Apresentação da Biblioteca Pandas

Prof. Dr. Fabiano B. Menegidio

1 Introdução

Pandas é uma biblioteca de código aberto para manipulação e análise de dados em Python. Ela fornece estruturas de dados rápidas, flexíveis e expressivas, projetadas para facilitar o trabalho com dados estruturados e rótulados, como tabelas de banco de dados e planilhas. A biblioteca Pandas é amplamente utilizada em ciência de dados, estatística e engenharia de software, especialmente em processos de análise, limpeza e transformação de dados.

Ao oferecer duas estruturas principais – **Series** e **DataFrame** – o Pandas permite manipular e processar grandes volumes de dados de maneira eficiente, oferecendo várias funcionalidades para leitura, filtragem, agregação e manipulação de dados.

2 Por que usar Pandas?

Pandas é uma das bibliotecas mais populares e fundamentais para o trabalho com dados em Python. Sua principal força reside na capacidade de trabalhar com grandes conjuntos de dados de forma eficiente, facilitando operações complexas, como fusão de dados, agrupamento e manipulação de colunas e índices.

Pandas é amplamente usado em áreas como *ciência de dados*, *machine learning*, *análise financeira*, *estatísticas* e até mesmo *visualização de dados*, integrando-se perfeitamente com bibliotecas como *Matplotlib* e *Seaborn*.

3 Principais Utilizações do Pandas

- **Manipulação de Dados Tabulares**: Pandas oferece o DataFrame, uma estrutura de dados tabular, similar a uma tabela SQL ou uma planilha Excel.
- **Leitura e Escrita de Arquivos**: Com Pandas, é possível ler e gravar arquivos de diversos formatos (CSV, Excel, JSON, SQL, etc.).
- **Análise de Dados Estatísticos**: A biblioteca fornece funções para calcular estatísticas descritivas, realizar agrupamentos, pivotagem e outras operações fundamentais.

- **Limpeza de Dados**: Pandas oferece métodos para tratamento de dados ausentes, substituição de valores e manipulação de strings.
- **Integração com Outras Ferramentas**: Pandas pode ser usado junto a bibliotecas como NumPy, Matplotlib e Scikit-learn.

4 Estrutura Básica do Pandas

O Pandas fornece duas estruturas de dados principais:

- **Series**: Um objeto unidimensional semelhante a uma lista ou vetor.
- **DataFrame**: Uma estrutura bidimensional (tabela) que permite armazenar e manipular dados em linhas e colunas com rótulos (índices).

4.1 Series

Uma `Series` é um array unidimensional que pode armazenar qualquer tipo de dado (inteiros, strings, floats, objetos Python). É semelhante a um array do NumPy, mas com um índice associado a cada elemento.

Exemplo de criação de uma Series:

```
import pandas as pd

# Criar uma Series
s = pd.Series([1, 2, 3, 4, 5], index=['a', 'b', 'c', 'd', 'e'])
print(s)
```

4.2 DataFrame

O `DataFrame` é a estrutura de dados mais poderosa do Pandas. Ele é uma tabela bidimensional com rótulos para linhas e colunas. Cada coluna pode conter diferentes tipos de dados (inteiros, floats, strings, etc.).

Exemplo de criação de um DataFrame:

```
import pandas as pd

# Criar um DataFrame
dados = {'Nome': ['Ana', 'Bruno', 'Carlos', 'Diana'],
         'Idade': [23, 35, 45, 28],
         'Cidade': ['S o Paulo', 'Rio de Janeiro', 'Belo Horizonte
    ', 'Curitiba']}
df = pd.DataFrame(dados)
print(df)
```

5 Principais Funções do Pandas

O Pandas oferece uma grande variedade de funções para trabalhar com dados. Abaixo estão algumas das funções mais importantes e amplamente utilizadas.

5.1 1. Leitura de Dados com Pandas

Pandas facilita a leitura de dados a partir de vários formatos de arquivo. Algumas das funções mais utilizadas incluem:

- `pd.read_csv()`: Leitura de arquivos CSV.
- `pd.read_excel()`: Leitura de arquivos Excel.
- `pd.read_json()`: Leitura de arquivos JSON.
- `pd.read_sql()`: Leitura de dados a partir de uma consulta SQL.

Exemplo de leitura de um arquivo CSV:

```
import pandas as pd

# Ler dados de um arquivo CSV
df = pd.read_csv('dados.csv')
print(df.head())  # Exibir as primeiras 5 linhas do DataFrame
```

5.2 2. Seleção e Fatiamento de Dados

Pandas facilita a seleção de linhas e colunas de um DataFrame de forma eficiente.

Exemplo de seleção de dados:

```
# Selecionar a coluna 'Nome'
nomes = df['Nome']

# Selecionar as primeiras duas linhas e a coluna 'Cidade'
primeiras_duas = df.loc[:1, 'Cidade']
```

5.3 3. Manipulação de Colunas e Linhas

Pandas permite adicionar, remover ou modificar colunas e linhas de um DataFrame de maneira fácil.

Exemplo de adição e remoção de colunas:

```
# Adicionar uma nova coluna 'Sal rio'
df['Sal rio'] = [5000, 6000, 7000, 8000]

# Remover a coluna 'Cidade'
df = df.drop('Cidade', axis=1)
```

5.4 4. Agrupamento e Agregação

Pandas oferece funções poderosas para agrupar dados e calcular agregações.

Exemplo de agrupamento:

```
# Agrupar por 'Cidade' e calcular a m dia de 'Idade'
media_idades = df.groupby('Cidade')['Idade'].mean()
print(media_idades)
```

5.5 5. Tratamento de Dados Ausentes

Pandas oferece métodos eficientes para lidar com dados ausentes.

Exemplo de preenchimento de valores ausentes:

```
# Preencher valores ausentes na coluna 'Sal rio' com a m dia da coluna
df['Sal rio'] = df['Sal rio'].fillna(df['Sal rio'].mean())
```

6 Conclusão

O Pandas é uma biblioteca indispensável para quem trabalha com manipulação e análise de dados em Python. Suas estruturas de dados, como Series e DataFrame, permitem a manipulação eficiente de grandes volumes de dados tabulares. Com uma ampla gama de funções para leitura, transformação, limpeza, agrupamento e agregação de dados, o Pandas facilita o trabalho com dados estruturados e é amplamente utilizado em ciência de dados, aprendizado de máquina e análise de negócios.