Text Analysis

Weekly assignment 6

Saurab den Butter

02 November, 2023

Contents

Loading required packages for the analysis

I start by loading the required packages for the analysis.

```
## Load required packages ----
if(!require(pacman)){
        install.packages('pacman')
}

p_load(tidyverse, tm, textclean,
       gt, janitor, ggthemes,
       kableExtra, SnowballC,
       RColorBrewer, wordcloud)

theme_set(ggthemes::theme_clean())
options(digits = 3)
options(scipen = 999)
```

Question 3: Stack Exchange topics and questions

Over the years, Facebook has hosted multiple competitions on `Kagle` to recruit new employees. This question is based on the third challenge.2 . The original competition tested text mining skills on a large data set from the Stack Exchange sites. The task was to predict the tags (a.k.a. keywords, topics, summaries), given only the question text and its title. The data set contains content from disparate stack exchange sites, containing

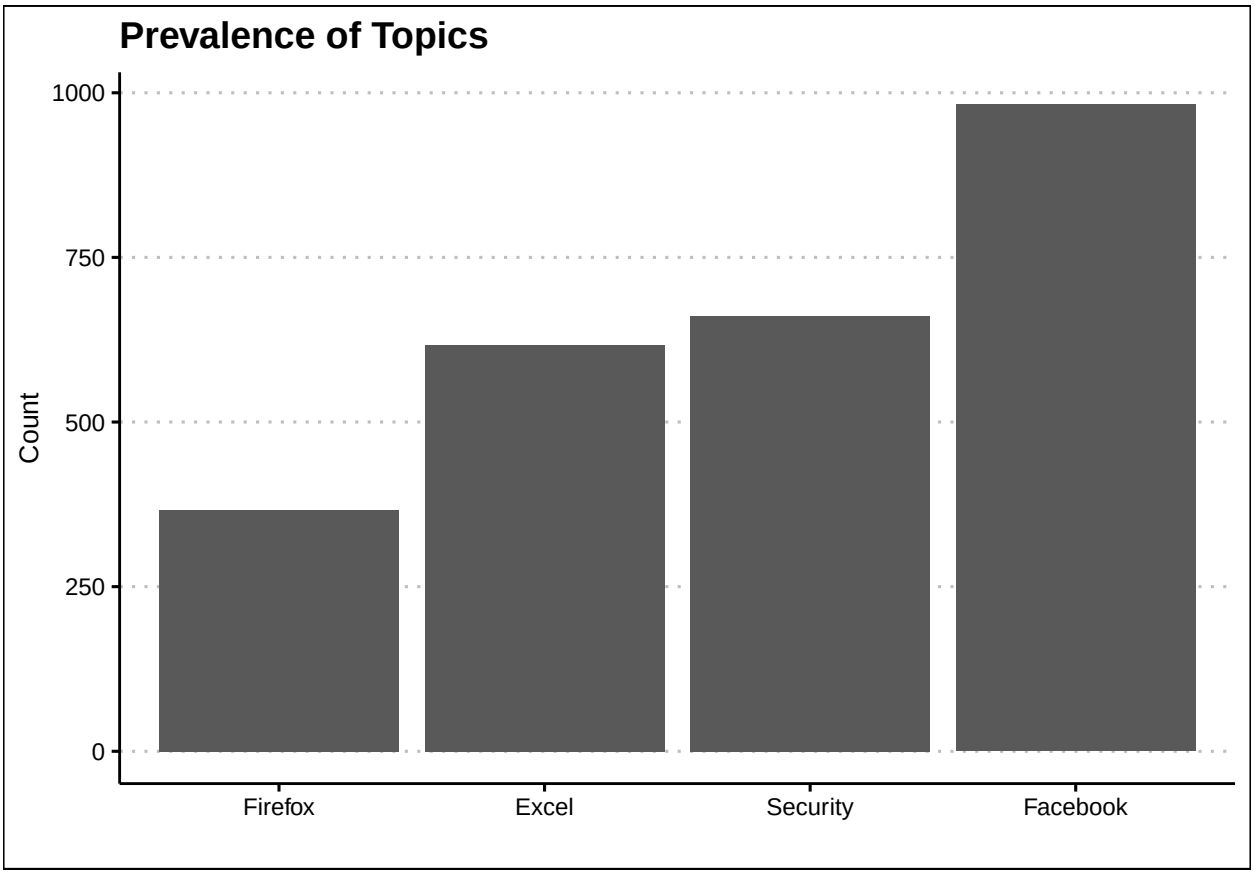
a mix of both technical and non-technical questions. A sample of this data set has been made available on Canvas, as stack.csv, and this sample provides three columns: the Topic of the post (one of four topics), the Title of the post, and the Body of the post which contains the actual question and explanation. This data set will be used for the rest of the assignment.

Distribution of posts over the various topics

The data set has 2626 observations and 3 variables.

TOPIC	COUNT	Prop
Facebook	982	0.374
Security	661	0.252
Excel	617	0.235
Firefox	366	0.139

The topic about `Facebook` is the most prevalent with 982 entries that accounts for about 37% of the observations. `Security` matters are also prominent with 661 observations (about 25%) of the posts. Posts about `Excel` and `Firefox` are the least prevalent with 617 (24%) and 366 (14%) prevalence, respectively.



Inspect a couple of question text bodies and report on your observations

The topic section contains HTML tags like <p> and <\p> and URLs for various websites referred to in the text. The text also contains digits that may not be of much use in a text analysis. Moreover, the existence of punctuation and, as in standard grammar, we have a mix of upper case and lower case letters in spelling of words (like at the start of a sentence). This could make two otherwise similar word be different. We also

have a lot of special characters (like *, #, @, %) that carry no meaning in text analysis. White spaces, and line breaks are also prevalent. Importantly, we have words that are indispensable in language but that carry no meaning on their own (called stop-words, for example the articles `a` and `the`, among others). p

```
## [1] "<p>In my favorite editor (vim), I regularly use ctrl-w to execute a certain action. Now, it qui
## [2] "<p>Aloha everyone,</p>\n\n<p>I have a class assignment in which I am tasked to build a MySql da
## [3] "<p>I recently started using Firefox as my primary web browser, and I would like to change some
## [4] "<p>I stepped over the following tutorial:\n<a href=\"http://afana.me/post/create-wizard-in-aspn
## [5] "<p>The site is set to WordPress 3.4.1, annexed widget Facebook for WordPress</p>\n\n<p>Later I
```

Discuss issues that need to be solved when cleaning the text later on.

1. HTML Tags: Because the text data comes from web sources, it contains HTML tags. I will remove these tags before commencing analysis.
2. Convert text to lower case: I will convert all text to lowercase to ensure consistency and prevent case-sensitive issues.
3. Punctuation: I will remove all punctuation marks (e.g., periods, commas, quotes) as they may not provide valuable information for analysis.
4. Special Characters: I will eliminate special characters, such as currency symbols or trademark signs, which may not be relevant to our analysis.
5. Stop Words: I will remove common stopwords (e.g., “and,” “the,” “in”) as they often don’t carry significant meaning for analysis. The `tm` package contains the list of the common English stopwords.
6. Numeric Character Removal: Because numbers are not relevant to our analysis, we remove them from the text. This is useful because we are focusing on language and not numerical data.
7. Whitespace and Line Breaks: I will remove extra whites pace and line breaks to ensure uniform formatting of the text before analysis. Without this, white spaces and line breaks could be mistaken for words and hence cloud our analysis.
8. Remove URLs: The text contains web links. I remove these links as they are not relevant to our analysis.
9. Tokenization: I will tokenize the text into words or phrases (tokens) to prepare it for further analysis.
10. Contractions: Some contractions in language complicate the analysis. In that case, I will replace all contractions into standard language (for instance, I’ll to I will).
11. Finally, I will be on the lookout for irrelevant meta data and non-standard character encoding that could affect the analysis.

Question 4: Preparation of the text

Choose two of the Topics for use in the rest of the assignment. Prepare (cleanse) the question body texts for further analysis by means of the following steps.

I select the following TWO topics;

- Facebook.
- Security.

Use functions from package textclean to remove URLs and make other desired adjustments.

To clean the data, I create a function that will automatically deal with all the issues identified above.

I use the above function to clean the data. I start with the facebook topic data.

Next, I clean the security topic data.

Use package tm to build a corpus from the vector with cleansed text for each of your chosen Topic;

In this section, I build a corpus and implement several cleansing steps as follow;

1. I remove all the English stopwords as described earlier. These words are important in grammar but carry no meaning on their own.
2. I remove all the numbers as they are not relevant in our text analysis.
3. Further, I eliminate all punctuation. Punctuation does not add value to a text analysis.
4. Next, I convert all text to lower case so that case sensitivity does not negatively affect our analysis.
5. Finally, I strip all the extra white spaces between words.

I then convert the output into a term document matrix.

Number of words and documents in the resulting corpus for each of your chosen Topic.

The facebook topic consists of one document with 9517 terms. There is zero sparse terms, meaning that each of the terms appears at least once. The maximal term length is 368.

```
## <<TermDocumentMatrix (terms: 9517, documents: 1)>>
## Non-/sparse entries: 9517/0
## Sparsity           : 0%
## Maximal term length: 368
## Weighting          : term frequency (tf)
```

The facebook topic consists of one document with 8647 terms. There is zero sparse terms, meaning that each of the terms appears at least once. The maximal term length is 676.

```
## <<TermDocumentMatrix (terms: 8647, documents: 1)>>
## Non-/sparse entries: 8647/0
## Sparsity           : 0%
## Maximal term length: 676
## Weighting          : term frequency (tf)
```

Question 5: Popular associates

For each of your chosen Topic, make an overview of the most frequent terms in the corpus and make an overview of the terms that have the higher correlations with that Topic. Comment on the results. Speculate as to whether there is possible predictive value in these results. [0.5 page]

In each case, I pick the terms that occur at least 400 times, starting with the Facebook topic below.

```
##  [1] "app"      "can"      "code"     "facebook" "frown"    "get"
##  [7] "like"     "page"     "pnn"      "post"     "skeptic"  "stick"
## [13] "tongu"    "tri"      "use"      "user"     "work"
```

Next, we look at the security topic.

```
## [1] "can"     "frown"   "secur"   "server"  "skeptic" "stick"   "tongu"
## [8] "use"     "user"
```

We see a clear distinction in the terms occurring in both topics. For the security topic, the terms `secur` has a high association with the topic. For Facebook, the term `facebook`, `app` have a high correlation with the topic. In our case, the terms "can", "skeptic", "use", and "User" have little predictive value in topic modelling. Overall, the output has some predictive power as there are words that clearly stand out in each of the topics. For instance, the term `server` is highly related to the security topic but is not among the top terms in the facebook topic. Likewise, the term `page` is more prevalent in the facebook topic than in the security topic. Such prevalence of terms gives rise the data some predictive power.

Question 6: Word clouds

Make a word cloud for each of your chosen Topic, and include these in your report. Compare the two graphs and discuss the result. [0.5 page]

I make a word cloud for each of the topics. Figure 2 shows the word cloud for the facebook topic while Figure 3 shows one for the security topic. In each case, we see that although there are commonalities in the word clouds, some terms clearly stand out in each of the topics. For instance, the term `Facebook` stand out in the facebook topic. The terms `secur` and `app` appears more in the security topic. There are some overlaps though. The terms `skeptic`, `user` and `use` are common in both topics. However, this overlap does not take away the usefulness of the data for predictive modelling.

Other terms that stand out in the facebook topic.

- Like.
- Page.
- frown.

Other terms that stand out in the security topic.

- Server.
- Password.
- Access.
- Data.

Terms that overlap in both topics include, among others the following:

- Use.
- User.
- Code.
- Skeptic.

World Cloud for Facebook Topic

Figure 1: WordCloud for the Facebook Topic

World Cloud for Security Topic

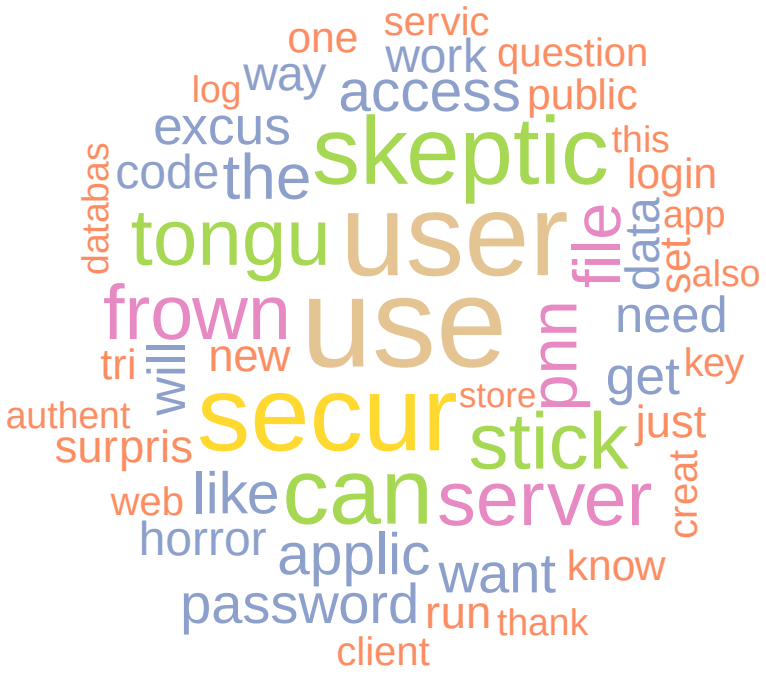


Figure 2: WordCloud for the Security Topic

References

Banks, G. C., Woznyj, H. M., Wesslen, R. S., & Ross, R. L. (2018). A review of best practice recommendations for text analysis in R (and a user-friendly app). **Journal of Business and Psychology**, 33, 445-459.

Welbers, K., Van Atteveldt, W., & Benoit, K. (2017). Text analysis in R. **Communication methods and measures**, 11(4), 245-265.

Code Appendix

```
## ----setup, include=FALSE-----------------------------------------------------------------------------------------
knitr::opts_chunk$set(echo = FALSE, warning = FALSE, message = FALSE)



## ----echo = TRUE--------------------------------------------------------------------------------------------------
## Load required packages ----
if(!require(pacman)){
        install.packages('pacman')
}

p_load(tidyverse, tm, textclean,
       gt, janitor, ggthemes,
       kableExtra, SnowballC,
       RColorBrewer, wordcloud)

theme_set(ggthemes::theme_clean())
options(digits = 3)
options(scipen = 999)
```

```
## ------------------------------------------------------------------------------------------------------------
## Load the data ----
chats <- read_csv('stack.csv') %>%
        clean_names()


## ------------------------------------------------------------------------------------------------------------
## Topics ----
counts <- chats %>%
        count(topic,
              sort = TRUE,
              name = "Count")



## ------------------------------------------------------------------------------------------------------------
## Prevalent topics table ----
counts %>%
        set_names(names(.) %>% str_to_upper()) %>%
        mutate(Prop = COUNT / nrow(chats)) %>%
        gt(caption = "Prevalence of Topics")



## ------------------------------------------------------------------------------------------------------------
## Prevalent topics ----
counts %>%
        mutate(topic = fct_reorder(topic, Count)) %>%
        ggplot(mapping = aes(x = topic, y = Count)) +
        geom_col() +
        labs(x = "", title = "Prevalence of Topics")


## ------------------------------------------------------------------------------------------------------------
## Chats body overview ----
chats %>%
        pull(body) %>%
        head(5)


## ------------------------------------------------------------------------------------------------------------
## text cleaning function using text clean ----
text_cleaner <- function(text){
        library(textclean)
        library(tidyverse)
        text %>%
                replace_contraction() %>%
                replace_date(replacement = "") %>%
                replace_email() %>%
                replace_emoji() %>%
                replace_emoticon() %>%
                replace_hash() %>%
                replace_html() %>%
```

```
                replace_internet_slang() %>%
                replace_white() %>%
                replace_number(remove = TRUE) %>%
                replace_tag() %>%
                replace_url() %>%
                replace_word_elongation()
}


## -----------------------------------------------------------------------------------------------------------
## Clean facebook data ----
clean_fb <- chats %>%
        filter(topic == "Facebook") %>%
        select(body) %>%
        text_cleaner()


## -----------------------------------------------------------------------------------------------------------
## Clean security data ----
clean_sec <- chats %>%
        filter(topic == "Security") %>%
        select(body) %>%
        text_cleaner()


## -----------------------------------------------------------------------------------------------------------
## Create a document term matrix for facebook topic ----
fb_dtm <- clean_fb %>%
        VectorSource() %>%
        Corpus() %>%
        tm_map(removeWords, stopwords('english')) %>%
        tm_map(removeNumbers) %>%
        tm_map(removePunctuation) %>%
        tm_map(tolower) %>%
        tm_map(stripWhitespace) %>%
        tm_map(stemDocument) %>%
        TermDocumentMatrix()



## -----------------------------------------------------------------------------------------------------------
## Create a document term matrix for security topic ----
sec_dtm <- clean_sec %>%
        VectorSource() %>%
        Corpus() %>%
        tm_map(removeWords, stopwords('english')) %>%
        tm_map(removeNumbers) %>%
        tm_map(removePunctuation) %>%
        tm_map(tolower) %>%
        tm_map(stripWhitespace) %>%
        tm_map(stemDocument) %>%
        TermDocumentMatrix()
```

```
## ------------------------------------------------------------------------------------------------------
## Documents and term counts in facebook topic ----
fb_dtm


## ------------------------------------------------------------------------------------------------------
## Documents and term counts in security topic ----
sec_dtm


## ------------------------------------------------------------------------------------------------------
## Popular terms- facebook ----
fb_dtm %>%
        findFreqTerms(lowfreq = 400) %>%
        head(30)


## ------------------------------------------------------------------------------------------------------
## Popular terms- security ----
sec_dtm %>%
        findFreqTerms(lowfreq = 400) %>%
        head(30)


## ----fig.cap="WordCloud for the Facebook Topic"-----------------------------------------------------
## Wordcloud- facebook ----
fb_df <- fb_dtm %>%
        as.matrix() %>%
        data.frame() %>%
        set_names("Count") %>%
        mutate(Term = row.names(.)) %>%
        arrange(desc(Count))

## Word cloud for facebook topic ----
wordcloud(
        words = fb_df$Term,
        freq = fb_df$Count,
         min.freq=1,
         max.words = 50,
         random.order = FALSE,
         colors=brewer.pal(7, "Set2"))


## ----fig.cap="WordCloud for the Security Topic"-----------------------------------------------------
## Wordcloud- security ----
sec_df <- sec_dtm %>%
        as.matrix() %>%
        data.frame() %>%
        set_names("Count") %>%
        mutate(Term = row.names(.)) %>%
        arrange(desc(Count))

## Word cloud for security topic ----
wordcloud(
```

```
words = sec_df$Term,
freq = sec_df$Count,
 min.freq=1,
 max.words = 50,
 random.order = FALSE,
 colors=brewer.pal(7, "Set2"))
```