SalesAnalysis

August 18, 2025

[1]:
```
#Import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
#Dataset Loading
df=pd.read_csv("sales_data.csv")
print("Dataset Loaded Successfully")
df.head(10)
```

```
Dataset Loaded Successfully
```

[2]:

	Date	Day	Month	Year	Customer_Age	Age_Group	\
0	2013-11-26	26	November	2013	19	Youth (<25)	
1	2015-11-26	26	November	2015	19	Youth (<25)	
2	2014-03-23	23	March	2014	49	Adults (35-64)	
3	2016-03-23	23	March	2016	49	Adults (35-64)	
4	2014-05-15	15	May	2014	47	Adults (35-64)	
5	2016-05-15	15	May	2016	47	Adults (35-64)	
6	2014-05-22	22	May	2014	47	Adults (35-64)	
7	2016-05-22	22	May	2016	47	Adults (35-64)	
8	2014-02-22	22	February	2014	35	Adults (35-64)	
9	2016-02-22	22	February	2016	35	Adults (35-64)	

	Customer_Gender	Country	State	Product_Category	Sub_Category	\
0	M	Canada	British Columbia	Accessories	Bike Racks	
1	M	Canada	British Columbia	Accessories	Bike Racks	
2	M	Australia	New South Wales	Accessories	Bike Racks	
3	M	Australia	New South Wales	Accessories	Bike Racks	
4	F	Australia	New South Wales	Accessories	Bike Racks	
5	F	Australia	New South Wales	Accessories	Bike Racks	
6	F	Australia	Victoria	Accessories	Bike Racks	
7	F	Australia	Victoria	Accessories	Bike Racks	
8	M	Australia	Victoria	Accessories	Bike Racks	
9	M	Australia	Victoria	Accessories	Bike Racks	

	Product	Order_Quantity	Unit_Cost	Unit_Price	Profit	Cost	\

```
0  Hitch Rack - 4-Bike               8         45         120      590   360
1  Hitch Rack - 4-Bike               8         45         120      590   360
2  Hitch Rack - 4-Bike              23         45         120     1366  1035
3  Hitch Rack - 4-Bike              20         45         120     1188   900
4  Hitch Rack - 4-Bike               4         45         120      238   180
5  Hitch Rack - 4-Bike               5         45         120      297   225
6  Hitch Rack - 4-Bike               4         45         120      199   180
7  Hitch Rack - 4-Bike               2         45         120      100    90
8  Hitch Rack - 4-Bike              22         45         120     1096   990
9  Hitch Rack - 4-Bike              21         45         120     1046   945

   Revenue
0      950
1      950
2     2401
3     2088
4      418
5      522
6      379
7      190
8     2086
9     1991
```

[3]:
```
#Basic EDA
print("Information of dataset:")
df.info()
```

```
Information of dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 113036 entries, 0 to 113035
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Date              113036 non-null  object
 1   Day               113036 non-null  int64
 2   Month             113036 non-null  object
 3   Year              113036 non-null  int64
 4   Customer_Age      113036 non-null  int64
 5   Age_Group         113036 non-null  object
 6   Customer_Gender   113036 non-null  object
 7   Country           113036 non-null  object
 8   State             113036 non-null  object
 9   Product_Category  113036 non-null  object
 10  Sub_Category      113036 non-null  object
 11  Product           113036 non-null  object
 12  Order_Quantity    113036 non-null  int64
 13  Unit_Cost         113036 non-null  int64
 14  Unit_Price        113036 non-null  int64
```

```
 15  Profit            113036 non-null  int64
 16  Cost              113036 non-null  int64
 17  Revenue           113036 non-null  int64
dtypes: int64(9), object(9)
memory usage: 15.5+ MB
```

[4]:
```
print("Description of dataset:")
df.describe()
```

```
Description of dataset:
```

[4]:

	Day	Year	Customer_Age	Order_Quantity	\
count	113036.000000	113036.000000	113036.000000	113036.000000	
mean	15.665753	2014.401739	35.919212	11.901660	
std	8.781567	1.272510	11.021936	9.561857	
min	1.000000	2011.000000	17.000000	1.000000	
25%	8.000000	2013.000000	28.000000	2.000000	
50%	16.000000	2014.000000	35.000000	10.000000	
75%	23.000000	2016.000000	43.000000	20.000000	
max	31.000000	2016.000000	87.000000	32.000000	

