DATA VISUALIZATION USING PYTHON

MatplotLib and Seaborn(Python Libraries)

Task By ShadowFox

To Create a comprehensive documentation guide for 2 of the following Python visualization libraries: Matplotlib, Seaborn, Plotly, Bokeh, and Pandas. Your guide should focus on the variety of graphs each library can generate and include practical examples with code snippets.

Mohd Osama from Babu Banarasi Das University
Mohdosama028316@gmail.com

Data Visualization using Python Libraries Matplotlib and Seaborn

Objective: Create a comprehensive documentation guide for 2 of the following visualization libraries: Matplotlib, Seaborn, Plotly, Bokeh, and Pandas. Your guide should focus on the variety of graphs each library can generate and include practical examples with code snippets.

Data Visualization by using MatplotLib :

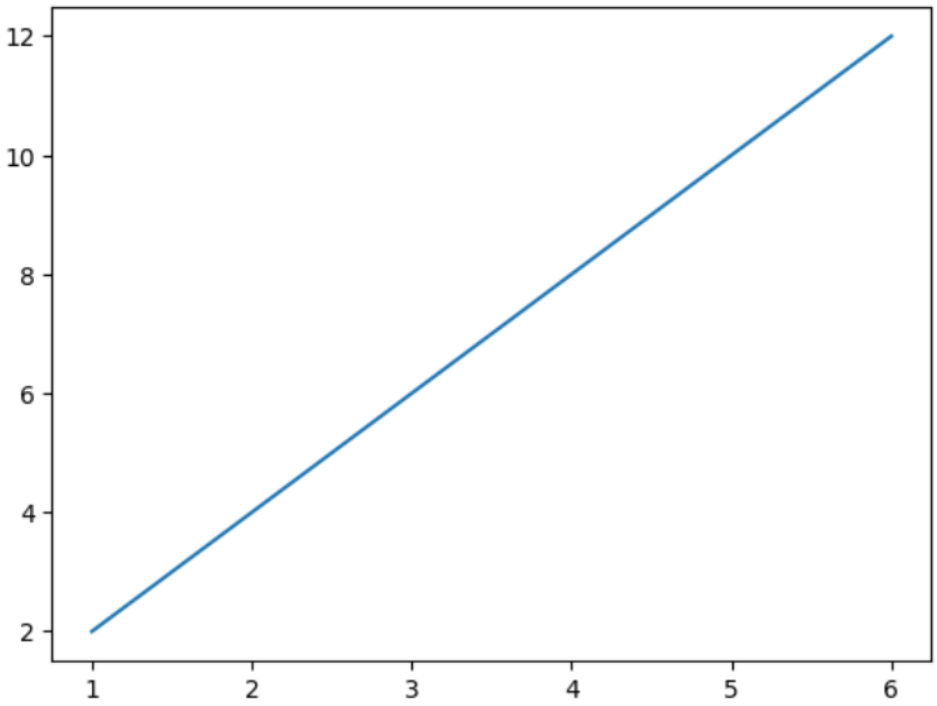
1. **Line plot:** A line plot in Matplotlib is used to visualize the relationship between two variables, typically with one variable along the x-axis and another along the y-axis. It is especially useful when you want to observe trends, patterns, or changes over a continuous range (e.g., time, distance, temperature).
   - First we have to import matplotlib library and other.

```
]:  1 import matplotlib.pyplot as plt
    2 import pandas as pd
    3 import numpy as np
```

   - Then visualize

```
1 x=[1,2,3,4,5,6]
2 y=[2,4,6,8,10,12]
```

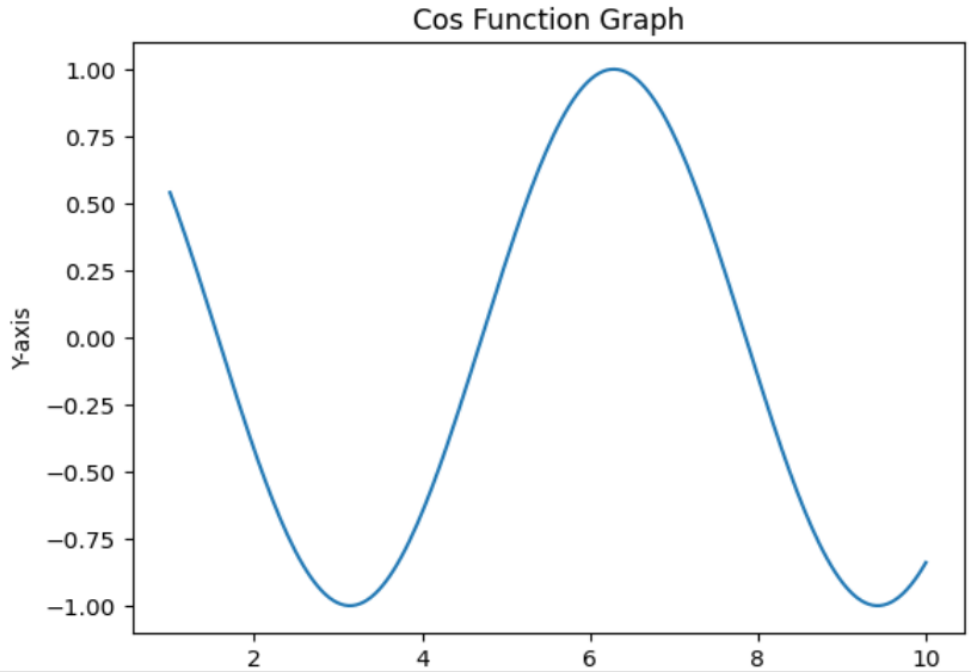
```
1 plt.plot(x,y)
2 plt.show()
```



➢ Making Cos function Graph using line plot

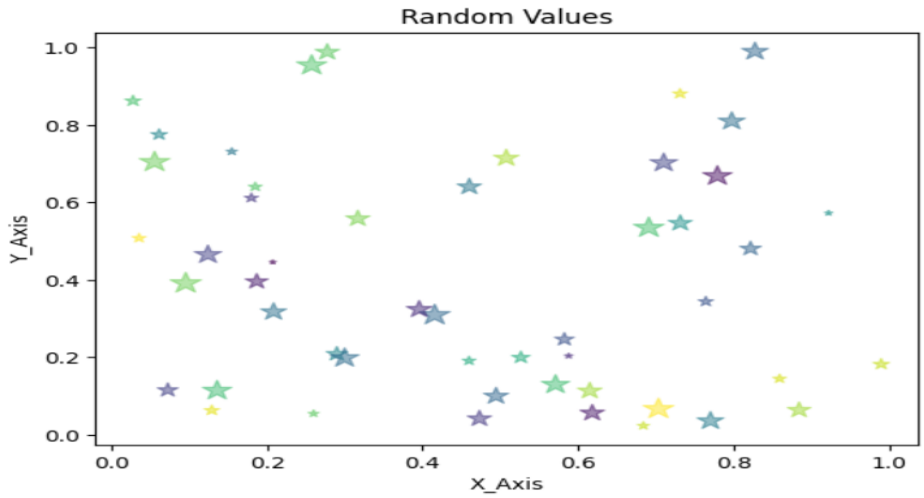
```
x1=np.linspace(1,10,200)
y1=np.cos(x1)
```

```
plt.plot(x1,y1)
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Cos Function Graph")
plt.show()
```



2. **Scatter plot:** A scatter plot in Matplotlib is a type of plot used to display the relationship between two variables. It visualizes individual data points, with one variable plotted along the x-axis and the other along the y-axis. Each point represents a single observation.

```
In [8]:
m=np.random.rand(50)
n=np.random.rand(50)
color=np.random.rand(50)
size=200*np.random.rand(50)
plt.scatter(m,n,c=color,s=size,alpha=.5,marker='*')
plt.xlabel("X_Axis")
plt.ylabel("Y_Axis")
plt.title("Random Values")
plt.show()
```



3. **Bar plot:** A bar plot in Matplotlib is a type of visualization used to represent data where values are displayed as rectangular bars. The length or height of each bar is proportional to the value it represents.

➢ Importing data

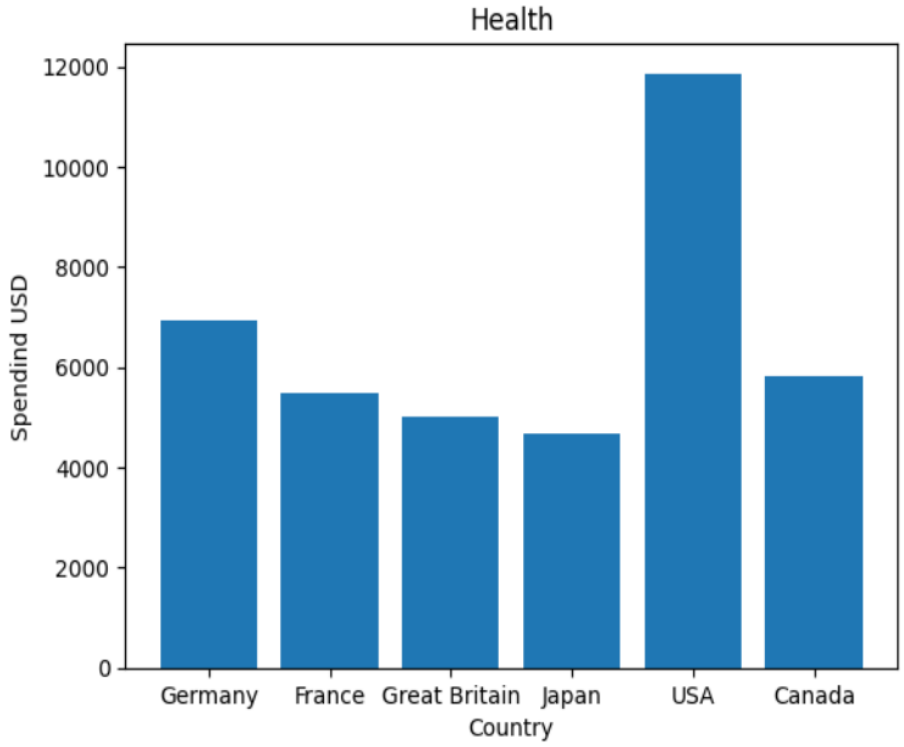
```
df=pd.read_csv("https://raw.githubusercontent.com/mwaskom/seaborn-data/master/healthexp.csv")
```

```
df.head()
```

	Year	Country	Spending_USD	Life_Expectancy
0	1970	Germany	252.311	70.6
1	1970	France	192.143	72.2
2	1970	Great Britain	123.993	71.9
3	1970	Japan	150.437	72.0
4	1970	USA	326.961	70.9

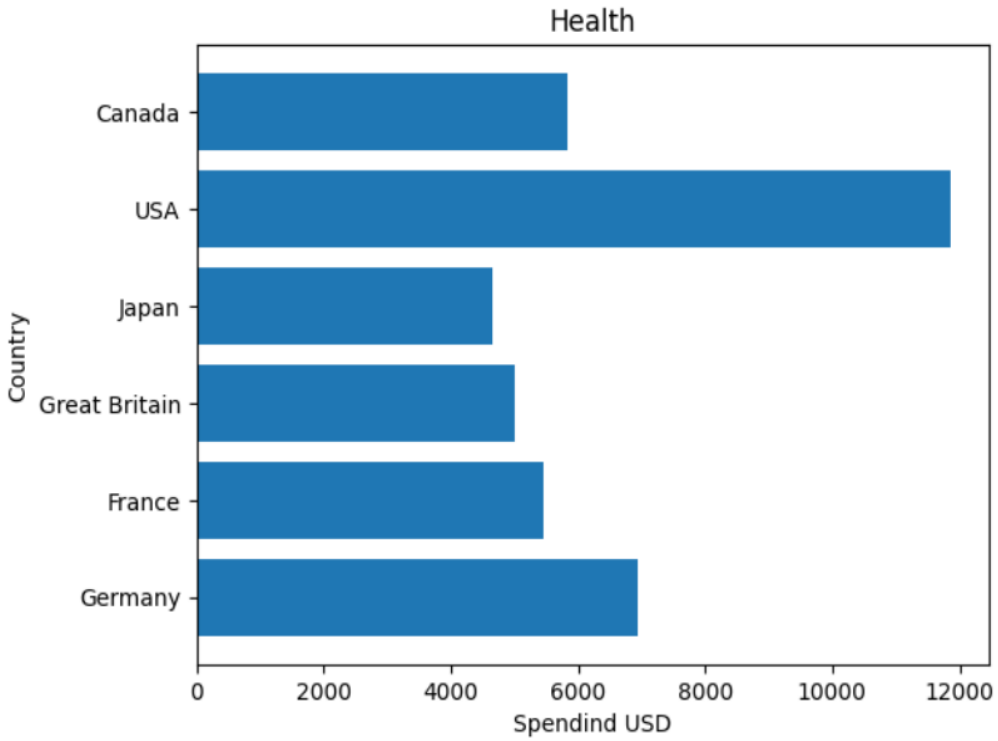
➢ **Vertical Bar:**

```
plt.bar(df['Country'],df["Spending_USD"])
plt.xlabel("Country")
plt.ylabel("Spendind USD")
plt.title("Health")
plt.show()
```



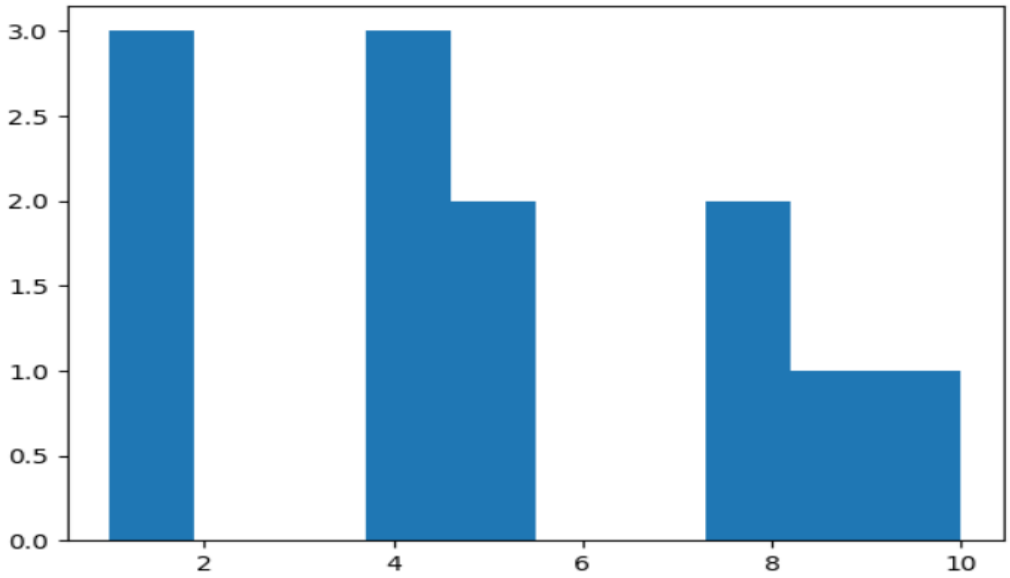
➢ **Horizontal Bar:**

```
plt.barh(df['Country'],df["Spending_USD"])
plt.ylabel("Country")
plt.xlabel("Spendind USD")
plt.title("Health")
plt.show()
```



4. Histogram : A histogram is a graphical representation used to visualize the distribution of a dataset. In Matplotlib, histogram plots are created using the plt.hist() function.

```
data=[1,1,1,4,4,5,8,8,9,10,5,4]
plt.hist(data)
plt.show()
```



4. **3D Plot:** 3D plotting in Matplotlib is a powerful tool for visualizing and analyzing multidimensional data. It is used across various domains for a wide range of purposes.

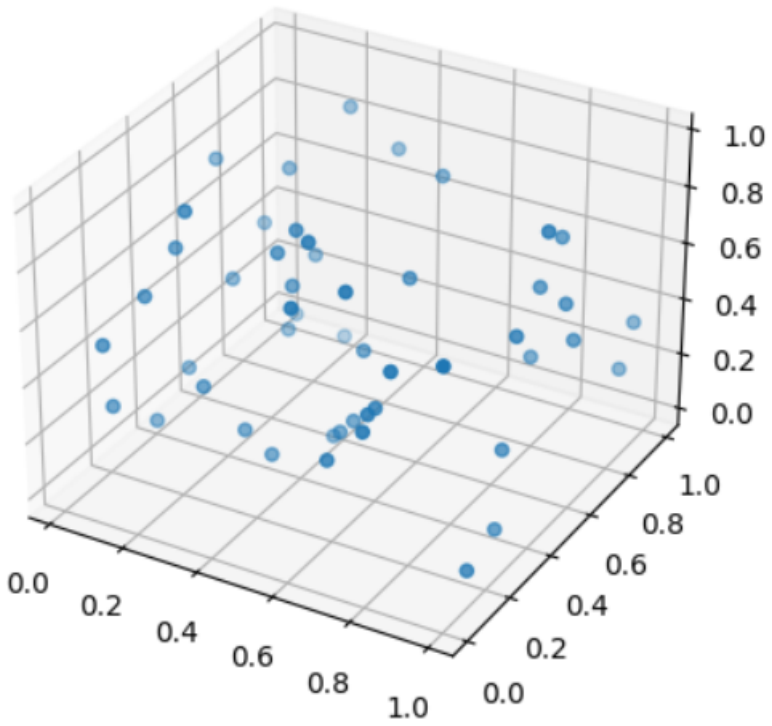
```
#3D

x=np.random.rand(50)
y=np.random.rand(50)
z=np.random.rand(50)

fig=plt.figure()
ax=fig.add_subplot(projection='3d')
ax.scatter(x,y,z)

plt.show()

```



Data Visualization by using Seaborn:

Seaborn is a Python library built on Matplotlib that simplifies statistical data visualization, providing aesthetically pleasing default styles, advanced plotting functions, and seamless integration with Pandas dataframes. It offers high-level tools for visualizing distributions, relationships, and statistical summaries, making complex plots like heatmaps, boxplots, and pair plots easy to create with minimal code. Seaborn's focus on statistical insights, combined with customizable themes and color palettes, makes it ideal for creating insightful and attractive visualizations for data analysis and storytelling.

- **Importing data and libraries:**

```
In [1]:
1 import seaborn as sns
2 import pandas as pd
3 import matplotlib.pyplot as plt
```

```
In [2]:
1 df=sns.load_dataset('iris')
```

```
In [3]:
1 df.head()
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

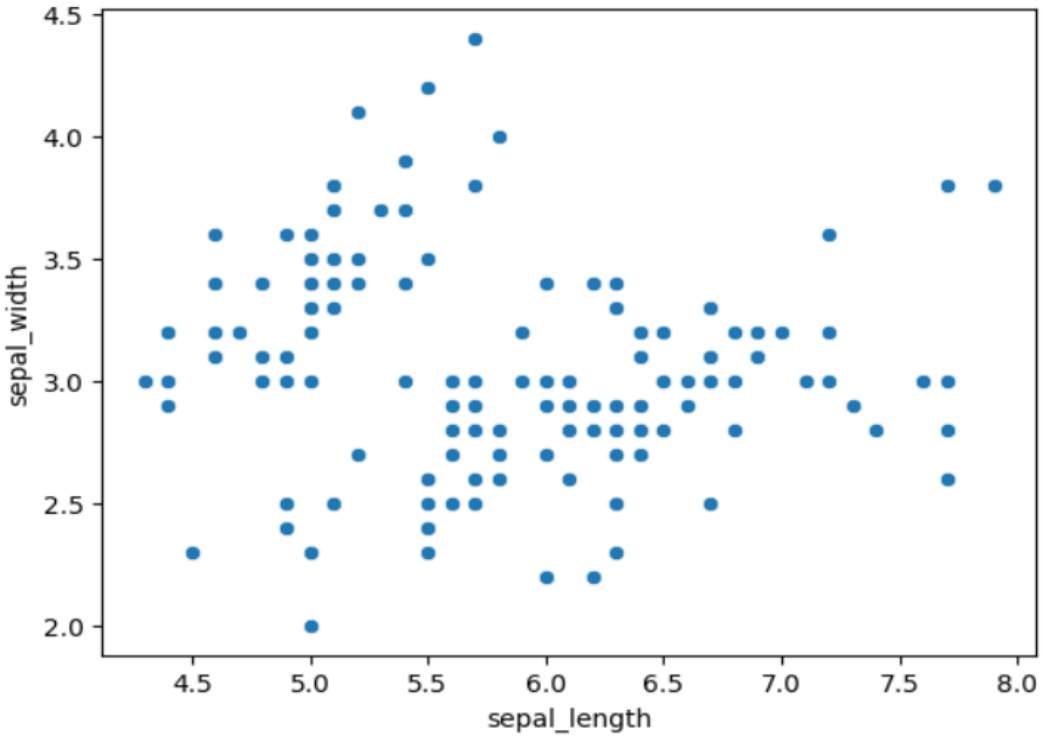
```
1 df1=sns.load_dataset('tips')
```

```
1 df1.head()
```

	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

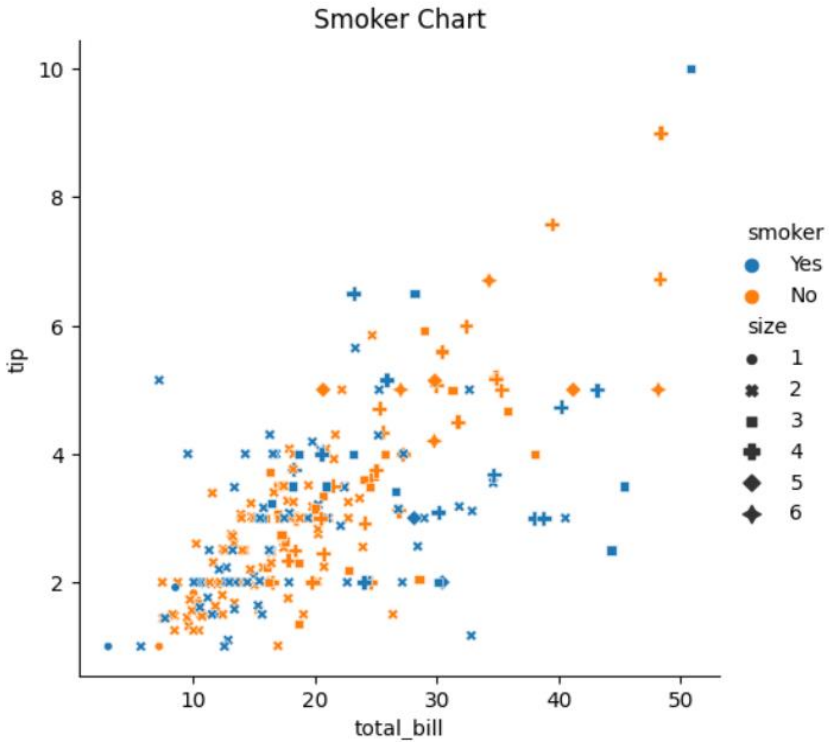
Scatter Plot:

```
sns.scatterplot(x=df.sepal_length,y=df.sepal_width)
plt.show()
```



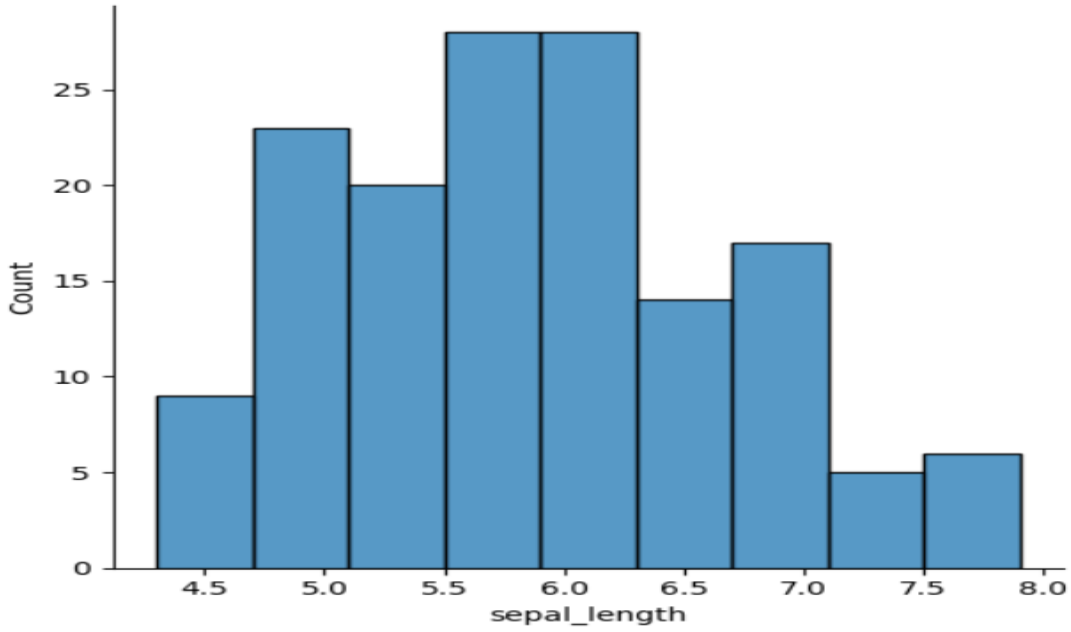
➢ Advance Scatter plot:

```
sns.relplot(x=df1.total_bill,y=df1.tip, data=df1, hue='smoker',style='size',size='size')
plt.title("Smoker Chart")
plt.show()
```



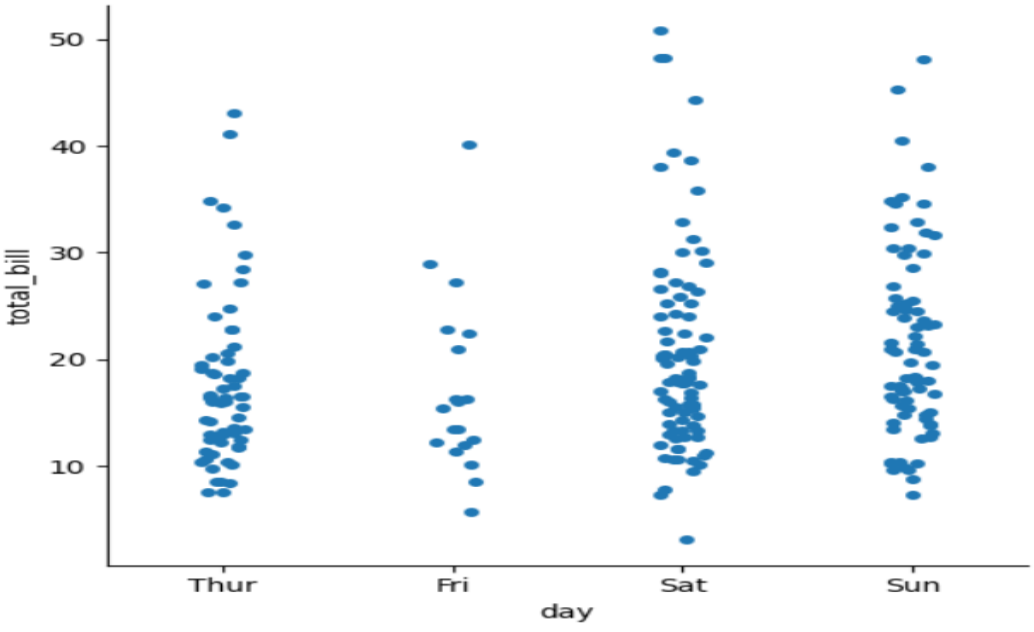
Distribution plot: The distribution plot (distplot) in Seaborn is used to visualize the distribution of a dataset. It combines a histogram and a kernel density estimate (KDE) by default to give insights into the underlying distribution of the data.

```
1 sns.displot(df['sepal_length'])
2 plt.show()
```

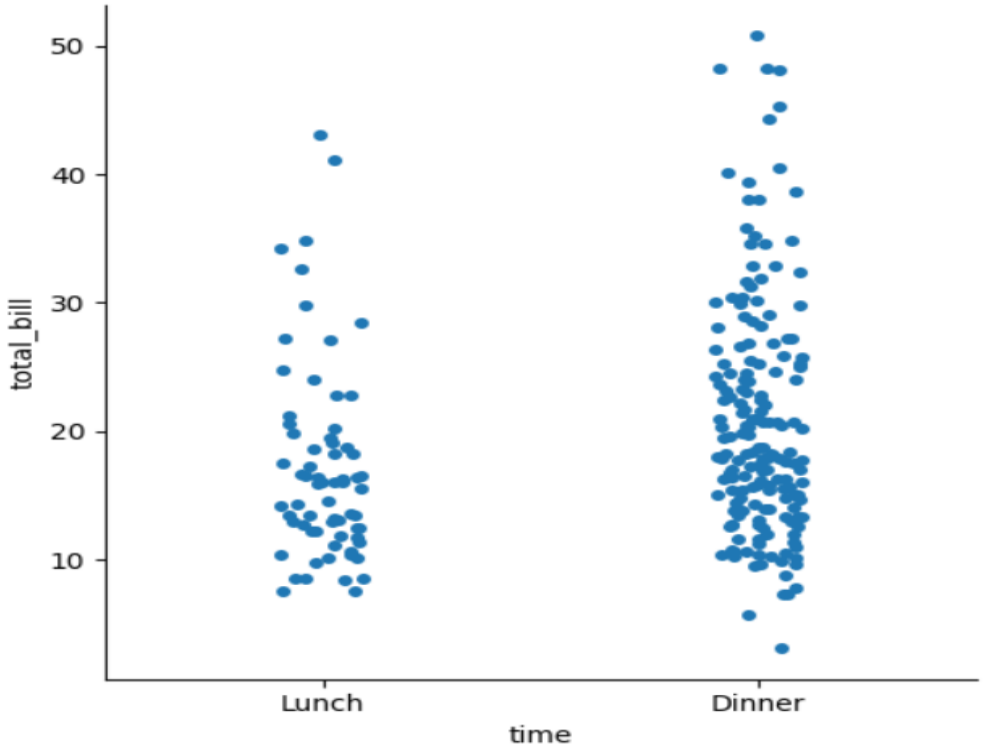


Cat plot: A catplot in Seaborn is a versatile and powerful visualization tool designed for creating categorical plots. It allows users to visualize relationships between a categorical variable and one or more numerical or categorical variables.

```
1 sns.catplot(x='day',y='total_bill',data=df1)
<seaborn.axisgrid.FacetGrid at 0x256f91eed10>
```

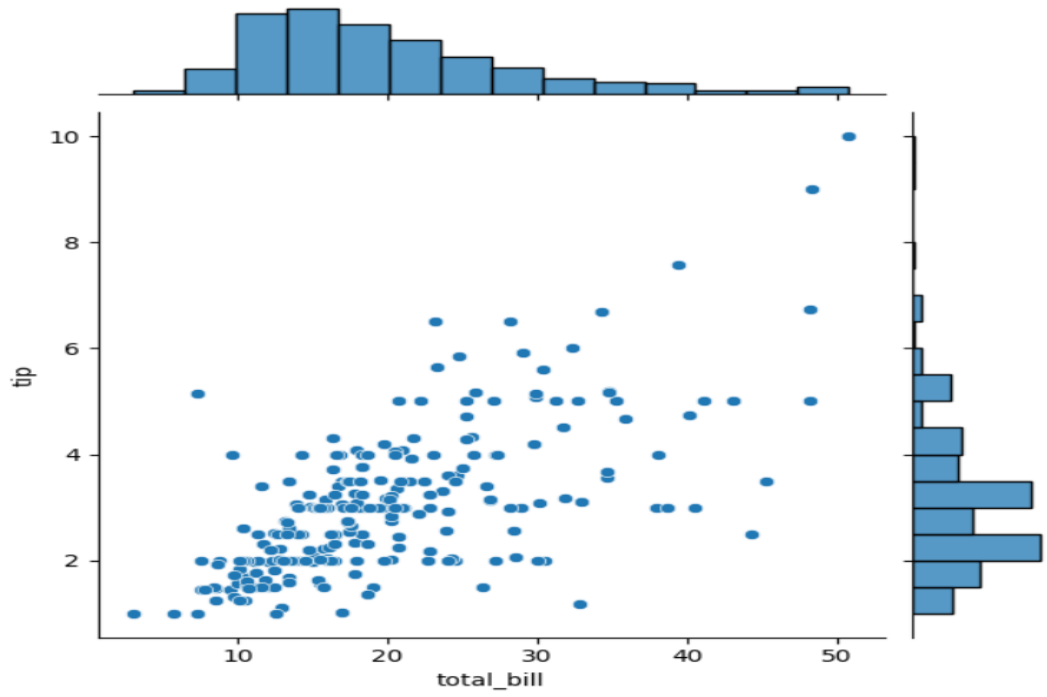


```
]:  1  sns.catplot(x='time',y='total_bill',data=df1)
    2  plt.show()
```



Joint plot: A joint plot in Seaborn visualizes the relationship between two variables, combining a scatter plot with marginal histograms or density plots to reveal correlations, distributions, patterns, and outliers effectively.

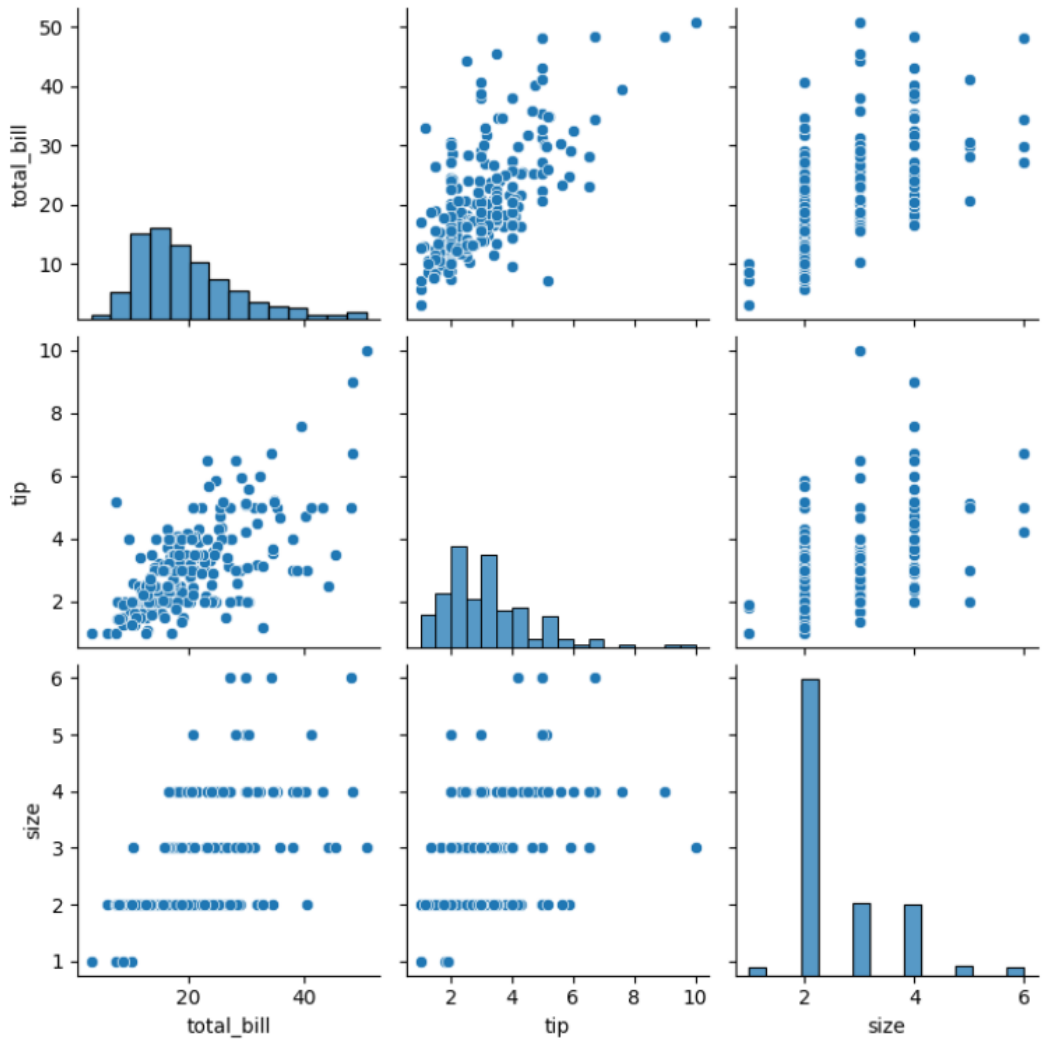
```
    1  sns.jointplot(x='total_bill',y='tip',data=df1)
<seaborn.axisgrid.JointGrid at 0x256fa458f40>
```



Pair plot: A pair plot in Seaborn visualizes relationships between multiple numerical variables by creating scatterplots for all possible pairs and histograms for individual variables, aiding in exploratory data analysis.

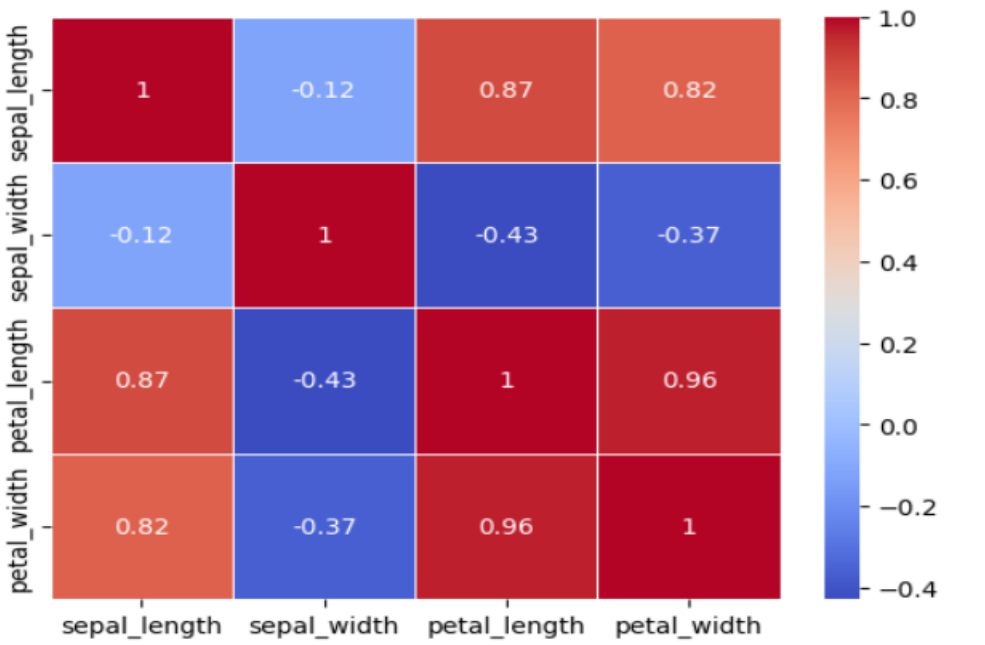
```
1 sns.pairplot(df1)
```

```
<seaborn.axisgrid.PairGrid at 0x256fa692590>
```



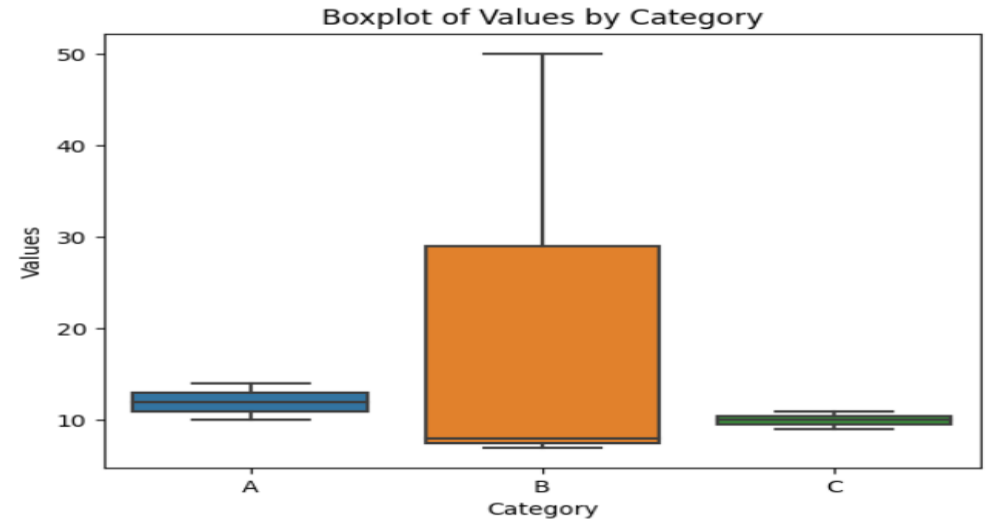
Heat Map: A heatmap in Seaborn is used to visualize data in a matrix form, highlighting patterns, correlations, or intensities through color gradients, making it ideal for exploring relationships, trends, or clusters in datasets.

```
]:  1  sns.heatmap(correlation,annot=True,cmap='coolwarm',linewidths=.5)
    2  plt.show()
```



Heat Map: A box plot in Seaborn is used to visualize the distribution, variability, and potential outliers of a dataset. It summarizes key statistics like median, quartiles, and range, facilitating comparison across groups or categories.

```
 2  data = {
 3      'Category': ['A', 'A', 'A', 'B', 'B', 'B', 'C', 'C', 'C'],
 4      'Values': [10, 12, 14, 8, 7, 50, 9, 11, 10]
 5  }
 6  # Create a DataFrame
 7  df = pd.DataFrame(data)
 8  # Create a boxplot
 9  sns.boxplot(data=df, x='Category', y='Values')
10  # Add title and labels
11  plt.title('Boxplot of Values by Category')
12  plt.xlabel('Category')
13  plt.ylabel('Values')
14
15  # Show the plot
16  plt.show()
17
```



Comparison of Matplotlib and Seaborn:

- Matplotlib provides extensive control and customization, allowing users to fine-tune every aspect of a plot. This makes it ideal for advanced users or situations where performance with large datasets is critical. However, it comes with a steeper learning curve and requires more code to create even simple plots.
- Seaborn, on the other hand, simplifies the plotting process with a high-level interface, making it easier and quicker to generate aesthetically pleasing visualizations. It's especially useful for statistical plots, but offers fewer customization options compared to Matplotlib. Seaborn may also struggle with performance when working with very large datasets, though its built-in aggregation functions can help.

Overall, Matplotlib is best suited for detailed and customized visualizations, while Seaborn is a great choice for quick and easy visual exploration of data.

Summary:

- Matplotlib: Ideal for detailed, highly customized visualizations and handling large datasets. Best for advanced users who need complete control.
- Seaborn: Excellent for quick, aesthetically pleasing visualizations with minimal effort, especially for statistical data analysis. It's a great choice for beginners or exploratory analysis.