**Seaborn** is a Python library built on top of **Matplotlib** that makes it easier to create attractive, informative statistical graphics with minimal code.

---

## 1. What Is Seaborn?

- **High-Level Interface for Statistical Plots**
  **Seaborn** provides a set of functions (like histplot, boxplot, scatterplot, heatmap, etc.) that automatically handle details such as color palettes, gridlines, and summary statistics (**confidence intervals, KDE curves, grouping).**
- **Works Seamlessly with Pandas DataFrames**
  You specify columns by name (e.g., x="column1", y="column2", hue="category"), and **Seaborn** takes care of splitting or aggregating data.
- **Beautiful Defaults**
  With one call to sns.set(), you get a “clean” theme, well-spaced axes, and an appealing color palette—no extra styling needed.

Below are **Seaborn-based** versions of the same visualizations we did in Matplotlib, all using the sales_data.csv file. We’ll walk through each plot step by step, showing both the code and a plain-English explanation of what it’s doing.

**Assumption:** You’ve already loaded **sales_data.csv** into a DataFrame called **df**. If you haven’t yet, here’s a quick reminder:

```
import pandas as pd
df = pd.read_csv(r'D:\csvfiles\sales_data.csv')  # or wherever your file lives
```

CSV should look like this (5 rows, 4 columns):

| Month | Sales | Profit | Customers |
|---|---|---|---|
| Jan | 1000 | 200 | 50 |
| Feb | 1500 | 300 | 60 |
| Mar | 1300 | 250 | 55 |
| Apr | 1700 | 400 | 70 |

|  |  |  |  |
|---|---|---|---|
| **May** | 1600 | 380 | 65 |

---

## 1. Distribution of Sales (Histogram with Kernel Density)

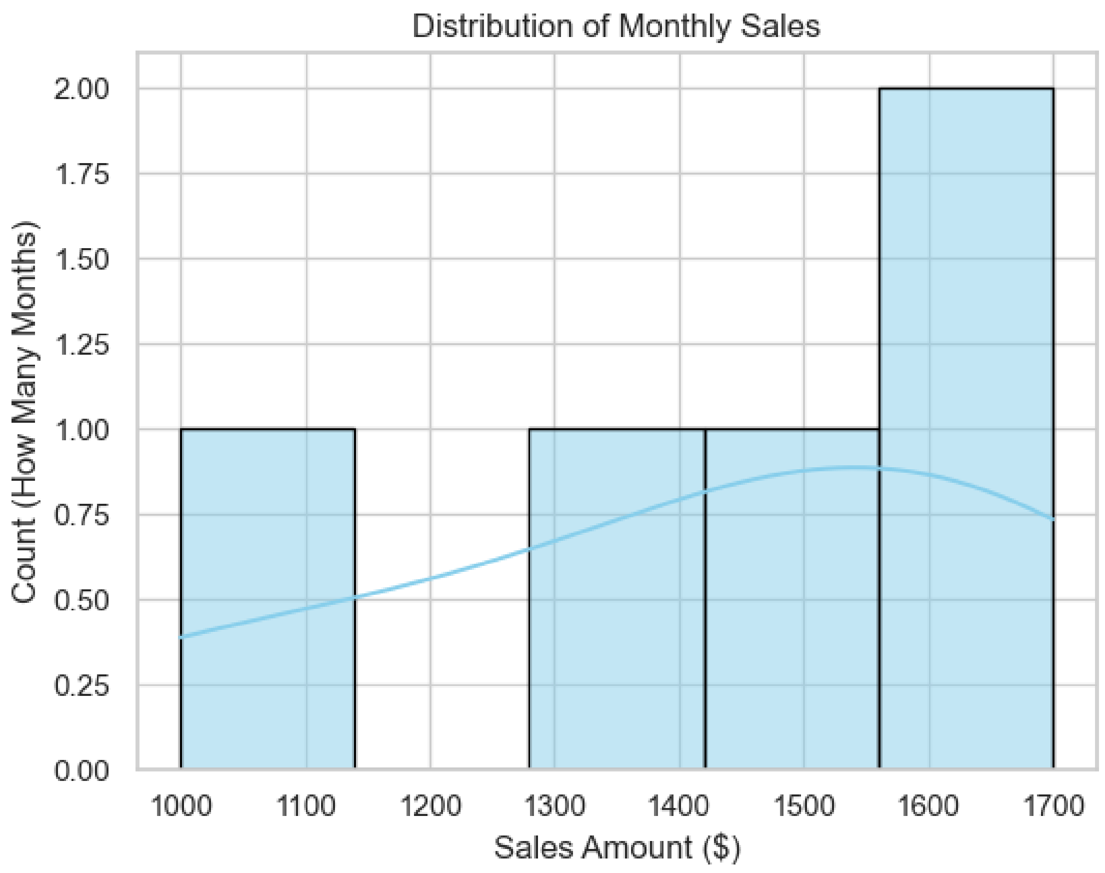
```
import seaborn as sns
import matplotlib.pyplot as plt

# 1. Set a seaborn style (optional, for nicer default colors/grids)
sns.set(style="whitegrid")

# 2. Plot a histogram + KDE (kernel density estimate) for the 'Sales'
column
sns.histplot(df['Sales'], bins=5, kde=True, color='skyblue',
edgecolor='black')

# 3. Add titles and labels
plt.title("Distribution of Monthly Sales")
plt.xlabel("Sales Amount ($)")
plt.ylabel("Count (How Many Months)")

plt.show()
```



## What It Shows

- **Histogram bars**: Each bar groups "Sales" into a range (bin).
  - We used `bins=5`, so Seaborn split the $1000–$1700 span into five equal bins.
  - The height of each bar is "how many months" had sales in that numeric range.
- **KDE (smooth curve)**: The **kde=True** adds a smooth curve on top of the bars, which helps you see the "shape" (density) of your data.
  - Where the curve is highest, most of your sales values cluster.
- **X-axis**: Dollar amounts (e.g. $1000, $1140, … up to $1700).
- **Y-axis**: How many months had sales in each bin.

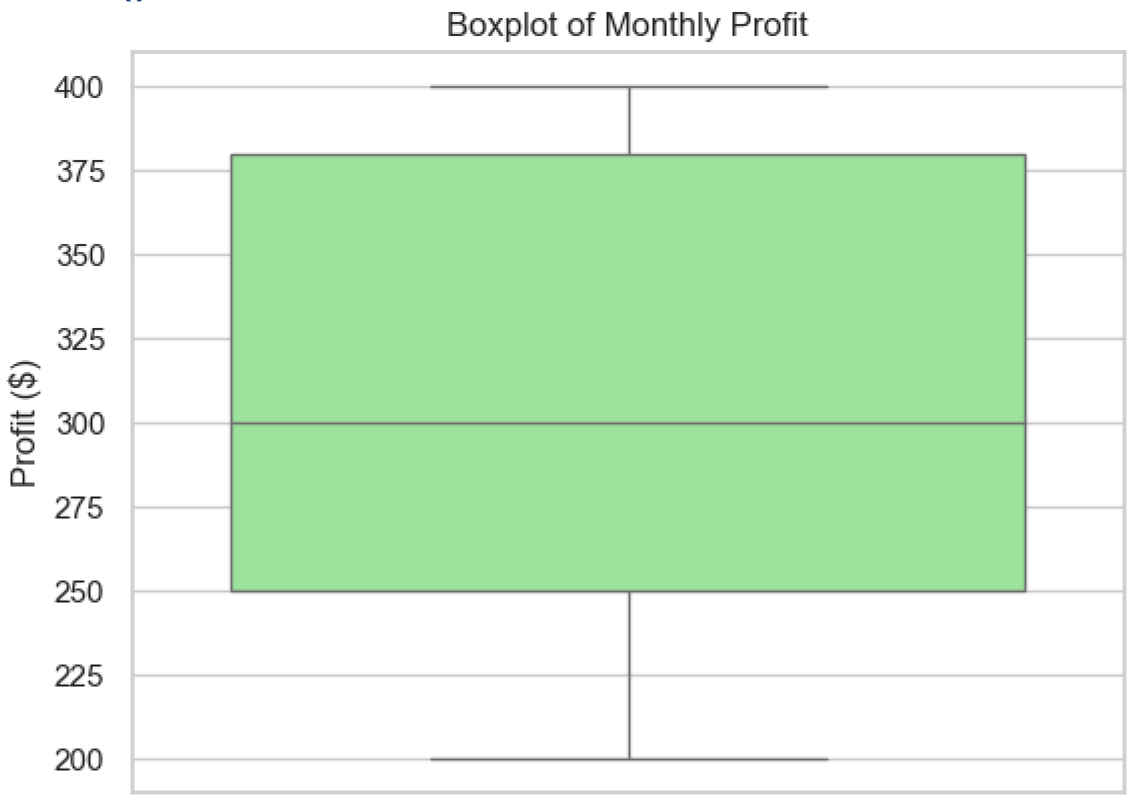
→ You will see one bar around $1000, one around $1300, one around $1500, and two months (Apr/May) in the top-end bin ($1560–$1700). The KDE curve simply "smooths" that information into a continuous curve.

