17ª SEMANA NACIONAL DE
CIÊNCIA E TECNOLOGIA
Inteligência Artificial: A Nova Fronteira da Ciência Brasileira

Inteligência Artificial:
Introdução a Linguagem R

Professor Danilo S. Almeida

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Triângulo Mineiro

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA,
INOVAÇÕES E COMUNICAÇÕES

PÁTRIA AMADA
BRASIL
GOVERNO FEDERAL

Quem **usa** e quem **pode usar**?

- R é free e open-source
  - Qualquer pessoa tem liberdade para usar e modificar
- Quem usa R?
  - Google
  - Facebook
  - Instituições de pesquisas
  - Várias outras

O que é o R?

- **R** pode ser definido como uma LINGUAGEM e AMBIENTE DE PROGRAMAÇÃO com ferramentas para:
  - Manipulação de dados
  - Cálculos
  - Apresentação gráfica

O que é o R?

- Como ambiente, entende-se um sistema coerente e totalmente planejado
- O **R** não é um software do tipo aplicativo
  - a preocupação não é com amigabilidade, mas com flexibilidade
  - capacidade de manipulação de dados
  - realização de análises

Como usar?

- Diretamente do terminal ou console (tela preta)
  - Linux: konsole, xterm, gnome-terminal
- Usando interfaces gráficas
  - Rstudio: http://rstudio.org/
  - Rkward: http://rkward.sourceforge.net/
  - Rcmdr: http://www.rcommander.com/
  - StatET: http://www.walware.de/goto/statet/

Sobre o R!

- Referência básica para usuários de R:
  - http://www.r-project.org/
    - inclui programas para download
    - listas de discussão
    - documentação e ajuda
- Baixar para Linux, Windows e MAC
  - https://cran.r-project.org/mirrors.html
- Instalação direta Ubuntu (Linux):
  - sudo apt install r-base

Obter

Local:

https://rstudio.com/

Online:

https://rdrr.io/snippets/

https://www.jdoodle.com/execute-r-online/

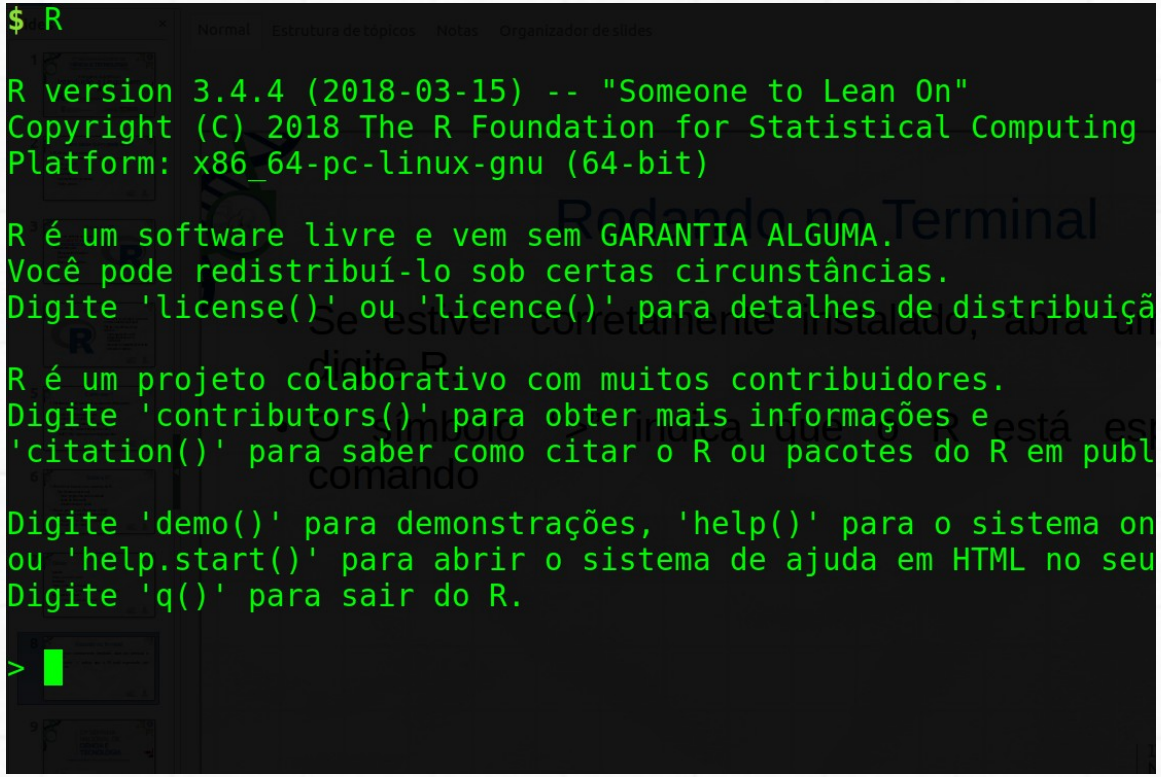
https://rextester.com/l/r_online_compiler

