

Ferramentas e Bibliotecas PYTHON para Data Science

Vanessa Sayuri Uchida

- Formada em Engenharia Civil - UFRN
- Graduanda em Tecnologia da Informação - UFRN



Sumário

- Ferramentas
 - Jupyter Notebook
 - Google Colab Notebook
- Bibliotecas
 - Numpy
 - Scipy
 - Pandas
 - Matplotlib
 - Seaborn



Ferramentas importantes

Jupyter Notebook
Google Colab Notebook

1

Jupyter Notebook



- Acrônimo das palavras Julia, Python e R;
- Permite editar e rodar notebooks via navegador;
- Pode ser usado mesmo sem acesso a internet.



Home



localhost:8888/tree/Palestra%20Grupy-rn



jupyter

Logout

Files

Running

Clusters

Select items to perform actions on them.

Upload

New



/ Palestra Grupy-rn

Name

Last Modified



..

segundos atrás



drive-download-20181028T172318Z-001

5 días atrás



Imagens

3 días atrás



Testes

3 días atrás



Análise de dados de Acidentes - PRF_2.ipynb

14 horas atrás



Análise de dados de Acidentes - PRF-Jupyter.ipynb

um dia atrás



TesteGraficos.ipynb

14 horas atrás



Untitled.ipynb

um dia atrás



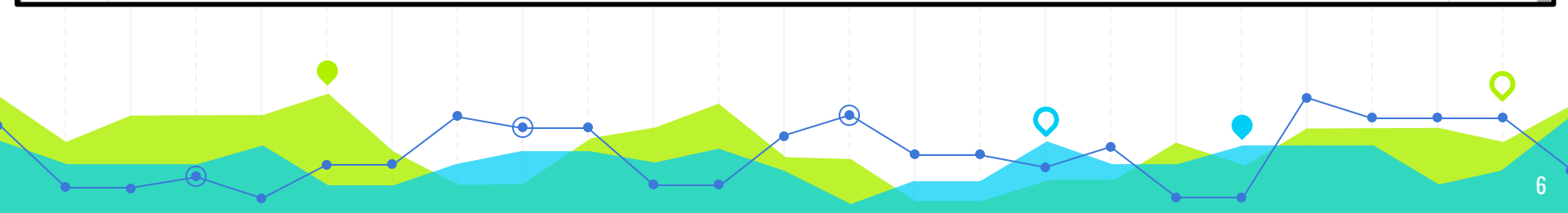
cheatsheet_pandas.jpg

6 días atrás



datatran2018.csv

5 días atrás



Home x Analise de dados de Acide x +

localhost:8888/notebooks/Palestra%20Grupy-rn/Analise%20de%20dados%20de%20Acidentes%20-%20PRF-Jupyter... ☆ V

jupyter Analise de dados de Acidentes - PRF-Jupyter (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown

Análise de Dados de Acidentes nas Rodovias Federais do Brasil em 2018

Ler o arquivo CSV

```
In [6]: import pandas as pd
prf_acidentes = pd.read_csv("datatran2018.csv", encoding = "Latin-1", sep=";")
```

```
In [7]: prf_acidentes
```

```
Out[7]:
```

	id	data_inversa	dia_semana	horario	uf	br	km	municipio	causa_acidente	tipo_acidente
0	99973.0	2018-01-01	segunda-feira	00:20:00	RJ	116.0	303,5	RESENDE ...	Condutor Dormindo ...	Saída de l ... carroçave
1	99976.0	2018-01-01	segunda-feira	00:40:00	SC	282.0	0,4	FLORIANOPOLIS ...	Não guardar distância de segurança ...	Colis ... traseira
2	99977.0	2018-01-01	segunda-feira	00:30:00	RJ	493.0	1	ITABORAÍ ...	Ultrapassagem Indevida ...	Colisão fro ...
3	99981.0	2018-01-01	segunda-feira	01:15:00	RS	386.0	134	SARANDI ...	Ingestão de Alcool ...	Colis ... transversa
4	99982.0	2018-01-01	segunda-feira	00:20:00	RS	293.0	151,7	CANDIOTA ...	Falta de Atenção à Condução ...	Saída de l ... carroçave



[Link do arquivo "datatran2018.csv"](#)



File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3

File Edit View Insert Cell Kernel Widgets Help

Run Markdown

Análise de Dados de Acidentes nas Rodovias Federais do Brasil em 2018

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3

File Edit View Insert Cell Kernel Widgets Help

Run Code

Análise de Dados de Acidentes nas Rodovias Federais do Brasil em 2018



Ler o arquivo CSV

```
In [10]: import numpy as np
prf_np = np.genfromtxt("datatran2018.csv", encoding = "Latin-1", delimiter=",")
```







Google Colab Notebook



- Permite abrir e executar arquivos com a extensão .ipynb sem a necessidade de instalar;
- Precisa de acesso a internet;
- Similar ao google docs.