## 2. Boxplot of Profit

```
# 1. Create a boxplot for the 'Profit' column
sns.boxplot(y=df['Profit'], color='lightgreen')

# 2. Annotate
plt.title("Boxplot of Monthly Profit")
plt.ylabel("Profit ($)")

plt.show()
```



**What It Shows:**

- **Box**:
  - The bottom of the box is Q1 (first quartile), the top is Q3 (third quartile).
  - The line inside the box is the **median** (middle value).
- **Whiskers**:
  - Vertical lines extend from Q1 down to the minimum profit and from Q3 up to the maximum profit, unless there are "outliers."
- **Outliers (dots beyond whiskers)**:

    - Any profit outside 1.5×IQR below Q1 or above Q3 would appear as a dot.
    - In our small dataset, none are that far out, so you simply see whiskers reaching down to $200 and up to $400.
- **Y-axis**: Profit in dollars.

→ You'll see that the "middle 50%" of profits sits roughly between $225 and $390, with a median at $300. The "whiskers" go down to $200 and up to $400.

---

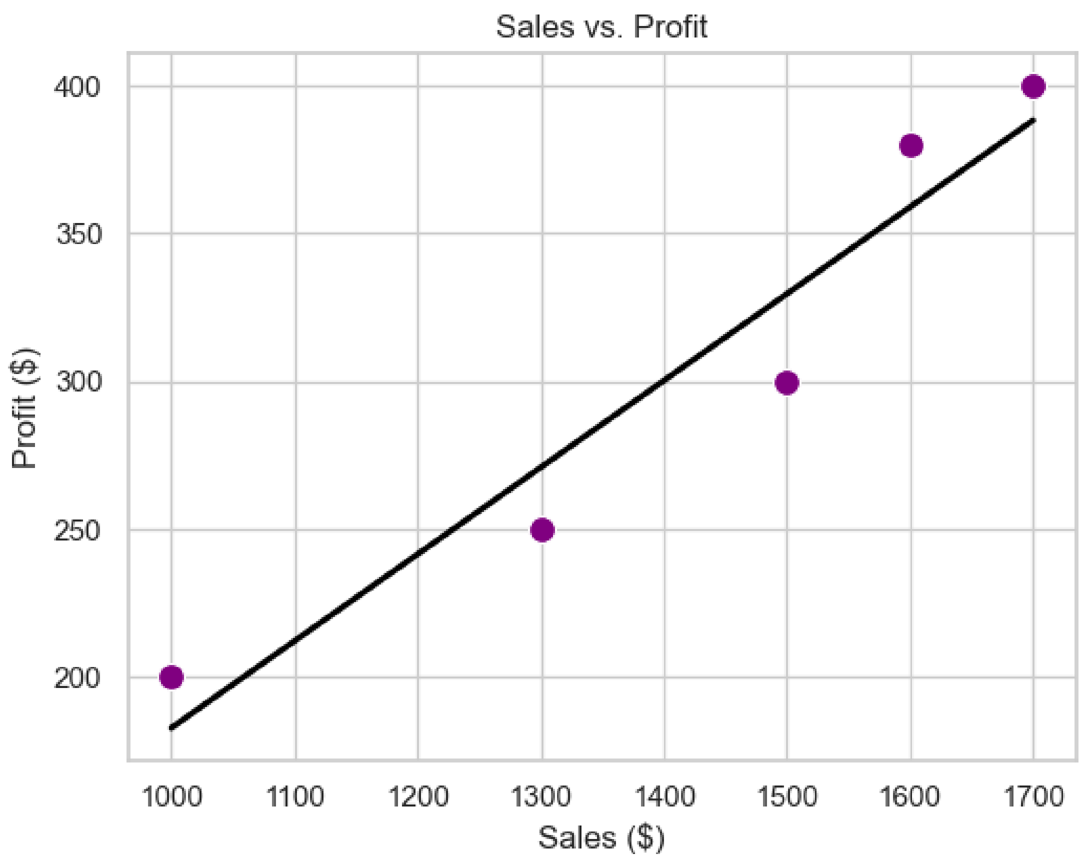
## 3. Scatter Plot: Sales vs. Profit

```
# 1. Scatterplot of Sales (x) vs. Profit (y), colored by default
sns.scatterplot(x='Sales', y='Profit', data=df, s=100, color='purple')

# 2. Add a simple linear trend line (regression) if you like:
sns.regplot(x='Sales', y='Profit', data=df, scatter=False, color='black', ci=None)

# 3. Annotate
plt.title("Sales vs. Profit")
plt.xlabel("Sales ($)")
plt.ylabel("Profit ($)")
plt.grid(True)

plt.show()
```



## What It Shows:

- **Dots**: Each dot represents one month, positioned by (Sales, Profit).
    - For example, the point at (1500, 300) is February.
- **Trend-line (optional)**: The `regplot(..., scatter=False)` draws a straight line that best fits your points (least-squares fit).
    - It visually tells you: "As sales rise, profit tends to rise in a roughly linear way."
- **X-axis**: Sales value.
- **Y-axis**: Profit value.

→ Since sales and profit both go up together in our dataset, you'll see a clear upward trend (dots slope up to the right). The black line confirms that relationship.

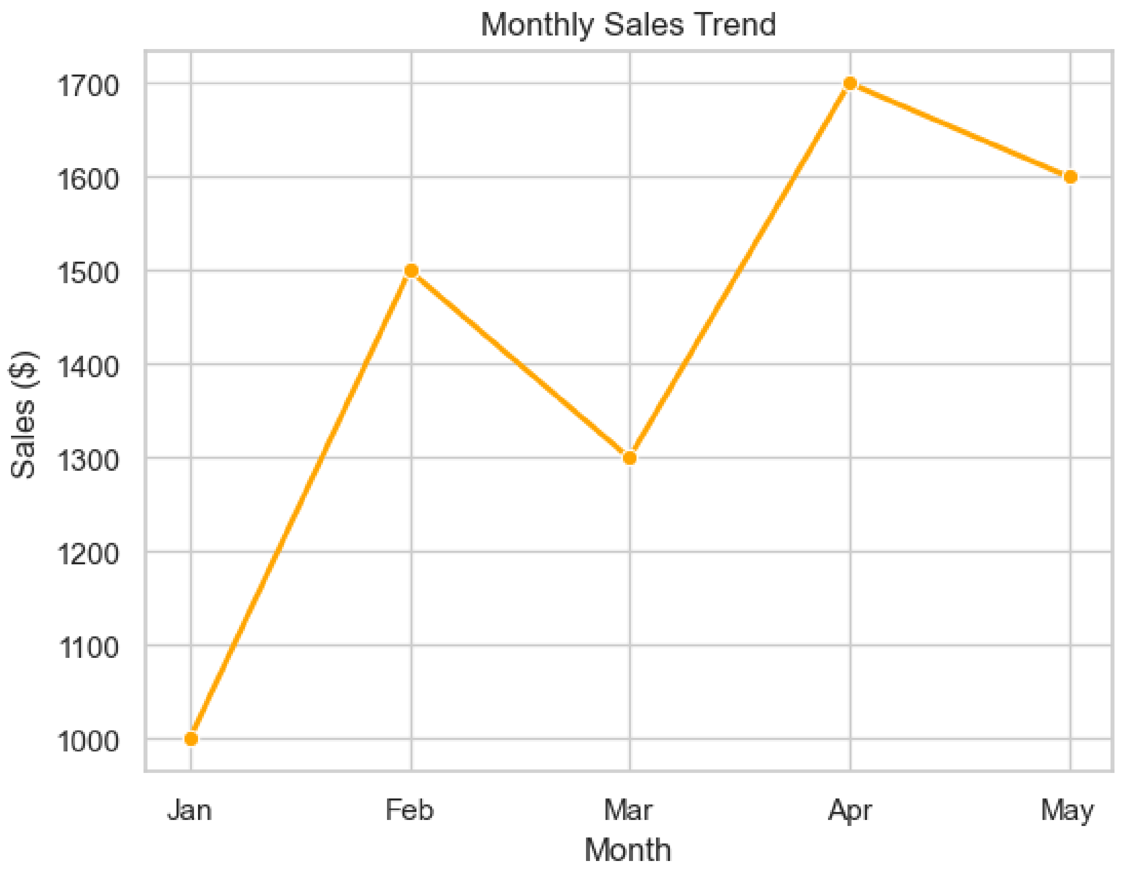
---

## 4. Line Plot: Sales Over Time

```
# 1. If 'Month' is a simple string (e.g. 'Jan', 'Feb', ...), seaborn will
plot in categorical order:
sns.lineplot(x='Month', y='Sales', data=df, marker='o', linewidth=2,
color='orange')

# 2. Annotate
plt.title("Monthly Sales Trend")
plt.xlabel("Month")
plt.ylabel("Sales ($)")
plt.grid(True)

plt.show()
```



### What It Shows:

- **Points connected by a line**:
    - Each month’s sales (Jan → Feb → Mar → ...) is plotted in order.

    - The **markers (“dots” at each month)** make it easy to see exact values, and the line joins them to show “trend over time.”
- **X-axis**: Categories “Jan”, “Feb”, “Mar”, etc. (Seaborn treats them in the order they appear in the DataFrame).
- **Y-axis**: Dollar amounts for sales.

→ You’ll see sales start at $1000 in January, rise to $1500 in February, dip slightly in March, and peak around $1700 in April/May.

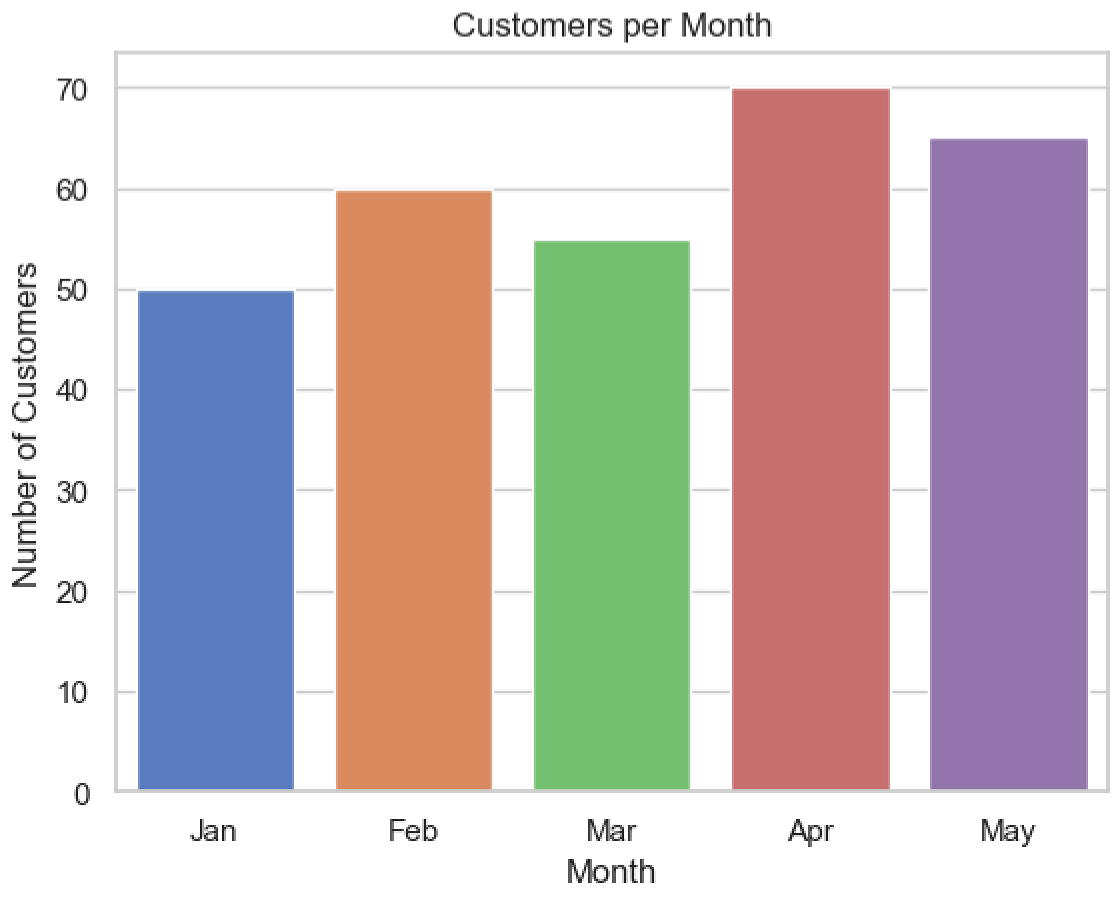
---

## 5. Bar Plot: Number of Customers per Month

```
# 1. Barplot for the 'Customers' column, grouped by 'Month'
sns.barplot(x='Month', y='Customers', data=df, palette='muted')

# 2. Annotate
plt.title("Customers per Month")
plt.xlabel("Month")
plt.ylabel("Number of Customers")

plt.show()
```



## What It Shows:

- **Bars**: Each bar's height shows how many customers visited that month.
- **X-axis**: "Jan", "Feb", "Mar", "Apr", "May."
- **Y-axis**: Count of customers.

→ Because there were 50 customers in January, 60 in February, 55 in March, 70 in April, and 65 in May, you'll see bars at those heights. It's an easy way to compare "foot traffic" month to month.

## Why Use Seaborn?

- **Simpler Syntax:** One function call often replaces several lines of Matplotlib.
- **Built-in Aesthetics:** With **sns.set(style="whitegrid"),** your charts automatically look cleaner (grid lines, pleasing default colors).
- **Annotations & Color Maps:** **sns.heatmap(..., annot=True)** or **sns.barplot(..., palette="muted")** make it very easy to get professional- looking plots in fewer lines.

---

## In a Nutshell

1. **Histogram with KDE**
   - Groups your sales numbers into bins and shows how many months fall into each bin. A smooth curve (KDE) sits on top to show the overall "shape" of your data.
2. **Boxplot**
   - Visually summarizes the distribution of profits. You see the middle 50% of values (the box), the median (line inside the box), and the "whiskers" (minimum/maximum that are not outliers).

3. **Scatter Plot + Trend Line**
   - Plots each month as a dot at (Sales, Profit). The trend line shows you that as sales go up, profit generally goes up too.
4. **Line Plot**
   - Connects monthly sales in chronological order (Jan → Feb → ...). You can immediately spot upticks or downturns across the first five months.
5. **Bar Plot**
   - Shows how many customers visited each month. Taller bars = more customers.