Simple Sales Data Visualization

Name: Gitika Pal

Unique ID: 202401100300112

Branch: CSE(AI)

Subject: Introduction to AI

Subject Code: AI101B

Simple Sales Data Visualization

Introduction

In today's competitive business environment, companies must make data-driven decisions to remain profitable and efficient. Sales data is one of the most critical metrics in understanding business performance, allowing companies to track revenue, monitor product success, and optimize inventory management. However, raw sales data can be overwhelming and difficult to interpret, especially when dealing with large datasets.

To overcome this challenge, businesses use **data visualization** techniques to present information in an easy-to-understand graphical format. Visualization tools help in spotting trends, identifying best-selling products, and making informed business strategies. This report focuses on a **Simple Sales Data Visualization** tool that allows users to input sales data and generates a **bar chart** to display revenue distribution among different products.

Problem Statement

Businesses often struggle with managing and analyzing their sales data effectively. Some common challenges include:

- **Data Overload:** Manually analysing large amounts of sales data can be time-consuming and prone to errors.
- **Lack of Insights:** Without proper visualization, it becomes difficult to identify sales trends and key revenue contributors.
- **Decision-Making Challenges:** Business owners and managers need quick access to meaningful insights to make strategic decisions.
- **Complexity of Advanced Tools:** Many data analysis tools require technical expertise, making it difficult for small business owners or non-technical users to leverage their benefits.

This project aims to address these challenges by developing a **simple and user-friendly** sales data visualization tool that automatically processes user input and generates a bar chart for easy interpretation.

Methodology

To achieve the objectives, we implemented the project using **Python**, a widely used programming language for data analysis and visualization. The following steps outline the approach used to solve the problem:

1. Data Collection

- The program prompts users to enter the number of sales records they want to input.
- For each record, users provide:
  - **Product Name** (to identify the item sold)
  - **Units Sold** (to determine the quantity of items sold)
  - **Revenue** (to measure the financial contribution of each product)

2. Data Processing

- The collected data is stored in a **Pandas DataFrame**, a data structure commonly used for handling tabular data.
- This structured format allows easy manipulation and retrieval of information.

3. Data Visualization

- The **Matplotlib** library is used to generate a **bar chart**, where:
  - The **x-axis** represents the product names.
  - The **y-axis** represents the revenue generated by each product.
- The chart is customized with labels, a title, and color enhancements for better readability.

4. Output & Analysis

- The program prints a tabular representation of the sales data, allowing users to verify their input.
- The generated **bar chart** visually presents the revenue distribution, making it easier to compare product performance.

Implementation

The project was implemented using **Python** with two essential libraries:

- **Pandas:** For data handling and manipulation.

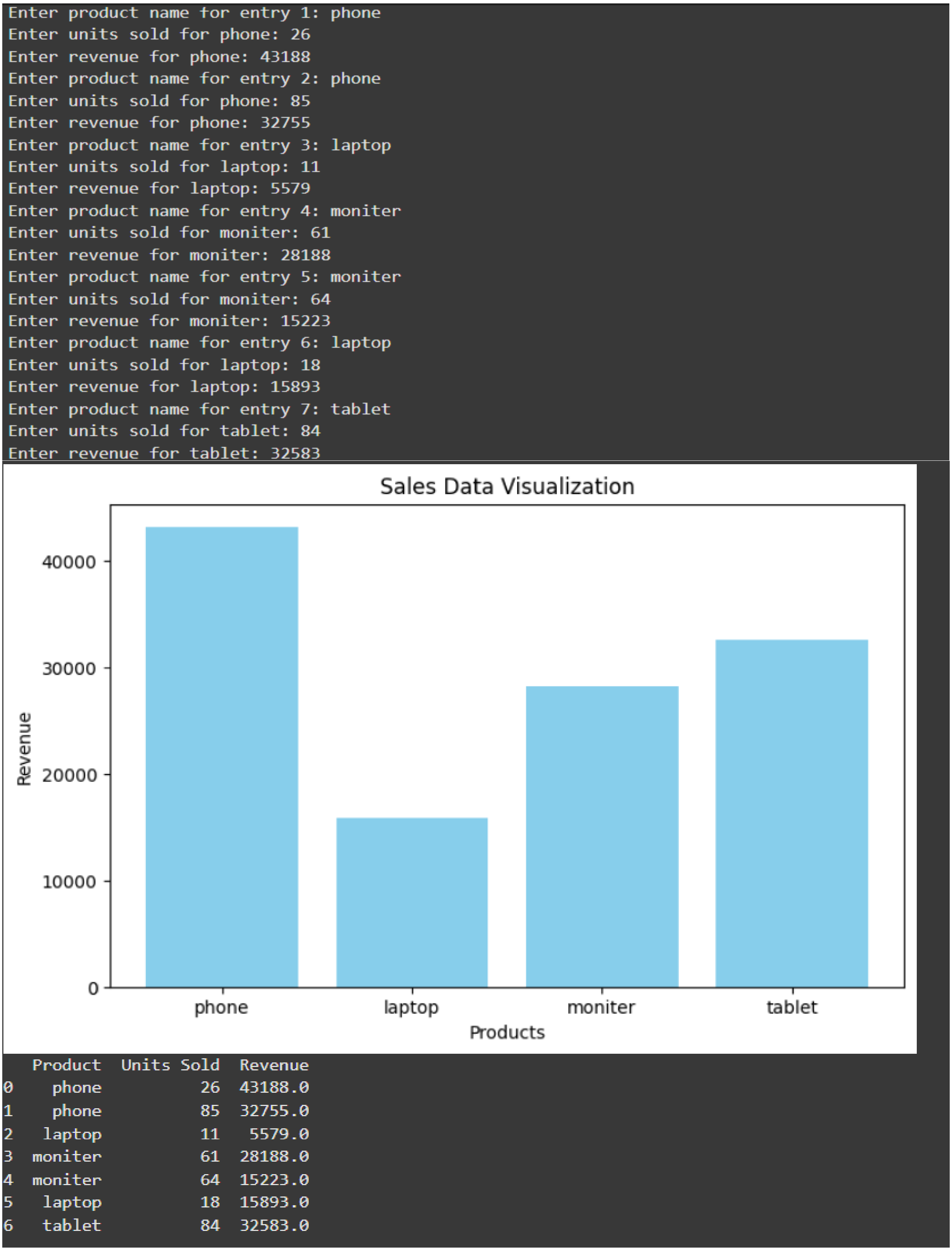
- **Matplotlib:** For generating the visualization.

The program follows a **user-input-driven approach**, where users enter their sales data interactively. After processing, a bar chart is displayed, providing an intuitive visual representation of revenue trends.

Code

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

# Get sales data from user
num_entries = int(input("Enter number of entries: "))
data = {'Product': [], 'Units Sold': [], 'Revenue': []}

for i in range(num_entries):
    product = input(f"Enter product name for entry {i+1}: ")
    units_sold = int(input(f"Enter units sold for {product}: "))
    revenue = float(input(f"Enter revenue for {product}: "))

    data['Product'].append(product)
    data['Units Sold'].append(units_sold)
    data['Revenue'].append(revenue)

# Convert data to DataFrame
df = pd.DataFrame(data)

# Create a bar chart
plt.figure(figsize=(8, 5))
plt.bar(df['Product'], df['Revenue'], color='skyblue')

# Add labels and title
plt.xlabel('Products')
plt.ylabel('Revenue')
plt.title('Sales Data Visualization')

# Show the plot
plt.show()

# Print sales data
print(df)
```

Output



```
Enter product name for entry 1: phone
Enter units sold for phone: 26
Enter revenue for phone: 43188
Enter product name for entry 2: phone
Enter units sold for phone: 85
Enter revenue for phone: 32755
Enter product name for entry 3: laptop
Enter units sold for laptop: 11
Enter revenue for laptop: 5579
Enter product name for entry 4: moniter
Enter units sold for moniter: 61
Enter revenue for moniter: 28188
Enter product name for entry 5: moniter
Enter units sold for moniter: 64
Enter revenue for moniter: 15223
Enter product name for entry 6: laptop
Enter units sold for laptop: 18
Enter revenue for laptop: 15893
Enter product name for entry 7: tablet
Enter units sold for tablet: 84
Enter revenue for tablet: 32583
```

```
  Product  Units Sold  Revenue
0   phone          26  43188.0
1   phone          85  32755.0
2  laptop          11   5579.0
3 moniter          61  28188.0
4 moniter          64  15223.0
5  laptop          18  15893.0
6  tablet          84  32583.0
```

Conclusion

The **Simple Sales Data Visualization** tool provides a straightforward way to analyze sales data, offering both tabular representation and visual insights. By leveraging Python’s Pandas and Matplotlib libraries, the project simplifies the process of data collection, processing, and visualization.

Future Enhancements

While this tool serves as a basic implementation, future improvements could include:

- **Integration with Databases:** To automatically fetch sales data instead of manual input.
- **Multiple Visualization Options:** Such as pie charts or line graphs for more detailed analysis.
- **Export Features:** Allowing users to save reports as PDFs or Excel files.
- **Real-Time Data Updates:** Connecting to an online sales database for live tracking.

By implementing these enhancements, the tool can be further developed into a more advanced and automated sales analysis system.