VISUALIZAÇÃO COMPUTACIONAL

Bibliotecas de visualização de dados do Python

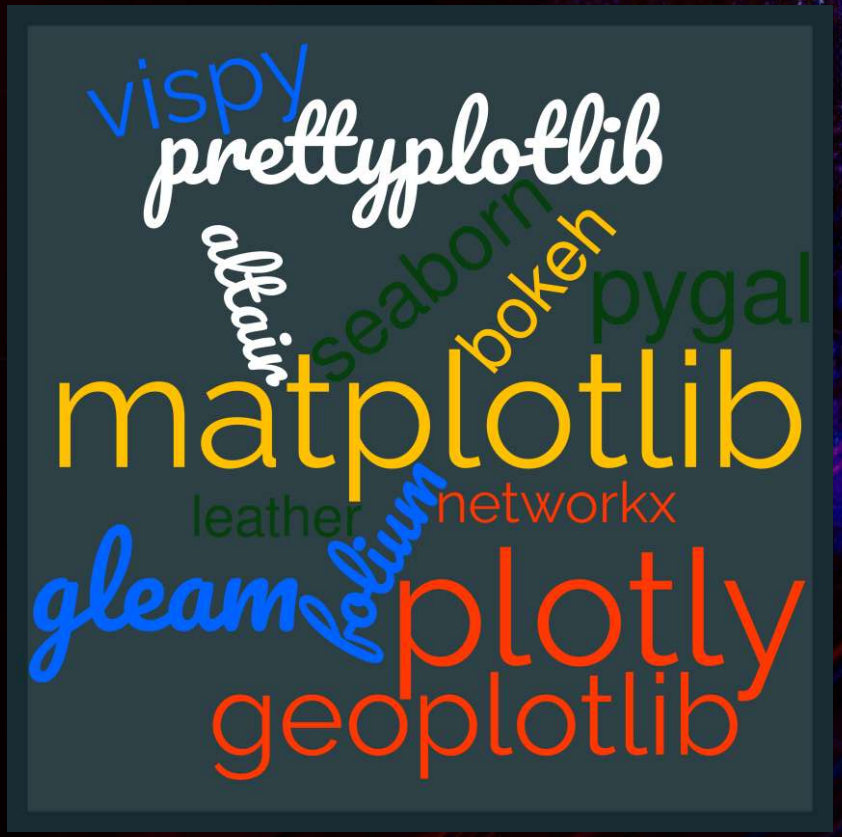
TÓPICOS

- Principais bibliotecas para visualização
- Exemplos

VISUALIZAÇÃO DE DADOS COM PYTHON

VÁRIAS BIBLIOTECAS:

- Forma de uso
- Diferentes técnicas de visualização

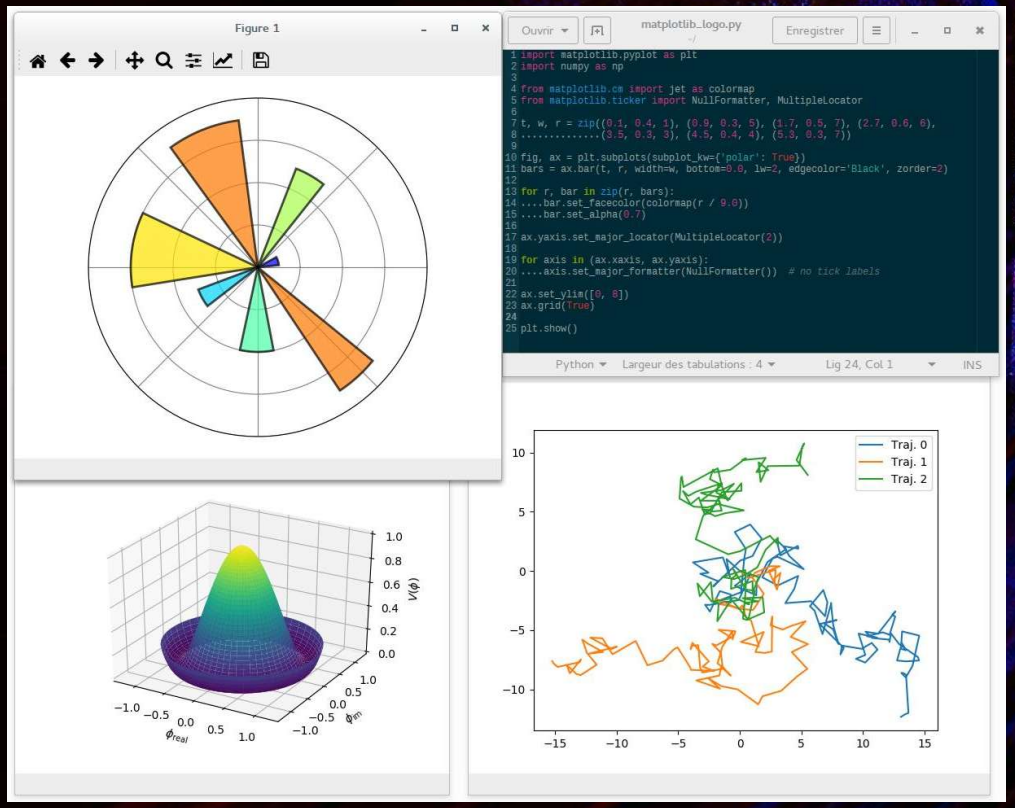


BIBLIOTECAS DE VISUALIZAÇÃO DE DADOS

Biblioteca	URL
Matplotlib	matplotlib.org
Seaborn	seaborn.pydata.org
Plotly	plotly.com
Folium	python-visualization.github.io/folium

MATPLOTLIB

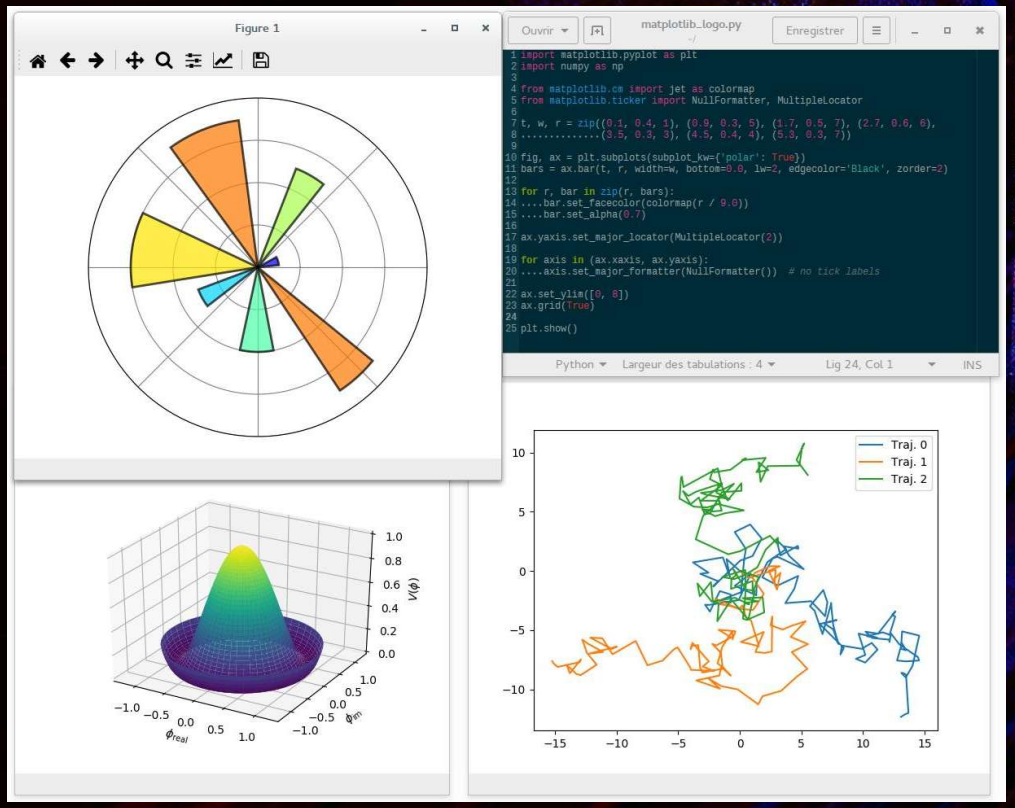
- Amplamente usado
- Relativamente fácil de usar
- Documentação vasta
- Controle fino:
  - ➢ Fontes, cores, partes
- Usado por:
  - ➢ Pandas, seaborn



Fonte: Adrien F. Vincent, Wikimedia Commons

MATPLOTLIB - DESVANTAGENS

- Gráficos mais básicos
- Exige muitos códigos
- Visualizações estáticas



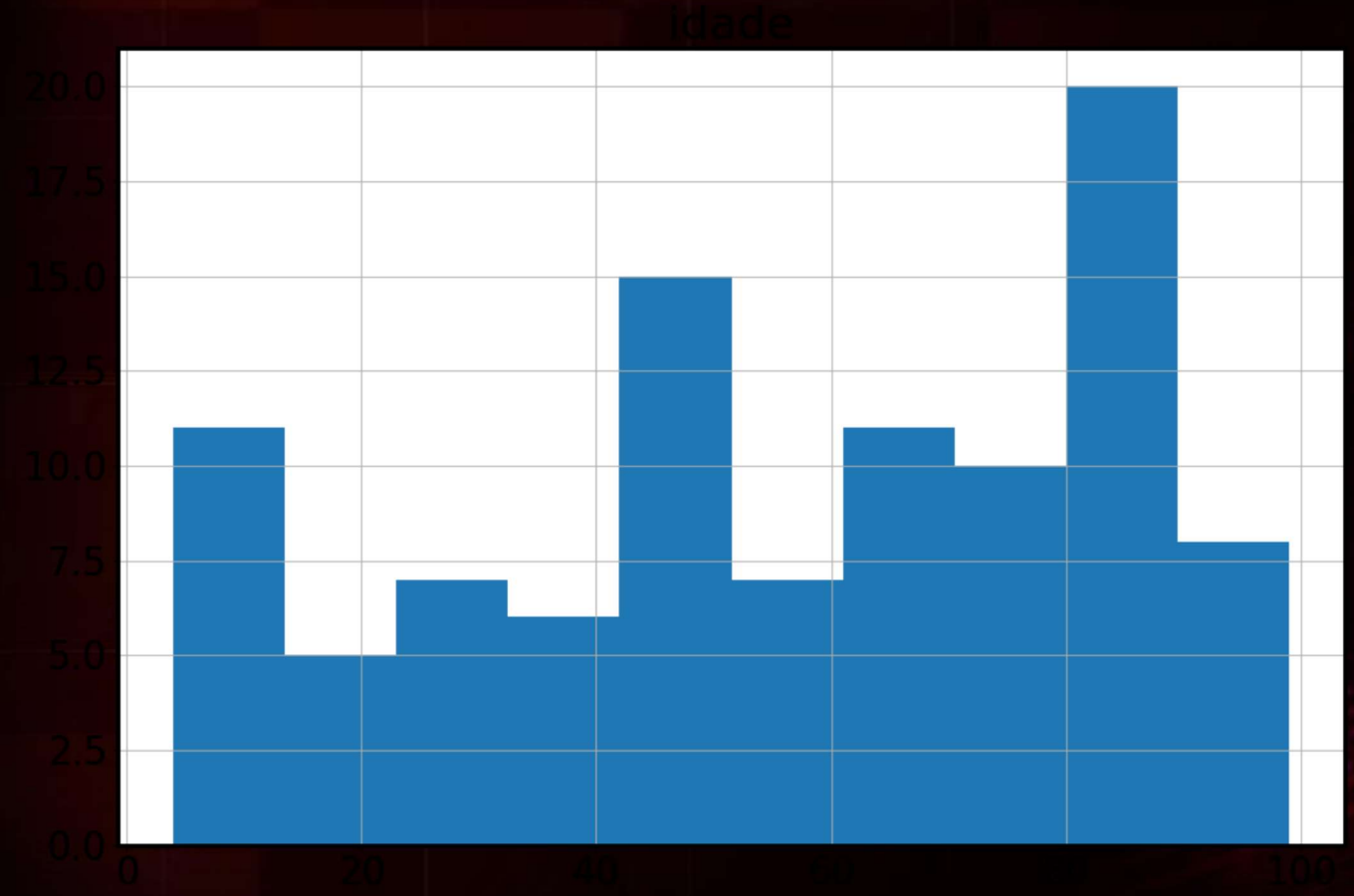
Fonte: Adrien F. Vincent, Wikimedia Commons

MATPLOTLIB - HISTOGRAMA

```
import pandas as pd
Import matplotlib.pyplot as plt

plt.rcParams['font.size'] = 26

dados = pd.read_csv('clientes.csv')
```

Nome	Idade
Maria	35
João	22
Ana	19
...	...
Carlos	48
Pedro	34
Júlia	28

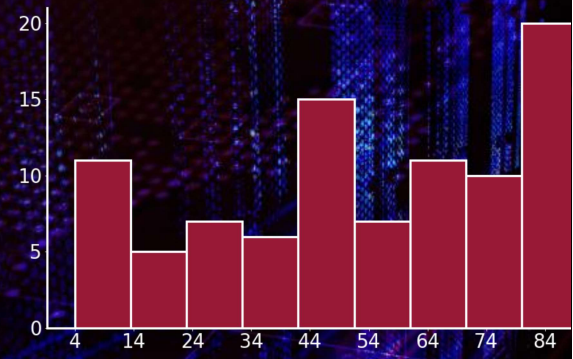
MATPLOTLIB - HISTOGRAMA




```
#pandas

dados.hist(bins=10)
```

