Explorando o Seaborn em Detalhes

Prof. Dr. Fabiano B. Menegidio

1 Introdução ao Seaborn

O **Seaborn** é uma biblioteca de visualização de dados construída sobre o **Matplotlib** e integrada ao **Pandas**. Seu objetivo é simplificar a criação de gráficos estatísticos com menos código e fornecer uma estética mais agradável para visualizações. Além disso, o Seaborn oferece suporte direto para trabalhar com estruturas de dados como *dataframes* do Pandas, tornando o processo de visualização mais intuitivo.

1.1 Recursos Principais do Seaborn

- **Estatísticas e Visualizações**: Gráficos prontos para análise de dados estatísticos, como *boxplots*, *violin plots*, *heatmaps*, e muito mais.
- **Integração com Pandas**: O Seaborn trabalha diretamente com *dataframes*, o que facilita a visualização de dados.
- **Visualizações de Regressão**: Suporte direto para adicionar linhas de tendência e regressão.
- **Facetas e Subplots Automáticos**: Fácil criação de gráficos múltiplos com base em diferentes categorias ou classes de dados.

2 Instalação do Seaborn

Se você ainda não tem o Seaborn instalado, pode instalá-lo com o seguinte comando:

```
pip install seaborn
```

3 Carregando Dados

O Seaborn possui alguns datasets embutidos que podem ser usados para testes e exemplos. Vamos começar carregando o dataset *titanic*, que é frequentemente utilizado.

```
import seaborn as sns

# Carregar um dataset embutido no Seaborn
df = sns.load_dataset('titanic')

# Ver as primeiras 5 linhas
print(df.head())
```

Agora, vamos explorar algumas funcionalidades do Seaborn com exemplos e explicações detalhadas.

4 Configurações de Estilo

O Seaborn fornece estilos prontos que podem ser aplicados aos gráficos para torná-los mais bonitos. Vamos ajustar o estilo global dos gráficos.

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

# Ajustar o estilo dos gr ficos
sns.set_style("whitegrid")

# Exemplo de um gr fico simples ap s ajustar o estilo
sns.countplot(x='class', data=df)
plt.show()
```

Explicação:

- `sns.set_style()` permite alterar o estilo dos gráficos. Aqui, usamos `"whitegrid"`, que adiciona uma grade branca ao f
  Outros estilos disponíveis incluem: `darkgrid`, `dark`, `white`, e `ticks`.

5 Gráficos de Distribuição

O Seaborn facilita a criação de gráficos que mostram a **distribuição** de variáveis numéricas.

5.1 Histogramas e KDE (Kernel Density Estimation)

```
# Histograma e gr fico de densidade da idade dos passageiros
sns.histplot(df['age'], kde=True)
plt.show()
```

Explicação:

- `histplot` cria um histograma que mostra a distribuição da variável `age`. O parâmetro `kde=True` adiciona uma curva de densidade para suavizar a distribuição e destacar os padrões.

5.2 Gráfico de Densidade (KDE)

```
# Apenas o gr fico de densidade (KDE)
sns.kdeplot(df['age'], shade=True)
plt.show()
```

Explicação:

- `kdeplot` exibe um gráfico de densidade baseado na distribuição dos valores da variável `age`. O parâmetro `shade=True` adiciona uma sombra abaixo da curva.

6 Gráficos de Dispersão (Scatter Plots)

Gráficos de dispersão são úteis para visualizar a relação entre duas variáveis numéricas.

```
# Gr fico de dispers o da idade vs. tarifa (age vs. fare)
sns.scatterplot(x='age', y='fare', data=df)
plt.show()
```

Explicação:

- `scatterplot` cria um gráfico de dispersão que mostra a relação entre `age` (idade) e `fare` (tarifa paga).

7 Gráficos de Regressão

O Seaborn facilita a visualização de uma regressão linear entre duas variáveis com uma linha de tendência.

```
# Gr fico de dispers o com linha de regress o
sns.lmplot(x='age', y='fare', data=df)
plt.show()
```

Explicação:

- `lmplot` cria um gráfico de dispersão com uma linha de regressão linear ajustada. Ele também desenha um intervalo de confiança ao redor da linha.

8 Gráficos Categóricos

8.1 Gráfico de Contagem (Countplot)

```
# Gr fico de contagem da classe dos passageiros
sns.countplot(x='class', data=df)
plt.show()
```

Explicação:

- `countplot` exibe a contagem de ocorrências em cada classe de passageiros (`class`). Este é um gráfico útil para dados categóricos.

8.2 Boxplot

O `boxplot` é uma excelente ferramenta para visualizar a distribuição de uma variável e identificar valores atípicos.

```
# Boxplot para a distribui  o de tarifas pagas em cada classe
sns.boxplot(x='class', y='fare', data=df)
plt.show()
```

Explicação:

- `boxplot` cria caixas que representam a distribuição da tarifa (`fare`) para cada classe de passageiros (`class`). As linhas no meio das caixas representam a mediana, e os "bigodes" (*whiskers*) indicam o intervalo de confiança.

8.3 Violin Plot

Um `violinplot` combina o boxplot com uma visualização da distribuição de densidade.

```
# Violin plot para distribui  o de tarifas em cada classe
sns.violinplot(x='class', y='fare', data=df)
plt.show()
```

Explicação:

- `violinplot` mostra a distribuição das tarifas (`fare`) de cada classe (`class`) com uma representação suavizada de densidade ao redor do gráfico de caixa.

9 Heatmaps (Mapas de Calor)

O `heatmap` é uma excelente forma de visualizar correlações entre variáveis numéricas.

```
# Criar uma matriz de correla  o entre vari veis num ricas
corr_matrix = df.corr()

# Exibir a matriz de correla  o como um heatmap
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.show()
```

Explicação:

- `corr()` calcula a matriz de correlação entre as variáveis numéricas do dataset.
- `heatmap` visualiza a matriz de correlação, onde as cores indicam a força da correlação entre as variáveis. O parâmetro `annot=True` exibe os valores numéricos dentro das células.

10 FacetGrid

O `FacetGrid` permite criar múltiplos gráficos com base em diferentes subgrupos dos dados.

```
# Criar facetas de gr ficos de dispers o por sexo
g = sns.FacetGrid(df, col="sex")
g.map(sns.scatterplot, "age", "fare")
plt.show()
```

Explicação:

- O `FacetGrid` cria gráficos separados para cada valor único na coluna `sex` (masculino e feminino). O método `map()` aplica um gráfico de dispersão para cada faceta.

11 Pairplot

O `pairplot` é uma forma eficiente de visualizar todas as combinações possíveis de gráficos de dispersão entre várias variáveis.

```
# Criar pairplot das vari veis num ricas selecionadas
sns.pairplot(df[['age', 'fare', 'pclass']])
plt.show()
```

Explicação:

- `pairplot` cria gráficos de dispersão para todas as combinações possíveis entre as variáveis numéricas selecionadas (`age`, `fare` e `pclass`).

12 Conclusão

O **Seaborn** é uma poderosa biblioteca para a criação de visualizações estatísticas em Python. Ele é integrado diretamente com os *dataframes* do Pandas, tornando o processo de visualização de dados mais fluido e simplificado. Além disso, seus gráficos possuem uma estética agradável e suporte nativo para técnicas estatísticas como visualizações de regressão e distribuição de dados. O uso do Seaborn em projetos de análise de dados pode ser extremamente vantajoso, pois permite a criação de gráficos sofisticados com poucas linhas de código.