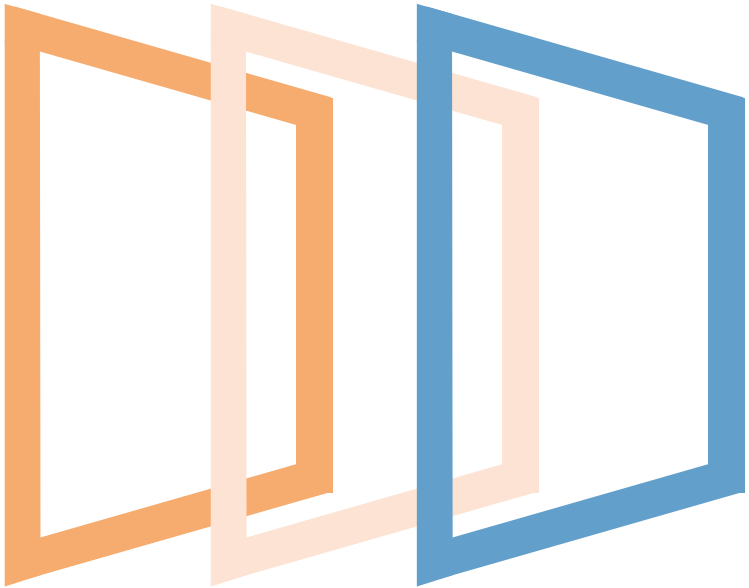
Análise Gráfica com Python

Thaís Ratis

Inteligência Artificial Brasil, 29.04.2025



minsait

An Indra company

Thaís Ratis

Instrutora

Formação Acadêmica

B.Sc. Ciência da Computação

Esp. Bioinformática

M.Sc. Bioinformática

Ph.D. Bioinformática

Área Profissional

Cientista de Dados

Contato no Teams

talmeidar@minsait.com

Conteúdo programático

Aula 01: Matplotlib

Aula 02: Seaborn

Aula 03: Plotly e Prova Teórica

Aula 04: Prova prática

Matplotlib

Aula 01

Objetivo

Capacitar os alunos a utilizarem as bibliotecas Matplotlib para criar visualizações de dados claras e impactantes. Pretende-se explorar técnicas de visualização e customização gráfica, ampliando as habilidades dos alunos na apresentação visual de informações. A meta é fornecer ferramentas essenciais para a comunicação eficaz de resultados de análise de dados.

Seaborn

Aula 02

Objetivo

Capacitar os alunos a utilizarem as bibliotecas Seaborn para criar visualizações de dados claras e impactantes. Pretende-se explorar técnicas de visualização e customização gráfica, ampliando as habilidades dos alunos na apresentação visual de informações. A meta é fornecer ferramentas essenciais para a comunicação eficaz de resultados de análise de dados.

Plotly e Prova Teórica

Aula 03

Objetivo

Capacitar os alunos a utilizar a biblioteca Plotly para criar visualizações interativas e dinâmicas de dados. Pretende-se explorar as capacidades avançadas de plotagem e interatividade oferecidas pelo Plotly, ampliando as habilidades dos alunos na apresentação e exploração visual de informações.

Formato avaliação - Teórica

Avaliação teórica: formulário;

Avaliar a capacidade dos participantes em aplicar os conceitos aprendidos de visualização de dados para serem aplicados em problemas reais de análise de dados.

Prova prática

Aula 04

Objetivo

Avaliar a capacidade dos participantes em aplicar os conceitos para resolver problemas reais de análise de dados. Os alunos serão desafiados a realizar tarefas de visualização e interpretação de dados utilizando as bibliotecas matplotlib, Seaborn e Plotly.

Formato avaliação - Pratica

Avaliação prática: será realizada em quarteto. Cada quarteto escolherá uma base no site, realizará a EDA com visualização. Por fim, deverão apresentar um *pitch* com um pptx em torno de 5 minutos com os principais pontos analisados, quais os *insights* e conclusões chegaram.

Visualização de Dados

Conteúdo programático

1. Crescimento dos Dados
2. Storytelling
3. Visualização de Dados
4. Matplotlib
5. Seaborn
6. Plotly

Crescimento dos dados

01

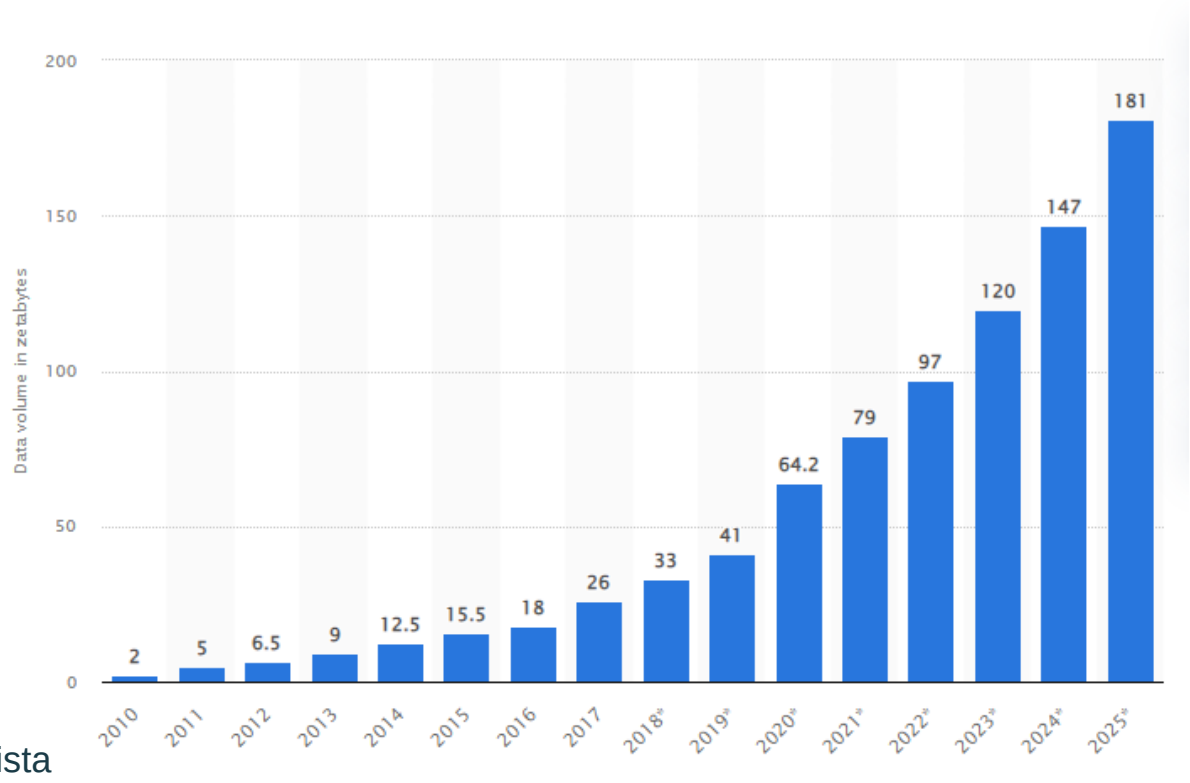
Crescimento dos dados

Fonte: Ouse tecnologia web

A quantidade de informações geradas, coletadas e armazenadas está aumentando exponencialmente, moldando o mundo em que vivemos.

O crescimento de dados é impulsionado por diversos fatores, como o aumento da conectividade, o avanço da tecnologia, a proliferação de dispositivos inteligentes e a digitalização de processos. Desde transações financeiras e registros médicos até interações nas redes sociais e dados coletados por sensores, cada vez mais aspectos da nossa vida cotidiana são registrados e transformados em dados.

Volume de dados/informações criados, capturados, copiados e consumidos mundialmente de 2010 a 2020, com previsões de 2021 a 2025



Captura de dados

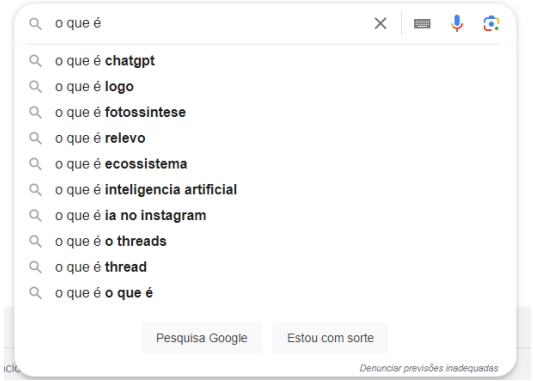
ChatGPT

Spotify®

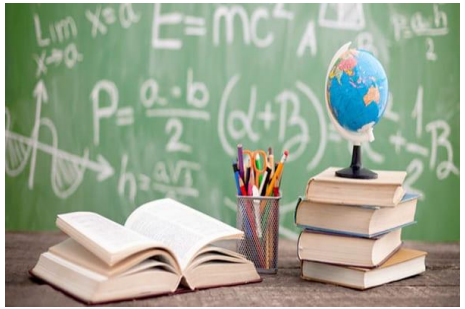
facebook

NETFLIX

Google



Áreas impactadas



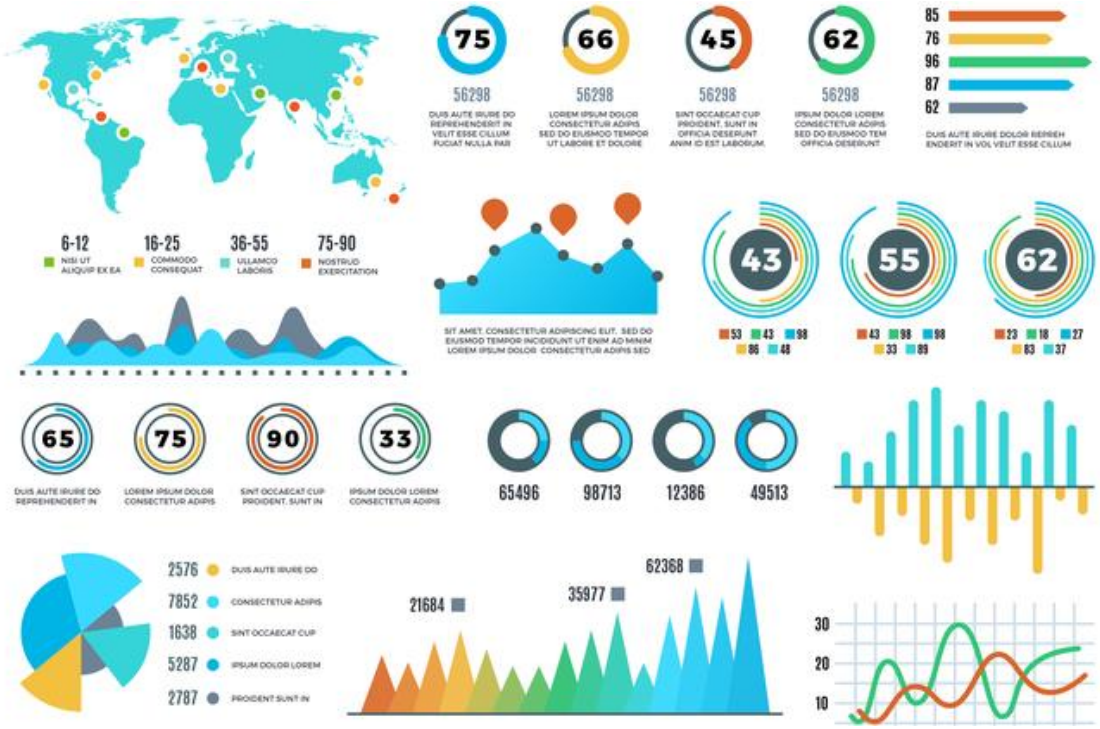
Storytelling

02

Storytelling

Storytelling em ciência de dados é a prática de contar histórias com base em *insights* e análises de dados. Envolve a habilidade de transformar informações complexas em narrativas acessíveis e envolventes, utilizando visualizações de dados, gráficos e exemplos concretos para comunicar descobertas e tendências. O *storytelling* em ciência de dados permite que os profissionais apresentem de forma clara e persuasiva os resultados de suas análises, conectando os dados a contextos relevantes e fornecendo *insights* acionáveis para tomadas de decisão informadas. Essa abordagem ajuda a comunicar a importância dos dados e a envolver o público em um processo de compreensão mais profunda e aplicação prática.

