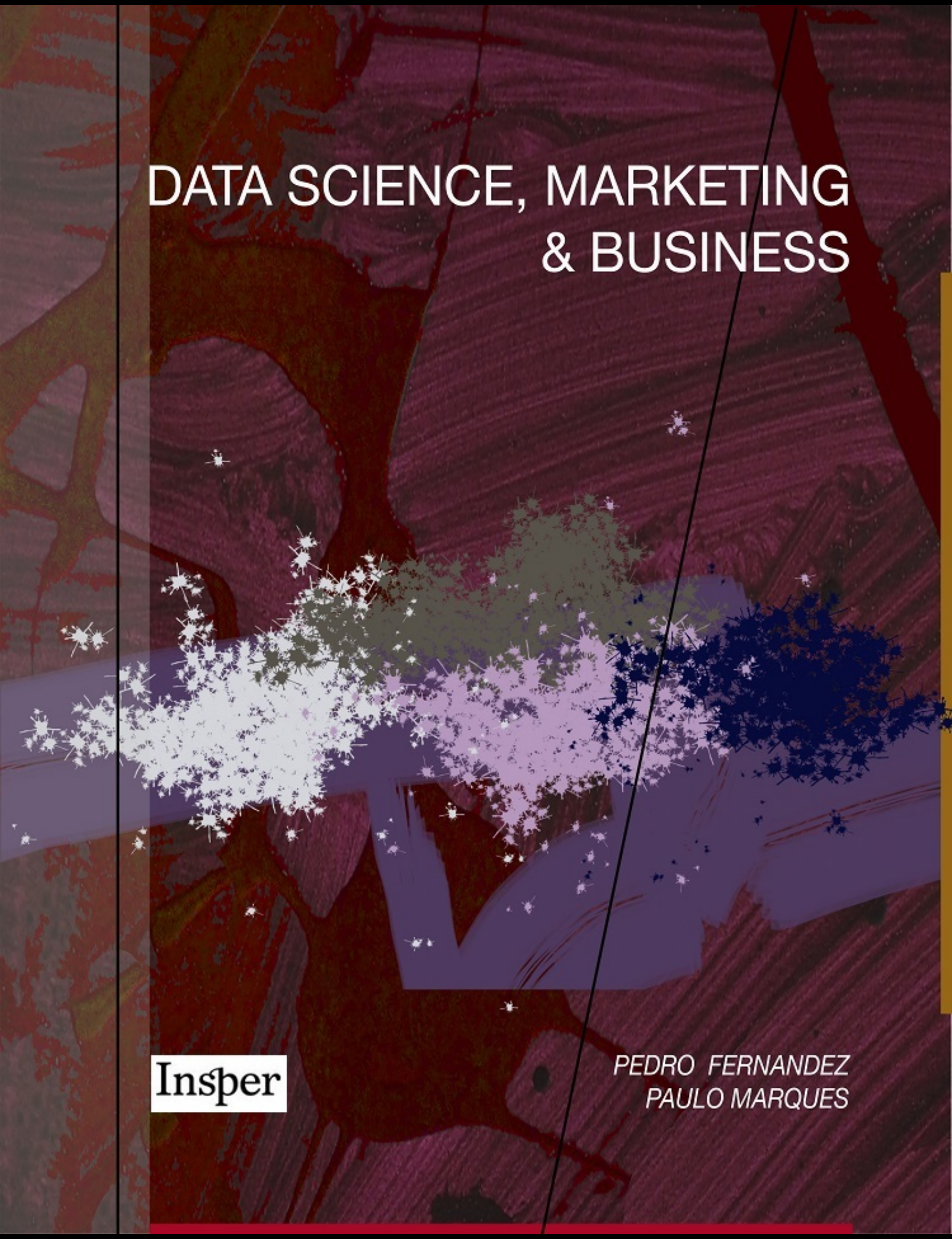
DATA SCIENCE, MARKETING & BUSINESS



Insper

PEDRO FERNANDEZ
PAULO MARQUES

Data Science, Marketing & Business

Pedro J. Fernandez
Paulo C. Marques F.

Versão 1.1

Conteúdo

Introdução

Este livro apresenta ao leitor uma série de problemas relacionados a dados mercadológicos. Para cada problema, começamos com uma discussão de natureza geral, que é subsequentemente formalizada. Cada questão discutida é exemplificada por um ou mais conjuntos de dados. O código computacional de todas as soluções é apresentado na íntegra.

Os processos de análise de dados vivem uma época de transformação e crise de identidade das disciplinas tradicionais. A demanda contemporânea é a integração total, sob o rótulo de *Data Science*, das técnicas e ferramentas originalmente desenvolvidas separadamente por disciplinas tais como Estatística, *Machine Learning* e Ciência da Computação. É um momento de renovação de currículos e integração de habilidades. O livro pretende traduzir o espírito da época, conjugando os aspectos matemáticos, inferenciais e computacionais envolvidos em questões de marketing e negócios.

O ambiente de código aberto criado pela linguagem `R` [15] ampara toda a nossa discussão. Fazemos um uso intenso de diversas bibliotecas específicas da linguagem, que formam um expressivo conjunto de ferramentas de *Data Science*. Não esperamos do leitor conhecimentos da linguagem maiores do que aqueles que podem ser adquiridos através de leituras preliminares de partes do livro de Wickham e Grolemund [20], disponível em `http://r4ds.had.co.nz/`.

Segue uma breve descrição do conteúdo dos capítulos do livro. No capítulo 1, apresentamos ferramentas e exemplos relacionados às questões de estrutura de mercado, nas quais procuramos entender geometricamente as relações de concorrência entre marcas e produtos. No capítulo 2, abordamos novamente a concorrência entre marcas, porém de um outro ponto de vista, adequado às situações em que temos informações mais detalhadas sobre as marcas e produtos, que vão além de uma simples medida de dissimilaridade.No capítulo 3, desenvolvemos técnicas para a descoberta de relações expressivas em bases de dados de transações de consumidores via regras de associação. No capítulo 4, exploramos os modelos de escolha discreta, que permitem representar e analisar as escolhas ao longo do tempo de um agente diante de um leque finito de bens ou serviços, envolvendo atributos tais como marcas, preços etc. No capítulo 5, discutimos a análise de dados textuais, introduzindo os conceitos básicos através da mineração de dados do *Twitter*. Em seguida, analisamos o modelo de tópicos, uma técnica que permite particionar um conjunto de documentos em *clusters* definidos por um modelo probabilístico baseado nas frequências de ocorrências das palavras em cada documento.

Diversos conjuntos de dados utilizados no livro foram gentilmente disponibilizados pela Ipsos. São dados reais, ligeiramente transformados, por questões de confidencialidade.

Agradecemos à Ipsos por essa contribuição vital ao projeto. Parcerias como essa, entre empresas e instituições de ensino e pesquisa, são sempre enriquecedoras e de grande valor social. Fica aqui o nosso agradecimento especial a Mariane Medina, diretora da Unidade Global de Modelagem da Ipsos, cujos esforços permitiram que os dados chegassem até nós em um formato adequado. Todos os dados utilizados nas análises estão disponíveis no site do livro em `http://datascience.insper.edu.br`.

Este projeto não teria sido possível sem a infraestrutura e o apoio do Insper - Instituto de Ensino e Pesquisa. Nossos agradecimentos especiais a Marcos Lisboa, Carolina da Costa e Paulo Furquim de Azevedo.

Agradecemos ao artista plástico Kakati de Paiva (`http://www.kakati.com.br/`) pela criação da capa do livro, que constitui a visão do artista de uma hipersuperfície de classificação.

Finalmente, Pedro J. Fernandez deixa seu agradecimento *in memorian* ao professor Elon Lages Lima do Instituto de Matemática Pura e Aplicada.

Sobre os autores

Pedro J. Fernandez

- **Formação Acadêmica**
  - Mestre em Economia e Matemática. PhD em Estatística Matemática.
- **Experiência Profissional**
  - Professor Assistente. Departamento de Estatística. UC Berkeley.
  - Professor Associado. IMPA. Rio de Janeiro.
  - CEO. Marketing Decision Systems. (1987-1989)
  - CEO. Novaction Marketing Consultants. (1989-1997)
  - CEO. Ipsos Brasil. (1997-2002)
  - Professor Associado. Escola de Negócios da FGV. (2002-2008)
  - Professor Visitante. Insper. (2016-2019)
- **Interesses Profissionais**
  - Métodos Quantitativos em Marketing e Pesquisa de Mercado.
  - Modelos de Resposta ao Marketing.
  - Previsão de Performance de Novos Produtos.
  - Métodos de Estatística, Econometria, *Machine Learning* e *Business Analytics* aplicados ao Marketing.

O autor tem vasta experiência criando e desenvolvendo empresas de sucesso no mercado de *Marketing Research*. É autor de diversos artigos e livros em suas áreas de interesse.

Paulo C. Marques F.

Bacharel em Física e Doutor em Estatística pela Universidade de São Paulo. Empreendedor na área de tecnologia e consultor, atualmente é Professor e Pesquisador do Insper, Instituto de Ensino e Pesquisa em São Paulo.

Tiago Mendonça dos Santos

Bacharel, Mestre e Doutorando em Estatística pela Universidade de São Paulo, atualmente é Professor do Insper, Instituto de Ensino e Pesquisa em São Paulo.

Hedibert F. Lopes

Bacharel e Mestre em Estatística pela Universidade Federal do Rio de Janeiro. PhD em Estatística pela Duke University. Foi Professor da Chicago Booth School of Business. Atualmente é Professor Titular de Estatística e Econometria do Insper, Instituto de Ensino e Pesquisa em São Paulo.

Capítulo 1

Concorrência

Pedro J. Fernandez, Tiago Mendonça dos Santos e Paulo C. Marques F.

Um dos principais problemas enfrentados por organizações que pretendem entrar no mercado de uma categoria de produtos é determinar a estrutura de concorrência entre os participantes do mercado. Na literatura, tais questões fazem parte de uma *Análise de Estrutura de Mercado* (*Market Structure Analysis*).

O primeiro passo neste tipo de problema é identificar as marcas/produtos presentes no mercado de interesse, além de informações tais como o preço ao consumidor e alguma medida de participação das marcas. Note-se que, muitas vezes, o mercado a ser considerado não fica definido explicitamente apenas pela natureza específica do produto. Por exemplo, ao analisar o mercado de refrigerantes, seria necessário considerarmos como concorrentes produtos tais como água mineral, chás e sucos de frutas? Essencialmente, o que está em jogo neste tipo de questão é a percepção dos consumidores, que podem ou não enxergar tais produtos como semelhantes entre si, ou seja, como sendo substituíveis no momento da compra.

Uma análise de estrutura de mercado envolve a obtenção de informações diretamente dos consumidores. Como veremos, tais questionamentos podem ser realizados de diversas maneiras. No entanto, as informações fornecidas pelos consumidores podem, em geral, ser resumidas pela construção de índices que medem a similaridade, ou a dissimilaridade, entre as marcas.

A técnica de *escalonamento multidimensional* (*multidimensional scaling* ou *MDS*) permite representar tais dissimilaridades entre os objetos do problema (produtos e marcas), cujos atributos, em geral, pertencem a um espaço de dimensão elevada, utilizando-se um espaço de dimensão menor, de maneira que tenhamos uma distorção mínima nas relações entre os objetos ao fazermos esta redução da dimensão original do problema. Este processo de redução dimensional permite visualizar e investigar mais facilmente as relações entre os objetos considerados.

Na próxima seção, discutimos a formulação matemática do escalonamento multidimensional. Em seguida, apresentamos três aplicações nas quais os dados são obtidos de maneiras distintas.

1.1 Escalonamento multidimensional

Dado um conjunto de objetos tais que seus atributos individuais são coordenadas de pontos de um espaço de dimensão possivelmente elevada, as técnicas de escalonamento multidimensional permitem representar os objetos como pontos de um novo espaço de dimensão relativamente pequena, tornando visualizável as dissimilaridades entre os objetos dos problemas.

O aspecto notável do escalonamento dimensional consiste no fato de que é suficiente conhecermos apenas as distâncias entre os objetos no espaço original, sem que as coordenadas de cada ponto sejam especificadas. De fato, veremos que basta conhecermos uma ordenação entre tais distâncias. Estas técnicas foram desenvolvidas fundamentalmente por Shepard [17, 18] e Kruskal [11, 12]. O livro de Borg e Groenen [3] é uma referência exaustiva sobre o tema.

Suponha que tenhamos um conjunto de n objetos (marcas ou produtos), cujos atributos $x_1, \ldots, x_n \in \mathbb{R}^p$ são conhecidos. Há diversas maneiras de definirmos a distância entre dois objetos. Por exemplo, a distância euclidiana usual, definida por

$$d_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$

Ou, quando cada atributo assume apenas os valores 0 e 1, podemos definir a distância de Jaccard [7, 8]

$$d_{ij} = 1 - \frac{\sum_{m=1}^{p} x_{im}x_{jm}}{\sum_{m=1}^{p} x_{im} + \sum_{m=1}^{p} x_{jm} - \sum_{m=1}^{p} x_{im}x_{jm}},$$

que será utilizada em nossa terceira aplicação. Para cada problema considerado, há uma definição de distância apropriada, que deve ser interpretada adequadamente.

Matematicamente, qualquer distância d_{ij} considerada entre os objetos do problema deve satisfazer as propriedades

1. $d_{ij} \geq 0$;
2. $d_{ij} = 0$ se e somente se $i = j$;
3. $d_{ij} = d_{ji}$;
4. $d_{ij} \leq d_{ik} + d_{kj}$,

para $i, j, k = 1, \ldots, n$. Uma *medida de dissimilaridade* é uma "distância enfraquecida", que não satisfaz as quatro propriedades acima.

Um primeiro exemplo de construção de uma medida de dissimilaridade seria solicitarmos a um grupo de indivíduos "notas" em uma escala de 0 a 10 que meçam a dissimilaridade entre os objetos do problema (quanto maior a nota, maior a dissimilaridade). Neste caso, as dissimilaridades d_{ij} seriam definidas pela média das notas dadas por todos os indivíduos. Em nossa primeira aplicação, as dissimilaridades são definidas desta maneira.

Um segundo exemplo de medida de dissimilaridade consiste em solicitarmos a cada um de n indivíduos que agrupe os objetos do problema de maneira arbitrária, de modo que objetos de um mesmo grupo sejam considerados similares. Neste caso, as dissimilaridades seriam definidas por

$$d_{ij} = \frac{n - (\text{quantas vezes } i \text{ e } j \text{ foram alocados no mesmo grupo})}{n}.$$

Conhecendo-se apenas as distâncias, ou apenas as dissimilaridades entre os objetos do problema, o escalonamento multidimensional determina pontos $z_1, \ldots, z_n \in \mathbb{R}^k$, em que k é menor do que p, de maneira a minimizar a função objetivo

$$S(z_1, \ldots, z_n) = \sum_{i,j=1}^{n} \left(d_{ij} - \sqrt{\sum_{m=1}^{k} (z_{im} - z_{jm})^2} \right)^2 .$$

Deste modo, os objetos passam a ser representados em um espaço de dimensão reduzida, mantendo-se de maneira aproximada a informação original sobre as distâncias ou dissimilaridades entre os objetos do problema. Neste procedimento de redução dimensional, conforme veremos em nossas aplicações, a escolha da dimensão k envolve a análise dos autovalores da matriz $D = (d_{ij})$. Detalhes matemáticos podem ser encontrados no capítulo 4 do livro de Fernandez e Yohai [5].

Nas próximas seções, apresentamos três aplicações, sendo que as duas primeiras utilizam a técnica de escalonamento multidimensional. É importante notar que os dados foram coletados de maneira distinta em cada aplicação, o que ilustra a questão de como podemos definir dissimilaridades e distâncias entre os objetos do nosso problema. Na primeira aplicação, os entrevistados indicam diretamente, numa escala de 10 pontos, a dissimilaridade entre países. Já na segunda aplicação, os indivíduos apenas agrupam aqueles carros que considerem semelhantes, e a partir destes agrupamentos é definida a medida de dissimilaridade. Finalmente, na terceira aplicação, os indivíduos indicam se consomem ou não regularmente certas marcas de uísques escoceses, e a partir desta informação calculamos a distância de Jaccard entre os produtos. Não necessitamos do escalonamento multidimensional nesta terceira aplicação, pois já partimos de uma distância bem definida entre as marcas de uísque.

1.2 Comparando países

Neste exemplo, consideramos os dados de questionários aplicados a estudantes de Ciências Políticas sobre as dissimilaridades percebidas entre 12 países: Bélgica - BEL, Brasil - BRA, China - CHI, Cuba - CUB, Egito - EGY, França - FRA, Índia - IND, Israel - ISR, Estados Unidos da América - USA, União das Repúblicas Socialistas Soviéticas - USS, Iugoslávia - YUG e Zaire - ZAI.

