@BigDataDoctor

Python para Dados

Colinha dos Fundamentos de Pandas

Aprenda Python Todos os Dias no Nosso Instagram & TikTok

Pandas

A biblioteca Pandas é construída em NumPy e fornece estruturas de dados fáceis de usar e ferramentas de análise de dados para a linguagem de programação Python.

Use a seguinte convenção de importação:

```
>>> importar pandas como pd
```

Estruturas de Dados do Pandas

Series

Um Array (ou arranjo) unidimensional capaz de carregar qualquer tipo de dado.

```
>>> s = pd.Series(
          data=[1, 2, 3, 3],
          index=[ 'A', 'B', 'C', 'D' ])
```

```
A    1
B    2
C    3
D    3
dtype: int64
```

DataFrame

Uma estrutura de dados rotulada (com nomes de colunas) de duas dimensões podendo conter diversos tipo de dados.

Semelhante a uma tabela de Excel.

```
>>> data = {
    'Nome': [
        'Elon Musk',
        'Mark Zuckerberg',
        'Jeff Bezos'
    ],'Idade': [50, 37, 57],
    'Altura (m)': [1.80,1.71,1.71]
}
>>> df = pd.DataFrame(data=data)
```

	Nome	Idade	Altura (m)
0	Elon Musk	50	1.80
1	Mark Zuckerberg	37	1.71
2	Jeff Bezos	57	1.71

Python para Dados

I/O (Lendo e Salvando Dados)

CSV (Comma-Separated Value)

```
>>> df.to_csv('tech_ceos.csv')
>>> pd.read_csv('tech_ceos.csv', index_col=0)
```

Excel

```
>>> df.to_excel('tech_ceos.xlsx', sheet_name='ceos')
>>> pd.read_excel('tech_ceos.xlsx', index_col=0)
```

SQL

```
>>> from sqlalchemy import create_engine
>>> engine = create_engine('sqlite:///:memory:')
>>> df.to_sql('CEOS', engine)
>>> pd.read_sql('SELECT * FROM CEOS;', engine)
```

Selecionando Dados

Getting (Acessando Dado)

```
>>> s['A'] # seleciona um elemento da série
1
```

```
>>> df[1:] # slicing -> seleciona subset de df
```

	Nome	Idade	Altura (m)
1	Mark Zuckerberg	37	1.71
2	Jeff Bezos	57	1.71

```
>>> df['Nome'] # seleciona coluna
```

```
0          Elon Musk
1    Mark Zuckerberg
2         Jeff Bezos
Name: Nome, dtype: object
```

```
>>> df.loc[0, 'Nome'] # seleciona [índice, coluna]
'Elon Musk'
```

```
>>> df.iloc[1, 0] # seleciona [indice x, indice y]
'Mark Zuckerberg'
```

Python para Dados

Filtrando e Sobrescrevendo

Filtro Booleano

```
>>> df[df['Altura (m)'] == 1.71] # altura igual a 1.71
>>> df[~df.Nome.str.startswith('E')] # nome NÃO inicia com E
>>> df[(df.Idade < 40) & (df.Idade > 30)] # entre 50 e 40
```

Setting (Sobrescrevendo Dado)

```
>>> s['A'] = 5
>>> df.iloc[1, 0] = 'Tio Zucker'
```

Acessando Metadados (Informações Gerais dos Dados)

Informações Básicas

```
>>> df.shape # formato do DataFrame
(3,3)
>>> df.index # informa índice
RangeIndex(start=0, stop=3, step=1)
>>> df.columns # informa colunas
Index(['Nome', 'Idade', 'Altura (m)'],
dtype='object')
>>> df.count()
Nome          3
Idade         3
Altura (m)    3
dtype: int64
```

```
>>> df.info() # overview dos dados
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Nome        3 non-null      object
 1   Idade       3 non-null      int64
 2   Altura (m)  3 non-null      float64
dtypes: float64(1), int64(1), object(1)
memory usage: 200.0+ bytes
```

Operações Básicas

```
>>> df.sum()
Nome          Elon MuskTio ZuckerJeff Bezos
Idade                               144
Altura (m)                         5.22
dtype: object
```

```
>>> df.cumsum()
```

	Nome	Idade	Altura (m)
0	Elon Musk	50	1.80
1	Elon MuskTio Zucker	87	3.51
2	Elon MuskTio ZuckerJeff Bezos	144	5.22

```
>>> df.describe()
```

	Idade	Altura (m)
count	3.000000	3.000000
mean	48.000000	1.740000
std	10.148892	0.051962
min	37.000000	1.710000
25%	43.500000	1.710000
50%	50.000000	1.710000
75%	53.500000	1.755000
max	57.000000	1.800000

Python para Dados

Manipulando os Dados

Cálculos

```
>>> df.Idade.apply(lambda x: x*2) # apenas coluna
0    100
1     74
2    114
Name: Idade, dtype: int64
>>> df.applymap(lambda x: x*2) # todo DataFrame
```

	Nome	Idade	Altura (m)
0	Elon MuskElon Musk	100	3.60
1	Tio ZuckerTio Zucker	74	3.42
2	Jeff BezosJeff Bezos	114	3.42

Removendo Valores

```
>>> s.drop(['A','B']) # remove itens da série
>>> df.drop('Idade', axis=1) # remove coluna do
DataFrame
```

Ordenando

```
>>> df.sort_index() # ordena índice
>>> df.sort_values('Idade') # ordena valores de coluna
>>> df.rank() # gera ranking (ordem) de colunas
```

Curtiu essas dicas?

Aprenda Python Todos os Dias no Nosso Instagram & TikTok