Storytelling



Storytelling

1. Conhecer o público;
2. Objetivo da apresentação;
3. Contexto do problema.

Uma história pode ser dividia em **3 atos:**

PREPARAÇÃO
Você vai mostrar ao público **o motivo de valer a pena ficar ali**, vai introduzir as informações mais relevantes e explicar o **contexto**

CONFLITO
Qual o **problema** que queremos resolver ou a **oportunidade** que temos de melhoria? O que justifica isso que estamos mostrando?

SOLUÇÃO
Apresentar as **ideias** que serão usadas para resolver o conflito e **chamar para a ação** ou gerar uma **discussão**

Fonte: hashtagtreinamentos

Storytelling - Conteúdo

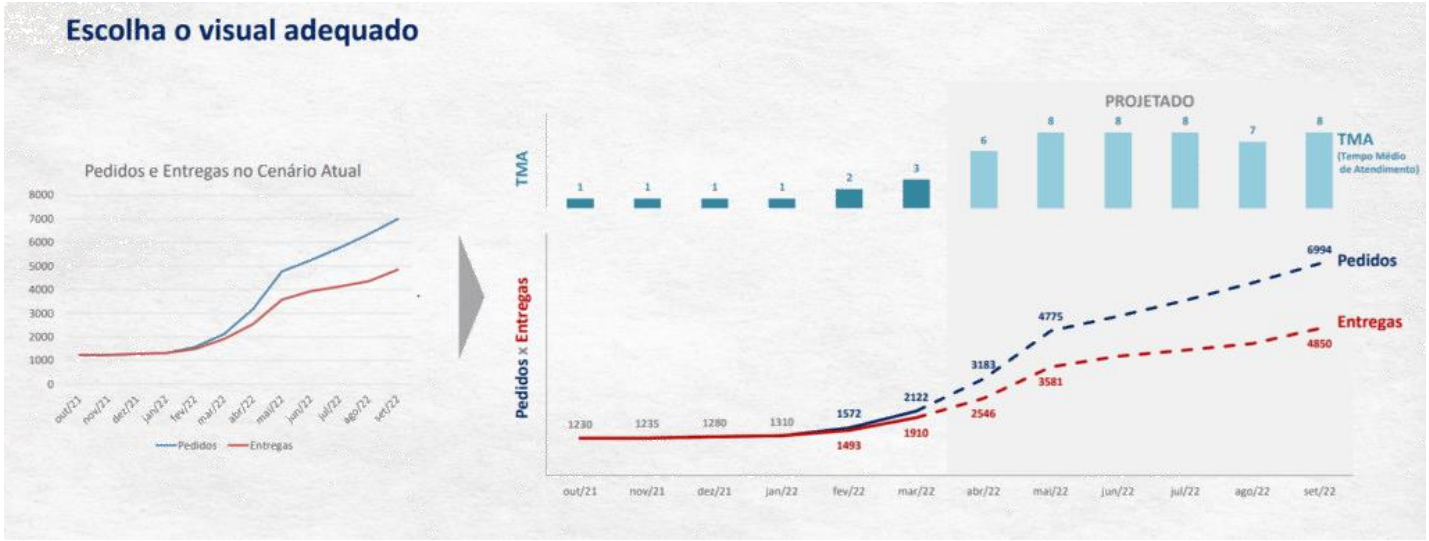
Mostre apenas o que é realmente importante

Retire a saturação
(elimine os ruídos)

Cuidado com os tipos de gráficos e muitas cores

Entenda o tipo de apresentação
(pessoal ou e-mail?)

Fonte: hashtagtreinamentos



Visualização de dados

03

Visualização de dados

Através da visualização de dados, podemos transformar conjuntos complexos de informações em gráficos e gráficos informativos, permitindo-nos descobrir padrões, tendências e *insights* ocultos.

Tipos de Gráficos

3.1

Tipos de Gráficos

Objetivo de transmitir uma informação importante de um único ponto de dado.

Tipos de Gráficos

Objetivo de comparação de valores ou categorias.

Tipos de Gráficos

Objetivo de exibir uma alteração ao longo do tempo ou local.

Tipos de Gráficos

Objetivo de exibir agrupamentos ou ranking de dados (informação ordinal ou hierárquica).

Tipos de Gráficos

Objetivo de exibir relações entre variáveis (muito utilizado em análise exploratória).

Gráfico de linha

Quando estamos interessados em expressar variáveis que representam passagem de tempo, mostrar a evolução histórica, acompanhar a evolução de uma variável em relação a um ou mais limites existentes.

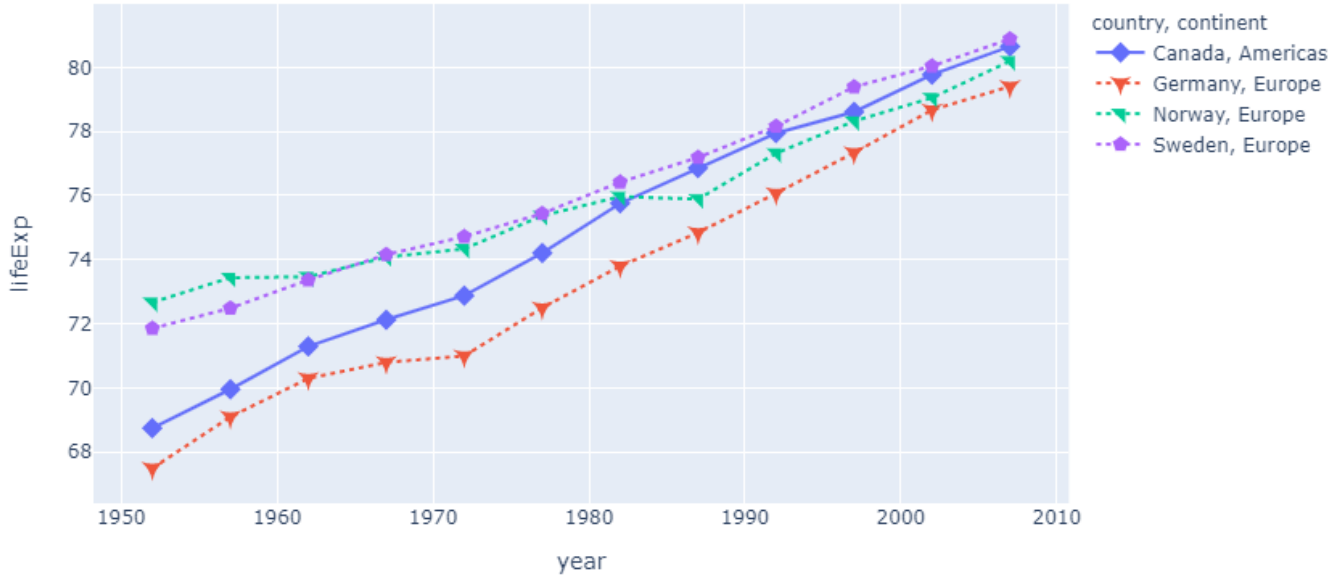


Gráfico de barras

Representação gráfica da distribuição de frequência de variáveis qualitativas (categorias).

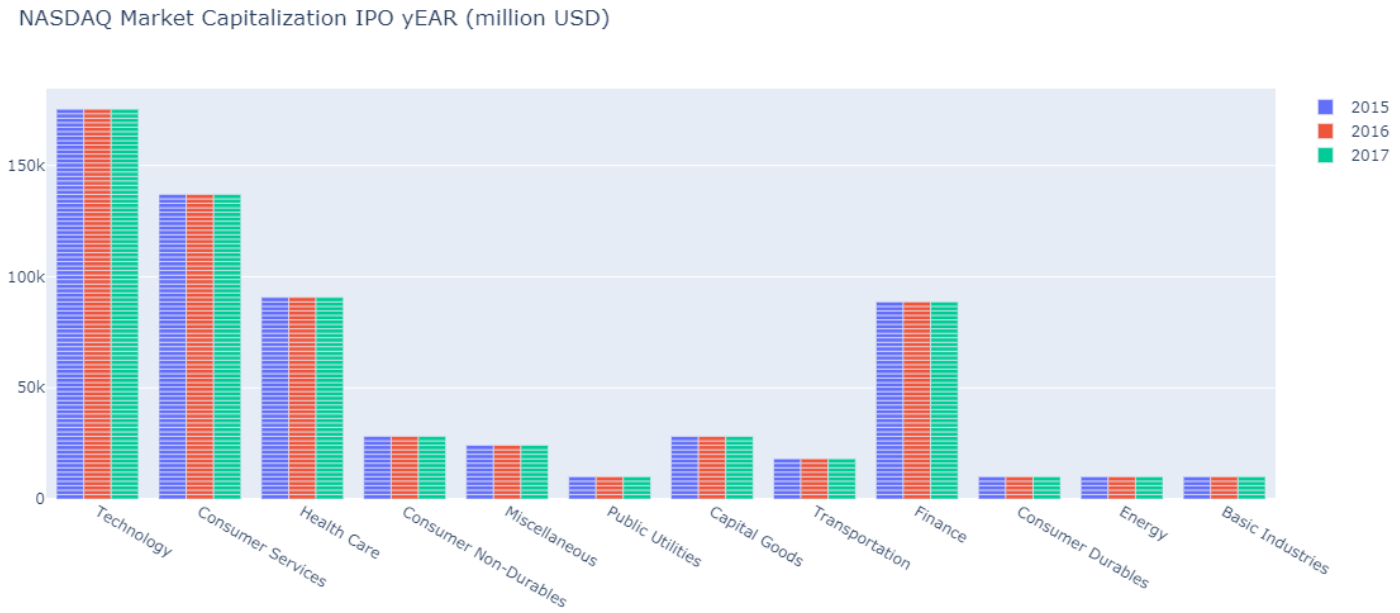


Gráfico de pizza ou gráfico de setores

Visualização que representa um valor relativo de cada categoria estabelecida em relação a um todo, são usados para visualizar partes de um todo, suas “fatias” devem sempre somar 100%.