Nesta pesquisa, cada entrevistado indicava, em uma escala de dez pontos, a dissimilaridade entre cada par de países. Após a coleta, os resultados foram agregados e considerou-se a dissimilaridade como sendo a nota média de cada par. Os dados deste estudo fazem parte da biblioteca `ElemStatLearn`. Começamos carregando as bibliotecas necessárias e lendo os dados.

```
library(tidyverse)
library(ggrepel)
library(factoextra)

countries <- read.csv("dados/concorrencia/countries.csv", row.names = 1)
```

	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG	ZAI
BEL	0.00	5.58	7.00	7.08	4.83	2.17	6.42	3.42	2.50	6.08	5.25	4.75
BRA	5.58	0.00	6.50	7.00	5.08	5.75	5.00	5.50	4.92	6.67	6.83	3.00
CHI	7.00	6.50	0.00	3.83	8.17	6.67	5.58	6.42	6.25	4.25	4.50	6.08
CUB	7.08	7.00	3.83	0.00	5.83	6.92	6.00	6.42	7.33	2.67	3.75	6.67
EGY	4.83	5.08	8.17	5.83	0.00	4.92	4.67	5.00	4.50	6.00	5.75	5.00
FRA	2.17	5.75	6.67	6.92	4.92	0.00	6.42	3.92	2.25	6.17	5.42	5.58
IND	6.42	5.00	5.58	6.00	4.67	6.42	0.00	6.17	6.33	6.17	6.08	4.83
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17	0.00	2.75	6.92	5.83	6.17
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75	0.00	6.17	6.67	5.67
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17	0.00	3.67	6.50
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	0.00	6.92
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92	0.00

Tabela 1.1: Dissimilaridades entre os países.

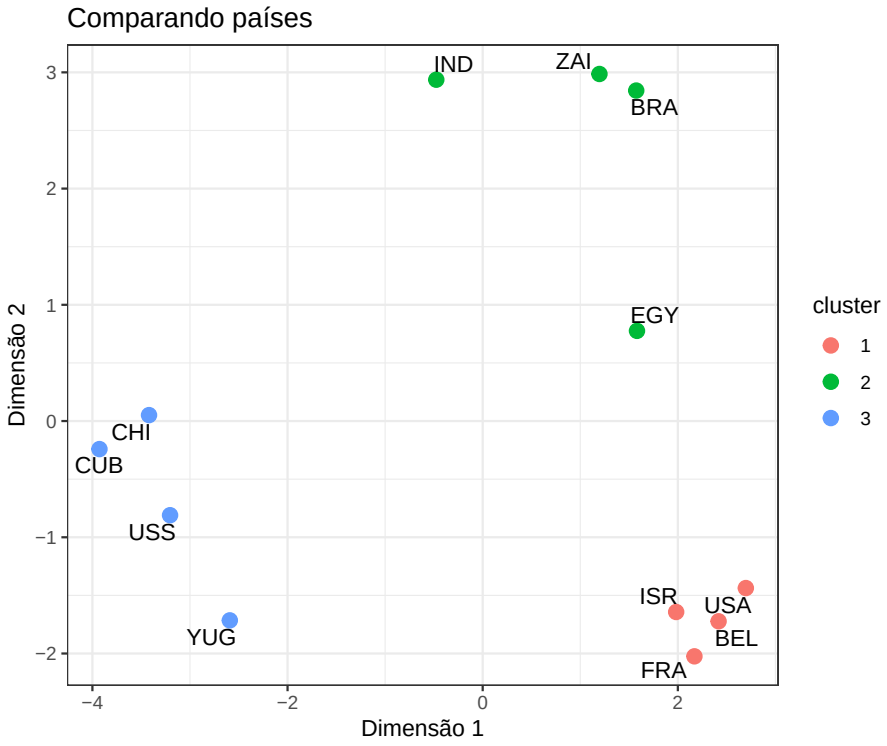
O escalonamento multidimensional destes dados é obtido através da função `cmdscale()`. Para interpretarmos a representação em dimensão reduzida dos dados gerada pelo escalonamento multidimensional, precisamos examinar os autovalores obtidos (veja [5]).

```
mds <- cmdscale(countries, k = nrow(countries) - 1, eig = TRUE)

mds$eig

##  [1]  7.249546e+01  4.171421e+01  2.582771e+01  1.560425e+01  1.116682e+01
##  [6]  9.310405e+00  6.025739e+00  2.956734e+00  1.154632e-14 -4.544210e-01
## [11] -2.784551e+00 -7.425256e+00

cumsum(mds$eig) / sum(mds$eig)

##  [1] 0.4155966 0.6547327 0.8027959 0.8922507 0.9562670 1.0096410 1.0441849
##  [8] 1.0611351 1.0611351 1.0585300 1.0425670 1.0000000
```

Note que temos 9 autovalores positivos e 3 autovalores negativos, o que indica que a matriz de dissimilaridades não define uma distância. No entanto, a soma dos autovalores positivos é muito maior do que a soma dos autovalores negativos. Ademais, note que apenas os dois primeiros autovalores representam aproximadamente 65% da soma total. Isto indica que um gráfico em duas dimensões pode representar a dissimilaridade dos dados de maneira satisfatória (veja [13]).

Tendo obtido as coordenadas dos objetos via escalonamento multidimensional, podemos agrupá-los em *clusters* com qualquer técnica de conglomeração. Neste exemplo, utilizamos o algoritmo k-médias [9, 6] para produzirmos três *clusters* de países.

```
df <- data.frame(country = rownames(mds$points),
                 z_1 = mds$points[, 1],
                 z_2 = mds$points[, 2])

df$cluster <- factor(kmeans(df[, 2:3], centers = 3, nstart = 100)$cluster)

df %>%
    ggplot(aes(x = z_1, y = z_2, label = country)) +
        geom_point(aes(color = cluster), size = 3) +
        labs(x = "Dimensão 1", y = "Dimensão 2", title = "Comparando países") +
        geom_text_repel()
```



Os *clusters* na figura acima agrupam os países de acordo a percepção dos estudantes entrevistados, de tal modo que países em um mesmo *cluster* são considerados semelhantes no que diz respeito às suas características gerais, sócio-econômicas etc.

1.3 Mercado automobilístico

Nesta aplicação, cada um de $n = 150$ consumidores recebeu 30 cartões com modelos de carros, e lhes foi solicitado que agrupassem os cartões de modo que dois carros alocados em um mesmo grupo fossem considerados equivalentes (substituíveis no momento da compra). Conforme discutido anteriormente, este processo define uma medida de dissimilaridade por

$$d_{ij} = \frac{n - (\text{quantas vezes } i \text{ e } j \text{ foram alocados no mesmo grupo})}{n}.$$

Assim, d_{ij} é igual a 0 quando os modelos i e j são alocados dentro do mesmo grupo por todos os consumidores, enquanto d_{ij} é igual a 1 se i e j não forem alocados dentro do mesmo grupo por nenhum consumidor. Começamos pela leitura da matriz de dissimilaridades gerada ao final das entrevistas. A tabela apresenta as dissimilaridades entre 10 dos modelos de carros envolvidos na pesquisa.

```
D <- read.csv("dados/concorrencia/cars.csv", row.names = 1)
```

	UNO	KA	VECT	CELT1	GOL	GOLF	CORSS	GOL1	ECO	BLA2
UNO	0.00	0.39	0.88	0.46	0.53	0.74	0.75	0.44	0.88	0.91
KA	0.39	0.00	0.90	0.38	0.53	0.73	0.74	0.44	0.92	0.94
VECT	0.88	0.90	0.00	0.87	0.83	0.75	0.56	0.86	0.90	0.85
CELT1	0.46	0.38	0.87	0.00	0.44	0.66	0.71	0.35	0.90	0.93
GOL	0.53	0.53	0.83	0.44	0.00	0.57	0.73	0.28	0.91	0.91
GOLF	0.74	0.73	0.75	0.66	0.57	0.00	0.71	0.56	0.89	0.87
CORSS	0.75	0.74	0.56	0.71	0.73	0.71	0.00	0.69	0.93	0.91
GOL1	0.44	0.44	0.86	0.35	0.28	0.56	0.69	0.00	0.91	0.91
ECO	0.88	0.92	0.90	0.90	0.91	0.89	0.93	0.91	0.00	0.34
BLA2	0.91	0.94	0.85	0.93	0.91	0.87	0.91	0.91	0.34	0.00

Tabela 1.2: Dissimilaridades entre os carros.

Fazendo o escalonamento muldimensional dos dados, observamos que um dos autovalores é negativo (pois $D = (d_{ij})$ não é uma matriz de distâncias), enquanto os dois primeiros autovalores têm magnitude muito maior do que os demais. Portanto, há novamente uma indicação de que os dados podem ser bem representados bidimensionalmente.

```
mds <- cmdscale(D, k = nrow(D) - 1, eig = TRUE)

mds$eig
```

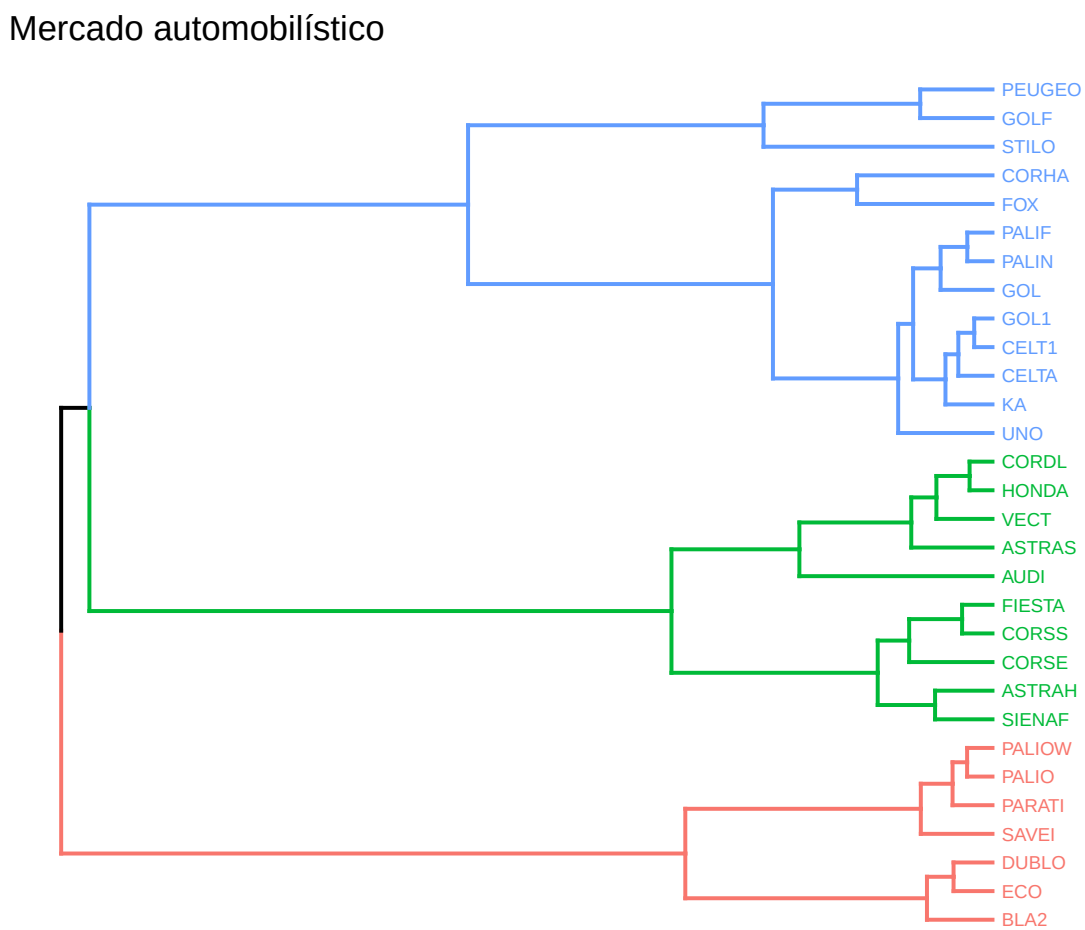
```
##  [1]  2.194767e+00  1.745268e+00  7.521428e-01  6.795032e-01  3.992741e-01
##  [6]  2.987542e-01  2.702497e-01  2.475952e-01  1.966893e-01  1.853476e-01
## [11]  1.721662e-01  1.519231e-01  1.388504e-01  1.252190e-01  1.106550e-01
## [16]  9.453951e-02  8.563329e-02  7.705235e-02  6.927064e-02  6.220930e-02
## [21]  5.623030e-02  3.805394e-02  3.452582e-02  2.937967e-02  2.107618e-02
## [26]  1.628979e-02  9.665003e-03  1.808790e-03  2.393918e-16 -1.046321e-02
```

Vale a pena recapitular o processo de análise de dados feito até o momento neste exemplo. Cada carro/modelo é um objeto complexo, cujos atributos poderiam ser representados em um espaço de dimensão elevada. No entanto, não temos acesso às coordenadas de cada carro neste espaço hipotético. Tudo o que temos é a informação sobre quão dissimilares são cada um dos pares de carros participantes na pesquisa, do ponto de vista dos consumidores entrevistados. O escalonamento multidimensional, utilizando apenas as informações sobre tais dissimilaridades, atribui a cada carro um ponto em um espaço

de duas dimensões, de maneira que carro mais próximos neste espaço sejam considerados mais semelhantes/substituíveis pelos consumidores entrevistados.

De posse das coordenadas bidimensionais de cada carro produzidas pelo escalonamento multidimensional, podemos utilizar qualquer técnica de conglomeração para agrupar os carros em *clusters*. Variando o procedimento utilizado no exemplo anterior, aqui optamos por uma técnica de conglomeração hierárquica (veja [9, 6]), cortando o dendrograma obtido de modo a formarmos três *clusters* de carros.

```
hc <- hclust(dist(mds$points[, 1:2]))

fviz_dend(hc, cex = 0.5, k = 3,
          main = "Mercado automobilístico",
          color_labels_by_k = TRUE, horiz = TRUE) + theme_void()
```



Neste tipo de problema a escolha do número apropriado de *clusters* deve ser feita examinando as configurações obtidas. Os dois extremos não são informativos: a opção por muitos *clusters* deixa poucos carros em cada um deles, insinuando que cada carro só

é semelhante a si mesmo, enquanto a opção por poucos *clusters* homegeneiza demais os carros.

Uma vez obtidos os *clusters*, o uso dos resultados é imediato. Por exemplo, parece razoável que o fabricante de um modelo específico acompanhe de perto o desenvolvimento de carros que pertencem ao próprio *cluster* no que diz respeito a preços, promoções, acessórios, serviços etc.

1.4 Uísques escoceses

Neste exemplo, consideramos dados de uma pesquisa sobre hábitos de consumo de uísque de 2.218 consumidores regulares. Os consumidores foram solicitados a indicar quais uísques, de uma lista de 21 marcas, consomem regularmente. A resposta de cada consumidor é representada por um vetor de 0's e 1's de comprimento 21. Nesta codificação, 1 indica que a marca é consumida regularmente, enquanto 0 indica a marca não é consumida regularmente. Estes dados fazem parte da biblioteca `bayesm`.

```
X <- as.matrix(read.csv("dados/concorrencia/whisky.csv"))

n <- nrow(X) # indivíduos
m <- ncol(X) # marcas
```

A tabela abaixo apresenta uma fatia dos dados, mostrando os hábitos de consumo de cinco consumidores para quatro das marcas de uísque escocês pesquisadas.

	Chivas.Regal	Dewar.s.White.Label	Johnnie.Walker.Black.Label	J.B
1	1	0	0	0
2	0	0	1	0
3	0	0	0	0
4	1	0	1	0
5	1	0	1	0

Tabela 1.3: Consumidoes e marcas de uísque escocês.

Assim, o primeiro entrevistado consome habitualmente a marca `Chivas.Regal`, o segundo consome habitualmente a marca `Johny.Walker.Black.Label` etc. Para este tipo de dado, utilizamos a distância de Jaccard [7, 8] definida anteriormente para comparar as diversas marcas de uísque. A intuição por detrás da definição da distância de Jaccard é apenas que marcas que possuem muitos consumidores em comum são mais parecidas, e vice-versa.

