

## Introdução às Bibliotecas de Visualização em Python

Matplotlib e Seaborn são duas das bibliotecas de visualização de dados mais comumente usadas em Python. Fornecem uma ampla gama de funcionalidades para criar uma variedade de gráficos 2D e 3D. **Matplotlib** é uma biblioteca de baixo nível que fornece muita flexibilidade e controlo sobre os detalhes do gráfico, enquanto **Seaborn** é uma biblioteca de nível superior que torna mais fácil criar gráficos estatísticos atraentes e informativos.

Além de Matplotlib e Seaborn, **Plotly** e **Bokeh** também são bibliotecas populares para criar visualizações interativas. Plotly é particularmente adequado para criar gráficos e painéis interativos, enquanto Bokeh é projetado para criar visualizações interativas para navegadores da web modernos.


A escolha da biblioteca depende das necessidades específicas do projeto e dos tipos de visualizações necessárias. Algumas bibliotecas podem ser mais adequadas para determinados tipos de dados ou visualizações, e algumas podem ser mais fáceis de usar do que outras, dependendo do nível de experiência do utilizador com Python e visualização de dados.

### ✓ Visualização de Dados com Seaborn (em Python)

#### O que é o Seaborn?

O **Seaborn** é uma biblioteca de visualização de dados baseada no Matplotlib que fornece uma interface de alto nível para a criação de gráficos estatísticos informativos e esteticamente agradáveis.

- Facilita a criação de **gráficos complexos** com poucas linhas de código.
- Está profundamente integrada com o `pandas`, permitindo trabalhar com `DataFrames` de forma eficiente.
- Suporta **gráficos estatísticos**, como gráficos de regressão, distribuição, boxplots, mapas de calor, entre outros.

 Documentação oficial: <https://seaborn.pydata.org>

### ✓ O Dataset Penguins

Neste notebook vamos utilizar o dataset **Palmer Penguins**, incluído no Seaborn.

Origem:

O conjunto de dados foi recolhido nas Ilhas Palmer (Antártica), e contém observações de três espécies de pinguins:

- **Adelie**
- **Gentoo**
- **Chinstrap**

Estrutura do Dataset:

Coluna	Descrição
species	Espécie do pinguim
island	Ilha onde foi observado
bill_length_mm	Comprimento do bico (em milímetros)
bill_depth_mm	Profundidade do bico (em milímetros)
flipper_length_mm	Comprimento da barbatana (em milímetros)
body_mass_g	Massa corporal (em gramas)
sex	Sexo do pinguim

Este dataset é ideal para explorar relações estatísticas entre variáveis numéricas e categóricas.

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Carregamento do dataset diretamente da biblioteca seaborn
df = sns.load_dataset("penguins")

# Remoção de registos com valores em falta (NaN)
df.dropna(inplace=True)

# Primeiras 5 linhas do conjunto de dados
df.head()
```

## ✓ Visualização de Dados com Seaborn (em Python)

### O que é o Seaborn?

O **Seaborn** é uma biblioteca de visualização de dados baseada no Matplotlib que fornece uma interface de alto nível para a criação de gráficos estatísticos informativos e esteticamente agradáveis.

- Facilita a criação de **gráficos complexos** com poucas linhas de código.
- Está profundamente integrada com o `pandas`, permitindo trabalhar com `DataFrames` de forma eficiente.
- Suporta **gráficos estatísticos**, como gráficos de regressão, distribuição, boxplots, mapas de calor, entre outros.

 Documentação oficial: <https://seaborn.pydata.org>

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Carregar um dataset limpo: 'penguins'
df = sns.load_dataset("penguins")
df.dropna(inplace=True) # Remover valores nulos
df.head()
```

### Estética e Estilo dos Gráficos com Seaborn

O Seaborn permite aplicar estilos e cores predefinidos para tornar os gráficos mais apelativos.

```
sns.set_style("whitegrid") # Outros estilos: darkgrid, white, ticks, dark
sns.set_palette("pastel") # Outros: deep, dark, colorblind, bright
```

Estes comandos podem ser aplicados no início do notebook para afetar todos os gráficos seguintes.

## ✓ Gráficos de Dispersão (`scatterplot`)

Usado para visualizar a relação entre duas variáveis numéricas.

```
# Criamos um gráfico de dispersão com as variáveis 'flipper_length_mm' no eixo X e 'body_mass_g' no eixo Y. Os pontos são coloridos por
sns.scatterplot(data=df, x="flipper_length_mm", y="body_mass_g", hue="species")
# Adicionamos um título ao gráfico.
plt.title("Comprimento da Barbatana vs Massa Corporal")
# Definimos o rótulo do eixo X.
plt.xlabel("Comprimento da Barbatana (mm)")
# Definimos o rótulo do eixo Y.
plt.ylabel("Massa Corporal (g)")
# Mostramos o gráfico.
plt.show()
```

## ✓ Gráficos de Linhas (`lineplot`)

Útil para dados sequenciais ou variações médias.

```
# Explicação:
# - `sns.lineplot()` cria um gráfico de linhas.
# - Ideal para mostrar tendências, mesmo em dados não temporais.
# - Representamos a massa corporal média em função do comprimento da barbatana, agrupado por espécie.

sns.lineplot(data=df, x="flipper_length_mm", y="body_mass_g", hue="species")
plt.title("Tendência do Comprimento da Barbatana e Massa Corporal")
plt.xlabel("Comprimento da Barbatana (mm)")
plt.ylabel("Massa Corporal (g)")
plt.show()
```

## ✓ Histogramas e Densidade (`histplot` e `kdeplot`)

Visualizam a distribuição de uma variável contínua.

```
# Criamos um histograma da variável 'flipper_length_mm', com curva de densidade (kde). Os dados são separados por espécie.
sns.histplot(data=df, x="flipper_length_mm", kde=True, hue="species", multiple="stack")
```

```
# Adicionamos um título ao gráfico.
plt.title("Distribuição do Comprimento da Barbatana por Espécie")
# Definimos o rótulo do eixo X.
plt.xlabel("Comprimento da Barbatana (mm)")
# Definimos o rótulo do eixo Y.
plt.ylabel("Frequência")
# Mostramos o gráfico.
plt.show()
```

## ✓ 📊 Gráficos de Caixa e Violino (boxplot e violinplot)

Úteis para visualizar distribuição, mediana e outliers.

```
# Criamos um gráfico de caixa (boxplot) com as espécies no eixo X e a massa corporal no eixo Y.
sns.boxplot(data=df, x="species", y="body_mass_g")
# Adicionamos um título ao gráfico.
plt.title("Boxplot da Massa Corporal por Espécie")
# Definimos o rótulo do eixo X.
plt.xlabel("Espécie")
# Definimos o rótulo do eixo Y.
plt.ylabel("Massa Corporal (g)")
# Mostramos o gráfico.
plt.show()
```

## ✓ 📊 Gráficos de Barras (barplot)

Mostra estatísticas agregadas (por padrão a média).

```
# Criamos um gráfico de barras com a massa corporal média por ilha.
sns.barplot(data=df, x="island", y="body_mass_g")
# Adicionamos o título ao gráfico.
plt.title("Massa Corporal Média por Ilha")
# Definimos o rótulo do eixo X.
plt.xlabel("Ilha")
# Definimos o rótulo do eixo Y.
plt.ylabel("Massa Corporal (g)")
# Mostramos o gráfico.
plt.show()
```

## ✓ 📊 Mapas de Calor (heatmap)

Visualização de correlações ou matrizes de dados.

```
# Calculamos a matriz de correlação entre as colunas numéricas do dataset.
correlacoes = df.corr(numeric_only=True)
# Criamos o mapa de calor (heatmap), com os valores anotados e uma paleta de cores 'coolwarm'.
sns.heatmap(correlacoes, annot=True, cmap="coolwarm")
# Adicionamos o título ao gráfico.
plt.title("Mapa de Calor das Correlações")
# Mostramos o gráfico.
plt.show()
```

## ✓ 📊 Gráficos de Pares (pairplot)

Permite visualizar todas as combinações entre múltiplas variáveis numéricas.

```
# Criamos uma matriz de gráficos de dispersão entre todas as variáveis numéricas, diferenciando por espécie.
sns.pairplot(df, hue="species")
# Adicionamos um título geral à figura com deslocamento vertical.
plt.suptitle("Gráficos de Pares das Variáveis Numéricas", y=1.02)
# Mostramos o gráfico.
plt.show()
```

## ✓ 📊 Controlo da Estética com Seaborn

O Seaborn permite personalizar a aparência geral dos gráficos através de quatro componentes principais:

1. **Estilos de fundo (set\_style)** – Define o fundo do gráfico.
2. **Paletas de cores (set\_palette)** – Define as cores dos elementos.

3. **Contexto visual** (`set_context`) – Adapta o tamanho dos elementos para diferentes usos.
4. **Tamanho da figura** (`plt.figure(figsize=(x, y))`) – Define o tamanho da imagem final.

## 1 Estilos de Fundo

Os estilos disponíveis são:

- "white"
- "dark"
- "whitegrid" (útil para dados com grelhas)
- "darkgrid"
- "ticks" (com marcas subtis nos eixos)

```
sns.set_style("whitegrid")
```

## 2 Paletas de Cores

Paletas predefinidas para uma estética mais consistente:

- "deep" (predefinido)
- "muted"
- "pastel"
- "bright"
- "dark"
- "colorblind"

```
sns.set_palette("pastel")
```

## 3 Contexto Visual

Ajusta automaticamente o tamanho dos elementos visuais (fontes, pontos, linhas):

- "paper" (ideal para impressão)
- "notebook" (predefinido)
- "talk" (para apresentações)
- "poster" (para gráficos grandes)

```
sns.set_context("talk")
```


## 4 Exemplo Combinado

Aplicação conjunta de estilo, paleta e contexto:

```
# Criamos um gráfico de caixa (boxplot) com as espécies no eixo X e a massa corporal no eixo Y.
sns.boxplot(data=df, x="species", y="body_mass_g")
# Adicionamos um título ao gráfico.
plt.title("Boxplot da Massa Corporal por Espécie")
# Definimos o rótulo do eixo X.
plt.xlabel("Espécie")
# Definimos o rótulo do eixo Y.
plt.ylabel("Massa Corporal (g)")
# Mostramos o gráfico.
plt.show()
```

## Lista de Exercícios com Seaborn

1. Cria um gráfico de dispersão entre o comprimento e a massa corporal, diferenciando as espécies com cores.
2. Representa a distribuição do comprimento da barbatana com `histplot`, incluindo `kde=True`.
3. Compara a massa corporal entre espécies com um `boxplot`.
4. Cria um `violinplot` para a variável `flipper_length_mm`.
5. Constrói um `heatmap` das correlações entre as variáveis numéricas.
6. Usa o `pairplot` para estudar a relação entre todas as variáveis numéricas do dataset.
7. Cria um `barplot` que mostre a média da massa corporal por ilha.

 Utiliza apenas `seaborn` para resolver os exercícios.