Sujet MMD 2016

MMD 2015

SB

17 March 2016

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`.

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:

Useful libraries and references

Tutoriel Kaggle
rstudio

```
libs <- c("ggplot2","knitr","plyr", "dplyr","magrittr","foreach","
result <-plyr::laply(.data=libs,
                     .fun=require,
                     character.only=TRUE,
                     quietly=TRUE,
                     warn.conflicts=FALSE)

tmlibs <- c("tm","NLP","tm.plugin.mail","textreuse","stringr")
resulttm <-plyr::laply(.data=tmlibs,
                     .fun=require,
                     character.only=TRUE,
                     quietly=TRUE,
                     warn.conflicts=FALSE)
```

Enron data

General presentation

- http://enrondata.org/content/research/
- http://www-2.cs.cmu.edu/~enron
- 619,446 messages belonging to 158 users
- The maildir folder is organised in 150 subfolders corresponding to different users.

Advices about the things to do before text mining

http://rforwork.info/2013/11/03/a-rather-nosy-topic-model-analysis-of-the-enron-email-corpus/

Archive exploration

Unpack the enron dataset in a subdirectory `maildir` of your project

Inspect the archive

```
enronpath <- paste(getwd(),"maildir",sep="/")
list_folders <- dir(enronpath)

head(dir(paste(getwd(),"maildir",sep="/")))
```

Spot the differences (if any)

between the subfolders (`_sent_mail`,`sent_items` and `_sent`)

Building a corpus

Building and shingling

A corpus is a collection of documents. The corpus is built from a collection of files using function `VCorpus` from package `tm` (or using a comparable function from `textreuse`).

Handling emails

The documents we handle are structured as mail documents, a dedicated plugin to `tm` helps handling the documents. It provides us with a dedicated reader that strips a lot of syntactical information from mails. Use the `readMail` reader from package `tm.plgin.email`.

Focus on mails *sent* by users

Shingling

tm provides many cleaning functions that operate on a corpus. Assume the corpus is named fooCorpus

```
Sys.time()
fooCorpus <- fooCorpus %>%
        tm_map(content_transformer(tolower)) %>%
        tm_map(removeWords,stopwords("english")) %>%
        tm_map(stemDocument) %>%
        tm_map(removePunctuation) %>%
        tm_map(stripWhitespace) %>%
        tm_map(removeNumbers)
Sys.time()
```

This is functional programming in disguise (tm_map is a text mining customization of map)
%>% is not a pipe. It is provided by magrittr

Building a document term matrix and exporatory analysis

Both kind of matrices inherit from triplet matrices (package `slam` Sparse Lightweight Arrays and Matrices).

```
# This may require more than a couple of seconds.
Sys.time()
foodtm <- DocumentTermMatrix(x=fooCorpus)
footdm <- TermDocumentMatrix(x=fooCorpus)
Sys.time()
```

Exploratory analysis of the term-document matrix

- Distribution of number of terms per document
- Distribution of number of occurrences of terms in corpus per words
- Fitting the Zipf distribution `Zipf_plot(foodtm)`
- Fitting a Heaps distribution `Heaps_plot(dtm)`
- Spotting some spurious terms : hhingling did not completely clean the corpus

```
findFreqTerms(tdm)[stringr::str_length(findFreqTerms(tdm))>25]
```

Looking for approximate nearest neighbors using LSH in `textreuse`

A hashing package for text mining `textreuse`

- git repo for textreuse
- Min hashing for dummies

Code inspection

LSH in `textreuse`

Preparing a corpus for LSH

textreuse has its own version of Corpora

The corpus building capabilities of textreuse seem less sophisticated than the corpus building capabilities of of tm

Transforming a tm corpus into a textreuse corpus

```
# takes time (10minutes on imac)
# add a progressbar
# try first on a small subset of fooCorpus ...
Sys.time()
barCorpus <- foreach (doc=fooCorpus, .combine="c") %do%
  TextReuseCorpus(text=doc$content,
                  tokenizer = tokenize_words,
                  progress = TRUE,
                  keep_tokens = TRUE,
                  keep_text = TRUE,
                  skip_short = TRUE)
Sys.time()
```

Analysis

Perform approximate nearest neighbor analysis

- Choose a grid of target similarity thresholds `seq(.9,.1,by=-.1)`
- For each similarity threshold, detect candidate pairs by tuning min-hashing
- Compute the connected components of the approximate nearest neighbor graphs (you may use package `igraph`)
- Associate typical terms with connected components

OR

Performing Latent Dirichlet Analysis

- You may also wish to attempt Latent Dirichlet Analysis using for example `topicmodels`