Text mining with R

You will work on Twitter data generated for an R discussion feed. You will perform:

Core text processing steps

Text association mining

**1: Data loading and text preprocessing**

1. Load the rdmTweets.RData file into memory.

**load("rdmTweets.RData")**

2. Load the twitteR, wordcloud, SnowballC, and tm libraries:

**library(twitteR)**

**library(tm)**

**library(wordcloud)**

**library(SnowballC)**

3. Convert the input to a data frame, and find the dimensionality of the data:

**dataframe <- do.call("rbind", lapply(rdmTweets, as.data.frame));**

**nrow(dataframe)**

**ncol(dataframe)**

4. Create a corpus, and display the first six lines in the file:

**myDocuments <- Corpus(VectorSource(dataframe $text));**

**inspect(myDocuments[1:6])**

5. Convert the text to lowercase, and display the first six lines in the file:

**myDocuments <-tm\_map( myDocuments,content\_transformer(tolower));**

**inspect(myDocuments[1:6])**

6. Remove all punctuation and display the first six lines in the file:

**myDocuments <- tm\_map(myDocuments, removePunctuation);**

**inspect(myDocuments[1:6])**

7. Remove numbers from the file and display the first six lines in the file:

**myDocuments <- tm\_map(myDocuments, removeNumbers);**

**inspect(myDocuments[1:6])**

8. Remove stop words from the file and display the first six lines in the file:

**myDocuments <- tm\_map(myDocuments, removeWords, stopwords("english"));**

**inspect(myDocuments[1:6])**

9. Take an original copy, perform stemming and display the first six lines in the file:

**originalBackup <- myDocuments;**

**myDocuments <- tm\_map(myDocuments, stemDocument);**

**inspect(myDocuments[1:6])**

1. Complete the stemming process. This is complicated by a bug in the current version of the stemming package, which can be fixed by modifying the stemCompletion function as follows:

stemCompletion\_fix <- function(x, dictionary)

{

words <- unlist(strsplit(as.character(x), " ")) words <- words[words != ""]

PlainTextDocument(stripWhitespace(paste(stemCompletion(words,

dictionary=dictionary), sep="", collapse=" ")))

}

**myDocuments <- lapply(myDocuments, stemCompletion\_fix, dictionary=originalBackup); myDocuments <- Corpus(VectorSource(myDocuments))**

11. Create a term document matrix and display a partial matrix:

**DTM <- TermDocumentMatrix(myDocuments, control=list(minWordLength=1))**

**inspect(DTM[80:105, 1:20])**

12. Display frequent terms with a frequency of 10 or higher:

**findFreqTerms(DTM, lowfreq=10)**

1. Convert the DTM to a matrix and calculate the word frequency. Sort the frequencies and display as a bar plot where a word appears in 15 or more tweets:

**matrix <- as.matrix(DTM)**

**dim(matrix)**

**wordFrequency <- rowSums(matrix)**

**sortedValues <- sort(wordFrequency, decreasing=TRUE)**

**topSortedWords <- subset(sortedValues, sortedValues >= 15)**

**barplot(topSortedWords, las=2)**

1. Build a word cloud where the size of the word is proportional to the frequency of occurrence:

**identifiers <- names(sortedValues )**

**wordcloud(words=identifiers,freq=sortedValues, min.freq=8)**

1. Experiment with changing min.freq and observe how output displayed on the screen changes.