SALES DATA ANALYSIS PROJECT

1

2

3

4

5

6

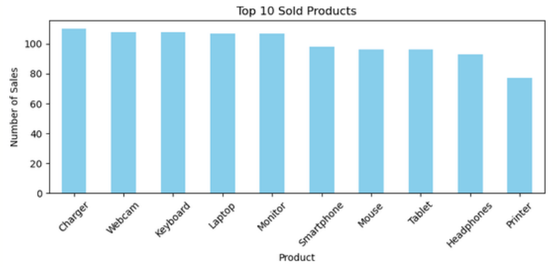
7

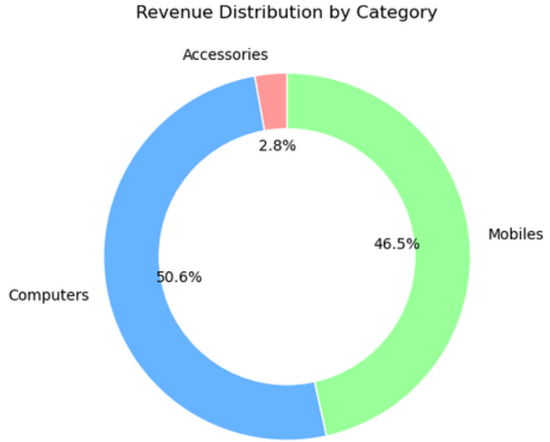
8

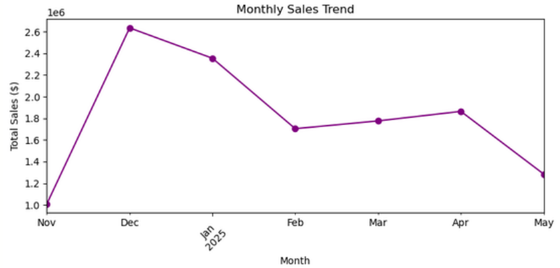
9

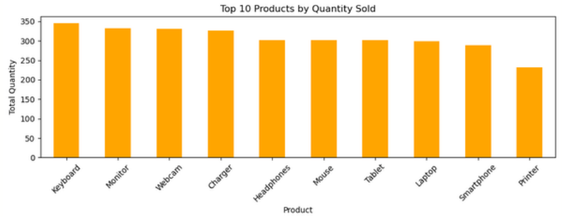
Hello everyone,

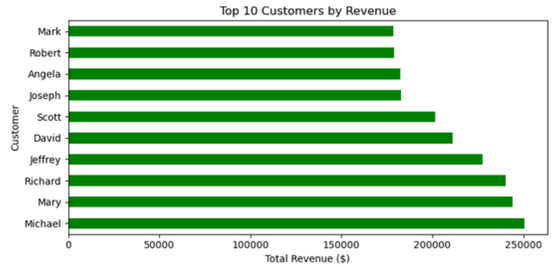
In this project, I performed a complete sales data analysis using Python and Pandas — starting from raw CSV data to delivering key business insights through visualizations created with Matplotlib. The project involved data cleaning, transformation, and exploratory data analysis to uncover trends, top-performing products, customer behavior, and monthly revenue patterns.











Objectives:

- To analyze electronic sales data from raw CSV files using Python and Pandas.
- To clean and transform the dataset for better usability and consistency.
- To uncover patterns and trends related to products, customers, and revenue.
- To present business insights through clear and engaging visualizations.
- To help stakeholders make informed decisions based on data-driven insights.

Business Request:

The company provided a raw sales dataset and requested an analysis to help them answer key business questions, such as:

- Which products are selling the most?
- What is the distribution of revenue across product categories?
- Who are the top customers?
- How are the monthly sales performing?
- What are the total quantities sold per product?

The goal was to transform the data into meaningful visual insights that could guide strategic decision-making.

