🔵 MATPLOTLIB Topics (Low-level control over plots)

Matplotlib gives you **fine control** over plot elements. It's the base library that Seaborn is built on.

---

1. Matplotlib Basics

- ✅ **What**: Core library for all plotting in Python
- 📘 **Why**: Full control over figure size, labels, titles, etc.
- 💼 **Where used**: Custom visuals in dashboards or publications
- 💡 `import matplotlib.pyplot as plt`

---

2. Line Plot

- ✅ Show **trends over time or continuous values**
- 📘 Used for time-series, sensor readings, stock prices
- 💼 Used in: Finance, sales analytics, forecasting
- 💡 `plt.plot(x, y)`

---

3. Bar Chart

- ✅ Compare categories (e.g., product sales)
- 📘 Used for comparing groups
- 💼 Used in: Business reports, marketing data, categorical data
- 💡 `plt.bar(x, height)`

---

4. Histogram

- ✅ Show **distribution** of numeric data
- 📘 Used to understand skewness, outliers, frequency
- 💼 Used in: EDA, statistics, feature analysis
- 💡 `plt.hist(data, bins=10)`

---

5. Scatter Plot

- ✅ Show **relationship** between 2 variables
- 📘 Used to detect correlation or clustering
- 💼 Used in: Regression, classification, clustering
- 💡 `plt.scatter(x, y)`

---

6. Pie Chart

- ✅ Show **proportions** of a whole
- 📘 Simple category share analysis
- 💼 Used in: Business, sales, presentation
- 💡 `plt.pie(sizes, labels=labels, autopct='%1.1f%%')`

---

7. Subplots

- ✅ Multiple plots in one figure
- 📘 Used for dashboards, comparisons
- 💡 `plt.subplot(2, 2, 1)`

- 💼 Used in: Exploratory analysis reports

---

8. Figure Customization

- ✅ Titles, labels, legends, grid
- 📘 For professional report-ready charts
- 💡 `plt.title()`, `plt.xlabel()`, `plt.grid()`

---

9. Figure Size & DPI

- ✅ Control size and resolution of plot
- 📘 Needed for export in publications
- 💡 `plt.figure(figsize=(10,5), dpi=100)`

---

10. Saving Plots

- ✅ Export as PNG, JPG, PDF, etc.
- 💼 Used in: Automated reporting
- 💡 `plt.savefig("plot.png")`

🟣 SEABORN Topics (High-level, statistical plotting)

Seaborn is **built on Matplotlib**, and is perfect for **statistical visualization**.

---

1. Seaborn Basics

- ✅ High-level visualization library
- 📘 Easy and beautiful plots
- 💼 Used in: EDA, data storytelling
- 💡 `import seaborn as sns`

---

2. Seaborn Themes and Styles

- ✅ Pre-defined professional styles
- 💼 Used in: Presentation, publication, branding
- 💡 `sns.set_style("whitegrid")`, `sns.set_palette("muted")`

---

3. Distribution Plot (`histplot`)

- ✅ Show data distribution
- 💼 Used for: EDA, normality check
- 💡 `sns.histplot(data=df, x='column')`

---

4. Box Plot

- ✅ Show median, IQR, outliers
- 💼 Used in: Outlier detection, comparison of groups
- 💡 `sns.boxplot(x='category', y='value', data=df)`

---

5. Violin Plot

- ✅ Boxplot + KDE (smoothed distribution)
- 💼 Used in: Feature comparison
- 💡 `sns.violinplot(x='group', y='value', data=df)`

---

6. Count Plot

- ✅ Show frequency of categories
- 💼 Used in: Categorical EDA
- 💡 `sns.countplot(x='gender', data=df)`

---

7. Bar Plot

- ✅ Show mean (with CI) for categories
- 💼 Used in: Statistical comparison
- 💡 `sns.barplot(x='group', y='value', data=df)`

---

8. Scatter Plot (`scatterplot`)

- ✅ Visualize relationships + grouping
- 💼 Used in: Clustering, feature relationships
- 💡 `sns.scatterplot(x='a', y='b', hue='target', data=df)`

---

9. Swarm & Strip Plot

- ✅ Show all points in a category

- 💼 Used in: Class distribution
- 💡 `sns.swarmplot(x='class', y='score', data=df)`

---

10. Heatmap (Correlation Matrix)

- ✅ Show variable relationships using color
- 💼 Used in: Feature selection for ML
- 💡

python
CopyEdit
```
corr = df.corr()
sns.heatmap(corr, annot=True, cmap='coolwarm')
```

---

11. Pairplot

- ✅ Matrix of scatter plots between all pairs
- 💼 Used in: ML model exploration
- 💡 `sns.pairplot(df, hue='target')`

---

12. Catplot

- ✅ Flexible multi-chart visualization by category
- 💼 Used in: Multi-variable group analysis
- 💡 `sns.catplot(x, y, hue, kind='bar', col='gender')`

---

13. Regplot / Lmplot

- ✅ Scatter with regression line
- 💼 Used in: Linear relationships, model validation
- 💡 `sns.regplot(x='x', y='y', data=df)`