Python para Data Science e Analysis

Conceitos básicos e essenciais

- *Ricardo Roberto de Lima*
- ***Analista de Tecnologia da Informação***
  Email: ricardo.rlima@dataprev.gov.br

Resumo dos Assuntos

Revisão de Python Bibliotecas para Cientistas de Dados

Lendo Dados, selecionando e filtrando dados, manipulando dados, ordenação, agrupamento, etc.

Plotagem dos Dados

Estatística Descritiva

Inferência Estatística

Python Bibliotecas para Ciência de Dados

Mais populares Python Ferramentas/Libs:

- NumPy
- SciPy
- Pandas
- SciKit-Learn

All these libraries are installed on the SCC

Bibliotecas de visualização

- matplotlib
- Seaborn

E muito mais …

Python Bibliotecas para Ciência de Dados

- Introduz objetos para matrizes multidimensionais e matrizes, bem como funções que permitem realizar facilmente operações matemáticas e estatísticas avançadas nesses objetos.

- Fornece vetorização de operações matemáticas em arrays e matrizes que melhoram significativamente o desempenho.

- Muitas outras bibliotecas Python são construídas em NumPy

Link: http://www.numpy.org/

Python Bibliotecas para Ciência de Dados

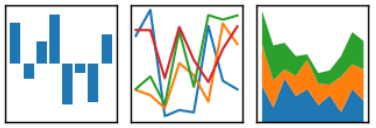
- Coleção de algoritmos para Álgebra Linear, equações diferenciais, integração numérica, otimização, estatísticas e muito mais.
- Parte da SciPy Stack
- Construído em NumPy

Link: https://www.scipy.org/scipylib/

Python Bibliotecas para Ciência de Dados

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



- Adiciona estruturas de dados e ferramentas projetadas para trabalhar com dados semelhantes a tabelas (semelhante a Series e Data Frames in R);
- Fornece ferramentas para manipulação de dados: reformulação, fusão, classificação, fatiamento, agregação, etc.
- Permite manipular dados ausentes.

Link: http://pandas.pydata.org/

Python Bibliotecas para Ciência de Dados

- Fornece os Algoritmos de Aprendizado de Máquina: Classificação, regressão, agrupamento, validação de modelo e estatística básica para os mesmo, etc.

- Criação e utilização com NumPy, SciPy and matplotlib

Link: http://scikit-learn.org/

Python Bibliotecas para Ciência de Dados

Biblioteca de plotagem 2D Python que produz figuras de qualidade de publicação em uma variedade de formatos impressos.

- Um conjunto de funcionalidades semelhantes às de MATLAB
- line plots, scatter plots, barcharts, histograms, pie charts etc.
- Nível relativamente baixo; algum esforço necessário para criar visualizações avançadas.

Link: https://matplotlib.org/

Python Bibliotecas para Ciência de Dados

***Seaborn**:*

- Baseado no Matplotlib
- Fornece interface de alto nível para desenhar gráficos estatísticos atraentes
- Similar (em estilo) à popular biblioteca ggplot2 em R.

Link: https://seaborn.pydata.org/

Práticas de Laboratório com o Jupyter

Use suas informações de login do SCC se você tiver uma conta do SCC.

- Se você estiver usando contas tutoriais, veja as informações no quadro-negro.
- Nota: sua senha não será exibida enquanto você a digitar.