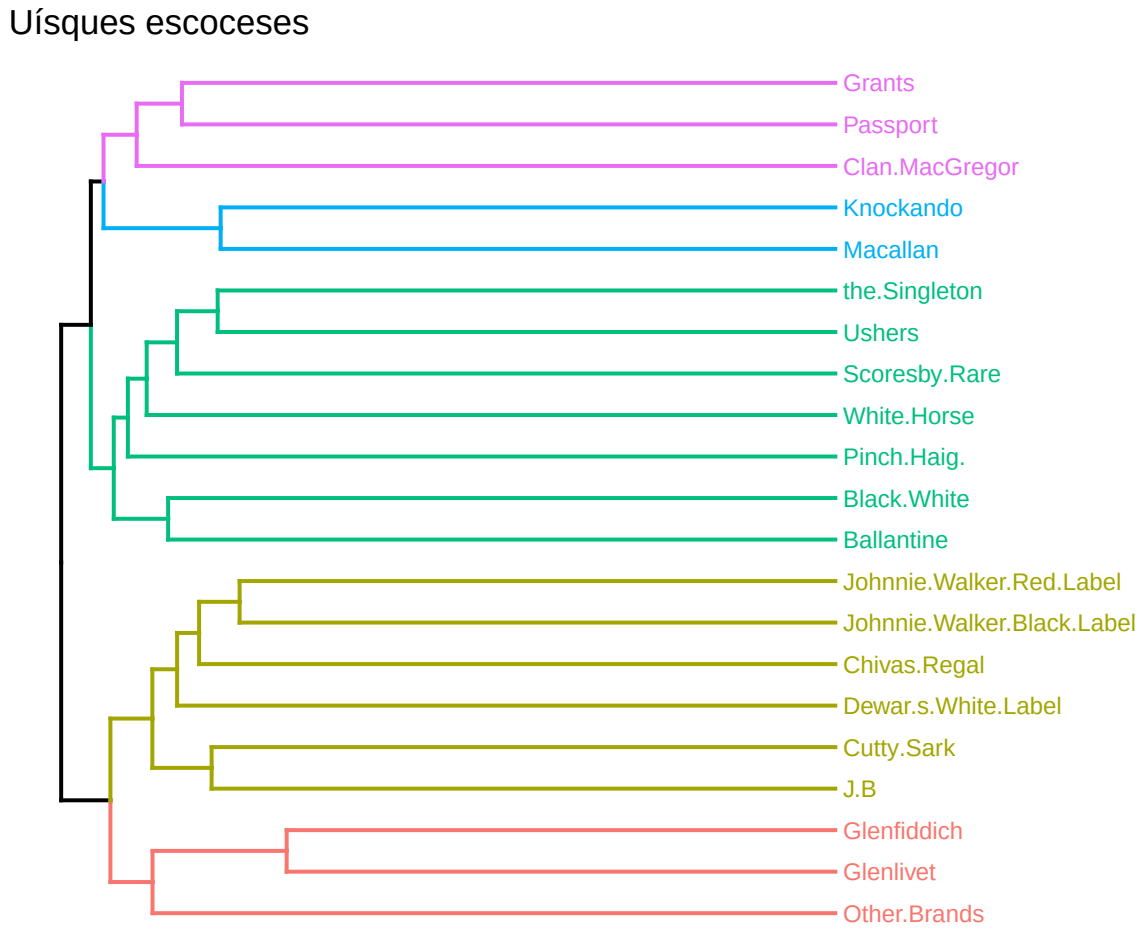
```
D <- matrix(0, nrow = m, ncol = m)

dimnames(D) <- list(colnames(X), colnames(X))

for (j in 1:m) {
    for (k in 1:j) {
        if (j != k) {
            D[j, k] <- 1 - sum(X[, j] * X[, k]) /
                          (sum(X[, j]) + sum(X[, k]) - sum(X[, j] * X[, k]))
            D[k, j] <- D[j, k]
        }
    }
}
```

Abaixo temos uma fatia da matriz de distâncias de Jaccard obtida, mostrando as relações entre três das marcas de uísque escocês pesquisadas.

```
brands <- c("Chivas.Regal", "Dewar.s.White.Label", "Passport")

D[brands, brands]

##                     Chivas.Regal Dewar.s.White.Label  Passport
## Chivas.Regal           0.0000000           0.8394737 0.9601874
## Dewar.s.White.Label    0.8394737           0.0000000 0.9527972
## Passport               0.9601874           0.9527972 0.0000000
```

Neste exemplo, é possível utilizar o escalonamento multidimensional para construir uma representação, digamos, bidimensional das marcas de uísque escocês. No entanto, se nosso propósito for apenas determinar a estrutura de concorrência entre as marcas, podemos utilizar uma técnica de conglomerados trabalhando diretamente com a matriz de distâncias de Jaccard. Abaixo temos o dendrograma obtido com o algoritmo de conglomeração hierárquica cortado para que tenhamos cinco *clusters*.

```
hc <- hclust(as.dist(D))

fviz_dend(hc, cex = 0.6, k = 5,
          main = "Uísques escoceses", labels_track_height = 0.35,
          color_labels_by_k = TRUE, horiz = TRUE) + theme_void()
```



Como em nosso exemplo anterior envolvendo automóveis, os *clusters* obtidos mapeiam diretamente a concorrência de uma determinada marca de uísque. Há diversos usos para a configuração de *clusters* obtida. Por exemplo, é possível usar a informação dos *clusters* para decidir como posicionar os uísques nas prateleiras de um ponto de venda. Uma outra possibilidade seria construir um sistema de recomendação simples, que sugerisse ao consumidor marcas do mesmo *cluster* de uma marca já adquirida. Finalmente, em uma situação na qual o ponto de venda não possui recursos para estocar todas as marcas, seria possível se concentrar em uma ou duas marcas de cada *cluster*, e ainda assim ter uma oferta representativa das marcas em questão.

Capítulo 2

Posicionamento

Pedro J. Fernandez, Tiago Mendonça dos Santos e Paulo C. Marques F.

Neste capítulo abordamos a concorrência entre as marcas/produtos de um determinado mercado, levando em consideração diversos atributos. Em contraste com o capítulo anterior, no qual contávamos apenas com uma medida da dissimilaridade entre as marcas, agora possuímos informações mais detalhadas sobre cada marca considerada.

Os estudos de *imagem e posicionamento de marcas* são uma parte importante do trabalho em Marketing. Nestes estudos, a categoria de produtos já se encontra definida e temos disponíveis informações genéricas, tais como *shares*, preços etc. Uma lista de atributos destas marcas é construída com o auxílio de: estudos qualitativos prévios, opiniões do departamento de marketing da empresa, informações da agência de publicidade envolvida no processo etc. Esta lista é composta de maneira a conter todos os atributos potenciais que influenciem a preferência entre as marcas, afetando as decisões de compras dos consumidores. Nesta etapa, é importante eliminar duplicações e atributos nitidamente irrelevantes.

Após definirmos as marcas e quais atributos serão considerados, os consumidores fazem a avaliação destas em relação a todos os atributos listados. Usualmente, esta avaliação é feita através de uma escala de Likert, de cinco ou sete pontos, sendo que as avaliações de cada indivíduo são feitas em um conjunto de marcas sobre as quais o indivíduo tenha uma opinião formada. Normalmente, são marcas que o indivíduo já comprou ou consumiu, bem como marcas que pretende comprar em um futuro próximo.

Em geral, é difícil trabalhar diretamente com os atributos originais das marcas, devido à sua profusão e ao fato de que tais atributos podem ser fortemente correlacionados. Um caminho natural consiste em reduzir a "dimensão" do problema, dada pelo número de atributos investigados, e passar a descrever os produtos em termos das *componentes principais* mais relevantes. No jargão do Marketing, tais componentes principais são denominadas *drivers* do problema.

Após obtermos estes *drivers*, podemos separar os resultados por marcas e calcular as médias dos *drivers* para cada marca. Os valores médios das marcas sobre os *drivers* são denominados *posicionamentos das marcas*. É conveniente, para auxiliar na interpretação, visualizar os resultados com o auxílio de gráficos de linha de todas as marcas sobre todos os *drivers*.

A técnica estatística (ou de aprendizagem não supervisionada) que utilizamos neste capítulo é a Análise de Componentes Principais (*Principal Components Analysis* ou *PCA*). Uma discussão formal da PCA, com detalhes e demonstrações dos resultados teóricos, é apresentada no livro de Fernandez e Yohai [5]. Na próxima seção, veremos a PCA em ação no estudo do posicionamento de marcas de alvejantes.

2.1 Mercado de alvejantes

Nesta aplicação, 750 consumidores de alvejantes avaliaram um conjunto reduzido de marcas respondendo sobre preferências e *ratings* de 29 atributos de imagem. Das 19 marcas consideradas, cada respondente avaliou no máximo 4 marcas. No total, há 1.661 avaliações de alvejantes na base de dados.

A base de dados está organizada em duas tabelas. Na primeira tabela, temos as avaliações de cada respondente, de tal modo que cada linha da tabela corresponde à avaliação de uma das marcas feita por um dos respondentes.

```
library(tidyverse)
library(factoextra)
library(ggrepel)

avaliacoes <- read_csv("dados/posicionamento/avaliacoes.csv")
```

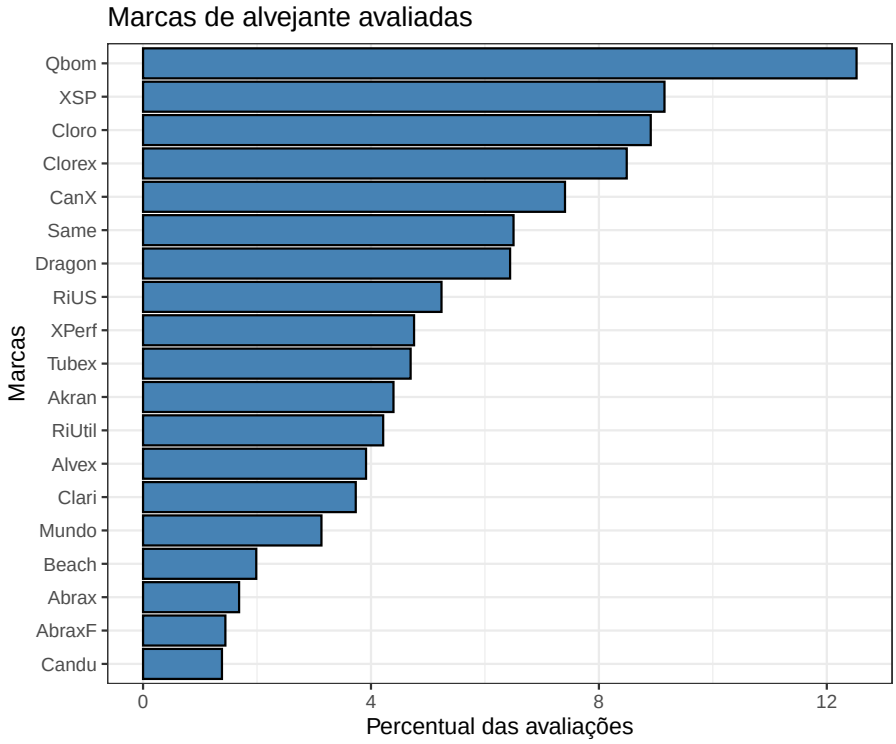
Como exemplo, temos abaixo as cinco avaliações de dois respondentes (identificados por uma chave única) para doze das questões apresentadas. Para cada questão, o respondente deu uma nota em uma escala de Likert de 0 a 7 pontos.

respondente	marca	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
1	RiUS	7	6	7	6	7	6	7	7	6	7	7	7
1	CanX	5	1	6	5	5	5	6	6	5	7	7	7
1	Candu	5	1	4	5	4	6	1	5	4	5	5	5
2	AbraxF	6	5	5	5	6	6	5	5	6	6	6	6
2	Mundo	5	5	6	3	3	5	5	5	4	4	4	5

Tabela 2.1: Avaliações dos alvejantes.

A figura seguinte apresenta os percentuais das avaliações que incluíram cada uma das marcas pesquisadas. Nos extremos, vemos que marca “Qbom” foi a mais avaliada pelos respondentes, enquanto a marca “Candu” apareceu no menor número de avaliações.

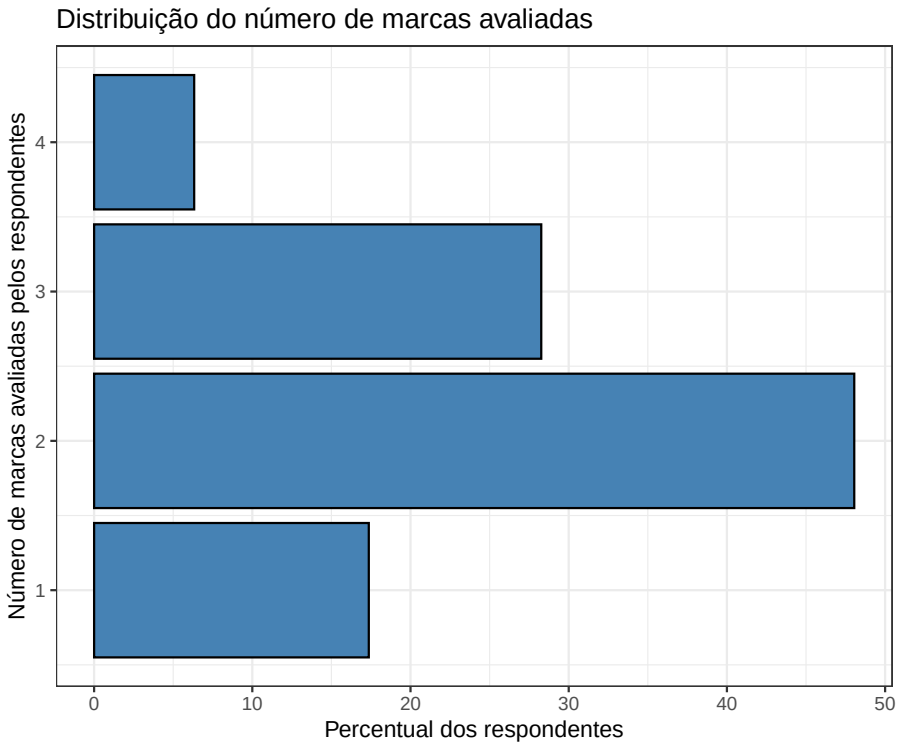
```
avaliacoes %>%
    count(marca) %>%
    mutate(percentual = 100 * prop.table(n)) %>%
    arrange(desc(percentual)) %>%
    ggplot(aes(x = reorder(marca, n), y = percentual)) +
        geom_bar(stat = "identity", fill = "steelblue", color = "black") +
        labs(x = "Marcas", y = "Percentual das avaliações",
             title = "Marcas de alvejante avaliadas") +
        coord_flip()
```



Queremos que o respondente avalie apenas aquelas marcas com as quais esteja familiarizado. O seguinte gráfico de barras apresenta a distribuição do número de marcas avaliadas pelos respondentes.

```
avaliacoes %>%
    group_by(respondente) %>%
    summarise(num_marcas_avaliadas = n()) %>%
    count(num_marcas_avaliadas) %>%
    mutate(percentual = 100 * prop.table(n)) %>%
    ggplot(aes(x = num_marcas_avaliadas, y = percentual)) +
        geom_bar(stat = "identity", fill = "steelblue", color = "black") +
        labs(x = "Número de marcas avaliadas pelos respondentes",
```

```
            y = "Percentual dos respondentes",
            title = "Distribuição do número de marcas avaliadas") +
        coord_flip()
```



A segunda tabela da base de dados contém as questões apresentadas aos respondentes. Como exemplo, vejamos o texto das dez primeiras questões.

```
(questoes <- read_csv("dados/posicionamento/questoes.csv"))
```

