TRUE) # Ordenamos

# Agrupamos por intervencion
total_Abascal_words <- Abascal_words %>%
  group_by(intervencion) %>%
  summarize(total = sum(n))

Abascal_words <- left_join(Abascal_words, total_Abascal_words)

Abascal_tf_idf <- Abascal_words %>%
```

```
  bind_tf_idf(word, intervencion, n) %>% # Calculamos las tf-idf de cada token (palabra)
  select(-total) %>%
  arrange(desc(tf_idf)) # Ordenamos de mayor a menor tf-idf

# Generamos la grafica
Abascal_tf_idf_graf <-    Abascal_tf_idf %>%
                          filter(tf_idf > Abascal_tf_idf[30,]$tf_idf) %>%
                          mutate(word = reorder(word, tf_idf)) %>%
                          ungroup() %>%
                          ggplot(aes(tf_idf, word)) +
                          geom_col() +
                          labs(y = NULL, title = "Santiago Abascal")
```

Gráfica conjunta de las frecuencias inversas

```
tf_idfs <-  ggarrange(Sanchez_tf_idf_graf, Casado_tf_idf_graf, Abascal_tf_idf_graf,
                            ncol = 3, nrow = 1)
annotate_figure(
  tf_idfs,
  top = text_grob("Top 30 palabras con mayor Tf-idf",
                  color = "black", face = "bold")
  )
```




