

***INTELIGÊNCIA ARTIFICIAL***

**UA02 / LABORATÓRIO # 1**

**1-) O que é o Pandas?**

Pandas é uma biblioteca do Python para manipulação e análise de dados. Ela é usada principalmente para trabalhar com dados tabulares (como tabelas do Excel ou bancos de dados).

**Estruturas principais do Pandas:**

1.) DataFrame:

- É uma tabela com linhas e colunas (como uma planilha do Excel).

- Cada coluna pode ter um tipo diferente (números, textos, datas).

Exemplo:

import pandas as pd

dados = {'Nome': ['Ana', 'João', 'Maria'], 'Idade': [25, 30, 28]}

df = pd.DataFrame(dados)

print(df)

Saída:

Nome Idade

0 Ana 25

1 João 30

2 Maria 28

2.) Series:

- É uma única coluna de um DataFrame (como uma lista, mas com rótulos).

- Exemplo:

idades = pd.Series([25, 30, 28], name="Idade")

print(idades)

Saída:

0 25

1 30

2 28

Name: Idade, dtype: int64

2.) Criando um DataFrame

Você pode criar um DataFrame de duas formas principais:

**Método 1: A partir de um dicionário**

dados = {

'Produto': ['Maçã', 'Banana', 'Laranja'],

'Preço': [3.50, 2.00, 4.20]

}

df = pd.DataFrame(dados)

print(df)

Saída:

Produto Preço

0 Maçã 3.5

1 Banana 2.0

2 Laranja 4.2

**Método 2: A partir de uma lista de listas**

dados = [['Maçã', 3.50], ['Banana', 2.00], ['Laranja', 4.20]]

df = pd.DataFrame(dados, columns=['Produto', 'Preço'])

print(df)

Saída igual ao exemplo anterior.

Produto Preço

0 Maçã 3.5

1 Banana 2.0

2 Laranja 4.2

3.) Lendo Dados de um Arquivo CSV

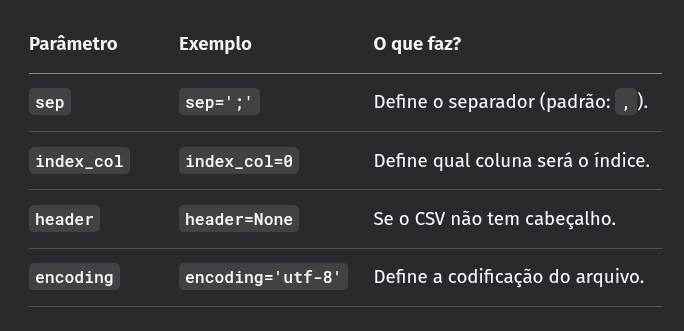
Um CSV é um arquivo de texto que armazena dados em formato de tabela, separados por vírgulas.

**Como ler um CSV com Pandas?**

df = pd.read\_csv('dados.csv') # Lê o arquivo 'dados.csv'

print(df.head()) # Mostra as 5 primeiras linhas

**Parâmetros úteis do `read\_csv`**



4.) Selecionando Dados em um DataFrame

Existem duas formas principais de selecionar dados:

A.) loc[] (seleção por RÓTULO)

- Usa nomes de linhas/colunas.

- Inclui o último elemento.

- Exemplos:

df.loc[0] # Pega a linha com índice 0

df.loc[0:2] # Pega as linhas 0, 1 e 2 (inclui o 2!)

df.loc[:, 'Produto'] # Pega TODAS as linhas da coluna 'Produto'

B.) iloc[] (seleção por POSIÇÃO)

- Usa números de posição.

- Exclui o último elemento (como listas do Python).

- Exemplos:

df.iloc[0] # Pega a primeira linha

df.iloc[0:2] # Pega as linhas 0 e 1 (exclui o 2!)

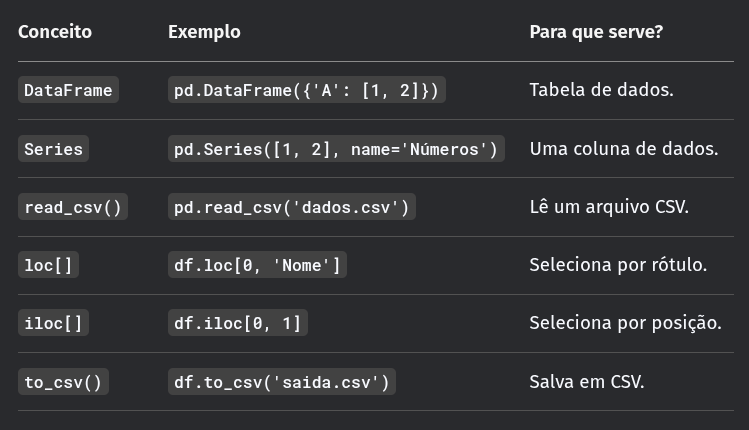
df.iloc[:, 1] # Pega TODAS as linhas da SEGUNDA coluna

5.) Salvando um DataFrame em CSV

Para salvar um DataFrame em um arquivo CSV:

df.to\_csv('novo\_arquivo.csv', index=False) # `index=False` evita salvar os índices

**Resumo Final**



**Dúvidas Comuns**

Qual a diferença entre **loc** e **iloc**?

- **loc** usa nomes (ex: df.loc[0, 'Nome']).

- **iloc** usa posições numéricas (ex: df.iloc[0, 1]).

Como pegar uma coluna específica?

df['Nome'] ou df.Nome (só funciona se o nome não tiver espaços).

Como filtrar linhas?

df[df['Idade'] > 25] → Retorna linhas onde a idade é maior que 25.