	Unit_Cost	Unit_Price	Profit	Cost	\
count	113036.000000	113036.000000	113036.000000	113036.000000	
mean	267.296366	452.938427	285.051665	469.318695	
std	549.835483	922.071219	453.887443	884.866118	
min	1.000000	2.000000	-30.000000	1.000000	
25%	2.000000	5.000000	29.000000	28.000000	
50%	9.000000	24.000000	101.000000	108.000000	
75%	42.000000	70.000000	358.000000	432.000000	
max	2171.000000	3578.000000	15096.000000	42978.000000	

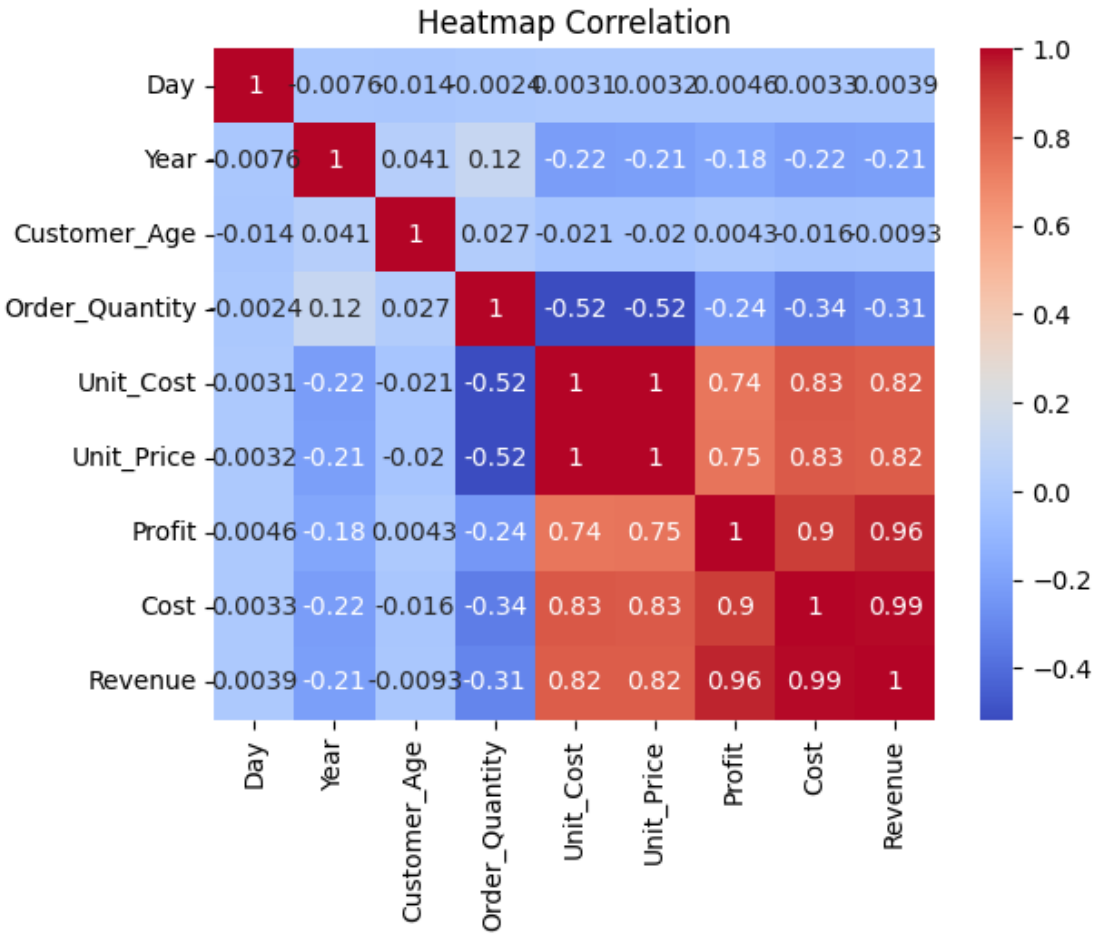
	Revenue
count	113036.000000
mean	754.370360
std	1309.094674
min	2.000000
25%	63.000000
50%	223.000000
75%	800.000000
max	58074.000000

[5]:
```
#Check missing values
df.isnull().sum()
```

[5]:
```
Date                0
Day                 0
Month               0
Year                0
```

```
Customer_Age          0
Age_Group             0
Customer_Gender       0
Country               0
State                 0
Product_Category      0
Sub_Category          0
Product               0
Order_Quantity        0
Unit_Cost             0
Unit_Price            0
Profit                0
Cost                  0
Revenue               0
dtype: int64
```

[6]:
```
Corr=df.corr(numeric_only=True)
plt.title("Heatmap Correlation")
sns.heatmap(Corr,annot=True,cmap='coolwarm')
plt.show()
```



[7]:
```
#KPI's Computation
#Convert to numeric
df['Order_Quantity']=pd.to_numeric(df['Order_Quantity'])
df['Unit_Price']=pd.to_numeric(df['Unit_Price'])
df['Cost']=pd.to_numeric(df['Cost'])
```

[8]:
```
#Total Revenue
df['Total_Revenue']=df['Order_Quantity'] * df['Unit_Price']
total_revenue=df['Total_Revenue'].sum()
print("Total Revenue is:",total_revenue)
```

```
Total Revenue is: 95176318
```

[9]:
```
#Total Profit
df['Profit']=df['Revenue'] - df['Cost']
total_profit=df['Profit'].sum()
print("Total Profit is:",total_profit)
```

```
Total Profit is: 32221100
```

[10]:
```
#Average Order Value
Avg=total_revenue/df['Order_Quantity']
print("Average of Orders are:",Avg)
```

```
Average of Orders are: 0         1.189704e+07
1         1.189704e+07
2         4.138101e+06
3         4.758816e+06
4         2.379408e+07
              …
113031    3.172544e+07
113032    4.326196e+06
113033    4.326196e+06
113034    3.965680e+06
113035    4.138101e+06
Name: Order_Quantity, Length: 113036, dtype: float64
```

[11]:
```
#Profit Margin
margin=(total_profit/total_revenue) *100
print("Profit Margin is:",margin)
```

```
Profit Margin is: 33.85411484398881
```

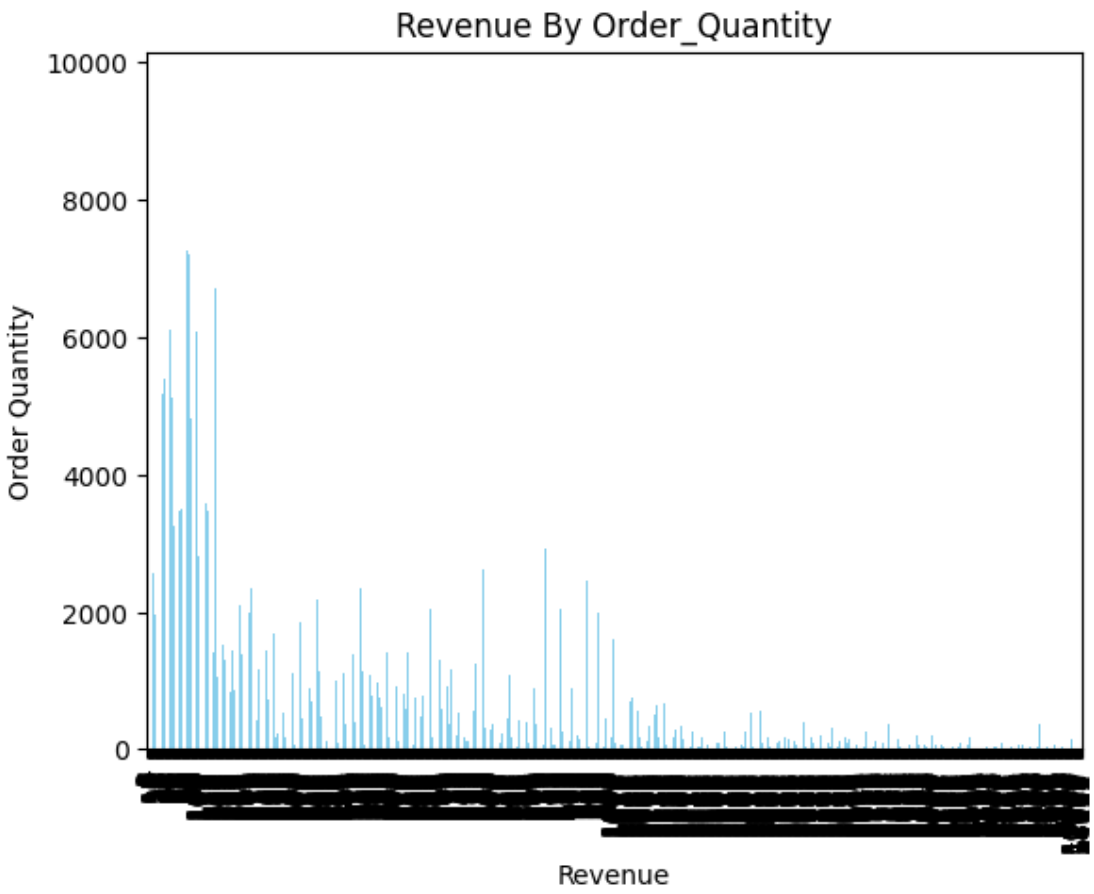
[12]:
```
#Sales of Unit
Unit_sold=sum(df['Order_Quantity'])
print("Units Sold are:",Unit_sold)
```

```
Units Sold are: 1345316
```

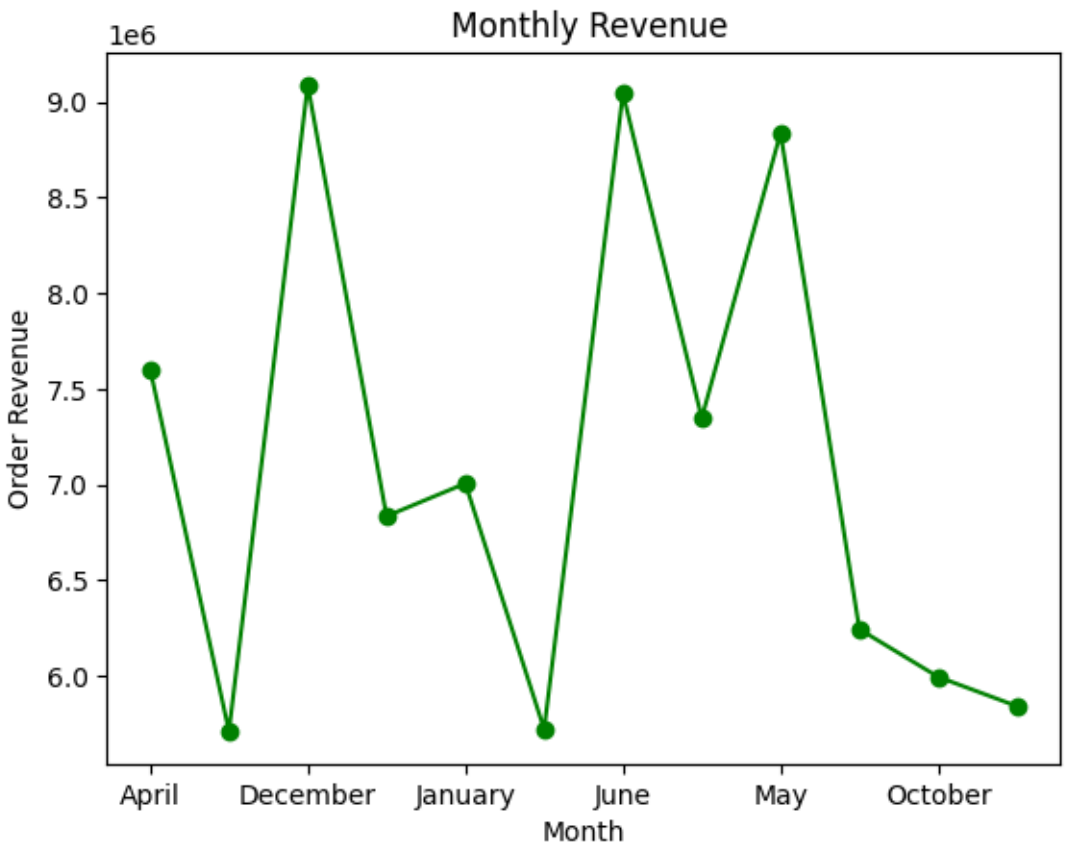
[13]:
```
#Orders per Customers
orders=df['Order_Quantity']/df['Customer_Age']
print("Orders per Customers are:",orders)
```

```
Orders per Customers are: 0         0.421053
1         0.421053
2         0.469388
3         0.408163
4         0.085106
            …
113031    0.073171
113032    1.222222
113033    1.222222
113034    0.648649
113035    0.621622
Length: 113036, dtype: float64
```

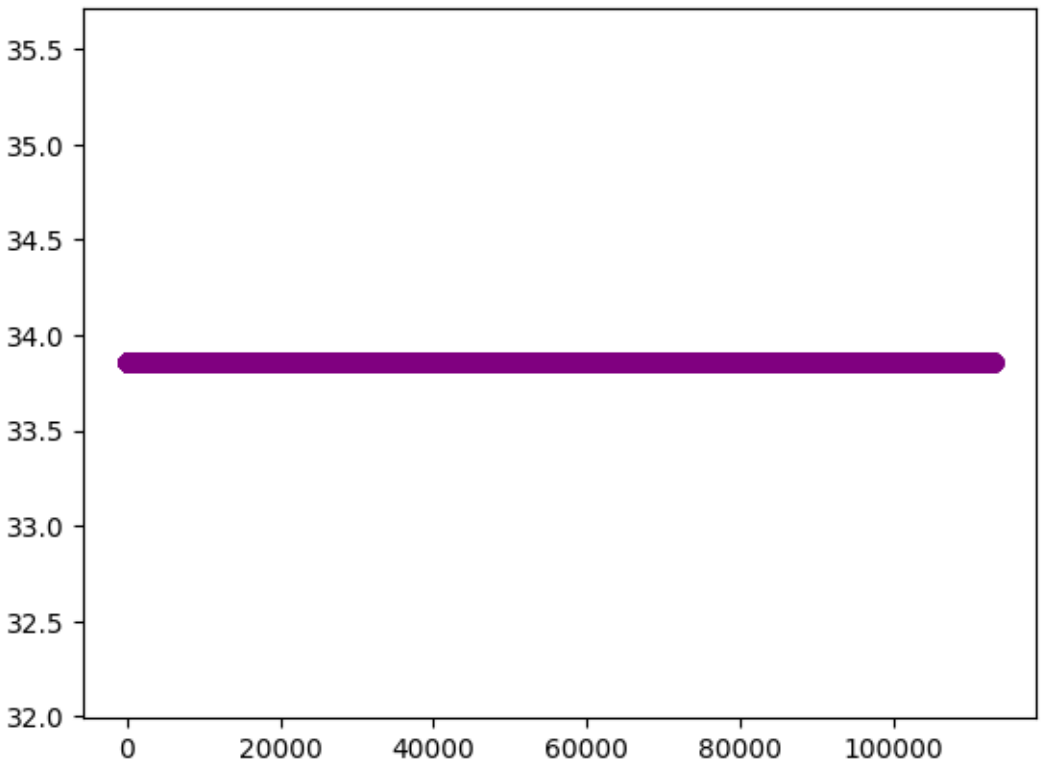
[14]:
```
#Visualization
#Revenue By OrderQuantity
grp=df.groupby("Revenue")["Order_Quantity"].sum()
grp.plot(kind='bar',color='skyblue')
plt.title("Revenue By Order_Quantity")
plt.xlabel("Revenue")
plt.ylabel("Order Quantity")
plt.show()
```



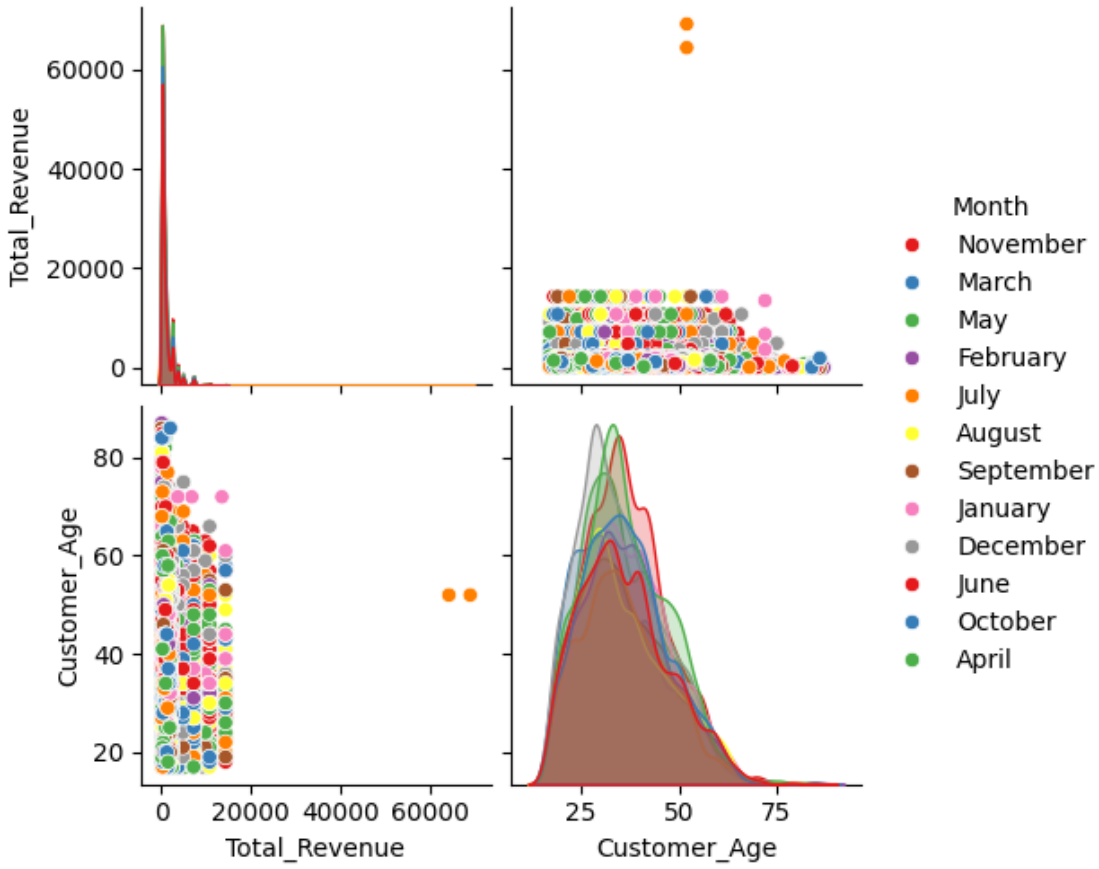
[15]:
```
#Monthly Revenue
grp=df.groupby("Month")['Revenue'].sum()
grp.plot(marker='o',color='green')
plt.title(" Monthly Revenue")
plt.xlabel("Month")
plt.ylabel("Order Revenue")
plt.show()
```



[28]:
```
#Profit margin by product category
df['margin']=(total_profit/total_revenue) *100
grp=df.groupby("Product_Category")["margin"]
grp.plot(marker='o',color='purple')
plt.show()
```



[29]:
```
#Visualize mupltiple columns
#Revenue by Customer_Age or Month
sns.pairplot(df[['Total_Revenue','Customer_Age','Month']], hue='Month',␣
 ↪palette='Set1')
plt.show()
```


Total_Revenue
60000
40000
20000
0
Customer_Age
80
60
40
20
0
20000
40000
60000
Total_Revenue
25
50
75
Customer_Age
Month
November
March
May
February
July
August
September
January
December
June
October
April

[ ]: