DATA MINING

LABDATA

FUNDAÇÃO INSTITUTO DE ADMINISTRAÇÃO

Análise de Dados e Data Mining

Tema da Aula: **Aula 2 - Introdução ao Pandas**

Prof.: **Dino Magri**

Coordenação:

Prof. Dr. Adolpho Walter Pimazzi Canton

Profa. Dra. Alessandra de Ávila Montini

- Contatos:
  - E-mail: professor.dinomagri@gmail.com
  - Twitter: https://twitter.com/prof_dinomagri
  - LinkedIn:http://www.linkedin.com/in/dinomagri
  - Site: http://www.dinomagri.com

Coordenação:

Prof. Dr. Adolpho Walter Pimazzi Canton

Profa. Dra. Alessandra de Ávila Montini

Currículo

- **(2014-Presente)** – Professor no curso de Extensão, Pós e MBA na Fundação Instituto de Administração (FIA) – www.fia.com.br
- **(2018-Presente)** – Pesquisa e Desenvolvimento de Big Data e Machine Learning na Beholder (http://beholder.tech)
- **(2013-2018)** – Pesquisa e Desenvolvimento no Laboratório de Arquitetura e Redes de Computadores (LARC) na Universidade de São Paulo – www.larc.usp.br
- **(2012)** – Bacharel em Ciência da Computação pela Universidade do Estado de Santa Catarina (UDESC) – www.cct.udesc.br
- **(2009/2010)** – Pesquisador e Desenvolvedor no Centro de Computação Gráfica – Guimarães – Portugal – www.ccg.pt
- **Lattes:** http://lattes.cnpq.br/5673884504184733

Material das aulas

- Caso esteja utilizando seu próprio computador, realize o download de todos os arquivos e salve na **Área de Trabalho** para facilitar o acesso.
  - Lembre-se de instalar os softwares necessários conforme descrito no documento de Instalação (**`InstalaçãoPython3v1.2.pdf`**).

- Nos computadores da FIA os arquivos já estão disponíveis, bem como a instalação dos softwares necessários.

Conteúdo da Aula

- Objetivo
- Introdução ao Pandas
- Exercícios
- Referências Bibliográficas

Conteúdo da Aula

- **Objetivo**
- Introdução ao Pandas
- Exercícios
- Referências Bibliográficas

Objetivo

- O objetivo dessa aula é introduzir os conceitos básicos sobre a biblioteca **Pandas** para realizar **análise de dados**.

Conteúdo da Aula

- Objetivo
- **Introdução ao Pandas**
- Exercícios
- Referências Bibliográficas

Pandas

- Pandas é uma biblioteca de código aberto para análise de dados em Python.
- Foi desenvolvido em 2008 por Wes McKinney.
  - Tem uma grande comunidade - http://pandas.pydata.org/community.html
  - Melhorias contínuas - https://github.com/pydata/pandas/
  - Diversas funcionalidades
  - Iteração rápida

- Essa biblioteca com o tempo teve uma **ótima adoção**, se tornando a **biblioteca padrão para análise de dados** em Python.

Pandas

- Principais características do Pandas:
  - É possível **processar diversos conjuntos de dados em diferentes formatos** (Series temporais, dados tabulares heterogêneos, e matrizes)
  - Podemos lidar com **diversas operações** nesses conjuntos de dados: filtragem, agrupamento, reordenamento, remodelação, junção, fatiamento, entre outros.

Pandas

- Principais características do Pandas
  - Facilidade de **importar dados** de diversas fontes como CSV, DB/SQL.
  - Facilidade de trabalhar com **dados** que estão **faltando**.
  - Tem uma boa integração com outras bibliotecas Python, como a statsmodels, SciPy, e scikit-lean.

Pandas

- Instalação
  - `pip install pandas`
    - `pytz` – biblioteca para calculo de `timezone`.
    - `numpy` – processamento de array numéricos.
    - `python-dateutil` – fornece extensão para o módulo de `datetime`.
    - `six` – fornece funções que permitem diminuir as diferenças entre as versões do Python 2 e 3.

