Amazon Sales Data Analysis

November 16, 2023

1 Amazon Sales Data Analysis

Sales management has gained importance to meet increasing competition and the need for improved methods of distribution to reduce cost and to increase profits. Sales management today is the most important function in a commercial and business enterprise.

Do ETL: Extract-Transform-Load some Amazon dataset and find for me

Sales-trend -> month-wise, year-wise, yearly_month-wise

Find key metrics and factors and show the meaningful relationships between

attributes. Do your own research and come up with your findings.

[1]:
```
#importing Nesesarry Libaries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df=pd.read_csv('Amazon Sales data.csv')
```

[3]:
```
df
```

[3]:

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012
2	Europe	Russia	Office Supplies	Offline	L	5/2/2014	341417157	5/8/2014
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits					
4	Sub-Saharan Africa	Rwanda	Office Supplies					
..	…	…	…					
95	Sub-Saharan Africa	Mali	Clothes					
96	Asia	Malaysia	Fruits					
97	Sub-Saharan Africa	Sierra Leone	Vegetables					
98	North America	Mexico	Personal Care					
99	Sub-Saharan Africa	Mozambique	Household					

```
3         Online              C   6/20/2014  514321792    7/5/2014
4        Offline              L    2/1/2013  115456712    2/6/2013
..           …            …          …         …          …
95        Online              M   7/26/2011  512878119    9/3/2011
96       Offline              L  11/11/2011  810711038  12/28/2011
97       Offline              C    6/1/2016  728815257   6/29/2016
98       Offline              M   7/30/2015  559427106    8/8/2015
99       Offline              L   2/10/2012  665095412   2/15/2012

    Units Sold  Unit Price  Unit Cost  Total Revenue  Total Cost  Total Profit
0         9925      255.28     159.42     2533654.00  1582243.50     951410.50
1         2804      205.70     117.11      576782.80   328376.44     248406.36
2         1779      651.21     524.96     1158502.59   933903.84     224598.75
3         8102        9.33       6.92       75591.66    56065.84      19525.82
4         5062      651.21     524.96     3296425.02  2657347.52     639077.50
..         …         …          …             …          …             …
95         888      109.28      35.84       97040.64    31825.92      65214.72
96        6267        9.33       6.92       58471.11    43367.64      15103.47
97        1485      154.06      90.93      228779.10   135031.05      93748.05
98        5767       81.73      56.67      471336.91   326815.89     144521.02
99        5367      668.27     502.54     3586605.09  2697132.18     889472.91

[100 rows x 14 columns]
```

[4]:
```
df.shape
```

[4]:
```
(100, 14)
```

We obserb data comprises of 100 rows and 14 columns

[5]:
```
df.describe
```

[5]:
```
<bound method NDFrame.describe of                                     Region
Country        Item Type  \
0               Australia and Oceania                Tuvalu        Baby Food
1   Central America and the Caribbean               Grenada           Cereal
2                              Europe                Russia  Office Supplies
3                  Sub-Saharan Africa  Sao Tome and Principe           Fruits
4                  Sub-Saharan Africa               Rwanda  Office Supplies
..                                …                   …              …
95                 Sub-Saharan Africa                 Mali          Clothes
96                               Asia              Malaysia           Fruits
97                 Sub-Saharan Africa          Sierra Leone       Vegetables
98                      North America                Mexico    Personal Care
99                 Sub-Saharan Africa            Mozambique        Household

   Sales Channel Order Priority  Order Date   Order ID   Ship Date  \
0        Offline              H   5/28/2010  669165933   6/27/2010
```

```
1         Online              C   8/22/2012  963881480   9/15/2012
2        Offline              L    5/2/2014  341417157    5/8/2014
3         Online              C   6/20/2014  514321792    7/5/2014
4        Offline              L    2/1/2013  115456712    2/6/2013
..           …            …          …         …          …
95        Online              M   7/26/2011  512878119    9/3/2011
96       Offline              L  11/11/2011  810711038  12/28/2011
97       Offline              C    6/1/2016  728815257   6/29/2016
98       Offline              M   7/30/2015  559427106    8/8/2015
99       Offline              L   2/10/2012  665095412   2/15/2012

    Units Sold  Unit Price  Unit Cost  Total Revenue  Total Cost  Total Profit
0         9925      255.28     159.42     2533654.00  1582243.50     951410.50
1         2804      205.70     117.11      576782.80   328376.44     248406.36
2         1779      651.21     524.96     1158502.59   933903.84     224598.75
3         8102        9.33       6.92       75591.66    56065.84      19525.82
4         5062      651.21     524.96     3296425.02  2657347.52     639077.50
..         …         …          …             …          …             …
95         888      109.28      35.84       97040.64    31825.92      65214.72
96        6267        9.33       6.92       58471.11    43367.64      15103.47
97        1485      154.06      90.93      228779.10   135031.05      93748.05
98        5767       81.73      56.67      471336.91   326815.89     144521.02
99        5367      668.27     502.54     3586605.09  2697132.18     889472.91

[100 rows x 14 columns]>
```

[6]:
```
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
```

[7]:
```
df.isnull().sum()#Checking for null values
```

[7]:
```
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
```

[58]:
```
#Transforming  data to colum to date timer format as it help the date to␣
 ↪extract month and year as it is in object data type
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['Ship Date'] = pd.to_datetime(df['Ship Date'])
```

[59]:
```
df.info()# in here we can see that the order date and Ship date has been␣
 ↪convert from object to date
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 17 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    datetime64[ns]
 6   Order ID        100 non-null    int64
 7   Ship Date       100 non-null    datetime64[ns]
 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
 14  Month           100 non-null    int64
```

```
 15  Year            100 non-null    int64
 16  Year_Month      100 non-null    period[M]
dtypes: datetime64[ns](2), float64(5), int64(4), object(5), period[M](1)
memory usage: 13.4+ KB
```

[10]:
```
#Extraction of Month, Year and Month_year from the Date to analyse
df['Month'] = df['Order Date'].dt.month
df['Year'] = df['Order Date'].dt.year
df['Year_Month'] = df['Order Date'].dt.to_period('M')
```

2 Data Analysis

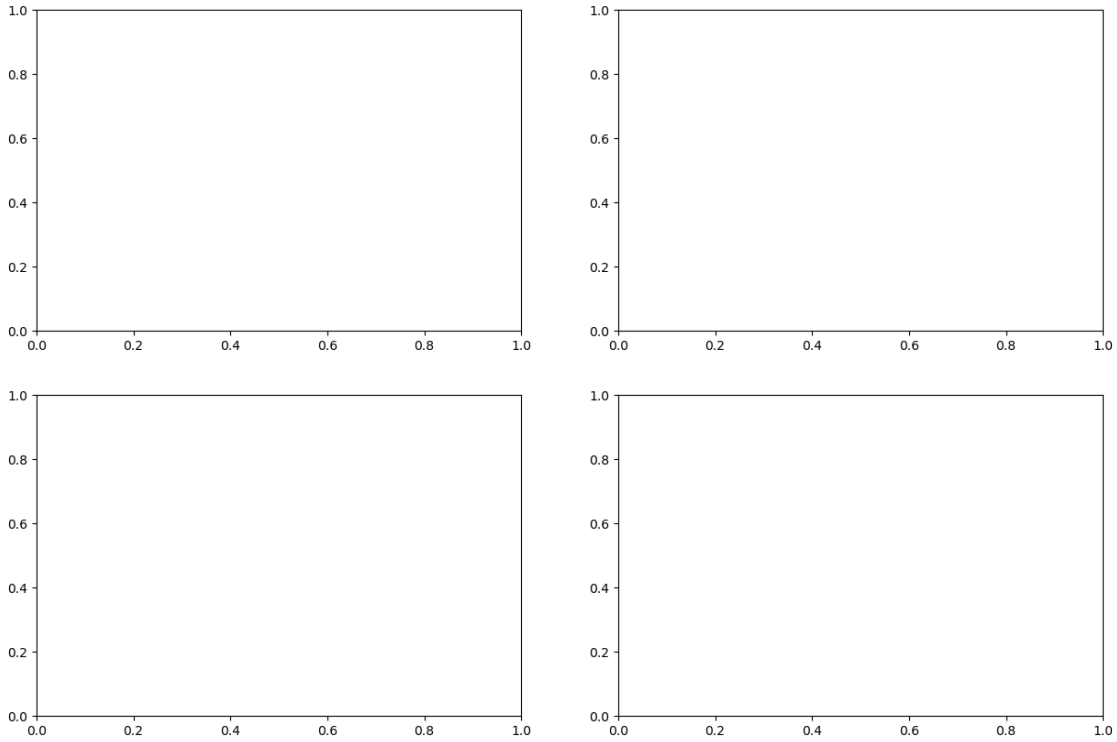
[11]:
```
# Monthly Sales Trends
monthly_sales = df.groupby('Month')['Total Profit'].sum()
```

[12]:
```
# Yearly sales trend
yearly_sales = df.groupby('Year')['Total Profit'].sum()
```

[13]:
```
# Yearly month-wise sales trend
yearly_monthly_sales = df.groupby('Year_Month')['Total Profit'].sum()
```

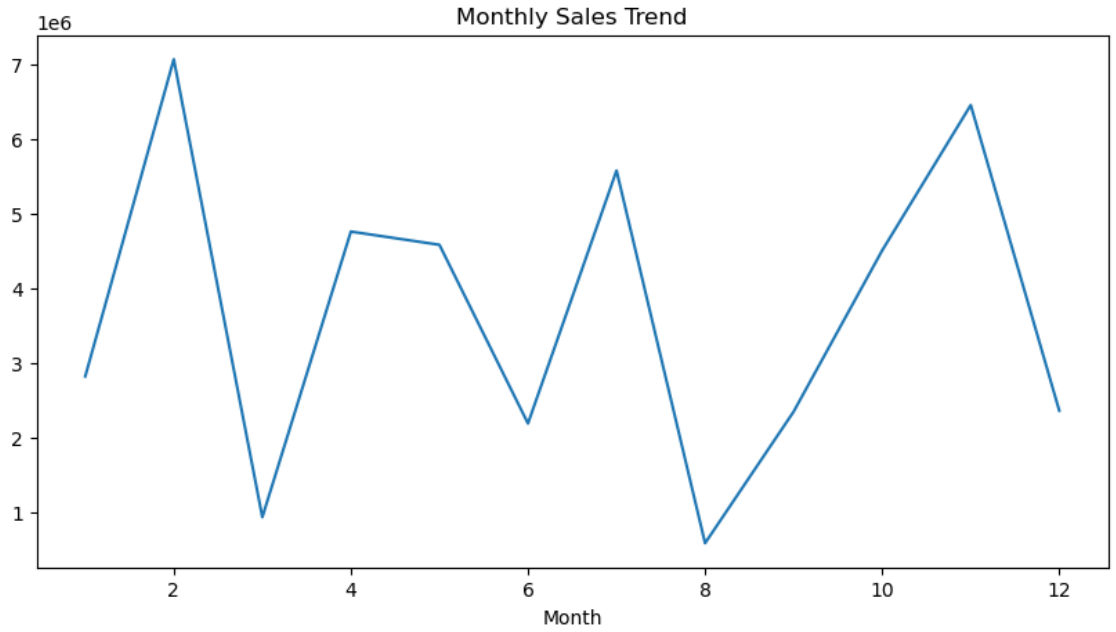
[14]:
```
# Key Metrics
avg_order_value = df['Total Profit'].mean()
total_orders = df['Order ID'].nunique()
```

[15]:
```
# Visualization
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(15, 10))
```



[16]:
```
# Monthly sales trend
plt.figure(figsize=(10, 5))
sns.lineplot(x=monthly_sales.index, y=monthly_sales.values)
plt.title('Monthly Sales Trend')
```

[16]: Text(0.5, 1.0, 'Monthly Sales Trend')



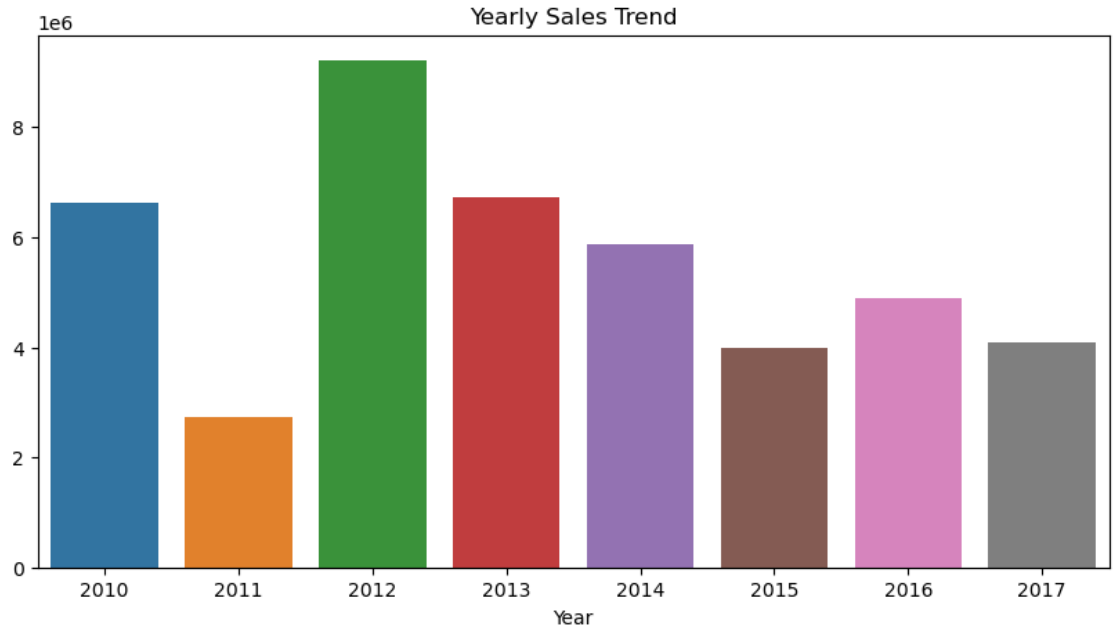
3 Observation

We obserb that in Febuary sales were good

and lowest in the month August

[17]:
```
# Yearly sales trend
plt.figure(figsize=(10,5))
sns.barplot(x=yearly_sales.index, y=yearly_sales.values)
plt.title('Yearly Sales Trend')
```

[17]:
```
Text(0.5, 1.0, 'Yearly Sales Trend')
```

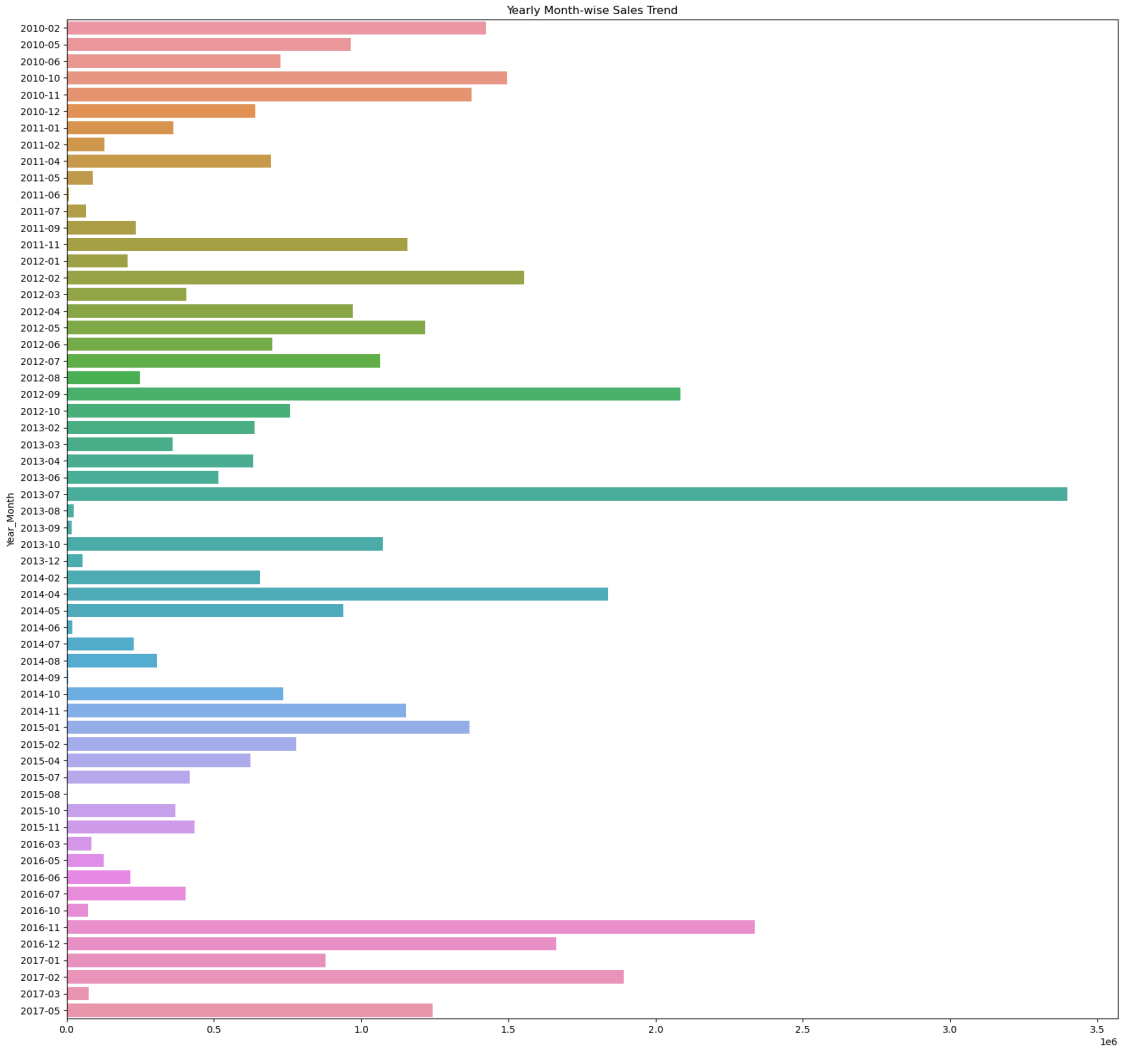


4 Observation

IN 2012 sales were Higest

in 2011 sales were lowest

[57]:
```
# Yearly_monthyly wise
plt.figure(figsize=(22, 19))
yr = sns.barplot(y=yearly_monthly_sales.index.astype(str),␣
 ↪x=yearly_monthly_sales.values, orient='h')
yr.set_yticklabels(yr.get_yticklabels(), rotation=0)
plt.subplots_adjust(left=0.2)
plt.title('Yearly Month-wise Sales Trend')
plt.show()
```



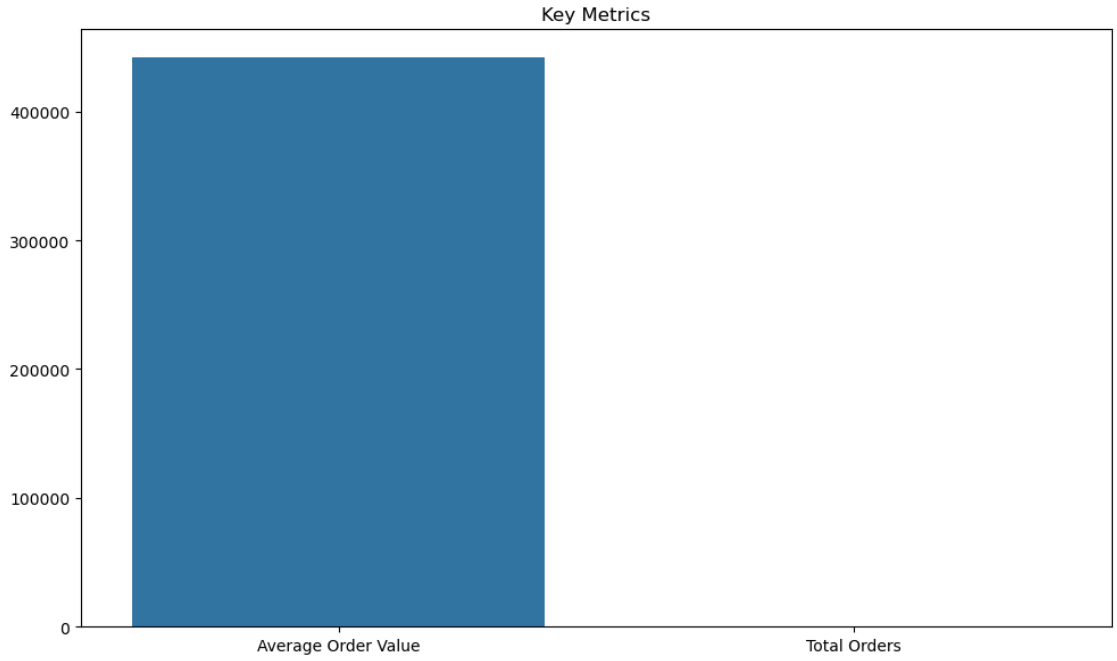
[ ]:

5 Observation

We obserb that in july yr of 2013 Sales were higest through out the year

and in 2011-06 sales were lowest through out the year

[61]:
```
# Key metrics
plt.figure(figsize=(10,6))
sns.barplot(x=['Average Order Value', 'Total Orders'], y=[avg_order_value,␣
  ↪total_orders])
plt.title('Key Metrics')
plt.tight_layout()
plt.show()
```


Key Metrics
400000
300000
200000
100000
0
Average Order Value
Total Orders

[ ]:

[ ]: