

Samplesuperstore Data analysis

April 26, 2024

Name:Rishikesh Manohar

0.0.1 importing libraries

```
[1]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

0.0.2 Dataset

```
[2]: df=pd.read_csv("SampleSuperstore.csv")
```

0.0.3 1) Data Exploration

0.0.4 Dimensions of data

```
[3]: df.shape
```

```
[3]: (9994, 13)
```

0.0.5 peek at the data

```
[4]: df.head(5)
```

```
[4]:
```

	Ship Mode	Segment	Country	City	State	\
0	Second Class	Consumer	United States	Henderson	Kentucky	
1	Second Class	Consumer	United States	Henderson	Kentucky	
2	Second Class	Corporate	United States	Los Angeles	California	
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	

	Postal Code	Region	Category	Sub-Category	Sales	Quantity	\
0	42420	South	Furniture	Bookcases	261.9600	2	
1	42420	South	Furniture	Chairs	731.9400	3	
2	90036	West	Office Supplies	Labels	14.6200	2	
3	33311	South	Furniture	Tables	957.5775	5	
4	33311	South	Office Supplies	Storage	22.3680	2	

	Discount	Profit
0	0.00	41.9136
1	0.00	219.5820
2	0.00	6.8714
3	0.45	-383.0310
4	0.20	2.5164

```
[5]: df.tail(5)
```

```
[5]:
```

	Ship Mode	Segment	Country	City	State	\
9989	Second Class	Consumer	United States	Miami	Florida	
9990	Standard Class	Consumer	United States	Costa Mesa	California	
9991	Standard Class	Consumer	United States	Costa Mesa	California	
9992	Standard Class	Consumer	United States	Costa Mesa	California	
9993	Second Class	Consumer	United States	Westminster	California	

	Postal Code	Region	Category	Sub-Category	Sales	Quantity	\
9989	33180	South	Furniture	Furnishings	25.248	3	
9990	92627	West	Furniture	Furnishings	91.960	2	
9991	92627	West	Technology	Phones	258.576	2	
9992	92627	West	Office Supplies	Paper	29.600	4	
9993	92683	West	Office Supplies	Appliances	243.160	2	

	Discount	Profit
9989	0.2	4.1028
9990	0.0	15.6332
9991	0.2	19.3932
9992	0.0	13.3200
9993	0.0	72.9480

```
[6]: # general overview of data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 13 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Ship Mode       9994 non-null   object
1   Segment         9994 non-null   object
2   Country         9994 non-null   object
3   City            9994 non-null   object
4   State           9994 non-null   object
5   Postal Code     9994 non-null   int64
6   Region          9994 non-null   object
7   Category        9994 non-null   object
```

```

8   Sub-Category  9994 non-null  object
9   Sales         9994 non-null  float64
10  Quantity      9994 non-null  int64
11  Discount      9994 non-null  float64
12  Profit        9994 non-null  float64
dtypes: float64(3), int64(2), object(8)
memory usage: 1015.1+ KB

```

```
[7]: df.describe()
```

```

[7]:      Postal Code      Sales      Quantity      Discount      Profit
count  9994.000000  9994.000000  9994.000000  9994.000000  9994.000000
mean   55190.379428   229.858001    3.789574    0.156203    28.656896
std    32063.693350   623.245101    2.225110    0.206452   234.260108
min     1040.000000     0.444000    1.000000    0.000000  -6599.978000
25%    23223.000000    17.280000    2.000000    0.000000    1.728750
50%    56430.500000    54.490000    3.000000    0.200000    8.666500
75%    90008.000000   209.940000    5.000000    0.200000   29.364000
max    99301.000000  22638.480000   14.000000    0.800000   8399.976000

```

0.0.6 checking for missing values,

```
[8]: df.isnull().sum()
```

```

[8]: Ship Mode      0
     Segment      0
     Country      0
     City         0
     State        0
     Postal Code   0
     Region       0
     Category     0
     Sub-Category  0
     Sales        0
     Quantity     0
     Discount     0
     Profit       0
dtype: int64

```

0.0.7 Missing values are not present in dataset

```
[9]: df.dtypes
```

```

[9]: Ship Mode      object
     Segment      object
     Country      object
     City         object

```

```
State          object
Postal Code    int64
Region         object
Category       object
Sub-Category   object
Sales          float64
Quantity       int64
Discount       float64
Profit         float64
dtype: object
```

0.0.8 2) Data cleaning

```
[10]: df.duplicated()
```

```
[10]: 0      False
      1      False
      2      False
      3      False
      4      False
      ...
      9989   False
      9990   False
      9991   False
      9992   False
      9993   False
      Length: 9994, dtype: bool
```

0.0.9 3) Descriptive Stastics

```
[11]: #Total sales
      total_sales=df['Sales'].sum()
```

```
[12]: total_sales
```

```
[12]: 2297200.8603000003
```

```
[13]: # Average order value
      average_order_value = df['Sales'].mean()
```

```
[14]: average_order_value
```

```
[14]: 229.85800083049833
```

```
[15]: median_sales = df['Sales'].median()
```

```
[16]: median_sales
```

```
[16]: 54.489999999999995
```

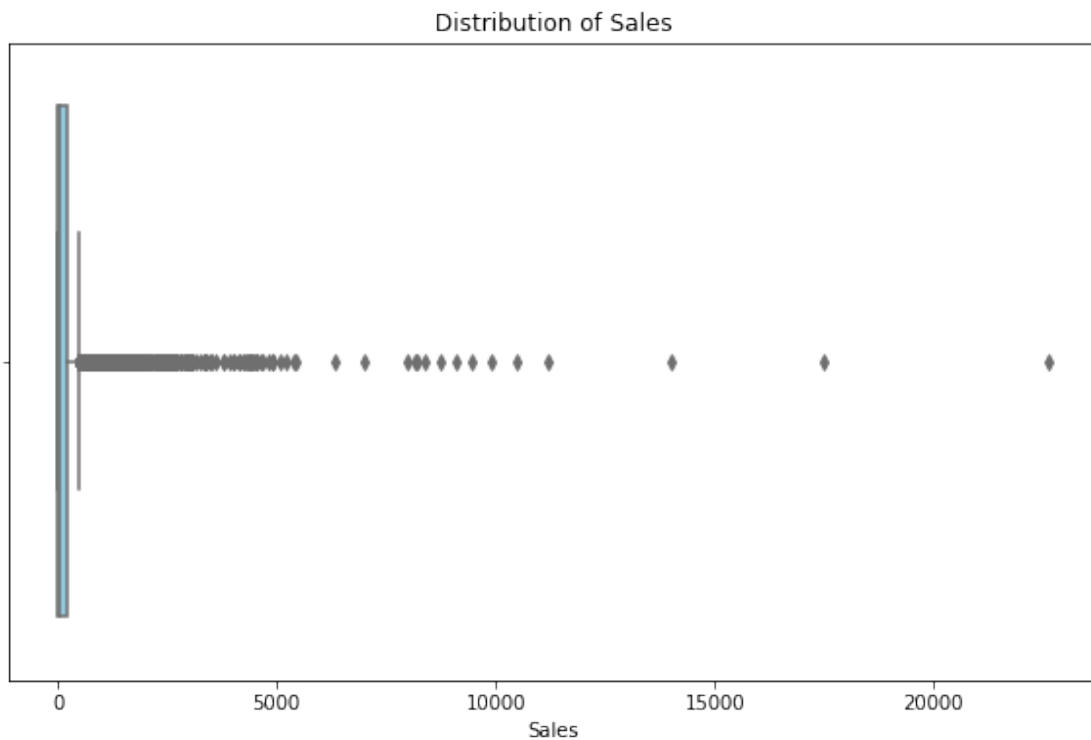
```
[17]: std_dev_sales = df['Sales'].std()
```

```
[18]: std_dev_sales
```

```
[18]: 623.2451005086807
```

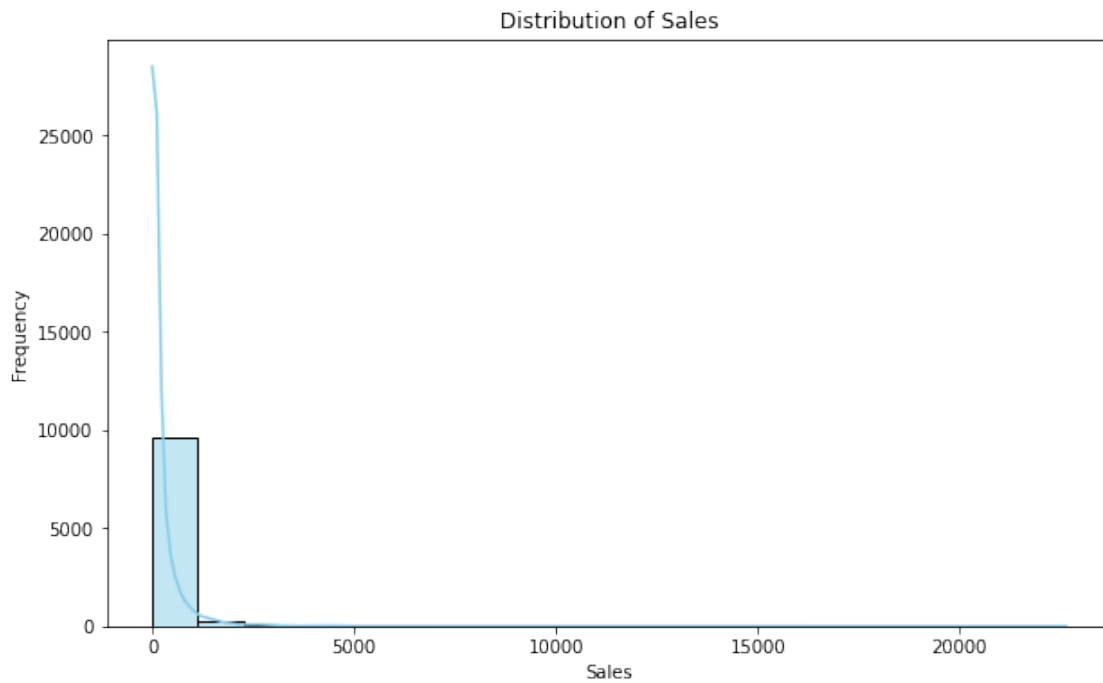
0.0.10 lets visualize distribution of sales

```
[19]: # visualize Distribution of sales by boxplot  
plt.figure(figsize=(10, 6))  
sns.boxplot(x=df['Sales'], color='skyblue')  
plt.title('Distribution of Sales')  
plt.xlabel('Sales')  
plt.show()
```

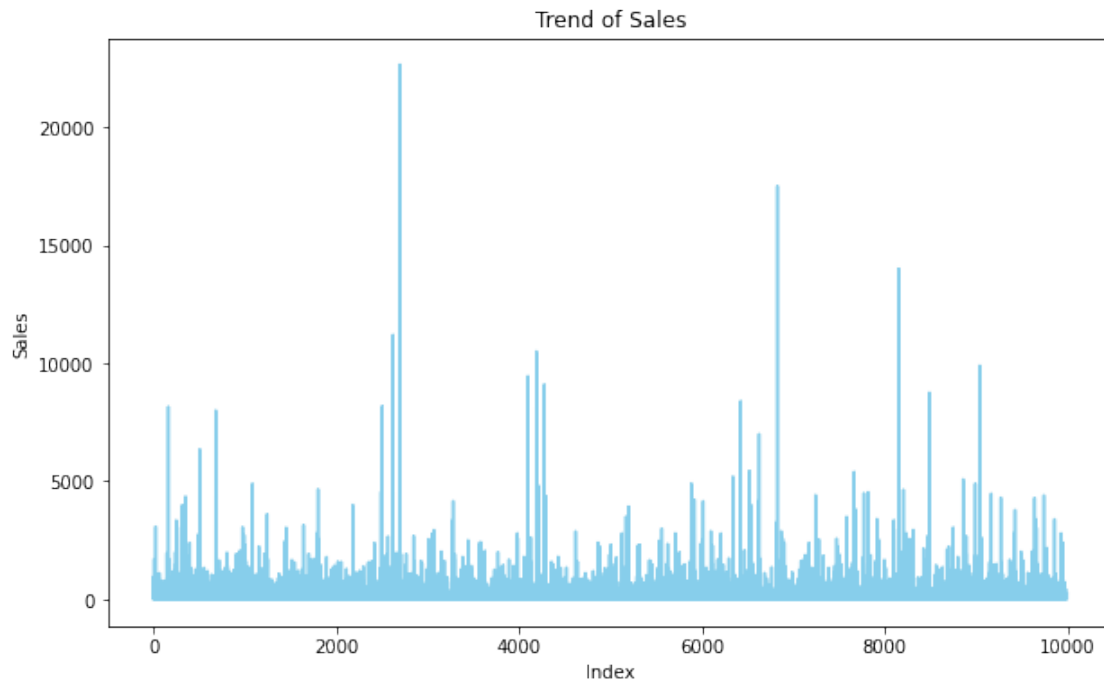


```
[20]: # visualize Distribution of sales by histplot  
plt.figure(figsize=(10, 6))  
sns.histplot(df['Sales'], bins=20, kde=True, color='skyblue')  
plt.title('Distribution of Sales')
```

```
plt.xlabel('Sales')
plt.ylabel('Frequency')
plt.show()
```

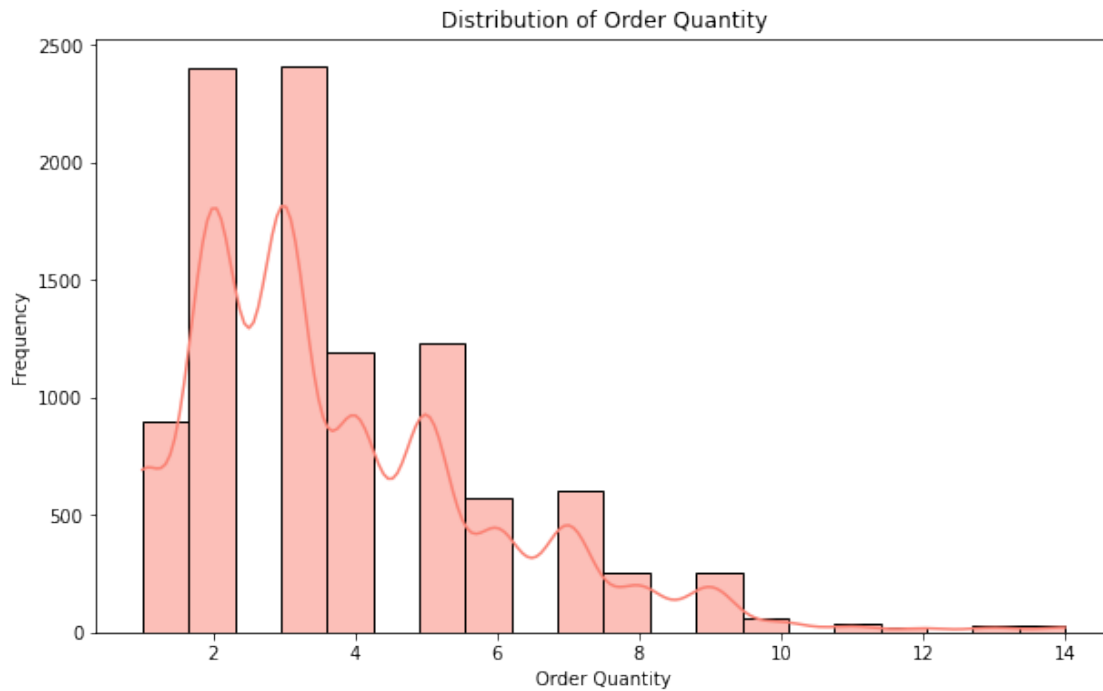