Aprende online:

https://learn.datacamp.com/

Rodando no Terminal

- Se estiver corretamente instalado, abra um terminal e digite R.
- O símbolo “>” indica que o R está esperando um comando



Encontrar comandos:

```
1> apropos('help') # busca por comandos similares
2> [1] "help"         "help.request" "help.search" "help.start"
3> apropos('mean')
4> [1] "colMeans"     "kmeans"       "mean"
5> [5] "mean.Date"    "mean.default" "mean.difftime"
6> [9] "mean.POSIXlt" "rowMeans"     "weighted.mean"
```

Obtendo Ajuda!

```
1> help('mean')                    # help em modo texto
2> ?mean                           # o mesmo que help('mean')
3> help('mean',help_type='html') # help em modo html
```

Sair do R

```
1> quit()
2> Save workspace image? [y/n/c]: n
```

Tipos de Dados

- character (“Danilo”)
- numeric (1.)
- integer (3)
- logical (TRUE or FALSE)
- vector (tipos homogêneos)
- list (parecidos vectors, mas heterogêneos)
- matrix
- dataframe (a maior parte dos dados estarão em dataframes)
- factors → Variáveis qualitativas que podem ser incluídas em modelos
- as.factor
- missing values (NA)

Detalhes da Linguagem R

- R é **case-sensitive**: então A e a são símbolos diferentes e se referem a diferentes variáveis
- Comandos diferentes são separados por ponto e vírgula “;”
- O conjunto de símbolos dependem do idioma e do sistema operacional onde se roda o R (tecnicamente, o *locale* em uso)
- Todos os símbolos alfanuméricos são permitidos, incluindo “.” e “_”
- Comentários começam com “#”

Detalhes da Linguagem R

```
1> #
2> A=2; a="banana"
3> print(A)
4> [1] 2
5> print(a)
6> [1] "banana"
7> #
8> .num=45; print(.num+1)
9> [1] 46
```

Detalhes da Linguagem R

- Como a maioria das linguagens de programação, R permite atribuir valores a variáveis
- A operação de atribuição tem a sintaxe objeto recebe valor
- Há dois operadores que atribuem valores a um objeto dessa maneira
  - sinal de menor seguido de hífen: **<-**
  - sinal de igual: **=**

Atribuição Menos usual

```
1> # Outra forma, menos usual é
2> "salsa" ->d ->e
3> print(d)
4> [1] "salsa"
5> print(e)
6> [1] "salsa"
7>
8> # Para saber mais
9> ?"="
10> ?"<-"
```

Apresentação de Resultados

- Em R, todo resultado é interpretado como um vetor
  - O “[1]” indica o índice do vetor
- No caso abaixo, os números entre colchetes indicam o índice do primeiro elemento de cada linha

```
1> # sequência de inteiros no intervalo [1,50]
2> 1:50
3> [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
4> [23] 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
5> [42] 42 43 44 45 46 47 48 49 50
```

Aritmética

```
1> # Usando o R como calculadora
2> 4+6
3> [1] 10
4> 3/2+1
5> [1] 2.6
6> 4*3**3                # potências são indicadas por ** ou ^
7> [1] 108
8> # Outras funções
9> sqrt(2)
10> [1] 1.414214
11> sin(pi)              # os ângulos são interpretados em radianos
12> [1] 1.224606e-16     # zero!
13> sqrt(sin(45*pi/180))
14> [1] 0.8408964
15> log(1)               # logaritmo neperiano (base e)
16> [1] 0
17> log(64,4)            # base 4
18> [1] 3
```

Criando um script

Exemplo um script: `exemplo.R`

```
1> # exemplo.R
2> a<-3
3> b<-6
4> print(a+b)
```

Criando um script

Executando o script de dentro do R

```
1> source('exemplo.R')
```

Chamando o script via terminal

```
1> R -f exemplo.R
```

Funções

```
1> # Atribui a função vetorial c(x+1, y+1) ao símbolo f
2> f <- function(x,y) {c(x+1, y+1)}
3> f(1,2)
4> [1] 2 3
5> # Para visualizar o conteúdo de f
6> f
7> function(x,y) {c(x+1, y+1)}
```

Números

```
1> # Números são interpretados literalmente
2> 1.1
3> [1] 1.1
4> 2^1023
5> [1] 8.988466e+307
6>
7> # Valores em notação hexadecimal comecam com "0x"
8> 0x1
9> [1] 1
10> 0xFFFF
11> [1] 65535
```

Números

```
1> # Qualquer número éinterpretado como ponto flutuante de dupla ↩
       precisão
2> typeof(1)
3> [1] "double"
4> ?typeof                      # mais sobre a função typeof()
5> typeof(as(1,"integer")) # se deseja um inteiro, use a função ↩
       "as"
6> [1] "integer"
```

Números

```
1> # Limites de precisão
2> (2^1023 + 1) == 2^1023 # 2^1023 éo máximo de precisão
3> [1] TRUE
4> # Limites de tamanho
5> 2^1024
6> [1] Inf                    # o universo so existe até 2^1023
7>
8> # R suporta complexos, escritos como (real) + (imaginário)i
9> 0+1i ^ 2
10> [1] -1+0i
11> sqrt(-1+0i)
12> [1] 0+1i
13> exp(0+1i * pi)
14> [1] -1+0i
```

Números

```
1> # A função sqrt() retorna um valor do mesmo tipo de entrada
2> sqrt(-1)
3> [1] NaN
4> Warning message:
5> In sqrt(-1) : NaNs produced
6> # O operador a:b retorna uma sequência de inteiros no ↩
       intervalo [a,b]
7> 1:5
8> [1] 1 2 3 4 5
9> typeof(1:5)
10> [1] "integer"
11> # Para combinar um conjunto de números em um vetor, use a ↩
       função c()
12> v <- c(173,12,1.12312,-93)
13> print(v)
14> [1] 173.00000 12.00000  1.12312 -93.0000000
```

Combinação

- O comando c() , de “combinar”, é a forma principal de criar vetores

Combinação

```
1> # Criando objetos
2> x<- c(1,2,3); print(x)
3> [1] 1 2 3
4> y<- 5:9; print(y)      # aqui temos uma sequência
5> [1]  5  6  7  8  9
6> z<-c(x,y); print(z)    # x e y são aglutinados em z
7> [1] 1 2 3 5 6 7 8 9
8> # Listando os elementos no espaço de trabalho
9> ls()
10> [1] "x" "y" "z"
11> # Apagando alguns deles
12> rm(x,y)
13> # Listando novamente
14> ls()
15> [1] "z"
16> rm(list=ls())          # apaga tudo
```

Vetores

```
1> # soma de vetores
2> c(1,2,3) + c(1,1,1)
3> [1] 2 3 4
4> # se não têm o mesmo tamanho, a menor sequência érepetida
5> c(1, 2, 3, 4) + 1
6> [1] 2 3 4 5
7> # o mesmo vale para divisão
8> 1 / c(1, 2, 3, 4, 5)
9> [1] 1.0000000 0.5000000 0.3333333 0.2500000 0.2000000
10>
11> # no caso abaixo, a menor sequência tem tamanho 2
12> c(1, 2, 3, 4) + c(10, 100)
13> [1] 11 102 13 104
```

Vetores

```
1> # A primeira sequência não émúltipla da segunda: 5 não é↩
       mútiplo de 2
2> c(1, 2, 3, 4, 5) + c(10, 100)
3> [1]  11 102 13 104 15
4> Mensagens de aviso perdidas:
5> In c(1, 2, 3, 4, 5) + c(10, 100) :
6>   comprimento do objeto maior não émúltiplo do comprimento do ↩
         objeto menor
7> # Em R podemos entrar com expressões de caracteres
8>  "Hello␣world."
9> [1] "Hello␣world."
10> # Um vetor de caracteres de comprimento 2
11> c("Hello␣world", "Hello␣R␣interpreter")
12> [1] "Hello␣world"
13> [2] "Hello␣R␣interpreter"
```

Vetores

```
1> # Formas de acessar os membros de um vetor
2> b = c(1:10)
3> b
4>  [1]  1  2  3  4  5  6  7  8  9 10
5> b[5]                    # acessa o 5o. elemento de b
6> [1] 5
7> b[2:7]                  # acessa uma fatia de b
8> [1] 2 3 4 5 6 7
9> b%%3                    # resto da divisão por 3
10>  [1] 1 2 0 1 2 0 1 2 0 1
```

Vetores

```
1> # Formas de acessar os membros de um vetor
2> b = c(1:10)
3> b
4>  [1]  1  2  3  4  5  6  7  8  9 10
5> b[5]                    # acessa o 5o. elemento de b
6> [1] 5
7> b[2:7]                  # acessa uma fatia de b
8> [1] 2 3 4 5 6 7
9> b%%3                    # resto da divisão por 3
10>  [1] 1 2 0 1 2 0 1 2 0 1
```

Vetores

```
1> # A operação abaixo retorna um vetor lógico
2> b%%3==0
3>  [1] FALSE FALSE TRUE FALSE FALSE TRUE FALSE FALSE TRUE FALSE
4>
5> # Elementos de b congruentes de 0 (mod 3)
6> b[b%%3==0]
7> [1] 3 6 9
8> #
9> b[c(2,6,9)]
10> [1] 2 6 9
```

Instruções de Controle

```
1> # Python-like for: uma iteração para cada elemento
2> x <- c(5,12,13)
3> for (n in x){
4> +    y = x/10
5> +    print(n^2+y)
6> + }
7> [1] 25.5
8> [1] 144.12
9> [1] 169.13
```

Instruções de Controle

```
1> # O mesmo vale para listas
2> l=list(p=21,k=c(1,2,3),z=NaN, f=function(a){return(a^2)})
3> for(x in l) print(x)
4> [1] 21
5> [1] 1 2 3
6> [1] NaN
7> function(a){return(a^2)}
```

Instruções de Controle

```
1> # C-style while
2> i <- 1
3> while (i <= 10) i <- i+4
4> i
5> [1] 13
6>
7> # Funcionamento básico do if
8> if (r == 4) {
9> +   x <- 1
10> +} else {
11> +   x <- 3
12> +   y <- 4
13> +}
```

+ no if é devido ao terminal

Instruções de Controle

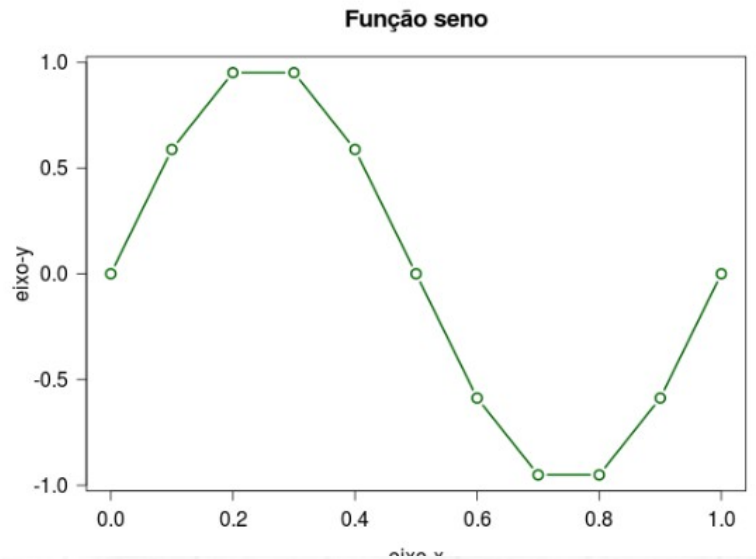
```
1> # Duas formas de representar a mesma operação
2> x <- 1
3> y <- if(x == 2) x else x+1
4> y
5> [1] 2
6> if(x == 2) y <- x else y <- x+1
7> y
8> [1] 2
```

Gráficos no R

- R inclui vários pacotes para a visualização de dados
- Se você é familiarizado com planilhas eletrônicas, você perceberá que o R pode gerar
  - gráficos de barras
  - gráficos de linhas
  - histogramas
  - gráficos de dispersão
  - ...
- Vamos verificar alguns casos por meio de exemplos

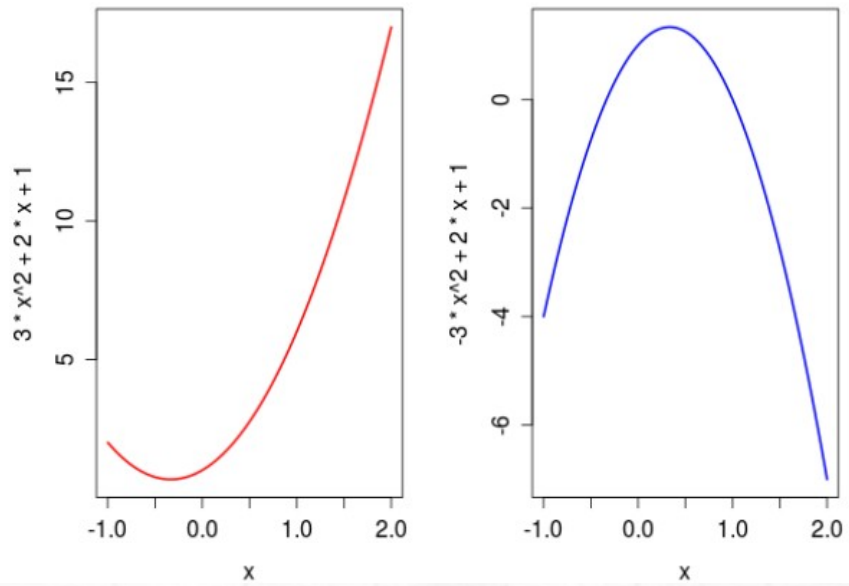
Gráficos no R

```
1> # Gráfico da função seno
2> x = seq(0,1,0.1)
3> y = sin(2*pi*x)
4> plot(x,y,type='b',col='darkgreen',main='Função␣↩
       seno',xlab='eixo-x', ylab='eixo-y',lwd=2)
```



