O mínimo que você precisa saber para aprender estatística no R.

Prof. MSc. Edgar Luiz de Lima

19/03/2022

Módulo 1 - Iniciando os trabalhos

Módulo 2 - Variáveis e tipo de dados

Usando o R como calculadora

Podemos utilizar o R para realizar operações básicas de som +, subtração -, multiplicação *, divisão / e exponiação ^.

```
2+2
```

```
## [1] 4
```

```
2*2
```

```
## [1] 4
```

```
2/2
```

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

```
2-2
```

```
## [1] 0
```

```
3^2
```

```
## [1] 9
```

Também podemos salvar os resultados dentro de um objeto, por exemplo: vamos elevar 3 ao quadrado e salvar dentro de objeto chamado A.

```
A<- 3^2
```

Note que o R não mostra o resultado, então temos que pedir para o R nos mostrar o resultado da operação.

Outra coisa importante é que o R diferencia A de a, o R interpreta letras maiúsclas diferentemente de letras minúsculas.

```
A
```

```
## [1] 9
```

Também podemos salvar letras e palavras dentro de um objeto, mas para isso devemos colocar a letra ou a palavra em aspas.

```
b<- "Hoje"
c<- "Eu"
d<- "Vou aprender R"
```

```
b
```

```
## [1] "Hoje"
```

```
c
```

```
## [1] "Eu"
```

```
d
```

```
## [1] "Vou aprender R"
```

Funções

O R possui diversas função que podemos utilizar parar realizar diferentes operações. O uso de uma função é feito escrevendo o nome da função e entre parêntese os argumentos da função, função (argumentos). Caso precise passar mais de um argumento para função, os argumentos são separados por vírgula.

```
log(10) # calculando o logarítimo natural de 10.
```

```
## [1] 2.302585
```

Agora podemos calcular o logarítmo de 10 na base 2. Note que temos dois argumentos separados por vírgula.

```
log(10,2)
```

```
## [1] 3.321928
```

A função prod calcula retona o produto de vários números.

```
prod(2,3,4,5,6)
```

```
## [1] 720
```

Temos também a função sqrt que retorna a raíz quadrada de um número.

```
sqrt(360)
```

```
## [1] 18.97367
```

A função round serve para indicar quantas cassas decimais queremos visualizar. Nela passamos um valor ou uma variável que guarda um valor, e indicamos quantas casa decimais queremos. Aqui iremos guardar o resultado da raíz quadrada de 360 dentro de um obejto chamado raiz, e pedir para o R devover o resultado com apenas duas casas decimais.

```
raiz<- sqrt(360)
round(raiz,2)
```

```
## [1] 18.97
```

Podemos também utilizar uma função que indica a classe da nossa variável.

```
a<- 10
class(a)
```

```
## [1] "numeric"
```

A variável a é uma variável numérica.

```
b<- "Eu vou aprender R"
class(b)
```

```
## [1] "character"
```

O objeto b é uma variável da classe character, pois é composto por letras ou simpolos.

Existem também as variáveis do tipo lógicas, são aquelas variáveis que guardam o resultado de uma comparação lógica, e pode ter o valor TRUE ou FALSE. Vamos fazer um teste lógico, iremos perguntar se a letra z é igual à 1 e vamos guardar o resultado detro de um obejeto chamado logica.

```
logica<- "z" ==1
logica
```

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

Obtemos um resultado FALSE, dizendo que a leta z não é igual a 1. A letra z está entre parentese, pq toda letra que não representa um objeto precisa estar entre aspas para ser interpretada pelo R.

Podemos agora perguntar se a leta z é diferente de 1.

```
logica2<- "z" != 1
logica2
```

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

Além de testar se a igualdade e a diferença entre variáveis, podemos também testar se 10 é maior que 0 ou se 2 é menor que 5 por exemplo.

```
10>0
```

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

```
2<5
```

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

Agora pra mostrar que o R interpreta letras maiúsculas de maneira diferente de letras minusculas, vamos fazer um teste de igualdade.

```
"A"=="a"
```

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

Como podemos ver, ele não considera A e a como tendo o mesmo valor. Vamos checar qual é a classe do objeto que guarda um resultado lógico?

```
logico3<- "A"=="a"
class(logico3)
```

```
## [1] "logical"
```

Podemos ver então que o obejeto logico3 é da classe logical.

Módulo 3 - Estrutura e manipulação de dados

Agora iremos aprender sobre as estruturas de dados e como manipulá-las. Os objetos de estrutura de dados que que iremos ver no curso são os objetos básicos do R.

1. Vetor: é uma sequência de valores que podem sem númericos caracteres ou lógicos;
2. Matrizes: é um objeto com duas dimensões, ou seja, possui linhas e colunas, as matrizes só podem armazenar variáveis de um tipo lógicas, númericas e caracteres.
3. Dataframe: assim como as matrizes, também é um objeto de duas dimensões, mas seu diferencial é que ela pode armazenar variáveis de diferentes tipo.

4 Listas: é um objeto que armazena outras estruturas de dados, podem armazenar vetores, matrizes e dataframes.

Vetores

Vamos começar então pelo estrutura de dados mais simples, os vetores. Podemos criar um vetor que armazena diferentes valores. Por exemplo, o vetor abaixo vai armazenar diferentes idades. Para criar um vetor iremos usar a função concatenar, que é chamada com a leta c.

```
idade<- c(18,25,30,28,10,15,60,55)
```

Agora pedimos pra ver oq tem dentro do vetor idade.

```
idade
```

```
## [1] 18 25 30 28 10 15 60 55
```

Como acima, podemos também armazenar letras e palavras em um vetor.

```
letras<- c("a", "b", "c", "d", "e")
letras
```

```
## [1] "a" "b" "c" "d" "e"
```

Podemos mesclar letras e palavras

```
palavras<- c("Edgar", "Curso", "de", "R")
palavras
```

```
## [1] "Edgar" "Curso" "de"    "R"
```

Nós podemos querer saber a soma das idades dentro do vetor, para isso existe a função sum.

```
sum(idade)
```

```
## [1] 241
```

Para saber a média das idades usamos a função mean, como não é "elegante" apresentar uma média sem uma medida de dispersão, iremos também calcular a variância e o desvio padrão.

```
mean(idade)
```

```
## [1] 30.125
```

```
var(idade)
```

```
## [1] 331.8393
```

```
sd(idade)
```

```
## [1] 18.21646
```

Para saber os valores mínimos e máximos podemos utilizar as funções min e max respectivamente.

```
min(idade)
```

```
## [1] 10
```

```
max(idade)
```

```
## [1] 60
```

Uma função muito importante se chama length, ela nos retona quantos elementos tem dentro do vetor, ou seja, ela retorna o tamanho do vetor.

```
length(idade)
```

```
## [1] 8
```

Podemos observar então que o nosso vetor idade tem comprimento 8, existem 8 valores guardados dentro deles.

É possível fazer outras operações com vetores, abaixo iremos somar 10 a cada valor do vetor.

```
idade+10
```

```
## [1] 28 35 40 38 20 25 70 65
```

Da mesma forma que foi possível somar, também é possível fazer qual quer uma das operações básicas.

Podemos também fazer uma operação entre dois vetores de mesmo tamanho.

```
idade<- c(18,25,30,28,10,15,60,55)
reducao<-c(2,4,6,8,10,12,14,16)
idade-reducao
```

```
## [1] 16 21 24 20  0  3 46 39
```

Agora vamos ver como retirar valores específicos de dentro de de vetor. Pra acessar uma possição dentro do vetor, não utilizamos colchetes e o número da posição que queremos retirar o valor. Por exemplo, eu quero retirar o 5° elemento de dentro do vetor.

```
idade[5]
```

```
## [1] 10
```

Então, o valor que está na quinta posição do vetor é o número 10.

Mas agora queremos retirar dois valores de dentro do vetor, vamos retirar o primeiro e o oitavo valor do vetor. Para isso utilizamos um vetor indicando as duas posições que queremos retirar.

```
idade[c(1,8)]
```

```
## [1] 18 55
```

Podemos também retirar valores com base em um teste lógico, por exemplo, queremos retirar valores maiores que 20, para isso precisamos fazer um teste lógico.

```
teste<- idade>20
```

Vamos olhar agora oq obtemos ao fazer esse teste.

```
teste
```

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

O vetor teste nos retona um vetor de valores lógicos, onde os valores TRUE são aqueles em que o vetor idade apresentou valores maiores que 20.

Nós podemos utilizar o vetor teste para retirar apenas os valores que foram maior que 20.

```
idade[teste]
```

```
## [1] 25 30 28 60 55
```

Podemos fazer uma substituição de valores, por exemplo, substituir o valor da quinta posição por outro valor.

```
idade[5]<- 100
idade
```

```
## [1]  18  25  30  28 100  15  60  55
```

Podemos criar um ou outro vetor que é um subconjunto do vetor idade.

```
idade2<-idade[c(2,4,6,8)]
idade2
```

```
## [1] 25 28 15 55
```

Matrizes

As matrizes são estrutura de dados com duas dimensões, ela possuem linhas e colunas e armazenam dados de apenas um tipo (ex. numérico, character...).

Podemos criar uma matrix utilizando a seguinte expressão: matrix(nrow= 3, ncol= 3), ou seja, iremos criar uma matriz com 3 linhas e 3 colunas.

```
matrix(nrow= 3, ncol= 3)
```

```
##      [,1] [,2] [,3]
## [1,]   NA   NA   NA
## [2,]   NA   NA   NA
## [3,]   NA   NA   NA
```

Como não foi indicado um conjunto de dados para a função, ela criou uma matriz preenchida com NA.

Agora vamos criar um matriz com a mesma dimensão, mas preenchida com o número 0. **Note que entre colchetes temos a indicação do número da coluna e de linhas.**

```
matrix(data= 0,nrow= 3, ncol= 3)
```

```
##      [,1] [,2] [,3]
## [1,]    0    0    0
## [2,]    0    0    0
## [3,]    0    0    0
```

BÔNUS - GRÁFICOS