

Mineração de texto aplicada à Lei de Acesso à informação - LAI

Packages for this routine

BASE DE DADOS E ANÁLISE EXPLORATÓRIA

Importação dos dados

Caminho do projeto

```
PATH = "../proj_eSIC_v10/textmining_pt/DATA/"
```

- Pedidos e-SIC

```
FILE = "/DATA/relatorio_pedidos.ods"
db_raw = readODS::read_ods(file = paste0(PATH,FILE), sheet = 1); # dim(db_raw)
dbnames = db_raw[1,]; db_raw = db_raw[-1,];
colnames(db_raw) = c("ID", "DataRegistro", "DATA_PRAZOATEND", "DESCRI_PEDIDO",
                    "RESUMO_PEDIDO", "DATA_RESPOSTA")
#View(head(db_raw))
LAI = db_raw
```

- Respostas e-SIC (DIRETORIAS EPE)

```
FILE1 = "DATA/relatorio_respostas.xlsx"
db1_raw = readxl::read_excel(paste0(PATH,FILE1), sheet = "DADOS", col_names = TRUE);
# dim(db1_raw); names(db1_raw)
colnames(db1_raw) = c("ID", "DATA", "SOLICITACAO", "DIRETORIA", "DATA_RESPOSTA")
#View(head(db1_raw))
LAI1 = db1_raw
```

- Stopwords

```
FILE2 = "DATA/stopwords_PT_FINAL.csv"
stopwords_pt = read.csv(paste0(PATH,FILE2), sep = ';', header = F, encoding = "UTF-8")
stopwords_pt = stopwords_pt[,-2];
cat(paste0("O nosso vetor de stopwords contém ",length(stopwords_pt), " palavras únicas"))
```

```
## O nosso vetor de stopwords contém 618 palavras únicas
```

```
## dim(stopwords_pt); class(stopwords_pt)
stopwords_pt = as.character(stopwords_pt)
stopwords_pt[1:14]
```

```
## [1] "a"      "à"      "acerca" "acesso" "adeus"  "agora"  "aí"
## [8] "ainda"  "alem"   "além"   "algmas" "algo"   "algumas" "alguns"
```

- Dicionário > BASE DE DADOS - REAL PRO TEXTO DO TCC

Dicionário de variáveis - PEDIDOS

```
dicionario = "DATA/Dicionario-Dados-Exportacao.txt"
dic_pedidos = read.delim(dicionario, sep = "-", skip = 3, header = FALSE, nrow = 21) %>%
  select(-V1)
```

```
colnames(dic_pedidos) = c("Nome das variáveis", "Tipo e descrição da variável")
#dimnames(dic_pedidos); View(dic_pedidos)
```

Dicionário de variáveis - RECURSOS

```
dic_recursos = read.delim(dicionario, sep = "-", skip = 30, header = FALSE, nrows = 17) %>%
  select(-V1)
colnames(dic_recursos) = c("Nome das variáveis", "Tipo e descrição da variável")
#dimnames(dic_recursos); View(dic_recursos)
```

Dicionário de variáveis - SOLICITANTES

```
dicionario = "DATA/Dicionario-Dados-Exportacao.txt"
dic_solicitantes = read.delim(file = dicionario, sep = "-", skip = 53, header = FALSE, nrows = 10) %>%
  select(-V1)
colnames(dic_solicitantes) = c("Nome das variáveis", "Tipo e descrição da variável")
#dimnames(dic_solicitantes); View(dic_solicitantes)
```

Pré-processamento dos dados

Pedidos por diretoria

- Tabela 01 número de solicitações/pedidos de informação

```
LAI1 %>%
  count(DIRETORIA, sort = TRUE, name = "total_pedidos") %>%
  kable("latex", caption = "Quantitativo de solicitações por Diretoria/EPE via e-SIC",
        booktabs = T) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 1: Quantitativo de solicitações por Diretoria/EPE via e-SIC

DIRETORIA	total_pedidos
DEA	210
DEE	197
DGC	115
DPG	24
OUTROS	19
SIC	1

```
diretorias0 = levels(as.factor(LAI1$DIRETORIA))
```

Verificamos a existência de 5 diretorias, sendo elas: *DEA*, *DEE*, *DGC*, *DPG*, *SIC* e *OUTROS*. Essa última é devido a existência de informações solicitadas que não são de competência direta de nenhuma das cinco diretorias, daí a necessidade de uma última categoria *OUTROS* para atender essas demandas.

A seguir, um passo importante de reclassificação será executado devido ao número pequeno de solicitações para a diretoria *SIC*. Apenas uma solicitação existente no nosso banco de dados para essa diretoria. Iremos, portanto, unificar essa demanda à categoria *OUTROS*.

- Respostas e-SIC - Reclassificação Diretorias

```
LAI1 = LAI1 %>%
  mutate(DIRETORIA = ifelse(DIRETORIA == diretorias0[6], diretorias0[5], DIRETORIA))
diretorias = levels(as.factor(LAI1$DIRETORIA))
```

```
#dim(LAI1)
#View(head(LAI1))
```

- Tabela 02 número de solicitações/pedidos de informação - após reclassificação

```
pedidos_diretoria = LAI1 %>%
  count(DIRETORIA, sort = TRUE, name = "total_pedidos")
pedidos_diretoria %>%
  kable("latex", caption = "Quantitativo de solicitações por Diretoria/EPE via e-SIC - após reclassificação",
        booktabs = T) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 2: Quantitativo de solicitações por Diretoria/EPE via e-SIC - após reclassificação

DIRETORIA	total_pedidos
DEA	210
DEE	197
DGC	115
DPG	24
OUTROS	20

As constatações anteriores foram feitas apenas na base de dados referente às respostas, donde temos a classificação das diretorias responsáveis por responder cada uma das demandas. É necessário, agora, unificar as bases de dados pertinentes a solicitações e respostas.

Unificando as duas bases

```
LAI = LAI %>% select(-DATA_RESPOSTA); #dim(LAI)
LAI1 = LAI1 %>% select(-DATA); #dim(LAI1)
DB = left_join(x = LAI, y = LAI1, by = "ID")
#View(head(DB))
```

Ver Anexo 01 c/ amostra dos dados da tabela que será utilizada para manipulação daqui pra frente.

Mineração de texto

Iniciamos as manipulações utilizando recursos da função `unnest_tokens()` do pacote `library(tidytext)` que nos permite trabalhar com textos em um formato `tidy` que coloca uma palavra por linha em uma única coluna, formando, assim, *termos/palavras* por linha. Utilizamos, também, ainda os recursos do pacote `library(dplyr)` para, posteriormente, agrupar esses termos por diretoria e calcular a frequência dos *termos*.

- Palavras

```
library(tidytext)
palavras <- DB %>%
  unnest_tokens(palavra, DESCR_PEDIDO) %>%
  count(palavra, sort = TRUE) %>%
  ungroup()
```

- Tabela 03 Palavras mais frequentes no conjunto de solicitações por diretoria

```
palavras[0:10,] %>%
  kable("latex", caption = "Principais palavras com stopwords",
```

```
booktabs = T, format.args = list(decimal.mark = ',', big.mark = "'")) %>%
kable_styling(latex_options = c("striped", "hold_position"))
```

Table 3: Principais palavras com stopwords

palavra	n
de	3'038
a	1'093
e	946
o	775
do	679
da	670
para	576
em	481
que	462
no	461

Verificamos que as 10 palavras mais frequentes em todos os pedidos realizados são palavras sem acréscimo contextual para alcançar o objetivo aqui proposto, pois não acrescentam nenhum sentido semântico, são essas: preposições (de, da, do, para), conjunção (e) e artigos(o,a).

Citar o que é preposição.

No passo seguinte iremos remover essas palavras, *stopwords*, e trabalhar apenas com palavras de sentido semântico relevante aos subjetivos solicitados às diretorias, acrescentando assim maior assertividade na classificação do nosso modelo, ainda a ser proposto no capítulo (indicar capítulo).

- Palavras por diretoria

```
library(tidytext)
diretoria_palavras <- DB %>%
  unnest_tokens(palavra, DESCRIPEDIDO) %>%
  count(DIRETORIA, palavra, sort = TRUE) %>%
  ungroup()
```

A tabela a seguir mostra a frequência das 10 palavras de maior ocorrência de todos os pedidos, agregados por diretoria.

- Tabela 04 Palavras mais frequentes no conjunto de solicitações por diretoria

```
diretoria_palavras[0:10,] %>%
  kable("latex", caption = "Palavras mais frequentes no conjunto de solicitações",
        booktabs = T, format.args = list(decimal.mark = ',', big.mark = "'")) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

- Total de palavras

```
total_palavras = diretoria_palavras %>%
  group_by(DIRETORIA) %>%
  summarize(total_palavras = sum(n))
```

- Tabela 05 Total de palavras por diretoria

```
total_palavras %>%
  kable("latex", caption = "Palavras mais frequentes no conjunto de solicitações",
        booktabs = T, format.args = list(decimal.mark = ',', big.mark = "'")) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 4: Palavras mais frequentes no conjunto de solicitações

DIRETORIA	palavra	n
DEA	de	1'127
DEE	de	979
DGC	de	736
DEE	a	364
DEA	a	350
DEA	e	329
DGC	a	304
DEE	e	273
DGC	e	266
DEA	o	262

Table 5: Palavras mais frequentes no conjunto de solicitações

DIRETORIA	total_palavras
DEA	14.079
DEE	12.907
DGC	10.355
DPG	1.434
OUTROS	1.022

É importante ressaltar aqui, a diferença extrema entre o número de palavras existente por diretoria, isso se dá devido ao número de pedidos realizados por diretoria já constatado anteriormente. Temos 210 solicitações registradas para a DEA, 197 para a DEE, 115 para a DGC e, apenas, 24 e 20 pedidos para a DPG e OUTROS, respectivamente.

Vamos, portanto, visualizar o número médio de palavras por pedido e diretoria. Para isso, vamos pegar o total de palavras por diretoria e dividir pelo total de pedidos por diretoria.

```
prop_palavras_pedido_dir = left_join(pedidos_diretoria,
                                     total_palavras, by = "DIRETORIA")
prop_palavras_pedido_dir$palavras_por_pedido = round(prop_palavras_pedido_dir$total_palavras / prop_pal
```

- Tabela 06 Número médio de palavras por pedido e diretoria

```
prop_palavras_pedido_dir %>%
  kable("latex", caption = "Palavras mais frequentes no conjunto de solicitações",
        booktabs = T, format.args = list(decimal.mark = ',', big.mark = ".")) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 6: Palavras mais frequentes no conjunto de solicitações

DIRETORIA	total_pedidos	total_palavras	palavras_por_pedido
DEA	210	14.079	67,04
DEE	197	12.907	65,52
DGC	115	10.355	90,04
DPG	24	1.434	59,75
OUTROS	20	1.022	51,10

- Junta informações

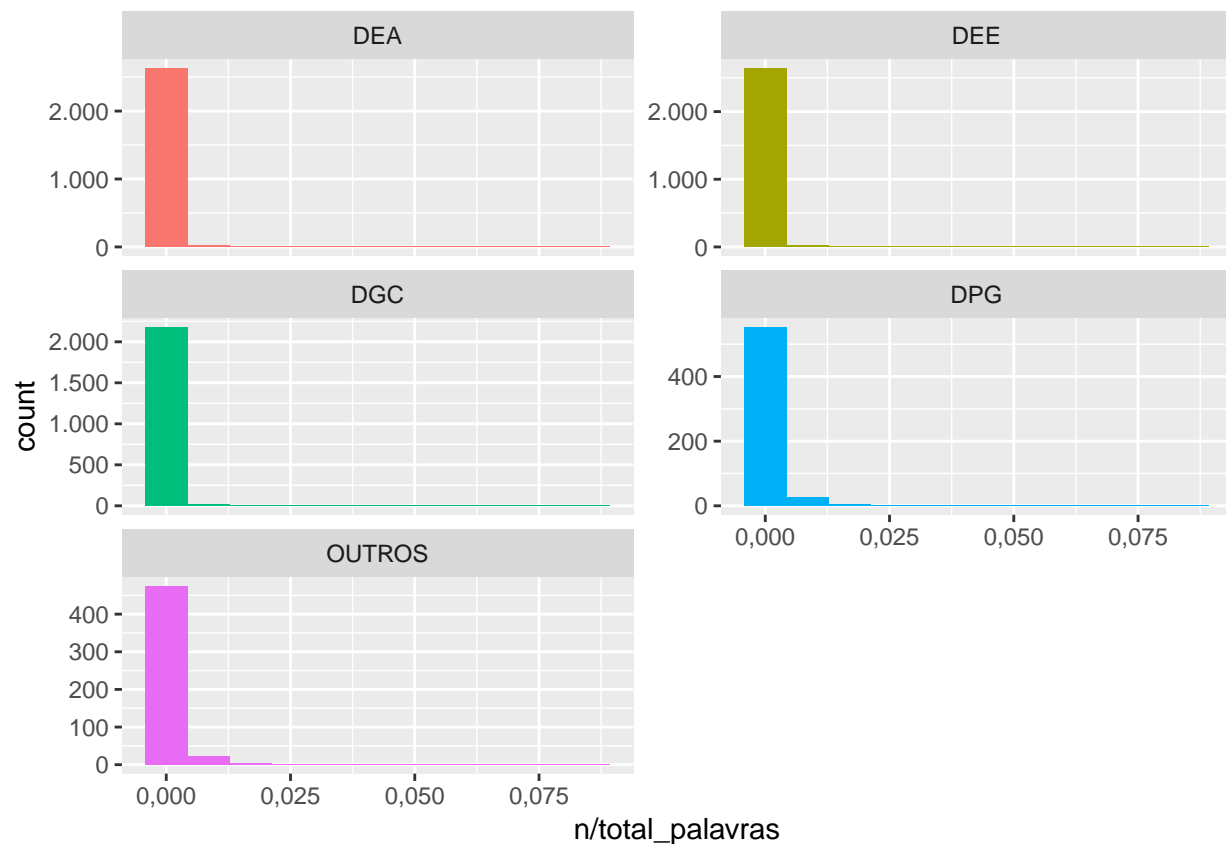
```
diretoria_palavras = left_join(diretoria_palavras, total_palavras, by = "DIRETORIA")
```

- Distribuição do nº de palavras usadas em solicitações por diretoria (histograma)

```
library(ggplot2)
gcomma <- function(x) format(x, big.mark = ".", decimal.mark = ",", scientific = FALSE)

ggplot(diretoria_palavras, aes(n/total_palavras, fill = DIRETORIA)) +
  geom_histogram(show.legend = FALSE, binwidth = .0085) + xlim(NA, .025) +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free_y") +
  scale_y_continuous(labels=gcomma) +
  scale_x_continuous(labels=gcomma)
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which
## will replace the existing scale.
```



- Palavras mais frequentes por diretoria

```
PROP_PALAVRA = diretoria_palavras %>%
  mutate(palavra = str_extract(palavra, "[a-z']+")) %>%
  count(DIRETORIA, palavra) %>%
  group_by(DIRETORIA) %>%
  mutate(proportion = n / sum(n)) %>%
  select(-n) %>%
  spread(DIRETORIA, proportion)
```

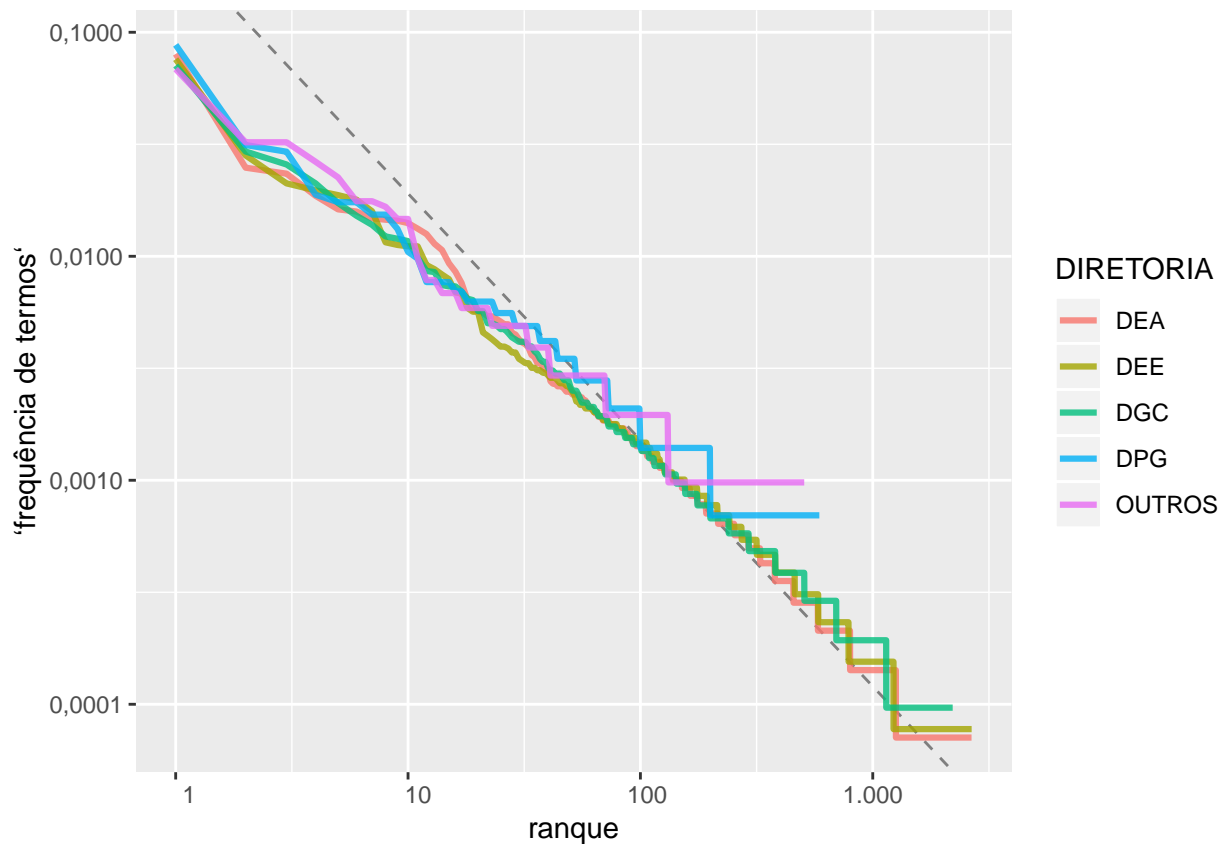
- Zipf's law

```

freq_by_rank <- diretoria_palavras %>%
  group_by(DIRETORIA) %>%
  mutate(ranque = row_number(),
         `frequência de termos` = n/total_palavras)

# Plot
freq_by_rank %>%
  ggplot(aes(ranque, `frequência de termos`, color = DIRETORIA)) +
  geom_abline(intercept = -0.62, slope = -1.1, color = "gray50", linetype = 2) +
  geom_line(size = 1.1, alpha = 0.8, show.legend = TRUE) +
  scale_x_log10(labels=gcomma) +
  scale_y_log10(labels=gcomma)

```



Frequência de palavras por diretoria

```

diretoria_palavras <- DB %>%
  unnest_tokens(palavra, DESCRIPEDIDO) %>%
  count(DIRETORIA, palavra, sort = TRUE) %>%
  ungroup()
#diretoria_palavras

plot_diretoria_palavras <- diretoria_palavras %>%
  bind_tf_idf(palavra, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(palavra = factor(palavra, levels = rev(unique(palavra)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",

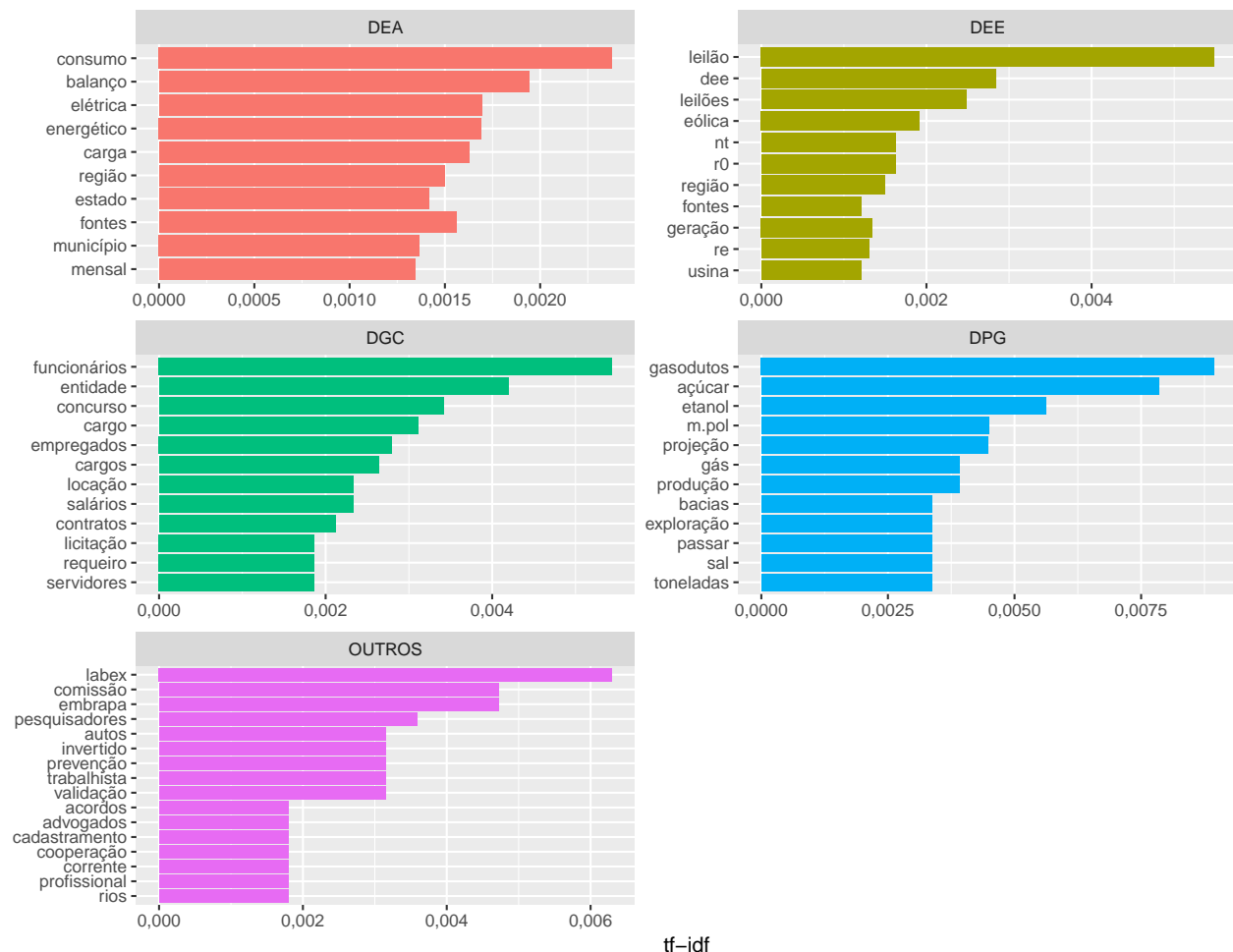
```

```

"DEE",
"DGC",
"DPG",
"OUTROS"))

#View(head(plot_diretoria_palavras))
#jpeg("02_freq_palavras_dir.jpeg")
plot_diretoria_palavras %>%
  group_by(DIRETORIA) %>%
  top_n(10, tf_idf) %>%
  ungroup() %>%
  mutate(palavra = reorder(palavra, tf_idf)) %>%
  ggplot(aes(palavra, tf_idf, fill = DIRETORIA)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
  coord_flip() +
  scale_y_continuous(labels=gcomma)

```



```
#dev.off()
```

Filtrando um pedaço de texto


```
DB %>%
  filter(str_detect(DESCRI_PEDIDO, "r0")) %>%
  select(DESCRI_PEDIDO) %>%
  head()
```

```
##
```

```
## 1
```

```
## 2
```

```
## 3 Solicitamos para nossa análise cópias dos relatórios nºs EPE-DEE-RE-147/2008-r0 que trata dos ESTU
```

```
## 4
```

```
## 5
```

```
## 6
```

Uma limpeza removendo palavras sem significado semântico (**stopwords**) pode auxiliar o algoritmo a retornar palavras ainda mais assertivas

Stemming

Podemos diminuir redundâncias por parte do algoritmo ensinando-o a compreender palavras que podem estar escritas de forma diferente mas que em significado semântico são semelhantes. Para isso, analisamos o radical de palavras com um mesmo prefixo mas com sufixos diferentes seja por quisistos como gênero ou plural.

Exemplos:

leilão \propto leilões estado \propto estados região \propto regiões

Falta implementar

Usando o pacote ptstem

```
library(ptstem)
ptstem(DB$DESCRI_PEDIDO[227])
```

```
## [1] "Sou funcionário da Simple Energy, empresa representante da UTE Manauara. Estou realizando um le
```

```
stemming1 = ptstem(DB$DESCRI_PEDIDO)
```

```
#ptstem(diretoria_palavras$palavra)
```

Frequência de palavras por diretoria

```
diretoria_palavras_stem1 <- DB %>%
  mutate(DESCRI_PEDIDO = stemming1) %>%
  unnest_tokens(palavra, DESCRI_PEDIDO) %>%
  count(palavra, sort = TRUE) %>%
  ungroup()
```

```
cat(paste0("Utilizando o algoritmo de stemming do pacote 'ptstem' o número de palavras chaves sem stemm
```

```
## Utilizando o algoritmo de stemming do pacote 'ptstem' o número de palavras chaves sem stemming reduz
```

Usando o pacote rslp

```
library(rslp)
rslp(DB$DESCRI_PEDIDO[227])
```

```
## [1] "Sou funcionario da Simple Energy, empresa representante da UTE Manauara. Estou realizando um le
```

```
stemming2 = rslp(DB$DESCRI_PEDIDO)
```

Frequência de palavras por diretoria

```
diretoria_palavras_stem2 <- DB %>%  
  mutate(DESCRI_PEDIDO = stemming2) %>%  
  unnest_tokens(palavra, DESCRI_PEDIDO) %>%  
  count(palavra, sort = TRUE) %>%  
  ungroup()
```

```
cat(paste0("Utilizando o algoritmo de stemming do pacote 'rslp' o número de palavras chaves sem stemming reduziu
```

Utilizando o algoritmo de stemming do pacote 'rslp' o número de palavras chaves sem stemming reduziu

Uma redução considerável no número de termos ocorreu ao usar o algoritmo `ptstem`, cerca de 55% de redução de termos versus 38% utilizando o algoritmo `rslp`.

Vale ressaltar, também, o tempo de processamento que ambos os algoritmos requerem.

```
temp_stem1 = proc.time()  
teste0 = ptstem(DB$DESCRI_PEDIDO)  
tempo_stem1 = proc.time() - temp_stem1  
remove(teste0)
```

Warning in remove(teste0): objeto 'teste0' não encontrado

```
temp_stem2 = proc.time()  
teste00 = rslp(DB$DESCRI_PEDIDO)  
tempo_stem2 = proc.time() - temp_stem2  
remove(teste00)
```

O tempo decorrido para processamento do algoritmo do `ptstem` foi de aproximadamente 12,4 segundos versus 1,0 segundo decorrido para o processamento do algoritmo do `rslp`. Logo, o `rslp` é quase 13 vezes mais eficiente em termos de tempo de processamento. Além disso, o `rslp` remove acentuações e caracteres como “ç”. Isso irá nos ajudar mais a frente quando utilizarmos os principais termos como variáveis binárias e predictoras do modelo de classificação.

Entretanto, o algoritmo mais lento, `ptstem`, foi mais interessante em termos de redução do número de termos únicos, cerca de 22% menos termos em relação ao outro algoritmo. Além disso, por se tratar de uma base de dados relativamente pequena, 565 pedidos, e o alto poder de processamento da máquina utilizada para esse estudo.

Optamos, portanto, por utilizar ambos algoritmos. Vamos, primeiro, aplicar o removedor de sufixos da língua portuguesa `rslp` seguido do `ptstem`.

Frequência de palavras por diretoria

```
stemming3 = ptstem(stemming2)  
diretoria_palavras_stem3 <- DB %>%  
  mutate(DESCRI_PEDIDO = stemming3) %>%  
  unnest_tokens(palavra, DESCRI_PEDIDO) %>%  
  count(DIRETORIA, palavra, sort = TRUE) %>%  
  ungroup()  
#diretoria_palavras
```

Vamos, agora, plotar as quinze palavras mais relevantes de acordo com a estatística `tf_idf`, por diretoria

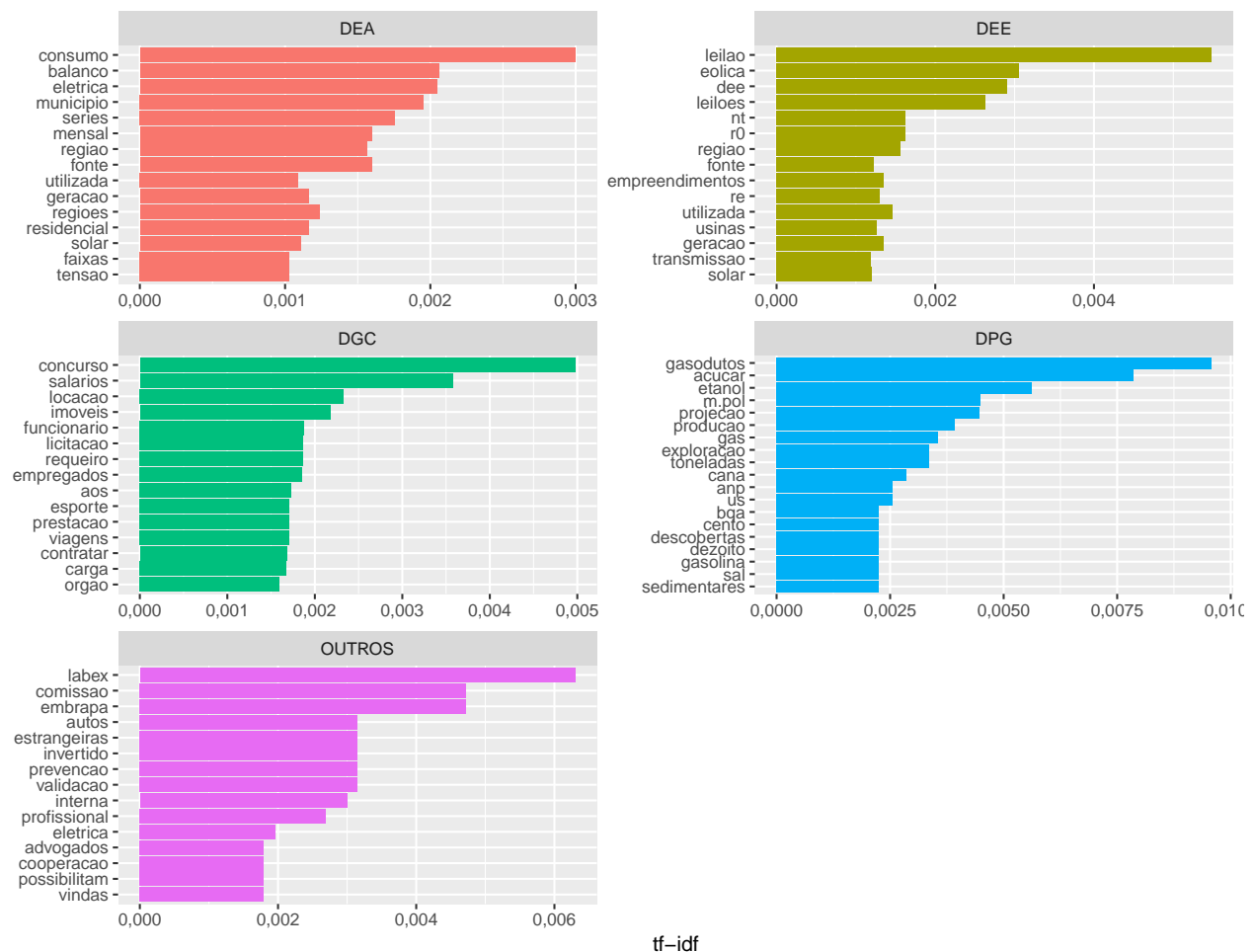
```
plot_diretoria_palavras_stem <- diretoria_palavras_stem3 %>%  
  bind_tf_idf(palavra, DIRETORIA, n) %>%  
  arrange(desc(tf_idf)) %>%  
  mutate(palavra = factor(palavra, levels = rev(unique(palavra)))) %>%
```

```

mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",
                                                "DEE",
                                                "DGC",
                                                "DPG",
                                                "OUTROS"))))

#View(head(plot_diretoria_palavras))
#jpeg("02_freq_palavras_dir.jpeg")
plot_diretoria_palavras_stem %>%
group_by(DIRETORIA) %>%
top_n(15, tf_idf) %>%
ungroup() %>%
mutate(palavra = reorder(palavra, tf_idf)) %>%
ggplot(aes(palavra, tf_idf, fill = DIRETORIA)) +
geom_col(show.legend = FALSE) +
labs(x = NULL, y = "tf-idf") +
facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
coord_flip() +
scale_y_continuous(labels=gcomma)

```



```
#dev.off()
```

Stopwords

Com o arquivo de stopwords previamente inserido vamos, primeiramente, transforma-lo em um data_frame a fim de futuramente utilizá-lo para extrair do texto palavras em comum.

Freq. de palavras sem stopwords por diretoria

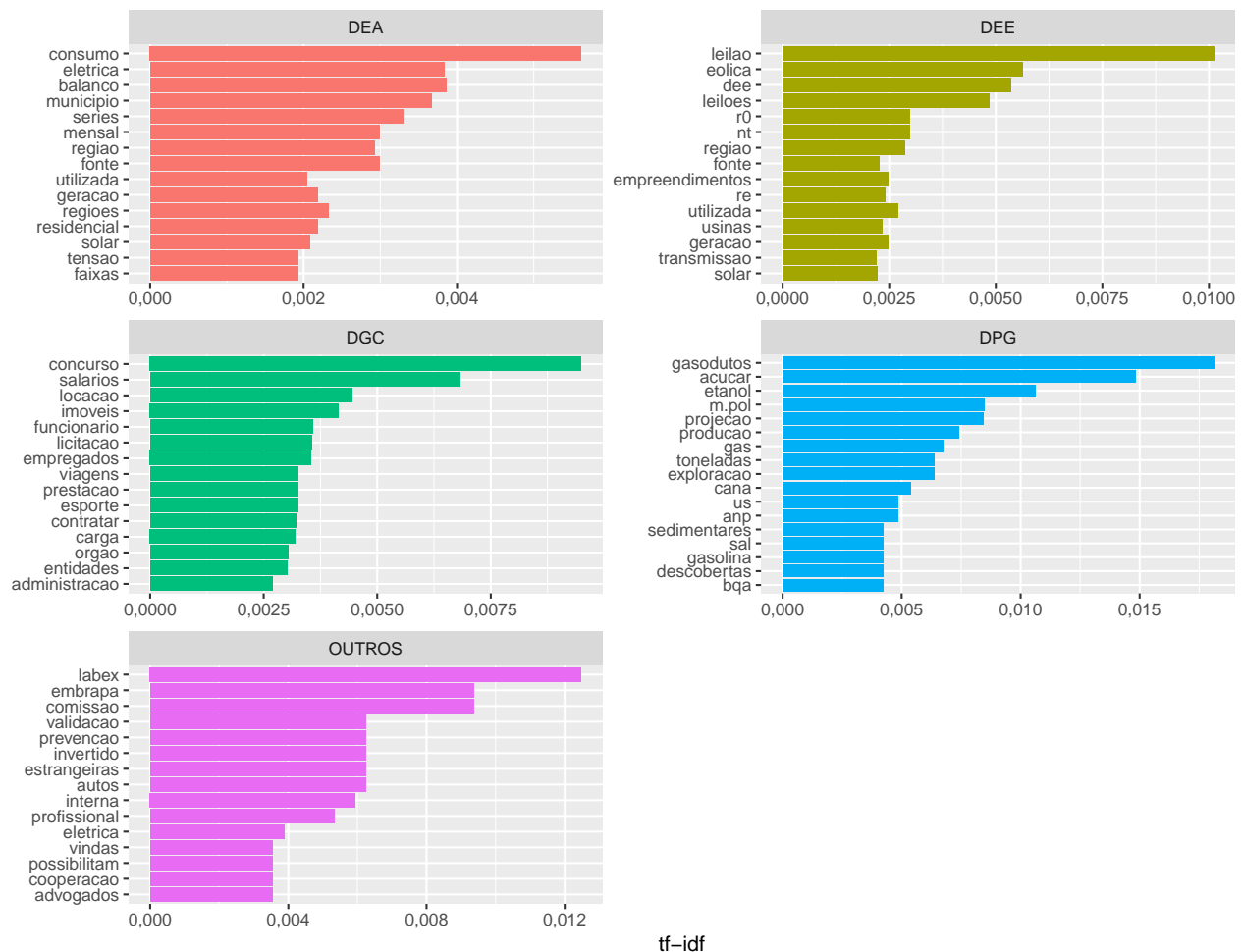
```
mystopwords <- data_frame(palavra = stopwords_pt)

## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.

diretoria_palavras_noSTOP <- anti_join(diretoria_palavras_stem3, mystopwords, by = "palavra")
#View(head(diretoria_palavras_noSTOP))

#diretoria_palavras_noSTOP_noSTOP
plot_diretoria_palavras_noSTOP <- diretorio_palavras_noSTOP %>%
  bind_tf_idf(palavra, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(palavra, levels = rev(unique(palavra)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",
                                                  "DEE",
                                                  "DGC",
                                                  "DPG",
                                                  "OUTROS")))

#plot_diretoria_palavras_noSTOP
#windows.options(width=10, height=10)
#jpeg("03_freq_palavras_dir_nostop.jpeg")
plot_diretoria_palavras_noSTOP %>%
  group_by(DIRETORIA) %>%
  top_n(15, tf_idf) %>%
  ungroup() %>%
  mutate(palavra = reorder(palavra, tf_idf)) %>%
  ggplot(aes(palavra, tf_idf, fill = DIRETORIA)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
  coord_flip() +
  scale_y_continuous(labels=gcomma)
```



```
#dev.off()
```

Gráficos de comparação de frequência de palavras por diretorias (2 a 2)

COM STOPWORDS

É importante ressaltar que os gráficos a seguir mostram, apenas, a comparação de frequência de palavras existentes em ambas diretorias. Ou seja, palavras existentes em apenas uma diretoria serão desconsideradas para a geração destes.

```
PROP_PALAVRA = diretoria_palavras_noSTOP %>%
  #mutate(palavra = str_extract(palavra, "[a-z']+")) %>%
  mutate(proportion = n / sum(n)) %>%
  select(-n) %>%
  spread(DIRETORIA, proportion)
```

- DEE X DEA

```
freq00 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`DEA`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq00, aes(x = proportion, y = `DEE`,
```

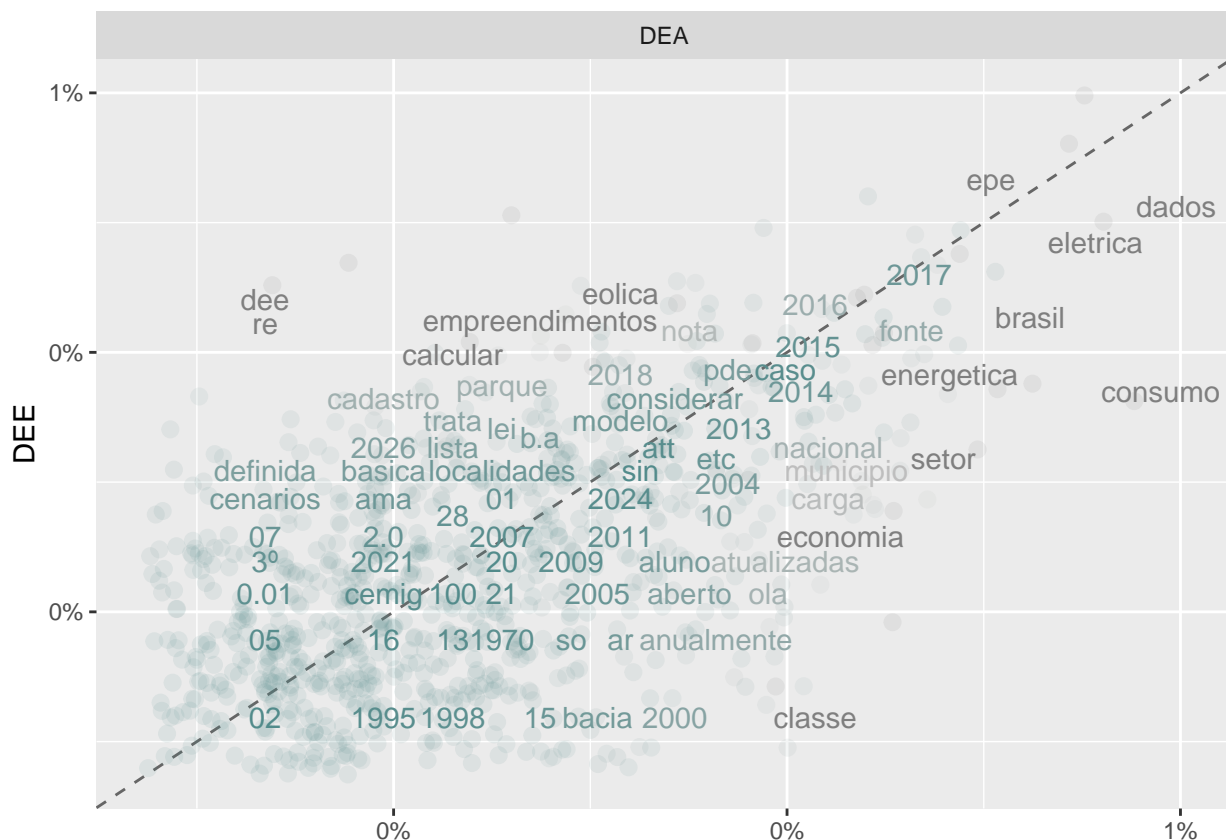
```

        color = abs(`DEE` - proportion))) +
geom_abline(color = "gray40", lty = 2) +
geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
scale_color_gradient(limits = c(0, 0.001),
                      low = "darkslategray4", high = "gray75") +
facet_wrap(~DIRETORIA, ncol = 1) +
theme(legend.position="none") +
labs(y = "DEE", x = NULL)

```

```
## Warning: Removed 2825 rows containing missing values (geom_point).
```

```
## Warning: Removed 2823 rows containing missing values (geom_text).
```



```

cor.test(data = freq00[freq00$DIRETORIA == "DEA",],
         ~ proportion + `DEE`)

```

```

##
## Pearson's product-moment correlation
##
## data:  proportion and DEE
## t = 30.221, df = 890, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6776757 0.7426100
## sample estimates:

```

```
##      cor
## 0.7116595
```

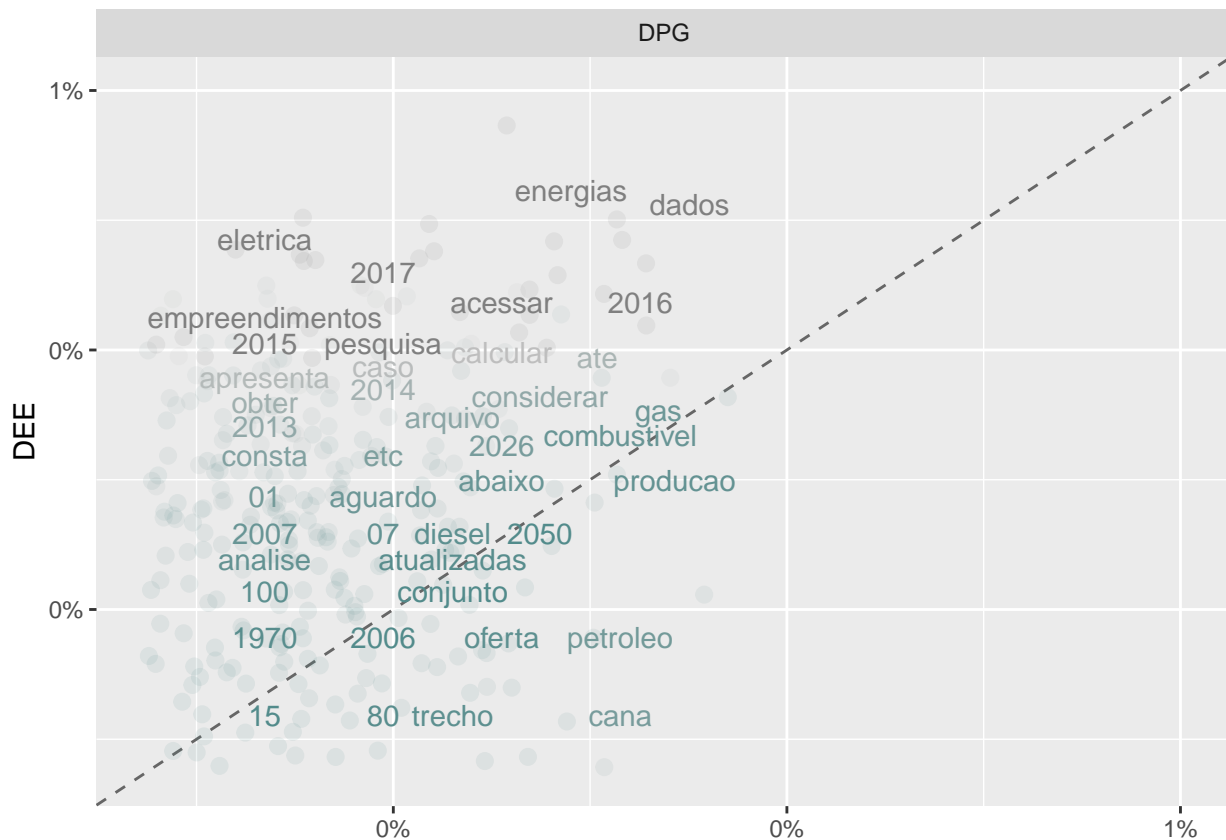
- DEE X DPG

```
freq01 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`DPG`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq01, aes(x = proportion, y = `DEE`,
  color = abs(`DEE` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_color_gradient(limits = c(0, 0.001),
    low = "darkslategray4", high = "gray75") +
  facet_wrap(~DIRETORIA, ncol = 1) +
  theme(legend.position="none") +
  labs(y = "DEE", x = NULL)
```

```
## Warning: Removed 3438 rows containing missing values (geom_point).
```

```
## Warning: Removed 3438 rows containing missing values (geom_text).
```



```
cor.test(data = freq01[freq01$DIRETORIA == "DPG",],
  ~ proportion + `DEE`)
```

```
##
## Pearson's product-moment correlation
##
## data: proportion and DEE
## t = 7.7037, df = 275, p-value = 2.406e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## 0.3193085 0.5136596
## sample estimates:
## cor
## 0.4213092
```

Warning messages:

```
1: Removed 4235 rows containing missing values (geom_point).
2: Removed 4236 rows containing missing values (geom_text).
```

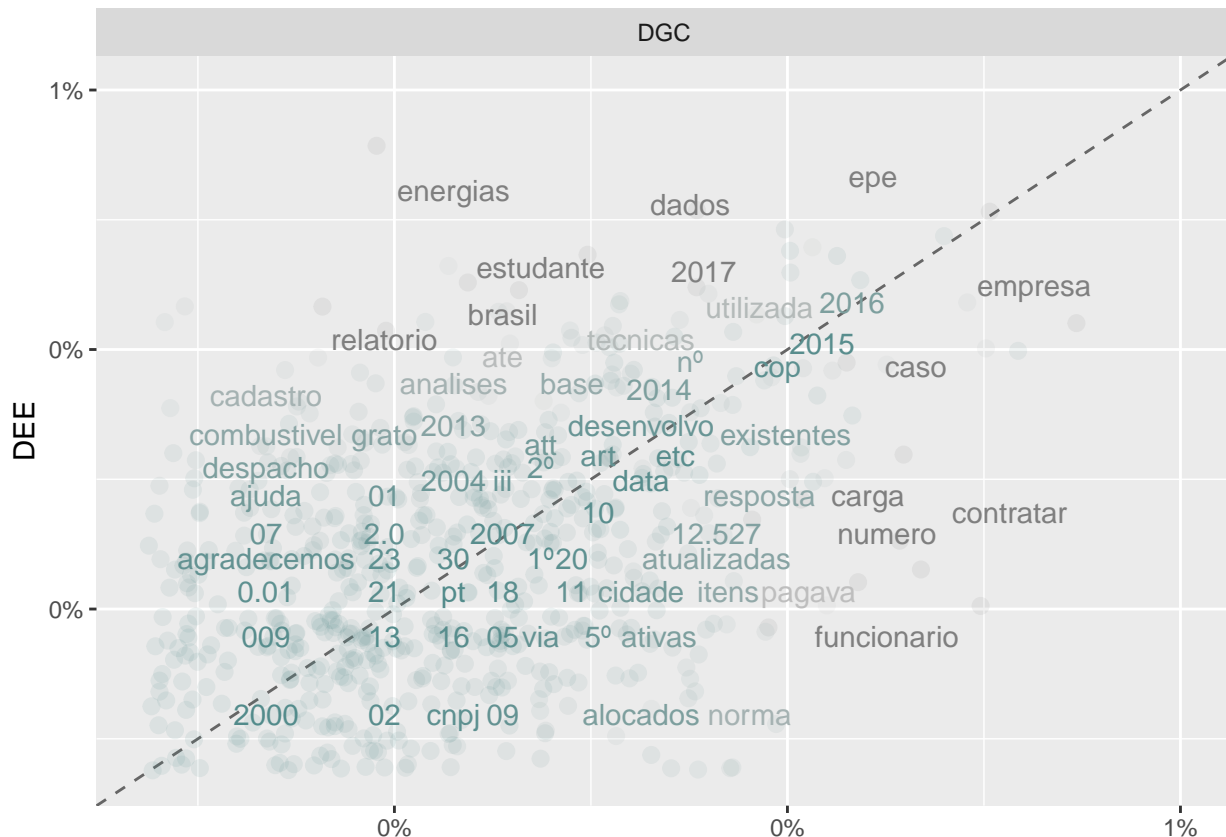
- DEE X DGC

```
freq02 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`DGC`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq02, aes(x = proportion, y = `DEE`,
  color = abs(`DEE` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0, 0.001),
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0, 0.001),
  scale_color_gradient(limits = c(0, 0.001),
    low = "darkslategray4", high = "gray75") +
  facet_wrap(~DIRETORIA, ncol = 1) +
  theme(legend.position="none") +
  labs(y = "DEE", x = NULL)
```

```
## Warning: Removed 3075 rows containing missing values (geom_point).
```

```
## Warning: Removed 3075 rows containing missing values (geom_text).
```

```
cor.test(data = freq02[freq02$DIRETORIA == "DGC",],
~ proportion + `DEE`)
```

```
##
## Pearson's product-moment correlation
##
## data: proportion and DEE
## t = 10.85, df = 638, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## 0.3271860 0.4581655
## sample estimates:
## cor
## 0.394679
```

Warning messages:

```
1: Removed 3794 rows containing missing values (geom_point).
2: Removed 3795 rows containing missing values (geom_text).
```

- DEE X OUTROS

```
freq03 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`OUTROS`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq03, aes(x = proportion, y = `DEE`,
  color = abs(`DEE` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
```


Warning messages:

- 1: Removed 4273 rows containing missing values (geom_point).
- 2: Removed 4274 rows containing missing values (geom_text).

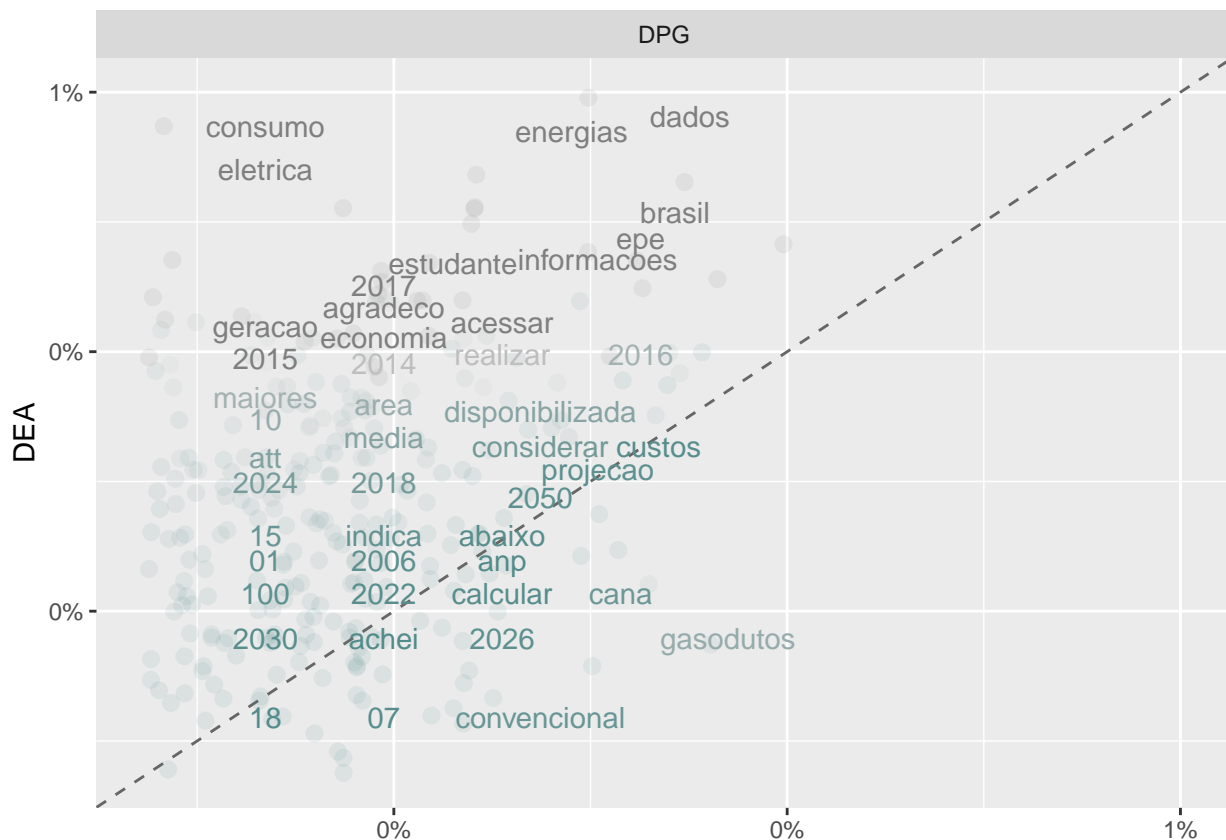
- DEA X DPG

```
freq04 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`DPG`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq04, aes(x = proportion, y = `DEA`,
  color = abs(`DEA` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_color_gradient(limits = c(0, 0.001),
    low = "darkslategray4", high = "gray75") +
  facet_wrap(~DIRETORIA, ncol = 1) +
  theme(legend.position="none") +
  labs(y = "DEA", x = NULL)
```

Warning: Removed 3438 rows containing missing values (geom_point).

Warning: Removed 3437 rows containing missing values (geom_text).



Warning messages:

```
1: Removed 4221 rows containing missing values (geom_point).  
2: Removed 4222 rows containing missing values (geom_text).
```

```
cor.test(data = freq04[freq04$DIRETORIA == "DPG",],  
         ~ proportion + `DEA`)
```

```
##  
## Pearson's product-moment correlation  
##  
## data: proportion and DEA  
## t = 6.1072, df = 276, p-value = 3.432e-09  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.2370151 0.4446322  
## sample estimates:  
## cor  
## 0.3450374
```

- DEA X DGC

```
freq05 <- PROP_PALAVRA %>%
```

```
  gather(DIRETORIA, proportion, c(`DGC`))
```

```
library(scales)
```

```
# expect a warning about rows with missing values being removed
```

```
ggplot(freq05, aes(x = proportion, y = `DEA`,
```

```
                  color = abs(`DEA` - proportion))) +
```

```
  geom_abline(color = "gray40", lty = 2) +
```

```
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
```

```
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
```

```
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c
```

```
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c
```

```
  scale_color_gradient(limits = c(0, 0.001),
```

```
                       low = "darkslategray4", high = "gray75") +
```

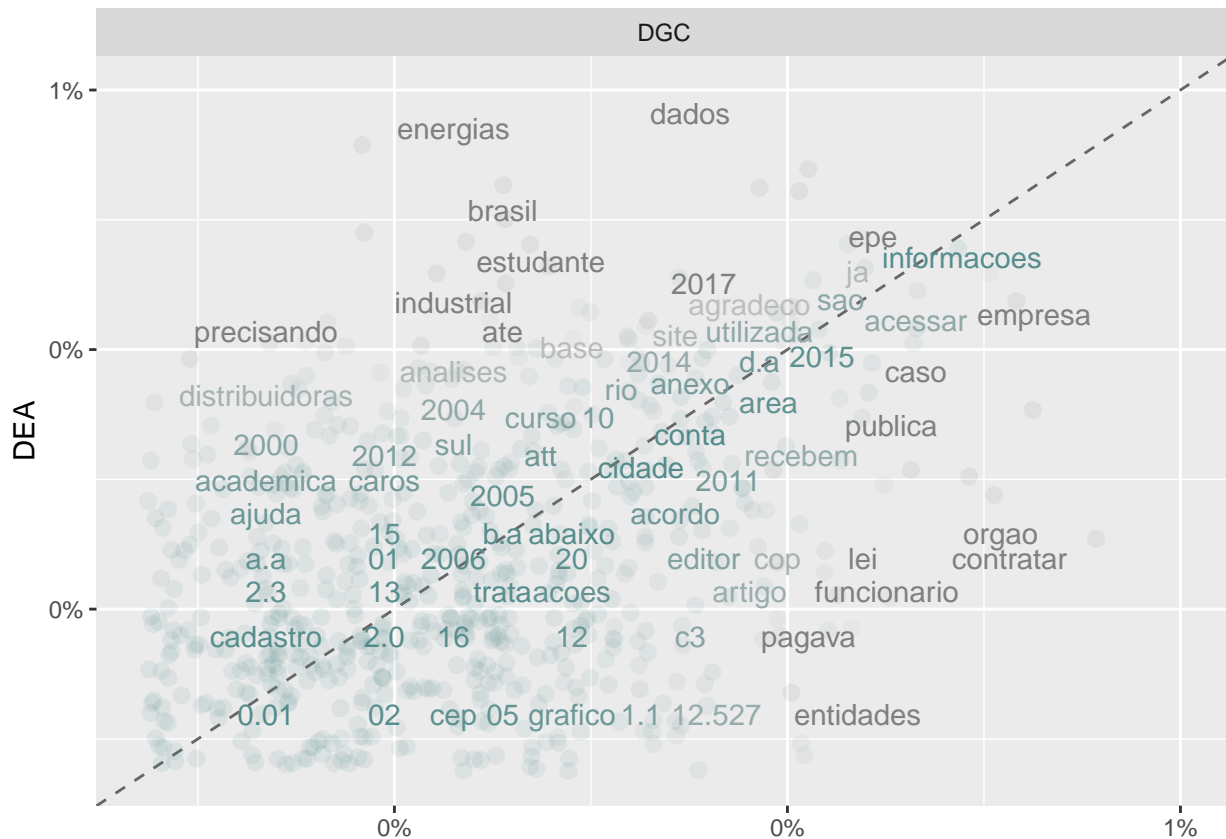
```
  facet_wrap(~DIRETORIA, ncol = 1) +
```

```
  theme(legend.position="none") +
```

```
  labs(y = "DEA", x = NULL)
```

```
## Warning: Removed 3087 rows containing missing values (geom_point).
```

```
## Warning: Removed 3086 rows containing missing values (geom_text).
```



Warning messages:

- 1: Removed 3812 rows containing missing values (geom_point).
- 2: Removed 3813 rows containing missing values (geom_text).

```
cor.test(data = freq05[freq05$DIRETORIA == "DGC",],
         ~ proportion + `DEA`)
```

```
##
## Pearson's product-moment correlation
##
## data: proportion and DEA
## t = 7.5906, df = 627, p-value = 1.158e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## 0.2168448 0.3601121
## sample estimates:
## cor
## 0.290103
```

• DEA X OUTROS

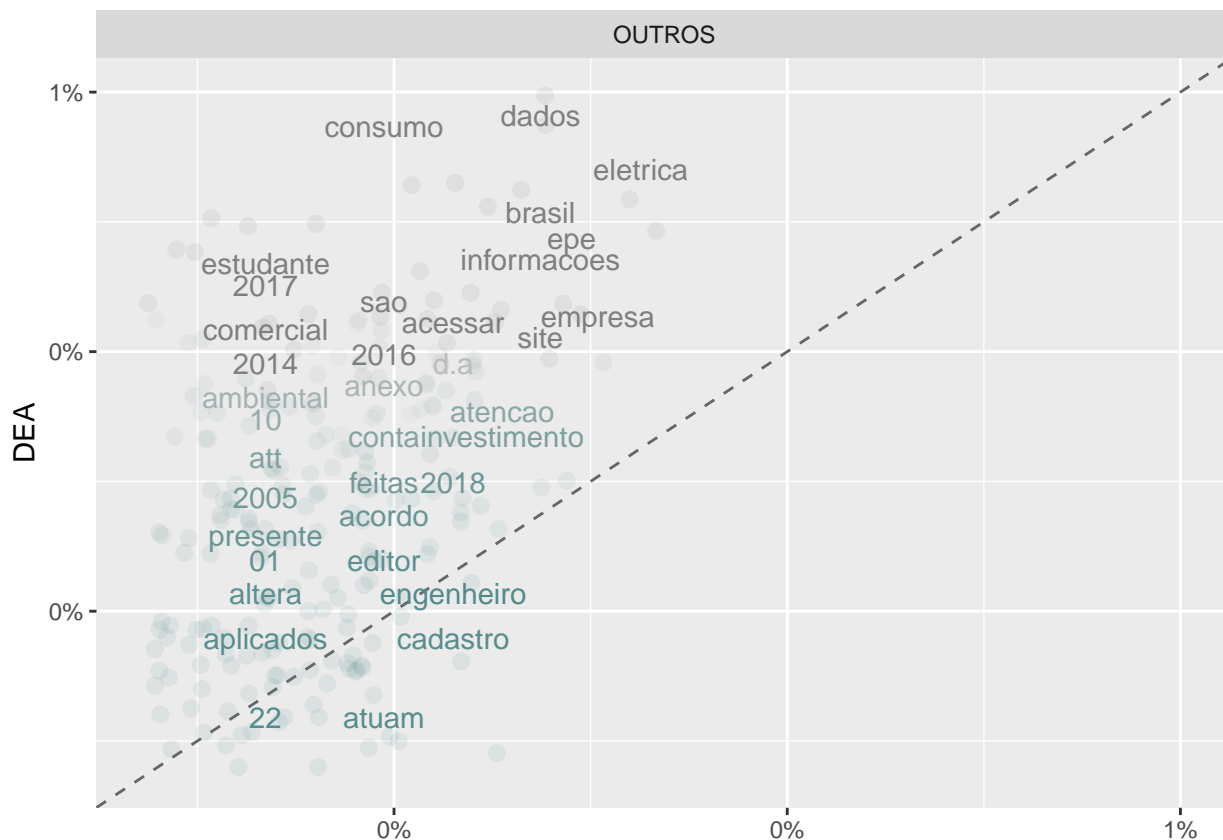
```
freq06 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`OUTROS`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq06, aes(x = proportion, y = `DEA`,
                  color = abs(`DEA` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
```

```
geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
scale_color_gradient(limits = c(0, 0.001),
  low = "darkslategray4", high = "gray75") +
facet_wrap(~DIRETORIA, ncol = 1) +
theme(legend.position="none") +
labs(y = "DEA", x = NULL)
```

```
## Warning: Removed 3499 rows containing missing values (geom_point).
```

```
## Warning: Removed 3498 rows containing missing values (geom_text).
```



```
Warning messages:
```

```
1: Removed 4303 rows containing missing values (geom_point).
```

```
2: Removed 4304 rows containing missing values (geom_text).
```

```
cor.test(data = freq06[freq06$DIRETORIA == "OUTROS",],
  ~ proportion + `DEA`)
```

```
##
```

```
## Pearson's product-moment correlation
```

```
##
```

```
## data: proportion and DEA
```

```
## t = 10.187, df = 215, p-value < 2.2e-16
```

```
## alternative hypothesis: true correlation is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## 0.4733414 0.6540412
## sample estimates:
##      cor
## 0.5705569
```

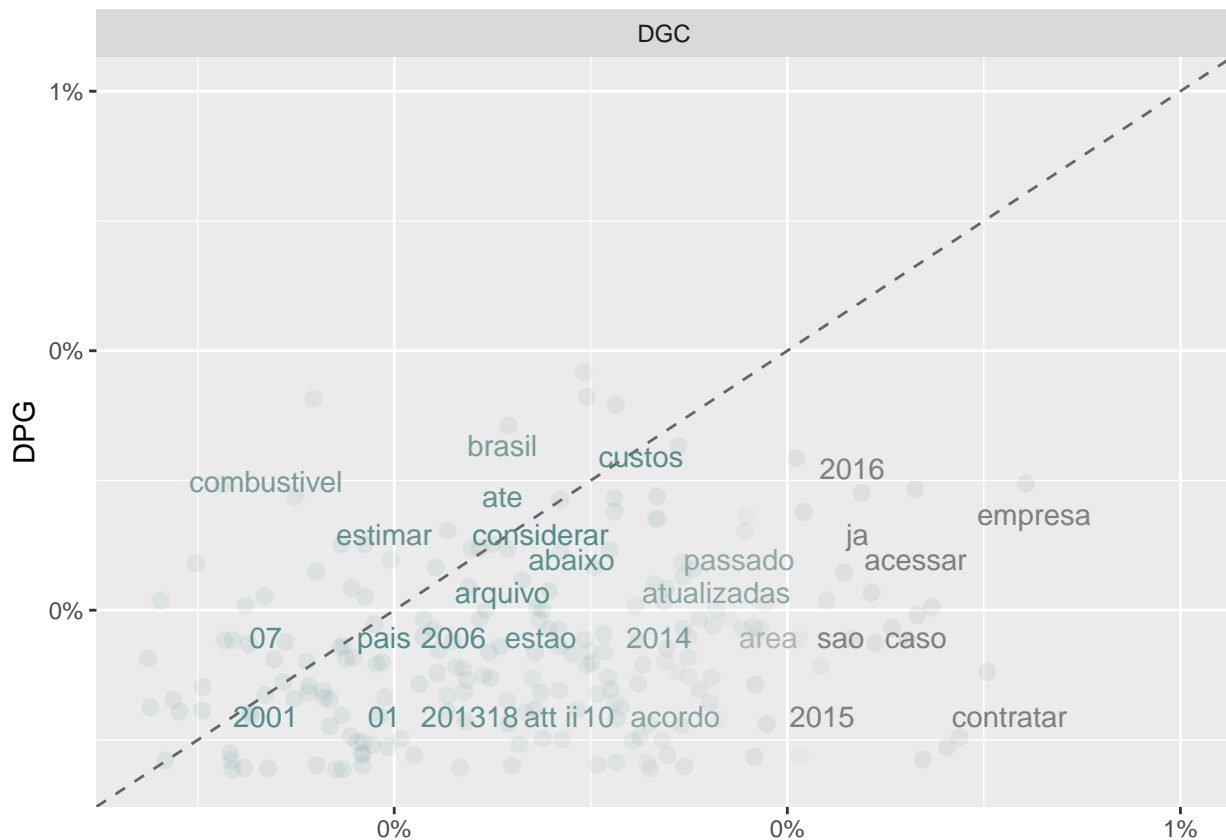
- DPG X DGC

```
freq07 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`DGC`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq07, aes(x = proportion, y = `DPG`,
  color = abs(`DPG` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.0001, 0.01)) +
  scale_color_gradient(limits = c(0, 0.001),
    low = "darkslategray4", high = "gray75") +
  facet_wrap(~DIRETORIA, ncol = 1) +
  theme(legend.position="none") +
  labs(y = "DPG", x = NULL)
```

```
## Warning: Removed 3507 rows containing missing values (geom_point).
```

```
## Warning: Removed 3507 rows containing missing values (geom_text).
```



Warning messages:

```
1: Removed 4296 rows containing missing values (geom_point).  
2: Removed 4297 rows containing missing values (geom_text).
```

```
cor.test(data = freq07[freq07$DIRETORIA == "DGC",],  
         ~ proportion + `DPG`)
```

```
##  
## Pearson's product-moment correlation  
##  
## data: proportion and DPG  
## t = 3.4202, df = 206, p-value = 0.0007543  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.09888554 0.35660374  
## sample estimates:  
## cor  
## 0.2318083
```

- DPG X OUTROS

```
freq08 <- PROP_PALAVRA %>%  
  gather(DIRETORIA, proportion, c(`OUTROS`))
```

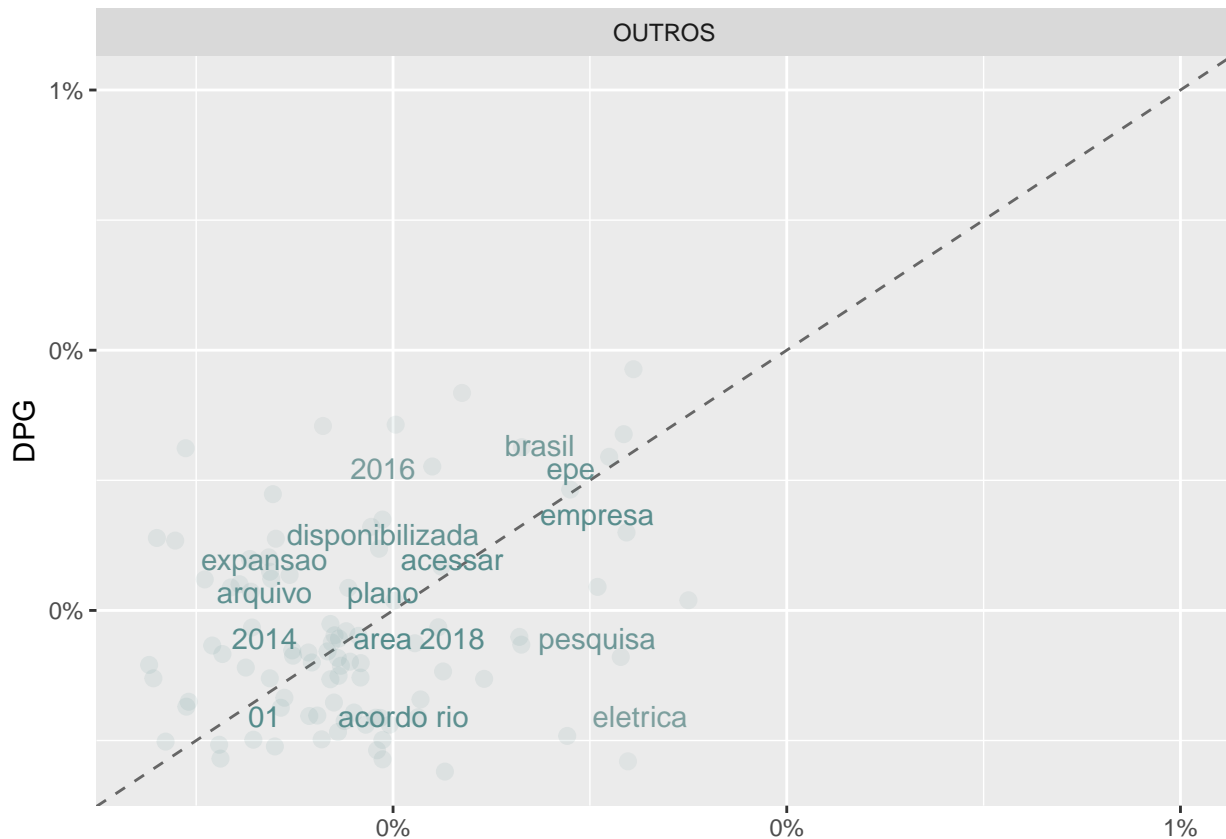
```
library(scales)
```

```
# expect a warning about rows with missing values being removed
```

```
ggplot(freq08, aes(x = proportion, y = `DPG`,  
                  color = abs(`DPG` - proportion))) +  
  geom_abline(color = "gray40", lty = 2) +  
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +  
  geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +  
  scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.001, 0.001)) +  
  scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.001, 0.001)) +  
  scale_color_gradient(limits = c(0, 0.001),  
                      low = "darkslategray4", high = "gray75") +  
  facet_wrap(~DIRETORIA, ncol = 1) +  
  theme(legend.position="none") +  
  labs(y = "DPG", x = NULL)
```

```
## Warning: Removed 3623 rows containing missing values (geom_point).
```

```
## Warning: Removed 3623 rows containing missing values (geom_text).
```

Warning messages:

```
1: Removed 4450 rows containing missing values (geom_point).
2: Removed 4451 rows containing missing values (geom_text).
```

```
cor.test(data = freq08[freq08$DIRETORIA == "OUTROS",],
         ~ proportion + `DPG`)
```

```
##
## Pearson's product-moment correlation
##
## data: proportion and DPG
## t = 5.0342, df = 90, p-value = 2.447e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## 0.2919524 0.6145567
## sample estimates:
## cor
## 0.4687405
```

• DGC X OUTROS

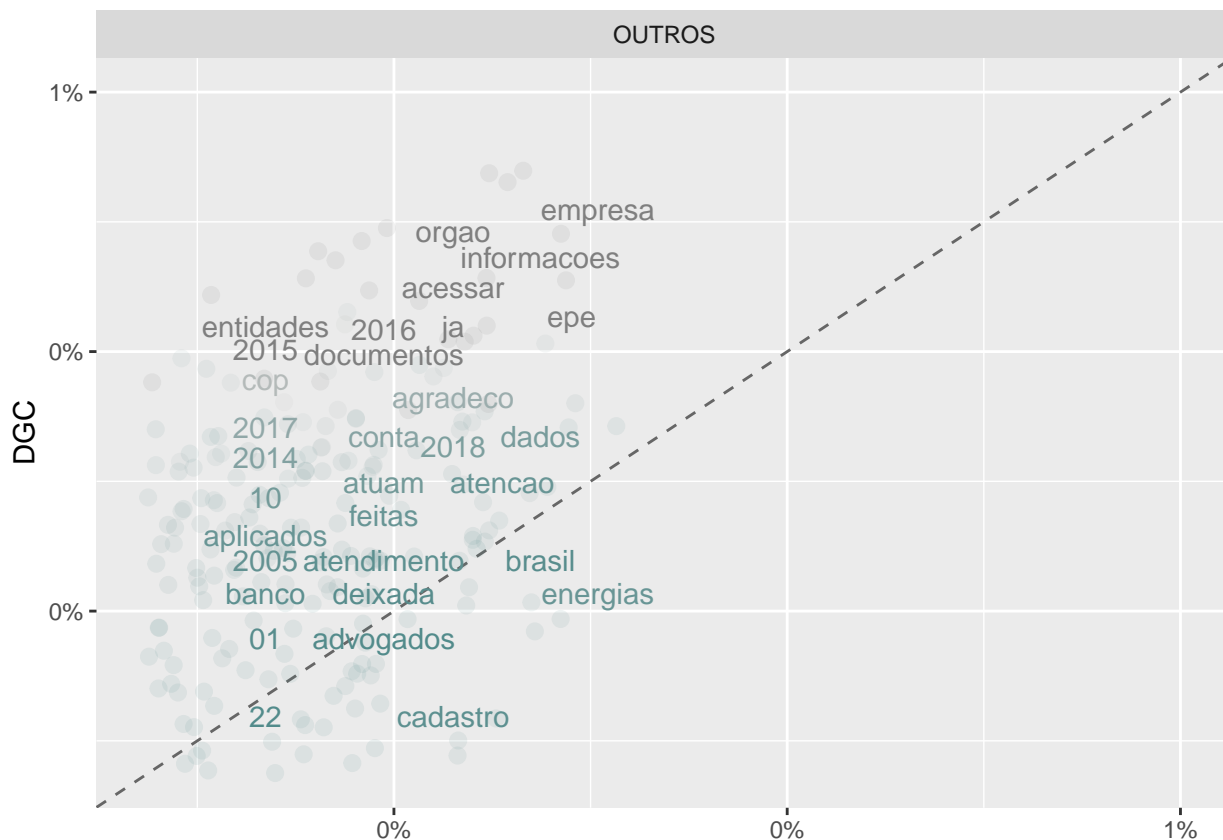
```
freq09 <- PROP_PALAVRA %>%
  gather(DIRETORIA, proportion, c(`OUTROS`))

library(scales)
# expect a warning about rows with missing values being removed
ggplot(freq08, aes(x = proportion, y = `DGC`,
                  color = abs(`DGC` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
```

```
geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
geom_text(aes(label = palavra), check_overlap = TRUE, vjust = 1.5) +
scale_x_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.001, 0.01)) +
scale_y_log10(labels = percent_format(big.mark = ".", decimal.mark = ",", accuracy = 1), limits = c(0.001, 0.01)) +
scale_color_gradient(limits = c(0, 0.001),
  low = "darkslategray4", high = "gray75") +
facet_wrap(~DIRETORIA, ncol = 1) +
theme(legend.position="none") +
labs(y = "DGC", x = NULL)
```

```
## Warning: Removed 3510 rows containing missing values (geom_point).
```

```
## Warning: Removed 3510 rows containing missing values (geom_text).
```



```
Warning messages:
```

```
1: Removed 4302 rows containing missing values (geom_point).
```

```
2: Removed 4303 rows containing missing values (geom_text).
```

```
cor.test(data = freq09[freq09$DIRETORIA == "OUTROS",],
  ~ proportion + `DGC`)
```

```
##
```

```
## Pearson's product-moment correlation
```

```
##
```

```
## data: proportion and DGC
```

```
## t = 8.4142, df = 203, p-value = 7.03e-15
```

```
## alternative hypothesis: true correlation is not equal to 0
```

```
## 95 percent confidence interval:
```

```
## 0.3992974 0.6034898
## sample estimates:
##      cor
## 0.508508
```

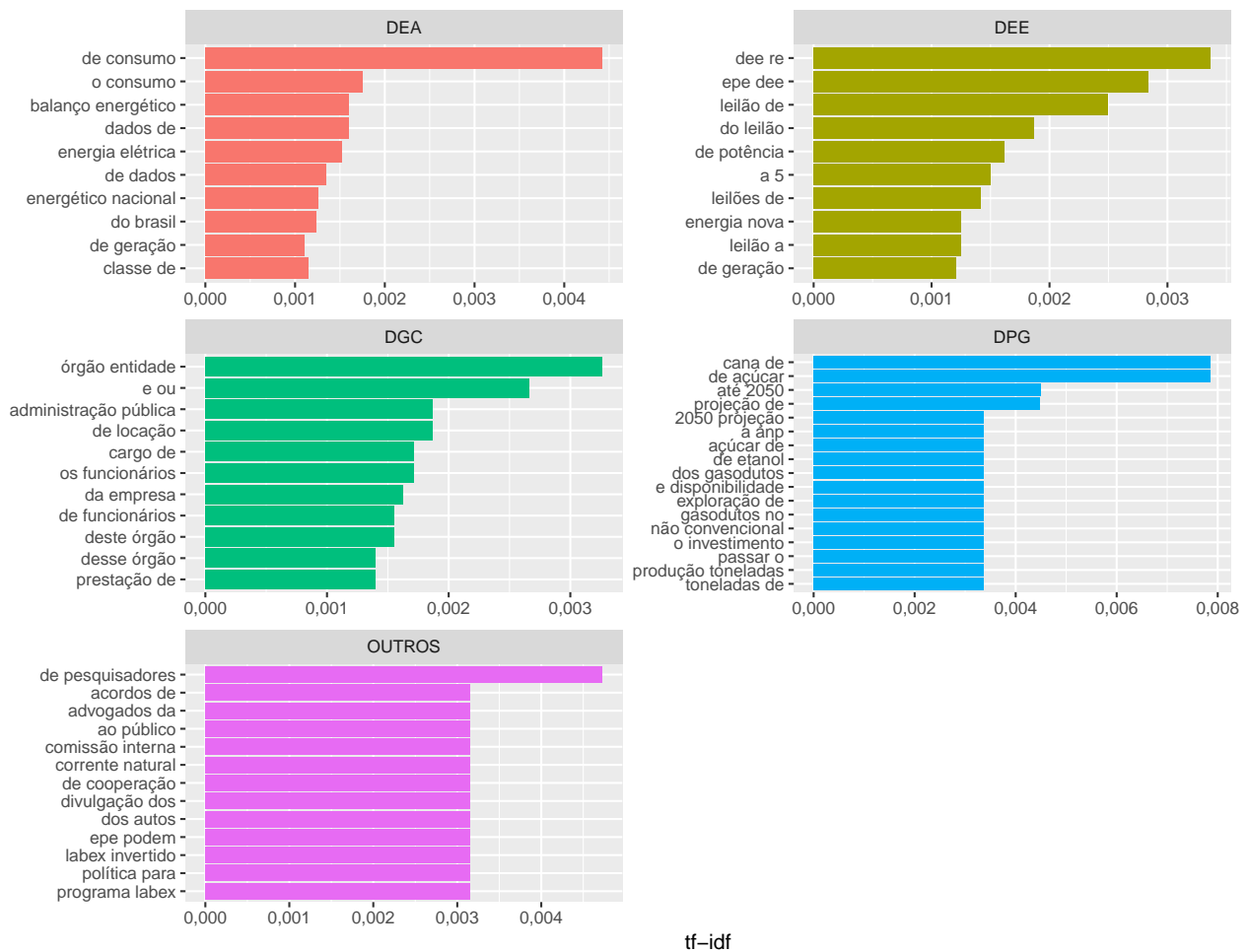
Usando bigram para n=2 palavras por token

Frequência de palavras por diretoria

```
diretoria_palavras_bigram <- DB %>%
  select(DESCRI_PEDIDO,DIRETORIA) %>%
  unnest_tokens(BIGRAM, DESCRI_PEDIDO, token = "ngrams", n = 2) %>%
  count(DIRETORIA, BIGRAM, sort = TRUE) %>%
  ungroup()
#diretoria_palavras_bigram

plot_diretoria_palavras_bigram <- diretoria_palavras_bigram %>%
  bind_tf_idf(BIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(BIGRAM = factor(BIGRAM, levels = rev(unique(BIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",
                                                    "DEE",
                                                    "DGC",
                                                    "DPG",
                                                    "OUTROS"))))

#View(head(plot_diretoria_palavras_bigram))
#jpeg("02_freq_palavras_dir.jpeg")
plot_diretoria_palavras_bigram %>%
  group_by(DIRETORIA) %>%
  top_n(10, tf_idf) %>%
  ungroup() %>%
  mutate(BIGRAM = reorder(BIGRAM, tf_idf)) %>%
  ggplot(aes(BIGRAM, tf_idf, fill = DIRETORIA)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
  coord_flip() +
  scale_y_continuous(labels=gcomma)
```



```
#dev.off()
```

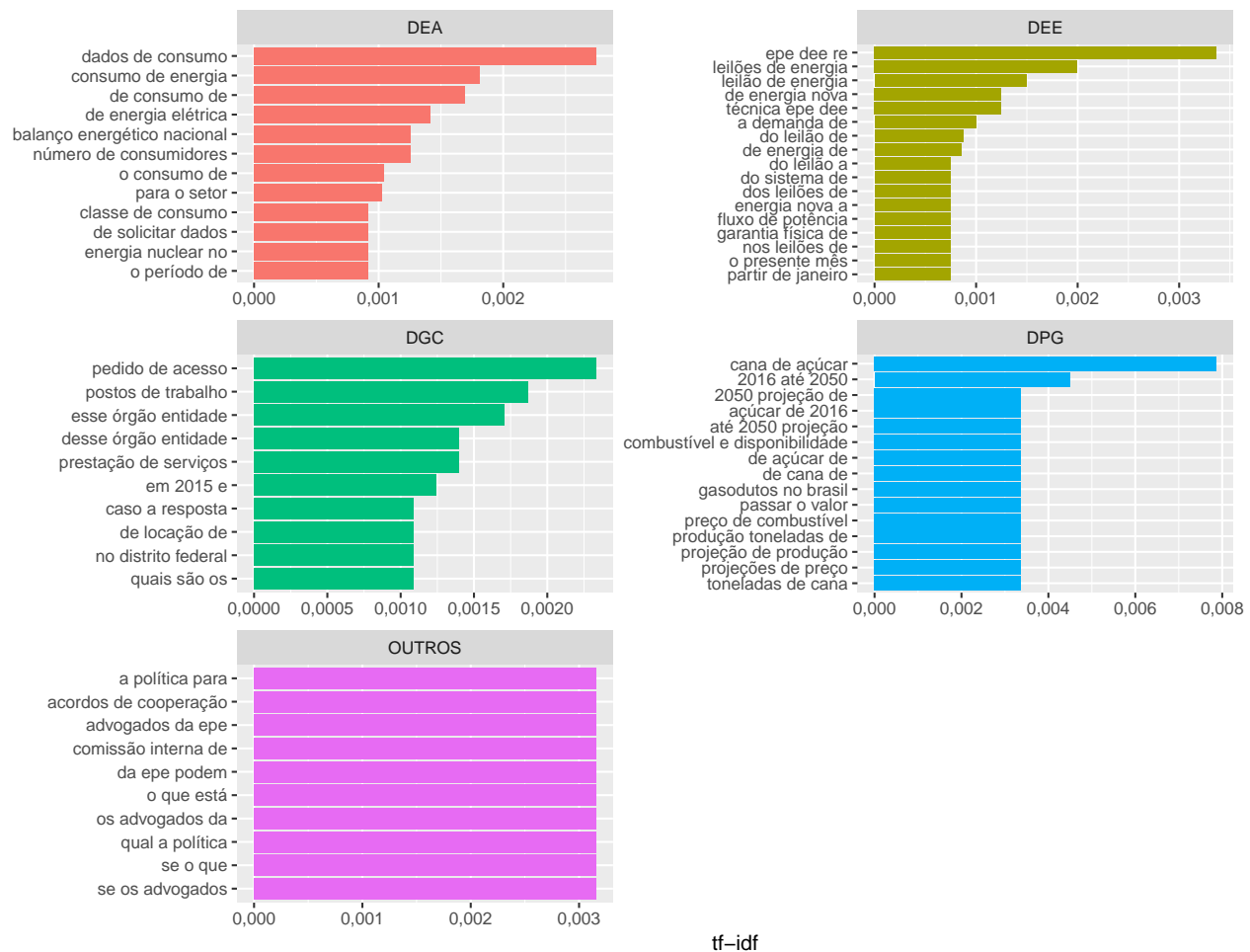
Usando bigram para n=3 palavras por token

Frequência de palavras por diretoria

```
diretoria_palavras_trigram <- DB %>%
  select(DESCRI_PEDIDO,DIRETORIA) %>%
  unnest_tokens(TRIGRAM, DESCRI_PEDIDO, token = "ngrams", n = 3) %>%
  count(DIRETORIA, TRIGRAM, sort = TRUE) %>%
  ungroup()
#diretoria_palavras_trigram

plot_diretoria_palavras_trigram <- diretoria_palavras_trigram %>%
  bind_tf_idf(TRIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(TRIGRAM = factor(TRIGRAM, levels = rev(unique(TRIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",
                                                    "DEE",
                                                    "DGC",
                                                    "DPG",
                                                    "OUTROS"))))
#View(head(plot_diretoria_palavras_trigram))
```

```
#jpeg("02_freq_palavras_dir.jpeg")
plot_diretoria_palavras_trigram %>%
  group_by(DIRETORIA) %>%
  top_n(10, tf_idf) %>%
  ungroup() %>%
  mutate(TRIGRAM = reorder(TRIGRAM, tf_idf)) %>%
  ggplot(aes(TRIGRAM, tf_idf, fill = DIRETORIA)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
  coord_flip() +
  scale_y_continuous(labels=gcomma)
```



```
#dev.off()
```

tidy object into document-term matrix

```
plot_diretoria_palavras <- diretoria_palavras %>%
  bind_tf_idf(palavra, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(palavra = factor(palavra, levels = rev(unique(palavra)))) %>%
```

```

mutate(DIRETORIA = factor(DIRETORIA, levels = c("DEA",
                                                "DEE",
                                                "DGC",
                                                "DPG",
                                                "OUTROS"))))

dtm = plot_diretoria_palavras %>%
cast_dtm(document = DIRETORIA, term = palavra, n)

```

Nuvem de palavras

Nuvem de palavras por diretoria - s/ steeming e/ c/ stopwords - onegram

```

#View(head(plot_diretoria_palavras))
library(wordcloud)
plot_diretorias_tf_idf = plot_diretoria_palavras %>%
  select(palavra, tf_idf, DIRETORIA) %>%
  mutate(palavra = reorder(palavra, tf_idf))

## DEE
#jpeg("XX_wordclou_tfidf_dir01_DEE.jpeg")
nuvem1 =
plot_diretorias_tf_idf %>%
  filter(DIRETORIA == "DEE") %>%
  select(-DIRETORIA, word = palavra, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(231321)
wordcloud(words = nuvem1$word, freq = nuvem1$freq, min.freq = 0.2,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))

```



```
as.data.frame()

set.seed(75437)
wordcloud(words = nuvem4$word, freq = nuvem4$freq, min.freq = 0.1,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))
```



```
## OUTROS
#jpeg("XX_wordclou_tf_idf_dir05_OUTROS.jpeg")
nuvem5 =
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "OUTROS") %>%
  select(-DIRETORIA, word = palavra, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(75437)
wordcloud(words = nuvem5$word, freq = nuvem5$freq, min.freq = 0.1,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))
```



```

#View(head(plot_diretoria_palavras))
library(wordcloud2)

plot_diretorias_tf_dif = plot_diretoria_palavras %>%
  select(palavra, tf_idf, DIRETORIA) %>%
  mutate(palavra = reorder(palavra, tf_idf))

## DEE
#jpeg("XX_wordclou_tfidf_dir01_DEE.jpeg")
set.seed(233115)
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "DEE") %>%
  top_n(150, tf_idf) %>%
  wordcloud2(shuffle = TRUE,
             color = "random-dark",
             shape = "circle")

## DGC
#jpeg("XX_wordclou_tfidf_dir01_DGC.jpeg")
set.seed(233115)
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "DGC") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## DEA
#jpeg("XX_wordclou_tfidf_dir01_DEA.jpeg")
set.seed(233115)
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "DEA") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## DPG
#jpeg("XX_wordclou_tfidf_dir04_DPG.jpeg")
set.seed(233115)
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "DPG") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## OUTROS
#jpeg("XX_wordclou_tfidf_dir01_OUTROS.jpeg")
set.seed(233115)
plot_diretorias_tf_dif %>%
  filter(DIRETORIA == "OUTROS") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

```

->

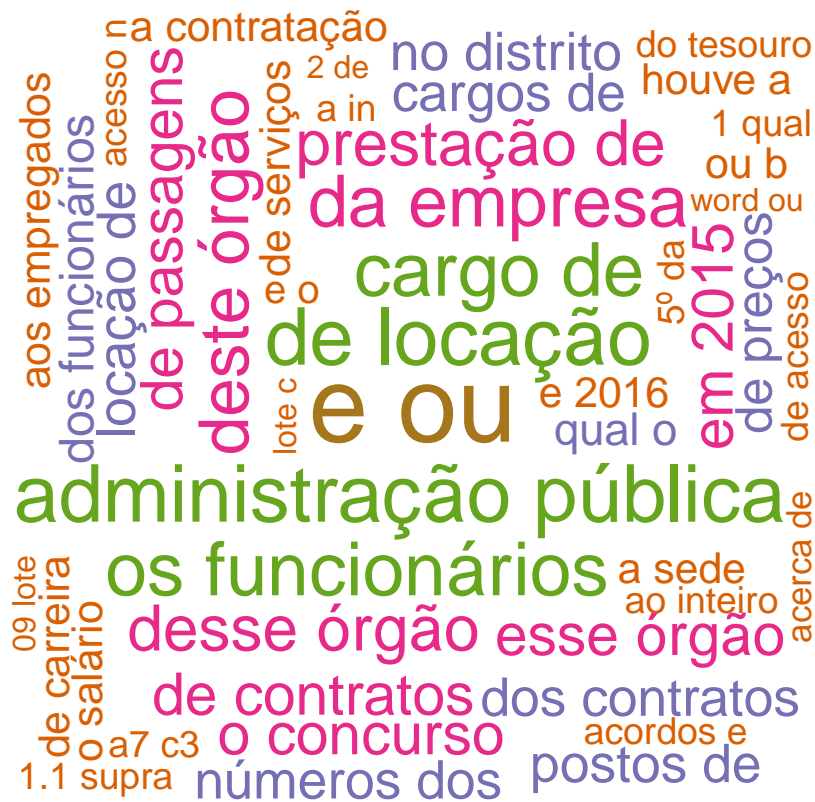
Nuvem de palavras por diretoria - s/ steeming e/ou remoção de stopwords - bigram


```

filter(DIRETORIA == "DGC") %>%
select(-DIRETORIA, word = BIGRAM,freq = tf_idf) %>%
#top_n(150, freq) %>%
as.data.frame()

set.seed(75437)
wordcloud(words = nuvem2.2$word, freq = nuvem2.2$freq, min.freq = 0.2,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))

```



```

## DEA
#jpeg("XX_wordclou_tfidf_dir03_DEA.jpeg")
nuvem3.2 =
plot_diretorias_tf_dif_bigram %>%
  filter(DIRETORIA == "DEA") %>%
  select(-DIRETORIA, word = BIGRAM,freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(543453)
wordcloud(words = nuvem3.2$word, freq = nuvem3.2$freq, min.freq = 0.2,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))

```




```
## OUTROS
#jpeg("XX_wordclou_tfidf_dir05_OUTROS.jpeg")
nuvem5.2 =
plot_diretorias_tf_dif_bigram %>%
  filter(DIRETORIA == "OUTROS") %>%
  select(-DIRETORIA, word = BIGRAM,freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(75437)
wordcloud(words = nuvem5.2$word, freq = nuvem5.2$freq, min.freq = 0.1,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```

de linhas a transmissão
 ser o labex invertido
 há no dos autos dia 20
 comissão interna
 advogados da
 acordos de 22 de
 ao público a política
 de cooperação
 epe podem
 programa labex
 iniciar a
 assim deixo
 lançado em

```
#View(head(plot_diretoria_palavras))
library(wordcloud2)

plot_diretorias_tf_dif_bigram = DB %>%
  select(DESCRI_PEDIDO,DIRETORIA) %>%
  unnest_tokens(BIGRAM, DESCRI_PEDIDO, token = "ngrams", n = 2) %>%
  count(DIRETORIA, BIGRAM, sort = TRUE) %>%
  bind_tf_idf(BIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(BIGRAM = factor(BIGRAM, levels = rev(unique(BIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA,levels=c("DEA","DEE","DGC","DPG","OUTROS"))) %>%
  select(BIGRAM, tf_idf, DIRETORIA)

## DEE
#jpeg("XX_wordclou_tfidf_dir01_DEE.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram %>%
  filter(DIRETORIA == "DEE") %>%
  top_n(150, tf_idf) %>%
  wordcloud2(shuffle = TRUE,
             color = "random-dark",
             shape = "circle")

## DGC
#jpeg("XX_wordclou_tfidf_dir01_DGC.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram %>%
```

```

filter(DIRETORIA == "DGC") %>%
top_n(150, tf_idf) %>%
wordcloud2()

## DEA
#jpeg("XX_wordclou_tfidf_dir01_DEA.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram %>%
  filter(DIRETORIA == "DEA") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## DPG
#jpeg("XX_wordclou_tfidf_dir01_DPG.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram %>%
  filter(DIRETORIA == "DPG") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

```

Separando palavras de um bigram em “palavra1” e “palavra2” p/ remover stopwords

Considerando já a exclusão de casos onde houver stopwords consecutivos na “palavra1” e “palavra2”, ou seja onde $palavra1 = stopword \wedge palavra2 = stopword$

```

bigrams = DB %>%
  select(DESCRI_PEDIDO,DIRETORIA) %>%
  unnest_tokens(BIGRAM, DESCRI_PEDIDO, token = "ngrams", n = 2) %>%
  count(DIRETORIA, BIGRAM, sort = TRUE)

separa_bigrams = bigrams %>%
  separate(BIGRAM, c("palavra1", "palavra2"), sep = " ")

junta_bigrams = separa_bigrams %>%
  unite(BIGRAM, palavra1, palavra2, sep = " ")
# levels(as.factor(junta_bigrams$BIGRAM == bigrams$BIGRAM)) # CHECK

## remove stopwords
bigrams2 = cbind(separa_bigrams,BIGRAM = junta_bigrams$BIGRAM) %>%
  filter(!palavra1 %in% mystopwords$palavra) %>%
  filter(!palavra2 %in% mystopwords$palavra) %>%
  filter(!palavra1 %in% "a") %>%
  filter(!palavra2 %in% "a") %>%
  filter(!palavra1 %in% "p") %>%
  filter(!palavra1 %in% "s") %>%
  filter(!palavra1 %in% "d") %>%
  filter(!palavra2 %in% "p") %>%
  filter(!palavra2 %in% "s") %>%
  filter(!palavra2 %in% "d") %>%
  filter(!palavra2 %in% "s.a") %>%
  filter(!str_detect(palavra1, "0")) %>%
  filter(!str_detect(palavra1, "1")) %>%
  filter(!str_detect(palavra1, "2")) %>%
  filter(!str_detect(palavra1, "3")) %>%

```



```

filter(!str_detect(palavra1, "4")) %>%
filter(!str_detect(palavra1, "5")) %>%
filter(!str_detect(palavra1, "6")) %>%
filter(!str_detect(palavra1, "7")) %>%
filter(!str_detect(palavra1, "8")) %>%
filter(!str_detect(palavra1, "9")) %>%
filter(!str_detect(palavra2, "0")) %>%
filter(!str_detect(palavra2, "1")) %>%
filter(!str_detect(palavra2, "2")) %>%
filter(!str_detect(palavra2, "3")) %>%
filter(!str_detect(palavra2, "4")) %>%
filter(!str_detect(palavra2, "5")) %>%
filter(!str_detect(palavra2, "6")) %>%
filter(!str_detect(palavra2, "7")) %>%
filter(!str_detect(palavra2, "8")) %>%
filter(!str_detect(palavra2, "9"))
#count(DIRETORIA, BIGRAM)

```

Nuvem de palavras por diretoria - s/ steeming c/ remoção de stopwords - bigram

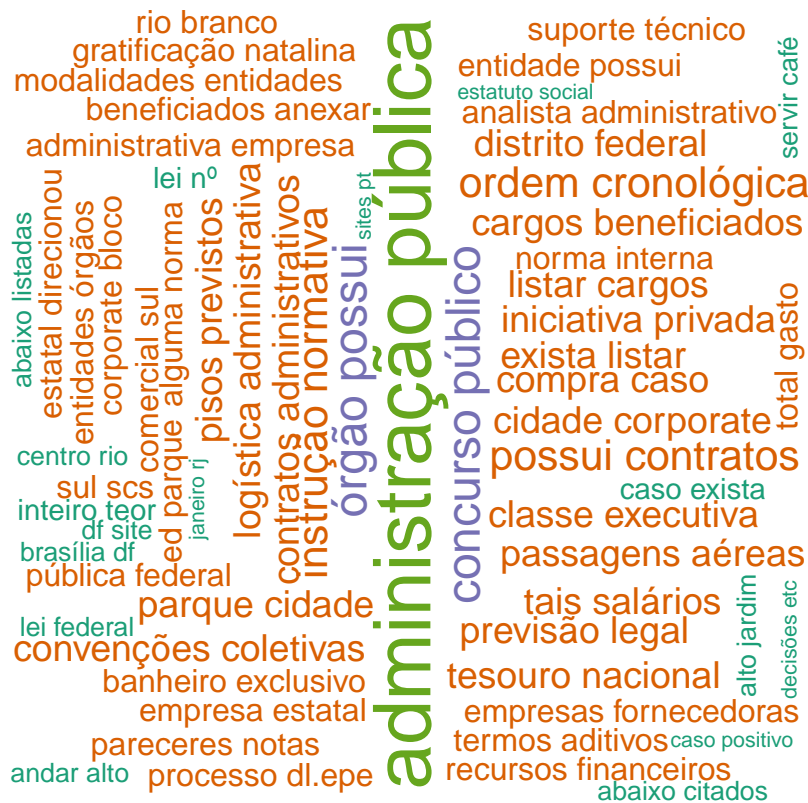
```

#View(head(plot_diretoria_palavras))
library(wordcloud2)
library(wordcloud)
plot_diretorias_tf_dif_bigram2 = bigrams2 %>%
  select(BIGRAM,n,DIRETORIA) %>%
  bind_tf_idf(BIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(BIGRAM = factor(BIGRAM, levels = rev(unique(BIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA,levels=c("DEA", "DEE", "DGC", "DPG", "OUTROS"))) %>%
  select(BIGRAM, tf_idf, DIRETORIA)

## DEE
#jpeg("XX_wordclou_tfidf_dir01_DEE.jpeg")
nuvem1.2 =
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DEE") %>%
  select(-DIRETORIA, word = BIGRAM,freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(231321)
wordcloud(words = nuvem1.2$word, freq = nuvem1.2$freq, min.freq = 0.2,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))

```

```
## DEA
#jpeg("XX_wordclou_tfidf_dir03_DEA.jpeg")
nuvem3.2 =
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DEA") %>%
  select(-DIRETORIA, word = BIGRAM,freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(543453)

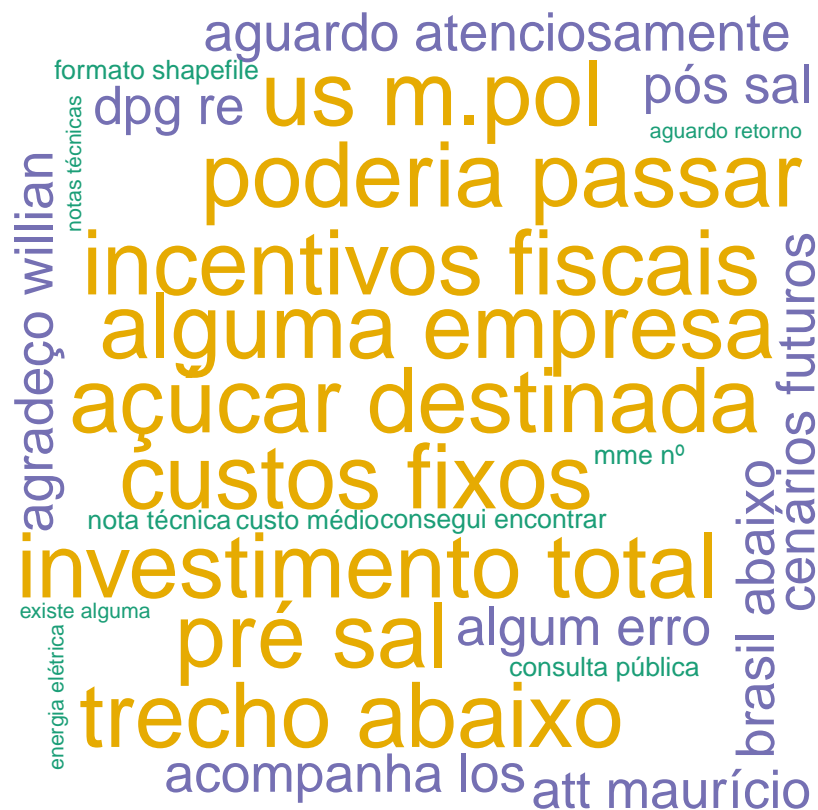
wordcloud(words = nuvem3.2$word, freq = nuvem3.2$freq,
          max.words=250, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(10, "Dark2"))
```

eia rima **impacto ambiental**



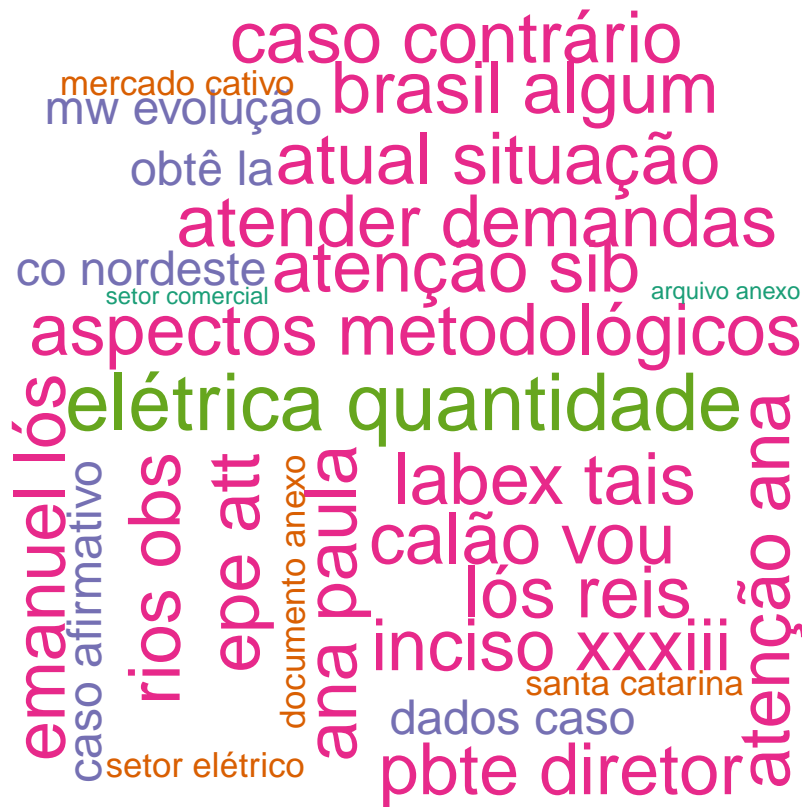
```
## DPG
#jpeg("XX_wordclou_tfidf_dir04_DPG.jpeg")
nuvem4.2 =
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DPG") %>%
  select(-DIRETORIA, word = BIGRAM, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(75437)
wordcloud(words = nuvem4.2$word, freq = nuvem4.2$freq, min.freq = 0.1,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```



```
## OUTROS
#jpeg("XX_wordclou_tfidf_dir05_OUTROS.jpeg")
nuvem5.2 =
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "OUTROS") %>%
  select(-DIRETORIA, word = BIGRAM, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(75437)
wordcloud(words = nuvem5.2$word, freq = nuvem5.2$freq, min.freq = 0.1,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```



```
#View(head(plot_diretoria_palavras))
library(wordcloud2)

plot_diretorias_tf_dif_bigram2 = bigrams2 %>%
  select(BIGRAM,n,DIRETORIA) %>%
  bind_tf_idf(BIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(BIGRAM = factor(BIGRAM, levels = rev(unique(BIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA,levels=c("DEA","DEE","DGC","DPG","OUTROS"))) %>%
  select(BIGRAM, tf_idf, DIRETORIA)

## DEE
#jpeg("XX_wordclou_tf_idf_dir01_DEE.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DEE") %>%
  top_n(150, tf_idf) %>%
  wordcloud2(shuffle = TRUE,
             color = "random-dark",
             shape = "circle")

## DGC
#jpeg("XX_wordclou_tf_idf_dir02_DGC.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DGC") %>%
  top_n(150, tf_idf) %>%
```

```

wordcloud2()

## DEA
#jpeg("XX_wordclou_tfidf_dir03_DEA.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DEA") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## DPG
#jpeg("XX_wordclou_tfidf_dir04_DPG.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "DPG") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## OUTROS
#jpeg("XX_wordclou_tfidf_dir05_OUTROS.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_bigram2 %>%
  filter(DIRETORIA == "OUTROS") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

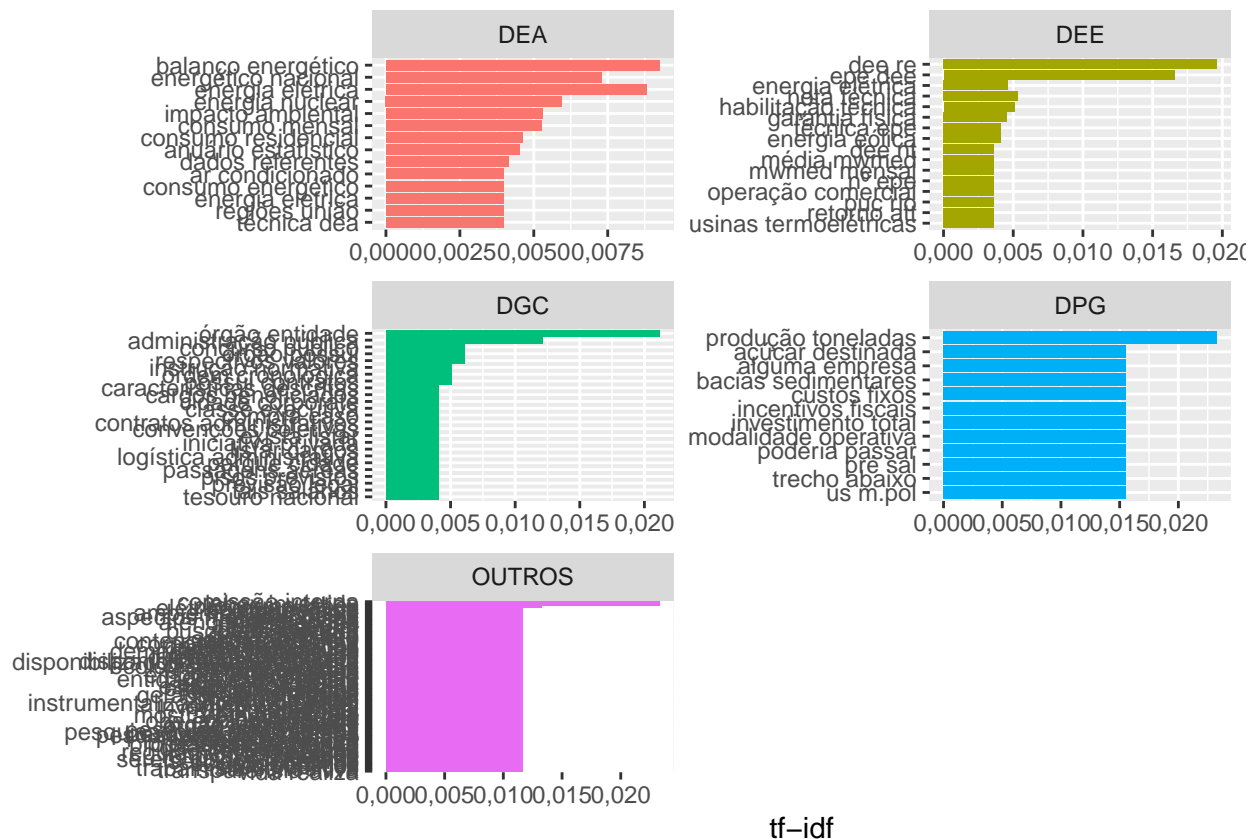
```

Gráfico da estatística tf_idf c/ remoção de stopwords

```

plot_diretorias_tf_dif_bigram2 %>%
  group_by(DIRETORIA) %>%
  top_n(10, tf_idf) %>%
  ungroup() %>%
  mutate(BIGRAM = reorder(BIGRAM, tf_idf)) %>%
  ggplot(aes(BIGRAM, tf_idf, fill = DIRETORIA)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~DIRETORIA, ncol = 2, scales = "free") +
  coord_flip() +
  scale_y_continuous(labels=gcomma)

```



Nuvem de palavras por diretoria - s/ stemming e/ou stopwords - trigram

```
#View(head(plot_diretoria_palavras))
library(wordcloud)
plot_diretorias_tf_idf_trigram = DB %>%
  select(DESCR_PEDIDO,DIRETORIA) %>%
  unnest_tokens(TRIGRAM, DESCR_PEDIDO, token = "ngrams", n = 3) %>%
  count(DIRETORIA, TRIGRAM, sort = TRUE) %>%
  bind_tf_idf(TRIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(TRIGRAM = factor(TRIGRAM, levels = rev(unique(TRIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA,levels=c("DEA","DEE","DGC","DPG","OUTROS"))) %>%
  select(TRIGRAM, tf_idf, DIRETORIA)

## DEE
#jpeg("wordcloud_tf_idf_dir01_DEE_trigram_comstop_semstemming.jpeg")
nuvem1.3 =
plot_diretorias_tf_idf_trigram %>%
  filter(DIRETORIA == "DEE") %>%
  select(-DIRETORIA, word = TRIGRAM,freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(8835)
wordcloud(words = nuvem1.3$word, freq = nuvem1.3$freq, min.freq = 0.2,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```




```
#dev.off()
```

```
## DGC
#jpeg("wordcloud_tf_idf_dir02_DGC_trigram_comstop_semstemming.jpeg")
nuvem2.3 =
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DGC") %>%
  select(-DIRETORIA, word = TRIGRAM, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(1273)
wordcloud(words = nuvem2.3$word, freq = nuvem2.3$freq, min.freq = 0.2,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```

de incentivo ao caso exista listar
o número de
nos termos do
em 2016 e
de cargos de
a razão da 09 lote c
a compra de
em 2015 e
2015 e 2016
a sede desse
a in 05
de acesso n
2004 a 2015
e a previsão
e ou no
ao inteiro teor
e gestão de
e ou repórter
ano de 2016
e tabela de
desse órgão entidade
e ou c de locação de nº 2 de
a folha de gráfico e ou c 3 a 7 c 3
características descritas no 4 qual o
pela administração pública
número de funcionários
as características descritas
à informação 1.1 os números dos
contratados pela administração

```
## DEA
#jpeg("XX_wordclou_tfidf_dir03_DEA_trigram_comstop_semstemming.jpeg")
nuvem3.3 =
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DEA") %>%
  select(-DIRETORIA, word = TRIGRAM, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(543453)
wordcloud(words = nuvem3.3$word, freq = nuvem3.3$freq,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```

entre os anos regiões união e de todas as
de 1980 a de 2000 a por classe de
do rio são
consumo mensal de
de solicitar dados
base de dados
união e estados
1990 a 2017
de impacto ambiental
da energia nuclear
banco de dados
o período de
classe de consumo
o consumo de
de consumo de
para o setor
energia nuclear no
mensal de energia
dos dados de
de energia elétrica
nota técnica de
anos de 2013
do estado de
brasil qual é
de energia elétrica
faixa de consumo
do grupo a
do brasil s.a
os anos de
os dados de
solicitar dados de

```
## DPG
#jpeg("XX_wordclou_tfidf_dir04_DPG_trigram_comstop_semstemming.jpeg")
nuvem4.3 =
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DPG") %>%
  select(-DIRETORIA, word = TRIGRAM, freq = tf_idf) %>%
  #top_n(150, freq) %>%
  as.data.frame()

set.seed(75437)
wordcloud(words = nuvem4.3$word, freq = nuvem4.3$freq, min.freq = 0.1,
  max.words=250, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(10, "Dark2"))
```


da constituição da
altera a corrente
a vinda de
a bancos e
se o que
o que está
transmissão de
transmissão e
que fazem a
a síntese desse
fazem a transmissão
ancorados em corrente
brasil tem a

```
#View(head(plot_diretoria_palavras))
library(wordcloud2)

plot_diretorias_tf_dif_trigram = DB %>%
  select(DESCRI_PEDIDO,DIRETORIA) %>%
  unnest_tokens(TRIGRAM, DESCRI_PEDIDO, token = "ngrams", n = 3) %>%
  count(DIRETORIA, TRIGRAM, sort = TRUE) %>%
  bind_tf_idf(TRIGRAM, DIRETORIA, n) %>%
  arrange(desc(tf_idf)) %>%
  mutate(TRIGRAM = factor(TRIGRAM, levels = rev(unique(TRIGRAM)))) %>%
  mutate(DIRETORIA = factor(DIRETORIA,levels=c("DEA","DEE","DGC","DPG","OUTROS"))) %>%
  select(TRIGRAM, tf_idf, DIRETORIA)

## DEE
#jpeg("XX_wordclou_tfidf_dir01_DEE.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DEE") %>%
  top_n(150, tf_idf) %>%
  wordcloud2(shuffle = TRUE,
             color = "random-dark",
             shape = "circle")

## DGC
#jpeg("XX_wordclou_tfidf_dir02_DGC.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_trigram %>%
```

```

filter(DIRETORIA == "DGC") %>%
top_n(150, tf_idf) %>%
wordcloud2()

## DEA
#jpeg("XX_wordclou_tfidf_dir03_DEA.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DEA") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## DPG
#jpeg("XX_wordclou_tfidf_dir04_DPG.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "DPG") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

## OUTROS
#jpeg("XX_wordclou_tfidf_dir05_OUTROS.jpeg")
set.seed(233115)
plot_diretorias_tf_dif_trigram %>%
  filter(DIRETORIA == "OUTROS") %>%
  top_n(150, tf_idf) %>%
  wordcloud2()

```

MODELAGEM - APLICAÇÃO E RESULTADOS

Preparação e partição de dados

```

key_DIR = plot_diretoria_palavras_noSTOP %>%
  group_by(DIRETORIA) %>%
  count(DIRETORIA)

key_DIR %>%
  kable("latex", caption = "Número de termos por diretoria",
        booktabs = T, format.args = list(decimal.mark = ',', big.mark = ".")) %>%
  kable_styling(latex_options = c("striped", "hold_position"))

```

Table 7: Número de termos por diretoria

DIRETORIA	n
DEA	1.829
DEE	1.921
DGC	1.507
DPG	408
OUTROS	348

selecionando as 20 palavras mais importantes de cada uma das 4 diretorias e da categoria 'OUTROS'.

```
# plot_diretoria_palavras_noSTOP$palavra <- gsub(".", "_", as.character(plot_diretoria_palavras_noSTOP$palavra))
# devo garantir que esses termos sao unicos dentro de cada diretoria
# devo, ainda, excluir acentos e pontos e caracteres q nao podem ser inclusos em nomes de variaveis no R
# Posteriormente devo refazer esses passos, juntamente com o stemming no arquivo de dados a ser inputado

termos_dir =
plot_diretoria_palavras_noSTOP %>%
group_by(DIRETORIA) %>%
top_n(50, tf_idf) %>%
select(palavra)
```

```
## Adding missing grouping variables: `DIRETORIA`
```

```
termos_dir <- unique(termos_dir);
```

Cria, antes, uma variável DESCRIPEDIDO1 que repete os passos feitos aos termos quanto ao stemming so que no texto fonte.

```
DB$DESCRIPEDIDO1 = DB$DESCRIPEDIDO
DB$DESCRIPEDIDO1 = ptstem(rslp(DB$DESCRIPEDIDO1))
```

```
fe <- matrix(data = 0, nrow = length(DB$DESCRIPEDIDO1), ncol = length(termos_dir$palavra))
fe <- data.frame(fe); colnames(fe) <- termos_dir$palavra
```

```
i=j=0
for(i in 1:length(DB$ID)){
  for(j in 1:length(termos_dir$palavra)){
    g <- grepl(termos_dir$palavra[j], DB$DESCRIPEDIDO1[i])
    if(g == TRUE){
      fe[i, j] <- 1
    }
  }
}
```

```
#REMOVE A VARIÁVEL geracao que apareceu em deduplicacao
fe = fe[, -50]
```

```
termo_dir1 = termos_dir %>%
select(DIRETORIA, palavra) %>%
mutate(termo = palavra)
```

```
NumTermos = as_tibble(rbind(apply(fe,2,sum)))
NumTermos = gather(NumTermos, key = "termo", value = "Num_Pedidos")
NumTermos = NumTermos[order(NumTermos$Num_Pedidos, decreasing = TRUE), ]
```

```
highchart() %>%
  hc_add_series(data = NumTermos$Num_Pedidos,
    type = "bar",
    name = "# de pedidos",
    showInLegend = FALSE,
    tooltip = list(valueDecimals = 0, valuePrefix = "", valueSuffix = ""), color="blue") %>%
  hc_yAxis(title = list(text = "Quantitativo de pedidos"),
    allowDecimals = TRUE, max = (max(NumTermos$Num_Pedidos)+103),
    labels = list(format = "{value}")) %>%
  hc_xAxis(title = list(text = "Termo"),
```

```

        categories = NumTermos$termo,
        tickmarkPlacement = "on",
        opposite = FALSE) %>%
hc_title(text = "Quantitativo de pedidos por termo (sem exclusividade)",
        style = list(fontWeight = "bold")) %>%
hc_subtitle(text = paste("")) %>%
    hc_tooltip(valueDecimals = 2,
               pointFormat = "{point.y} pedidos")%>%
               #pointFormat = "Variável: {point.x} <br> Missing: {point.y}")
    hc_credits(enabled = TRUE,
               text = "Fonte: CGU, e-SIC (2019). Elaboração: Ewerson Pimenta.",
               style = list(fontSize = "10px")) %>%
hc_exporting(enabled = TRUE, filename = "F3-filmes-genero-Pimenta")

```

```
db_modelo = as_tibble(cbind(select(DB,DIRETORIA),fe))
```

```
# __Porcentagem de ZEROS por variável__
```

```

zeros <- (colSums(fe==0)/nrow(fe)*100); var <- names(fe)
db_zero <- data.frame(var,zeros); rownames(db_zero) <- NULL
db_zero <- db_zero[order(db_zero$zeros, decreasing = TRUE), ]

```

```

hc4_1 <- highchart() %>%
  hc_add_series(data = db_zero$zeros,
               type = "bar",
               name = "Porcentagem de zeros",
               showInLegend = FALSE,
               tooltip = list(valueDecimals = 2, valuePrefix = "", valueSuffix = " %"), color="pink")
  hc_yAxis(title = list(text = "Porcentagem de zero"),
           allowDecimals = TRUE, max = 100,
           labels = list(format = "{value}%")) %>%
  hc_xAxis(categories = db_zero$var,
           tickmarkPlacement = "on",
           opposite = FALSE) %>%
  hc_title(text = "Porcentagem de zeros por variável",
          style = list(fontWeight = "bold")) %>%
  hc_subtitle(text = paste("")) %>%
    hc_tooltip(valueDecimals = 2,
               pointFormat = "Zeros: {point.y}")%>%
               #pointFormat = "Variável: {point.x} <br> Missing: {point.y}")
    hc_credits(enabled = TRUE,
               text = "Fonte: IMDB/KAGGLE. Elaboração: Ewerson Pimenta.",
               style = list(fontSize = "10px")) %>%
  hc_exporting(enabled = TRUE, filename = "Fig00-Pimenta")
#hc <- hc %>%
# hc_add_theme(hc_theme_darkunica())
hc4_1; remove(hc4_1, var, zeros)

```


Modelos de classificação

Partição dos dados

Particionando a base de dados em Treino e Teste, esses dois (Treino e Teste) também terão armazenados as diretorias que foram responsáveis por cada pedido via amostragem probabilística dos dados originais separadamente das bases de Treino e Teste.

Para amostragem aleatória simples

```
intrain <- createDataPartition(y = db_modelo$DIRETORIA, p = 0.65, list = FALSE)
training <- db_modelo[intrain,]
testing <- db_modelo[-intrain,]
```

Modelagem 1 - Random Forest (RF)

Random Forest (RF) - Metodologia

- Descrição**
1. Random Forest foi desenvolvido para agregar árvores de decisão (modelo de classificação);
 2. Pode ser usado para modelo de classificação (p/ var. resposta categórica) ou regressão (no caso de haver variável resposta contínua);
 3. Evita *overfitting*;
 4. Permite trabalhar com um largo número de características de um conjunto de dados;
 5. Auxilia na seleção de variáveis baseada em um algoritmo que calcula a importância por variável (assim, tendo conhecimento de quais variáveis são mais importantes, podemos usar essa informação para outros modelos de classificação);
 6. User-friendly: apenas 2 parâmetros livres:
 - Trees - *ntrees*, default 500 (Nº de árvores);
 - Variáveis selecionadas via amostragem aleatória candidatas à cada “split” (quebra da árvore) - *mtry*, default \sqrt{p} p/ classificação e $\frac{p}{3}$ p/ regressão (p: nº de features/variáveis);

Passo-a-Passo

É realizado em 3 passos:

1. Desenha as amostras via bootstrap do número de árvores *ntrees*;
2. Para cada amostra via bootstrap, cresce o número de árvores “un-puned” para a escolha da melhor quebra da árvore baseado na amostra aleatória do valor predito de *mtry* a cada nó da árvore;
- 3. Faz classificação de novos valores usando a maioria de votos p/ classificação e usa a média p/ regressão baseada nas amostras de *ntrees*.

Random Forest - Aplicação e Resultados

Inicialmente utilizaremos o pacote **randomForest** que implementa o algoritmo de Random Forest de Breiman (baseado na clusterização de Breiman, originalmente codificada em Fortran) que tem por finalidade classificar e/ou criar regressão. Além disso, pode ser usado em um modelo não supervisionado para avaliar proximidades entre pontos.

Estamos usando, a partir daqui, a base de treino.

```
#library(randomForest)
#library(rpart)
#library(rpart.plot)
#rf <- randomForest(proximity = T, ntree = 38, do.trace = T, WR~., data=training)
```

```

set.seed(9984512)
# Training with classification tree
rf <- rpart(DIRETORIA ~ ., data=training, method="class", xval = 4)
print(rf, digits = 3)

## n= 369
##
## node), split, n, loss, yval, (yprob)
##      * denotes terminal node
##
##  1) root 369 232 DEA (0.37 0.35 0.2 0.043 0.035)
##    2) consumo>=0.5 72   8 DEA (0.89 0.083 0 0.014 0.014) *
##    3) consumo< 0.5 297 175 DEE (0.25 0.41 0.25 0.051 0.04)
##      6) orgao< 0.5 272 152 DEE (0.26 0.44 0.2 0.055 0.04)
##        12) concurso< 0.5 262 142 DEE (0.27 0.46 0.17 0.057 0.042)
##          24) contratar< 0.5 247 129 DEE (0.29 0.48 0.13 0.057 0.045)
##            48) interna< 0.5 237 119 DEE (0.3 0.5 0.11 0.059 0.038)
##              96) rs>=0.5 39   16 DEA (0.59 0.33 0.051 0 0.026) *
##                97) rs< 0.5 198   93 DEE (0.24 0.53 0.12 0.071 0.04)
##                  194) industrial>=0.5 7    1 DEA (0.86 0.14 0 0 0) *
##                    195) industrial< 0.5 191   87 DEE (0.21 0.54 0.13 0.073 0.042)
##                      390) carga< 0.5 182   80 DEE (0.22 0.56 0.099 0.077 0.044) *
##                        391) carga>=0.5 9    3 DGC (0.11 0.22 0.67 0 0) *
##                          49) interna>=0.5 10    3 DGC (0.1 0 0.7 0 0.2) *
##                            25) contratar>=0.5 15    4 DGC (0.067 0.13 0.73 0.067 0) *
##                              13) concurso>=0.5 10    0 DGC (0 0 1 0 0) *
##                                7) orgao>=0.5 25    4 DGC (0.04 0.08 0.84 0 0.04) *

attributes(rf)

## $names
## [1] "frame"           "where"           "call"
## [4] "terms"           "cptable"         "method"
## [7] "parms"           "control"         "functions"
## [10] "numresp"         "splits"          "variable.importance"
## [13] "y"               "ordered"
##
## $xlevels
## named list()
##
## $ylevels
## [1] "DEA"   "DEE"   "DGC"   "DPG"   "OUTROS"
##
## $class
## [1] "rpart"

# Predict the testing set with the trained model
predictions1 <- predict(rf, testing, type = "class")

# Accuracy and other metrics
confusionMatrix(predictions1, as.factor(testing$DIRETORIA))

## Confusion Matrix and Statistics
##
##              Reference

```

```
## Prediction DEA DEE DGC DPG OUTROS
##   DEA      36   7   0   2       0
##   DEE      29  53  13   6       6
##   DGC       8   8  27   0       1
##   DPG       0   0   0   0       0
##   OUTROS    0   0   0   0       0
##
## Overall Statistics
##
##           Accuracy : 0.5918
##           95% CI : (0.5195, 0.6613)
##   No Information Rate : 0.3724
##   P-Value [Acc > NIR] : 3.966e-10
##
##           Kappa : 0.3991
##   McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: DEA Class: DEE Class: DGC Class: DPG
## Sensitivity           0.4932      0.7794      0.6750      0.00000
## Specificity           0.9268      0.5781      0.8910      1.00000
## Pos Pred Value        0.8000      0.4953      0.6136      NaN
## Neg Pred Value        0.7550      0.8315      0.9145      0.95918
## Prevalence            0.3724      0.3469      0.2041      0.04082
## Detection Rate        0.1837      0.2704      0.1378      0.00000
## Detection Prevalence  0.2296      0.5459      0.2245      0.00000
## Balanced Accuracy      0.7100      0.6788      0.7830      0.50000
##
##           Class: OUTROS
## Sensitivity            0.00000
## Specificity            1.00000
## Pos Pred Value         NaN
## Neg Pred Value         0.96429
## Prevalence             0.03571
## Detection Rate         0.00000
## Detection Prevalence   0.00000
## Balanced Accuracy      0.50000
```

Olhando as 6 primeiras observações real X predito

```
p1 <- predict(rf,training)
head(p1)
```

```
##           DEA      DEE      DGC      DPG      OUTROS
## 1 0.8888889 0.08333333 0.0000000 0.01388889 0.01388889
## 2 0.8888889 0.08333333 0.0000000 0.01388889 0.01388889
## 3 0.0400000 0.08000000 0.8400000 0.00000000 0.04000000
## 4 0.8888889 0.08333333 0.0000000 0.01388889 0.01388889
## 5 0.2197802 0.56043956 0.0989011 0.07692308 0.04395604
## 6 0.8888889 0.08333333 0.0000000 0.01388889 0.01388889
```

```
head(training$DIRETORIA)
```

```
## [1] "DEA" "DEA" "DEA" "DEE" "DEA" "DEA"
```

Selecioneando uma árvore

```
rp <- rpart::rpart(formula = DIRETORIA~.,data=training)
```

```
set.seed(09986755)
rf1 <- randomForest(DIRETORIA ~ ., data=training,
                    importance = TRUE,
                    proximity = TRUE)
```

Tuning do modelo

a partir de $n = 200$ árvores o erro OOB tende a estabilizar

Aparentemente $mtry = 18$ é um bom palpite

Reescrevendo, então, o modelo

```
set.seed(09986755)
# Training with classification tree
rf2 <- randomForest(as.factor(DIRETORIA) ~ ., data=training,
                   ntree = 200,
                   mtry = 18,
                   importance = TRUE,
                   proximity = TRUE)

rf2
```

```
# Predict the testing set with the trained model
predictions2 <- predict(rf2, testing, type = "class")
```

```
# Accuracy and other metrics
confusionMatrix(predictions2, as.factor(testing$DIRETORIA))
```

```
RF_importance = randomForest::importance(rf2)[order(randomForest::importance(rf2)[,1], decreasing = TRUE)]
randomForest::varImpPlot(rf2)
#rpart::plotcp(rf2)
#rpart.plot(rf2)
#rpart.plot.version1(rf2)
```

Taxa de Erro Random Forest

```
plot(rf2)
legend('topright', colnames(rf2$serr.rate), col=1:5, fill=1:5)
```

Histograma do Número de nós por árvore

```
hist(treesize(rf2), probability = T,
     main = "Distribuição do nº de nós por árvore",
     col = "green")
```

```
edit(MDSplot)
fig.align="center"
(MDIM_treino = MDSplot(rf2, training$DIRETORIA, pch=20))
#(MDIM_teste = MDSplot(rf2, testing$DIRETORIA, pch=20))
sum(MDIM_treino$eig[1:2])
```