cap4

Washington Candeia

May 15, 2018

Contents

1 Capítulo 4 - Notação R

Agoa que você tem um baralho de cartas, é preciso uma forma de fazer coisas com isso, do tipo que se faz com cartas em um baralho.
Primeiro, você vai querer reembaralhar as cartas de tempos em tempos. Depois, você irá querer jogar as cartas do baralho (uma carta por vez).

Para fazer estas coisas, você precisará trabalhar com valores individuais dentro de seu *data frame*, uma tarefa essencial em *data science*. Por exemplo, para jogar as cartas do topo de seu baralho, você precisará escrever uma função que seleciona a primeira linha de valores em seu *data frame*, como abaixo:

```
# Load data
deck <- read.csv("./deck.csv", stringsAsFactors = FALSE)

# Transformando em data frame
df <- data.frame(deck, stringsAsFactors = FALSE)

# Salvar arquivo
write.csv(deck, file = "cards2.csv", row.names = FALSE)

# Observando o conteúdo do data frame com função str
str(df)
```

```
## 'data.frame':    52 obs. of  3 variables:
##  $ face : chr  "king" "queen" "jack" "ten" ...
##  $ suit : chr  "spades" "spades" "spades" "spades" ...
##  $ value: int  13 12 11 10 9 8 7 6 5 4 ...
```

```
df
```

```
##     face     suit value
## 1   king   spades    13
## 2  queen   spades    12
## 3   jack   spades    11
## 4    ten   spades    10
## 5   nine   spades     9
## 6  eight   spades     8
## 7  seven   spades     7
## 8    six   spades     6
## 9   five   spades     5
## 10  four   spades     4
## 11 three   spades     3
## 12   two   spades     2
## 13   ace   spades     1
## 14  king    clubs    13
## 15 queen    clubs    12
## 16  jack    clubs    11
## 17   ten    clubs    10
## 18  nine    clubs     9
## 19 eight    clubs     8
## 20 seven    clubs     7
## 21   six    clubs     6
## 22  five    clubs     5
## 23  four    clubs     4
## 24 three    clubs     3
## 25   two    clubs     2
## 26   ace    clubs     1
## 27  king diamonds    13
## 28 queen diamonds    12
## 29  jack diamonds    11
## 30   ten diamonds    10
## 31  nine diamonds     9
## 32 eight diamonds     8
## 33 seven diamonds     7
## 34   six diamonds     6
## 35  five diamonds     5
## 36  four diamonds     4
## 37 three diamonds     3
## 38   two diamonds     2
## 39   ace diamonds     1
## 40  king   hearts    13
## 41 queen   hearts    12
## 42  jack   hearts    11
## 43   ten   hearts    10
## 44  nine   hearts     9
## 45 eight   hearts     8
## 46 seven   hearts     7
## 47   six   hearts     6
## 48  five   hearts     5
## 49  four   hearts     4
## 50 three   hearts     3
## 51   two   hearts     2
## 52   ace   hearts     1
```

```
#Atributos
attributes(df)
```

```
## $names
## [1] "face"  "suit"  "value"
##
## $row.names
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
## [47] 47 48 49 50 51 52
##
## $class
## [1] "data.frame"
```

```
unclass(df)
```

```
## $face
##  [1] "king"  "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"
##  [9] "five"  "four"  "three" "two"   "ace"   "king"  "queen" "jack"
## [17] "ten"   "nine"  "eight" "seven" "six"   "five"  "four"  "three"
## [25] "two"   "ace"   "king"  "queen" "jack"  "ten"   "nine"  "eight"
## [33] "seven" "six"   "five"  "four"  "three" "two"   "ace"   "king"
## [41] "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"   "five"
## [49] "four"  "three" "two"   "ace"
##
## $suit
##  [1] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
##  [7] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
## [13] "spades"   "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [19] "clubs"    "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [25] "clubs"    "clubs"    "diamonds" "diamonds" "diamonds" "diamonds"
## [31] "diamonds" "diamonds" "diamonds" "diamonds" "diamonds" "diamonds"
## [37] "diamonds" "diamonds" "diamonds" "hearts"   "hearts"   "hearts"
## [43] "hearts"   "hearts"   "hearts"   "hearts"   "hearts"   "hearts"
## [49] "hearts"   "hearts"   "hearts"   "hearts"
##
## $value
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
##
## attr(,"row.names")
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
## [47] 47 48 49 50 51 52
```

Na página 57 do livro, é possível encontrar o endereço para baixar o arquivo ***deck*** [http://bit.ly/deck_CSV].

Para extrair valores de um data frame, escreva o nome do data frame seguido de colchetes:

```
# Slice
df[,]      # Todas as linhas e todas as colunas
```

```
##     face     suit value
## 1   king   spades    13
## 2  queen   spades    12
## 3   jack   spades    11
```

```
## 4    ten   spades    10
## 5   nine   spades     9
## 6  eight   spades     8
## 7  seven   spades     7
## 8    six   spades     6
## 9   five   spades     5
## 10  four   spades     4
## 11 three   spades     3
## 12   two   spades     2
## 13   ace   spades     1
## 14  king    clubs    13
## 15 queen    clubs    12
## 16  jack    clubs    11
## 17   ten    clubs    10
## 18  nine    clubs     9
## 19 eight    clubs     8
## 20 seven    clubs     7
## 21   six    clubs     6
## 22  five    clubs     5
## 23  four    clubs     4
## 24 three    clubs     3
## 25   two    clubs     2
## 26   ace    clubs     1
## 27  king diamonds    13
## 28 queen diamonds    12
## 29  jack diamonds    11
## 30   ten diamonds    10
## 31  nine diamonds     9
## 32 eight diamonds     8
## 33 seven diamonds     7
## 34   six diamonds     6
## 35  five diamonds     5
## 36  four diamonds     4
## 37 three diamonds     3
## 38   two diamonds     2
## 39   ace diamonds     1
## 40  king   hearts    13
## 41 queen   hearts    12
## 42  jack   hearts    11
## 43   ten   hearts    10
## 44  nine   hearts     9
## 45 eight   hearts     8
## 46 seven   hearts     7
## 47   six   hearts     6
## 48  five   hearts     5
## 49  four   hearts     4
## 50 three   hearts     3
## 51   two   hearts     2
## 52   ace   hearts     1
```

```r
df[1,]     # Linha 1, valores de todas as colunas
```

```
##   face   suit value
## 1 king spades    13
```

```
df[, 1]    # Somente coluna 1
```

```
##  [1] "king"  "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"
##  [9] "five"  "four"  "three" "two"   "ace"   "king"  "queen" "jack"
## [17] "ten"   "nine"  "eight" "seven" "six"   "five"  "four"  "three"
## [25] "two"   "ace"   "king"  "queen" "jack"  "ten"   "nine"  "eight"
## [33] "seven" "six"   "five"  "four"  "three" "two"   "ace"   "king"
## [41] "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"   "five"
## [49] "four"  "three" "two"   "ace"
```

```
df[,2]     # Somente coluna 3
```

```
##  [1] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
##  [7] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
## [13] "spades"   "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [19] "clubs"    "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [25] "clubs"    "clubs"    "diamonds" "diamonds" "diamonds" "diamonds"
## [31] "diamonds" "diamonds" "diamonds" "diamonds" "diamonds" "diamonds"
## [37] "diamonds" "diamonds" "diamonds" "hearts"   "hearts"   "hearts"
## [43] "hearts"   "hearts"   "hearts"   "hearts"   "hearts"   "hearts"
## [49] "hearts"   "hearts"   "hearts"   "hearts"
```

```
df[14,2]   # Linha 14 coluna 2
```

```
## [1] "clubs"
```

```
df[1:3, 2] # Da linha 1 à 3, da coluna 2
```

```
## [1] "spades" "spades" "spades"
```

São os **índices** (*indexes*) que dizem a R aonde ir, o que buscar, que valores retornar.

