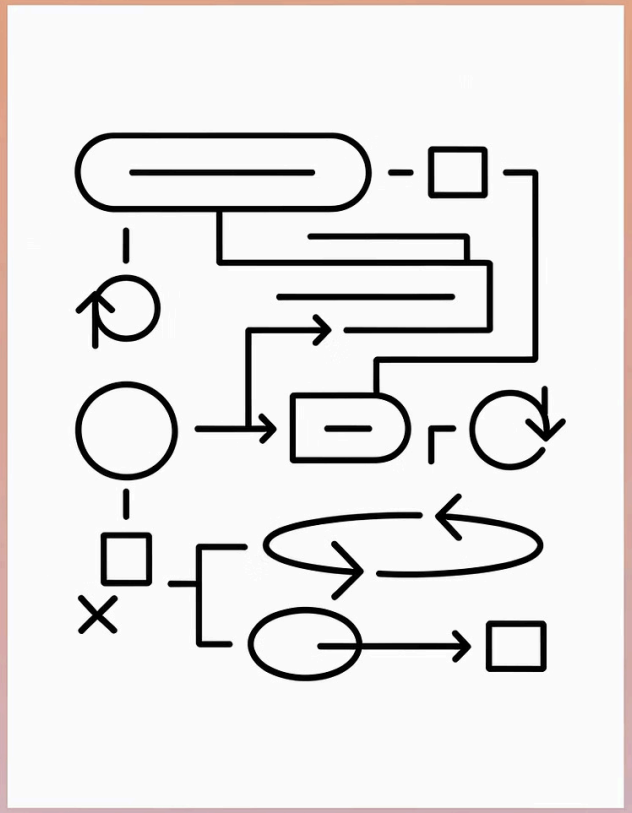


CEFET/RJ

Iteração Implícita em Python

Em Python, operações sobre vetores e tabelas não precisam de loops explícitos. Bibliotecas como NumPy e pandas permitem aplicar funções a muitos dados de uma vez, em um estilo de programação chamado de **processamento vetorizado**. Esse método é mais rápido, mais legível e mais seguro do que usar estruturas repetitivas tradicionais.

```
import numpy as np
import pandas as pd
```

Eduardo Ogasawara
eduardo.ogasawara@cefet-rj.br
https://eic.cefet-rj.br/~eogasawara

FUNDAMENTOS

Estruturas Vetorizadas

NumPy representa dados numéricos como **arrays vetorizados**, permitindo aplicar funções matemáticas em blocos inteiros de dados. Pandas usa essas estruturas internamente para gerenciar tabelas de forma eficiente, eliminando a necessidade de loops explícitos.

```
x = np.array([1, 2, 3, 4])
np.mean(x)
```

Saída: 2.5

EXEMPLO PRÁTICO

Criando uma Tabela com Pandas

Vamos criar uma tabela contendo peso, altura e identificação de pessoas. Esses dados demonstrarão como funcionam as operações vetorizadas. Cada coluna é uma série de valores, e o DataFrame organiza tudo em formato tabular estruturado.

```
data = {
   "weight": [60, 72, 57, 90, 95, 72],
   "height": [1.75, 1.80, 1.65, 1.90, 1.74, 1.91],
   "subject": ["A", "B", "C", "D", "E", "F"]
}
df = pd.DataFrame(data)
```

Resultado

Uma tabela com 6 linhas e 3 colunas: weight, height e subject

Selecionando Colunas Numéricas

Antes de calcular estatísticas como médias ou mínimos, é essencial selecionar apenas as colunas numéricas. Isso evita erros ao tentar processar colunas de texto. O pandas facilita esse filtro automaticamente.

```
num_cols = df.select_dtypes(
    include="number"
)
```

Por que filtrar?

- Evita erros de tipo
- Isola dados quantitativos
- Prepara para cálculos
- Melhora performance

Médias por Coluna (Vetorizado)

Cálculo Automático

O pandas aplica a função `mean()` internamente sem necessidade de loops explícitos

Resultado Imediato

Uma série com um valor por coluna é retornada instantaneamente

Processamento Vetorizado

Toda a operação acontece em nível de C, garantindo máxima eficiência

```
num_cols.mean()
```

Saída:

weight 74.333333
height 1.791667

NUMPY

Operações com Axis em NumPy

Quando trabalhamos com NumPy, podemos escolher a direção do cálculo usando o parâmetro `axis`. Isso controla como a função percorre a matriz multidimensional.

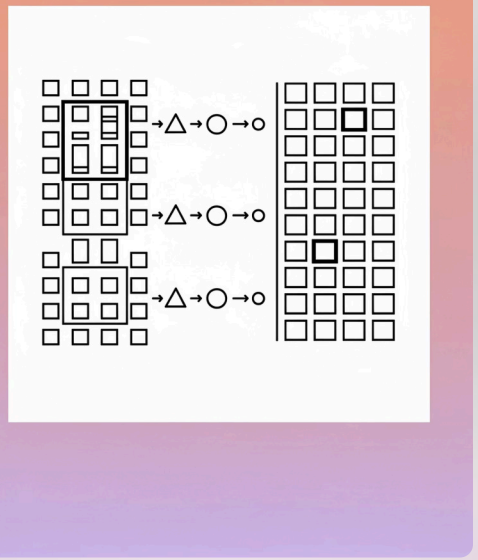
axis=0

Processa **por coluna**, percorrendo verticalmente

axis=1

Processa **por linha**, percorrendo horizontalmente

```
m = num_cols.to_numpy()
```



Mínimo por Linha e por Coluna

Por Linha (axis=1)

Encontra o menor valor em cada linha, comparando as colunas entre si.

```
np.min(m, axis=1)
```

Saída:

array([1.75, 1.8, 1.65, 1.9, 1.74, 1.91])

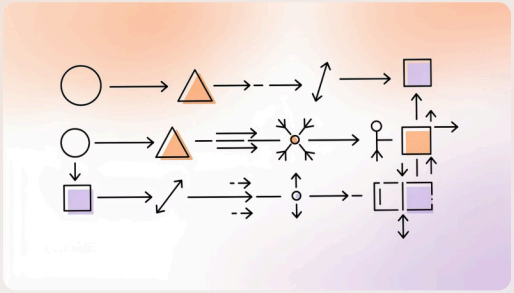
Por Coluna (axis=0)

Encontra o menor valor em cada coluna, comparando todas as linhas.

```
np.min(m, axis=0)
```

Saída:

array([57., 1.65])



Broadcasting Entre Colunas

01

Alinhamento Automático

NumPy e pandas alinham vetores compatíveis automaticamente

02

Operações Elemento a Elemento

Cálculos matemáticos são aplicados em cada posição correspondente

03

Broadcasting

Mecanismo que permite criar novas variáveis sem loops explícitos

```
df["height"] ** 2
```

Saída: array([3.0625, 3.2400, 2.7225, 3.6100, 3.0276, 3.6481])

APLICAÇÃO PRÁTICA

Cálculo Derivado sem Loop

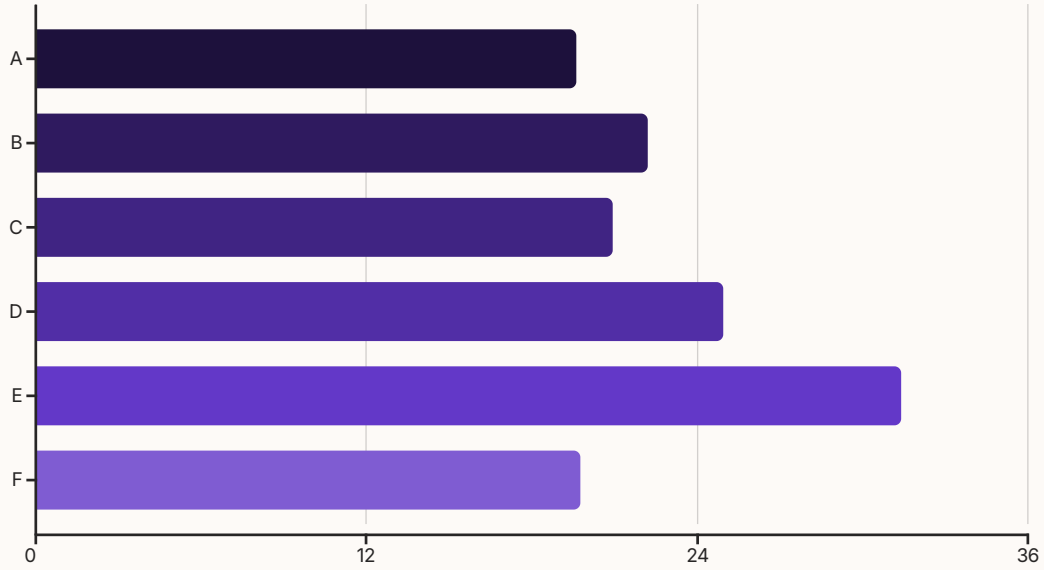
Podemos combinar colunas para criar novas métricas de forma vetorizada. Aqui calculamos o **Índice de Massa Corporal (IMC)**, onde cada linha é processada automaticamente sem necessidade de iteração explícita.

Fórmula do IMC

IMC = peso / (altura²)

Implementação

```
df["bmi"] = df["weight"] /
   (df["height"] ** 2)
```



Aplicar Função a Várias Colunas

Às vezes precisamos aplicar a mesma função a múltiplas colunas de forma organizada. Uma compreensão de dicionário mantém cada valor separado por coluna, oferecendo clareza e estrutura ao código.

Definir Colunas

```
cols = ["weight", "height"]
```

Aplicar Função

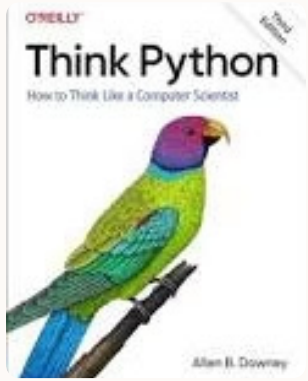
```
means = {col: df[col].mean()
    for col in cols}
```

Resultado

weight: 74.33
height: 1.79

Dica: Compreensões de dicionário são ideais para aplicar operações uniformes mantendo a legibilidade do código.

Referências



Think Python

Downey, A. *Think Python: How to Think Like a Computer Scientist.* O'Reilly Media.

An essential introduction to programming fundamentals and computational thinking using Python.

Python Data Science Handbook

VanderPlas, J. *Python Data Science Handbook.* O'Reilly Media.

A comprehensive guide to essential tools for working with data in Python, including NumPy, Pandas, and visualization libraries.

Data Science from Scratch

Grus, J. *Data Science from Scratch.* O'Reilly Media.

Learn data science fundamentals by building algorithms and tools from the ground up using Python.