# Minería de Texto en R – WordCloud

Para los análisis de minería de texto, se recomienda que las fuentes se encuentren dentro de archivos planos (.txt), y se recomienda que su codificación de caracteres sea UTF-8, ya que algunas funciones no trabajan adecuadamente con otras codificaciones. En el presente laboratorio se le facilitan 2 archivos de texto:

- ***SteveJobsStanford.txt:*** Texto del discurso de Steve Jobs en la universidad de Stanford. Se encuentra en idioma inglés.

- ***LGSTraspasoPresidencial.txt:*** Texto del discurso del presidente Luis Guillermo Solís durante la ceremonia de traspaso de poderes. Se encuentra en idioma español.

- ***CATraspasoPresidencial.txt:*** Texto del discurso del presidente Luis Guillermo Solís durante la ceremonia de traspaso de poderes. Se encuentra en idioma español.

## Paso 1

Instalar las librerías que se requieren para poder realizar este tipo de análisis:

```
# Instalar
install.packages("tm")# para mineria de texto
install.packages("SnowballC")# para text stemming
install.packages("wordcloud")# generador de word-cloud
install.packages("RColorBrewer")# paletas de color
install.packages("sm")# stopwords
install.packages("devtools")# para poder cargar la siguiente librería desde
un repo dfirente a CRAN
library("devtools")
devtools::install_github("lchiffon/wordcloud2")
```

## Paso 2

Cargar las librerías recién instaladas para que sus funcionalidades estén disponibles durante los análisis de texto.

```
# Cargar
library("tm")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")
library("sm")
```

## Paso 3

Se debe cargar el texto a R, para lo cual se utilizará la función Corpus(), de la librería tm (text mining). Corpus crea una lista de documentos, sin embargo, para este laboratorio solamente contendrá uno a la vez.

Para iniciar correremos el siguiente código. El cual abrirá una ventana de selección, en la cual se debe escoger el archivo que se va a utilizar. Para el primer caso se iniciará con el discurso de Steve Jobs, por lo tanto, éste será el archivo que se debe seleccionar.

```
text <- readLines(file.choose())
```

En caso que el texto que se quiera analizar se encuentre en una ruta que pueda accederse por una URL, se podría utilizar el siguiente código:

```
# Leer el arhcivo de texto de internet
filePath <- http://www.example.com/speech.txt

text <- readLines(filePath)
```

Una vez que ya se tiene el texto, se debe leer como Corpus:

```
# Cargar los datos como un Corpus
docs <- Corpus(VectorSource(text))
```

Si requiere visualizar el contenido del archivo, puede utilizar el siguiente comando:

```
# Revisar el contenido
inspect(docs)
```

Después de ejecutar este comando puede notar lo siguiente en el texto proporcionado, cada párrafo se encuentra separado por un identificador o número de párrafo, es por esto que la variable que se creó anteriormente llamada *text* indicaba que contenía una lista de X objetos, los cuales corresponden uno a cada párrafo encontrado en el archivo.

## Paso 4

Lo primero que debe realizarse una vez que se tiene el texto en memoria es poder realizar algunas transformaciones al texto para lo cual se usará la función tm_map().

Entre las transformaciones que se pueden realizar se encuentra el poder reemplazar algunos caracteres especiales con espacios en blanco para poder realizar un análisis más eficiente y acertado del texto. Para eso usaremos el siguiente código:

```
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))

docs <- tm_map(docs, toSpace, "/")
docs <- tm_map(docs, toSpace, "@")
docs <- tm_map(docs, toSpace, "\\|")
```

## Paso 5

Una vez que se han realizado algunas transformaciones sobre el texto, conviene realizar algunos pasos de limpieza al texto.

Para este paso, se utilizará la misma función tm_map(), la cual también puede ser utilizada para eliminar espacios innecesarios dentro del texto, además poder convertir todo el texto a minúsculas, y también para eliminar todos aquellos *stopwords* como lo es "we".

Los *stopwords*, son aquellas palabras que se encuentran dentro del texto, pero que en realidad no tienen ningún aporte de relevancia para el análisis. Entre estos, se encuentran algunos monosílabos, o palabras propias de una región. Para esto, ya existen algunas listas de palabras predeterminadas por idioma, y además se pueden realizar listados personalizados.

Para realizar este tipo de limpiezas se podrían utilizar algunos de los siguientes comandos. Revise lo que se encuentra en cada uno antes de realizar la ejecución del mismo, y realice los cambios necesarios para que el código tenga sentido al problema que se está tratando de resolver:

```
# Convertir el texto en lower case
docs <- tm_map(docs, content_transformer(tolower))

# Eliminar numeros
docs <- tm_map(docs, removeNumbers)

# Eliminar stopwords en ingles
docs <- tm_map(docs, removeWords, stopwords("english"))

# Eliminar stopwords personalizados
docs <- tm_map(docs, removeWords, c("blabla1", "blabla2"))

# Eliminar puntuaciones
docs <- tm_map(docs, removePunctuation)

# Elimnar espacios extra en blanco
docs <- tm_map(docs, stripWhitespace)

# Text stemming
# docs <- tm_map(docs, stemDocument)
```

## Paso 6

Se debe crear una tabla matriz donde se encuentra la frecuencia de aparición de cada una de las palabras. Para realizar estos pasos se puede ejecutar el siguiente código:

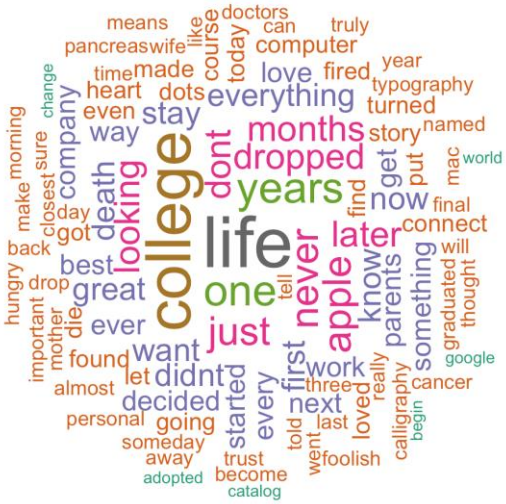
```
dtm <- TermDocumentMatrix(docs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)
```

## Paso 7

Una vez que se han realizado todos los pasos de limpieza y transformación al texto, viene el momento de ilustrar el WordCloud. Para esto se usarán los siguientes comandos:

```
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.words=100,
random.order=FALSE, rot.per=0.35, colors=brewer.pal(8, "Dark2"))
```

Una vez que se ha presentado el diagrama, visualice las palabras que tienen un mayor tamaño, estas justamente son las que más se mencionaron dentro del texto analizado.




Para comprender mejor la función que se utilizó, se presenta el siguiente cuadro donde se explican un poco más los parámetros que se utilizaron.

| | |
|---|---|
| words | Las palabras que se van a representar. |

| | |
|---|---|
| freq | Su frecuencia. |
| min.freq | Palabras que se encuentren inferior a esta frecuencia no se representarán en el gráfico. |
| max.words | Máxima cantidad de palabras a representar. |
| random.order | Presentar las palabras en orden aleatorio. Si es falso, se presentarán en orden decreciente a la frecuencia. |
| rot.per | Proporción de rotación de las palabras. |
| colors | Utilizar colores para las frecuencias. Se puede decir un único color si se desea que no se muestren múltiples. |

## Paso 8

Se puede realizar un análisis adicional para poder visualizar algunos detalles de las frecuencias de las palabras encontradas. Para eso con el siguiente código se buscará las palabras que al menos tengan una frecuencia de 4:

```
findFreqTerms(dtm, lowfreq = 4)
```

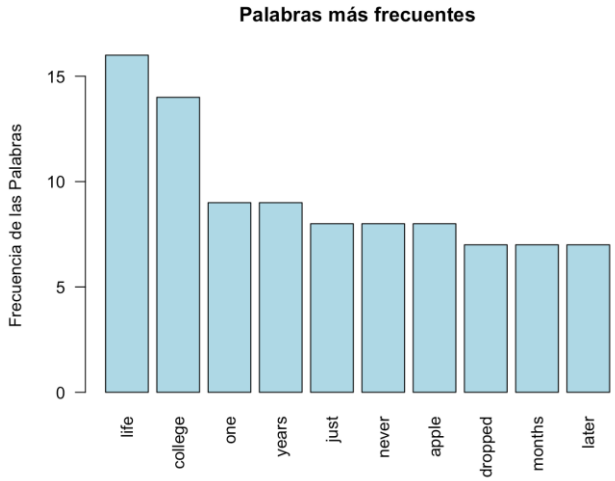
Además, se puede realizar un análisis de asociación entre las frecuencias de los términos (correlación entre los términos), para esto, se utiliza la función findAssocs(). En el siguiente código se analiza la asociación de los términos con la palabra "*college*":

```
findAssocs(dtm, terms = "college", corlimit = 0.3)
```

## Paso 9

Además, se puede realizar un análisis de las 10 palabras con más apariciones dentro del texto, para eso se podrá utilizar el siguiente código:

```
barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word, col ="lightblue", main
="Palabras más frecuentes", ylab = "Frecuencia de las Palabras")
```


Palabras más frecuentes
Frecuencia de las Palabras
0
5
10
15
life
college
one
years
just
never
apple
dropped
months
later

# Laboratorio WordCloud

## Parte 1

Correr nuevamente el ejercicio para el archivo ***LGSTraspasoPresidencial.txt***, es importante que tome en cuenta que éste se encuentra en español, por lo que a la hora de correr algunos comandos debe realizar algunos cambios, además la representación será un poco diferente, pues para realizar los cálculos, en muchos casos los caracteres especiales se eliminan.

Una vez que haya ejecutado el laboratorio, documente los cambios que tuvo que realizar a los comandos y además por medio de imágenes, muestre los resultados obtenidos.

## Parte 2

Como una evolución de los WordCloud, se lanzó una librería llamada wordcloud2 que ofrece opciones avanzadas de visualización.

Para poder ejecutar esta segunda parte del laboratorio se debe instalar dicha librería y ejecutar los siguientes comandos. Para eso se utilizará un dataset de ejemplo que viene incluido con R, sin embargo, si desea utilizar los dataset creados durante el ejercicio, debe sustituir el parámetro demoFreq por d.

Como prueba de la realización de esta parte, se le solicita que documente por medio de imágenes los resultados obtenidos de cada una de las ejecuciones, y una breve explicación de las características de cada una de las visualizaciones. Puede utilizar cualquiera de los 2 archivos de texto que se le suministraron, o simplemente correr con el dataset de ejemplo mencionado.

- wordcloud2(demoFreq, size = 1,shape = 'star')
- wordcloud2(demoFreq, size = 1.0)
- wordcloud2(demoFreq, size = 0.5)
- wordcloud2(demoFreq, size = 0.5, color = "random-light")
- wordcloud2(demoFreq, size = 0.5, backgroundColor = "black")
- wordcloud2(demoFreq, size = 0.5, minRotation = -pi/2, maxRotation = -pi/2)
- wordcloud2(demoFreq, size = 0.5, minRotation = -pi/6, maxRotation = -pi/6, rotateRatio=1)
- wordcloud2(demoFreq, size = 0.5, shape = "diamond")
- wordcloud2(demoFreq, size = 0.5, ellipticity = 0.1)
- wordcloud2(demoFreq, size = 0.5, ellipticity = 5.0)
- wordcloud2(demoFreq, color = "random-light", backgroundColor = "grey")
- letterCloud(demoFreq, word = "R", size = 2)
- letterCloud(demoFreq, word = "WORDCLOUD2", wordSize = 1)
- wordcloud2(demoFreqC, size = 2, fontFamily = "微软雅黑", color = "random-light", backgroundColor = "grey")

Si se desea guardar los resultados de las ejecuciones de wordcloud2 ya sea en un PDF o en formato HTML, se puede utilizar el siguiente código:

```
install.packages("webshot")
library(webshot)
webshot::install_phantomjs()

# Crear cualquiera de los graficos que se quiera guardar
my_graph=wordcloud2(demoFreq, size=1.5)

# Guardarlo en HTML
library("htmlwidgets")
saveWidget(my_graph,"tmp.html",selfcontained = F)

# Guardarlo en PDF, como se puede visualizar se requiere el HTML primero
webshot("tmp.html","fig_1.pdf", delay =5, vwidth = 480, vheight=480)
```

## Parte 3

A continuación, se muestra el código usando la librería *shiny* (visualizaciones dinámicas), la cual permite cambiar el tamaño de la visualización en tiempo de ejecución.

Se recomienda que corra el código y que documente por medio de imágenes y una breve explicación, los resultados obtenidos.

```
if(require(shiny)){
  library(wordcloud2)
   # Global variables can go here
   n <- 1


   # Define the UI
   ui <- bootstrapPage(
      numericInput('size', 'Size of wordcloud', n),
     wordcloud2Output('wordcloud2')
   )



   # Define the server code
```

```
server <- function(input, output) {
   output$wordcloud2 <- renderWordcloud2({
   # wordcloud2(demoFreqC, size=input$size)
     wordcloud2(demoFreq, size=input$size)
   })
}
# Return a Shiny app object
# Sys.setlocale("LC_CTYPE","chs") #if you use Chinese character
## Do not Run!
shinyApp(ui = ui, server = server)
}
```

## Parte 4

En los siguientes ejercicios, se muestra como por medio de la función wordcloud2 se pueden realizar algunas visualizaciones utilizando una silueta de imagen como template para la forma que se desea tenga la nube de palabras.

Para realizar estos ejercicios asegurese de contar con las figuras que se le suministraron y que toda se encuentren almacenadas en el directorio de trabajo de R.

Para efectos del laboratorio, debe correr los scripts y mostrar por medio de imágenes y una breve explicación los resultados obtenidos.

- wordcloud2(demoFreq, figPath = "twitter.png", size = 1.5, color = "skyblue")
- wordcloud2(demoFreq, figPath = "costarica.png", size = 1.5, color = "random-dark")
- wordcloud2(demoFreq, figPath = "apple.png", size = 1, color = "black")