```
[21]: # Trend of Sales
plt.figure(figsize=(10, 6))
sns.lineplot(data=df['Sales'], color='skyblue')
plt.title('Trend of Sales')
plt.xlabel('Index')
plt.ylabel('Sales')
plt.show()
```



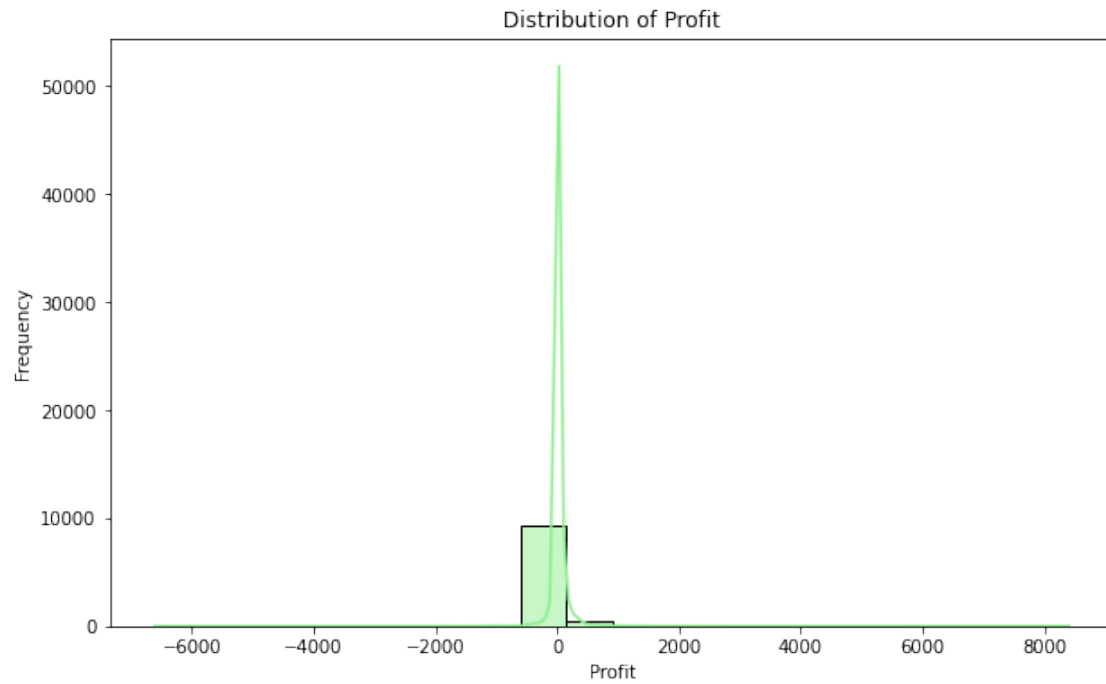
0.0.11 visualize order quantity

```
[22]: # lets visualize the order quantity
plt.figure(figsize=(10, 6))
sns.histplot(df['Quantity'], bins=20, kde=True, color='salmon')
plt.title('Distribution of Order Quantity')
plt.xlabel('Order Quantity')
plt.ylabel('Frequency')
plt.show()
```

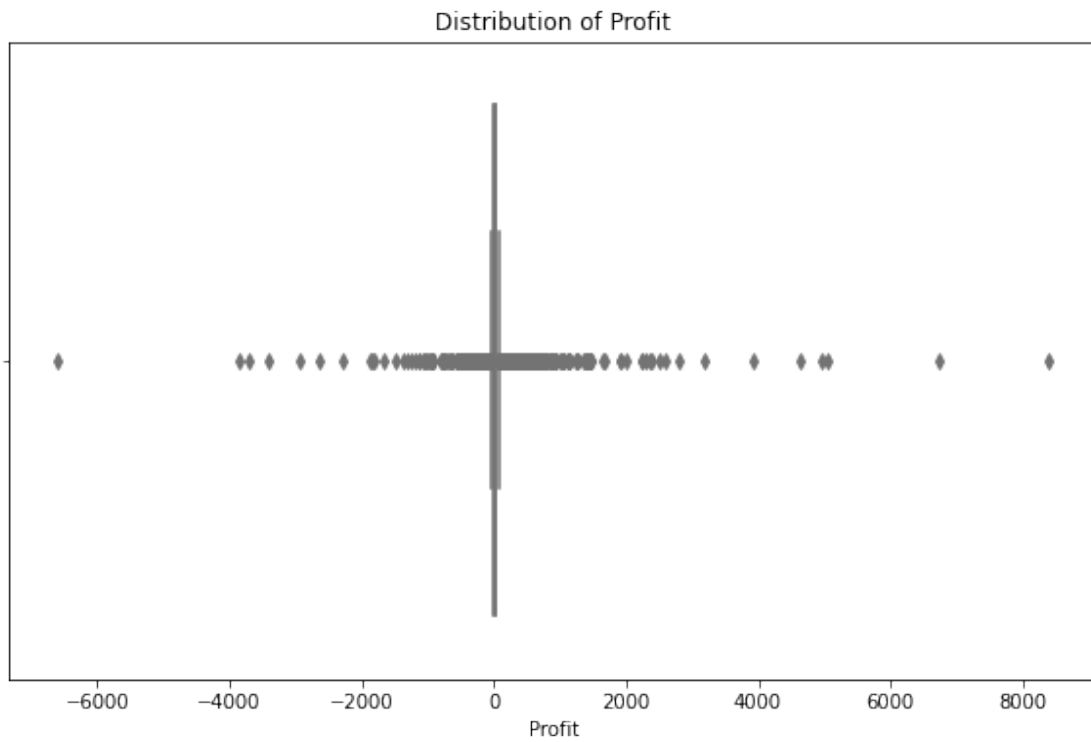


0.0.12 visualize profit

```
[23]: # lets visualize the profit
plt.figure(figsize=(10, 6))
sns.histplot(df['Profit'], kde=True, bins=20, color='lightgreen')
plt.title('Distribution of Profit')
plt.xlabel('Profit')
plt.ylabel('Frequency')
plt.show()
```

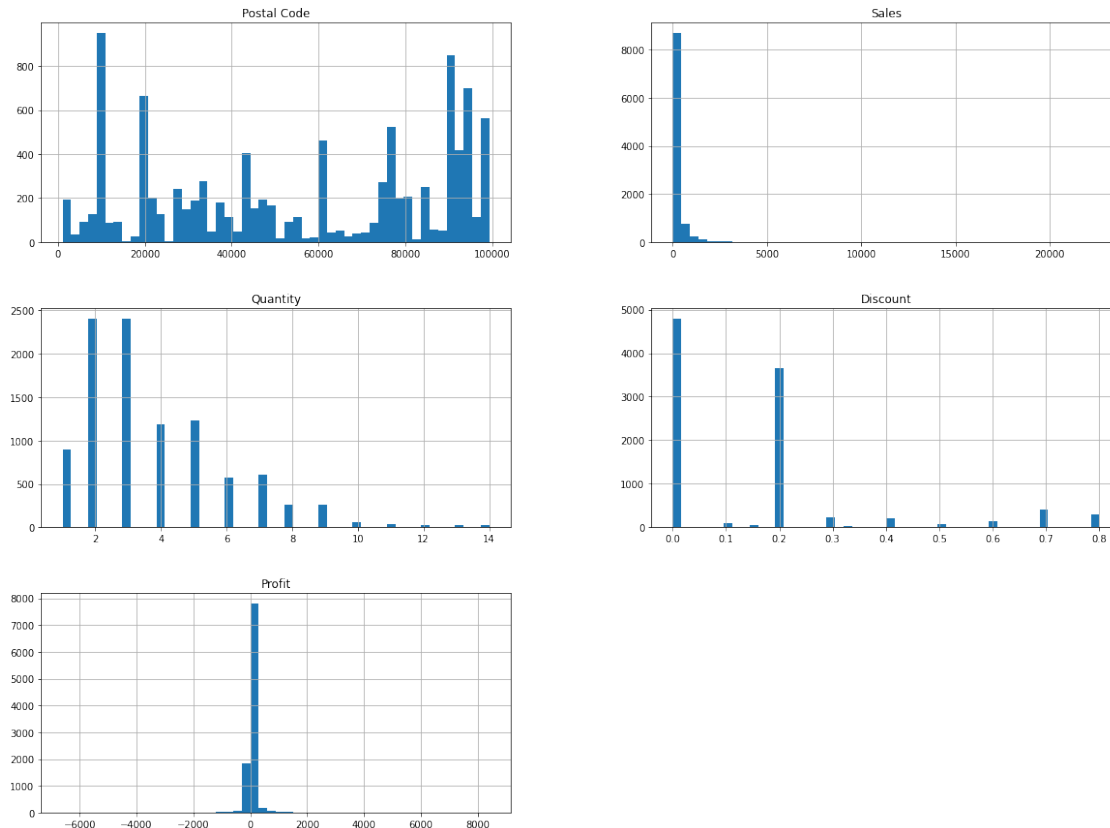



