

# ✓ TASK 1 : MASTERING DATA VISUALIZATION WITH PYTHON

## 🎯 Objective :

Visualization is not just about making attractive graphics—it's a critical part of data science that helps us **understand complex datasets, reveal hidden insights, and communicate findings effectively**.

This task focuses on **two popular Python libraries: Matplotlib and Seaborn**, demonstrating their capabilities through various plot types with code examples and outputs.

---

## 1. Library Overview

### Matplotlib

- Foundational Python library for creating static, animated, and interactive visualizations.
- Highly customizable; serves as a base for other libraries.
- **Use Cases:** Time-series analysis, exploratory data analysis, academic and research plots.

### Seaborn

- High-level interface built on Matplotlib.
  - Beautiful default themes and built-in statistical plotting.
  - **Use Cases:** Statistical data visualization, machine learning EDA, visual comparison across categories.
- 

## 2. Graph Types with Code and Outputs

### Matplotlib

#### 1. Line Plot

**Use Case:** Display trends over time.

```
import matplotlib.pyplot as plt  
  
x = [1, 2, 3, 4, 5]  
y = [10, 20, 15, 25, 30]  
  
plt.plot(x, y, marker='o', linestyle='--', color='green')  
  
plt.title("Line Plot Example")
```

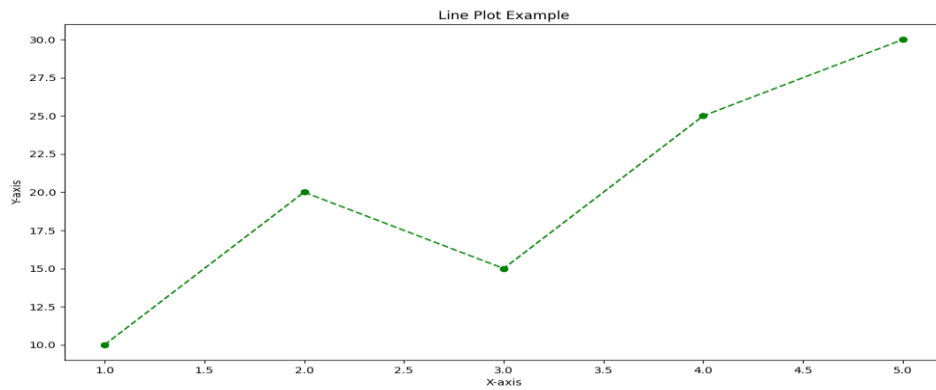
```
plt.xlabel("X-axis")
```

```
plt.ylabel("Y-axis")
```

```
plt.show()
```



**Output:**



## 2. Scatter Plot

**Use Case:** Show correlation between two variables.

```
plt.scatter(x, y, color='red')
```

```
plt.title("Scatter Plot Example")
```

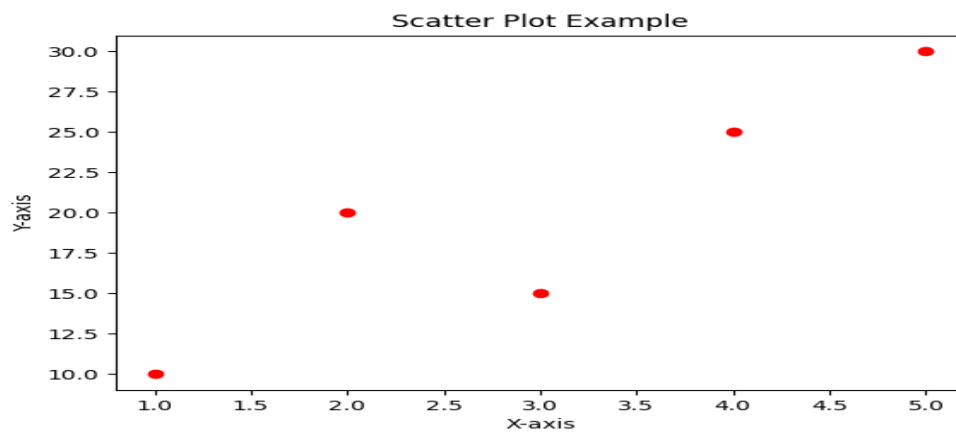
```
plt.xlabel("X-axis")
```

```
plt.ylabel("Y-axis")
```

```
plt.show()
```



**Output:**



---

### 3. Bar Chart

**Use Case:** Compare categorical values.

```
categories = ['A', 'B', 'C', 'D']
```

```
values = [5, 7, 3, 8]
```

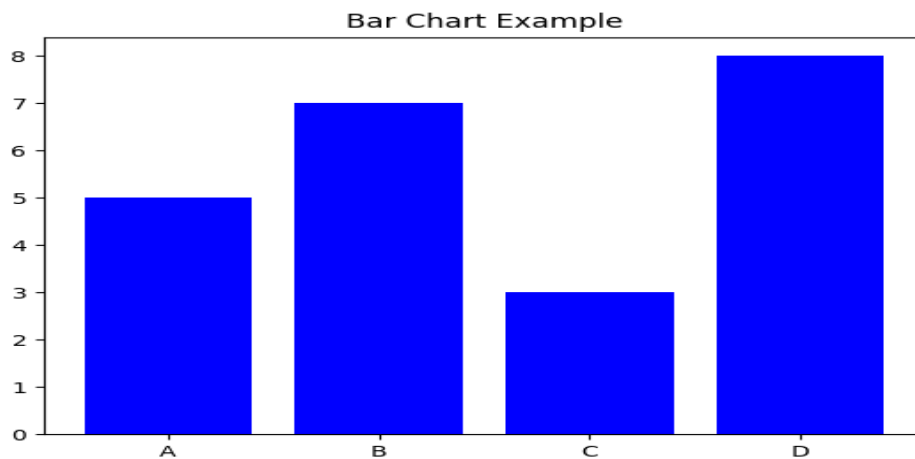
```
plt.bar(categories, values, color='blue')
```

```
plt.title("Bar Chart Example")
```

```
plt.show()
```



**Output:**



---

### 4. Histogram

**Use Case:** Show data distribution.

```
data = [12, 17, 15, 12, 18, 20, 22, 15, 17]
```

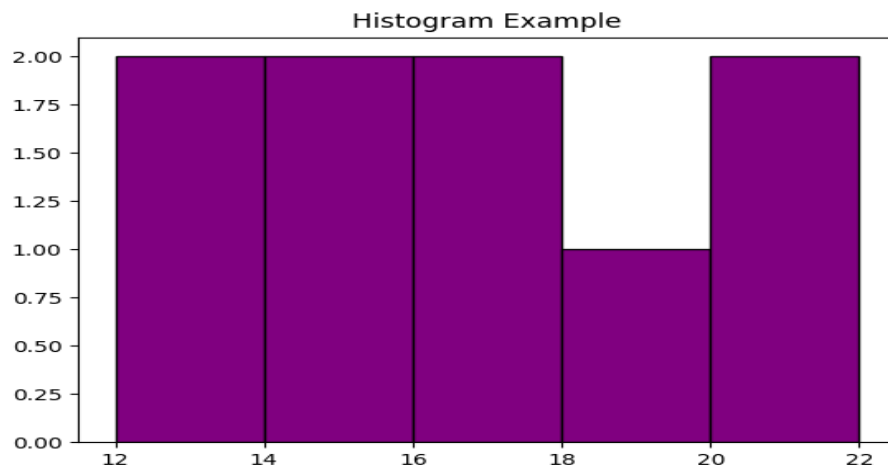
```
plt.hist(data, bins=5, color='purple', edgecolor='black')
```

```
plt.title("Histogram Example")
```

```
plt.show()
```



**Output:**



---

## 5. Pie Chart

**Use Case:** Show proportion of categories.

```
sizes = [30, 25, 20, 25]
```

```
labels = ['A', 'B', 'C', 'D']
```

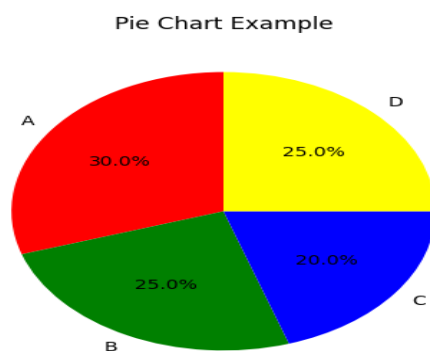
```
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90,  
colors=['red','green','blue','yellow'])
```

```
plt.title("Pie Chart Example")
```

```
plt.show()
```



**Output:**



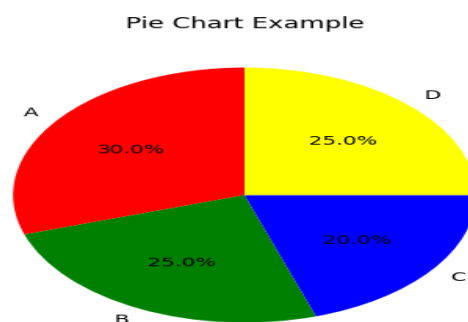
---

## ✓ Seaborn

```
import seaborn as sns  
  
import matplotlib.pyplot as plt  
  
# Load sample dataset  
  
data = sns.load_dataset("tips")
```



**Output:**

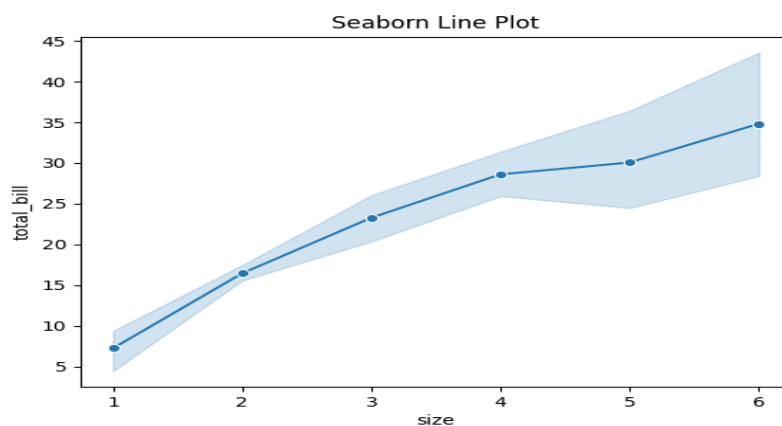


## 1. Line Plot

```
sns.lineplot(x='size', y='total_bill', data=data, marker='o')  
  
plt.title("Seaborn Line Plot")  
  
plt.show()
```



**Output:**



---

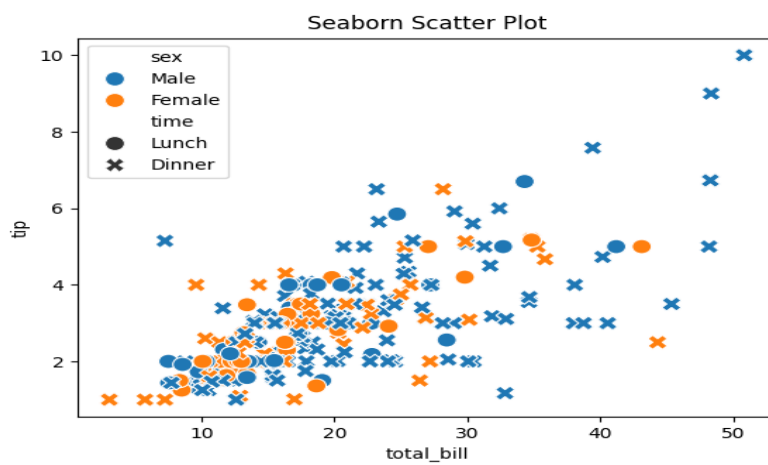
## 2. Scatter Plot

```
sns.scatterplot(x='total_bill', y='tip', hue='sex', style='time', data=data, s=100)

plt.title("Seaborn Scatter Plot")

plt.show()
```

 **Output:**



---

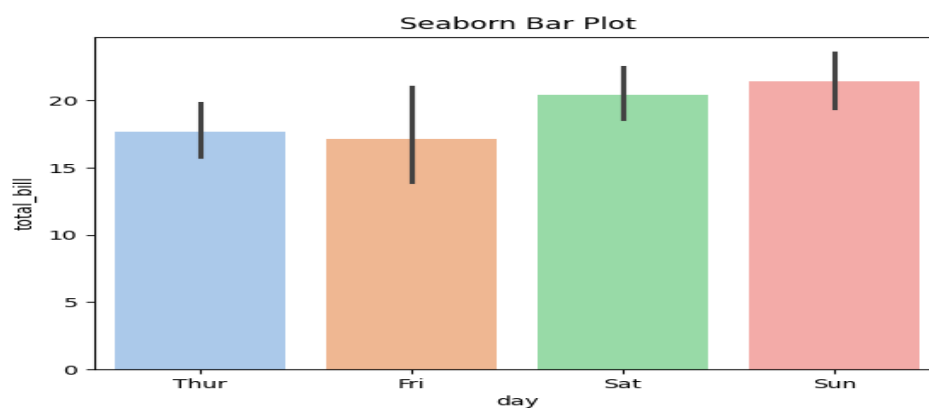
## 3. Bar Plot

```
sns.barplot(x='day', y='total_bill', data=data, palette='pastel')

plt.title("Seaborn Bar Plot")

plt.show()
```

 **Output:**



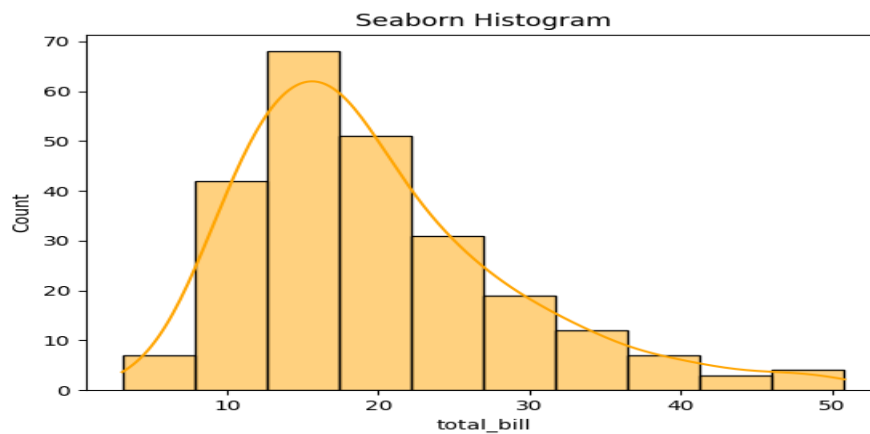
---

## 4. Histogram

```
sns.histplot(data['total_bill'], bins=10, kde=True, color='orange')  
  
plt.title("Seaborn Histogram")  
  
plt.show()
```



**Output:**



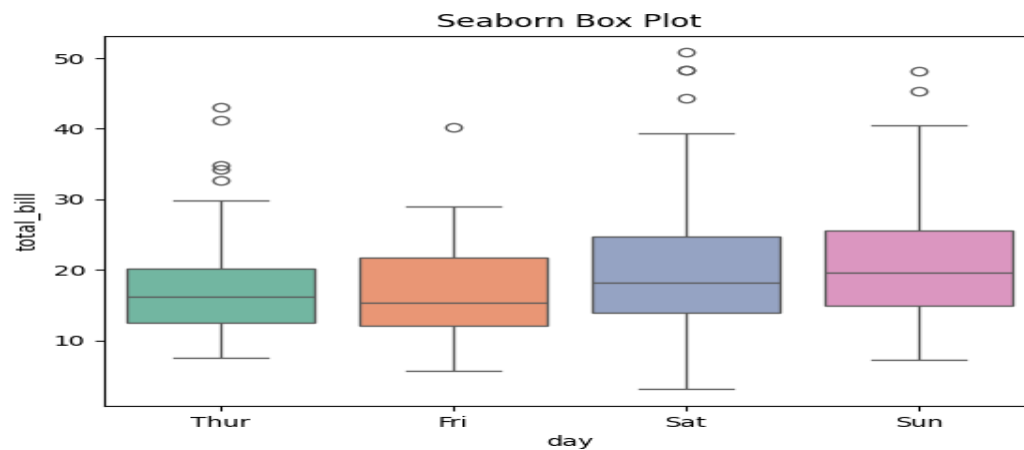
---

## 5. Box Plot

```
sns.boxplot(x='day', y='total_bill', data=data, palette='Set2')  
  
plt.title("Seaborn Box Plot")  
  
plt.show()
```



**Output:**



---

### 3. Comparison of Libraries

Feature	Matplotlib	Seaborn
Ease of Use	Medium	Easy
Customization	Very High	Medium
Default Styling	Basic	Attractive
Statistical Plots	Limited	Excellent
Interactivity	No	No
Large Dataset Handling	Good	Good

#### ✓ Summary:

- Use **Matplotlib** for complete control and detailed customization.
- Use **Seaborn** for attractive, high-level statistical plots with minimal code

---

#### ✓ Conclusion:

Mastering data visualization in Python is an essential skill for any aspiring data scientist. Through this task, we explored **Matplotlib** and **Seaborn**, two powerful libraries that enable both **customizable and aesthetically pleasing visualizations**.

---

### 4. References

1. Matplotlib Documentation: [https://matplotlib.org/stable/users/explain/quick\\_start.html](https://matplotlib.org/stable/users/explain/quick_start.html)
2. Seaborn Documentation: <https://seaborn.pydata.org/tutorial/introduction.html>
3. Plotly Python Distplot: <https://plotly.com/python/distplot/>
4. Bokeh User Guide: [https://docs.bokeh.org/en/latest/docs/user\\_guide/basic.html](https://docs.bokeh.org/en/latest/docs/user_guide/basic.html)
5. Pandas Visualization Guide: [https://pandas.pydata.org/docs/user\\_guide/index.html](https://pandas.pydata.org/docs/user_guide/index.html)
6. McKinney, W. *Python for Data Analysis*, 2nd Edition, O'Reilly Media, 2017.
7. VanderPlas, Jake. *Python Data Science Handbook*, O'Reilly Media, 2016.