Tratamento do Deck de Cultura

ASR

2023-02-08

# Instalação e Carregamento dos Pacotes  
pacotes <- c("tm", "dplyr", "wordcloud", "stringr", "tidyverse", "tidytext", "tibble", "utils")  
  
if(sum(as.numeric(!pacotes %in% installed.packages())) != 0){  
 instalador <- pacotes[!pacotes %in% installed.packages()]  
 for(i in 1:length(instalador)) {  
 install.packages(instalador, dependencies = T)  
 break()}  
 sapply(pacotes, require, character = T)   
} else {  
 sapply(pacotes, require, character = T)   
}

## tm dplyr wordcloud stringr tidyverse tidytext tibble utils   
## TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

# Lendo o deck de cultura  
deck\_cultura\_df <- tibble::tibble(text = readLines("deck de cultura.txt",   
 encoding = 'UTF-8'))  
dplyr::summarise(deck\_cultura\_df, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 134

utils::str(deck\_cultura\_df)

## tibble [134 × 1] (S3: tbl\_df/tbl/data.frame)  
## $ text: chr [1:134] "A nossa Cultura" "Este documento expressa em parte quem já somos, e em parte quem aspiramos ser" "Neste documento apresentamos:" "Nosso Propósito e Nossa Missão" ...

# Stopwords proveniente do Python  
stopwords\_py <- read.csv("stopwords\_pt\_py.csv",   
 encoding = 'UTF-8',   
 header = TRUE)  
# Stopwords proveniente do R  
stopwords\_R <- tibble::tibble(word = stopwords('pt'))  
# Unindo as duas listas de stopwords  
stopwords\_full\_join <- dplyr::full\_join(stopwords\_py,  
 stopwords\_R,  
 by = "word",  
 keep = FALSE)  
dplyr::summarise(stopwords\_full\_join, observações=n())

## observações  
## 1 207

utils::str(stopwords\_full\_join)

## 'data.frame': 207 obs. of 1 variable:  
## $ word: chr "a" "à" "ao" "aos" ...

# Tratamento do Deck de Cultura  
deck\_cultura\_df$text <- deck\_cultura\_df$text %>%  
 # Aplicando função tolower a todas as linhas no tibble  
 lapply(tolower) %>%  
 # Removendo caracter especial  
 lapply(gsub, pattern = '[^[:alnum:]]', replace = ' ') %>%  
 # Removendo "não palavras"  
 lapply(gsub, pattern = '\\W', replace = ' ') %>%  
 # Removendo dígitos  
 lapply(gsub, pattern = '\\d', replace = '') %>%  
 # Removendo stopwords  
 lapply(tm::removeWords, stopwords\_full\_join$word) %>%  
 # Caso necessite remover acentuação do português  
 #lapply(iconv, from = 'UTF-8', to = 'ASCII//TRANSLIT') %>%  
 lapply(tm::stripWhitespace) %>%  
 # Multiplos espaços em branco são convertidos em um único espaço  
 trimws()  
  
dplyr::summarise(deck\_cultura\_df, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 134

utils::str(deck\_cultura\_df)

## tibble [134 × 1] (S3: tbl\_df/tbl/data.frame)  
## $ text: chr [1:134] "cultura" "documento expressa parte parte aspiramos" "neste documento apresentamos" "propósito missão" ...

#dplyr::glimpse(deck\_cultura\_df)  
#utils::head(deck\_cultura\_df, n=15)   
#print(deck\_cultura\_df)   
#dim(deck\_cultura\_df)   
#names(deck\_cultura\_df)

# Criando corpus\_deck separando cada palavra em um novo registro  
corpus\_deck <- deck\_cultura\_df %>% tidytext::unnest\_tokens(word,  
 text,  
 token = "words")  
dplyr::summarise(corpus\_deck, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 1298

utils::str(corpus\_deck)

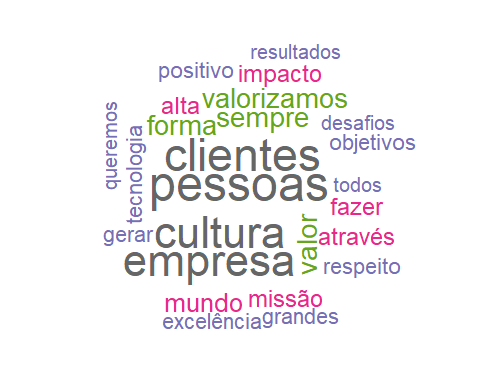
## tibble [1,298 × 1] (S3: tbl\_df/tbl/data.frame)  
## $ word: chr [1:1298] "cultura" "documento" "expressa" "parte" ...

# Contando aparições de palavras usando uma função dplyr  
word\_count <- corpus\_deck %>% dplyr::count(word,   
 sort = TRUE)  
dplyr::summarise(word\_count, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 770

utils::str(word\_count)

## tibble [770 × 2] (S3: tbl\_df/tbl/data.frame)  
## $ word: chr [1:770] "pessoas" "clientes" "cultura" "empresa" ...  
## $ n : int [1:770] 19 18 17 17 10 10 10 10 9 8 ...



# Gerar ngrams para ser usado na análise de sentimentos  
# Número de palavras no ngram  
k\_grams <- 2  
# Gerando ngram baseada em k\_grams  
kgrams <- deck\_cultura\_df %>%   
 tidytext::unnest\_tokens(output = word,   
 input = text,   
 token = 'ngrams',   
 n = k\_grams) %>%  
 tidyr::separate(word, c('word1', 'word2'), sep = ' ') %>%  
 dplyr::filter(!word1 %in% stopwords\_full\_join) %>%  
 dplyr::filter(!word2 %in% stopwords\_full\_join) %>%  
 tidyr::unite(word, word1, word2, sep = ' ', na.rm = TRUE) %>%  
 dplyr::count(word, sort = TRUE)  
# Multiplos espaços em branco são convertidos em um único espaço  
kgrams$word <- kgrams$word %>% trimws()  
# Remove "NA" e registros vazios  
kgrams <- kgrams[!apply(is.na(kgrams) | kgrams$word == "", 1, all),]  
  
dplyr::summarise(kgrams, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 1072

utils::str(kgrams)

## tibble [1,072 × 2] (S3: tbl\_df/tbl/data.frame)  
## $ word: chr [1:1072] "impacto positivo" "alta performance" "gerar valor" "ajudar organizações" ...  
## $ n : int [1:1072] 7 5 5 3 3 3 3 3 3 3 ...

# Número de ngram no dicionário de sentimentos  
max\_ngram <- 10

# Gerando dicionário para análise de sentimentos  
# Adicionando ao dicionário de sentimentos as "max\_words" palavras que mais aparecem  
dictionary <- tibble::tibble(line = 1:max\_words,  
 text = word\_count$word[1:max\_words])  
# Adicionando ao dicionário de sentimentos os "max\_ngram" kgrams que mais aparecem  
dictionary <- dictionary %>%  
 tibble::add\_row(line = (max\_words+1):(max\_words+max\_ngram),  
 text = kgrams$word[1:max\_ngram])  
dictionary <- dictionary %>%   
 tibble::add\_column(value = 0)  
  
# Definindo valor 5 de sentimento para os ngrams  
dictionary$value[(max\_words+1):(max\_words+max\_ngram)] <- 5  
  
# Definindo valor 4 de sentimento para as palavras  
dictionary$value[1:max\_words] <- 4  
# Criando valor aleatório entre 0 e 4  
#dictionary$value[1:max\_words] <- floor(runif(max\_words, min = 0, max = 4))  
  
dplyr::summarise(dictionary, observações=n())

## # A tibble: 1 × 1  
## observações  
## <int>  
## 1 35

utils::str(dictionary)

## tibble [35 × 3] (S3: tbl\_df/tbl/data.frame)  
## $ line : int [1:35] 1 2 3 4 5 6 7 8 9 10 ...  
## $ text : chr [1:35] "pessoas" "clientes" "cultura" "empresa" ...  
## $ value: num [1:35] 4 4 4 4 4 4 4 4 4 4 ...