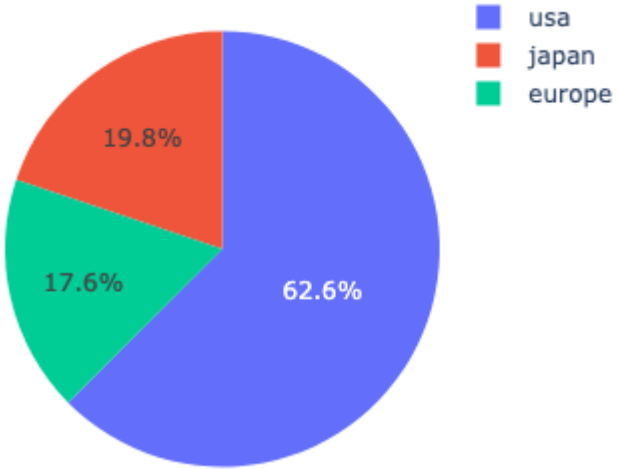
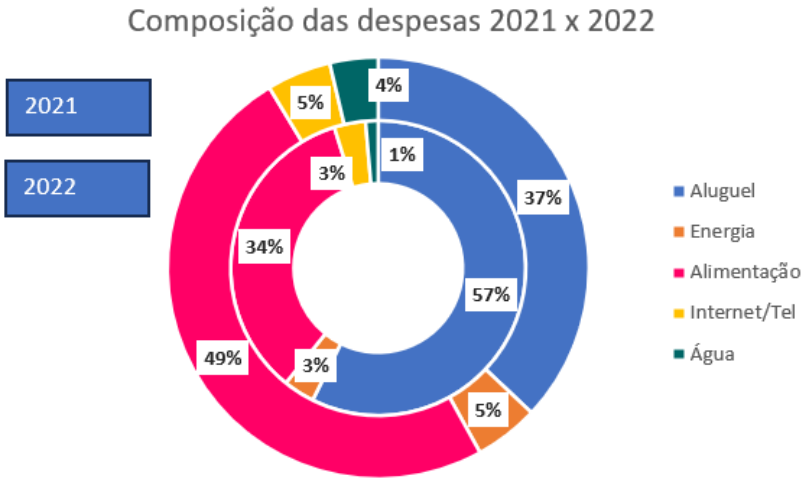


Gráfico de rosca

É uma variação de um gráfico de pizza padrão. Um gráfico de rosca normalmente mostra as proporções de dados categóricos em que o tamanho de cada pedaço da rosca comunica a proporção de cada categoria.



Fonte: hashtagtreinamentos

Gráfico de dispersão

Tipo de gráfico utilizado para representar a relação entre duas variáveis quantitativas., por exemplo, para saber se existe correlação entre variáveis.

Gráfico de bolha

Variação do gráfico de dispersão. Quando temos uma terceira variável numérica, como população, uma variação comum é em relação ao tamanho dos pontos, onde pontos maiores indicam valores mais altos.

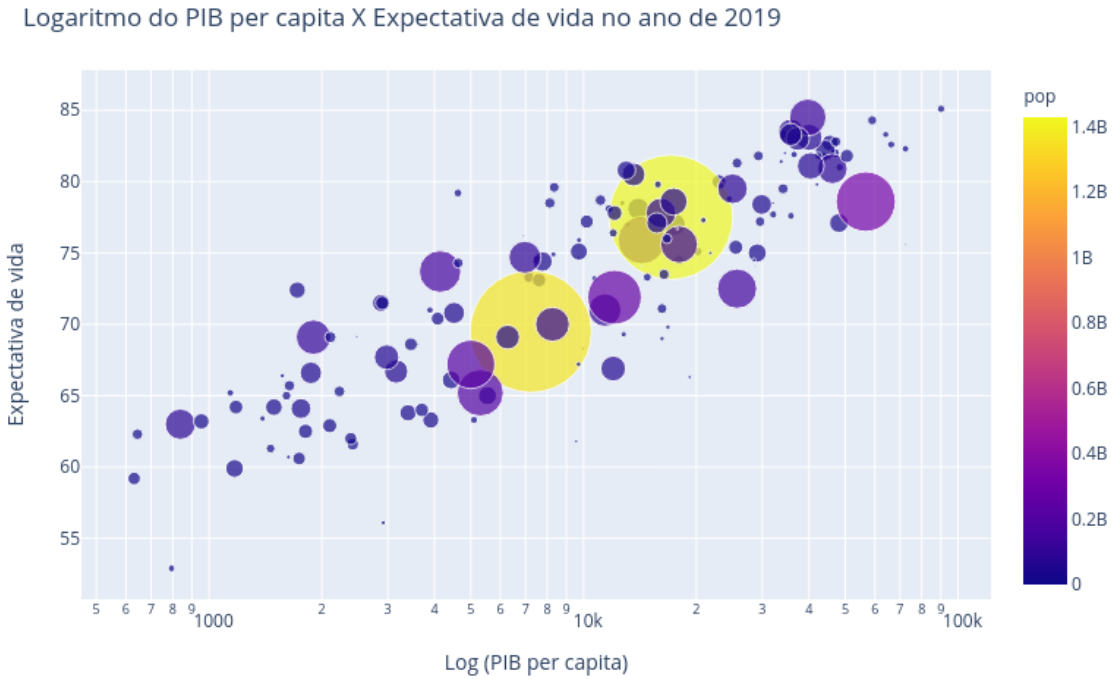
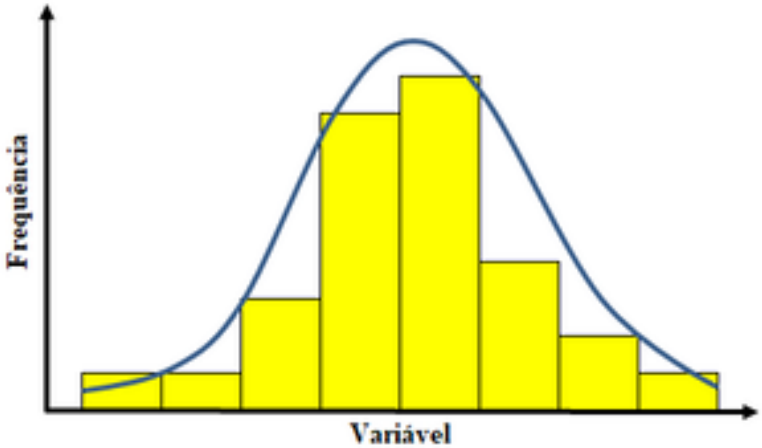


Gráfico de histograma

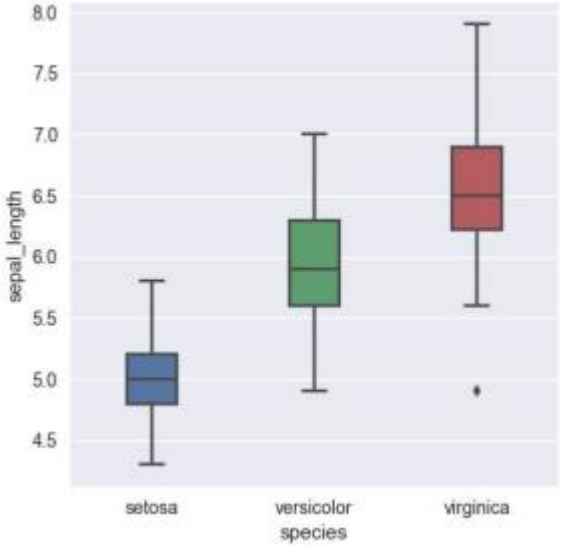
É uma espécie de gráfico de barras que demonstra uma distribuição de frequências. No histograma, a base de cada uma das barras representa uma classe e a altura representa a quantidade ou frequência absoluta com que o valor de cada classe ocorre.



Fonte: hashtagtreinamentos

Gráfico de boxplot

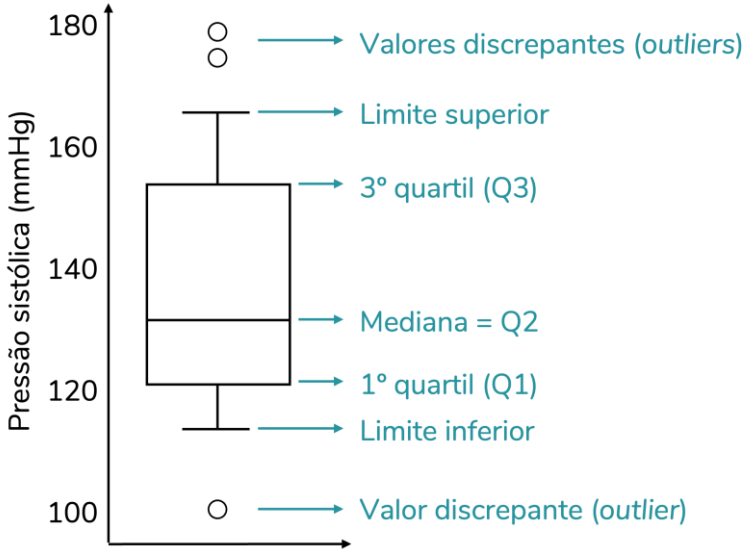
O Box Plot, também chamado diagrama de caixa, é uma ferramenta gráfica utilizada para ilustrar um conjunto de dados. Útil para visualizar outliers, distribuição dos dados, mediana dos dados e Permite ver a inclinação da distribuição. Por meio dele, é possível visualizar a distribuição de dados com base em cinco estatísticas: o mínimo; o primeiro quartil (Q1); a mediana; o terceiro quartil (Q3); o máximo.



Fonte: hashtagtreinamentos

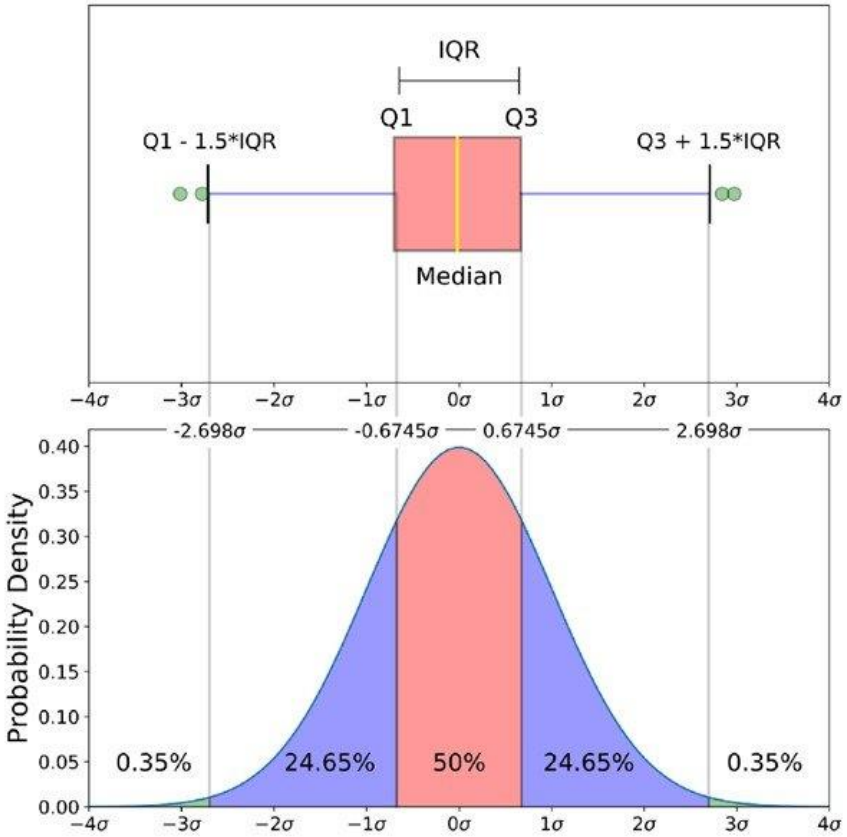
Gráfico de boxplot

- O boxplot é formado por quartis.
- Os quartis são usados para definir o comprimento da caixa.
- Os valores acima e abaixo dos quartis usados para construir as hastes (mínimos e máximos).
- Valores fora da caixa e das hastes são considerados *Outliers*.



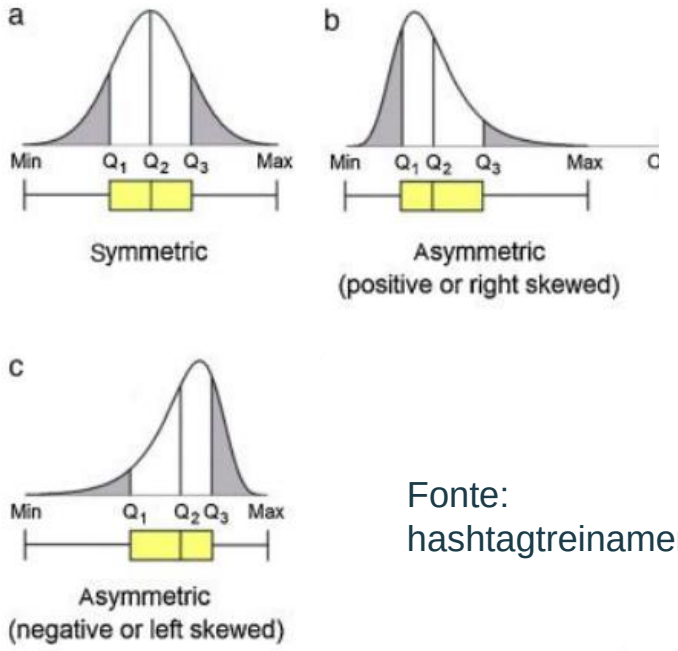
Fonte: hashtagtreinamentos

Gráfico de boxplot



Fonte: hashtagtreinamentos

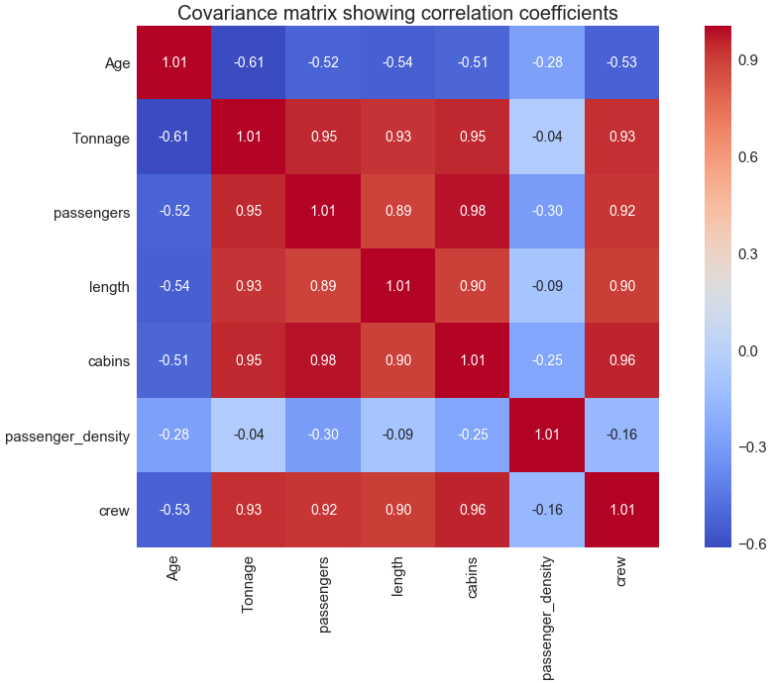
Gráfico de boxplot - Assimetrias



Fonte: hashtagtreinamentos

Mapa de calor (*heatmap*)

Um mapa de calor (ou *heatmap*) é uma representação gráfica de dados em que os valores são representados por cores.



Fonte: hashtagtreinamentos

Escolhendo cores

3.2

Escolhendo cores

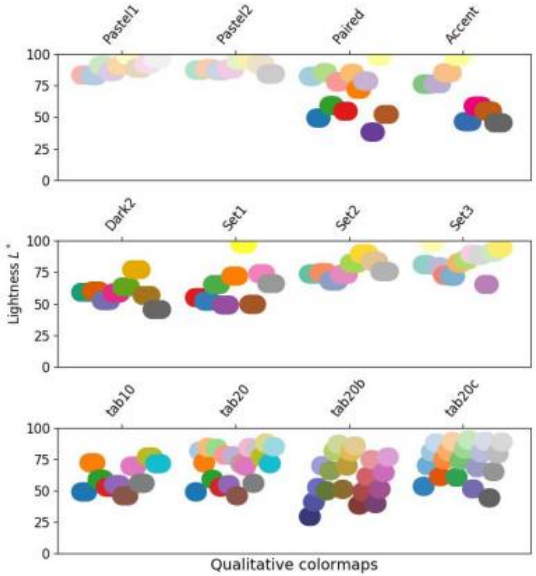
- Cores são importantes e podem ser usados para revelar padrões nos dados;
- Normalmente um mapa de cor é melhor entendido pela audiência do que apenas uma única cor;
- Mudanças em luminosidade são melhor percebidas do que mudanças em tonalidade.

Existem três tipos principais de paleta de cores para visualização de dados:

- Paletas qualitativas
- Paletas sequenciais
- Paletas divergentes

Paletas qualitativas

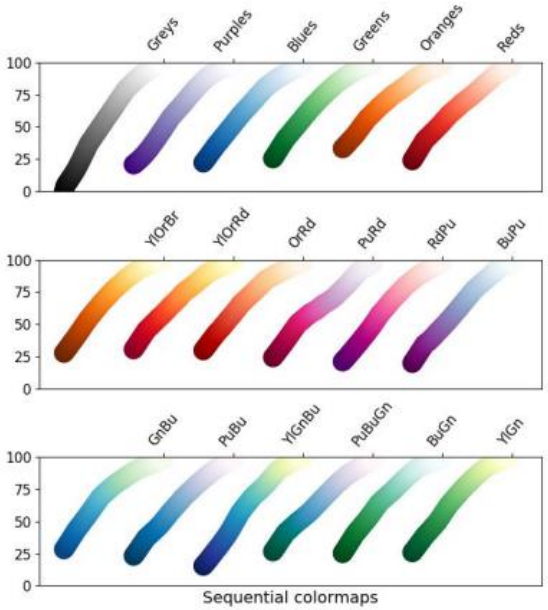
Uma paleta qualitativa é usada quando a variável é de natureza categórica. Variáveis categóricas são aquelas que assumem rótulos distintos sem ordenação inerente. Os exemplos incluem país ou estado, raça e gênero. Cada valor possível da variável é atribuído a uma cor de uma paleta qualitativa.



Em uma paleta qualitativa, as cores atribuídas a cada grupo precisam ser distintas.

Paletas sequencial

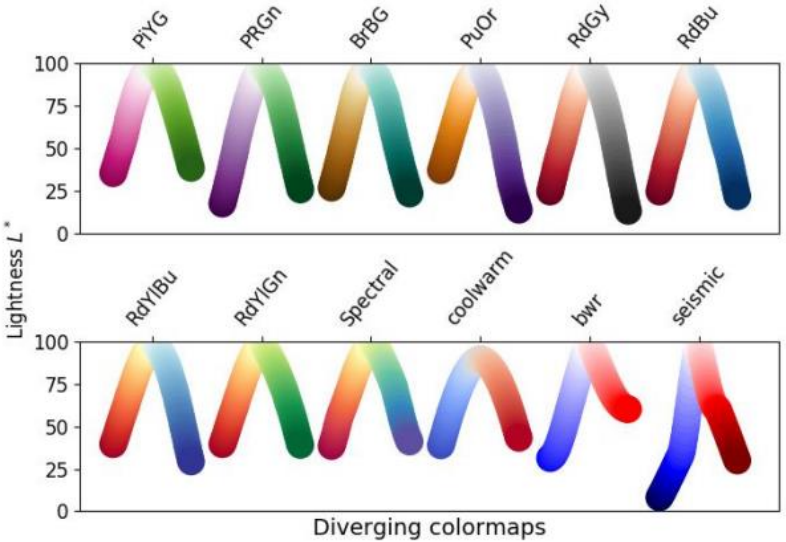
Quando a variável atribuída para ser colorida é numérica ou tem valores ordenados, então ela pode ser representada com uma paleta sequencial. As cores são atribuídas a valores de dados, geralmente com base em matiz, luminosidade ou ambos.



A dimensão de cor mais proeminente para uma paleta sequencial é sua leveza. Normalmente, valores mais baixos estão associados a cores mais claras e valores mais altos a cores mais escuras.

