Price Elasticity of demand

SHUVADEEP MAITY

Contents

Abstract - The price elasticity of demand (PED) measures the percentage change in quantity demanded by consumers as a result of a percentage change in price. This measurement of price elasticity of demand is calculated by dividing the % change in quantity demanded by the % change in price, represented in the PED ratio.

$$\text{Price Elasticity of Demand} = \frac{\text{Percentage change in quantity demanded}}{\text{Percentage change in price}} = \frac{(Q_1-Q_0)\div(Q_1+Q_0)}{(P_1-P_0)\div(P_1+P_0)}$$

Q_0 = Initial quantity demanded when P = 0
Q_1 = New quantity demanded when P_0 changes to P_1
P_0 = Initial price, immediately prior to the price change
P_1 = The new price, immediately post the price change

Introduction - Price elasticity modelling is a fundamental analytical technique used by businesses to understand how changes in product prices impact consumer demand. It measures the sensitivity of demand to changes in price, indicating how much the quantity demanded will change in response to a change in price. Our use of this technique is to identify products that we may offer at a discount to increase sales and revenue.

The formula for price elasticity using the point slope method is given below where points refer to the mean values of price and quantity.

$$\text{Price Elasticity} = \frac{\Delta Q}{\Delta P} \times \frac{P_{\text{mean}}}{Q_{\text{mean}}}$$

Where,

$$\text{Price Elasticity} = \text{Price Elasticity of Demand}$$

$$\Delta Q = \text{Change in Quantity}$$

$$\Delta P = \text{Change in Price}$$

$$P_{\text{mean}} = \text{Mean Price}$$

$$Q_{\text{mean}} = \text{Mean Quantity}$$

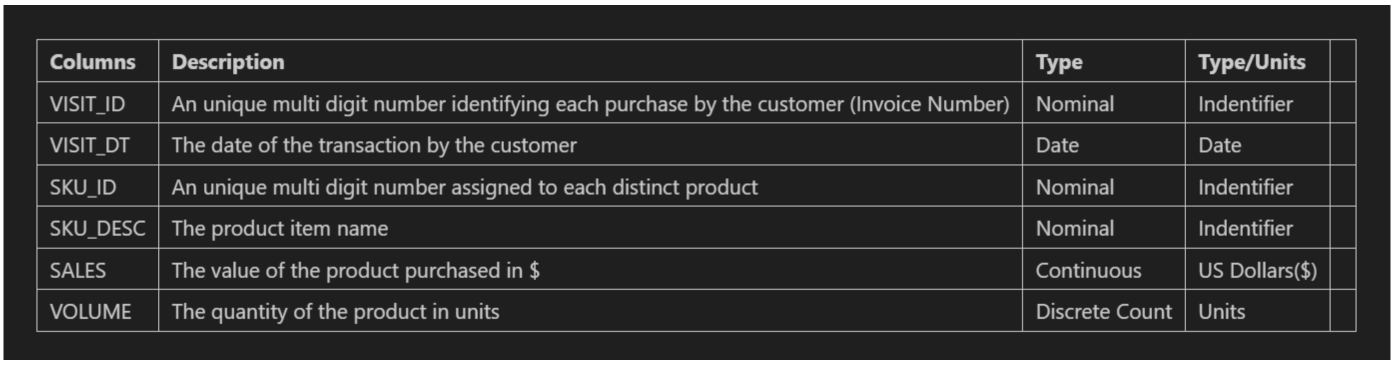
Problem statement- In the pursuit of revenue optimization, businesses often face the challenge of determining the price elasticity of demand (PED) for their products or services. The problem lies in understanding how sensitive consumers are to changes in price and how to leverage this information to make informed pricing decisions that maximize revenue. Without an accurate grasp of PED, businesses risk suboptimal pricing strategies and an inability to adapt to market dynamics and competitive pressures. Consequently, there is a pressing need to develop effective strategies and tools that provide real-time insights into PED, enabling businesses to achieve revenue optimization by making data-driven pricing decisions and maintaining a competitive edge in a dynamic marketplace.

Objective - In a retail store, understanding the price elasticity of demand (PED) serves specific objectives that are tailored to the retail environment. The objectives of studying PED in a retail store are -

Pricing Strategy - Retailers aim to set prices that maximize their revenue and profitability. By analysing the price elasticity of demand, they can determine the optimal price points for their products. For instance, if a product has elastic demand, a retailer might lower the price to attract more customers.

Promotions and Discounts - Retailers often run promotions, discounts, and sales events. Understanding PED helps them assess the likely impact of these promotions on sales and profitability. For instance, they can gauge whether a 10% discount will significantly increase sales of a product or if a 20% discount is needed.

Data Overview - The data that has been used to demonstrate is 15 months of transactional data of a particular retail store in the United States. The data has been anonymized to protect the privacy of the customers.

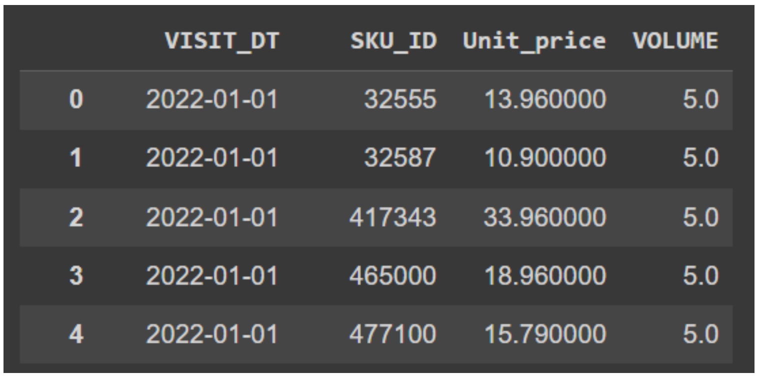


Columns	Description	Type	Type/Units
VISIT_ID	An unique multi digit number identifying each purchase by the customer (Invoice Number)	Nominal	Indentifier
VISIT_DT	The date of the transaction by the customer	Date	Date
SKU_ID	An unique multi digit number assigned to each distinct product	Nominal	Indentifier
SKU_DESC	The product item name	Nominal	Indentifier
SALES	The value of the product purchased in $	Continuous	US Dollars($)
VOLUME	The quantity of the product in units	Discrete Count	Units

Methods - To perform price elasticity modelling, we typically start by analysing data on the volume of sales of a product at different price points with respect to time.

Step 1- A new column added to the DataFrame 'df'. This new column is calculated by dividing the values in the "SALES" column by the values in the "VOLUME" column. Essentially, it computes the unit price for each row in the DataFrame. The result is that for each row in the DataFrame, you now have a "Unit_price" value that represents the price per unit.

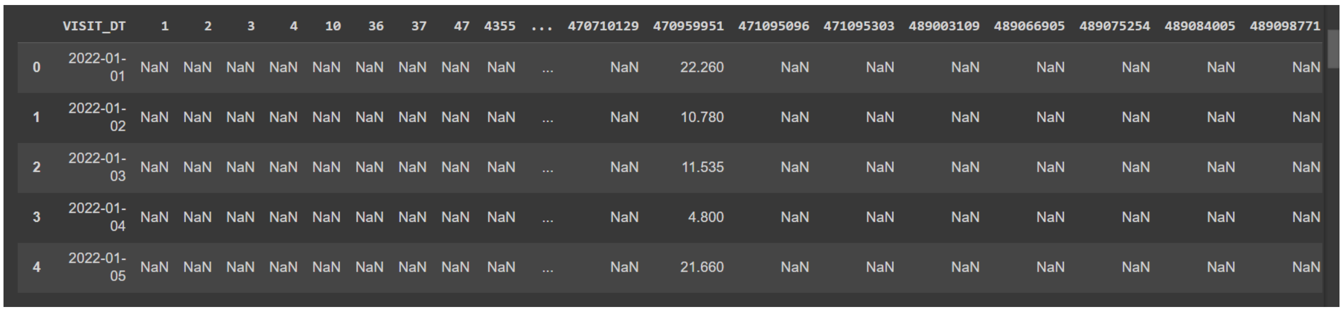
Calculates the mean of the "Unit_price" and "VOLUME" columns for each group created by the 'groupby' operation. So, for each combination of "VISIT_DT" and "SKU_ID," it calculates the mean unit price and the mean VOLUME.



