PYTHON & POWER BI EXERCISE

GLOBAL SUPERSTORE DATASET

DHANUSHA SRI M J

20MIY0054

Introduction

The "Global Superstore Sales Analysis" project leverages Python to explore and visualize sales data, aiming to uncover significant trends and insights. The analysis includes data cleaning, transformation, and visualization, providing a comprehensive view of sales performance, customer behavior, and regional differences.

Detailed Analysis

1. Data Exploration

- Libraries and Data Importation: Utilized NumPy, pandas, seaborn, and matplotlib for data handling. The dataset includes 24 columns and 51,290 rows.
- Column Overview: Important columns include Order Date, Ship Mode, Customer ID, Segment, Category, Sales, Quantity, and Profit.
- Data Inspection: Used data.info() and data.describe() to examine the dataset structure, detect missing values, and obtain summary statistics.
- Order Priority Distribution: Analyzed the frequency of each order priority level to understand its distribution.

2. Data Cleaning

- Copying and Renaming Columns: Created a copy of the dataset and renamed columns for clarity and consistency.
- Date Conversion: Converted order_date and ship_date from strings to date formats to facilitate time-based analysis.
- Adding Sales Year Column: Added a sales_year column to segment data by year.
- Categorical Data Optimization: Converted categorical columns to the category datatype to improve processing efficiency.

3. Data Analysis

- Correlation Analysis: Calculated the correlation between shipping costs and sales, as well as shipping costs and profits, to understand their relationships.
- Order Priority and Profit Analysis: Aggregated sales and profit by order priority and category to identify high-performing segments.

- Sales Performance by Region: Aggregated sales data by region to analyze regional performance.
- Customer Segmentation: Analyzed sales and profit by customer ID, segmented customers based on sales volume into 'Low', 'Medium', and 'High' categories.
- Top Products: Identified the top 5 products by sales and profit.
- Profit Margin Analysis: Calculated profit margins for product categories and sub-categories, sorting by profitability.
- Segment Contributions: Analyzed sales contributions by segment to understand their impact.
- Monthly Trends: Analyzed sales and profit trends on a monthly basis to observe temporal variations.

4. Data Visualization

- Total Sales and Profit by Category: Created bar charts to visualize total sales and profit across categories.
  Profit Margin Distribution: Plotted a histogram of profit margins by sub-category, showing the frequency distribution.
- Sales by Category per Segment: Used a stacked bar chart to display sales distribution by category across different customer segments.
- Monthly Sales and Profit Trends: Created line charts to visualize monthly trends in sales and profit.
- Sales Contribution by Segment: Visualized sales contributions by segment using a bar chart.

5. Conclusion

The **Global Superstore Sales Analysis** project provides valuable insights into sales performance, customer behavior, and regional differences. Key findings include:

⇒ Understanding how shipping costs impact sales and profits.
⇒ Identifying top-performing categories and products.
⇒ Analyzing customer segments and their sales contributions.
⇒ Observing trends in sales and profit over time.

The visualizations and analyses support informed decision-making and strategic planning, offering a clear picture of sales dynamics within the global superstore context.

GLOBAL SUPERSTORE SALES ANALYSIS

The "Global Superstore Sales Analysis" project involves analyzing sales data using Python to uncover key trends. The task requires data exploration and visualization, with a well-documented Python script or Jupyter Notebook to highlight insights.