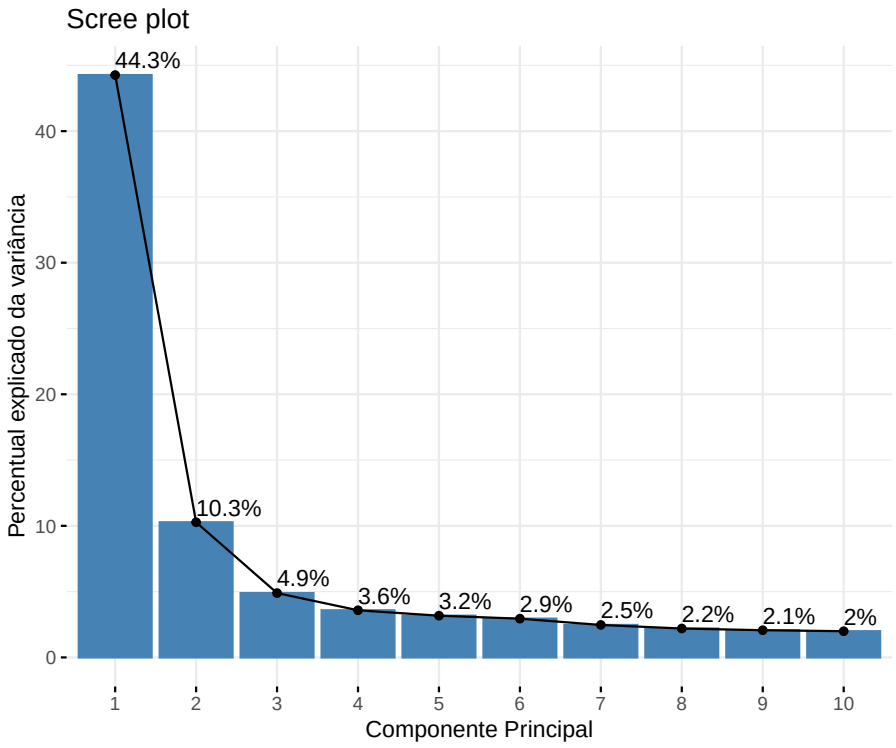
```
## # A tibble: 29 x 2
##    numero pergunta
##    <chr>  <chr>
##  1 Q1     Deixa as roupas mais brancas
##  2 Q2     É adequada para roupas coloridas
##  3 Q3     É a melhor para a remoção de manchas de gordura
##  4 Q4     Desinfeta melhor
##  5 Q5     É boa para limpar gordura
##  6 Q6     Deixa um aroma agradável na casa
##  7 Q7     É suave para as mãos
##  8 Q8     É adequada para a limpeza pesada
##  9 Q9     Deixa um aroma agradável nas roupas
## 10 Q10    É um produto para ser usado tanto na cozinha como no banheiro
## # ... with 19 more rows
```

Ao fazermos a PCA destes dados, cada componente principal obtida será uma combinação linear das notas dadas às questões originalmente apresentadas. Note que as colunas foram padronizadas.

```
pca <- avaliacoes %>%
    select(starts_with("Q")) %>%
    prcomp(scale = TRUE)
```

Um *scree plot*, ou "gráfico de cotovelo", indica que as três primeiras componentes já explicam 59% da variabilidade dos dados.

```
fviz_eig(pca, addlabels = TRUE) +  # scree plot
    labs(x = "Componente Principal", y = "Percentual explicado da variância")
```




```
(cumsum(pca$sdev^2) / sum(pca$sdev^2))[1:3]

## [1] 0.44 0.55 0.59
```

O exame dos autovalores corrobora a ideia de que os dados podem ser bem resumidos por três componentes principais. O critério de Kaiser [10] diz que um autovalor maior do que 1 indica que a respectiva componente principal é responsável por mais variabilidade do que aquela associada a cada uma das variáveis originais. Este critério só é válido para dados que tenham sido devidamente padronizados, como fizemos.

```
get_eigenvalue(pca)[1:10, 1:2]

##        eigenvalue variance.percent
## Dim.1       12.84             44.3
## Dim.2        2.98             10.3
## Dim.3        1.42              4.9
## Dim.4        1.04              3.6
## Dim.5        0.92              3.2
## Dim.6        0.85              2.9
## Dim.7        0.72              2.5
## Dim.8        0.64              2.2
## Dim.9        0.60              2.1
## Dim.10       0.58              2.0
```

O fato de não termos adotado a quarta componente principal como um de nossos *drivers* reflete um compromisso entre a explicação da variabilidade dos dados e a interpretabilidade dos *drivers* obtidos, como veremos na próxima seção.

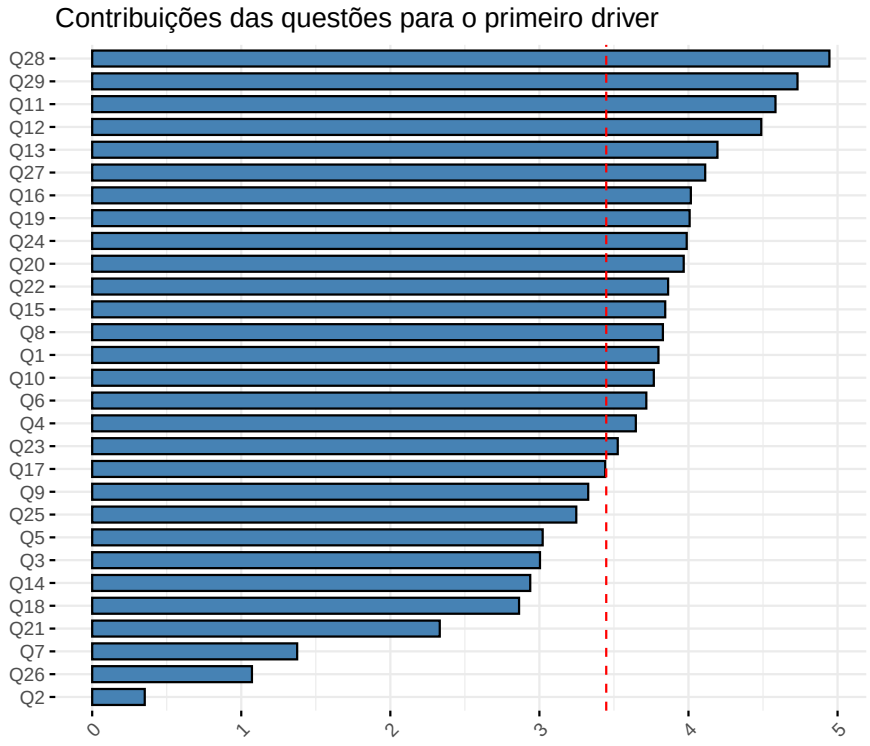
2.2 Interpretando os *drivers*

Uma vez obtidos os *drivers*, precisamos interpretá-los examinando a natureza das questões que contribuem com maior peso para cada um deles. Nosso objetivo é resumir os atributos abordados pelas questões, de maneira que possamos associar a cada *driver* um único aspecto geral dos atributos pesquisados.

Para cada *driver*, a contribuição percentual de cada questão é igual a 100 vezes o quadrado da carga correspondente dividido pela soma dos quadrados de todas as cargas.

A figura seguinte mostra a contribuição das questões para o primeiro *driver*. A linha tracejada de referência seria o valor de uma contribuição percentual igual para cada questão, ou seja, 100 dividido pelo número de questões. O parâmetro `axes` determina o número do *driver* que está sendo examinado.

```
pca %>% fviz_contrib(choice = "var", axes = 1, sort.val = "asc",
                     fill = "steelblue", color = "black") +
    labs(x = "", title = "Contribuições das questões para o primeiro driver") +
    coord_flip()
```




```
Phi <- pca$rotation

Z <- pca$x[, 1:3]
colnames(Z) <- sprintf("driver_%d", 1:3)

get_driver <- function(Phi, questoes, drv, top) {
    tibble(numero = rownames(Phi), carga = Phi[, drv]) %>%
        left_join(questoes) %>%
        mutate(contribuicao = carga^2 / sum(carga^2)) %>%
        arrange(desc(contribuicao)) %>%
        head(n = top)
}
```

Os sinais relativos das cargas e dos escores são importantes. Por exemplo, com os sinais originais, a questão 28 com escore mais negativo indicaria mais limpeza.

```
(driver_1 <- get_driver(Phi, questoes, drv = 1, top = 6))

## # A tibble: 6 x 4
##   numero  carga pergunta                                    contribuicao
##   <chr>   <dbl> <chr>                                              <dbl>
## 1 Q28    -0.222 É eficiente na limpeza da casa toda               0.0495
## 2 Q29    -0.218 Deixa a casa com aroma de limpeza                 0.0473
```

```
## 3 Q11    -0.214 Facilita a tarefa da dona de casa na limpeza da ca~       0.0458
## 4 Q12    -0.212 Dá a melhor sensação de desinfecção                       0.0449
## 5 Q13    -0.205 É a mais adequada para a lavagem de roupas                0.0419
## 6 Q27    -0.203 Deixa um aroma agradável nos lugares onde foi usada       0.0411
```

É fácil ajustar os sinais relativos para tornar mais direta a interpretação do primeiro *driver*.

```
Phi[, 1] <- -Phi[, 1]

Z[, 1] <- -Z[, 1]

(driver_1 <- get_driver(Phi, questoes, drv = 1, top = 6))
```

```
## # A tibble: 6 x 4
##   numero carga pergunta                                          contribuicao
##   <chr>  <dbl> <chr>                                                  <dbl>
## 1 Q28    0.222 É eficiente na limpeza da casa toda                    0.0495
## 2 Q29    0.218 Deixa a casa com aroma de limpeza                      0.0473
## 3 Q11    0.214 Facilita a tarefa da dona de casa na limpeza da casa   0.0458
## 4 Q12    0.212 Dá a melhor sensação de desinfecção                    0.0449
## 5 Q13    0.205 É a mais adequada para a lavagem de roupas             0.0419
## 6 Q27    0.203 Deixa um aroma agradável nos lugares onde foi usada    0.0411
```

Deste modo, dado o teor das questões mais relevantes para este primeiro *driver*, denominaremos o mesmo por *driver* "limpeza". Repetindo esta análise para o segundo *driver*, vemos que este reflete a dimensão "suavidade" dos alvejantes.

```
(driver_2 <- get_driver(Phi, questoes, drv = 2, top = 10))
```

```
## # A tibble: 10 x 4
##    numero  carga pergunta                                          contribuicao
##    <chr>   <dbl> <chr>                                                  <dbl>
##  1 Q7      0.349 É suave para as mãos                                   0.122
##  2 Q26     0.317 Não deixa um aroma forte e ruim nas mãos               0.101
##  3 Q2      0.299 É adequada para roupas coloridas                       0.0896
##  4 Q9      0.280 Deixa um aroma agradável nas roupas                    0.0787
##  5 Q23    -0.261 É eficiente para desinfetar vasos sanitários e ra~     0.0679
##  6 Q8     -0.241 É adequada para a limpeza pesada                      0.0581
##  7 Q6      0.223 Deixa um aroma agradável na casa                      0.0497
##  8 Q27     0.210 Deixa um aroma agradável nos lugares onde foi usa~    0.0442
##  9 Q24     0.207 Tem um aroma adequado para ser usada na casa toda     0.0427
## 10 Q14     0.205 É fácil de enxaguar                                   0.0419
```

Finalmente, vemos que o terceiro *driver* captura a dimensão "intensidade" dos alvejantes. Note que sempre precisamos levar em conta os sinais das cargas de cada questão durante o processo de interpretação.

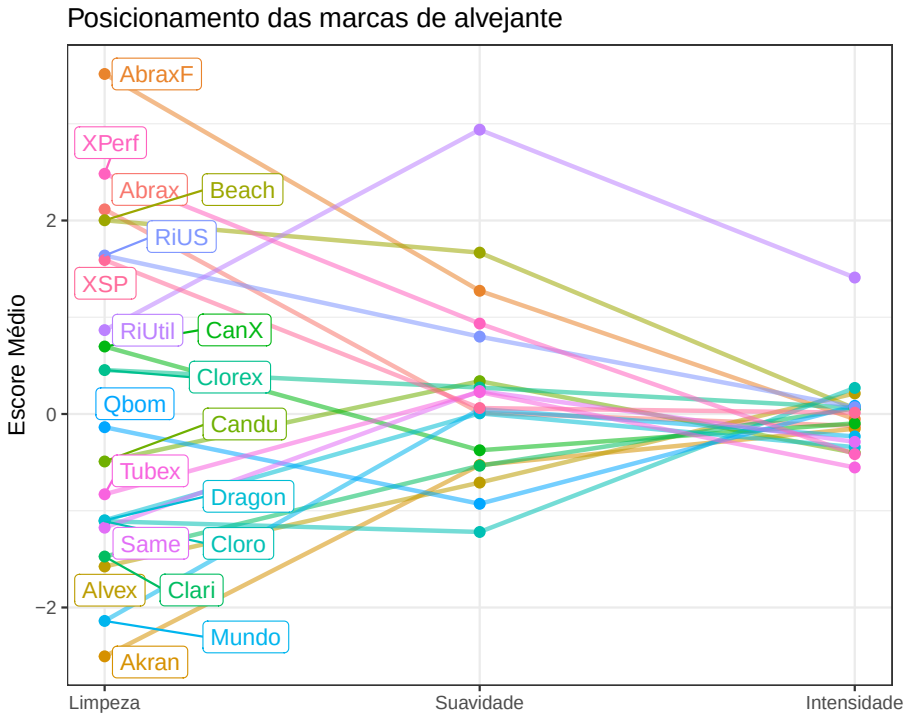
```
(driver_3 <- get_driver(Phi, questoes, drv = 3, top = 5))
```

```
## # A tibble: 5 x 4
##   numero  carga pergunta                                          contribuicao
##   <chr>   <dbl> <chr>                                                    <dbl>
## 1 Q3      0.291 É a melhor para a remoção de manchas de gordura          0.0846
## 2 Q20     0.277 É econômica no uso                                       0.0768
## 3 Q10    -0.266 É um produto para ser usado tanto na cozinha como ~      0.0709
## 4 Q19     0.254 É a melhor marca de alvejante do mercado                 0.0646
## 5 Q16     0.254 Rende mais                                               0.0646
```

2.3 Posicionando as marcas

O posicionamento das marcas é determinado pelo valor médio destas sobre os *drivers*.

```
tb <- tibble(marca = avaliacoes$marca) %>%
    bind_cols(as_tibble(Z))

tb %>%
    group_by(marca) %>%
    summarise_all(mean) %>%
    gather(key = "driver", value = "escore_medio", driver_1:driver_3) %>%
    ggplot(aes(x = driver, y = escore_medio,
               group = marca, color = marca,
               label = ifelse(driver == "driver_1", marca, ""))) +
        geom_line(size = 1, alpha = 0.55) +
        geom_point(size = 2) +
        labs(x = "", y = "Escore Médio",
             title = "Posicionamento das marcas de alvejante") +
        geom_label_repel(direction = "both") +
        scale_x_discrete(breaks = sprintf("driver_%d", 1:3),
                         labels = c("Limpeza",
                                    "Suavidade",
                                    "Intensidade"),
                         expand = c(0.05, 0)) +
        theme(legend.position = "none")
```



Vemos neste gráfico de linhas que a marca “AbraxF“ é a mais bem posicionada no quesito “Limpeza“, enquanto a marca “Akran” é a pior posicionada neste quesito. Ademais, a marca “RiUtil” é a mais bem posicionada nos quesitos “Suavidade” e “Intensidade”.

A informação sobre o posicionamento das marcas é um guia importante na composição de peças publicitárias que enfatizem de maneira diferenciada os aspectos cobertos por cada *driver*, sejam da marca anunciada, ou de suas concorrentes. Finalmente, em um eventual *redesign* do produto, sua embalagem etc, o fabricante pode investir na melhoria dos quesitos daqueles *drivers* nos quais seu produto não esteja bem posicionado.

Capítulo 3

Sacolas de compras

Pedro J. Fernandez e Paulo C. Marques F.

Neste capítulo, introduzimos um conjunto de técnicas originalmente desenvolvidas para encontrar relações expressivas em bases de dados de transações de consumidores. A busca e interpretação de tais relações formam uma análise de sacola de compras (*market basket analysis*), cujas técnicas permitem determinar quais itens são comprados juntos com maior frequência, possibilitando uma série de decisões de marketing, planejamento e vendas.

O capítulo está organizado da seguinte maneira. Na seção 3.1, apresentamos as definições básicas necessárias em uma análise de sacolas de compras, introduzido o conceito de regra de associação e as principais métricas relacionadas (suporte, confiança e *lift*). A aplicação na seção 3.2 ilustra o uso mais tradicional das regras de associação em uma análise de itens comprados em mercearias. Na seção 3.3, analisamos hábitos de consumidores de chá, exemplificando o fato de que os dados analisados podem ser mais gerais do que apenas itens em sacolas de compras.

3.1 Regras de associação

Suponha que tenhamos um conjunto $\Omega = \{L, C, S, R, V\}$, formado pelos *itens* Leite (L), Café (C), Suco (S), Refrigerante (R) e Vinho (V). Uma *sacola de compras* B é um subconjunto não vazio de Ω que representa os itens adquiridos por um consumidor em uma determinada compra.

Nossa base de dados consiste em um conjunto de sacolas de compras. Por exemplo, imagine que tenhamos as sacolas de compras

$$B_1 = \{L, C, R\}, \quad B_2 = \{L, C, R, V\}, \quad B_3 = \{L, R\},$$

$$B_4 = \{L, C, S\}, \quad B_5 = \{S, V\} \quad \text{e} \quad B_6 = \{S, R\}.$$

Suponha que queiramos determinar quais itens aparecem com maior frequência nas sacolas de compras. Para um conjunto de itens $X \subset \Omega$, o *suporte* (*support*) de X, denotado por $\text{supp}(X)$, é a fração das sacolas de compras que contém os itens de X. Em nosso exemplo, os conjuntos de itens com suporte não menor do que $1/3$ são

$$X_1 = \{L\}, \quad X_2 = \{C\}, \quad X_3 = \{S\}, \quad X_4 = \{R\},$$

$$X_5 = \{L, C\}, \quad X_6 = \{L, R\}, \quad X_7 = \{C, S\}, \quad X_8 = \{C, R\} \quad \text{e} \quad X_9 = \{L, C, R\};$$

em particular, $\text{supp}(X_5) = 1/2$.

