Ecommercepythonproject

February 27, 2025

[1]:
```
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"
```

[2]:
```
dataset = pd.read_csv(r'C:\Users\Admin\Downloads\Sample - Superstore.csv',␣
↪encoding='latin-1')
```

[3]:
```
dataset.head()
```

[3]:

	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]
```

[4]:
```
dataset.describe()
```

[4]:
```
            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
```

[5]:
```
dataset.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 Converting date columns

[6]:
```
dataset['Order Date'] = pd.to_datetime(dataset['Order Date'])
dataset['Ship Date'] = pd.to_datetime(dataset['Ship Date'])
```

[7]:
```
dataset.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
```

[8]:
```
dataset.head()
```

[8]:

	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]

[9]:
```
dataset['Order Month']  = dataset['Order Date'].dt.month
dataset['Order Year'] = dataset['Order Date'].dt.year
dataset['Order Day Of Week'] = dataset['Order Date'].dt.dayofweek
```

[10]:
```
dataset.head()
```

[10]:
```
   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 Month  Order Year  Order Day Of Week
0     0.00   41.9136           11        2016                  1
1     0.00  219.5820           11        2016                  1
2     0.00    6.8714            6        2016                  6
3     0.45 -383.0310           10        2015                  6
4     0.20    2.5164           10        2015                  6

[5 rows x 24 columns]
```

2 Monthly Sales Analysis

[11]:
```
#You need to calculate monthly sales pf the store and identify whih ,onth had␣
↪the highest and lowest sale.
```

[12]:
```
Sales_by_month = dataset.groupby('Order Month')['Sales'].sum().reset_index()
```

[13]:
```
Sales_by_month
```

[13]:

	Order Month	Sales
0	1	94924.8356
1	2	59751.2514
2	3	205005.4888
3	4	137762.1286
4	5	155028.8117
5	6	152718.6793
6	7	147238.0970
7	8	159044.0630
8	9	307649.9457
9	10	200322.9847
10	11	352461.0710
11	12	325293.5035

[14]:
```
fig = px.line(Sales_by_month,
              x= 'Order Month',
              y= 'Sales',
              title= 'Monthly Sales Analysis')
fig.show()
```

3 sales by category

[15]:
```
#you need to analyze sales based on product categories which category has the␣
 ↪lowest and highest sale
```

[16]:
```
sales_by_category = dataset.groupby(['Category'])['Sales'].sum().reset_index()
```

[17]:
```
sales_by_category
```

[17]:

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

[18]:
```
fig = px.pie(sales_by_category,
             values= 'Sales',
             names= 'Category',
             hole=0.3,
             color_discrete_sequence= px.colors.qualitative.Pastel)

fig.update_traces(textposition='inside', textinfo= 'percent+label')
fig.update_layout(title_text='Sales Analysis by Category',␣
 ↪title_font=dict(size=24))

fig.show()
```

4 Sales analysis by sub category

[19]:
```
#sales analysis needs to be done based on sub-categories
```

[20]:
```
sales_by_subcategory = dataset.groupby('Sub-Category')['Sales'].sum().
↪reset_index()
```

[21]:
```
sales_by_subcategory
```

[21]:

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

[22]:
```
fig = px.bar(sales_by_subcategory, x= 'Sub-Category', y= 'Sales', title ="Sales␣
↪analysis by sub category")

fig.show()
```

5 monthly profit analysis

[23]:
```
#you need to analyze monthly profit from sales and detemine which month has␣
↪highest profit
```

[24]:
```
profit_by_month = dataset.groupby('Order Month')['Profit'].sum().reset_index()
```

[25]:
```
profit_by_month
```

[25]:

	Order Month	Profit
0	1	9134.4461

```
1              2  10294.6107
2              3  28594.6872
3              4  11587.4363
4              5  22411.3078
5              6  21285.7954
6              7  13832.6648
7              8  21776.9384
8              9  36857.4753
9             10  31784.0413
10            11  35468.4265
11            12  43369.1919
```

[26]:
```
fig = px.line(profit_by_month, x= 'Order Month', y= 'Profit', title = "Monhly␣
 ↪profit analysis")

fig.show()
```

6 profit by category

[27]:
```
#Profit by category and sub-category
```

[28]:
```
profit_by_category= dataset.groupby('Category')['Profit'].sum().reset_index()
```

[29]:
```
profit_by_category
```

[29]:
```
          Category       Profit
0        Furniture   18451.2728
1  Office Supplies  122490.8008
2       Technology  145454.9481
```

[30]:
```
fig = px.pie(profit_by_category,
             values= 'Profit',
             names= 'Category',
             hole=0.3,
             color_discrete_sequence= px.colors.qualitative.Pastel)

fig.update_traces(textposition='inside', textinfo= 'percent+label')
fig.update_layout(title_text='Profit Analysis by Category',␣
 ↪title_font=dict(size=24))

fig.show()
```

7 Profit by sub category

[31]:
```
profit_by_subcategory = dataset.groupby('Sub-Category')['Profit'].sum().
 ↪reset_index()

fig= px.bar(profit_by_subcategory, x='Sub-Category',
            y = 'Profit',
            title= "Profit Analysis by Sub- Category")

fig.show()
```

8 sales and profit - customer segment

[32]:
```
#analyze sales and profit by customer segment
```

[33]:
```
sales_profit_by_segment = dataset.groupby('Segment').agg({'Sales': 'sum',␣
 ↪'Profit' : 'sum'}).reset_index()

color_palette = colors.qualitative.Pastel

fig= go.Figure()
fig.add_trace(go.Bar(x =sales_profit_by_segment['Segment'],
                     y =sales_profit_by_segment['Sales'],
                    name='Sales',
                    marker_color=color_palette[0]))

fig.add_trace(go.Bar(x = sales_profit_by_segment['Segment'],
                     y = sales_profit_by_segment['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()
```

9 Profit ratio

[34]:
```
#Analyze sales profit ratio
```

[35]:
```
sales_profit_by_segment = dataset.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']
print(sales_profit_by_segment[['Segment', 'Sales_to_Profit_Ratio']])
```

```
        Segment  Sales_to_Profit_Ratio
0      Consumer               8.659471
1     Corporate               7.677245
2   Home Office               7.125416
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: