Analyzing Amazon Sales data Analysis

Data Preparation and Cleaning

Import Analytics and Visulization Lab's

In [184…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Load CSV File using Panda

In [185…
```
df = pd.read_csv("C:/Users/24/Desktop/Analyzing Amazon Sales data Analysis/Amazon sale
```

In [186…
```
df.head()
```

Out[186]:

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	15
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	11
2	Europe	Russia	Office Supplies	Offline	L	5/2/2014	341417157	5/8/2014	1779	651.21	52
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	7/5/2014	8102	9.33	
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	52

In [187…
```
df.tail()
```

Out[187]:

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	F
95	Sub-Saharan Africa	Mali	Clothes	Online	M	7/26/2011	512878119	9/3/2011	888	10
96	Asia	Malaysia	Fruits	Offline	L	11/11/2011	810711038	12/28/2011	6267	
97	Sub-Saharan Africa	Sierra Leone	Vegetables	Offline	C	6/1/2016	728815257	6/29/2016	1485	15
98	North America	Mexico	Personal Care	Offline	M	7/30/2015	559427106	8/8/2015	5767	8
99	Sub-Saharan Africa	Mozambique	Household	Offline	L	2/10/2012	665095412	2/15/2012	5367	66

In [188…
```
df.shape
```

Out[188]:
```
(100, 14)
```

In [189…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Region          100 non-null    object 
 1   Country         100 non-null    object 
 2   Item Type       100 non-null    object 
 3   Sales Channel   100 non-null    object 
 4   Order Priority  100 non-null    object 
 5   Order Date      100 non-null    object 
 6   Order ID        100 non-null    int64  
 7   Ship Date       100 non-null    object 
 8   Units Sold      100 non-null    int64  
 9   Unit Price      100 non-null    float64
 10  Unit Cost       100 non-null    float64
 11  Total Revenue   100 non-null    float64
 12  Total Cost      100 non-null    float64
 13  Total Profit    100 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 11.1+ KB
```

In [190…
```
df.columns
```

Out[190]:
```
Index(['Region', 'Country', 'Item Type', 'Sales Channel', 'Order Priority',
       'Order Date', 'Order ID', 'Ship Date', 'Units Sold', 'Unit Price',
       'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit'],
      dtype='object')
```

In [191…
```
df.describe()
```

Out[191]:

	Order ID	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
count	1.000000e+02	100.000000	100.000000	100.000000	1.000000e+02	1.000000e+02	1.000000e+02
mean	5.550204e+08	5128.710000	276.761300	191.048000	1.373488e+06	9.318057e+05	4.416820e+05
std	2.606153e+08	2794.484562	235.592241	188.208181	1.460029e+06	1.083938e+06	4.385379e+05
min	1.146066e+08	124.000000	9.330000	6.920000	4.870260e+03	3.612240e+03	1.258020e+03
25%	3.389225e+08	2836.250000	81.730000	35.840000	2.687212e+05	1.688680e+05	1.214436e+05
50%	5.577086e+08	5382.500000	179.880000	107.275000	7.523144e+05	3.635664e+05	2.907680e+05
75%	7.907551e+08	7369.000000	437.200000	263.330000	2.212045e+06	1.613870e+06	6.358288e+05
max	9.940222e+08	9925.000000	668.270000	524.960000	5.997055e+06	4.509794e+06	1.719922e+06

Data Analysis

Checking for Missing Data

In [192…
```
df.shape
```

Out[192]:
```
(100, 14)
```

In [193…
```
df.isnull().sum()
```

Out[193]:
```
Region            0
Country           0
Item Type         0
Sales Channel     0
Order Priority    0
Order Date        0
Order ID          0
Ship Date         0
Units Sold        0
Unit Price        0
Unit Cost         0
Total Revenue     0
Total Cost        0
Total Profit      0
dtype: int64
```

In [194…
```
df.dropna(subset=["Total Profit"],axis=0,inplace=True)
```

In [195…
```
df.shape
```

Out[195]:
```
(100, 14)
```

In [196…
```
df.isnull().sum()
```

Out[196]:
```
Region            0
Country           0
Item Type         0
Sales Channel     0
Order Priority    0
Order Date        0
Order ID          0
Ship Date         0
Units Sold        0
Unit Price        0
Unit Cost         0
Total Revenue     0
Total Cost        0
Total Profit      0
dtype: int64
```

Checking for Correlation between variables

In [197…
```
df.corr()
```

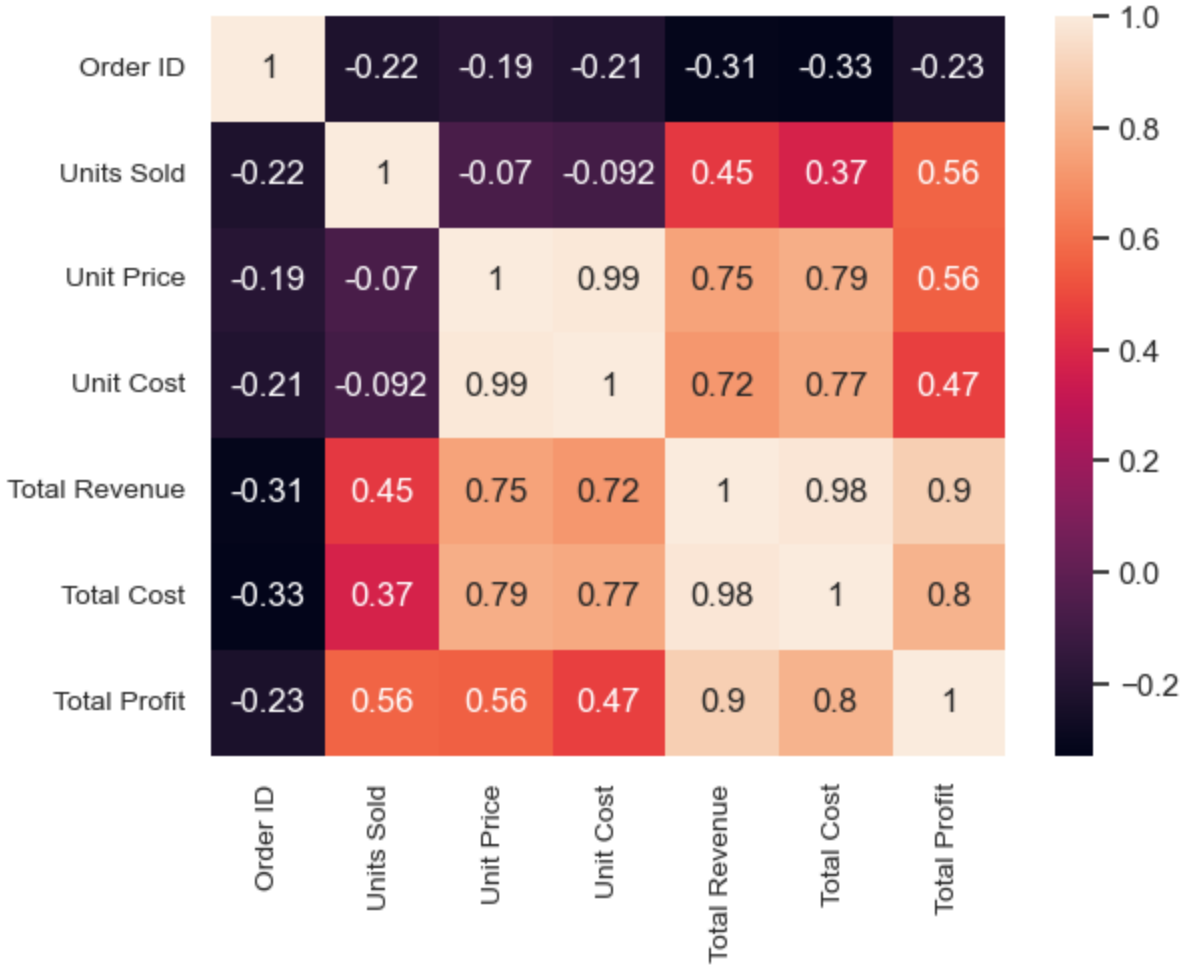
Out[197]:

	Order ID	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
Order ID	1.000000	-0.222907	-0.190941	-0.213201	-0.314688	-0.328944	-0.234638
Units Sold	-0.222907	1.000000	-0.070486	-0.092232	0.447784	0.374746	0.564550
Unit Price	-0.190941	-0.070486	1.000000	0.987270	0.752360	0.787905	0.557365
Unit Cost	-0.213201	-0.092232	0.987270	1.000000	0.715623	0.774895	0.467214
Total Revenue	-0.314688	0.447784	0.752360	0.715623	1.000000	0.983928	0.897327
Total Cost	-0.328944	0.374746	0.787905	0.774895	0.983928	1.000000	0.804091
Total Profit	-0.234638	0.564550	0.557365	0.467214	0.897327	0.804091	1.000000

In [ ]:
```
```

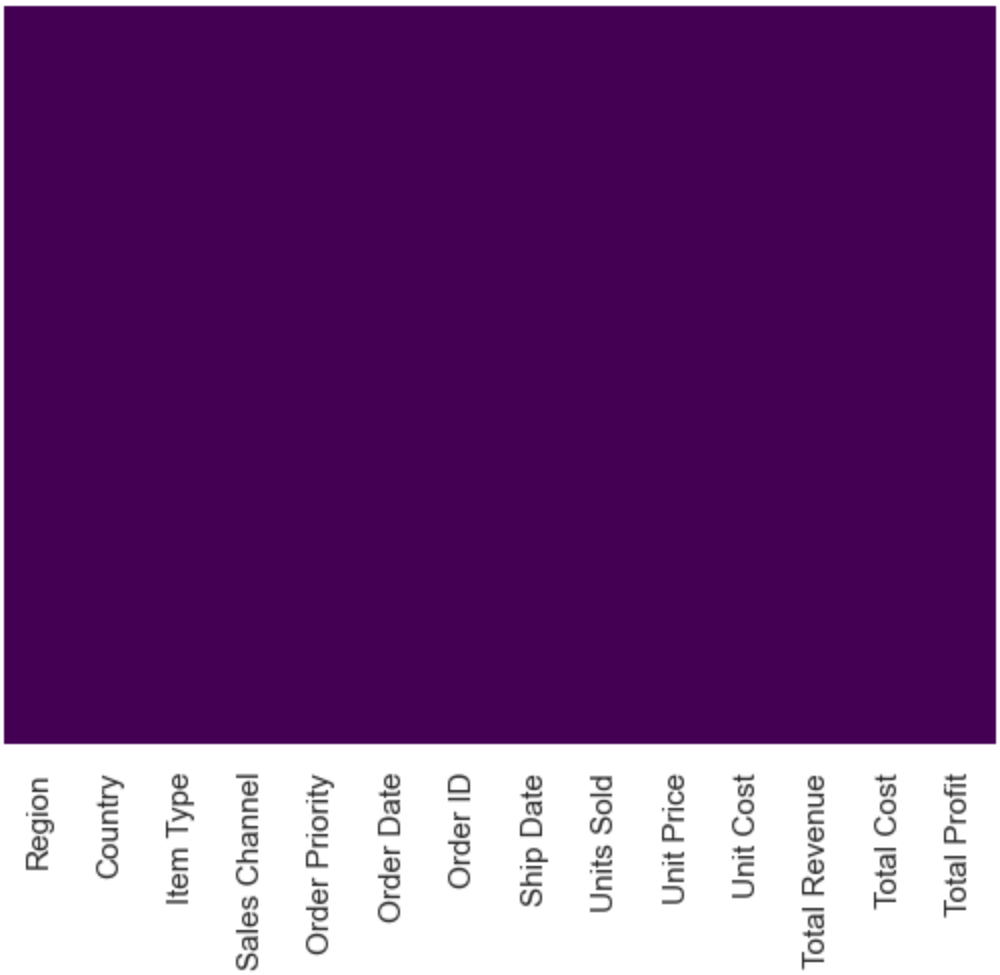
In [198…
```
plt.tick_params(labelsize=10)
sns.heatmap(df.corr(),annot=True)
```

Out[198]:
```
<AxesSubplot:>
```



In [199…
```
sns.heatmap(df.isnull(),yticklabels =False, cbar= False, cmap="viridis")
```

Out[199]:
```
<AxesSubplot:>
```



Variable Analysis

Country Variable

In [200…
```
df.columns
```

Out[200]:
```
Index(['Region', 'Country', 'Item Type', 'Sales Channel', 'Order Priority',
       'Order Date', 'Order ID', 'Ship Date', 'Units Sold', 'Unit Price',
       'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit'],
      dtype='object')
```

In [201…
```
df.Country.unique()
```

Out[201]:
```
array(['Tuvalu', 'Grenada', 'Russia', 'Sao Tome and Principe', 'Rwanda',
       'Solomon Islands', 'Angola', 'Burkina Faso',
       'Republic of the Congo', 'Senegal', 'Kyrgyzstan', 'Cape Verde',
       'Bangladesh', 'Honduras', 'Mongolia', 'Bulgaria', 'Sri Lanka',
       'Cameroon', 'Turkmenistan', 'East Timor', 'Norway', 'Portugal',
       'New Zealand', 'Moldova ', 'France', 'Kiribati', 'Mali',
       'The Gambia', 'Switzerland', 'South Sudan', 'Australia', 'Myanmar',
       'Djibouti', 'Costa Rica', 'Syria', 'Brunei', 'Niger', 'Azerbaijan',
       'Slovakia', 'Comoros', 'Iceland', 'Macedonia', 'Mauritania',
       'Albania', 'Lesotho', 'Saudi Arabia', 'Sierra Leone',
       "Cote d'Ivoire", 'Fiji', 'Austria', 'United Kingdom', 'San Marino',
       'Libya', 'Haiti', 'Gabon', 'Belize', 'Lithuania', 'Madagascar',
       'Democratic Republic of the Congo', 'Pakistan', 'Mexico',
       'Federated States of Micronesia', 'Laos', 'Monaco', 'Samoa ',
       'Spain', 'Lebanon', 'Iran', 'Zambia', 'Kenya', 'Kuwait',
       'Slovenia', 'Romania', 'Nicaragua', 'Malaysia', 'Mozambique'],
      dtype=object)
```

In [202…
```
df.Country.nunique()
```

Out[202]:
```
76
```

This dataset encodes agriculture data for 76 country

In [203…
```
df.Country.value_counts()
```

Out[203]:
```
The Gambia               4
Sierra Leone             3
Sao Tome and Principe    3
Mexico                   3
Australia                3
                        ..
Comoros                  1
Iceland                  1
Macedonia                1
Mauritania               1
Mozambique               1
Name: Country, Length: 76, dtype: int64
```

In [204…
```
df.Region.unique()
```

Out[204]:
```
array(['Australia and Oceania', 'Central America and the Caribbean',
       'Europe', 'Sub-Saharan Africa', 'Asia',
       'Middle East and North Africa', 'North America'], dtype=object)
```

In [205…
```
df.Region.nunique()
```

Out[205]:
```
7
```

For further dissection we get data for 76 countries and 7 regions

In [206…
```
df.Region.value_counts()
```

Out[206]:
```
Sub-Saharan Africa                  36
Europe                              22
Australia and Oceania               11
Asia                                11
Middle East and North Africa        10
Central America and the Caribbean    7
North America                        3
Name: Region, dtype: int64
```

In [207…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Region          100 non-null    object 
 1   Country         100 non-null    object 
 2   Item Type       100 non-null    object 
 3   Sales Channel   100 non-null    object 
 4   Order Priority  100 non-null    object 
 5   Order Date      100 non-null    object 
 6   Order ID        100 non-null    int64  
 7   Ship Date       100 non-null    object 
 8   Units Sold      100 non-null    int64  
 9   Unit Price      100 non-null    float64
 10  Unit Cost       100 non-null    float64
 11  Total Revenue   100 non-null    float64
 12  Total Cost      100 non-null    float64
 13  Total Profit    100 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 11.1+ KB
```

In [208…
```
df.axes[0]
```

Out[208]:
```
RangeIndex(start=0, stop=100, step=1)
```

In [209…
```
df.axes[1]
```

Out[209]:
```
Index(['Region', 'Country', 'Item Type', 'Sales Channel', 'Order Priority',
       'Order Date', 'Order ID', 'Ship Date', 'Units Sold', 'Unit Price',
       'Unit Cost', 'Total Revenue', 'Total Cost', 'Total Profit'],
      dtype='object')
```

In [210…
```
df.dtypes
```

Out[210]:
```
Region             object
Country            object
Item Type          object
Sales Channel      object
Order Priority     object
Order Date         object
Order ID            int64
Ship Date          object
Units Sold          int64
Unit Price        float64
Unit Cost         float64
Total Revenue     float64
Total Cost        float64
Total Profit      float64
dtype: object
```

In [211…
```
df.index
```

Out[211]:
```
RangeIndex(start=0, stop=100, step=1)
```

In [212…
```
df.columns.isnull()
```

Out[212]:
```
array([False, False, False, False, False, False, False, False, False,
       False, False, False, False, False])
```

In [213…
```
df.loc[:,["Total Revenue","Total Profit"]].iloc[:]
```

Out[213]:

	Total Revenue	Total Profit
0	2533654.00	951410.50
1	576782.80	248406.36
2	1158502.59	224598.75
3	75591.66	19525.82
4	3296425.02	639077.50
...	...	...
95	97040.64	65214.72
96	58471.11	15103.47
97	228779.10	93748.05
98	471336.91	144521.02
99	3586605.09	889472.91

100 rows × 2 columns

In [214…
```
df.shape
```

Out[214]:
```
(100, 14)
```

In [215…
```
np.corrcoef(df.loc[:,"Total Revenue"].iloc[:], df.loc[:,"Total Profit"].iloc[:])
```

Out[215]:
```
array([[1.        , 0.89732687],
       [0.89732687, 1.        ]])
```

In [216…
```
df.set_index("Order ID",inplace=True)
```

In [217…
```
df.head()
```

Out[217]:

Order ID	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Ship Date	Units Sold	Unit Price	Un Co
669165933	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	6/27/2010	9925	255.28	159.4
963881480	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	9/15/2012	2804	205.70	117.1
341417157	Europe	Russia	Office Supplies	Offline	L	5/2/2014	5/8/2014	1779	651.21	524.9
514321792	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	7/5/2014	8102	9.33	6.9
115456712	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	2/6/2013	5062	651.21	524.9

In [218…
```
df.cov()
```

Out[218]:

	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
Units Sold	7.809144e+06	-4.640481e+04	-4.850918e+04	1.826973e+09	1.135124e+09	6.918495e+08
Unit Price	-4.640481e+04	5.550370e+04	4.377593e+04	2.587902e+08	2.012054e+08	5.758482e+07
Unit Cost	-4.850918e+04	4.377593e+04	3.542232e+04	1.966455e+08	1.580833e+08	3.856216e+07
Total Revenue	1.826973e+09	2.587902e+08	1.966455e+08	2.131684e+12	1.557145e+12	5.745386e+11
Total Cost	1.135124e+09	2.012054e+08	1.580833e+08	1.557145e+12	1.174922e+12	3.822231e+11
Total Profit	6.918495e+08	5.758482e+07	3.856216e+07	5.745386e+11	3.822231e+11	1.923155e+11

The high value of Pearson correlation coefficient between Total Revenue and Total Profit indicates that these two variables are closely related to each other.

If revenue generated is high, then more profit will be generated and vice versa.

The negative value of correlation coefficient among Units Sold and Unit Cost implies that quantity of products is inversely proportional to their cost. Same is the scenario with Units Sold and Units Price. Lesser the number of units of a product available, more will be it's price.

In [219…
```
np.average(df["Total Profit"])
```
Out[219]:
```
441681.98399999994
```

At an average, the profit generated for a product is 441681.98

Total Profit

In [220…
```
np.max(df["Total Profit"])
```
Out[220]:
```
1719922.04
```

In [221…
```
np.min(df["Total Profit"])
```
Out[221]:
```
1258.02
```

In [222…
```
np.var(df["Total Profit"])
```
Out[222]:
```
190392340968.9648
```

Maximum and minimum profit generated are 1719922.04 and 1258.09 respectively

Total Revenue

In [223…
```
np.max(df["Total Revenue"])
```
Out[223]:
```
5997054.98
```

In [224…
```
np.min(df["Total Revenue"])
```
Out[224]:
```
4870.26
```

In [225…
```
np.var(df["Total Revenue"])
```
Out[225]:
```
2110366986501.2166
```

In [226…
```
np.mean(df["Total Revenue"])
```
Out[226]:
```
1373487.6830999998
```

In [227…
```
np.std(df["Total Revenue"])
```
Out[227]:
```
1452710.2211044075
```

In [228…
```
np.median(df["Total Revenue"])
```

Out[228]: 752314.36

In [229…
```
np.percentile(df["Total Revenue"],50,axis=0,overwrite_input=True)
```

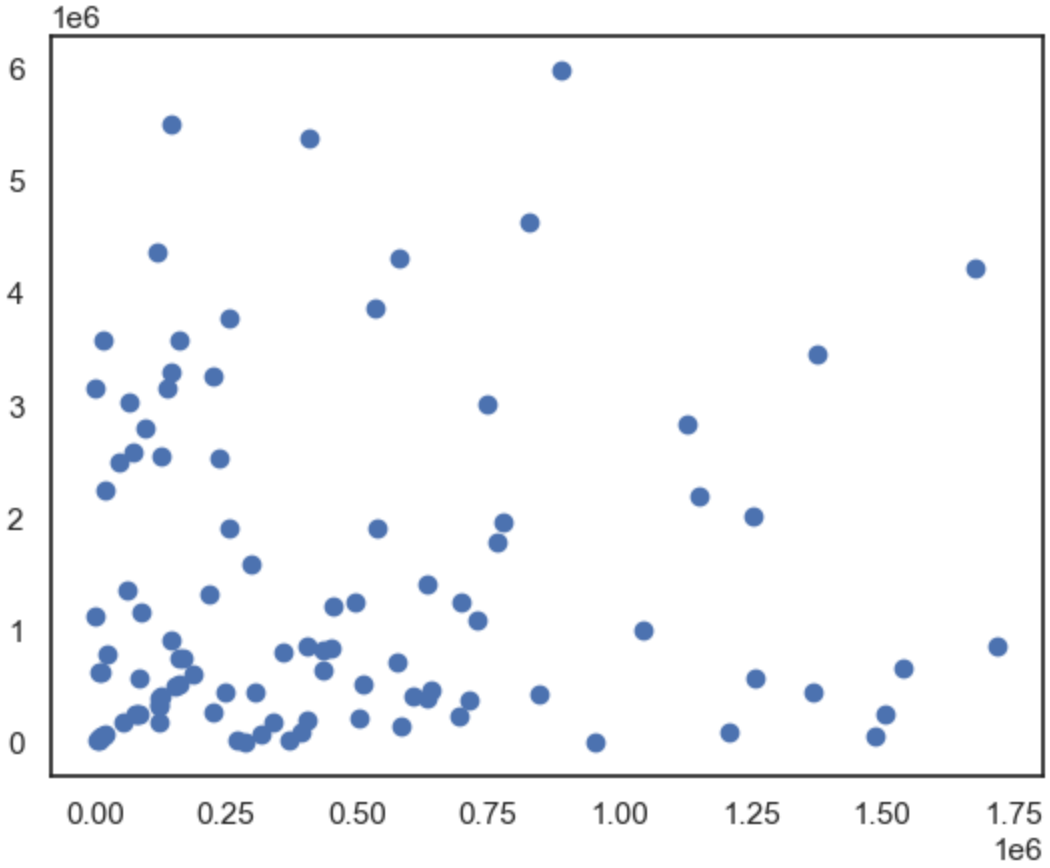
Out[229]: 752314.36

Maximum and minimum revenue generated by the product are ₹ 5997054.98 and ₹ 4870.26.

Revenue has very high variability in it's distribution. The median revenue generated is ₹ 752314.36.¶

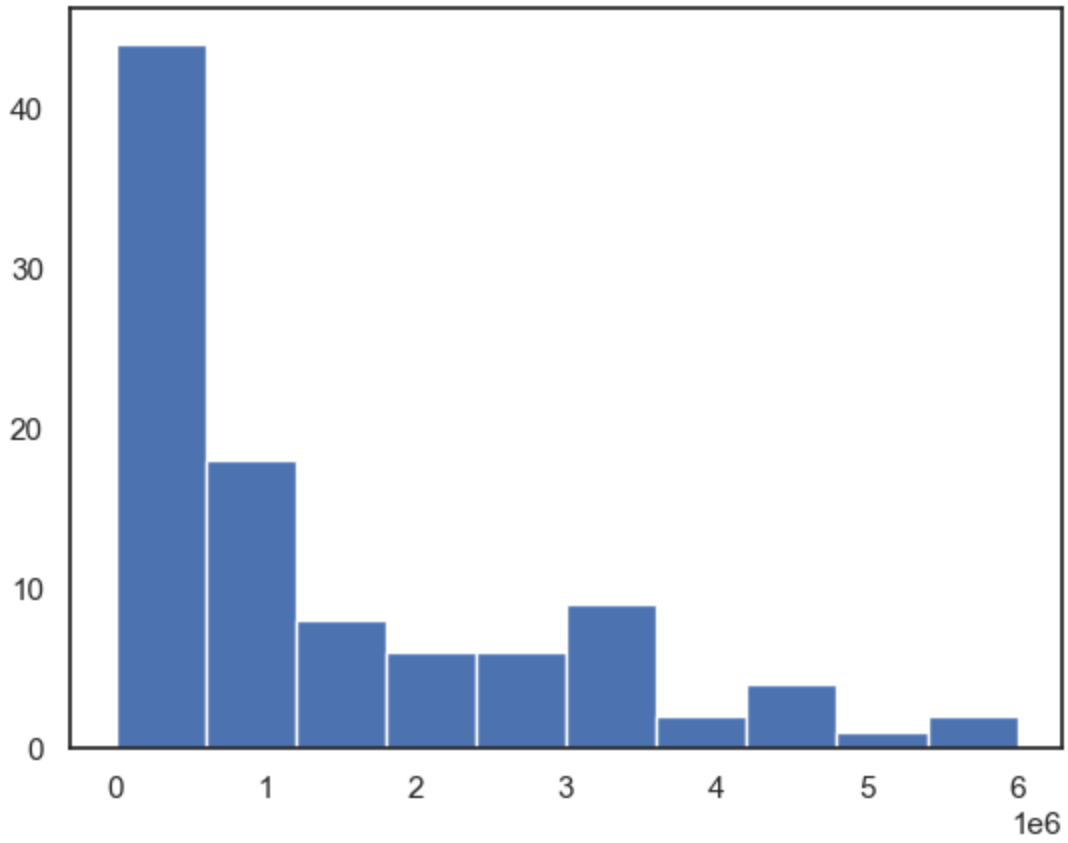
In [230…
```
plt.scatter(df["Total Profit"],df["Total Revenue"])
```

Out[230]: <matplotlib.collections.PathCollection at 0x1e24c226370>



In [231…
```
plt.hist(df["Total Revenue"])
```

Out[231]:
```
(array([44., 18.,  8.,  6.,  6.,  9.,  2.,  4.,  1.,  2.]),
 array([4.87026000e+03, 6.04088732e+05, 1.20330720e+06, 1.80252568e+06,
        2.40174415e+06, 3.00096262e+06, 3.60018109e+06, 4.19939956e+06,
        4.79861804e+06, 5.39783651e+06, 5.99705498e+06]),
 <BarContainer object of 10 artists>)
```



In [232…
```
np.correlate(df["Total Revenue"],df["Total Profit"])
```

Out[232]:
```
array([6.48023895e+13])
```

In [233…
```
df.head()
```

Out[233]:

Order ID	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Ship Date	Units Sold	Unit Price	Un Co
669165933	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	6/27/2010	9925	255.28	159.4
963881480	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	9/15/2012	2804	205.70	117.1
341417157	Europe	Russia	Office Supplies	Offline	L	5/2/2014	5/8/2014	1779	651.21	524.9
514321792	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	7/5/2014	8102	9.33	6.9
115456712	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	2/6/2013	5062	651.21	524.9

In [234…
```
np.max(df["Total Cost"])
```

Out[234]:
```
4509793.96
```

In [235…
```
np.min(df["Total Cost"])
```

Out[235]:
```
3612.24
```

In [236…
```
np.var(df["Total Cost"])
```

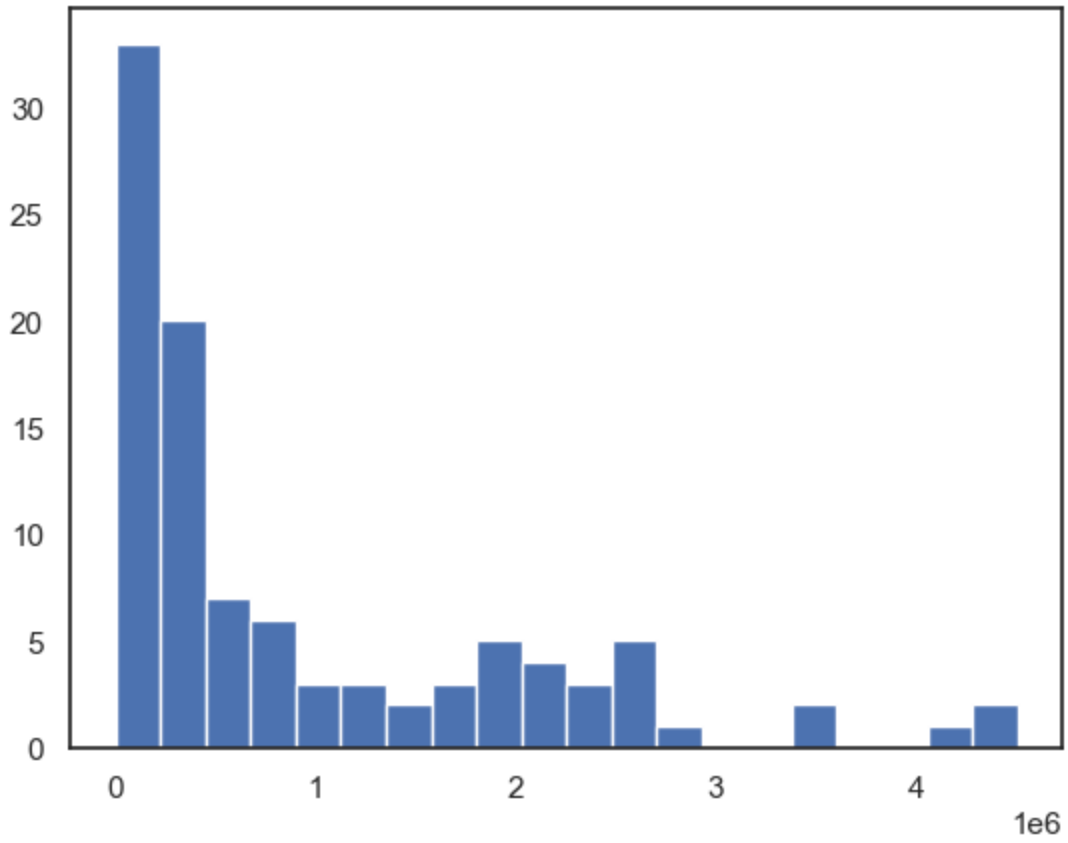
Out[236]:
```
1163172913211.59
```

In [237…
```
np.histogram(df["Total Cost"],bins=10)
```

Out[237]:
```
(array([53, 13,  6,  5,  9,  8,  1,  2,  0,  3], dtype=int64),
 array([3.61224000e+03, 4.54230412e+05, 9.04848584e+05, 1.35546676e+06,
        1.80608493e+06, 2.25670310e+06, 2.70732127e+06, 3.15793944e+06,
        3.60855762e+06, 4.05917579e+06, 4.50979396e+06]))
```

In [238…
```
plt.hist(df["Total Cost"],bins=20)
```

Out[238]:
```
(array([33., 20.,  7.,  6.,  3.,  3.,  2.,  3.,  5.,  4.,  3.,  5.,  1.,
         0.,  0.,  2.,  0.,  0.,  1.,  2.]),
 array([3.61224000e+03, 2.28921326e+05, 4.54230412e+05, 6.79539498e+05,
        9.04848584e+05, 1.13015767e+06, 1.35546676e+06, 1.58077584e+06,
        1.80608493e+06, 2.03139401e+06, 2.25670310e+06, 2.48201219e+06,
        2.70732127e+06, 2.93263036e+06, 3.15793944e+06, 3.38324853e+06,
        3.60855762e+06, 3.83386670e+06, 4.05917579e+06, 4.28448487e+06,
        4.50979396e+06]),
 <BarContainer object of 20 artists>)
```



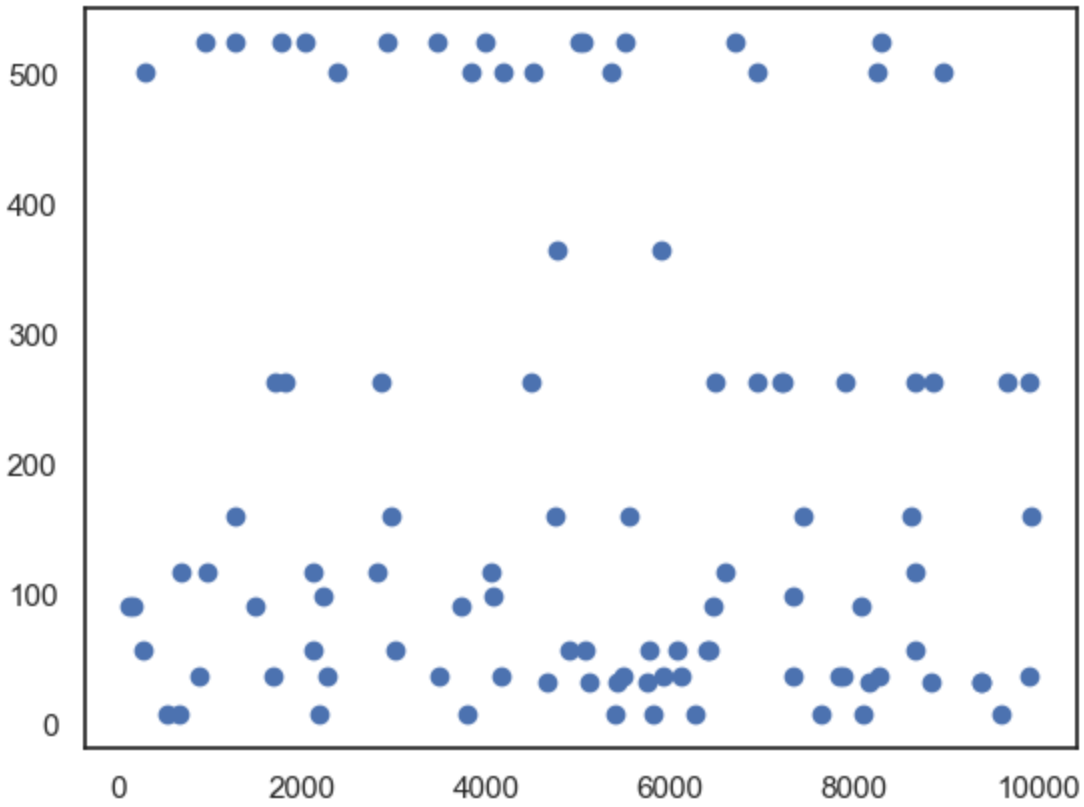
In [239…
```
df.corr(method="pearson")
```

Out[239]:

	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
Units Sold	1.000000	-0.070486	-0.092232	0.127583	0.374746	0.564550
Unit Price	-0.070486	1.000000	0.987270	0.007902	0.787905	0.557365
Unit Cost	-0.092232	0.987270	1.000000	-0.001689	0.774895	0.467214
Total Revenue	0.127583	0.007902	-0.001689	1.000000	-0.009489	0.065280
Total Cost	0.374746	0.787905	0.774895	-0.009489	1.000000	0.804091
Total Profit	0.564550	0.557365	0.467214	0.065280	0.804091	1.000000

In [240…
```
plt.scatter(df["Units Sold"],df["Unit Cost"])
```

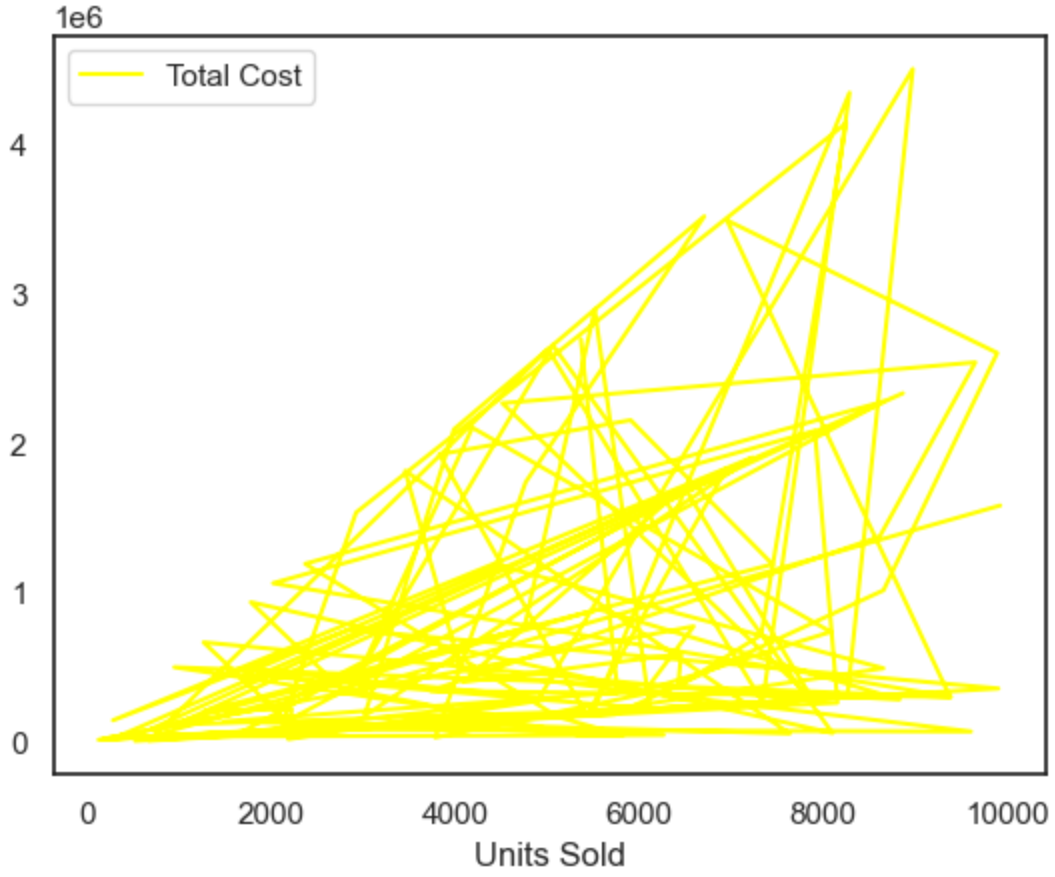
Out[240]:
```
<matplotlib.collections.PathCollection at 0x1e24661a670>
```



The above scatter plot implies that the two variables 'Units Sold' and 'Unit Cost' are inversely proportional to each other to some extent. When more units of a product are sold, the unit cost of that product becomes lesser and vice versa.

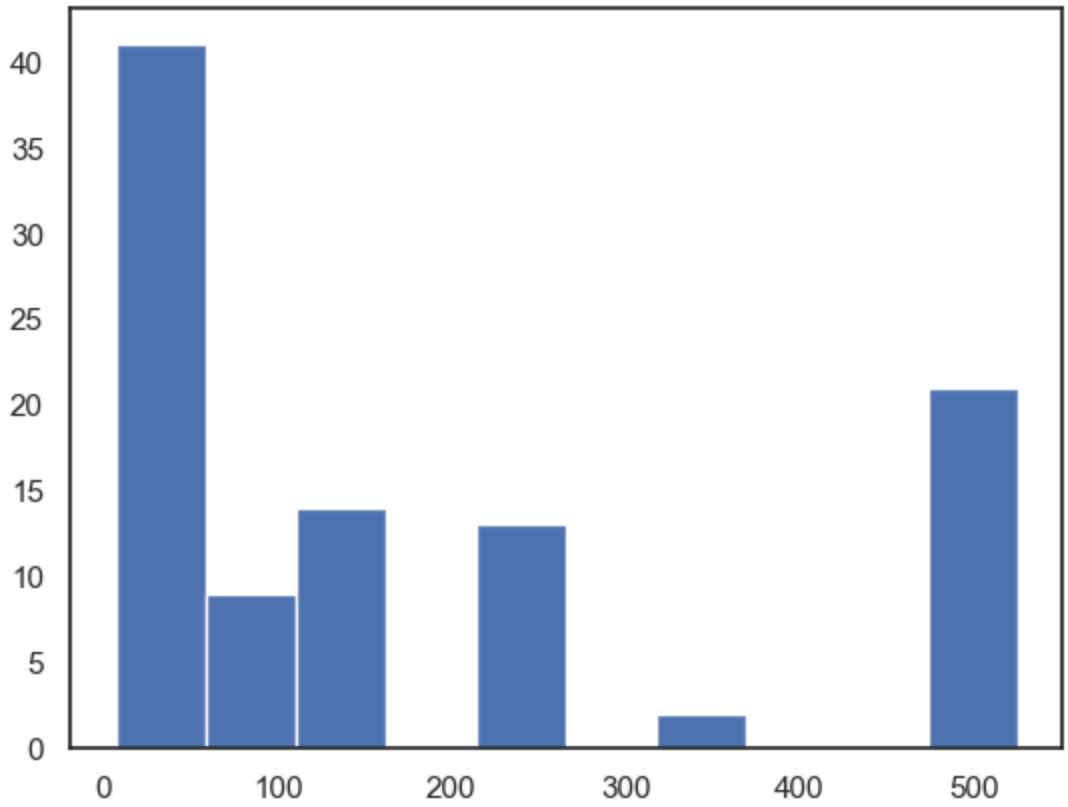
In [241…
```
df.plot.line(x="Units Sold",y="Total Cost",subplots=True, color={"Total Cost": "yellow
```

Out[241]:
```
array([<AxesSubplot:xlabel='Units Sold'>], dtype=object)
```



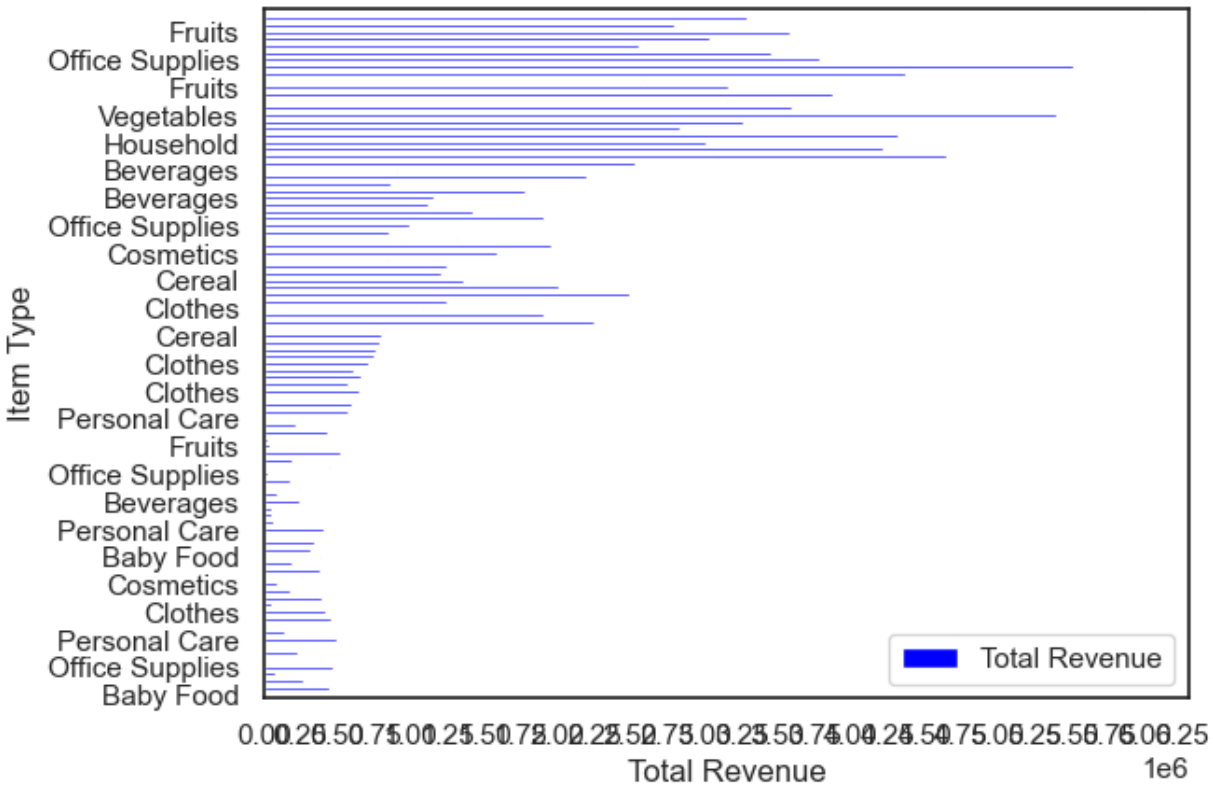
In [242…
```
plt.hist(df["Unit Cost"])
```

Out[242]:
```
(array([41.,  9., 14.,  0., 13.,  0.,  2.,  0.,  0., 21.]),
 array([  6.92 ,  58.724, 110.528, 162.332, 214.136, 265.94 , 317.744,
        369.548, 421.352, 473.156, 524.96 ]),
 <BarContainer object of 10 artists>)
```



In [243…
```
df.plot.barh(x="Item Type",y="Total Revenue",color="blue")
plt.locator_params(nbins=28)
plt.xlabel("Total Revenue")
```

Out[243]:
```
Text(0.5, 0, 'Total Revenue')
```

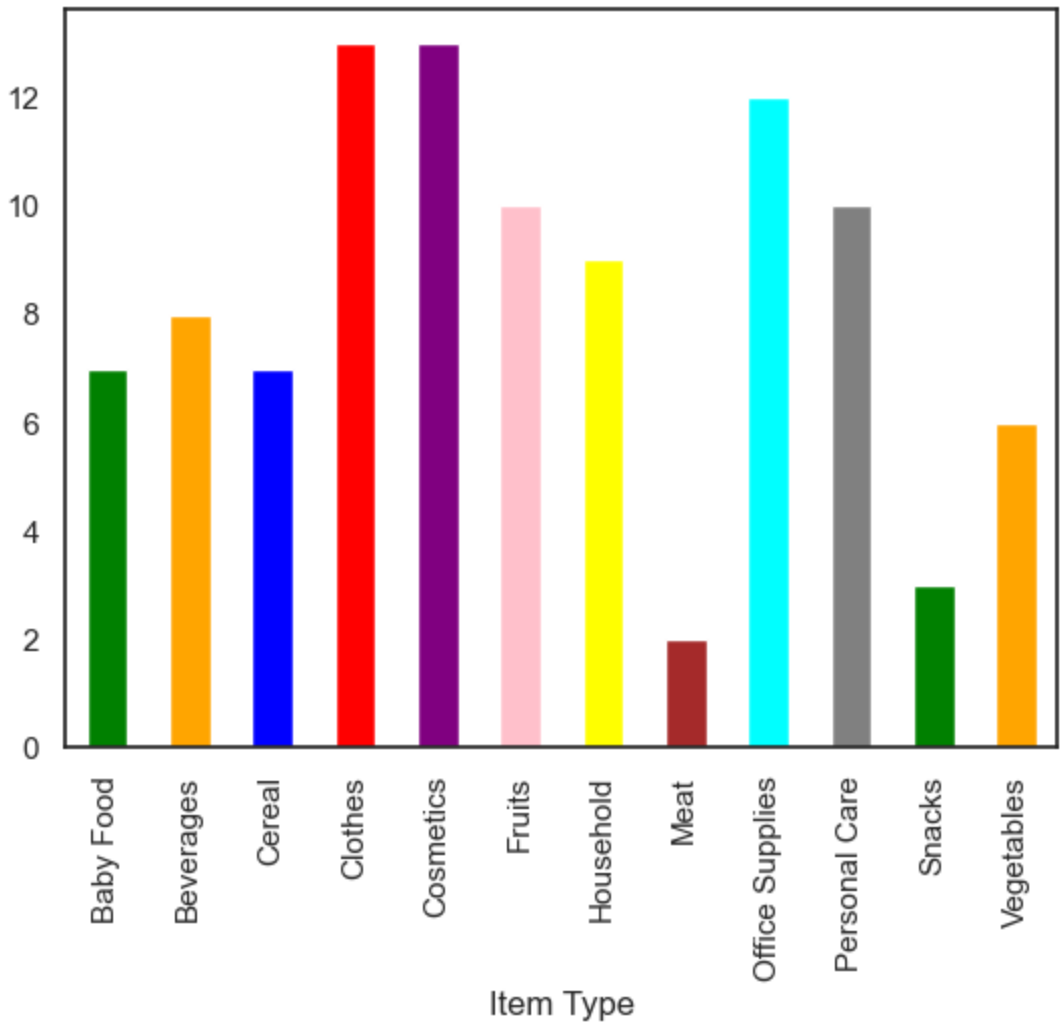


In [244…
```
df["Item Type"].unique()
```

Out[244]:
```
array(['Baby Food', 'Cereal', 'Office Supplies', 'Fruits', 'Household',
       'Vegetables', 'Personal Care', 'Clothes', 'Cosmetics', 'Beverages',
       'Meat', 'Snacks'], dtype=object)
```

In [245…
```
df.groupby("Item Type")["Total Revenue"].count().plot(kind="bar",color=["green","orang
```

Out[245]:
```
<AxesSubplot:xlabel='Item Type'>
```



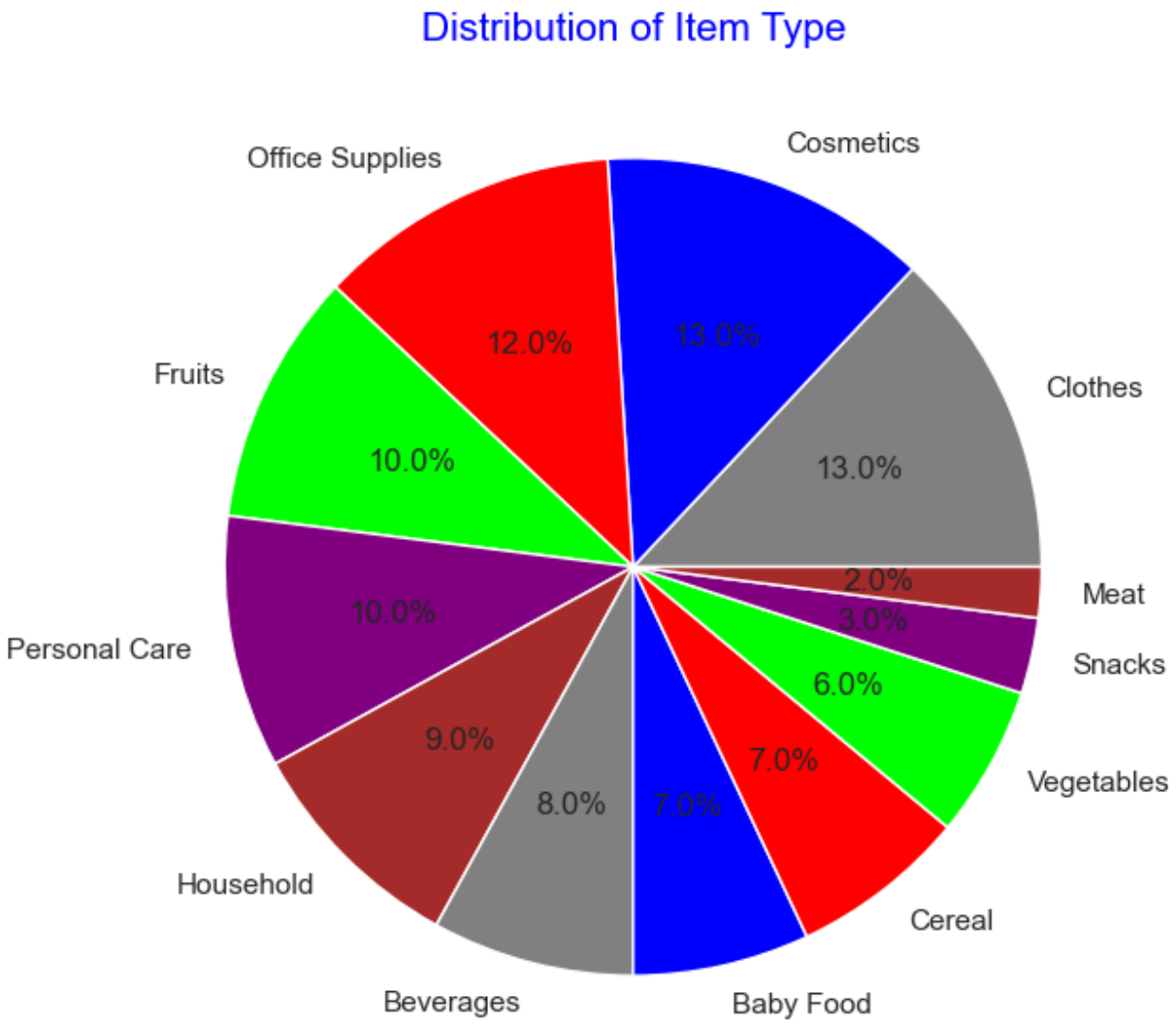
In [246…
```
df["Item Type"].dropna(inplace=True)
```

In [247…
```
labels =df["Item Type"].value_counts().index
```

In [248…
```
size = df["Item Type"].value_counts().values
colors =["grey","blue","red","lime","purple","brown"]
```

In [249…
```
plt.figure(figsize=(7,7))
plt.pie(size,labels=labels,colors=colors,autopct="%1.1f%%")
plt.title("Distribution of Item Type",fontsize=15,color="blue")
```

Out[249]:
```
Text(0.5, 1.0, 'Distribution of Item Type')
```



Clothes and cosmetics are the most purchased items while meat and snacks are the least purchased ones.

In [250…

```
prob =df.groupby(by=df.Country)["Total Profit","Country"].sum().reset_index().sort_val
prob
```

```
C:\Users\24\AppData\Local\Temp\ipykernel_14316\2669750807.py:1: FutureWarning: Indexi
ng with multiple keys (implicitly converted to a tuple of keys) will be deprecated, u
se a list instead.
  prob =df.groupby(by=df.Country)["Total Profit","Country"].sum().reset_index().sort_
values(by="Total Profit", ascending=False).head(10)
```

Out[250]:

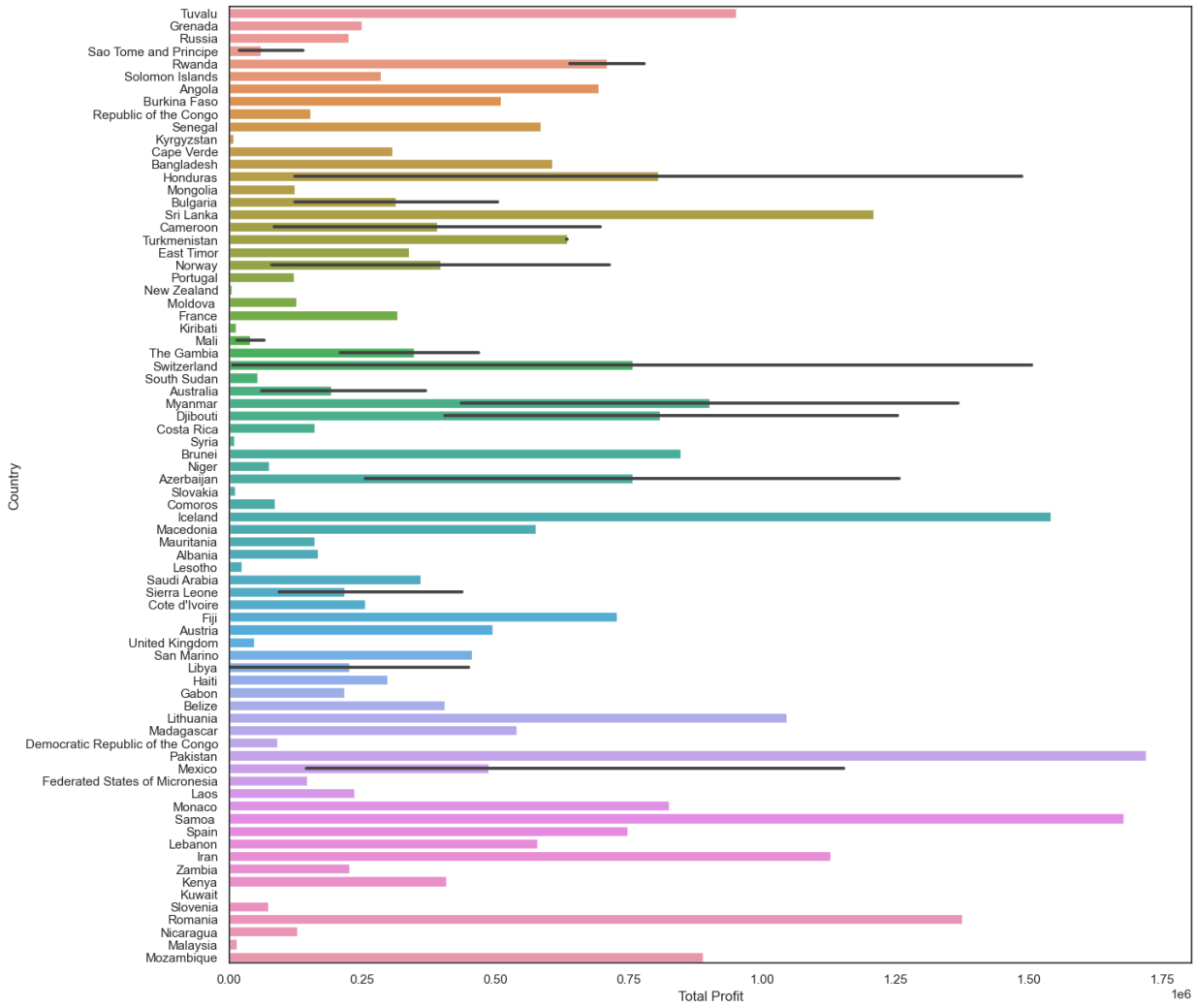
	Country	Total Profit
16	Djibouti	2425317.87
46	Myanmar	1802771.70
51	Pakistan	1719922.04
57	Samoa	1678540.98
24	Honduras	1609947.52
25	Iceland	1541705.29
4	Azerbaijan	1512926.83
69	Switzerland	1512729.45
41	Mexico	1457942.76
56	Rwanda	1417493.49

In [251…

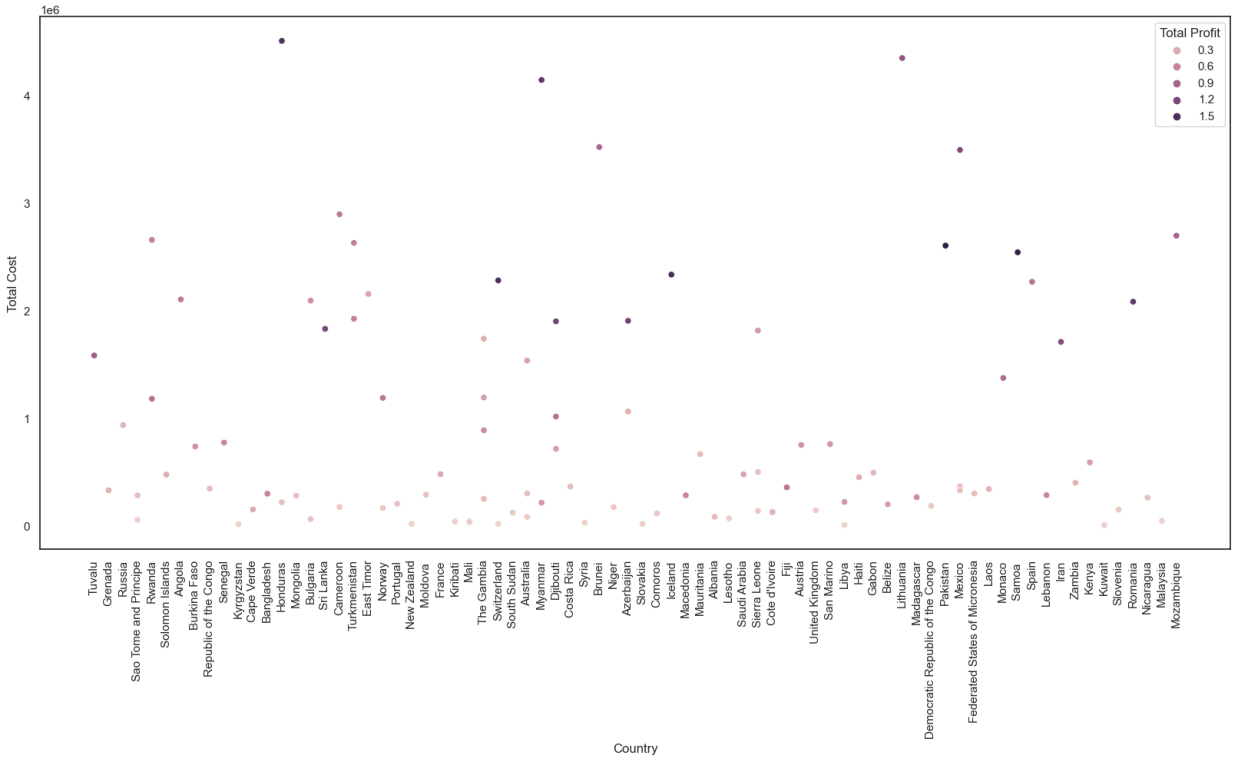
```
plt.figure(figsize=(15,15))
sns.barplot(x=df["Total Profit"],y=df["Country"],orient="h")
```

Out[251]:

```
<AxesSubplot:xlabel='Total Profit', ylabel='Country'>
```



In [252…
```
data_explore=df.copy()
df_Country=data_explore.copy()
plt.figure(figsize=(20,9))
sns.scatterplot(data=data_explore,x="Country",y="Total Cost",hue="Total Profit")
plt.xticks(rotation=90)
plt.show()
```



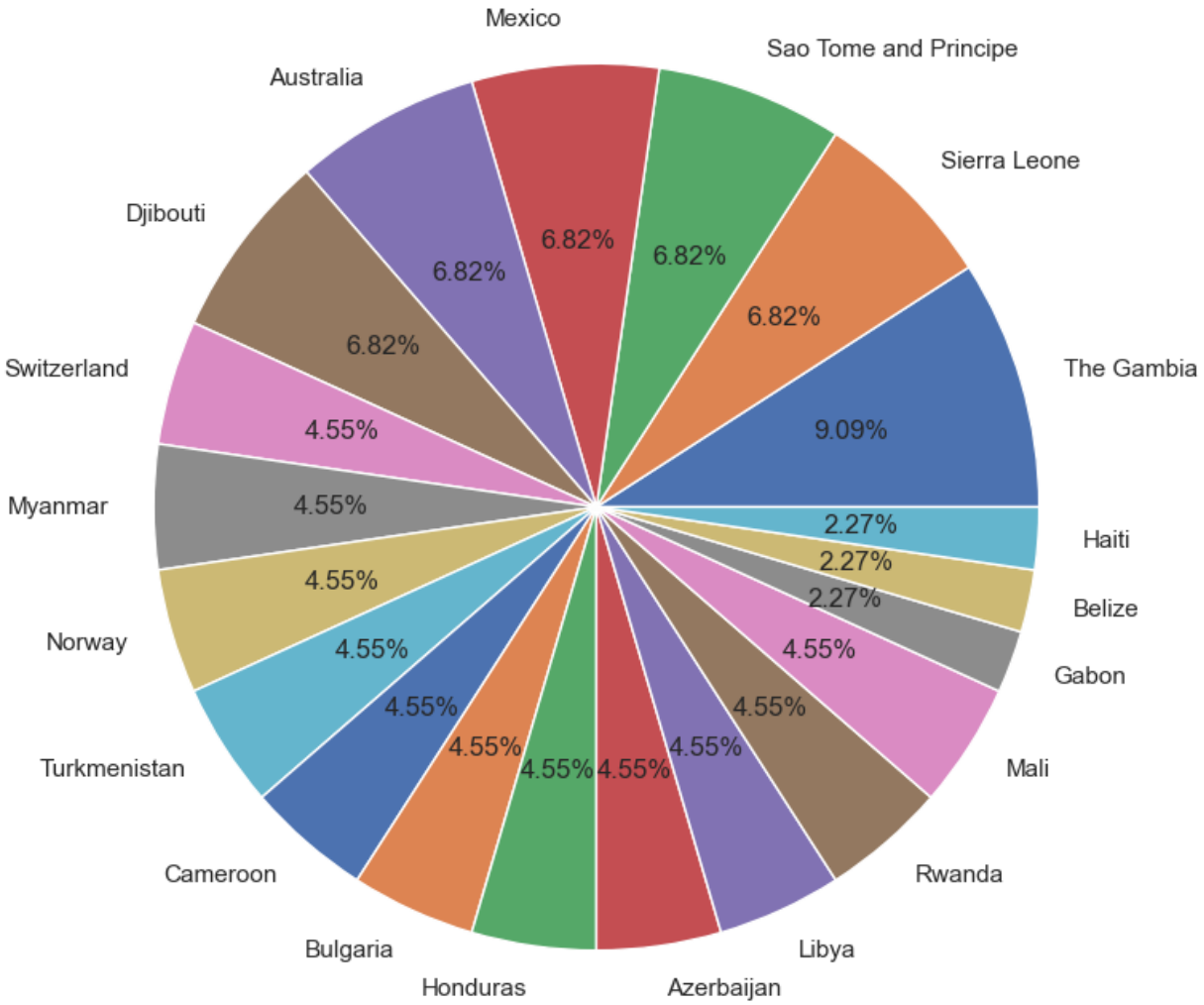
In [253…
```
df.isnull().sum()
```

Out[253]:
```
Region            0
Country           0
Item Type         0
Sales Channel     0
Order Priority    0
Order Date        0
Ship Date         0
Units Sold        0
Unit Price        0
Unit Cost         0
Total Revenue     0
Total Cost        0
Total Profit      0
dtype: int64
```

In [254…
```
df["Country"].value_counts()
```

Out[254]:
```
The Gambia               4
Sierra Leone             3
Sao Tome and Principe    3
Mexico                   3
Australia                3
                        ..
Comoros                  1
Iceland                  1
Macedonia                1
Mauritania               1
Mozambique               1
Name: Country, Length: 76, dtype: int64
```

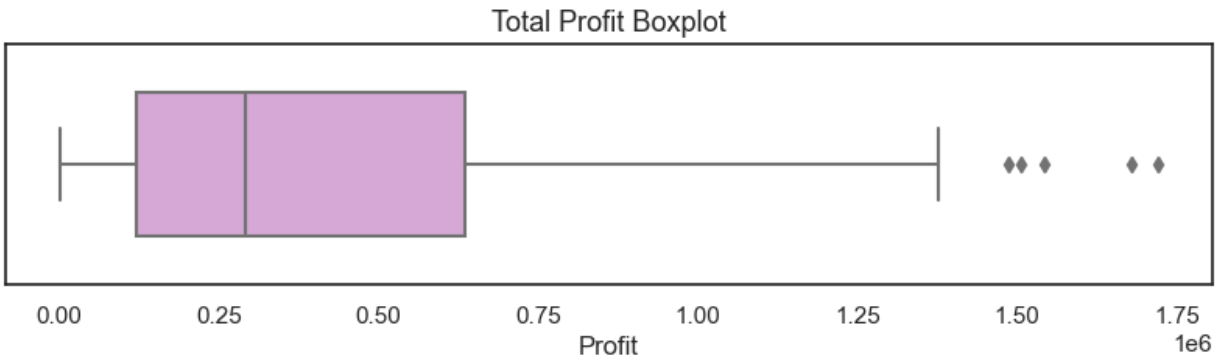
In [255…
```
import matplotlib.pyplot as plt
Country = df.Country.value_counts().index
country=df.Country.value_counts().values
fig,ax= plt.subplots(figsize=(9,9))
ax.pie(country_val[:20],labels=Country[:20],autopct="%1.2f%%")
plt.show()
```



In [256…
```
sns.set(style="white")
fig,ax=plt.subplots(figsize=(10, 2))
sns.boxplot(df["Total Profit"],color="plum", width=.6)
```

```
plt.title("Total Profit Boxplot", fontsize=13)
plt.xlabel("Profit")
plt.show()
```

```
C:\Users\24\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid posit
ional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  warnings.warn(
```

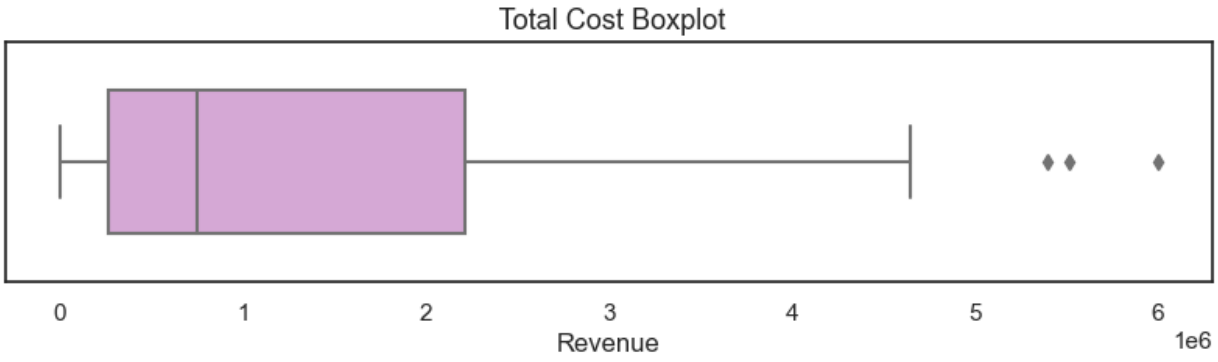


In [268…

```
sns.set(style="white")
fig,ax=plt.subplots(figsize=(10, 2))
sns.boxplot(df["Total Revenue"],color= "plum", width=.6)

plt.title(" Total Cost Boxplot", fontsize=13)
plt.xlabel("Revenue")
plt.show()
```

```
C:\Users\24\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pas
s the following variable as a keyword arg: x. From version 0.12, the only valid posit
ional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  warnings.warn(
```

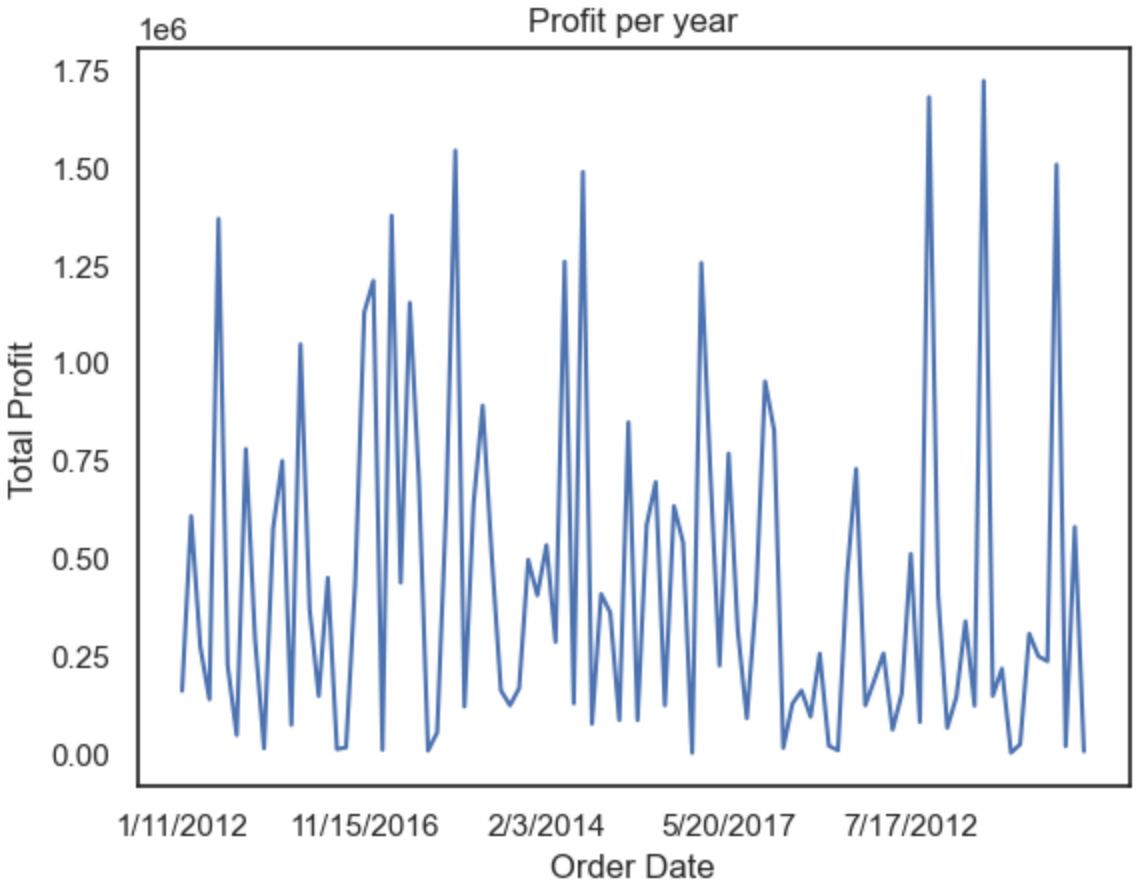


In [287…

```
df.groupby("Order Date")["Total Profit"].mean().plot()
plt.xlabel("Order Date")
plt.ylabel("Total Profit")
plt.title("Profit per year")
```

Out[287]:

```
Text(0.5, 1.0, 'Profit per year')
```


Profit per year
1e6
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Total Profit
1/11/2012
11/15/2016
2/3/2014
5/20/2017
7/17/2012
Order Date

In [ ]: