The five main steps for creating a word cloud using R software

Step 1: Create a text file

In the following examples, I'll process the "**I have a dream speech**" from "**Martin Luther King**" but you can use any text you want :

- Copy and paste the text in a plain text file (e.g : ml.txt)
- Save the file

Note that, the text should be saved in a plain text (.txt) file format using your favorite text editor.

Step 2 : Install and load the required packages

text mining and **wordcloud** packages are required.

They can be installed and loaded using the R code below :

```
# Install
install.packages("tm")  # for text mining
install.package("SnowballC") # for text stemming
install.packages("wordcloud") # word-cloud generator
install.packages("RColorBrewer") # color palettes


# Load
library("tm")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")
```

Step 3 : Text mining

load the text

The text is loaded using **Corpus()** function from **text mining** (tm) package. Corpus is a list of a document (in our case, we only have one document).

In the example below, I loaded a .txt file hosted on STHDA website but you can use one from your computer.

```
# Read the text file
filePath <- "http://www.sthda.com/sthda/RDoc/example-files/martin-luther-king-i-have-a-dream-speech.txt"
```

```
text <- readLines(filePath)

# Load the data as a corpus
docs <- Corpus(VectorSource(text))
```

VectorSource() function creates a corpus of character vectors

The content of the document can be inspected as follow :

```
inspect(docs)
```

Text transformation

Transformation is performed using **tm_map()** function to replace, for example, special characters from the text.

Replacing “/”, “@” and “|” with space

```
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "/")
docs <- tm_map(docs, toSpace, "@")
docs <- tm_map(docs, toSpace, "\\|")
```

Cleaning the text

the **tm_map()** function is used to remove unnecessary white space, to convert the text to lower case, to remove common stopwords like ‘the’, “we”.

The information value of ‘stopwords’ is near zero due to the fact that they are so common in a language. Removing this kind of words is useful before further analysis. For ‘stopwords’, supported languages are danish, dutch, english, finnish, french, german, hungarian, italian, norwegian, portuguese, russian, spanish and swedish. Language names are case sensitive.

I’ll also show you how to make your own list of stopwords to remove from the text.

You could also remove numbers and punctuation with **removeNumbers** and **removePunctuation** arguments.

Another important preprocessing step is to make a **text stemming** which reduces words to their root form. In other words, this process removes suffixes from words to make it simple and to get the common origin. For example, a stemming process reduces the words “moving”, “moved” and “movement” to the root word, “move”.

Note that, text stemming require the package ‘SnowballC’.

The R code below can be used to clean your text :

```
# Convert the text to lower case
docs <- tm_map(docs, content_transformer(tolower))

# Remove numbers
```

```
docs <- tm_map(docs, removeNumbers)

# Remove english common stopwords
docs <- tm_map(docs, removeWords, stopwords("english"))

# Remove your own stop word
# specify your stopwords as a character vector
docs <- tm_map(docs, removeWords, c("blabla1", "blabla2"))

# Remove punctuations
docs <- tm_map(docs, removePunctuation)

# Eliminate extra white spaces
docs <- tm_map(docs, stripWhitespace)

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

Step 4 : Build a term-document matrix

Document matrix is a table containing the frequency of the words. Column names are words and row names are documents. The function *TermDocumentMatrix()* from **text mining** package can be used as follow :

```
dtm <- TermDocumentMatrix(docs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)
          word freq
will       will   17
freedom   freedom   13
ring       ring   12
day         day   11
dream      dream   11
let         let   11
every      every    9
able       able    8
one         one    8
together together    7
```

Step 5 : Generate the Word cloud

The importance of words can be illustrated as a **word cloud** as follow :

```
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
        max.words=200, random.order=FALSE, rot.per=0.35,
        colors=brewer.pal(8, "Dark2
```

The above **word cloud** clearly shows that “Will”, “freedom”, “dream”, “day” and “together” are the five most important words in the “**I have a dream speech**” from **Martin Luther King**.

Arguments of the **word cloud generator** function :

- words : the words to be plotted
- freq : their frequencies
- min.freq : words with frequency below min.freq will not be plotted
- max.words : maximum number of words to be plotted
- random.order : plot words in random order. If false, they will be plotted in decreasing frequency
- rot.per : proportion words with 90 degree rotation (vertical text)
- colors : color words from least to most frequent. Use, for example, colors ="black" for single color.

Go further

Explore frequent terms and their associations

You can have a look at the frequent terms in the term-document matrix as follow. In the example below we want to find words that occur at least four times :

```
findFreqTerms(dtm, lowfreq = 4)
 [1] "able"    "day"     "dream"   "every"   "faith"   "free"    "freedom"  "let"     "mountain" "nation"
[11] "one"     "ring"    "shall"   "together" "will"
```

You can analyze the association between frequent terms (i.e., terms which correlate) using findAssocs() function. The R code below identifies which words are associated with "freedom" in **I have a dream speech** :

```
findAssocs(dtm, terms = "freedom", corlimit = 0.3)
           freedom
let           0.89
ring          0.86
mississippi   0.34
mountainside  0.34
stone         0.34
every         0.32
mountain      0.32
state         0.32
```

The frequency table of words

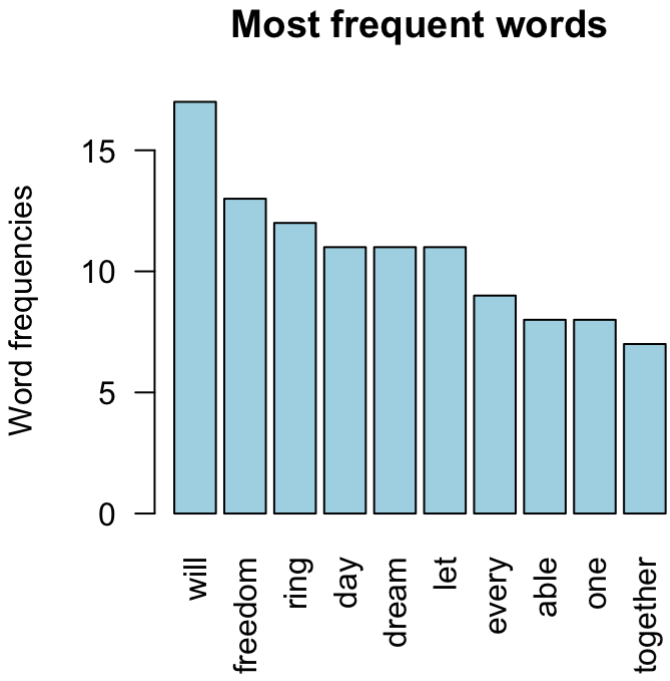
```
head(d, 10)
           word freq
will       will   17
freedom   freedom   13
ring       ring   12
day         day   11
dream      dream   11
let         let   11
every      every    9
```

```
able       able   8
one         one   8
together together  7
```

Plot word frequencies

The frequency of the first 10 frequent words are plotted :

```
barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word,
        col ="lightblue", main ="Most frequent words",
        ylab = "Word frequencies")
```



A stemming algorithm reduces the words "fishing", "fished", and "fisher" to the root word, "fish". On the other hand, "argue", "argued", "argues", "arguing", and "argus" reduce to the stem "argu"

2.A **corpus** (plural *corpora*) or **text corpus** is a large and structured set of texts (nowadays usually electronically stored and processed). They are used to do statistical analysis.

3. **Word Cloud** Example