Introdução à Computação (MAC113 - FEA – 2020)

Ana C V de Melo

Manipulação de data-frames

Sobre esta Aula

O que já vimos

- ...
- Vetores, Listas, matrizes: criar, acessar os elementos, modificar elementos, adicionar elementos, remover elementos.
- Data-frames:
  - criar, acessar os elementos, modificar elementos, adicionar/remover linhas e colunas
  - como selecionar partes dos data-frames (subset())

O que veremos hoje

- Data-frames:
  - selecionar linhas por valores em colunas (subset, %in%, which)
  - operadores relacionais e lógicos
  - fazer gráficos a partir dos dados

Data-frames - aula anterior

A estrutura

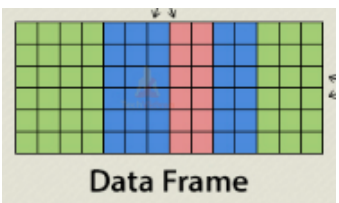


Figure 1:

Data-frames - criar

```
dados_alunos <- data.frame(
    Nome = c("Alexandre", "Allan", "Andre", "Bernardo"),
    P1 = c(3.5, 5, 8 , 9),
    P2 = c(5.5, 6, 7.5, 5.5)
)
print(dados_alunos)
```

```
##        Nome  P1  P2
## 1 Alexandre 3.5 5.5
## 2     Allan 5.0 6.0
## 3     Andre 8.0 7.5
## 4  Bernardo 9.0 5.5
```

```
typeof(dados_alunos)
```

```
## [1] "list"
```

Adicionar linhas a um data-frame - 1

-cria um novo data-frame com 1 ou mais linhas

```
dados_alunos_novos <- data.frame(
  Nome = c("Bruna", "Camila"),
  P1 = c(9, 6.5),
  P2 = c(4.6, 7.3)
)

print(dados_alunos_novos)
```

```
##     Nome  P1  P2
## 1  Bruna 9.0 4.6
## 2 Camila 6.5 7.3
```

Adicionar linhas a um data-frame - 2

- adiciona ao final do data-frame existente

```
dados_alunos <- rbind(dados_alunos, dados_alunos_novos)
print(dados_alunos)
```

```
##        Nome  P1  P2
## 1 Alexandre 3.5 5.5
## 2     Allan 5.0 6.0
## 3     Andre 8.0 7.5
## 4  Bernardo 9.0 5.5
## 5     Bruna 9.0 4.6
## 6    Camila 6.5 7.3
```

Adicionar coluna ao data-frame

- cria um data-frame com a nova coluna e adiciona ao data-frame existente

```
dados_colunaSub <- data.frame(Sub=c(6.5, NA, 5.0, NA, 8.0, NA))
dados_alunos <- cbind(dados_alunos, dados_colunaSub)
print(dados_alunos)
```

```
##        Nome  P1  P2 Sub
## 1 Alexandre 3.5 5.5 6.5
## 2     Allan 5.0 6.0  NA
## 3     Andre 8.0 7.5 5.0
## 4  Bernardo 9.0 5.5  NA
## 5     Bruna 9.0 4.6 8.0
## 6    Camila 6.5 7.3  NA
```

- NA - Not Available

Adicionar coluna ao data-frame

- cria uma nova coluna com nome "Rec" e os respectivos valores

```
dados_alunos$Rec <- c(8.5, NA, 8.5, NA, NA, NA)
print(dados_alunos)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 3     Andre 8.0 7.5 5.0 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
## 6    Camila 6.5 7.3  NA  NA
```

Mais operações sobre data-frames - seleção por valores

- alunos com nota maior ou igual a 8 na P1
- todas as linhas que satisfazem essa condição

```
alunos_p1 <- subset(dados_alunos, dados_alunos$P1 >= 8)
print(alunos_p1)
```

```
##       Nome P1  P2 Sub Rec
## 3    Andre  8 7.5   5 8.5
## 4 Bernardo  9 5.5  NA  NA
## 5    Bruna  9 4.6   8  NA
```

Mais operações sobre data-frames - seleção por valores

- alunos que Não (!) obtiveram nota maior ou igual a 8 na P1

```
alunos_p1<-subset(dados_alunos, !(dados_alunos$P1 >= 8))
print(alunos_p1)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 6    Camila 6.5 7.3  NA  NA
```

Mais operações sobre data-frames - seleção por valores

- alunos com nota entre 3 e (&) 7 na P1

```
alunos_p1<- subset(dados_alunos, dados_alunos$P1 >= 3 &
                                 dados_alunos$P1 <= 7)
print(alunos_p1)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 6    Camila 6.5 7.3  NA  NA
```

Mais operações sobre data-frames - seleção por valores

- alunos com nota menor que 5 ou (|) maior que 8 na P1

```
alunos_p1<- subset(dados_alunos, dados_alunos$P1 < 5 |
                                 dados_alunos$P1 > 8)
print(alunos_p1)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
```

Os operadores relacionais e lógicos: significado

Operadores relacionais

- relação entre valores
- têm como resultados *TRUE* ou *FALSE* :

$<$	menor que
$>$	maior que
$<=$	menor ou igual
$>=$	maior ou igual
$==$	igual
$!=$	diferente

Operadores relacionais: exemplos

```
5 > 4
```

```
## [1] TRUE
```

```
5 <= NA      # valores indefinidos (NA)?
```

```
## [1] NA
```

```
print(dados_alunos[5,])
```

```
##    Nome P1  P2 Sub Rec
## 5 Bruna  9 4.6   8  NA
```

```
dados_alunos[5,"Sub"] <= 6.8
```

```
## [1] FALSE
```

Operadores relacionais e a função subset()

- aplicada sobre vetores - testa cada elemento
- seleciona apenas os elementos com resultado TRUE

```
v <- c(12, 34, 2, 55, 32)
subset(v, v > 30)
```

```
## [1] 34 55 32
```

Lembrando o data-frame que criamos. . .

```
print(dados_alunos)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 3     Andre 8.0 7.5 5.0 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
## 6    Camila 6.5 7.3  NA  NA
```

Operadores relacionais e a função subset() sobre 1 coluna

- cada coluna do data-frame é um vetor!
- observe se selecionarmos apenas uma coluna

```
colunaP1 <- dados_alunos$P1
print(colunaP1)
```

```
## [1] 3.5 5.0 8.0 9.0 9.0 6.5
```

```
subset(colunaP1, colunaP1 > 5)
```

```
## [1] 8.0 9.0 9.0 6.5
```

Operadores relacionais e a função subset() sobre data-frames

- seleciona todas as linhas com o teste TRUE
- usamos todo o data-frame e a condição é sobre a coluna P1

```
subset(dados_alunos, dados_alunos$P1 > 5)
```

```
##       Nome  P1  P2 Sub Rec
## 3    Andre 8.0 7.5   5 8.5
## 4 Bernardo 9.0 5.5  NA  NA
## 5    Bruna 9.0 4.6   8  NA
## 6   Camila 6.5 7.3  NA  NA
```

Operadores relacionais e a função subset() sobre data-frames

- o que acontece se tivermos na coluna valores indefinidos (NA)?
- só seleciona os que resultam em TRUE

```
subset(dados_alunos, dados_alunos$Sub > 5)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 5     Bruna 9.0 4.6 8.0  NA
```

- poderemos elaborar essa seleção usando operadores lógicos. . .

Operadores Lógicos

Operadores (aplicados sobre vetores - subset())

!	não
|	ou
&	e

Tabela Verdade

P	Q	P&Q	P|Q	!P
T	T	T	T	F
T	F	F	T	
F	T	F	T	T
F	F	F	F	

- decisão formada por condições

Novas Funções sobre data-frames

Função is.na() aplicada a vetores

- testa cada elemento do vetor -> TRUE ou FALSE

```
colRec <- dados_alunos$Rec
print(colRec)
```

```
## [1] 8.5  NA 8.5  NA  NA  NA
```

```
is.na(colRec)
```

```
## [1] FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

função is.na() aplicada a data-frames

- usada junto com subset() - parte da decisão para a seleção
- quais alunos *não* fizeram a Rec? os que têm nota NA...

```
subset(dados_alunos, is.na(dados_alunos$Rec))
```

```
##       Nome  P1  P2 Sub Rec
## 2    Allan 5.0 6.0  NA  NA
## 4 Bernardo 9.0 5.5  NA  NA
## 5    Bruna 9.0 4.6   8  NA
## 6   Camila 6.5 7.3  NA  NA
```

Exemplos

```
v <- c(12, 34, 2, 55, 32)
subset(v, !(v > 30))    # valor não superior a 30
```

```
## [1] 12  2
```

```
subset(v, (v > 30) &    # intervalo entre 30 e (&)
          (v <= 55))    # 55 (inclusive)
```

```
## [1] 34 55 32
```

```
subset(v, (v < 30) |    # intervalos menores 30 ou (|)
          (v > 34))     # maiores que 34
```

```
## [1] 12  2 55
```

identificar elementos indefinidos (is.na())

- como identificar se um valor está indefinido?

```
print(dados_alunos[3,"Rec"])
```

```
## [1] 8.5
```

```
is.na(dados_alunos[3,"Rec"])  #é um valor indefinido?
```

```
## [1] FALSE
```

```
print(dados_alunos[4,"Rec"])
```

```
## [1] NA
```

```
is.na(dados_alunos[4,"Rec"])  #é um valor indefinido?
```

```
## [1] TRUE
```

quais valores não estão indefinidos (!(is.na()))

- is.na() com o not (!)
- pergunta se não é NA

```
print(colRec)
```

```
## [1] 8.5  NA 8.5  NA  NA  NA
```

```
is.na(colRec)     # quais são NA?
```

```
## [1] FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

```
!(is.na(colRec))  # quais não são NA?
```

```
## [1]  TRUE FALSE  TRUE FALSE FALSE FALSE
```

função is.na() aplicada a data-frames

- quais alunos fizeram a Rec? os que têm nota diferente de NA...

```
subset(dados_alunos, !(is.na(dados_alunos$Rec)))
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 3     Andre 8.0 7.5 5.0 8.5
```

função is.na() aplicada a data-frames

- quais alunos não fizeram a Rec e fizeram a Sub

```
subset(dados_alunos, (is.na(dados_alunos$Rec)) &
                     !(is.na(dados_alunos$Sub)))
```

```
##    Nome P1  P2 Sub Rec
## 5 Bruna  9 4.6   8  NA
```

- pense em várias outras situações para extrair dados dos data-frames

Outras operações úteis para data-frames

- encontrar linhas do data-frames com determinados valores
- extrato do data-frame para os alunos "Camila" e "Bruna"

```
cambru <- subset(dados_alunos,
         dados_alunos$Nome %in% c("Camila", "Bruna"))
print(cambru)
```

```
##     Nome  P1  P2 Sub Rec
## 5  Bruna 9.0 4.6   8  NA
## 6 Camila 6.5 7.3  NA  NA
```

```
cambru1 <- dados_alunos[dados_alunos$Nome %in%
                                c("Camila", "Bruna"),]
print(cambru1)
```

```
##     Nome  P1  P2 Sub Rec
## 5  Bruna 9.0 4.6   8  NA
## 6 Camila 6.5 7.3  NA  NA
```

Outras operações úteis para data-frames

- qual o índice da linha para um determinado valor?
- a linha onde está a aluna "Camila"

```
print(dados_alunos)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 3     Andre 8.0 7.5 5.0 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
## 6    Camila 6.5 7.3  NA  NA
```

```
which(dados_alunos$Nome == "Camila")
```

```
## [1] 6
```

Outras operações úteis para data-frames

- índices das linhas dos alunos com nota superior a 7 na P2

```
print(dados_alunos)
```

```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 3     Andre 8.0 7.5 5.0 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
## 6    Camila 6.5 7.3  NA  NA
```

```
which(dados_alunos$P2 > 7)
```

```
## [1] 3 6
```

Outras operações úteis para data-frames

- índices das linhas dos alunos com nota superior a 7 na P2, ou inferior a 5 na P1

```
print(dados_alunos)
```

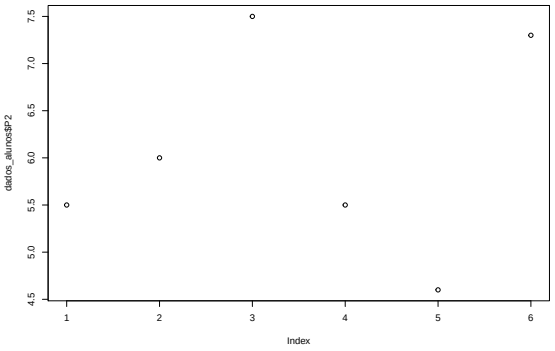
```
##        Nome  P1  P2 Sub Rec
## 1 Alexandre 3.5 5.5 6.5 8.5
## 2     Allan 5.0 6.0  NA  NA
## 3     Andre 8.0 7.5 5.0 8.5
## 4  Bernardo 9.0 5.5  NA  NA
## 5     Bruna 9.0 4.6 8.0  NA
## 6    Camila 6.5 7.3  NA  NA
```

```
which(dados_alunos$P2 > 7 | dados_alunos$P1 < 5)
```

```
## [1] 1 3 6
```

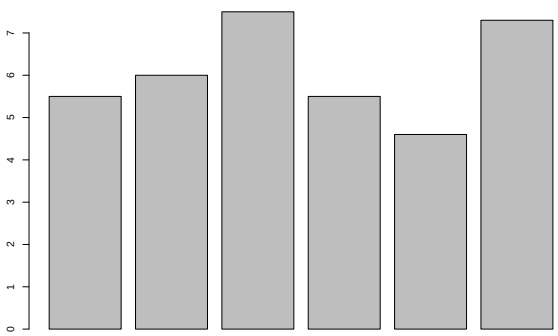
Dados de forma gráfica

```
plot(dados_alunos$P2)
```



Dados de forma gráfica

```
barplot(dados_alunos$P2)
```



Dados de forma gráfica

```
hist(dados_alunos$P2)
```