Pandas

- Pandas foi construído em cima do NumPy e ele fornece diversas outras funcionalidades que não estão disponíveis no NumPy.
- **Permite criar estruturas de dados de fácil entendimento e que são rápidas**.
- Desta forma possibilita preencher a lacuna que existia entre Python e linguagens de programação como R.

Pandas

- As 2 principais estruturas de dados no Pandas:
  - **Series** e **DataFrame**

- Para fazer uso dessas estruturas, primeiro precisamos importar a biblioteca.

```
>>> import pandas as pd
```

Pandas - Series

- **Series** é na verdade um array NumPy de 1 dimensão **com rótulos**.
- Podemos criar Series da seguinte maneira:

```
>>> s = pd.Series(dados)
```

- Onde, `dados` pode ser um dos itens abaixo:
  - Um `numpy.ndarray`
  - Um `dicionário`
  - Um `valor escalar`

Pandas - Series

- Além da criação, podemos realizar operações como **fatiamento** (slicing), **atribuições**, aplicar funções aritméticas e estatísticas, entre tantos outros.

</> Abra o arquivo **"aula2-parte1-series.ipynb"**

Pandas - DataFrame

- **DataFrame** é um array 2D com rótulos nas colunas e nas linhas.
- Conceitualmente é semelhante a uma tabela ou planilha de dados.
- Tem as seguintes características:
  - Colunas podem ser de diferentes tipos: `float64, int, bool`.
  - Uma coluna do DataFrame é uma `Series`.
  - Podemos pensar que é um dicionário de Series, onde as colunas e linhas são indexadas, denota-se `index` para linhas e `columns` no caso de colunas.
  - Os índices são necessários para acesso rápido aos dados.
  - Seu **tamanho** é **mutável**: colunas podem ser inseridas e deletadas

Pandas - DataFrame

- Podemos criar DataFrame da seguinte maneira:

```
>>> df = pd.DataFrame(dados)
```

- Onde, `dados` pode ser:
  - Dicionário de ndarrays de 1D, listas, dicionários, ou Series
  - Array 2D do NumPy
  - Estruturado
  - Series
  - Outra estrutura DataFrame

Pandas - DataFrame

- Podemos realizar inúmeras operações, como **seleção**, **atribuição**, **remoção**, alinhamento, aplicar funções aritméticas e estatísticas entre outros.

</> Abra o arquivo **"aula2-parte2-dataframe.ipynb"**

Outras operações

- Conforme vimos o fatiamento e a indexação podem ser um pouco confusos.
  - Por exemplo, se uma Series tem um índice explicito de inteiros, uma operação como `s1[1]` irá utilizar o índice **explícito**, enquanto que uma operação de fatiamento como `s1[1:3]` irá utilizar o índice **implícito**, no mesmo estilo de Python. Exemplo:

```
>>> s1 = pd.Series(['a', 'b', 'c'], index=[1,3,5])
1    a
3    b
5    c
dtype: object
```

Outras operações

- Utilizando o índice **explicito** quando se está indexando irá produzir o resultado:

```
>>> s1[1]
'a'
```

- Utilizando o índice **implícito** quando se está fatiando irá produzir o resultado:

```
>>> s1[1:3]
3    b
5    c
dtype: object
```

Outras operações

- Como podemos perceber isso pode ser um pouco confuso no caso de índices de números inteiros.
- Por isso, Pandas fornece alguns **indexadores especiais** que explicitamente contém esquemas de acesso aos índices.
- Eles não são métodos funcionais e sim **atributos que expõe uma interface de fatiamento particular para o dados**.
- São eles: `loc`, `iloc`

Outras operações

- O atributo `loc` permite indexar e fatiar sempre utilizando o índice **explícito**.

```
>>> s1
1    a
3    b
5    c
>>> s1.loc[1]
'a'
>>> s1.loc[1:3]
1    a
3    b
```

Outras operações

- O atributo `loc` permite indexar e fatiar sempre utilizando o índice **implícito**.

```
>>> s1
1    a
3    b
5    c
>>> s1.iloc[1]
'b'
>>> s1.iloc[1:3]
3    b
5    c
```

</> Abra o arquivo **"aula2-parte2-dataframe.ipynb"**

Conteúdo da Aula

- Objetivo
- Introdução ao Pandas
- **Exercícios**
- Referências Bibliográficas

Exemplo prático

- Iremos utilizar o arquivo `capitais.csv` que é um arquivo que tem todas as capitais do Brasil, bem como a população e a área de cada capital (km2).
- O Pandas disponibiliza diversos métodos para carregar diferentes tipos de dados, segue alguns deles:
  - `pd.read_csv('caminho-ate-arquivo.csv', sep=';')`
  - `pd.read_excel('caminho-ate-arquivo.xlsx', 'Sheet1')`
  - `sql.read_frame(query, connection)` – necessita do módulo pandas.io

</> Abra o arquivo **"aula2-parte3-exemplo-pratico.ipynb"**

Exercícios

Utilizando o DataFrame `capitais` criado anteriormente, faça os seguintes exercícios:

- **Exercício 1** - Selecione todas as capitais que tenham área maior que 400 km2. Quantas foram?

- **Exercício 2** - Selecione as capitais que tenham população maior que 2 milhões.

Exercícios

- **Exercício 3** - Crie uma função que retorne uma lista contendo somente as capitais que começam com uma determinada letra. A função deve receber dois parâmetros:
  - O primeiro parâmetro será uma lista com **todas as capitais**.
  - O segundo será uma **letra**.
- Para testar a função, selecione as capitais que começam com as letras **B** e **Z**. Lembre-se de tratar possíveis erros.

```
>>> def capitais_com_letra(todas_capitais, letra):
```

Exercícios

- **Exercício 4** - Utilizando a função criada no exercício 3, calcule o total da população para as capitais que começam com **S**. Por fim, imprima a seguinte frase:

```
As capitais X, Y e Z tem W pessoas.
```

Exercícios

- **Exercício 5** - Selecione os itens que tenham população maior que 1 milhão e área menor que 500 km2.

- **Exercício 6** - Selecione os itens que tenham população maior que 5 milhões ou área maior que 5000 km2.

Conteúdo da Aula

- Objetivo
- Introdução ao Pandas
- Exercícios
- **Referências Bibliográficas**

Referências Bibliográficas

- **Mastering pandas** – Femi Anthony – Packt Publishing, 2015.
- **Python for Data Analysis** – Wes McKinney – USA: O'Reilly, 2013.
- Tutoriais disponíveis no site oficial do Pandas - http://pandas.pydata.org/pandas-docs/stable/
- Livro de receitas disponíveis no site oficial do Pandas - http://pandas.pydata.org/pandas-docs/stable/cookbook.html

Referências Bibliográficas

- **Use a Cabeça! Python** – Paul Barry - Rio de Janeiro, RJ: Alta Books, 2012.

- **Use a Cabeça! Programação** – Paul Barry & David Griffiths – Rio de Janeiro RJ: Alta Books, 2010.

- **Aprendendo Python: Programação orientada a objetos** – Mark Lutz & David Ascher – Porto Alegre: Bookman, 2007

Referências Bibliográficas

- **Python for kids – A playful Introduction to programming** – Jason R. Briggs – San Francisco – CA: No Starch Press, 2013.

- **Python for Data Analysis** – Wes McKinney – USA: O'Reilly, 2013.

- **Python Cookbook** – David Beazley & Brian K. Jones – O'Reilly, 3th Edition, 2013.

- As referências de links utilizados podem ser visualizados em http://urls.dinomagri.com/refs