R based Introduction to Tekstmining as a Part of Bigdata Course

John Aa. Sørensen, lektor
Section for Information Technology, DTU Diplom

23 March 2017

Use packages tm and wordcloud

```
##############################################################
#       Enter the main R Language documentation.
#
help.start() # Main entry to the R project documentation.
opar.org <- par(no.readonly=TRUE) # Original parameters.
loc='C:/R_packages/'              # Folder for packages.
```

Use packages tm and wordcloud

```
#####################################################
#
#  Verify a single txt document analysis example using the
#  packages "tm", "wordcloud" and their required packages.
#  Required txt document in RStudio working directory:
#  doc_1.csv which, in advance, is generated from the
#  R_data_export_import.pdf obtained from [R Core, 2015a].
#
#  Notice: This section does not require a pdf2txt
#          conversion.
#
install.packages("tm",lib=loc)  # Text mining package.
install.packages("NLP",lib=loc) # Nat. Lang. Proc. package.
install.packages("wordcloud",lib=loc) # Wordcloud package.
install.packages("RColorBrewer",lib=loc) # Color package.
#
library("NLP",lib=loc)
library("tm",lib=loc)
library("RColorBrewer",lib=loc)
library("wordcloud", lib=loc)
```

Use package tm

```
#-----------------------------------------------
docs <- read.csv("doc_1.csv", colClasses="character",
                 stringsAsFactors=FALSE)
head(docs,20)
str(docs)  # Inspect the first lines in the doc_1.csv file
class(docs) # docs is a data frame
attributes(docs)
(nrow(docs)) # Print the number of rows in data frame.
docs[1:100,] # Inspect the 100 top lines in document.
docs[200:209,] # Print 10 rows from row no. 200 in
               # dataframe docs.
#-----------------------------------------------
```

Use package tm

```
#-----------------------------------------------
# Paste all lines with a space into one vector.
#
docs_txt <- paste(docs, collapse=" ")
str(docs_txt)
class(docs_txt)
attributes(docs_txt)
summary(docs)
```

Use package tm

```
#
# Create a corpus with one document.
#
docs_source <- VectorSource(docs_txt)
str(docs_source)
corpus <- Corpus(docs_source) # Convert to corpus.
corpus     # List what's in the corpus.
str(corpus)
head(corpus,15)
```

Use package tm

```
# Print document no. 1 in the corpus.
writeLines(as.character(corpus[[1]]))
#
# Cleaning the corpus
corpus <- tm_map(corpus,
          content_transformer(tolower))
?tm_map()
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, stripWhitespace)
corpus <- tm_map(corpus, removeWords,
                  stopwords("english"))
str(corpus)  # Inspect structure of corpus
```

Use package tm

```
# Create the document term matrix.
?DocumentTermMatrix()  # Inspect the manual.
#
# Create a document term matrix with one row.
dtm <- DocumentTermMatrix(corpus)
inspect(dtm[1,1700:1750]) # Display 51 positions
str(dtm)                  # Inspect internal structure.
dtm2 <- as.matrix(dtm)
str(dtm2)                 # Inspect internal structure.
```

Use package tm

```
#
# Frequency
frequency <- colSums(dtm2)
str(frequency)
class(frequency)
#
# Sort to find the words with highest frequency.
frequency <- sort(frequency, decreasing=TRUE)
frequency
#
```

Use package tm

```
# Inspect the 30 most frequently represented words.
head(frequency,30)
#
# List the 200 most frequent words.
words <- names(frequency) # List words to be cluttered.
words[1:200]
```

Use package wordcloud

```
#
# Plot the top 200 words in a wordcloud.
# The word size is representing the frequency of that word.
# random.order=TRUE => the words are plotted in random order.
# rot.per= 0.5 => half of the words are rotated 90 degrees.
#
par(opar.org)
wordcloud(words[1:200], frequency[1:200], random.order=TRUE,
          rot.per=0.50)
par(opar.org)
```

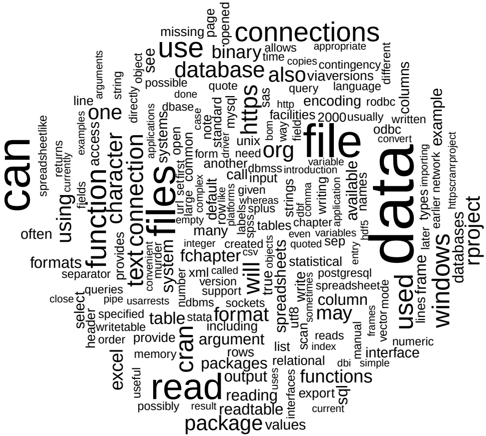
Use package wordcloud

```
#
# Plot the top 200 words in a wordcloud.
# The word size is representing the frequency of that word.
# random.order=FALSE => the words are plotted in order of
#                       decreasing frequency.
# rot.per = 0 => no rotation of the words.
#
wordcloud(words[1:200], frequency[1:200], random.order=FALSE,
          rot.per=0)
```

Use package wordcloud

```
#
par(opar.org)
pdf("fig_7_1_200_words_random.pdf")
wordcloud(words[1:200], frequency[1:200], random.order=TRUE,
          rot.per=0.5)
dev.off()
par(opar.org)
```

Example on 200 wordcloud, random order, 50% rotated



Use package wordcloud

```
#
par(opar.org)
pdf("fig_7_1_200_words_not_random.pdf")
wordcloud(words[1:200], frequency[1:200], random.order=FALSE,
          rot.per=0)
dev.off()
par(opar.org)
```

Example on 200 wordcloud, not random order, not rotated

Use package wordcloud, plot 3, 10 and 30 words clouds

```
#
par(mfrow=c(1,3))  # 1 rows and 3 columns.
wordcloud(words[1:3], frequency[1:3], rot.per=0,
                                    random.order=FALSE)
wordcloud(words[1:10], frequency[1:10], rot.per=0,
                                    random.order=FALSE)
wordcloud(words[1:30], frequency[1:30], rot.per=0,
                                    random.order=FALSE)
par(opar.org)
```

Use package wordcloud, pdf plot 3, 10 and 30 words clouds

```
#
par(opar.org) # Restore original parameters.
pdf(file = "fig_7_2_wcloud.pdf")
par(mfrow=c(1,3))          # 1 row and 3 columns.
wordcloud(words[1:3], frequency[1:3], rot.per=0,
random.order=FALSE)
wordcloud(words[1:10], frequency[1:10], rot.per=0,
random.order=FALSE)
wordcloud(words[1:30], frequency[1:30], rot.per=0,
random.order=FALSE)
dev.off()
par(opar.org)
```

Example on 3 wordclouds, not random order, not rotated