Superstore Retail Analysis

November 9, 2022

1 Exploratory Data Analysis - Retail

Highlights : * Perform Exploratory Data Analysis (EDA) on dataset 'SampleSuperstore' * As a business manager, find the weak areas to work to make more profit * Derive business problems by exploring the data

This task will extract relevant, representative, and sufficient case study data from a reputable and reliable online source. Appropriate preprocessing adjustments and data exploration will be performed on the data to ensure reliable and reasonable outcomes and outputs. All significant interpretations and observations will be noted and considered for future improvements.

Following the purpose of this task, the primary focus will be on profit-related factors, which are the attributes 'Sales' and 'Profit' in American Dollars (USD$) measurements, as well as the integer value of 'Quantity' and percentage values of 'Discount' for each sales transaction. Analysing these will help to identify and assess concern areas.

1.1 import libraries

[19]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

1.2 reading data

[2]:
```
# import data
df = pd.read_csv('SampleSuperstore.csv')
df.head()
```

[2]:

	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
```

[3]:
```
# return the object type, which is dataframe
type(df)
```

[3]:
```
pandas.core.frame.DataFrame
```

The dataframe format type will facilitate the use of a wider variety of syntax and methods for data analysis, including describe() and info().

1.3 Data Preprocessing

[4]:
```
# display the number of entries, the number and names of the column attributes,␣
 ↪the data type and
# digit placings, and the memory space used
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
```

[5]:
```
# display unique categories for qualitative attributes
catagorical_features = [i for i in df.columns if df.dtypes[i] == 'object']
for j in catagorical_features:
    print(j)
    print(sorted(df[j].unique())) # sort in alphabetical order
```

```
Ship Mode
['First Class', 'Same Day', 'Second Class', 'Standard Class']
Segment
['Consumer', 'Corporate', 'Home Office']
Country
['United States']
City
['Aberdeen', 'Abilene', 'Akron', 'Albuquerque', 'Alexandria', 'Allen',
'Allentown', 'Altoona', 'Amarillo', 'Anaheim', 'Andover', 'Ann Arbor',
'Antioch', 'Apopka', 'Apple Valley', 'Appleton', 'Arlington', 'Arlington
Heights', 'Arvada', 'Asheville', 'Athens', 'Atlanta', 'Atlantic City', 'Auburn',
'Aurora', 'Austin', 'Avondale', 'Bakersfield', 'Baltimore', 'Bangor',
'Bartlett', 'Bayonne', 'Baytown', 'Beaumont', 'Bedford', 'Belleville',
'Bellevue', 'Bellingham', 'Bethlehem', 'Beverly', 'Billings', 'Bloomington',
'Boca Raton', 'Boise', 'Bolingbrook', 'Bossier City', 'Bowling Green', 'Boynton
Beach', 'Bozeman', 'Brentwood', 'Bridgeton', 'Bristol', 'Broken Arrow',
'Broomfield', 'Brownsville', 'Bryan', 'Buffalo', 'Buffalo Grove', 'Bullhead
City', 'Burbank', 'Burlington', 'Caldwell', 'Camarillo', 'Cambridge', 'Canton',
'Carlsbad', 'Carol Stream', 'Carrollton', 'Cary', 'Cedar Hill', 'Cedar Rapids',
'Champaign', 'Chandler', 'Chapel Hill', 'Charlotte', 'Charlottesville',
'Chattanooga', 'Chesapeake', 'Chester', 'Cheyenne', 'Chicago', 'Chico', 'Chula
Vista', 'Cincinnati', 'Citrus Heights', 'Clarksville', 'Cleveland', 'Clifton',
'Clinton', 'Clovis', 'Coachella', 'College Station', 'Colorado Springs',
'Columbia', 'Columbus', 'Commerce City', 'Concord', 'Conroe', 'Conway', 'Coon
Rapids', 'Coppell', 'Coral Gables', 'Coral Springs', 'Corpus Christi', 'Costa
Mesa', 'Cottage Grove', 'Covington', 'Cranston', 'Cuyahoga Falls', 'Dallas',
'Danbury', 'Danville', 'Davis', 'Daytona Beach', 'Dearborn', 'Dearborn Heights',
'Decatur', 'Deer Park', 'Delray Beach', 'Deltona', 'Denver', 'Des Moines', 'Des
Plaines', 'Detroit', 'Dover', 'Draper', 'Dublin', 'Dubuque', 'Durham', 'Eagan',
'East Orange', 'East Point', 'Eau Claire', 'Edinburg', 'Edmond', 'Edmonds', 'El
Cajon', 'El Paso', 'Elkhart', 'Elmhurst', 'Elyria', 'Encinitas', 'Englewood',
'Escondido', 'Eugene', 'Evanston', 'Everett', 'Fairfield', 'Fargo',
'Farmington', 'Fayetteville', 'Florence', 'Fort Collins', 'Fort Lauderdale',
'Fort Worth', 'Frankfort', 'Franklin', 'Freeport', 'Fremont', 'Fresno',
'Frisco', 'Gaithersburg', 'Garden City', 'Garland', 'Gastonia', 'Georgetown',
'Gilbert', 'Gladstone', 'Glendale', 'Glenview', 'Goldsboro', 'Grand Island',
'Grand Prairie', 'Grand Rapids', 'Grapevine', 'Great Falls', 'Greeley', 'Green
Bay', 'Greensboro', 'Greenville', 'Greenwood', 'Gresham', 'Grove City',
'Gulfport', 'Hackensack', 'Hagerstown', 'Haltom City', 'Hamilton', 'Hampton',
```

'Harlingen', 'Harrisonburg', 'Hattiesburg', 'Helena', 'Hempstead', 'Henderson', 'Hendersonville', 'Hesperia', 'Hialeah', 'Hickory', 'Highland Park', 'Hillsboro', 'Holland', 'Hollywood', 'Holyoke', 'Homestead', 'Hoover', 'Hot Springs', 'Houston', 'Huntington Beach', 'Huntsville', 'Independence', 'Indianapolis', 'Inglewood', 'Iowa City', 'Irving', 'Jackson', 'Jacksonville', 'Jamestown', 'Jefferson City', 'Johnson City', 'Jonesboro', 'Jupiter', 'Keller', 'Kenner', 'Kenosha', 'Kent', 'Kirkwood', 'Kissimmee', 'Knoxville', 'La Crosse', 'La Mesa', 'La Porte', 'La Quinta', 'Lafayette', 'Laguna Niguel', 'Lake Charles', 'Lake Elsinore', 'Lake Forest', 'Lakeland', 'Lakeville', 'Lakewood', 'Lancaster', 'Lansing', 'Laredo', 'Las Cruces', 'Las Vegas', 'Laurel', 'Lawrence', 'Lawton', 'Layton', 'League City', 'Lebanon', 'Lehi', 'Leominster', 'Lewiston', 'Lincoln Park', 'Linden', 'Lindenhurst', 'Little Rock', 'Littleton', 'Lodi', 'Logan', 'Long Beach', 'Longmont', 'Longview', 'Lorain', 'Los Angeles', 'Louisville', 'Loveland', 'Lowell', 'Lubbock', 'Macon', 'Madison', 'Malden', 'Manchester', 'Manhattan', 'Mansfield', 'Manteca', 'Maple Grove', 'Margate', 'Marietta', 'Marion', 'Marlborough', 'Marysville', 'Mason', 'Mcallen', 'Medford', 'Medina', 'Melbourne', 'Memphis', 'Mentor', 'Meriden', 'Meridian', 'Mesa', 'Mesquite', 'Miami', 'Middletown', 'Midland', 'Milford', 'Milwaukee', 'Minneapolis', 'Miramar', 'Mishawaka', 'Mission Viejo', 'Missoula', 'Missouri City', 'Mobile', 'Modesto', 'Monroe', 'Montebello', 'Montgomery', 'Moorhead', 'Moreno Valley', 'Morgan Hill', 'Morristown', 'Mount Pleasant', 'Mount Vernon', 'Murfreesboro', 'Murray', 'Murrieta', 'Muskogee', 'Naperville', 'Nashua', 'Nashville', 'New Albany', 'New Bedford', 'New Brunswick', 'New Castle', 'New Rochelle', 'New York City', 'Newark', 'Newport News', 'Niagara Falls', 'Noblesville', 'Norfolk', 'Normal', 'Norman', 'North Charleston', 'North Las Vegas', 'North Miami', 'Norwich', 'Oak Park', 'Oakland', 'Oceanside', 'Odessa', 'Oklahoma City', 'Olathe', 'Olympia', 'Omaha', 'Ontario', 'Orange', 'Orem', 'Orland Park', 'Orlando', 'Ormond Beach', 'Oswego', 'Overland Park', 'Owensboro', 'Oxnard', 'Palatine', 'Palm Coast', 'Park Ridge', 'Parker', 'Parma', 'Pasadena', 'Pasco', 'Passaic', 'Paterson', 'Pearland', 'Pembroke Pines', 'Pensacola', 'Peoria', 'Perth Amboy', 'Pharr', 'Philadelphia', 'Phoenix', 'Pico Rivera', 'Pine Bluff', 'Plainfield', 'Plano', 'Plantation', 'Pleasant Grove', 'Pocatello', 'Pomona', 'Pompano Beach', 'Port Arthur', 'Port Orange', 'Port Saint Lucie', 'Portage', 'Portland', 'Providence', 'Provo', 'Pueblo', 'Quincy', 'Raleigh', 'Rancho Cucamonga', 'Rapid City', 'Reading', 'Redding', 'Redlands', 'Redmond', 'Redondo Beach', 'Redwood City', 'Reno', 'Renton', 'Revere', 'Richardson', 'Richmond', 'Rio Rancho', 'Riverside', 'Rochester', 'Rochester Hills', 'Rock Hill', 'Rockford', 'Rockville', 'Rogers', 'Rome', 'Romeoville', 'Roseville', 'Roswell', 'Round Rock', 'Royal Oak', 'Sacramento', 'Saginaw', 'Saint Charles', 'Saint Cloud', 'Saint Louis', 'Saint Paul', 'Saint Peters', 'Saint Petersburg', 'Salem', 'Salinas', 'Salt Lake City', 'San Angelo', 'San Antonio', 'San Bernardino', 'San Clemente', 'San Diego', 'San Francisco', 'San Gabriel', 'San Jose', 'San Luis Obispo', 'San Marcos', 'San Mateo', 'Sandy Springs', 'Sanford', 'Santa Ana', 'Santa Barbara', 'Santa Clara', 'Santa Fe', 'Santa Maria', 'Scottsdale', 'Seattle', 'Sheboygan', 'Shelton', 'Sierra Vista', 'Sioux Falls', 'Skokie', 'Smyrna', 'South Bend', 'Southaven', 'Sparks', 'Spokane', 'Springdale', 'Springfield', 'Sterling Heights', 'Stockton', 'Suffolk', 'Summerville', 'Sunnyvale', 'Superior', 'Tallahassee',

