

Linguagens de programação para ciência de dados (Python com Spark)



PÓS-GRADUAÇÃO

Manipulação de dados com Python


Bloco 1

Anderson Paulo Ávila Santos






► Objetivos

- Apresentar as formas de manipulação de dados em Python.
 - Preparar bibliotecas de limpeza de dados.
 - Apresentar biblioteca Pandas para manipulação de dados.
- 



► Motivação

- A manipulação de dados é de extrema importância para a ciência de dados.
 - Pesquisas mostram que cientistas de dados passam a maior parte do tempo preparando os dados em vez de minerar.
 - Gastam 60% de seu tempo limpando e organizando dados.
 - O Python disponibiliza uma série de bibliotecas que visam facilitar e proporcionar maior agilidade nessas tarefas.
- 



► Principais bibliotecas

- Python disponibiliza para a manipulação de dados. As principais são Numpy, Scipy e o Pandas.
- O Numpy é uma biblioteca Python que dá suporte de operações numéricas.
- O Scipy é voltado para a programação científica.
- Pandas disponibiliza uma estrutura para manipulação de dados em formato tabular; implementa a estrutura de data.frames do R.

► Pandas

- Pandas é uma biblioteca do Python e para ser utilizada é necessário que sua instalação seja feita previamente:
`python -m pip install --upgrade pandas`
- Pode ser criado usando dicionários de dados como no exemplo:

Figura 1 – Criando dicionários

```
import pandas as pd

df_data = {'pais': ['Brasil', 'Argentina', 'Argentina', 'Brasil', 'Chile', 'Chile'],
           'ano': [2005, 2006, 2005, 2006, 2007, 2008],
           'populacao': [170.1, 30.5, 32.2, 172.6, 40.8, 42.0]}
df = pd.DataFrame(df_data)
print(df)
```

Fonte: elaborado pelo autor.

► Pandas

- Há a possibilidade de importação de arquivos e transformação dos mesmos em *DataFrame*.
- É simples adicionar colunas e valores em um *DataFrame*.

Figura 2 – Adição de colunas e valores

```
import numpy as np
import pandas as pd

df = pd.DataFrame(data=np.array([[1, 2, 3],
                                [4, 5, 6], [7, 8, 9]]),
                  columns=['A', 'B', 'C'])

# Usando a propriedade `.index`
df['D'] = df.index
# Imprimindo o DataFrame `df`
print(df)
```

Fonte: elaborado pelo autor.

Porque Python, Spark e Hadoop e preparação do ambiente em Spark e Python

Bloco 2

Anderson Paulo Ávila Santos



► Pandas exclusão

- Para possibilitar exclusão de uma coluna do *DataFrame*, pode ser utilizado o método `drop()`.
- A deleção é simples, pode ser feita utilizando o nome da coluna ou seu respectivo índice:

Figura 3 – Deleção de colunas

```
# Deleta a coluna com o índice 'A'  
df.drop('A', axis=1, inplace=True)
```

```
# Deleta a coluna com posição 1  
df.drop(df.columns[[1]], axis=1)
```

Fonte: elaborado pelo autor.

► Pandas exclusão e renomeação

- É possível remover linhas do seu *DataFrame*, levando em conta apenas os valores duplicados que existem em uma coluna.

```
df.drop_duplicates([48], keep='last')
```

- É possível renomear o índice ou colunas de um *DataFrame* do Pandas.

► Pandas exclusão e renomeação

- Como renomear o índice ou colunas de um *DataFrame* do Pandas?

Figura 4 – Renomeação de um *DataFrame*.

```
# Define os novos nomes para as colunas
newcols = {
    'A': 'new_column_1',
    'B': 'new_column_2',
    'C': 'new_column_3'
}

# Usa o comando 'rename()' para renomear as colunas
df.rename(columns=newcols, inplace=True)

# renomeia os índices
df.rename(index={1: 'a'})
```

Fonte: elaborado pelo autor .

PÓS-GRADUAÇÃO

Teoria em prática

Bloco 3

Anderson Paulo Ávila Santos





► Teoria em prática

- Em Python existem algumas bibliotecas que são disponibilizadas para facilitar a manipulação de dados.
- Uma das principais e mais utilizadas é o Pandas, que disponibiliza um tipo de estrutura de manipulação de dados chamado *DataFrame*.
- Para a manipulação de dados sobre clientes, uma empresa precisa importar os dados presentes em arquivos CSV e remover/ alterar algumas informações contidas nesses dados.

► Exercício 1

Utilizando *DataFrame*, qual o comando deverá ser utilizado para realizar a importação desses dados?

- a) `import_csv()`.
- b) `to_csv()`.
- c) `read_csv()`.
- d) `import_file()`.
- e) `export_csv()`.

► Exercício 2

Caso exista a necessidade de se substituir todas as ocorrências de uma sequência de caracteres em um *DataFrame*, qual seria o comando e os parâmetros a serem utilizados?

- a) `replace(list)`.
- b) `sub(list,list)`.
- c) `replace(list,list)`.
- d) `change(list)`.
- e) `sub(list)`.

► Exercício 3

Para melhor formatação de uma coluna de dados, utilizando função lambda, qual função que deve ser utilizada?

- a) map().
- b) lamb().
- c) lambda().
- d) map_func().
- e) lamb_func().

PÓS-GRADUAÇÃO

Dica do professor

Bloco 4

Anderson Paulo Avila Santos





► Dicas de leitura

- Documentação do Pandas - *Pandas: powerful Python data analysis toolkit*, disponível no site oficial do Pandas.
- Documentação do Numpy e Scipy – *Numpy and Scipy Documentation*, disponível no site oficial do Scipy.

► Bibliografia

PRESS, G. Cleaning Big Data: most time-consuming, least enjoyable data science task, Survey Says. **Forbes**, [s.l.], 2016. Disponível em: <https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#326d062d6f63>. Acesso em: 15 jan. 2020.

WILLEMS, K. Apache Spark in Python: Beginner's Guide: a beginner's guide to Spark in Python based on 9 popular questions, such as how to install PySpark in Jupyter Notebook, best practices. **DataCamp Community**, [s.l.], 2017. Disponível em: <https://www.datacamp.com/community/tutorials/apache-spark-python>. Acesso em: 15 jan. 2020.

SILVA, R. O.; SILVA, I. R. S. Linguagem de programação Python. **Tecnologias em projeção**, v. 10, n. 1, 2019.