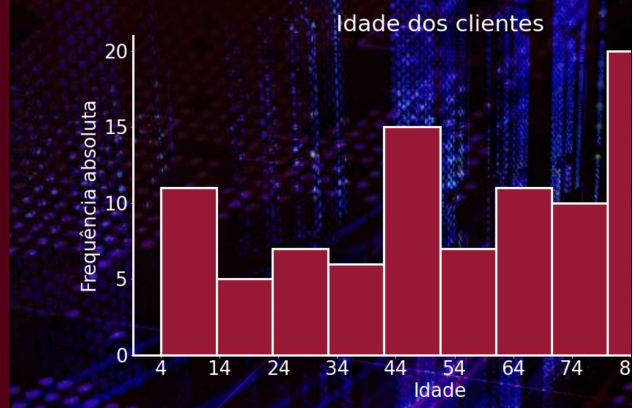
MATPLOTLIB - HISTOGRAMA

```
figura, eixos = plt.subplots(figsize=(15,8))

eixos.hist(dados['idade'], bins=10, color='#991A37',
                linewidth=3, edgecolor='white')

eixos.set(xticks=np.arange(min(dados['idade']),
                                    max(dados['idade'])+1,10))

eixos.spines['top'].set_visible(False)
eixos.spines['right'].set_visible(False)
eixos.spines['bottom'].set_color('white')
eixos.spines['left'].set_color('white')
eixos.tick_params(axis='x', colors='white')
eixos.tick_params(axis='y', colors='white')
```



MATPLOTLIB - HISTOGRAMA

```
plt.title('Idade dos clientes').set_color('white')
plt.xlabel('Idade').set_color('white')
plt.ylabel('Frequência absoluta').set_color('white')

plt.savefig('histograma.png', transparent=True)
plt.show()
```



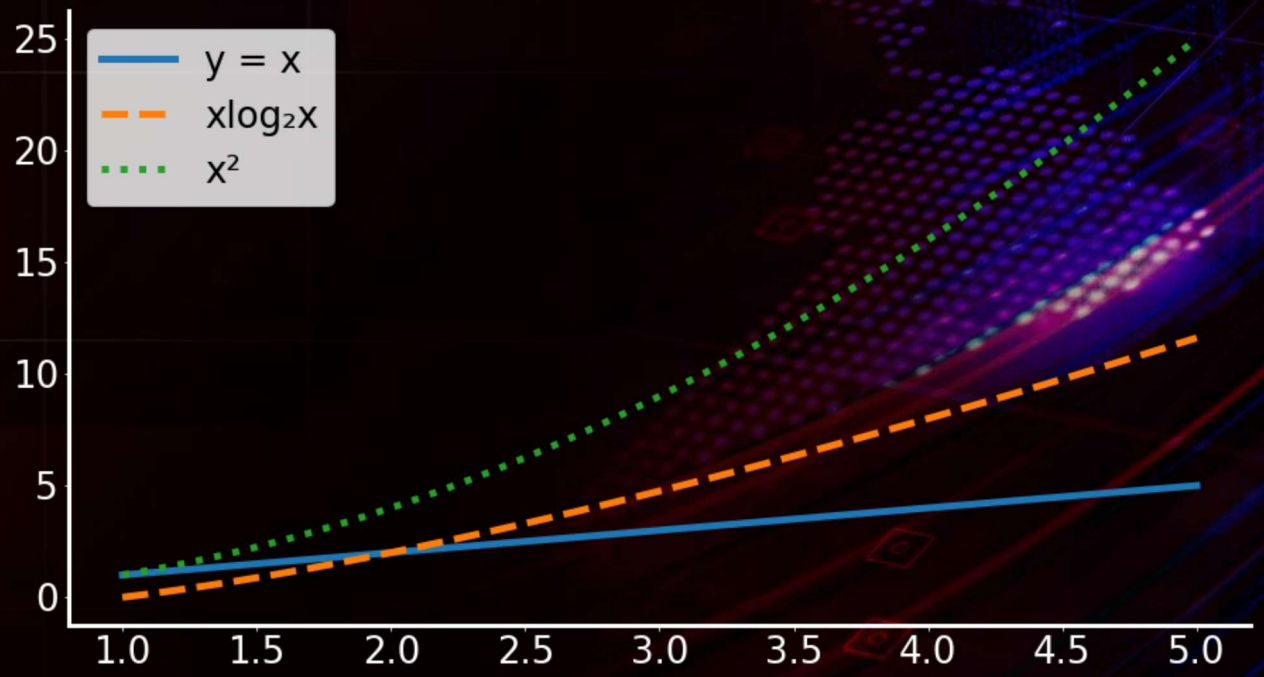
MATPLOTLIB - HISTOGRAMA

MATPLOTLIB - LINHAS

x	y = x	$x\log_2(x)$	x^2
1	1	0.0	1.0
1.1	1.1	0.15	1.21
1.2	1.2	0.32	1.44
...	...	...	...
4.8	4.8	10.86	23.04
4.9	4.9	11.23	24.01
5.0	5.0	11.61	25

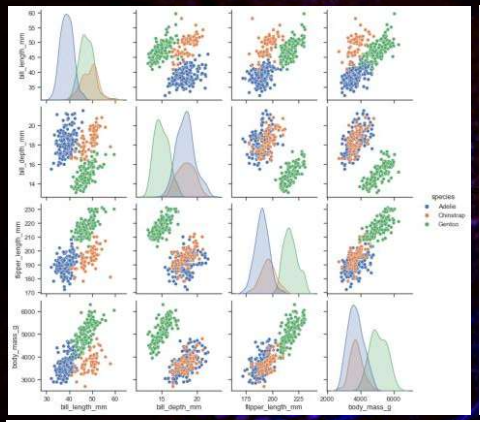
MATPLOTLIB - LINHAS

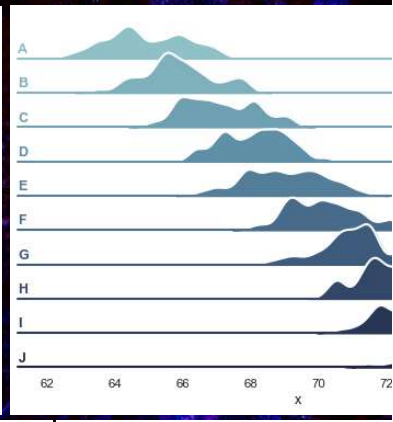
```
plt.plot(x, x, linewidth=5, linestyle='solid', label='y = x')
plt.plot(x, xlog2x, linewidth=5, linestyle='dashed', label='xlog\u2082x')
plt.plot(x, x2, linewidth=5, linestyle='dotted', label='x\u00b2')

plt.legend()
plt.show()
```

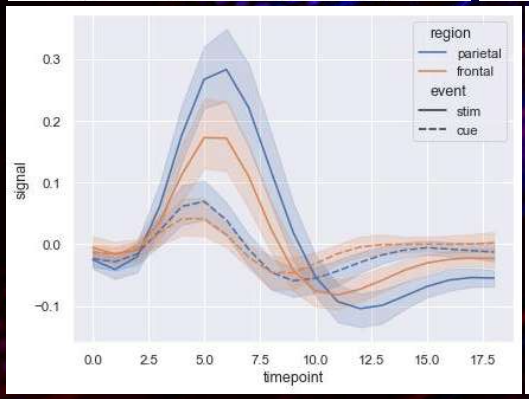


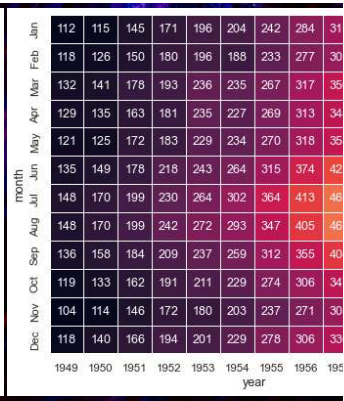
SEABORN

- Baseada no Matplotlib
- Menos códigos
- Compatível com estruturas de Pandas e NumPy
- Vários temas prontos
- Variedade de visualizações
  - Visual mais elaborado



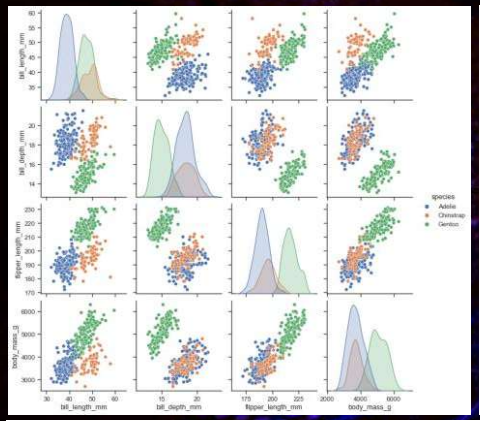


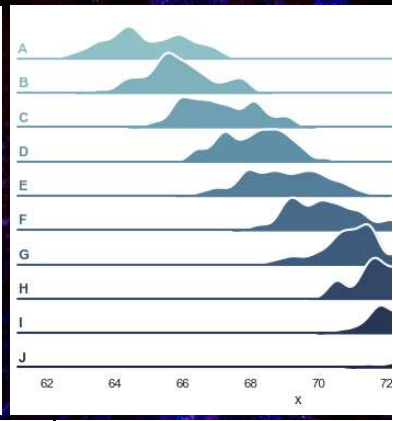


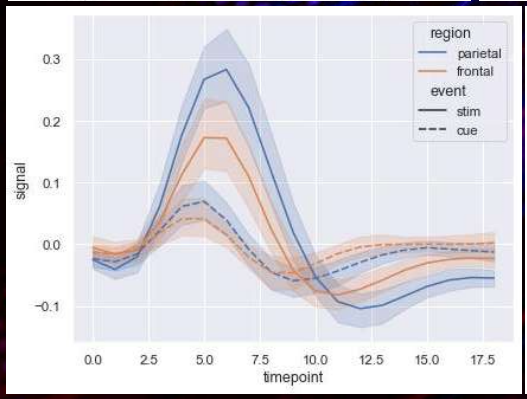


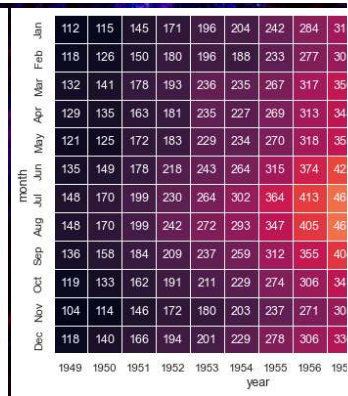
SEABORN - DESVANTAGENS

- Customização limitada
- Poucas visualizações interativas

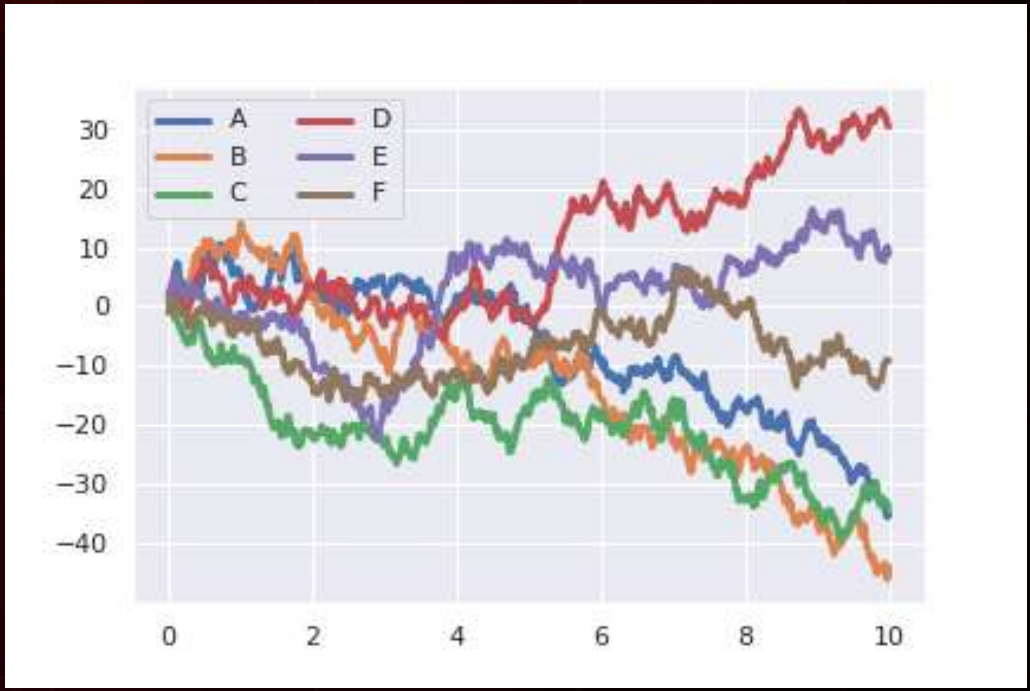




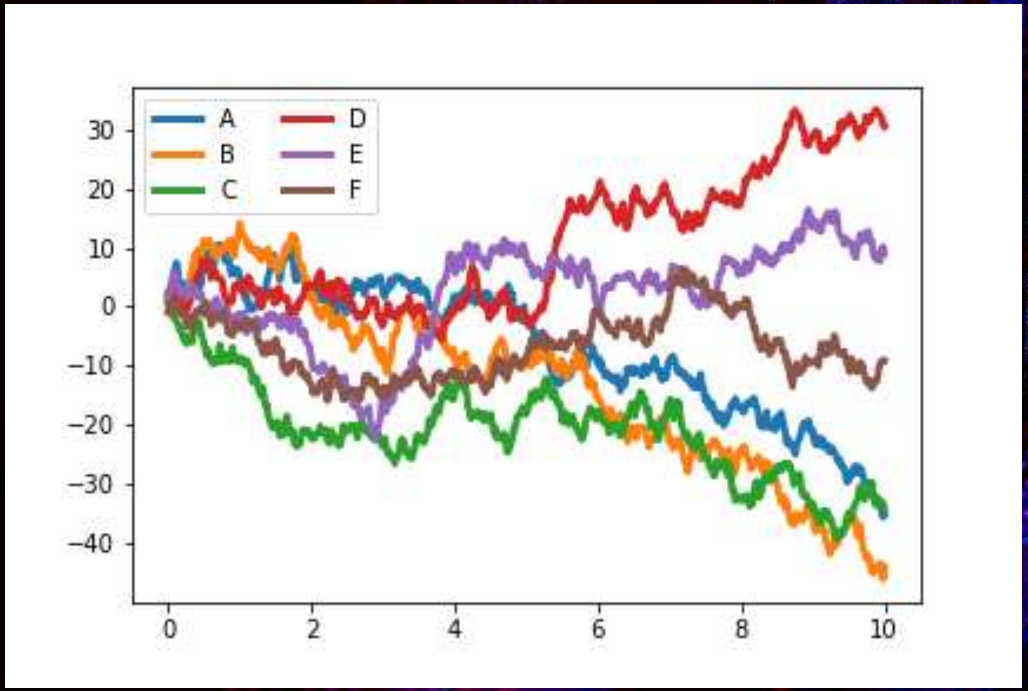




SEABORN X MATPLOTLIB



seaborn



matplotlib

SEABORN - CONFIGURAÇÕES

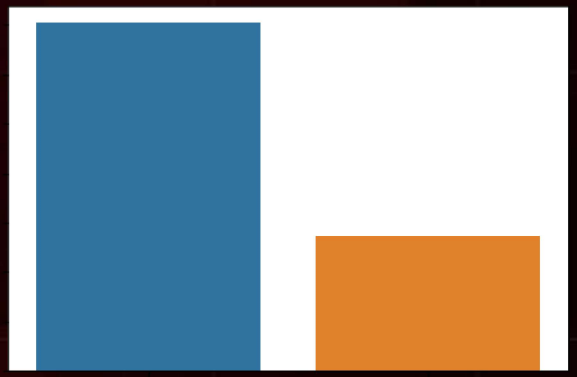
```
sb.set_theme(style='white', palette='dark')
```

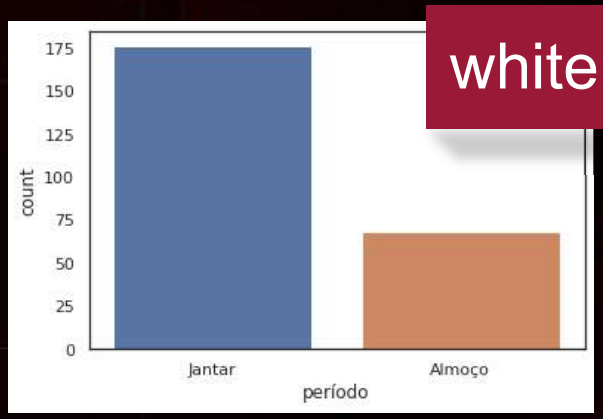
```
sb.countplot(data='dados', style='whitegrid',
palette='dark', context='notebook')
```

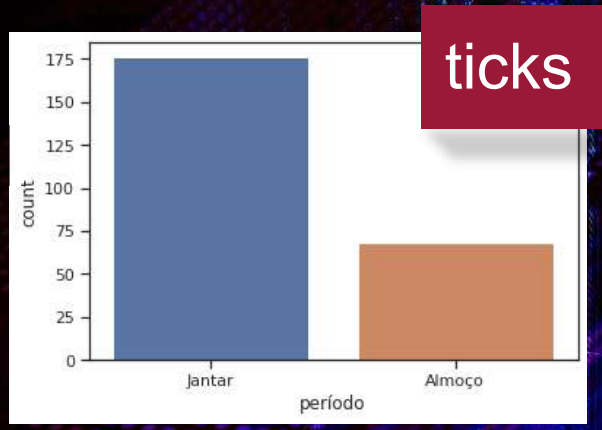
```
sb.set_palette('dark')
```

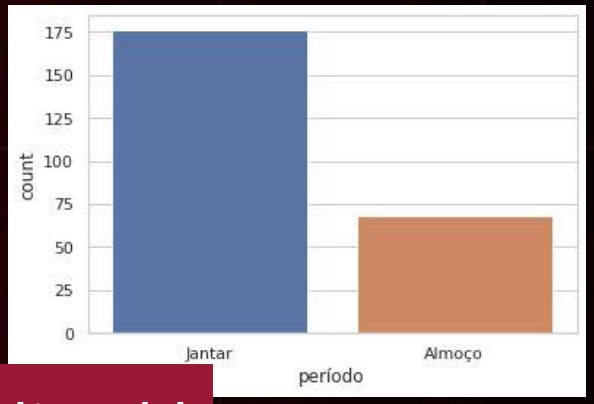
SEABORN - TEMAS

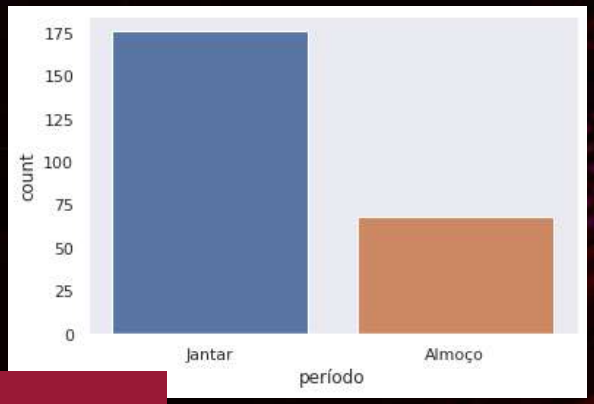
sb.set_theme(style='white')





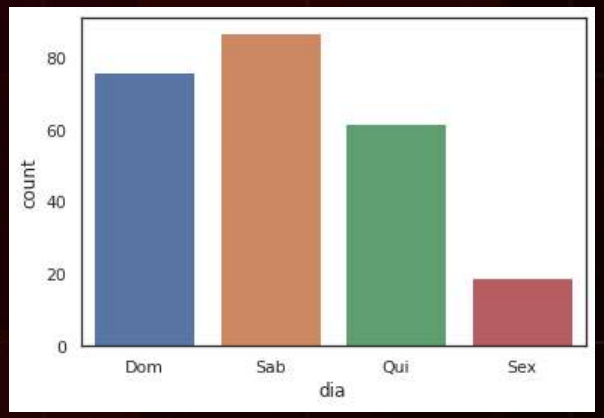


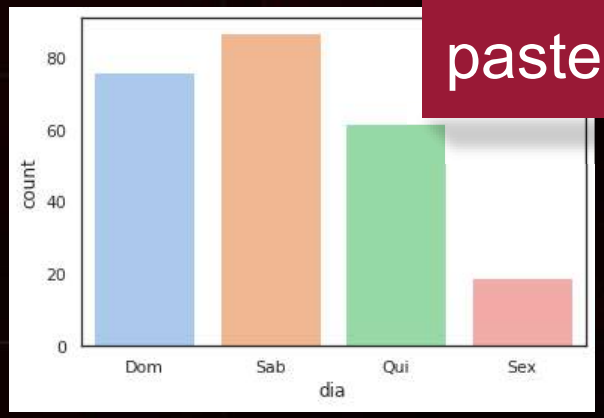




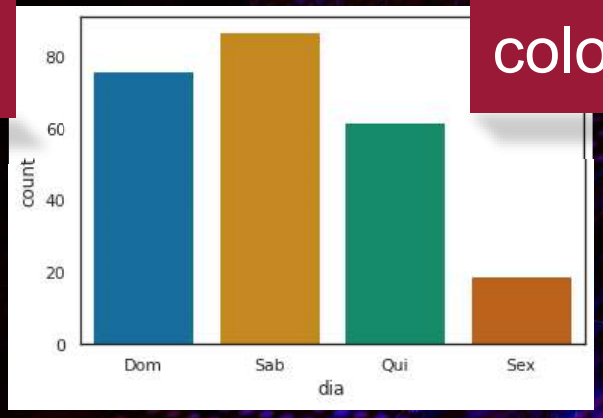
SEABORN - CORES

sb.set_theme(palette='pastel')

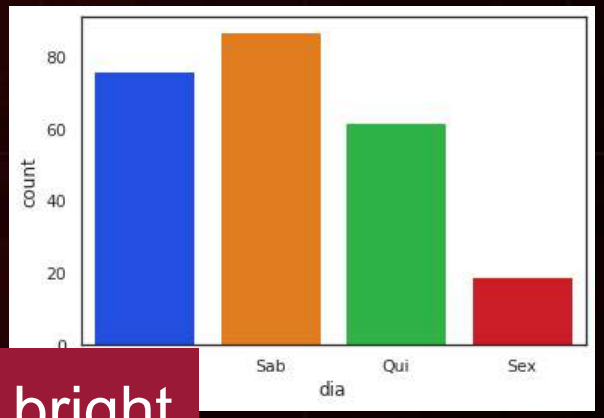




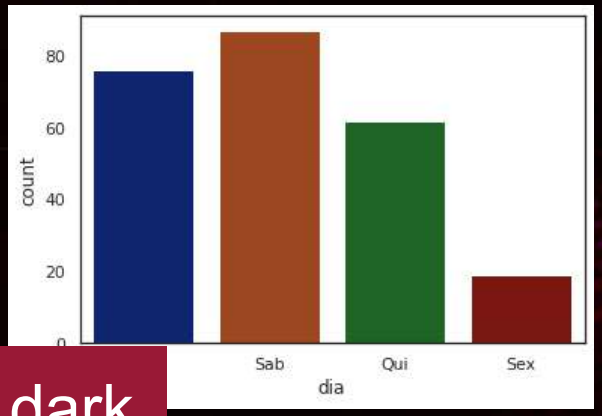
pastel



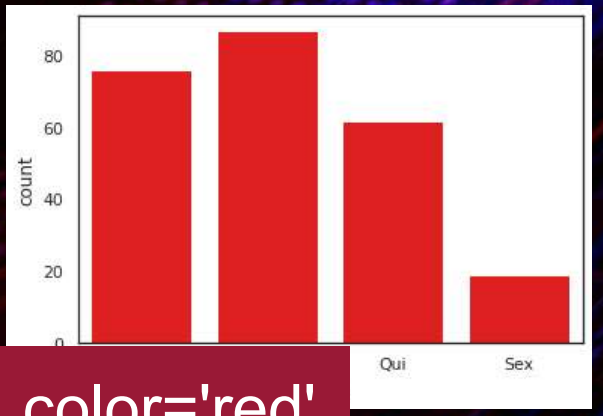
colorblind



bright



dark

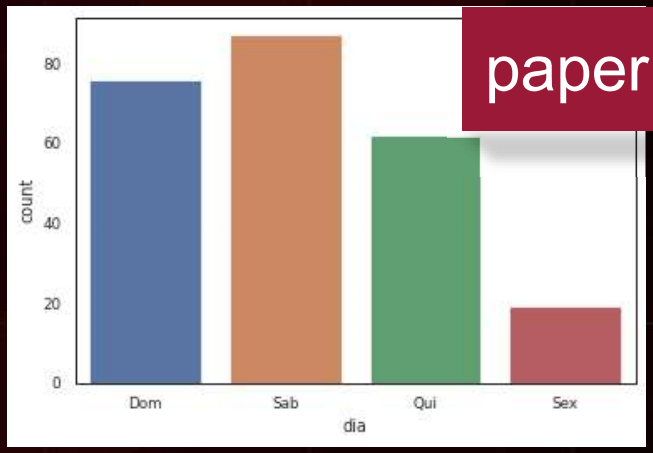


color='red'

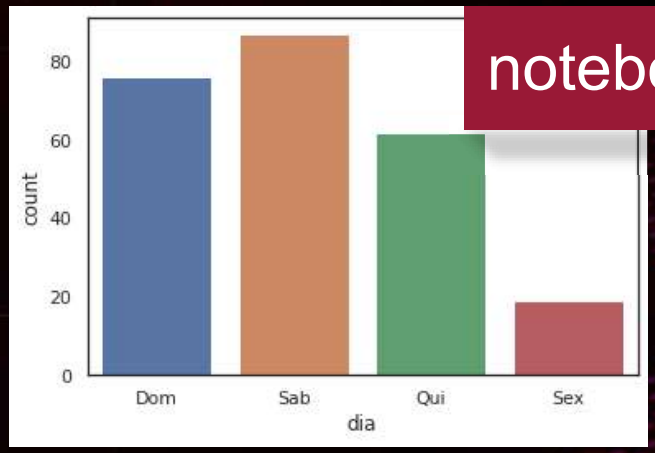
SEABORN - CONTEXTO

sb.set_theme(context='paper')