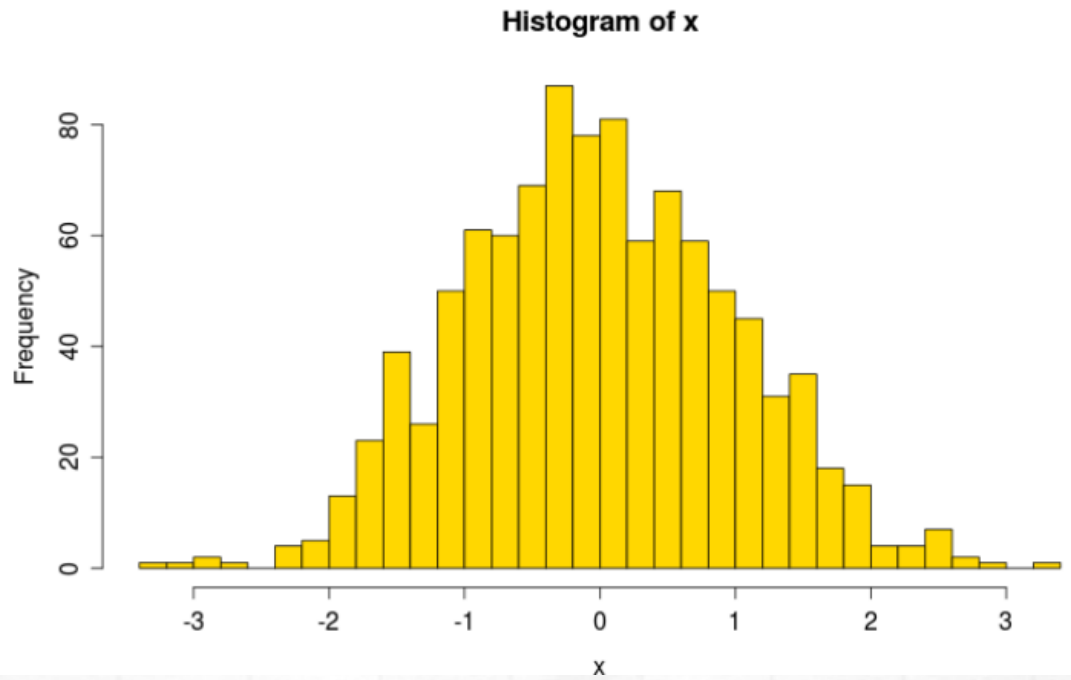
Gráficos no R

```
1> # Gráficos de funções paramétricas de uma variável
2> par(mfrow=c(1,2))                    # uma linha e duas colunas
3> curve( 3*x^2+2*x+1, -1, 2,col="red",lwd=2)
4> curve(-3*x^2+2*x+1, -1, 2,col="blue",lwd=2)
```



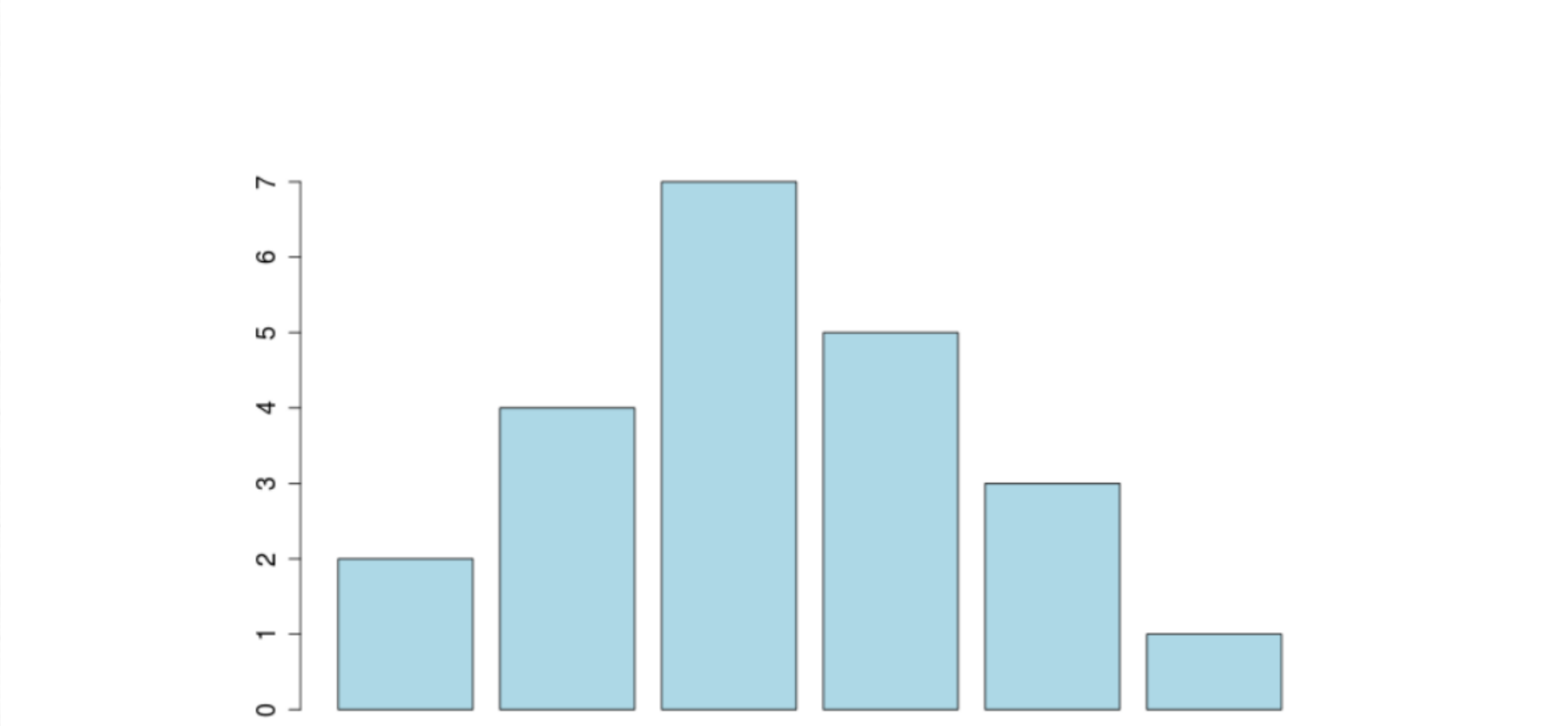
Gráficos no R

```
1> # Histograma de uma série dados com distribuição normal
2> x = rnorm(1000)
3> hist(x,col="yellow",breaks=40)
```



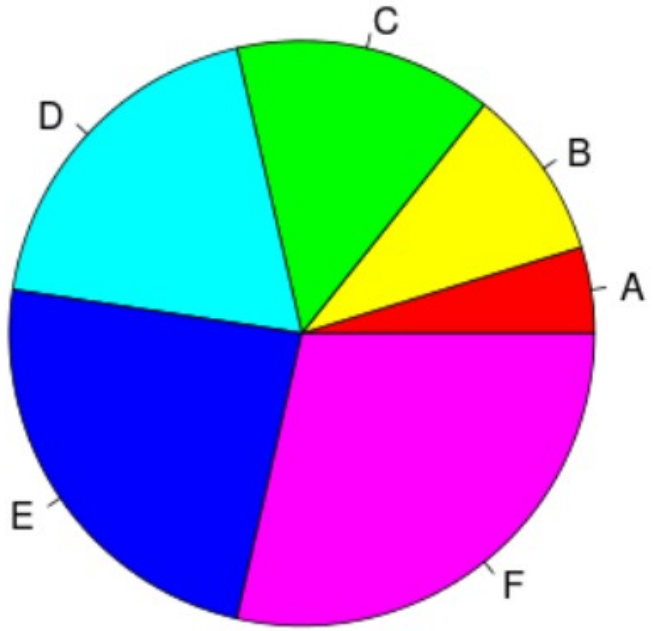
Gráficos no R

```
1> # Um gráfico de barras
2> barplot(c(2,4,7,5,3,1),col='lightblue')
```



Gráficos no R

```
1> # Gráfico de pizza
2> pie(1:6, col = rainbow(6), labels= LETTERS[1:6], radius = 0.9)
```



Bônus

- Algorítimo de Machine Learning
  - Pacote: **e1071**

Agradecimentos

@professordaniloalmeida

youtube.com/professordaniloalmeida

Obrigado a todos (as)!

www.professordanilo.com.br

17ª SEMANA NACIONAL DE CIÊNCIA E TECNOLOGIA

INSTITUTO FEDERAL
Triângulo Mineiro

17ª SEMANA NACIONAL DE CIÊNCIA E TECNOLOGIA

Inteligência Artificial: A Nova Fronteira da Ciência Brasileira

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Triângulo Mineiro

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA,
INOVAÇÕES E COMUNICAÇÕES

PÁTRIA AMADA
BRASIL
GOVERNO FEDERAL