# E- COMMERCE PROJECT

-BY

R. SUKANYA

Date of Submission: 10th September, 2023

# INDEX

## 1. Analysis of Data

First and foremost, the dataset needs to be analysed. Data analysis involves cleaning, changing, and processing raw data and extracting useful information that can help the businesses to take informed decisions.

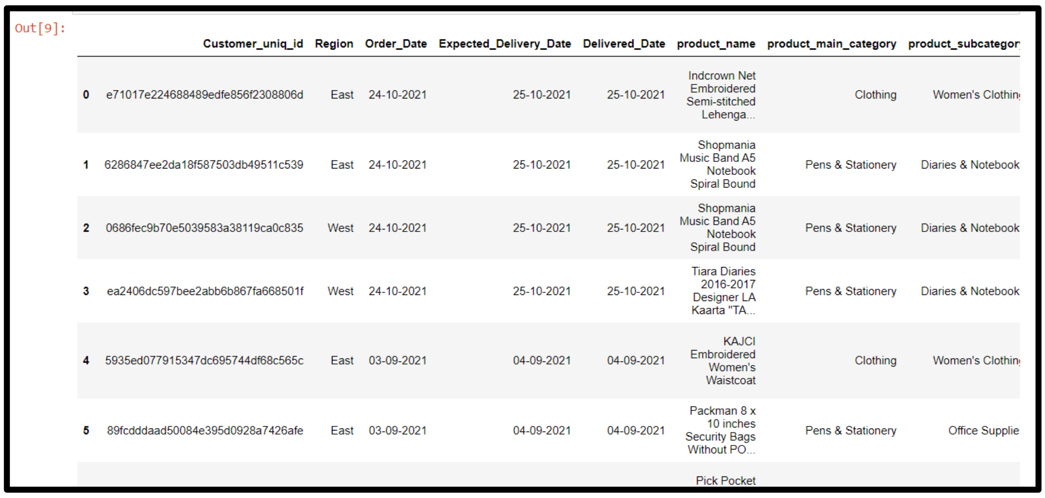
It is most important to get familiar with the dataset. For this, open the E-Commerce.csv file in Excel and look at the various columns that make up the dataset and their features. In order to do the same in python, we will first read the E-Commerce dataset and load it in a dataset named ECom_Data.

Before issuing any commands, all the necessary libraries need to be imported. For our analysis, numpy, pandas, matplotlib and seaborn have to be imported.

### a) First ten records and Last ten records of the dataset:

We print the first ten records and the last ten records of the E-Commerce dataset using the head and the tail commands respectively.

***Output:***



Out[9]:

| | Customer_uniq_id | Region | Order_Date | Expected_Delivery_Date | Delivered_Date | product_name | product_main_category | product_subcategor |
|---|---|---|---|---|---|---|---|---|
| **0** | e71017e224688489edfe856f2308806d | East | 24-10-2021 | 25-10-2021 | 25-10-2021 | Indcrown Net Embroidered Semi-stitched Lehenga... | Clothing | Women's Clothin |
| **1** | 6286847ee2da18f587503db49511c539 | East | 24-10-2021 | 25-10-2021 | 25-10-2021 | Shopmania Music Band A5 Notebook Spiral Bound | Pens & Stationery | Diaries & Notebook |
| **2** | 0686fec9b70e5039583a38119ca0c835 | West | 24-10-2021 | 25-10-2021 | 25-10-2021 | Shopmania Music Band A5 Notebook Spiral Bound | Pens & Stationery | Diaries & Notebook |
| **3** | ea2406dc597bee2abb6b867fa668501f | West | 24-10-2021 | 25-10-2021 | 25-10-2021 | Tiara Diaries 2016-2017 Designer LA Kaarta "TA... | Pens & Stationery | Diaries & Notebook |
| **4** | 5935ed077915347dc695744df68c565c | East | 03-09-2021 | 04-09-2021 | 04-09-2021 | KAJCI Embroidered Women's Waistcoat | Clothing | Women's Clothin |
| **5** | 89fcdddaad50084e395d0928a7426afe | East | 03-09-2021 | 04-09-2021 | 04-09-2021 | Packman 8 x 10 inches Security Bags Without PO... | Pens & Stationery | Office Supplie |
| | | | | | | Pick Pocket | | |

### b. No. of rows and columns in the dataset:

There are two ways to find the number of rows and columns in the dataset.

1) Use the shape command to get the number of rows and columns. The shape() method gives the dimensions of a Pandas dataset.

2) Use the length function with axis = 0 for number of rows and axis = 1 for number of columns in the dataset.

***Output:***

```
(8906, 17)
```

***There are 8906 rows and 17 columns in the E-Commerce dataset.***

**c. To find the number of object datatypes in the given dataset:**

The dtypes() function is used to find the datatype of a column in the dataset. We can use dtypes.value_counts() to get the number of different datatypes in the dataset. In the E-Commerce dataset, there are 14 object datatypes in the dataset.

