Economics

Assignment

[MS1101]

Estimating Demand Curve using OLS

Submitted By:

Soubhik Sen

Roll No: 24MA4108

Dept : Mtech in Operations Research

Professor:
Prof. Dr. Kaushik Mandal

Nov 10, 2024

1 Estimating Demand Curve using OLS Regression

1.1 Step 1: Collecting and Cleaning Data

For this analysis, I am collecting data on the **price** and **quantity** of **tomatoes** in **Paschim Bardhaman vegetable market** from **September 1** to **October 30, 2024**. The data is sourced from Agmarknet, a government platform providing market prices and related agricultural information. This data will form the basis for estimating the demand curve for tomatoes in the region using ordinary least squares (OLS) regression. The collected data will include the price of tomatoes and the corresponding quantity sold in various markets during the specified period.

- The data has been sorted by reported date and only the relevant columns are recorded.
- Since linear regression is sensitive to scale, changing the price unit from **Rs./Quintal** to **Rs./kg** and quantity from **Tonnes** to **Quintals** so that price and quantity are on the same level of magnitude and it becomes easier for us to calculate the regression coefficients.

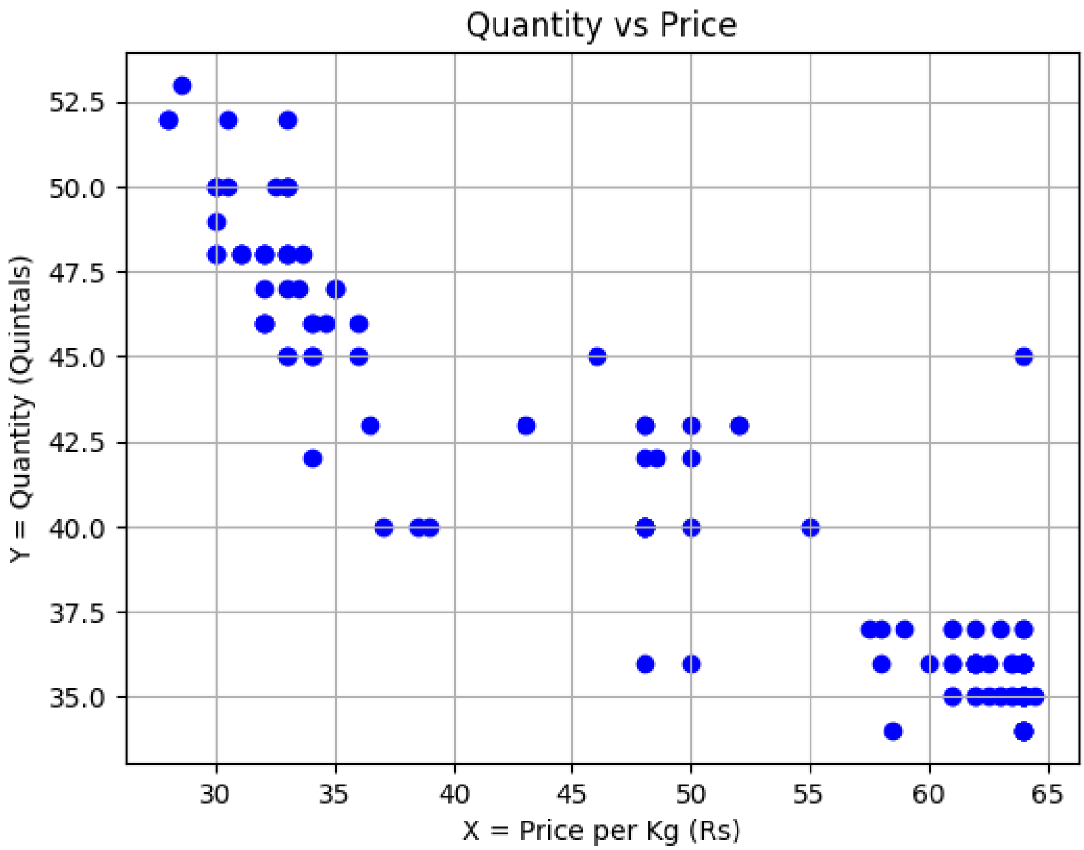
[25]:

```
     Modal Price (Rs./kg)  Quantity (Quintals)
0                    28.0                 52.0
1                    28.5                 53.0
2                    28.0                 52.0
3                    30.0                 50.0
4                    30.5                 52.0
..                    ...                  ...
114                  62.5                 35.0
115                  57.5                 37.0
116                  58.0                 36.0
117                  59.0                 37.0
118                  58.0                 37.0

[119 rows x 2 columns]
```

The **modal price** is the independent variable (**Y**) and **quantity demanded/sold** as the dependent variable (**X**).

[26]:

```
        X     Y
0    28.0  52.0
1    28.5  53.0
2    28.0  52.0
3    30.0  50.0
4    30.5  52.0
..    ...   ...
114  62.5  35.0
115  57.5  37.0
116  58.0  36.0
117  59.0  37.0
118  58.0  37.0

[119 rows x 2 columns]
```



There are 119 datapoints.

1.2 Step 2: Creating Regression Model

Our linear regression model is

$$Y = A + BX$$

For each x_i, let $\hat{y}_i$ be the fitted value. Then, the error for every datapoint i is given by

$$e_i = y_i - \hat{y}_i = y_i - A - Bx_i$$

The sum of the squared errors is given by

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - A - Bx_i)^2$$

Let sum of the squared errors be represented by the function $E(A, B)$.

$$E(A, B) = \sum_{i=1}^{n} e_i^2$$

To find the best fit for the data, we find the values of A and B such that $E(A, B)$ is minimized. This can be done by taking partial derivatives with respect to A and B, and setting them to zero. We obtain

$$\frac{\partial E}{\partial A} = \sum_{i=1}^{n} 2(-1)(y_i - A - Bx_i) = 0$$

$$\frac{\partial E}{\partial B} = \sum_{i=1}^{n} 2(-x_i)(y_i - A - Bx_i) = 0.$$

By solving the above two equations we get

$$\hat{B} = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n} (x_i - \overline{x})^2}$$

$$\hat{A} = \overline{y} - \hat{B}\overline{x}$$

where

$$\overline{x} = \frac{x_1 + x_2 + ... + x_n}{n}$$

$$\overline{y} = \frac{y_1 + y_2 + ... + y_n}{n}$$

So finally, the above formulas give us the regression line

$$\hat{y}_i = \hat{A} + \hat{B}x_i$$

where $\hat{A}$, $\hat{B}$ are **least square estimates** of A and B.

1.3 Step 3: Calculation

First, we will calculate the means of x and y.

[28]:
```
x_mean = df_cleaned['X'].mean().round(4)
y_mean = df_cleaned['Y'].mean().round(4)
print(f"Mean of X (Modal Price): {x_mean}")
print(f"Mean of Y (Quantity): {y_mean}")
```

```
Mean of X (Modal Price): 48.6193
Mean of Y (Quantity): 41.1765
```

Now, we will calculate the other necessary summation values.

[30]:
```
df_cleaned['X - X'] = df_cleaned['X'] - x_mean
df_cleaned['Y - Y'] = df_cleaned['Y'] - y_mean
df_cleaned['(X - X)(Y - Y)'] = df_cleaned['X - X'] * df_cleaned['Y - Y']
df_cleaned['(X - X)^2'] = df_cleaned['X - X'] ** 2

df_cleaned = df_cleaned.round(4)
pd.set_option('display.max_rows', None)
df_cleaned
```

	X	Y	X - $\bar{X}$	Y - $\bar{Y}$	(X - $\bar{X}$)(Y - $\bar{Y}$)	(X - $\bar{X}$)^2
0	28.0	52.0	-20.6193	10.8235	-223.1730	425.1555
1	28.5	53.0	-20.1193	11.8235	-237.8805	404.7862
2	28.0	52.0	-20.6193	10.8235	-223.1730	425.1555
3	30.0	50.0	-18.6193	8.8235	-164.2874	346.6783
4	30.5	52.0	-18.1193	10.8235	-196.1142	328.3090
5	30.0	48.0	-18.6193	6.8235	-127.0488	346.6783
6	30.0	49.0	-18.6193	7.8235	-145.6681	346.6783
7	30.0	50.0	-18.6193	8.8235	-164.2874	346.6783
8	30.0	48.0	-18.6193	6.8235	-127.0488	346.6783
9	32.0	47.0	-16.6193	5.8235	-96.7825	276.2011
10	32.0	46.0	-16.6193	4.8235	-80.1632	276.2011
11	33.0	50.0	-15.6193	8.8235	-137.8169	243.9625
12	31.0	48.0	-17.6193	6.8235	-120.2253	310.4397
13	33.0	48.0	-15.6193	6.8235	-106.5783	243.9625
14	30.5	50.0	-18.1193	8.8235	-159.8756	328.3090
15	34.0	46.0	-14.6193	4.8235	-70.5162	213.7239
16	31.0	48.0	-17.6193	6.8235	-120.2253	310.4397
17	32.5	50.0	-16.1193	8.8235	-142.2286	259.8318
18	31.0	48.0	-17.6193	6.8235	-120.2253	310.4397
19	32.0	48.0	-16.6193	6.8235	-113.4018	276.2011
20	32.0	48.0	-16.6193	6.8235	-113.4018	276.2011
21	33.0	52.0	-15.6193	10.8235	-169.0555	243.9625
22	33.0	50.0	-15.6193	8.8235	-137.8169	243.9625
23	33.0	48.0	-15.6193	6.8235	-106.5783	243.9625

24	33.0	50.0	-15.6193	8.8235	-137.8169	243.9625
25	33.5	47.0	-15.1193	5.8235	-88.0472	228.5932
26	32.0	46.0	-16.6193	4.8235	-80.1632	276.2011
27	33.0	48.0	-15.6193	6.8235	-106.5783	243.9625
28	32.0	48.0	-16.6193	6.8235	-113.4018	276.2011
29	33.0	45.0	-15.6193	3.8235	-59.7204	243.9625
30	32.0	46.0	-16.6193	4.8235	-80.1632	276.2011
31	33.0	47.0	-15.6193	5.8235	-90.9590	243.9625
32	33.6	48.0	-15.0193	6.8235	-102.4842	225.5794
33	33.0	45.0	-15.6193	3.8235	-59.7204	243.9625
34	34.0	42.0	-14.6193	0.8235	-12.0390	213.7239
35	34.0	45.0	-14.6193	3.8235	-55.8969	213.7239
36	34.0	46.0	-14.6193	4.8235	-70.5162	213.7239
37	35.0	47.0	-13.6193	5.8235	-79.3120	185.4853
38	34.0	45.0	-14.6193	3.8235	-55.8969	213.7239
39	37.0	40.0	-11.6193	-1.1765	13.6701	135.0081
40	36.5	43.0	-12.1193	1.8235	-22.0995	146.8774
41	39.0	40.0	-9.6193	-1.1765	11.3171	92.5309
42	38.5	40.0	-10.1193	-1.1765	11.9054	102.4002
43	36.0	46.0	-12.6193	4.8235	-60.8692	159.2467
44	36.0	45.0	-12.6193	3.8235	-48.2499	159.2467
45	34.6	46.0	-14.0193	4.8235	-67.6221	196.5408
46	35.0	47.0	-13.6193	5.8235	-79.3120	185.4853
47	43.0	43.0	-5.6193	1.8235	-10.2468	31.5765
48	46.0	45.0	-2.6193	3.8235	-10.0149	6.8607
49	48.0	42.0	-0.6193	0.8235	-0.5100	0.3835
50	50.0	43.0	1.3807	1.8235	2.5177	1.9063
51	48.0	40.0	-0.6193	-1.1765	0.7286	0.3835
52	48.0	43.0	-0.6193	1.8235	-1.1293	0.3835
53	48.0	40.0	-0.6193	-1.1765	0.7286	0.3835
54	48.0	43.0	-0.6193	1.8235	-1.1293	0.3835
55	50.0	42.0	1.3807	0.8235	1.1370	1.9063
56	48.0	40.0	-0.6193	-1.1765	0.7286	0.3835
57	52.0	43.0	3.3807	1.8235	6.1647	11.4291
58	48.0	40.0	-0.6193	-1.1765	0.7286	0.3835
59	50.0	36.0	1.3807	-5.1765	-7.1472	1.9063
60	48.0	36.0	-0.6193	-5.1765	3.2058	0.3835
61	52.0	43.0	3.3807	1.8235	6.1647	11.4291
62	48.0	40.0	-0.6193	-1.1765	0.7286	0.3835
63	48.5	42.0	-0.1193	0.8235	-0.0982	0.0142
64	50.0	40.0	1.3807	-1.1765	-1.6244	1.9063
65	60.0	36.0	11.3807	-5.1765	-58.9122	129.5203
66	55.0	40.0	6.3807	-1.1765	-7.5069	40.7133
67	63.0	37.0	14.3807	-4.1765	-60.0610	206.8045
68	64.0	45.0	15.3807	3.8235	58.8081	236.5659
69	58.5	34.0	9.8807	-7.1765	-70.9088	97.6282
70	61.0	35.0	12.3807	-6.1765	-76.4694	153.2817

71	62.0	36.0	13.3807	-5.1765	-69.2652	179.0431
72	63.5	36.0	14.8807	-5.1765	-77.0299	221.4352
73	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
74	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
75	63.0	35.0	14.3807	-6.1765	-88.8224	206.8045
76	64.0	37.0	15.3807	-4.1765	-64.2375	236.5659
77	62.0	36.0	13.3807	-5.1765	-69.2652	179.0431
78	63.0	35.0	14.3807	-6.1765	-88.8224	206.8045
79	64.0	34.0	15.3807	-7.1765	-110.3796	236.5659
80	63.5	35.0	14.8807	-6.1765	-91.9106	221.4352
81	62.5	36.0	13.8807	-5.1765	-71.8534	192.6738
82	63.5	35.0	14.8807	-6.1765	-91.9106	221.4352
83	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
84	64.5	35.0	15.8807	-6.1765	-98.0871	252.1966
85	64.0	34.0	15.3807	-7.1765	-110.3796	236.5659
86	64.0	34.0	15.3807	-7.1765	-110.3796	236.5659
87	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
88	64.0	34.0	15.3807	-7.1765	-110.3796	236.5659
89	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
90	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
91	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
92	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
93	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
94	64.0	34.0	15.3807	-7.1765	-110.3796	236.5659
95	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
96	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
97	63.5	36.0	14.8807	-5.1765	-77.0299	221.4352
98	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
99	64.0	36.0	15.3807	-5.1765	-79.6182	236.5659
100	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
101	61.0	37.0	12.3807	-4.1765	-51.7080	153.2817
102	61.0	36.0	12.3807	-5.1765	-64.0887	153.2817
103	62.0	35.0	13.3807	-6.1765	-82.6459	179.0431
104	62.0	36.0	13.3807	-5.1765	-69.2652	179.0431
105	64.0	35.0	15.3807	-6.1765	-94.9989	236.5659
106	64.0	37.0	15.3807	-4.1765	-64.2375	236.5659
107	62.0	37.0	13.3807	-4.1765	-55.8845	179.0431
108	62.0	35.0	13.3807	-6.1765	-82.6459	179.0431
109	61.0	37.0	12.3807	-4.1765	-51.7080	153.2817
110	62.0	36.0	13.3807	-5.1765	-69.2652	179.0431
111	61.0	35.0	12.3807	-6.1765	-76.4694	153.2817
112	62.0	36.0	13.3807	-5.1765	-69.2652	179.0431
113	61.0	36.0	12.3807	-5.1765	-64.0887	153.2817
114	62.5	35.0	13.8807	-6.1765	-85.7341	192.6738
115	57.5	37.0	8.8807	-4.1765	-37.0902	78.8668
116	58.0	36.0	9.3807	-5.1765	-48.5592	87.9975
117	59.0	37.0	10.3807	-4.1765	-43.3550	107.7589

```
118   58.0   37.0    9.3807   -4.1765            -39.1785      87.9975
```

[31]:
```python
s_xy = df_cleaned['(X - X)(Y - Y)'].sum()
s_xx = df_cleaned['(X - X)^2'].sum()
print(f"Σ(X - X)(Y - Y) = {s_xy:.3f}")
print(f"Σ(X - X)^2 = {s_xx:.3f}")
print(f"Mean of X (X) = {x_mean:.3f}")
print(f"Mean of Y (Y) = {y_mean:.3f}")
```

```
Σ(X - X̄)(Y - Ȳ) = -8874.306
Σ(X - X̄)^2 = 22734.522
Mean of X (X̄) = 48.619
Mean of Y (Ȳ) = 41.176
```

[32]:
```python
B = s_xy / s_xx
A = y_mean - B * x_mean

print(f"Slope (B) = {B:.2f}")
print(f"Intercept (A) = {A:.2f}")
```

```
Slope (B) = -0.39
Intercept (A) = 60.15
```

So we have,

$$\hat{B} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2} = \frac{-8874.306}{22734.522} = -0.39$$

$$\hat{A} = \overline{y} - \hat{B}\overline{x} = 41.176 - (-0.39)48.619 = 60.15$$

Finally, the demand curve equation is

$$\boxed{Y = 60.15 - 0.39X}$$

R² Value R² measures how well the regression predictions approximate the actual data points.

Calculating predicted Y values ($\hat{y}_i$) for every x_i.

[33]:
```python
df_cleaned['Y_pred'] = A + B * df_cleaned['X']
ss_res = ((df_cleaned['Y'] - df_cleaned['Y_pred']) ** 2).sum().round(4)
ss_tot = ((df_cleaned['Y'] - y_mean) ** 2).sum().round(4)
r_squared = 1 - (ss_res / ss_tot)
print(f"ss_res = {ss_res}, ss_tot = {ss_tot}, R² = {r_squared:.4f}")
```

```
ss_res = 499.2542, ss_tot = 3963.2941, R² = 0.8740
```

So we have

$$R^2 = 1 - \frac{\mathrm{SS_{res}}}{\mathrm{SS_{tot}}}$$

where:

$$\text{Residual sum of squares} = \mathrm{SS_{res}} = \sum(y_i - \hat{y}_i)^2 = 499.2542$$

$$\text{Total sum of squares} = \mathrm{SS}_{\mathrm{tot}} = \sum (y_i - \overline{y})^2 = 3963.2941$$

Substituting in these values:

$$R^2 = 1 - \frac{499.2542}{3963.2941}$$

And when calculated, this yields:

$$R^2 = 0.8740$$

$$\text{Percentage of } R^2 = 87.4\%$$

Adjusted R² Adjusted R^2 helps account for the number of predictors in the model, preventing the R^2 value from being too optimistic (especially when multiple predictors are involved).

`n = 119, k = 1, Adjusted R² = 0.8730`

Given:

$$n = 119, \quad k = 1, \quad R^2 = 0.8740$$

The formula for Adjusted R^2 is:

$$\text{Adjusted } R^2 = 1 - \left(\frac{1 - R^2}{n - k - 1} \right) \times (n - 1)$$

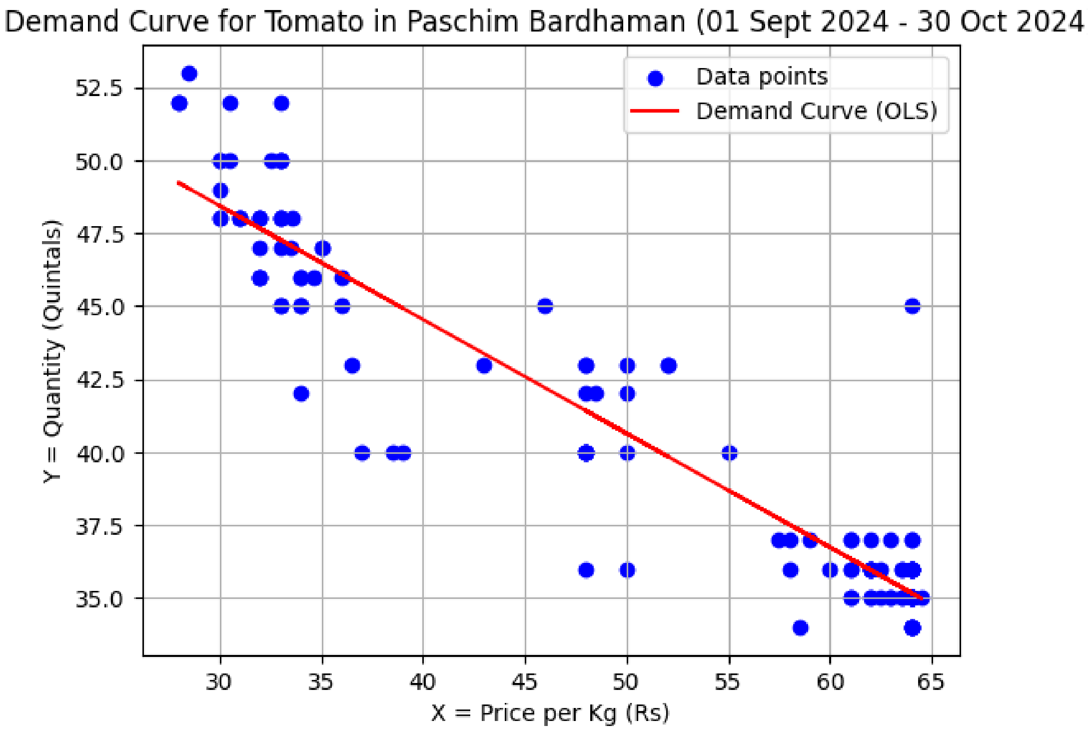
Substituting the values:

$$\text{Adjusted } R^2 = 1 - \left(\frac{1 - 0.8740}{119 - 1 - 1} \right) \times (119 - 1)$$

After calculation:

$$\text{Adjusted } R^2 = 0.8730$$

1.4 Step 4: Plotting the data points and the best fitted line



1.5 Step 5: Conclusion

The plot clearly shows the demand curve, which represents the relationship between the price of tomatoes and the quantity demanded. As observed from the plot, the best-fit line indicates a negative slope, suggesting an inverse relationship between the price and quantity demanded. This means that, as the price of tomatoes increases, the quantity demanded decreases, which is consistent with the **Law of Demand** in economics.

Additionally, the coefficient of determination $R^2 = 0.8740$ suggests that the model explains approximately 87.4% of the variance in the quantity demanded, indicating a strong fit. The Adjusted R^2 value of 0.8730 further confirms the robustness of the model, accounting for the single predictor in the regression.