	VISIT_DT	SKU_ID	Unit_price	VOLUME
0	2022-01-01	32555	13.960000	5.0
1	2022-01-01	32587	10.900000	5.0
2	2022-01-01	417343	33.960000	5.0
3	2022-01-01	465000	18.960000	5.0
4	2022-01-01	477100	15.790000	5.0

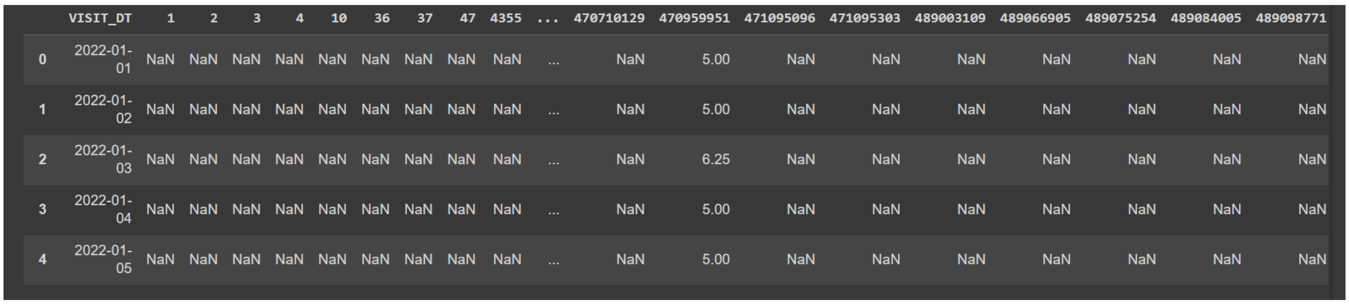
<u>Step 2-</u> During this stage we get pivot tables for each product with:

- Pivot table called x values as the price points with respect to time with the product as columns



	VISIT_DT	1	2	3	4	10	36	37	47	4355	...	470710129	470959951	471095096	471095303	489003109	489066905	489075254	489084005	489098771
0	2022-01-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	22.260	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	2022-01-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	10.780	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2022-01-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	11.535	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2022-01-04	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	4.800	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	2022-01-05	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	21.660	NaN	NaN	NaN	NaN	NaN	NaN	NaN

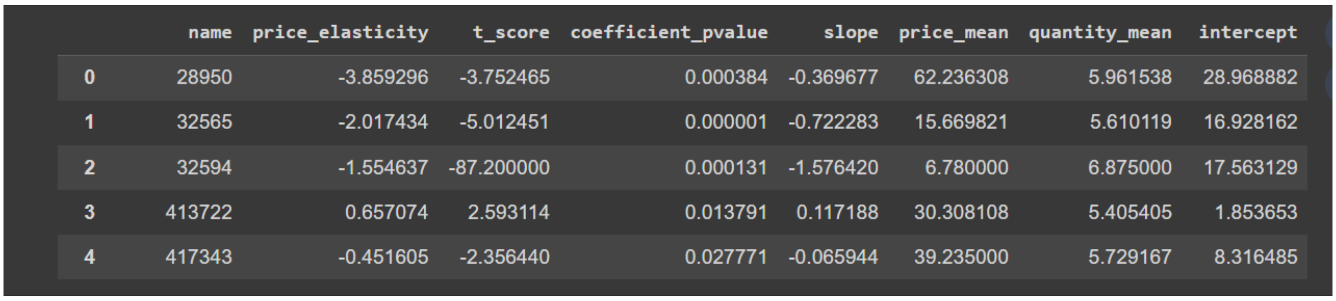
- Pivot table called y values as the volume of sales of the product with respect to time as columns



	VISIT_DT	1	2	3	4	10	36	37	47	4355	...	470710129	470959951	471095096	471095303	489003109	489066905	489075254	489084005	489098771
0	2022-01-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	5.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	2022-01-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	5.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2022-01-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	6.25	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2022-01-04	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	5.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	2022-01-05	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	5.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Now performs price elasticity modelling on a dataset with x_values (unit price) and y_values (volume). The code loops through each column in x_values and creates a list of points (x_value, y_value) for each column. It then creates a dataframe df from the list of points and performs linear regression using the statsmodels library.

Then performs a null hypothesis test to determine if the coefficient has a p-value less than 0.05. If the p-value is less than 0.05, calculates the price elasticity of demand using the slope of the linear regression model and the mean price and mean quantity of the product. Then appends the results of the analysis into a dictionary results_values and creates a final dataframe df_elasicity from the dictionary.



	name	price_elasticity	t_score	coefficient_pvalue	slope	price_mean	quantity_mean	intercept
0	28950	-3.859296	-3.752465	0.000384	-0.369677	62.236308	5.961538	28.968882
1	32565	-2.017434	-5.012451	0.000001	-0.722283	15.669821	5.610119	16.928162
2	32594	-1.554637	-87.200000	0.000131	-1.576420	6.780000	6.875000	17.563129
3	413722	0.657074	2.593114	0.013791	0.117188	30.308108	5.405405	1.853653
4	417343	-0.451605	-2.356440	0.027771	-0.065944	39.235000	5.729167	8.316485

The resulting df_elasticity dataframe contains the name of each product, its price elasticity, t-score, coefficient p-value, slope, mean price, mean quantity, and intercept. The resulting model has various metrics to gauge its performance. One of the key metrics is the models p value or the probability value. A common threshold of the pvalue is 0.05. This value is used to determine if the model is statistically significant.

The p-value calculated is used to determine whether there is sufficient evidence to reject the null hypothesis in favour of the alternative hypothesis. If the p-value is less than the chosen significance level (alpha), in our case it is 0.05, it suggests that the slope coefficient is statistically significantly different from zero. In this case, the null hypothesis is rejected, indicating that there is a significant relationship between price and quantity demanded (i.e., price elasticity is not zero). On the other hand, if the p-value is greater than alpha, there is insufficient evidence to reject the null hypothesis, implying that the slope coefficient is not significantly different from zero, and there may not be a significant relationship between price and quantity demanded.

Step 3- Based on the values of the price elasticity we can classify the products into the following categories.

1. Elastic Products:

- Products with elastic demand have a price elasticity less than -1 ($|E| < -1$).

- When the price of elastic products increases, the quantity demanded decreases significantly, and when the price decreases, the quantity demanded increases significantly.

- Consumers are highly responsive to price changes for these products.

2. Unit Elastic Products:

- Unit elastic products have a price elasticity from 1 to -1 ($-1 < |E| < 1$).

- For unit elastic products, the percentage change in quantity demanded is exactly equal to the percentage change in price.

- Total revenue remains constant with price changes for unit elastic products.

- While relatively rare in practice, some products may exhibit unitary elasticity under specific circumstances.

3. Inelastic Products:

- Inelastic products are less sensitive to price changes, meaning that changes in price have a relatively small impact on the quantity demanded.

- Even if the price of inelastic products increases, the quantity demanded decreases only slightly, and if the price decreases, the quantity demanded increases moderately.

- In our use case after computing the elastic and unit elastic products, rest of the products are denoting as Inelastic Products.

For this use case we are mainly concerned with elastic products for obvious reasons.

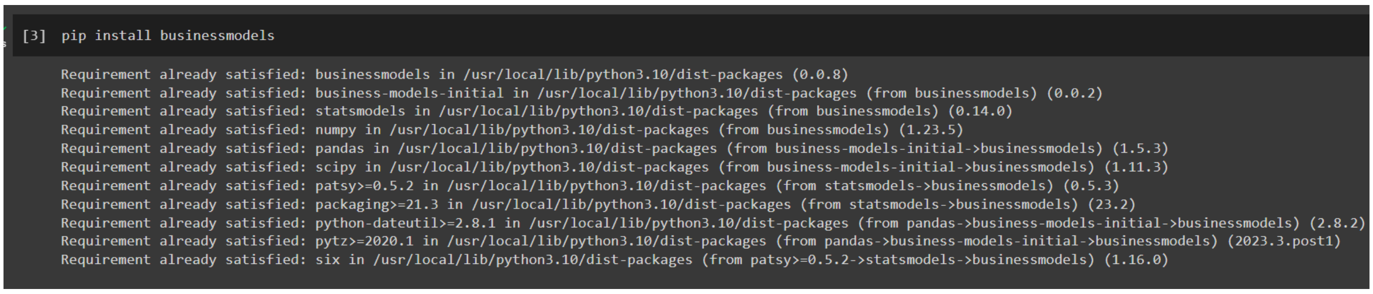
For example, if the price elasticity of demand is -2, then a 1% increase in price will result in a 2% decrease in demand. We then use this information to devise promotional schemes.

Implementation –

1.Installation-

Following code is to install package.

pip install businessmodels



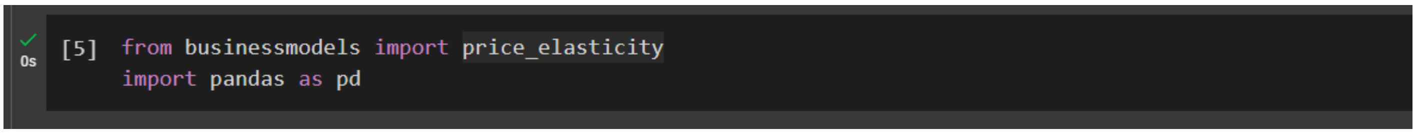
```
[3]  pip install businessmodels

     Requirement already satisfied: businessmodels in /usr/local/lib/python3.10/dist-packages (0.0.8)
     Requirement already satisfied: business-models-initial in /usr/local/lib/python3.10/dist-packages (from businessmodels) (0.0.2)
     Requirement already satisfied: statsmodels in /usr/local/lib/python3.10/dist-packages (from businessmodels) (0.14.0)
     Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from businessmodels) (1.23.5)
     Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from business-models-initial->businessmodels) (1.5.3)
     Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from business-models-initial->businessmodels) (1.11.3)
     Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.10/dist-packages (from statsmodels->businessmodels) (0.5.3)
     Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels->businessmodels) (23.2)
     Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas->business-models-initial->businessmodels) (2.8.2)
     Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->business-models-initial->businessmodels) (2023.3.post1)
     Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.2->statsmodels->businessmodels) (1.16.0)
```

Import the 'businessmodels' library and its 'price_elasticity' module

from businessmodels import price_elasticity

Import the 'pandas' library as 'pd'

import pandas as pd



```
[5]  from businessmodels import price_elasticity
     import pandas as pd
```

2.Data Load-

Read a CSV file from Google Drive or any type of storage it in the 'df' DataFrame

df = pd.read_csv('/content/drive/MyDrive/store_1586_q1_2022_q1_2023.csv')

Display the first few rows of the DataFrame

df.head()

3.Function Call -

# Calculate price elasticity using the 'pricing' function from the 'price_elasticity' module

Separated = price_elasticity.pricing(df)

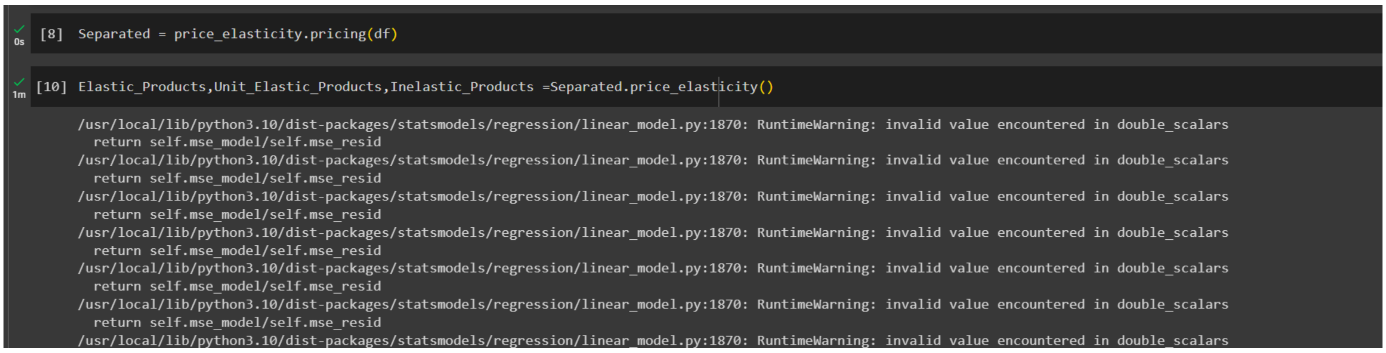
# Get the results of price elasticity, separating products into different categories

# 'Elastic_Products' are products with elastic demand

# 'Unit_Elastic_Products' are products with unitary elastic demand

# 'Inelastic_Products' are products with inelastic demand

Elastic_Products, Unit_Elastic_Products, Inelastic_Products = Separated.price_elasticity()



```
[8]  Separated = price_elasticity.pricing(df)

[10] Elastic_Products,Unit_Elastic_Products,Inelastic_Products =Separated.price_elasticity()

     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
       return self.mse_model/self.mse_resid
     /usr/local/lib/python3.10/dist-packages/statsmodels/regression/linear_model.py:1870: RuntimeWarning: invalid value encountered in double_scalars
```

# Display the products with elastic demand

print("\nElastic Products:")

print(Elastic_Products)

# Display the products with unitary elastic demand

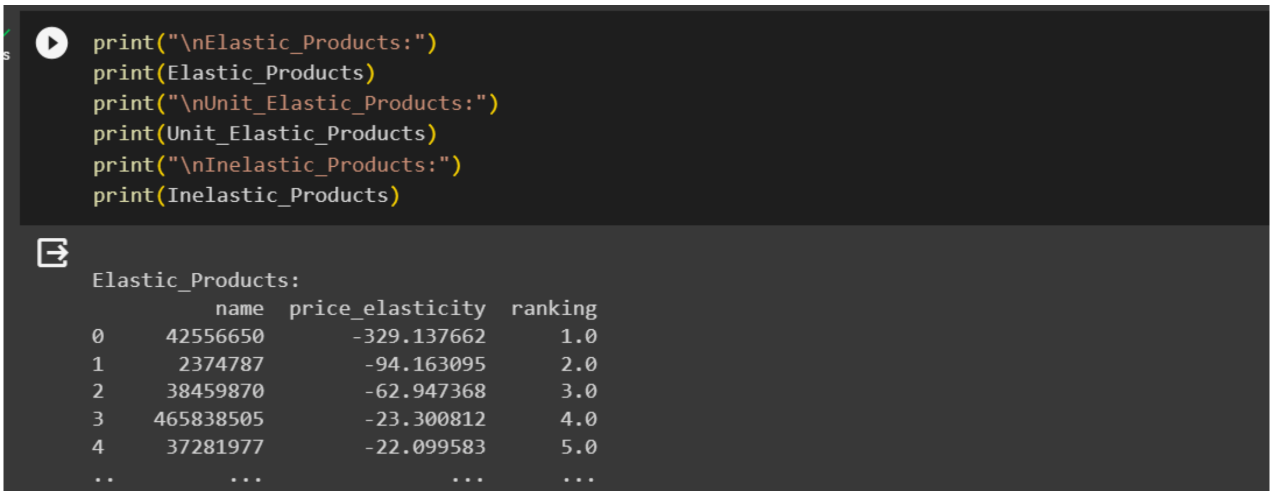
print("\nUnit Elastic Products:")

print(Unit_Elastic_Products)

# Display the products with inelastic demand

print("\nInelastic Products:")

print(Inelastic_Products)



```
print("\nElastic_Products:")
print(Elastic_Products)
print("\nUnit_Elastic_Products:")
print(Unit_Elastic_Products)
print("\nInelastic_Products:")
print(Inelastic_Products)

Elastic_Products:
           name  price_elasticity  ranking
0      42556650       -329.137662      1.0
1       2374787        -94.163095      2.0
2      38459870        -62.947368      3.0
3     465838505        -23.300812      4.0
4      37281977        -22.099583      5.0
..          ...               ...      ...
```

<u>**Outcomes -**</u> The modelling resulted in 272 products being elastic, about 2.11% of total products. These products are the ones that are most sensitive to price changes. Therefore, we can use this list to identify products that can be used as targeted campaigns to increase sales and revenue.

For example, they can be targeted to High-Frequency Low-Value (HFLM) customers to increase their average order value.

Additionally, there are 11,203 inelastic products, about 86.82% of total products. These products are the ones that are least sensitive to price changes.

Conclusion - Price elasticity calculations allowed us to pinpoint 272 products, which were most sensitive to price changes. This information was used to suggest that the business should focus on promoting these products through discounts and promotions.

And these products are inelastic, businesses have more flexibility in adjusting their prices without significantly affecting the demand. Businesses can consider increasing the prices of these products to boost revenue and profits.