What I Did:

1
2
3
4
5
6
7
8
9

I started by reading and exploring the raw data, followed by cleaning and transforming it. Then, I conducted exploratory data analysis (EDA) using Pandas and visualized key insights with Matplotlib. Below are selected sections of my code along with their respective outputs:

```python
# Import necessary libraries once
import pandas as pd
import matplotlib.pyplot as plt
```

Imported pandas and matplotlib for data handling and plotting

```python
# Read the CSV file into a DataFrame
df = pd.read_csv("electronics_sales_data.csv")
df['Date'] = pd.to_datetime(df['Date'])

# Basic Overview
print("Shape:", df.shape)
print("\nData Types:\n", df.dtypes)
print("\nSummary Statistics:\n", df.describe())

# Unique Counts
print("\nUnique Products:", df['Product'].nunique())
print("Unique Categories:", df['Category'].nunique())
print("Unique Customers:", df['Customer'].nunique())

# Check for missing values
print("\nMissing Values:\n", df.isnull().sum())
```

Loaded and explored the dataset: checked structure, types, stats, unique counts, and missing values.

```
Shape: (1000, 8)

Data Types:
 Order ID             int64
Date         datetime64[ns]
Customer             object
Product              object
Category             object
Quantity              int64
Price                 int64
Total                 int64
dtype: object

Summary Statistics:
          Order ID     Quantity         Price          Total
count  1000.000000  1000.000000   1000.000000   1000.000000
mean    500.500000     3.058000   4283.700000  12626.400000
std     288.819436     1.426419   5313.410076  18216.592224
min       1.000000     1.000000    100.000000    100.000000
25%     250.750000     2.000000    200.000000    600.000000
50%     500.500000     3.000000    350.000000   1750.000000
75%     750.250000     4.000000   8000.000000  16000.000000
max    1000.000000     5.000000  15000.000000  75000.000000

Unique Products: 10
Unique Categories: 3
Unique Customers: 357

Missing Values:
 Order ID    0
Date         0
Customer     0
Product      0
Category     0
Quantity     0
Price        0
Total        0
dtype: int64
```

Displayed the first 5 rows to get an initial look at the dataset's structure and content

```
# Show the first 5 rows of the data
df.head()
```

	Order ID	Date	Customer	Product	Category	Quantity	Price	Total
0	1	2025-02-02	Michael	Webcam	Accessories	5	350	1750
1	2	2025-03-17	Phillip	Webcam	Accessories	5	350	1750
2	3	2025-03-24	Joshua	Laptop	Computers	2	15000	30000
3	4	2024-11-20	Brandon	Headphones	Accessories	3	200	600
4	5	2025-02-19	Lisa	Webcam	Accessories	5	350	1750

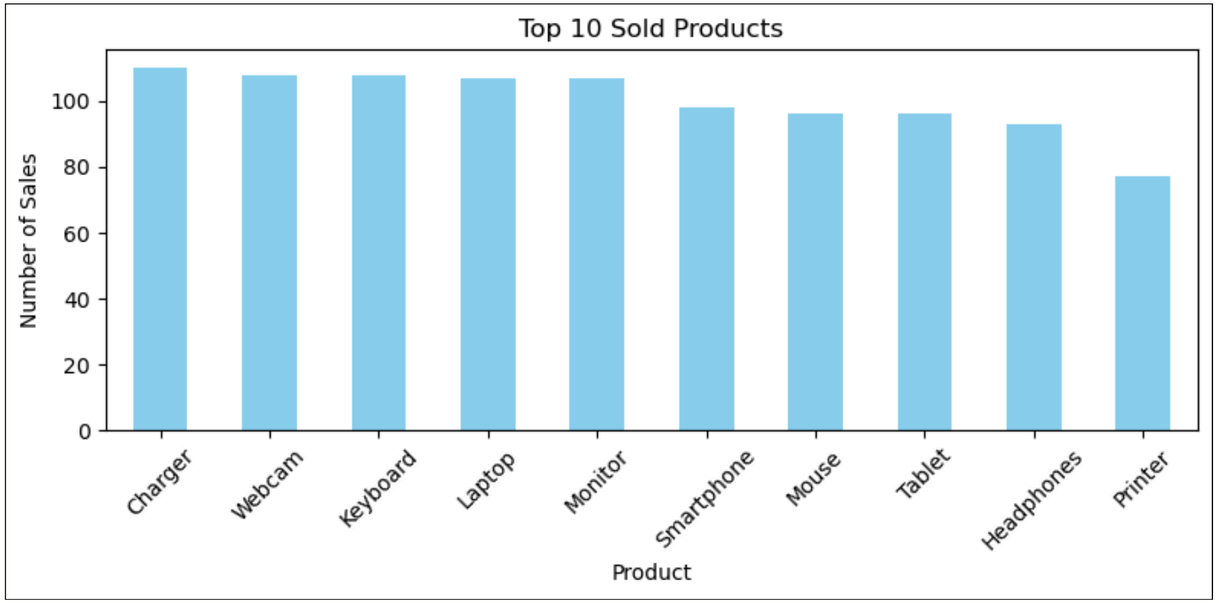
```
# Top 10 most sold products
top_products = df['Product'].value_counts().head(10)
top_products
```

```
Charger       110
Webcam        108
Keyboard      108
Laptop        107
Monitor       107
Smartphone     98
Mouse          96
Tablet         96
Headphones     93
Printer        77
Name: Product, dtype: int64
```

