Insper

Análise de texto

Aula 2

Magno Severino
PADS - Aprendizagem Estatística de Máquina II

Objetivos de aprendizagem

Ao final dessa aula você deverá ser capaz de

- compreender como representar dados textuais em formato adequado para processamento e modelagem;
- calcular e interpretar métricas que mensuram a importância de termos para um documento em uma coleção de documentos;
- implementar o modelo não supervisionado de tópicos LDA para agrupar documentos semelhantes.

CLASSICAL MACHINE LEARNING

Data is pre-categorized or numerical

Data is not labeled in any way

SUPERVISED

Predict a category

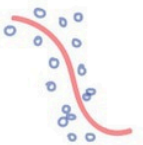
Predict a number

CLASSIFICATION

«Divide the socks by color»

REGRESSION

«Divide the ties by length»



UNSUPERVISED

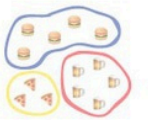
Divide by similarity

Find hidden dependencies

Identify sequences

CLUSTERING

«Split up similar clothing into stacks»



ASSOCIATION

«Find what clothes I often wear together»

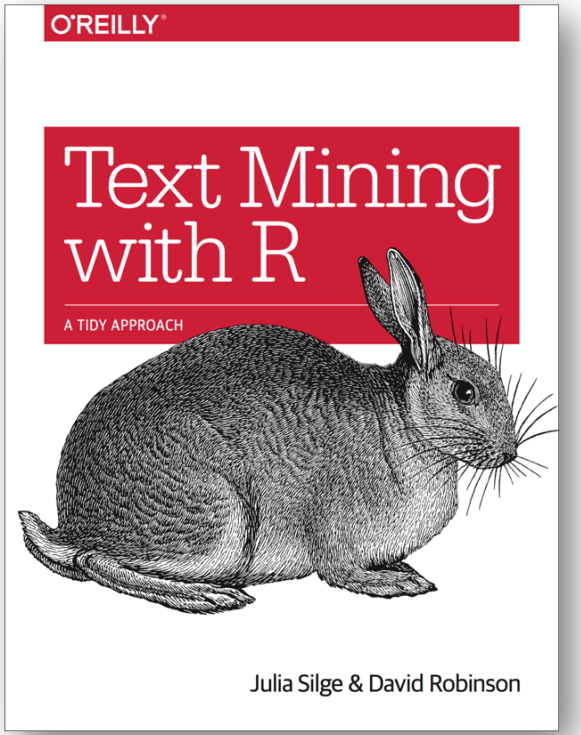
DIMENSION REDUCTION (generalization)

«Make the best outfits from the given clothes»

Introdução

- Textos são uma fonte valiosa de informações relevantes para diversos aspectos da vida social.
- Opiniões de consumidores a respeito de produtos e serviços, expressas em redes sociais, afetam as chances de bons resultados de um negócio.
- Opiniões sobre figuras políticas e personalidades midiáticas acabam por selar trajetórias de sucesso ou fracasso.
- O crescimento das redes sociais deu proeminência a este tipo de dado e sua modelagem.
- De fato, a incorporação efetiva da informação de dados textuais em análises mais tradicionais ainda é um problema com muito espaço para desenvolvimento e inovação.

Referência



- Disponível em https://www.tidytextmining.com/.
- Diversas análises e estudos de caso.

Definições

- **corpus**: uma coleção de documentos de texto;
- **tokens**: é uma unidade de texto. Normalmente é dado por apenas uma palavra, mas pode ser formado por um n-grama, parágrafo, linha ou caractere. Tokenization é o processo de dividir o texto em unidades de texto;
- **n-gramas**: sequência de n palavras consecutivas. Por vezes é interessante considerar uma combinação de termos como um novo termo;
- **stop words**: palavras que não apresentam informação significativa. Por exemplo, "de", "que" e "ao";
- **stemming**: redução de palavras para uma palavra base. Por exemplo, "químic" pode ser considerada a base das palavras "química", "químicas", "químico" e "químicos".

Sacola de palavras (bag-of-words)

Nessa abordagem, o que importa são os termos e não a ordem em que aparecem.

Então, os documentos abaixo são considerados iguais

```
doc1 <- "Magno foi à casa de Yasmin"

doc2 <- "Yasmin foi à casa de Magno"
```

pois que contém as mesmas palavras:

```
## [1] "à"      "casa"   "de"     "foi"    "Magno"  "Yasmin"
```

Representação tabular:

doc	casa	de	foi	magno	yasmin	à
1	1	1	1	1	1	1
2	1	1	1	1	1	1

Neste caso, podemos definir os tri-gramas "casa de Yasmin" e "casa de Magno".

Corpus

```
library(tm)

frases <- c("Não sei se é fato ou se é fita. Não sei se é fita ou fato. O fato é
que você me fita e fita mesmo de fato.",

            "Se cada um vai a casa de cada um é porque cada um quer que cada um
vá lá. Porque se cada um não fosse a casa de cada um é porque cada um não queria
que cada um fosse lá.",

            "O doce perguntou pro doce qual é o doce mais doce que o doce de
batata-doce. O doce respondeu pro doce que o doce mais doce que o doce de batata-
doce é o doce de doce de batata-doce.")

(corpus <- VCorpus(VectorSource(frases)))
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 3
```

O **corpus** é o conjunto de documentos com os quais estamos trabalhando. Nesse caso, o conjunto de trava-línguas.

Corpus

```
inspect(corpus)
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 3
##
## [[1]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 106
##
## [[2]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 169
##
## [[3]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 182
```

Corpus

```
inspect(corpus[[1]])
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 106
##
## Não sei se é fato ou se é fita. Não sei se é fita ou fato. O fato é que você me
fita e fita mesmo de fato.
```

```
meta(corpus[[1]])
```

```
##   author       : character(0)
##   datetimestamp: 2024-10-22 21:43:28
##   description  : character(0)
##   heading      : character(0)
##   id           : 1
##   language     : en
##   origin       : character(0)
```

Corpus

Pode ser do nosso interesse organizar os dados de acordo com algum critério. A função **tm_map** permite fazer diversas operações, como colocar todos caracteres em minúsculo, remover espaços em branco repetidos, remover palavras específicas, remover números, remover pontuação e *stemizar* as palavras.

```
corpus <- corpus %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(stripWhitespace) %>%
  tm_map(removeNumbers) %>%
  #tm_map(removeWords, c("palavra1", "palavra2")) %>%
  tm_map(removePunctuation) %>%
  tm_map(removeWords,
         iconv(stopwords("portuguese"), "UTF-8")) %>%
  tm_map(stemDocument, language = "portuguese")
```

Corpus

```
frases[1] # doc 1 original
```

```
## [1] "Não sei se é fato ou se é fita. Não sei se é fita ou fato. O fato é que
você me fita e fita mesmo de fato."
```

```
corpus[[1]]$content # doc 1 processado
```

```
## [1] "sei é fato é fita sei é fita fato fato é fita fita fato"
```

```
frases[2] # doc 2 original
```

```
## [1] "Se cada um vai a casa de cada um é porque cada um quer que cada um vá lá.
Porque se cada um não fosse a casa de cada um é porque cada um não queria que cada
um fosse lá."
```

```
corpus[[2]]$content # doc 2 processado
```

```
## [1] "cada vai casa cada é porqu cada quer cada vá lá porqu cada casa cada é
porqu cada queria cada lá"
```

Corpus

```
frases[3] # doc 3 original
```

```
## [1] "O doce perguntou pro doce qual é o doce mais doce que o doce de batata-
doce. O doce respondeu pro doce que o doce mais doce que o doce de batata-doce é o
doce de doce de batata-doce."
```

```
corpus[[3]]$content # doc 3 processado
```

```
## [1] "doce perguntou pro doce é doce doce doce batatadoc doce respondeu pro doce
doce doce doce batatadoc é doce doce batatadoc"
```

Document term matrix (dtm)

Vamos transformar o corpus numa matriz documento-termo. Nesse tipo de objeto, as linhas indicam os documentos e as colunas os termos.

```
dtm <- DocumentTermMatrix(corpus)

dtm
```

```
## <<DocumentTermMatrix (documents: 3, terms: 14)>>
## Non-/sparse entries: 14/28
## Sparsity           : 67%
## Maximal term length: 9
## Weighting          : term frequency (tf)
```

Em *Non-/sparse entries*, 14 é o número de entradas não vazias da matriz e 28 é o número de entradas vazias. Assim, a esparsidade da matriz é de $67\% \approx 100 \times \frac{28}{14+28}$.

Document term matrix (dtm)

```
inspect(dtm)
```

```
## <<DocumentTermMatrix (documents: 3, terms: 14)>>
## Non-/sparse entries: 14/28
## Sparsity           : 67%
## Maximal term length: 9
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs batatadoc cada casa doce fato fita perguntou porqu pro sei
##    1         0    0    0    0    4    4         0     0   0   2
##    2         0    8    2    0    0    0         0     3   0   0
##    3         3    0    0   12    0    0         1     0   2   0
```

Veja o que o comando `as.matrix(dtm)` faz.

Abordagem `tidy`

```
library(tidytext)
tidy(dtm)
```

document	term	count
1	fato	4
1	fita	4
1	sei	2
2	cada	8
2	casa	2
2	porqu	3
2	quer	1
2	queria	1
2	vai	1
3	batatadoc	3

Abordagem `tidy`

A função `unnest_tokens` permite obter os *tokens* (palavras) a partir de frases ou documentos.

```
corpus2 <- tibble(
  id = 1:3,
  text = c("Não sei se é fato ou se é fita. Não sei se é fita ou fato. O fato é
que você me fita e fita mesmo de fato.",

           "Se cada um vai a casa de cada um é porque cada um quer que cada um vá
lá. Porque se cada um não fosse a casa de cada um é porque cada um não queria que
cada um fosse lá.",

           "O doce perguntou pro doce qual é o doce mais doce que o doce de
batata-doce. O doce respondeu pro doce que o doce mais doce que o doce de batata-
doce é o doce de doce de batata-doce."))

corpus2 %>%
  unnest_tokens(term, text) %>%
  count(id, term)
```

Abordagem `tidy`

A função `unnest_tokens` permite obter os *tokens* (palavras) a partir de frases ou documentos.

id	term	n
1	fato	4
1	fita	4
1	é	4
1	se	3
1	não	2
2	cada	8
2	um	8
2	porque	3
2	a	2
2	casa	2
3	doce	15
3	o	7
3	de	4
3	batata	3
3	que	3

Note que aqui não fizemos nenhum tratamento nas palavras, como feito anteriormente.

Abordagem `tidy`

Podemos usar a função `unnest_tokens` considerando bi-gramas.

```
corpus2 %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  count(id, bigram) %>%
  arrange(desc(n))
```

id	bigram	n
2	cada um	8
3	o doce	7
3	doce de	4
1	se é	3
3	batata doce	3
3	de batata	3
3	doce que	3
3	que o	3
1	não sei	2
1	sei se	2
1	é fita	2
2	a casa	2

Análise de frequência de palavras

- Uma questão importante em *text mining* e procesamento de linguagem natural é como definir o assunto que um texto aborda;
- Uma medida de quão importante uma palavra pode ser é a frequência do termo (*term frequency - tf*), que mede a frequência de ocorrência de uma palavra em um documento;
- Entretanto, existem palavras em um documento que ocorrem muitas vezes mas não são importantes. Em português, essas palavras poderiam ser "o", "a", "é", "de", etc;
- Podemos adicionar tais palavras em uma lista (*stop words*) e removê-las antes da análise, mas é possível que algumas dessas palavras sejam mais importantes em alguns documentos que em outros. Portanto, essa não é uma abordagem sofisticada para computar a frequência de termos;
- Uma alternativa é utilizar a *inverse document frequency* (idf), que diminui o peso de termos usados frequentemente e aumenta o peso de palavras que não são usadas com muita frequência em uma coleção de documentos;
- Essa métrica pode ser combinada com a *term frequency* para obter a *tf-idf* de um termo (tf multiplicado por idf), a frequência de um termo ajustado por quão raro ele é utilizado.

The statistic tf-idf is intended to measure how important a word is to a document in a collection (or *corpus*) of documents, for example, to one novel in a collection of novels or to one website in a collection of websites.

Retirado do livro Text Mining with R Silge, J. and Robinson, D. 2017.

Definições de *tf* e *idf*

A *term frequency* de um dado termo é definida por

$$tf(\text{term}) = \frac{\text{frequência do termo no documento}}{\text{número de termos no documento}}.$$

A *inverse document frequency* de um dado termo é definida por

$$idf(\text{term}) = \ln\left(\frac{\text{número de documentos}}{\text{número de documentos contendo o termo}}\right).$$

Consideramos a quantidade $tf \times idf$ para avaliar a importância do termo para um documento. Veja a tabela a seguir com os termos do corpus dos trava-línguas.

doc	term	freq	n_terms_doc	n_docs_contendo	tf	idf	tf_idf
1	de	1	28	3	?	?	?
1	e	1	28	1	?	?	?
1	fato	4	28	1	?	?	?
1	fita	4	28	1	?	?	?
1	me	1	28	1	?	?	?
1	mesmo	1	28	1	?	?	?

Definições de *tf* e *idf*

A *term frequency* de um dado termo é definida por

$$tf(\text{term}) = \frac{\text{frequência do termo no documento}}{\text{número de termos no documento}}.$$

A *inverse document frequency* de um dado termo é definida por

$$idf(\text{term}) = \ln\left(\frac{\text{número de documentos}}{\text{número de documentos contendo o termo}}\right).$$

Consideramos a quantidade $tf \times idf$ para avaliar a importância do termo para um documento. Veja a tabela a seguir com os termos do corpus dos trava-línguas.

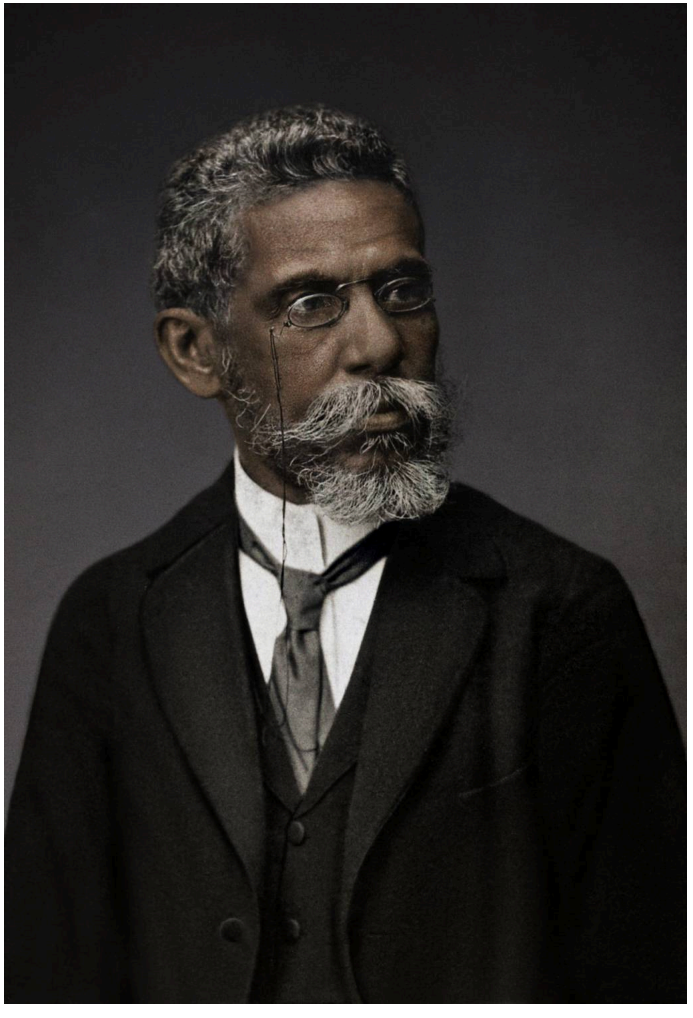
doc	term	freq	n_terms_doc	n_docs_contendo	tf	idf	tf_idf
1	de	1	28	3	0.036	0.000	0.000
1	e	1	28	1	0.036	1.099	0.040
1	fato	4	28	1	0.143	1.099	0.157
1	fita	4	28	1	0.143	1.099	0.157
1	me	1	28	1	0.036	1.099	0.040
1	mesmo	1	28	1	0.036	1.099	0.040

tf-idf com `tidytext`

```
corpus2 %>%
  unnest_tokens(term, text) %>%
  count(id, term) %>%
  bind_tf_idf(term, id, n)
```

id	term	n	tf	idf	tf_idf
1	de	1	0.0357	0.0000	0.0000
1	e	1	0.0357	1.0986	0.0392
1	fato	4	0.1429	1.0986	0.1569
1	fita	4	0.1429	1.0986	0.1569
1	me	1	0.0357	1.0986	0.0392
1	mesmo	1	0.0357	1.0986	0.0392
1	não	2	0.0714	0.4055	0.0290
1	o	1	0.0357	0.4055	0.0145
1	ou	2	0.0714	1.0986	0.0785
1	que	1	0.0357	0.0000	0.0000
1	se	3	0.1071	0.4055	0.0434
1	sei	2	0.0714	1.0986	0.0785
1	você	1	0.0357	1.0986	0.0392
1	é	4	0.1429	0.0000	0.0000
2	a	2	0.0488	1.0986	0.0536

Obras de Machado de Assis



Vamos analisar duas obras de Joaquim Maria Machado de Assis:

- Memórias Póstumas de Brás Cubas,
- Dom Casmurro.

Dom Casmurro

Memórias Póstumas de B. Cubas

Frequência dos termos

O gráfico abaixo mostra os 15 termos mais frequentes em cada obra.

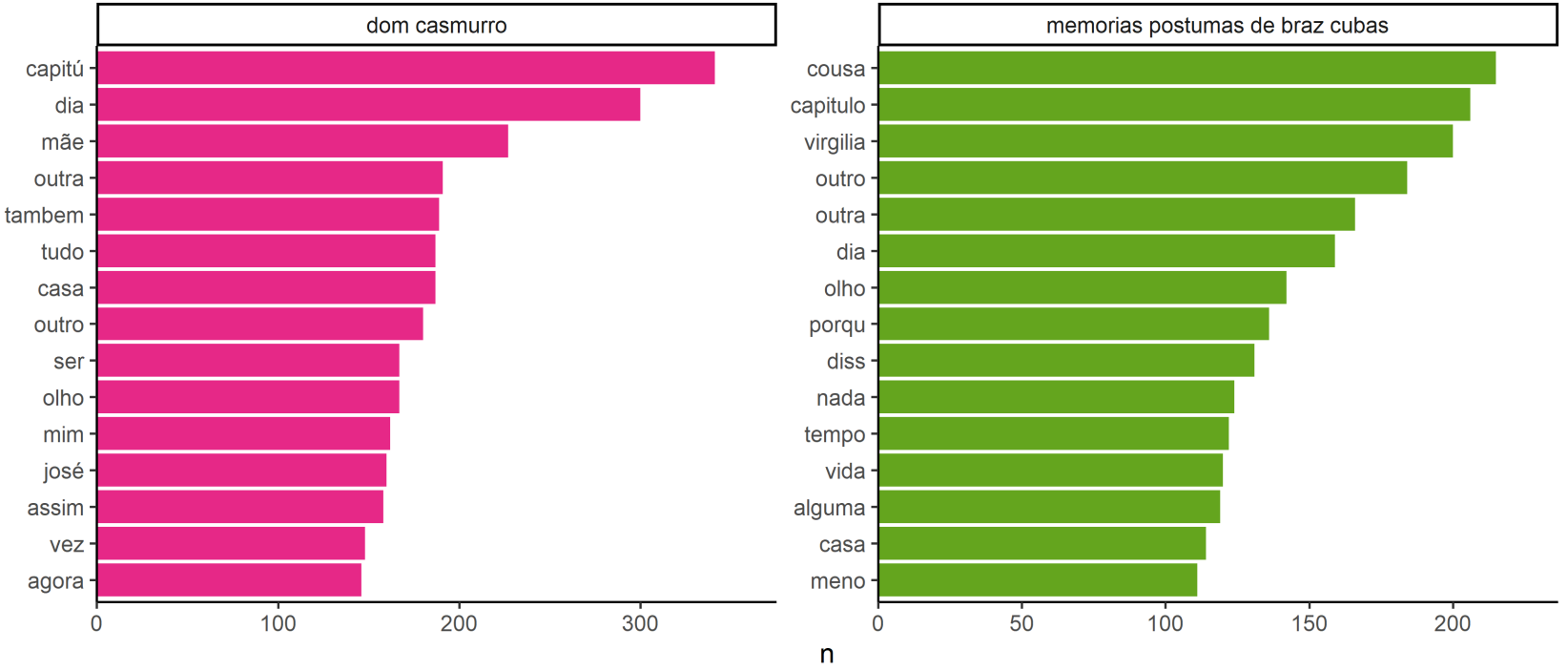
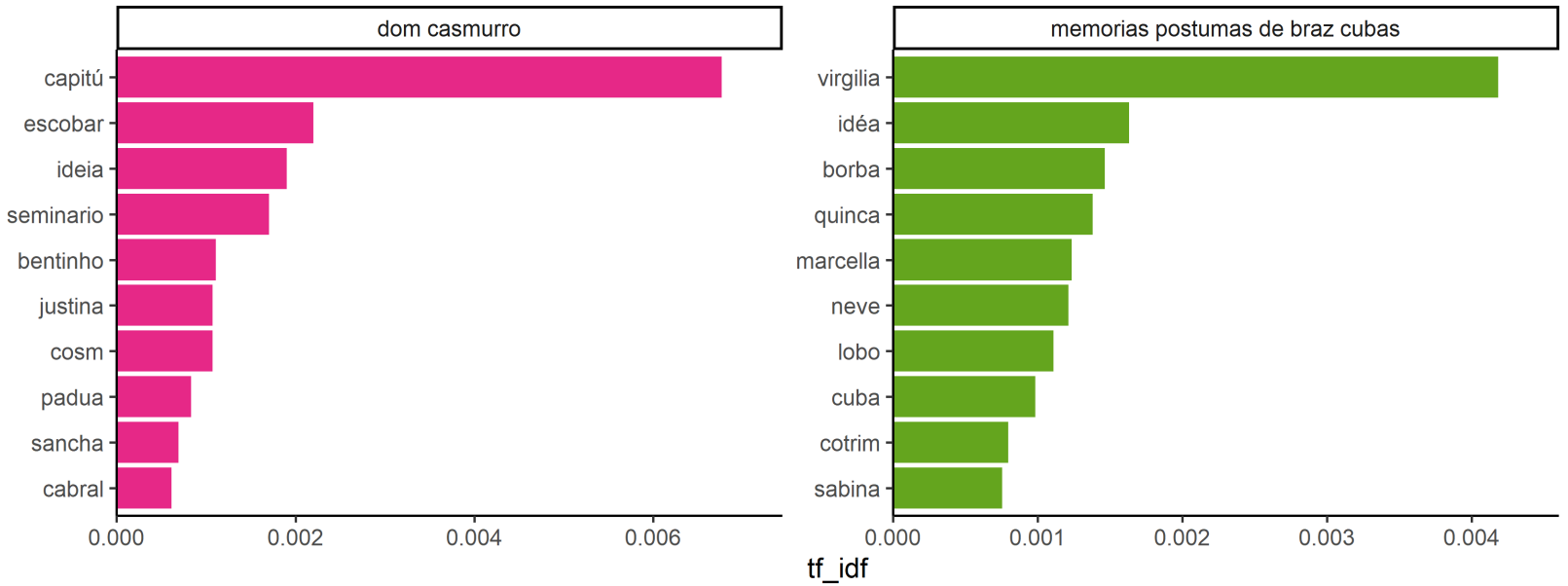


Gráfico *idf*



A estatística $tf\text{-}idf$ é utilizada para avaliar a importância do termo para um documento. Multiplica-se o fator abaixo pela frequência relativa do termo no documento em questão.

$$idf(\text{term}) = \ln\left(\frac{n_{\text{documents}}}{n_{\text{documents containing term}}}\right).$$

Latent Dirichlet allocation

Latent Dirichlet allocation (LDA) is one of the most common algorithms for topic modeling. Without diving into the math behind the model, we can understand it as being guided by two principles.

- **Every document is a mixture of topics**. We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B."

- **Every topic is a mixture of words**. For example, we could imagine a two-topic model of American news, with one topic for "politics" and one for "entertainment." The most common words in the politics topic might be "President", "Congress", and "government", while the entertainment topic may be made up of words such as "movies", "television", and "actor". Importantly, words can be shared between topics; a word like "budget" might appear in both equally.

LDA is a mathematical method for estimating both of these at the same time: finding the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document.

Retirado do livro Text Mining with R Silge, J. and Robinson, D. 2017.

Corpus da BBC

Neste exemplo trabalharemos com um *corpus* com notícias da BBC.

Os documentos estão classificados de acordo com os seguintes assuntos:

- business
- entertainment
- politics
- sport
- tech

Os assuntos serão desconsiderados para verificarmos se conseguimos recuperá-los usando o modelo LDA.

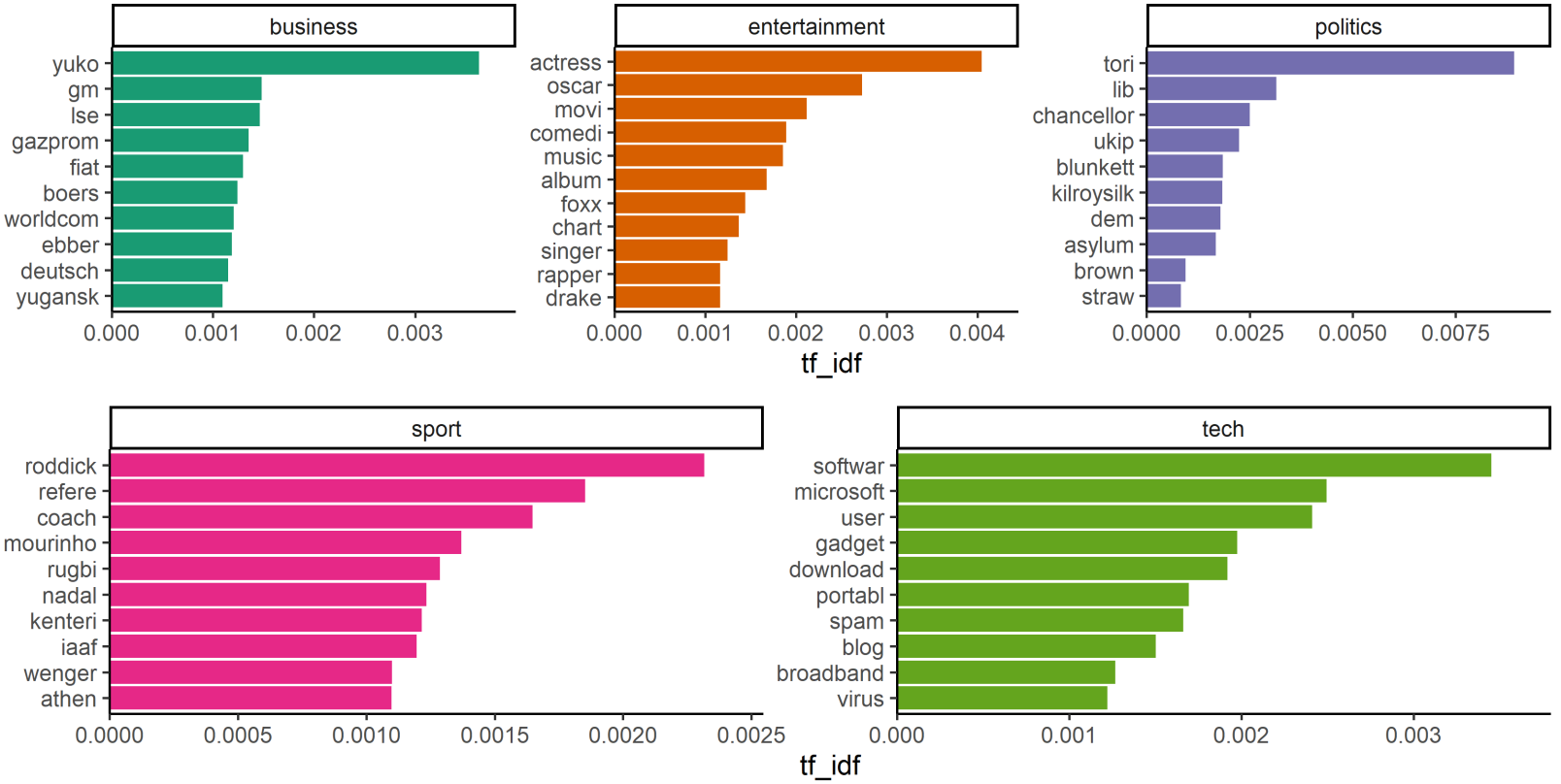
BBC

```
## <<DocumentTermMatrix (documents: 2225, terms: 9635)>>
## Non-/sparse entries: 286774/21151101
## Sparsity           : 99%
## Maximal term length: NA
## Weighting          : term frequency (tf)
```

Portanto, temos um total de 2.225 documentos com 9.635 termos e 99% de esparsidade.

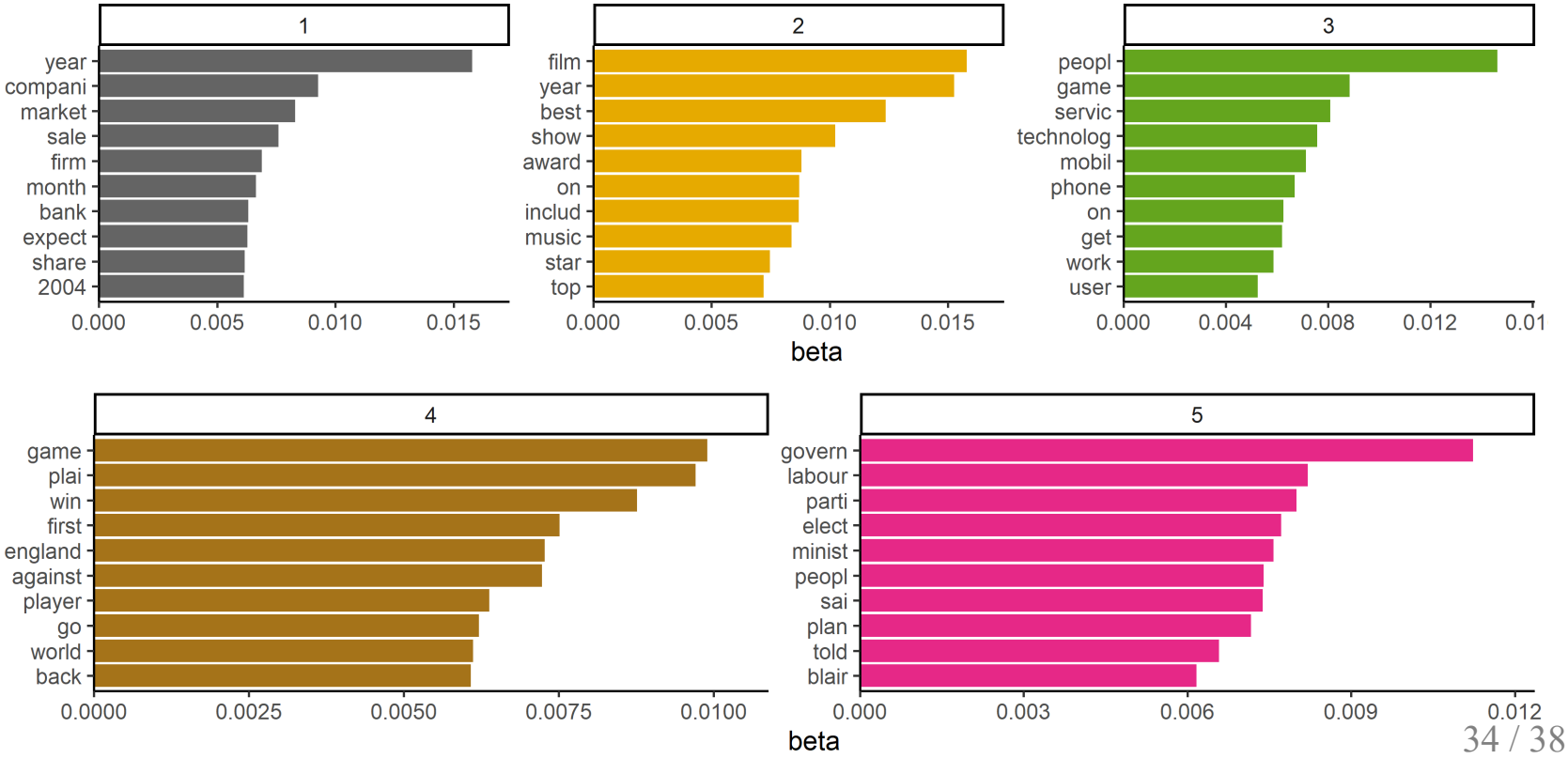
BBC - *tf-idf*

O gráfico abaixo mostra os termos com maior valor da estatística *tf_idf* para cada um dos tópicos considerados. Atenção: eixo x está em escala diferente em cada gráfico.



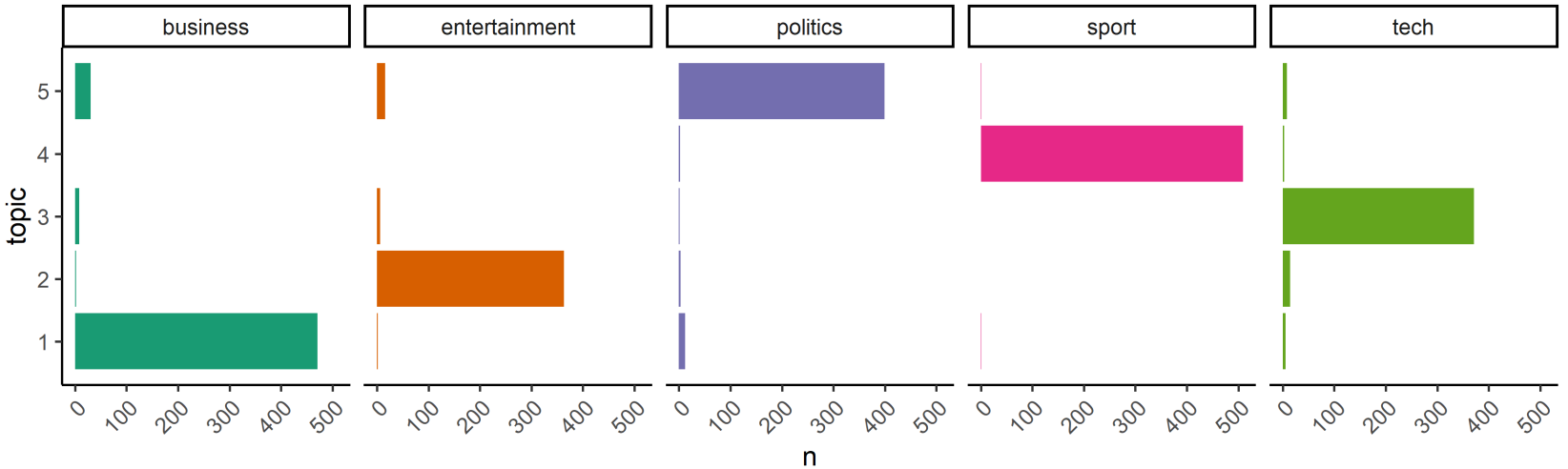
BBC - *word-topic probabilities*

Podemos analisar as probabilidades de cada palavra aparecer em cada tópico de acordo com o modelo LDA através da função `tidy(modelo, matrix = "beta")`. A partir dela, podemos obter o gráfico abaixo, que mostra as palavras mais frequentes em cada tópico. Atenção: eixo x está em escala diferente em cada gráfico.



BBC - *document-topic probabilities*

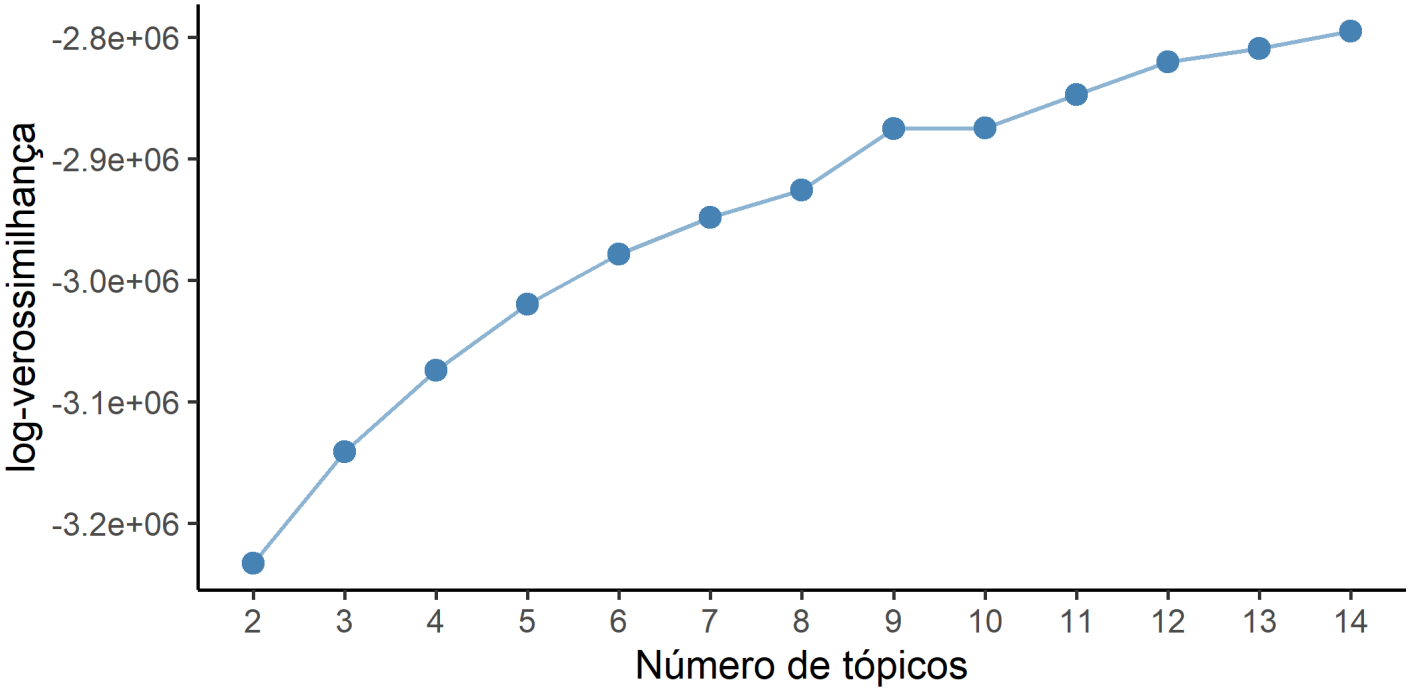
Podemos analisar, por documento, as probabilidades de pertencer a cada tópico com o modelo LDA através da função `tidy(modelo, matrix = "gamma")`. A partir dela, podemos obter o gráfico abaixo que nos ajuda a nomear os tópicos de 1 a 5, considerando os tópicos de fato observados.



Considerando essa abordagem, conseguimos agrupar corretamente 94.88% dos documentos.

BBC - número de tópicos

Uma pergunta natural quando não temos o número de tópicos especificado a priori é quantos tópicos devemos considerar (problema similar ao que temos ao considerar k-médias). Para ter uma direção, podemos utilizar a curva de cotovelo.



Resumindo

- Dados textuais são uma fonte valiosa de informações relevantes;
- Usar a abordagem *tidy* facilita a exploração e entendimento do conteúdo dos textos, usando ferramentas que estamos habituados (`tidyverse`).
- O modelo de tópicos LDA permite separar documentos em clusters, pois assume que cada documento é uma mistura de tópicos e cada tópico é uma mistura de palavras.

Obrigado!

magnotfs@insper.edu.br