```
'Tamarac', 'Tampa', 'Taylor', 'Temecula', 'Tempe', 'Texarkana', 'Texas City',
'The Colony', 'Thomasville', 'Thornton', 'Thousand Oaks', 'Tigard', 'Tinley
Park', 'Toledo', 'Torrance', 'Trenton', 'Troy', 'Tucson', 'Tulsa', 'Tuscaloosa',
'Twin Falls', 'Tyler', 'Urbandale', 'Utica', 'Vacaville', 'Vallejo',
'Vancouver', 'Vineland', 'Virginia Beach', 'Visalia', 'Waco', 'Warner Robins',
'Warwick', 'Washington', 'Waterbury', 'Waterloo', 'Watertown', 'Waukesha',
'Wausau', 'Waynesboro', 'West Allis', 'West Jordan', 'West Palm Beach',
'Westfield', 'Westland', 'Westminster', 'Wheeling', 'Whittier', 'Wichita',
'Wilmington', 'Wilson', 'Woodbury', 'Woodland', 'Woodstock', 'Woonsocket',
'Yonkers', 'York', 'Yucaipa', 'Yuma']
State
['Alabama', 'Arizona', 'Arkansas', 'California', 'Colorado', 'Connecticut',
'Delaware', 'District of Columbia', 'Florida', 'Georgia', 'Idaho', 'Illinois',
'Indiana', 'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland',
'Massachusetts', 'Michigan', 'Minnesota', 'Mississippi', 'Missouri', 'Montana',
'Nebraska', 'Nevada', 'New Hampshire', 'New Jersey', 'New Mexico', 'New York',
'North Carolina', 'North Dakota', 'Ohio', 'Oklahoma', 'Oregon', 'Pennsylvania',
'Rhode Island', 'South Carolina', 'South Dakota', 'Tennessee', 'Texas', 'Utah',
'Vermont', 'Virginia', 'Washington', 'West Virginia', 'Wisconsin', 'Wyoming']
Region
['Central', 'East', 'South', 'West']
Category
['Furniture', 'Office Supplies', 'Technology']
Sub-Category
['Accessories', 'Appliances', 'Art', 'Binders', 'Bookcases', 'Chairs',
'Copiers', 'Envelopes', 'Fasteners', 'Furnishings', 'Labels', 'Machines',
'Paper', 'Phones', 'Storage', 'Supplies', 'Tables']
```

[6]:
```
# count of unique values
df.nunique()
```

[6]:
```
Ship Mode          4
Segment            3
Country            1
City             531
State             49
Postal Code      631
Region             4
Category           3
Sub-Category      17
Sales           5825
Quantity          14
Discount          12
Profit          7287
dtype: int64
```

Duplicated rows or records can now by dropped from the dataset, as this redundancy may cause inaccurate results and outcomes (an assumption on the dataset).

[7]:
```
# detect duplicated records
df[df.duplicated(subset = None, keep = False)]
```

[7]:

	Ship Mode	Segment	Country	City	State
568	Standard Class	Corporate	United States	Seattle	Washington
591	Standard Class	Consumer	United States	Salem	Oregon
935	Standard Class	Home Office	United States	Philadelphia	Pennsylvania
950	Standard Class	Home Office	United States	Philadelphia	Pennsylvania
1186	Standard Class	Corporate	United States	Seattle	Washington
1479	Standard Class	Consumer	United States	San Francisco	California
2803	Standard Class	Consumer	United States	San Francisco	California
2807	Second Class	Consumer	United States	Seattle	Washington
2836	Standard Class	Consumer	United States	Los Angeles	California
3127	Standard Class	Consumer	United States	New York City	New York
3405	Standard Class	Home Office	United States	Columbus	Ohio
3406	Standard Class	Home Office	United States	Columbus	Ohio
3412	Standard Class	Corporate	United States	San Francisco	California
3670	Standard Class	Consumer	United States	Salem	Oregon
4117	Standard Class	Consumer	United States	Los Angeles	California
4553	Standard Class	Consumer	United States	San Francisco	California
5372	Standard Class	Corporate	United States	Houston	Texas
5493	Same Day	Home Office	United States	San Francisco	California
5905	Same Day	Home Office	United States	San Francisco	California
6146	Standard Class	Corporate	United States	San Francisco	California
6245	Standard Class	Home Office	United States	Seattle	Washington
6334	Standard Class	Consumer	United States	New York City	New York
6357	Standard Class	Corporate	United States	Seattle	Washington
6409	First Class	Consumer	United States	Houston	Texas
7608	Standard Class	Consumer	United States	San Francisco	California
7735	Standard Class	Corporate	United States	Seattle	Washington
7759	Standard Class	Corporate	United States	Houston	Texas
8032	First Class	Consumer	United States	Houston	Texas
8095	Second Class	Consumer	United States	Seattle	Washington
8457	Second Class	Corporate	United States	Chicago	Illinois
8533	Standard Class	Consumer	United States	Detroit	Michigan
9262	Standard Class	Consumer	United States	Detroit	Michigan
9363	Standard Class	Home Office	United States	Seattle	Washington
9477	Second Class	Corporate	United States	Chicago	Illinois

	Postal Code	Region	Category	Sub-Category	Sales	Quantity
568	98105	West	Office Supplies	Paper	19.440	3
591	97301	West	Office Supplies	Paper	10.368	2
935	19120	East	Office Supplies	Paper	15.552	3
950	19120	East	Office Supplies	Paper	15.552	3
1186	98103	West	Office Supplies	Paper	25.920	4
1479	94122	West	Office Supplies	Paper	25.920	4
2803	94122	West	Office Supplies	Paper	12.840	3

2807	98115	West	Office Supplies	Paper	12.960	2
2836	90036	West	Office Supplies	Paper	19.440	3
3127	10011	East	Office Supplies	Paper	49.120	4
3405	43229	East	Furniture	Chairs	281.372	2
3406	43229	East	Furniture	Chairs	281.372	2
3412	94122	West	Office Supplies	Art	11.760	4
3670	97301	West	Office Supplies	Paper	10.368	2
4117	90036	West	Office Supplies	Paper	19.440	3
4553	94122	West	Office Supplies	Paper	12.840	3
5372	77041	Central	Office Supplies	Paper	15.552	3
5493	94122	West	Office Supplies	Labels	41.400	4
5905	94122	West	Office Supplies	Labels	41.400	4
6146	94122	West	Office Supplies	Art	11.760	4
6245	98105	West	Furniture	Furnishings	22.140	3
6334	10011	East	Office Supplies	Paper	49.120	4
6357	98103	West	Office Supplies	Paper	25.920	4
6409	77041	Central	Office Supplies	Paper	47.952	3
7608	94122	West	Office Supplies	Paper	25.920	4
7735	98105	West	Office Supplies	Paper	19.440	3
7759	77041	Central	Office Supplies	Paper	15.552	3
8032	77041	Central	Office Supplies	Paper	47.952	3
8095	98115	West	Office Supplies	Paper	12.960	2
8457	60653	Central	Office Supplies	Binders	3.564	3
8533	48227	Central	Furniture	Chairs	389.970	3
9262	48227	Central	Furniture	Chairs	389.970	3
9363	98105	West	Furniture	Furnishings	22.140	3
9477	60653	Central	Office Supplies	Binders	3.564	3

	Discount	Profit
568	0.0	9.3312
591	0.2	3.6288
935	0.2	5.4432
950	0.2	5.4432
1186	0.0	12.4416
1479	0.0	12.4416
2803	0.0	5.7780
2807	0.0	6.2208
2836	0.0	9.3312
3127	0.0	23.0864
3405	0.3	-12.0588
3406	0.3	-12.0588
3412	0.0	3.1752
3670	0.2	3.6288
4117	0.0	9.3312
4553	0.0	5.7780
5372	0.2	5.4432
5493	0.0	19.8720