Identified the top 10 most frequently sold products based on sales count

Visualized the top 10 best-selling products using a bar chart for quick comparison

```
# Top 10 Sold Products
top_products = df['Product'].value_counts().head(10)
plt.figure(figsize=(8,4))
top_products.plot(kind='bar', title='Top 10 Sold Products', color='skyblue')
plt.xlabel('Product')
plt.ylabel('Number of Sales')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



```python
# Total Revenue
total_revenue = df['Total'].sum()
print("\nTotal Revenue: $", total_revenue)
```

```
Total Revenue: $ 12626400
```

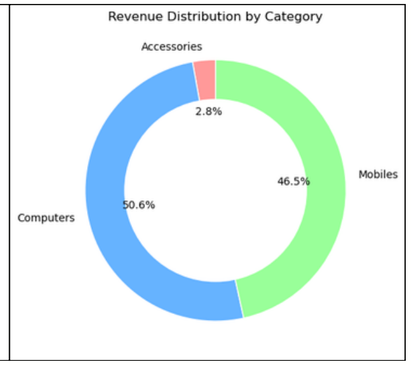
Calculated the overall total revenue generated from all sales

```python
# Group data by 'Category' and sum total revenue for each category
category_revenue = df.groupby('Category')['Total'].sum().sort_values(ascending=False)
print(category_revenue)
```

```
Category
Computers      6393000
Mobiles        5876000
Accessories     357400
Name: Total, dtype: int64
```

Grouped sales by category to identify which product categories generated the most revenue

```python
# Revenue Distribution by Category (Donut Chart Only)
category_revenue = df.groupby('Category')['Total'].sum()
colors = ['#ff9999','#66b3ff','#99ff99']
plt.pie(category_revenue, labels=category_revenue.index, autopct='%1.1f%%', startangle=90, colors=colors, wedgeprops={'
centre_circle = plt.Circle((0,0),0.70,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.title('Revenue Distribution by Category')
plt.tight_layout()
plt.show()
```



Visualized category-wise revenue share using a donut chart to highlight the contribution of each category

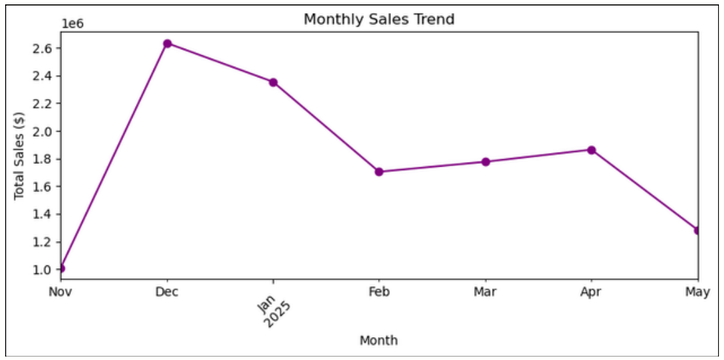
```python
# Extract month and year from 'Date' column to analyze monthly sales
df['Month'] = df['Date'].dt.to_period('M')

# Group data by 'Month' and sum the total sales for each month, then sort descending
monthly_sales = df.groupby('Month')['Total'].sum().sort_values(ascending=False)
print(monthly_sales)
```

```
Month
2024-12    2634900
2025-01    2354550
2025-04    1863750
2025-03    1775900
2025-02    1703950
2025-05    1284650
2024-11    1008700
Freq: M, Name: Total, dtype: int64
```

Aggregated total sales by month and sorted months by sales volume to identify top-performing periods

```python
# Monthly Sales Trend
df['Month'] = df['Date'].dt.to_period('M')
monthly_sales = df.groupby('Month')['Total'].sum().sort_index()
plt.figure(figsize=(8,4))
monthly_sales.plot(kind='line', marker='o', title='Monthly Sales Trend', color='purple')
plt.xlabel('Month')
plt.ylabel('Total Sales ($)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Plotted monthly sales trend over time to visualize sales patterns and seasonality

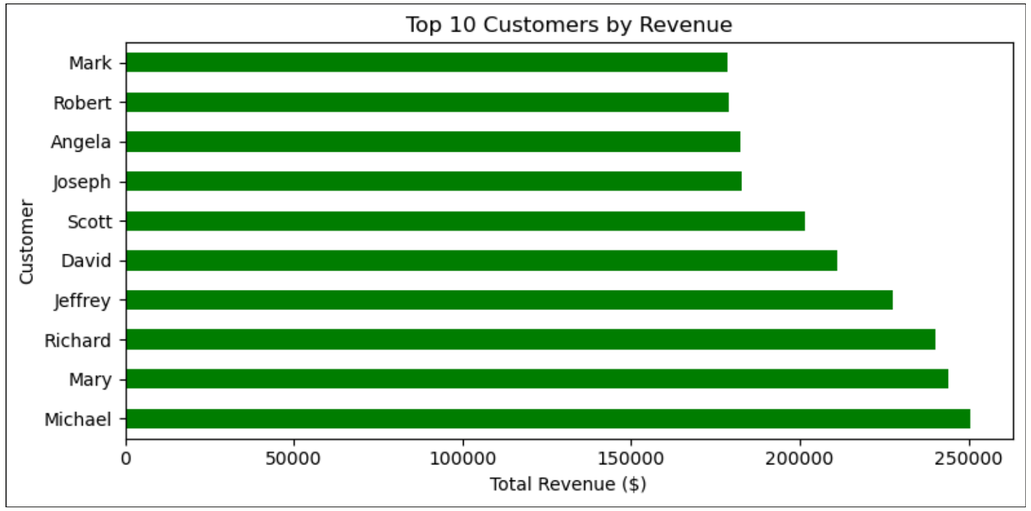
```python
# Find top 10 customers by total revenue they contributed
top_customers = df.groupby('Customer')['Total'].sum().sort_values(ascending=False).head(10)
print(top_customers)
```

```
Customer
Michael    250650
Mary       244050
Richard    240300
Jeffrey    227700
David      211300
Scott      201400
Joseph     182950
Angela     182250
Robert     178950
Mark       178450
Name: Total, dtype: int64
```

Identified the top 10 customers contributing the highest total revenue

```python
# Top 10 Customers by Revenue
top_customers = df.groupby('Customer')['Total'].sum().sort_values(ascending=False).head(10)
plt.figure(figsize=(8,4))
top_customers.plot(kind='barh', color='green', title='Top 10 Customers by Revenue')
plt.xlabel('Total Revenue ($)')
plt.ylabel('Customer')
plt.tight_layout()
plt.show()
```

Visualized the top 10 customers by their total revenue using a horizontal bar chart



```python
# Average Order Value
average_order_value = df['Total'].mean()
print("\nAverage Order Value: $", round(average_order_value, 2))
```

```
Average Order Value: $ 12626.4
```

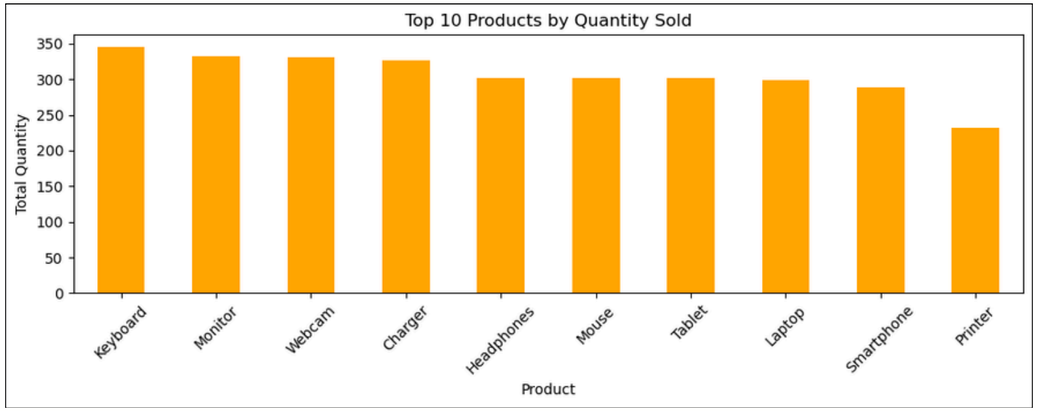
Calculated the average order value across all sales transactions

```python
# Total quantity sold per product
product_quantity = df.groupby('Product')['Quantity'].sum().sort_values(ascending=False)
print(product_quantity)
```

```
Product
Keyboard      345
Monitor       332
Webcam        331
Charger       327
Headphones    301
Mouse         301
Tablet        301
Laptop        299
Smartphone    289
Printer       232
Name: Quantity, dtype: int64
```

Summed the total quantity sold for each product to identify best-selling items by volume

```python
# Total Quantity Sold per Product
product_quantity = df.groupby('Product')['Quantity'].sum().sort_values(ascending=False)
plt.figure(figsize=(10,4))
product_quantity.head(10).plot(kind='bar', color='orange', title='Top 10 Products by Quantity Sold')
plt.xlabel('Product')
plt.ylabel('Total Quantity')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Displayed a bar chart showing the top 10 products by total quantity sold, highlighting customer demand.

Sales Data Analysis Report – Electronics Store

1 2 3 4 5 6 7 8 9 10

1. Overview

- Total Orders:1,000
- Unique Products: 10
- Unique Categories: 3
- Unique Customers: 357
- Total Revenue: $12,626,400

2. Key Insights

2.1 Top Selling Products (by Quantity)

- The most sold product is Keyboard, followed by Monitor and Webcam.
- Indicates consistent customer demand for input and accessory devices.

2.2 Revenue by Product Category

- Computers category generated the highest total revenue: $6,393,000, followed by Mobiles and Accessories.
- Highlights how high-priced items like laptops and monitors drive more revenue despite lower quantities.

2.3 Monthly Sales Trend

- December 2024 had the highest total sales, followed by January 2025 and April 2025.
- November 2024 recorded the lowest revenue.- Suggests seasonal shopping peaks related to holidays and promotions.

2.4 Top Customers (by Total Spending)

- The highest paying customer is Michael, contributing $250,650 in total.
- A small group of top customers contributed a significant share of the total revenue

2.5 Average Order Value

- The average amount spent per order is approximately $12,626.40.

3. Visualizations Used:

- Top 10 Sold Products- Bar Chart
- Revenue by Product Category- Donut Chart
- Monthly Revenue Trends- Line Chart
- Top Customers by Revenue- Horizontal Bar Chart
- Total Quantity Sold per Product- Bar Chart

4. Recommendations

- Focus marketing efforts on best-selling products like Keyboards, Monitors, and Webcams.
- Bundle Accessories with high-value products to increase total order value.
- Offer loyalty rewards or exclusive discounts to top customers for retention.
- Launch sales campaigns during peak months (e.g., December, January).
- Investigate reasons behind low performance in November and adapt strategy accordingly.

5. Next Steps

- Integrate geographic or demographic data for enhanced segmentation.
- Track promotional campaigns to analyze their effect on revenue trends.
- Explore further metrics such as payment methods, delivery zones, or time-based purchase behavior for deeper insights

Thanks a lot for your attention!

Phone : +201003173523

Email : a.hares1999@gmail.com

Linkedin: https://www.linkedin.com/in/abdelrhman-mahrous-174b191aa/