Sales_Analysis

January 26, 2025

1 Sales Data Analysis and Visualizations

1.1 The management team has outlined seven critical questions they would like to address through analysis of this dataset.

1.1.1 Here are the Research Questions:

1. You need to calculate the monthly sales the store and identify which month had the highest sales and which month had the lowest sales.
2. You need to analyze sales based on product categories and determine which category has the lowest sales and which category has the highest sales.
3. The sales analysis needs to be done based on sub-categories.
4. You need to analyze the monthly profit from sales and determine which month had the highest profit.
5. Analyze the profit by category and sub-category.
6. Analyze the sales and profit by customer segment.
7. Analyze the sales to profit ratio.

1.Pandas for Data Cleaning 2. Plotly.Express For For Data Visualizations (Advance, Fast and Provide more functionalities) 3. Plotly.grapah_objects For Advance and customize Graphs 4. Plotly.io For Graphs Templates 5. Plotly.colors For Colors to use in Graphs 6. Pio.templates For By Default theme color

[178]:
```
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
import plotly.colors as colors
pio.templates.default = 'plotly_white'
```

[180]:
```
df = pd.read_csv("Superstore_dataset.csv", encoding = 'latin-1') #Encoding␣
↪means that as we have underscoore and slash like / in the dataset so our␣
↪machine can understand this as well properly
```

[601]:
```
df.columns
```

[601]:
```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State',
       'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category',
       'Product Name', 'Sales', 'Quantity', 'Discount', 'Profit', 'Order Year',
       'Order Month', 'Day of the Week'],
      dtype='object')
```

[182]:
```
df.head()
```

[182]:
```
   Row ID        Order ID  Order Date   Ship Date       Ship Mode Customer ID  \
0       1  CA-2016-152156   11/8/2016  11/11/2016    Second Class    CG-12520
1       2  CA-2016-152156   11/8/2016  11/11/2016    Second Class    CG-12520
2       3  CA-2016-138688   6/12/2016   6/16/2016    Second Class    DV-13045
3       4  US-2015-108966  10/11/2015  10/18/2015  Standard Class    SO-20335
4       5  US-2015-108966  10/11/2015  10/18/2015  Standard Class    SO-20335

     Customer Name    Segment        Country             City  …  \
0      Claire Gute   Consumer  United States        Henderson  …
1      Claire Gute   Consumer  United States        Henderson  …
2  Darrin Van Huff  Corporate  United States      Los Angeles  …
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …

  Postal Code  Region       Product ID         Category Sub-Category  \
0       42420   South  FUR-BO-10001798        Furniture    Bookcases
1       42420   South  FUR-CH-10000454        Furniture       Chairs
2       90036    West  OFF-LA-10000240  Office Supplies       Labels
3       33311   South  FUR-TA-10000577        Furniture       Tables
4       33311   South  OFF-ST-10000760  Office Supplies      Storage

                                        Product Name     Sales  Quantity  \
0                  Bush Somerset Collection Bookcase  261.9600         2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,…  731.9400         3
2  Self-Adhesive Address Labels for Typewriters b…   14.6200         2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775         5
4                     Eldon Fold 'N Roll Cart System   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 rows x 21 columns]
```

[184]:
```
df.describe()
```

[184]:
```
            Row ID   Postal Code         Sales     Quantity     Discount  \
count  9994.000000   9994.000000   9994.000000  9994.000000  9994.000000
mean   4997.500000  55190.379428    229.858001     3.789574     0.156203
std    2885.163629  32063.693350    623.245101     2.225110     0.206452
min       1.000000   1040.000000      0.444000     1.000000     0.000000
25%    2499.250000  23223.000000     17.280000     2.000000     0.000000
50%    4997.500000  56430.500000     54.490000     3.000000     0.200000
75%    7495.750000  90008.000000    209.940000     5.000000     0.200000
max    9994.000000  99301.000000  22638.480000    14.000000     0.800000

            Profit
count  9994.000000
mean     28.656896
std     234.260108
min   -6599.978000
25%       1.728750
50%       8.666500
75%      29.364000
max    8399.976000
```

[186]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   object
 3   Ship Date      9994 non-null   object
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
```

```
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

1.1.2 Converting Date columns

[189]:
```
df['Order Date'] = pd.to_datetime(df['Order Date'])
```

[191]:
```
df['Ship Date'] = pd.to_datetime(df['Ship Date'])
```

[193]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: datetime64[ns](2), float64(3), int64(3), object(13)
memory usage: 1.6+ MB
```

[195]:
```
df.head()
```

[195]:
```
   Row ID        Order ID Order Date  Ship Date       Ship Mode Customer ID  \
0       1  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
1       2  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
2       3  CA-2016-138688 2016-06-12 2016-06-16    Second Class    DV-13045
3       4  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335
4       5  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335
```

```
     Customer Name    Segment        Country            City  …  \
0      Claire Gute   Consumer  United States       Henderson  …
1      Claire Gute   Consumer  United States       Henderson  …
2  Darrin Van Huff  Corporate  United States     Los Angeles  …
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …

  Postal Code  Region       Product ID         Category Sub-Category  \
0       42420   South  FUR-BO-10001798        Furniture    Bookcases
1       42420   South  FUR-CH-10000454        Furniture       Chairs
2       90036    West  OFF-LA-10000240  Office Supplies       Labels
3       33311   South  FUR-TA-10000577        Furniture       Tables
4       33311   South  OFF-ST-10000760  Office Supplies      Storage

                                       Product Name     Sales  Quantity  \
0                 Bush Somerset Collection Bookcase  261.9600         2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,…  731.9400         3
2  Self-Adhesive Address Labels for Typewriters b…   14.6200         2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775         5
4                     Eldon Fold 'N Roll Cart System   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 rows x 21 columns]
```

1.1.3 We want to Analayze the data in monthly and Yearly wise so let's extract the month and year from the order date column

[544]:
```
df['Order Year'] =  df['Order Date'].dt.year
df['Order Month'] = df['Order Date'].dt.month
df['Day of the Week'] = df['Order Date'].dt.dayofweek
```

[546]:
```
df.head()
```

[546]:
```
   Row ID        Order ID Order Date  Ship Date       Ship Mode Customer ID  \
0       1  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
1       2  CA-2016-152156 2016-11-08 2016-11-11    Second Class    CG-12520
2       3  CA-2016-138688 2016-06-12 2016-06-16    Second Class    DV-13045
3       4  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335
4       5  US-2015-108966 2015-10-11 2015-10-18  Standard Class    SO-20335
```

```
     Customer Name    Segment        Country             City  …  \
0      Claire Gute   Consumer  United States        Henderson  …
1      Claire Gute   Consumer  United States        Henderson  …
2  Darrin Van Huff  Corporate  United States      Los Angeles  …
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …

          Category  Sub-Category  \
0        Furniture     Bookcases
1        Furniture        Chairs
2  Office Supplies        Labels
3        Furniture        Tables
4  Office Supplies       Storage

                                        Product Name     Sales Quantity  \
0                  Bush Somerset Collection Bookcase  261.9600        2
1  Hon Deluxe Fabric Upholstered Stacking Chairs,…  731.9400        3
2  Self-Adhesive Address Labels for Typewriters b…   14.6200        2
3      Bretford CR4500 Series Slim Rectangular Table  957.5775        5
4                         Eldon Fold 'N Roll Cart System   22.3680        2

  Discount    Profit  Order Year  Order Month  Day of the Week
0     0.00   41.9136        2016           11                1
1     0.00  219.5820        2016           11                1
2     0.00    6.8714        2016            6                6
3     0.45 -383.0310        2015           10                6
4     0.20    2.5164        2015           10                6

[5 rows x 24 columns]
```

[548]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 24 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Row ID           9994 non-null   int64
 1   Order ID         9994 non-null   object
 2   Order Date       9994 non-null   datetime64[ns]
 3   Ship Date        9994 non-null   datetime64[ns]
 4   Ship Mode        9994 non-null   object
 5   Customer ID      9994 non-null   object
 6   Customer Name    9994 non-null   object
 7   Segment          9994 non-null   object
 8   Country          9994 non-null   object
 9   City             9994 non-null   object
 10  State            9994 non-null   object
 11  Postal Code      9994 non-null   int64
 12  Region           9994 non-null   object
 13  Product ID       9994 non-null   object
 14  Category         9994 non-null   object
 15  Sub-Category     9994 non-null   object
 16  Product Name     9994 non-null   object
 17  Sales            9994 non-null   float64
 18  Quantity         9994 non-null   int64
 19  Discount         9994 non-null   float64
 20  Profit           9994 non-null   float64
 21  Order Year       9994 non-null   int32
 22  Order Month      9994 non-null   int32
 23  Day of the Week  9994 non-null   int32
dtypes: datetime64[ns](2), float64(3), int32(3), int64(3), object(13)
memory usage: 1.7+ MB
```

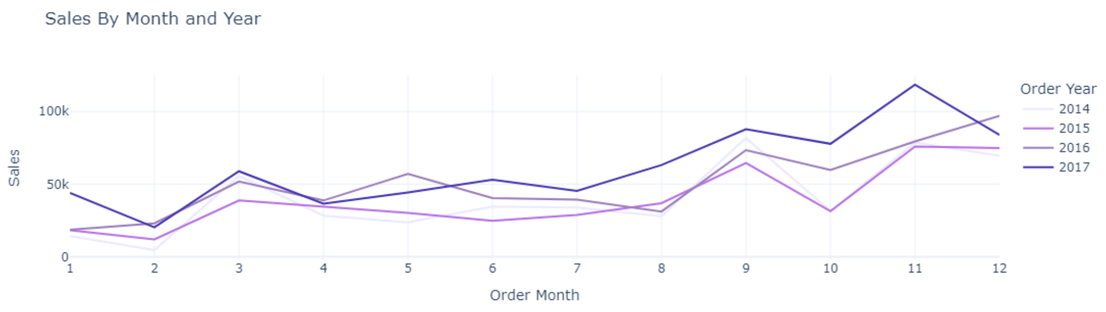
1.1.4 1. You need to calculate the monthly sales of the store and identify which month had the highest sales and which month had the lowest sales.

[591]:
```
monthly_sales = df.groupby(['Order Year', 'Order Month'],␣
 ↪observed=False)['Sales'].sum().reset_index()
monthly_sales
```

[591]:
```
    Order Year  Order Month        Sales
0         2014            1   14236.8950
1         2014            2    4519.8920
2         2014            3   55691.0090
3         2014            4   28295.3450
4         2014            5   23648.2870
5         2014            6   34595.1276
6         2014            7   33946.3930
7         2014            8   27909.4685
8         2014            9   81777.3508
9         2014           10   31453.3930
10        2014           11   78628.7167
11        2014           12   69545.6205
12        2015            1   18174.0756
13        2015            2   11951.4110
14        2015            3   38726.2520
15        2015            4   34195.2085
16        2015            5   30131.6865
17        2015            6   24797.2920
18        2015            7   28765.3250
19        2015            8   36898.3322
20        2015            9   64595.9180
21        2015           10   31404.9235
22        2015           11   75972.5635
```

```
23          2015            12    74919.5212
24          2016             1    18542.4910
25          2016             2    22978.8150
26          2016             3    51715.8750
27          2016             4    38750.0390
28          2016             5    56987.7280
29          2016             6    40344.5340
30          2016             7    39261.9630
31          2016             8    31115.3743
32          2016             9    73410.0249
33          2016            10    59687.7450
34          2016            11    79411.9658
35          2016            12    96999.0430
36          2017             1    43971.3740
37          2017             2    20301.1334
38          2017             3    58872.3528
39          2017             4    36521.5361
40          2017             5    44261.1102
41          2017             6    52981.7257
42          2017             7    45264.4160
43          2017             8    63120.8880
44          2017             9    87866.6520
45          2017            10    77776.9232
46          2017            11   118447.8250
47          2017            12    83829.3188
```

[597]:
```python
purple_shades = ['#EBEAFF', '#B771E5','#9B7EBD', '#4635B1']
fig = px.line(monthly_sales, x = 'Order Month', y = 'Sales',
title = 'Sales By Month and Year', color='Order Year',
color_discrete_sequence=purple_shades)
fig.update_xaxes(type='category')
fig.show()
```



1.1.5 Insight:

1. Highest Sales Month: December consistently shows the highest sales across the years, peaking especially in 2017.
2. Lowest Sales Month: January has the lowest sales in most years.

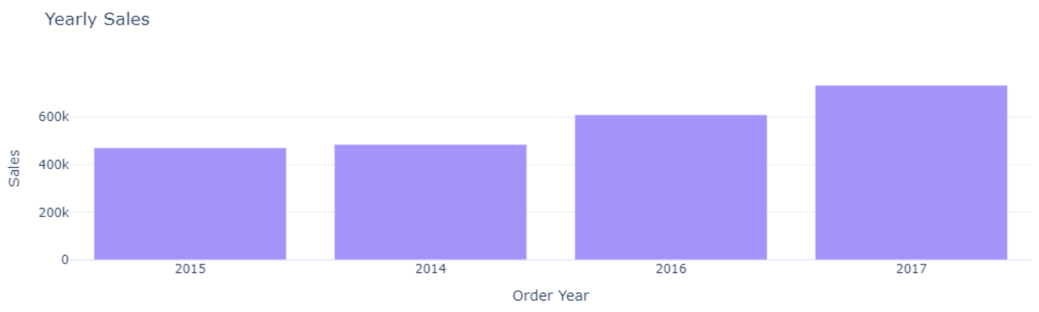
1.1.6 Recommendation:

1. Leverage December as the primary sales month by offering holiday promotions and discounts to maximize revenue.
2. Introduce targeted marketing campaigns or discounts during January to boost sales in the slowest month.

```
[587]: df['Order Year'] = df['Order Year'].astype('category')
       Yearly_sales = df.groupby(['Order Year'],observed=False)['Sales'].sum().
        ↪reset_index().round(2)
       Yearly_sales
```

```
[587]:   Order Year      Sales
       0       2014  484247.50
       1       2015  470532.51
       2       2016  609205.60
       3       2017  733215.26
```

```
[563]: purple_shades = ['#A294F9']
       fig = px.bar(Yearly_sales,
                    x= 'Order Year',
                    y='Sales',
                    title = 'Yearly Sales',
                    color_discrete_sequence = purple_shades
       )
       fig.update_layout(xaxis={'categoryorder': 'total ascending'})
       fig.update_xaxes(type='category')
       fig.show()
```



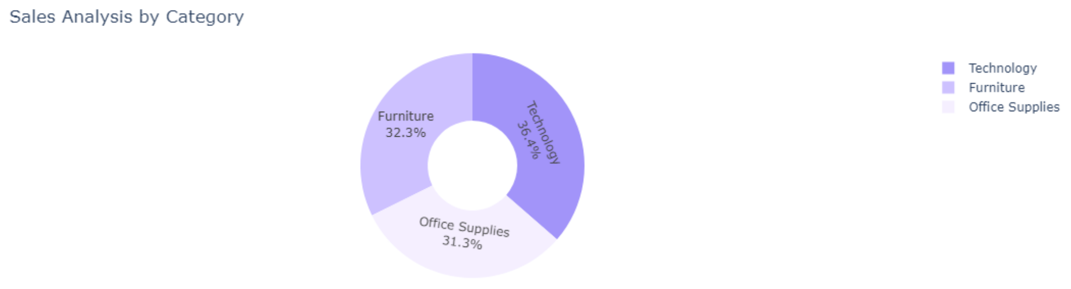
1.1.7 2. You need to analyze sales based on product categories and determine which category has the lowest sales and which category has the highest sales.

[212]:
```
sales_by_category = df.groupby('Category')['Sales'].sum().reset_index()
sales_by_category
```

[212]:
```
          Category        Sales
0        Furniture  741999.7953
1  Office Supplies  719047.0320
2       Technology  836154.0330
```

[413]:
```
purple_shades = ['#A294F9', '#CDC1FF', '#F5EFFF']

fig = px.pie(
    sales_by_category,
    names='Category',
    values='Sales',
    hole=0.4,
    color_discrete_sequence=purple_shades
)
fig.update_traces(textposition='inside', textinfo='percent+label')
fig.update_layout(title_text='Sales Analysis by Category')
fig.show()
```



1.1.8 Insight:

1. Highest Sales Category: Technology contributes the highest sales at 36.4%.
2. Lowest Sales Category: Office Supplies has the lowest sales at 31.3%.

1.1.9 Recommendation:

1. Focus on maintaining inventory and providing attractive promotions for Phones and Chairs to sustain their strong performance.
2. Investigate the low sales in Fasteners and Labels to determine if they are essential to the product mix; if not, consider reducing inventory or discontinuing these items.

1.1.10 3. The sales analysis needs to be done based on sub-categories.

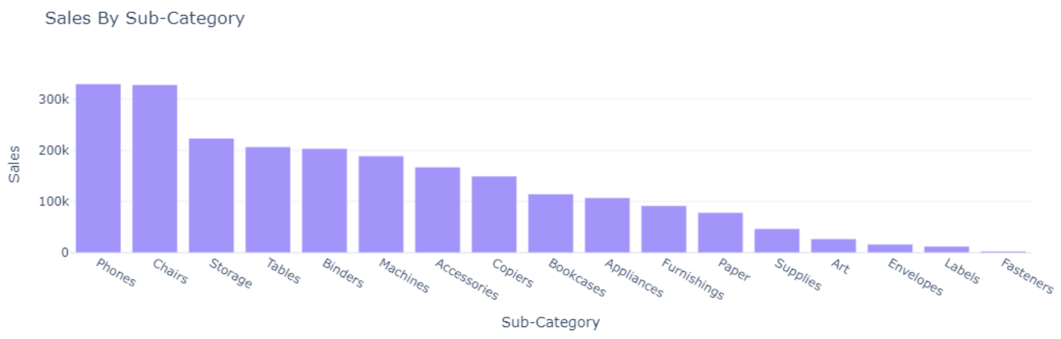
```
[217]: Counting = len(pd.unique(df['Sub-Category']))
       Counting
```

```
[217]: 17
```

```
[348]: sales_by_subcategory = df.groupby('Sub-Category')['Sales'].sum().reset_index()
       sales_by_subcategory.sort_values(by='Sales')
```

```
[348]:    Sub-Category        Sales
       8     Fasteners    3024.2800
       10       Labels   12486.3120
       7     Envelopes   16476.4020
       2           Art   27118.7920
       15     Supplies   46673.5380
       12        Paper   78479.2060
       9   Furnishings   91705.1640
       1    Appliances  107532.1610
       4     Bookcases  114879.9963
       6       Copiers  149528.0300
       0   Accessories  167380.3180
       11     Machines  189238.6310
       3       Binders  203412.7330
       16       Tables  206965.5320
       14      Storage  223843.6080
       5        Chairs  328449.1030
       13       Phones  330007.0540
```

```
[368]: purple_shades = ['#A294F9']
       fig = px.bar(sales_by_subcategory,
                    x= 'Sub-Category',
                    y='Sales',
                    title = 'Sales By Sub-Category',
                    color_discrete_sequence=purple_shades
       )
       fig.update_layout(xaxis={'categoryorder': 'total descending'})
       fig.show()
```



1.1.11 Insight:

1. Top Sub-Categories by Sales: Phones and Chairs are the highest-performing sub-categories, each generating over 300k in sales.
2. Lowest Sub-Categories by Sales: Fasteners and Labels contribute the least to sales.

1.1.12 Recommendation:

1. Focus on maintaining inventory and providing attractive promotions for Phones and Chairs to sustain their strong performance.
2. Investigate the low sales in Fasteners and Labels to determine if they are essential to the product mix; if not, consider reducing inventory or discontinuing these items.

1.1.13 4. You need to analyze the monthly profit from sales and determine which month had the highest profit.

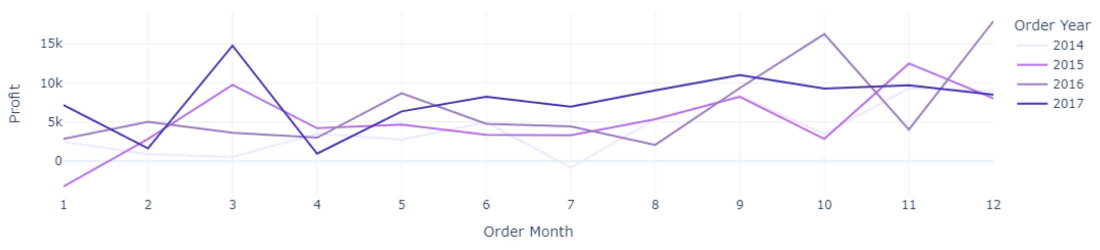
[585]:
```
monthly_profit = df.groupby(['Order Year', 'Order Month'],␣
 ↪observed=False)['Profit'].sum().reset_index()
monthly_profit
```

[585]:
```
    Order Year  Order Month     Profit
0         2014            1  2450.1907
1         2014            2   862.3084
2         2014            3   498.7299
3         2014            4  3488.8352
4         2014            5  2738.7096
5         2014            6  4976.5244
6         2014            7  -841.4826
7         2014            8  5318.1050
8         2014            9  8328.0994
9         2014           10  3448.2573
10        2014           11  9292.1269
11        2014           12  8983.5699
12        2015            1 -3281.0070
```

```
13        2015             2   2813.8508
14        2015             3   9732.0978
15        2015             4   4187.4962
16        2015             5   4667.8690
17        2015             6   3335.5572
18        2015             7   3288.6483
19        2015             8   5355.8084
20        2015             9   8209.1627
21        2015            10   2817.3660
22        2015            11  12474.7884
23        2015            12   8016.9659
24        2016             1   2824.8233
25        2016             2   5004.5795
26        2016             3   3611.9680
27        2016             4   2977.8149
28        2016             5   8662.1464
29        2016             6   4750.3781
30        2016             7   4432.8779
31        2016             8   2062.0693
32        2016             9   9328.6576
33        2016            10  16243.1425
34        2016            11   4011.4075
35        2016            12  17885.3093
36        2017             1   7140.4391
37        2017             2   1613.8720
38        2017             3  14751.8915
39        2017             4    933.2900
40        2017             5   6342.5828
41        2017             6   8223.3357
42        2017             7   6952.6212
43        2017             8   9040.9557
44        2017             9  10991.5556
45        2017            10   9275.2755
46        2017            11   9690.1037
47        2017            12   8483.3468
```

[595]:
```
purple_shades = ['#EBEAFF', '#B771E5','#9B7EBD', '#4635B1']
fig = px.line(monthly_profit, x = 'Order Month', y = 'Profit',
title = 'Profit By Month and Year', color='Order Year',
color_discrete_sequence=purple_shades)
fig.update_xaxes(type='category')
fig.show()
```

Profit By Month and Year



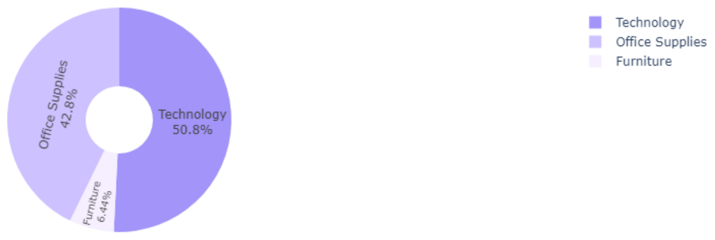
1.1.14 5. Analyze the profit by category and sub-category.

[261]:
```
profit_by_category = df.groupby('Category')['Profit'].sum().reset_index()
profit_by_category
```

[261]:
```
          Category        Profit
0        Furniture    18451.2728
1  Office Supplies   122490.8008
2       Technology   145454.9481
```

[370]:
```
purple_shades = ['#A294F9', '#CDC1FF', '#F5EFFF']
fig = px.pie(profit_by_category,
             names = 'Category',
             values = 'Profit',
             hole = 0.3,
             color_discrete_sequence = purple_shades
             )
fig.update_traces(textposition = 'inside', textinfo = 'percent+label')
fig.update_layout(title_text = 'Profit Analysis by Category')
fig.show()
```

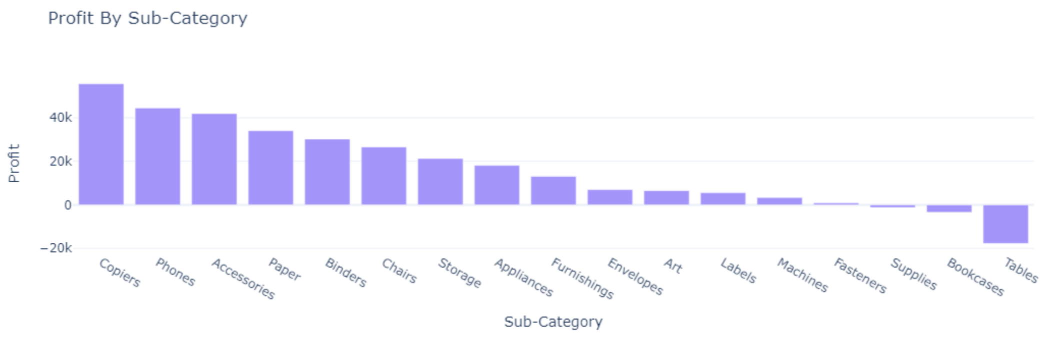
Profit Analysis by Category



[275]:
```
profit_by_sub_category = df.groupby('Sub-Category')['Profit'].sum().
  ↪reset_index()
profit_by_sub_category
```

[275]:
```
   Sub-Category      Profit
0   Accessories  41936.6357
1    Appliances  18138.0054
2           Art   6527.7870
3       Binders  30221.7633
4     Bookcases  -3472.5560
5        Chairs  26590.1663
6       Copiers  55617.8249
7     Envelopes   6964.1767
8     Fasteners    949.5182
9   Furnishings  13059.1436
10       Labels   5546.2540
11     Machines   3384.7569
12        Paper  34053.5693
13       Phones  44515.7306
14      Storage  21278.8264
15     Supplies  -1189.0995
16       Tables -17725.4811
```

[380]:
```
purple_shades = ['#A294F9']
fig = px.bar(profit_by_sub_category,
            x= 'Sub-Category',
            y='Profit',
            title = 'Profit By Sub-Category',
            color_discrete_sequence = purple_shades
)
fig.update_layout(xaxis={'categoryorder': 'total descending'})
fig.show()
```



1.1.15 Insight:

1. Copiers, Phones, and Accessories generate the highest profits, indicating strong market demand and effective pricing strategies.
2. Tables and Bookcases result in significant losses, suggesting inefficiencies or poor demand in these sub-categories. ### Recommendations:
3. Focus on High-Performing Sub-Categories: Increase marketing efforts and inventory for Copiers, Phones, and Accessories to further capitalize on their profitability.
4. Reassess Loss-Making Sub-Categories: Evaluate pricing, demand, and supply chain for Tables and Bookcases. Consider discontinuing or re-strategizing these products.
5. Optimize Marginal Sub-Categories: Improve efficiency and promotion for sub-categories with marginal profits, such as Art and Labels, to boost overall profitability.

1.1.16 6. Analyze the sales and profit by customer segment.

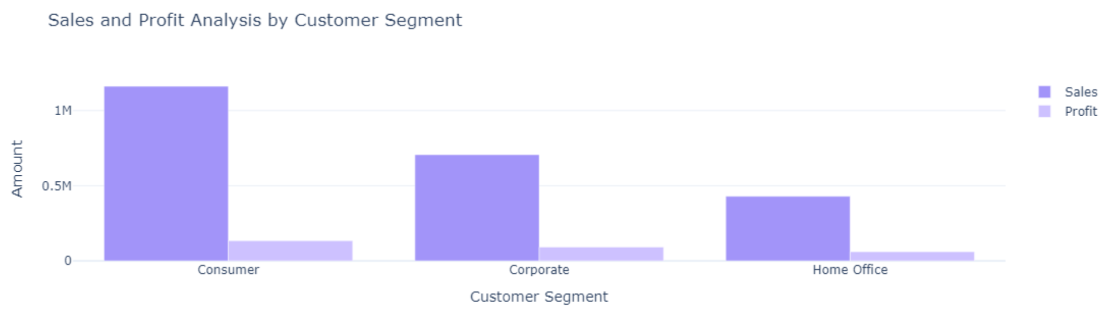
[324]:
```
sales_profit_bySegment = df.groupby('Segment').agg({'Sales' : 'sum', 'Profit':␣
↪'sum'}).reset_index()
sales_profit_bySegment
```

[324]:
```
       Segment         Sales       Profit
0     Consumer  1.161401e+06  134119.2092
1    Corporate  7.061464e+05   91979.1340
2  Home Office  4.296531e+05   60298.6785
```

[384]:
```
color_palette = ['#A294F9', '#CDC1FF']
fig = go.Figure()
fig.add_trace(go.Bar(
    x=sales_profit_bySegment['Segment'],
    y=sales_profit_bySegment['Sales'],
    name='Sales',
    marker_color=color_palette[0]))

fig.add_trace(go.Bar(
    x=sales_profit_bySegment['Segment'],
    y=sales_profit_bySegment['Profit'],
    name='Profit',
    marker_color=color_palette[1]))

fig.update_layout(title='Sales and Profit Analysis by Customer Segment',
                  xaxis_title='Customer Segment', yaxis_title='Amount')
fig.show()
```



1.1.17 Insight:

1. The Consumer segment generates the highest sales and profit, followed by Corporate, while Home Office has the lowest profit despite decent sales. ### Recommendations:
2. Focus on Home Office: Investigate the low profitability in the Home Office segment and review pricing or discount strategies.
3. Leverage Consumer Segment: Invest in targeted marketing campaigns to further capitalize on the high-performing Consumer segment.
4. Optimize Corporate Strategies: Explore ways to increase profitability in the Corporate segment, such as enhancing customer loyalty programs or reducing operational costs.

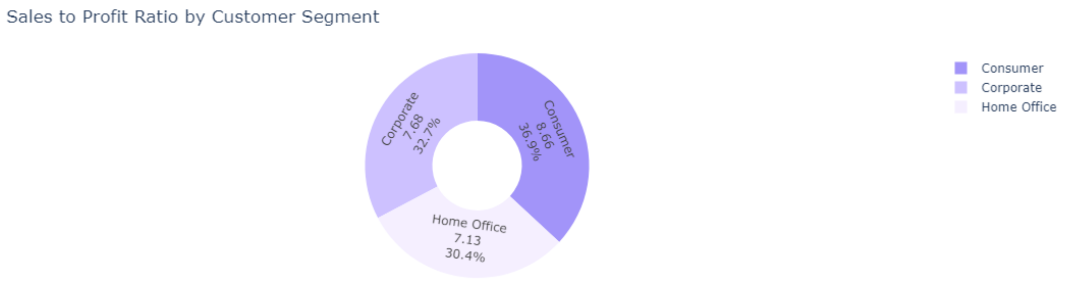
1.1.18 7. Analyze the sales to profit ratio.

[400]:
```
sales_profit_by_segment = df.groupby('Segment').agg({'Sales' : 'sum', 'Profit':␣
 ↪'sum'}).reset_index()
sales_profit_by_segment['Sales_to_Profit_Ratio'] =␣
 ↪sales_profit_by_segment['Sales']/sales_profit_by_segment['Profit']
sales_profit_by_segment['Sales_to_Profit_Ratio'] =␣
 ↪sales_profit_by_segment['Sales_to_Profit_Ratio'].round(2)
print(sales_profit_by_segment[['Segment', 'Sales_to_Profit_Ratio']])
```

```
       Segment  Sales_to_Profit_Ratio
0     Consumer                   8.66
1    Corporate                   7.68
2  Home Office                   7.13
```

[415]:
```
color_palette = ['#A294F9', '#CDC1FF', '#F5EFFF']
fig = px.pie(
    sales_profit_by_segment,
    names='Segment',
    values='Sales_to_Profit_Ratio',
    hole=0.4,
    color_discrete_sequence=color_palette
)
```

```
fig.update_traces(textposition='inside', textinfo='label+percent+value')
fig.update_layout(title='Sales to Profit Ratio by Customer Segment')
fig.show()
```



1.1.19 Insight:

1. The Consumer segment has the highest sales-to-profit ratio (36.9%), indicating strong profitability.
2. The Corporate segment follows closely (32.7%), while Home Office has the lowest ratio (30.4%), suggesting inefficiencies or higher costs. ### Recommendations:
3. Improve Home Office Efficiency: Identify cost drivers in the Home Office segment and optimize operational processes or pricing strategies.
4. Maintain Consumer Profitability: Continue investing in the Consumer segment to sustain its high sales-to-profit ratio.
5. Increase Corporate Focus: Enhance profitability in the Corporate segment through strategic upselling or optimizing shipping costs.