```
# Import libraries for data manipulation
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.preprocessing import LabelEncoder
from sklearn.model_import MinMaxScaler
```

Data Loading

```
# Load the dataset
df = pd.read_csv('/kaggle/input/superstore-dataset/Superstore.csv')
df.head()
```

	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
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.96	2	0.00	41.9184
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs, Round Back	731.9400	3	0.00	219.5820
2	3	CA-2016-138688	6/12/2016	6/16/2016	First Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters by Universal	14.62	2	0.00	6.8714
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.031
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold Flat Components Storage	22.368	2	0.20	2.516

```
# Display basic information about the dataset
df.info()
```

```
<class'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 Row ID         9994 non-null int64
 Order ID       9994 non-null object
 Order Date     9994 non-null object
 Ship Date      9994 non-null object
 Ship Mode      9994 non-null object
 Customer ID    9994 non-null object
 Customer Name  9994 non-null object
 Segment        9994 non-null object
 Country        9994 non-null object
 City           9994 non-null object
 State          9994 non-null object
 Postal Code    9994 non-null int64
 Region         9994 non-null object
 Product ID     9994 non-null object
 Category       9994 non-null object
 Sub-Category   9994 non-null object
 Product Name   9994 non-null object
 Sales          9994 non-null float64
 Quantity       9994 non-null int64
 Discount       9994 non-null float64
 Profit         9994 non-null float64
dtypes: float64(4), int64(3), object(14)
memory usage: 1.6+ MB
```

- **Observation:** No missing values

Data Cleaning and Preparation

```
# Check for missing values in each column
df.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
Country        0
City           0
State          0
Postal Code    0
Region         0
Product ID     0
Category       0
Sub-Category   0
Product Name   0
Sales          0
Quantity       0
Discount       0
Profit         0
dtypes: int64
```

- **Observation:** No missing values

```
# Check if there are any duplicate rows
df.duplicated().sum
```

```
0
```

- **Observation:** No duplicate rows

```
# Convert dates to datetime format
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['Ship Date'] = pd.to_datetime(df['Ship Date'])
df.head()
```

	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
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.96	2	0.00	41.9184
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs, Round Back	731.9400	3	0.00	219.5820
2	3	CA-2016-138688	2016-06-12	2016-06-16	First Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters by Universal	14.62	2	0.00	6.8714
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.031
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold Flat Components Storage	22.368	2	0.20	2.516

```
# Check unique values in categorical columns
df['Ship Mode'].unique()
df['Segment'].unique()
df['Country'].unique()
df['Region'].unique()
df['Category'].unique()
df['Sub-Category'].unique()
df['Ship Mode'].unique()
df['Segment'].unique()
df['Country'].unique()
df['Region'].unique()
df['Category'].unique()
df['Sub-Category'].unique()
```

```
array['Second Class', 'First Class', 'Standard Class', 'Same Day']
array['Consumer', 'Corporate', 'Home Office']
array['United States']
array['South', 'West', 'Central', 'East']
array['Furniture', 'Office Supplies', 'Technology']
array['Bookcases', 'Chairs', 'Labels', 'Tables', 'Storage',
       'Furnishings', 'Art', 'Phones', 'Binders', 'Accessories',
       'Machines', 'Copiers', 'Bookcases', 'Chairs', 'Labels',
       'Tables', 'Storage', 'Furnishings', 'Art', 'Phones',
       'Binders', 'Accessories', 'Machines', 'Copiers']
array['Second Class', 'First Class', 'Standard Class', 'Same Day']
array['Consumer', 'Corporate', 'Home Office']
array['United States']
array['South', 'West', 'Central', 'East']
array['Furniture', 'Office Supplies', 'Technology']
array['Bookcases', 'Chairs', 'Labels', 'Tables', 'Storage',
       'Furnishings', 'Art', 'Phones', 'Binders', 'Accessories',
       'Machines', 'Copiers', 'Bookcases', 'Chairs', 'Labels',
       'Tables', 'Storage', 'Furnishings', 'Art', 'Phones',
       'Binders', 'Accessories', 'Machines', 'Copiers']
```

- **Observation:** Only one country (United States) in the dataset, so this column will be dropped.

```
# Drop Country column since it's all United States
df = df.drop(['Country'], axis=1)
df.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	State	Postal Code	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount	Profit
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.96	2	0.00	41.9184
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs, Round Back	731.9400	3	0.00	219.5820
2	3	CA-2016-138688	2016-06-12	2016-06-16	First Class	DV-13045	Darrin Van Huff	Corporate	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters by Universal	14.62	2	0.00	6.8714
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	Bretford CR4500 Series Slim Rectangular Table	957.5775	5	0.45	-383.031
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	Eldon Fold Flat Components Storage	22.368	2	0.20	2.516

- **Observation:** Dropped the Country column because it contains only one country (the United States)

```
# Create a copy of the dataset for analysis
df_analysis = df.copy()
print("Dataset shape:", df_analysis.shape)
```

```
Dataset shape: (9994, 20)
```

Exploratory Data Analysis

```
# Basic statistics for numerical columns
df.describe()
```

	Row ID	Order Date	Ship Date	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994	9994	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500500	2017-05-07 00:07:12.288000256	2017-05-11 08:33:07.220031488	55190.379428	229.858001	3.789574	0.156203	28.656896
min	1.000000	2014-01-03 00:00:00	2014-01-07 00:00:00	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	2015-05-23 00:00:00	2015-05-27 00:00:00	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	2016-06-26 00:00:00	2016-06-29 00:00:00	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	2017-05-14 00:00:00	2017-05-18 00:00:00	90008.000000	210.605000	5.000000	0.400000	29.364000
max	9994.000000	2017-12-30 00:00:00	2018-01-05 00:00:00	99301.000000	22638.480000	14.000000	0.800000	8399.976000
std	2885.163629	NaN	NaN	32063.693350	623.245101	2.236035	0.206291	234.260108

```
# Check distribution of numerical columns
df.describe(include=['object'])
```

	Order ID	Ship Mode	Customer ID	Customer Name	Segment	City	State	Region	Product ID	Category	Sub-Category	Product Name
count	9994	9994	9994	9994	9994	9994	9994	9994	9994	9994	9994	9994
unique	5009	4	793	793	3	531	49	4	1862	3	17	1894
top	CA-2017-100111	Standard Class	WB-21850	William Brown	Consumer	New York City	California	West	OFF-PA-10001970	Office Supplies	Binders	Staples
freq	14	5968	37	37	5191	915	2001	3203	48	6026	1523	46

```
# Create a copy for analysis
df_analysis = df.copy()
print("Dataset shape:", df_analysis.shape)
```

```
Dataset shape: (9994, 20)
```

```
# Sales distribution by segment
plt.figure(figsize=(12,6))
sns.boxplot(x='Segment', y='Sales', data=df)
plt.title('Sales Distribution by Segment')
plt.ylim(0, 1000)
plt.show()
```

```
# Sales distribution by ship mode
plt.figure(figsize=(12,6))
sns.boxplot(x='Ship Mode', y='Sales', data=df)
plt.title('Sales Distribution by Ship Mode')
plt.ylim(0, 1000)
plt.show()
```

```
# Sales distribution by category
plt.figure(figsize=(12,6))
sns.boxplot(x='Category', y='Sales', data=df)
plt.title('Sales Distribution by Category')
plt.ylim(0, 1000)
plt.show()
```

```
# Sales distribution by sub-category
plt.figure(figsize=(15,8))
sns.boxplot(x='Sub-Category', y='Sales', data=df)
plt.title('Sales Distribution by Sub-Category')
plt.ylim(0, 1000)
plt.xticks(rotation=45)
plt.show()
```

OBSERVATION

- Standard Class is the most preferred shipping mode.
- Category Family Office Supplies is shipped more number of time.
- Consumer segment is the most frequent customer and number of time.
- Sales distribution is a little higher in United States.
- Consumer segment has the highest shipping record.

```
# Sales distribution by profit
plt.figure(figsize=(12,6))
sns.boxplot(x='Profit', y='Sales', data=df)
plt.title('Sales Distribution by Profit')
plt.ylim(0, 1000)
plt.xlim(-500, 500)
plt.show()
```

```
# Sales distribution by quantity
plt.figure(figsize=(12,6))
sns.boxplot(x='Quantity', y='Sales', data=df)
plt.title('Sales Distribution by Quantity')
plt.ylim(0, 1000)
plt.show()
```

```
# Sales distribution by discount
plt.figure(figsize=(12,6))
sns.boxplot(x='Discount', y='Sales', data=df)
plt.title('Sales Distribution by Discount')
plt.ylim(0, 1000)
plt.show()
```

```
# Correlation matrix for numerical columns
numerical_cols = ['Sales', 'Quantity', 'Discount', 'Profit']
correlation = df[numerical_cols].corr()
plt.figure(figsize=(8,6))
sns.heatmap(correlation, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Matrix')
plt.show()
```

```
# Scatter plots for relationships
plt.figure(figsize=(12,4))
plt.subplot(1,3,1)
sns.scatter(df['Sales'], df['Profit'])
plt.title('Sales vs Profit')
plt.subplot(1,3,2)
sns.scatter(df['Quantity'], df['Profit'])
plt.title('Quantity vs Profit')
plt.subplot(1,3,3)
sns.scatter(df['Discount'], df['Profit'])
plt.title('Discount vs Profit')
plt.tight_layout()
plt.show()
```

OBSERVATION

- Sales and Profit are positively correlated.
- Discount and Profit are negatively correlated.
- Quantity and Profit show weak correlation.

```
# Sales by category
plt.figure(figsize=(12,6))
df.groupby('Category')['Sales'].sum.plot(kind='bar')
plt.title('Total Sales by Category')
plt.xlabel('Category')
plt.ylabel('Total Sales')
plt.xticks(rotation=0)
plt.show
```


OBSERVATION

- Technology is the most profitable category followed by Office Supplies and then Furniture.
- Technology sub-category is sales are category is profitable and Tables is sales profitable.
- In Technology category, the sub-category Copiers is highest profitable.
- Furniture sub-category is sales are category is profitable and Tables is sales profitable.
- Office Supplies sub-category is sales are category is profitable and Binders is sales profitable.

```
# Profit by category
plt.figure(figsize=(12,6))
df.groupby('Category')['Profit'].sum.plot(kind='bar')
plt.title('Total Profit by Category')
plt.xlabel('Category')
plt.ylabel('Total Profit')
plt.xticks(rotation=0)
plt.show
```

```
# Sales by ship mode
plt.figure(figsize=(12,6))
df.groupby('Ship Mode')['Sales'].sum.plot(kind='bar')
plt.title('Total Sales by Ship Mode')
plt.xlabel('Ship Mode')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show
```

```
# Profit by ship mode
plt.figure(figsize=(12,6))
df.groupby('Ship Mode')['Profit'].sum.plot(kind='bar')
plt.title('Total Profit by Ship Mode')
plt.xlabel('Ship Mode')
plt.ylabel('Total Profit')
plt.xticks(rotation=45)
plt.show
```

```
# Sales by segment
plt.figure(figsize=(12,6))
df.groupby('Segment')['Sales'].sum.plot(kind='bar')
plt.title('Total Sales by Segment')
plt.xlabel('Segment')
plt.ylabel('Total Sales')
plt.xticks(rotation=0)
plt.show
```



```
# Profit by segment
plt.figure(figsize=(12,6))
df.groupby('Segment')['Profit'].sum.plot(kind='bar')
plt.title('Total Profit by Segment')
plt.xlabel('Segment')
plt.ylabel('Total Profit')
plt.xticks(rotation=0)
plt.show
```



```
# Sales by sub-category
plt.figure(figsize=(15,8))
df.groupby('Sub-Category')['Sales'].sum.plot(kind='bar')
plt.title('Total Sales by Sub-Category')
plt.xlabel('Sub-Category')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show
```



```
# Profit by sub-category
plt.figure(figsize=(15,8))
df.groupby('Sub-Category')['Profit'].sum.plot(kind='bar')
plt.title('Total Profit by Sub-Category')
plt.xlabel('Sub-Category')
plt.ylabel('Total Profit')
plt.xticks(rotation=45)
plt.show
```



```
# Top 10 states by sales
top_states = df.groupby('State')['Sales'].sum.sort_values(ascending=False).head(10)
print("Top 10 States by Sales:")
top_states
```

```
Top 10 States by Sales:
State
California              457687.6315
New York                310876.2710
Texas                   170188.0458
Washington              138641.2700
Pennsylvania            116511.9140
Florida                 89473.7080
Illinois                80166.1010
Ohio                    78258.1360
Michigan                76269.6140
Virginia                70636.7200
Name: Sales, dtype: float64
```

```
# Top 10 states by profit
top_states_profit = df.groupby('State')['Profit'].sum.sort_values(ascending=False).head(10)
print("Top 10 States by Profit:")
top_states_profit
```

```
Top 10 States by Profit:
State
California              76381.3871
New York                74070.2640
Washington              33402.6510
Michigan                24463.1876
Virginia                18597.9504
Indiana                 18382.9363
Minnesota               10823.1874
Delaware                9977.3748
Kentucky                9301.7855
Arizona                 8958.5760
Name: Profit, dtype: float64
```

```
# Top 10 states by number of orders
top_states_orders = df.groupby('State')['Order ID'].count.sort_values(ascending=False).head(10)
print("Top 10 States by Number of Orders:")
top_states_orders
```

```
Top 10 States by Number of Orders:
State
California              2001
New York                1915
Texas                   1695
Washington              1148
Pennsylvania            1035
Florida                 763
Illinois                635
Ohio                    469
Michigan                424
Virginia                352
Name: Order ID, dtype: int64
```

```
# Top 10 states by average sales
top_states_avg = df.groupby('State')['Sales'].mean.sort_values(ascending=False).head(10)
print("Top 10 States by Average Sales:")
top_states_avg
```

```
Top 10 States by Average Sales:
State
District of Columbia    1027.000000
California              228.857859
New York                162.339861
Texas                   100.392224
Washington              120.761028
Massachusetts           81.777964
Maryland                71.931836
Utah                    64.283333
Nevada                  60.996667
Rhode Island            146.192105
Name: Sales, dtype: float64
```

```
# Top 10 states by average profit
top_states_avg_profit = df.groupby('State')['Profit'].mean.sort_values(ascending=False).head(10)
print("Top 10 States by Average Profit:")
top_states_avg_profit
```

```
Top 10 States by Average Profit:
State
District of Columbia    512.000000
California              38.190935
New York                38.689819
Texas                   5.909164
Washington              29.094347
Massachusetts           24.533389
Maryland                21.579551
Utah                    19.285000
Nevada                  18.298000
Rhode Island            43.857632
Name: Profit, dtype: float64
```

```
# Top 10 states by profit
top_states_profit = df.groupby('State')['Profit'].sum.sort_values(ascending=False).head(10)
plt.figure(figsize=(12,6))
top_states_profit.plot(kind='bar')
plt.title('Top 10 States by Profit')
plt.xlabel('State')
plt.ylabel('Total Profit')
plt.xticks(rotation=45)
plt.show
```



```
# Top 10 states by sales
top_states = df.groupby('State')['Sales'].sum.sort_values(ascending=False).head(10)
plt.figure(figsize=(12,6))
top_states.plot(kind='bar')
plt.title('Top 10 States by Sales')
plt.xlabel('State')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show
```