```
[24]: plt.figure(figsize=(10, 6))
sns.boxplot(x=df['Profit'], color='lightgreen')
plt.title('Distribution of Profit')
plt.xlabel('Profit')
plt.show()
```



0.0.13 Histogram of data

```
[25]: df.hist(bins=50,figsize=(20,15))  
plt.show()
```

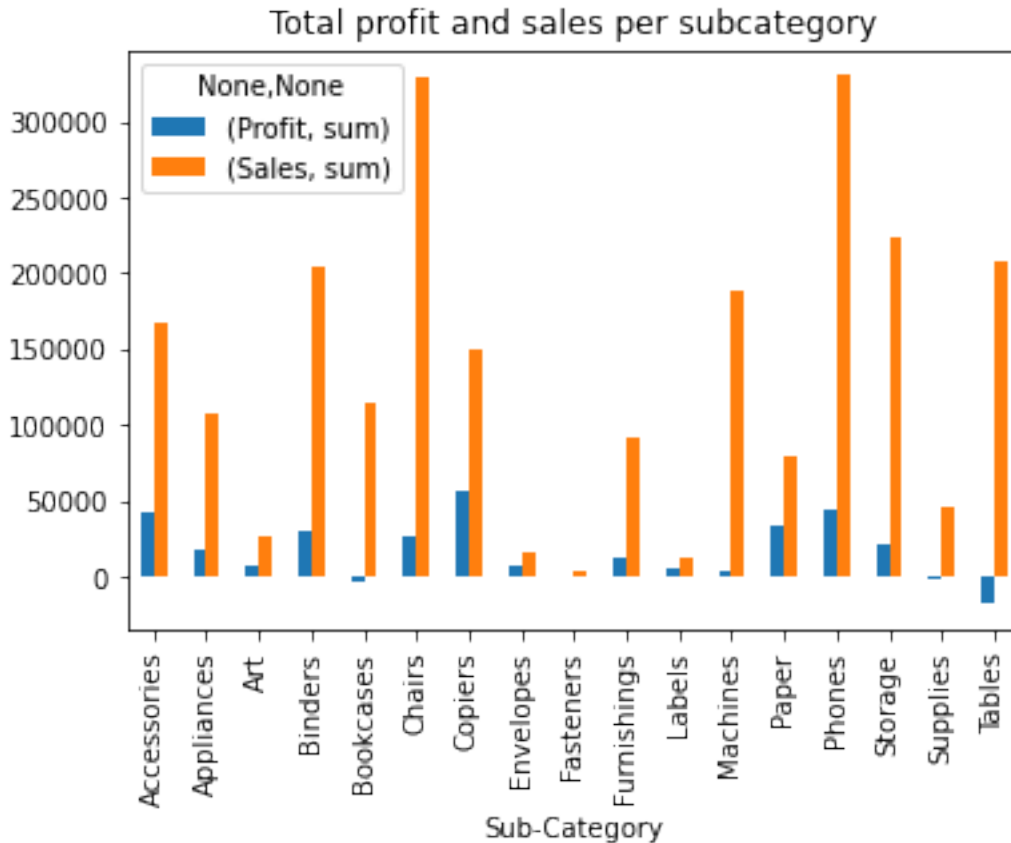


0.0.14 the sales and profit

```
[26]: df.groupby('Sub-Category')['Profit','Sales'].agg(['sum']).plot.bar()
plt.title('Total profit and sales per subcategory')
plt.figure(figsize=[10,8])
plt.show()
```

/tmp/ipykernel_77/3991613971.py:1: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) will be deprecated, use a list instead.

```
df.groupby('Sub-Category')['Profit','Sales'].agg(['sum']).plot.bar()
```



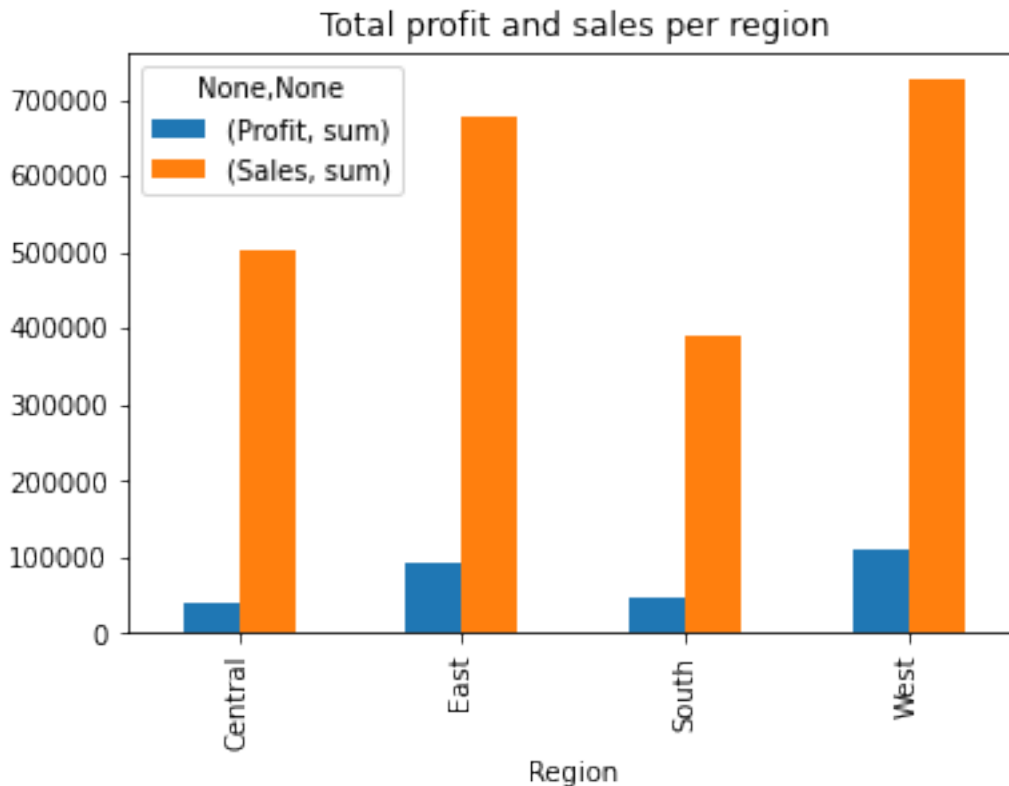
<Figure size 720x576 with 0 Axes>

- 0.1 highest profit is earned in copiers while selling price of chairs and phones are extremely high compared to other product.
- 0.2 people dont prefer to buy tables and bookcases from superstore hence these departments are in loss.

```
[27]: df.groupby('Region')['Profit','Sales'].agg(['sum']).plot.bar()
plt.title('Total profit and sales per region')
plt.figure(figsize=[10,8])
plt.show()
```

/tmp/ipykernel_77/647084091.py:1: FutureWarning: Indexing with multiple keys (implicitly converted to a tuple of keys) will be deprecated, use a list instead.

```
df.groupby('Region')['Profit','Sales'].agg(['sum']).plot.bar()
```



<Figure size 720x576 with 0 Axes>

0.2.1 Sales in west and east region is high whereas sales in south is low.

0.2.2 the profit in central region is very less and profit earned in west is highest.

0.2.3 4) Customer Segmentation

```
[28]: # types of customers
types_of_customers = df["Segment"].unique()
print(types_of_customers)
```

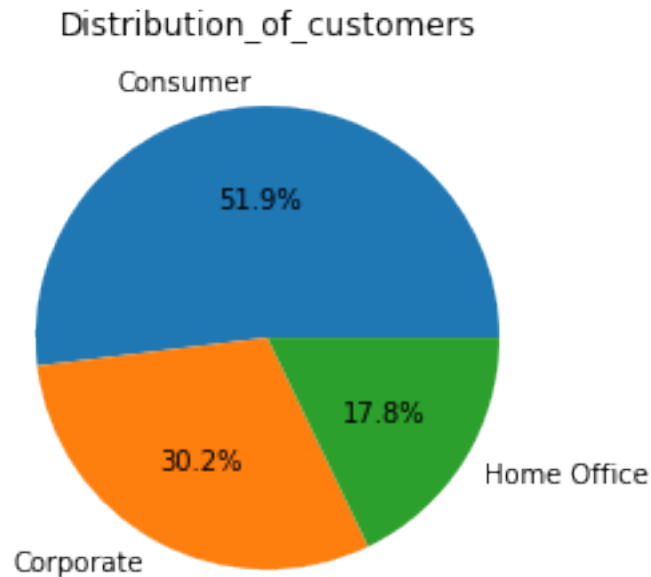
```
['Consumer' 'Corporate' 'Home Office']
```

```
[29]: # number of customers in each segment
number_of_customers= df["Segment"].value_counts().reset_index()
number_of_customers=number_of_customers.rename(columns={'index':'Customer_
↪Type', 'Segment':'Total Customers'})
print(number_of_customers)
```

```
Customer Type  Total Customers
0      Consumer             5191
```

1	Corporate	3020
2	Home Office	1783

```
[30]: #ploting pie chart
plt.pie(number_of_customers['Total_↵
↵Customers'],labels=number_of_customers['Customer Type'],autopct='%1.1f%%')
#set pie chart labels
plt.title('Distribution_of_customers')
plt.show()
```

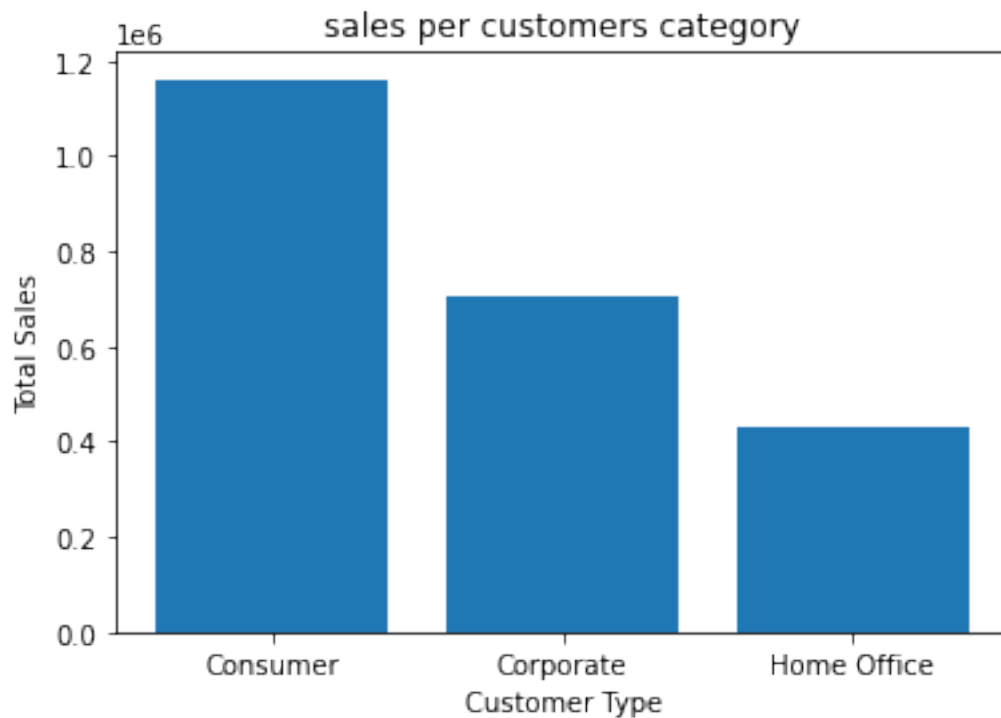


```
[31]: # sales per category
sales_per_category=df.groupby('Segment')['Sales'].sum().reset_index()
sales_per_category=sales_per_category.rename(columns={'Segment': 'Customer_↵
↵Type', 'Sales': 'Total Sales'})
print(sales_per_category)
```

	Customer Type	Total Sales
0	Consumer	1.161401e+06
1	Corporate	7.061464e+05
2	Home Office	4.296531e+05

```
[32]: # sales per customer category
plt.bar(sales_per_category['Customer Type'],sales_per_category['Total Sales'])
#label
plt.title('sales per customers category')
plt.xlabel('Customer Type')
```

```
plt.ylabel('Total Sales')
plt.show()
```



0.2.4 The segment of home office is very less and store has to focus more on home office to improve the sales

```
[33]: # sorting unique values in the ship mode column into new series
types_of_shipping=df['Ship Mode'].unique()
print(types_of_shipping)
```

```
['Second Class' 'Standard Class' 'First Class' 'Same Day']
```

```
[34]: # frequency use of shipping method
shipping_mode= df['Ship Mode'].value_counts().reset_index()
shipping_mode=shipping_mode.rename(columns={'index':'Mode of Shipment','Ship_
↪Mode':'Use Frequency'})
print(shipping_mode)
```

	Mode of Shipment	Use Frequency
0	Standard Class	5968
1	Second Class	1945
2	First Class	1538
3	Same Day	543

```
[35]: state=df['State'].value_counts().reset_index()
state=state.rename(columns={'index':'State','State':'Number of customers'})
print(state.head(7))
```

	State	Number of customers
0	California	2001
1	New York	1128
2	Texas	985
3	Pennsylvania	587
4	Washington	506
5	Illinois	492
6	Ohio	469

0.3 Customer Segmentation by RFM method

```
[36]: frequency_df = df.groupby('City').size().reset_index(name='Frequency')
```

```
[37]: monetary_df = df.groupby('City')['Sales'].sum().reset_index()
monetary_df.rename(columns={'Sales': 'Monetary'}, inplace=True)
```

```
[38]: rfm_df = pd.merge(frequency_df, monetary_df, on='City')
```

```
[39]: print(rfm_df)
```

	City	Frequency	Monetary
0	Aberdeen	1	25.500
1	Abilene	1	1.392
2	Akron	21	2729.986
3	Albuquerque	14	2220.160
4	Alexandria	16	5519.570
..
526	Woonsocket	4	195.550
527	Yonkers	15	7657.666
528	York	5	817.978
529	Yucaipa	1	50.800
530	Yuma	4	840.865

[531 rows x 3 columns]

```
[40]: f_score = rfm_df['Frequency'].quantile(0.75)
m_score = rfm_df['Monetary'].quantile(0.75)
```

```
[41]: def rfm_segment(row):
    if row['Frequency'] >= f_score and row['Monetary'] >= m_score:
        return 'High Value customer'
    elif row['Frequency'] >= f_score:
        return 'Frequent customers'
```



```

elif row['Monetary'] >= m_score:
    return 'Big Spenders'
else:
    return 'Other'

```