Práticas de Laboratório com o Jupyter

Utilizando o Ambiente dom Jupyter Notebook

```
[scc1 ~] jupyter notebook
```

Práticas de Laboratório com o Jupyter

Carregando as Bibliotecas:

In [ ]:
```
#Import Python Libraries
import numpy as np
import scipy as sp
import pandas as pd
import matplotlib as mpl
import seaborn as sns
```

Press `Shift + Enter` to execute the *jupyter* cell

Práticas de Laboratório com o Jupyter

Lendo os dados utilizando o Pandas:

```
#Read csv file
df = pd.read_csv("http://rcs.bu.edu/examples/python/data_analysis/Salaries.csv")
```

Existe um número de comandos pandas para ler outros formatos de dados:

```
pd.read_excel('myfile.xlsx',sheet_name='Sheet1', index_col=None, na_values=['NA'])
pd.read_stata('myfile.dta')
pd.read_sas('myfile.sas7bdat')
pd.read_hdf('myfile.h5','df')
```

Práticas de Laboratório com o Jupyter

Explorando o DATAFRAME:

In [3]:
```
#List first 5 records
df.head()
```

Out[3]:

	rank	discipline	phd	service	sex	salary
0	Prof	B	56	49	Male	186960
1	Prof	A	12	6	Male	93000
2	Prof	A	23	20	Male	110515
3	Prof	A	40	31	Male	131205
4	Prof	B	20	18	Male	104800

Hands-on Exercícios

Mão na massa

Tente ler os primeiros 10, 20 e 50 registros.

Você consegue adivinhar como visualizar os últimos registros?

Data Frame Tipos de Dados

Pandas Type	Native Python Type	Description
object	string	The most general dtype. Will be assigned to your column if column has mixed types (numbers and strings).
int64	int	Numeric characters. 64 refers to the memory allocated to hold this character.
float64	float	Numeric characters with decimals. If a column contains numbers and NaNs(see below), pandas will default to float64, in case your missing value has a decimal.
datetime64, timedelta[ns]	N/A (but see the datetime module in Python's standard library)	Values meant to hold time data. Look into these for time series experiments.

Data Frame Tipos de Dados

In [4]:
```
#Check a particular column type
df['salary'].dtype
```

Out[4]:
```
dtype('int64')
```

In [5]:
```
#Check types for all the columns
df.dtypes
```

Out[4]:

rank	object
discipline	object
phd	int64
service	int64
sex	object
salary	int64

dtype: object

Data Frame Tipos de Dados

Objetos Python possuem atributos e métodos

df.attribute	descrição
dtypes	Lista dos tipos de colunas
columns	Lista dos Nomes das Colunas
axes	Listar os rótulos das linhas e os nomes das colunas
ndim	Número de dimensões
size	Número de elementos
shape	Retorno das tuplas representando as dimensões.
values	Representação NumPy dos Dados

Hands-on Exercícios

Mão na massa

Encontre quantos registros esse quadro de dados possui?

Quantos elementos existem?

Quais são os nomes das colunas?

Quais tipos de colunas nós temos nesse quadro de dados?

Data Frame Métodos

Ao contrário dos atributos, os métodos Python possuem parênteses.
Todos os atributos e métodos podem ser listados com uma função `dir()` function: `dir(df)`

df.method()	description
head([n]), tail([n])	Primeiras / Últimas n Linhas.
describe()	Gerar estatísticas descritivas (somente para colunas numéricas).
max(), min()	Retorna o max / min valores para todas as colunas numéricas.
mean(), median()	Retorna a média/mediana de valores para todas as colunas numericas.
std()	Desvio Padrão.
sample([n])	Retorna o exemplo randômico para o data frame.
dropna()	Apaga todos os registros com valores ausentes.

Hands-on Exercícios

Data Frame (Métodos)

Forneça o resumo das colunas numéricas no conjunto de dados?

Calcule o desvio padrão para todas as colunas numéricas?

Quais são os valores médios dos primeiros 50 registros no conjunto de dados?

Dica: *use o método head() para subdividir os primeiros 50 registros e depois calcule a média.*

Selecione uma coluna com Data Frame

Colunas dentro de um Data Frame

Método 1: Subconjunto do quadro de dados usando o nome da coluna:

```
df['sex']
```

Método 2: Use o nome da coluna como um atributo:

```
df.se
x
```

Nota: há uma classificação de atributo para quadros de dados de pandas, portanto, para selecionar uma coluna com um nome “rank”, devemos usar o método 1:

Hands-on Exercícios

Data Frame (Métodos)

Calcular as estatísticas básicas para a coluna salarial;

Encontre quantos valores na coluna salarial (use o método count);

Calcule o salário médio.

