**Step-by-Step Guide: How to Create Subplots Using a For Loop in Python**

When analyzing datasets with multiple numerical variables, visualizing each feature’s distribution is essential. One efficient way to do this is by plotting **multiple subplots in a loop** using Matplotlib and Seaborn.

Project\_title: Agricultural product demand forecasting

**✅ Step 1: Import Necessary Libraries**

import matplotlib.pyplot as plt

import seaborn as sns

These libraries will help in plotting the histograms and styling the visuals.

**✅ Step 2: Define Your Numeric Variables**

Make a list of the numerical columns in your dataset you want to visualize.

numeric\_vars = ['Quantity\_Sold', 'Revenue', 'Temperature\_Celsius',

'Rainfall\_mm', 'Transportation\_Cost', 'Labor\_Cost',

'Quality\_Score', 'Inventory\_Level']

**✅ Step 3: Set Up the Plot Canvas (Figure Size)**

Define the overall size of the figure that will contain your subplots.

python

CopyEdit

plt.figure(figsize=(20, 15)) # Width = 20, Height = 15 (in inches)

**✅ Step 4: Loop Through the Variables and Create Subplots**

Using enumerate, iterate through the list of variables and create one subplot per variable.

for i, var in enumerate(numeric\_vars, 1): # Start index at 1

plt.subplot(3, 3, i) # Create subplot: 3 rows, 3 columns, position i

sns.histplot(df[var], bins=50, kde=True) # Histogram with KDE line

plt.axvline(df[var].mean(), color='red', linestyle='dashed', linewidth=2) # Vertical mean line

plt.title(f'Distribution of {var}') # Add title to each subplot

💡 **Tips:**

* 3, 3, i means 3 rows and 3 columns, filling position i.
* sns.histplot() gives a histogram.
* kde=True adds a curve to show the distribution shape.
* plt.axvline() marks the mean with a vertical red line.

**✅ Step 5: Tidy the Layout to Prevent Overlaps**

plt.tight\_layout()

This ensures the plots do not overlap and all titles and labels are visible.

**✅ Step 6: Display the Plots**

plt.show()

This renders all your plots in a neat 3x3 grid layout.

**🌟 Benefits of Using a for Loop with Subplots**

| **Advantage** | **Description** |
| --- | --- |
| ✅ **Efficient & DRY** | No need to repeat code for each variable—looping handles it automatically. |
| ✅ **Scalable** | Works for any number of variables; just update the list. |
| ✅ **Consistent Styling** | Ensures each plot uses the same structure, size, and format. |
| ✅ **Readable Output** | Clear, side-by-side comparison of distributions across all features. |
| ✅ **Professional Visuals** | Makes your exploratory data analysis (EDA) clean and presentable. |