7 Text Analysis

Jonathan De Los Santos

3/8/2021

# Prompt

This assignment asks you to examine text mining for classification. Provide your answers to the questions in a Word document named Assign6\_LastName.doc along with your source code that should be saved as Assign6\_LastName, and click the title link to upload and subit them.

The BC3 text file includes news related to Blockchain technologies, and the DS text file contains news regarding Ebola issues. The preprocessing for text files is critical. Notably, in this assignment, you are requested to conduct mining text files. Therefore, the preprocessing becomes much more critical to complete the below two tasks. Please carefully read the directions and answer the questions. Please treat yourself as a data scientist and come up with solutions. To provide the best answers to the below questions, you may need to iterate the procedures several times.

# Data Preprocessing:

## A. Import the necessary packages.

library(tm)  
library(stringr)  
library(Matrix)  
library(tidytext)  
library(topicmodels)

## B. Create the Corpus

Import the two text files, and create the corpus. You can create one corpus containing two text files or generate two different corpora (or corpuses) for each text file. If you create two corpora (or corpuses) for two text files, you need to combine two document-term matrices to answer the below questions. (10 pts)

Beginning with our Ebola and Blockchain text files, we will read in both files as lines then execute the VectorSource() function to pass in these lines as a document to a corpus object. The first line of each file is displayed to demonstrate what a single document looks like.

textDS = file("DS.txt", open = "r")  
textDS.decomposition = readLines(textDS, encoding="UTF-8")  
textDS.decomposition[1]

## [1] "In a statement, the Travel and Transport Task Force called for international cooperation of governments and the transport sector in following the recommendations of the International Health Regulations Emergency Committee on Ebola, convened by the UN World Health Organization (WHO)."

corpusDS <- Corpus(VectorSource(textDS.decomposition))  
#orpusDS  
  
textBC = file("BC3.txt", open = "r")  
textBC.decomposition = readLines(textBC, encoding="UTF-8")  
textBC.decomposition[1]

## [1] "That explanation of where this new technology sits within history feels right: While prognosticators love to talk about crypto and blockchain as a bubble, it is likely just very early days. And while 1999 marked what seemed like a high point for the internet before a precipitous fall, it proved to only be the first stage of the internet\xa1\xafs ascendance. Yes, there were implosions like Pets.com, but there was also Amazon."

corpusBC3 <- Corpus(VectorSource(textBC.decomposition))  
#corpusBC

## C. Clean the Data

Clean the data by applying the necessary steps. In the process of cleaning the data, please come up with reasonable “self-stopwords” for both text files. If you have two corpora (or corpuses), perform the cleaning process separately for each corpus, and combine the document-term matrix after cleaning the data. There is no single way to conduct this task; hence you’re strongly encouraged to provide detailed comments on your code. (35 pts)

We have decided to create two different corpora for the DS and BC3 files and we will begin by cleaning each corpus individually. This includes standard cleaning steps:

* Lowering characters
* Removing numbers
* Removing Punctuation
* Creating custom stop words
* Removing both these custom stop words and the common English words
* Stripping whitespace
* Stemming (pairing to the root word)

### DS Data Cleaning

Perform the above cleaning steps on the DS corpus object. Here I have selected the words “Ebola”, “UN”, “WHO”, and “Virus” as terms that are common and likely not useful to our analysis. We write the first line again below after the whitespace stripping and stemming phases. The stemming appears to be a little aggressive, but hopefully it is consistent and does not affect the overall analysis.

# Basic cleaning  
  
corpusDS <- tm\_map(corpusDS, tolower)

## Warning in tm\_map.SimpleCorpus(corpusDS, tolower): transformation drops  
## documents

corpusDS <- tm\_map(corpusDS, removeNumbers)

## Warning in tm\_map.SimpleCorpus(corpusDS, removeNumbers): transformation drops  
## documents

corpusDS <- tm\_map(corpusDS, removePunctuation)

## Warning in tm\_map.SimpleCorpus(corpusDS, removePunctuation): transformation  
## drops documents

# Stopwords  
  
selfstopwordsDS <- c("Ebola", "UN", "WHO", "Virus")  
  
corpusDS<- tm\_map(corpusDS, removeWords, c(stopwords("en"), selfstopwordsDS))

## Warning in tm\_map.SimpleCorpus(corpusDS, removeWords, c(stopwords("en"), :  
## transformation drops documents

# Strip Whitespace  
corpusDS <- tm\_map(corpusDS, stripWhitespace)

## Warning in tm\_map.SimpleCorpus(corpusDS, stripWhitespace): transformation drops  
## documents

writeLines(as.character(corpusDS[[1]]))

## statement travel transport task force called international cooperation governments transport sector following recommendations international health regulations emergency committee ebola convened un world health organization

# Stemming  
corpusDS <- tm\_map(corpusDS, stemDocument)

## Warning in tm\_map.SimpleCorpus(corpusDS, stemDocument): transformation drops  
## documents

writeLines(as.character(corpusDS[[1]]))

## statement travel transport task forc call intern cooper govern transport sector follow recommend intern health regul emerg committe ebola conven un world health organ

### BC3 Data Cleaning

Perform the above cleaning steps on the BC3 corpus object. Here I have selected the words“crypto”, “cryptocurrency”, “blockchain”, and “technology” as terms that are common and likely not useful to our analysis. However, I did leave the word “bitcoin” as it was not used as frequently as I expected and may reveal some interesting relationships. The first lines are written out again after the whitespace stripping and stemming.

# Basic cleaning  
  
corpusBC3 <- tm\_map(corpusBC3, tolower)

## Warning in tm\_map.SimpleCorpus(corpusBC3, tolower): transformation drops  
## documents

corpusBC3 <- tm\_map(corpusBC3, removeNumbers)

## Warning in tm\_map.SimpleCorpus(corpusBC3, removeNumbers): transformation drops  
## documents

corpusBC3 <- tm\_map(corpusBC3, removePunctuation)

## Warning in tm\_map.SimpleCorpus(corpusBC3, removePunctuation): transformation  
## drops documents

# Stopwords  
  
selfstopwordsBC3 <- c("crypto", "cryptocurrency", "blockchain", "technology")  
  
corpusBC3<- tm\_map(corpusBC3, removeWords, c(stopwords("en"), selfstopwordsBC3))

## Warning in tm\_map.SimpleCorpus(corpusBC3, removeWords, c(stopwords("en"), :  
## transformation drops documents

# Strip Whitespace  
corpusBC3 <- tm\_map(corpusBC3, stripWhitespace)

## Warning in tm\_map.SimpleCorpus(corpusBC3, stripWhitespace): transformation drops  
## documents

writeLines(as.character(corpusBC3[[1]]))

## explanation new sits within history feels right prognosticators love talk bubble likely just early days marked seemed like high point internet precipitous fall proved first stage internet¡¯s ascendance yes implosions like petscom also amazon

# Stemming  
corpusBC3 <- tm\_map(corpusBC3, stemDocument)

## Warning in tm\_map.SimpleCorpus(corpusBC3, stemDocument): transformation drops  
## documents

writeLines(as.character(corpusBC3[[1]]))

## explan new sit within histori feel right prognost love talk bubbl like just earli day mark seem like high point internet precipit fall prove first stage internet¡¯ ascend yes implos like petscom also amazon

### Create Document-Term Matrices

This task ends by creating the Document-Term Matrices for each corpus individually. Then we use the combine (c()) function from the tm package to create one master DTM.

# Create DS TDM  
  
corpusDS.dtm <- DocumentTermMatrix(corpusDS)  
#corpusDS.tdm  
  
# Create BC TDM  
  
corpusBC.dtm <- DocumentTermMatrix(corpusBC3)  
#corpusBC.tdm  
  
# Combine matrices  
  
corpus.dtm <- c(corpusDS.dtm, corpusBC.dtm)  
#corpus.dtm  
  
#write.csv(as.matrix(corpus.dtm, file = "CombinedDTM\_Test"))

# LDA Analysis

Apply the Latent Dirichlet Allocation (LDA) technique to complete the following.

## D. Find Optimal K

Find the optimal number of topics k for the text data. (15 pts)

To find the ideal number of topics , we need to calculate **perplexity**. This will be done from a list of candidate values to reduce computational needs. First create some possible values of and create an empty matrix for them to be filled in with their perplexity below.

can\_k <- c(2,3,4,5,10,20,40,60,100)  
  
results <- matrix(0, nrow = length(can\_k), ncol = 2)  
colnames(results) <- c("k", "Perplexity")

Next we will loop through the corpus enough times to cover each candidate k (9), perform an LDA, calculate the perplexity, and store the resulting value in the matrix created above.

Based on this analysis, we choose **4 as our k-value** based on lowest calculated perplexity.

for (i in 1:length(can\_k)){  
 k <- can\_k[i]  
 SEED <- 1234  
 text.lda <- LDA(corpus.dtm, k=k, method = "Gibbs",  
 control = list(seed = SEED, burnin = 1000,  
 thin = 100, iter = 1000))  
 results[i,] <- c(k, perplexity(text.lda, newdata = corpus.dtm))  
}  
  
results\_df <- as.data.frame(results)  
results\_df

## k Perplexity  
## 1 2 671.7070  
## 2 3 653.8910  
## 3 4 642.1854  
## 4 5 640.5554  
## 5 10 627.3939  
## 6 20 635.4652  
## 7 40 668.2208  
## 8 60 695.0336  
## 9 100 680.7704

## E. Display Term and Document Probabilities

Display the term-probability and document-probability by setting the number of topics at k for each industry. (20 pts)

Using our of 10 from above, we can create objects to show the probability distributions for terms and documents. These are long, so uncomment the write.csv lines to export the results.

SEED <- 1234  
text.lda <- LDA(corpus.dtm, k = 10, method = "Gibbs",  
 control = list(seed = SEED, burnin = 1000,  
 thin = 100, iter = 1000))  
  
Terms <- posterior(text.lda)$terms  
Topics <- posterior(text.lda)$topics  
  
#write.csv(t(as.matrix(Terms)), file = "Terms\_Output.csv")  
#write.csv(as.matrix(Topics), file = "Topics\_Output.csv")

## F. Verify Results

Verify your outcomes from (a) and (b) above with the tidy() function. (20 pts)

Our term and document-term matrices are created below by using matrix types of “beta” and “gamma” respectively.

Looking at line 3 of the term-matrix, we see that there is a 0.35% probability that the term “call” is generated from topic 3.

Line 1 of the document-term matrix implies that there is a 9% probability that topic 1 will be found in document 1.

tidy\_topics <- tidy(text.lda, matrix = "beta")  
head(tidy\_topics, 10)

## # A tibble: 10 x 3  
## topic term beta  
## <int> <chr> <dbl>  
## 1 1 call 0.000306  
## 2 2 call 0.000346  
## 3 3 call 0.000376  
## 4 4 call 0.000371  
## 5 5 call 0.000351  
## 6 6 call 0.000352  
## 7 7 call 0.00400   
## 8 8 call 0.00359   
## 9 9 call 0.000302  
## 10 10 call 0.000330

tidy\_docs <- tidy(text.lda, matrix = "gamma")  
head(tidy\_docs, 10)

## # A tibble: 10 x 3  
## document topic gamma  
## <chr> <int> <dbl>  
## 1 1 1 0.123   
## 2 2 1 0.0857  
## 3 3 1 0.0735  
## 4 4 1 0.0875  
## 5 5 1 0.0824  
## 6 6 1 0.103   
## 7 7 1 0.0690  
## 8 8 1 0.107   
## 9 9 1 0.0649  
## 10 10 1 0.0685