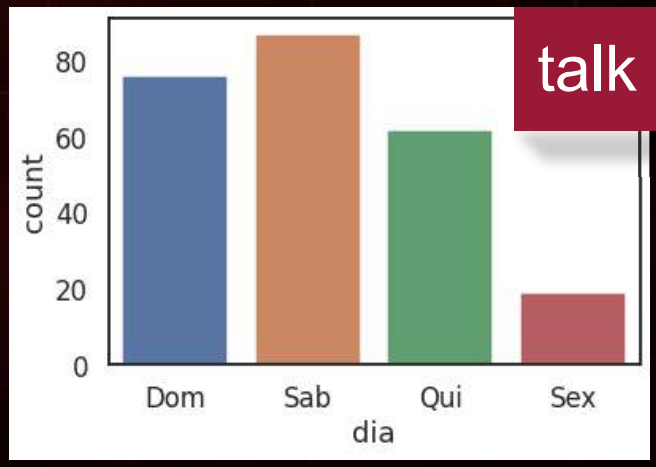
paper



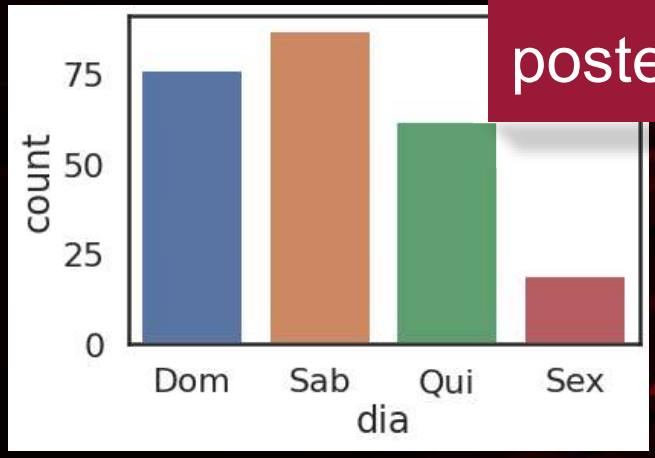
notebook



talk

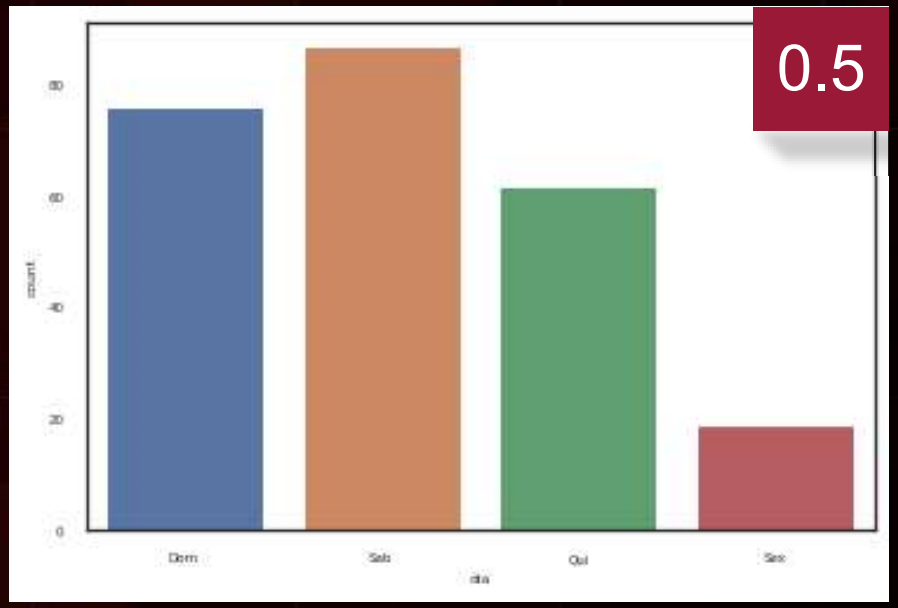


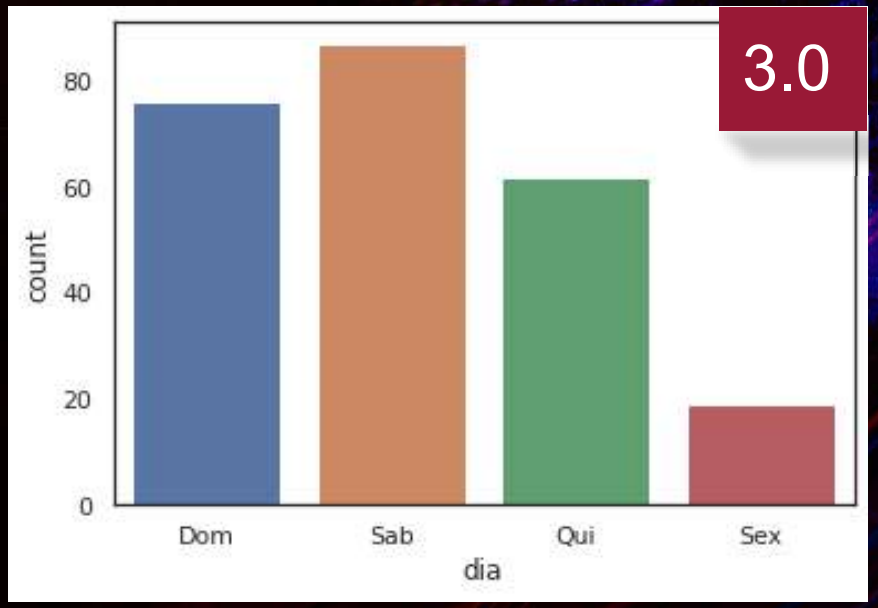
poster



SEABORN - ESCALA DA FONTE

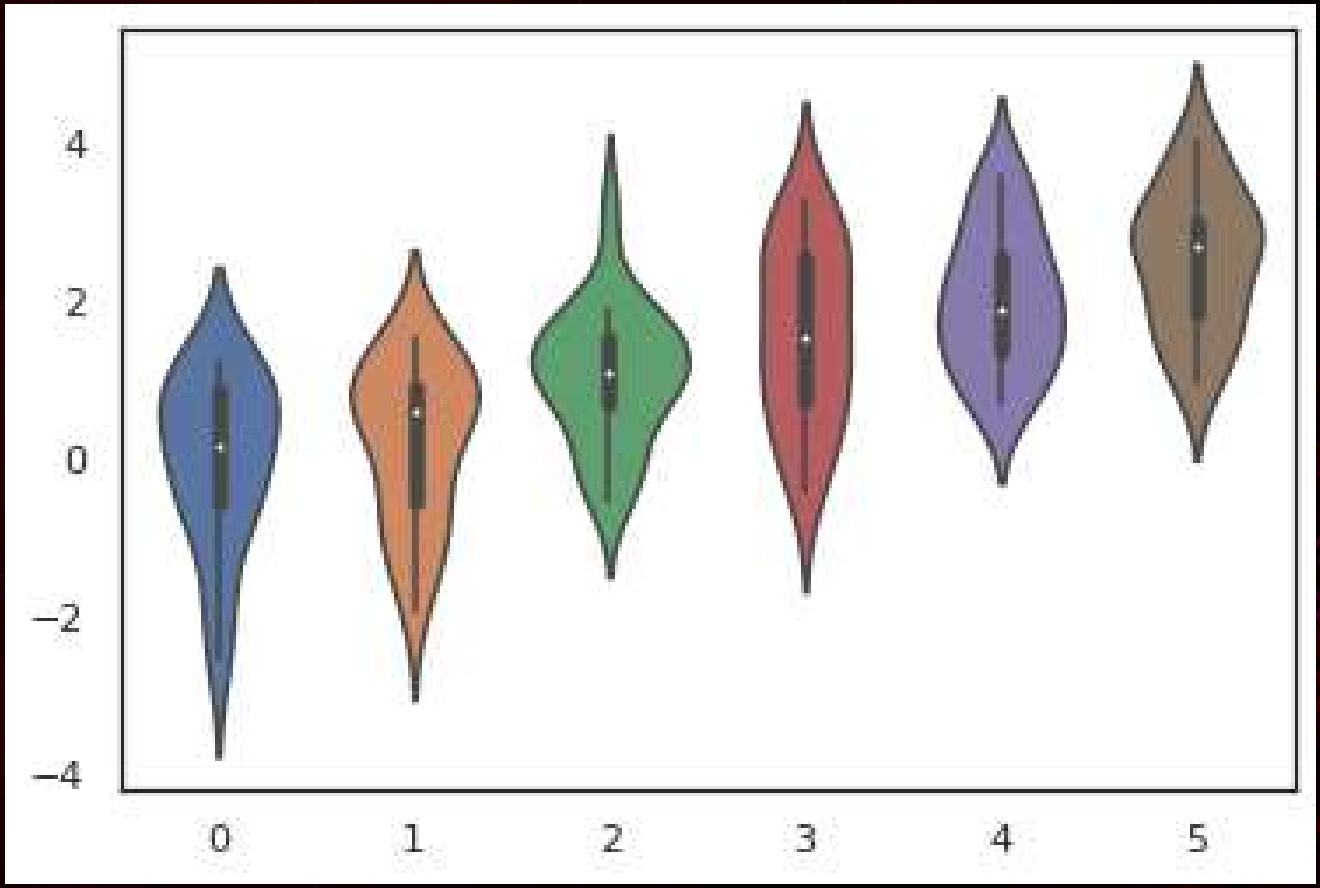
sb.set_theme(font_scale=1.0)





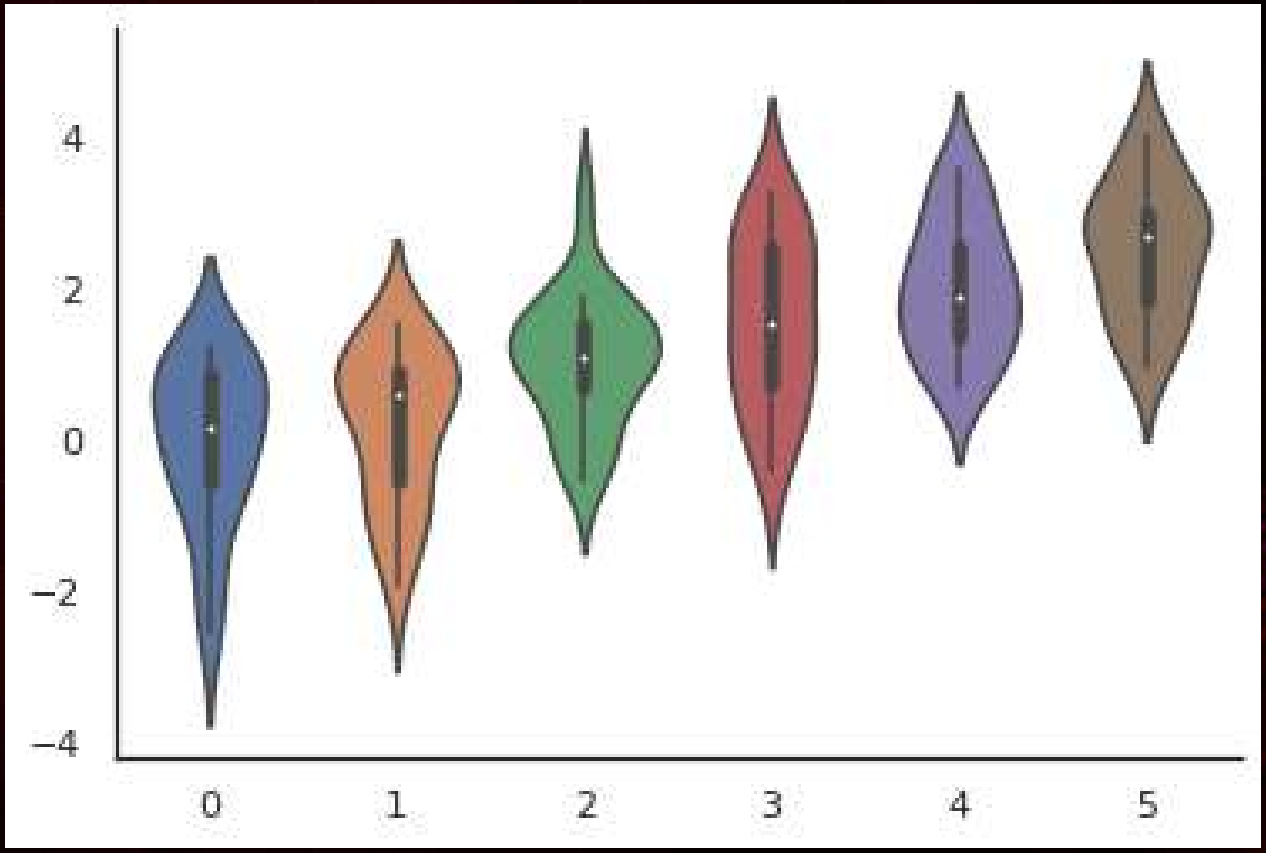
SEABORN - VIOLINPLOT

sb.violinplot(data=dados)



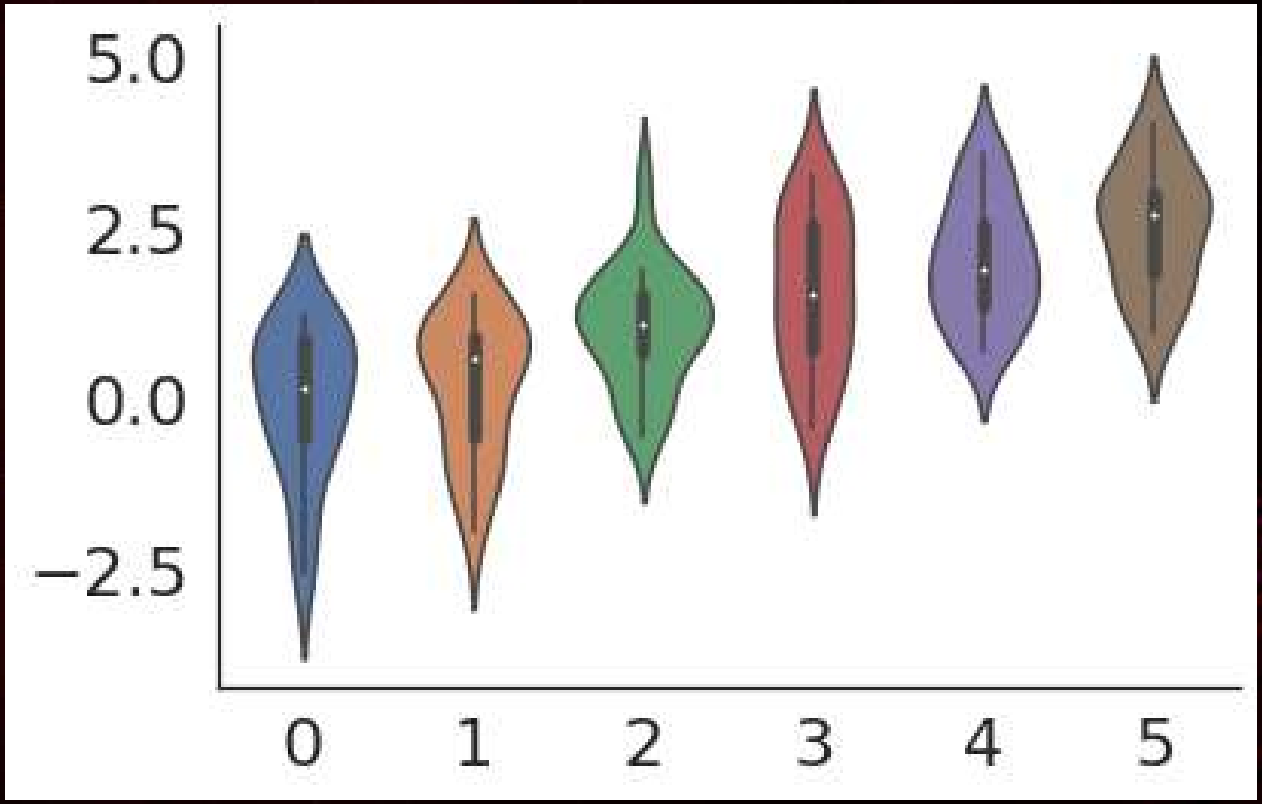
SEABORN - VIOLINPLOT

#removendo as linhas de cima e da direita
sb.despine()



