Biblioteca
Pandas

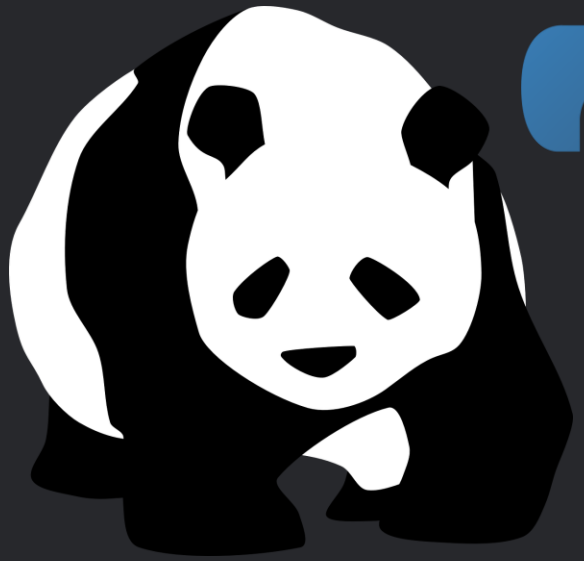
Prof. Dr. Diego Bruno

Education Tech Lead na DIO
Doutor em Robótica e *Machine Learning* pelo ICMC-USP

Biblioteca **Pandas**

Prof. Dr. Diego Bruno

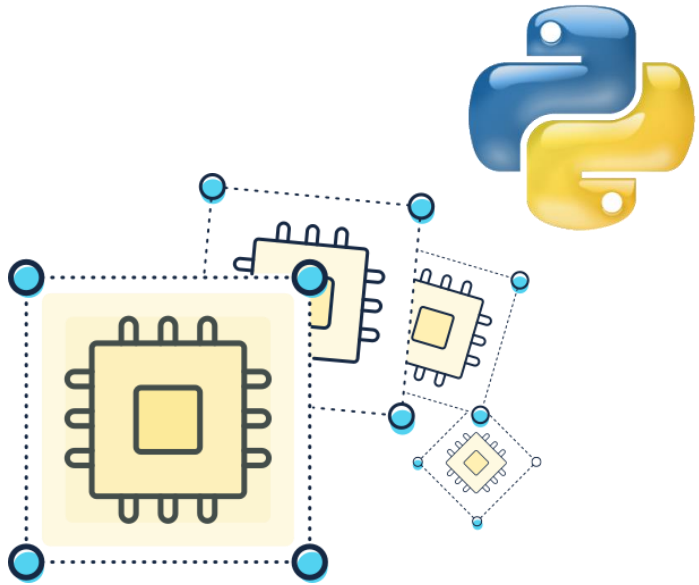
Machine Learning



Utilizando a biblioteca

O pandas permite trabalhar com diferentes tipos de dados, por exemplo:

→ Dados tabulares, como uma planilha Excel ou uma tabela SQL;
→ Dados ordenados de modo temporal ou não;
Matrizes;
→ Qualquer outro conjunto de dados,
que não necessariamente precisem estar rotulados;



Bibliotecas de exemplo:

pillow

scikit-image
image processing in python

Estruturas de Dados

- Os dois principais objetos da biblioteca Pandas são as **Series** e os **DataFrames**:

- → Uma Serie é uma matriz unidimensional que contém uma sequência de valores que apresentam uma indexação (que podem ser numéricos inteiros ou rótulos), muito parecida com uma única coluna no Excel.

- → Já o DataFrame é uma estrutura de dados tabular, semelhante a planilha de dados do Excel, em que tanto as linhas quanto as colunas apresentam rótulos.

Estruturas de Dados

- Os dois principais objetos da biblioteca Pandas são as **Series** e os **DataFrames**:

Series

	nome
0	Maria
1	José
2	Ana
3	Paulo

Índice

	idade
0	25
1	56
2	31
3	43

Linhas

DataFrame

Colunas

	nomes	idade
0	Maria	25
1	José	56
2	Ana	31
3	Paulo	43

Quais as vantagens?

- Comandos equivalentes, com a mesma funcionalidade no Pandas;

R	pandas
`dim(df)`	`df.shape`
`head(df)`	`df.head()`
`slice(df, 1:10)`	`df.iloc[:9]`
`filter(df, col1 == 1, col2 == 1)`	`df.query('col1 == 1 & col2 == 1')`
`df[df$col1 == 1 & df$col2 == 1,]`	`df[(df.col1 == 1) & (df.col2 == 1)]`

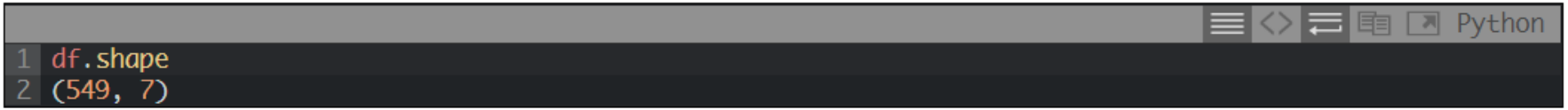
DataFrame

- Agora se visualizarmos novamente os primeiros 4 dados do nosso conjunto, veremos que todos os valores passados para ***na_values***, além dos próprios dados ausentes, foram substituídos por NaN.

```
1 df.head(n=4)
```

	id	data_aq	produto	quantidade	valor UN	TotaL	setor
0	0	01/01/2019	toalha	6	R$ 35,00	R$ 210,00	Mesa_banho
1	1	02/01/2019	toalha	6	R$ 35,00	R$ 210,00	NaN
2	2	03/01/2019	toalha	2	R$ 35,00	R$ 70,00	mesa_banho
3	3	01/02/2019	toalha	5	R$ 35,00	R$ 175,00	NaN

df.shape



df.info

```python
df.info()
```

Já para saber que formato se encontram os dados em cada coluna, além da quantidade de memória para ler esse conjunto de dados, podemos utilizar o comando *info*:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 549 entries, 0 to 548
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   id          549 non-null    int64 
 1   data_aq     549 non-null    object
 2   produto     542 non-null    object
 3   quantidade  549 non-null    int64 
 4   valor UN    549 non-null    object
 5   TotaL       549 non-null    object
 6   setor       535 non-null    object
dtypes: int64(2), object(5)
memory usage: 30.1+ KB
```

Alterações

Na sequência, podemos visualizar quais são nossas colunas existentes e até mesmo alterar esses nomes, basta passar o novo conjunto de nomes desejados com a mesma quantidade de colunas existente no conjunto original:

```
df.columns
Index(['id', 'data_aq', 'produto', 'quantidade', 'valor UN', 'Total', 'setor'], dtype='object')

df.columns = ['id', 'data_aq', 'produto', 'quantidade', valor_un', 'valor_total', setor']
```

Verificação

Para verificar quantos dados faltantes existem em nosso conjunto

```
1 df.isnull().sum()
```

```
id              0
data_aq         0
produto         7
quantidade      0
valor_un        0
valor_total     0
setor          14
dtype: int64
```

Valores únicos

No nosso objeto Serie podemos verificar quais os valores únicos existem naquela coluna, com o método *unique*:

```
1 df['setor'].unique()
```

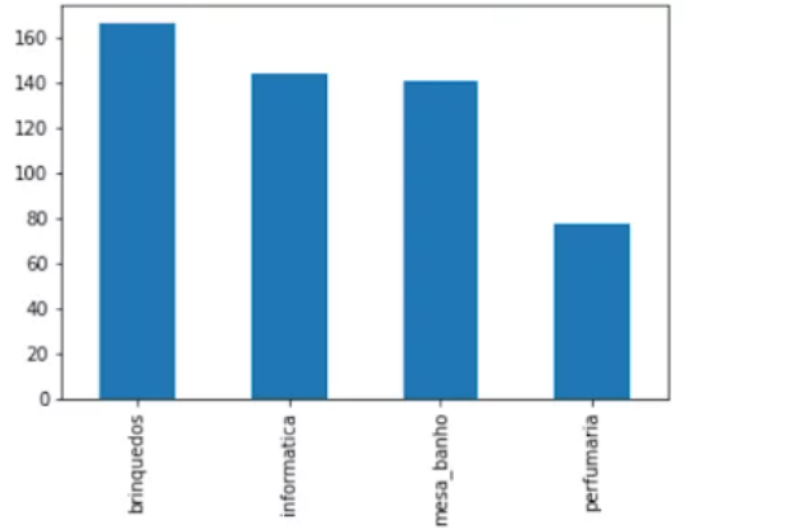
```
array(['Mesa_banho', 'mesa_banho', 'mesa_Banho', 'perfumaria',
       'informatica', 'Informatica', 'informaticA', 'brinquedos',
       'brinquEdos', 'Brinquedos'], dtype=object)
```

Agrupamentos

Ainda a partir desse método podemos gerar uma visualização simples e rápida com o resultado. Como? Com o método *plot*.

```
1 df['setor'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f18fd27be90>
```



Dados estatísticos

o Pandas colabora na exibição de algumas informações estatísticas a respeito do nosso conjunto de dados e permite que possamos realizar facilmente uma análise com o nosso objeto DataFrame

```
1 df.describe()
```

	id	quantidade	valor_total
count	528.000000	528.000000	528.000000
mean	275.229167	6.032197	280.739848
std	157.431587	3.207100	383.826301
min	0.000000	1.000000	2.990000
25%	137.750000	3.000000	64.950000
50%	276.000000	6.000000	142.890000

Obrigado!

Machine Learning

Prof. Dr. Diego Bruno