> nome_objeto_data[linha, coluna]

Além disso, há diferentes formas de criar índices em R:

> Inteiros positivos Inteiros negativos
> Zero
> Espaços em branco
> Valores lógicos
> Nomes

1.1 Inteiros Positivos

```
# Usando diferentes formas de buscar valores por índices
deck[14,2]           # Linha 14 coluna 2
```

```
## [1] "clubs"
```

```
deck[1, 2]           # Linha 1 coluna 2
```

```
## [1] "spades"
```

```
# Com vetores
deck[1, c(1, 2, 3)]  # Linha 1, colunas 1, 2 e 3
```

```
##   face   suit value
## 1 king spades    13
```

```
deck[2, c(1, 3)]     # Linha 2, colunas 1 e 3
```

```
##    face value
## 2 queen    12
```

Obs.: https://rmarkdown.rstudio.com/pdf_document_format.html

```
# drop = FALSE
deck[1:2, 1:2]    # Retorna um data frame
```

```
##    face   suit
## 1  king spades
## 2 queen spades
```

```
deck[1:2, 1]      # Retorna um vetor
```

```
## [1] "king"  "queen"
```

```
deck[1:2, 2]      # Retorna um vetor
```

```
## [1] "spades" "spades"
```

```
deck[1:2, 3]      # Retorna um vetor
```

```
## [1] 13 12
```

```
# Para retornar um vetor, mesmo quando se coloca uma simples coluna
# utiliza-se o argumento drop = FALSE
deck[1:2, 1, drop = FALSE]  # Retorna um data frame
```

```
##    face
## 1  king
## 2 queen
```

```
deck[1:2, 1]               # Retorna um vetor
```

```
## [1] "king"  "queen"
```

```
deck[1:6, 2, drop = FALSE]  # Retorna um data frame
```

```
##     suit
## 1 spades
## 2 spades
## 3 spades
## 4 spades
## 5 spades
## 6 spades
```

1.2 Inteiros negativos

Fazem o oposto dos inteiros positivos. Quando se coloca um inteiro negativo em um slice ou index, o interpretador R excluirá estes últimos itens apontados. Ou seja, R retornará todos os elementos, exceto, os elementos apontados no índices negativos.

```
deck[-1, 1:3]         # Exceto a linha 1, todas as linhas restantes e colunas
```

```
##     face     suit value
## 2  queen   spades    12
## 3   jack   spades    11
## 4    ten   spades    10
```

```
## 5   nine   spades     9
## 6  eight   spades     8
## 7  seven   spades     7
## 8    six   spades     6
## 9   five   spades     5
## 10  four   spades     4
## 11 three   spades     3
## 12   two   spades     2
## 13   ace   spades     1
## 14  king    clubs    13
## 15 queen    clubs    12
## 16  jack    clubs    11
## 17   ten    clubs    10
## 18  nine    clubs     9
## 19 eight    clubs     8
## 20 seven    clubs     7
## 21   six    clubs     6
## 22  five    clubs     5
## 23  four    clubs     4
## 24 three    clubs     3
## 25   two    clubs     2
## 26   ace    clubs     1
## 27  king diamonds    13
## 28 queen diamonds    12
## 29  jack diamonds    11
## 30   ten diamonds    10
## 31  nine diamonds     9
## 32 eight diamonds     8
## 33 seven diamonds     7
## 34   six diamonds     6
## 35  five diamonds     5
## 36  four diamonds     4
## 37 three diamonds     3
## 38   two diamonds     2
## 39   ace diamonds     1
## 40  king   hearts    13
## 41 queen   hearts    12
## 42  jack   hearts    11
## 43   ten   hearts    10
## 44  nine   hearts     9
## 45 eight   hearts     8
## 46 seven   hearts     7
## 47   six   hearts     6
## 48  five   hearts     5
## 49  four   hearts     4
## 50 three   hearts     3
## 51   two   hearts     2
## 52   ace   hearts     1
```

```
deck[-(2:52), 1:3]    # A linha 1 exceptuando-se as outras, e todas as colunas
```

```
##   face   suit value
## 1 king spades    13
```

1.3 Zero

Usar um zero como índice cria um objeto vazio.

1.4 Espaços em Branco

Espaços em branco são usados para especificar que se pode utilizar **todos** os indices daquela posição. Isto significa todas as linhas ou todas as colunas.

```
deck[ , 3]   # Todas as linhas, itens da coluna 3
```

```
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
```

```
deck[52, ]   # Apenas a linha 52, todas as colunas
```

```
##    face   suit value
## 52  ace hearts     1
```

```
deck[ , 2]   # Todos as linhas da coluna 2
```

```
##  [1] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
##  [7] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
## [13] "spades"   "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [19] "clubs"    "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [25] "clubs"    "clubs"    "diamonds" "diamonds" "diamonds" "diamonds"
## [31] "diamonds" "diamonds" "diamonds" "diamonds" "diamonds" "diamonds"
## [37] "diamonds" "diamonds" "diamonds" "hearts"   "hearts"   "hearts"
## [43] "hearts"   "hearts"   "hearts"   "hearts"   "hearts"   "hearts"
## [49] "hearts"   "hearts"   "hearts"   "hearts"
```

1.5 Valores Lógicos

Se um vetor contiver valores lógicos, TRUE ou FALSE, como índice, R atribuirá cada um destes a uma linha do data frame (ou da coluna, dependendo de que index se usa). Assim, R retornará cada linha correspondente a TRUE e deixará de lado quaisquer outras que tenham FALSE como correspondência.

```
deck[1, c(1, 2, 3)]            # Com inteiros
```

```
##   face   suit value
## 1 king spades    13
```

```
deck[1, c(TRUE, TRUE, FALSE)]  # Com lógicos
```

```
##   face   suit
## 1 king spades
```

1.6 Nomes

Também é possível obter as entradas ou valores do data frame utilizando nomes, como *strings* de caracteres.

```
deck[1, c("face", "suit", "value")]
```

```
##   face   suit value
## 1 king spades    13
```

```r
# O mesmo que
deck[1, c(1, 2, 3)]            # Linha 1, todas as três colunas
```

```
##   face   suit value
## 1 king spades    13
```

```r
deck[1, c(TRUE, TRUE, TRUE)]   # Linha 1, todas as três colunas
```

```
##   face   suit value
## 1 king spades    13
```

```r
deck[1, 1:3]                   # Linha 1, todas as três colunas
```

```
##   face   suit value
## 1 king spades    13
```

```r
# Apenas um deles
deck[ , "value"]               # Todas as entradas (linhas) na coluna value (3)
```

```
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
```

```r
deck[ , "suit"]                # Todas as entradas (linhas) na coluna suit(2)
```

```
##  [1] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
##  [7] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
## [13] "spades"   "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [19] "clubs"    "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [25] "clubs"    "clubs"    "diamonds" "diamonds" "diamonds" "diamonds"
## [31] "diamonds" "diamonds" "diamonds" "diamonds" "diamonds" "diamonds"
## [37] "diamonds" "diamonds" "diamonds" "hearts"   "hearts"   "hearts"
## [43] "hearts"   "hearts"   "hearts"   "hearts"   "hearts"   "hearts"
## [49] "hearts"   "hearts"   "hearts"   "hearts"
```

2 Retire uma Carta (Deal a Card)

Exercício da página 70 - 71

2.1 Exercício

Complete o seguinte código para fazer uma função que retorna a primeira linha de um data frame:

```r
# função para completar:
deal <- function(cards){
  #?
}

# Complete:
deal <- function(cards){
  first_card  <- cards[1, ]
  print(first_card)
}
```

```r
# Usando a função
deal(deck)
```

```
##   face   suit value
## 1 king spades    13
```

Porém, toda vez que usarmos a função `deal` ela sempre trará o rei de espadas, com valor 13. É preciso embaralhar (*shuffling*) estas cartas.

3 Embaralhe o Baralho (Shuffle the Deck)

Para embaralhar é preciso rearranjar de forma aleatória as cartas em um baralho.
Neste nosso cojunto de dados, cada carta é **uma linha** no *data frame*, e cada coluna trás uma informação desta carta. Para embaralhá-las a solução é reordenar as **linhas** deste *data frame* de forma aleatória.

Lembrando:

Usando argumento `replace` na função `sample` é uma forma de conseguir utilizá-la para obter duas amostras independentes. Assim, é como se jogássemos o dado duas ou mais vezes (lembrando de especificar como argumento em `size` nesta função).

```r
# Função sample
dado <- c(1, 2, 3, 4, 5, 6)
sample(dado, size = 2, replace = TRUE)
```

```
## [1] 3 5
```

Usar a função `sample` evita ter que embaralhar carta por carta salvando em diferentes variáveis, conforme o livro exemplifica.

```r
# Veja que é possível randomizar qualquer valor
# Você poderia criar as variáveis:
deck2 <- deck[1:52, ]   # 52 cartas (linhas), todas as colunas (informações das cartas)
deck2
```

```
##     face     suit value
## 1   king   spades    13
## 2  queen   spades    12
## 3   jack   spades    11
## 4    ten   spades    10
## 5   nine   spades     9
## 6  eight   spades     8
## 7  seven   spades     7
## 8    six   spades     6
## 9   five   spades     5
## 10  four   spades     4
## 11 three   spades     3
## 12   two   spades     2
## 13   ace   spades     1
## 14  king    clubs    13
## 15 queen    clubs    12
## 16  jack    clubs    11
## 17   ten    clubs    10
## 18  nine    clubs     9
## 19 eight    clubs     8
## 20 seven    clubs     7
```

```
## 21   six    clubs     6
## 22  five    clubs     5
## 23  four    clubs     4
## 24 three    clubs     3
## 25   two    clubs     2
## 26   ace    clubs     1
## 27  king diamonds    13
## 28 queen diamonds    12
## 29  jack diamonds    11
## 30   ten diamonds    10
## 31  nine diamonds     9
## 32 eight diamonds     8
## 33 seven diamonds     7
## 34   six diamonds     6
## 35  five diamonds     5
## 36  four diamonds     4
## 37 three diamonds     3
## 38   two diamonds     2
## 39   ace diamonds     1
## 40  king   hearts    13
## 41 queen   hearts    12
## 42  jack   hearts    11
## 43   ten   hearts    10
## 44  nine   hearts     9
## 45 eight   hearts     8
## 46 seven   hearts     7
## 47   six   hearts     6
## 48  five   hearts     5
## 49  four   hearts     4
## 50 three   hearts     3
## 51   two   hearts     2
## 52   ace   hearts     1
```

```
# Escolhendo a segunda carta (linha), depois a primeira, por fim as útlimas 52
# E todas as colunas
deck3 <- deck[c(2, 1, 3:52), ]
deck3
```

```
##     face     suit value
## 2  queen   spades    12
## 1   king   spades    13
## 3   jack   spades    11
## 4    ten   spades    10
## 5   nine   spades     9
## 6  eight   spades     8
## 7  seven   spades     7
## 8    six   spades     6
## 9   five   spades     5
## 10  four   spades     4
## 11 three   spades     3
## 12   two   spades     2
## 13   ace   spades     1
## 14  king    clubs    13
## 15 queen    clubs    12
## 16  jack    clubs    11
```

```
## 17   ten    clubs    10
## 18  nine    clubs     9
## 19 eight    clubs     8
## 20 seven    clubs     7
## 21   six    clubs     6
## 22  five    clubs     5
## 23  four    clubs     4
## 24 three    clubs     3
## 25   two    clubs     2
## 26   ace    clubs     1
## 27  king diamonds    13
## 28 queen diamonds    12
## 29  jack diamonds    11
## 30   ten diamonds    10
## 31  nine diamonds     9
## 32 eight diamonds     8
## 33 seven diamonds     7
## 34   six diamonds     6
## 35  five diamonds     5
## 36  four diamonds     4
## 37 three diamonds     3
## 38   two diamonds     2
## 39   ace diamonds     1
## 40  king   hearts    13
## 41 queen   hearts    12
## 42  jack   hearts    11
## 43   ten   hearts    10
## 44  nine   hearts     9
## 45 eight   hearts     8
## 46 seven   hearts     7
## 47   six   hearts     6
## 48  five   hearts     5
## 49  four   hearts     4
## 50 three   hearts     3
## 51   two   hearts     2
## 52   ace   hearts     1
```

Logo, é possível aleatorizar os valores das cartas - linhas - passando a variável `random` que contém todas as 52 cartas, como índice de `deck`. Assim, cada valor que estiver aleatorizado em `random` será uma pisição no índice de `deck`.

```
# Criando uma variável random para armazenar cartas
random <- sample(1:52, size = 52)   # Das cartas 1 a 52, tamanho 52, pois são 52 amostras por vez

# Aleatorizando
deck4 <- deck[random, ]    # Os valores de random são passados como indices de linhas
deck4
```

```
##     face     suit value
## 40  king   hearts    13
## 8    six   spades     6
## 31  nine diamonds     9
## 48  five   hearts     5
## 41 queen   hearts    12
## 43   ten   hearts    10
```

```
## 10  four   spades     4
## 35  five diamonds     5
## 6  eight   spades     8
## 1   king   spades    13
## 16  jack    clubs    11
## 22  five    clubs     5
## 13   ace   spades     1
## 38   two diamonds     2
## 29  jack diamonds    11
## 47   six   hearts     6
## 19 eight    clubs     8
## 11 three   spades     3
## 36  four diamonds     4
## 2  queen   spades    12
## 33 seven diamonds     7
## 24 three    clubs     3
## 15 queen    clubs    12
## 23  four    clubs     4
## 3   jack   spades    11
## 32 eight diamonds     8
## 42  jack   hearts    11
## 45 eight   hearts     8
## 20 seven    clubs     7
## 26   ace    clubs     1
## 46 seven   hearts     7
## 4    ten   spades    10
## 21   six    clubs     6
## 52   ace   hearts     1
## 18  nine    clubs     9
## 27  king diamonds    13
## 51   two   hearts     2
## 34   six diamonds     6
## 14  king    clubs    13
## 12   two   spades     2
## 30   ten diamonds    10
## 25   two    clubs     2
## 44  nine   hearts     9
## 50 three   hearts     3
## 5   nine   spades     9
## 17   ten    clubs    10
## 39   ace diamonds     1
## 9   five   spades     5
## 28 queen diamonds    12
## 49  four   hearts     4
## 37 three diamonds     3
## 7  seven   spades     7
```

3.1 Exercício, p.73

Use as ideias anteriores para escrever uma função *shuffle*. Ela deve pegar um data frame e retornar uma cópia embaralhada deste data frame.

```r
# Função embaralhar
# Para isso, usar sample
# Embaralhar 52 cartas, 52 como tamanho independente
# cards é o input da função, o argumento para o data frame colocado
# random contém os valores embaralhados de indices
# Cada indice é uma linha, então, mantem-se todas as colunas
shuffle <- function(cartas){
  random <- sample(1:52, size = 52)
  cartas[random, ]
}

shuffle(deck)
```

```
##     face     suit value
## 10  four   spades     4
## 29  jack diamonds    11
## 40  king   hearts    13
## 19 eight    clubs     8
## 36  four diamonds     4
## 43   ten   hearts    10
## 13   ace   spades     1
## 25   two    clubs     2
## 28 queen diamonds    12
## 7  seven   spades     7
## 12   two   spades     2
## 3   jack   spades    11
## 20 seven    clubs     7
## 49  four   hearts     4
## 5   nine   spades     9
## 34   six diamonds     6
## 21   six    clubs     6
## 17   ten    clubs    10
## 6  eight   spades     8
## 26   ace    clubs     1
## 35  five diamonds     5
## 27  king diamonds    13
## 50 three   hearts     3
## 33 seven diamonds     7
## 45 eight   hearts     8
## 44  nine   hearts     9
## 24 three    clubs     3
## 9   five   spades     5
## 38   two diamonds     2
## 37 three diamonds     3
## 51   two   hearts     2
## 52   ace   hearts     1
## 31  nine diamonds     9
## 42  jack   hearts    11
## 11 three   spades     3
## 30   ten diamonds    10
## 14  king    clubs    13
## 41 queen   hearts    12
## 18  nine    clubs     9
## 46 seven   hearts     7
```

```
## 1   king   spades    13
## 16  jack    clubs    11
## 48  five   hearts     5
## 8    six   spades     6
## 39   ace diamonds     1
## 4    ten   spades    10
## 15 queen    clubs    12
## 23  four    clubs     4
## 22  five    clubs     5
## 47   six   hearts     6
## 32 eight diamonds     8
## 2  queen   spades    12
```

Então, se fosse simular um jogo de cartas já temos duas coisas:

1. A função `deal` para escolher a carta que está em cima do monte de baralhos;
2. A função `shuffle` para embaralhar antes que formos retirar uma carta com `deal`;

```
# Retire uma carta
deal(deck)
```

```
##   face   suit value
## 1 king spades    13
```

```
# Embaralhe
deck5 <- shuffle(deck)

# Retire outra carta
deal(deck5)
```

```
##    face   suit value
## 48 five hearts     5
```

3.2 Dollar Signs ($) e Colchetes Duplos ([[]])

Dois objetos em R aceitam um segundo sistema de notação. Pode-se extrari valores de *data frames* e listas utilizando a sintaxe **$**.

3.2.1 Dollar Signs $

Por exemplo, para selecionar uma coluna em um data frame é só usar o nome do data.frame**$**coluna.

```
# Notação com uso de index
deck[, "value"]   # Ultima coluna
```

```
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
```

```
deck[ , 3]        # Ultima coluna
```

```
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
```

```
# Notação com sinal de dólar
deck$value        # Ultima coluna
```

```
##  [1] 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7  6  5  4
## [24]  3  2  1 13 12 11 10  9  8  7  6  5  4  3  2  1 13 12 11 10  9  8  7
## [47]  6  5  4  3  2  1
```

```
deck$suit         # Segunda coluna
```

```
##  [1] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
##  [7] "spades"   "spades"   "spades"   "spades"   "spades"   "spades"
## [13] "spades"   "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [19] "clubs"    "clubs"    "clubs"    "clubs"    "clubs"    "clubs"
## [25] "clubs"    "clubs"    "diamonds" "diamonds" "diamonds" "diamonds"
## [31] "diamonds" "diamonds" "diamonds" "diamonds" "diamonds" "diamonds"
## [37] "diamonds" "diamonds" "diamonds" "hearts"   "hearts"   "hearts"
## [43] "hearts"   "hearts"   "hearts"   "hearts"   "hearts"   "hearts"
## [49] "hearts"   "hearts"   "hearts"   "hearts"
```

```
deck$face         # Primeira coluna
```

```
##  [1] "king"  "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"
##  [9] "five"  "four"  "three" "two"   "ace"   "king"  "queen" "jack"
## [17] "ten"   "nine"  "eight" "seven" "six"   "five"  "four"  "three"
## [25] "two"   "ace"   "king"  "queen" "jack"  "ten"   "nine"  "eight"
## [33] "seven" "six"   "five"  "four"  "three" "two"   "ace"   "king"
## [41] "queen" "jack"  "ten"   "nine"  "eight" "seven" "six"   "five"
## [49] "four"  "three" "two"   "ace"
```

Pode-se usar a notação de $ em **listas** também. Cada subconjunto da lista será outra lista.

```
# Criando um lista
lst <- list(numbers = c(1, 2),
            logical = TRUE,
            strings = c("a", "b", "c"))

## Os três elmentos da lista são numbers, logical, strings
print(lst)
```

```
## $numbers
## [1] 1 2
##
## $logical
## [1] TRUE
##
## $strings
## [1] "a" "b" "c"
```

```
# Cada valor será retornado como uma lista

# Porém, usando notação $, os valores retornados são valores puros, sem ser na estrutura de lista
lst$numbers
```

```
## [1] 1 2
```

```
# Eles podem até ser usados
mean(lst$numbers)
```

```
## [1] 1.5
```

```
sum(lst$numbers)
```

```
## [1] 3
```

3.2.2 Colchetes duplos ([[ ]])

Se os elmentos da lista não tiverem nomes, pode-se usar a notação de colchetes duplos. Eles equivalem ao uso do $.

```
# Notação de colchetes duplo para primeiro elmento da lista
lst[[1]]          # Liberam os valores dentro desse subconjunto da lista
```

```
## [1] 1 2
```

```
lst$numbers       # A mesma coisa
```

```
## [1] 1 2
```

```
# Usar colchetes duplos é diferente de um só
lst[1]            # Devolve um subconjunto da lista
```

```
## $numbers
## [1] 1 2
```

```
lst["numbers"]    # O mesmo que o uso acima
```

```
## $numbers
## [1] 1 2
```

```
# Usando duplo colchetes
lst[[1]]          # Valor puro
```

```
## [1] 1 2
```

```
lst[["numbers"]]  # Valor puro
```

```
## [1] 1 2
```

```
lst$numbers       # Valor puro
```

```
## [1] 1 2
```

```
# Diferente
lst[1]            # Subconjunto
```

```
## $numbers
## [1] 1 2
```

```
lst["numbers"]    # Subconjunto
```

```
## $numbers
## [1] 1 2
```

Cada lista é um trem que carrega em seus vagões os valores puros.
Cada vagão é, desta forma, um elemento da lista (numbers, logical, strings do nosso exemplo).

> Usar colchetes simples seleciona o vagão.
> Usar colchetes duplos seleciona o conteúdo do vagão.