EDS MINI PROJECT

GROUP MEMBERS:

GUNGUN WAHANE 666
SANDESH WALUNJ 667
YASH SASANE 681
OM WABALE 685
Shreyash Patil 678

INTRODUCTION:

In this analysis, we delve into an e-commerce company's sales data to gain valuable insights. Using Python libraries like NumPy, pandas, and matplotlib.pyplot, we perform data manipulation and visualization tasks.

MOTIVATION:

- This project aims to utilize Python and its libraries to uncover hidden insights and patterns in the e-commerce company's sales data.
- By analyzing the dataset, the project aims to understand product performance, identify trends, and make data-driven predictions, enabling informed business decisions and improved competitiveness.

DATASET:

	A	B	C	D	E
1	Product ID	Category	Sales Quar	Price	
2	1	Electronics	10	500	
3	2	Clothing	20	1000	
4	3	Electronics	15	800	
5	4	Clothing	30	1200	
6	5	Electronics	12	700	
7	6	Furniture	8	1500	
8	7	Clothing	25	900	
9	8	Electronics	18	600	
10	9	Furniture	10	2000	
11	10	Clothing	35	1100	
12					

DETAILS OF DATASET:

The dataset consists of 10 records representing individual products, with information on product ID, category, sales quantity, and price. By analyzing sales trends, category-wise sales, and price distribution, we extract valuable insights to inform data-driven decision-making for the company.

DATA MANIPULATION:

We utilize Python libraries like NumPy and pandas to perform various data manipulation tasks.

These include:

- calculating the total sales quantity

total_sales_quantity = df['Sales Quantity'].sum()

- finding the average price

average_price = df['Price'].mean()

- identifying the maximum sales quantity

max_sales_quantity = df['Sales Quantity'].max()

- determining the number of products in specific categories

electronics_products = df[df['Category'] == 'Electronics']

- num_electronics_products = len(electronics_products

```
plt.bar(df['Product ID'], df['Sales Quantity'])
plt.xlabel('Product ID')
plt.ylabel('Sales Quantity')
plt.title('Sales Quantity per Product')
```

```
plt.plot(df['Time'], df['Sales Quantity'])
plt.xlabel('Time')
plt.xticks(rotation=45)
plt.ylabel('Sales Quantity')
plt.title('Sales Quantity Trend over Time')
```

```
# to add time column
df['Time'] = pd.date_range(start='2023-01-01',
        periods=len(df), freq='D')
```

```
category_counts = df['Category'].value_counts()
plt.pie(category_counts,
labels=category_counts.index, autopct='%1.1f%%')
plt.title('Product Distribution by Category')
```

```
category_counts = df['Category'].value_counts()
plt.pie(category_counts, labels=category_counts.index,
        autopct='%1.1f%%')
plt.title('Product Distribution by Category')
```

Library Used to Plot Graph - MATPLOTLIB

• The library commonly used to plot graphs in Python is called Matplotlib. Matplotlib is a popular data visualization library that provides a wide range of functionalities for creating various types of plots, charts, and graphs.

• Matplotlib provides many customization options and supports various types of plots such as scatter plots, bar plots, histograms, etc.

• You can customize every aspect of your plot, including colors, line styles, markers , labels, titles, axes and legends

DATA VISUALISATION:

- To enhance the understanding of the sales data, we utilize the matplotlib.pyplot library to create visually appealing graphs and charts.
- These include bar charts depicting sales quantity per product, line graphs showcasing sales quantity trends over time
- pie charts illustrating the percentage distribution of products in each category
- Histograms displaying the price distribution of the products.

```
total_sales_quantity = df['Sales Quantity'].sum()
print("Total sales quantity:", total_sales_quantity)
```


```
total_sales_quantity = df['Sales Quantity'].sum()
print("Total sales quantity:", total_sales_quantity)
```


```
max_sales_quantity = df['Sales Quantity'].max()
print("Maximum sales quantity:",
max_sales_quantity)
```

```
electronics_products = df[df['Category'] == 'Electronics']
num_electronics_products = len(electronics_products)
print("Number of products under Electronics category:",
num_electronics_products)
```

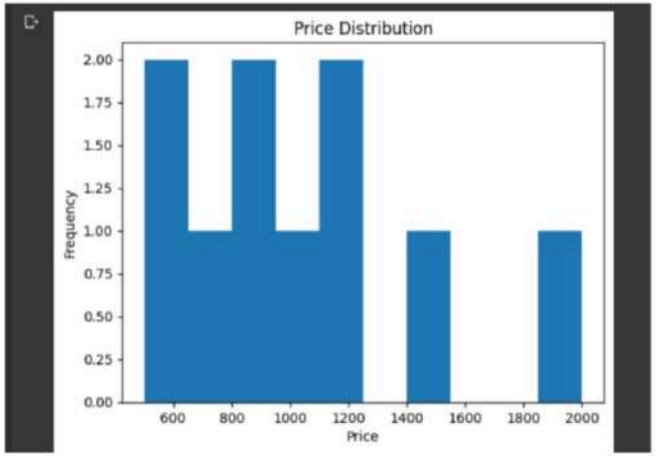
```python
# Question 5: Calculate the total sales quantity for each category
total_sales_by_category = df.groupby('Category')['Sales
Quantity'].sum()
print("Total sales quantity by category:")
print(total_sales_by_category)
```

```
Total sales quantity by category:
Category
Clothing       110
Electronics     55
Furniture       18
Name: Sales Quantity, dtype: int64
```

```python
# Question 6: Find the product(s) with the highest price
max_price = df['Price'].max()
products_with_max_price = df[df['Price'] == max_price]
print("Product(s) with the highest price:")
print(products_with_max_price)
```

```
Product(s) with the highest price:
   Product ID   Category  Sales Quantity  Price
8           9  Furniture              10   2000
```

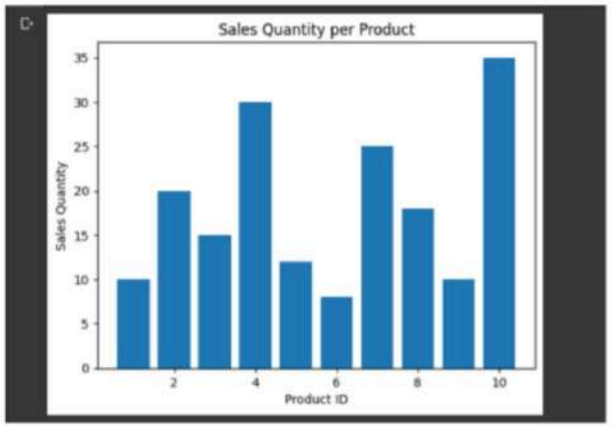
```python
# Question 10: Generate a histogram to display the price distribution
of the products
plt.hist(df['Price'], bins=10)
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.title('Price Distribution')
plt.show()
```



```
# Question 7: Create a bar chart to visualize the sales quantity for
each product
plt.bar(df['Product ID'], df['Sales Quantity'])
plt.xlabel('Product ID')
```

```
plt.ylabel('Sales Quantity')
plt.title('Sales Quantity per Product')
plt.show()
```

OUTPUT:




```
df['Time'] = pd.date_range(start='2023-01-01', periods=len(df),
freq='D')


# Question 8: Plot a line graph to show the trend of sales quantity
over time
plt.plot(df['Time'], df['Sales Quantity'])
plt.xlabel('Time')
plt.xticks(rotation=45)
plt.ylabel('Sales Quantity')
plt.title('Sales Quantity Trend over Time')
plt.show()
```

STUDIO SHODWE

OUTPUT:

The output of this analysis will include valuable insights into sales trends, category-wise sales, and price distribution. Through data visualization and predictive techniques, we aim to provide actionable information that will aid in making informed business decisions.


STUDIO SHODWE
Thank You
SLIDE PRESENTATIONS DESIGN