DS UNIT 3

Created @May 13, 2025 12:58 PM

Unit III: Data Visualization with Matplotlib and Seaborn

1. Introduction to Matplotlib Library

What is Matplotlib?

- **Matplotlib** is a popular Python library for creating static, animated, and interactive data visualizations.
- Developed by John D. Hunter in 2003; now widely used and maintained.
- Used for line plots, scatter plots, bar charts, histograms, 3D plots, and more.

Features of Matplotlib

- **Versatile**: Supports many plot types (line, scatter, bar, hist, pie, etc.)
- **Highly Customizable**: Control over line styles, colors, markers, axes, annotations, fonts.
- **Integration**: Works seamlessly with NumPy arrays and Pandas DataFrames.
- **Publication-Quality Figures**: High control over plot appearance.
- **Extensible**: Works with toolkits (Seaborn, Basemap, Cartopy, etc.)
- **Cross-Platform**: Works on Windows, macOS, Linux.
- **Interactive**: Supports widgets, dynamic plots (esp. with Jupyter/IPython).

Installation

```
pip install matplotlib
```

- To check the installation:

```
import matplotlib
print(matplotlib.__version__)
```

2. Components of Matplotlib

- **Figure**: The overall window or page where everything is drawn.
- **Axes**: An individual plot or graph (contains the data, x/y labels, title, etc.).
- **Axis**: The x-axis and y-axis of each plot.
- **Artist**: Everything visualized on a figure (e.g., lines, text, labels).
- **pyplot**: The state-based interface (usually imported as `import matplotlib.pyplot as plt`).

---

3. Types of Plots in Matplotlib

- **Line Graph**
- **Stem Plot**
- **Bar Chart**
- **Histogram**
- **Scatter Plot**
- **Stack Plot**
- **Box Plot**
- **Pie Chart**
- **Error Plot**
- **Violin Plot**
- **3D Plots**

---

4. Plotting Styles and Methods

- **Pyplot (plt) interface**
- **Figure Class**, **Axes Class**
- *set_ methods*for customizing plots (color, font, grid, etc.).
- **Custom Legends** and **Ticks/Labels**
- **Multiple Subplots**
- **Image plotting**

---

5. Subplots and Multiple Plots

Creating Subplots

Using `plt.subplot`

- Parameters: `(nrows, ncols, index)`
- Example: Create two plots side by side

```
import matplotlib.pyplot as plt
import numpy as np
x = np.array([0, 1, 2, 3])
y1 = np.array([3, 8, 1, 10])
y2 = np.array([10, 20, 30, 40])
plt.subplot(1, 2, 1)
plt.plot(x, y1)
plt.subplot(1, 2, 2)
plt.plot(x, y2)
plt.show()
```

Using `plt.subplots`

- Returns a Figure and Axes object/array.

```
fig, axs = plt.subplots(2, 2)
axs[0, 0].plot([1,2,3],[4,5,6])
```

```
axs[0, 1].scatter([1,2,3],[4,5,6])
axs[1, 0].bar([1,2,3],[4,5,6])
axs[1, 1].hist([1,2,2,3,3,3,3,3,3,4,4,5])
plt.show()
```

- Subplot grids can be 1D or 2D arrays; index them appropriately.

---

6. Controlling Axes, Ticks, Labels, Legends

Axes Control

- **xlim/ylim**: Set range of axes.

```
plt.xlim(0, 5)
plt.ylim(0, 35)
```

- **xticks/yticks**: Set tick locations and labels.

```
plt.xticks([1, 2, 3, 4], ['One', 'Two', 'Three', 'Four'])
plt.yticks([10, 20, 30], ['Low', 'Medium', 'High'])
```

- **Tick Label Rotation**:

```
plt.xticks(rotation=45)
plt.yticks(rotation=90)
```

Labels and Title

- **xlabel/ylabel/title**:

```
plt.xlabel('X Axis', fontsize=14, color='blue')
plt.ylabel('Y Axis', fontsize=14, color='red')
plt.title('Plot Title', fontsize=16)
```

Legend

- **Adding and Customizing**:

```
plt.plot([1,2,3],[10,20,25], label='Line 1')
plt.plot([1,2,3],[30,25,20], label='Line 2')
plt.legend(loc='upper left') # Position can be customized
```

7. Annotations and Saving Plots

Annotations

- **Adding text and arrows** using `plt.annotate()`:

```
plt.annotate('Peak', xy=(3,5), xytext=(3.5,6),
             arrowprops=dict(facecolor='black', arrowstyle='->'), fontsize=10)
```

- **Bounding boxes** and custom arrows are possible via `arrowprops` and `bbox` arguments.

Saving Plots

- **Syntax:** `plt.savefig(fname, dpi=None, bbox_inches='tight', format=None, ...)`
- Examples:

```
plt.savefig('plot.png')
plt.savefig('plot.pdf', dpi=300)
```

- Common file types: .png, .jpg, .svg, .pdf
- Use parameters like dpi for resolution, bbox_inches for tight layout, etc.

---

8. Introduction to Seaborn Library

What is Seaborn?

- **Seaborn** is a statistical data visualization library built on Matplotlib.

- Provides shapely, attractive, and informative statistical graphics with much less code.
- Integrates well with Pandas and NumPy.

Features

- High-level statistical plotting functions
- Beautiful default themes and color palettes
- Built-in support for complex visualizations like Pairplot, FacetGrid
- Quick integration with DataFrames
- User-friendly API, well-documented

Installation

```
pip install seaborn
```

9. Major Plot Types in Seaborn

Scatter Plot

```
import seaborn as sns
import matplotlib.pyplot as plt
data = {"x": [1,2,3,4,5], "y": [2,5,3,8,10]}
sns.scatterplot(x="x", y="y", data=data)
plt.title("Scatter Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

Line Plot

```
x = [1,2,3,4,5]
y = [2,5,3,8,10]
```

```
sns.lineplot(x=x, y=y)
plt.title("Sample Line Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```

Histogram

```
import numpy as np
data = np.random.randn(1000)
plt.hist(data, density=True)
plt.title("Histogram")
plt.xlabel("Value")
plt.ylabel("Density")
plt.show()
```

Boxplot

- Displays distribution, median, quartiles, and outliers.

```
data = {"A": [1,2,3,4,5], "B": [2,4,6,8,10], "C":[3,6,9,12,15]}
sns.boxplot(data=data)
plt.title("Box Plot")
plt.xlabel("Columns")
plt.ylabel("Values")
plt.show()
```

- To highlight outliers:

```
df = pd.DataFrame({"A": [1,2,3,4,5,30], "B":[2,4,6,8,7,28], "C":[3,6,9,5,2,7]})
sns.boxplot(data=df, showmeans=True, whis=1.5)
plt.title("Box Plot with Outliers")
plt.show()
```

Pairplot

- Plots pairwise relationships between numeric columns.

```
df = pd.DataFrame({"A":[1,2,3,4,5,8],"B":[2,4,6,8,10,7], "C":[3,6,9,2,5,8]})
sns.pairplot(df)
plt.show()
```

- With hue:

```
df = pd.DataFrame({
    "A": [10,20,30,40,50],
    "B": [5,10,15,20,25],
    "C": [8,12,16,20,24],
    "D": ["A","B","A","B","A"]
})
sns.pairplot(df, hue="D")
plt.show()
```

10. Multiple Plots and Grids in Seaborn

FacetGrid

- Display same plot type for subsets of data (categorical layout).

```
sns.FacetGrid(df, col="hue", height=4).map(sns.scatterplot, "x", "y")
plt.show()
```

- By row:

```
sns.FacetGrid(df, row="hue", height=4).map(sns.scatterplot, "x", "y")
plt.show()
```

Jointplot

- Shows scatter (or other) plots plus histograms for the variables.

```
sns.jointplot(x='x', y='y', kind="scatter", data=df)
plt.show()
```

Gridspec and Subplots

- Custom layouts using plt.GridSpec or plt.subplots.

```
import matplotlib.pyplot as plt
import seaborn as sns
Grid_plot = plt.GridSpec(2,3, wspace=0.8, hspace=0.6)
plt.subplot(Grid_plot[0,0]); plt.grid(True)
plt.subplot(Grid_plot[0,1:]); plt.grid(True)
plt.subplot(Grid_plot[1,:2]); plt.grid(True)
plt.subplot(Grid_plot[1,2]); plt.grid(True)
plt.show()
```

11. Scatterplot, Lineplot, Histogram, and Boxplot

Summary

- **Scatterplot:** Compares two numeric variables, each dot = observation.
- **Lineplot:** Shows trend or relationship between variables, often over time.
- **Histogram:** Displays distribution/frequency of a numeric variable.
- **Boxplot:** Summarizes numeric data via quartiles, highlights median/outliers.

---

12. Playing with Text in Plots

Word Clouds

```
from wordcloud import WordCloud
text = "word cloud example example example"
wordcloud = WordCloud(width=800, height=400, background_color="white").
generate(text)
```

```
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show()
```

Text Annotations

- Use `plt.text(x, y, s, ...)` or Seaborn's plot with Matplotlib's text:

```
sns.scatterplot(x=[1,2,3], y=[4,5,6])
for xi, yi in zip([1,2,3], [4,5,6]):
    plt.text(xi, yi, f"({xi},{yi})", ha='center')
plt.show()
```

Text as Plot Elements

- Use `ax.text()` for absolute control:

```
fig, ax = plt.subplots()
ax.text(0.5, 0.5, "Sample Text", ha='center', va='center', transform=ax.transAx
es)
plt.show()
```

13. 3D Plots

- **Requires:** `from mpl_toolkits.mplot3d import Axes3D`
- Types: 3D scatter, line, and surface plots.

3D Scatter Plot

```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter([1,2,3], [2,4,5], [10,15,20], c='blue', s=100)
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
```

```
ax.set_zlabel('Z-axis')
plt.title('3D Scatter Plot')
plt.show()
```

3D Line Plot

```
x = [1,2,3,4,5]
y = [2,4,3,5,1]
z = [4,5,3,2,6]
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot3D(x, y, z)
plt.title('3D Line Plot')
plt.show()
```

3D Surface Plot

```
import numpy as np
x = np.linspace(0, 5, 10)
y = np.linspace(0, 5, 10)
X, Y = np.meshgrid(x, y)
Z = X**2 + Y**2
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis', linewidth=0, antialiased=True)
ax.set_xlabel('X')
ax.set_ylabel('Y')
plt.title('3D Surface Plot')
plt.show()
```

14. Quick Reference Table: Common Plot Types

Plot Type	Matplotlib (plt)	Seaborn	Usage Example
Line	plt.plot()	sns.lineplot()	Trends/time series

Scatter	plt.scatter()	sns.scatterplot()	Correlation between variables
Histogram	plt.hist()	sns.histplot()/sns.distplot()	Distribution of a variable
Boxplot	plt.boxplot()	sns.boxplot()	Summary of distribution and outliers
Pairplot	N/A	sns.pairplot()	Pairwise relationships
FacetGrid	N/A	sns.FacetGrid()	Grouped/tiled plots by variable
Jointplot	N/A	sns.jointplot()	Bivariate + marginal histograms
3D Plot	mpl_toolkits.mplot3d	(Matplotlib)	3D visualization
Save Plot	plt.savefig()	plt.savefig()	Export plot

## 15. Conclusion			

- **Matplotlib** is the foundation for Python plotting—highly flexible, powerful, and detailed.
- **Seaborn** builds on Matplotlib—user-friendly, attractive for statistical and exploratory data analysis.
- Master both for complete control and fast development of visualization in data science and analytics.
- Always label axes, include legends/titles, and properly annotate your plots for maximum clarity!