Data Frame (Group by) Métodos

Usando o método “group by”, podemos:
- Divida os dados em grupos com base em alguns critérios;
- Calcular estatísticas (ou aplicar uma função) para cada grupo;
- Semelhante à função dplyr() em R.

In [ ]:
```
#Group data using rank
df_rank = df.groupby(['rank'])
```

In [ ]:
```
#Calculate mean value for each numeric column per each group
df_rank.mean()
```

rank	phd	service	salary
AssocProf	15.076923	11.307692	91786.230769
AsstProf	5.052632	2.210526	81362.789474
Prof	27.065217	21.413043	123624.804348

Data Frame (Group by) Métodos

Uma vez que o objeto groupby é criado, podemos calcular várias estatísticas para cada grupo:

In [ ]:
```
#Group data using rank
df_rank = df.groupby(['rank'])
```

In [ ]:
```
#Calculate mean value for each numeric column per each group
df_rank.mean()
```

rank	phd	service	salary
AssocProf	15.076923	11.307692	91786.230769
AsstProf	5.052632	2.210526	81362.789474
Prof	27.065217	21.413043	123624.804348

Nota: Se forem usados colchetes únicos para especificar a coluna (por exemplo, salário), a saída será o objeto Série Pandas. Quando colchetes duplos são usados, a saída é um quadro de dados.

Data Frame (Groupby) Métodos

groupby Anotações de Performance:

- Nenhum agrupamento / divisão ocorre até que seja necessário. Criar o objeto groupby somente verifica se você passou por um mapeamento válido;

- Por padrão, as chaves do grupo são classificadas durante a operação groupby. Você pode querer passar sort = False para o potencial aumento de velocidade:

In [ ]:

```
#Calculate mean salary for each professor rank:
df.groupby(['rank'], sort=False)[['salary']].mean()
```

Data Frame (Groupby) Métodos

Para subdividir os dados, podemos aplicar a indexação booleana. Essa indexação é comumente conhecida como um filtro. Por exemplo, se quisermos subdividir as linhas nas quais o valor do salário é maior que $120k:

```
#Calculate mean salary for each professor rank:
df_sub = df[ df['salary'] > 120000 ]
```

Qualquer operador booleano pode ser usado para subconjunto dos dados:

> greater; >= greater or equal;

< less; <= less or equal;

== equal; != not equal;

In [ ]:

```
#Select only those rows that contain female professors:
df_f = df[ df['sex'] == 'Female' ]
```

Data Frame: Slicing

Existem várias maneiras de usar um subconjunto do Quadro de Dados:

- Uma ou mais colunas;
- Uma ou mais linhas;
- Um subconjunto de linhas e colunas.

Linhas e colunas podem ser selecionadas por sua posição ou rótulo.

Data Frame: Slicing

Ao selecionar uma coluna é possível usar um único conjunto de colchetes, mas o objeto resultante será uma série (não um DataFrame):

```
In [ ]:#Select column salary:
       df['salary']
```

Quando precisamos selecionar mais de uma coluna e/ou fazer com que a saída seja um DataFrame, devemos usar colchetes duplos:

```
In [ ]: #Select column salary:
        df[['rank','salary']]
```

Data Frame: Selecionando Linhas

Se precisarmos selecionar um intervalo de linhas, podemos especificar o intervalo usando ":"

```
In [ ]: #Select rows by their position:
        df[10:20]
```

Observe que a primeira linha tem uma posição 0 e o último valor no intervalo é omitido:

Então, para o intervalo de 0:10, as primeiras 10 linhas são retornadas com as posições começando com 0 e terminando com 9.

Data Frame: Método loc

Se precisarmos selecionar um intervalo de linhas, usando seus rótulos, podemos usar o método loc:

In [ ]:
```
#Select rows by their labels:
df_sub.loc[10:20,['rank','sex','salary']]
```

Out[ ]:

	rank	sex	salary
10	Prof	Male	128250
11	Prof	Male	134778
13	Prof	Male	162200
14	Prof	Male	153750
15	Prof	Male	150480
19	Prof	Male	150500

Data Frame: Método iloc

Se precisarmos selecionar um intervalo de linhas e/ou colunas, usando suas posições, podemos usar o método iloc:

In [ ]:
```
#Select rows by their labels:
df_sub.iloc[10:20,[0, 3, 4, 5]]
```

Out[ ]:

	rank	service	sex	salary
26	Prof	19	Male	148750
27	Prof	43	Male	155865
29	Prof	20	Male	123683
31	Prof	21	Male	155750
35	Prof	23	Male	126933
36	Prof	45	Male	146856
39	Prof	18	Female	129000
40	Prof	36	Female	137000
44	Prof	19	Female	151768
45	Prof	25	Female	140096

Data Frame: Método iloc (Resumo)

```
df.iloc[0]  # First row of a data frame
df.iloc[i]  #(i+1)th row
df.iloc[-1] # Last row
```

```
df.iloc[:, 0]  # First column
df.iloc[:, -1] # Last column
```

```
df.iloc[0:7]         #First 7 rows
df.iloc[:, 0:2]      #First 2 columns
df.iloc[1:3, 0:2]    #Second through third rows and first 2 columns
df.iloc[[0,5], [1,3]]  #1st and 6th rows and 2nd and 4th columns
```

Data Frame: Classificação (Sorting)

Podemos classificar os dados por um valor na coluna. Por padrão, a classificação ocorrerá em ordem crescente e um novo quadro de dados será retornado.

In [ ]:
```
# Create a new data frame from the original sorted by the column Salary
df_sorted = df.sort_values( by ='service')
df_sorted.head()
```

Out[ ]:

	rank	discipline	phd	service	sex	salary
55	AsstProf	A	2	0	Female	72500
23	AsstProf	A	2	0	Male	85000
43	AsstProf	B	5	0	Female	77000
17	AsstProf	B	4	0	Male	92000
12	AsstProf	B	1	0	Male	88000

Data Frame: Classificação (Sorting)

Podemos classificar os dados em duas ou mais Colunas:

In [ ]:
```
df_sorted = df.sort_values( by =['service', 'salary'],
ascending = [True, False])
df_sorted.head(10)
```

Out[ ]:

	rank	discipline	phd	service	sex	salary
52	Prof	A	12	0	Female	105000
17	AsstProf	B	4	0	Male	92000
12	AsstProf	B	1	0	Male	88000
23	AsstProf	A	2	0	Male	85000
43	AsstProf	B	5	0	Female	77000
55	AsstProf	A	2	0	Female	72500
57	AsstProf	A	3	1	Female	72500
28	AsstProf	B	7	2	Male	91300
42	AsstProf	B	4	2	Female	80225
68	AsstProf	A	4	2	Female	77500

Valores Ausentes

Valores ausentes são marcados como: NaN

In [ ]:
```
# Read a dataset with missing values
flights =
pd.read_csv("http://rcs.bu.edu/examples/python/data_analysis/
flights.csv")
```

In [ ]:
```
# Select the rows that have at least one missing value
flights[flights.isnull().any(axis=1)].head()
```

Out[ ]:

	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
330	2013	1	1	1807.0	29.0	2251.0	NaN	UA	N31412	1228	EWR	SAN	NaN	2425	18.0	7.0
403	2013	1	1	NaN	NaN	NaN	NaN	AA	N3EHAA	791	LGA	DFW	NaN	1389	NaN	NaN
404	2013	1	1	NaN	NaN	NaN	NaN	AA	N3EVAA	1925	LGA	MIA	NaN	1096	NaN	NaN
855	2013	1	2	2145.0	16.0	NaN	NaN	UA	N12221	1299	EWR	RSW	NaN	1068	21.0	45.0
858	2013	1	2	NaN	NaN	NaN	NaN	AA	NaN	133	JFK	LAX	NaN	2475	NaN	NaN

Valores Ausentes

Existem várias métodos para lidar com valores ausentes no DataFrame:

df.method()	description
dropna()	Elimina observações faltantes
dropna(how='all')	Elimina observações em que todas as células são NA
dropna(axis=1, how='all')	Alarga a coluna se todos os valores estiverem em falta
dropna(thresh = 5)	Elimina linhas que contém menos de 5 valores não ausentes
fillna(0)	Substituir valores ausentes por zeros
isnull()	Retorna True se o valor estiver faltando
notnull()	Retorna True para valores não ausentes

Valores Ausentes

- Ao somar os dados, os valores omissos serão tratados como zero;
- Se todos os valores estiverem faltando, a soma será igual a NaN;
- Os métodos cumsum() e cumprod() ignoram os valores ausentes, mas os preservam nos arrays resultantes;
- Valores ausentes do método GroupBy são excluídos (assim como no R);
- Muitos métodos de estatística descritiva têm a opção skipna para controlar se os dados ausentes devem ser excluídos. Este valor é definido como True por padrão (diferente de R).

Funções de Agregação em Pandas

Aggregation – calcular uma estatística de resumo sobre cada grupo, ou seja.

- Computar somas ou meios de grupo
- Computar tamanhos de grupos de computação / contagens

Funções de Agregação comuns:

min, max
count, sum, prod
mean, median, mode, mad
std, var

Funções de Agregação em Pandas

Agg() - são métodos usados quando múltiplas estatísticas são computadas por coluna:

In [ ]:
```
flights[['dep_delay','arr_delay']].agg(['min','mean','max'])
```

Out[ ]:

	dep_delay	arr_delay
min	-16.000000	-62.000000
mean	9.384302	2.298675
max	351.000000	389.000000

Estatística Descritiva Básica

df.method()	description
describe	Basic statistics (count, mean, std, min, quantiles, max)
min, max	Minimum and maximum values
mean, median, mode	Arithmetic average, median and mode
var, std	Variance and standard deviation
sem	Standard error of mean
skew	Sample skewness
kurt	kurtosis

Gráficos e exploração dos Dados

O pacote **Seaborn** é construído sobre o **matplotlib**, mas fornece uma interface de alto nível para desenhar gráficos estatísticos atraentes, semelhante à biblioteca **ggplot2** do **R**.

Ele visa especificamente a visualização de dados estatísticos.

Para mostrar gráficos dentro do Python Notebook, inclua a diretiva inline:

```
In [ ]: %matplotlib inline
```

Gráficos – Tipos deles

Nome	description
distplot	histogram
barplot	estimate of central tendency for a numeric variable
violinplot	similar to boxplot, also shows the probability density of the data
jointplot	Scatterplot
regplot	Regression plot
pairplot	Pairplot
boxplot	boxplot
swarmplot	categorical scatterplot
factorplot	General categorical plot

Análise estatística básica

statsmodel e scikit-learn – Ambos têm várias funções para Análise Estatística

O primeiro é usado principalmente para análises regulares usando fórmulas de estilo R, enquanto o Sckit-learnin é mais adaptado para o Machine Learning (Aprendizado de Máquina).

statsmodels:

- linear regressions
- ANOVA tests
- hypothesis testings
- many more ...

scikit-learn:

- kmeans
- support vector machines
- random forests
- many more ...

Contato

Ricardo Roberto de Lima

Plataforma Lattes (CNPQ) -http://lattes.cnpq.br/8352116355228450
Email: ricardorobertolima@gmail.com