Visualization Library Documentation

Libraries : Pandas

Pandas is a powerful open-source Python library used for **data analysis and data manipulation**.
It provides easy-to-use data structures like:

- **Series** (1-D data)
- **DataFrame** (2-D tabular data)

Pandas is commonly used for:

- Data cleaning
- Data preparation
- Reading/writing data from CSV, Excel, SQL, JSON
- Statistical analysis
- Creating simple visualizations (built on top of Matplotlib)

Pandas makes it easy to create graphs directly from a DataFrame.

1. Line Plot:

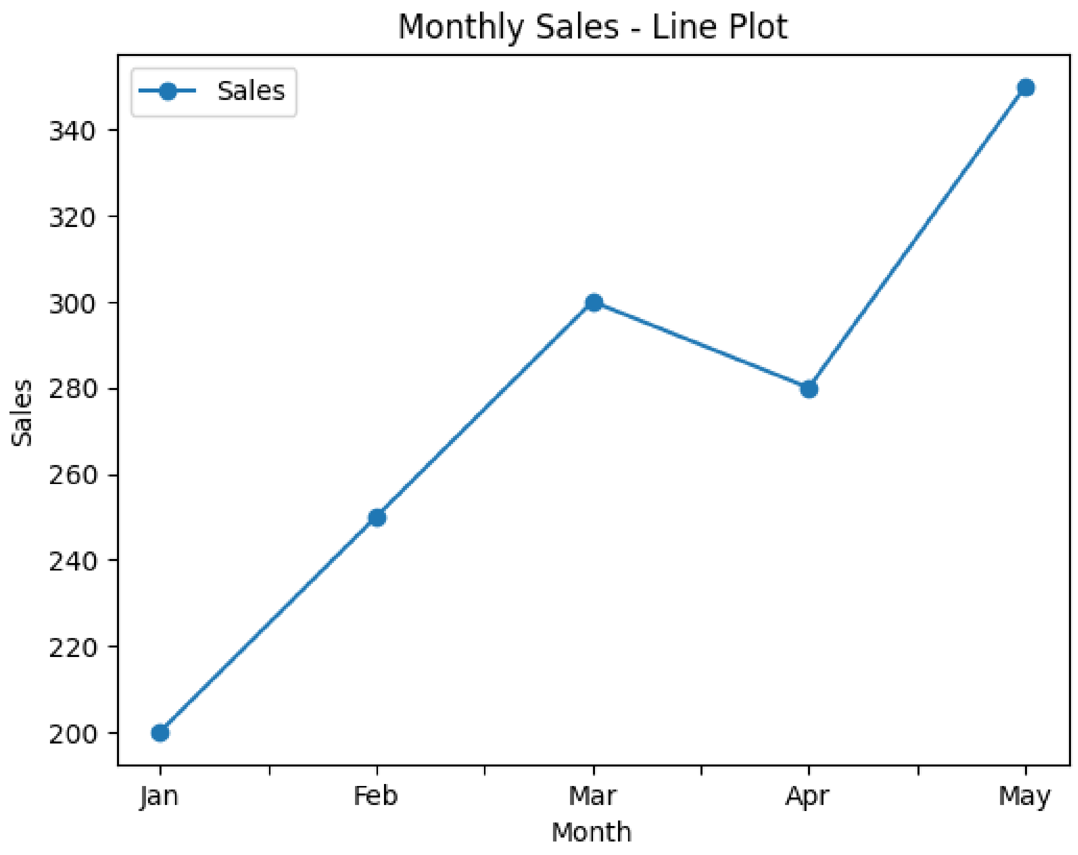
   A **line plot** displays data points connected by straight lines.
It is used to show **trends over time**.

Program:

```
        import pandas as pd
import matplotlib.pyplot as plt
data = {
   'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],
   'Sales': [200, 250, 300, 280, 350],
   'Profit': [50, 70, 65, 80, 90]
}
df = pd.DataFrame(data)
plt.figure(figsize=(4,4))
df.plot(x='Month', y='Sales', kind='line', marker='o')
plt.title('Monthly Sales - Line Plot')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.show()
```

Output:

```
<Figure size 400x400 with 0 Axes>
```



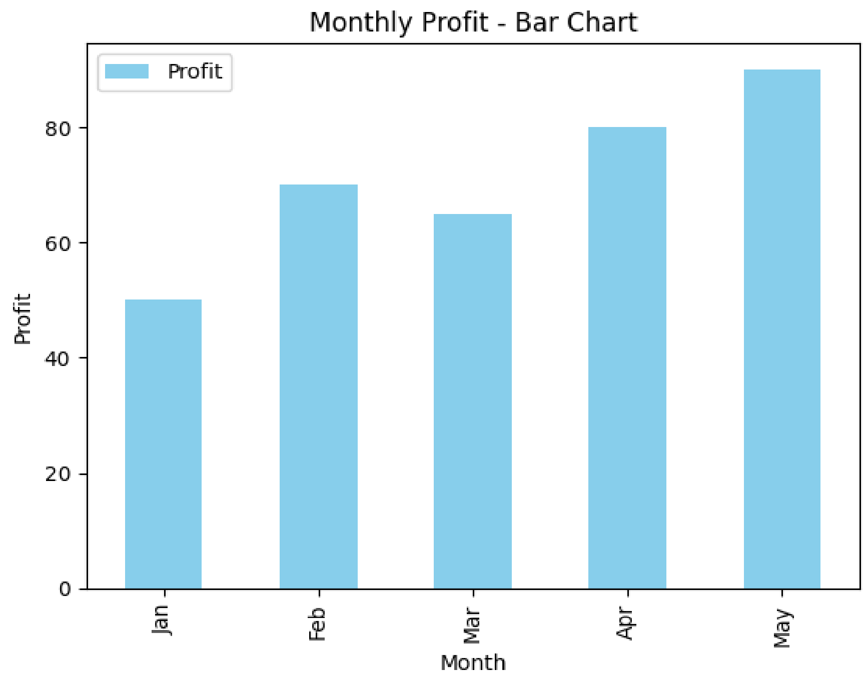
2. Bar Chart:

   A **bar chart** uses rectangular bars to represent data values.
   It compares **categories** or **groups**.

Program:

```
import pandas as pd

import matplotlib.pyplot as plt

data = {

    'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],

    'Sales': [200, 250, 300, 280, 350],

    'Profit': [50, 70, 65, 80, 90]

}

df = pd.DataFrame(data)

plt.figure(figsize=(6,4))

df.plot(x='Month', y='Profit', kind='bar', color='skyblue')

plt.title('Monthly Profit - Bar Chart')

plt.xlabel('Month')

plt.ylabel('Profit')

plt.show()
```

Output: <Figure size 600x400 with 0 Axes>



3. Histogram:

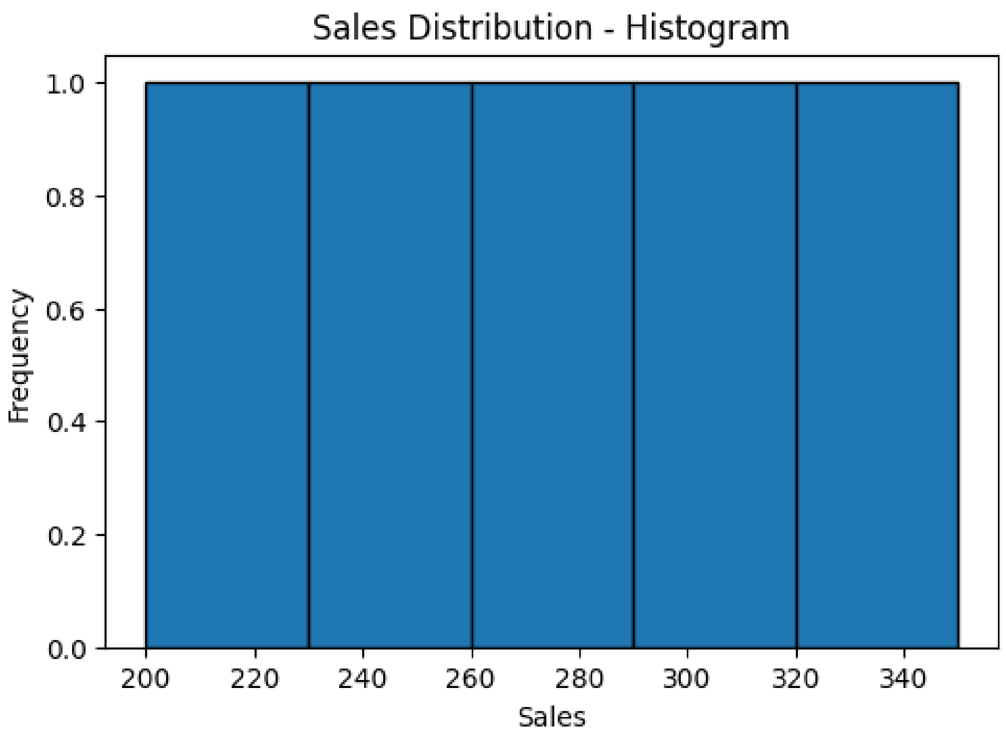
   A **histogram** shows the **distribution** of numerical data by dividing it into **bins**.

Program:

```
import pandas as pd

import matplotlib.pyplot as plt

data = {

   'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],

   'Sales': [200, 250, 300, 280, 350],

   'Profit': [50, 70, 65, 80, 90]

}

df = pd.DataFrame(data)

plt.figure(figsize=(6,4))
```

```
df['Sales'].plot(kind='hist', bins=5, edgecolor='black')
plt.title('Sales Distribution - Histogram')
plt.xlabel('Sales')
plt.show()
```

Output:



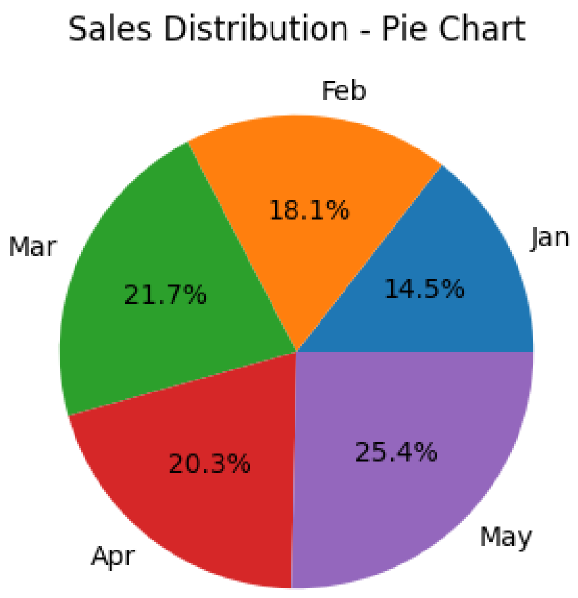
4. Pie Chart:

   A **pie chart** shows data as slices of a circle, representing **percentages or proportions**.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
data = {
    'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],
    'Sales': [200, 250, 300, 280, 350],
    'Profit': [50, 70, 65, 80, 90]
}
df = pd.DataFrame(data)
plt.figure(figsize=(6,4))
df.set_index('Month')['Sales'].plot(kind='pie', autopct='%1.1f%%')
plt.title('Sales Distribution - Pie Chart')
plt.ylabel('')
plt.show()
```

Output:



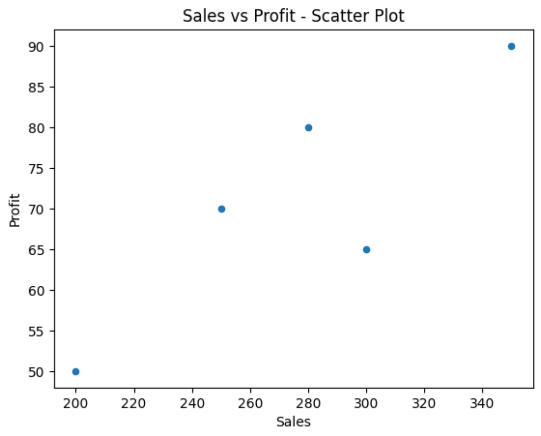
5. Scatter Plot

   A **scatter plot** displays points representing the relationship between **two numerical variables**.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
data = {
    'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],
    'Sales': [200, 250, 300, 280, 350],
    'Profit': [50, 70, 65, 80, 90]
}
df = pd.DataFrame(data)
plt.figure(figsize=(6,4))
df.plot(kind='scatter', x='Sales', y='Profit')
plt.title('Sales vs Profit - Scatter Plot')
plt.xlabel('Sales')
plt.ylabel('Profit')
plt.show()
```

Output:
```
<Figure size 200x200 with 0 Axes>
```



Library : Matplotlib

Matplotlib is the most popular and widely used Python library for data visualization.
It allows you to create high-quality **2D and 3D plots**, such as:

- Line plots
- Bar charts
- Histograms
- Pie charts
- Scatter plots
- Boxplots
- Heatmaps, etc.

Key Features

- Highly customizable (colors, labels, styles)
- Works well with NumPy and Pandas
- Supports simple to advanced plots
- Produces publication-quality figures
- Used in data science, machine learning, and scientific research

1. Line Plot:

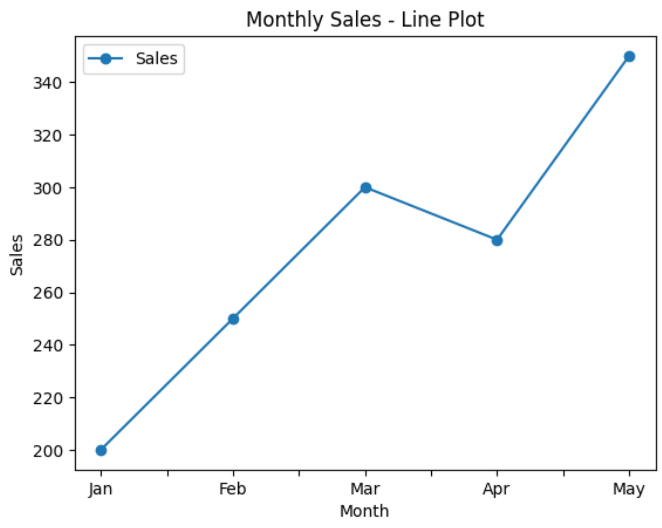
   Shows data points connected by lines.
Used for **time-series trends**.

Program:

```
import matplotlib.pyplot as plt

x = ['Jan', 'Feb', 'Mar', 'Apr', 'May']

y = [200, 250, 300, 280, 350]

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

plt.title('Line Plot - Monthly Sales')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.show()
```

Output:

```
<Figure size 400x400 with 0 Axes>
```



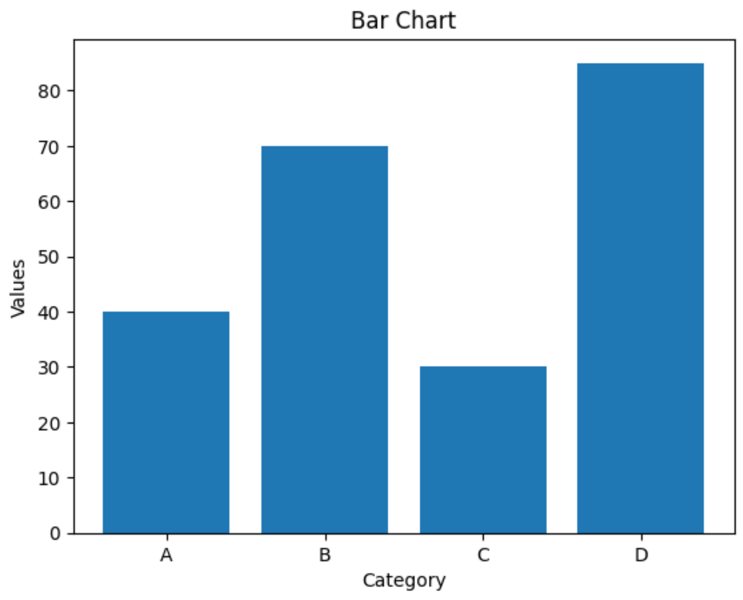
2. Bar Chart:
   Represents data with **vertical or horizontal bars**.

Program:

```
import matplotlib.pyplot as plt
categories = ['A', 'B', 'C', 'D']
values = [40, 70, 30, 85]
plt.bar(categories, values)
plt.title('Bar Chart')
plt.xlabel('Category')
plt.ylabel('Values')
plt.show()
```

Output:

```
<Figure size 600x400 with 0 Axes>
```



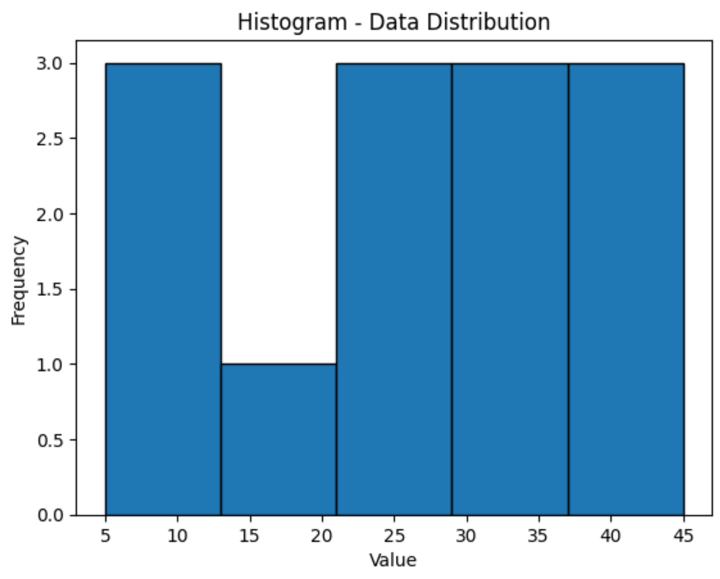
3.Histogram:

Shows the **distribution** of numerical data by grouping values into bins.

Program:

```
import matplotlib.pyplot as plt
data = [5, 10, 12, 20, 22, 22, 25, 30, 32, 35, 40, 42, 45]
plt.hist(data, bins=5, edgecolor='black')
plt.title('Histogram - Data Distribution')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```

Output:



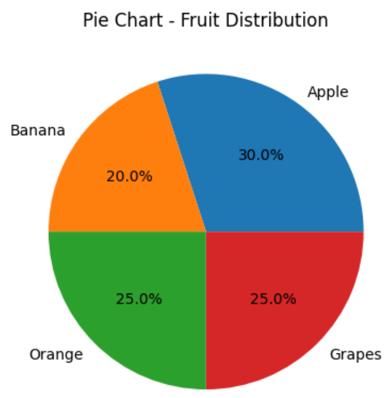
4.Pie Chart:

Represents data as slices of a circle.
Shows **proportions or percentages**.

Program:

```
import matplotlib.pyplot as plt
labels = ['Apple', 'Banana', 'Orange', 'Grapes']
sizes = [30, 20, 25, 25]
plt.pie(sizes, labels=labels, autopct='%1.1f%%')
plt.title('Pie Chart - Fruit Distribution')
plt.show()
```

Output:



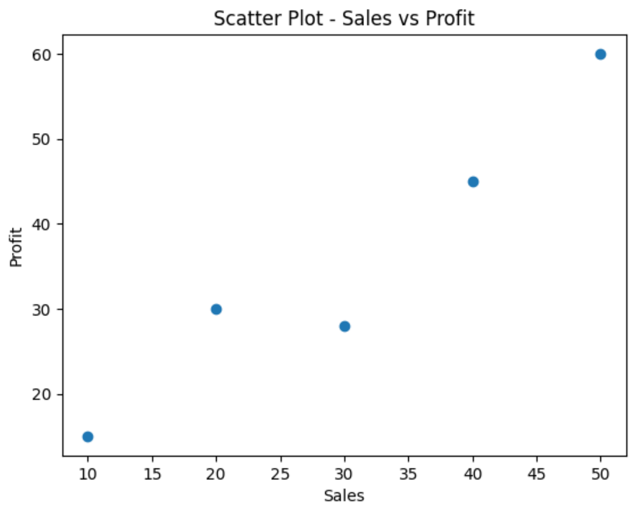
5.Scatter Plot:

Displays points representing two numeric variables. Used for **correlation analysis**.

Program:

```
import matplotlib.pyplot as plt

x = [10, 20, 30, 40, 50]

y = [15, 30, 28, 45, 60]

plt.scatter(x, y)

plt.title('Scatter Plot - Sales vs Profit')

plt.xlabel('Sales')

plt.ylabel('Profit')

plt.show()
```

Output:



Comparision between the libraries Pandas and Matplotlib:

Criteria	**Pandas**	**Matplotlib**
Purpose	Data analysis tool with built-in visualization	Dedicated plotting and visualization library
Foundation	Built on top of Matplotlib	Core library (base for Pandas, Seaborn, etc.)
Visualization Style	High-level, simple, automatic	Low-level, fully customizable
Code Requirement	Very few lines of code	Requires more lines and detailed setup
Data Handling	Works directly with DataFrame & Series	Works with arrays/lists; DataFrame support via Pandas
Plot Generation Speed	Faster for simple plots	Slower due to customization but more powerful

Criteria	**Pandas**	**Matplotlib**
Customizability	Limited (basic options only)	Extremely high—colors, fonts, grids, axes, styles
Learning Difficulty	Easy for beginners	Medium to advanced depending on features used
Plot Range	Only basic plots (line, bar, pie, scatter, hist)	Wide range (boxplot, stem, heatmap, 3D plot, polar, error bars, etc.)
Output Style	Default styling (less aesthetic)	Professional, publication-quality visuals
Use Case	Quick EDA (Exploratory Data Analysis)	Detailed analysis, reports, dashboards
Subplots Support	Limited & basic	Strong support with flexible layouts
Community Use	Preferred for data analysis	Preferred for visualization tasks
Dependencies	Dependent on Matplotlib	No dependency on Pandas
Advanced Features	Limited	Animation, interactive plotting, figure-level controls

Resources:

Pandas:

W3Schools Pandas Tutorial

https://www.w3schools.com/python/pandas/default.asp

GeeksforGeeks Pandas Guide

https://www.geeksforgeeks.org/python-pandas/

Kaggle Pandas Course

https://www.kaggle.com/learn/pandas

Matplotlib:

W3Schools Matplotlib Tutorial

https://www.w3schools.com/python/matplotlib_intro.asp

GeeksforGeeks Matplotlib Guide

https://www.geeksforgeeks.org/python-matplotlib-tutorial/