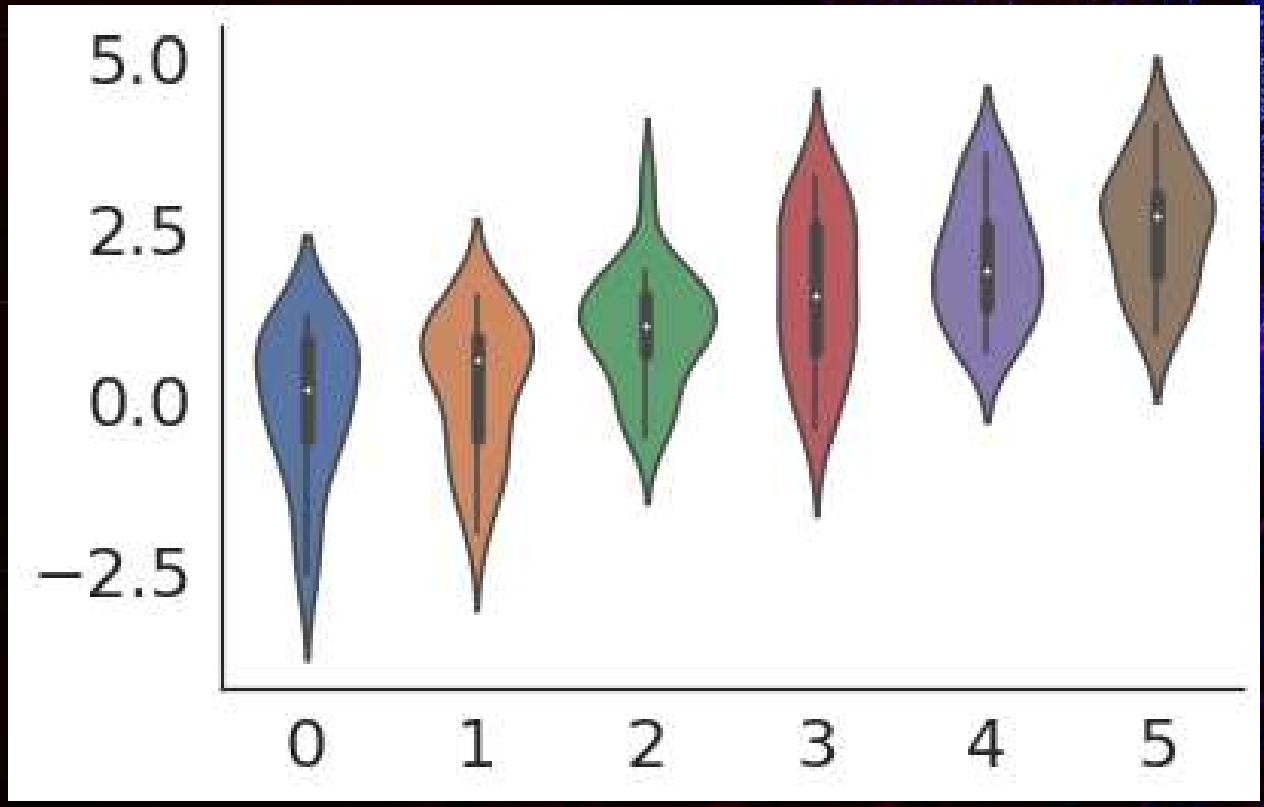
SEABORN - VIOLINPLOT

```
#aumentando a fonte dos eixos
sb.set_theme(font_scale=2.0)
```



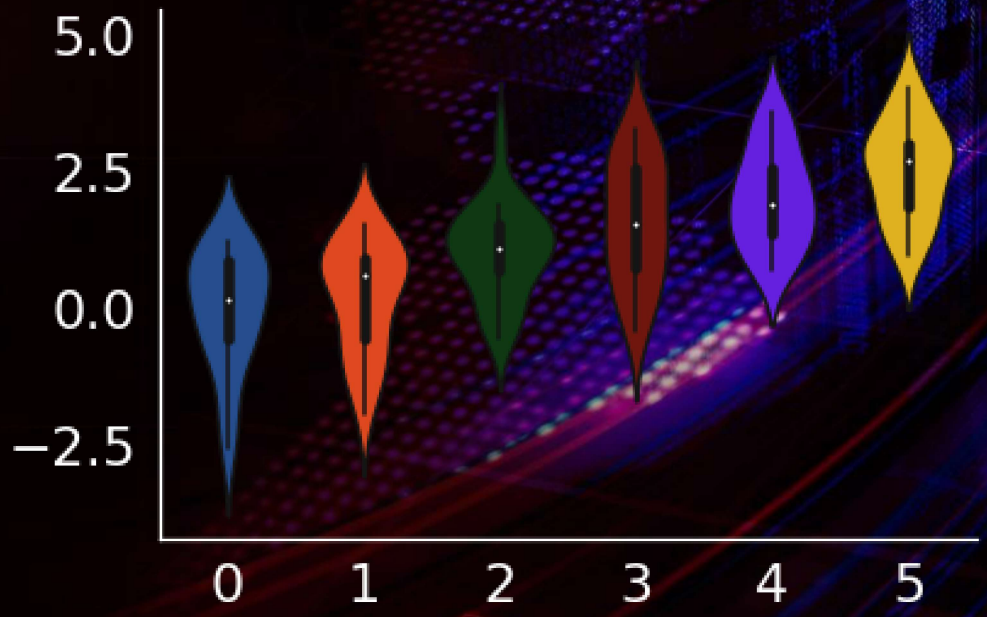
SEABORN - VIOLINPLOT

```
#mudando as cores
cores = [
  '#14489C',
  '#FF3700',
  '#07400E',
  '#800900',
  '#5D00FF',
  '#FFC000'
]
sb.set_theme(
  palette=cores,
  font_scale=2.0
)
```

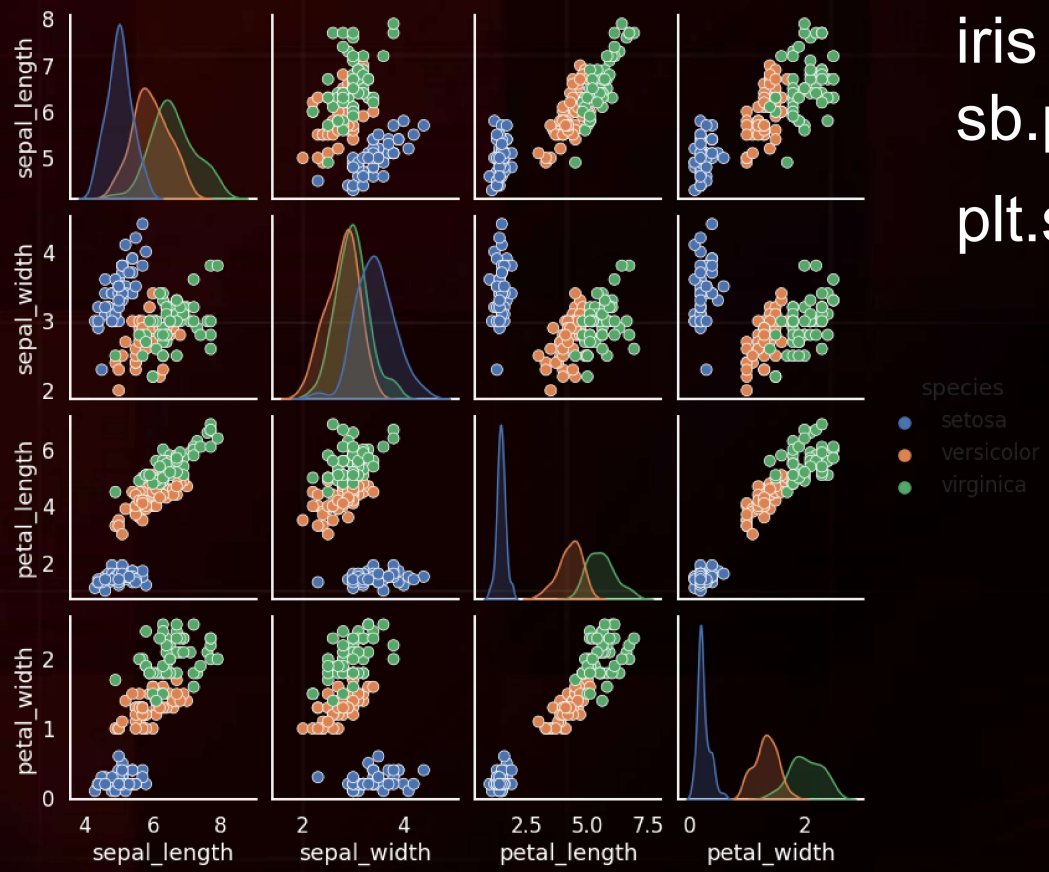


SEABORN - VIOLINPLOT

```
#fundo transparente e eixos em branco
parametros = {
  'axes.facecolor':(0,0,0,0),
  'figure.facecolor':(0,0,0,0),
  'xtick.color': 'white',
  'ytick.color': 'white',
  'axes.edgecolor': 'white'
}
sb.set_theme(
  palette=cores,
  font_scale=2.0,
  rc=parametros
)
```



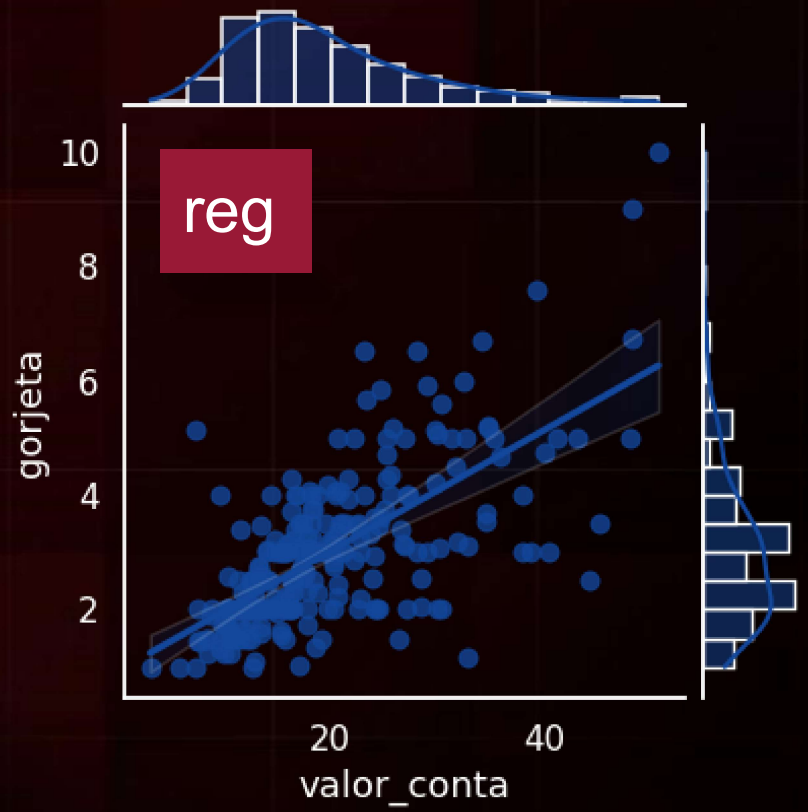
SEABORN - PAIRPLOT

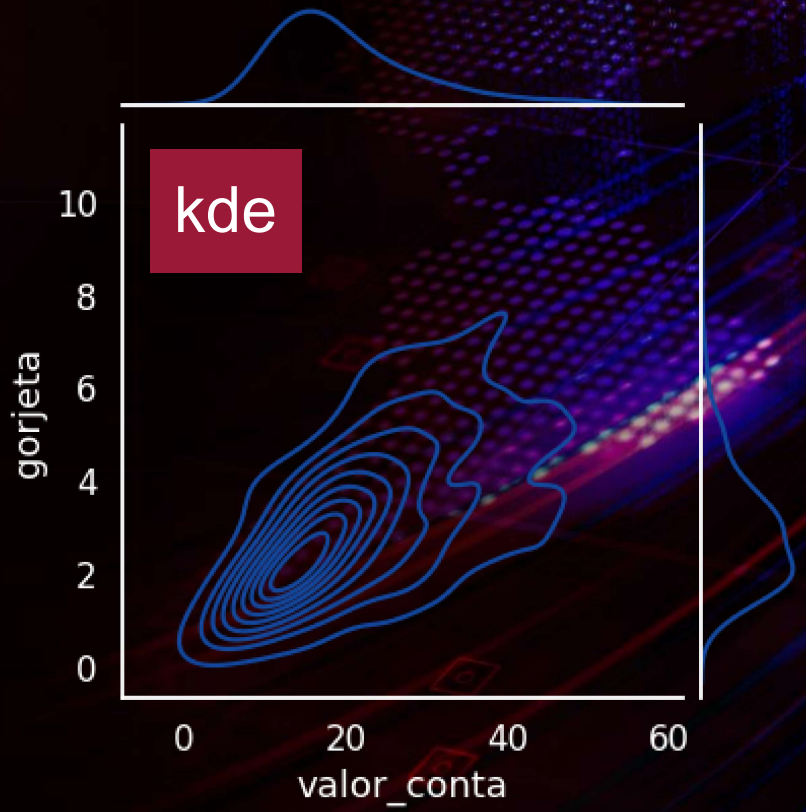



```
iris = sb.load_dataset('iris')
sb.pairplot(iris, hue='species')
plt.show()
```

SEABORN - JOINTPLOT

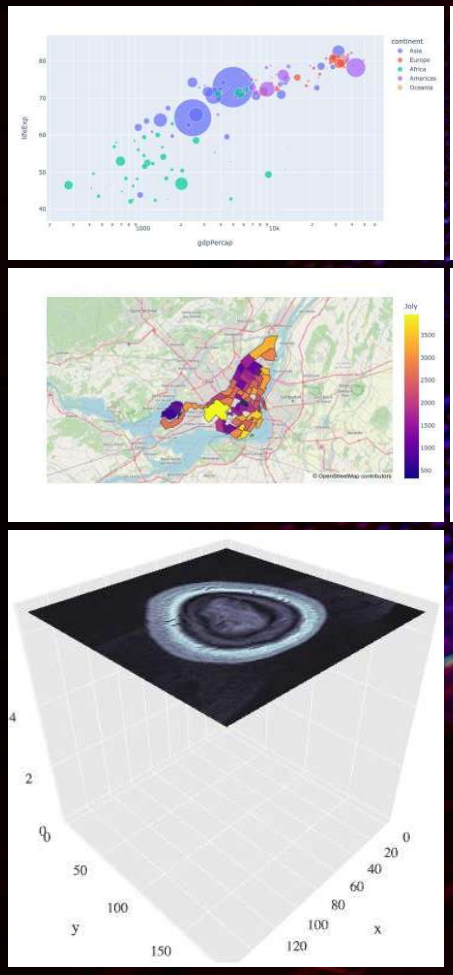
```
sb.jointplot(x='valor_conta',y='gorjeta',data=tips, kind='reg',color='#144890
```





PLOTLY

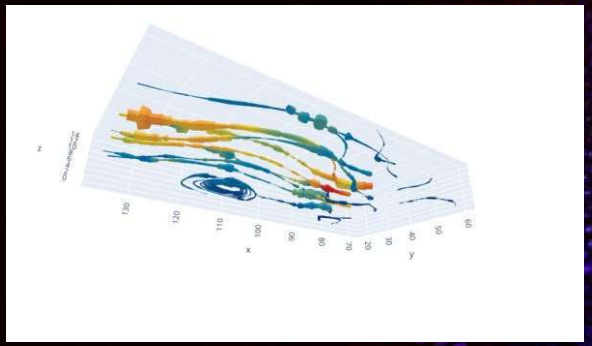
- Baseada na plotly.js
- Visualizações interativas:
  - 3D, animações, filtros
- Sintaxe simples
- Exportar para HTML
- Variedade de visualizações
  - Visual elaborado

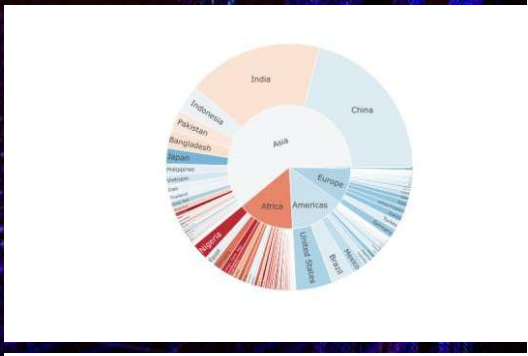


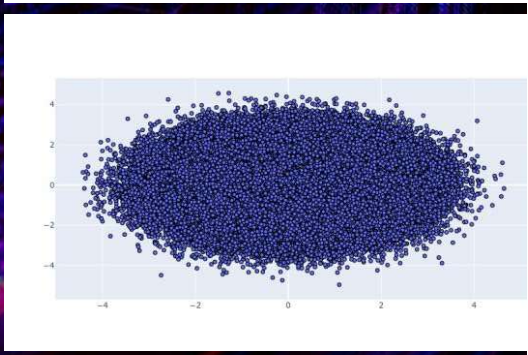
Fonte: Plotly

PLOTLY - DESVANTAGENS

- Integração com as bases de dados
- Documentação desatualizada

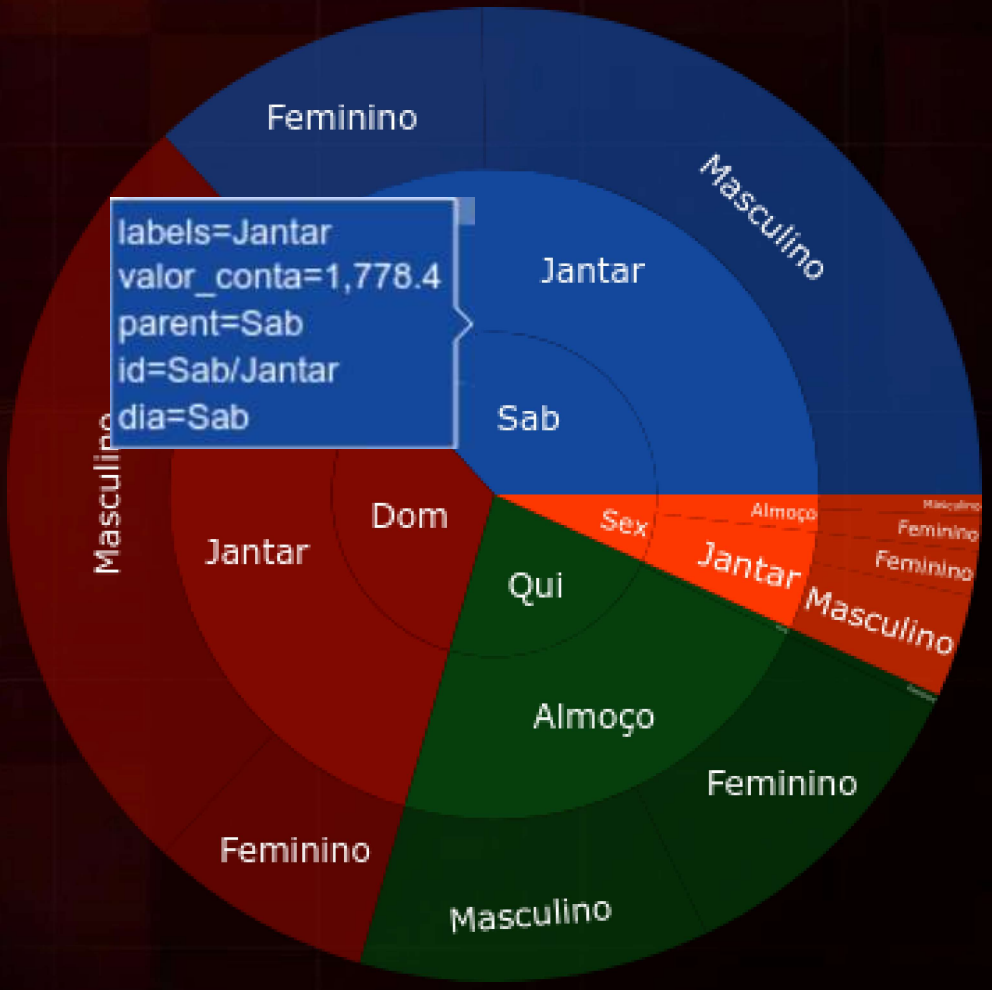






Fonte: Plotly

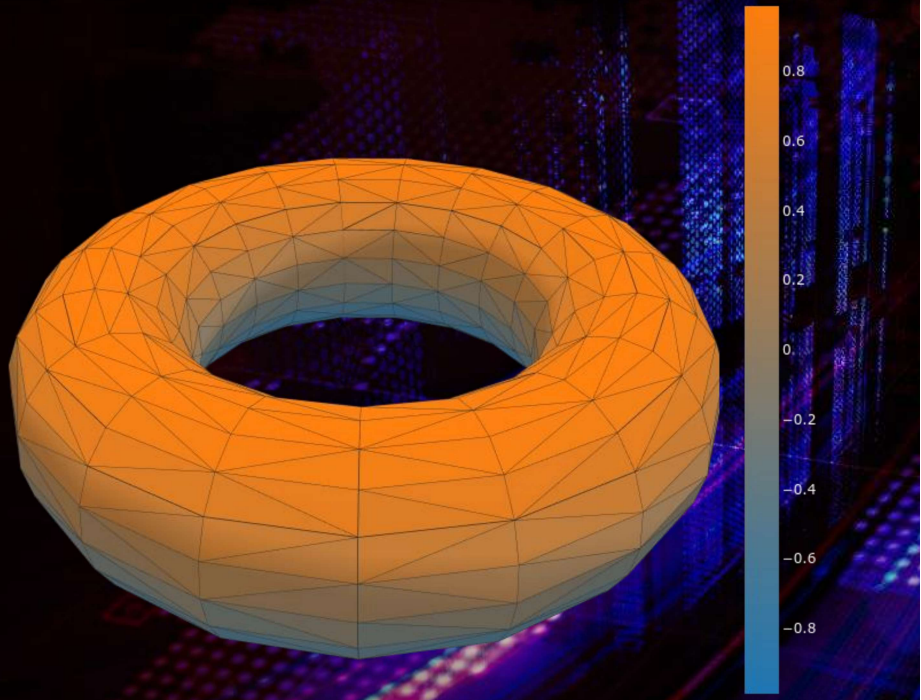
PLOTLY - SUNBURST




```
import plotly.express as px

figura = px.sunburst(tips, path=['dia', 'período', 'sexo'], values='valor_conta')

figura.show()
```

PLOTLY - SUNBURST

```
import plotly.figure_factory as ff
import numpy as np
from scipy.spatial import Delaunay

# gera posições no espaço 3D: x, y, z

#gera malha do objeto: vertices

malha = ff.create_trisurf(x=x, y=y, z=z,
        simplices=vertices,
        aspectratio=dict(x=1, y=1, z=0.3))
malha.show()
```

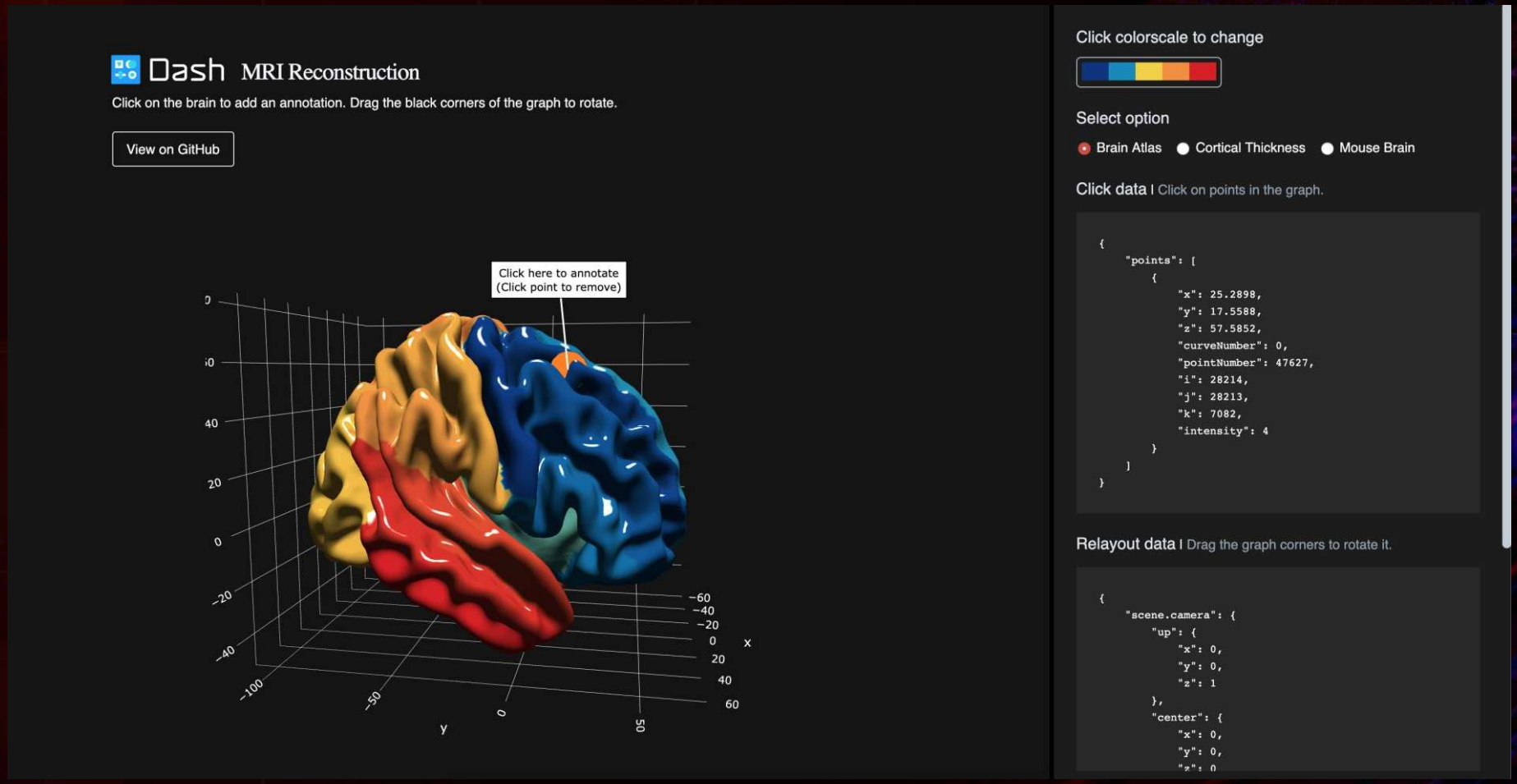


PLOTLY - CONTROLES (DASH)

URL:
https://plotly.com/pyth
scatter-plots/

```
from dash import Dash, dcc, html, Input, Output
app = Dash(__name__)
```

PLOTLY - CONTROLES (DASH)

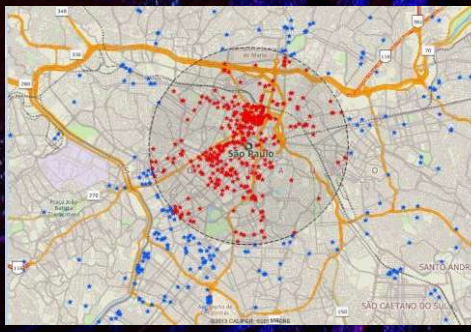


Dash Brain Viewer
Fonte: Dash Sample Apps, GitHub

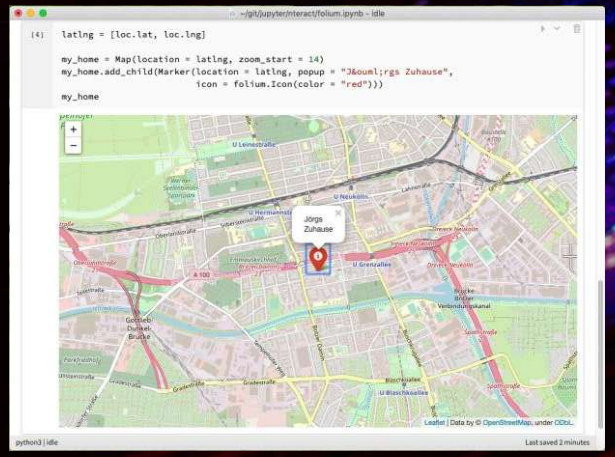
FOLIUM

- Baseada na leaflet.js
- Mapas estáticos e interativos
- Facilita a criação de mapas
- OpenStreetMap
- Variedade de camadas
- Exportar para HTML

Fonte: planemad, Wikimedia Commons



Fonte: Rafael Chimidt, Medium

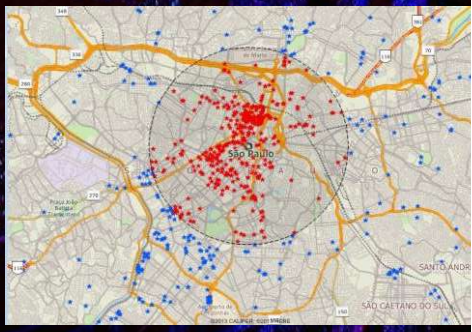


Fonte: Jorg Kantel, Flickr

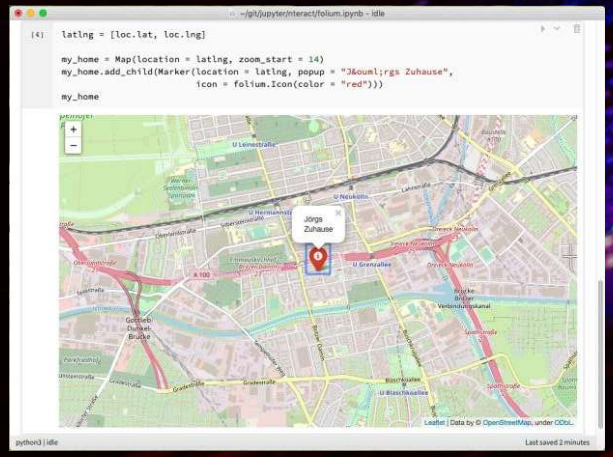
FOLIUM - DESVANTAGENS

- Obtenção dos mapas (basemap)

Fonte: planemad, Wikimedia Commons



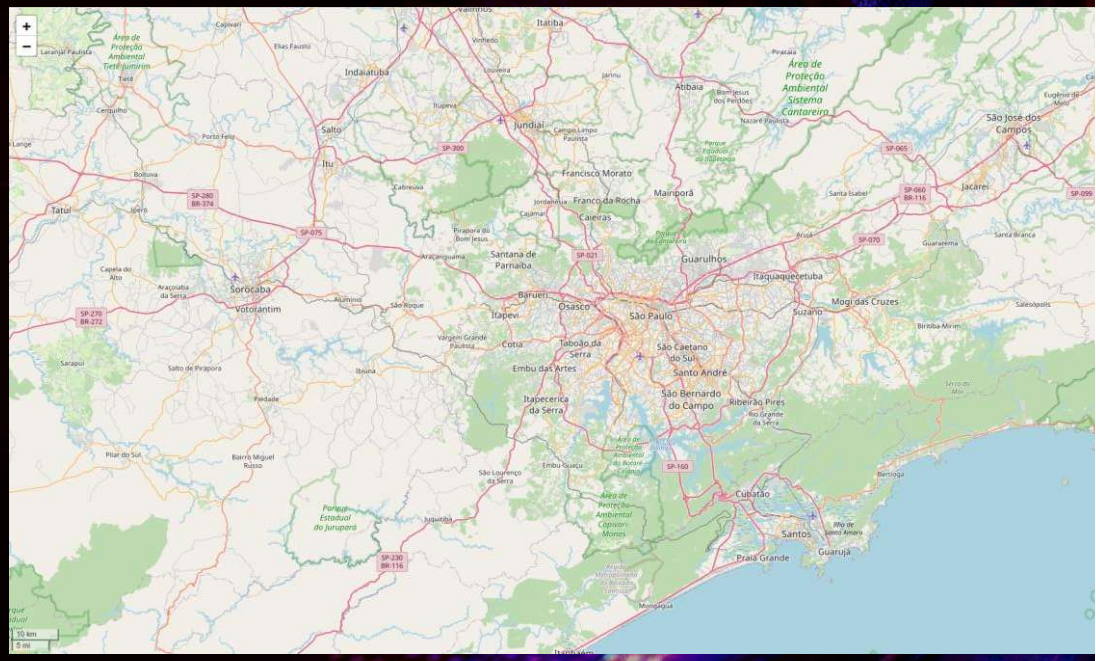
Fonte: Rafael Chimidt, Medium



Fonte: Jorg Kantel, Flickr

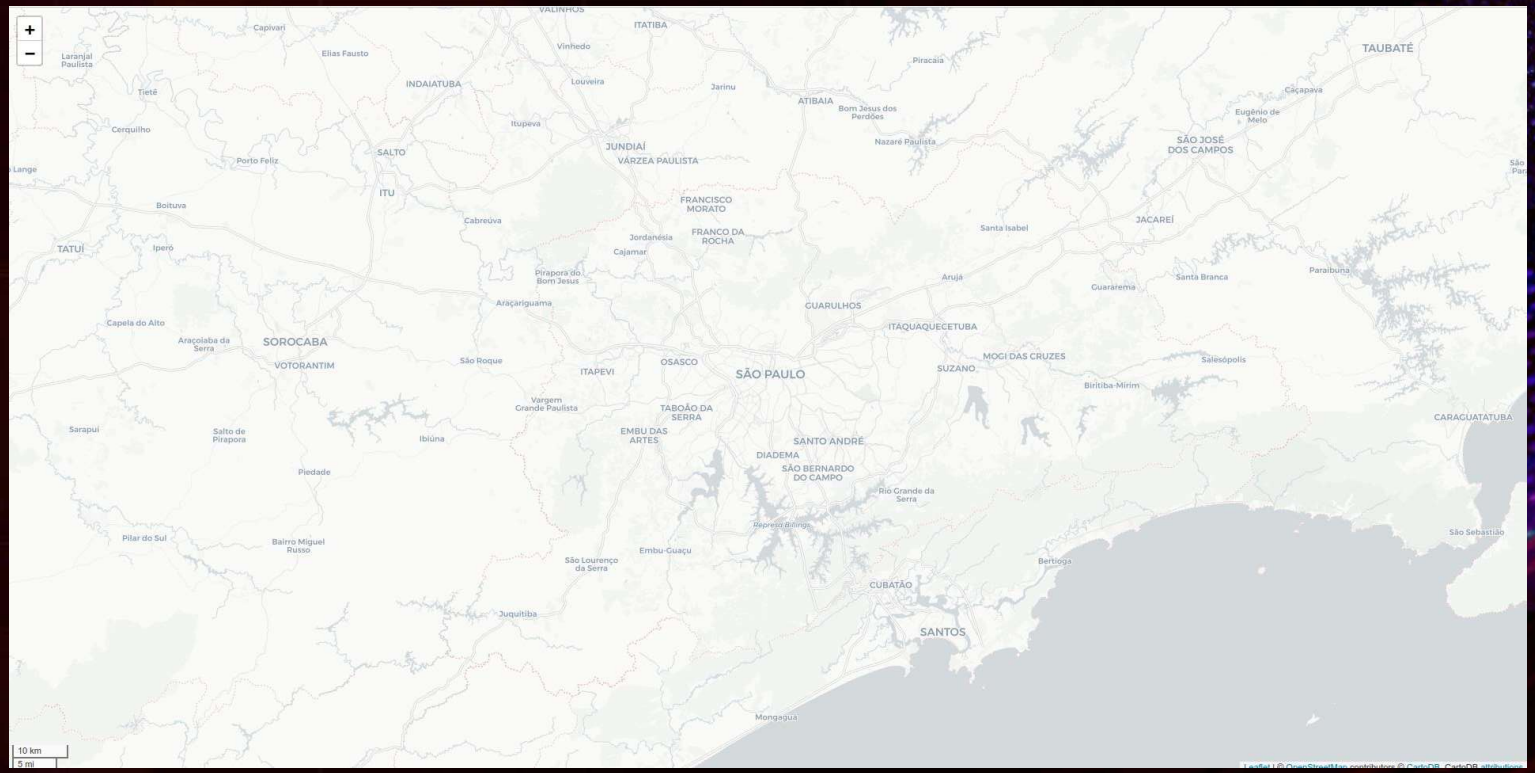
FOLIUM - MAPA

```
import folium

lat_sp = -23.5789
lon_sp = -46.6388
area = [lat_sp, lon_sp]

mapa = folium.Map(location=area,
                  control_scale = True,
                  zoom_start=10)
display(mapa)
```



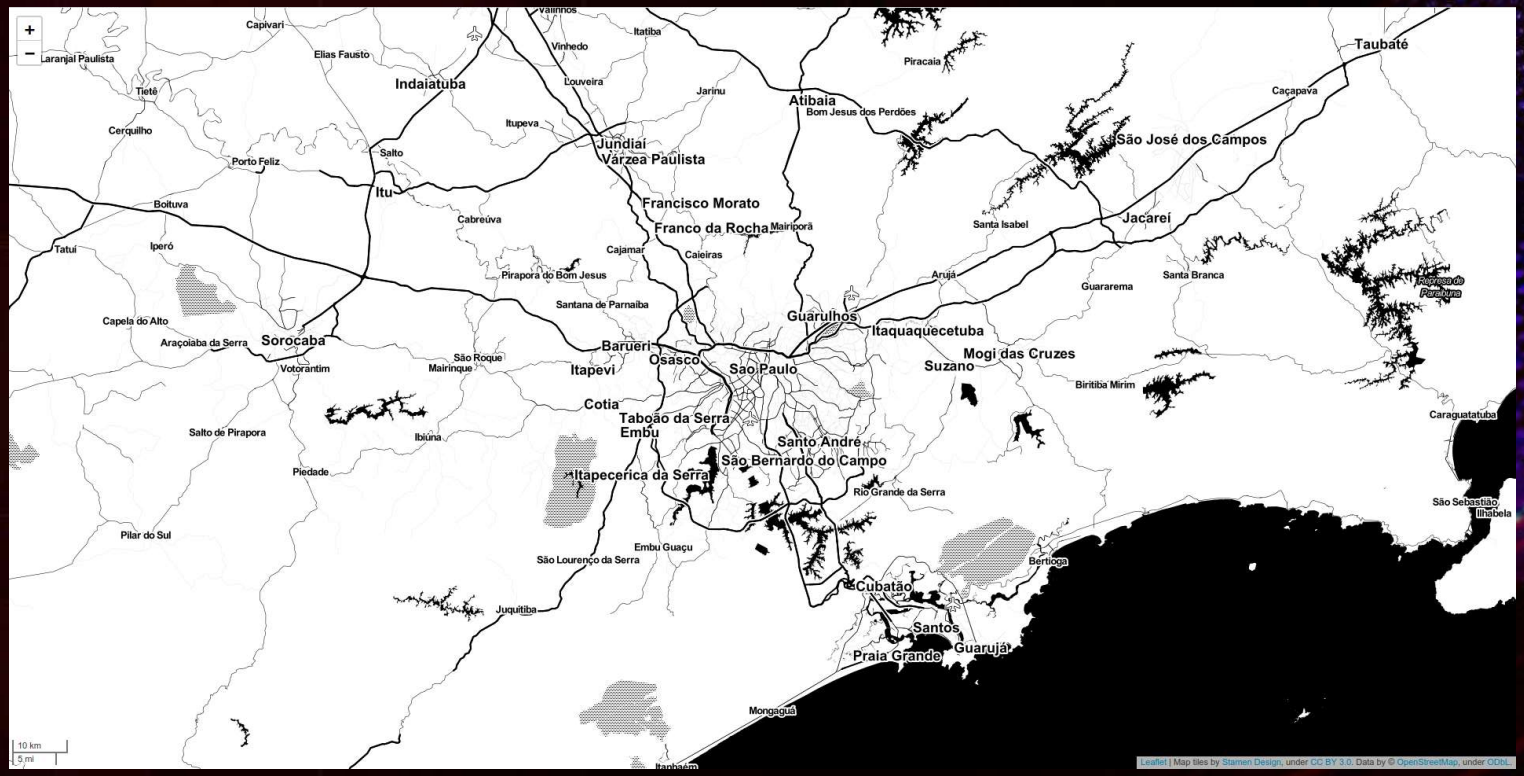
FOLIUM - CAMADAS

```
mapa = folium.Map(location=area,
                  control_scale = Tr
                  zoom_start=10,
                  tiles='cartodbposit
```

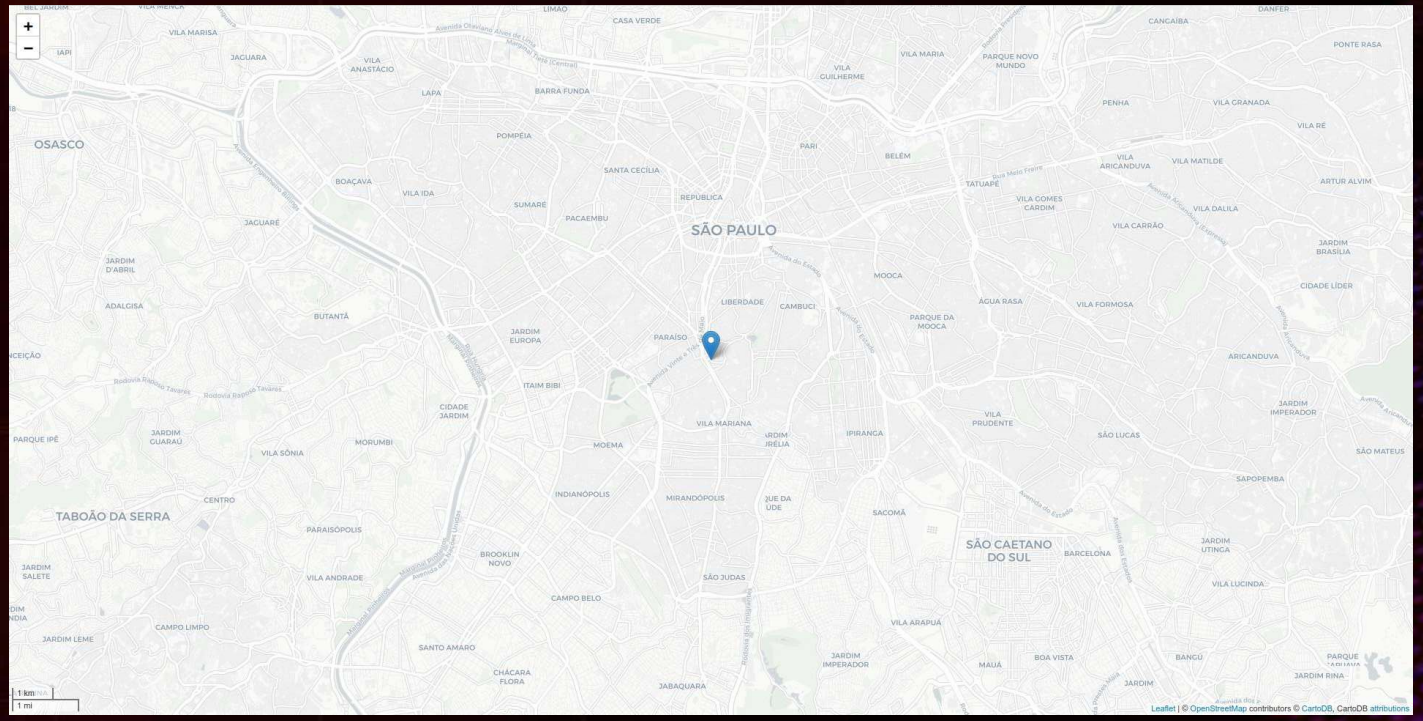


FOLIUM - CAMADAS

```
mapa = folium.Map(location=area,
                  control_scale = Tr
                  zoom_start=10,
                  tiles='stamentoner
```

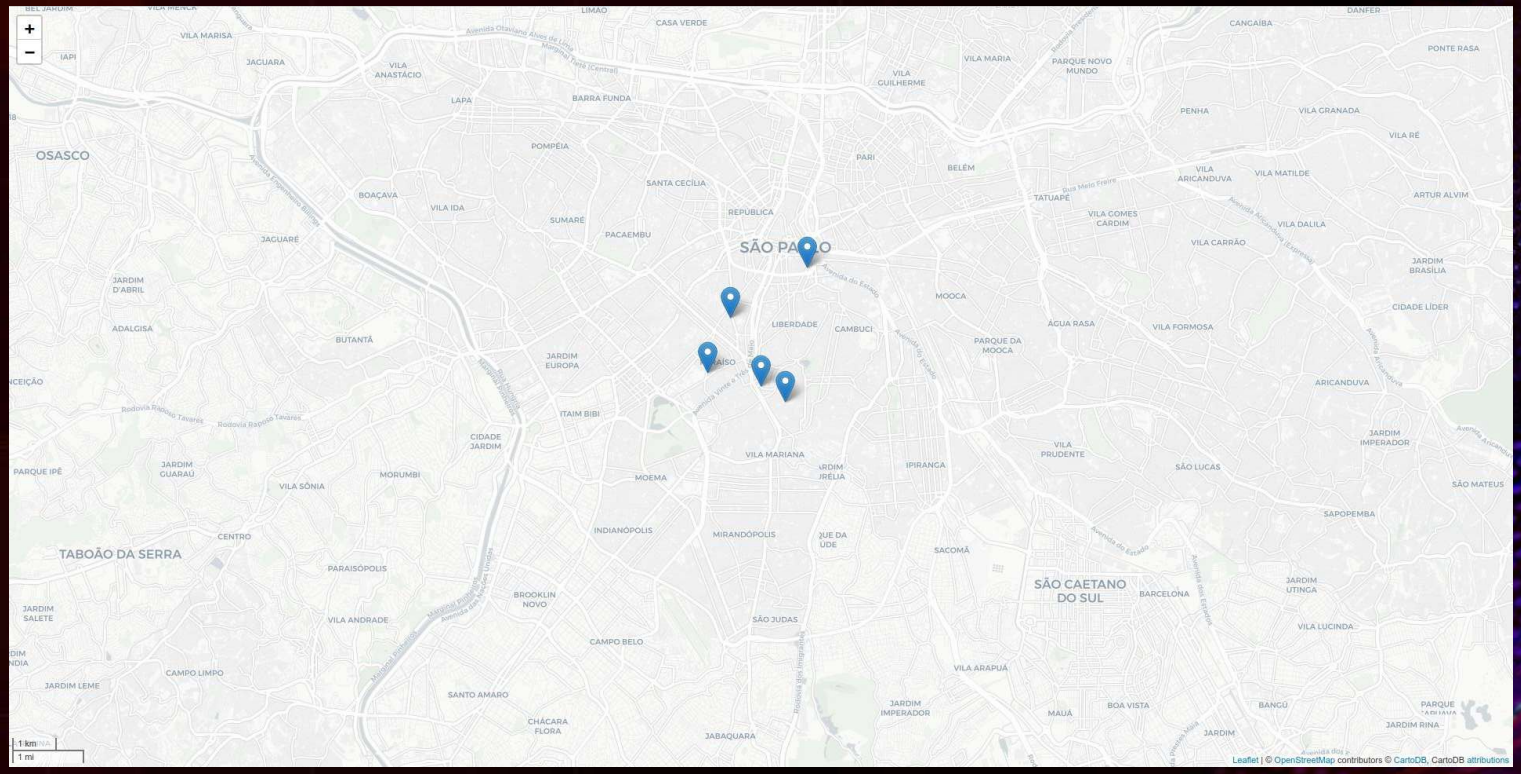


FOLIUM - MARCADORES



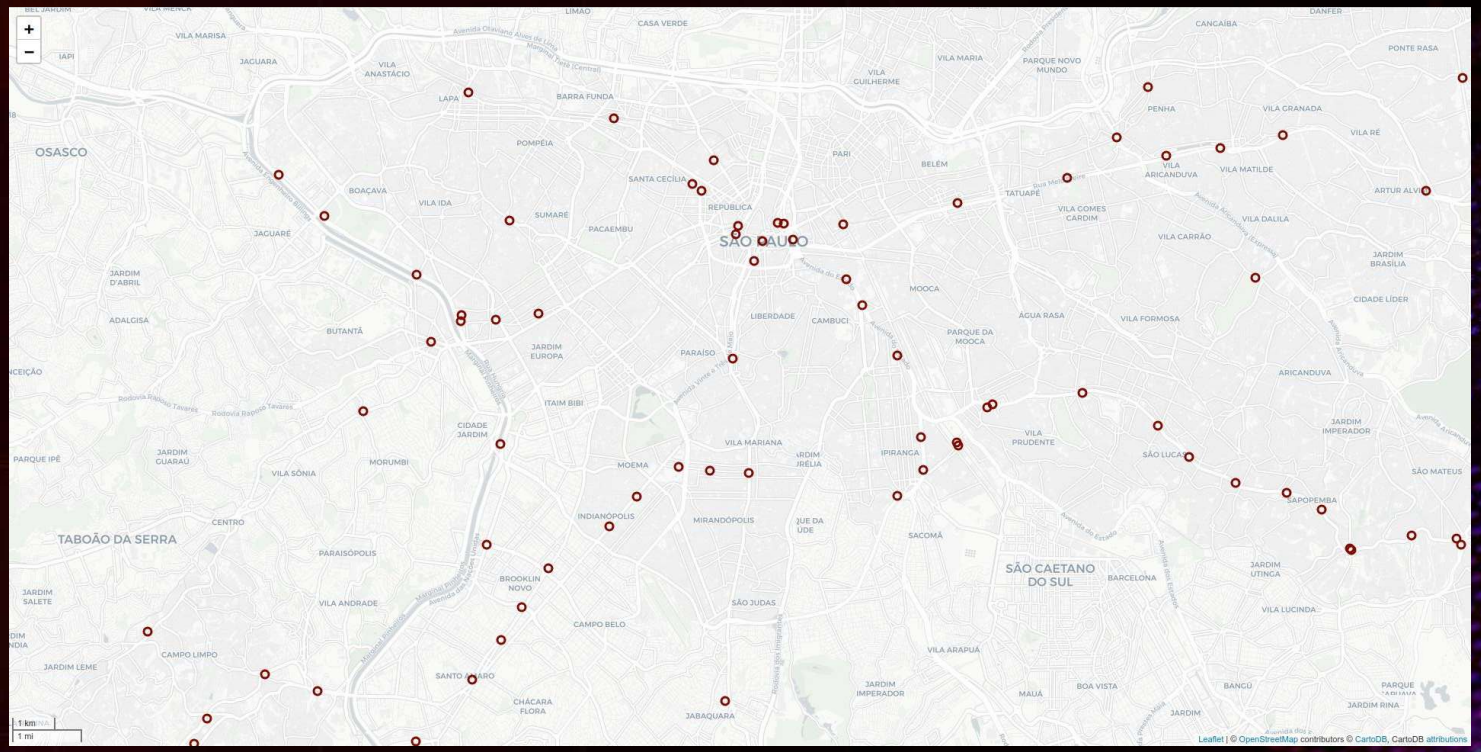

```
mapa = folium.Map(location=area, zoom_start=12)

folium.Marker(location=[-23.5789,-46.6388],
                        popup='Centro').add_to(mapa)
```

FOLIUM - MARCADORES




```
lats=[-23.5789,-23.5763,-23.5652,-23.5821,-23.5552]
lons=[-46.6388,-46.6505,-46.6456,-46.6336,-46.6287]
for i in range(0,len(lats)):
    folium.Marker(location=[lats[i],lons[i]]).add_to(mapa)
```

FOLIUM - MARCADORES



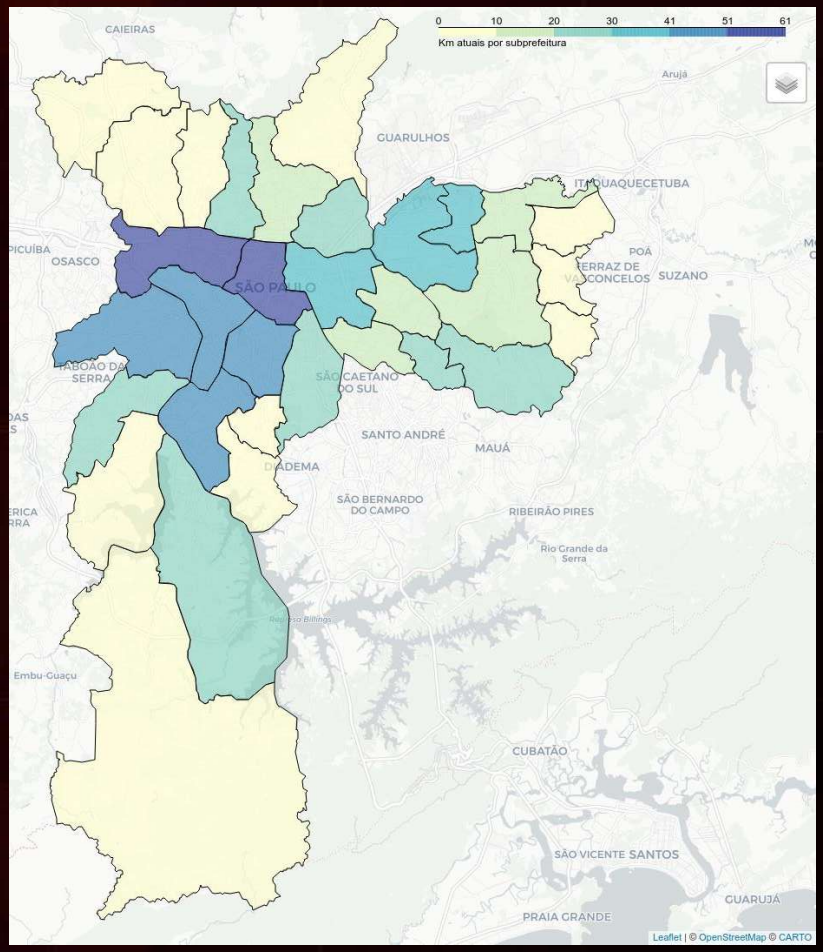

```
pontos = gpd.read_file('arquivo.shp')

for index, ponto in pontos.iterrows():
  folium.CircleMarker(location=[ponto.geometry.y,
                                ponto.geometry.x]).add_to(mapa)
```

FOLIUM - LINHAS


```
estilo = lambda styl
  'color': 'blue',
  'weight': 2
  }
```

```
ciclovias = gpd.read_file('arquivo.shp')
ciclovias.crs = {'init': 'epsg:4326'}

folium.GeoJson(ciclovias, style_function=estilo).add_to(mapa)
```

FOLIUM - POLÍGONOS




```
subprefeituras_km = gpd.read_file('arquivo.shp')

choropleth = folium.Choropleth(
    geo_data=subprefeituras_km,
    data=subprefeituras_km,
    columns=['sp_nome','km'],
    key_on='feature.properties.sp_nome',
    legend_name='Km atuais por subprefeitura',
    fill_color = 'YlGnBu'
).add_to(fmap)
```

REFERÊNCIAS

Visualização de dados. Capítulo: Frameworks de visualização de dados. Alessandra M. Paz Milani et al. Editora SAGAH, 2020.

Matplotlib - Documentação: https://matplotlib.org/

Seaborn - Documentação: https://seaborn.pydata.org/

Seaborn - Controlling figure aesthetics (Controlando a estética das figuras). Michael Waskon, Seaborn.
Link: https://seaborn.pydata.org/tutorial/aesthetics.html

Plotly - Documentação: https://plotly.com/python/

Dash Brain Viewer
Link: https://github.com/plotly/dash-sample-apps/tree/main/apps/dash-brain-viewer

Folium - Documentação: https://python-visualization.github.io/folium