Comprehensive Notes on Matplotlib for Machine Learning

Prepared for ML Learning

Contents

1 Introduction to Matplotlib **3**

2 Basic Workflow of Matplotlib **3**

3 Figure and Axes **3**
3.1 Creating figure and axes 4

4 Types of Plots **4**
4.1 Line Plot 4
4.2 Scatter Plot 4
4.3 Bar Plot 4
4.4 Histogram 4
4.5 Multiple Plots (Subplots) 5

5 Customization **5**
5.1 Labels and Title 5
5.2 Adding Legends 5
5.3 Colors and Line Styles 5
5.4 Grid 5
5.5 Figure Size 5

6 Saving Figures **6**

7 Matplotlib with Numpy **6**

8 Matplotlib for Machine Learning **6**
8.1 Plotting Loss Curve 6
8.2 Plotting Accuracy Curve 6
8.3 Visualizing Classification Results 7
8.4 Confusion Matrix Visualization 7

9 Advanced Matplotlib Concepts **7**
9.1 Object-Oriented VS pyplot API 7
9.2 OO-style Example 7
9.3 3D Plots 7
9.4 Heatmaps 8

10 Best Practices for ML Projects **8**

11 Conclusion **8**

1 Introduction to Matplotlib

Matplotlib is the most widely used plotting library in Python. It plays a crucial role in Machine Learning for:

- Visualizing datasets
- Understanding data distributions
- Observing model performance
- Plotting loss curves, accuracy curves, etc.
- Creating scatter plots for classification problems

The main module used is:

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

2 Basic Workflow of Matplotlib

Almost every Matplotlib plot follows the same sequence:

1. Prepare the data
2. Create a figure and axes
3. Plot the data
4. Add labels, legend, title
5. Display plot

```python
import matplotlib.pyplot as plt

x = [1,2,3,4]
y = [2,4,6,8]

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

3 Figure and Axes

Matplotlib uses a hierarchical structure:

- **Figure**: Whole window
- **Axes**: Actual plot area inside the figure

3.1 Creating figure and axes

```
fig, ax = plt.subplots()

ax.plot([1,2,3], [4,5,6])
ax.set_title("Using Axes")
plt.show()
```

4 Types of Plots

Matplotlib supports many plot types. Below are the most important for ML.

4.1 Line Plot

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

Useful for:

- Loss curves
- Learning rate schedules

4.2 Scatter Plot

```
plt.scatter(x, y)
plt.show()
```

Useful for:

- Visualizing clusters (K-Means)
- Visualizing classification boundaries

4.3 Bar Plot

```
categories = ["A","B","C"]
values = [10, 30, 20]

plt.bar(categories, values)
plt.show()
```

4.4 Histogram

```
import numpy as np

data = np.random.randn(1000)
plt.hist(data, bins=30)
plt.show()
```

Useful for:

- Inspecting feature distribution
- Detecting outliers

4.5 Multiple Plots (Subplots)

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

ax[0].plot(x, y)
ax[1].scatter(x, y)

plt.show()
```

5 Customization

Matplotlib offers many customization options.

5.1 Labels and Title

```
plt.xlabel("X values")
plt.ylabel("Y values")
plt.title("My Plot")
```

5.2 Adding Legends

```
plt.plot(x, y, label="line1")
plt.legend()
```

5.3 Colors and Line Styles

```
plt.plot(x, y, color="red", linestyle="--", linewidth=2)
```

5.4 Grid

```
plt.grid(True)
```

5.5 Figure Size

```
plt.figure(figsize=(8,4))
```

6 Saving Figures

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

7 Matplotlib with Numpy

Numpy arrays work perfectly with Matplotlib.

```
import numpy as np

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

plt.plot(x, y)
plt.show()
```

8 Matplotlib for Machine Learning

This is where Matplotlib becomes essential.

8.1 Plotting Loss Curve

```
epochs = range(1,11)
loss = [0.9,0.7,0.6,0.5,0.4,0.35,0.3,0.28,0.25,0.2]

plt.plot(epochs, loss, marker='o')
plt.title("Training Loss Curve")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.grid(True)
plt.show()
```

8.2 Plotting Accuracy Curve

```
accuracy = [0.60,0.65,0.70,0.72,0.74,0.78,0.80,0.82,0.85,0.87]

plt.plot(epochs, accuracy, marker='o')
plt.title("Training Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.show()
```

8.3 Visualizing Classification Results

```
plt.scatter(class0_x, class0_y, label="Class 0")
plt.scatter(class1_x, class1_y, label="Class 1")
plt.legend()
plt.show()
```

8.4 Confusion Matrix Visualization

```
import seaborn as sns
import numpy as np

cm = np.array([[50, 10], [5, 80]])

sns.heatmap(cm, annot=True, cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```

9 Advanced Matplotlib Concepts

9.1 Object-Oriented VS pyplot API

Two ways to create plots:

- **pyplot:** quick and simple
- **OO-style:** recommended for ML projects

9.2 OO-style Example

```
fig, ax = plt.subplots()
ax.plot(x,y)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_title("OO Style Plot")
plt.show()
```

9.3 3D Plots

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

ax.scatter(x, y, z)
plt.show()
```

9.4 Heatmaps

```
plt.imshow(matrix, cmap='viridis')
plt.colorbar()
plt.show()
```

10 Best Practices for ML Projects

- Always label axes
- Use legends for multi-line graphs
- Use grid for readability
- Save figures with high DPI
- Use subplots to compare models
- Keep consistent colors across plots

11 Conclusion

Matplotlib is one of the most important tools in Machine Learning for visualizing:

- Data distribution
- Model training metrics
- Classification boundaries
- Clustering
- Confusion matrices

Learning Matplotlib deeply will significantly improve your ML workflow.