O suporte de um conjunto de itens X admite uma interpretação probabilística bastante simples. Suponha que escolhamos uma das sacolas de compras da nossa base de dados com probabilidade uniforme. Definindo o evento $\mathscr{E}_X$ como sendo "encontrar os itens de X em uma das sacola de compras", é imediato que $\text{supp}(X)$ é igual à probabilidade $\Pr(\mathscr{E}_X)$. Note que, ao contrário do que ocorre com a medida de probabilidade $\Pr(\,\cdot\,)$, o suporte é antimonótono: quanto mais itens houver em X, menor o suporte, pois seria menos provável selecionarmos uma sacola de compras em que todos os itens de X estivessem presentes.

Dados dois conjuntos de itens X e Y, tais que $X \cap Y = \emptyset$, uma *regra de associação* é uma expressão simbólica da forma $X \Rightarrow Y$, em que X é o *antecedente* e Y é o *consequente*. Definimos o *suporte da regra de associação* $X \Rightarrow Y$, denotado por $\text{supp}(X \Rightarrow Y)$, como sendo $\text{supp}(X \cup Y)$, ou seja, a fração das sacolas de compras que contém todos os itens de X e todos os itens de Y. Como ilustração, em nosso exemplo, temos que

$$\text{supp}(\{L, C\} \Rightarrow \{R\}) = \frac{1}{3}.$$

Em termos da interpretação probabilística, $\text{supp}(X \Rightarrow Y)$ é igual à probabilidade conjunta $\Pr(\mathscr{E}_X \cap \mathscr{E}_Y)$, uma vez que o evento $\mathscr{E}_X \cap \mathscr{E}_Y$ equivale a "encontrar todos os itens de X e de Y em uma sacola de compras".

A *confiança* (*confidence*) de uma regra de associação $X \Rightarrow Y$, denotada por $\text{conf}(X \Rightarrow Y)$, é definida pela razão entre $\text{supp}(X \cup Y)$ e $\text{supp}(X)$. Assim, em nosso exemplo, temos que

$$\text{conf}(\{L, C\} \Rightarrow \{R\}) = \frac{2/6}{3/6} = \frac{2}{3}.$$

Em termos da interpretação probabilística, $\text{conf}(X \Rightarrow Y)$ é igual à probabilidade condicional $\Pr(\mathscr{E}_Y \mid \mathscr{E}_X)$.

A estratégia ao minerar regras de associação em uma base de dados é estabelecer um suporte mínimo, de maneira a limitar o número de regras de associação que iremos examinar, procurando entra as regras remanescentes aquelas com confiança mais alta, uma vez que tais regras indicariam associações interessantes.

Em nosso exemplo, vimos que, dado que a sacola de compras contém Leite e Café, a probabilidade condicional da sacola também conter Refrigerante é igual a 2/3, que é a confiança obtida para a regra de associação $\{L, C\} \Rightarrow \{R\}$.

A princípio, parece que encontramos uma associação relevante entre os itens, mas há um risco nesta interpretação, pois de fato pode ocorrer que a probabilidade condicional

$\Pr(\mathscr{E}_{\{R\}} \mid \mathscr{E}_{\{L,C\}})$ seja "alta" apenas porque $\Pr(\mathscr{E}_{\{R\}})$ seja "alta" e os eventos $\mathscr{E}_{\{R\}}$ e $\mathscr{E}_{\{L,C\}}$ sejam independentes, de modo que $\Pr(\mathscr{E}_{\{R\}} \mid \mathscr{E}_{\{L,C\}}) = \Pr(\mathscr{E}_{\{R\}})$. De fato, isso é exatamente o que ocorre em nosso exemplo. Para distinguirmos casos como este, precisamos de uma métrica adicional que meça a dependência entre os eventos definidos pelo antecedente e o consequente da regra de associação considerada.

O *lift* (*levante*) de uma regra de associação $X \Rightarrow Y$, denotado por $\text{lift}(X \Rightarrow Y)$, é definido pela razão entre $\text{supp}(X \cup Y)$ e o produto $\text{supp}(X) \times \text{supp}(Y)$. Em termos da interpretação probabilística, $\text{lift}(X \Rightarrow Y)$ é igual à razão de probabilidades

$$\frac{\Pr(\mathscr{E}_X \cap \mathscr{E}_Y)}{\Pr(\mathscr{E}_X)\Pr(\mathscr{E}_Y)}.$$

Note que, se os eventos $\mathscr{E}_X$ e $\mathscr{E}_Y$ forem independentes, então $\text{lift}(X \Rightarrow Y) = 1$. Assim, o *lift* pode ser entendido como uma medida da nossa surpresa ao encontrarmos os itens de X e de Y juntos em uma mesma sacola de compras, tomando como referência a situação em que os eventos $\mathscr{E}_X$ e $\mathscr{E}_Y$ sejam independentes.

Conforme observamos, em nosso exemplo, temos que $\text{lift}(\{L, C\} \Rightarrow \{R\}) = 1$. Para fins de comparação, a regra de associação $\{L, R\} \Rightarrow \{C\}$ também possui confiança igual a $2/3$. No entanto, $\text{lift}(\{L, R\} \Rightarrow \{C\}) \approx 1{,}33$.

Estas considerações completam a estratégia de mineração de regras de associação expressivas em uma base de dados de sacolas de compras: 1) Fixamos um suporte mínimo para limitarmos o número de regras de associação que iremos examinar; 2) Selecionamos as regras de associação com confiança "alta" e *lift* maior do que um.

Dada uma base de dados com informações sobre sacolas de compras, não é viável examinar de maneira simples, por enumeração direta e exaustiva, todas as regras de associação existentes que satisfaçam determinados critérios. A simples determinação de todos os conjuntos de itens com suporte superior a um certo *threshold* já enfrenta uma explosão combinatória. O engenhoso algoritmo *Apriori* de Agrawal et al. [1] permite obter as regras de associação de maneira eficiente, mediante a fixação de valores para métricas tais como o suporte, a confiança e o *lift*.

O algoritmo *Apriori* é considerado uma das grandes invenções na área de mineração de dados (*data mining*). Valendo-se da antimonotonicidade do suporte, o algoritmo *Apriori* torna tratável o problema de determinação das regras de associação em bases de dados de tamanho considerável. Por exemplo, em um grande supermercado norte-americano podemos ter mais de cem mil itens à venda e uma base de dados com mais de um bilhão de sacolas de compras, cujas regras de associação são mineradas via algoritmo *Apriori* sem grande dificuldade.

Na próxima seção, mineramos regras de associação aplicando o algoritmo *Apriori* a uma base de dados de compras em mercearias.

3.2 Compras em mercearias

O algoritmo *Apriori* é implementado pela biblioteca `arules`, da qual faz parte o objeto `Groceries`, uma base de dados com 169 itens distribuídos em 9.835 sacolas de compras.

```
library(arules)

data(Groceries)

summary(Groceries)

## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.0261
##
## most frequent items:
##       whole milk other vegetables       rolls/buns             soda
##             2513             1903             1809             1715
##           yogurt          (Other)
##             1372            34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     2.0     3.0     4.4     6.0    32.0
##
## includes extended item information - examples:
##        labels  level2           level1
## 1 frankfurter sausage meat and sausage
## 2     sausage sausage meat and sausage
## 3  liver loaf sausage meat and sausage
```

Vamos procurar as regras de associação com suporte não menor do que 0,1% e confiança não menor do que 80%.

```
rules <- apriori(Groceries, parameter = list(sup = 0.001, conf = 0.8))

## Apriori
```

```
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.01s].
## writing ... [410 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Obtivemos 410 regras. Podemos inspecioná-las individualmente.

```
inspect(rules[30])
```

```
##     lhs                    rhs          support confidence lift count
## [1] {other vegetables,
##      yogurt,
##      specialty cheese} => {whole milk} 0.00132      0.812 3.18    13
```

Ou, por exemplo, listar as 10 primeiras regras obtidas.

```
inspect(rules[1:10])
```

```
##      lhs                          rhs                support confidence  lift count
## [1]  {liquor,
##       red/blush wine}         => {bottled beer}     0.00193      0.905 11.24    19
## [2]  {curd,
##       cereals}                => {whole milk}       0.00102      0.909  3.56    10
## [3]  {yogurt,
```

```
##       cereals}                => {whole milk}       0.00173      0.810  3.17    17
## [4]  {butter,
##       jam}                    => {whole milk}       0.00102      0.833  3.26    10
## [5]  {soups,
##       bottled beer}           => {whole milk}       0.00112      0.917  3.59    11
## [6]  {napkins,
##       house keeping products} => {whole milk}       0.00132      0.812  3.18    13
## [7]  {whipped/sour cream,
##       house keeping products} => {whole milk}       0.00122      0.923  3.61    12
## [8]  {pastry,
##       sweet spreads}          => {whole milk}       0.00102      0.909  3.56    10
## [9]  {turkey,
##       curd}                   => {other vegetables} 0.00122      0.800  4.13    12
## [10] {rice,
##       sugar}                  => {whole milk}       0.00122      1.000  3.91    12
```

As regras podem ser ordenadas pela confiança.

```
inspect(head(rules, n = 5, by = "confidence"))
```

```
##     lhs                     rhs          support confidence lift count
## [1] {rice,
##      sugar}              => {whole milk} 0.00122          1 3.91    12
## [2] {canned fish,
##      hygiene articles}   => {whole milk} 0.00112          1 3.91    11
## [3] {root vegetables,
##      butter,
##      rice}               => {whole milk} 0.00102          1 3.91    10
## [4] {root vegetables,
##      whipped/sour cream,
##      flour}              => {whole milk} 0.00173          1 3.91    17
## [5] {butter,
##      soft cheese,
##      domestic eggs}      => {whole milk} 0.00102          1 3.91    10
```

Também podemos ordenar as regras pelo *lift*.

```
inspect(head(rules, n = 5, by = "lift"))
```

```
##     lhs                     rhs              support confidence  lift count
## [1] {liquor,
##      red/blush wine}     => {bottled beer}    0.00193      0.905 11.24    19
```

```
## [2] {citrus fruit,
##      other vegetables,
##      soda,
##      fruit/vegetable juice} => {root vegetables} 0.00102      0.909  8.34    10
## [3] {tropical fruit,
##      other vegetables,
##      whole milk,
##      yogurt,
##      oil}                   => {root vegetables} 0.00102      0.909  8.34    10
## [4] {citrus fruit,
##      grapes,
##      fruit/vegetable juice} => {tropical fruit}  0.00112      0.846  8.06    11
## [5] {other vegetables,
##      whole milk,
##      yogurt,
##      rice}                  => {root vegetables} 0.00132      0.867  7.95    13
```

Ou por meio de uma ordenação composta, envolvendo mais de uma métrica.

```
inspect(head(rules, n = 5, by = c("confidence", "lift")))
```

```
##     lhs                     rhs                 support confidence lift count
## [1] {citrus fruit,
##      root vegetables,
##      soft cheese}        => {other vegetables} 0.00102          1 5.17    10
## [2] {pip fruit,
##      whipped/sour cream,
##      brown bread}        => {other vegetables} 0.00112          1 5.17    11
## [3] {tropical fruit,
##      grapes,
##      whole milk,
##      yogurt}             => {other vegetables} 0.00102          1 5.17    10
## [4] {ham,
##      tropical fruit,
##      pip fruit,
##      yogurt}             => {other vegetables} 0.00102          1 5.17    10
## [5] {ham,
##      tropical fruit,
##      pip fruit,
##      whole milk}         => {other vegetables} 0.00112          1 5.17    11
```

O uso típico em marketing seria fixar um conjunto de itens como alvo para o consequente e obter as regras de associação correspondentes. Por exemplo, suponha que fixemos cerveja engarrafada como sendo o único item do consequente.

```
target <- subset(rules, subset = rhs %in% "bottled beer")

inspect(target)

##     lhs                          rhs            support confidence lift count
## [1] {liquor,red/blush wine} => {bottled beer} 0.00193 0.905      11.2 19
```

Assim, dado que um consumidor adquiriu bebidas destiladas e vinho tinto ou rosé, a probabilidade condicional de que adquira cerveja engarrafada é de 90,5%. A regra de associação encontrada é relevante, uma vez que seu *lift* é de 11,2.

Portanto, é provável que seja vantajoso para a mercearia dispor em posições próximas nas prateleiras os itens bebidas destiladas, vinho tinto ou rosé e cerveja engarrafada, uma vez que a probabilidade do consumidor interessado em destilados e vinhos comprar cerveja engarrafada por associação é alta. Uma outra possibilidade seria criar algum tipo de venda promocional de todos esses itens, de maneira a favorecer o lucro da mercearia.

A aplicação da próxima seção estende o uso das regras de associação para dados que não são estritamente sacolas de compras.

3.3 Consumidores de chá

O uso das regras de associação pode ser estendido a problemas mais gerais. Aqui temos uma base de dados com informações sobre consumidores de chá obtida em `http://factominer.free.fr/book/tea.csv`. Agrupamos as idades dos consumidores em faixas etárias e selecionamos nove variáveis categóricas.

```
db <- read.table("dados/sacolas/tea.csv", sep = ";", header = TRUE)

db$age <- cut(db$age,
              breaks = c(0, 24, 34, 44, 59, 200),
              labels = c("15-24", "25-34", "35-44", "45-59", "60+"))

tea <- db[, c(13, 14, 16, 15, 17, 2, 20, 22, 19)]
```

1. Que tipo de chá você consome com maior frequência?

```
table(tea$variety)

##
##     black flavoured     green
##        74       193        33
```

2. Como você consome seu chá?

```
table(tea$how)

##
##         lemon          milk nothing.added         other
##            33            63           195             9
```

3. Como é embalado o chá que você consome?

```
table(tea$format)

##
##        loose       sachet sachet+loose
##           36          170           94
```

4. Você toma chá com açúcar?

```
table(tea$sugar)

##
## not.sugar     sugar
##       155       145
```

5. Em que tipo de loja você compra seu chá?

```
table(tea$place.of.purchase)

##
##         specialist.shop             supermarket supermarket+specialist.
##                      30                     192                      78
```

6. Você consome chá no período da tarde?

```
table(tea$afternoon.tea)

##
##   afternoon.tea Not.afternoon.t
##             169             131
```

7. Profissão.

```
table(tea$profession)

##
##          employee        management   manual labourer        other work
##                59                40                12                20
## senior management           student        unemployed
##                35                70                64
```

8. Faixa etária.

```
table(tea$age)

##
## 15-24 25-34 35-44 45-59   60+
##    92    69    40    61    38
```

9. Sexo.

```
table(tea$sex)

##
##   F   M
## 178 122
```

A partir deste data frame em que todas as variáveis são categóricas, criamos uma lista de transações.

```
trn <- as(tea, "transactions")

summary(trn)
```

```
## transactions as itemMatrix in sparse format with
##  300 rows (elements/itemsets/transactions) and
##  31 columns (items) and a density of 0.29
##
## most frequent items:
##             how=nothing.added             variety=flavoured
##                           195                           193
## place.of.purchase=supermarket                       sex=F
##                           192                           178
##                 format=sachet                      (Other)
##                           170                          1772
##
## element (itemset/transaction) length distribution:
## sizes
##   9
## 300
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       9       9       9       9       9       9
##
## includes extended item information - examples:
##              labels variables    levels
## 1     variety=black   variety     black
## 2 variety=flavoured   variety flavoured
## 3     variety=green   variety     green
##
## includes extended transaction information - examples:
##   transactionID
## 1             1
## 2             2
## 3             3
```

O próximo passo é minerar as regras de associação com suporte não menor do que 20% e confiança não menor do que 80%.

