# 📊 Matplotlib for Data Analysis & Machine Learning

Matplotlib is a powerful Python library for **data visualization**. While libraries like Seaborn, Plotly, or Altair build on top of it, **Matplotlib is the foundation** and gives you the most flexibility.

---

## 1. Why Matplotlib in Data Analysis & ML?

- Helps understand **data distribution** (histograms, boxplots).
- Useful for **feature analysis** (scatter plots, pair plots).
- Essential for **model evaluation** (learning curves, confusion matrices, ROC curves).
- Can integrate with **NumPy, Pandas, and Scikit-learn** seamlessly.

---

## 2. Installation & Setup

```
pip install matplotlib numpy pandas scikit-learn
```

Basic import:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

---

## 3. Basic Plotting

### Line Plot

```
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y, label="sin(x)")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Line Plot Example")
plt.legend()
plt.show()
```

👉 Line plots are useful to show **trends over time** or continuous functions.

---

### Bar Plot

```
categories = ["A", "B", "C", "D"]
values = [5, 7, 3, 8]

plt.bar(categories, values, color="teal")
plt.title("Bar Plot Example")
plt.show()
```

👉 Bar plots are good for comparing **categorical variables**.

---

### Scatter Plot

```
x = np.random.rand(50)
y = np.random.rand(50)

plt.scatter(x, y, color="red")
plt.title("Scatter Plot Example")
plt.show()
```

👉 Scatter plots are used to find **relationships between variables**.

---

## 4. Customization

You can style almost everything:

```
plt.plot(x, y, color="green", linestyle="--", marker="o")
plt.xlabel("X values")
plt.ylabel("Y values")
plt.title("Customized Plot")
plt.grid(True)
plt.show()
```

---

## 5. Subplots

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

axs[0].plot(x, np.sin(x), "b")
axs[0].set_title("Sine")
```

```
axs[1].plot(x, np.cos(x), "r")
axs[1].set_title("Cosine")

plt.show()
```

👉 Subplots let you **compare multiple variables** side by side.

## 6. Data Analysis with Matplotlib + Pandas

Matplotlib integrates with Pandas for **EDA (Exploratory Data Analysis)**.

```
df = pd.DataFrame({
    "Age": np.random.randint(18, 60, 100),
    "Salary": np.random.randint(30000, 100000, 100),
    "Department": np.random.choice(["HR", "IT", "Sales"], 100)
})

# Histogram of Age
df["Age"].plot(kind="hist", bins=10, color="skyblue")
plt.title("Age Distribution")
plt.show()

# Bar chart of average salary per department
df.groupby("Department")["Salary"].mean().plot(kind="bar", color="orange")
plt.title("Average Salary by Department")
plt.show()
```

## 7. Advanced Visualizations for ML

### 7.1 Correlation Heatmap

Visualize feature correlations before ML.

```
corr = df.corr()
plt.imshow(corr, cmap="coolwarm", interpolation="nearest")
plt.colorbar()
plt.title("Correlation Heatmap")
plt.show()
```

### 7.2 Feature Distribution

Helps check skewness/outliers.

```
plt.boxplot(df["Salary"])
plt.title("Boxplot of Salary")
plt.show()
```

## 7.3 Model Evaluation – Confusion Matrix

```
from sklearn.metrics import confusion_matrix
import seaborn as sns

y_true = [1, 0, 1, 1, 0, 1, 0, 0, 1, 0]
y_pred = [1, 0, 1, 0, 0, 1, 1, 0, 1, 0]

cm = confusion_matrix(y_true, y_pred)

plt.imshow(cm, cmap="Blues")
plt.title("Confusion Matrix")
plt.colorbar()
plt.xticks([0, 1], ["Pred 0", "Pred 1"])
plt.yticks([0, 1], ["True 0", "True 1"])
plt.show()
```

## 7.4 ROC Curve

For classification model performance.

```
from sklearn.metrics import roc_curve, auc

y_true = np.array([0, 0, 1, 1])
y_scores = np.array([0.1, 0.4, 0.35, 0.8])

fpr, tpr, _ = roc_curve(y_true, y_scores)
roc_auc = auc(fpr, tpr)

plt.plot(fpr, tpr, color="blue", label=f"ROC curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], "r--")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```

## 7.5 Learning Curves

To check for overfitting/underfitting.

```
train_sizes = [50, 100, 150, 200]
train_scores = [0.6, 0.7, 0.8, 0.85]
test_scores = [0.55, 0.65, 0.68, 0.7]

plt.plot(train_sizes, train_scores, "o-", label="Train Score")
plt.plot(train_sizes, test_scores, "o-", label="Test Score")
plt.xlabel("Training Size")
plt.ylabel("Score")
plt.title("Learning Curve")
plt.legend()
plt.show()
```

# 8. Saving Figures

```
plt.plot(x, y)
plt.title("Example Plot")
plt.savefig("figure.png", dpi=300)
plt.show()
```

# 9. Exercises

1. Plot the distribution of exam scores (random normal data, 1000 samples).
2. Use a scatter plot to visualize the relationship between age and salary.
3. Plot a confusion matrix for a simple binary classifier.
4. Generate and plot a ROC curve with random scores.
5. Compare two models’ learning curves on the same graph.

# 10. Mini Project – Titanic Data Visualization

```
import seaborn as sns
```

```
# Load Titanic dataset
titanic = sns.load_dataset("titanic")

# Survival count
titanic["survived"].value_counts().plot(kind="bar", color=["red", "green"])
plt.title("Survival Count")
plt.show()

# Survival by class
pd.crosstab(titanic["pclass"], titanic["survived"]).plot(kind="bar", stacked=True)
plt.title("Survival by Passenger Class")
plt.show()

# Age distribution by survival
titanic[titanic["survived"] == 1]["age"].plot(kind="hist", bins=20, alpha=0.5, label="Survi
titanic[titanic["survived"] == 0]["age"].plot(kind="hist", bins=20, alpha=0.5, label="Not S
plt.legend()
plt.title("Age Distribution by Survival")
plt.show()
```

👉 This kind of analysis is exactly what you'd do before building ML models.

---

✅ Now you've got:

- **Matplotlib basics**
- **EDA tools for Data Analysis**
- **Machine Learning visualizations**