 Overview of Colaboratory Features 

File Edit View Insert Runtime Tools Help

 CODE  TEXT  CELL  CELL  COPY TO DRIVE




CONNECT  EDITING 

Table of contents Code snippets Files 

Cells


- Code cells
- Text cells
- Adding and moving cells

Working with python

- System aliases
- Magics
- Tab-completion and exploring code
- Exception Formatting
- Rich, interactive outputs

Integration with Drive

- Commenting on a cell

 SECTION

Cells

A notebook is a list of cells. Cells contain either explanatory text or executable code and its output. Click a cell to select it.


Code cells

Below is a **code cell**. Once the toolbar button indicates CONNECTED, click in the cell to select it and execute the contents in the following ways:

- Click the **Play icon** in the left gutter of the cell;
- Type **Cmd/Ctrl+Enter** to run the cell in place;
- Type **Shift+Enter** to run the cell and move focus to the next cell (adding one if none exists); or
- Type **Alt+Enter** to run the cell and insert a new code cell immediately below it.

There are additional options for running some or all cells in the **Runtime** menu.

```
[ ] a = 10
a
```

 10

Text cells

This is a **text cell**. You can **double-click** to edit this cell. Text cells use markdown syntax. To learn more, see our [markdown guide](#).

You can also add math to text cells using [LaTeX](#) to be rendered by [MathJax](#). Just place the statement within a pair of \$ signs. For example $\sqrt{3x-1} + (1+x)^2$ becomes $\sqrt{3x-1} + (1+x)^2$.

Adding and moving cells

You can add new cells by using the **+ CODE** and **+ TEXT** buttons that show when you hover between cells. These buttons are also in the toolbar above the notebook

```
[ ] # Uploading files from your local file system
```

```
from google.colab import files
uploaded = files.upload()
for fn in uploaded.keys():
    print('User uploaded file "{name}" with length {length} bytes'.format(
        name=fn, length=len(uploaded[fn])))
```



Escolher arquivos datatran2018.csv

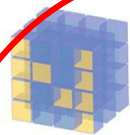
- **datatran2018.csv**(application/vnd.ms-excel) - 33096569 bytes, last modified: 20/09/2018 - 100% done
- Saving datatran2018.csv to datatran2018.csv
User uploaded file "datatran2018.csv" with length 33096569 bytes

Código disponível em:

<https://colab.research.google.com/notebooks/io.ipynb>



Bibliotecas 2



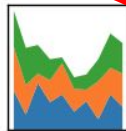
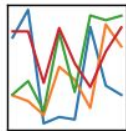
NumPy



SciPy

pandas

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



seaborn

matplotlib



NumPy



- Cálculos numéricos
- Multidimensional arrays e matrizes
- Não precisa de loops para criar arrays

```
import numpy as np
a1 = np.array([(1, 2), (5, 7)])
a2 = np.full((2,2),5)
a3 = np.random.random((3,3))
```

```
print(a1)
```

```
[[1 2]
 [5 7]]
```

```
print(a2)
```

```
[[5 5]
 [5 5]]
```

```
print(a3)
```

```
[[0.59476188 0.92656242 0.61528635]
 [0.53514719 0.84787068 0.50845682]
 [0.19158773 0.76393059 0.78621118]]
```



Multiplicação de arrays

```
a1 * a2
```

```
array([[ 5, 10],  
       [25, 35]])
```



```
print(a1)
```

```
[[1 2]  
 [5 7]]
```

```
print(a2)
```

```
[[5 5]  
 [5 5]]
```

Multiplicação de matrizes

```
a1.dot(a2)
```

```
array([[15, 15],  
       [60, 60]])
```


Slicing



```
a4 = np.array([(1, 2, 3), (4, 5, 7), (10, 20, 30) ])
```

```
print(a4)
```

