Analyse de sentiments avec sentometrics

François Pelletier

Note : Ce document combine les quatre fichiers Rmd originaux. Les packages R `RSQLite` et `sentometrics` sont nécessaires pour exécuter le code, mais ne sont pas installés dans l'environnement actuel. Tout le code R est donc présenté avec `eval: false` à titre de référence.

Préparation des données Sentometrics

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library("jsonlite")
library("tidyverse")
library("RSQLite")
library("DBI")
library("lubridate")
```

```
blog_exemple <- jsonlite::read_json("google_news_blogs/blogs/blogs_0000001.json")
```

- Identifiant

```
blog_exemple$uuid
```

- Date

```
blog_exemple$published
```

- Contenu

```
blog_exemple$text
```

- Features
  - Persons

```
blog_exemple$entities$persons %>% sapply(FUN = function(x) x$name)
```

- Organizations

```
blog_exemple$entities$organizations %>% sapply(FUN = function(x) x$name)
```

- Locations

```
blog_exemple$entities$locations %>% sapply(FUN = function(x) x$name)
```

Dataframes

Core

```
extract_names <- function(list_entities){
  name_entities <- list_entities %>% sapply(FUN = function(x) x$name)
  if (length(name_entities) > 0)
    return(name_entities)
  else
    return(NA)
}

generer_core_df <- function(json_contents){
  tibble(uuid = json_contents$uuid %>% coalesce(""),
       site = json_contents$thread$site %>% coalesce(""),
       site_type = json_contents$thread$site_type %>% coalesce(""),
       country = json_contents$thread$country %>% coalesce(""),
       published = lubridate::as_datetime(json_contents$thread$published) %>% coalesce(ISOdat
       title_full = json_contents$thread$title_full %>% coalesce(""),
       text = json_contents$text %>% coalesce(""))
}

generer_entities_df <- function(json_contents){
  this_df <- bind_rows(tibble(uuid = json_contents$uuid,
                              entity_type="persons",
```

```
                              entity=json_contents$entities$persons %>%
                                extract_names) ,
                            tibble(uuid = json_contents$uuid,
                              entity_type="organizations",
                              entity=json_contents$entities$organizations %>%
                                extract_names),
                            tibble(uuid = json_contents$uuid,
                              entity_type="locations",
                              entity=json_contents$entities$locations %>%
                                extract_names))
  this_df <- na.omit(this_df)
}
```

```
core_df <- generer_core_df(blog_exemple)
core_df %>% glimpse
```

```
entities_df <- generer_entities_df(blog_exemple)
entities_df %>% glimpse
```

Création des schémas de la base de données

```
if(file.exists("google_news.sqlite"))
  file.remove("google_news.sqlite")
con = dbConnect(drv = RSQLite::SQLite(), dbname="google_news.sqlite")
dbCreateTable(con,"core",core_df)
dbCreateTable(con,"entities",entities_df)
```

Importation des données

```
file_blogs <- list.files(path = "google_news_blogs/blogs",pattern = "*.json",full.names = TRU
file_news <- list.files(path = "google_news_blogs/news",pattern = "*.json",full.names = TRUE)
```

```
traiter_json <- function(file_path){

  json_contents <- jsonlite::read_json(file_path)
  core_df <- generer_core_df(json_contents)
  entities_df <- generer_entities_df(json_contents)
```

```
  dbAppendTable(con,"core",core_df)
  dbAppendTable(con,"entities",entities_df)
}
```

Traitement des fichiers

```
i <- 0 # itérateur
for (file_blog in file_blogs){
  if(!(i %% 1000)){
    print(paste0(i,": Traitement de ",file_blog))
  }
  traiter_json(file_blog)
  i <- i+1
}

ii <- 0 # itérateur
for (file_article in file_news){
  if(!(ii %% 1000)){
    print(paste0(ii,": Traitement de ",file_article))
  }
  traiter_json(file_article)
  ii <- ii+1
}
```

Formatage des données

```
library("tidyverse")
library("RSQLite")
library("DBI")
library("sentometrics")
```

```
con = dbConnect(drv = RSQLite::SQLite(), dbname="google_news.sqlite")
```

Aperçu

```
tbl(con,"core") %>% head(10) %>% collect() %>% glimpse()
```

Top 10 de modalités

```
top_10_sites <- tbl(con,"core") %>%
  select(site) %>%
  group_by(site) %>%
  count() %>%
  arrange(desc(n)) %>%
  head(10) %>%
  collect()
saveRDS(top_10_sites,"top_10_sites.RDS")
top_10_sites
```

```
top_10_country <- tbl(con,"core") %>%
  select(country) %>%
  mutate(country = ifelse(country=="","XX",country)) %>%
  group_by(country) %>%
  count() %>%
  arrange(desc(n)) %>%
  head(10) %>%
  collect()
saveRDS(top_10_country,"top_10_country.RDS")
top_10_country
```

Entities features

Compteurs

```
entities_count <- tbl(con,"entities") %>% group_by(uuid,entity_type) %>% count %>% collect()
```

```
entities_count_t <- entities_count %>% reshape2::dcast(uuid~paste0("entity_",entity_type),fu
entities_count_t %>% head(10) %>% glimpse()
```

Aperçu

```
tbl(con,"entities") %>% group_by(entity_type, entity) %>% count() %>% arrange(desc(n)) %>% he
```

Core features

```
core_features_corpus <- tbl(con,"core") %>% collect() %>%
  transmute(
    id=uuid,
    date=lubridate::as_datetime(published),
    texts=paste(title_full,text,sep = "\n"),
    # Site
    site_01 = ifelse(site==top_10_sites$site[1],1,0),
    site_02 = ifelse(site==top_10_sites$site[2],1,0),
    site_03 = ifelse(site==top_10_sites$site[3],1,0),
    site_04 = ifelse(site==top_10_sites$site[4],1,0),
    site_05 = ifelse(site==top_10_sites$site[5],1,0),
    site_06 = ifelse(site==top_10_sites$site[6],1,0),
    site_07 = ifelse(site==top_10_sites$site[7],1,0),
    site_08 = ifelse(site==top_10_sites$site[8],1,0),
    site_09 = ifelse(site==top_10_sites$site[9],1,0),
    site_10 = ifelse(site==top_10_sites$site[10],1,0),
    # Site type
    is_blog = ifelse(site_type=="blogs",1,0),
    # Country
    country_01 = ifelse(country==top_10_country$country[1],1,0),
    country_02 = ifelse(country==top_10_country$country[2],1,0),
    country_03 = ifelse(country==top_10_country$country[3],1,0),
    country_04 = ifelse(country==top_10_country$country[4],1,0),
    country_05 = ifelse(country==top_10_country$country[5],1,0),
    country_06 = ifelse(country==top_10_country$country[6],1,0),
    country_07 = ifelse(country==top_10_country$country[7],1,0),
    country_08 = ifelse(country==top_10_country$country[8],1,0),
    country_09 = ifelse(country==top_10_country$country[9],1,0),
    country_10 = ifelse(country==top_10_country$country[10],1,0)
  ) %>% left_join(entities_count_t,by=c("id"="uuid")) %>% sento_corpus()
```

```
saveRDS(core_features_corpus,file = "core_features_corpus.RDS")
```

Analyse BD

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library("sentometrics")
library("tidyverse")
library("plotly")
```

```
core_features_corpus.RDS <- readRDS("core_features_corpus.RDS")
top_10_country <- readRDS("top_10_country.RDS")
top_10_sites <- readRDS("top_10_sites.RDS")
corpusSample <- quanteda::corpus_sample(core_features_corpus.RDS, size = 200)
```

Définition des lexiques

```
data("list_valence_shifters", package = "sentometrics")
data("list_lexicons", package = "sentometrics")

lexIn <- list_lexicons[c("FEEL_en_tr")]
valIn <- list_valence_shifters[["en"]]

l1 <- sento_lexicons(lexIn,valIn)
```

Calcul des sentiments

```
c_sentiments_sample <- compute_sentiment(x = corpusSample,
                                         lexicons = l1,
                                         how = "counts",
                                         nCore = 8)
c_sentiments_sample
```

```
c_control_compute <- ctr_agg(howWithin = "proportional",
                             howDocs = "equal_weight",
                             howTime = "equal_weight",
                             lag = 7,
                             by = "day")
```

```
c_sentiments <- sento_measures(sento_corpus = core_features_corpus.RDS,
                               lexicons = l1,
                               ctr = c_control_compute)
```

```
c_measures <- as.data.table(c_sentiments)
```

```
c_measures_g <- measures_global(c_sentiments)
```

Sentiment par site

```
c_measures_melt <- c_measures %>%
  select(date,starts_with("FEEL_en_tr--site")) %>%
  `colnames<-`(c("date",top_10_sites$site)) %>%
  melt(id="date",variable.name = "site")
plot_site <- ggplot(data=c_measures_melt,
       aes(x=date, y=value, colour=site))+
  geom_line()
ggplotly(plot_site)
```

Sentiment par pays

```
c_measures_melt <- c_measures %>%
  select(date,starts_with("FEEL_en_tr--country")) %>%
  `colnames<-`(c("date",top_10_country$country)) %>%
  melt(id="date",variable.name = "country")
plot_country <- ggplot(data=c_measures_melt,
       aes(x=date, y=value, colour=country))+
  geom_line()
ggplotly(plot_country)
```

Sentiment par compteur d'entités

```
c_measures_melt <- c_measures %>%
  select(date,starts_with("FEEL_en_tr--entity")) %>%
  melt(id="date",variable.name = "entity")
```

```r
plot_entity <- ggplot(data=c_measures_melt,
       aes(x=date, y=value, colour=entity))+
  geom_line()
ggplotly(plot_entity)
```

Sentometrics — Présentation CAAMD

```r
knitr::opts_chunk$set(echo = FALSE)
```

Sentometrics

Présentation basée sur un atelier présenté par Keven Bluteau à **R à Québec 2019**.

D'où vient le nom ?

- Mélange d'analyse de sentiments et d'économétrie
- Type d'analyse de plus en plus fréquent en finance, en marketing et en politique.

Quelle forme prend le produit ?

- Package R
- Services conseils ($)

Article de référence

The R Package sentometrics to Compute, Aggregate and Predict with Textual Sentiment

Pourquoi ?

Les données qualitatives sont de plus en plus utilisées pour raffiner les analyses prédictives, car elle donnent une rétroaction sur la passé et le futur, contrairement aux données numériques qui donnent toujours une image passée ou présente d'une réalité.

Historique des packages R

- tm (2008)
- openNLP (2016)
- quanteda (2018)
- tidytext (2016)

Les bases

Sentometrics est construit sur `quanteda` et `data.table`. Les modèles sont estimés avec `glmnet` et `caret`.

Les fonctionnalités

Functionality	Functions	Output
1. Corpus management		
(a) Creation	`sento_corpus()`	*sentocorpus*
(b) Manipulation	**quanteda** corpus functions (*e.g.*, `docvars()`, `corpus_sample()`, or `corpus_subset()`)	
(c) Conversion	`to_sentocorpus()`	
(d) Features generation	`add_features()`	
2. Sentiment computation		
(a) Lexicon management	`sento_lexicons()`	*sentolexicons*
(b) Computation	`compute_sentiment()`	*sentiment*
(c) Manipulation	`sentiment_bind()`, `to_sentiment()`	
(d) Summarization	`peakdocs()`	
3. Sentiment aggregation		
(a) Specification	`ctr_agg()`	
(b) Aggregation	`sento_measures()`, `aggregate.sentiment()`	*sentomeasures*
(c) Manipulation	`measures_delete()`, `measures_fill()`, `measures_global()`, `measures_merge()`, `measures_select()`, `measures_subset()`, `diff()`, `scale()`, `get_measures()`	
(d) Visualization	`plot.sentomeasures()`	
(e) Summarization	`summary()`, `peakdates()`, `nobs()`, `nmeasures()`, `get_dimensions()`, `get_dates()`	
4. Modelling		
(a) Specification	`ctr_model()`	
(b) Estimation	`sento_model()`	*sentomodel*, *sentomodeliter*
(c) Prediction	`predict.sentomodel()`	
(d) Diagnostics	`summary()`, `get_loss_data()`, `attributions()`	*attributions*
(e) Visualization	`plot.sentomodeliter()`, `plot.attributions()`	

Table 1: Taxonomy of the R package **sentometrics**. This table displays the functionalities of the **sentometrics** package along with the associated functions and S3 output objects.

Calcul des sentiments

- Unigrammes: somme pondérée des scores pour tous les mots apparaissant dans un lexique
- Bigrammes avec décalage de polarité (valence shifting)
  - Va intégrer l'impact de mots négatifs par exemple (good vs. not good)
- Groupements avec décalage de polarité (valence shifting)
  - Fenêtre mobile avant et après le mot

Aggrégation des sentiments

Les sentiments sont aggrégés en deux phases:

- Pour tous les documents durant une période donnée
- Pour plusieurs périodes consécutives

Création des métriques

- Aggrégation à l'intérieur du document (howWithin)
- Aggrégation à l'intérieur d'un intervalle de temps (howDocs)
- Aggrégation au fil du temps (howTime)

Modélisation

- Régression avec Elastic Net
- Configuration des hyperparamètres avec `ctr_model()`
- Entraînement avec `sento_model()`