Introdução a Data Science usando Python

Jonathan da Silva Braga - SECEM II 2017

- site: http://jonathan.orbt.tech
- github: https://github.com/jonathanbraga
- e-mail: jonathanb2br@gmail.com
- empresa: http://orbt.tech

- Pandas é contruido em cima do pacote Numpy, porém com alguns atributos voltados especificamente para o estudo de dados. Um maior exemplo disse é o tipo array que no numpy, precisa ter um tipo especifico para os dados. No pandas os "arrays" podem ter tipos diversos, não precisando ser homogênio (Series e DataFrame).

```
import pandas as pd
```

```
dado = pd.Series([0.30, 0.9, 6])
dado.index
```

```
RangeIndex(start=0, stop=3, step=1)
```

- start: número inicial da sequência.
- stop: gere números até, mas não incluindo esse número.
- step: diferença entre cada número na seqüência.

Índices

```
dado_indice = pd.Series([1,2,3], index=['a','b','c'])
dado_indice
```

```
a    1
b    2
c    3
dtype: int64
```

Acessando dados via índice

```
print(dado_indice['b'])
print(dado_indice[1])
print(dado_indice[['a','b']])
```

```
2
2
a    1
b    2
dtype: int64
```

Filtros no acesso aos dados

```
dado_indice[dado_indice > 1]
```

```
b    2
c    3
dtype: int64
```

Dicionários

```
dado_dicionario = {'Bola': 1, 'Caixa': 3, 'Cadeira': 7, 'Mesa': 10}
dado_dicionario
```

```
{'Bola': 1, 'Cadeira': 7, 'Caixa': 3, 'Mesa': 10}
```

```
dado_dicionario_serie = pd.Series(dado_dicionario)
dado_dicionario_serie
```

```
Bola        1
Cadeira     7
Caixa       3
Mesa       10
```

```
dtype: int64
```

```
lista = ['Maçã', 'Banana', 'Uva', 'Pera', 'Laranja']
quantidade = [5,6, 8, 2, 9]
dado_list = pd.Series(quantidade, index=lista)
dado_list
```

```
Maçã       5
Banana     6
Uva        8
Pera       2
Laranja    9
dtype: int64
```

```
lista2 = ['Maçã', 'Banana', 'Uva', 'Pera', 'Laranja', 'Goiaba']
quantidade2 = [5,6, 8, 2, 9, 10]
dado_list2 = pd.Series(quantidade2, index=lista2)
dado_list2
```

```
Maçã        5
Banana      6
Uva         8
Pera        2
Laranja     9
Goiaba     10
dtype: int64
```

```
total = dado_list + dado_list2
total
```

```
Banana     12.0
Goiaba      NaN
Laranja    18.0
Maçã       10.0
Pera        4.0
Uva        16.0
dtype: float64
```

NAN

- Indica a falta de dados (not a number)

Técnicas para poder tratar ou eliminar o NAN

```
total[total.notnull()]
```

```
Banana     12.0
Laranja    18.0
Maçã       10.0
Pera        4.0
Uva        16.0
dtype: float64
```

```
total.dropna()
```

```
Banana     12.0
Laranja    18.0
Maçã       10.0
Pera        4.0
Uva        16.0
dtype: float64
```

```
total.fillna(999)
```

```
Banana      12.0
Goiaba     999.0
Laranja     18.0
Maçã        10.0
Pera         4.0
Uva         16.0
dtype: float64
```

Métodos eficientes para se especificar um dado

```
data = pd.Series(['a','b',
```

```
'c','d','e','f','g','h','i','j','l','m','n','o','p','q','r'], index=
[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17])
data
```

```
1     a
2     b
3     c
4     d
5     e
6     f
7     g
8     h
9     i
10    j
11    l
12    m
13    n
14    o
15    p
16    q
17    r
dtype: object
```

```
data[1]
```

```
'a'
```

```
data[3:10]
```

```
4     d
5     e
6     f
7     g
8     h
9     i
10    j
dtype: object
```

LOC x ILOC

- Loc: É um método baseado em rótulos. Isso significa que levará em consideração os nomes ou rótulos
- ILoc: Toma fatias com base na posição do índice. Para aqueles familiarizados com Python, ele se comporta como um corte regular. Você apenas indica o número do índice de posição

```
print(data.loc[1])
```

```
a
```

```
print(data.iloc[1])
```

```
b
```

DataFrame

Assim, DataFrame Pandas equivale a uma Matriz com índices explícitos, porém, diferentemenete dos arrays NumPy, os elementos podem ser de tipos diferentes.

```
dadosFrame = pd.DataFrame(data)
dadosFrame
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0
1	a
2	b
3	c
4	d
5	e

6	f
7	g
8	h
9	i
10	j
11	l
12	m
13	n
14	o
15	p
16	q
17	r

```
preco = [2,7,4,8,9,10]
carro = ['Uno', 'Fusco','Ferrari', 'Corsa', 'Prisma', 'Palio']
cidade = ['Fortaleza', 'Natal', 'Recife', 'Mossoró', 'Assu', 'Santa Cruz']
data_carros = pd.DataFrame({'Preço':preco, 'Carro':carro, 'Cidade':cidade})
data_carros
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	Carro	Cidade	Preço
0	Uno	Fortaleza	2
1	Fusco	Natal	7
2	Ferrari	Recife	4
3	Corsa	Mossoró	8
4	Prisma	Assu	9

5	Palio	Santa Cruz	10

```
data_carros['Cidade']
```

```
0     Fortaleza
1         Natal
2        Recife
3       Mossoró
4          Assu
5    Santa Cruz
Name: Cidade, dtype: object
```

