CIND 110-Assignment 2

Amarpreet Kaur

06-Dec-2022

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

#Install and load required packages

```
#https://cran.r-project.org/web/packages/tm/index.html
#install.packages("tm", dependencies = TRUE)
#https://cran.r-project.org/web/packages/RWeka/index.html
#install.packages("RWeka", dependencies = TRUE)
#https://cran.r-project.org/web/packages/textstem/index.html
#install.packages("textstem", dependencies = TRUE)
#https://cran.r-project.org/web/packages/textclean/index.html
#install.packages("textclean", dependencies = TRUE)
#https://cran.r-project.org/web/packages/text2vec/index.html
#install.packages("text2vec", dependencies = TRUE)
lstPackages <- c('tm', 'RWeka', 'textstem', 'textclean', 'text2vec')
lapply(lstPackages, library, character.only = TRUE)
```

```
## Warning: package 'tm' was built under R version 4.2.2
```

```
## Loading required package: NLP
```

```
## Warning: package 'RWeka' was built under R version 4.2.2
```

```
## Warning: package 'textstem' was built under R version 4.2.2
```

```
## Loading required package: koRpus.lang.en
```

```
## Warning: package 'koRpus.lang.en' was built under R version 4.2.2
```

```
## Loading required package: koRpus
```

```
## Warning: package 'koRpus' was built under R version 4.2.2
```

```
## Loading required package: sylly
```

```
## Warning: package 'sylly' was built under R version 4.2.2
```

```
## For information on available language packages for 'koRpus', run
##
##   available.koRpus.lang()
##
## and see ?install.koRpus.lang()
```

```
##
## Attaching package: 'koRpus'
```

```
## The following object is masked from 'package:tm':
##
##     readTagged
```

```
## Warning: package 'textclean' was built under R version 4.2.2
```

```
## Warning: package 'text2vec' was built under R version 4.2.2
```

```
## [[1]]
## [1] "tm"        "NLP"       "stats"     "graphics"  "grDevices" "utils"
## [7] "datasets"  "methods"   "base"
##
## [[2]]
##  [1] "RWeka"     "tm"        "NLP"       "stats"     "graphics"  "grDevices"
##  [7] "utils"     "datasets"  "methods"   "base"
##
## [[3]]
##  [1] "textstem"       "koRpus.lang.en" "koRpus"         "sylly"
##  [5] "RWeka"          "tm"             "NLP"            "stats"
##  [9] "graphics"       "grDevices"      "utils"          "datasets"
## [13] "methods"        "base"
##
## [[4]]
##  [1] "textclean"      "textstem"       "koRpus.lang.en" "koRpus"
##  [5] "sylly"          "RWeka"          "tm"             "NLP"
##  [9] "stats"          "graphics"       "grDevices"      "utils"
## [13] "datasets"       "methods"        "base"
##
## [[5]]
##  [1] "text2vec"       "textclean"      "textstem"       "koRpus.lang.en"
##  [5] "koRpus"         "sylly"          "RWeka"          "tm"
##  [9] "NLP"            "stats"          "graphics"       "grDevices"
## [13] "utils"          "datasets"       "methods"        "base"
```

##Read The Sample Data QUESTION 2

```
# Read The Sample Dataset
rawData <- read.csv(file = 'booksCSV.csv', header = T, sep = ",")
numberofDocs <- length(rawData$id)
rawData$id <- paste0("Doc", c(1:numberofDocs))
```

Prepare The Corpora

```
# Prepare The Corpora
listofDocs <- tm::VectorSource(rawData$description)
listofDocs$Names <- names(rawData$id)
corporaData <- tm::VCorpus(listofDocs)
```

```
## Warning in as.POSIXlt.POSIXct(Sys.time(), tz = "GMT"): unable to identify current timezone
'C':
## please set environment variable 'TZ'
```

```
#listofDocs
```

Cleaning and Preprocessing the text (Cleansing)

```
#Replacing number with words
for(i in 1:12)
{
  corporaData[[i]]$content <-
  as.character(textclean::replace_number(corporaData[[i]]$content))
}

#Utilizing a Thesaurus
for(i in 1:12)
{
    corporaData[[i]]$content <-
    textstem::lemmatize_strings(corporaData[[i]]$content,
                                dictionary = lexicon::hash_lemmas)
}

#Stemming
corporaData <- tm::tm_map(corporaData, stemDocument)

#Stopword Removal
corporaData <- tm::tm_map(corporaData, removeWords, stopwords('english'))
corporaData <- tm::tm_map(corporaData, removeWords, stopwords('SMART'))

#Other Pre-processing Steps: Punctuation Marks, Extra Whitespaces, etc
corporaData <- tm::tm_map(corporaData, content_transformer(tolower))
corporaData <- tm::tm_map(corporaData, removePunctuation,
                      ucp = TRUE,
                      preserve_intra_word_contractions = FALSE,
                      preserve_intra_word_dashes = FALSE)
#corporaData <- tm::tm_map(corporaData, removeNumbers)
corporaData <- tm::tm_map(corporaData, stripWhitespace)

corporaData[[12]]$content
```

```
## [1] "microsoft visual studio explor depth visual basic visual c++ c asp+ integr comprehens de
velop environment"
```

```
corporaData
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 12
```

##Create a uni-gram Term Document Matrix-QUESTION 2 & 3 CREATING TDM-UNI

```
# Create a uni-gram Term Document Matrix
term.doc.matrix.1g <- tm::TermDocumentMatrix(corporaData)
tm::inspect(term.doc.matrix.1g[1:12,1:12])
```

```
## <<TermDocumentMatrix (terms: 12, documents: 12)>>
## Non-/sparse entries: 14/130
## Sparsity           : 90%
## Maximal term length: 9
## Weighting          : term frequency (tf)
## Sample             :
##            Docs
## Terms       1 11 12 2 3 4 5 6 8 9
##   after     0  0  0 0 1 0 0 0 0 1
##   agent     0  0  0 0 0 1 0 0 0 0
##   anoth     0  0  0 0 0 0 1 0 0 0
##   antholog  0  0  0 0 0 0 0 0 1 0
##   apocalyps 0  0  0 0 0 1 0 0 0 0
##   applic    1  0  0 0 0 0 0 0 0 0
##   architect 0  0  0 1 0 0 0 0 0 0
##   ascendant 0  0  0 0 0 1 0 0 0 0
##   asp+      0  0  1 0 0 0 0 0 0 0
##   battl     0  0  0 1 0 0 1 0 0 0
```

```
# Represent TDM in a matrix format and display its dimensions
term.doc.matrix.unigram <- as.matrix(term.doc.matrix.1g)
dim(term.doc.matrix.unigram)
```

```
## [1] 106  12
```

```
head(term.doc.matrix.unigram)
```

```
##            Docs
## Terms       1 2 3 4 5 6 7 8 9 10 11 12
##   after     0 0 1 0 0 0 0 0 1  0  0  0
##   agent     0 0 0 1 0 0 0 0 0  0  0  0
##   anoth     0 0 0 0 1 0 0 0 0  0  0  0
##   antholog  0 0 0 0 0 0 0 1 0  0  0  0
##   apocalyps 0 0 0 1 0 0 0 0 0  0  0  0
##   applic    1 0 0 0 0 0 0 0 0  0  0  0
```

Declaring weights (TF-IDF variants)

```
# Declaring weights (TF-IDF variants)
tf.idf.weights <- function(tf.vec) {
  # Computes tfidf weights from term frequency vector
  n.docs <- length(tf.vec)
  doc.frequency <- length(tf.vec[tf.vec > 0])
  weights <- rep(0, length(tf.vec))
  relative.frequency <- tf.vec[tf.vec > 0] / sum(tf.vec[tf.vec > 0])
  weights[tf.vec > 0] <-  relative.frequency * log10(n.docs/doc.frequency)
  return(weights)
}
```

###Compute the TF-IDF (unigram)

```
#Compute the TF-IDF (unigram)
tfidf.matrix.uni <- t(apply(as.matrix(term.doc.matrix.unigram), 1,
                        FUN = function(row) {tf.idf.weights(row)}))

colnames(tfidf.matrix.uni) <- rawData$id
head(tfidf.matrix.uni)
```

```
##
## Terms           Doc1 Doc2      Doc3     Doc4     Doc5 Doc6 Doc7     Doc8
##   after     0.000000    0 0.3890756 0.000000 0.000000    0    0 0.000000
##   agent     0.000000    0 0.0000000 1.079181 0.000000    0    0 0.000000
##   anoth     0.000000    0 0.0000000 0.000000 1.079181    0    0 0.000000
##   antholog  0.000000    0 0.0000000 0.000000 0.000000    0    0 1.079181
##   apocalyps 0.000000    0 0.0000000 1.079181 0.000000    0    0 0.000000
##   applic    1.079181    0 0.0000000 0.000000 0.000000    0    0 0.000000
##
## Terms            Doc9 Doc10 Doc11 Doc12
##   after     0.3890756     0     0     0
##   agent     0.0000000     0     0     0
##   anoth     0.0000000     0     0     0
##   antholog  0.0000000     0     0     0
##   apocalyps 0.0000000     0     0     0
##   applic    0.0000000     0     0     0
```

```
dim(tfidf.matrix.uni)
```

```
## [1] 106  12
```