Paletas sequencial

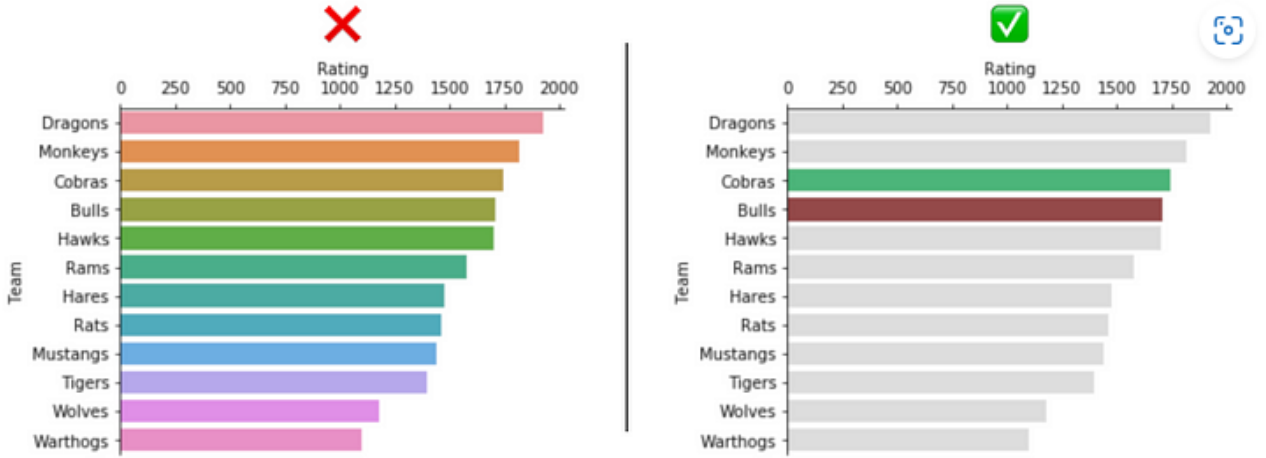
Se nossa variável numérica tiver um valor central significativo, como zero, podemos aplicar uma paleta divergente. Uma paleta divergente é essencialmente uma combinação de duas paletas sequenciais com um ponto final compartilhado situado no valor central. Valores maiores que o centro são atribuídos a cores em um lado do centro, enquanto valores menores são atribuídos a cores no lado oposto.



Usada para cada uma das paletas sequenciais componentes para facilitar a distinção entre valores positivos e negativos em relação ao centro. Como acontece com as paletas sequenciais, o valor central geralmente é atribuído a uma cor clara, de modo que as cores mais escuras indicam uma distância maior do centro.

Dicas - Evite o uso desnecessário de cores

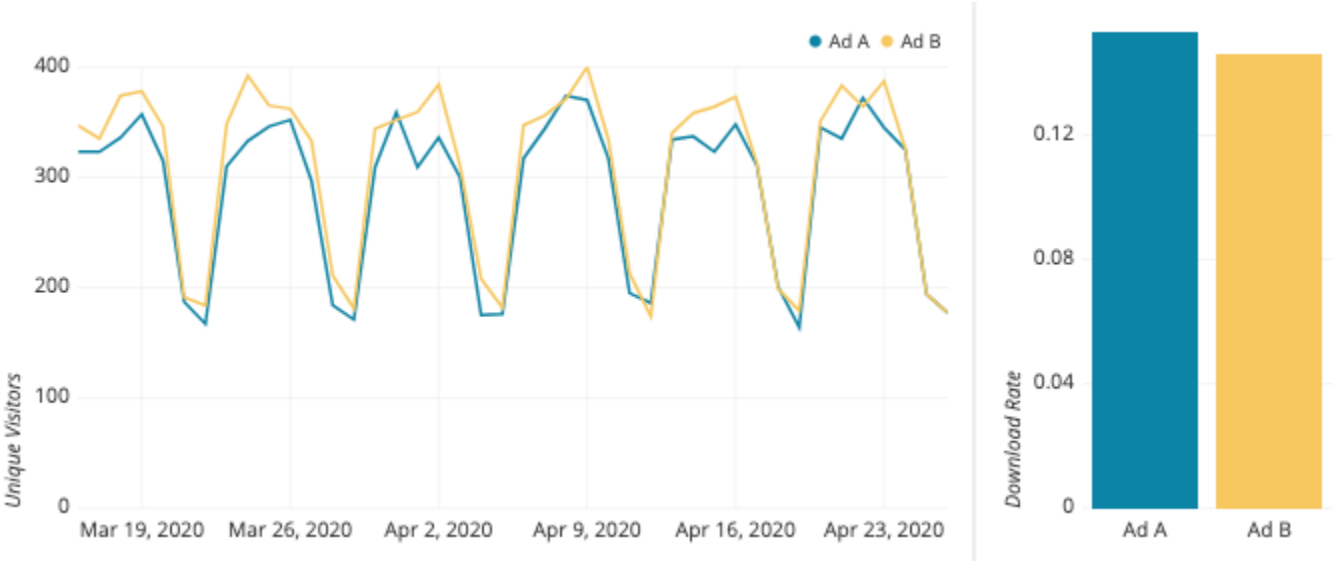
Mesmo que a cor seja uma parte importante da visualização de dados, é aconselhável exercer contenção e usar a cor apenas quando apropriado. Nem todo gráfico que você criar exigirá várias cores. Se você tiver apenas duas variáveis para representar graficamente, elas provavelmente serão codificadas por posições ou comprimentos verticais e horizontais.



As cores da barra do arco-íris à esquerda não são significativas e devem ser evitadas. À direita, a maioria das barras é cinza neutro para destacar a comparação das duas barras coloridas.

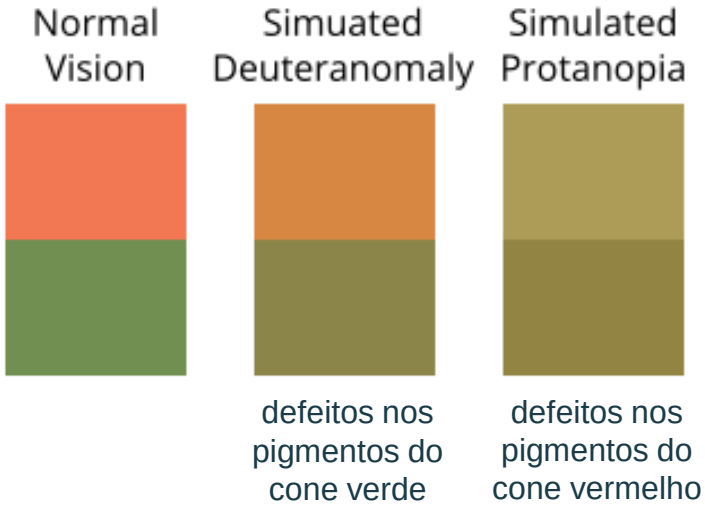
Dicas - Seja consistente com as cores nos gráficos

Se você tiver um painel ou relatório que inclui vários gráficos, é uma boa ideia combinar as cores entre os gráficos quando eles se referem ao mesmo grupo ou entidade. Se as cores mudarem de significado entre os gráficos, isso pode tornar mais difícil para o leitor entender o gráfico.



Dicas - Não se esqueça do daltonismo

As formas mais comuns de daltonismo causam confusão entre certos tons de vermelho e verde, embora também existam formas de daltonismo que fazem com que os tons de azul e amarelo pareçam iguais.



Matplotlib

04

Matplotlib

Instalação:

pip install matplotlib

conda install matplotlib

Importação:

import matplotlib.pyplot as plt

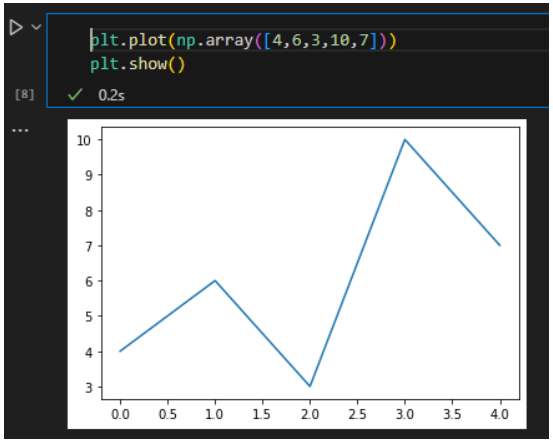
Matplotlib

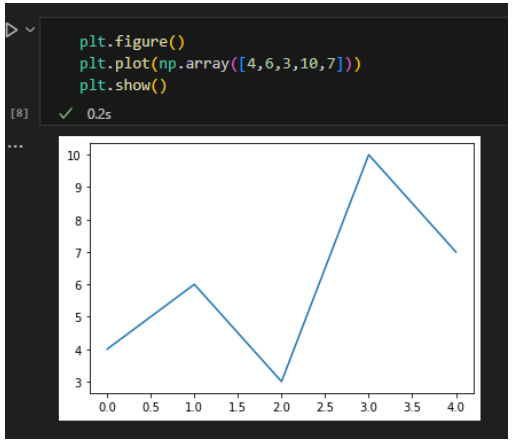
Criando uma figura como área de desenho. Você pode ter várias figuras em uma análise:

plt.figure() //argumento padrão está vazio, mas você usa um número, uma string, etc. como entrada

Plotando os dados usando um tipo de gráfico de sua escolha (gráfico de linhas, gráfico de barras, histograma, gráfico de pizza, etc).

plt.plot(np.array([4,6,3,10,7])) // usa gráfico de linha aqui



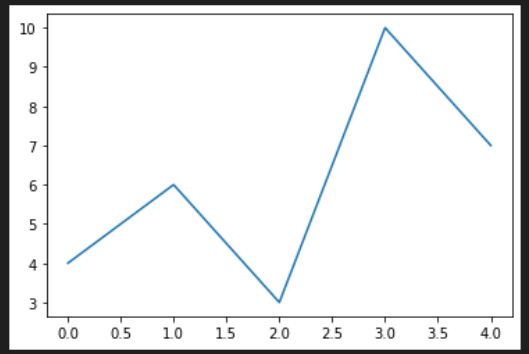


Matplotlib

```
matplotlib.pyplot.figure(num=None, figsize=None, dpi=None, *, facecolor=None,
edgecolor=None, frameon=True, FigureClass=<class 'matplotlib.figure.Figure'>,
clear=False, **kwargs)                                                [source]
```

```
plt.figure()
plt.plot(np.array([4,6,3,10,7]))
plt.show()
```

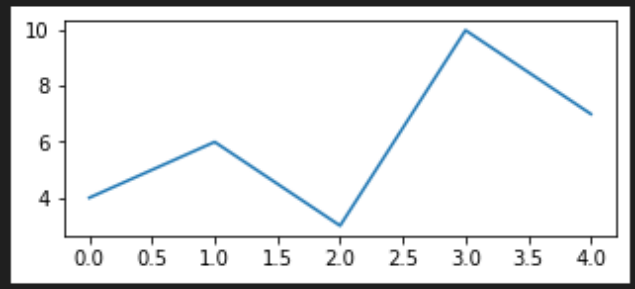
[8] ✓ 0.2s



5 polegadas de largura e 2 de altura

```
plt.figure(figsize=(5,2))
plt.plot(np.array([4,6,3,10,7]))
plt.show()
```

[9] ✓ 0.2s



Matplotlib

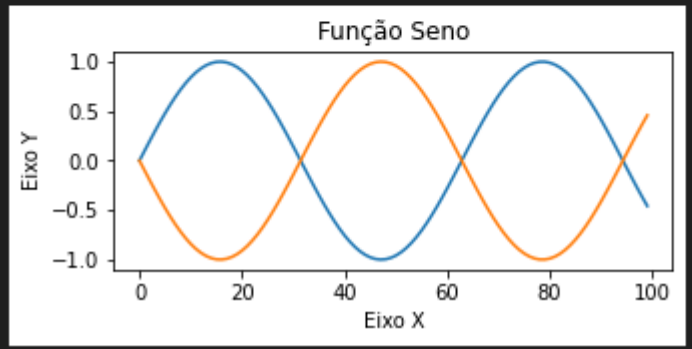
Editando os eixos

plt.xlabel("Eixo X")
plt.ylabel("Eixo Y")

Editando o titulo
plt.title(“Título“)

Adicionando legendas
plt.legend()

```
##  Alterando nomes dos eixos e acrescentando titulo
figura = plt.figure(figsize=(5,2))
#colocando o titulo
plt.title("Função Seno")
#plots de seno
plt.plot(seno)
plt.plot(seno*-1)
#colocando os eixos
plt.xlabel("Eixo X")
plt.ylabel("Eixo Y")
plt.show()
```



Matplotlib

Gráfico de barras: plt.bar()

Gráfico de dispersão: plt.scatter()

Gráfico de linha: plt.plot()

Boxplot: plt.boxplot()

Gráfico de pizza e rosca: plt.pie()

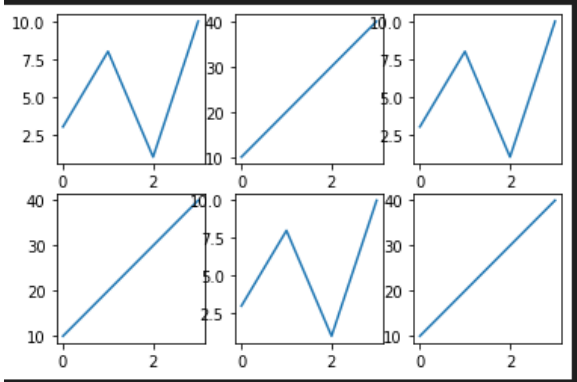
Histograma: plt.hist()

Matplotlib - Subplot

```
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 1)
plt.plot(x,y)

x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 2)
plt.plot(x,y)

x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 3)
plt.plot(x,y)

x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 4)
plt.plot(x,y)
```

```
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 5)
plt.plot(x,y)

x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 6)
plt.plot(x,y)

plt.show()
```

✓ 0.7s



JUPYTER NOTEBOOK
+
ATIVIDADE PRÁTICA

Seaborn

05

Seaborn

Instalação:

pip install seaborn

conda install seaborn

Importação:

import seaborn as sns

Seaborn

Gráfico de barras: seaborn.barplot() e seaborn.countplot()

Gráfico de dispersão e gráfico de bolhas: seaborn.scatterplot()

Gráfico de linha: seaborn.lineplot()

Boxplot: seaborn.boxplot()

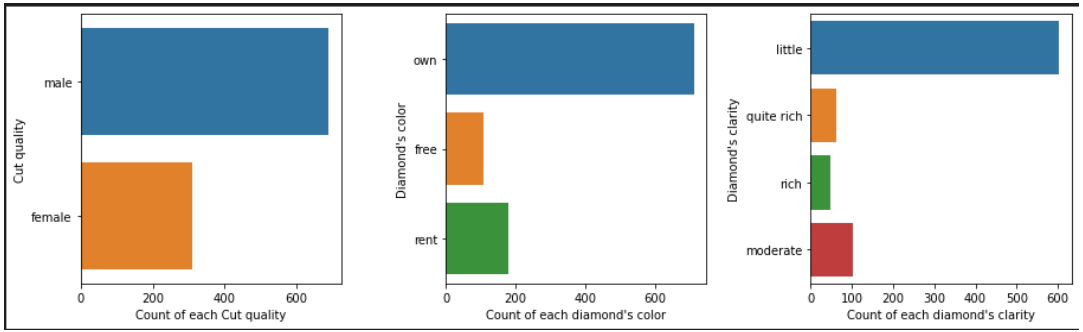
Gráfico de pizza e rosca: plt.pie()

Heatmap: seaborn.heatmap()

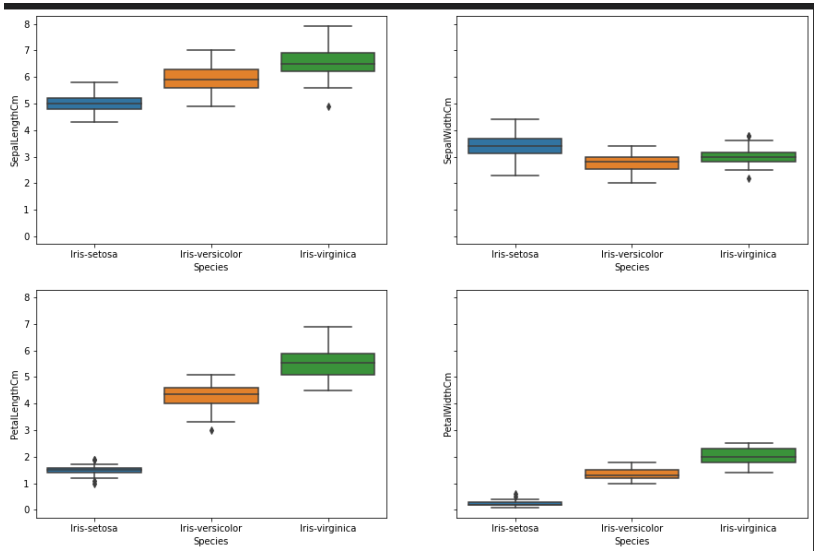
Histograma: seaborn.histplot()

Seaborn - Subplot

```
fig = plt.figure(figsize=(15,10))
fig.subplots_adjust(hspace=0.4, wspace=0.4)

plot1 = plt.subplot(2, 3, 1)
sns.countplot(y=df.Sex ,data=df)
plt.xlabel("Count of each Cut quality")
plt.ylabel("Cut quality")

plot2 = plt.subplot(2, 3, 2)
sns.countplot(y=df.Housing ,data=df)
plt.xlabel("Count of each diamond's color")
plt.ylabel("Diamond's color")

plot3 = plt.subplot(2, 3, 3)
sns.countplot(y=df['Saving accounts'] ,data=df)
plt.xlabel("Count of each diamond's clarity")
plt.ylabel("Diamond's clarity")
plt.show()
```



Seaborn - Subplot




```
fig, ax = plt.subplots(2, 2, figsize=(15,10), sharey=True)


sns.boxplot(x='Species',y='SepalLengthCm',data=df_iris, ax=ax[0][0])
sns.boxplot(x='Species',y='SepalWidthCm',data=df_iris, ax=ax[0][1])
sns.boxplot(x='Species',y='PetalLengthCm',data=df_iris, ax=ax[1][0])
sns.boxplot(x='Species',y='PetalWidthCm',data=df_iris, ax=ax[1][1])
```

JUPYTER NOTEBOOK
+
ATIVIDADE PRÁTICA

Plotly

06

Plotly

A biblioteca plotly Python é uma biblioteca de plotagem interativa e de código aberto que oferece suporte a mais de 40 tipos de gráficos exclusivos, abrangendo uma ampla variedade de casos de uso estatísticos, financeiros, geográficos, científicos e tridimensionais.

Construído sobre a biblioteca Plotly JavaScript (plotly.js), o plotly permite que os usuários do Python criem belas visualizações interativas baseadas na web que podem ser exibidas em notebooks Jupyter, salvas em arquivos HTML independentes ou servidas como parte da aplicação web construída em Python usando o Dash. A biblioteca plotly Python às vezes é chamada de "plotly.py" para diferenciá-la da biblioteca JavaScript.

Plotly

Instalação:

pip install plotly

conda install -c plotly plotly

Importação:

```
import plotly.offline as pyo
import plotly.graph_objs as go
import plotly.express as px
```

Plotly

Criando uma figura como área de desenho. Você pode ter várias figuras em uma análise:

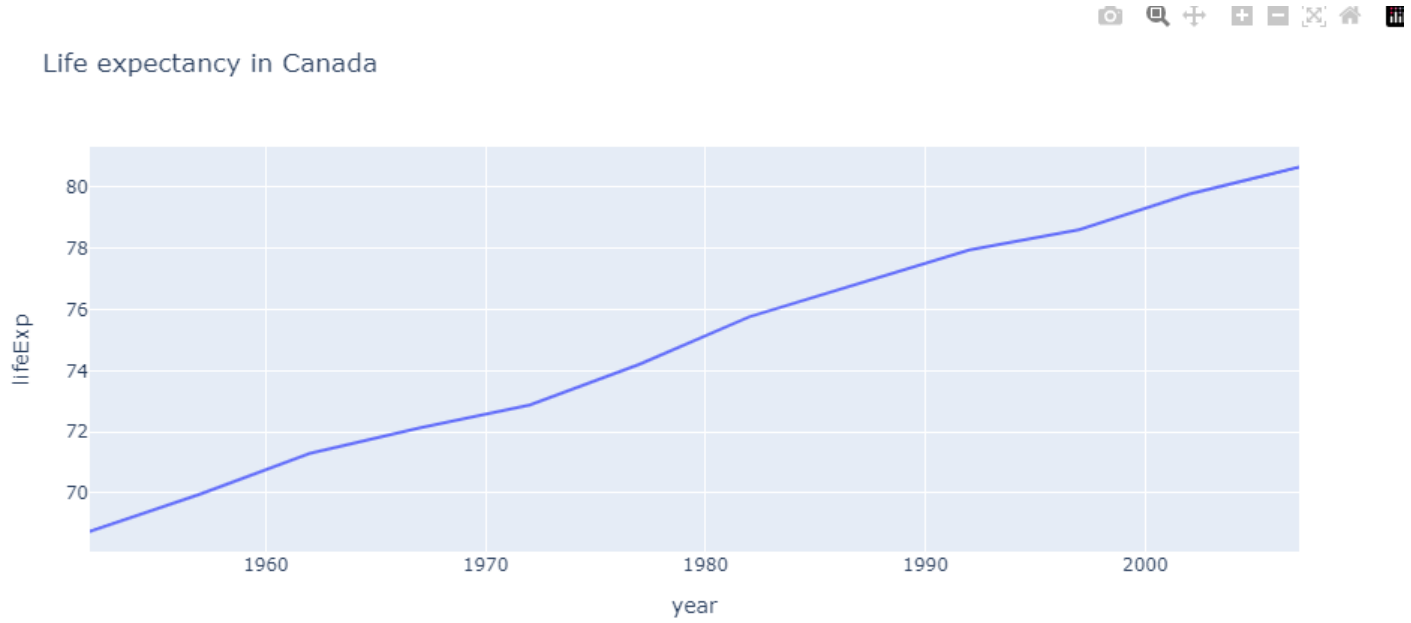
```
fig = go.Figure(data=data, layout=layout)
```

Plotando os dados usando um tipo de gráfico de sua escolha (gráfico de linhas, gráfico de barras, histograma, gráfico de pizza, etc).

```
fig = px.line(df.query("country=='Canada'"), x="year", y="lifeExp", title='Life expectancy in Canada')

trace2 = go.Scatter(
    x = df_canada['year'],
    y = df_canada['lifeExp'],
    mode = 'lines',
    name = 'lines'
)
```

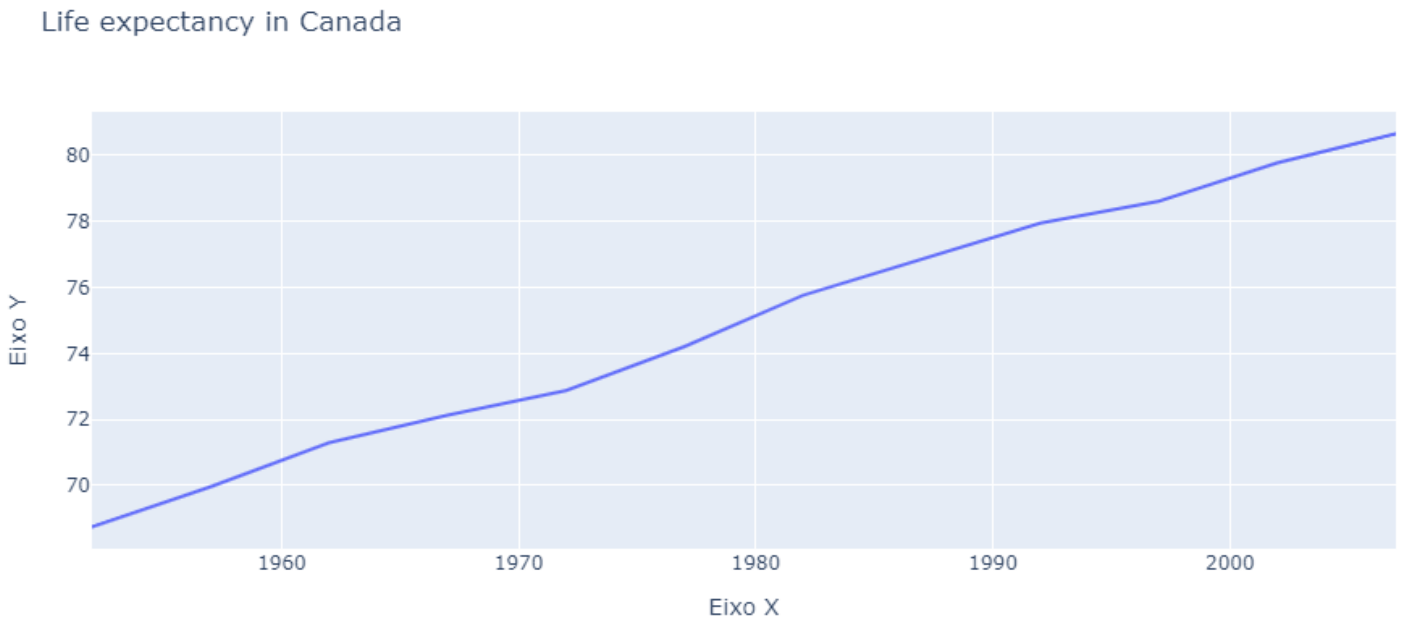
Plotly

```
fig = px.line(df.query("country=='Canada'"), x="year", y="lifeExp", title='Life expectancy in Canada')
```



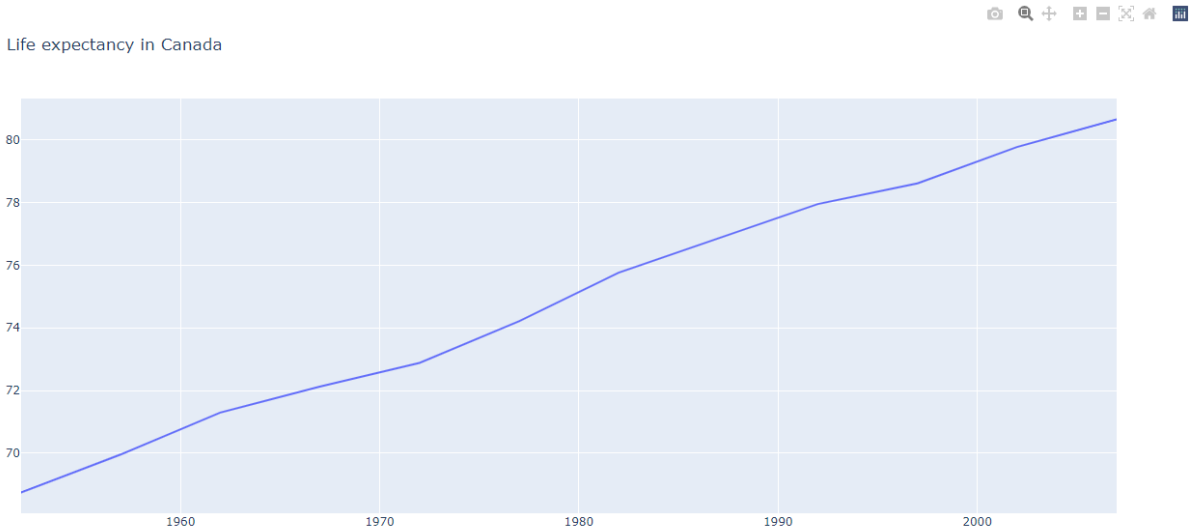
Plotly

```
fig = px.line(df.query("country=='Canada'"), x="year", y="lifeExp", title='Life expectancy in Canada')
fig.update_layout(
    xaxis_title="Eixo X", yaxis_title="Eixo Y"
 )
```



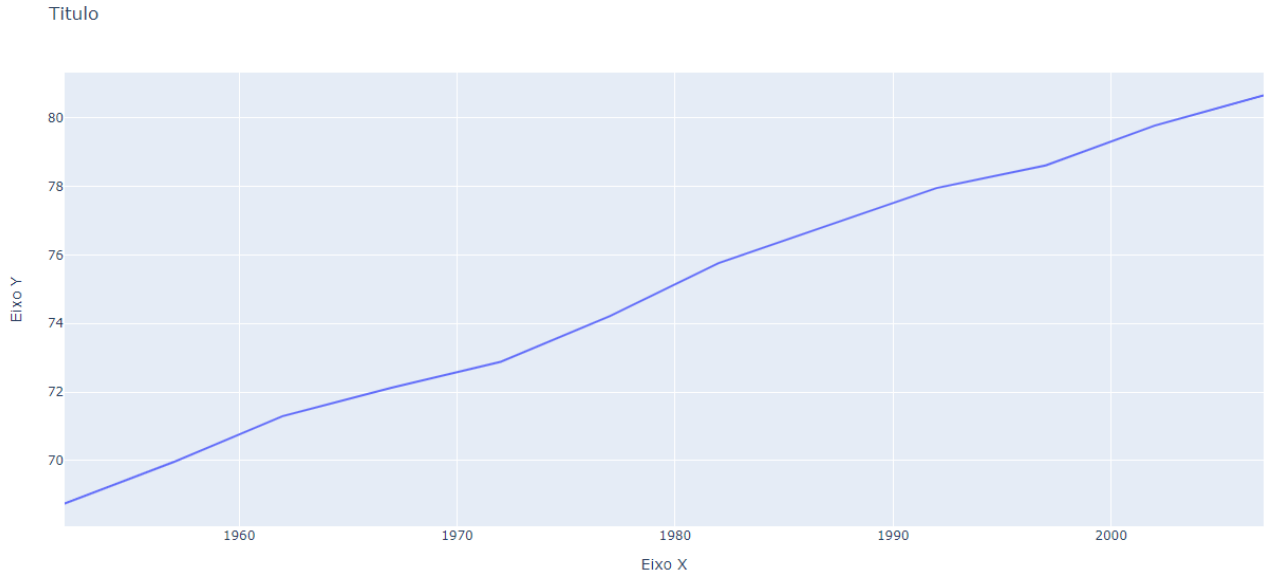
Plotly

```
trace2 = go.Scatter(
    x = df_canada['year'],
    y = df_canada['lifeExp'],
    mode = 'lines',
    name = 'lines'
)
layout = go.Layout(
    title = 'Life expectancy in Canada'
)
data = [trace2]
fig = go.Figure(data=data,layout=layout)
pyo.plot(fig, filename='line1.html')
```



Plotly

```
trace2 = go.Scatter(
    x = df_canada['year'],
    y = df_canada['lifeExp'],
    mode = 'lines',
    name = 'lines'
)
layout = go.Layout(
    title = 'Titulo',
    xaxis = dict(
        title = 'Eixo X'
    ),
    yaxis = dict(
        title = 'Eixo Y'
    )
)
data = [trace2]
fig = go.Figure(data=data,layout=layout)
pyo.iplot(fig)
```



Plotly

Gráfico de barras: px.bar() ou go.bar()

Gráfico de dispersão e gráfico de bolhas: px.scatter() ou go.Scatter()

Gráfico de linha: px.line() ou go.Scatter(mode = 'lines')

Boxplot: px.box() ou go.Box()

Gráfico de pizza e rosca: px.pie() e px.pie(hole = 0.4) ou go.Pie() e go.Pie(hole = 0.4)

Heatmap: px.imshow() ou go.Heatmap()

Histograma: px.histogram() ou go.Histogram()

Plotly - Subplot

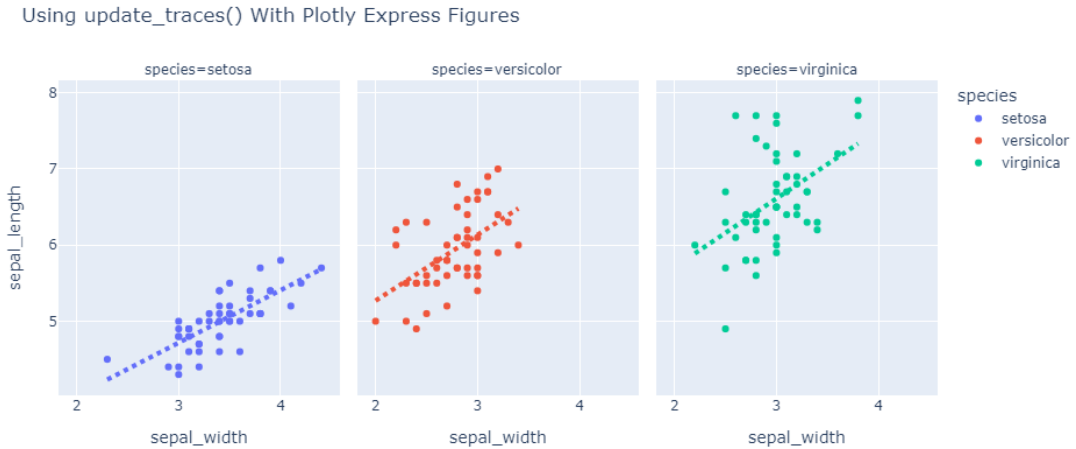
```
from plotly.subplots import make_subplots

fig = make_subplots(rows=3, cols=1)

fig.add_trace(
    go.Scatter(x=[1, 2, 3], y=[4, 5, 6]),
    row=1, col=1
)

fig.add_trace(
    go.Scatter(x=[20, 30, 40], y=[50, 60, 70]),
    row=2, col=1
)

fig.add_trace(
    go.Scatter(x=[30, 40, 50], y=[60,100,200]),
    row=3, col=1
)

fig.update_layout(height=600, width=600, title_text="Stacked Subplots")
```

✓ 0.0s

Plotly - Subplot

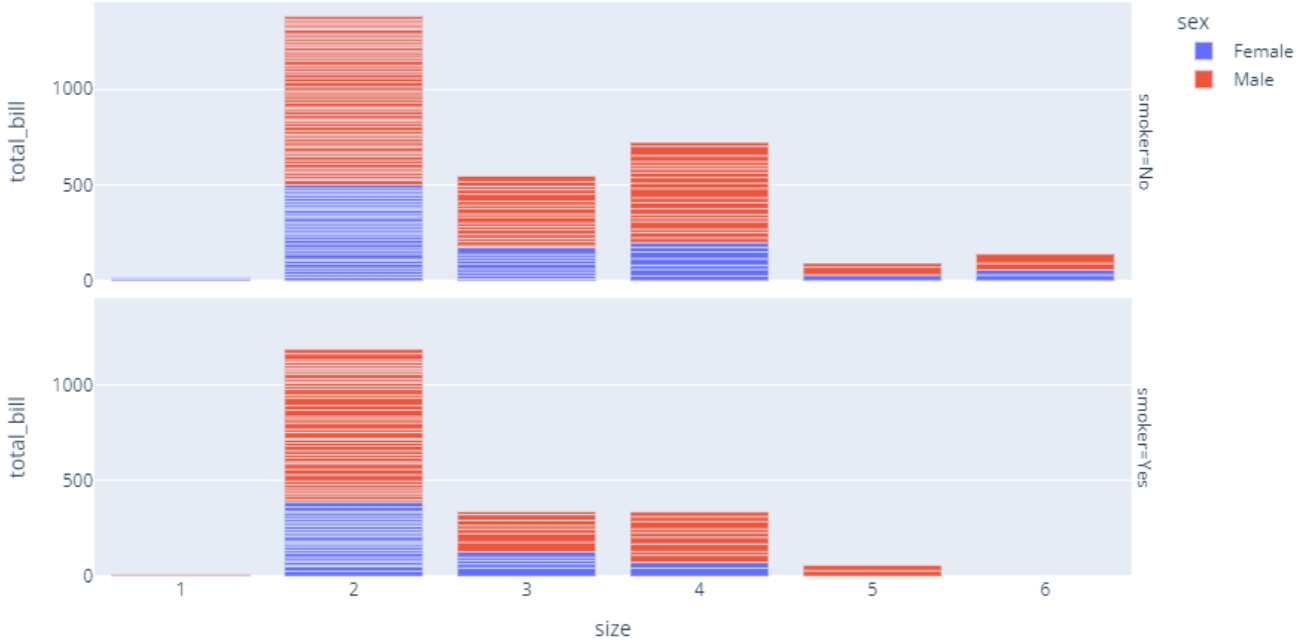
```
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species",
                 facet_col="species",  trendline="ols", title="Using update_traces() With Plotly Express Figures")
                 # facet_cols separa os gráficos de acordo com a coluna específica.
                 # trendline = "ols" indica a função de regressão a ser utilizad.


fig.update_traces(
    line=dict(dash="dot", width=4) # faz al inha tracejada
    )

fig.show()
```

✓ 0.1s



Plotly - Subplot

```
import plotly.express as px
df = px.data.tips()
fig = px.bar(df, x="size", y="total_bill", color="sex", facet_row="smoker")
fig.show()
```

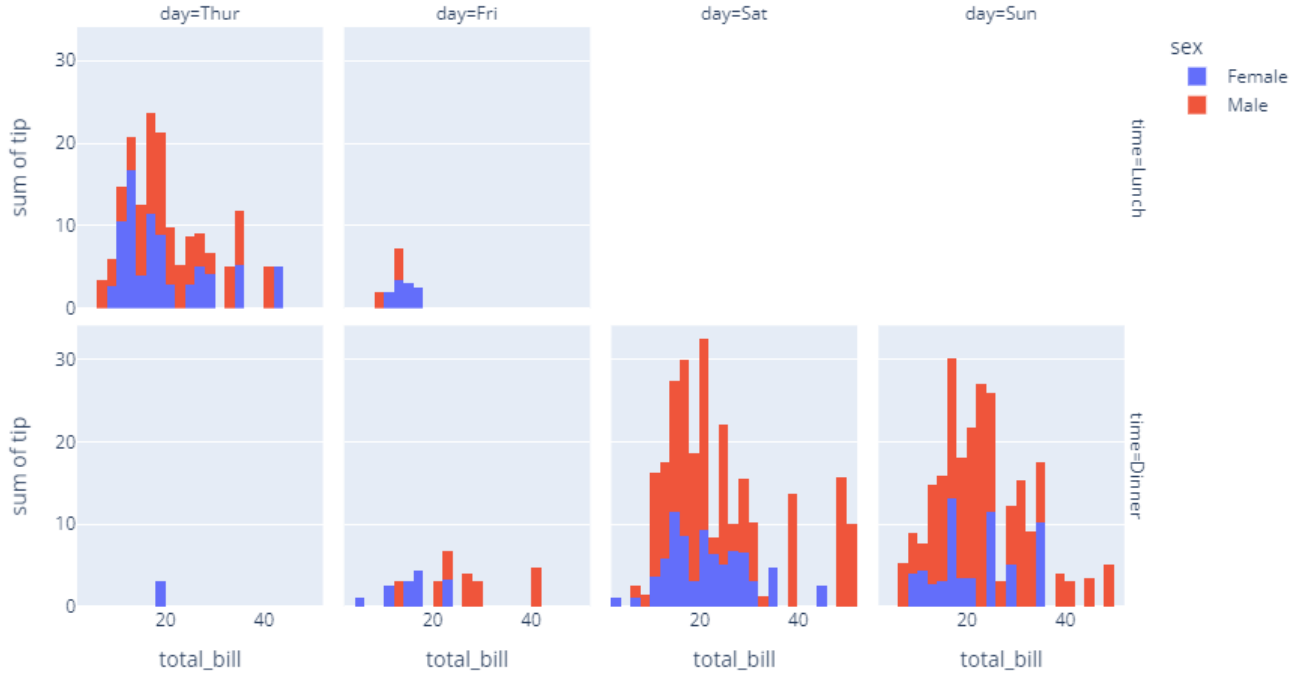


Plotly - Subplot

```
import plotly.express as px
df = px.data.gapminder()
fig = px.scatter(df, x='gdpPercap', y='lifeExp', color='continent', size='pop',
                 facet_col='year', facet_col_wrap=4)
fig.show()
```

Plotly - Subplot

```
import plotly.express as px
df = px.data.tips()
fig = px.histogram(df, x="total_bill", y="tip", color="sex", facet_row="time", facet_col="day",
       category_orders={"day": ["Thur", "Fri", "Sat", "Sun"], "time": ["Lunch", "Dinner"]})
fig.show()
```



JUPYTER NOTEBOOK
+
ATIVIDADE PRÁTICA

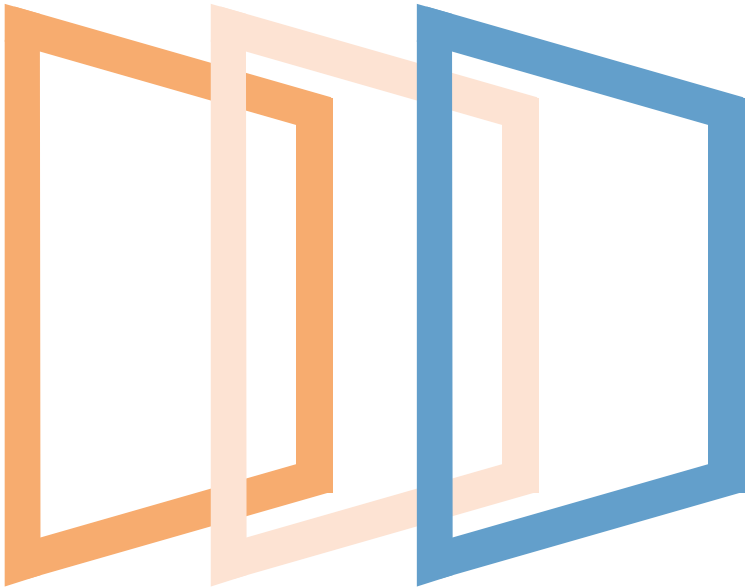
Referências

- Como escolher as cores para suas visualizações de dados (ichi.pro)
- Box plot: o que é e para que serve? – OPENCADD
- AULA_08.pptx (live.com)
- Scatter plot: Um Guia Completo para Gráficos de Dispersão – FLAI
- Matplotlib Subplot (w3schools.com)
- Faça um gráfico de pizza em Python usando Matplotlib – Acervo Lima
- Donut Chart using Matplotlib in Python – GeeksforGeeks
- web.stanford.edu/class/archive/cs/cs106a/cs106a.1228/lectures/22-matplotlib/slides
- Matplotlib.pdf (ohio-state.edu)

Referências

- API reference — seaborn 0.12.2 documentation (pydata.org)
- Plotly Python Graphing Library
- Interactive Python Dashboards with Plotly and Dash (udemy.com)
- https://stack-academy.memberkit.com.br/32408-data-science-do-zero

Análise Gráfica com Python



Thaís Ratis

Inteligência Artificial Brasil, 15.05.2024

minsait

An Indra company