BIG DATA ANALYSIS

October 2019

Text Mining

Kadek Ardya Novi Diani
09211850096004

Text Mining is a method to generate an unstructured data in text form, in order to analyze the main topic of the text, and many other advance purposes. But in this paper, the purpose will be limited only up to find the main topic of the article or text.
Basic concept of text mining is to structuring text data from sentences into words by tokenizing, and then cleansing the words from punctuation and other symbols, and stemming the words back to the basic words.
This paper will simulate text mining method with 2 kind of data source, which is twitter crawled data and journal text data in pdf form. Both analyses will be utilizing R as the analysis tools.
Before start to the simulation, make sure that packages below already installed in R.

```
library("rtweet")
library("NLP")
library("tm")
library("SnowballC")
library("RColorBrewer")
library("wordcloud")
library("textclean")
library("katadasaR")
library("tokenizers")
library("dplyr")
```

A. Text Data Twitter

1. Crawling Text Data

Before being able to crawl data from twitter, user has to create an API keyword from Twitter Developer, to get 4 keywords that will be used by R to connect the application to twitter. After getting the API keywords, input the keywords into R through below command:

```
token <- create_token(
  consumer_key = "xxxxxzeuoM91OJgKpRtcKdPG",
  consumer_secret = "xxxxxoETxiB9AdilFgqfC4N3kuDE2RqDAuUBGvtNvqiPiSAW",
  access_token = "xxxxx70-fbzM9v2qw0Qyj6cha2tnTghPqlPieyM3WDLAmYbYK",
  access_secret = "xxxxxFqWilAir77lNQAKxwIUeGFDRP6o0VHmhZbYVOK")
```

In this paper, we will try to crawl text data regarding 2 big fintech company in Indonesia, which are Gopay and Ovo.
Here is the command in twitter to crawling data:

```
tweet <- search_tweets(q = "gopay", n = 1000)      #data result in matrix form
text <- tweet[,5]                                  #to take column that containing the text
docs <- Corpus(VectorSource(text))                 #forming the text become a corpus
```

2. Cleansing Text Data

After collecting the text data, next process is to cleansing the text data from unnecessary symbols and replacing the typo or unformal words.
This process can be done in R by utilizing “textclean” package that already exist in R.
Cleansing text data command:

```
docs<-docs %>%
  replace_html() %>%                                                    #delete html
  replace_url() %>%                                                     #delete url
  replace_emoji() %>%                                                   #delete emoji
  replace_tag(docs, pattern = "@([A-Za-z0-9_]+)",replacement="") %>%    #delete mention
  replace_hash(docs, pattern = "#([A-Za-z0-9_]+)",replacement="")       #delete hashtag
```

In order to words fixing from typo or informal words, some dictionary in Bahasa can be utilized, by download it from www.github.com, save it, and then call it into R environment.
The command to words fixing is as below:

```
#calling dictionary into R environtment
spell.lex <- read.csv("C:/Users/Ardya Novi/OneDrive/Documents/Back up ID & Doc/MMT Business
Analytics/Big Data Analysis/kamus-alay-master/colloquial-indonesian-lexicon.csv")

#replacing typo & informal words
docs <- replace_internet_slang(docs, slang = paste0("\\b", spell.lex$slang, "\\b"),
replacement = spell.lex$formal, ignore.case = TRUE)
```

3. Stemming & Tokenizing Text Data

To do stemming process, package “katadasaR” can be utilized.
This package is still under development. So, to install this package must be through “devtools” package, and by command “install_github”.
Here is the detail command to do stemming process

```
library(devtools)
install_github("nurandi/katadasaR")
library(“katadasaR”)

stemming <- function(x){
  paste(lapply(x,katadasar),collapse = " ")}
docs <- lapply(tokenize_words(docs[]), stemming)
```

Tokenizing sentences into words, with command below:

```
docs <- tokenize_words(docs)
```

Next process is to filtering the words from unnecessary words, like pronoun, expressions, etc. Command “stopwords” can be utilized for this process. The list of stopwords can be saved in .txt form, and call it back into R environment. Some stopwords list can be downloaded from www.github.com.
Here is the command to words filtering:

```
#calling stopwords list from .txt file
myStopwords <- readLines("C:/Users/Ardya Novi/OneDrive/Documents/Back up ID & Doc/MMT
Business Analytics/Big Data Analysis/bilp-master/stoplist.txt")

#filtering words
docs <- as.character(docs)
docs <- tokenize_words(docs, stopwords = myStopwords)
```

4. Generating Graph

To visualizing the text data result based on quantity of the text occurring in the tweet, package “wordcloud” will be utilized.
Here is the command to create the word cloud:

```
class(docs)
docs <- as.character(docs)
set.seed(1234)
wordcloud(docs, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```

Here is the result of data text crawling, with keywords “gopay” and “ovo.

Based on chart above, it shows that when people talk about gopay, they also talk about the man behind the company. But when people talk about ovo, they mostly talk about the features, program benefit, and the competitor.

B. Reading Text Data from pdf Journal’s Abstract

1. Reading Text Data from pdf File

Package “pdftools” canbe used in order to read pdf text data into R. Here is the command:

```
#read text from pdf, save it as variable abs
abs<-pdf_text("C:/Users/Ardya Novi/OneDrive/Documents/Back up ID & Doc/MMT Business
Analytics/Big Data Analysis/Task 2 - Text Mining/TOP_10_NEURAL_NETWORK_PAPERS.pdf")

#splitting the elements of the text based on sentences
abs<-abs %>%
     strsplit(split = "\n")

#load the data into corpus
abstract <- Corpus(VectorSource(abs))
```

2. Cleansing Text Data

There is also other way to cleansing data than utilizing package “textclean”, which is by package “tm”. The command to cleansing text through package “tm” is as below:

```
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
abstract <- tm_map(abstract, toSpace, "/")                          #delete “/”
abstract <- tm_map(abstract, toSpace, "@")                          #delete “@”
abstract <- tm_map(abstract, toSpace, "\\|")                        #delete “\\|”
abstract <- tm_map(abstract, content_transformer(tolower))          #lowercase
abstract <- tm_map(abstract, removeNumbers)                         #delete numbers
abstract <- tm_map(abstract, removePunctuation)                     #delete punctuation
abstract <- tm_map(abstract, stripWhitespace)                       #delete extra space
```

3. Stemming & Tokenizing Text Data

Since the text data language is in English, so the command to do stemming is slightly different with case above (twitter data case).

In order to filtering text data, English stopwords is already built up in R, so it can be easily used just by command below:

```
abstract <- tm_map(abstract, removeWords, stopwords("english"))
```

User also be able to add up list of stopwords that is not covered yet in english stopwords, by command below:

```
abstract <- tm_map(abstract, removeWords, c("using", "ieee", "vol", "http", "ppr", "used"))
```

After the text data being clean, next process is to transform the text data into matix form, by command below:

```
dtm <- TermDocumentMatrix(abstract)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
```

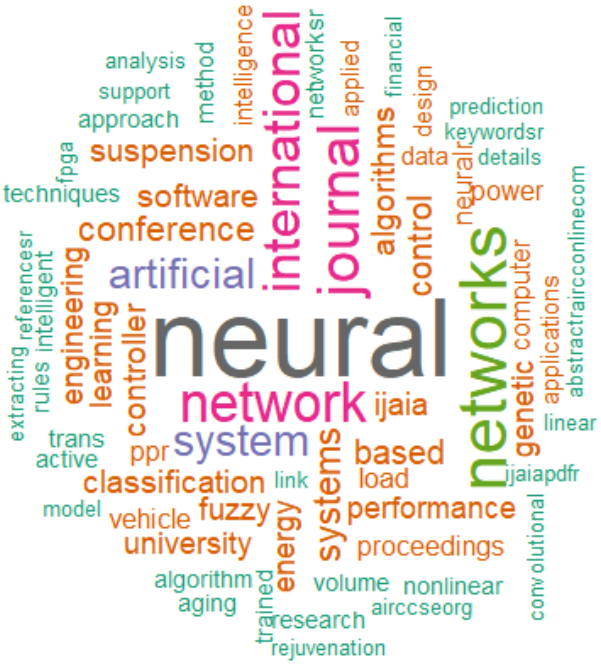
4. **Generating Graph**

Same as twitter data case, in order to visualize the content of the journal based on quantity of the words that is used, package "wordcloud" will be utilized.

Here is the command:

```
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 10,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```

The word cloud result is as below:



It is shown that the pdf file is mainly talks about neural network, and indicating that it is the international journals.