```
[42]: rfm_df['RFM_Segment'] = rfm_df.apply(rfm_segment, axis=1)
```

```
[43]: print(rfm_df)
```

	City	Frequency	Monetary	RFM_Segment
0	Aberdeen	1	25.500	Other
1	Abilene	1	1.392	Other
2	Akron	21	2729.986	High Value customer
3	Albuquerque	14	2220.160	Frequent customers
4	Alexandria	16	5519.570	High Value customer
..
526	Woonsocket	4	195.550	Other
527	Yonkers	15	7657.666	High Value customer
528	York	5	817.978	Other
529	Yucaipa	1	50.800	Other
530	Yuma	4	840.865	Other

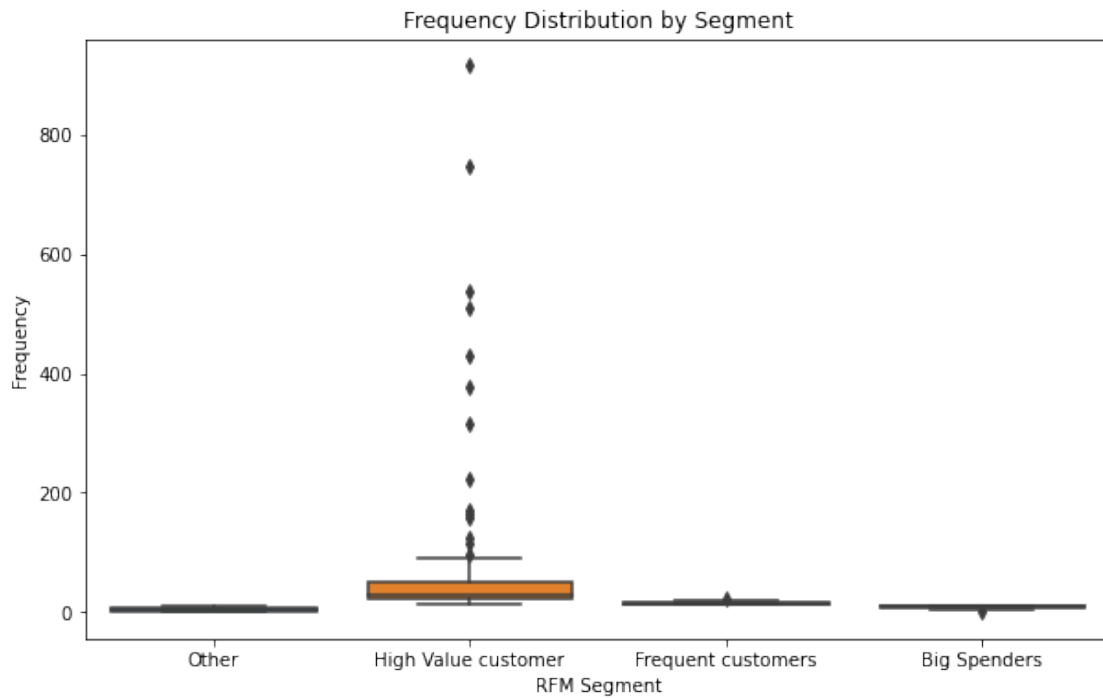
[531 rows x 4 columns]

```
[44]: segment_analysis = rfm_df.groupby('RFM_Segment').agg({
    'Frequency': 'mean',
    'Monetary': 'mean'
}).reset_index()
```

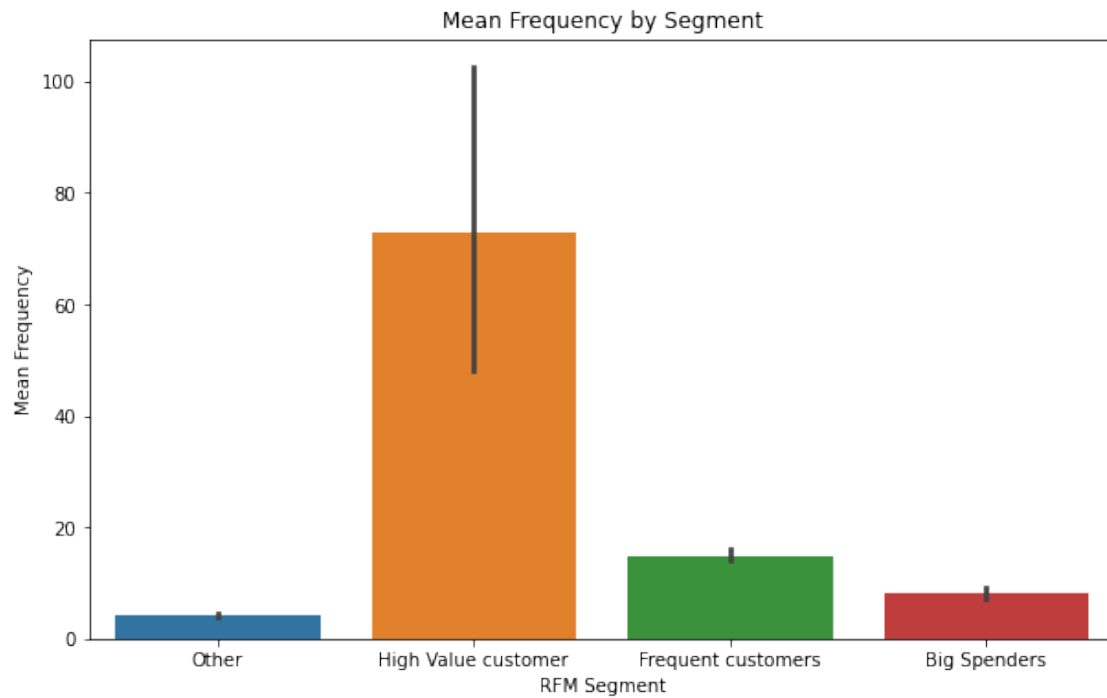
```
[45]: print(segment_analysis)
```

	RFM_Segment	Frequency	Monetary
0	Big Spenders	8.153846	3990.861754
1	Frequent customers	14.928571	1719.205161
2	High Value customer	73.018692	17767.308691
3	Other	4.191892	659.996433

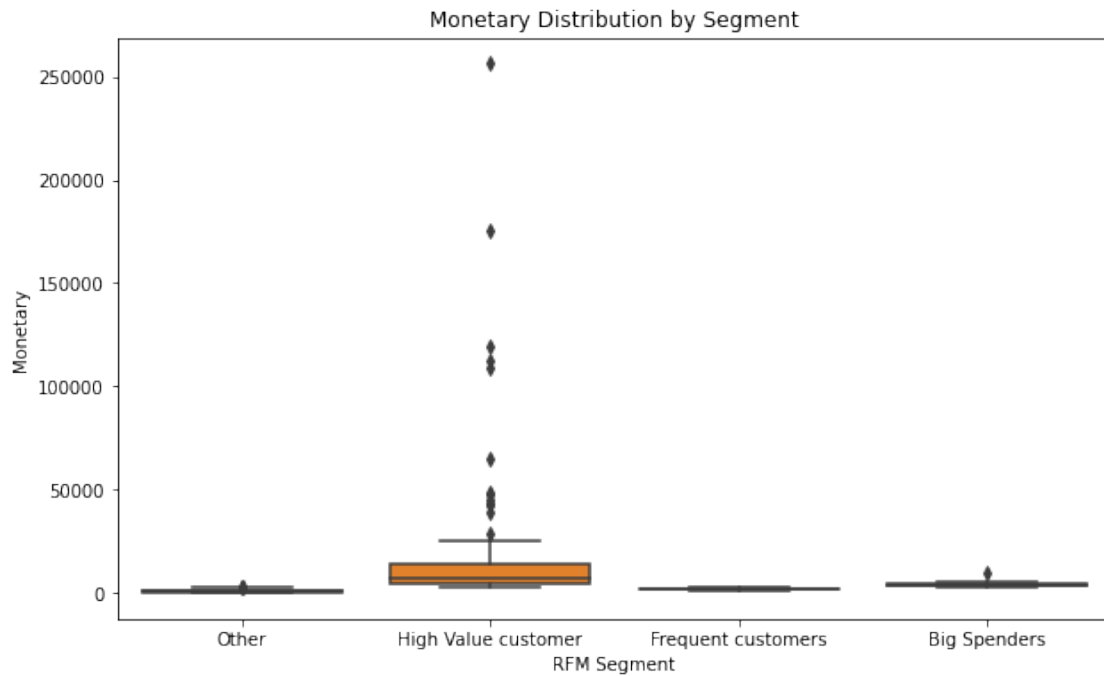
```
[46]: plt.figure(figsize=(10, 6))
sns.boxplot(data=rfm_df, x='RFM_Segment', y='Frequency')
plt.title('Frequency Distribution by Segment')
plt.xlabel('RFM Segment')
plt.ylabel('Frequency')
plt.show()
```



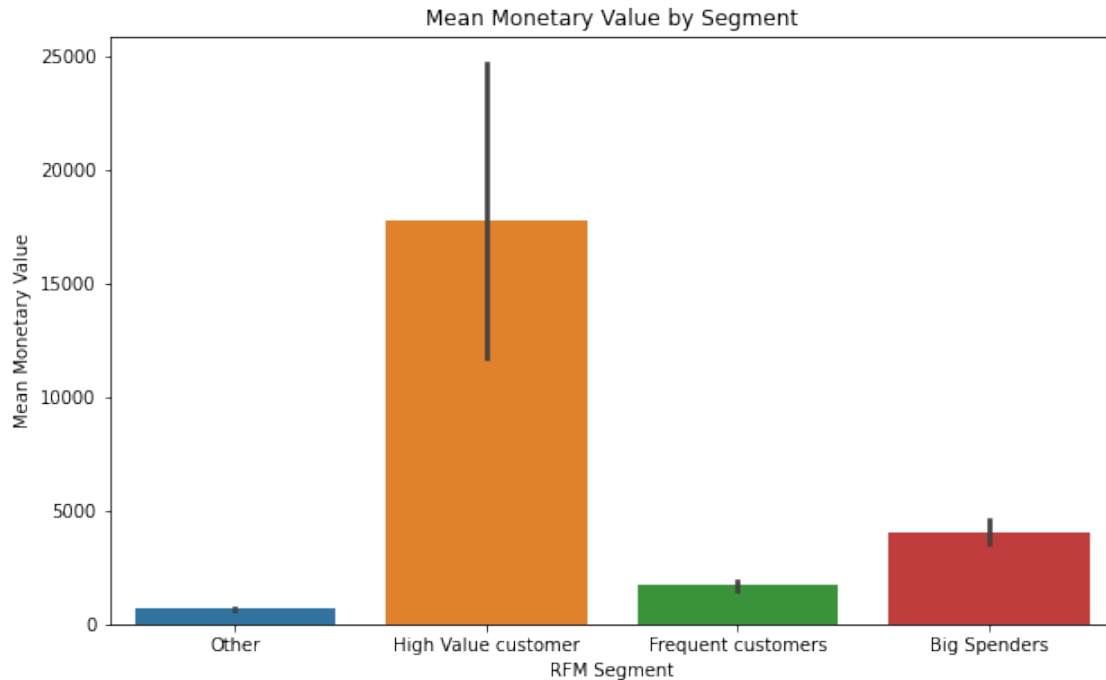
```
[47]: plt.figure(figsize=(10, 6))
sns.barplot(data=rfm_df, x='RFM_Segment', y='Frequency')
plt.title('Mean Frequency by Segment')
plt.xlabel('RFM Segment')
plt.ylabel('Mean Frequency')
plt.show()
```



```
[48]: plt.figure(figsize=(10, 6))
sns.boxplot(data=rfm_df, x='RFM_Segment', y='Monetary')
plt.title('Monetary Distribution by Segment')
plt.xlabel('RFM Segment')
plt.ylabel('Monetary')
plt.show()
```



```
[49]: plt.figure(figsize=(10, 6))
sns.boxplot(data=rfm_df, x='RFM_Segment', y='Monetary')
plt.title('Mean Monetary Value by Segment')
plt.xlabel('RFM Segment')
plt.ylabel('Mean Monetary Value')
plt.show()
```



0.3.1 5)Product Analysis

```
[50]: subcategory_count=df.groupby('Category')['Sub-Category'].nunique().reset_index()
# sort by ascending order
subcategory_count=subcategory_count.
↳sort_values(by='Sub-Category',ascending=False)
print(subcategory_count.reset_index(drop=True))
```

	Category	Sub-Category
0	Office Supplies	9
1	Furniture	4
2	Technology	4

```
[51]: # sales per each category
category_sales=df.groupby(['Category'])['Sales'].sum().reset_index()
print(category_sales)
```

	Category	Sales
0	Furniture	741999.7953
1	Office Supplies	719047.0320
2	Technology	836154.0330

```
[52]: top_selling_categories = df.groupby('Category').agg({'Sales': 'sum'}).
↳reset_index()
```

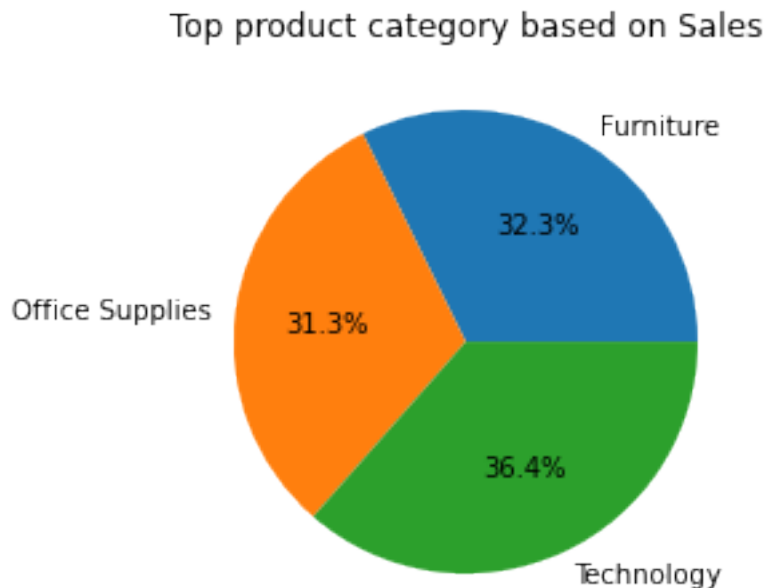
```
[53]: top_selling_categories = top_selling_categories.sort_values(by='Sales',
↳ascending=False)
```

```
[54]: print("\nTop Selling Categories:")
print(top_selling_categories.head(5))
```

Top Selling Categories:

	Category	Sales
2	Technology	836154.0330
0	Furniture	741999.7953
1	Office Supplies	719047.0320

```
[55]: # plotting a pie chart
plt.pie(category_sales['Sales'],labels=category_sales['Category'],autopct='%1.
↳1f%%')
#set labels
plt.title('Top product category based on Sales')
plt.show()
```



```
[56]: top_selling_products = df.groupby('Sub-Category').agg({'Sales': 'sum'}).
↳reset_index()
```

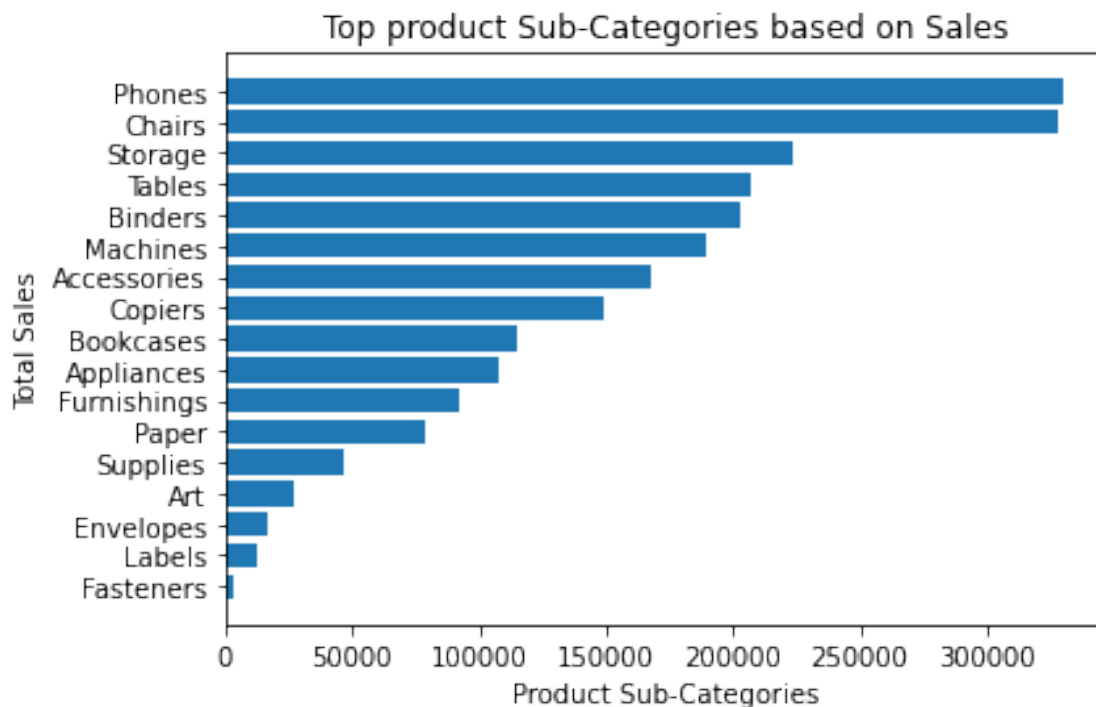
```
[57]: top_selling_products = top_selling_products.sort_values(by='Sales',
↳ascending=False)
```

```
[58]: print("Top Selling Products:")
      print(top_selling_products.head(10))
```

Top Selling Products:

	Sub-Category	Sales
13	Phones	330007.0540
5	Chairs	328449.1030
14	Storage	223843.6080
16	Tables	206965.5320
3	Binders	203412.7330
11	Machines	189238.6310
0	Accessories	167380.3180
6	Copiers	149528.0300
4	Bookcases	114879.9963
1	Appliances	107532.1610

```
[59]: # plotting horizontal bar graph
top_selling_products=top_selling_products.sort_values(by='Sales',ascending=True)
plt.barh(top_selling_products['Sub-Category'],top_selling_products['Sales'])
# labels
plt.title('Top product Sub-Categories based on Sales')
plt.xlabel('Product Sub-Categories')
plt.ylabel('Total Sales')
plt.show()
```



0.3.2 6)Time series Analysis

0.3.3 To examin the Sales trend over different time periods the dataset does not content the data like Date,Time,Month,year.

0.3.4 to examin sales trend over different time period the data of order date or year,month is required.

```
[60]: region_sales = df.groupby('Region')['Sales'].sum().reset_index()
```

```
[61]: region_sales = region_sales.sort_values(by='Sales', ascending=False)
```

```
[62]: print("\nSales Distribution by Region:")
      print(region_sales)
```

Sales Distribution by Region:

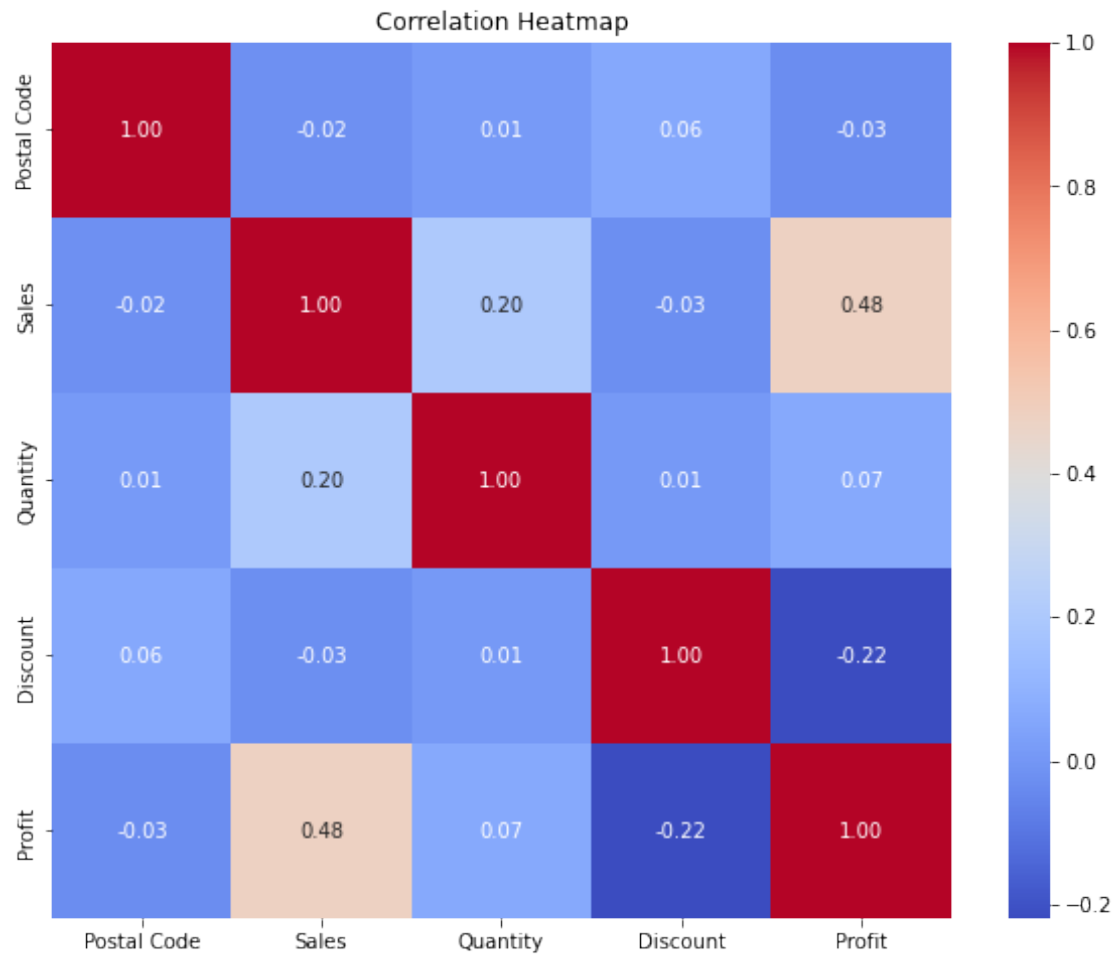
	Region	Sales
3	West	725457.8245
1	East	678781.2400
0	Central	501239.8908
2	South	391721.9050

```
[63]: correlation_matrix = df.corr()
```

/tmp/ipykernel_77/4214245630.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

```
correlation_matrix = df.corr()
```

```
[64]: plt.figure(figsize=(10, 8))
      sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
      plt.title('Correlation Heatmap')
      plt.show()
```

- 0.3.5 Correlation between profit and discount is in negative and correlation between profit and sales is postive
- 0.3.6 7) Conclusions
- 0.3.7 profit in south and central region is less
- 0.3.8 Profit in east and west region is better than south and central
- 0.3.9 highest profit is earned in copiers while selling price of chairs and phones are extremely high compared to other product.
- 0.3.10 people dont prefer to buy tables and bookcases from superstore hence these departments are in loss
- 0.3.11 The store has wide variety of office supplies especialy in binders and paper.
- 0.3.12 Negative correlation between and profit and discount
- 0.3.13 Total sum of profit in sale of table is negative.
- 0.3.14 profit is more in sale of copiers.
- 0.3.15 No or very less profit in supplies.
- 0.3.16 Technology segment is more profitable.
- 0.3.17 Analyzed customer segments based on purchasing behavior
- 0.3.18 Identified high-value customers, frequent customers, and other segments to marketing strategies and customer experiences
- 0.3.19 Trend Analysis (without Time Data):
- 0.3.20 Conducted trend analysis based on aggregated data or other available features, despite the absence of explicit date-related information.
- 0.4 Recommendation for improving sales :
- 0.4.1 the correlation between profit and discount is negative Discounts can attract customers and lead to higher revenue by encouraging more purchases.
- 0.4.2 Offering discounts lowers the overall revenue per sale, which can decrease profit margins.
- 0.4.3 If not strategically planned, discounts may impact sales effectiveness
- 0.4.4 Product Performance: The sales of phones and chairs are highest it is frequently buyed products store has to manage there stocks to avoid shortage of quantity.
- 0.4.5 The sales of envelopes ,fasteners and art products are very less store has focus on this to increase sales by giving attractive discounts or schemes to the customers
- 0.4.6 The sales in west and east region is very high Keep track of inventory levels to avoid stockouts or overstock situations. Use inventory management tools to ensure you have the right products available at the right time
- 0.4.7 Repeat customers tend to spend more money with a business over time. Their loyalty translates into higher revenue for the company.
- 0.4.8 To increase sales into south and central region store has to conduct the Customer Loyalty Programs: These programs encourage frequent customer continue doing business by offering exclusive rewards such as discounts, seasonal coupons,

[]: