Sentiment analysis and LSF

Replication of Lima-Lopes (2020) - Part 3

Rodrigo Esteves de Lima-Lopes
State University of Campinas
rll307@unicamp.br

Contents

1 Introduction

This script will discuss how to prepare our dictionaries and count the sentiments. My analysis replicates Lima-Lopes (2020) and is based on SFL (Systemic-Functional Linguistics) approach (Halliday and Mathissen, 2013), mostly specifically the Appraisal System (Martin and White, 2005).

Our objectives here are twofold:

1. Prepare the dictionaries
2. Process the counting and normalisation

In both cases, we are using Quanteda as our main package.

2 Packages

```
library(quanteda)
library(dplyr)
```

Each package has a function:

- `quanteda`: text analysis (counting the features)
- `dplyr`: data manipulation

3 Dictionaries

Dictionaries are formal classes which associate keys (some classificatory values) with elements in a list. The idea of a dictionary is to identify `character/strings` with a pre-defined category, in order to tell us how any of each class is present in a corpus. It is important to notice that a corpus might have one or more texts in its structure.

There are a number of possibilities, packages and methods, in R to work with dictionaries. In this tutorial, we are going to use **quanteda** for its easiness of use and because it brings tools for other analysis as we will discuss latter.

3.1 Creating the dictionaries

Our first step is to create a number of vectors with each categories we would need to analyse in the corpus.

```
adj.1 <-  c("squat", "filthy", "dirty", "behaviour", "very", "unrestricted", "awful", "black", "hungry"
  unique()

adj.2 <- c("unskilled migrant", "low-skilled", "skilled-worker", "jobless",  "qualification", "sharp mi
  unique()

adj.3 <-  c("legal", "illegal", "unlawful", "unlawfully", "asylum seekers", "refugee", "genuine refugee
  unique()

MC.1 <- c("spain", "from outside eu", "ethnic minorities", "portugal", "non-eea countries", "from europ

MC.2 <-  c("home secretary", "theresa may", "mrs. may", "cabinet source", "mr johnson", "chairman", "mr
  unique()

MC.3 <- c("particularly", "very", "dramatic", "passionately", "directly", "quickly", "completely", "cer
  unique()

vp <- c("show", "stop illegal immigrants", "stop migrants", "says", "say", "said", "squat", "stow", "st
  unique()


EM <- c("should", "would", "must", "need", "will", "can", "can't", "shouldn't", "might","may","must", "


NP.1 <- c("poorer EU countries", "poorer member states", "poorer EU member", "European economies", "eur
  unique()

NP.2 <- c("control", "borders", "border", "wave", "waves", "surge", "increase", "uncontrolled", "immedi
  unique()
```

Now we create a list with our vectors

```
dic.list <- list(adj.1, adj.2, adj.3, MC.1, MC.2, MC.3, vp, EM, NP.1, NP.2)
```

Then we name such a list

```
names(dic.list) <- c("adj.1", "adj.2", "adj.3", "MC.1", "MC.2", "MC.3", "vp", "EM", "NP.1", "NP.2")
```

Finally, we create the dictionary

```
dic.brexit <- dictionary(dic.list)
```

3.2 Creating the corpus

Before we apply the dictionary, we will create a corpus like we did in **02_Sentiment_pre_processing_network**. However, we will not lemmatise the corpus.

```
# one by one
TS.Corpus <- corpus(TS.df, text_field = 'Content')
```

```r
DS.Corpus <- corpus(DS.df, text_field = 'Content')
TT.Corpus <- corpus(TT.df, text_field = 'Content')

#General
News.df <- rbind(TT.df,TS.df, DS.df)
News.Corpus <- corpus(News.df, text_field = 'Content')

#tokens
TS.tokens  <- tokens(TS.Corpus,
                     what = "word",
                     remove_punct = TRUE,
                     remove_symbols = TRUE,
                     remove_numbers = TRUE,
                     remove_url = TRUE,
                     split_hyphens = FALSE,
                     include_docvars = TRUE,
                     padding = FALSE,
                     verbose = quanteda_options("verbose")
)

DS.tokens <- tokens(DS.Corpus,
                    what = "word",
                    remove_punct = TRUE,
                    remove_symbols = TRUE,
                    remove_numbers = TRUE,
                    remove_url = TRUE,
                    split_hyphens = FALSE,
                    include_docvars = TRUE,
                    padding = FALSE,
                    verbose = quanteda_options("verbose")
)
TT.tokens <- tokens(TT.Corpus,
                    what = "word",
                    remove_punct = TRUE,
                    remove_symbols = TRUE,
                    remove_numbers = TRUE,
                    remove_url = TRUE,
                    split_hyphens = FALSE,
                    include_docvars = TRUE,
                    padding = FALSE,
                    verbose = quanteda_options("verbose")
)
News.tokens <- News.Corpus |>
  tokens(what = "word",
         remove_punct = TRUE,
         remove_symbols = TRUE,
         remove_numbers = TRUE,
         remove_url = TRUE,
         split_hyphens = FALSE,
         include_docvars = TRUE,
         padding = FALSE,
         verbose = quanteda_options("verbose")
  )
```

3.3 Counting the occurences

Now let us do the actual counting

```
News.counting <- dfm(News.tokens, dictionary = dic.brexit)
DS.counting <- dfm(DS.tokens, dictionary = dic.brexit)
TS.counting <- dfm(TS.tokens, dictionary = dic.brexit)
TT.counting <- dfm(TT.tokens, dictionary = dic.brexit)
```

Saving as data frame for latter use

```
DS.counting <- convert(DS.counting, to = "data.frame")
TS.counting <- convert(TS.counting, to = "data.frame")
TT.counting <- convert(TT.counting, to = "data.frame")
News.counting <- convert(News.counting, to = "data.frame")
```

3.3.1 Normalising and preparing the analysis

Now we are going to add a total column and use the name of the documents to name de rows

```
News.counting$Total <- rowSums(News.counting[,2:10])
rownames(News.counting) <- News.counting$doc_id
```

Now we normalise and round the data

```
News.Cal  <- News.counting |>
  mutate_at(
    vars(-matches(c("Total", "doc_id"))),
    ~ .x * 100 / Total) |>
  mutate_at(
    vars(-matches(c("Total", "doc_id"))),
    ~ round(.x, 2)
  )
```