***Output:***

```
object     14
int64       2
float64     1
dtype: int64
```

**d. To check if Boolean datatype is in the dataset:**

The dtypes() function can be used to find all the different datatypes in the dataset like int, float, object, Boolean etc.

***Output: There is no Boolean datatype in the dataset.***

## 2. Altering the dataset

### a. Dropping columns:

In order to eliminate the non-informative columns from the dataset, the drop function is used. The drop function explicitly specifies the column to be deleted. The axis=1 in the drop function suggests that a column has to be deleted. However, the column is temporarily deleted in this way is still present in the original dataset.

### b. Dropping columns permanently:

In order to delete the columns permanently in the dataset, inplace=True has to be specified in the drop() function. When a column is deleted in this way, it is permanently deleted from the dataset. On describing the dataset now, you will not find this column anymore.

***Output: The columns product_specifications and description have been permanently deleted by using the drop() function with inplace=True.***

## 3. Summarizing data

### a. Number of unique brands:

In order to get the number of unique brands in the dataset, use the nunique() function on the Brand column of the Ecom_Data dataset.

***Output:***

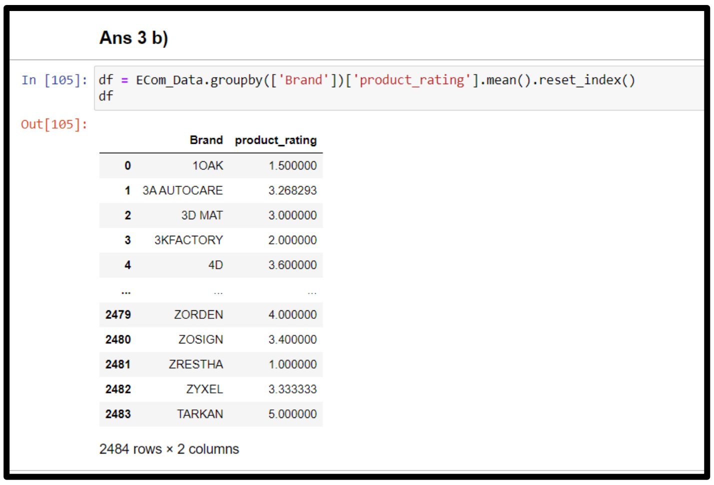
***2484***

***There are 2484 unique brands in the dataset.***

### b. Average product rating within each brand:

The average product rating within each brand can be found by grouping the two columns, namely, Brand and product_rating and taking the mean of the product ratings of each brand. This will give you the required average rating of each brand.

***Output:***



Ans 3 b)

```
In [105]: df = ECom_Data.groupby(['Brand'])['product_rating'].mean().reset_index()
          df
```

Out[105]:

|  | Brand | product_rating |
|---|---|---|
| 0 | 1OAK | 1.500000 |
| 1 | 3A AUTOCARE | 3.268293 |
| 2 | 3D MAT | 3.000000 |
| 3 | 3KFACTORY | 2.000000 |
| 4 | 4D | 3.600000 |
| ... | ... | ... |
| 2479 | ZORDEN | 4.000000 |
| 2480 | ZOSIGN | 3.400000 |
| 2481 | ZRESTHA | 1.000000 |
| 2482 | ZYXEL | 3.333333 |
| 2483 | TARKAN | 5.000000 |

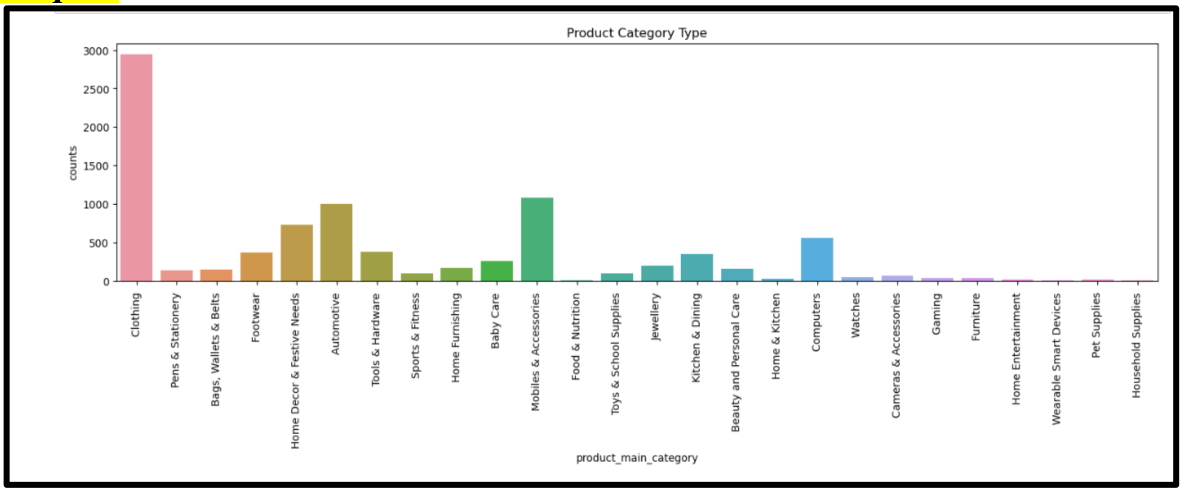
2484 rows × 2 columns

## 4. Main Categories of Products

### a. Count of items ordered for each product_main_category:

The count of items ordered for each product can be found out by drawing the bar chart. For this, countplot() can be used. The plot can be made with product_main_category on the x-axis while the counts will be on the y-axis. For the labels to be very clear on the x-axis, use the xticks() function with rotation = $90^0$. The bar graph should be customized by giving an appropriate title by using the title() command. Specify the scale by using the figsize() function.

***Output:***



**b. Maximum orders and Minimum orders:**

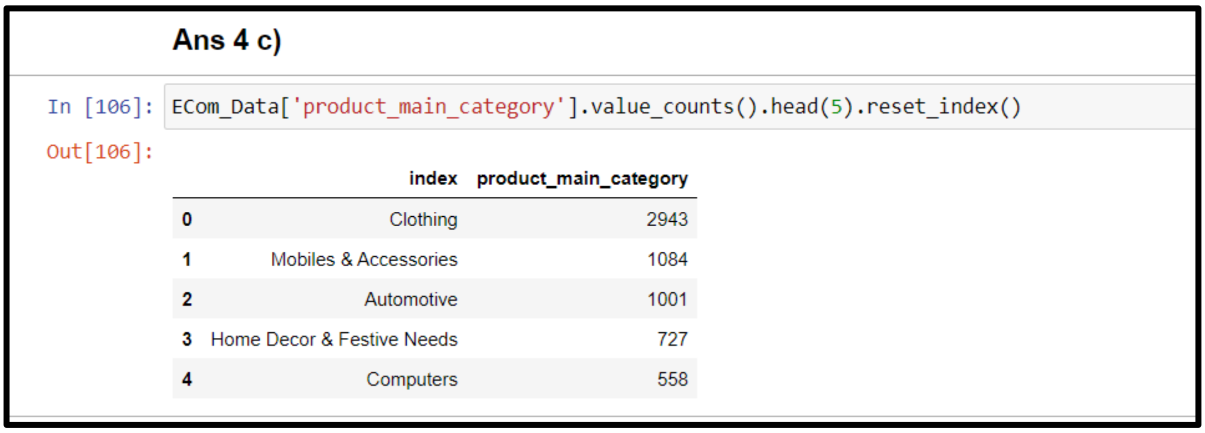
From the bar graph, it is very clearly visible that maximum orders were placed for Clothing category and Mobiles and Accessories category.

Minimum orders were placed for Wearable and Smart Devices category and Food & Nutrition category.

**c. Top 5 product main category:**

On using the value_counts() function on the product_main_category column of the ECom_Data dataset, we get the top 5 categories in the descending order.

***Output:***



**Ans 4 c)**

```
In [106]: ECom_Data['product_main_category'].value_counts().head(5).reset_index()
Out[106]:
                          index  product_main_category
          0                Clothing                  2943
          1   Mobiles & Accessories                  1084
          2              Automotive                  1001
          3  Home Decor & Festive Needs               727
          4               Computers                   558
```

## 5. Revenue Computation

Both the retailers and the company have to make profits to sustain in the business.

**a. Total Revenue generated by the E-Commerce companies:**

Revenue = 0.25*(discounted_price > 600) + 0.15*( discounted_price > 350 & discounted_price <= 600) + 0.10*( discounted_price > 100 & discounted_price <= 350) + 0.05*( discounted_price <= 100)

Four columns can be added to the E-Commerce dataset with the four conditions of revenue generation, namely, 25%, 15%, 10% and 5%. Then merge the four columns using the fillna() function into one column called revenue. Then, take the sum() of the revenue column of the dataset.

***Output:***

```
2217486.85
```

***The total Revenue is 2217486.85***

## 6. Brand Revenue and top 10 brands

**a. Total Brand Revenue:**

The Brand Revenue of each brand is calculated by subtracting the revenue from the discounted price.

As the column called revenue has been added to the dataset, subtract the values of the two columns and create a new column called Brand Revenue.

Then, calculate the sum of all the rows of Brand Revenue column using the sum() function.

***Output:***

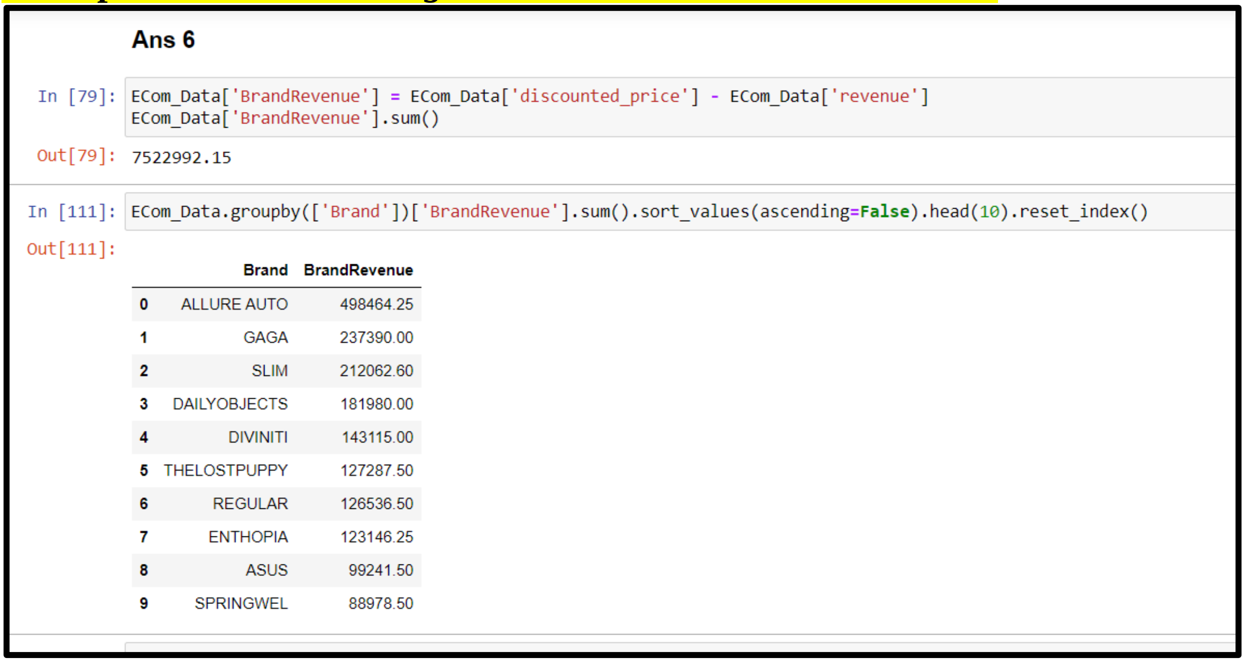
```
7522992.15
```

***The total Brand Revenue is*** `7522992.15`

**b. Top 10 Brands having maximum revenue:**

Use the groupby() function to group the two columns namely, Brand and Brand Revenue. Use the sum() function to calculate the sum of the Brand revenues of the same brands and then sort_values() function sorts these values. Use the ascending = False to sort the values in descending order.

***Output:***
***The top 10 brands according to their Brand revenues are shown:***



**Ans 6**

```
In [79]: ECom_Data['BrandRevenue'] = ECom_Data['discounted_price'] - ECom_Data['revenue']
         ECom_Data['BrandRevenue'].sum()

Out[79]: 7522992.15
```

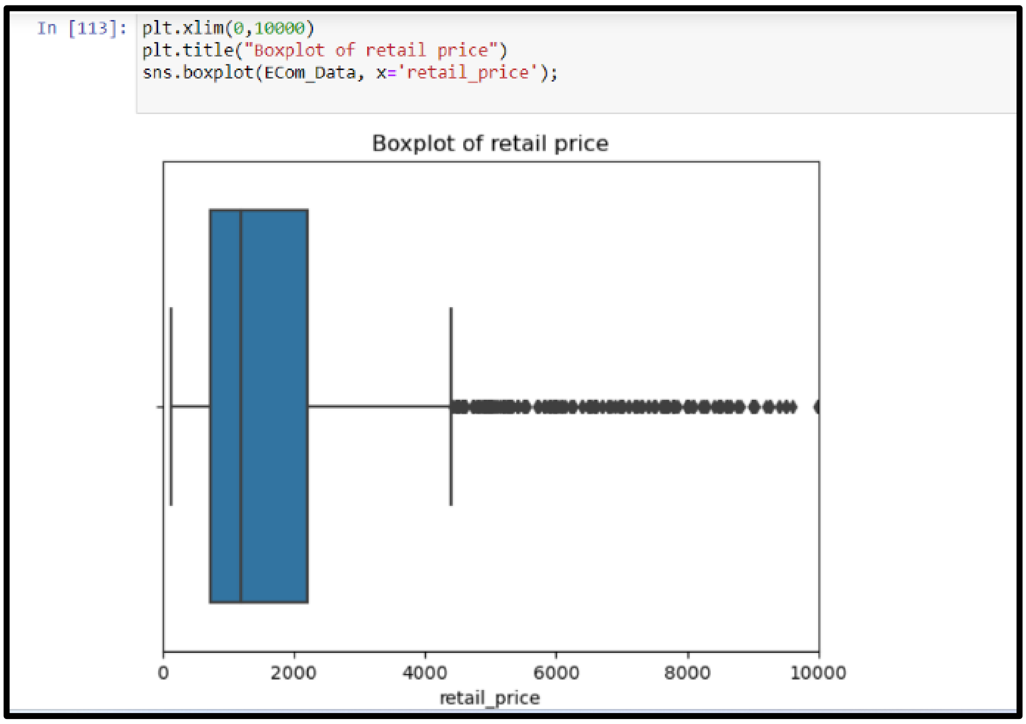
```
In [111]: ECom_Data.groupby(['Brand'])['BrandRevenue'].sum().sort_values(ascending=False).head(10).reset_index()
Out[111]:
```

| | Brand | BrandRevenue |
|---|---|---|
| 0 | ALLURE AUTO | 498464.25 |
| 1 | GAGA | 237390.00 |
| 2 | SLIM | 212062.60 |
| 3 | DAILYOBJECTS | 181980.00 |
| 4 | DIVINITI | 143115.00 |
| 5 | THELOSTPUPPY | 127287.50 |
| 6 | REGULAR | 126536.50 |
| 7 | ENTHOPIA | 123146.25 |
| 8 | ASUS | 99241.50 |
| 9 | SPRINGWEL | 88978.50 |

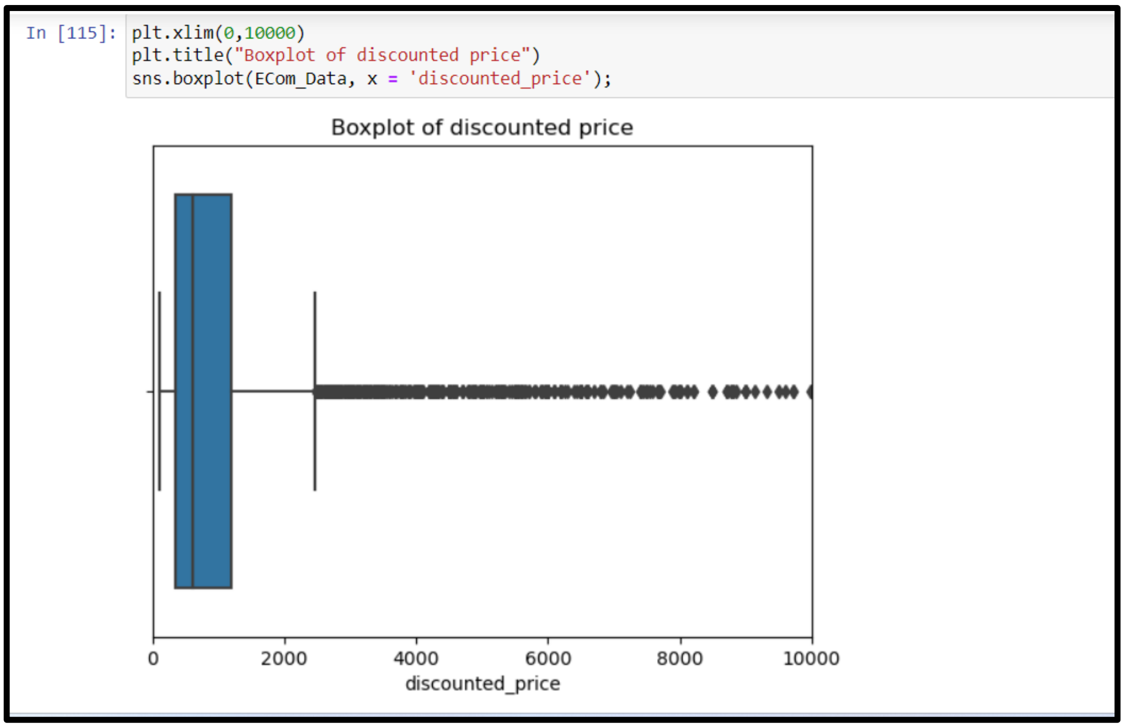
## 7. Boxplot and Scatterplot

### a. Boxplots of retail price and discounted price:

Use the boxplot() function to draw the boxplots of retail price and discounted price.



```
In [113]: plt.xlim(0,10000)
          plt.title("Boxplot of retail price")
          sns.boxplot(ECom_Data, x='retail_price');
```



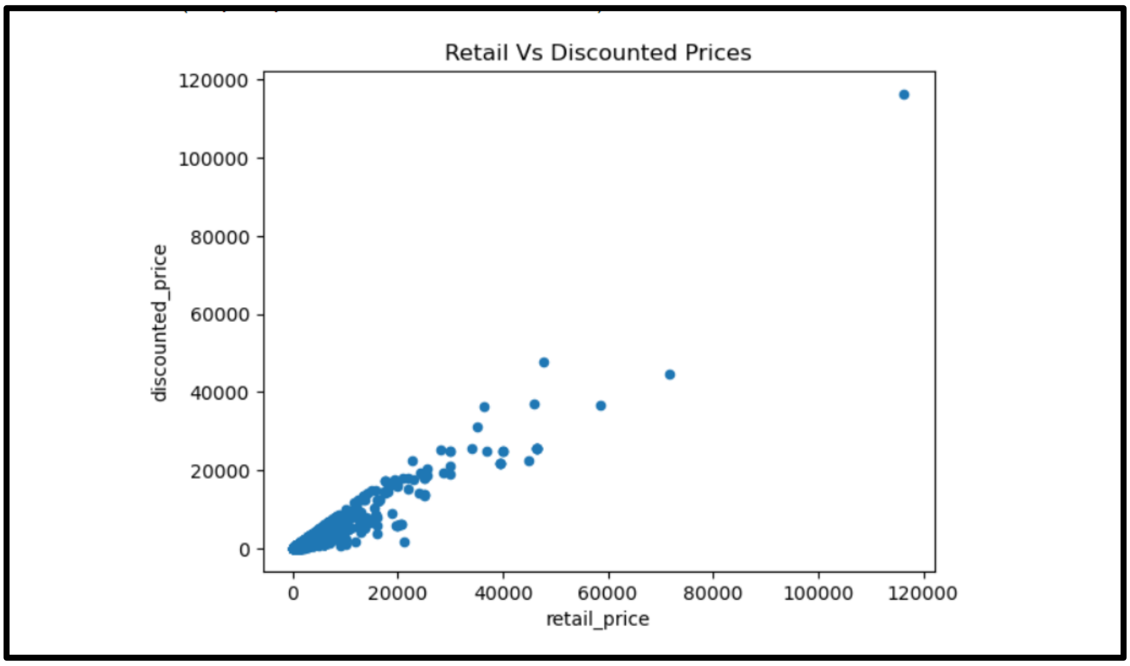
```
In [115]: plt.xlim(0,10000)
          plt.title("Boxplot of discounted price")
          sns.boxplot(ECom_Data, x = 'discounted_price');
```

There are outliers in both the boxplots shown by the dots outside the maximum.

**b. Scatter plot of retail price vs discounted price**

Use the scatter() function to take the data as ECom_Data and specify the x-axis as retail price and y-axis as discounted price.

***Output:***



## 8. New Dataframe for Brand specific information

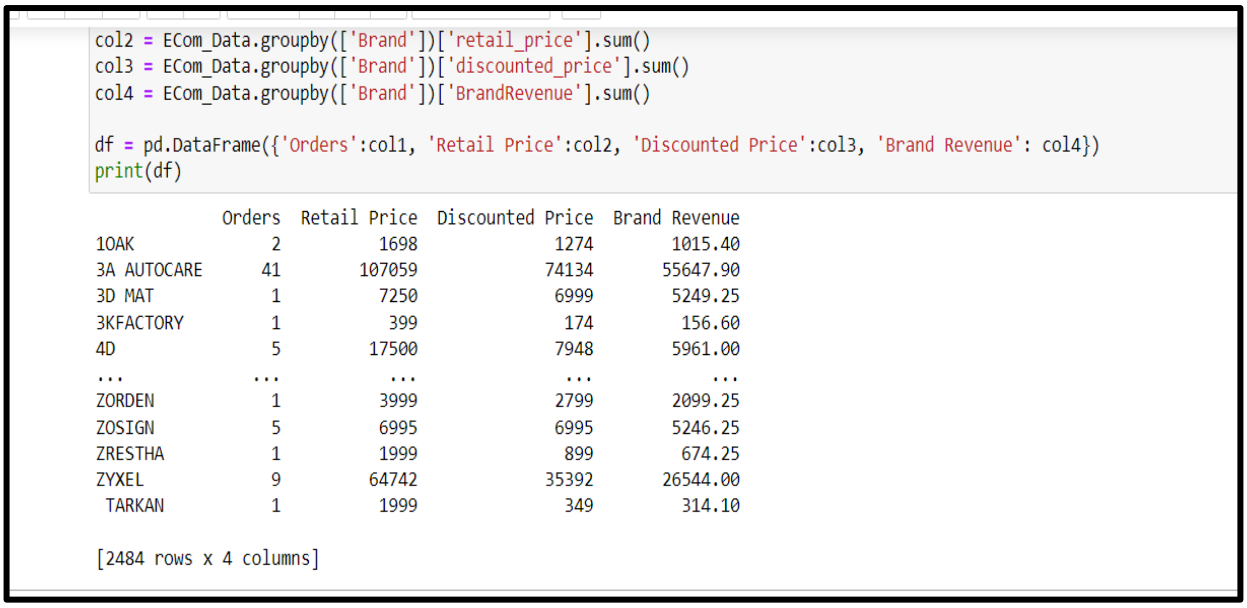
The first column of the new dataframe comprises of the total number of orders placed per Brand. Use the value_counts() function on the Brand column to get this information.

The second column comprises of the total retail price per brand. Use groupby() function to group Brand and retail price and use the sum() on the retail price column to get the second column values.

The third column comprises of the total discounted price per brand. Again, use the groupby() function to group Brand and discounted price and use the sum() function on the discounted price column.

The fourth column comprises of the total Brand Revenue generated per brand. This information can be procured by grouping the Brand and Brand revenue columns and applying sum() on the Brand revenue.

The new dataframe looks like this:

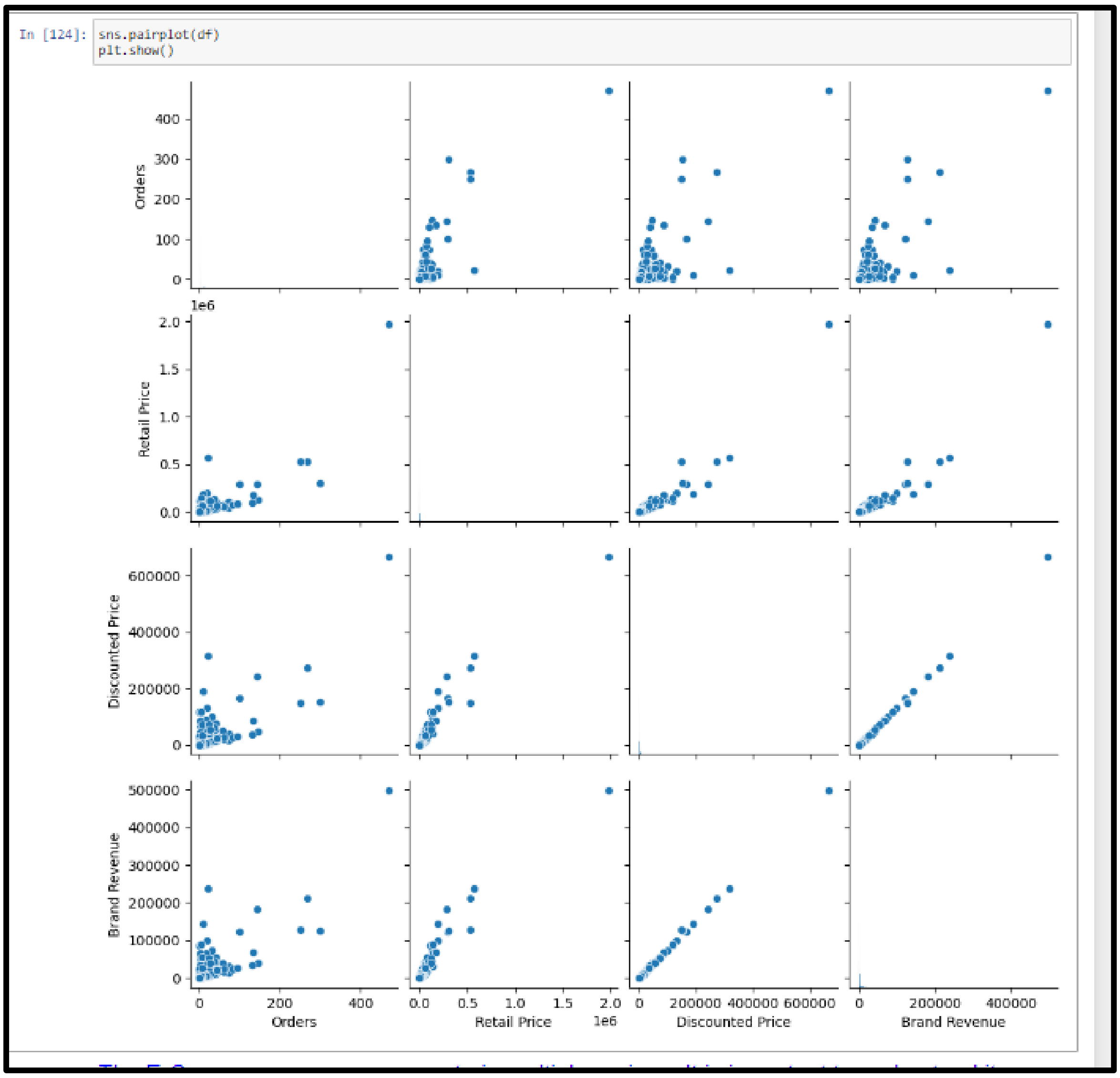


```
col2 = ECom_Data.groupby(['Brand'])['retail_price'].sum()
col3 = ECom_Data.groupby(['Brand'])['discounted_price'].sum()
col4 = ECom_Data.groupby(['Brand'])['BrandRevenue'].sum()

df = pd.DataFrame({'Orders':col1, 'Retail Price':col2, 'Discounted Price':col3, 'Brand Revenue': col4})
print(df)
```

```
              Orders  Retail Price  Discounted Price  Brand Revenue
1OAK               2          1698              1274        1015.40
3A AUTOCARE       41        107059             74134       55647.90
3D MAT             1          7250              6999        5249.25
3KFACTORY          1           399               174         156.60
4D                 5         17500              7948        5961.00
...              ...           ...               ...            ...
ZORDEN             1          3999              2799        2099.25
ZOSIGN             5          6995              6995        5246.25
ZRESTHA            1          1999               899         674.25
ZYXEL              9         64742             35392       26544.00
 TARKAN            1          1999               349         314.10

[2484 rows x 4 columns]
```

### a. Pairplot for the new dataframe:

Use the pairplot() function to get the pairplot of the new dataframe.



```
In [124]: sns.pairplot(df)
          plt.show()
```

## 9. Comparison of Performance Regionwise:

### a. Lineplot for each region:

Lineplot for each region is plotted using the lineplot() function in seaborn.

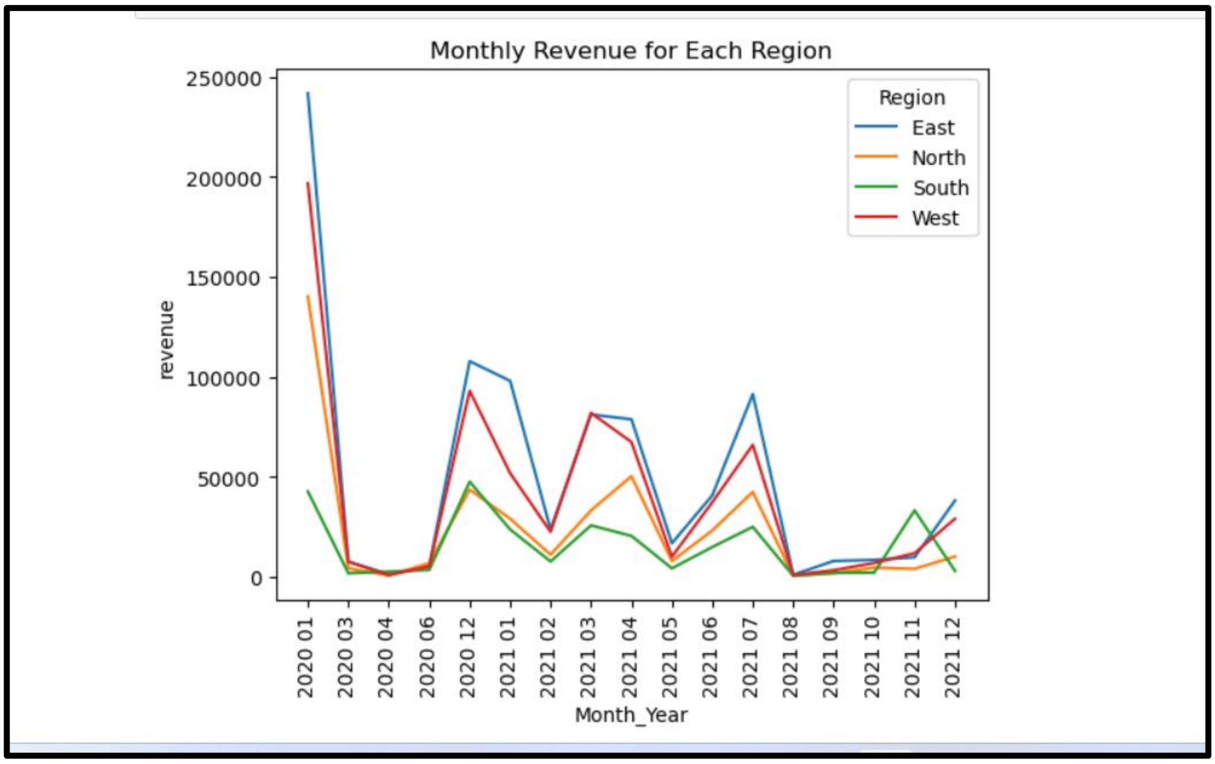
First, change the datatype of the Order_Date column from string to datetime using to_datetime() function.

Then, extract the year and the month using the .dt.year and .dt.month formats and make new columns. Also, extract the year and month together into a Month_Year column in the dataset.

Now, make a new dataframe with columns Region, Month_Year and total revenue in that particular month and year.

Using this dataframe, plot a lineplot with hue as Region, x-axis and Month_Year and y-axis as revenue respectively.

***Output:***



**b. Best and Worst performing months for every region**

East Region:
Best Performing Month - January, 2020
Worst Performing Month - August, 2021

North Region:
Best Performing Month - January, 2020
Worst Performing Month - August, 2021

South Region:
Best Performing Month - January, 2020
Worst Performing Month - August, 2021

West Region:
Best Performing Month - January, 2020
Worst Performing Month - August, 2021

In this way, we have analysed the dataset given to us and extracted the requisite information from the same to help us visualize the dataset in a better way.