```
# Top 10 states by average sales
top_states_avg = df.groupby('State')['Sales'].mean.sort_values(ascending=False).head(10)
plt.figure(figsize=(12,6))
top_states_avg.plot(kind='bar')
plt.title('Top 10 States by Average Sales')
plt.xlabel('State')
plt.ylabel('Average Sales')
plt.xticks(rotation=45)
plt.show
```



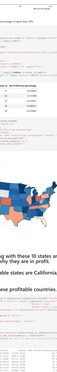
```
# Top 10 states by average profit
top_states_avg_profit = df.groupby('State')['Profit'].mean.sort_values(ascending=False).head(10)
plt.figure(figsize=(12,6))
top_states_avg_profit.plot(kind='bar')
plt.title('Top 10 States by Average Profit')
plt.xlabel('State')
plt.ylabel('Average Profit')
plt.xticks(rotation=45)
plt.show
```


OBSERVATION

- The top 10 states by sales are California from West region, New York from East region, Virginia from South region and Michigan from Midwest.
- These are over 70 states having by accounting for one-sixth percentage.
- The top 10 states by profit are California from West region, Washington, Ohio from East region, Florida, Tennessee, North Carolina from South region and Minnesota, New Jersey from Midwest region.

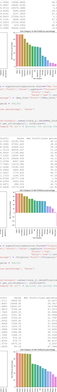
```
# count the top 10 sub-categories by loss in the Binders group
top10 = df.groupby('Sub-Category').sum().sort_values('Profit').head(10)
top10.reset_index(inplace=True)
sns.barplot(data=top10,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 10 Sub-Categories by Loss',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0      Binders  1.700000 -3972.  3723.                   -106.692
1       Tables  1.400000 -1708.  9626.                    -17.746
2     Machines  1.000000 -3384.  3384.                   -100.000
3      Bookcases  0.800000 -3099.  5395.                    -57.441
4       Supplies  0.800000 -1189.  1647.                    -72.221
5       Storage  0.800000 -1078.  3158.                    -34.135
6        Copiers  0.800000  1780.  6480.                     27.469
7        Phones  0.800000  1729.  8465.                     20.429
8       Chairs  0.800000  2546.  8027.                     31.718
9       Furnishings  0.800000  2364.  9772.                     24.187
```



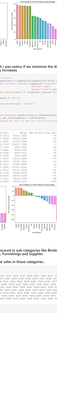

```
# Top 5 sub-categories based on profit
top5 = df.groupby('Sub-Category').sum().sort_values('Profit').tail(5)
top5.reset_index(inplace=True)
sns.barplot(data=top5,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 5 Sub-Categories by Profit',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0       Copiers  0.800000  1780.  6480.                     27.469
1        Phones  0.800000  1729.  8465.                     20.429
2       Chairs  0.800000  2546.  8027.                     31.718
3      Furnishings  0.800000  2364.  9772.                     24.187
4       Storage  0.800000  1078.  3158.                     34.135
```




```
# Top 10 sub-categories based on profit
top10 = df.groupby('Sub-Category').sum().sort_values('Profit').tail(10)
top10.reset_index(inplace=True)
sns.barplot(data=top10,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 10 Sub-Categories by Profit',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0       Copiers  0.800000  1780.  6480.                     27.469
1        Phones  0.800000  1729.  8465.                     20.429
2       Chairs  0.800000  2546.  8027.                     31.718
3      Furnishings  0.800000  2364.  9772.                     24.187
4       Storage  0.800000  1078.  3158.                     34.135
5       Accessories  0.800000  2076.  9048.                     22.944
6        Labels  0.800000   366.  1771.                     20.664
7       Envelopes  0.800000   594.  2965.                     20.034
8       Paper  0.800000  3420.  7781.                     43.948
9       Art  0.800000  1044.  3709.                     28.147
```




```
# Top 10 sub-categories based on profit
top10 = df.groupby('Sub-Category').sum().sort_values('Profit').head(10)
top10.reset_index(inplace=True)
sns.barplot(data=top10,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 10 Sub-Categories by Loss',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0      Binders  1.700000 -3972.  3723.                   -106.692
1       Tables  1.400000 -1708.  9626.                    -17.746
2     Machines  1.000000 -3384.  3384.                   -100.000
3      Bookcases  0.800000 -3099.  5395.                    -57.441
4       Supplies  0.800000 -1189.  1647.                    -72.221
5       Storage  0.800000 -1078.  3158.                    -34.135
6        Copiers  0.800000  1780.  6480.                     27.469
7        Phones  0.800000  1729.  8465.                     20.429
8       Chairs  0.800000  2546.  8027.                     31.718
9       Furnishings  0.800000  2364.  9772.                     24.187
```



```
# Top 5 sub-categories based on profit
top5 = df.groupby('Sub-Category').sum().sort_values('Profit').head(5)
top5.reset_index(inplace=True)
sns.barplot(data=top5,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 5 Sub-Categories by Loss',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0      Binders  1.700000 -3972.  3723.                   -106.692
1       Tables  1.400000 -1708.  9626.                    -17.746
2     Machines  1.000000 -3384.  3384.                   -100.000
3      Bookcases  0.800000 -3099.  5395.                    -57.441
4       Supplies  0.800000 -1189.  1647.                    -72.221
```



```
# Top 10 sub-categories based on profit
top10 = df.groupby('Sub-Category').sum().sort_values('Profit').head(10)
top10.reset_index(inplace=True)
sns.barplot(data=top10,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 10 Sub-Categories by Loss',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0      Binders  1.700000 -3972.  3723.                   -106.692
1       Tables  1.400000 -1708.  9626.                    -17.746
2     Machines  1.000000 -3384.  3384.                   -100.000
3      Bookcases  0.800000 -3099.  5395.                    -57.441
4       Supplies  0.800000 -1189.  1647.                    -72.221
5       Storage  0.800000 -1078.  3158.                    -34.135
6        Copiers  0.800000  1780.  6480.                     27.469
7        Phones  0.800000  1729.  8465.                     20.429
8       Chairs  0.800000  2546.  8027.                     31.718
9       Furnishings  0.800000  2364.  9772.                     24.187
```


OBSERVATION

Zoo customer pattern which I was notice if we minimise the discounts and increase the sales the profits will automatically increases

```
# Top 10 sub-categories based on profit
top10 = df.groupby('Sub-Category').sum().sort_values('Profit').head(10)
top10.reset_index(inplace=True)
sns.barplot(data=top10,x='Profit',y='Sub-Category',palette='Paired')
plt.xlabel('Profit',fontsize=15,fontweight='bold')
plt.ylabel('Sub-Category',fontsize=15,fontweight='bold')
plt.title('Top 10 Sub-Categories by Loss',fontsize=20,fontweight='bold')
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.rcParams['figure.dpi']=150
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.rcParams['figure.figure.size']=(8,6)
plt.show()
```

```
Sub-Category  Discount  Profit  Sales  Row_Profit_Sales_percentage
0      Binders  1.700000 -3972.  3723.                   -106.692
1       Tables  1.400000 -1708.  9626.                    -17.746
2     Machines  1.000000 -3384.  3384.                   -100.000
3      Bookcases  0.800000 -3099.  5395.                    -57.441
4       Supplies  0.800000 -1189.  1647.                    -72.221
5       Storage  0.800000 -1078.  3158.                    -34.135
6        Copiers  0.800000  1780.  6480.                     27.469
7        Phones  0.800000  1729.  8465.                     20.429
8       Chairs  0.800000  2546.  8027.                     31.718
9       Furnishings  0.800000  2364.  9772.                     24.187
```


Final Solution

we have to minimize the discount in sub-categories like Binders, Tables, Machines, Bookcases, Supplies, Storage and increase in Copiers and Phones

Also we have to increase the sales in those categories

```
print('THEN WHY THINK')
```

```
THEN WHY THINK THINK THINK
```