1901202051.SuryajirajeBhosale.ML1S2

May 26, 2021

[185]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import copy
```

Loading Data:

[782]:
```
df = pd.read_excel('/Users/user/Desktop/Superstore.xls')
```

Displaying first few lines of the data:

[783]:
```
df.head()
```

[783]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	…	Postal Code	Region	Product ID	Category	Sub-Category	Product Name	Sales	Quantity
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	…	42420	South	FUR-BO-10001798	Furniture	Bookcases	Bush Somerset Collection Bookcase	261.9600	2
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	…	42420	South	FUR-CH-10000454	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,…	731.9400	3
2	3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	…	90036	West	OFF-LA-10000240	Office Supplies	Labels			
3	4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	…	33311	South	FUR-TA-10000577	Furniture	Tables			
4	5	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	…	33311	South	OFF-ST-10000760	Office Supplies	Storage			

```
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]
```

Basic descriptives and correlation matrix:

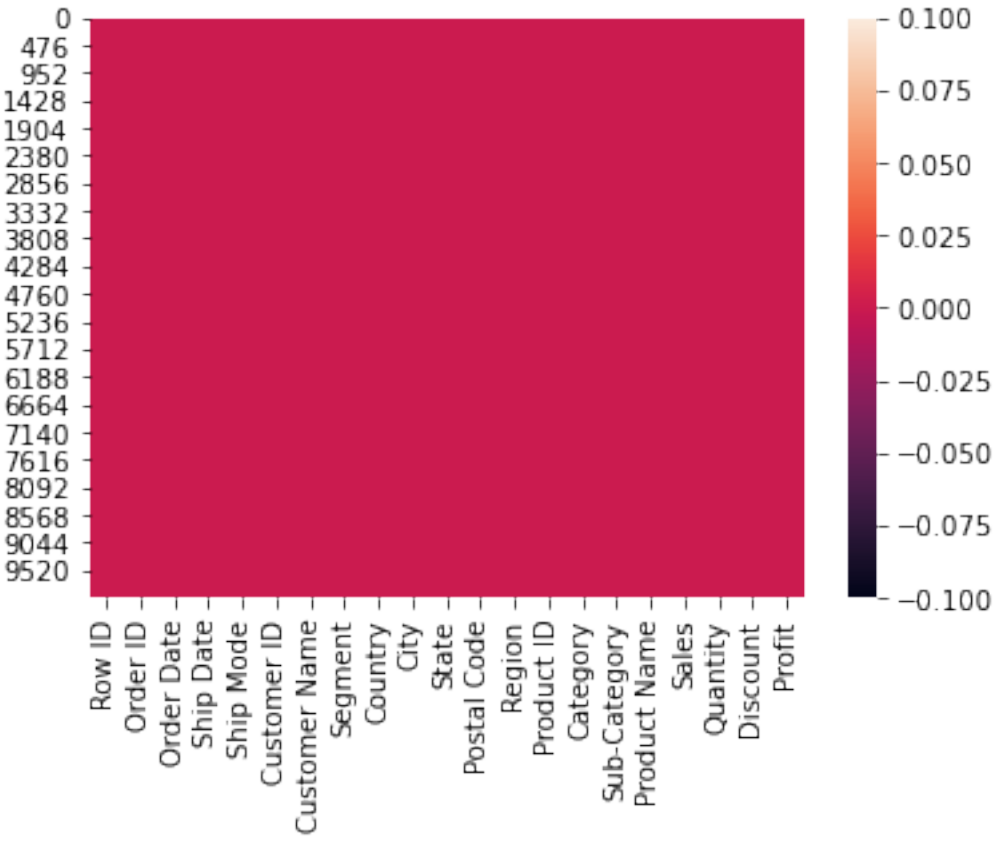
[784]:
```
print(df.describe(), '\n')
print(df.corr(), '\n')
```

```
            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

               Row ID  Postal Code     Sales  Quantity  Discount    Profit
Row ID       1.000000     0.009671 -0.001359 -0.004016  0.013480  0.012497
Postal Code  0.009671     1.000000 -0.023854  0.012761  0.058443 -0.029961
Sales       -0.001359    -0.023854  1.000000  0.200795 -0.028190  0.479064
Quantity    -0.004016     0.012761  0.200795  1.000000  0.008623  0.066253
Discount     0.013480     0.058443 -0.028190  0.008623  1.000000 -0.219487
Profit       0.012497    -0.029961  0.479064  0.066253 -0.219487  1.000000
```

Checking the presence of null values:

[785]:
```
sns.heatmap(df.isnull())
```

[785]: <AxesSubplot:>



Treating column names to delete spaces in between variable names:

[786]:
```
df.columns = df.columns.str.replace(' ', '')
df.head()
```

[786]:

	RowID	OrderID	OrderDate	ShipDate	ShipMode	CustomerID	\
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	

	CustomerName	Segment	Country	City	…	PostalCode	\
0	Claire Gute	Consumer	United States	Henderson	…	42420	
1	Claire Gute	Consumer	United States	Henderson	…	42420	
2	Darrin Van Huff	Corporate	United States	Los Angeles	…	90036	

```
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …      33311
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …      33311

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

                                          ProductName     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]
```

Grouping was initiated for varibales - Category & Subcategory:

[787]:
```
df_prof = copy.deepcopy(df.groupby(['Category','Sub-Category']).sum())
df_prof = df_prof.drop(['RowID','PostalCode'], axis = 1)
df_prof = df_prof.reset_index()
df_prof
```

[787]:

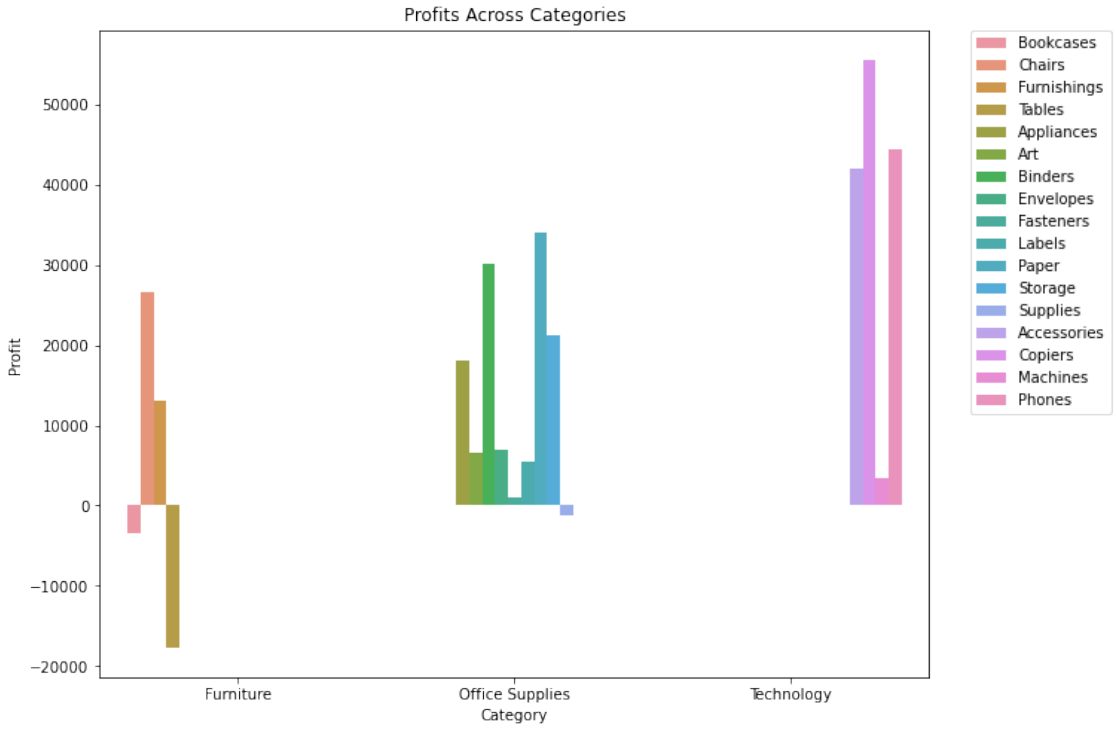
	Category	Sub-Category	Sales	Quantity	Discount	Profit
0	Furniture	Bookcases	114879.9963	868	48.14	-3472.5560
1	Furniture	Chairs	328449.1030	2356	105.00	26590.1663
2	Furniture	Furnishings	91705.1640	3563	132.40	13059.1436
3	Furniture	Tables	206965.5320	1241	83.35	-17725.4811
4	Office Supplies	Appliances	107532.1610	1729	77.60	18138.0054
5	Office Supplies	Art	27118.7920	3000	59.60	6527.7870
6	Office Supplies	Binders	203412.7330	5974	567.00	30221.7633
7	Office Supplies	Envelopes	16476.4020	906	20.40	6964.1767
8	Office Supplies	Fasteners	3024.2800	914	17.80	949.5182
9	Office Supplies	Labels	12486.3120	1400	25.00	5546.2540
10	Office Supplies	Paper	78479.2060	5178	102.60	34053.5693
11	Office Supplies	Storage	223843.6080	3158	63.20	21278.8264
12	Office Supplies	Supplies	46673.5380	647	14.60	-1189.0995

13	Technology	Accessories	167380.3180	2976	60.80	41936.6357
14	Technology	Copiers	149528.0300	234	11.00	55617.8249
15	Technology	Machines	189238.6310	440	35.20	3384.7569
16	Technology	Phones	330007.0540	3289	137.40	44515.7306

Graph 1: Profitability Of Each Item In Each Category

[788]:
```
fig_dims = (10, 8)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x="Category",y="Profit",hue="Sub-Category",data=df_prof,ax=ax)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0)
plt.title('Profits Across Categories')
```

[788]: Text(0.5, 1.0, 'Profits Across Categories')



Inference On Graph 1: The graph above depicts the profitabily across all at a atomic level. We can see the most profitable and the least profitable categories:

Most Profitable: Technology

Moderate Profitable: Office Supplies

Least Profitable: Furniture

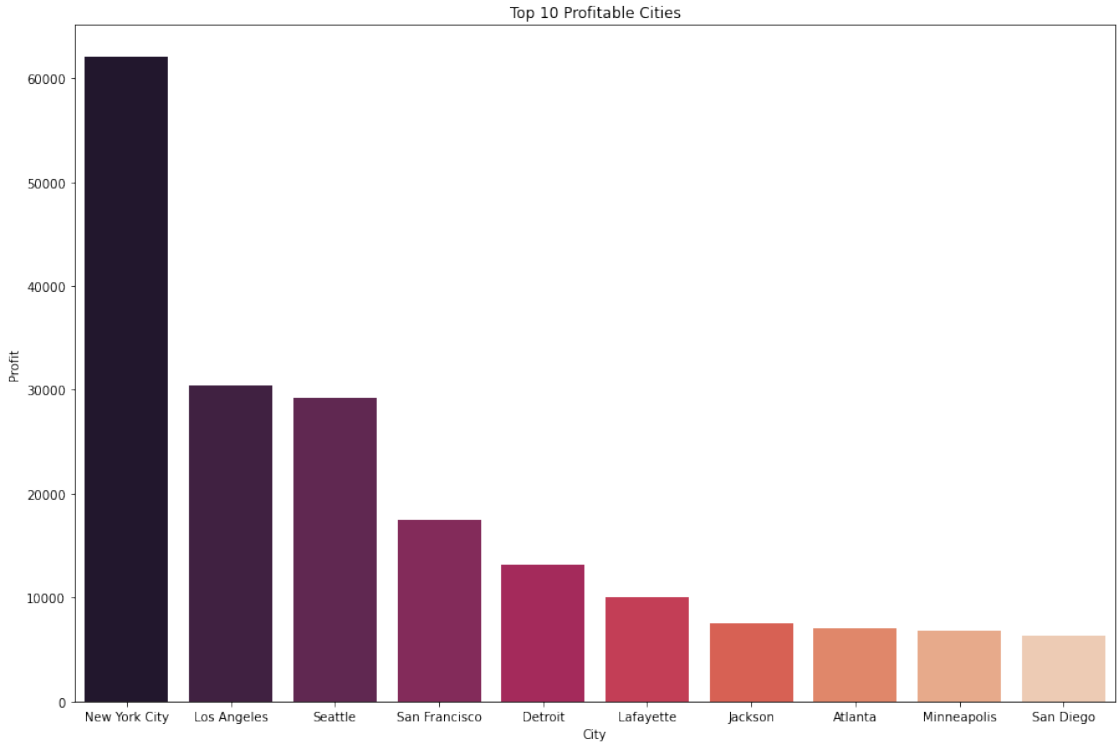
We conclude that the most profitable category being technology within which phones and copiers have been the most profitable across all subcategories. The commodity that shows negative profits

or a loss are tables. The office supplies category has shown stable profits with paper and binders posting the highest profits.

Graph 2: Top 10 Most Profitable Cities

[789]:
```
df_city = copy.deepcopy(df.groupby(['City'])).sum()
df_city = df_city.reset_index()
df_city_max = df_city.sort_values(by = ['Profit'], ascending = False)
df_city_max = df_city_max.iloc[:10,:]
fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x='City', y= 'Profit', data=df_city_max, ax=ax, palette='rocket')
plt.title('Top 10 Profitable Cities')
```

[789]:
```
Text(0.5, 1.0, 'Top 10 Profitable Cities')
```



Graph 3: Top 10 Cities With The Highest Sales (per Quantity):

[790]:
```
df_city = copy.deepcopy(df.groupby(['City'])).sum()
df_city = df_city.reset_index()
df_city_Sale_max = df_city.sort_values(by = ['Quantity'], ascending = False)
df_city_Sale_max = df_city_Sale_max.iloc[:10,:]
fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
```

```
sns.barplot(x='City', y= 'Quantity', data=df_city_Sale_max, ax=ax,␣
 ↪palette='mako')
plt.title('Top 10 Selling Cities')
```

[790]: Text(0.5, 1.0, 'Top 10 Selling Cities')

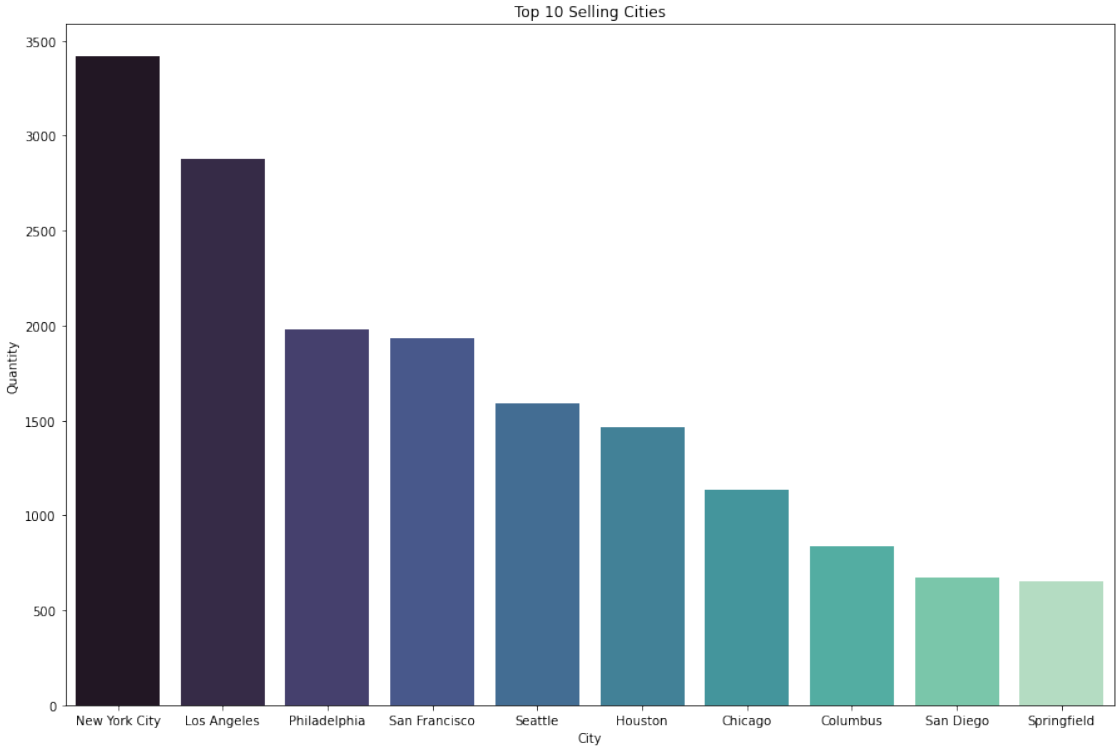


Table for Above Graph:

[791]:
```
df_city_Sale_max = df_city.sort_values(by = ['Quantity'], ascending = False)
df_city_Sale_max.iloc[:10,:]
```

[791]:

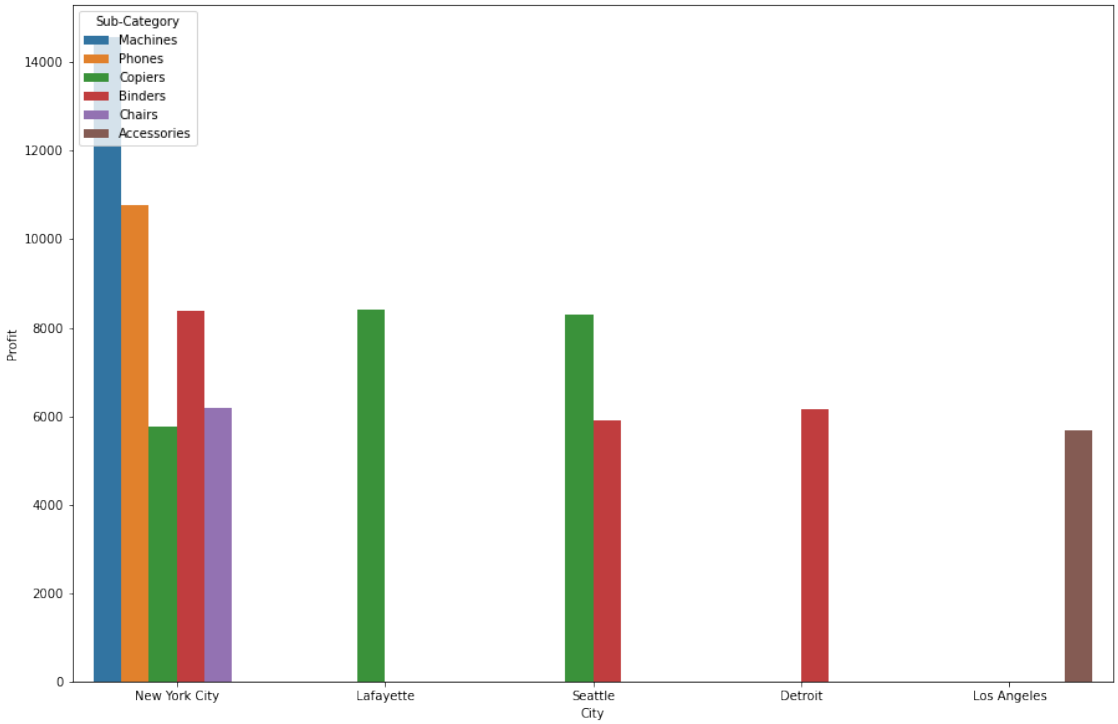
	City	RowID	PostalCode	Sales	Quantity	Discount	\
329	New York City	4450928	9168909	256368.1610	3417	51.40	
266	Los Angeles	3785302	67252887	175851.3410	2879	55.50	
374	Philadelphia	2531454	10275302	109077.0130	1981	175.50	
438	San Francisco	2462471	47998395	112669.0920	1935	34.00	
452	Seattle	2199185	41989758	119540.7420	1590	27.80	
207	Houston	2017125	29052387	64504.7604	1466	143.14	
80	Chicago	1643465	19037248	48539.5410	1132	120.50	
94	Columbus	1076696	8961990	38706.2430	836	38.50	
437	San Diego	827518	15650880	47521.0290	670	13.60	
464	Springfield	819285	9016357	43054.3420	649	23.20	

```
          Profit
329  62036.9837
266  30440.7579
374 -13837.7674
438  17507.3854
452  29156.0967
207 -10153.5485
80   -6654.5688
94    5897.1013
437   6377.1960
464   6200.6974
```

Graph 4: Highest Grosssing SubCategroies Across Cities

[792]:
```
df_cat_cit = copy.deepcopy(df.groupby(['City','Category','Sub-Category']).sum())
df_cat_cit = df_cat_cit.reset_index()
df_cat_cit = df_cat_cit.sort_values(by=['Profit'], ascending = False)
df_cat_cit_T10 = df_cat_cit.iloc[:10,:]
df_cat_cit_T10
fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x='City',y='Profit', hue='Sub-Category', data=df_cat_cit_T10, ax=ax)
```

[792]: <AxesSubplot:xlabel='City', ylabel='Profit'>

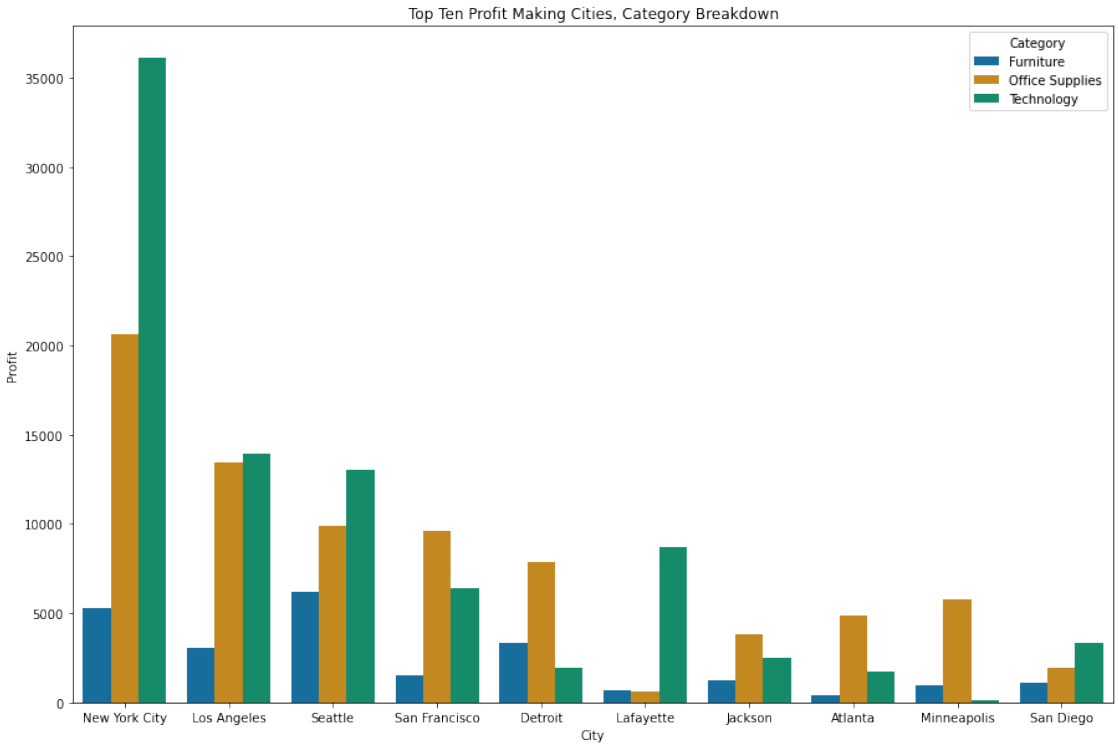


Inference On Graph 4: The above depicts that New York City is the most porfitable city across most profitable subcategories with copiers being the dominating subcat across most cities as it is the most profitable commodity overall. Binders come in second as the most profitable subcat from office appliances and machines were the most successful technology subcat in the most profitable city.

Graph 5: Most Profitable Sectors Across Top Ten Most Profitable Cities

[793]:
```
top10prof = pd.DataFrame()
for i in df_city_max.City:
    df_top10_cat = df[df.City == i]
    df_top10_cat = df_top10_cat.groupby(['City','Category']).sum()
    top10prof = top10prof.append(df_top10_cat)


top10prof = top10prof.drop(['PostalCode','RowID'], axis=1)
top10prof = top10prof.reset_index()

fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x='City', y='Profit', hue = 'Category', data=top10prof, ax=ax,␣
 ↪palette='colorblind')
plt.title('Top Ten Profit Making Cities, Category Breakdown')
```

[793]: Text(0.5, 1.0, 'Top Ten Profit Making Cities, Category Breakdown')

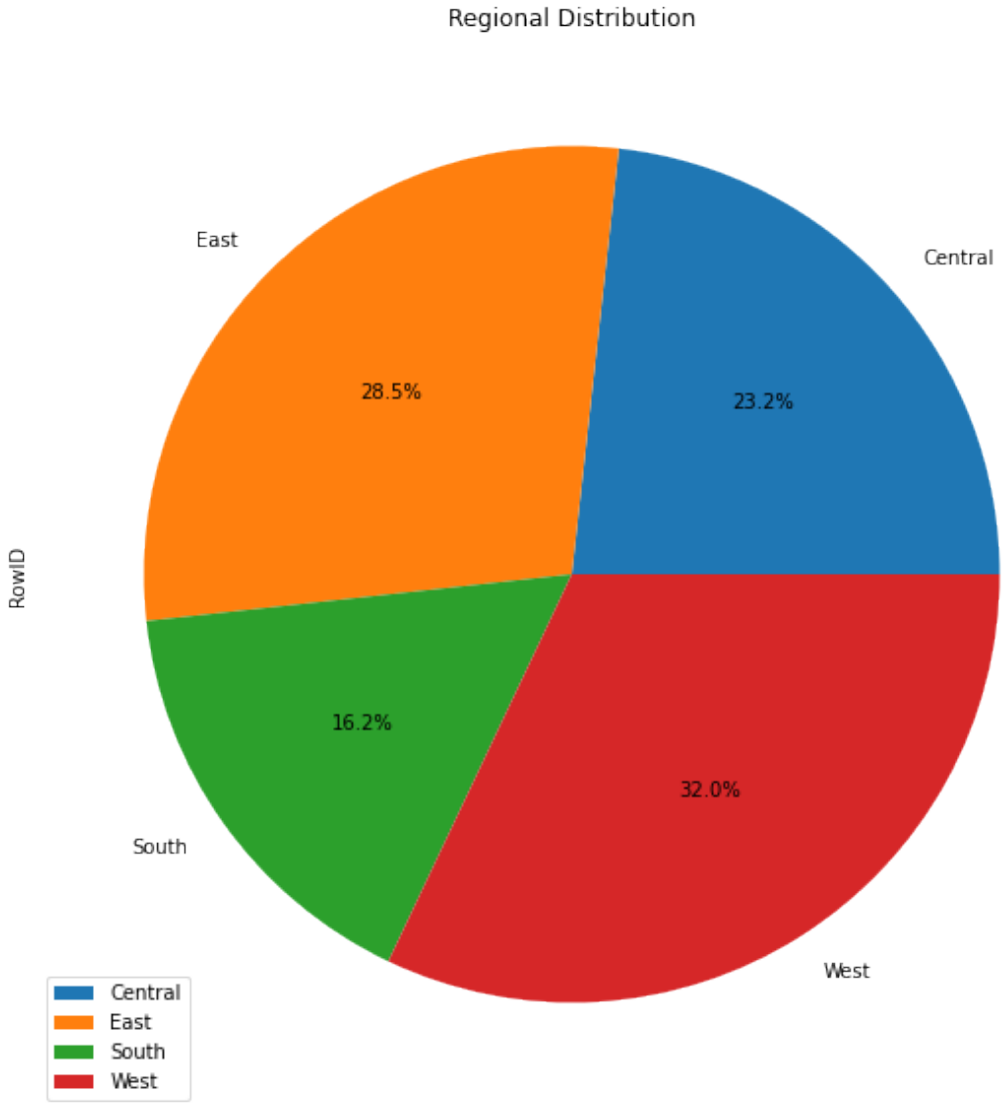


[794]:
```
uni_reg = df.Region.unique()
l1 = []
for i in uni_reg:
    v = df.Region.str.count(i).sum()
    l1.append(v)

res = dict(zip(uni_reg,l1))
res = pd.DataFrame([res])
```

Graph 6: Pie Chart Of Proporation Of Each Regions To The Total Turnover

[795]:
```
data = df.groupby(['Region']).count().iloc[:,:1]
fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
data.plot.pie(autopct='%.1f%%', subplots = True, ax=ax)
plt.title('Regional Distribution')
```

[795]:
```
Text(0.5, 1.0, 'Regional Distribution')
```



[796]:
```
df.head()
```

[796]:

	RowID	OrderID	OrderDate	ShipDate	ShipMode	CustomerID	…
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	

	CustomerName	Segment	Country	City	…	PostalCode	\
0	Claire Gute	Consumer	United States	Henderson	…	42420	
1	Claire Gute	Consumer	United States	Henderson	…	42420	
2	Darrin Van Huff	Corporate	United States	Los Angeles	…	90036	

```
3   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …      33311
4   Sean O'Donnell   Consumer  United States  Fort Lauderdale  …      33311

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

                                        ProductName     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]
```

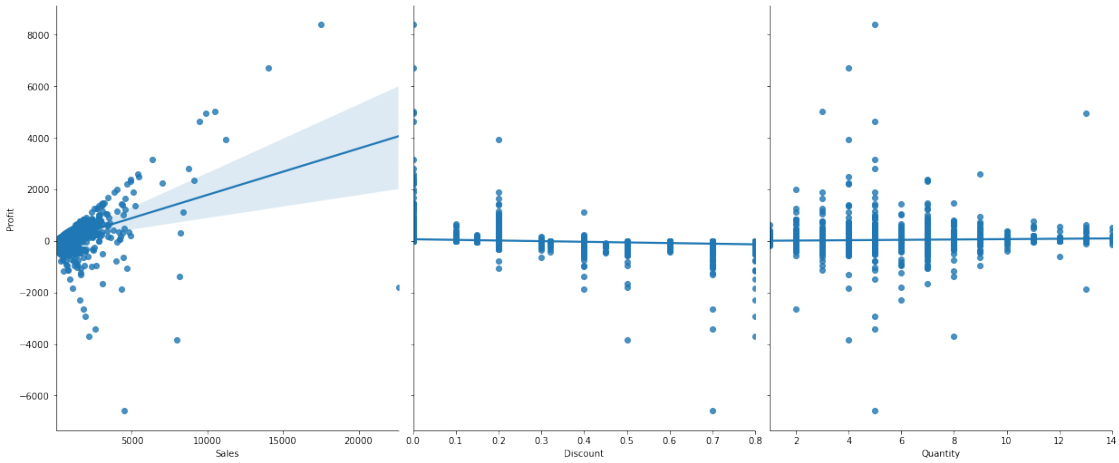
[797]:
```
print(df.dtypes)
```

```
RowID                    int64
OrderID                 object
OrderDate       datetime64[ns]
ShipDate        datetime64[ns]
ShipMode                object
CustomerID              object
CustomerName            object
Segment                 object
Country                 object
City                    object
State                   object
PostalCode               int64
Region                  object
ProductID               object
Category                object
Sub-Category            object
ProductName             object
Sales                  float64
Quantity                 int64
```

```
Discount                float64
Profit                  float64
dtype: object
```

Chart 7: Regression Plot Of Sales As Dependent & Proftability As Independent

[818]:
```
sale_prof = sns.pairplot(df, x_vars=['Sales','Discount', 'Quantity'],␣
↪y_vars='Profit', height =7, aspect=0.8, kind = 'reg')
```



Regression Model Of Predicting Profitability By Sales:

The score of the model is extremely poor due to a lot of key association and variables not being added to the model. This was not done due to restriction on time.

[831]:
```
from sklearn.model_selection import train_test_split
X = df[['Sales','Discount', 'Quantity']]
Y = df.Profit
X = np.array(X)
Y = np.array(Y)
X_test,X_train,Y_test,Y_train = train_test_split(X,Y,test_size=1/3,random_state␣
↪= 0)
```

[832]:
```
X_train = X_train.reshape(-1,3)
Y_train = Y_train.reshape(-1,1)
```

[833]:
```
from sklearn.linear_model import LinearRegression
lmod = LinearRegression()
lmod.fit(X_train, Y_train)
def predict_my_profit(x,y,z):
    return str(lmod.predict([[x,y,z]])).lstrip('[').rstrip(']')
```

[834]:
```
lmod.score(X_train, Y_train)
```

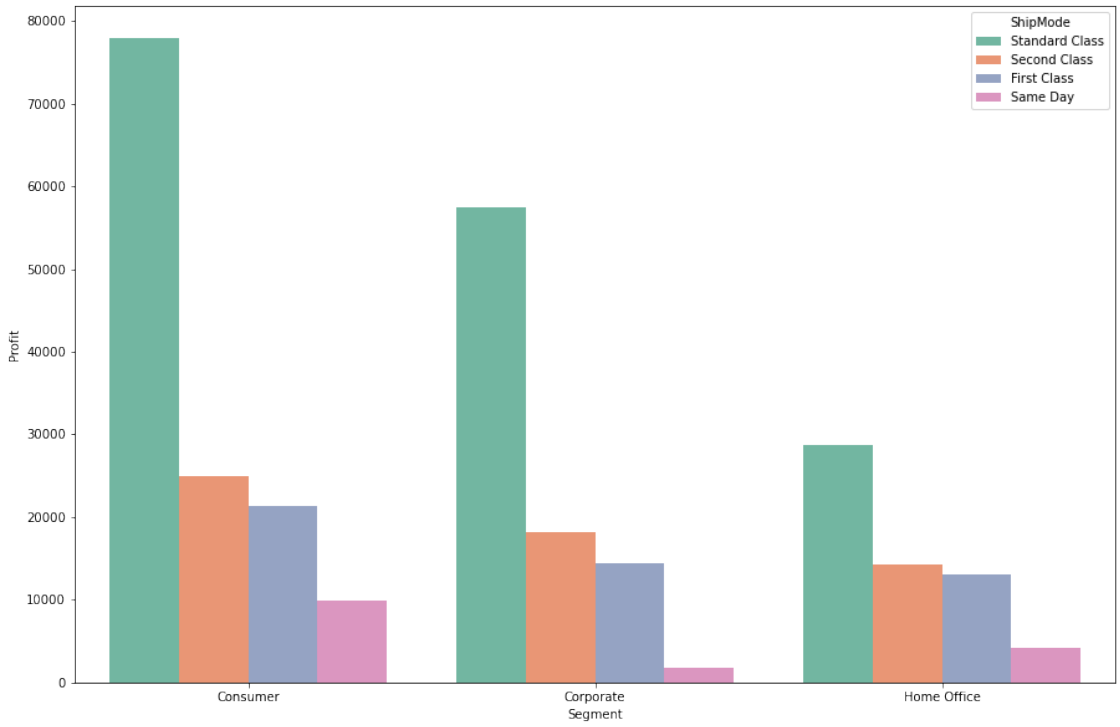
[834]:
```
0.40728110093069914
```

[842]:
```
predict_my_profit(20000,2,50)
```

[842]:
```
'4285.91809123'
```

[843]:
```
df_seg = df.groupby(['Segment','ShipMode']).sum()
df_seg = df_seg.drop(['RowID','PostalCode'], axis = 1)
df_seg = df_seg.reset_index()
df_seg = df_seg.sort_values(by = 'Profit', ascending = False)
fig_dims = (15, 10)
fig, ax = plt.subplots(figsize=fig_dims)
sns.barplot(x='Segment',y='Profit', hue='ShipMode',data=df_seg, palette='Set2')
```

[843]:
```
<AxesSubplot:xlabel='Segment', ylabel='Profit'>
```



[844]:
```
dt = copy.deepcopy(df)
dt
```

[844]:

	RowID	OrderID	OrderDate	ShipDate	ShipMode	CustomerID	\
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
…     …             …        …          …              …        …
9989   9990  CA-2014-110422 2014-01-21 2014-01-23    Second Class   TB-21400
9990   9991  CA-2017-121258 2017-02-26 2017-03-03  Standard Class   DB-13060
9991   9992  CA-2017-121258 2017-02-26 2017-03-03  Standard Class   DB-13060
9992   9993  CA-2017-121258 2017-02-26 2017-03-03  Standard Class   DB-13060
9993   9994  CA-2017-119914 2017-05-04 2017-05-09    Second Class   CC-12220

          CustomerName    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  …
…                  …        …             …                …  …
9989  Tom Boeckenhauer   Consumer  United States            Miami  …
9990       Dave Brooks   Consumer  United States       Costa Mesa  …
9991       Dave Brooks   Consumer  United States       Costa Mesa  …
9992       Dave Brooks   Consumer  United States       Costa Mesa  …
9993      Chris Cortes   Consumer  United States      Westminster  …

     PostalCode  Region       ProductID         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
…         …     …              …               …           …
9989      33180   South  FUR-FU-10001889        Furniture  Furnishings
9990      92627    West  FUR-FU-10000747        Furniture  Furnishings
9991      92627    West  TEC-PH-10003645       Technology       Phones
9992      92627    West  OFF-PA-10004041  Office Supplies        Paper
9993      92683    West  OFF-AP-10002684  Office Supplies   Appliances

                                             ProductName     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
…                                                  …       …       …
9989                           Ultra Door Pull Handle   25.2480         3
9990  Tenex B1-RE Series Chair Mats for Low Pile Car…   91.9600         2
9991                             Aastra 57i VoIP phone  258.5760         2
9992  It's Hot Message Books with Stickers, 2 3/4" x 5"   29.6000         4
9993  Acco 7-Outlet Masterpiece Power Center, Wihtou…  243.1600         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
…          …       …
9989      0.20    4.1028
9990      0.00   15.6332
9991      0.20   19.3932
9992      0.00   13.3200
9993      0.00   72.9480

[9994 rows x 21 columns]
```

Regression Model To Predict the ShipDate while taking OrderDate as input:

[845]:
```
dt.OrderDate = pd.to_numeric(dt.OrderDate)
dt.ShipDate = pd.to_numeric(dt.ShipDate)
from sklearn.model_selection import train_test_split
X = np.array(dt.OrderDate)
Y = np.array(dt.ShipDate)
X_test,X_train,Y_test,Y_train = train_test_split(X,Y,test_size=1/3,random_state␣
↪= 0)
```

[846]:
```
X_train = X_train.reshape(-1,1)
Y_train = Y_train.reshape(-1,1)
```

[847]:
```
from sklearn.linear_model import LinearRegression
lmod1 = LinearRegression()
lmod1.fit(X_train,Y_train)
lmod1.score(X_train,Y_train)
```

[847]:
```
0.9999827897059588
```

[524]:
```
dt
```

[524]:
```
      RowID         OrderID            OrderDate             ShipDate  \
0         1  CA-2016-152156  1478563200000000000  1478822400000000000
1         2  CA-2016-152156  1478563200000000000  1478822400000000000
2         3  CA-2016-138688  1465689600000000000  1466035200000000000
3         4  US-2015-108966  1444521600000000000  1445126400000000000
4         5  US-2015-108966  1444521600000000000  1445126400000000000
…       …              …                  …                    …
9989   9990  CA-2014-110422  1390262400000000000  1390435200000000000
9990   9991  CA-2017-121258  1488067200000000000  1488499200000000000
9991   9992  CA-2017-121258  1488067200000000000  1488499200000000000
```

```
9992   9993  CA-2017-121258  1488067200000000000  1488499200000000000
9993   9994  CA-2017-119914  1493856000000000000  1494288000000000000

            ShipMode CustomerID      CustomerName    Segment        Country  \
0       Second Class   CG-12520       Claire Gute   Consumer  United States
1       Second Class   CG-12520       Claire Gute   Consumer  United States
2       Second Class   DV-13045   Darrin Van Huff  Corporate  United States
3     Standard Class   SO-20335    Sean O'Donnell   Consumer  United States
4     Standard Class   SO-20335    Sean O'Donnell   Consumer  United States
…                …          …                 …          …              …
9989    Second Class   TB-21400  Tom Boeckenhauer   Consumer  United States
9990  Standard Class   DB-13060       Dave Brooks   Consumer  United States
9991  Standard Class   DB-13060       Dave Brooks   Consumer  United States
9992  Standard Class   DB-13060       Dave Brooks   Consumer  United States
9993    Second Class   CC-12220      Chris Cortes   Consumer  United States

                City  … PostalCode  Region        ProductID  \
0          Henderson  …      42420   South  FUR-BO-10001798
1          Henderson  …      42420   South  FUR-CH-10000454
2        Los Angeles  …      90036    West  OFF-LA-10000240
3     Fort Lauderdale  …      33311   South  FUR-TA-10000577
4     Fort Lauderdale  …      33311   South  OFF-ST-10000760
…                 …  …          …       …              …
9989           Miami  …      33180   South  FUR-FU-10001889
9990       Costa Mesa  …      92627    West  FUR-FU-10000747
9991       Costa Mesa  …      92627    West  TEC-PH-10003645
9992       Costa Mesa  …      92627    West  OFF-PA-10004041
9993      Westminster  …      92683    West  OFF-AP-10002684

             Category Sub-Category  \
0           Furniture    Bookcases
1           Furniture       Chairs
2     Office Supplies       Labels
3           Furniture       Tables
4     Office Supplies      Storage
…                 …            …
9989        Furniture  Furnishings
9990        Furniture  Furnishings
9991       Technology       Phones
9992  Office Supplies        Paper
9993  Office Supplies   Appliances

                                              ProductName     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
…                                                    …       …       …
9989                              Ultra Door Pull Handle   25.2480         3
9990  Tenex B1-RE Series Chair Mats for Low Pile Car…   91.9600         2
9991                                Aastra 57i VoIP phone  258.5760         2
9992  It's Hot Message Books with Stickers, 2 3/4" x 5"   29.6000         4
9993  Acco 7-Outlet Masterpiece Power Center, Wihtou…  243.1600         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
…          …       …
9989      0.20    4.1028
9990      0.00   15.6332
9991      0.20   19.3932
9992      0.00   13.3200
9993      0.00   72.9480

[9994 rows x 21 columns]
```

[848]:
```
def pred_ship_date(x):
    d = str(x)
    d = pd.to_datetime(d)
    d = pd.Series(d)
    d = pd.to_numeric(d)
    d = lmod1.predict([d])
    d = d[0][0]
    d = pd.to_datetime(d)
    return d
```

[852]:
```
pred_ship_date('2015-06-10')
```

[852]:
```
Timestamp('2015-06-14 01:11:35.502362624')
```