Seaborn: A Comprehensive Guide to Data Visualization

Seaborn is a powerful and versatile Python library built on top of Matplotlib, designed for creating informative and visually appealing statistical graphics. It simplifies complex data visualization tasks, making it easier to explore and understand data through a wide range of plots. With Seaborn, users can effortlessly create aesthetic and meaningful charts with just a few lines of code, utilizing various built-in datasets and customizable options.

This guide walks you through the major plots in Seaborn, showcasing how to implement and customize them for effective data analysis.

Import Libraries and Load Dataset

```
import seaborn as sns
import matplotlib.pyplot as plt
# Load the 'tips' dataset

tips = sns.load_dataset("tips")
tips.head()
```

OUTPUT:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

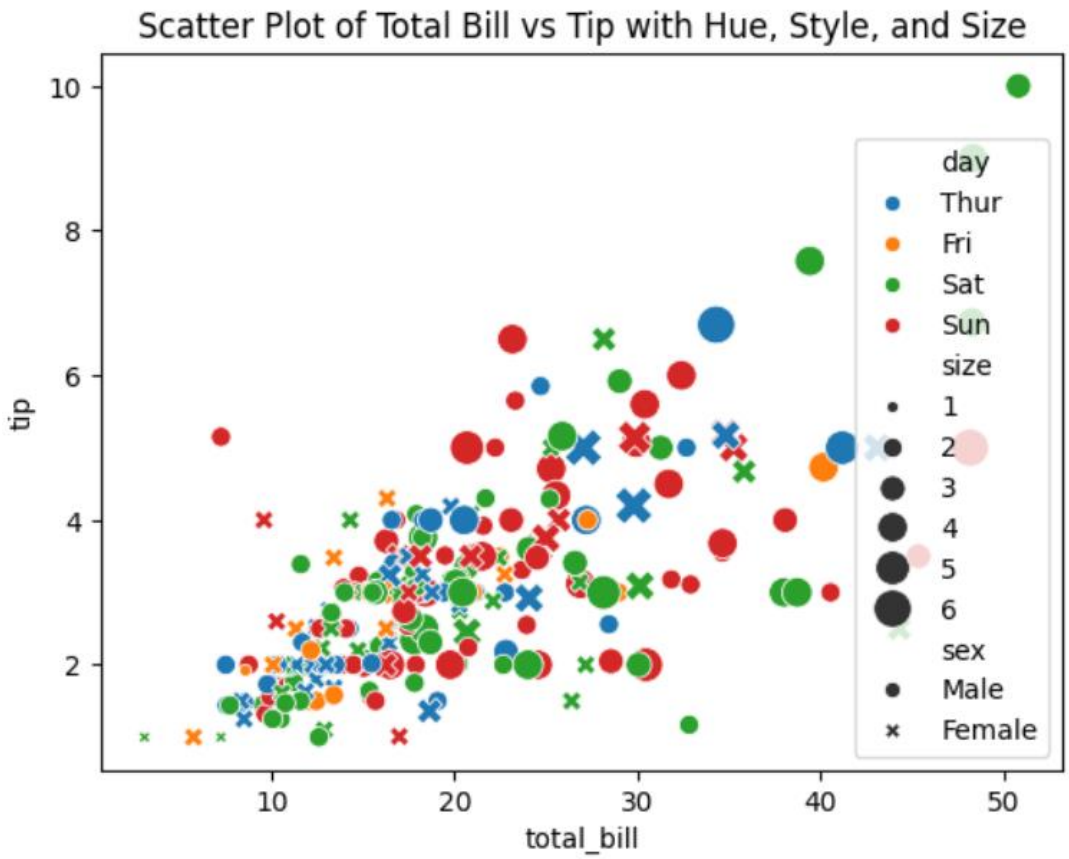
1. Scatter Plot

Scatter Plot Parameters:

- hue: Adds color grouping based on another variable.
- style: Differentiates data points by marker style based on another variable.
- size: Changes the size of the data points based on another variable.

```
# Scatter plot of total_bill vs tip with hue, style, and size
sns.scatterplot(data=tips, x='total_bill', y='tip', hue='day', style='sex', size='size', sizes=(20, 200))
```

```
plt.title('Scatter Plot of Total Bill vs Tip with Hue, Style, and Size')
plt.show()
```



Explanation:

- hue: Color codes the points based on the "day" column.
- style: Uses different markers based on the "sex" column.
- size: Changes the size of points based on the "size" column, with point size ranging between 20 and 200.

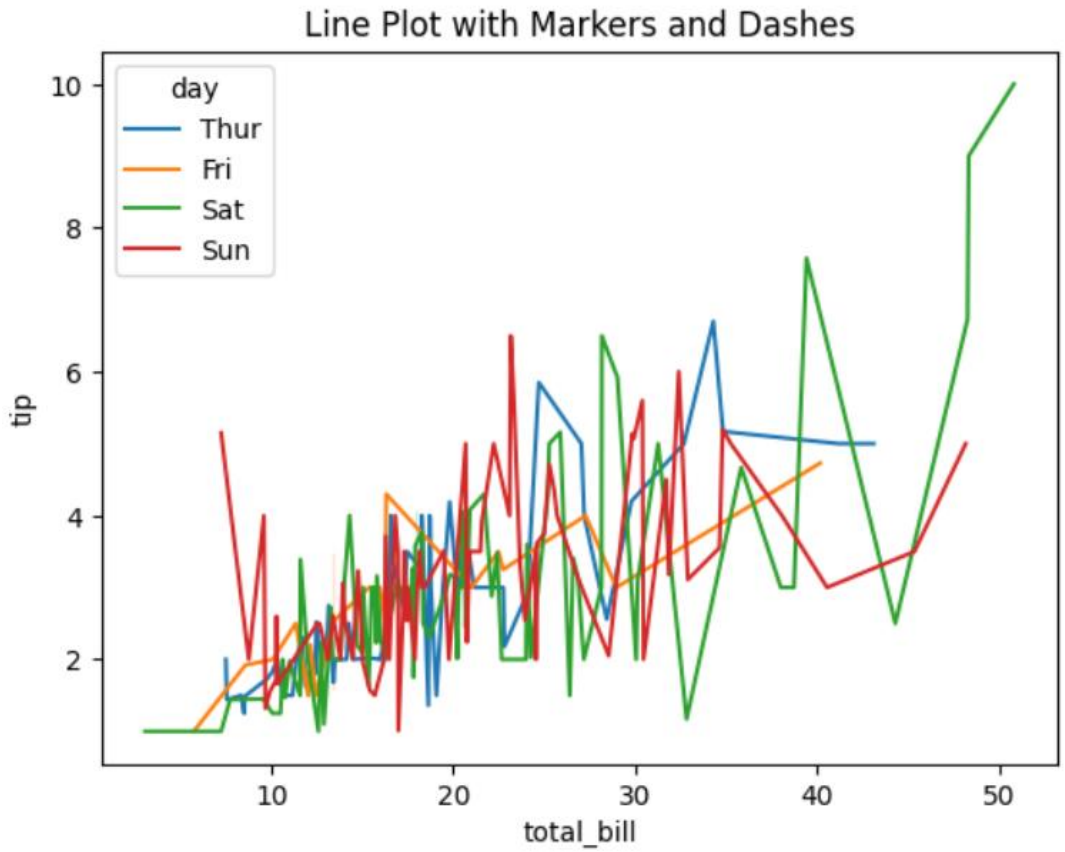
---

2. Line Plot

Line Plot Parameters:

- markers: Adds markers at data points.
- dashes: Customizes the dash style of the lines.

```
# Line plot with markers and dashes
sns.lineplot(data=tips, x='total_bill', y='tip', hue='day', markers=True, dashes=False)
plt.title('Line Plot with Markers and Dashes')
plt.show()
```



Explanation:

- markers: Adds markers to data points for better visibility.
- dashes: If set to False, it creates solid lines for all categories.

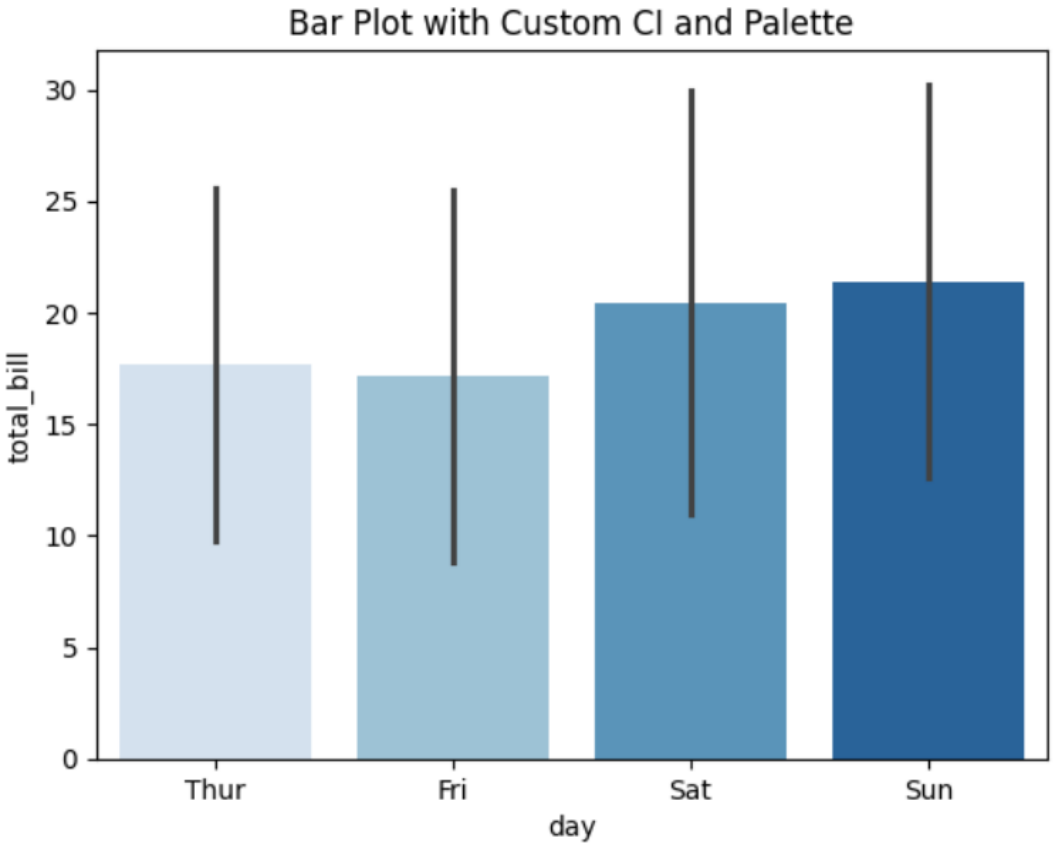
---

3. Bar Plot

Bar Plot Parameters:

- ci: Controls confidence interval.
- palette: Changes the color palette.

```
# Bar plot of total_bill with custom confidence interval and palette
sns.barplot(data=tips, x='day', y='total_bill', ci='sd', palette='Blues')
plt.title('Bar Plot with Custom CI and Palette')
plt.show()
```



Explanation:

- ci: Shows the confidence interval of standard deviation ('sd').
- palette: Customizes the color palette (in this case, shades of blue).

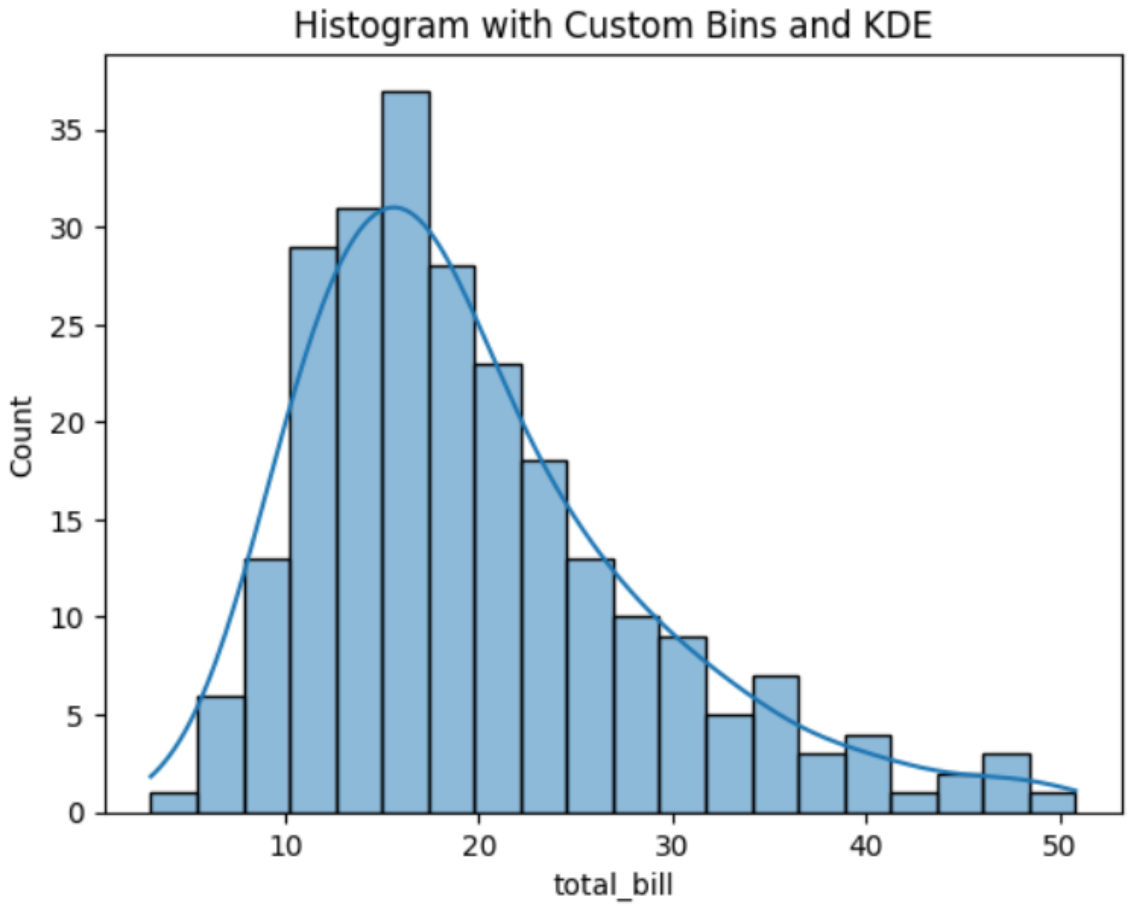
---

4. Histogram

Histogram Parameters:

- bins: Adjusts the number of bins (intervals) for the histogram.
- kde: Adds a Kernel Density Estimate curve.

```
# Histogram of total_bill with custom bins and KDE
sns.histplot(data=tips, x='total_bill', bins=20, kde=True)
plt.title('Histogram with Custom Bins and KDE')
plt.show()
```



Explanation:

- bins: Sets the number of bins to 20.
- kde: Adds a smoothed density curve (KDE) to the histogram.

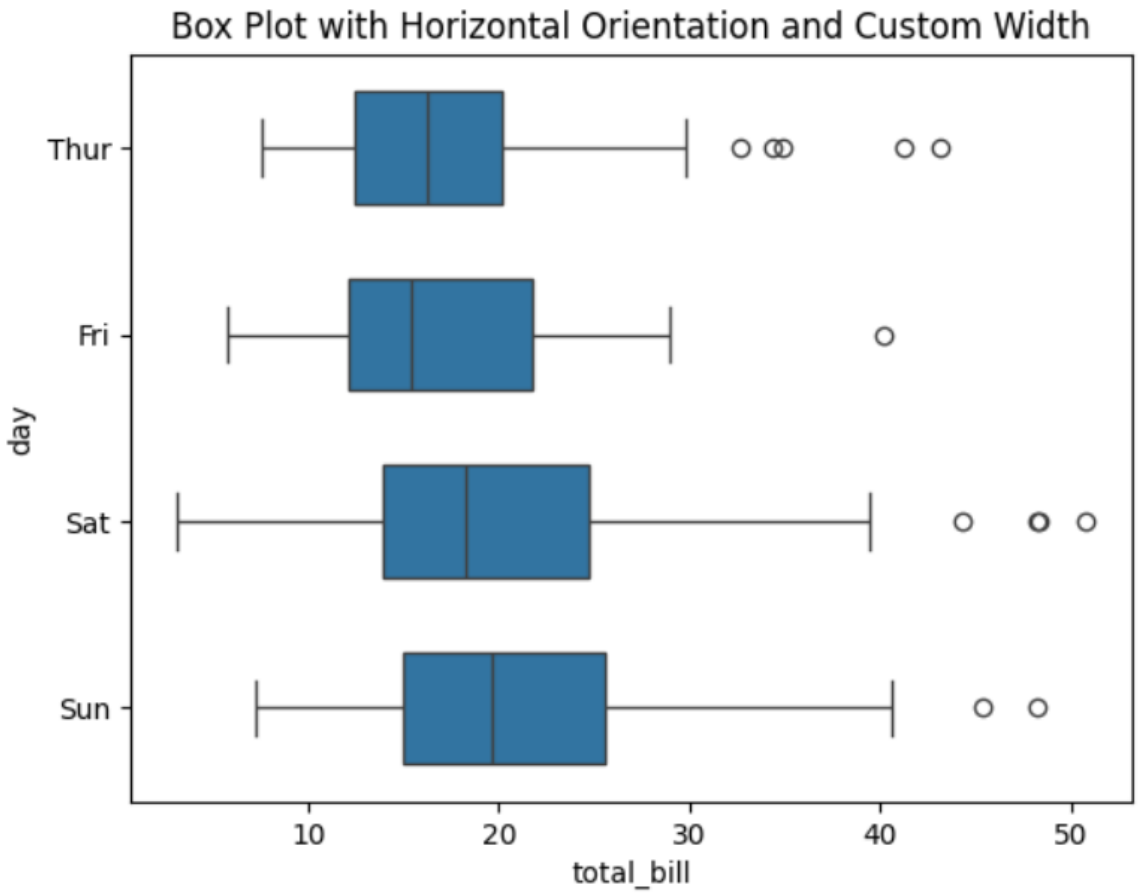
---

5. Box Plot

Box Plot Parameters:

- orient: Changes the orientation of the plot.
- width: Adjusts the width of the box plot.

```
# Box plot of total_bill with horizontal orientation
sns.boxplot(data=tips, x='total_bill', y='day', orient='h', width=0.6)
plt.title('Box Plot with Horizontal Orientation and Custom Width')
plt.show()
```



Explanation:

- orient: Changes the orientation to horizontal ('h').
- width: Sets the width of the boxes to 0.6 for a balanced look.

