**Module 8: Data Visualization using Matplotlib and Seaborn**

We use **Matplotlib** and **Seaborn** in Python primarily for **data visualization**. Both libraries help you understand data better by creating plots, charts, and graphs — but they have different strengths.

**✅ Why We Use Them:**

* **Matplotlib**
* It's the **foundation** of data visualization in Python.
* Allows you to create **basic to advanced plots** (line, bar, scatter, histogram, etc.).
* Highly **customizable** (control every element of a plot).
* Works well with **NumPy** and **Pandas**.
* **Seaborn**
* Built **on top of Matplotlib**.
* Designed for **statistical visualizations**.
* Works very well with **Pandas DataFrames**.
* Has **beautiful default styles** and **easier syntax** for complex plots (e.g., boxplots, heatmaps, violin plots).
* **Line plots**

Create a new Python file, e.g., line\_plot\_example.py

import matplotlib.pyplot as plt

# Sample data

x = [1, 2, 3, 4, 5]

y = [10, 20, 25, 30, 40]

# Create line plot

plt.plot(x, y, marker='o', linestyle='-', color='blue', label='Sales')

# Add labels and title

plt.xlabel("Days")

plt.ylabel("Revenue")

plt.title("Sales Over Time")

plt.legend()

# Show the plot

plt.show()

A graph with a line

AI-generated content may be incorrect.

* **Multiline plots**

import matplotlib.pyplot as plt

# Sample data

days = [1, 2, 3, 4, 5]

product\_A\_sales = [10, 15, 20, 25, 30]

product\_B\_sales = [5, 10, 15, 10, 20]

product\_C\_sales = [12, 18, 22, 28, 35]

# Plotting multiple lines

plt.plot(days, product\_A\_sales, label="Product A", marker='o')

plt.plot(days, product\_B\_sales, label="Product B", marker='s')

plt.plot(days, product\_C\_sales, label="Product C", marker='^')

# Add labels, title, and legend

plt.xlabel("Days")

plt.ylabel("Sales")

plt.title("Sales Comparison Over Days")

plt.legend()

# Optional: Add grid

plt.grid(True)

# Show plot

plt.show()

A graph with different colored lines and numbers

AI-generated content may be incorrect.

* **Bar plot**

import matplotlib.pyplot as plt

# Sample data

products = ['Product A', 'Product B', 'Product C', 'Product D']

sales = [150, 200, 180, 130]

# Create a bar plot

plt.bar(products, sales, color='skyblue')

# Add labels and title

plt.xlabel('Products')

plt.ylabel('Sales')

plt.title('Sales by Product')

# Show the plot

plt.show()

A graph of blue bars

AI-generated content may be incorrect.

* **Histogram**

import matplotlib.pyplot as plt

# Sample data: student marks

marks = [45, 55, 65, 70, 75, 80, 85, 90, 92, 95, 45, 55, 60, 65, 70, 75, 80]

# Create histogram

plt.hist(marks, bins=5, color='orange', edgecolor='black')

# Add labels and title

plt.xlabel('Marks Range')

plt.ylabel('Number of Students')

plt.title('Distribution of Student Marks')

# Show plot

plt.show()

A graph of a number of students

AI-generated content may be incorrect.

* **Pie chart**

import matplotlib.pyplot as plt

# Sample data

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

sales = [250, 150, 100, 200]

# Create pie chart

plt.pie(sales, labels=labels, autopct='%1.1f%%', startangle=90)

# Add title

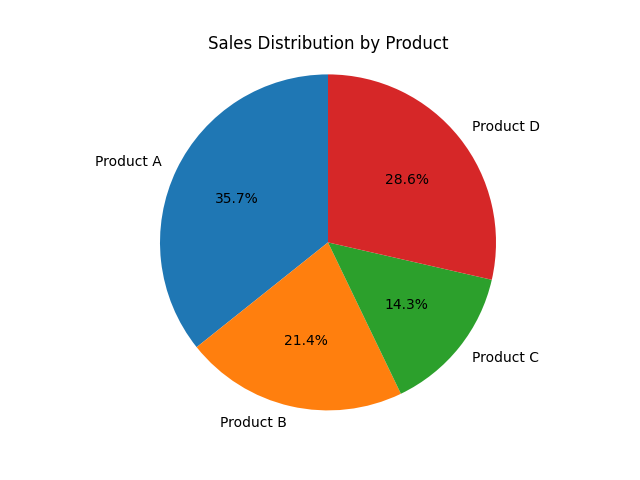
plt.title('Sales Distribution by Product')

# Equal aspect ratio ensures the pie is drawn as a circle.

plt.axis('equal')

# Show plot

plt.show()

****

* **Scatter plot**

import matplotlib.pyplot as plt

# Sample data

x = [5, 7, 8, 7, 6, 9, 5, 8, 7, 9]

y = [99, 86, 87, 88, 100, 86, 103, 87, 94, 78]

# Create scatter plot

plt.scatter(x, y, color='blue', marker='o', s=100, alpha=0.7)

# Labels and title

plt.xlabel("X Values")

plt.ylabel("Y Values")

plt.title("Simple Scatter Plot Example")

# Show grid for better readability

plt.grid(True)

# Display plot

plt.show()

A graph with blue dots

AI-generated content may be incorrect.

* **Boxplot**

import matplotlib.pyplot as plt

# Sample data

data = [7, 8, 5, 6, 9, 12, 15, 14, 8, 7, 6, 9, 10, 12, 11]

# Create boxplot

plt.boxplot(data)

# Labels and title

plt.xlabel("Dataset")

plt.ylabel("Values")

plt.title("Simple Boxplot Example")

# Show grid for clarity

plt.grid(True)

# Display the plot

plt.show()

A graph with lines and a line in the center

AI-generated content may be incorrect.

* **Saving charts**

In both **Matplotlib** and **Seaborn**, you can save charts to files (PNG, JPG, PDF, SVG, etc.) using the same underlying Matplotlib function:

import seaborn as sns

import matplotlib.pyplot as plt

import pandas as pd

# Sample DataFrame

df = pd.DataFrame({"x": [1, 2, 3, 4], "y": [10, 20, 25, 30]})

# Plot

sns.lineplot(data=df, x="x", y="y")

plt.title("Seaborn Example")

# Save chart

plt.savefig("seaborn\_chart.png", dpi=300, bbox\_inches='tight')

plt.show()

**Tips for Saving**

dpi=300 → High resolution (good for print).

bbox\_inches='tight' → Trims extra whitespace around the chart.

Change file extension (.jpg, .pdf, .svg) to save in different formats.

Save before plt.show() — otherwise, the figure might reset.

* **Grids**

In Matplotlib and Seaborn, grids help make charts easier to read by showing horizontal/vertical reference lines.

import matplotlib.pyplot as plt

x = [1, 2, 3, 4]

y = [10, 20, 25, 30]

plt.plot(x, y, marker='o')

# Add grid

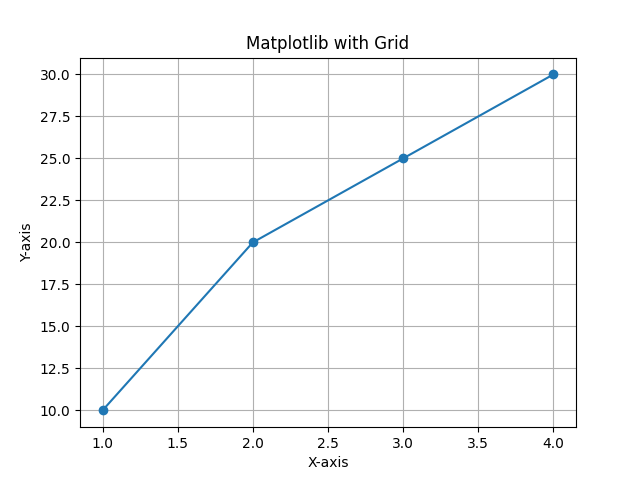
plt.grid(True) # Default grid

plt.title("Matplotlib with Grid")

plt.xlabel("X-axis")

plt.ylabel("Y-axis")

plt.show()



Customizing the Grid

plt.grid(

color='gray', # Grid line color

linestyle='--', # Dashed lines

linewidth=0.7, # Line thickness

alpha=0.7 # Transparency

)

You can also control which axis shows the grid:

plt.grid(axis='x') # Only vertical grid lines

plt.grid(axis='y') # Only horizontal grid lines

* **Subplots**

Subplots let you display multiple charts in one figure.

import matplotlib.pyplot as plt

x = [1, 2, 3, 4]

y1 = [10, 20, 25, 30]

y2 = [5, 15, 20, 25]

# Create figure and subplots (1 row, 2 columns)

fig, ax = plt.subplots(1, 2, figsize=(10, 4))

# First subplot

ax[0].plot(x, y1, marker='o')

ax[0].set\_title("Line Chart 1")

ax[0].grid(True)

# Second subplot

ax[1].bar(x, y2)

ax[1].set\_title("Bar Chart 2")

ax[1].grid(True)

plt.tight\_layout() # Adjust spacing

plt.show()

A graph and chart with blue bars

AI-generated content may be incorrect.

**Subplots with Seaborn**

import seaborn as sns

import pandas as pd

import matplotlib.pyplot as plt

# Sample DataFrame

df = pd.DataFrame({

"x": [1, 2, 3, 4],

"y1": [10, 20, 25, 30],

"y2": [5, 15, 20, 25]

})

# Create subplots

fig, ax = plt.subplots(1, 2, figsize=(10, 4))

# First subplot (line)

sns.lineplot(data=df, x="x", y="y1", ax=ax[0])

ax[0].set\_title("Line Chart")

ax[0].grid(True)

# Second subplot (bar)

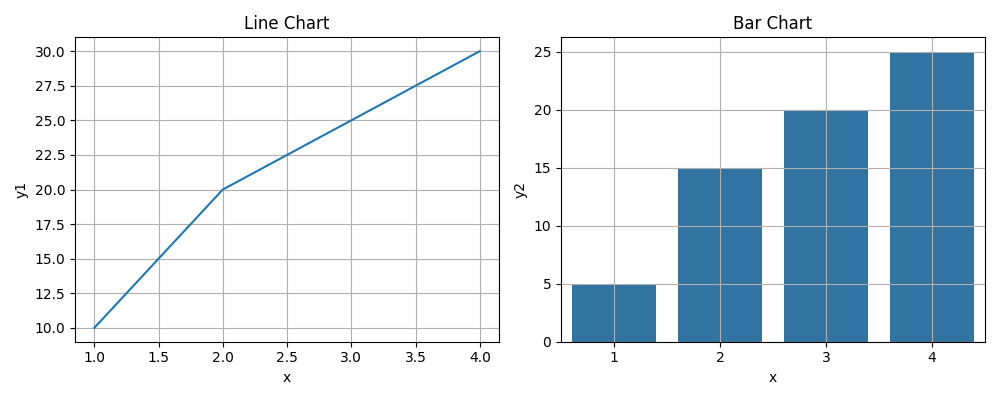
sns.barplot(data=df, x="x", y="y2", ax=ax[1])

ax[1].set\_title("Bar Chart")

ax[1].grid(True)

plt.tight\_layout()

plt.show()



* **Heatmaps**

**Heatmap with Seaborn (Correlation Matrix Example)**

import seaborn as sns

import pandas as pd

import matplotlib.pyplot as plt

# Sample Data

df = pd.DataFrame({

"A": [1, 2, 3, 4],

"B": [4, 3, 2, 1],

"C": [10, 20, 25, 30]

})

# Compute correlation matrix

corr = df.corr()

# Create heatmap

sns.heatmap(corr, annot=True, cmap="coolwarm", linewidths=0.5)

plt.title("Correlation Heatmap")

plt.show()

A diagram of heatmap

AI-generated content may be incorrect.