Fondamenti SeaBorn

Python

Introduzione a Seaborn

Seaborn è una libreria Python per la visualizzazione di dati basata su Matplotlib.

Seaborn semplifica la creazione di grafici statistici eleganti e complessi con meno codice e include stili predefiniti e palette di colori che migliorano l'aspetto dei grafici rispetto a quelli creati con Matplotlib da solo.

Seaborn è particolarmente utile quando si lavora con dati strutturati, come DataFrame di Pandas, e offre diverse funzionalità avanzate che rendono più facile visualizzare relazioni, tendenze e distribuzioni.

Principali Funzionalità di Seaborn

Grafici statistici complessi: Seaborn consente di creare grafici statistici come violin plot, box plot, heatmap, pairplot e molti altri con pochissimo codice.

Integrazione con Pandas: Seaborn lavora perfettamente con strutture di dati Pandas, come DataFrame, semplificando la visualizzazione dei dati.

Stili e Temi predefiniti: Include stili di grafici eleganti e professionali, migliorando l'aspetto visivo delle visualizzazioni.

Palette di colori: Fornisce diverse palette di colori che possono essere facilmente applicate ai grafici per un'estetica migliorata e coerente.

Gestione delle variabili categoriali: Seaborn facilita la visualizzazione di variabili categoriali, con grafici come catplot, boxplot e barplot.

Elementi Chiave di Seaborn

1. Grafici di Distribuzione

Seaborn offre diverse opzioni per visualizzare la distribuzione di un set di dati. Tra queste ci sono il histplot (istogramma), kdeplot (stima della densità del kernel) e jointplot (combinazione di distribuzione e grafici a dispersione).

2. Grafici per Variabili Categoriali

Per le variabili categoriali, Seaborn offre diversi grafici come il barplot, countplot, boxplot e violinplot. Questi grafici sono ottimi per esplorare come le diverse categorie si confrontano tra loro.

3. Grafici di Relazione

Seaborn facilita la creazione di grafici a dispersione e regressione, come scatterplot e regplot, che visualizzano la relazione tra due variabili. È anche possibile utilizzare pairplot per mostrare tutte le relazioni tra variabili di un DataFrame.

4. Heatmap

Le heatmap sono utilizzate per rappresentare matrici di dati numerici con colori. Sono particolarmente utili per visualizzare le correlazioni tra variabili in un dataset.

Metodi per Visualizzare la Distribuzione dei Dati

- **sns.histplot(): Crea un istogramma, che rappresenta la distribuzione di una singola variabile.**

```
sns.histplot(data, bins=30)
```

- **sns.kdeplot(): Stima la densità del kernel per visualizzare la distribuzione dei dati in una forma liscia e continua.**

```
sns.kdeplot(data, shade=True)
```

- **sns.displot(): Combina istogrammi e KDE per visualizzare la distribuzione dei dati in un singolo grafico.**

```
sns.displot(data, kind="kde")
```

- **sns.ecdfplot(): Crea un grafico della funzione di distribuzione cumulativa empirica (ECDF), utile per mostrare come i valori si accumulano nel tempo.**

```
sns.ecdfplot(data)
```

Metodi per Relazioni tra Variabili

- **sns.scatterplot(): Crea un grafico a dispersione (scatter plot), visualizzando la relazione tra due variabili continue.**

1. **sns.scatterplot(x='x', y='y', data=data)**

- **sns.lineplot(): Crea un grafico a linee, spesso usato per mostrare tendenze temporali o relazioni tra variabili continue.**

1. **sns.lineplot(x='x', y='y', data=data)**

Metodi per Variabili Categoriali

- **sns.barplot(): Visualizza la media di una variabile continua rispetto a una variabile categoriale, con barre di errore.**

```
sns.barplot(x='categoria', y='valore', data=data)
```

- **sns.countplot(): Conta il numero di occorrenze di ciascuna categoria e visualizza i risultati come barre.**

```
sns.countplot(x='categoria', data=data)
```

- **sns.boxplot(): Mostra la distribuzione dei dati utilizzando una scatola che rappresenta i quartili e le code. Utile per confrontare variabili continue tra gruppi categoriali.**

```
sns.boxplot(x='categoria', y='valore', data=data)
```

Esempi di Utilizzo

Esempio 1: Creazione di un Grafico a Dispersione (Scatter Plot) con Regressione

Seaborn può essere utilizzato per visualizzare un grafico a dispersione con una linea di regressione sovrapposta, che mostra la relazione tra due variabili.

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

# Creazione di un dataset di esempio
np.random.seed(0)
x = np.random.rand(100)
y = 2 * x + np.random.normal(0, 0.1, 100)
data = pd.DataFrame({'x': x, 'y': y})

# Creazione di un grafico a dispersione con linea di regressione
sns.regplot(x='x', y='y', data=data)

# Aggiunta di titolo
plt.title('Grafico a Dispersione con Regressione')

# Mostra il grafico
plt.show()
```

Spiegazione dell'Esempio

- **Dati: Creiamo un dataset fittizio con 100 punti utilizzando Pandas.**
- **Grafico a Dispersione: Il comando sns.regplot() disegna un grafico a dispersione con una linea di regressione calcolata automaticamente.**
- **Personalizzazione: Viene aggiunto un titolo con il comando plt.title().**

Esempio 2: Creazione di una Heatmap

Le heatmap sono utili per visualizzare matrici di correlazione o altri tipi di dati tabulari in cui ogni cella rappresenta un valore con una scala di colori

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

# Creazione di dati casuali per la heatmap
data = np.random.rand(10, 12)

# Creazione della heatmap
sns.heatmap(data, cmap='viridis', annot=True, linewidths=0.5)

# Aggiunta di titolo
plt.title('Heatmap con Annotazioni')

# Mostra il grafico
plt.show()

```

Spiegazione dell'Esempio

- **Dati: Creiamo una matrice casuale di dati di dimensioni 10x12 utilizzando NumPy.**
- **Heatmap: Il comando sns.heatmap() crea una heatmap, dove ogni cella rappresenta un valore della matrice. annot=True mostra i valori numerici all'interno di ogni cella e linewidths=0.5 aggiunge una sottile linea di separazione tra le celle.**
- **Personalizzazione: Aggiungiamo un titolo con plt.title().**

Differenze tra Seaborn e Matplotlib

- **Sintassi più semplice: Seaborn riduce il codice necessario per creare grafici complessi rispetto a Matplotlib.**
- **Stile predefinito: I grafici Seaborn hanno uno stile moderno e professionale senza necessità di configurazioni manuali.**
- **Palette di colori: Seaborn offre una vasta gamma di colori predefiniti che migliorano l'estetica dei grafici.**
- **Focus su variabili categoriali: Seaborn ha comandi specifici per visualizzare dati categoriali, come barplot, violin plot, ecc.**

Conclusione

Seaborn è uno strumento eccellente per creare grafici statistici e visualizzazioni eleganti con meno codice rispetto a Matplotlib.

Grazie alla sua integrazione con Pandas e alle sue funzionalità avanzate, è una libreria molto utile per chi lavora con dati strutturati e vuole esplorare relazioni tra variabili, distribuzioni e categorie.


Buon MasterD a tutti