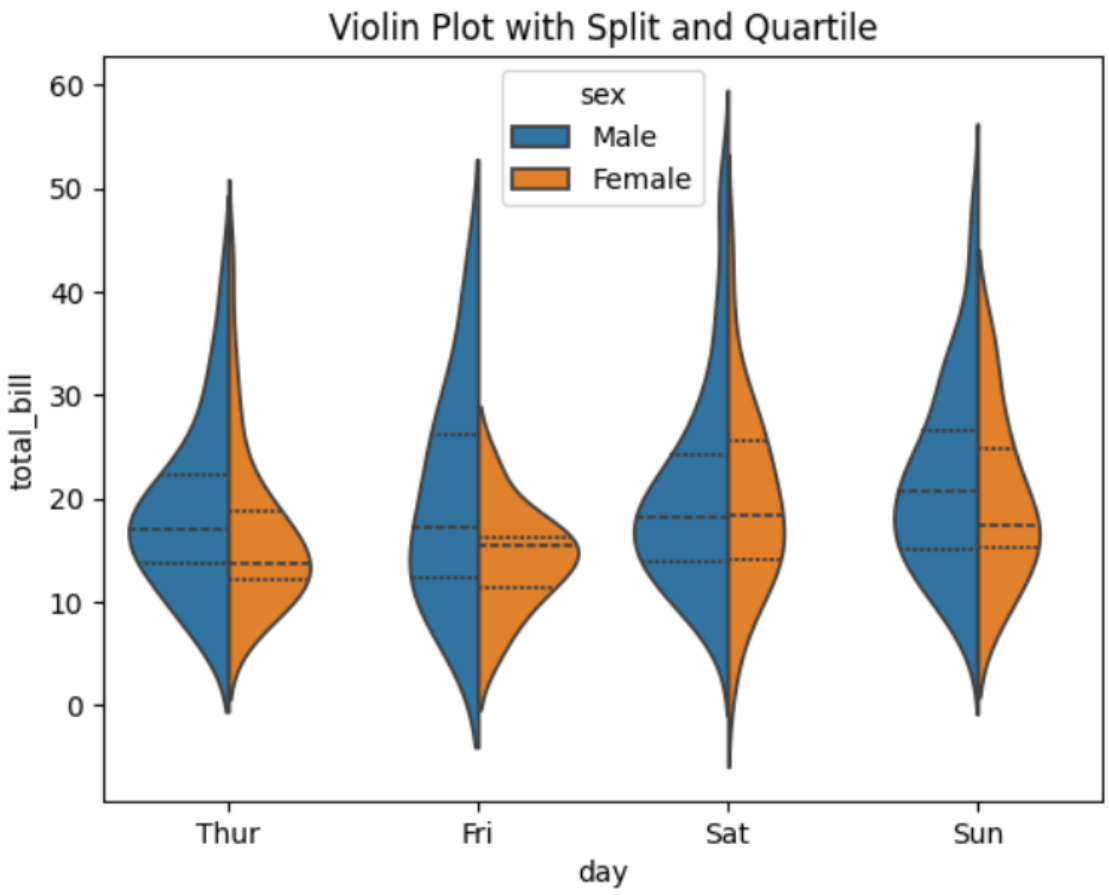
---

6. Violin Plot

Violin Plot Parameters:

- split: Shows data on the same violin for different categories.
- inner: Controls the plot inside the violin (e.g., box, quartile, or stick).

```
# Violin plot with split by sex and inner as quartile
sns.violinplot(data=tips, x='day', y='total_bill', hue='sex', split=True, inner='quartile')
plt.title('Violin Plot with Split and Quartile')
plt.show()
```



Explanation:

- split: Splits the violin by "sex" within each day category.
- inner: Displays quartiles inside the violin plot.

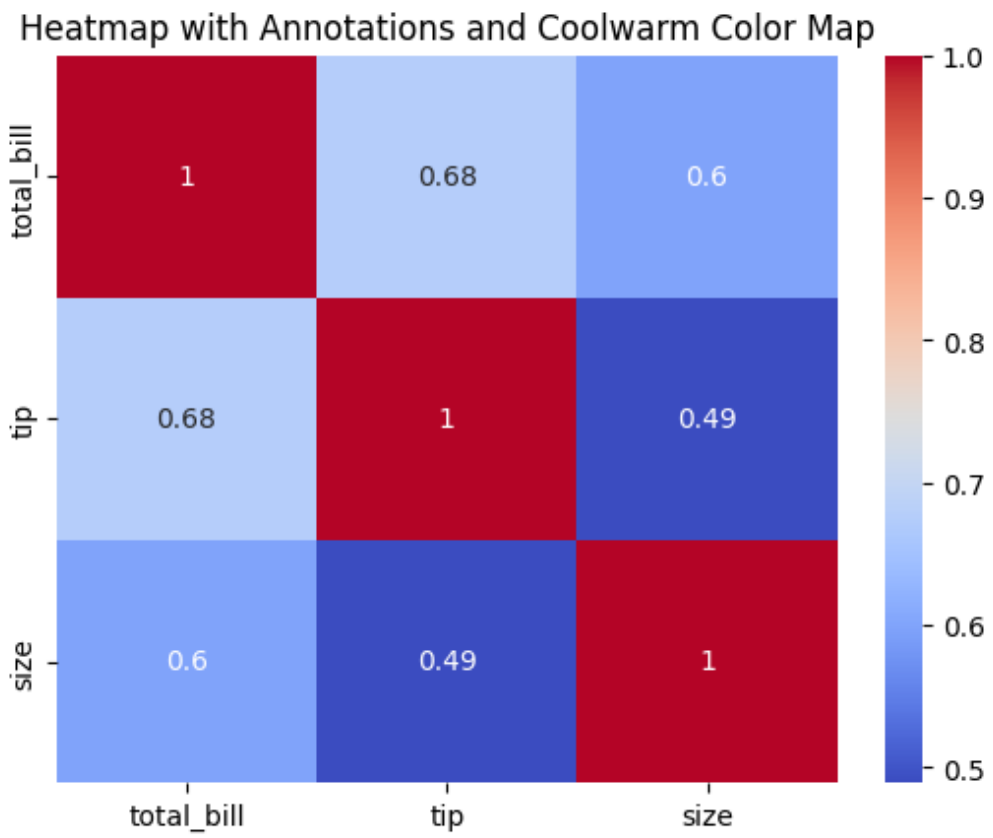
---

7. Heatmap

Heatmap Parameters:

