

UNIVERSIDADE DA REGIÃO DE JOINVILLE - UNIVILLE

Bacharelado em Engenharia de Software (BES)

Estatística para computação

Professora Priscila Ferraz Franczak

Engenheira Ambiental - UNIVILLE

Mestre em Ciência e Engenharia de Materiais - UDESC

Doutora em Ciência e Engenharia de Materiais - UDESC

priscila.franczak@gmail.com

Plano de Aula

1. Fatores

1.1 Criando fatores

2. Data.frames

2.1 Índices dos data.frames

2.2 Manipulando um data.frame

2.3 Separando um data.frame por grupos

3. Listas

3.1 Alguns comandos que retornam listas

4. Exercícios

1. Fatores

- Fatores proporcionam uma forma fácil e compacta de lidar com dados categóricos (ou nominais).
- Exemplos: sexo, raça, tipo sanguíneo, fator Rh.

- Estes tipos de dados podem ser vistos como variáveis que podem tomar como valores possíveis um conjunto finito de etiquetas (por exemplo uma variável que armazene o estado civil de uma pessoa).
- Há várias funções do R (ligadas à visualização de dados) que tiram partido do fato de guardarmos informação categórica como fatores em vez de usarmos *strings*.

1.1 Criando fatores

Exemplo: suponhamos que pretendemos guardar o sexo de 10 indivíduos num vetor:

```
> s <- c("f", "m", "m", "m", "f", "m", "f", "m",  
"f", "f")
```

```
> s
```

```
[1] "f" "m" "m" "m" "f" "m" "f" "m" "f" "f"
```



Ao transformarmos o vetor de caracteres em um fator, o R nos dá a possibilidade, por exemplo, de contar quantas vezes ocorre cada valor desse fator:

```
>s<-factor(s)
>s
>[1] f m m m f m f m f f
Levels: f m
```

s deixou de ser um vetor de caracteres, transformando-se num fator de dois níveis: **f** e **m**.

2. Data.frames

- São semelhantes às matrizes, pois possuem linhas e colunas.
- Porém, colunas diferentes podem armazenar elementos de tipos diferentes.
- Por exemplo: a primeira coluna pode ser numérica, quanto a segunda, constituída de caracteres.



A distinção mais importante é que em um `data.frame` (ao contrário de uma lista), **todos os membros devem ser vetores de igual comprimento.**
(The Book of R)

- O data.frame é uma das ferramentas mais importantes e frequentemente usadas em R para análise de dados estatísticos.
- Cada linha representa uma unidade (instância ou observação);
- Cada coluna representa uma variável, observada em cada unidade.

Exemplo: experimento para avaliar o desempenho escolar de uma classe de alunos. **Para cada aluno (unidade)**, são registrados os dados abaixo:

Nome	Idade	Sexo	NF
José Santos	17	M	92
Angela Dias	17	F	75
Aline Souza	16	F	81
Mayara Costa	15	F	87
Lara Lins	15	F	90
Nicolas Barros	13	M	88

“Nome” é um vetor de caracteres;

“Idade” e “Nota Final (NF)” são vetores numéricos;

“Sexo” é um fator.

- Pode-se montar o data.frame criando cada uma das colunas e juntando-as posteriormente:

Nome:

```
>Nome<-c("José Santos",  
+ "Angela Dias",  
+ "Aline Souza",  
+ "Mayara Costa",  
+ "Lara Lins",  
+ "Nicolas Barros")  
>Nome #exibe Nome  
[1] "José Santos" "Angela Dias"  
[3] "Aline Souza" "Mayara Costa"  
[5] "Lara Lins" "Nicolas Barros"
```

Idade:

```
>Idade<-c(17,17,16,15,15,13)
#criando o vetor idade
>Idade #exibe Idade
[1] 17 17 16 15 15 13
```

Sexo:

```
>Sexo<-factor(c("M","F","F","F","F","M"))
#fator
>Sexo #exibe sexo
[1] M F F F F M
Levels: F M
```


NF:

```
>NF<-c(92,75,81,87,90,88) #vetor NF  
>NF #exibe NF  
[1] 92 75 81 87 90 88
```

Agora que cada vetor foi criado, reunimos tudo em um objeto com a estrutura de dados de um data.frame:

```
> escola<-data.frame(Nome,Idade,Sexo,NF)
> escola
```

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88

A numeração automática de 1 a 6 é devido aos data.frames, que ao serem criados, possuem atributo que representa nomes das linhas (row.names), que, por padrão, são números inteiros.

2.1 Índices dos data.frames

- Nos objetos armazenados com data.frames, podemos acessar os elementos como em matrizes:

```
>escola[2,1] #elemento da linha 2, coluna 1  
[1] Angela Dias  
6 Levels: Aline Souza Angela Dias José Santos  
Lara Lins ... Nicolas Barros
```

```
>escola[2,] #toda a linha 2  
      Nome Idade Sexo NF  
2 Angela Dias   17   F  75
```

- O “Nome” foi originalmente criado como um vetor de caracteres, porém o R passou a entendê-lo como um fator dentro do data.frame.
- Isso acontece porque o argumento `stringsAsFactors`, por padrão definido como `TRUE`, converte vetores de caracteres em fatores na construção do data.frame.

- Pode-se converter a estrutura de dados de cada uma das colunas de um data.frame usando `as`, seguido de um ponto e o nome da nova estrutura desejada:

```
>escola[,1] #verificamos que realmente a coluna é um fator
[1] José Santos Angela Dias Aline Souza Mayara
Costa Lara Lins
[6] Nicolas Barros
```

Levels: Aline Souza Angela Dias José Santos
Lara Lins Mayara Costa Nicolas Barros

```
>escola[,1]<-as.character(escola[,1]) #converte a
coluna para caractere.
```

```
>escola[,1]
[1] "José Santos" "Angela Dias" "Aline Souza"
"Mayara Costa" "Lara Lins"
[6] "Nicolas Barros"
```

- A outra maneira de acessar elementos de um data.frame é usar o nome do objeto, o símbolo \$ e o nome da coluna de interesse:

Acessando a coluna “Nome”:

```
>escola$Nome
```

```
[1] "José Santos" "Angela Dias"  
[3] "Aline Souza"  "Mayara Costa"  
[5] "Lara Lins"    "Nicolas Barros"
```


Acessando o segundo elemento da coluna “Nome”:

```
>escola$Nome[2]  
[1] "Angela Dias"
```

Acessando do primeiro ao terceiro elemento da coluna “Nome”:

```
>escola$Nome[1:3]  
[1] "José Santos" "Angela Dias" "Aline  
Souza"
```

2.2 Manipulando um data.frame

- Podemos adicionar ou remover colunas ou linhas, como nas matrizes.
- `cbind()` e `rbind()`

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88

- Adicionando coluna:

```
> escola<-cbind(escola, Conceito=c("A", "C", "B", "B", "A", "B"))  
> escola
```

	Nome	Idade	Sexo	NF	Conceito
1	José Santos	17	M	92	A
2	Angela Dias	17	F	75	C
3	Aline Souza	16	F	81	B
4	Mayara Costa	15	F	87	B
5	Lara Lins	15	F	90	A
6	Nicolas Barros	13	M	88	B

- Adicionando linha:

```
> escola<-rbind(escola, "linha 7"=c("Caio Pio", 12, "M", 99, "A"))  
> escola
```

	Nome	Idade	Sexo	NF	Conceito
1	José Santos	17	M	92	A
2	Angela Dias	17	F	75	C
3	Aline Souza	16	F	81	B
4	Mayara Costa	15	F	87	B
5	Lara Lins	15	F	90	A
6	Nicolas Barros	13	M	88	B
7	Caio Pio	12	M	99	A

- Pode-se utilizar índices com o sinal negativo para eliminar linhas ou colunas de um data.frame:

```
> escola<-escola[,-5]
```

```
> escola
```

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88
7	Caio Pio	12	M	99

- Pode-se também selecionar um subgrupo de um data.frame e armazená-lo em um outro objeto:

```
> v<-escola[1:6,]
```

```
> v
```

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88

- Pode-se exibir apenas uma classe:

```
> escola[escola$Sexo=="M",] #exibindo só masculinos
```

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
6	Nicolas Barros	13	M	88

- Ordenar linhas de um data.frame segundo os dados contidos em determinada coluna também é extremamente útil:

```
> escola[order(escola$NF),] #ordena por NF
```

	Nome	Idade	Sexo	NF
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
6	Nicolas Barros	13	M	88
5	Lara Lins	15	F	90
1	José Santos	17	M	92

- Ordenar por ordem inversa (decrescente):

```
> escola[rev(order(escola$NF)),]
```

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88
4	Mayara Costa	15	F	87
3	Aline Souza	16	F	81
2	Angela Dias	17	F	75


- É possível ordenar por mais de uma coluna:

```
> escola[order(escola$Sexo,escola$NF),]
```

	Nome	Idade	Sexo	NF
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90
6	Nicolas Barros	13	M	88
1	José Santos	17	M	92

2.3 Separando um data.frame por grupos

- Podemos separar o data.frame em grupos, como por exemplo por **Sexo**:


> split(escola, Sexo)

\$F

	Nome	Idade	Sexo	NF
2	Angela Dias	17	F	75
3	Aline Souza	16	F	81
4	Mayara Costa	15	F	87
5	Lara Lins	15	F	90

\$M

	Nome	Idade	Sexo	NF
1	José Santos	17	M	92
6	Nicolas Barros	13	M	88

Objeto não
salvo!

2.4 Agrupando data.frames

`merge()`

- Permite que dois data.frames sejam unidos por uma coluna ou linha que **tenham em comum**.

```
> novo<-data.frame(Nome=escola$Nome,Número=1:6)
```

```
> novo
```

	Nome	Número
1	José Santos	1
2	Angela Dias	2
3	Aline Souza	3
4	Mayara Costa	4
5	Lara Lins	5
6	Nicolas Barros	6


```
> merge(escola,novo,by="Nome")
```

	Nome	Idade	Sexo	NF	Número
1	Aline Souza	16	F	81	3
2	Angela Dias	17	F	75	2
3	José Santos	17	M	92	1
4	Lara Lins	15	F	90	5
5	Mayara Costa	15	F	87	4
6	Nicolas Barros	13	M	88	6

Considerações:

- Caso dois data.frames a serem unidos não tenham uma coluna exatamente igual (mesmos elementos), o resultado será um data.frame contendo apenas as linhas em que ambos têm em comum.
- Uma alternativa é usar o argumento `all=T`, assim, aos elementos faltantes em quaisquer dos data.frames serão atribuídos NA.

3. Listas

- São objetos usados para combinar diferentes estruturas de dados em um mesmo objeto, ou seja, vetores, matrizes, arrays, data.frames e até outras listas.

- Exemplo:

```
> pes<-list(idade=32, nome="Aline", notas=c(98,85,96))  
> pes  
$idade  
[1] 32  
  
$nome  
[1] "Aline"  
  
$notas  
[1] 98 85 96
```

- Quando você exibe um objeto que é uma lista, cada componente é mostrado com seu nome (precedido do símbolo \$) e valor.

- Cada componente da lista pode ser acessado individualmente:

```
>pes$nome #componente "nome" da lista pes  
[1] "Aline"  
>pes$notas[2] #segundo elemento de $notas  
[1] 85
```

3.1 Alguns comandos que retornam listas

- Muitos comandos do R retornam seu resultado na forma de listas.
- Um exemplo pode ser mostrado com o uso do comando `t.test()`, que retorna um objeto que é uma lista.

Exemplo:

- Execute um teste t simples para dois conjuntos de números:

```
> x<-c(1,3,2,3,4)
> y<-c(4,5,5,4,4)
> tt<-t.test(x,y,var.equal=T)
> tt
```

Two Sample t-test

data: x and y

t = -3.182, df = 8, p-value = 0.01296

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-3.1044729 -0.4955271

sample estimates:

mean of x mean of y

2.6

4.4

- Podemos comprovar que o objeto tt é uma lista:

```
>is.list(tt)
```

```
[1] TRUE
```

```
>mode(tt)
```

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

- Podemos exibir os componentes da lista com o comando `names()`:

```
> names(tt)
```

```
[1] "statistic" "parameter" "p.value" "conf.int" "estimate"  
[6] "null.value" "alternative" "method" "data.name"
```


- Podemos extrair elementos do objeto tt individualmente:

```
>tt$p.value  
[1] 0.01295879
```

```
>tt$statistic  
t  
-3.181981
```



4. Exercícios