```
# Limpiamos memoria
rm(list = ls())
```

Bibliografía

Especiales agradecimientos a los siguientes rincones de internet, sin la información expuesta en ellos este estudio no habría sido posible.

- Text Mining with R
- Sentiment analysis with tidytext (R case study, 2021) de John Little

CONCLUSIONES Y OBSERVACIONES

Observación:

Antes de pararnos a escudriñar los resultados debemos tener en cuenta las limitaciones de este estudio. Mi ordenador tienen 8GB de RAM y un procesador Intel Core i5 de 8º generación (1.60GHz), no es un ordenador potente. Por este motivo he estado limitado en el número de PDFs a analizar. Si quería que los procesos no fuesen demasiado largos y que mi ordenador no explotase tenía que limitarme a un número de PDFs modesto (62 en este caso). Sin embargo, invito a quien quiera y pueda a ejecutar este código con un mayor numero de PDFs de Plenos, los resultados serán mucho más precisos y reveladores. Es tan fácil como aumentar el número de páginas a recorrer en el programa Python del scrapeo del Congreso, y obtendrá un número mayor de PDFs para analizar.

Yo ejecute el programa de scrapeo con 5 páginas y obtuve 62 PDFs. En los cuales había:

- 93 Intervenciones de Pablo Casado
- 176 Intervenciones de Pedro Sánchez
- 22 Intervenciones de Santiago Abascal

El número de intervenciones es proporcional a la exactitud de los resultados. Por eso creo que al tener un número de intervenciones significativamente bajo en el caso de Santiago Abascal sus resultados no son del todo veraces.

En cuanto al análisis sentimental he utilizado el diccionario en español de Kaggle, al cual le veo algunas fallas como considerar la palabra “Derecha” positiva y demás. Yo no puedo ponerme a construir un diccionario de análisis sentimental en español o este estudio no terminaría nunca. El resultado obtenido del cálculo de frecuencias inversas por documento creo que deja mucho que desear, pero el problema es que no se como mejorarlo.

Por último aclarar que el caso de estudio es político, por lo que no cabe duda de que mis conclusiones pueden verse afectadas por mi ideología o punto de vista social. Al fin y al cabo, cualquier conclusión de este estudio esta sujeta a la subjetividad del analista.

Conclusiones:

Palabras más utilizadas

Tanto Santiago Abascal como Pablo Casado dirigen su discurso prácticamente a Pedro Sánchez, ambos tienen como palabra más utilizada “Sánchez”. Está claro que los dos tratan de dialogar con él y sus intervenciones tienen como principal objetivo nuestro presidente.

Llama la atención el pequeño matiz que diferencia la tercera y cuarta palabra entre Pablo Casado y Santiago Abascal. La palabra “España” se encuentra en el tercer puesto de Casado y cuarto puesto de Abascal, mientras que la palabra “Españoles” es la cuarta de Casado y tercera de Abascal. Creo que esto no es ninguna coincidencia. En mi opinión esto denota matices en las ideas que quieren comunicar estos partidos.

El Partido Popular se dirige a España como nación, como territorio soberano, como país… Quiere hacer llegar un mensaje de gobernanza, de autoridad, como si estuvieran dirigiéndose al país que gobiernan. Por otro lado, VOX trata de dirigirse al individuo, a la persona que habita territorio español, a los españoles… Con esto, en mi opinión, intentan que su mensaje sea más personal, que la persona que lo escuche sienta y de algún modo se vea incitada a la reflexión.

El Partido Popular ha sido y es un partido fuerte, cabeza de la oposición. Tienen que reflejar una imagen de poder y capacidad de asumir un gobierno. Sin embargo, VOX es un partido que está cogiendo fuerzas, tiene que proyectar una imagen más individual, de captación de votos y reformista.

Además, la palabra que sigue a "Españoles", en el Top de Casado, es "Millones", mientras que en el de Abascal casi no se encuentra esta palabra. Esto induce a pensar que muchas veces que Casado emplea la palabra "Españoles" lo hace anteponiendo la palabra "Millones": "Millones de Españoles" que es casi lo mismo que decir "España". En el caso de Abascal esto no ocurre. Mensaje a la nación contra mensaje personal.

Otra observación interesante es la ausencia de la palabra "Gracias" en el discurso de Abascal.

En cuanto a las palabras utilizadas por Sánchez se pueden observar distintas curiosidades. La palabra españoles no está en su Top 20, tiene "España" en segundo puesto y "Española" en el décimo séptimo. Contiene palabras que no tienen sus opositores como: "País", "Precisamente" (Que intuyo que será una muletilla a la hora de afrontar sus defensas y debates), "acuerdo" y otras. Las palabras más mencionadas por Sánchez dan a entender que se defiende, trata de dirigir su discurso a términos generales de España y su nación ("unidad", "recuperación", "social", etc).

Análisis de sentimientos

Pablo casado encabeza su lista de palabras con significado sentimental con la palabra negativa "Crisis", la cual saca una considerable diferencia a la segunda palabra "Gracias". Es su gráfico dominan las palabras negativas respecto a las positivas. Nos indica una postura de ataque e inconformismo respecto al gobierno en sus intervenciones en el Congreso de los Diputados.

Santiago Abascal no es una excepción, de hecho en su caso son aun más acentuadas las palabras negativas. Su primera palabra positiva, "Libertades", está en la décimo primera posición. En mi opinión Abascal utiliza palabras con un carácter negativo muy acentuado, más que las de Casado. "Mentiras", "Ruina", "Peor", "Miseria"; son de sus palabras más utilizadas.

En cambio, la gráfica de Pedro Sánchez no se parece a la de sus compañeros políticos. La suya es mucho más homogénea. Las palabras negativas y positivas iteran como si estuviese casi planeado. Es el único que encabeza la lista con una palabra positiva: "Gracias", la cual se repite casi el doble que la segunda, esta negativa, "Alarma". Es sin duda el que más palabras positivas utiliza y sus palabras negativas son ligeras, nada que ver con las de Abascal.

Frecuencia inversa de palabras por documento

Como datos interesantes:

- La palabra con más peso para Abascal es "Pido", la de Casado "Celebramos" y la de Sánchez "Información".
- Vemos como algunas de las palabras con más tf-idf de Abascal tienen connotaciones negativas o son fáciles de emplear en contextos hirientes como: "demagogia", "respeta", "ínteres", "padre", "factura" y más. Sin embargo a su compañero diputado Pablo Casado le ocurre lo mismo, con palabras como: "incumplido", "juramento", "engañado", "atraviesan", …
- Me hace gracia ver que la segunda palabra más significativa de Casado es "Acabo".
- Pedro Sánchez tiene en tercer lugar "Pido" y en segundo "Valoro". Aquí si que se aprecia que la inmensa mayoría de palabras son positivas y me atrevería a decir populistas, que agradan a cualquiera: "dinámica", "diplomática", "solución", "escuchado", "deber", "feminismo", "vacaciones", "recarguen" y bastantes más.