- annot: Adds numeric annotations to the cells.
- cmap: Changes the color mapping.

```
# Heatmap of correlation matrix with annotations and a different color map
# Select only numeric columns
tips_numeric = tips.select_dtypes(include=['float64', 'int64'])
# Compute the correlation matrix on numeric data
corr = tips_numeric.corr()
# Create a heatmap with annotations and the 'coolwarm' color map
 sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Heatmap with Annotations and Coolwarm Color Map')
plt.show()
```



Explanation:

- annot: Adds the correlation values to each cell.
- cmap: Uses the 'coolwarm' colormap for a visually appealing look.

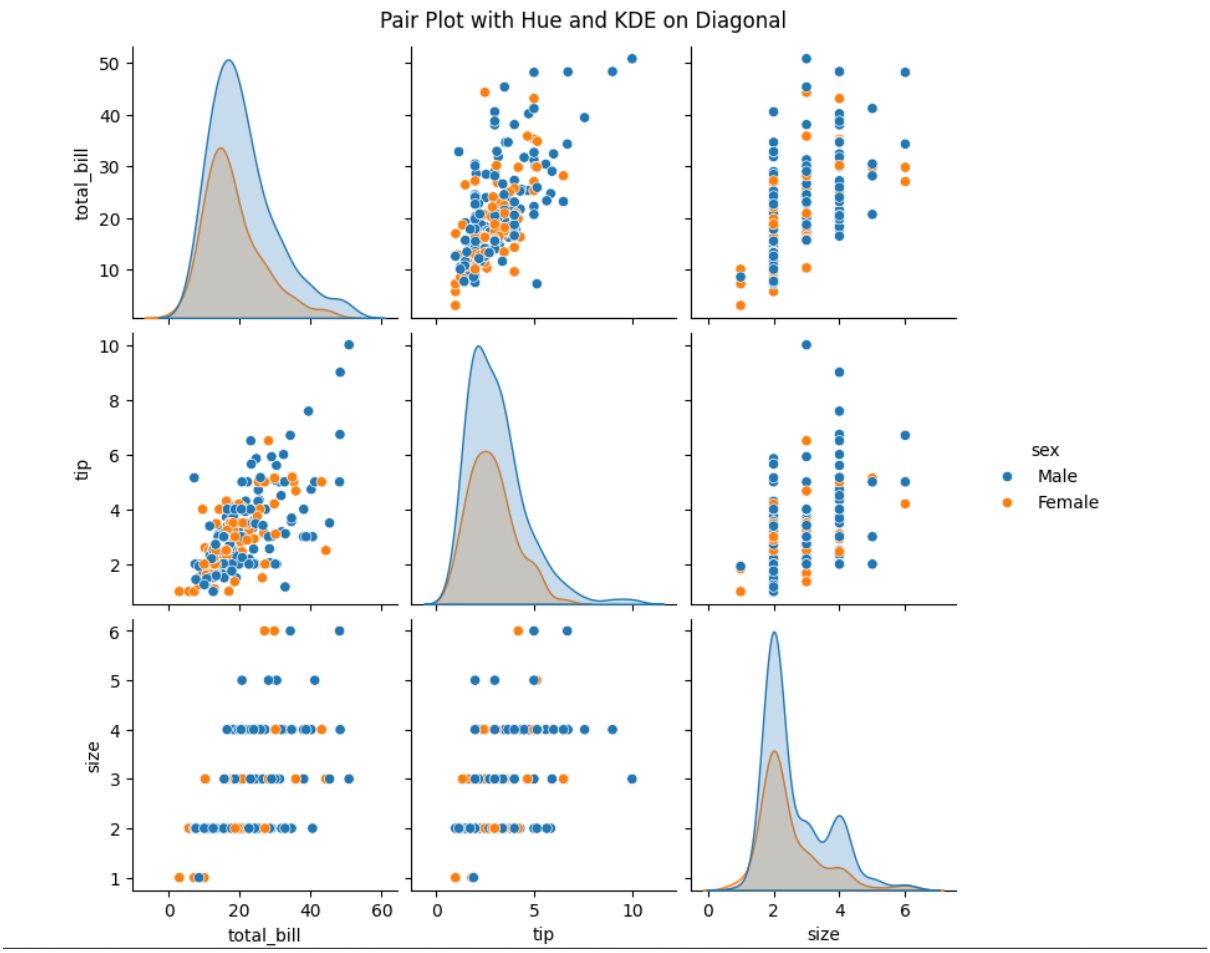
---

8. Pair Plot

Pair Plot Parameters:

- hue: Adds color for grouping based on a column.
- diag_kind: Defines the type of plot on the diagonal (e.g., kde or hist).

```
# Pair plot with hue and diagonal kind as KDE
sns.pairplot(tips, hue='sex', diag_kind='kde')
plt.suptitle('Pair Plot with Hue and KDE on Diagonal', y=1.02)
plt.show()
```



Explanation:

- hue: Groups the data points by "sex" using different colors.
- diag_kind: Uses KDE (Kernel Density Estimate) on the diagonal.

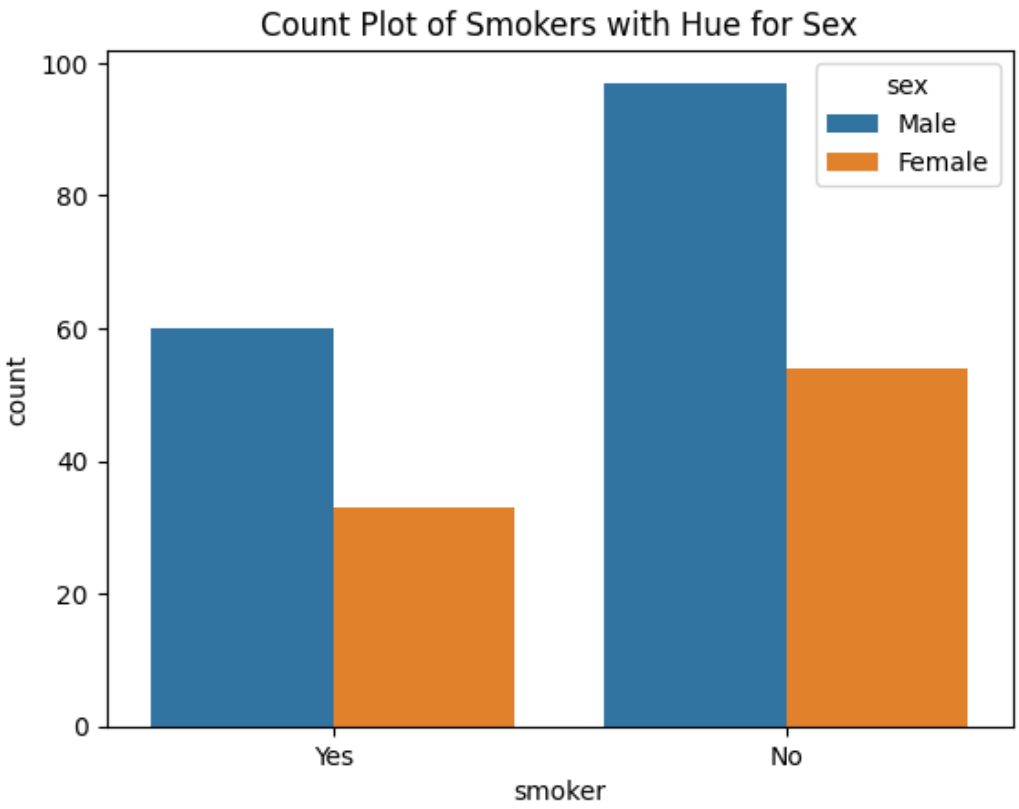
---

9. Count Plot

Count Plot Parameters:

- hue: Adds color coding for another categorical variable.
- dodge: Adjusts the bars when hue is applied.

```
# Count plot of smokers with hue for sex
sns.countplot(data=tips, x='smoker', hue='sex', dodge=True)
plt.title('Count Plot of Smokers with Hue for Sex')
plt.show()
```



Explanation:

- hue: Adds separate bars for "sex" within each category of "smoker".
- dodge: Ensures that the bars for "sex" are placed next to each other.