Tratando NaN em dataframe

```
dataF = pd.DataFrame([[1., 6.5, 3.], [1., None, None], [None, None, None],
[None, 6.5, 3.]])
dataF
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0	1	2
0	1.0	6.5	3.0
1	1.0	NaN	NaN
2	NaN	NaN	NaN
3	NaN	6.5	3.0

```
dataF.dropna(how='all')
```

```
.dataframe thead th {
```

```
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0	1	2
0	1.0	6.5	3.0
1	1.0	NaN	NaN
3	NaN	6.5	3.0

```
df = pd.DataFrame(np.random.randn(7, 3))
df
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0	1	2
0	0.708095	-1.700263	0.246891
1	-1.025046	-1.369640	-1.097429
2	-1.816795	-0.584778	-0.529114
3	2.270005	-0.584807	0.368513
4	-1.913609	-1.111615	0.833532
5	0.754387	0.216073	1.177428
6	0.015802	1.345364	1.141771

```
df.ix[:, 2] = None;
df
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:1:
DeprecationWarning:
.ix is deprecated. Please use
.loc for label based indexing or
.iloc for positional indexing

See the documentation here:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#deprecate_ix
  """Entry point for launching an IPython kernel.
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0	1	2
0	0.708095	-1.700263	None
1	-1.025046	-1.369640	None
2	-1.816795	-0.584778	None
3	2.270005	-0.584807	None
4	-1.913609	-1.111615	None
5	0.754387	0.216073	None
6	0.015802	1.345364	None

```
df.fillna(0)
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

	0	1	2

0	0.708095	-1.700263	0
1	-1.025046	-1.369640	0
2	-1.816795	-0.584778	0
3	2.270005	-0.584807	0
4	-1.913609	-1.111615	0
5	0.754387	0.216073	0
6	0.015802	1.345364	0

Grtáficos

```
import matplotlib.pyplot as plt
```

```
data_g =
pd.read_csv('https://raw.githubusercontent.com/jakevdp/PythonDataScienceHandbo
ok/master/notebooks/data/state-population.csv')
data_g.head()
```

```
.dataframe thead th {
    text-align: left;
}

.dataframe tbody tr th {
    vertical-align: top;
}
```

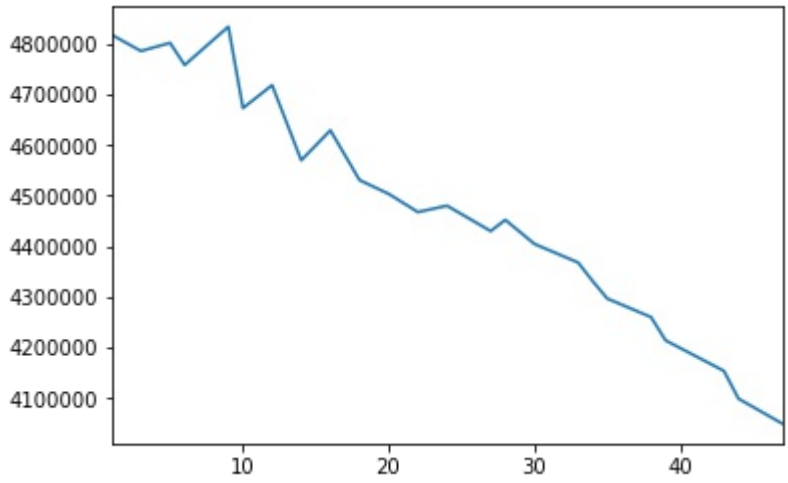
	state/region	ages	year	population
0	AL	under18	2012	1117489.0
1	AL	total	2012	4817528.0
2	AL	under18	2010	1130966.0
3	AL	total	2010	4785570.0
4	AL	under18	2011	1125763.0

```
x_aux = data_g.iloc[:,0] == 'AL'
valores = data_g[x_aux][data_g.ages == 'total']
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:2:
UserWarning: Boolean Series key will be reindexed to match DataFrame index.
```

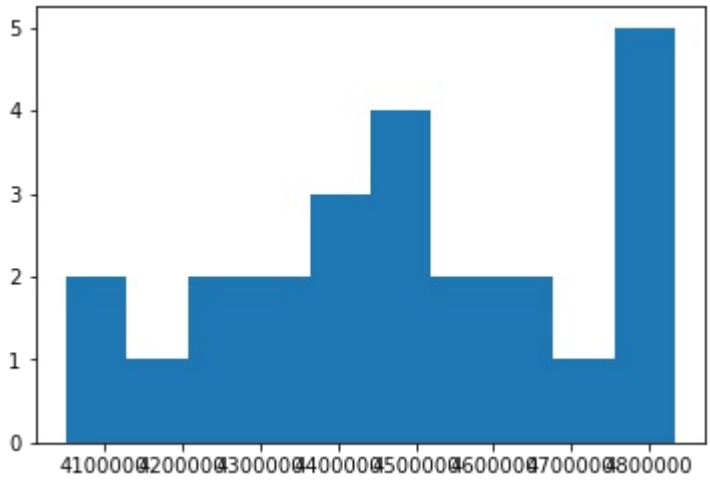
```
%matplotlib inline
valores['population'].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2d19be1e8d0>
```




```
plt.hist(valores['population'])
```

```
(array([ 2.,  1.,  2.,  2.,  3.,  4.,  2.,  2.,  1.,  5.]),
 array([ 4050055. ,  4128421.7,  4206788.4,  4285155.1,  4363521.8,
         4441888.5,  4520255.2,  4598621.9,  4676988.6,  4755355.3,
         4833722. ]),
 <a list of 10 Patch objects>)
```




```
plt.scatter(valores.year, valores.population)
```

```
<matplotlib.collections.PathCollection at 0x2d19d73a748>
```