TASK-3

Author: ISHWARYA D

THE SPARKS FOUNDATION INTERNSHIP

Exploratory Data Analysis - Retail

● To find out the weak areas where you can,work to make more profit. ● And to find What all business problems we can derive by exploring the data.

EXPLORING THE DATASET

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from statsmodels.formula.api import ols
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
```

In [2]:
```
data = pd.read_csv('SampleSuperstore.csv')
data.head(5)
```

Out[2]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quantity	Discount	
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.9600	2	0.00	4
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.9400	3	0.00	21
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.6200	2	0.00	
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.5775	5	0.45	-38
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.3680	2	0.20	

In [3]:
```
data.describe()
```

Out[3]:

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

In [4]:
```
data.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
```

In [4]:
```
data['Sales'].mean()
```

Out[4]:
```
229.8580008304938
```

In [5]:
```
data['Sales'].quantile(.25)
```

Out[5]:
```
17.28
```

In [6]:
```
data['Quantity'].unique()
```

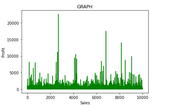
Out[6]:
```
array([ 2,  3,  5,  7,  4,  6,  9,  1,  8, 14, 11, 13, 10, 12],
      dtype=int64)
```

In [7]:
```
np.std(data['Sales'],ddof=1)
```

Out[7]:
```
623.2451005086818
```

PLOTTING THE GRAPH

In [8]:
```
plt.plot(data['Sales'],color='green')
plt.xlabel('Sales')
plt.ylabel('Profit')
plt.title("GRAPH")
plt.show()
```



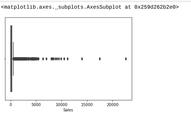
In [9]:
```
zip_condition_data=data.groupby(['Sales','Profit'])['Discount'].mean()
zip_condition_data
```

Out[9]:
```
Sales     Profit    
0.444     -1.1100       0.8
0.556     -0.9452       0.8
0.836     -1.3376       0.8
0.852     -0.3408       0.8
0.876     -1.4016       0.8
                       ... 
10499.970 5039.9856     0.0
11199.968 3919.9888     0.2
13999.960 6719.9808     0.0
17499.950 8399.9760     0.0
22638.480 1811.0784     0.5
Name: Discount, Length: 7657, dtype: float64
```

BOXPLOT FOR SALES

In [10]:
```
sns.boxplot(x=data['Sales'])
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x259d262b2e0>
```



Interquartile range for Sales

In [11]:
```
q1=data['Sales'].quantile(.25)
q3=data['Sales'].quantile(.75)
```

In [12]:
```
iqr=q3-q1
iqr
```

Out[12]:
```
192.66
```

In [13]:
```
upper_limit=q3 +1.5*iqr
lower_limit=q1 -1.5*iqr
upper_limit,lower_limit
```

Out[13]:
```
(490.93, -271.7100000000000004)
```

In [14]:
```
data.dropna(inplace=True,axis=0,subset=['Sales'])
```

In [16]:
```
numerical_columns=['Ship Mode','Segment','Country','City','State','Postal Code','Region','Category','Sub-Category','Sales','Quantity','Discount','Profit']
```

In [17]:
```
column=data["Postal Code"].values.reshape(-1,1)
column.shape
```

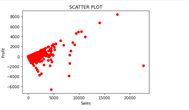
Out[17]:
```
(9994, 1)
```

In [18]:
```
column=data["Postal Code"].values.reshape(-1,1)
imputer=SimpleImputer(missing_values =np.nan, strategy='most_frequent')
data["Postal Code"]=imputer.fit_transform(column)
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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   object 
 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: 1.1+ MB
```

SALES VS PROFIT

In [19]:
```
plt.scatter(x=data['Sales'],y=data['Profit'],COLOR='RED')
plt.ylabel("Profit")
plt.xlabel("Sales")
plt.title("SCATTER PLOT")
plt.show()
```



ANOVA TABLE

In [20]:
```
mod=ols('Sales ~ Profit',data=data).fit()
Anova_Table=sm.stats.anova_lm(mod, typ = 2)
print(Anova_Table)
```

```
                sum_sq      df           F  PR(>F)
Profit    8.908433e+08     1.0  2976.247088     0.0
Residual  2.990782e+09  9992.0         NaN     NaN
```

DATA BINNING

In [21]:
```
Zip_Table=data.groupby('Postal Code').agg({'Sales':'mean'}).sort_values('Sales',ascending=True)
Zip_Table.head()
```

Out[21]:

	Sales
Postal Code	
79605	1.392
44035	1.824
33458	2.064
32503	2.214
32174	2.808

In [22]:
```
Zip_Table['Postal Code_group']=pd.cut(Zip_Table['Sales'],bins=10,
                                      labels=['Zipcode_group_0',
                                              'Zipcode_group_1',
                                              'Zipcode_group_2',
                                              'Zipcode_group_3',
                                              'Zipcode_group_4',
                                              'Zipcode_group_5',
                                              'Zipcode_group_6',
                                              'Zipcode_group_7',
                                              'Zipcode_group_8',
                                              'Zipcode_group_9'],
                                     include_lowest=True)
```

In [23]:
```
y=data.iloc[:,0]
x=data.iloc[:,1:31]
x.head(5)
```

Out[23]:

	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quantity	Discount	Profit
0	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.9600	2	0.00	41.9136
1	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.9400	3	0.00	219.5820
2	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.6200	2	0.00	6.8714
3	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.5775	5	0.45	-383.0310
4	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.3680	2	0.20	2.5164

TRAINING THE MODEL

In [24]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [25]:
```
x_train.shape
```

Out[25]:
```
(6995, 12)
```

In [26]:
```
x_test.shape
```

Out[26]:
```
(2999, 12)
```

ACTUAL SALES PRICE VS PROFIT

In [27]:
```
plt.figure(dpi=100)
k=range(0, len(data))
plt.scatter(k,data['Sales'].sort_values(),color='blue',label='Actual Sale Price')
plt.plot(k,data['Profit'].sort_values(),color='blue',label='Profit')
plt.xlabel('Fitted points (Ascending)')
plt.ylabel('Sales')
plt.title("overall mean")
plt.legend()
```

Out[27]:
```
<matplotlib.legend.Legend at 0x259d3790340>
```