05 Text Mining

Fabian Blasch

05/31/2022

1 Load Data

```
# source AUX
source("./../Misc/Auxilliary.R")

# packages
get.package(c("bizdays", "lubridate", "tidytext", "stopwords", "stringi",
              "SnowballC", "wordcloud"))

# load data
auctions <- readRDS("./../../Data/Bid Tab RDS/Bid_Tabs_Cleaned.RDS")
```

2 Final Checks before Mining

```
# check for descriptions that are shorter than 50 characters after cleaning
lapply(auctions, \(x){

  # over project ID
  lapply(x, \(y){

    # Description length
    Dlen <- y$Text["Contract Description"] |> nchar()

    # required min char count
    dlen_check <- Dlen > 50

    # return
    list("Description" = Dlen,
         "Dlcheck" = dlen_check)

  })

}) -> filter_dat

# pull index

lapply(filter_dat, \(x){

  sapply(x, \(x){

    # extr
    x[["Dlcheck"]]
```

```
  })

}) -> ind

# files that will be removed
(!unlist(ind)) |> sum()
```

```
## [1] 4
```

```
# subset
Map(\(au, ind) au[ind], auctions, ind) -> auctions
```

3 Data Mining

3.1 Tokenization

```
# exmpl
desc <- pull_varT(auctions, "Contract Description")

# obtain Contract ID
CID <- strsplit(names(desc), "[.]") |> sapply("[[", 2)

# df for tokenization
desc_tm <- data.frame("Num" = seq_along(CID), "Contract_id" = CID,
                      "Description" = desc)

# generate tokens
desc_tok <- unnest_tokens(desc_tm, word, Description)

# unique words
uniq_words <- desc_tok[, "word"] |> unique()

# num
uniq_words |> length()
```

```
## [1] 3323
```

3.2 Mispelled Words and Stopwords

```
# potentially misspelled words
miss <- hunspell::hunspell(uniq_words)

# to vector
miss_vec <- unlist(unique(miss))

# suggestions
sugg <- hunspell::hunspell_suggest(miss_vec)

# pick first suggestion and merge
sugg_1 <- sapply(sugg, "[", 1)

# merge and export
# openxlsx::write.xlsx(as.data.frame(cbind(miss_vec, sugg_1)),
```

```
#                       "./../../Data/Misc Data/Word_corr.xlsx")

# import
corr <- openxlsx::read.xlsx("./../../Data/Misc Data/Word_corr.xlsx")
corr[is.na(corr[, "sugg_1"]), "sugg_1"] <- ""

# stopwords
do.call(c, lapply(c("snowball", "stopwords-iso",
                    "smart", "marimo", "nltk"), \(x){

           # generate stopwords
           stopwords(language = "en", source = x)

         })) |> unique() -> stopw

# add stopwords to removal
corrs <- rbind(corr, cbind("miss_vec" = stopw,"sugg_1" = ""))

# remove
descrm <- stringi::stri_replace_all_regex(desc,
                                          paste0("\\b", corrs[, "miss_vec"], "\\b"),
                                          corrs[, "sugg_1"],
                                          vectorize_all = FALSE)

# this stop word list still results in noisy descriptions
# look at most frequent words that are not of importance

# df for tokenization
desc_tm_r2 <- data.frame("Num" = seq_along(CID), "Contract_id" = CID,
                         "Description" = descrm)

# generate tokens
desc_tok_r2 <- unnest_tokens(desc_tm_r2, word, Description)

# frequ
f_tab_words <- sort(table(desc_tok_r2$word), decreasing = TRUE)

# write to exceland check
# openxlsx::write.xlsx(data.frame(names(f_tab_words), f_tab_words),
#                       "./../../Data/Misc Data/StopWord_corr.xlsx")

# read back
corr_2 <- openxlsx::read.xlsx("./../../Data/Misc Data/StopWord_corr.xlsx")
corr_2[is.na(corr_2[, "Replacement"]), "Replacement"] <- ""

# second round of removal
descrm_2 <- stringi::stri_replace_all_regex(descrm,
                                            paste0("\\b", corr_2[, "Word"], "\\b"),
                                            corr_2[, "Replacement"],
                                            vectorize_all = FALSE)
# retokenize

# df for tokenization
```

```r
desc_tm_r3 <- data.frame("Num" = seq_along(CID), "Contract_id" = CID,
                         "Description" = descrm_2)

# generate tokens
desc_tok_r3 <- unnest_tokens(desc_tm_r3, word, Description)

# remove digits
desc_tok_r3 <- desc_tok_r3[grepl(".*(\\D).*", desc_tok_r3[, "word"]),]
```

3.3 Stemming

```r
# stemming
desc_stem <- cbind(desc_tok_r3,
                   "Stem" = SnowballC::wordStem(desc_tok_r3[, "word"],
                                                language = "en"))
# unique stems
uniq_stem <- tapply(desc_stem[, "Stem"], desc_stem[, "Num"], unique)

# length of list in each year
split_ind <- sapply(auctions, length)

# ind
split_lst <- list(1:120,
                  121:234,
                  235:339,
                  340:452,
                  453:474)

# concatinate
Map(\(ind, auc){

  # temp stem
  tmp <- uniq_stem[ind]

  # concatinate
  Map(\(a, s){

    # assign
    a[["Stem"]] <- s

    # return
    return(a)

  }, auc, tmp)

}, split_lst, auctions) |> setNames(2015:2019) -> Bid_Tab_Stem

# write
# saveRDS(Bid_Tab_Stem, "./../../Data/Bid Tab RDS/Bid_Tab_Stem.RDS")

# glimpse into description and stem

# lapply(Bid_Tab_Stem, \(x){
```

```
#
#   lapply(x, \(y){
#
#     print(y$Text["Contract Description"])
#     print(paste(y$Stem, collapse = ","))
#
#   })
#
# })
```

4 Some Illustrations

```
# table of stems
stem_tab <- sort(table(desc_stem$Stem), decreasing = TRUE)

# align
par(mfrow = c(1, 2), mar = c(4, 3, 2, 2))

# top 20 most common stemmed words
barplot(stem_tab[1:40], las = 2,
        ylim = c(0, 250), cex.names = 0.5, col = 4, main = "Top 40 Stemmed Words")

# wordcloud
wordcloud(names(stem_tab), stem_tab, max.words = 50, random.order = FALSE, rot.per = 0.35,
          colors = brewer.pal(8, "Dark2"))
```