1. Data Exploration

Import libraries

```
#IMPORTING LIBRARIES
import numpy as np
import pandas as pd
import seaborn as sns

import matplotlib.pyplot as plt
from matplotlib import ticker as mtick

#IMPORTING DATASET
data = pd.read_csv('C:/Users/DHANUSHA/Desktop/GD.csv',  encoding = 
'unicode_escape', engine ='python')

data
```

```
       Row ID         Order ID  Order Date   Ship Date       Ship Mode
\
0       32298   CA-2012-124891   7-31-2012   7-31-2012        Same Day

1       26341    IN-2013-77878  02-05-2013  02-07-2013    Second Class

2       25330    IN-2013-71249  10-17-2013  10-18-2013     First Class

3       13524  ES-2013-1579342   1-28-2013   1-30-2013     First Class

4       47221     SG-2013-4320  11-05-2013  11-06-2013        Same Day

...       ...              ...         ...         ...             ...

51285   29002    IN-2014-62366   6-19-2014   6-19-2014        Same Day

51286   35398   US-2014-102288   6-20-2014   6-24-2014  Standard Class

51287   40470   US-2013-155768  12-02-2013  12-02-2013        Same Day

51288    9596   MX-2012-140767   2-18-2012   2-22-2012  Standard Class
```

```
51289    6147   MX-2012-134460   5-22-2012   5-26-2012    Second Class


      Customer ID     Customer Name      Segment           City  \
0        RH-19495       Rick Hansen     Consumer  New York City
1        JR-16210     Justin Ritter    Corporate      Wollongong
2        CR-12730      Craig Reiter     Consumer        Brisbane
3        KM-16375  Katherine Murray  Home Office          Berlin
4         RH-9495       Rick Hansen     Consumer           Dakar
...           ...               ...          ...             ...
51285    KE-16420   Katrina Edelman    Corporate            Kure
51286    ZC-21910   Zuschuss Carroll    Consumer         Houston
51287    LB-16795    Laurel Beltran  Home Office          Oxnard
51288    RB-19795        Ross Baird  Home Office        Valinhos
51289    MC-18100     Mick Crebagga     Consumer        Tipitapa

                 State  ...        Product ID        Category Sub-
Category  \
0             New York  ...   TEC-AC-10003033       Technology
Accessories
1      New South Wales  ...   FUR-CH-10003950        Furniture
Chairs
2           Queensland  ...   TEC-PH-10004664       Technology
Phones
3               Berlin  ...   TEC-PH-10004583       Technology
Phones
4                Dakar  ...  TEC-SHA-10000501       Technology
Copiers
...                ...  ...               ...              ...
...
51285        Hiroshima  ...   OFF-FA-10000746  Office Supplies
Fasteners
51286            Texas  ...   OFF-AP-10002906  Office Supplies
Appliances
51287       California  ...   OFF-EN-10001219  Office Supplies
Envelopes
51288        S\o Paulo  ...   OFF-BI-10000806  Office Supplies
Binders
51289          Managua  ...   OFF-PA-10004155  Office Supplies
Paper

                                            Product Name     Sales
Quantity  \
0      Plantronics CS510 - Over-the-Head monaural Wir...  2309.650
7
1             Novimex Executive Leather Armchair, Black  3709.395
9
2                     Nokia Smart Phone, with Caller ID  5175.171
9
3                        Motorola Smart Phone, Cordless  2892.510
```

```
5
4                             Sharp Wireless Fax, High-Speed  2832.960
8
...                                                      ...       ...
...
51285                         Advantus Thumb Tacks, 12 Pack    65.100
5
51286  Hoover Replacement Belt for Commercial Guardsm...     0.444
1
51287       #10- 4 1/8" x 9 1/2" Security-Tint Envelopes    22.920
3
51288                               Acco Index Tab, Economy    13.440
2
51289            Eaton Computer Printout Paper, 8.5 x 11    61.380
3

      Discount    Profit  Shipping Cost  Order Priority
0          0.0  762.1845         933.57        Critical
1          0.1 -288.7650         923.63        Critical
2          0.1  919.9710         915.49          Medium
3          0.1  -96.5400         910.16          Medium
4          0.0  311.5200         903.04        Critical
...        ...       ...            ...             ...
51285      0.0    4.5000           0.01          Medium
51286      0.8   -1.1100           0.01          Medium
51287      0.0   11.2308           0.01            High
51288      0.0    2.4000           0.00          Medium
51289      0.0    1.8000           0.00            High

[51290 rows x 24 columns]
```

```
data = pd.DataFrame(data)
```

```
#DATA HEAD
data.head()
```

```
   Row ID         Order ID  Order Date   Ship Date     Ship Mode
Customer ID  \
0   32298   CA-2012-124891   7-31-2012   7-31-2012      Same Day
RH-19495
1   26341    IN-2013-77878  02-05-2013  02-07-2013  Second Class
JR-16210
2   25330    IN-2013-71249  10-17-2013  10-18-2013   First Class
CR-12730
3   13524  ES-2013-1579342   1-28-2013   1-30-2013   First Class
KM-16375
4   47221     SG-2013-4320  11-05-2013  11-06-2013      Same Day
RH-9495

      Customer Name      Segment          City           State  ...
```

```
\
0       Rick Hansen     Consumer  New York City         New York  ...

1     Justin Ritter    Corporate     Wollongong  New South Wales  ...

2      Craig Reiter     Consumer       Brisbane       Queensland  ...

3  Katherine Murray  Home Office         Berlin           Berlin  ...

4       Rick Hansen     Consumer          Dakar            Dakar  ...


         Product ID    Category Sub-Category  \
0   TEC-AC-10003033  Technology  Accessories
1   FUR-CH-10003950   Furniture       Chairs
2   TEC-PH-10004664  Technology       Phones
3   TEC-PH-10004583  Technology       Phones
4  TEC-SHA-10000501  Technology      Copiers

                                        Product Name     Sales
Quantity  \
0  Plantronics CS510 - Over-the-Head monaural Wir...  2309.650
7
1          Novimex Executive Leather Armchair, Black  3709.395
9
2                  Nokia Smart Phone, with Caller ID  5175.171
9
3                     Motorola Smart Phone, Cordless  2892.510
5
4                     Sharp Wireless Fax, High-Speed  2832.960
8

  Discount    Profit  Shipping Cost  Order Priority
0      0.0  762.1845         933.57        Critical
1      0.1 -288.7650         923.63        Critical
2      0.1  919.9710         915.49          Medium
3      0.1  -96.5400         910.16          Medium
4      0.0  311.5200         903.04        Critical

[5 rows x 24 columns]
```

Column Description

Row ID, Order ID, Order Date , Ship Date , Ship Mode , Customer ID , Customer Name , Segment , City , State , Country , Postal Code , Market , Region , Product ID , Category , Sub-Category , Product Name , Sales , Quantity , Discount , Profit , Shipping Cost , Order Priority

```
#GETTING DATA INFO
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Row ID          51290 non-null  int64  
 1   Order ID        51290 non-null  object 
 2   Order Date      51290 non-null  object 
 3   Ship Date       51290 non-null  object 
 4   Ship Mode       51290 non-null  object 
 5   Customer ID     51290 non-null  object 
 6   Customer Name   51290 non-null  object 
 7   Segment         51290 non-null  object 
 8   City            51290 non-null  object 
 9   State           51290 non-null  object 
 10  Country         51290 non-null  object 
 11  Postal Code     9994 non-null   float64
 12  Market          51290 non-null  object 
 13  Region          51290 non-null  object 
 14  Product ID      51290 non-null  object 
 15  Category        51290 non-null  object 
 16  Sub-Category    51290 non-null  object 
 17  Product Name    51290 non-null  object 
 18  Sales           51290 non-null  float64
 19  Quantity        51290 non-null  int64  
 20  Discount        51290 non-null  float64
 21  Profit          51290 non-null  float64
 22  Shipping Cost   51290 non-null  float64
 23  Order Priority  51290 non-null  object 
dtypes: float64(5), int64(2), object(17)
memory usage: 9.4+ MB

#GETTING SUMMARY
data.describe()

            Row ID   Postal Code         Sales      Quantity
Discount  \
count  51290.00000   9994.000000  51290.000000  51290.000000
51290.000000
mean   25645.50000  55190.379428    246.490581      3.476545
0.142908
std    14806.29199  32063.693350    487.565361      2.278766
0.212280
min        1.00000   1040.000000      0.444000      1.000000
0.000000
25%    12823.25000  23223.000000     30.758625      2.000000
0.000000
50%    25645.50000  56430.500000     85.053000      3.000000
0.000000
75%    38467.75000  90008.000000    251.053200      5.000000
```

```
0.200000
max    51290.00000  99301.000000  22638.480000     14.000000
0.850000

             Profit  Shipping Cost
count  51290.000000   51290.000000
mean      28.610982      26.375915
std      174.340972      57.296804
min    -6599.978000       0.000000
25%        0.000000       2.610000
50%        9.240000       7.790000
75%       36.810000      24.450000
max     8399.976000     933.570000
```

The Global Superstore dataset contains 24 columns and 51290 rows

```
#ORDER PRIORITY DISTRIBUTION
priority_distribution = df['Order Priority'].value_counts()
print(priority_distribution)
```

```
Medium      29433
High        15501
Critical     3932
Low          2424
Name: Order Priority, dtype: int64
```

```
#DATA WITH MISSING VALUES
data.isnull().sum()
```

```
Row ID                0
Order ID              0
Order Date            0
Ship Date             0
Ship Mode             0
Customer ID           0
Customer Name         0
Segment               0
City                  0
State                 0
Country               0
Postal Code       41296
Market                0
Region                0
Product ID            0
Category              0
Sub-Category          0
Product Name          0
Sales                 0
Quantity              0
Discount              0
```

```
Profit                    0
Shipping Cost             0
Order Priority            0
Year                      0
Month                     0
dtype: int64

#GETTING DATA TYPES
data.dtypes

Row ID              int64
Order ID           object
Order Date         object
Ship Date          object
Ship Mode          object
Customer ID        object
Customer Name      object
Segment            object
City               object
State              object
Country            object
Postal Code       float64
Market             object
Region             object
Product ID         object
Category           object
Sub-Category       object
Product Name       object
Sales             float64
Quantity            int64
Discount          float64
Profit            float64
Shipping Cost     float64
Order Priority     object
dtype: object
```

2. Data Cleaning

1. Copied the dataset for cleaning.

```
data = data.copy()
```

2. Renamed column names for clear understanding.

```
data.columns = data.columns.str.replace(' ', '_').str.lower()
data.columns

Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'city', 'state',
'country',
       'postal_code', 'market', 'region', 'product_id', 'category',
       'sub-category', 'product_name', 'sales', 'quantity',
'discount',
       'profit', 'shipping_cost', 'order_priority'],
      dtype='object')
```

3. Converted strings to dates.

```
data.columns = data.columns.str.replace(' ', '_').str.lower()
data.columns

Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'city', 'state',
'country',
       'postal_code', 'market', 'region', 'product_id', 'category',
       'sub-category', 'product_name', 'sales', 'quantity',
'discount',
       'profit', 'shipping_cost', 'order_priority'],
      dtype='object')

data[['order_date', 'ship_date']].dtypes
print('Initial date - ' + str(data['order_date'].min()))
print('Final date - ' + str(data['order_date'].max()))

Initial date - 01-01-2011
Final date - 9-30-2014
```

4. Verified the changes.

```
data.dtypes

row_id              int64
order_id           object
order_date         object
ship_date          object
ship_mode          object
customer_id        object
customer_name      object
segment            object
city               object
state              object
country            object
postal_code       float64
market             object
region             object
product_id         object
category           object
sub-category       object
product_name       object
sales             float64
quantity            int64
```

```
discount          float64
profit            float64
shipping_cost     float64
order_priority     object
dtype: object
```

5. Added a new column.

```
data['sales_year'] = pd.DatetimeIndex(data['order_date']).year
```

6. Converted categorical columns from object to category.

```
new = ['order_priority', 'ship_mode', 'country', 'state', 'region',
'sub-category', 'segment', 'market', 'category']
data[new] = data[new].astype('category')
```

7. Confirmed data types for better optimizations.

```
data.dtypes

row_id              int64
order_id           object
order_date         object
ship_date          object
ship_mode        category
customer_id        object
customer_name      object
segment          category
city               object
state            category
country          category
postal_code       float64
market           category
region           category
product_id         object
category         category
sub-category     category
product_name       object
sales             float64
quantity            int64
discount          float64
profit            float64
shipping_cost     float64
order_priority   category
sales_year          int64
dtype: object
```

3. Data Analysis

```
#DISCOUNT IMPACT ANALYSIS
correlation_sales = df[['Shipping Cost', 'Sales']].corr().iloc[0, 1]
correlation_profit = df[['Shipping Cost', 'Profit']].corr().iloc[0, 1]

print(f"Correlation between Shipping Cost and Sales:
{correlation_sales:.2f}")
print(f"Correlation between Shipping Cost and Profit:
{correlation_profit:.2f}")
```

```
Correlation between Shipping Cost and Sales: 0.77
Correlation between Shipping Cost and Profit: 0.35
```

```
#ORDER PRIORITY ANALYSIS
priority_summary = df.groupby('Order Priority').agg({'Sales': 'sum',
'Profit': 'sum'}).reset_index()
print(priority_summary)
```

```
  Order Priority         Sales         Profit
0       Critical  9.862355e+05  124224.16428
1           High  3.807548e+06  420373.51340
2            Low  5.678259e+05   58655.85098
3         Medium  7.280892e+06  864203.76262
```

```
#TOTAL PROFIT
profit = data.groupby('category')
['profit'].sum().sort_values(ascending=False)
profit
```

```
category
Technology         663778.73318
Office Supplies    518473.83430
Furniture          285204.72380
Name: profit, dtype: float64
```

```
# TOTAL SALES AND PROFIT BY CATEGORY
category_summary = df.groupby('Category').agg({'Sales': 'sum',
'Profit': 'sum'}).reset_index()
print(category_summary)
```

```
          Category         Sales         Profit
0        Furniture  4.110874e+06  285204.72380
1  Office Supplies  3.787070e+06  518473.83430
2       Technology  4.744557e+06  663778.73318
```

```
#SALES PERFORMANCE BY REGION
sales_by_region = df.groupby('Region').agg({'Sales':
'sum'}).reset_index()
print(sales_by_region)
```

```
            Region         Sales
0           Africa  7.837732e+05
1           Canada  6.692817e+04
2        Caribbean  3.242809e+05
3          Central  2.822303e+06
4     Central Asia  7.528266e+05
5             EMEA  8.061613e+05
6             East  6.787812e+05
7            North  1.248166e+06
8       North Asia  8.483098e+05
9          Oceania  1.100185e+06
10           South  1.600907e+06
11  Southeast Asia  8.844232e+05
12            West  7.254578e+05
```

```
#TOTAL SALES
sales = data.groupby('category')
['sales'].sum().sort_values(ascending=False)
sales
```

```
category
Technology         4.744557e+06
Furniture          4.110874e+06
Office Supplies    3.787070e+06
Name: sales, dtype: float64
```

```
#CUSTOMER SEGMENTATION ANALYSIS
customer_summary = df.groupby('Customer ID').agg({'Sales': 'sum',
'Profit': 'sum'}).reset_index()
bins = [0, 500, 1000, 2000]
labels = ['Low', 'Medium', 'High']
customer_summary['Sales Segment'] = pd.cut(customer_summary['Sales'],
bins=bins, labels=labels)
print(customer_summary)
```

```
     Customer ID         Sales      Profit Sales Segment
0       AA-10315  13747.41300   447.69050           NaN
1       AA-10375   5884.19500   677.47740           NaN
2       AA-10480  17695.58978  1516.47518           NaN
3       AA-10645  15343.89070  3051.43900           NaN
4         AA-315   2243.25600   535.56600           NaN
...          ...          ...         ...           ...
1585    YS-21880  18703.60600  3091.59430           NaN
1586    ZC-11910      7.17300   -15.56700           Low
1587    ZC-21910  28472.81926   452.50326           NaN
1588    ZD-11925   2951.22600   478.41600           NaN
1589    ZD-21925   9479.34440  -276.67890           NaN

[1590 rows x 4 columns]
```

```
#TOP PRODUCTS BY SALES AND PROFIT
top_products_sales = df.sort_values(by='Sales',
ascending=False).head(5)
top_products_profit = df.sort_values(by='Profit',
ascending=False).head(5)

print("Top 5 Products by Sales:")
print(top_products_sales)
print("\nTop 5 Products by Profit:")
print(top_products_profit)
```

```
Top 5 Products by Sales:
       Row ID        Order ID Order Date   Ship Date       Ship
Mode  \
12887   33994  CA-2011-145317 2011-03-18   3-23-2011  Standard Class

329     38123  CA-2013-118689 2013-10-03  10-10-2013  Standard Class

14843   39450  CA-2014-140151 2014-03-24   3-26-2014     First Class

7399    33920  CA-2014-127180 2014-10-23  10-25-2014     First Class

290     35487  CA-2014-166709 2014-11-18  11-23-2014  Standard Class


      Customer ID Customer Name      Segment           City
State  ...  \
12887    SM-20320   Sean Miller  Home Office   Jacksonville
Florida  ...
329      TC-20980  Tamara Chand    Corporate      Lafayette
Indiana  ...
14843    RB-19360  Raymond Buch     Consumer        Seattle
Washington  ...
7399     TA-21385  Tom Ashbrook  Home Office  New York City    New
York  ...
290      HL-15040  Hunter Lopez     Consumer         Newark
Delaware  ...

      Sub-Category                                       Product Name
\
12887     Machines  Cisco TelePresence System EX90 Videoconferenci...

329        Copiers              Canon imageCLASS 2200 Advanced Copier

14843      Copiers              Canon imageCLASS 2200 Advanced Copier

7399       Copiers              Canon imageCLASS 2200 Advanced Copier

290        Copiers              Canon imageCLASS 2200 Advanced Copier
```

```
           Sales Quantity Discount     Profit Shipping Cost Order
Priority  \
12887  22638.480        6      0.5 -1811.0784         24.29
Medium
329    17499.950        5      0.0  8399.9760        349.07
Medium
14843  13999.960        4      0.0  6719.9808         20.00
Medium
7399   11199.968        4      0.2  3919.9888         45.98
High
290    10499.970        3      0.0  5039.9856        363.19
Medium

       Year  Month
12887  2011      3
329    2013     10
14843  2014      3
7399   2014     10
290    2014     11

[5 rows x 26 columns]

Top 5 Products by Profit:
       Row ID        Order ID Order Date   Ship Date       Ship
Mode  \
329     38123  CA-2013-118689 2013-10-03  10-10-2013  Standard Class

14843   39450  CA-2014-140151 2014-03-24   3-26-2014     First Class

290     35487  CA-2014-166709 2014-11-18  11-23-2014  Standard Class

122     40336  CA-2013-117121 2013-12-18  12-22-2013  Standard Class

45      35395  CA-2011-116904 2011-09-23   9-28-2011  Standard Class


      Customer ID  Customer Name    Segment         City
State  ...  \
329      TC-20980   Tamara Chand  Corporate    Lafayette
Indiana  ...
14843    RB-19360   Raymond Buch   Consumer      Seattle
Washington  ...
290      HL-15040   Hunter Lopez   Consumer       Newark
Delaware  ...
122      AB-10105  Adrian Barton   Consumer      Detroit
Michigan  ...
45       SC-20095   Sanjit Chand   Consumer  Minneapolis
Minnesota  ...

      Sub-Category                                       Product
```

```
Name  \
329        Copiers             Canon imageCLASS 2200 Advanced Copier

14843      Copiers             Canon imageCLASS 2200 Advanced Copier

290        Copiers             Canon imageCLASS 2200 Advanced Copier

122        Binders  GBC Ibimaster 500 Manual ProClick Binding System

45         Binders              Ibico EPK-21 Electric Binding System


          Sales Quantity Discount     Profit Shipping Cost Order
Priority  \
329    17499.95        5      0.0  8399.9760        349.07
Medium
14843  13999.96        4      0.0  6719.9808         20.00
Medium
290    10499.97        3      0.0  5039.9856        363.19
Medium
122     9892.74       13      0.0  4946.3700        498.70
Medium
45      9449.95        5      0.0  4630.4755        655.61
Medium

       Year  Month
329    2013     10
14843  2014      3
290    2014     11
122    2013     12
45     2011      9

[5 rows x 26 columns]
```

```
# PROFIT MARGIN
sales_category = data.groupby(['category', 'sub-category'],
as_index=False)[['sales', 'profit']].sum()
sales_category['profit_margin'] = sales_category['profit'] /
sales_category['sales']
sales_category =  sales_category.sort_values(by='profit_margin',
ascending=False)
sales_category
```

```
           category sub-category         sales        profit
profit_margin
29  Office Supplies        Paper  2.442917e+05   59207.68270
0.242365
27  Office Supplies       Labels  7.340403e+04   15010.51200
0.204492
24  Office Supplies    Envelopes  1.709043e+05   29601.11630
```

```
0.173203
34       Technology  Accessories  7.492370e+05  129626.30620
0.173011
40       Technology      Copiers  1.509436e+06  258567.54818
0.171301
20  Office Supplies      Binders  4.619115e+05   72449.84600
0.156848
19  Office Supplies          Art  3.720920e+05   57953.91090
0.155752
18  Office Supplies   Appliances  1.011064e+06  141680.58940
0.140130
25  Office Supplies    Fasteners  8.324232e+04   11525.42410
0.138456
47       Technology       Phones  1.706824e+06  216717.00580
0.126971
9         Furniture  Furnishings  3.855783e+05   46967.42550
0.121810
4         Furniture    Bookcases  1.466572e+06  161924.41950
0.110410
31  Office Supplies      Storage  1.127086e+06  108461.48980
0.096232
5         Furniture       Chairs  1.501682e+06  140396.26750
0.093493
32  Office Supplies     Supplies  2.430742e+05   22583.26310
0.092907
45       Technology     Machines  7.790601e+05   58867.87300
0.075563
16        Furniture       Tables  7.570419e+05  -64083.38870      -
0.084650
0         Furniture  Accessories  0.000000e+00       0.00000
NaN
1         Furniture   Appliances  0.000000e+00       0.00000
NaN
2         Furniture          Art  0.000000e+00       0.00000
NaN
3         Furniture      Binders  0.000000e+00       0.00000
NaN
6         Furniture      Copiers  0.000000e+00       0.00000
NaN
7         Furniture    Envelopes  0.000000e+00       0.00000
NaN
8         Furniture    Fasteners  0.000000e+00       0.00000
NaN
10        Furniture       Labels  0.000000e+00       0.00000
NaN
11        Furniture     Machines  0.000000e+00       0.00000
NaN
12        Furniture        Paper  0.000000e+00       0.00000
NaN
```

```
13        Furniture        Phones  0.000000e+00        0.00000
NaN
14        Furniture       Storage  0.000000e+00        0.00000
NaN
15        Furniture      Supplies  0.000000e+00        0.00000
NaN
17  Office Supplies   Accessories  0.000000e+00        0.00000
NaN
21  Office Supplies     Bookcases  0.000000e+00        0.00000
NaN
22  Office Supplies        Chairs  0.000000e+00        0.00000
NaN
23  Office Supplies       Copiers  0.000000e+00        0.00000
NaN
26  Office Supplies   Furnishings  0.000000e+00        0.00000
NaN
28  Office Supplies      Machines  0.000000e+00        0.00000
NaN
30  Office Supplies        Phones  0.000000e+00        0.00000
NaN
33  Office Supplies        Tables  0.000000e+00        0.00000
NaN
35       Technology    Appliances  0.000000e+00        0.00000
NaN
36       Technology           Art  0.000000e+00        0.00000
NaN
37       Technology       Binders  0.000000e+00        0.00000
NaN
38       Technology     Bookcases  0.000000e+00        0.00000
NaN
39       Technology        Chairs  0.000000e+00        0.00000
NaN
41       Technology     Envelopes  0.000000e+00        0.00000
NaN
42       Technology     Fasteners  0.000000e+00        0.00000
NaN
43       Technology   Furnishings  0.000000e+00        0.00000
NaN
44       Technology        Labels  0.000000e+00        0.00000
NaN
46       Technology         Paper  0.000000e+00        0.00000
NaN
48       Technology       Storage  0.000000e+00        0.00000
NaN
49       Technology      Supplies  0.000000e+00        0.00000
NaN
50       Technology        Tables  0.000000e+00        0.00000
NaN
```

```
#SEGMENT WISE CONTRIBUTIONS
sales_seg = data.groupby('segment').sum()
sales_seg
```

```
C:\Users\DHANUSHA\AppData\Local\Temp\ipykernel_6140\2279748982.py:2:
FutureWarning: The default value of numeric_only in
DataFrameGroupBy.sum is deprecated. In a future version, numeric_only
will default to False. Either specify numeric_only or select only
columns which should be valid for the function.
  sales_seg = data.groupby('segment').sum()
```

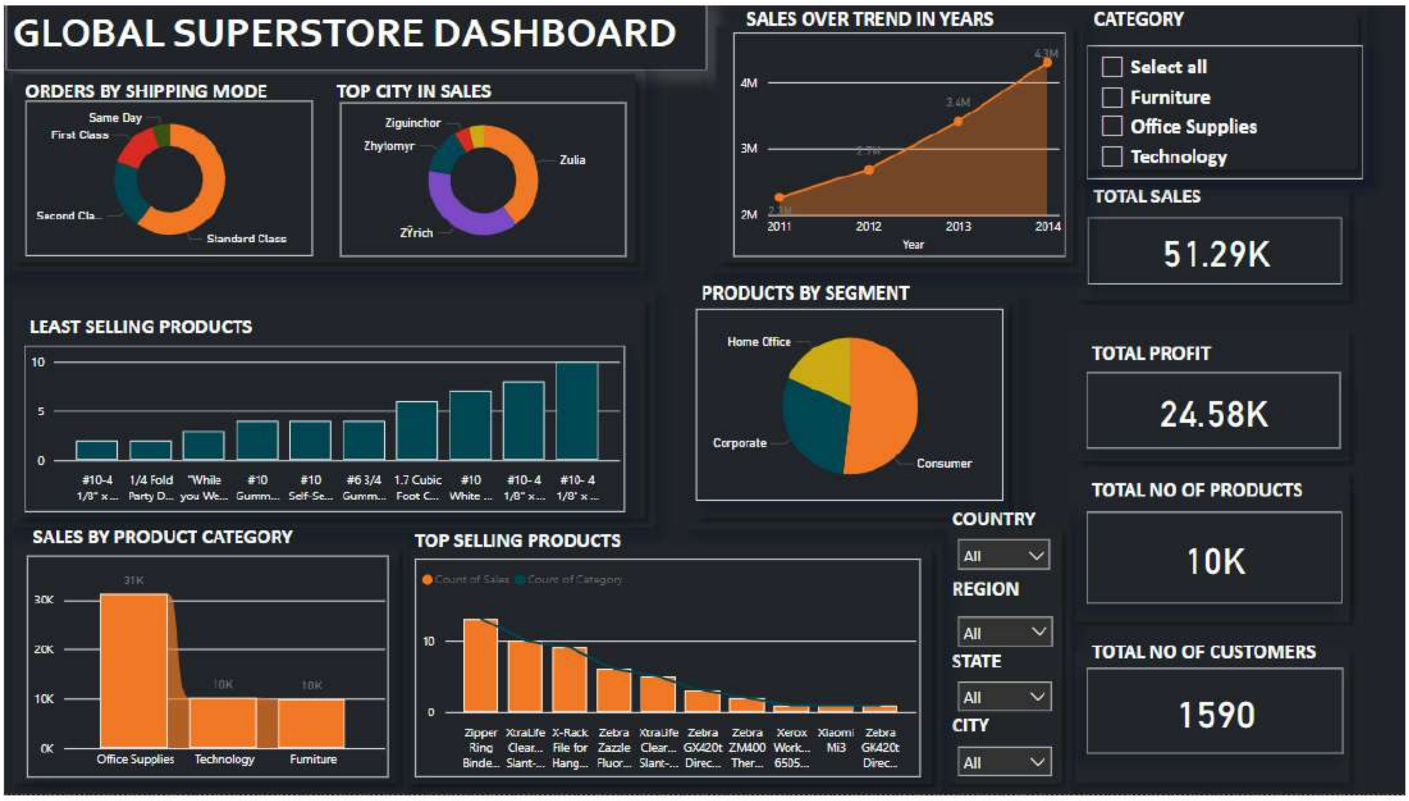
```
                row_id  postal_code         sales  quantity  discount
\
segment

Consumer     679274966  288878609.0  6.507949e+06     92157  3808.042

Corporate    395087151  164536330.0  3.824698e+06     53565  2205.284

Home Office  240995578   98157713.0  2.309855e+06     32590  1316.402


                   profit  shipping_cost  sales_year
segment
Consumer     749239.78206      697300.64    53374470
Corporate    441208.32866      410474.46    31055044
Home Office  277009.18056      245045.59    18805829
```

```
sales_seg
```

```
                row_id  postal_code         sales  quantity  discount
\
segment

Consumer     679274966  288878609.0  6.507949e+06     92157  3808.042

Corporate    395087151  164536330.0  3.824698e+06     53565  2205.284

Home Office  240995578   98157713.0  2.309855e+06     32590  1316.402


                   profit  shipping_cost  sales_year
segment
Consumer     749239.78206      697300.64    53374470
Corporate    441208.32866      410474.46    31055044
Home Office  277009.18056      245045.59    18805829
```

```
#ANALYSING MONTHLY TRENDS
df = pd.DataFrame(data)

df['Order Date'] = pd.to_datetime(df['Order Date'])
```

POWER BI DASHBOARD



6. Reference

- https://docs.google.com/spreadsheets/d/1KagwoQLy1quKvT_82amuS-x3UnsolX4J6p02ewbjQNA/edit?usp=sharing
- https://docs.google.com/document/d/1IuBvcyB81k5Ucz2QcZjlr8xQRJpdpMhMLSVVYGX_k78/edit?usp=sharing
- https://www.geeksforgeeks.org/exploratory-data-analysis-in-python/
- https://www.analyticsvidhya.com/blog/2015/04/comprehensive-guide-data-exploration-sas-using-python-numpy-scipy-matplotlib-pandas/
- https://medium.com/@jscvcds/data-exploration-in-python-with-examples-30a5324472aa
- https://openstax.org/books/introduction-python-programming/pages/15-5-data-visualization