📊 Python Visualization Guide: Matplotlib & Seaborn

🧰 1. Library Overview

Matplotlib

Matplotlib is the foundational plotting library in Python. It provides fine-grained control over plots and supports a wide range of chart types.

- **Best for**: Static visualizations
- **Use Cases**: Academic plots, reports, publications
- **Features**:
    - Highly customizable
    - Integrates with NumPy, Pandas
    - Generates static, animated, and interactive plots

Seaborn

Seaborn is built on top of Matplotlib and designed for statistical visualizations. It offers high-level interfaces for drawing attractive and informative graphics.

- **Best for**: Statistical plots and data exploration
- **Use Cases**: Data analysis, heatmaps, trend analysis
- **Features**:
    - Better default aesthetics
    - Simple syntax
    - Supports complex datasets (especially with Pandas)

📈 2. Graph Types with Examples

📌 Matplotlib Graph Types

1. Line Plot

- **Use Case**: Trends over time

python

CopyEdit

```
import matplotlib.pyplot as plt


x = [1, 2, 3, 4]
y = [10, 20, 25, 30]


plt.plot(x, y)
plt.title("Line Plot")
plt.xlabel("X Axis")
plt.ylabel("Y Axis")
plt.show()
```

2. Bar Chart

- **Use Case**: Comparing categorical data

python

CopyEdit

```
categories = ['A', 'B', 'C']
values = [4, 7, 1]


plt.bar(categories, values)
plt.title("Bar Chart")
plt.show()
```

3. Histogram

- **Use Case**: Distribution of data

python

CopyEdit

```
import numpy as np


data = np.random.randn(1000)
```

```
plt.hist(data, bins=30)
plt.title("Histogram")
plt.show()
```

4. Scatter Plot

- **Use Case**: Relationship between two variables

```
python
CopyEdit
x = [5, 7, 8, 7]
y = [99, 86, 87, 88]

plt.scatter(x, y)
plt.title("Scatter Plot")
plt.show()
```

5. Pie Chart

- **Use Case**: Proportions in a whole

```
python
CopyEdit
labels = ['Python', 'Java', 'C++', 'Ruby']
sizes = [215, 130, 245, 210]

plt.pie(sizes, labels=labels, autopct='%1.1f%%')
plt.title("Pie Chart")
plt.show()
```

📌 **Seaborn Graph Types**

1. Line Plot

- **Use Case**: Time series or trends

python

CopyEdit

```
import seaborn as sns
import pandas as pd

df = pd.DataFrame({"x": [1, 2, 3, 4], "y": [5, 6, 7, 8]})
sns.lineplot(data=df, x="x", y="y")
```

2. Bar Plot

- **Use Case**: Comparing values

python

CopyEdit

```
sns.barplot(x=["A", "B", "C"], y=[3, 7, 5])
```

3. Histogram (displot)

- **Use Case**: Distribution of a variable

python

CopyEdit

```
sns.displot([1, 1, 2, 2, 2, 3, 4, 4])
```

4. Scatter Plot (relplot)

- **Use Case**: Correlation analysis

python

CopyEdit

```
tips = sns.load_dataset("tips")
sns.relplot(data=tips, x="total_bill", y="tip")
```

5. Box Plot

- **Use Case**: Visualizing spread and outliers

python

CopyEdit

```
sns.boxplot(data=tips, x="day", y="total_bill")
```

6. Heatmap

- **Use Case**: Correlation matrix or matrix values

python

CopyEdit

import numpy as np

data = np.random.rand(4, 4)

sns.heatmap(data, annot=True)

⚖️ 3. Comparison: Matplotlib vs Seaborn

Feature	Matplotlib	Seaborn
Ease of Use	Moderate – requires more code	Easy – higher-level functions
Customization	Excellent – full control	Good – built-in styles, some limitations
Aesthetics	Basic default styles	Beautiful by default
Interactivity	Limited (static by default)	Static – depends on Matplotlib
Performance	High – works well with large data	Medium – slower for very large datasets
Integration	Works with NumPy, Pandas	Works best with Pandas
Best for	Precise, customized visuals	Quick statistical insights

---

📚 4. Resources

- **Matplotlib Quick Start**: https://matplotlib.org/stable/users/explain/quick_start.html#quick-start
- **Seaborn Introduction**: https://seaborn.pydata.org/tutorial/introduction.html

---

✅ **Summary**

- **Choose Matplotlib** when you need fine control and custom layouts.
- **Choose Seaborn** when you want quick, beautiful statistical plots using Pandas DataFrames.