Data Visualization

Turning Data into Insights

matplotlib

Dated: 30 July 2025

Instructor : Fawad Bahadur

Github: bahadurCorvit

Introduction to Data

What is Data?

Data is raw facts and figures collected for analysis and decision-making.

Types of Data:

1. Structured Data

Organized in rows and columns

Example: Excel sheets, SQL databases

2. Unstructured Data

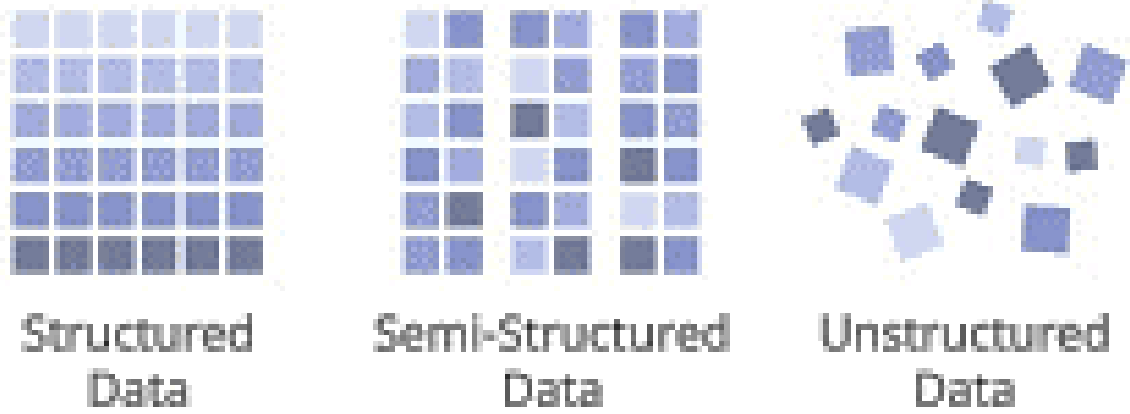
No predefined format

Example: Images, videos, emails, social media posts

3. Semi-structured Data

Partially organized

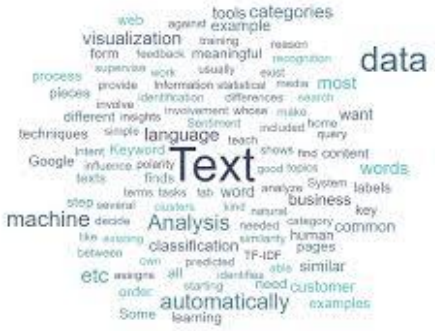
Example: JSON, XML files

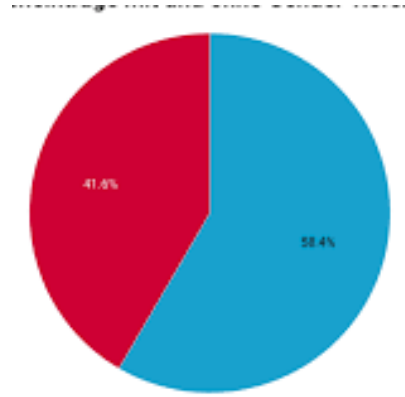


Instructor : Fawad Bahadur

Conti...

Common Data Format	Example
Text	Tweets, Documents
Numerical	Sales Figures, Temperature
Categorical	Gender, Country
Time Series	Stock Prices over time
Multimedia	Photo, Audio Recordings





Instructor : Fawad Bahadur

Where Does Data Come From?

1. Internal Sources

- Company databases
- Transaction records
- CRM systems
- Employee feedback

2. External Sources

- Social media platforms (Facebook, Twitter)
- Public datasets (government, research organizations)
- Web scraping
- Third-party data providers

3. Sensors and IoT Devices

- Smart devices
- Wearables
- Environmental sensors

4. Surveys and Questionnaires

- Customer feedback
- Market research

5. Multimedia Sources

- Images, videos, audio recordings
- Satellite imagery

Why is Data Important?

1. Informed Decision-Making

- Supports business, healthcare, education & policy decisions
- Enables data
- Driven strategies

2. Improved Efficiency

- Identifies patterns to optimize processes
- Reduces cost & time through automation

3. Personalization

- Powers recommendation systems (e.g., Netflix, Amazon)
- Enhances user experience

4. Innovation & Growth

- Drives AI, machine learning, and product development
- Enables predictive analytics

5. Competitive Advantage

- Companies leveraging data outperform competitors
- Helps in trend prediction and market insights

Data Visualization and Its Importance

What is Data Visualization?

The graphical representation of information a
data using charts, graphs, maps, and dashboar

Data Visualization Important

1. Simplifies Complex Data
2. Reveals Patterns and Trends
3. Improves Decision-Making
4. Enhances Communication
5. Supports Data Storytelling

Common Data Visualization Tools

1. Matplotlib (Python)

- Basic plotting library
- Highly customizable
- Great for line, bar, scatter plots

2. Seaborn (Python)

- Built on top of Matplotlib
- Beautiful statistical plots
- Works well with pandas DataFrames

3. Plotly (Python/JS)

- Interactive and web-ready plots
- 3D, animated and dashboard capabilities

4. Tableau

- Drag-and-drop visual analytics tool
- Ideal for business dashboards and reports

5. Power BI

- Microsoft's BI platform
- Integrates well with Excel and Azure

6. Excel

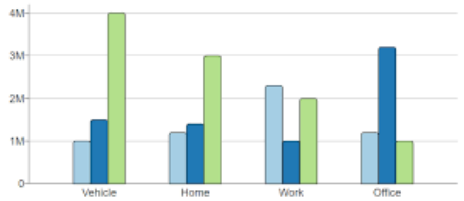
- Basic visualizations for quick analysis
- Widely used in business settings

Types of Data Visualizations

1. Bar Chart

Compare values across categories

Example: Sales by region



2. Line Chart

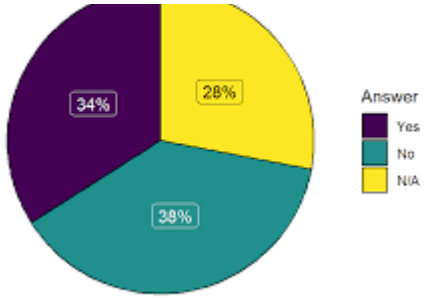
Show trends over time

Example: Monthly temperature

3. Pie Chart

Show parts of a whole

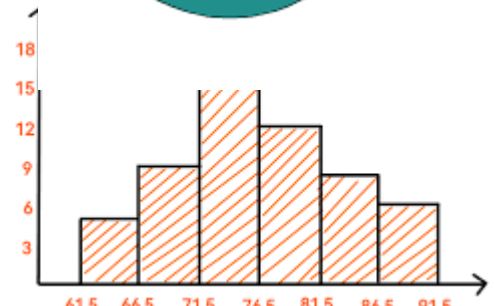
Example: Market share by company



4. Histogram

Distribution of numeric data

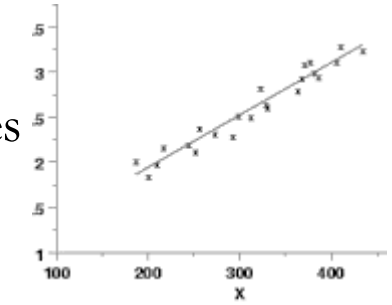
Example: Exam score frequency



5. Scatter Plot

Show relationships between two variables

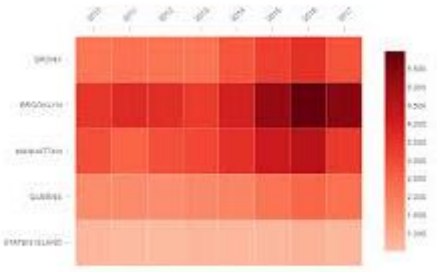
Example: Height vs. weight



6. Heatmap

Visualize correlations or intensity

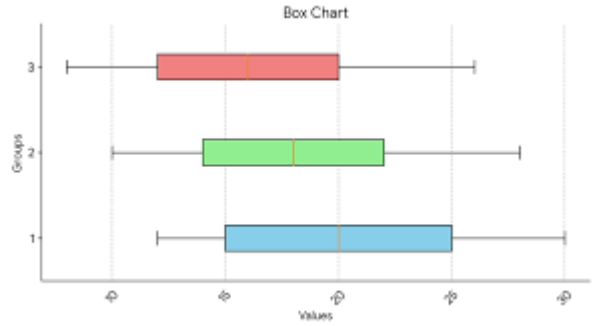
Example: Correlation matrix



7. Box Plot

Summary of distribution (median, quartiles, outliers)

Example: Salary distribution



Introduction to Matplotlib

What is Matplotlib?

Matplotlib is a popular Python library used for creating static, animated, and interactive visualizations.

Key Features:

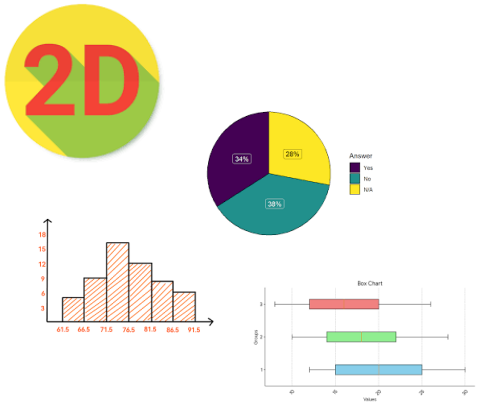
- 2D plotting library
- Supports line, bar, scatter, pie, histogram, and more
- Highly customizable (colors, labels, styles)
- Works well with NumPy and pandas

Use Cases:

- Data exploration and reporting
- Visual summaries in machine learning
- Creating dashboards and reports

Installation

```
pip install matplotlib
conda install matplotlib
```



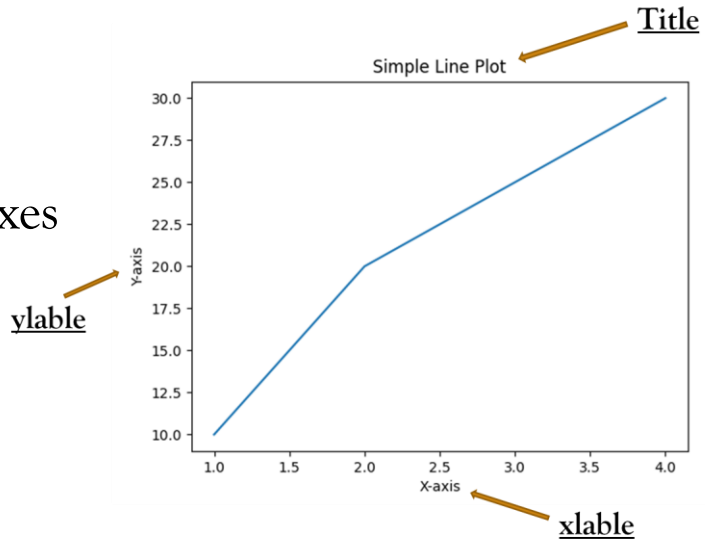
Example

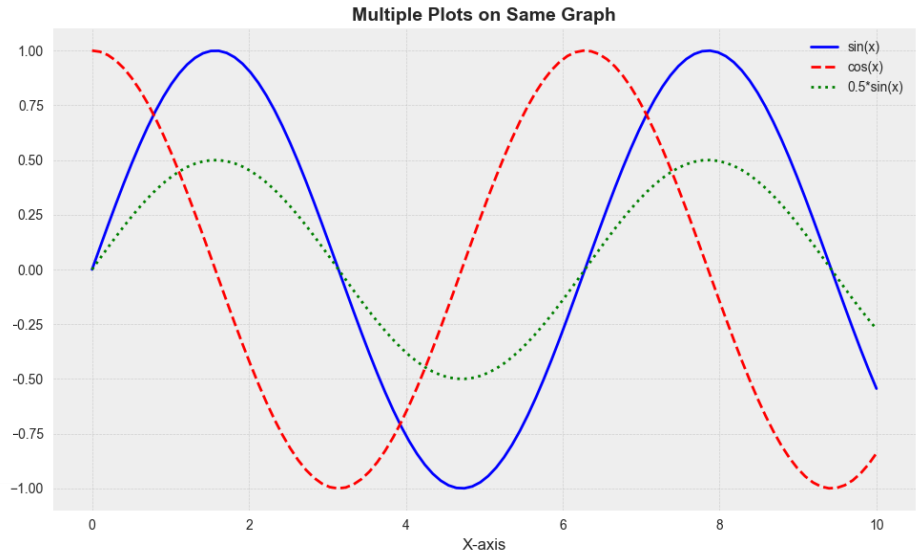
```
import matplotlib.pyplot as plt

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

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

Creating a Simple Line Plot

Key Elements of a Plot

- plt.plot() – Draws the line graph
- plt.title() – Adds a title
- plt.xlabel() / plt.ylabel() – Labels axes
- plt.show() – Displays the plot





Code

Python

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [10, 20, 15, 25, 30]
plt.plot(x, y)
plt.title("Basic Line Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

Tip

- Matplotlib also supports multiple plots on the same graph using plt.plot() multiple times before plt.show().

Customizing Plots in Matplotlib

1. Changing Colors, Line Styles & Markers

Python

```
plt.plot(x, y, color='green', linestyle='--', marker='o')
```

- color = 'red', 'blue', 'green', etc.
- linestyle = '-', '--', '-.', ':'
- marker = 'o' (circle), 's' (square), '*' (star), etc.

Code

```
plt.plot(x, y, color='purple', linestyle='-.', marker='s')
plt.title("Customized Line Plot")
plt.xlabel("Time")
plt.ylabel("Value")
plt.grid(True)
plt.legend(["Trend"])
plt.show()
```

2. Adding Grid & Legend

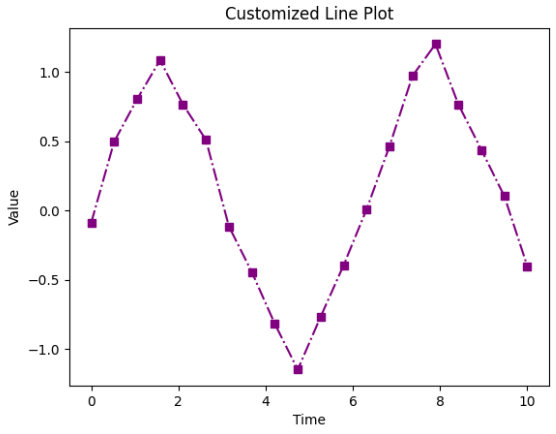
Python

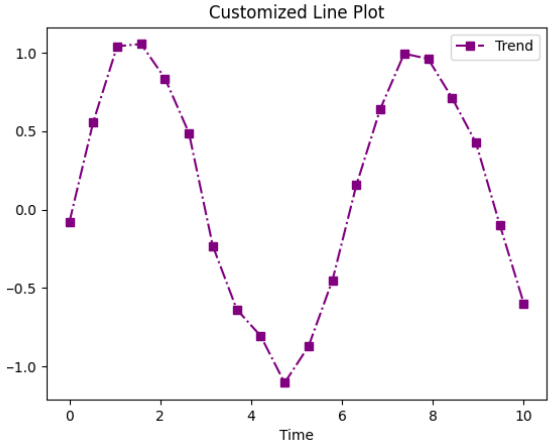
```
plt.grid(True)
plt.legend(["Sales"])
```

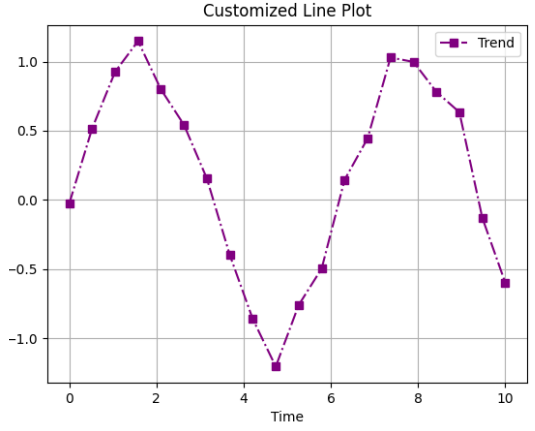
3. Adjusting Figure Size

Python

```
plt.figure(figsize=(8, 5))
```





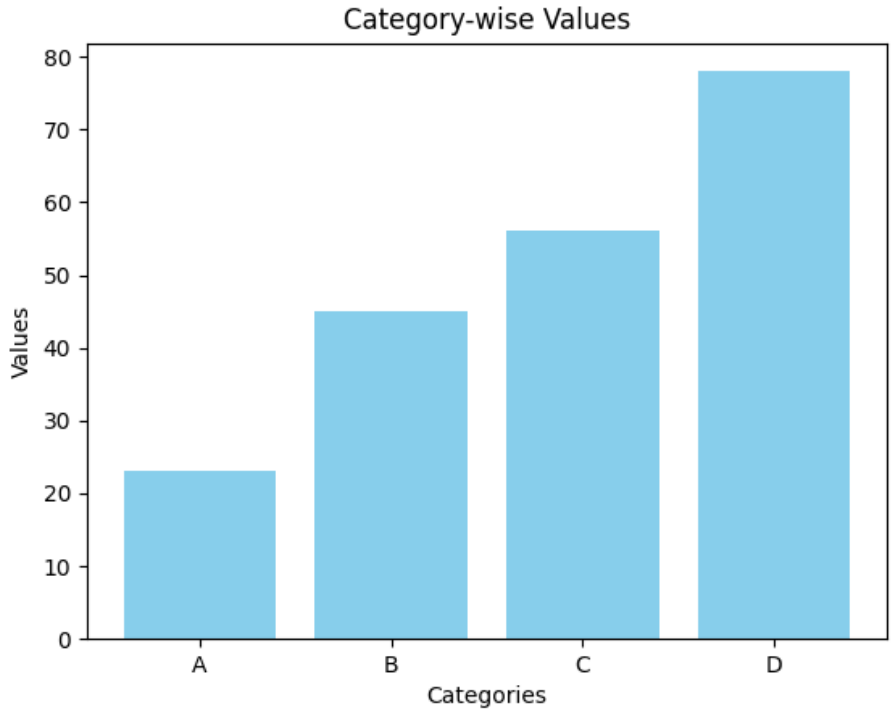


Creating Bar Charts in Matplotlib

1. Vertical Bar Chart

Python

```
import matplotlib.pyplot as plt

categories = ['A', 'B', 'C', 'D']
values = [23, 45, 56, 78]

plt.bar(categories, values, color='skyblue')
plt.title("Category-wise Values")
plt.xlabel("Categories")
plt.ylabel("Values")
plt.show()
```



Use Case

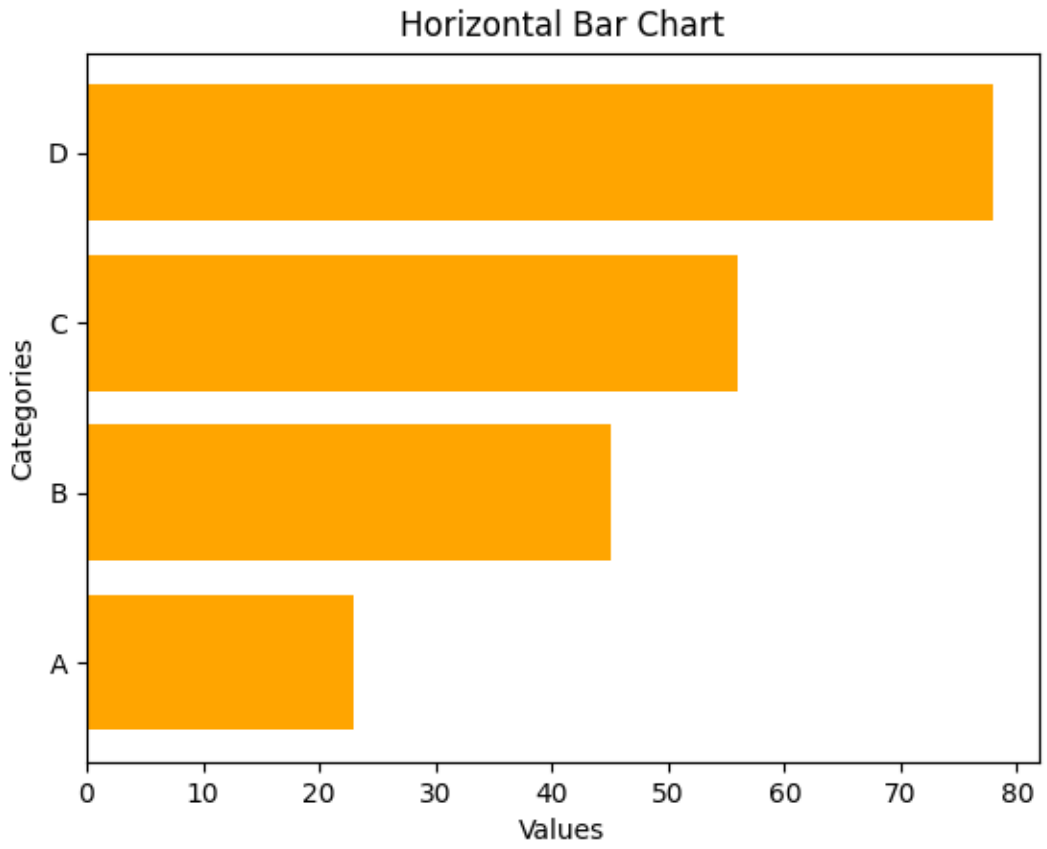
Compare quantities across discrete categories (e.g., sales per product)

Contin...

2. Horizontal Bar Chart

Python

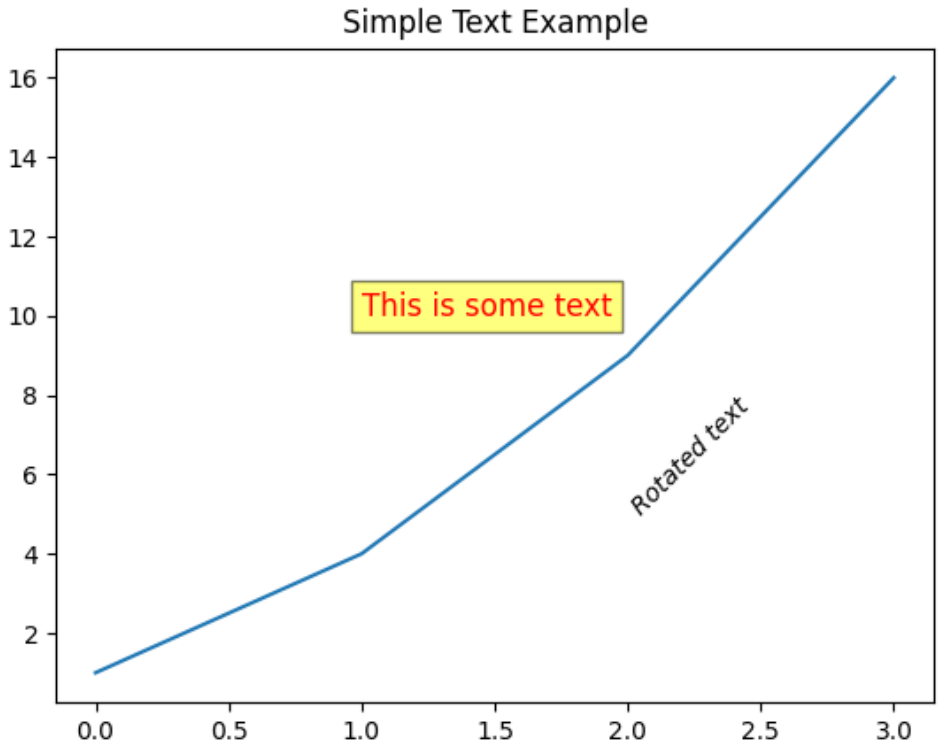
```
categories = ['A', 'B', 'C', 'D']
values = [23, 45, 56, 78]

plt.barh(categories, values, color='orange')
plt.title("Horizontal Bar Chart")
plt.xlabel("Values")
plt.ylabel("Categories")
plt.show()
```



Creating Bar Charts in Matplotlib

3. Customization Options

Change bar color, width, and alignment
Add value labels using plt.text()



Use Case

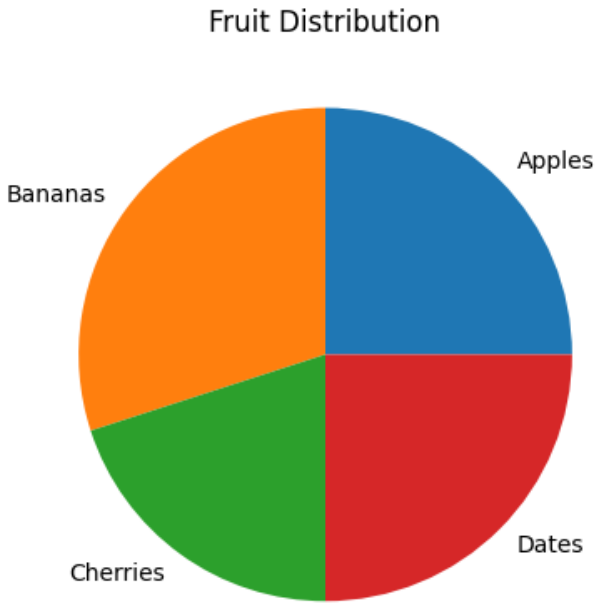
Compare quantities across discrete categories (e.g., sales per product)

Creating Pie Charts in Matplotlib

1. Basic Pie Chart

Python

```
import matplotlib.pyplot as plt

labels = ['Apples', 'Bananas', 'Cherries', 'Dates']
sizes = [25, 30, 20, 25]

plt.pie(sizes, labels=labels)
plt.title("Fruit Distribution")
plt.show()
```



Use Cases

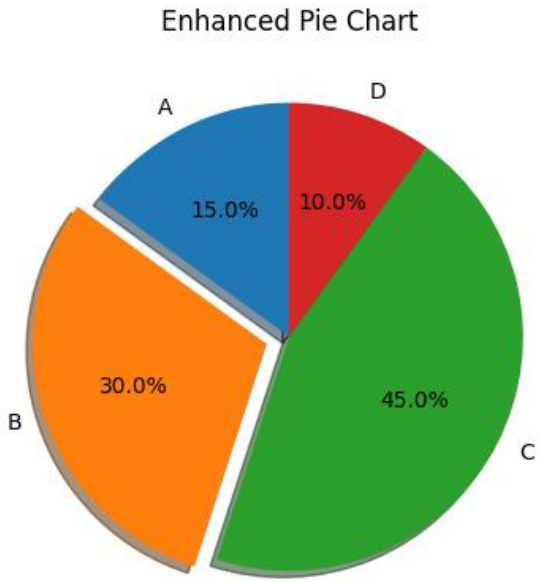
Represent part-to-whole relationships (percentages, shares)

Creating Pie Charts in Matplotlib

2. Adding Customization

Python

```
plt.pie(
    sizes,
    labels=labels,
    autopct='%1.1f%%',      # Show percentages
    startangle=90,          # Start from top
    explode=[0, 0.1, 0, 0],  # Emphasize one slice
shadow=True)
plt.title("Enhanced Pie Chart")
plt.show()
```



Use Cases

Represent part-to-whole relationships (percentages, shares)

Creating Histograms in Matplotlib

1. 1. What is a Histogram?

A histogram shows the distribution of a numeric variable by grouping data into bins.

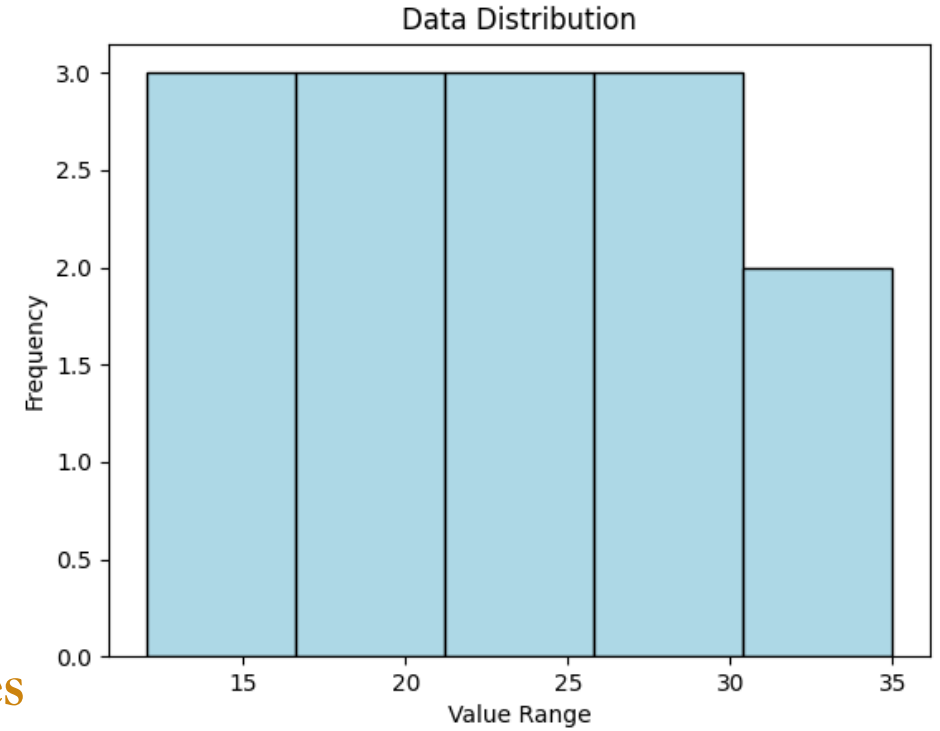
2. Basic Histogram

Python

```
import matplotlib.pyplot as plt

data = [12, 15, 13, 18, 19, 21, 22, 23, 25, 26, 28, 30, 32, 35]
plt.hist(data, bins=5, color='skyblue', edgecolor='black’)
plt.title("Data Distribution")
plt.xlabel("Value Range")
plt.ylabel("Frequency")
plt.show()
```

3. Customization Options

- bins: Number of intervals
- color & edgecolor: Visual styling
- Add grid, mean lines, etc. for clarity



Use Cases

Understand data spread, detect skewness, spot outliers

Creating Scatter Plots in Matplotlib

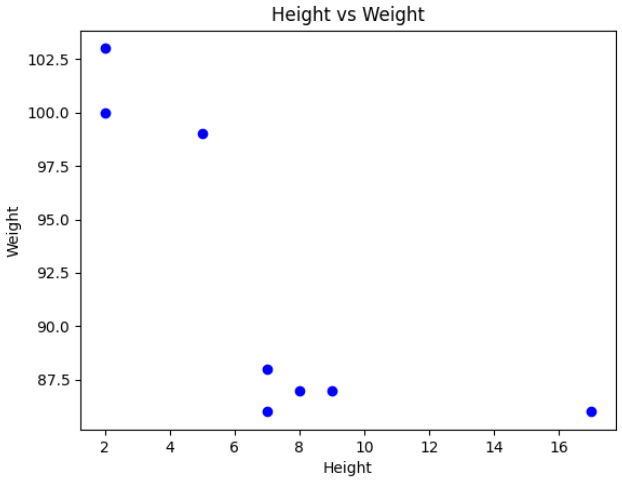
1. What is a Scatter Plot?

A scatter plot displays values for two variables as points on a 2D plane. It shows relationships, patterns, or correlations.

2. Basic Scatter Plot

Python

```
import matplotlib.pyplot as plt

x = [5, 7, 8, 7, 2, 17, 2, 9]
y = [99, 86, 87, 88, 100, 86, 103, 87]
plt.scatter(x, y, color='blue')
plt.title("Height vs Weight")
plt.xlabel("Height")
plt.ylabel("Weight")
plt.show()
```

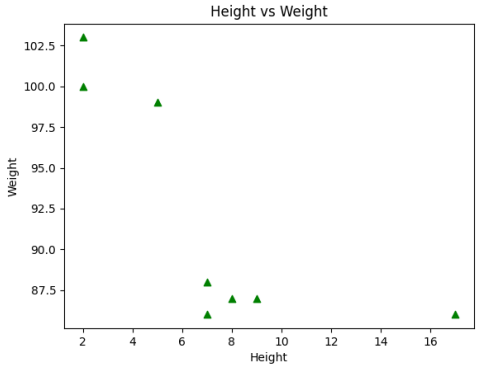


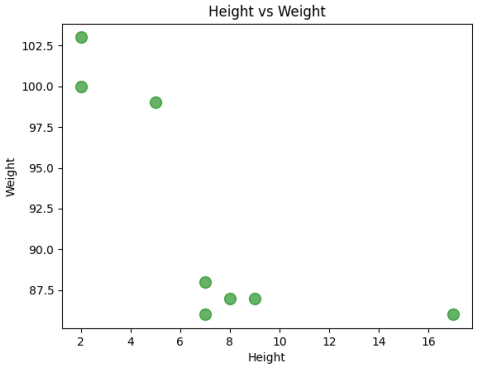
3. Customization Options

- **color:** e.g., 'red', 'green'
- **marker:** e.g., 'o', '^', 's'
- **size:** using s= parameter for point sizes
- **alpha:** for transparency

Python

```
plt.scatter(x, y, color='green', s=100, alpha=0.6)
```





Use Cases

Visualizing relationships between variables (e.g., age vs income)

Creating Subplots in Matplotlib

1. What are Subplots?

Subplots allow multiple plots to be displayed in a single figure.

2. Using plt.subplot()

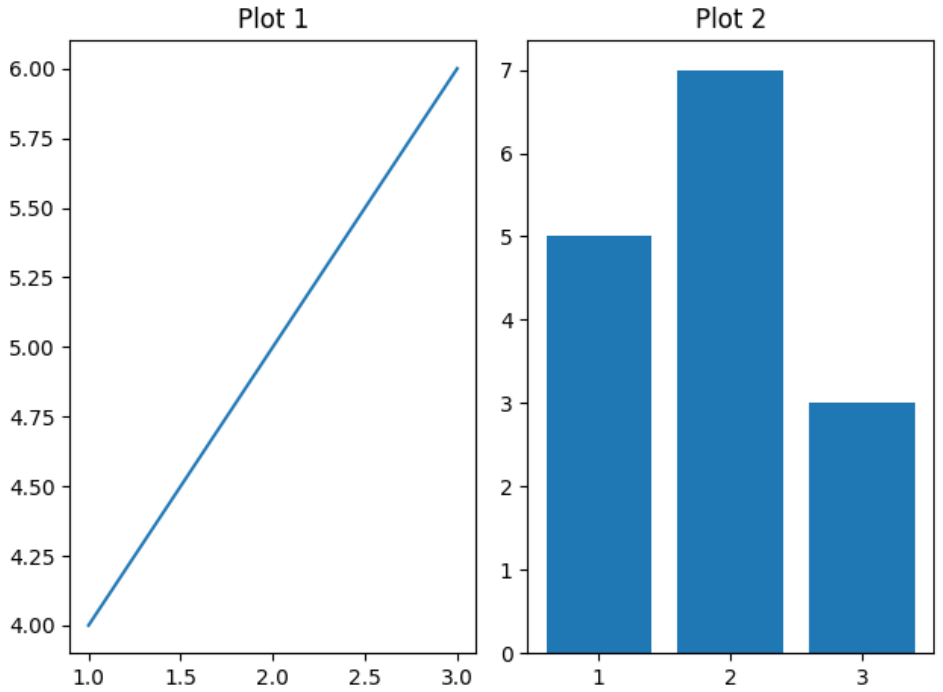
Python

```
import matplotlib.pyplot as plt
```

```
import matplotlib.pyplot as plt
plt.subplot(1, 2, 1)  # 1 row, 2 columns, 1st plot
plt.plot([1, 2, 3], [4, 5, 6])
plt.title("Plot 1")
plt.subplot(1, 2, 2)  # 2nd plot
plt.bar([1, 2, 3], [5, 7, 3])
plt.title("Plot 2")
plt.tight_layout()  # Adjust spacing
plt.show()
```

3. Grid Structure

- plt.subplot(rows, columns, index)
- Index starts at 1



Use Cases

Compare multiple visualizations side by side

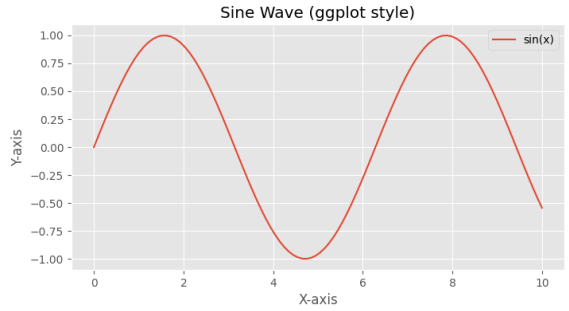
Plot Styling & Themes in Matplotlib

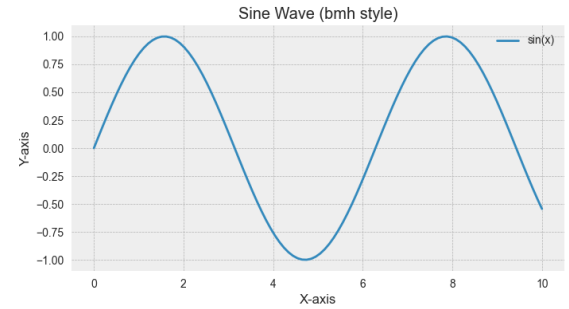
1. Built-in Styles

Apply a pre-defined style to your plots:

Python

```
plt.style.use('ggplot')
```





Popular styles:-

- 'ggplot'
- 'seaborn'
- 'bmh'
- 'dark_background'
- 'fivethirtyeight'

2. Customize Fonts & Sizes

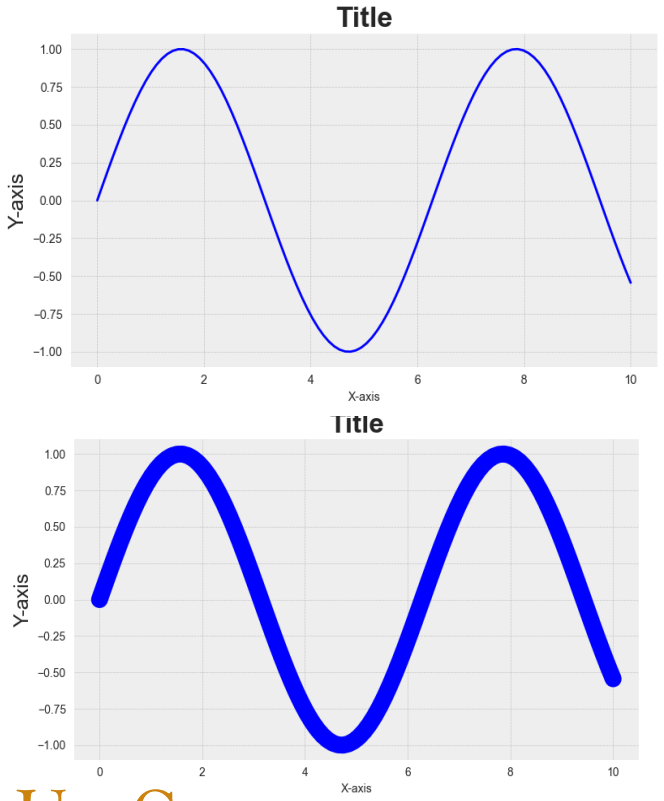
Python

- `plt.title("Title", fontsize=14, fontweight='bold')`
- `plt.xlabel("X-axis", fontsize=12)`
- `plt.ylabel("Y-axis", fontsize=12)`

3. Line and Marker Customization

Python

```
plt.plot(x,
         y,
         color='purple',
         linewidth=2,
         marker='o',
         markersize=8)
```



Use Case

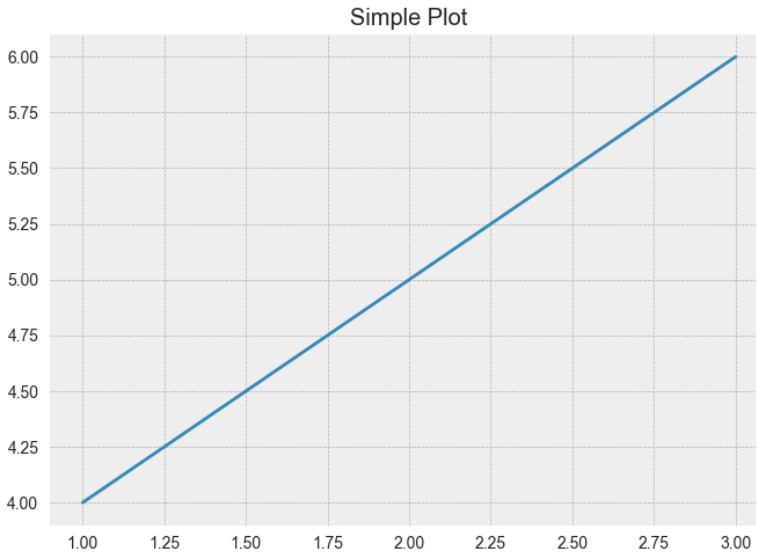
Enhance readability, match publication or brand styles.

Saving Plots to Files in Matplotlib--

1. Save a Plot as an Image File

Python

```
plt.plot([1, 2, 3], [4, 5, 6])
plt.title("Simple Plot")
plt.savefig("plot.png")  # Save as PNG
```



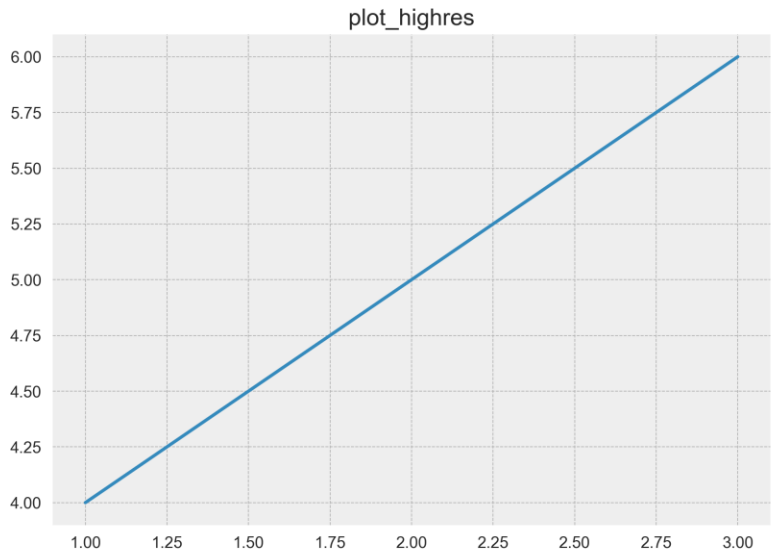
2. Supported File Formats

- .png – Most common
- .jpg, .jpeg – Compressed image
- .pdf – For high-quality printing
- .svg – Scalable vector graphics

3. Set Resolution (DPI)

Python

```
plt.savefig("plot_highres.png", dpi=300)
```



4. Save Without Displaying

- Call plt.savefig() before plt.show()
- Once plt.show() is used, the plot is cleared

Use Case

Include plots in reports, articles, or websites

Matplotlib Chart Comparison

Plot Type	Function	Use Case	Example Function
Line Plot	plot()	Trends over continuous data (e.g., time series)	plt.plot(x, y)
Scatter Plot	scatter()	Relationship between two variables	plt.scatter(x, y)
Bar Chart	bar()	Compare categories using rectangular bars	plt.bar(categories, values)
Pie Chart	pie()	Show proportion of categories as slices of a pie	plt.pie(sizes, labels=...)
Histogram	hist()	Frequency distribution of continuous data	plt.hist(data, bins=...)

THANK YOU