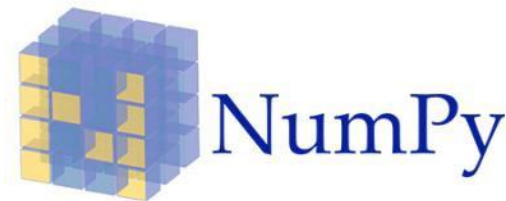
```
[[ 1  2  3]
 [ 4  5  7]
 [10 20 30]]
```

```
slicing = a4[1:3, 0:2]
```

```
print(slicing)
```

```
[[ 4  5]
 [10 20]]
```

Leitura de arquivo CSV



```
import numpy as np
prf_np = np.genfromtxt("datatran2018.csv", encoding = "Latin-1", delimiter=";")
```

prf_np

```
array([[ nan,    nan,    nan, ...,    nan,    nan,    nan],
       [99973.,    nan,    nan, ...,    nan,    nan,    nan],
       [99976.,    nan,    nan, ...,    nan,    nan,    nan],
       ...,
       [160497.,    nan,    nan, ...,    nan,    nan,    nan],
       [160598.,    nan,    nan, ...,    nan,    nan,    nan],
       [160605.,    nan,    nan, ...,    nan,    nan,    nan]])
```

- Resolução de problemas em computação científica;
- Módulos para aplicações: interpolação, integração, estatística, etc.

Álgebra Linear



● Cálculo de determinante

```
>>> import numpy as np
>>> from scipy import linalg
>>> A = np.array([[1,2],[3,4]])
>>> A
array([[1, 2],
       [3, 4]])
>>> linalg.det(A)
-2.0
```

Exemplo disponível em: <https://docs.scipy.org/doc/scipy-1.1.0/reference/tutorial/linalg.html>

Álgebra Linear



- Cálculo de matriz inversa

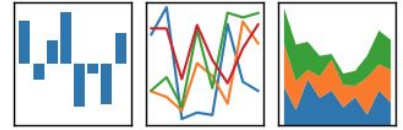
```
>>> import numpy as np
>>> from scipy import linalg
>>> A = np.array([[1,3,5],[2,5,1],[2,3,8]])
>>> A
array([[1, 3, 5],
       [2, 5, 1],
       [2, 3, 8]])
>>> linalg.inv(A)
array([[ -1.48,  0.36,  0.88],
       [ 0.56,  0.08, -0.36],
       [ 0.16, -0.12,  0.04]])
```

Exemplo disponível em: <https://docs.scipy.org/doc/scipy-1.1.0/reference/tutorial/linalg.html>

Pandas

pandas

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

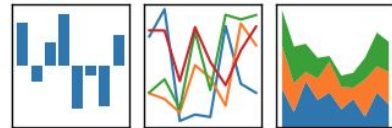


- Análise de dados e ciência de dados
- Manipulação e leitura de dados.

Leitura de arquivo CSV

pandas

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

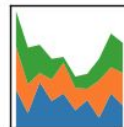
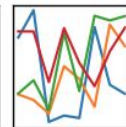


```
import pandas as pd
prf_acidentes = pd.read_csv("datatran2018.csv", encoding = "Latin-1", sep=";")
```

Visualização do arquivo CSV

pandas

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



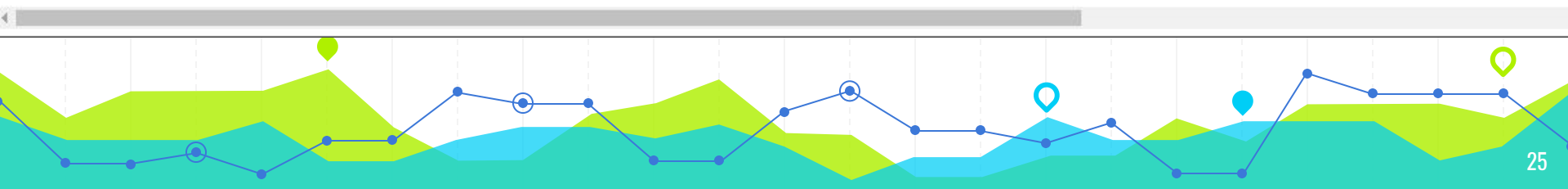
```
prf_acidentes.head()
```

	id	data_inversa	dia_semana	horario	uf	br	km	municipio	causa_acidente	tipo_acidente	...	feridos_graves	ilesos	ig
0	99973.0	2018-01-01	segunda-feira	00:20:00	RJ	116.0	303,5	RESENDE ...	Condutor Dormindo ...	Saída de leito carroçável	4	1	
1	99976.0	2018-01-01	segunda-feira	00:40:00	SC	282.0	0,4	FLORIANOPOLIS ...	Não guardar distância de segurança ...	Colisão traseira	2	1	
2	99977.0	2018-01-01	segunda-feira	00:30:00	RJ	493.0	1	ITABORAI ...	Ultrapassagem Indevida ...	Colisão frontal	0	3	
3	99981.0	2018-01-01	segunda-feira	01:15:00	RS	386.0	134	SARANDI ...	Ingestão de Álcool ...	Colisão transversal	0	2	
4	99982.0	2018-01-01	segunda-feira	00:20:00	RS	293.0	151,7	CANDIOTA ...	Falta de Atenção à Condução ...	Saída de leito carroçável	1	0	

5 rows x 30 columns


```
prf_datafr = prf_acidentes.loc[:, ['data_inversa', 'dia_semana', 'horario', 'uf', 'br',  
                                  'municipio', 'causa_acidente', 'tipo_acidente', 'fase_dia',  
                                  'condicao_metereologica', 'tipo_pista', 'tracado_via', 'veiculos']]  
prf_datafr.head()
```

	data_inversa	dia_semana	horario	uf	br	municipio	causa_acidente	tipo_acidente	fase_dia
0	2018-01-01	segunda-feira	00:20:00	RJ	116.0	RESENDE ...	Condutor Dormindo ...	Saída de leito carroçável ...	Plena Noite
1	2018-01-01	segunda-feira	00:40:00	SC	282.0	FLORIANOPOLIS ...	Não guardar distância de segurança ...	Colisão traseira ...	Plena Noite
2	2018-01-01	segunda-feira	00:30:00	RJ	493.0	ITABORAI ...	Ultrapassagem Indevida ...	Colisão frontal ...	Plena Noite
3	2018-01-01	segunda-feira	01:15:00	RS	386.0	SARANDI ...	Ingestão de Álcool ...	Colisão transversal ...	Pleno dia
4	2018-01-01	segunda-feira	00:20:00	RS	293.0	CANDIOTA ...	Falta de Atenção à Condução ...	Saída de leito carroçável ...	Plena Noite

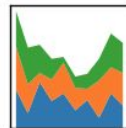
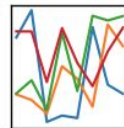
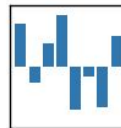


```
prf_datafr.info(.
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 46497 entries, 0 to 46496  
Data columns (total 13 columns):  
data_inversa          46497 non-null object  
dia_semana            46497 non-null object  
horario               46497 non-null object  
uf                    46497 non-null object  
br                    46421 non-null float64  
municipio             46497 non-null object  
causa_acidente        46497 non-null object  
tipo_acidente         46497 non-null object  
fase_dia              46497 non-null object  
condicao_metereologica 46497 non-null object  
tipo_pista            46497 non-null object  
tracado_via           46497 non-null object  
veiculos              46497 non-null int64  
dtypes: float64(1), int64(1), object(11)  
memory usage: 4.6+ MB
```

pandas

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

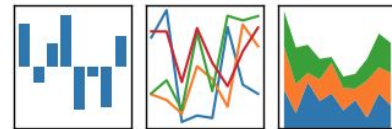


```
#Soma da quantidade de dados faltantes  
prf_datafr.isnull().sum()
```

data_inversa	0
dia_semana	0
horario	0
uf	0
br	76
municipio	0
causa_acidente	0
tipo_acidente	0
fase_dia	0
condicao_metereologica	0
tipo_pista	0
tracado_via	0
veiculos	0
dtype:	int64

pandas

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



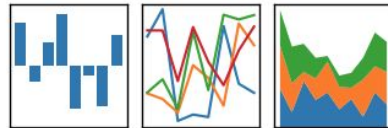
```
#Remoção dos dados faltantes  
prf_datafr.dropna(inplace=True)
```

```
#Confirmação que os dados faltantes foram removidos  
prf_datafr.isnull().sum()
```

```
data_inversa      0  
dia_semana        0  
horario           0  
uf                0  
br               0  
municipio         0  
causa_acidente    0  
tipo_acidente     0  
fase_dia          0  
condicao_meteorologica 0  
tipo_pista        0  
tracado_via       0  
veiculos          0  
dtype: int64
```

pandas

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



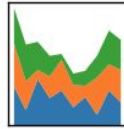
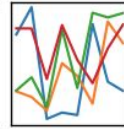
```
prf_datafr["dia_semana"].value_counts()
```

sábado	7407
domingo	7335
sexta-feira	7273
segunda-feira	6375
quinta-feira	6133
quarta-feira	5965
terça-feira	5933

Name: dia_semana, dtype: int64

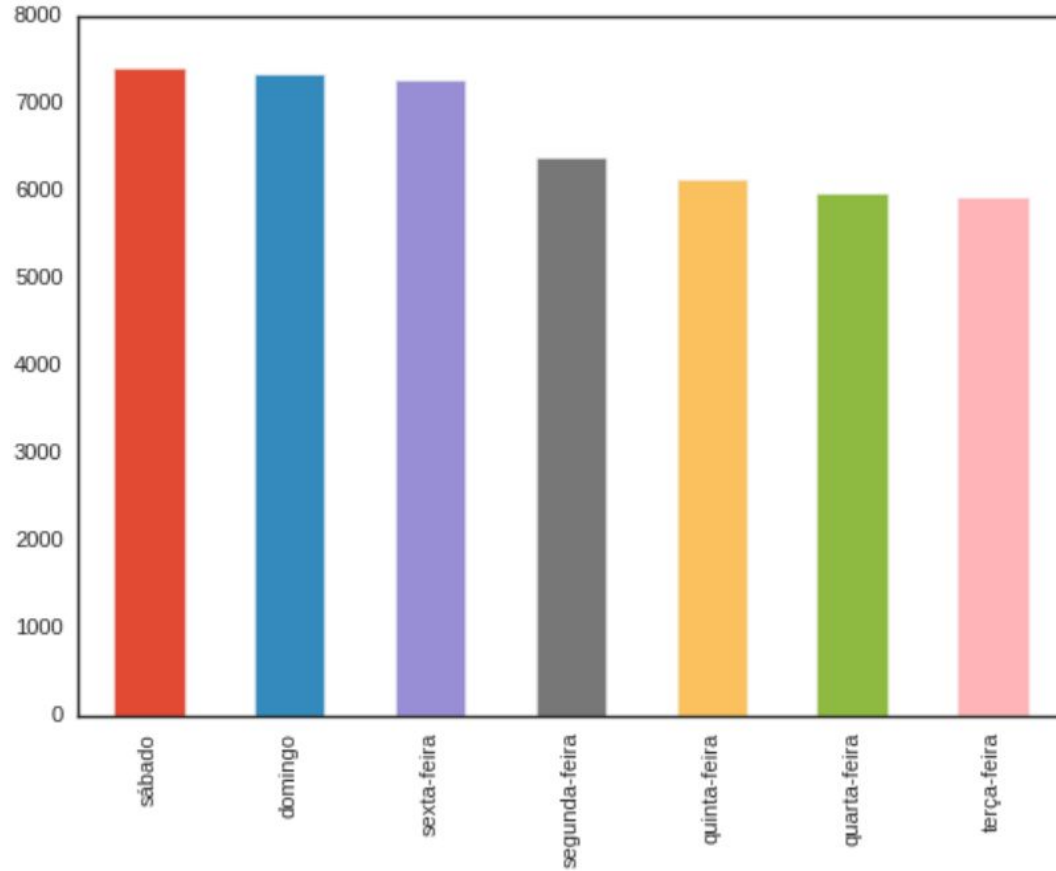
pandas

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

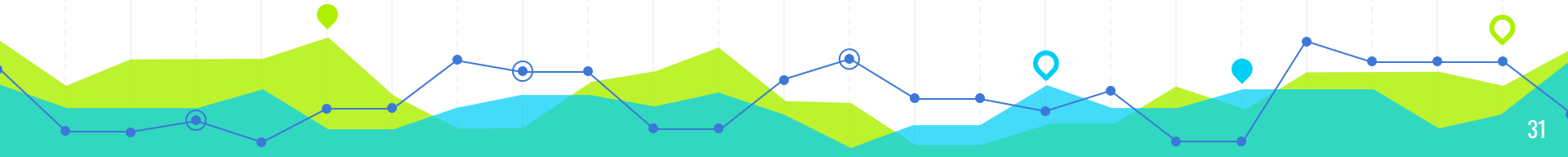


```
prf_datafr["dia_semana"].value_counts().plot.bar()
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fe9c9746080>



● Customização e visualização de gráficos

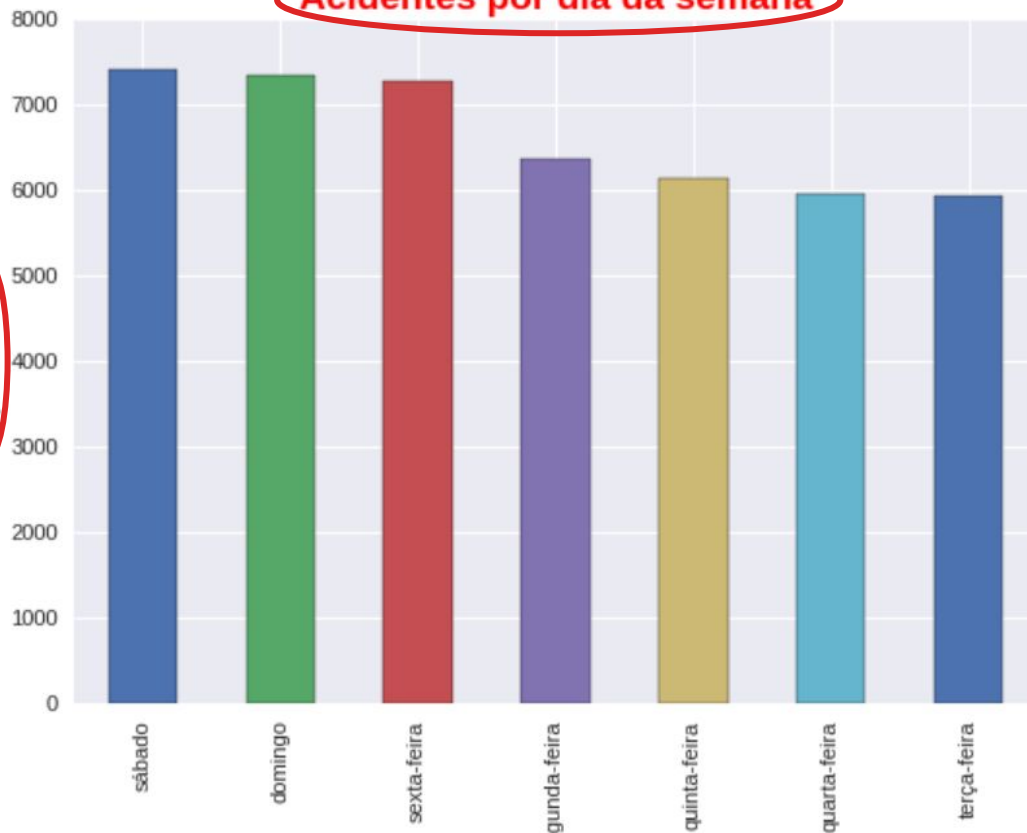


```
import matplotlib.pyplot as plt

plt.style.use('seaborn')
prf_datafr["dia_semana"].value_counts().plot.bar()
plt.title('Acidentes por dia da semana',
          fontsize='xx-large',
          weight = 'heavy', color='r')
plt.xlabel('Dias')
plt.ylabel('Quantidade')
```

Quantidade

Acidentes por dia da semana



Dias

Seaborn

seaborn

- Visualização de gráficos mais complexos
- Usuários de python de nível intermediário



```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()

# Load the example flights dataset and convert to long-form
flights_long = sns.load_dataset("flights")
flights = flights_long.pivot("month", "year", "passengers")

# Draw a heatmap with the numeric values in each cell
f, ax = plt.subplots(figsize=(9, 6))
sns.heatmap(flights, annot=True, fmt="d", linewidths=.5, ax=ax)
```



Exemplo disponível em:

https://seaborn.pydata.org/examples/heatmap_annotation.html



Obrigada!

Perguntas?



github.com/VanessaSayuri



sayurivanessa@gmail.com

CREDITS

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival