Let’s compare **Matplotlib** and **ggplot (plotnine)** using the same dataset:

**Dataset**

import pandas as pd

data = {

"Hours Worked": [1, 2, 3, 4, 5],

"Money Earned": [100, 200, 300, 400, 500]

}

df = pd.DataFrame(data)

**Using Matplotlib**

import matplotlib.pyplot as plt

plt.plot(df["Hours Worked"], df["Money Earned"], marker='o', color='blue')

plt.title("Hours Worked vs Money Earned")

plt.xlabel("Hours Worked")

plt.ylabel("Money Earned (₹)")

plt.grid(True)

plt.show()

**Explanation of various functions:**

* plt.plot() → plots a line graph.
* marker='o' → shows points on the line.
* plt.title() → sets the title.
* plt.xlabel() and plt.ylabel() → label axes.
* plt.grid(True) → adds a grid for better readability.
* **Very flexible**, but you need to write separate lines for every detail.

**Using ggplot (plotnine)**

from plotnine import ggplot, aes, geom\_line, geom\_point, labs

plot = (

ggplot(df, aes(x="Hours Worked", y="Money Earned"))

+ geom\_line(color='blue')

+ geom\_point(color='red')

+ labs(title="Hours Worked vs Money Earned",

x="Hours Worked",

y="Money Earned (₹)")

)

print(plot)

**Explanation of various functions:**

* ggplot(df, aes(...)) → define the dataset and mapping of variables.
* geom\_line() → draw the line.
* geom\_point() → draw points on the line.
* labs() → set title and axis labels.
* Everything is **combined in a single chain** using +.
* **Declarative style** → you focus on *what* you want, not *how to draw it* step by step.

**Summary:**

* Often, **both are used together** in practice.