Text Mining 1: Indexing and Querying Text

300958 Social Web Analysis

Week 5 Lab Solutions

- Examine the help page for `sort` and work out how to obtain the top 20 occurring words from the table.

```
sort(word.table, decreasing=TRUE)[1:20]
```

- To perform each of these tasks, `tm` provides the functions `removeNumbers`, `removePunctuation`, `stripWhitespace`, `tolower`, `removeWords` and `stemDocument` and the application function `tm_map`. Use your knowledge of R, the help pages and your favourite Web search engine to work out how to perform the six tasks, then implement them on our corpus. You can examine the changes in the corpus by printing the contents of the first document using `tweet.corpus[[1]]`. **Hint**: `tm_map` applies a function to all documents in th `tm_map` help page.


```
corpus = tm_map(corpus, function(x) iconv(x, to='ASCII                    ecial characters
corpus = tm_map(corpus, removeNumbers) # remove numbers
corpus = tm_map(corpus, removePunctuation) # remove punctuation
corpus = tm_map(corpus, stripWhitespace) # remove whitespace
corpus = tm_map(corpus, tolower) # convert all to lowercase
corpus = tm_map(corpus, removeWords, stopwords()) # remove stopwords
corpus = tm_map(corpus, PlainTextDocument)
corpus = tm_map(corpus, stemDocument) # convert all words to their stems
```

- Compute the weighted document term matrix `tweet.weighted.matrix` containing the values of $w_{d,t}$.

```
N = nrow(tweet.matrix)
IDF = log(N/colSums(tweet.matrix > 0))
TF = log(tweet.matrix + 1)
tweet.weighted.matrix = TF %*% diag(IDF)
```

- Sum the weights in `tweet.weighted.matrix` to obtain an overall weight for each term.

```
w = colSums(tweet.weighted.matrix)
```

- Locate the position of the top 20 words, according to the overall word weight. Use the vector `colnames(tweet.matrix)` to locate the word names.

```
o = order(w, decreasing = TRUE)[1:20]
colnames(tweet.matrix)[o]
```