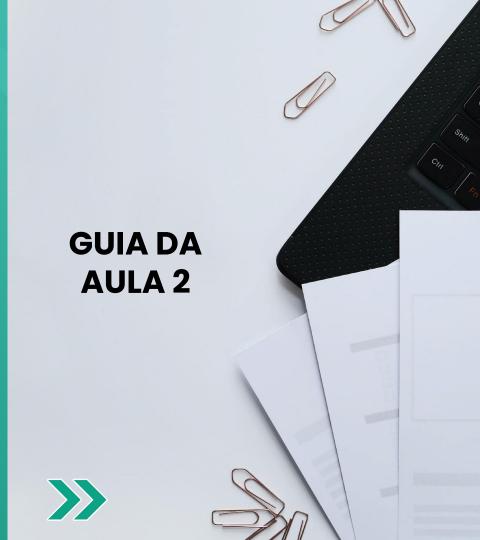


Profissão: Analista de dados





FUNDAMENTOS DE MATEMÁTICA







Conheça os arrays Numpy

- Arrays vs Listas
- Arrays 1D: Vetores
- Arrays 2D:
 Matrizes
- Arrays 3D, 4D etc.



Acompanhe aqui os temas que serão tratados na videoaula







Arrays vs Listas

As listas Python e os arrays NumPy são similares em muitos sentidos: ambos servem para armazenar dados sequencialmente na memória, possuem sintaxe parecidas etc.
Contudo, é importante maximizar as suas diferenças:

 Manipulação algébrica: Arrays apresentam uma sintaxe mais simples e eficiente (velocidade e memória) por realizar operações de forma vetorial. Listas sempre trabalham de forma escalar. Exemplo:

```
In []: # escalar

11 = [1, 2, 3]
12 = [4, 5, 6]

13 = [a + b for a, b in zip(11, 12)] print(13)
```





```
In []:  # vetorial
    a1 =
        np.array(11) a2
        = np.array(12)
        a3 = a1 + a2
        print(a3)
```

- Tipo: Arrays trabalham melhor com elementos do mesmo tipo.
- Mutabilidade: Arrays são menos eficientes quanto a inserção e remoção de elementos.





Arrays 1D: Vetores

Arrays NumPy de uma dimensão (1D) são conhecidos como vetores, como listas de uma linha ou mais colunas.

• Criação:

```
a1 = np.array([2, 4, 6,
        8]) print(a1)
        a1 = np.arange(0, 10,
        2) print(a1)
In [ ]:
        np.zeros(10)
        print(a1)
        a1 = 10 *
        np.ones(10)
        print(a1)
                                     (( | | ))
```



• Manipulação:

```
In []: a1 = np.array([2, 4, 6,
8]) print(a1)
In []: a1[0]
In []: a1[0:2]
In []: a1[a1 > 4]
```





Atributos:

```
In [ ]: al.ndim
In [ ]: al.shape
In [ ]: al.size
In [ ]: al.dtype
```





• Métodos:

```
In []: a1.sort() # "inplace"
    print(a1)
In []: a1.tolist()
```





Arrays 2D: Matrizes

Arrays NumPy de duas dimensões (2D) são conhecidos como matrizes, como tabelas, com linhas e colunas.

• Criação:

```
In []:
    m1 = np.array([[1, 2, 3], [4, 5, 6]]) # vetores como linhas
    print(m1)
```





• Manipulação:

```
In [ ]:    m1[1,2] # linha x coluna
In [ ]:    m1[1,:]
In [ ]:    m1[:,1]
```





Atributos:

```
In [ ]:    m1.ndim
In [ ]:    m1.shape
In [ ]:    m1.size
In [ ]:    m1.dtype
```





Métodos:

```
In []:     m1.sort() # "inplace"
     print(m1)
In []:     m1.tolist()
```





Arrays 3D, 4D etc.

Arrays NumPy de três ou mais dimensões são apenas estruturas de dados com mais de duas dimensões.