```
rules <- apriori(trn, parameter = list(sup = 0.2, conf = 0.8))

## Apriori
##
## Parameter specification:
```

```
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.2      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 60
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[31 item(s), 300 transaction(s)] done [0.00s].
## sorting and recoding items ... [19 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [20 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

summary(rules)

## set of 20 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3  4
##  2 11  7
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    3.00    3.00    3.25    4.00    4.00
##
## summary of quality measures:
##     support        confidence          lift           count
##  Min.   :0.200   Min.   :0.800   Min.   :1.24   Min.   : 60.0
##  1st Qu.:0.213   1st Qu.:0.836   1st Qu.:1.34   1st Qu.: 64.0
##  Median :0.235   Median :0.860   Median :1.35   Median : 70.5
##  Mean   :0.259   Mean   :0.859   Mean   :1.38   Mean   : 77.7
##  3rd Qu.:0.272   3rd Qu.:0.873   3rd Qu.:1.42   3rd Qu.: 81.5
##  Max.   :0.487   Max.   :0.910   Max.   :1.53   Max.   :146.0
##
## mining info:
```

```
##  data ntransactions support confidence
##   trn           300     0.2        0.8
```

Vejamos as dez melhores regras de associação, ordenadas pela confiança e pelo *lift*.

```
inspect(head(rules, n = 10, by = c("confidence", "lift")))
```

```
##      lhs                                rhs                           support confidence lift count
## [1]  {how=nothing.added,
##       format=sachet,
##       sex=F}                         => {place.of.purchase=supermarket}   0.203      0.910 1.42    61
## [2]  {variety=flavoured,
##       how=nothing.added,
##       format=sachet}                 => {place.of.purchase=supermarket}   0.233      0.909 1.42    70
## [3]  {how=nothing.added,
##       format=sachet}                 => {place.of.purchase=supermarket}   0.327      0.907 1.42    98
## [4]  {variety=flavoured,
##       format=sachet,
##       sugar=sugar}                   => {place.of.purchase=supermarket}   0.213      0.901 1.41    64
## [5]  {format=sachet,
##       sugar=sugar}                   => {place.of.purchase=supermarket}   0.267      0.899 1.40    80
## [6]  {variety=flavoured,
##       sugar=sugar,
##       place.of.purchase=supermarket} => {format=sachet}                   0.213      0.865 1.53    64
## [7]  {variety=flavoured,
##       format=sachet,
##       sex=F}                         => {place.of.purchase=supermarket}   0.213      0.865 1.35    64
## [8]  {variety=flavoured,
##       format=sachet}                 => {place.of.purchase=supermarket}   0.337      0.863 1.35   101
## [9]  {format=sachet,
##       afternoon.tea=afternoon.tea}   => {place.of.purchase=supermarket}   0.250      0.862 1.35    75
## [10] {format=sachet,
##       sex=F}                         => {place.of.purchase=supermarket}   0.287      0.860 1.34    86
```

Suponha que estejamos interessados em uma ação de vendas envolvendo chás "saborizados". Especificando este consequente, chegamos às seguintes regras de associação.

```
target <- subset(rules, subset = rhs %in% "variety=flavoured")

inspect(target)
```

```
##     lhs                                rhs                  support confidence lift count
## [1] {age=15-24}                     => {variety=flavoured}   0.257      0.837 1.30    77
## [2] {sugar=sugar,
##      sex=F}                         => {variety=flavoured}   0.203      0.859 1.34    61
## [3] {format=sachet,
##      sugar=sugar,
##      place.of.purchase=supermarket} => {variety=flavoured}   0.213      0.800 1.24    64
```

Observando as regras obtidas, podemos personalizar nossa ação de vendas. Por exemplo, a primeira regra de associação sugere como público alvo consumidores na faixa etária de 15 a 24 anos.

Esperamos que esta breve introdução às regras de associação abra um vasto universo de *data mining* e leve o leitor a aplicações que transcendam as análises de sacola de compras. A técnicas apresentadas são bastante gerais, podendo ser utilizadas na análise de dados textuais, em Genética, na classificação de imagens, na análise de tráfego em redes de computadores, entre outras aplicações.

Capítulo 4

Modelos de escolha

Pedro J. Fernandez, Tiago Mendonça dos Santos e Paulo C. Marques F.

Os Modelos de Escolha Discreta (*Discrete Choice Models*) são de especial interesse para os praticantes do Marketing Analítico. Amplamente disseminados, estes modelos representam de maneira flexível diversos aspectos das escolhas feitas por consumidores. Aplicações dos modelos de escolha discreta englobam a estimação da demanda por produtos de giro rápido e bens duráveis (*fast-moving consumer goods* e *durable goods*, respectivamente), serviços, atividades recreacionais etc. Uma referência importante que discute aplicações dos modelos de escolha discreta é o livro de Train [19].

4.1 Utilidade latente

Suponha que um consumidor enfrente uma decisão de escolha entre um número finito de alternativas (daí o nome modelos de escolha "discreta"). Para o consumidor, tais escolhas apresentam características e custos distintos. Formalmente, para $i = 1, \ldots, n$, o consumidor i é apresentado a um conjunto de k alternativas, atribuindo utilidades $U_{ij} \in \mathbb{R}$ às alternativas $j = 1, \ldots, k$, de maneira que estas utilidades ordenem suas preferências relativas dentro do conjunto das alternativas que lhe foram apresentadas (veja [14] e [19]). Isto define uma função de utilidade que é conhecida para o consumidor, mas não para o pesquisador. O consumidor i escolhe a alternativa j se e somente se $U_{ij} > U_{i\ell}$, para todo $\ell \neq j$.

Mesmo não podendo observar diretamente as utilidades U_{ij}, o pesquisador possui informações sobre atributos das alternativas, conforme percebidos por cada consumidor. Denotamos estas informações por $x_{ij} \in \mathbb{R}^p$, para $i = 1, \ldots, n$ e $j = 1, \ldots, k$, em que p é o número de atributos de cada alternativa. Por exemplo, em um contexto de escolha entre marcas de sabão em pó, podemos conhecer atributos tais como: nível de conhecimento do consumidor sobre o fabricante; percepção individual da eficiência do produto na limpeza; opinião do consumidor quanto à degradação de tecidos submetidos ao produto etc.

Além das informações relativas às alternativas, o pesquisador pode observar variáveis que dependam somente do consumidor, tais como classe social, idade, número de filhos, ou número de membros da família. Nos referimos a estas variáveis como "sociodemográficas", denotando o vetor de suas componentes para o consumidor i por $z_i \in \mathbb{R}^q$. O conteúdo

destas variáveis pode ser bastante genérico; basta que seja dado em função apenas do consumidor considerado.

Ademais, o pesquisador observa, para $i = 1, \ldots, n$, as escolhas $y_i \in \mathbb{R}^k$ feitas pelos consumidores entre as alternativas existentes, em que o vetor y_i é definido de maneira que sua j-ésima componente seja igual a 1 se o consumidor i fez a escolha j, e as demais componentes sejam iguais a 0 (convenção denominada *one-hot encoding* pelos praticantes de *Machine Learning*).

Suponha que tenhamos definido uma função $V(x_{ij}, z_i)$, de maneira que a utilidade U_{ij} (latente, não observada) possa ser escrita como $U_{ij} = V(x_{ij}, z_i) + \epsilon_{ij}$, em que ϵ_{ij} é uma variável aleatória que representa todos os fatores não observados que afetam a utilidade U_{ij}. Em um modelo de escolha, estamos interessados na probabilidade π_{ij} do indivíduo i fazer a escolha j, dada por

$$\begin{aligned}\pi_{ij} &= \Pr\{U_{ij} > U_{i\ell}, \text{para todo } \ell \neq j\} \\ &= \Pr\{\epsilon_{i\ell} - \epsilon_{ij} < V_{ij} - V_{i\ell}, \text{para todo } \ell \neq j\},\end{aligned}$$

na qual utilizamos a abreviação $V_{ij} = V(x_{ij}, z_i)$. É evidente que tais probabilidades dependem da distribuição dos ϵ_{ij}. Em particular, McFadden [14] demonstrou que, se os ϵ_{ij} são independentes e possuem distribuição de valor extremo do tipo I, então estas probabilidades podem ser escritas na forma

$$\pi_{ij} = \frac{\exp(V_{ij})}{\sum_{\ell=1}^{k} \exp(V_{i\ell})}.$$

Apresentamos a demonstração deste resultado no final deste capítulo.

Na próxima seção, para que possamos descrever um painel de consumidores que fazem escolhas sucessivas, generalizamos esta discussão do problema de escolha de alternativas de consumo, introduzindo o modelo bayesiano hierárquico que será utilizado em nossa aplicação.

4.2 Modelo Bayesiano hierárquico

Generalizando o cenário descrito anteriormente, suponha que um mesmo consumidor faça escolhas sucessivas ao longo do tempo, de modo que os atributos x_{ij} introduzidos na seção anterior sejam estendidos para $x_{itj} \in \mathbb{R}^p$, em que o índice t denota os diversos momentos em que as escolhas são feitas pelo consumidor i. De maneira análoga, as escolhas correspondentes feitas pelo consumidor i ao longo do tempo passam a ser denotadas por $y_{it} \in \mathbb{R}^k$.

Introduzimos uma matriz de parâmetros (coeficientes de regressão) $\Delta \in \mathbb{R}^{q\times p}$ e vetores aleatórios $w_i \in \mathbb{R}^p$, condicionalmente independentes e identicamente distribuídos, com distribuição normal multivariada, tendo vetor de médias nulo e matriz de covariâncias Σ, para $i = 1, \ldots, n$. Definimos vetores $\beta_i \in \mathbb{R}^p$ por

$$\beta_i = \Delta^\top z_i + w_i,$$

para $i = 1, \ldots, n$. Guiados por McFadden [14], mas sem utilizarmos seus resultados de maneira estrita, definimos

$$\pi_{itj} = \frac{\exp(x_{itj}^\top \beta_i)}{\sum_{\ell=1}^k \exp(x_{it\ell}^\top \beta_i)},$$

e postulamos que $y_{it} \in \mathbb{R}^k$ tenha condicionalmente distribuição categórica com probabilidades π_{itj}.

Para completarmos este modelo bayesiano hierárquico, precisamos de distribuições *a priori* para Δ e Σ. Uma vez especificado o modelo, exploramos a distribuição *a posteriori* dos parâmetros utilizando um método de Monte Carlo de cadeia de Markov. Os detalhes das distribuições *a priori* e da técnica de simulação podem ser encontrados em [16].

Na próxima seção, este modelo de escolha é aplicado a um problema com dados de consumidores interessados em cruzeiros marítimos.

4.3 Cruzeiros marítimos

A seguir, apresentamos um exemplo de modelo de escolha aplicado ao mercado de cruzeiros marítimos. Neste exemplo, utilizamos dados fornecidos gentilmente pela *Sawtooth Software* (`http://www.sawtoothsoftware.com/`).

Durante o mês de outubro de 2015, a *Sawtooth Software* coletou dados utilizando um painel comercial gerenciado pela *Survey Sampling International* (SSI). Neste estudo, observamos escolhas de férias de 7 a 11 dias de duração em cruzeiros marítimos feitas por 600 painelistas.

A cada painelista são apresentadas 15 *tarefas*. Cada uma das tarefas envolve a escolha de um entre quatro *conceitos* de cruzeiro. Um conceito é determinado pelos valores de 6 atributos. Os valores possíveis dos atributos estão listados no final deste capítulo. A Tabela 4.1 exemplifica uma das tarefas apresentadas aos painelistas.

```
library(tidyverse)

(tasks <- read_csv("dados/escolhas/tasks.csv"))
```

Atributo	**Conceito 1**	**Conceito 2**	**Conceito 3**	**Conceito 4**
Destination	Alaska	Mexican Riviera	Western Caribean	Mediterranean
Cruise line	Disney	Norwegian	Royal Caribbean	Princess
Number of days	7 days	10 days	7 days	8 days
Stateroom	Oceanview stateroom, porthole window	Inside stateroom (no windows)	Balcony stateroom, sliding door to private balcony	Oceanview stateroom, porthole window
Ship Amenities	Fewer Amenities	More Amenities	Fewer Amenities	More Amenities
Price per person per day	$125	$100	$200	$100

Tabela 4.1: Uma das possíveis tarefas apresentadas aos painelistas.

```
## # A tibble: 36,000 x 10
##    CaseID  Task Concept Dest   Cline   Days  State  Amenit Price Response
##     <dbl> <dbl>   <dbl> <chr>  <chr>   <chr> <chr>  <chr>  <chr>    <dbl>
##  1      1     1       1 ECab   Holland 8D    Oview  Few    $125         0
##  2      1     1       2 WCab   Norge   7D    Insid  Few    $100         1
##  3      1     1       3 Alask  Princ   9D    Oview  More   $200         0
##  4      1     1       4 Mex    Royal   11D   Balcon More   $150         0
##  5      1     2       1 Medit  Carni   9D    Insid  Few    $100         1
##  6      1     2       2 WCab   Disney  8D    Balcon More   $125         0
##  7      1     2       3 Medit  Norge   7D    Oview  More   $175         0
##  8      1     2       4 Norway Disney  10D   Balcon Few    $200         0
##  9      1     3       1 ECab   Princ   11D   Balcon Few    $175         1
## 10      1     3       2 WCab   Holland 9D    Oview  More   $150         0
## # ... with 35,990 more rows
```

Assim, vemos, por exemplo, que o painelista identificado pelo `CaseID` 1, ao responder a `Task` 1, escolheu o conceito 2 (observe o *one-hot encoding* dos quatro conceitos desta tarefa na coluna `Response`).

Nesta base de dados, temos uma série de variáveis demográficas e atitudinais disponíveis para cada painelista.

```
(panel <- read_csv("dados/escolhas/panel.csv"))
```

```
## # A tibble: 600 x 20
##    CaseID Cruisedbefore Howmany Intend Abiltravel Abilspend Motivation_1
##     <dbl>         <dbl>   <dbl>  <dbl>      <dbl>     <dbl>        <dbl>
##  1      1             1       3      1          1         2            0
##  2      3             1       3      2          1         2            1
##  3      5             1       5      1          2         2            0
##  4      6             2       0      1          2         2            0
##  5      7             1       2      2          1         2            1
##  6      9             1       1      1          2         2            1
##  7     10             2       0      2          2         2            1
##  8     12             1       1      2          2         2            1
##  9     14             1       3      1          1         1            1
## 10     16             1       2      2          2         2            0
## # ... with 590 more rows, and 13 more variables: Motivation_2 <dbl>,
## #   Motivation_3 <dbl>, Motivation_4 <dbl>, Motivation_5 <dbl>,
## #   Motivation_6 <dbl>, Motivation_7 <dbl>, Motivation_8 <dbl>,
## #   TopDestination <dbl>, TopCruiseline <dbl>, TopLength <dbl>, Demos1 <dbl>,
## #   Demos2 <dbl>, Demos3 <dbl>
```

Em nosso exemplo, trabalharemos com apenas duas variáveis demográficas: `Howmany`, que determina quantos cruzeiro marítimos o painelista já vez; e `Abiltravel`, que determina se o painelista considera ter mais tempo de férias do que outras pessoas que ele conhece, codificada como um ordinal com os valores 1, 2 e 3 (quanto maior o valor, melhor a auto-percepção do painelista em relação a este quesito).

O modelo Bayesiano hierárquico é implementado pela biblioteca `bayesm`. Nesta simulação, adotamos um *burn-in* de 500 iterações, com um comprimento efetivo da cadeia de Markov de 5000 iterações.

```
X <- tasks %>%
    select(CaseID) %>%
    bind_cols(model.matrix(~ Dest + Cline + Days + State + Amenit + Price,
                           data = tasks)[, -1] %>% as_tibble())

Z <- scale(panel %>% select(Howmany, Abiltravel),
           center = TRUE, scale = FALSE)
```

```
ids  <- unique(tasks$CaseID)
lgt <- vector("list", length(ids))

for (i in 1:length(ids)) {
    lgt[[i]] <- list(y = tasks %>%
                             filter(CaseID == ids[i], Response == 1) %>%
                             select(Concept) %>%
                             as.matrix(),
                     X = X %>%
                             filter(CaseID == ids[i]) %>%
                             select(-CaseID) %>%
                             as.matrix())
}

BURN_IN <- 500
N <- 5000
```

```
library(bayesm)

model <- rhierMnlRwMixture(Data = list(lgtdata = lgt, Z = Z, p = 4),
                           Prior = list(ncomp = 1),
                           Mcmc = list(R = BURN_IN + N))
```

4.4 Simulando *shares*

O uso mais importante dos modelos de escolha se dá na previsão dos *shares* de novos produtos. Como exemplo, façamos a comparação entre os três produtos determinados pelos conceitos da Tabela 4.2. Note que neste exemplo estamos variando apenas dois dos atributos entre os conceitos apresentados: a linha do cruzeiro e o preço diário por pessoa. A generalização deste exemplo para cenários com mais produtos, bem como uma maior variedade dos valores dos atributos, é imediata.

```
dummies <- colnames(X %>% select(-CaseID))

attrib <- list(c("DestECab", "ClineHolland", "Days7D",
                 "StateOview", "AmenitFewer", "Price$125"),
               c("DestECab", "ClinePrinc", "Days7D",
                 "StateOview", "AmenitFewer", "Price$150"),
```

Atributo	**Conceito 1**	**Conceito 2**	**Conceito 3**
Destination	Eastern Caribbean	Eastern Caribbean	Eastern Caribbean
Cruise line	Holland America	Princess	Disney
Number of days	7 days	7 days	7 days
Stateroom	Oceanview	Oceanview	Oceanview
Ship Amenities	Fewer Amenities	Fewer Amenities	Fewer Amenities
Price per person per day	$125	$150	$150

Tabela 4.2: Conceitos de três produtos em potencial.

```
               c("DestECab", "ClineDisney", "Days7D",
                 "StateOview", "AmenitFewer", "Price$155"))

concepts <- matrix(0, nrow = 3, ncol = length(dummies))
colnames(concepts) <- dummies

for (i in 1:length(attrib)) {
    concepts[i, dummies %in% attrib[[i]]] <- 1
}

concepts %>% as_tibble()

## # A tibble: 3 x 21
##   DestECab DestMedit DestMex DestNorway DestWCab ClineDisney ClineHolland
##      <dbl>     <dbl>   <dbl>      <dbl>    <dbl>       <dbl>        <dbl>
## 1        1         0       0          0        0           0            1
## 2        1         0       0          0        0           0            0
## 3        1         0       0          0        0           1            0
## # ... with 14 more variables: ClineNorge <dbl>, ClinePrinc <dbl>,
## #   ClineRoyal <dbl>, Days11D <dbl>, Days7D <dbl>, Days8D <dbl>, Days9D <dbl>,
## #   StateInsid <dbl>, StateOview <dbl>, AmenitMore <dbl>, `Price$125` <dbl>,
## #   `Price$150` <dbl>, `Price$175` <dbl>, `Price$200` <dbl>
```

A distribuição *a posteriori* dos parâmetros do modelo nos fornece estimativas pontuais dos *shares* dos três conceitos analisados.

```
posterior_beta <- model$betadraw[, , (BURN_IN + 1):(BURN_IN + N)]

n <- dim(posterior_beta)[1]
```

```
shares <- array(dim = c(n, nrow(concepts), N))

for (t in 1:N) {
    for (i in 1:n) {
        U <- concepts %*% posterior_beta[i, , t]
        shares[i, , t] <- exp(U) / sum(exp(U))
    }
}

agg_shares <- 100 * apply(shares, 2:3, mean)

round(apply(agg_shares, 1, mean), 2)

## [1] 32.5 27.3 40.2
```

Deste modo, esperamos que o terceiro conceito seja escolhido com maior frequência, com um *share* estimado de 40,2%. A distribuição *a posteriori* dos parâmetros do modelo também nos fornece intervalos de credibilidade para as três estimativas dos *shares*. Por exemplo, com 95% de probabilidade, temos os seguintes intervalos de credibilidade.

```
round(apply(agg_shares, 1, quantile, probs = c(0.025, 0.975)), 2)

##       [,1] [,2] [,3]
## 2.5%  29.4 24.6 37.6
## 97.5% 35.8 30.1 42.9
```

Modelo de McFadden

Suponha que a utilidade seja escrita como $U_{ij} = V_{ij} + \epsilon_{ij}$, em que os ϵ_{ij} são independentes e têm distribuição de valor extremo do tipo I. Assim, a densidade de cada ϵ_{ij} é dada por

$$f(t) = \exp(-t)\exp(-\exp(-t)),$$

para $t \in \mathbb{R}$; e a função de distribuição correspondente é

$$F(t) = \Pr\{\epsilon_{ij} \leq t\} = \int_{-\infty}^{t} f(u)\,du = \exp(-\exp(-t)).$$

Queremos calcular

$$\begin{aligned}\pi_{ij} &= \Pr\{U_{ij} > U_{i\ell}, \text{para todo } \ell \neq j\} = \Pr(\cap_{\ell\neq j}\{\epsilon_{i\ell} < \epsilon_{ij} + V_{ij} - V_{i\ell}\}) \\ &= \int_{-\infty}^{\infty} \Pr(\cap_{\ell\neq j}\{\epsilon_{i\ell} < \epsilon_{ij} + V_{ij} - V_{i\ell}\} \mid \epsilon_{ij} = t)\, f(t)\, dt \\ &= \int_{-\infty}^{\infty} \Pr(\cap_{\ell\neq j}\{\epsilon_{i\ell} < t + V_{ij} - V_{i\ell}\})\, f(t)\, dt \\ &= \int_{-\infty}^{\infty} \left(\prod_{\ell\neq j} \Pr\{\epsilon_{i\ell} < t + V_{ij} - V_{i\ell}\}\right) f(t)\, dt,\end{aligned}$$

na qual utilizamos o teorema da probabilidade total e a independência dos $\epsilon_{i\ell}$.

Observando que

$$\prod_{\ell\neq j} \Pr\{\epsilon_{i\ell} < t + V_{ij} - V_{i\ell}\} = \frac{\prod_{\ell=1}^{k} \Pr\{\epsilon_{i\ell} < t + V_{ij} - V_{i\ell}\}}{\Pr\{\epsilon_{ij} < t\}}$$

e utilizando as expressões de $f(t)$ e $F(t)$, temos que

$$\pi_{ij} = \int_{-\infty}^{\infty} \exp\left(-\exp(-t)\exp(-V_{ij})\sum_{\ell=1}^{k}\exp(V_{i\ell})\right)\exp(-t)\, dt.$$

Fazendo a mudança de variável $u = \exp(-t)$, de modo que $du = -\exp(-t)\, dt$, transformando e invertendo os novos limites de integração, temos que

$$\pi_{ij} = \int_{0}^{\infty} \exp\left(-\left(\exp(-V_{ij})\sum_{\ell=1}^{k}\exp(V_{i\ell})\right)u\right) du = \frac{\exp(V_{ij})}{\sum_{\ell=1}^{k}\exp(V_{i\ell})},$$

como queríamos.

Atributos

Temos abaixo todos os valores possíveis dos seis atributos pesquisados nas baterias.

Destination

1. Mexican Riviera
2. Eastern Caribbean
3. Western Caribbean
4. Alaska
5. Norway and Northern Europe

6. Mediterranean

Cruise Line

1. Norwegian
2. Disney
3. Royal Caribbean
4. Princess
5. Holland America
6. Carnival

Number of Days

1. 7 days
2. 8 days
3. 9 days
4. 10 days
5. 11 days

Stateroom

1. Inside stateroom (no windows)
2. Oceanview stateroom, porthole window
3. Balcony stateroom, sliding door to private balcony

Ship Amenities

1. Fewer amenities.
2. More amenities.

Price per person per Day

1. $100 per person per day
2. $125 per person per day
3. $150 per person per day
4. $175 per person per day
5. $200 per person per day

Capítulo 5

Dados textuais

Pedro J. Fernandez, Paulo C. Marques F. e Hedibert F. Lopes

Neste capítulo, iremos trabalhar com um tipo especial de dado, os dados em forma de texto, ou dados textuais (*text data*), que são uma fonte valiosa de informações relevantes para diversos aspectos da vida social. Por exemplo, as opiniões de consumidores a respeito de produtos e serviços, expressas em redes sociais, afetam as chances de bons resultados de um negócio. De maneira análoga, 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. Entre as variantes de dados textuais, temos:

1. Pequenas mensagens do *Twitter* (*tweets*), com no máximo duzentos e quarenta caracteres;
2. Textos de tamanho intermediário, tais como avaliações de consumidores sobre livros, restaurantes, filmes etc;
3. Textos mais longos, tais como críticas profissionais de livros e filmes, editoriais e notícias em jornais ou sites, comunicações para acionistas, registros médicos, discursos políticos e decisões judiciais.

5.1 Elementos textuais

Em um conjunto de dados textuais cada unidade de informação é denominada *documento*. O conjunto de todos os documentos em que estamos interessados é denominado *corpus*.

No tipo de análise que iremos considerar, cada documento é reduzido a um conjunto de palavras, sem levar-se em conta a posição relativa das palavras dentro do documento, ou as estruturas sintáticas. Tal representação é denominada *sacola-de-palavras* (*bag-of-words*). Nesta representação, cada documento fica identificado a uma lista de *termos* e suas respectivas frequências de ocorrência no documento. Tais termos não são formados necessariamente por palavras individuais.

Na preparação de um conjunto de dados textuais, podemos identificar *tokens* (símbolos) que formam termos com mais de uma palavra. Por exemplo, a expressão "bem mal" pode ser considerada um único termo (um *bigrama*) formado pelas palavras "bem" e "mal", tendo um sentido diferente das palavras "bem" e "mal" consideradas separadamente. Por exemplo, considere as sentenças "Decidiu fazer o *bem* e combater o *mal*" e "Foi *bem mal* no exame". Do mesmo modo, podemos tratar "Ministério da Justiça" como um único termo (um *trigrama*). Em geral, um termo formado por n palavras é denominado *n-grama*.

Esta possibilidade de definirmos n-gramas aumenta substancialmente o poder expressivo da representação via sacola-de-palavras. Por exemplo, considere dois documentos, o primeiro deles contendo a frase "João foi à casa de Maria", e o segundo contendo a frase "Maria foi à casa de João". Do ponto de vista dos unigramas "João", "Maria", "foi", "à", "casa", "de", os dois documentos são indistinguíveis, apesar da evidente distinção semântica. No entanto, a definição dos trigramas "casa de João" e "casa de Maria" permite distinguir adequadamente os dois documentos.

5.2 Mensagens do *Twitter*

Uma boa maneira de explorar os conceitos básicos relacionados a dados textuais é minerar algumas mensagens do *Twitter* utilizando a biblioteca `twitteR`.

Além de possuir uma conta regular do *Twitter*, para rodar os exemplos a seguir é necessário criar uma conta (gratuita) de desenvolvedor. Nossa conta de exemplo está acessível em `https://twitter.com/BayesianFactory`.

O primeiro passo é conectar-se e realizar a autenticação. Abaixo, por uma questão de privacidade, ocultamos os valores das chaves de acesso. Utilize suas próprias chaves quando reproduzir os exemplos.

```
library(tidyverse)
library(twitteR)
library(ROAuth)

setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
```

Uma funcionalidade básica da biblioteca `twitteR`, que pode ser utilizada para a construção de "robôs", é enviar *Tweets* utilizando a função `tweet()`.

```
tweet("Testando!")
```

Também podemos verificar as *Twitter trends* de uma certa localidade, identificada pelo *woeid* (*Where On Earth IDentifier*). Para descobrir o *woeid* da localidade mais próxima a partir de uma latitude e uma longitude (por exemplo, as do bairro da Vila Olímpia, em São Paulo), utilizamos a função `closestTrendLocations()`.

```
closestTrendLocations(lat = -23.53, long = -46.67)
```

```
##        name country  woeid
## 1 São Paulo  Brazil 455827
```

As buscas a seguir foram feitas em Fevereiro de 2020. Estas eram as dez principais *Twitter trends* de São Paulo.

```
trends <- getTrends(woeid = 455827)
```

```
trends$name[1:10]
```

```
##  [1] "#TheVoiceKids" "#Globeleza"    "Tiago Nunes"   "#Carnaval2020"
##  [5] "Pyong"         "Exato"         "#ForaBianca"   "#DingDong"
##  [9] "Tatum"         "Celtics"
```

Utilizando a função `searchTwitter()` podemos fazer uma busca das mensagens mais recentes escritas em Português que contém a *hashtag* `Carnaval2020`.

```
results <- searchTwitter("#Carnaval2020", n = 1000, lang = "pt")
tweets <- twListToDF(results)
```

O objeto `tweets` é um `data.frame` com as seguintes colunas.

```
names(tweets)
```

```
##  [1] "text"          "favorited"     "favoriteCount" "replyToSN"
##  [5] "created"       "truncated"     "replyToSID"    "id"
##  [9] "replyToUID"    "statusSource"  "screenName"    "retweetCount"
## [13] "isRetweet"     "retweeted"     "longitude"     "latitude"
```

Este é o conteúdo do segundo *tweet*.

```
writeLines(strwrap(tweets$text[2], 80))

## RT @centraldotimao: Fala aí Fiel! Gostou do desfile? #CantaGavioes
```

A partir dos textos dos *tweets* criamos o *corpus* com as funções da biblioteca de *text mining* `tm` (veja [4]).

```
library(tm)

txt <- sapply(tweets$text,
              function(row) iconv(row, "UTF-8", "ASCII//TRANSLIT", sub = ""))

corpus <- Corpus(VectorSource(txt))
```

Antes de examinarmos o *corpus* criado, iremos transformá-lo, deixando todas as palavras em letras minúsculas e removendo: espaços em branco dobrados, números, palavras irrelevantes, pontuações e *stop words* (artigos, preposições etc).

```
corpus <- corpus %>%
    tm_map(content_transformer(tolower)) %>%
    tm_map(stripWhitespace) %>%
    tm_map(removeNumbers) %>%
    tm_map(removeWords, c("https", "rt", "via", "carnaval", "sobre")) %>%
    tm_map(removePunctuation) %>%
    tm_map(removeWords, iconv(stopwords("portuguese"),
                              from = "UTF-8", to = "ASCII//TRANSLIT"))
```

Na fase de preparação do *corpus*, uma transformação importante é o processo de *stemming*, pelo qual palavras derivadas de um radical comum são reduzidas a um único termo. A própria função `tm_map()` nos permite fazer o *stemming* do *corpus* da seguinte maneira.

```
corpus <- tm_map(corpus, stemDocument, language = "portuguese")
```

No entanto, não ficamos satisfeitos com os resultados deste algoritmo quando aplicado ao nosso idioma. Consequentemente, na seguinte *nuvem de palavras* temos o *corpus* antes do processo de *stemming*. Nesta nuvem de palavras, o tamanho da fonte de um termo é proporcional à sua frequência de ocorrência no *corpus*.

```
library(wordcloud)

wordcloud(corpus, max.words = 75, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))
```

Na representação de sacola-de-palavras, o *corpus* fica identificado matematicamente a uma *matriz documento-termo*, cujas linhas e colunas correspondem a documentos e termos, respectivamente, e as entradas da matriz são os números de ocorrências dos termos em cada documento.

```
dtm <- DocumentTermMatrix(corpus)
```

De posse desta matriz, podemos procurar os termos com ocorrência mais frequente no *corpus*, bem como examinar as associações entre os termos.

```
findFreqTerms(dtm, lowfreq = 40)
```

```
##  [1] "bloco"                 "acadmicosdevigriogeral" "bolsonaro"
##  [4] "carnavalrj"            "criticar"               "escola"
##  [7] "jairbolsonaro"         "plenonews"              "samba"
## [10] "vaiada"                "dia"                    "vai"
## [13] "pra"                   "hoje"                   "salvador"
## [16] "paga"
```

```
findAssocs(dtm, terms = "bolsonaro", corlimit = 0.3)

## $bolsonaro
## acadmicosdevigriogeral              criticar             plenonews
##                   0.97                  0.97                  0.97
##                 vaiada            carnavalrj          jairbolsonaro
##                   0.97                  0.96                  0.96
##                 escola                 samba
##                   0.93                  0.83
```

5.3 Modelo de tópicos

Suponha que você tenha em mãos um *corpus* cuja extensão torne a leitura de todos os documentos proibitiva. Por exemplo, você tem acesso a milhares de decisões judiciais de ações trabalhistas relacionadas a um certo setor da indústria, ou centenas de milhares de *e-mails* de uma grande corporação envolvida em esquemas de corrupção.

Um *modelo de tópicos probabilístico* é uma técnica de aprendizagem não supervisionada que permite agrupar os documentos do *corpus* de acordo com os seus conteúdos. Ou seja, com o modelo de tópicos criamos *clusters* de documentos levando em conta suas similaridades e dissimilaridades. A ideia é que há um certo número de *tópicos*, que são variáveis aleatórias latentes (não observadas), e que as palavras dos documentos têm uma distribuição de probabilidades que depende do tópico específico do documento ao qual elas pertencem.

O objetivo desta modelagem é obter a probabilidade *a posteriori* dos tópicos de cada documento, dados os conteúdos de todos os documentos do *corpus*. O modelo probabilístico dominante na área de modelagem de tópicos é a *Latent Dirichlet Allocation*, ou LDA [2]. Ao final do treinamento do modelo LDA, temos as palavras mais prováveis de cada tópico, bem como o tópico mais provável de cada documento do *corpus*.

5.4 Um *corpus* da BBC

Vamos analisar um *corpus* de documentos com notícias da BBC (`http://mlg.ucd.ie/datasets/bbc.htm`). Os documentos foram pré-classificados pela BBC por assunto (tópico). Para examinarmos o potencial de uma aprendizagem não supervisionada com estes dados, iremos "esquecer" os tópicos originais definidos pela BBC e tentaremos recuperá-los.

Dizemos que uma matriz é *esparsa* quando esta possui muitos elementos nulos; tenham estes elementos o valor 0, ou sejam apenas valores ausentes `NA`. Uma matriz esparsa é representada de maneira que tais valores nulos não sejam armazenados na memória. O armazenamento dos documentos da BBC está estruturado em torno de uma matriz esparsa, no formato `MatrixMarket`, que conecta as listas de termos e documentos do *corpus*. Para lermos esta matriz esparsa utilizamos a função `readMM()` da biblioteca `Matrix`.

```
library(Matrix)

mtx <- readMM("dados/textos/bbc.mtx")
```

A partir dessa matriz esparsa `mtx`, construímos a matriz documento-termo utilizando as funções da biblioteca `tm`.

```
library(tm)

tdm <- as.TermDocumentMatrix(mtx, weighting = weightTf)

dtm <- t(tdm)
dtm$dimnames$Terms <- scan("dados/textos/bbc.terms", what = "character")
dtm$dimnames$Docs <- scan("dados/textos/bbc.docs", what = "character")
```

Conforme discutido na seção anterior, cada linha da matriz `dtm` corresponde a um documento do *corpus* e cada coluna corresponde a um termo; e esta matriz armazena as frequências de ocorrências dos termos nos documentos. As dimensões de `dtm` quantificam o tamanho do *corpus* que estamos examinando: 2.225 documentos e 9.635 termos.

```
dim(dtm)

## [1] 2225 9635
```

Não podemos inspecionar os valores da matriz `dtm` diretamente, devido à forma como esta matriz é armazenada. A biblioteca `tm` fornece as funções para acessarmos os elementos de `dtm`. Por exemplo, estes são os primeiros 5 documentos e as frequências de 10 de seus termos.

```
inspect(dtm[1:5, 1:10])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 10)>>
## Non-/sparse entries: 19/31
## Sparsity           : 62%
## Maximal term length: 9
## Weighting          : term frequency (tf)
## Sample             :
##               Terms
## Docs           ad boost giant jump media profit quarterli sale time warner
##   business.001  1     2     1    1     1     10         1    5    3      4
##   business.002  0     1     0    0     0      0         0    0    2      0
##   business.003  0     0     1    0     0      0         0    4    0      0
##   business.004  0     0     0    0     0      4         1    1    0      0
##   business.005  0     0     1    0     0      0         0    0    1      0
```

Utilizando as conversões de tipos adequadas, podemos listar todos os termos de um documento, por exemplo, o documento `business.007`.

```
doc_terms <- as.matrix(dtm["business.007",])
print(sort(dtm$dimnames$Terms[doc_terms > 0]), quote = FALSE)
```

```
##  [1] 000        133        146        157        190        2001
##  [7] 2004       52         ad         administr  albeit     amount
## [13] analyst    boost      bush       celebr     chief      clearview
## [19] condit     continu    creat      creation   decemb     depart
## [25] deputi     dollar     econom     economi    economist  elect
## [31] employ     end        environ    expand     expect     fall
## [37] favour     fewer      figur      financi    fine       firm
## [43] first      gain       get        given      got        group
## [49] growth     herbert    interest   issu       januari    job
## [55] kei        ken        labor      level      limit      low
## [61] lowest     margin     market     mayland    mean       moder
## [67] net        novemb     number     offic      opportun   pace
## [73] payrol     posit      presid     presidenti produc     push
## [79] rate       record     rel        result     revis      rick
## [85] satisfi    septemb    slow       strong     suggest    term
## [91] territori  three      unemploy   valu       worker     year
```

Observando a convenção utilizada para nomear os documentos, podemos extrair os tópicos correspondentes e calcular a quantidade de documentos existentes em cada tópico.

```
document_topic <- sapply(strsplit(rownames(dtm), "[.]"), function(x) x[1])
table(document_topic)

## document_topic
##      business entertainment      politics         sport          tech
##           510           386           417           511           401
```

Explorando o *corpus*, estes são os 7 termos que ocorrem com maior frequência.

```
freq <- colSums(as.matrix(dtm))
print(format(sort(freq, decreasing = TRUE)[1:7], width = 9), quote = FALSE)

##      year     peopl        on      game      time     first     govern
##      2830      2044      1838      1640      1487      1283      1246
```

Podemos representar graficamente as frequências de ocorrências dos termos neste *corpus* da BBC por uma nuvem de palavras, lembrando que nessa nuvem o tamanho da fonte de cada termo é proporcional à sua frequência de ocorrência no *corpus*.

```
library(wordcloud)

wordcloud(names(freq), freq, max.words = 75, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))
```

Note que as palavras deste *corpus* já foram submetidas ao processo de *stemming*. Por exemplo, as palavras *company* e *companies* foram reduzidas ao termo *compani*. Para cada tópico, estes são os termos mais frequentes.

```
for (topic in unique(document_topic)) {
    cat(topic, ":\n", sep = "")
    print(format(sort(colSums(as.matrix(dtm)[grepl(topic, rownames(dtm)),]),
                      decreasing = TRUE)[1:6], width = 9), quote = FALSE)
}
```

```
## business:
##      year   compani      firm    market      bank      sale
##       884       627       557       539       459       414
## entertainment:
##      film      year      best     music     award      star
##       964       594       590       540       522       429
## politics:
##    labour    govern     parti     elect     peopl     blair
##       760       759       709       670       623       575
## sport:
##      game      plai       win    player   england   against
##       648       624       590       474       459       454
## tech:
##     peopl      game technolog     mobil     phone        on
##       960       875       631       595       540       519
```

Agora, suponha que "perdemos" a informação sobre os tópicos originais a que pertencem os documentos deste *corpus* da BBC. Utilizando a biblioteca `topicmodels`, iremos treinar um modelo LDA com 5 tópicos.

```
library(topicmodels)

model <- LDA(dtm, k = 5, method = "Gibbs",
             control = list(seed = 4321, burnin = 250, thin = 2, iter = 1000))
```

Estes são os 10 termos com maior probabilidade de ocorrência em cada tópico formado pelo modelo LDA.

```
print(terms(model, 10), quote = FALSE)
```

```
##       Topic 1 Topic 2 Topic 3   Topic 4 Topic 5
##  [1,] year    film    peopl     plai    govern
```

```
##  [2,] compani year    game      game    labour
##  [3,] market  best    servic    win     parti
##  [4,] sale    show    technolog england elect
##  [5,] firm    award   mobil     against peopl
##  [6,] share   includ  phone     first   plan
##  [7,] expect  on      on        back    minist
##  [8,] bank    music   get       player  sai
##  [9,] month   top     work      two     told
## [10,] price   star    user      time    blair
```

Estes são os tópicos mais prováveis para os cinco primeiros documentos do *corpus*.

```
topics(model, 3)[, 1:5]
```

```
##      business.001 business.002 business.003 business.004 business.005
## [1,]            1            1            1            1            1
## [2,]            3            5            5            4            5
## [3,]            2            3            4            3            2
```

Note que para o modelo LDA os tópicos são apenas rótulos (`Topic 1`, `Topic 2` etc), sem nenhum significado especial. No entanto, voltando aos tópicos originais definidos pela BBC, conseguimos estabelecer uma correspondência entre os rótulos do modelo LDA e os tópicos reais.

```
true_topic <- as.factor(document_topic)
names(true_topic) <- rownames(dtm)
predicted_topic <- factor(topics(model),
                          levels = c(1, 2, 5, 4, 3),
                          labels = c("business", "entertainment", "politics",
                                     "sport", "tech"),
                          ordered = TRUE)
confusion <- table(predicted_topic, true_topic,
                   dnn = c("Predicted Topic", "True Topic"))
print(confusion)
```

```
##                 True Topic
## Predicted Topic business entertainment politics sport tech
##   business           475             4        9     2    7
##   entertainment        3           363        5     2   12
##   politics            24            14      400     1    9
##   sport                0             0        2   506    3
##   tech                 8             5        1     0  370
```

Deste modo, conseguimos responder a questão crucial: quantos documentos foram alocados corretamente nos tópicos originais? O percentual de alocação correta é impressionante.

```
(sprintf("Alocados corretamente: %.2f%%\n",
         100 * (sum(diag(confusion)) / sum(confusion))))
```

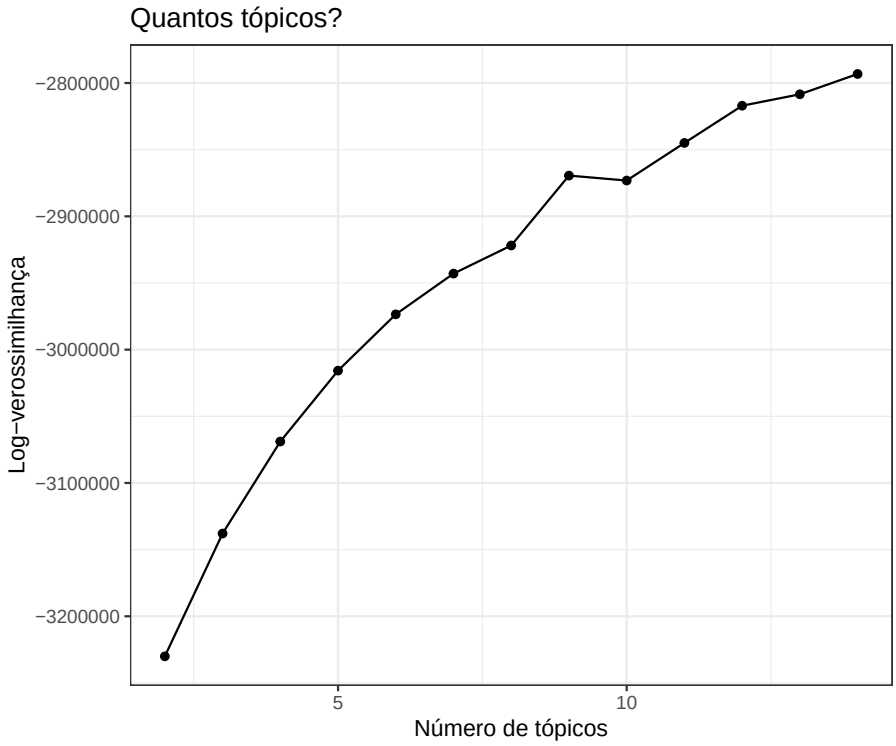
```
## [1] "Alocados corretamente: 95.01%\n"
```

Em uma análise de *clusters*, determinar o número adequado de *clusters* é sempre uma questão delicada. Em geral, é necessário examinar os *clusters* obtidos para decidir se o número é adequado. Em certos problemas, o número de *clusters* é definido por questões de natureza prática: por exemplo, o número de especialistas que irão analisar os documentos dos *clusters* criados.

Uma técnica muito comum que ampara a decisão sobre o número de *clusters* é treinar vários modelos LDA com números de tópicos distintos e procurar um "cotovel" na curva definida pelas log-verossimilhanças dos modelos LDA treinados. Podemos apreciar a funcionalidade deste método utilizando o *corpus* da BBC, para o qual sabemos que temos cinco tópicos.

```
k_range <- 2:14
log_L <- numeric()
for (k in k_range) {
    mdl <- LDA(dtm, k = k, method = "Gibbs",
               control = list(seed = 4321, burnin = 250, thin = 2, iter = 1000))
    log_L <- c(log_L, mdl@loglikelihood)
}
```

```
qplot(k_range, log_L, type = "b", geom = c("point", "line"),
      xlab = "Número de tópicos", ylab = "Log-verossimilhança",
      main = "Quantos tópicos?")
```



No gráfico acima, parece razoável encontrar o ”cotovelo”por volta de 5 ou 6 tópicos, o que coincide com a realidade dos dados.

Finalizamos com a observação de que o Modelo de Tópicos também é utilizado quando os *tokens* não correspondem a palavras de uma linguagem natural. Por exemplo, há aplicações do Modelo de Tópicos em Genética, na identificação de *clusters* de genes que são marcadores de alterações fisiológicas.

Anexo

Documento número 1 do *corpus* da BBC. O texto abaixo foi classificado pela BBC como pertencente ao tópico *business*.

Ad sales boost Time Warner profit

Quarterly profits at US media giant TimeWarner jumped 76% to $1.13bn (600m pounds) for the three months to December, from $639m year-earlier.

The firm, which is now one of the biggest investors in Google, benefited from sales of high-speed internet connections and higher advert sales. TimeWarner said fourth quarter sales rose 2% to $11.1bn from $10.9bn. Its profits were buoyed by one-off gains which offset a profit dip at Warner Bros, and less users for AOL.

Time Warner said on Friday that it now owns 8% of search-engine Google. But its own internet business, AOL, had has mixed fortunes. It lost 464,000 subscribers in the fourth quarter profits were lower than in the preceding three quarters. However, the company said AOL's underlying profit before exceptional items rose 8% on the back of stronger internet advertising revenues. It hopes to increase subscribers by offering the online service free to TimeWarner internet customers and will try to sign up AOL's existing customers for high-speed broadband. TimeWarner also has to restate 2000 and 2003 results following a probe by the US Securities Exchange Commission (SEC), which is close to concluding.

Time Warner's fourth quarter profits were slightly better than analysts' expectations. But its film division saw profits slump 27% to $284m, helped by box-office flops Alexander and Catwoman, a sharp contrast to year-earlier, when the third and final film in the Lord of the Rings trilogy boosted results. For the full-year, TimeWarner posted a profit of $3.36bn, up 27% from its 2003 performance, while revenues grew 6.4% to $42.09bn. "Our financial performance was strong, meeting or exceeding all of our full-year objectives and greatly enhancing our flexibility," chairman and chief executive Richard Parsons said. For 2005, TimeWarner is projecting operating earnings growth of around 5%, and also expects higher revenue and wider profit margins.

TimeWarner is to restate its accounts as part of efforts to resolve an inquiry into AOL by US market regulators. It has already offered to pay $300m to settle charges, in a deal that is under review by the SEC. The company said it was unable to estimate the amount it needed to set aside for legal reserves, which it previously set at $500m. It intends to adjust the way it accounts for a deal with German music publisher Bertelsmann's purchase of a stake in AOL Europe, which it had reported as advertising revenue. It will now book the sale of its stake in AOL Europe as a loss on the value of that stake.

Referências

[1] R. Agrawal e R. Srikant. "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th VLDB Conference*, Santiago, Chile (1994)

[2] D. Blei, A. Ng e M. Jordan. "Latent Dirichlet Allocation", *Journal of Machine Learning Research*, **3**, 993 (2003).

[3] I. Borg e P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*, segunda edição, Springer (2005).

[4] I. Feinerer, K. Hornik e D. Meyer. "Text Mining Infrastructure in R", *Journal of Statistical Software*, **25**, 1 (2008).

[5] P.J. Fernandez e V. Yohai. *Introdução à Análise Exploratória de Dados Multivariados*, IMPA (2014). `https://impa.br/wp-content/uploads/2017/04/PM_41.pdf`

[6] T. Hastie, R. Tibshirani e J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, segunda edição, Springer (2016)

[7] P. Jaccard. "Étude Comparative de la Distribution Florale dans une Portion des Alpes et des Jura", *Bulletin de la Société Vaudoise des Sciences Naturelles*, **37**, 547 (1901).

[8] P. Jaccard. "The Distribution of the Flora in the Alpine Zone", *New Phytologist*, **11**, 37 (1912).

[9] G. James, D. Witten, T. Hastie e R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R*, Springer (2017).

[10] H.F. Kaiser. "A Note on Guttman's Lower Bound for the Number of Common Factors", *British Journal of Statistical Psychology*, **14**, 1 (1961).

[11] J.B. Kruskal. "Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis", *Psychometrika*, **29**, 1 (1964).

[12] J.B. Kruskal. "Nonmetric Multidimensional Scaling: a Numerical Method", *Psychometrika*, **29**, 28 (1964).

[13] K.V. Mardia, J.T. Kent e J.M. Bibby. *Multivariate Analysis*, Academic Press (1979).

[14] D. McFadden. “Conditional Logit Analysis of Qualitative Choice Behavior”, *Frontiers in Econometrics*, 105 (1974).

[15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2019).

[16] P.E. Rossi, G.M. Allenby e R. McCulloch. *Bayesian Statistics and Marketing*, Wiley (2005).

[17] R.N. Shepard. “The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. I”, *Psychometrika*, **27**, 125 (1962).

[18] R.N. Shepard. “The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. II”, *Psychometrika*, **27**, 219 (1962).

[19] K. Train. *Quantitative Choice Analysis*, MIT Press (1986).

[20] H. Wickham e G. Grolemund. *R for Data Science*, O’Reilly Media (2017).