```
5905        0.0  19.8720
6146        0.0   3.1752
6245        0.0   6.4206
6334        0.0  23.0864
6357        0.0  12.4416
6409        0.2  16.1838
7608        0.0  12.4416
7735        0.0   9.3312
7759        0.2   5.4432
8032        0.2  16.1838
8095        0.0   6.2208
8457        0.8  -6.2370
8533        0.0  35.0973
9262        0.0  35.0973
9363        0.0   6.4206
9477        0.8  -6.2370
```

[8]:
```
# drop duplicated records, retain only one copy for each
df = pd.DataFrame.drop_duplicates(df)
df.shape
```

[8]:
```
(9977, 13)
```

now we can drop dummy data (the data we don't need)

[9]:
```
# drop Country & Postal Code
df = df.drop(['Postal Code'], axis = 1)
df = df.drop(['Country'], axis = 1)
```

A correlation heatmap is used to list all the correlation coefficients in order to identify multi-collinearity, in other words high intercorrelation above an absolute value of 0.5 between the a pair of attributes. For a pair of attributes with multicollinearity, one of them will be dropped since it would be redudant to include both of them with almost mirroring values. Another reason is to prevent overfitting.

[10]:
```
# compare linear relationships between attributes using correlation coefficient␣
 ↪generated using
# correlation matrix
sns.heatmap(df.corr(), cmap = 'PuBu', annot = True)
plt.show()
```

2 Exploratory Data Analysis (EDA) Retail Analysis

EDA aims to perform initial investigations on data before formal modeling and graphical representations and visualisations, in order to discover patterns, look over assumptions, and test hypothesis. The summarised information on main characteristics and hidden trends in data can help the Superstore to identify concern areas and problems, and the resolution of these can boost their profits.

First, the summary statistics will be considered.

[11]:
```
df.describe()
```

[11]:

	Sales	Quantity	Discount	Profit
count	9977.000000	9977.000000	9977.000000	9977.00000
mean	230.148902	3.790719	0.156278	28.69013
std	623.721409	2.226657	0.206455	234.45784
min	0.444000	1.000000	0.000000	-6599.97800
25%	17.300000	2.000000	0.000000	1.72620
50%	54.816000	3.000000	0.200000	8.67100
75%	209.970000	5.000000	0.200000	29.37200
max	22638.480000	14.000000	0.800000	8399.97600

[12]:
```
# total Sales
round(sum(df['Sales']), 2)
```

[12]:
```
2296195.59
```

[13]:
```
# total Quantity sold
sum(df['Quantity'])
```

[13]: 37820

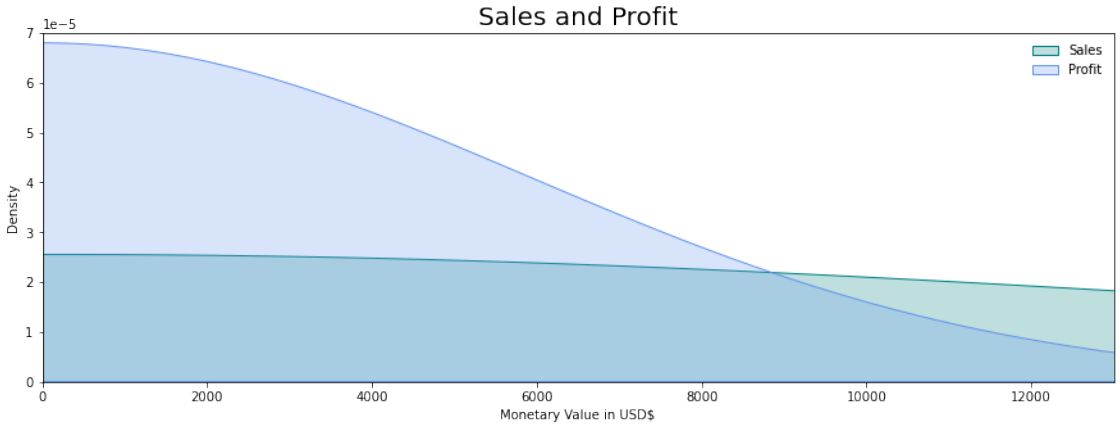
[14]:
```
# total Profit
round(sum(df['Profit']), 2)
```

[14]: 286241.42

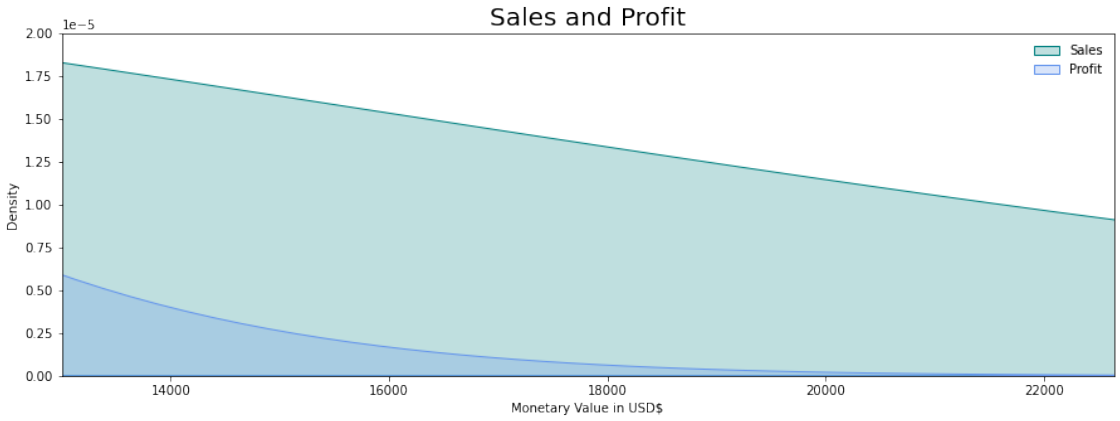
```
Total sales value was USD$2296195.59, and USD\$230.15 on average for each transaction. This ca

Total quantity sold was 37,820, and 4 on average for each transaction. This can range from 1 t

Average discount was 16% for each transaction. This can range from no discount to a notable hi

Overall, the Superstore made a considerably good profit of USD$28,6241.42, and USD\$28.69 on a
```

Diving deeper into sales and profit data, these are compared through density plotting to visualise data frequency for overall data distribution. Histogram density plots are chosen to illustrate the overall data distribution, as well as the data distributions of Sales and Profit. Kernel density estimate (KDE) plots, using the kde() method, will visualise the overall distribution through a continuous probability density curve. This generates two histogram density plots and their continuous probability density curves for Sales and Profit in the same figure space, and clearly differentiates them by specifying different colours in its parameters.

[20]:
```
plt.figure(figsize = (15, 5))
# plot Sales and Profit for comparisons
sns.kdeplot(df['Sales'], color = 'Teal', label = 'Sales', shade = True, bw = 25)
sns.kdeplot(df['Profit'], color = 'Cornflowerblue', label = 'Profit', shade =␣
 ↪True, bw = 25)
plt.xlim([0, 13000])
plt.ylim([0, 0.00007])
plt.ylabel('Density')
plt.xlabel('Monetary Value in USD$')
plt.title('Sales and Profit', fontsize = 20)
plt.legend(loc = 'upper right', frameon = False)
plt.show()
```



[21]:
```
plt.figure(figsize = (15, 5))
# plot Sales and Profit for comparisons
sns.kdeplot(df['Sales'], color = 'Teal', label = 'Sales', shade = True, bw = 25)
sns.kdeplot(df['Profit'], color = 'Cornflowerblue', label = 'Profit', shade =␣
↪True, bw = 25)
plt.xlim([13000, 22640])
plt.ylim([0, 0.00002])
plt.ylabel('Density')
plt.xlabel('Monetary Value in USD$')
plt.title('Sales and Profit', fontsize = 20)
plt.legend(loc = 'upper right', frameon = False)
plt.show()
```



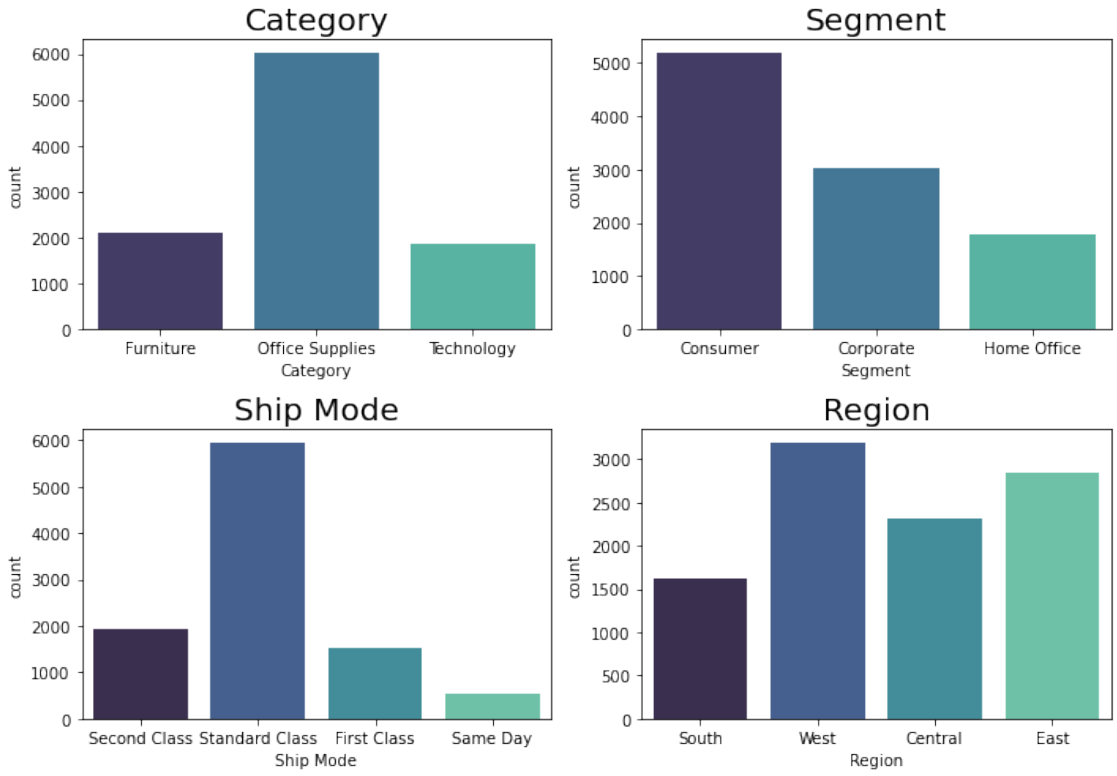
Profits are mostly above sales, indicating good business. However, there are some instances wh

The histogram density plots' highest points in the curves show the pattern of more sales transa

Taking a closer look at the individual categories of important qualitative attributes, as well as their

frequency of occurences :

[23]:
```
# count of each Category, Segment, Ship Mode, and Region
fig, axs = plt.subplots(nrows = 2, ncols = 2, figsize=(10, 7));
sns.countplot(df['Category'], ax = axs[0][0], palette = 'mako')
sns.countplot(df['Segment'], ax = axs[0][1], palette = 'mako')
sns.countplot(df['Ship Mode'], ax = axs[1][0], palette = 'mako')
sns.countplot(df['Region'], ax = axs[1][1], palette = 'mako')
axs[0][0].set_title('Category', fontsize = 20)
axs[0][1].set_title('Segment', fontsize = 20)
axs[1][0].set_title('Ship Mode', fontsize = 20)
axs[1][1].set_title('Region', fontsize = 20)
plt.tight_layout()
```



This clearly illustrates that data available for “Office Supplies” has almost 3 times the proportion than that for the other two categories, which will be taken note of for further data visualisations and analysis later on. This clearly illustrates that data available for “Consumer” is the sum of that of other two categories, which will be taken note of for further data visualisations and analysis later on. This clearly illustrates that data available for “Standard Class” has almost 3 times the proportion than that for “Second Class” and “First Class” categories, and 12 times that for “Same Day”. This clearly illustrates that data available for all 4 categories are differing, and this will be taken note of for further data visualisations and analysis later on. This generates two histogram density plots and their continuous probability density curves for Sales and Profit in the same figure

space, and clearly differentiates them by specifying different colours in its parameters.

Moving on, scatter plot allows detailed observation of the overall spread and relationships between Sales and Profit for all transactions.

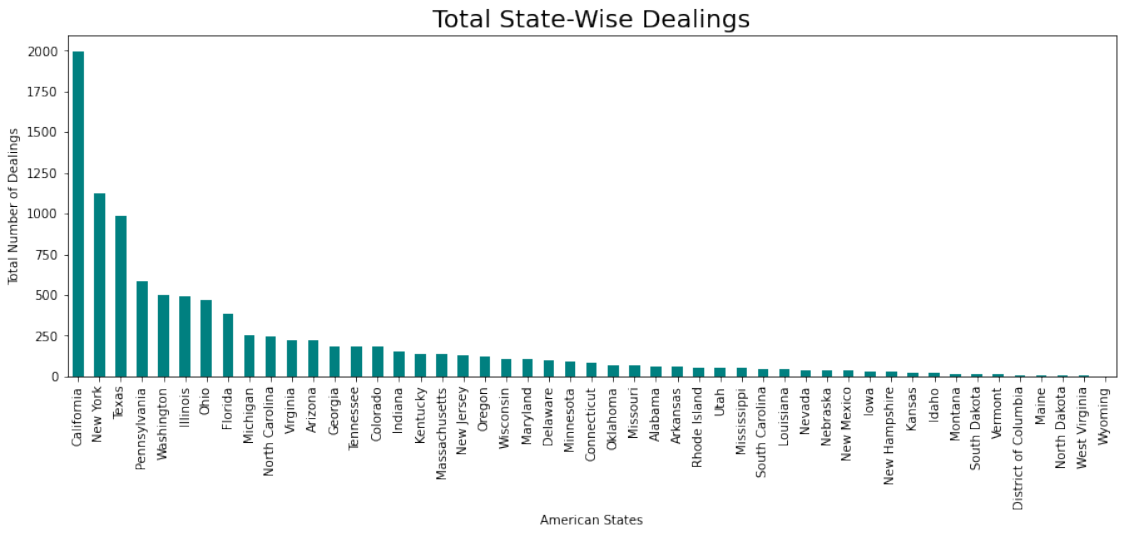
[24]:
```
fig, ax = plt.subplots(figsize = (10, 6))
# scatterplot of Sales and Profit
ax.scatter(df["Sales"] , df["Profit"], color = 'Teal')
ax.set_xlabel('Sales in USD$')
ax.set_ylabel('Profit/Loss in USD$')
plt.title('Sales and Profit', fontsize = 20)
plt.show()
```

Most sales are less than USD$5k. A significant number of transactions under USD$2.5k result in a loss. One likely reason behind smaller transaction amounts is after accounting for higher discount deductions, where this will decrease overall profits and can even cause a loss. Larger sales above USD$2.5k are very likely to result in a profit. Profit margins may be higher, after economies of scale in cost components such as procurement, packaging, and delivery.

A state-wise analysis is carried out below.

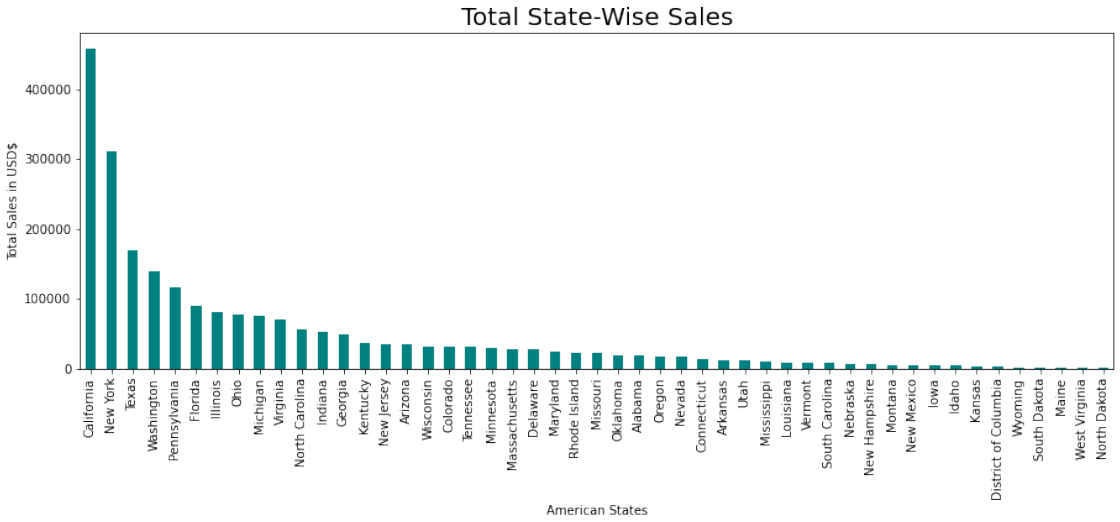
[28]:
```
# total dealings for each State
df_state_dealings = df.groupby('State')['Quantity'].count().
↪sort_values(ascending = False).plot.bar(figsize = (15, 5),
                                                                    ␣
↪                    color = 'Teal')
```

```
plt.ylabel('Total Number of Dealings')
plt.xlabel('American States')
plt.title('Total State-Wise Dealings', fontsize = 20)
plt.show()
```



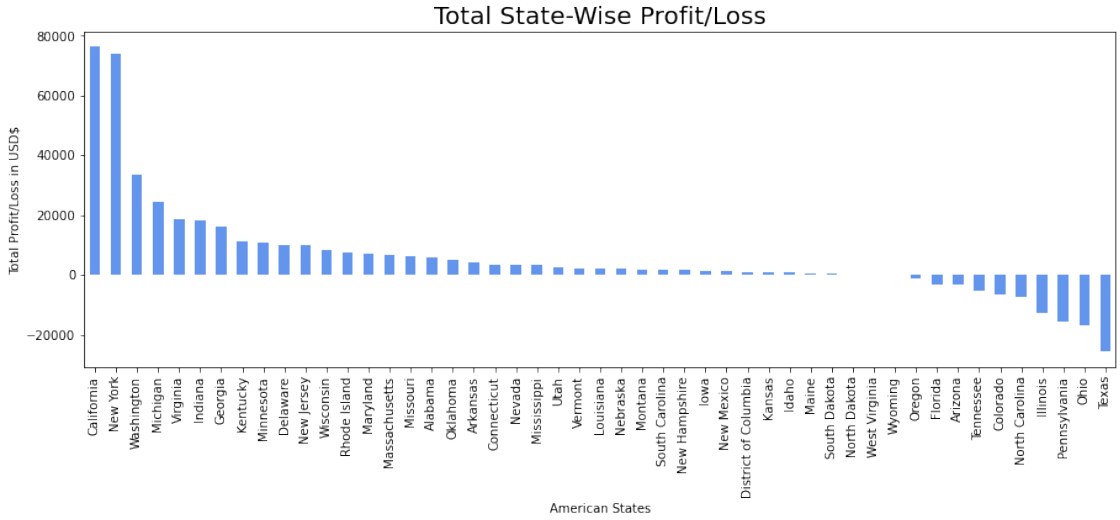
Superstore has the notable highest dealings in the state of California, with almost 2K of total dealings. With a big trailing gap, New York has the second highest dealings, with around 1,125 of total dealings. Texas is third with almost 1K of total dealings. Even if the Superstore outlets here are newly opened, marketing strategies should be improved in these areas as well as the states with less than 100 total dealings.

[27]:
```
# total Sales for each State
df_state_sales = df.groupby('State')['Sales'].sum().sort_values(ascending =␣
↪False).plot.bar(figsize = (15, 5),
                                                                              ␣
↪             color = 'Teal')
plt.ylabel('Total Sales in USD$')
plt.xlabel('American States')
plt.title('Total State-Wise Sales', fontsize = 20)
plt.show()
```

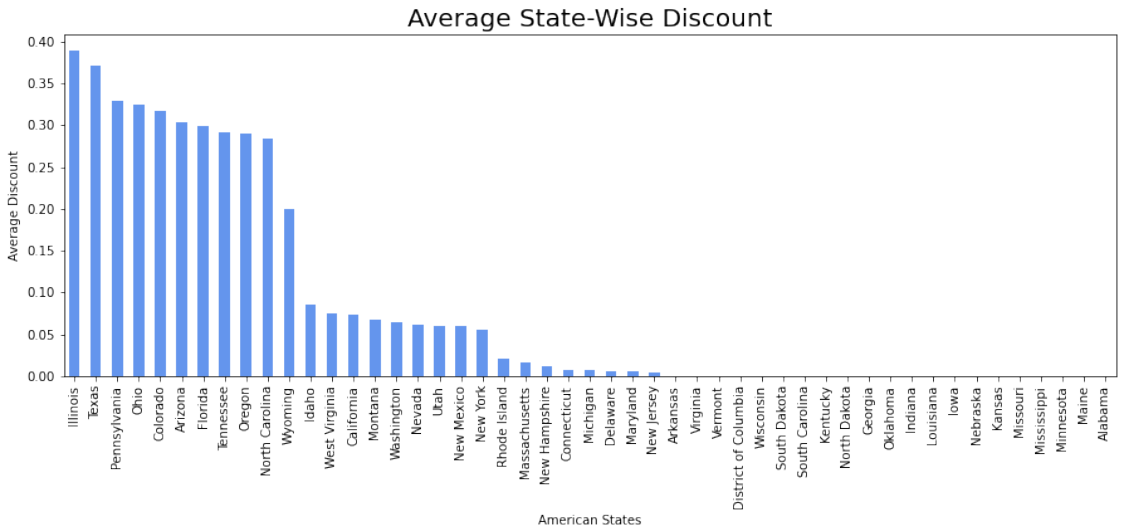


"The top 3 states here is same as for the previous analysis on number of dealings. Superstore has a notable highest sales in the state of California, with over USD$ 450K of total sales. With a big trailing gap, New York has the second highest sales, with over USD300$k of total sales. With another big trailing gap, Texas is third with around USD$ 170K of total sales. Even if the Superstore outlets here are newly opened, marketing strategies should be improved in these areas as well as the states with less than USD$20k total sales."

[29]:
```
# total Profit for each State
df_state_profit = df.groupby('State')['Profit'].sum().sort_values(ascending =␣
↪False).plot.bar(figsize = (15, 5),

↪                color = 'Cornflowerblue')
plt.ylabel('Total Profit/Loss in USD$')
plt.xlabel('American States')
plt.title('Total State-Wise Profit/Loss', fontsize = 20)
plt.show()
```



[30]:
```
# average Discount for each State
df_state_profit = df.groupby('State')['Discount'].mean().sort_values(ascending
 ↪= False).plot.bar(figsize = (15, 5),
                                                                              
 ↪                  color = 'Cornflowerblue')
plt.ylabel('Average Discount')
plt.xlabel('American States')
plt.title('Average State-Wise Discount', fontsize = 20)
plt.show()
```

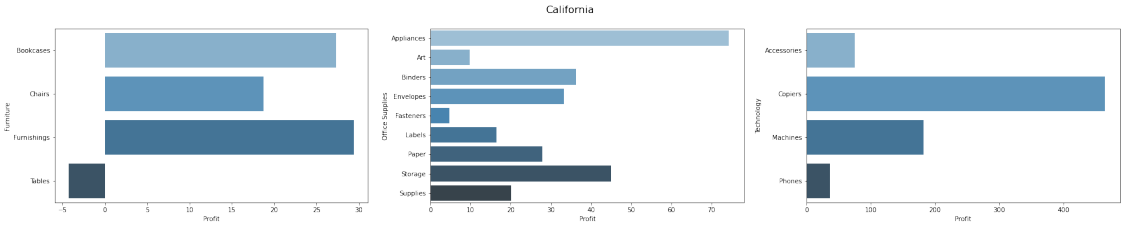


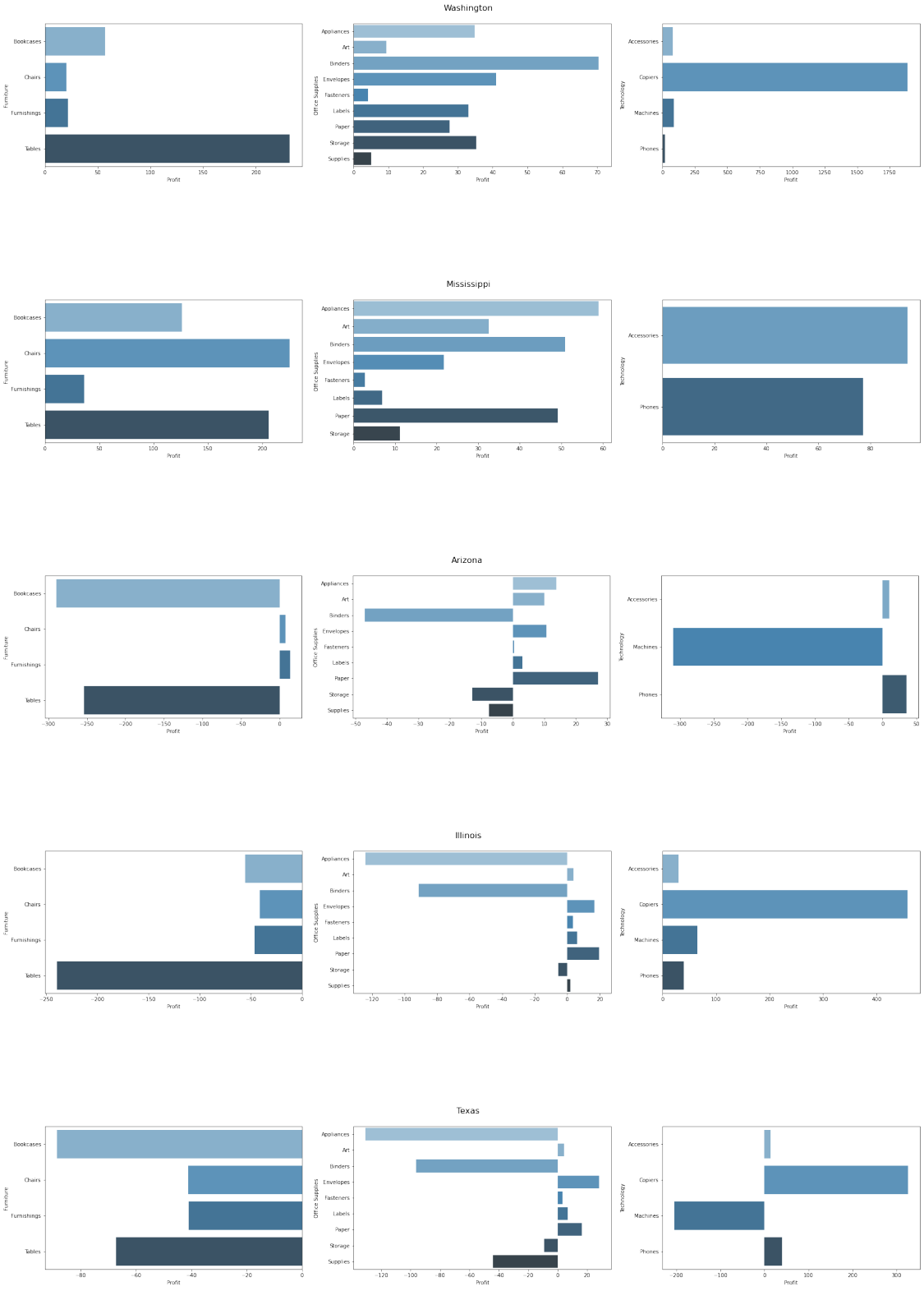
Superstore is advised to reduce discount levels in Texas, and instead switch to other promotional

strategies, in order to minimise losses. Pennsylvania, Illinois, and Ohio are the third, first, and fourth state offering larger discounts respectively, and this may be the reason behind them resulting in the third, fourth, and second biggest loss of around USD$15k. California gives out considerably lower discounts, which may be one of the main reasons behind it being top in sales and profits by a significant margin. This is indicative that the promotional strategy of offering slight discounts is highly effective in the state of California.

Which products are popular in profit-making states? Which products are commonly purchased in loss-bearing states? Which product categories and sub-categories can be improved in order to increase profits? reasons behind it being top in sales and profits by a significant margin. This is indicative that the promotional strategy of offering slight discounts is highly effective in the state of California. More than half the states make little to no profit, and a significant number of these even suffer from a notable loss. A majority of states offer slight discounts under 10%

[32]:
```
def state_data_viewer(states):
# plot profit of product categories and sub-categories for the chosen states
    product_data = df.groupby(['State'])
    for state in states:
        data = product_data.get_group(state).groupby(['Category'])
        fig, ax = plt.subplots(1, 3, figsize = (30, 5))
        fig.suptitle(state, fontsize = 16)
        ax_index = 0
        # plot a chart for each category
        for category in ['Furniture', 'Office Supplies', 'Technology']:
            # plot sub-categories in each category
            category_data = data.get_group(category).groupby(['Sub-Category']).
 ↪mean()
            sns.barplot(x = category_data.Profit, y = category_data.index,
                        ax = ax[ax_index], palette = 'Blues_d')
            ax[ax_index].set_ylabel(category)
            ax_index += 1
# chosen States based on profit/loss categories
states = ['California', 'Washington', 'Mississippi', 'Arizona', 'Illinois',␣
 ↪'Texas']
state_data_viewer(states)
```

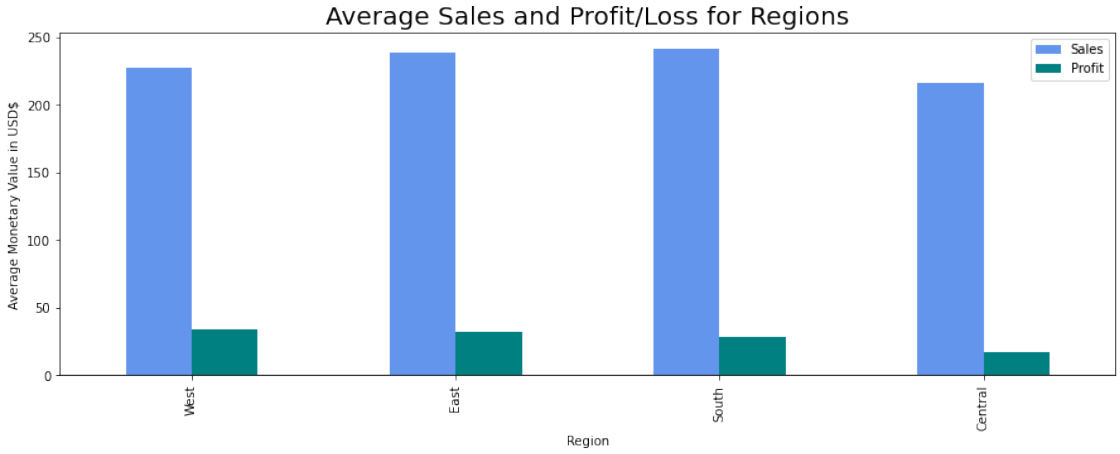




In high profit states such as California, all products sold across all categories and sub-categories are

generating comparatively good profits, except for Tables in Furniture category which is suffering from an overall loss of USD$4 per unit sold. In low profit states such as Mississippi, all products sold across all categories and sub-categories are generating good profits. Superstore is advised to make marketing strategies one of the main focuses to ensure increasing profitable sales and consistent future gains in long term customers for these 2 sub-categories. In low loss states such as Arizona, most sub-categories generate little to no profits, and those that suffer from up to USD$300 of notable losses are Bookcases, Tables, Binders, and Machines. Superstore is advised to make marketing strategies one of the main focuses to ensure increasing profitable sales and consistent future gains in long term customers for Copiers. In medium loss states such as Illinois, most sub-categories generate little to no profits, and those that suffer from up to USD$240 of notable losses are Tables, Appliances, and Binders. The Technology category has the best performance with Copiers making the highest profit of around USD$450 profit per unit sold. In high loss states such as Texas, the Technology category has the best performance with Copiers making the highest profit of around USD$325 profit per unit sold, while the Furniture category's performance is going entirely in loss. The overall observed negative trend is that all Furniture sub-categories as well as Appliances and Binders under the Office Supplies category contribute to a majority of losses.

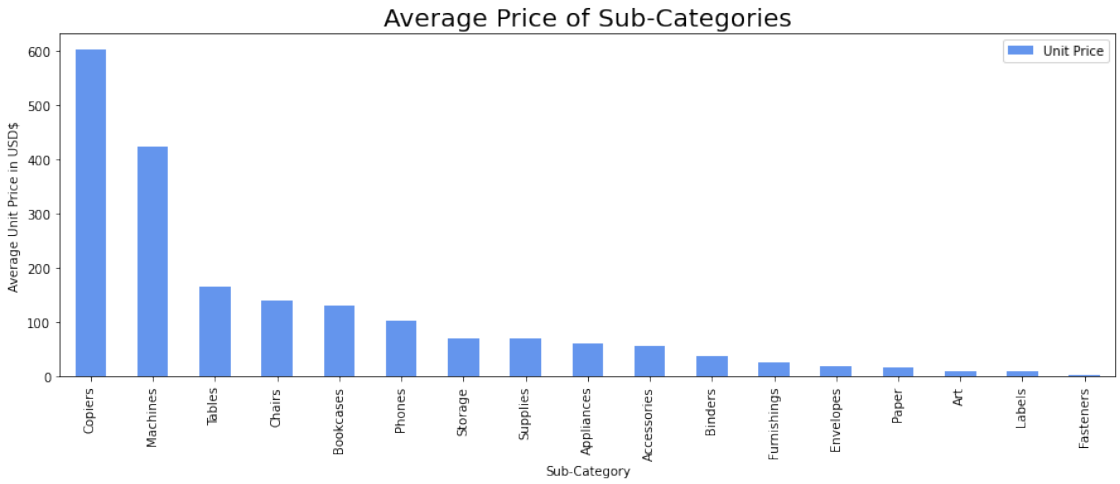
[33]:
```
# average Sales and profit/loss for Region
colors = ['Cornflowerblue', 'Teal']
df_region = df.groupby(['Region'])[['Sales', 'Discount', 'Profit']].mean()
df_region.sort_values('Profit', ascending = False)[['Sales', 'Profit']].
 ↪plot(kind = 'bar',
                                                                        ␣
 ↪figsize = (15, 5),
                                                                        ␣
 ↪color = colors)

plt.ylabel('Average Monetary Value in USD$')
plt.xlabel('Region')
plt.title('Average Sales and Profit/Loss for Regions', fontsize = 20)
plt.show()
```



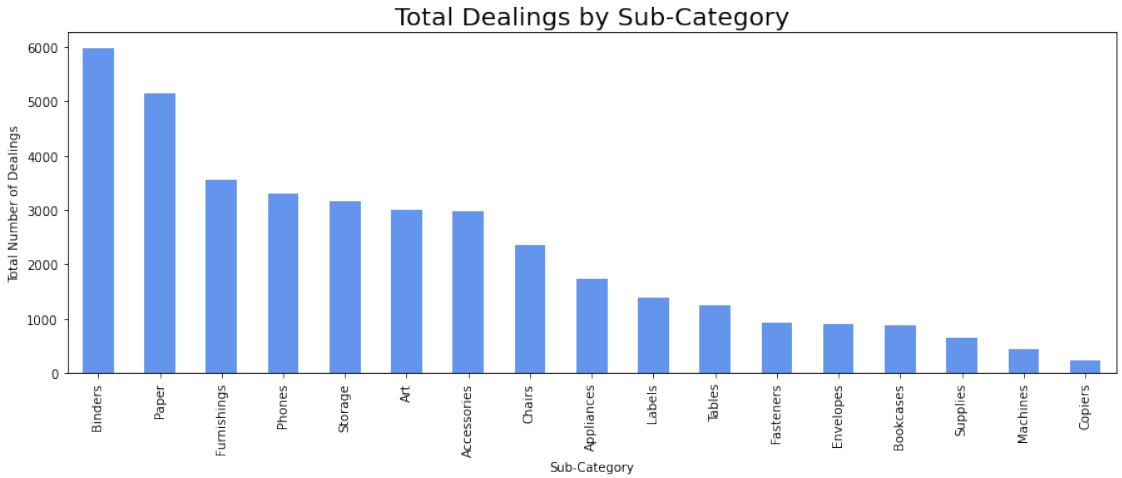
For Superstore outlets that are newly opened, marketing strategies should be one of the main focuses to ensure increasing profitable sales and consistent future gains in long term customers. Moving on to analyse the effect of customer segments on sales and profit : While the Consumer customer segment made up almost 51% of total sales, the profits contributed was only 47%. This is indicative that although there is good targeting strategy, the Superstore is bearing a slight loss in the Consumer segment due to lower profit margins. This is indicative that Superstore is conducting good target strategy with these segments at appropriate profit margins.

Diving deeper into product Categories and Sub-Categories, the spreads and trends of prices, quantities, sales, and profits are analysed.

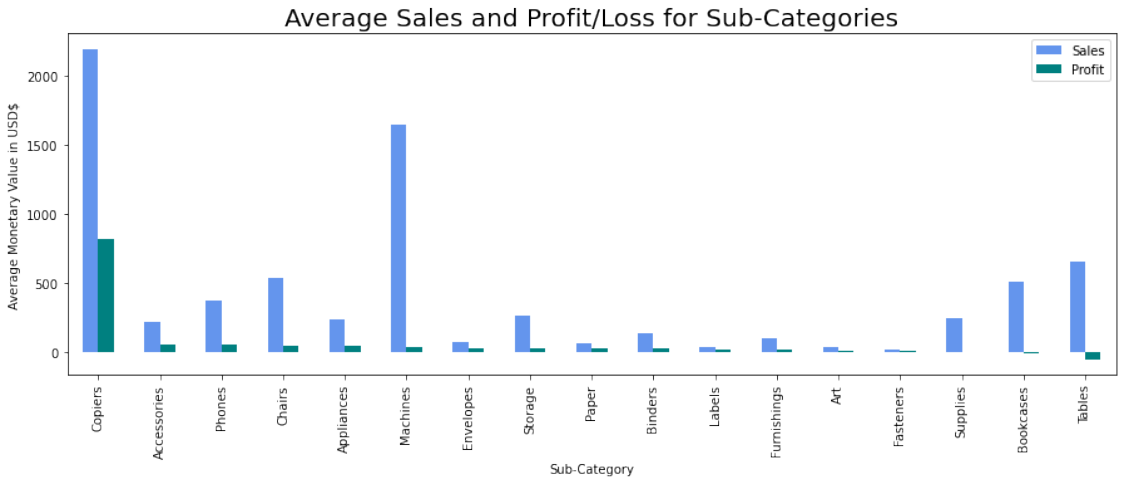
[35]:
```
# Unit Price calculation
df['Unit Price'] = df.Sales / df.Quantity

# Unit Profit calculation
df['Unit Profit'] = df.Profit / df.Quantity

# Unit Price of Sub-Category
df_subcategory = df.groupby(['Sub-Category'])[['Sales', 'Discount', 'Profit',␣
↪'Unit Price', 'Unit Profit']].mean()
df_subcategory.sort_values('Unit Price', ascending = False)[['Unit Price']].
↪plot(kind = 'bar',

↪ figsize = (15, 5),

↪ color = 'Cornflowerblue')
plt.ylabel('Average Unit Price in USD$')
plt.xlabel('Sub-Category')
plt.title('Average Price of Sub-Categories', fontsize = 20)
plt.show()
```



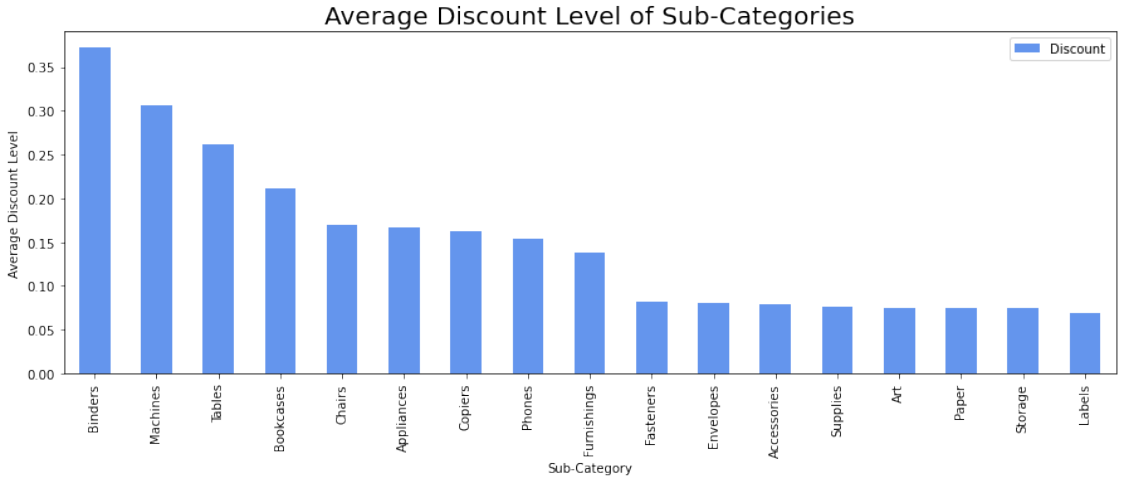
[36]:
```
# total dealings for each Sub-Category
df_state_dealings = df.groupby('Sub-Category')['Quantity'].sum().
 ↪sort_values(ascending = False).plot.bar(figsize = (15, 5),

 ↪                          color = 'Cornflowerblue')
plt.ylabel('Total Number of Dealings')
plt.xlabel('Sub-Category')
plt.title('Total Dealings by Sub-Category', fontsize = 20)
plt.show()
```



[37]:
```
colors = ['Cornflowerblue', 'Teal']
# Average Sales and profit/loss for Sub-Category
df_subcategory.sort_values('Profit', ascending = False)[['Sales', 'Profit']].
 ↪plot(kind = 'bar',

 ↪  figsize = (15, 5),

 ↪  color = colors)

plt.ylabel('Average Monetary Value in USD$')
plt.xlabel('Sub-Category')
plt.title('Average Sales and Profit/Loss for Sub-Categories', fontsize = 20)
plt.show()
```



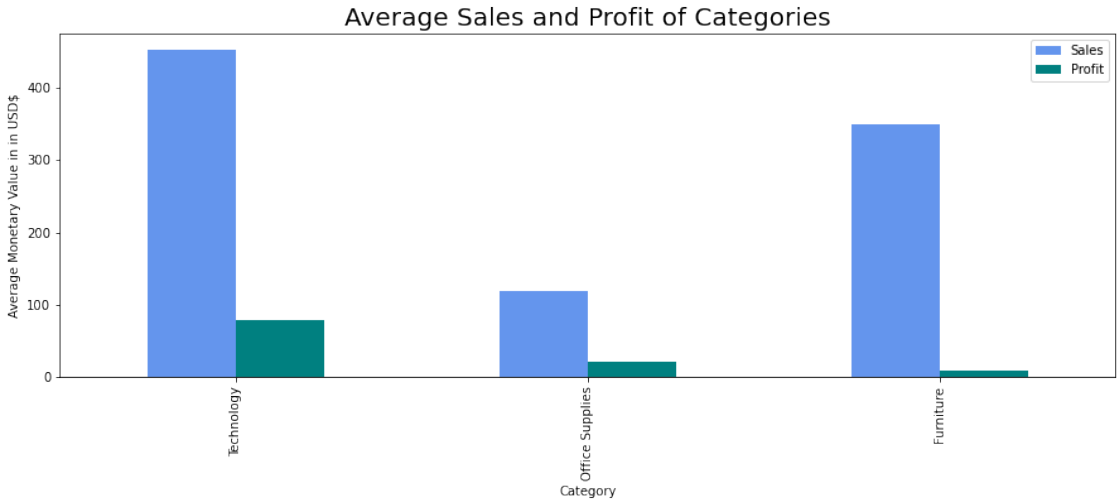
[38]:
```
# Discount of Sub-Category
df_subcategory.sort_values('Discount', ascending = False)[['Discount']].
 ↪plot(kind = 'bar',
                                                                        ␣
 ↪figsize = (15, 5),
                                                                        ␣
 ↪color = 'Cornflowerblue')
plt.ylabel('Average Discount Level')
plt.xlabel('Sub-Category')
plt.title('Average Discount Level of Sub-Categories', fontsize = 20)
plt.show()
```



The overall performance of Chairs is observed to generate the highest profits from its high sales, despite giving out relatively high average discounts of almost 17.5%. Furnitures category has the

highest average discounts, which lowers its profitability in terms of both profits and profit margins. Binders have the highest average discounts of around 37.5%, which lowers its profitability in terms of both profits and profit margins. Overall, all sub-categories have good profit margins. The Office Supplies Category has the lowest discounts. This Category has the highest profitability in terms of profit margins, which is likely due to low average discounts. Machine has the lowest profit margin as its notably high sales only bring in almost negligible profits. Phone and Accessories have high profits despite lower sales, as compared to similar profits gained from Chairs, Appliances, and Machines. Profit margins are good. This Category has the highest overall sales and profits

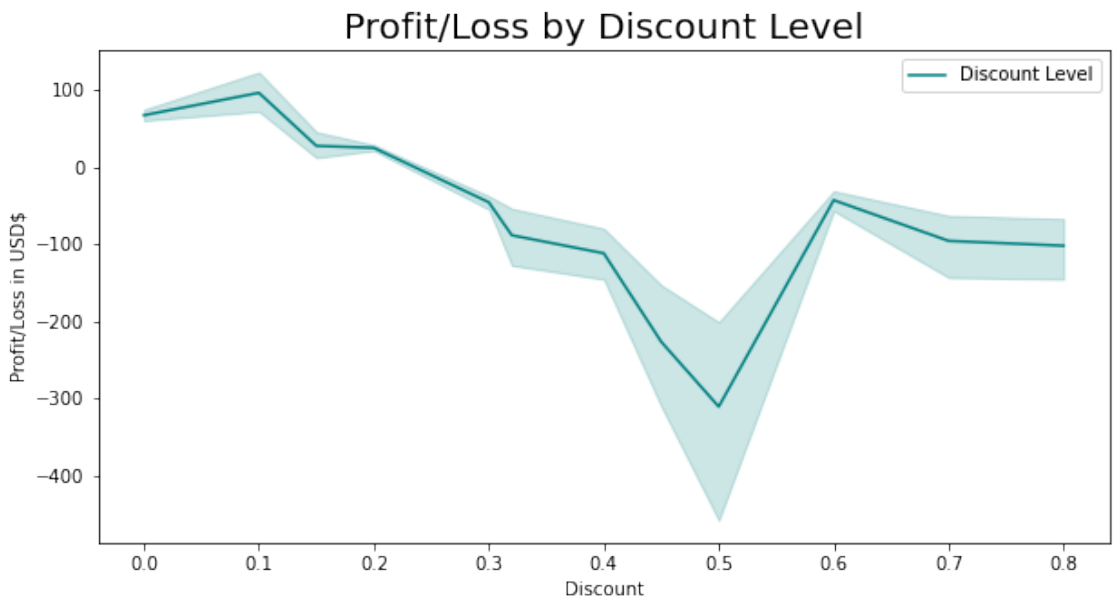
[39]:
```
# average Sales and Profit of Category
df_category = df.groupby(['Category'])[['Sales', 'Discount', 'Profit', 'Unit␣
 ↪Price', 'Unit Profit']].mean()
df_category.sort_values('Profit', ascending = False)[["Sales", "Profit"]].
 ↪plot(kind = 'bar',
                                                                          ␣
 ↪figsize = (15, 5),
                                                                          ␣
 ↪color = colors)

plt.ylabel('Average Monetary Value in in USD$')
plt.xlabel('Category')
plt.title('Average Sales and Profit of Categories', fontsize = 20)
plt.show()
```



The final EDA will observe the effect of discounts on profit/loss.

[40]:
```
plt.figure(figsize = (10, 5))
# profit/loss by Discount level
sns.lineplot('Discount', 'Profit', data = df, color = 'Teal', label = 'Discount␣
 ↪Level')
```

```
plt.ylabel('Profit/Loss in USD$')
plt.title('Profit/Loss by Discount Level', fontsize = 20)
plt.show()
```



Losses will likely occur for higher discount levels above 20%. In other words, between 0% and 1

The worst losses occured when discount approximates 50%. This may indicate festivals, end-of-s

Superstore is most profitable when discount levels lower than 10% are offered. This allows less

[42]:
```
#transactions with Discount
df_discounted = df[df['Discount'] > 0]

# transactions without Discount
df_no_discount = df[df['Discount'] == 0]

print ("On Average :")

print("Discounted Sales per transaction is USD$", round(df_discounted['Sales'].
 ↪mean(), 2),
      ", without is USD$", round(df_no_discount['Sales'].mean(), 2))

print("Discounted Unit Price is USD$", round(df_discounted['Unit Price'].
 ↪mean(), 2),
      ", without is USD$", round(df_no_discount['Unit Price'].mean(), 2))
```

```
print("Discounted Quantity purchased per transaction is ",␣
↪round(df_discounted['Quantity'].mean(), 2),
      ", without is ", round(df_no_discount['Quantity'].mean(), 2))

print("Discounted Profit per transaction is USD$",␣
↪round(df_discounted['Profit'].mean(), 2),
      ", without is USD$", round(df_no_discount['Profit'].mean(), 2))

print("Discounted Unit Profit is USD$", round(df_discounted['Unit Profit'].
↪mean(), 2),
      ", without is USD$", round(df_no_discount['Unit Profit'].mean(), 2))

print(" ")

print ("In Total :")

print("Discounted Total Sales is USD$", round(df_discounted['Sales'].sum(), 2),
      ", without is USD$", round(df_no_discount['Sales'].sum(), 2))

print("Discounted Total Quantity is ", round(df_discounted['Quantity'].sum(),␣
↪2),
      ", without is ", round(df_no_discount['Quantity'].sum(), 2))

print("Discounted Total Profit is USD$", round(df_discounted['Profit'].sum(),␣
↪2),
      ", without is USD$", round(df_no_discount['Profit'].sum(), 2))
```

```
On Average :
Discounted Sales per transaction is USD$ 232.93 , without is USD$ 227.13
Discounted Unit Price is USD$ 62.82 , without is USD$ 59.0
Discounted Quantity purchased per transaction is  3.77 , without is  3.81
Discounted Profit per transaction is USD$ -6.67 , without is USD$ 67.02
Discounted Unit Profit is USD$ -1.23 , without is USD$ 17.61

In Total :
Discounted Total Sales is USD$ 1208918.03 , without is USD$ 1087277.56
Discounted Total Quantity is  19590 , without is  18230
Discounted Total Profit is USD$ -34602.98 , without is USD$ 320844.41
```

Considering both average and total sales, customers tend to spend more when there are discounts

The average price of a discounted product is USD$63 as compared to USD\$59 for a non-discounted

Considering both average and total sales quantities, customers tend to buy more products when t

However, discounts will affect profits. On average, the sale of a discounted product results in

Thank you!

[ ]: