**Minor in AI**

**Notes**

04 March

**Title: Mastering Data Visualization with Matplotlib: From Basics to Stunning Plots!**

Syllabus:

* Customize Plots for Better Visualization
* Modify colors, markers, and line styles for better clarity.
* Add legends, annotations, and grid lines to improve interpretability.
* Handle Multiple Subplots: Use plt.subplot() or plt.subplots() to create multiple visualizations in a single figure.
* Adjust layout spacing using plt.tight\_layout().
* Save and Display Plots Efficiently, Export figures in different formats (.png, .jpg, .pdf), Optimize figure size and resolution for presentations and reports.

Activities carried out:

* Design of the city: Place Charles de Gaulle, 12 avenues
* NetLogo model discussed: Path
* We don’t need plots at all places – Stats dashboard from Championship Trophy
* Worm Graphs

What Intern did:

Case 1: An Intern was asked to analyze data on ice cream sales and shark attacks over the summer months. He noticed that as ice cream sales go up, shark attacks also increase! To prove his point, he made a line plot connecting the two trends.

"Look! The lines move together! That must mean eating ice cream causes sharks to attack!"

Case 2: Intern was asked to analyze how concert ticket prices impact the number of people attending a concert for different events. He decided to use a histogram to show the relationship.

"I made a histogram showing how many concerts had different ticket price ranges! That should explain everything, right?"

**Code Snippets that we did hands-on**

(Refer to video and collab for more)

import matplotlib.pyplot as plt

# Create a figure

plt.figure(figsize=(10, 5))

# Sample datasets

datasets = [1, 2, 3, 4, 5]

# Accuracy of Model A on different datasets

accuracy\_model\_a = [85, 88, 90, 87, 92]

# Accuracy of Model B on different datasets

accuracy\_model\_b = [80, 83, 85, 86, 89]

# First subplot - Model A Accuracy

plt.subplot(1, 2, 1)

plt.plot(datasets, accuracy\_model\_a, marker='o', color='blue', label="Model A")

plt.title("Model A Accuracy")

plt.xlabel("Dataset")

plt.ylabel("Accuracy (%)")

plt.ylim(auto=True)

plt.grid()

plt.legend()

# Second subplot - Model B Accuracy

plt.subplot(1, 2, 2)

plt.plot(datasets, accuracy\_model\_b, marker='s', color='red', label="Model B")

plt.title("Model B Accuracy")

plt.xlabel("Dataset")

plt.ylabel("Accuracy (%)")

plt.ylim(auto=True)

plt.grid()

plt.legend()

# Adjust layout and show the plots

plt.tight\_layout()

plt.show()

import matplotlib.pyplot as plt

# Training cycles (iterations where AI learns more memes)

training\_cycles = list(range(1, 11))

# Accuracy of AI in recognizing dank memes over time

meme\_recognition\_accuracy = [30, 40, 50, 60, 70, 78, 83, 87, 90, 93]

# Create the plot

plt.figure(figsize=(8, 5))

plt.plot(training\_cycles, meme\_recognition\_accuracy, marker='o', linestyle='-', color='purple', label="AI Meme Detector")

# Add titles and labels

plt.title("AI Learning to Recognize Dank Memes")

plt.xlabel("Training Cycle (More Memes Shown)")

plt.ylabel("Accuracy (%)")

plt.ylim(20, 100)  # Setting limits for better visualization

plt.grid()

plt.legend()

# Save the figure as an image

plt.savefig("meme\_ai\_accuracy.png", dpi=300)

# Show the plot

plt.show()

import matplotlib.pyplot as plt

import numpy as np

# Generate sample data

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

y = np.sin(x)

# Create a figure with optimized size and resolution

plt.figure(figsize=(10, 6), dpi=300)

# Plot data

plt.plot(x, y, label='Sine Wave', linewidth=2)

# Labels and title

plt.xlabel('X-axis', fontsize=14)

plt.ylabel('Y-axis', fontsize=14)

plt.title('Optimized Plot for Presentation and Reports', fontsize=16)

# Grid and legend

plt.grid(True, linestyle='--', alpha=0.6)

plt.legend(fontsize=12)

# Adjust layout

plt.tight\_layout()

# Save as high-resolution image

plt.savefig('optimized\_plot.png', dpi=300, bbox\_inches='tight')

# Show plot

plt.show()