

ASSIGNMENT -1 CUSTOMER ANALYTICS
Armaan Dhanda 906310394

Date: 9/4/2024

Q1.1

(p, q): 0.003 0.214
m: 34.405

Q1.2

p: 0.00102
q: 0.202

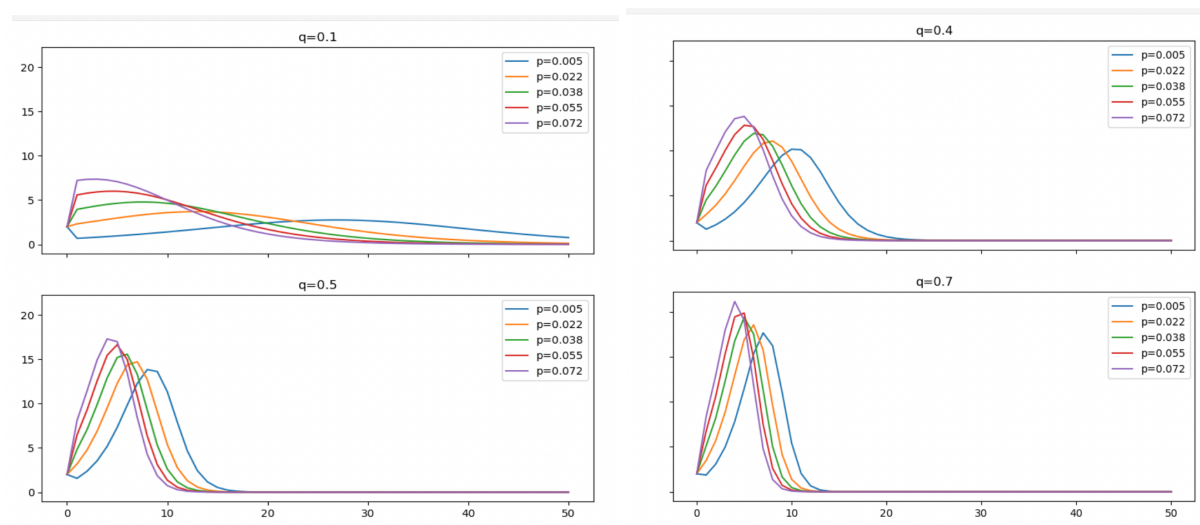
Q1.3

n(30): 4.89

Q1.4

p: 0.00114
q: 0.178
N(30): 4.5

Option Q2:



The adoption curve's shape is influenced by both p and q , with lower values of these parameters, particularly when close to zero, resulting in a more gradual curve and a delayed acceptance of the product. When either p or q increases, the shape of the curve changes

significantly. A higher p leads to a steeper initial climb, indicating faster uptake and an earlier peak adoption. Conversely, a higher q results in a more pronounced peak and a sharper rise and fall in the curve, suggesting faster spread post-initial acceptance.

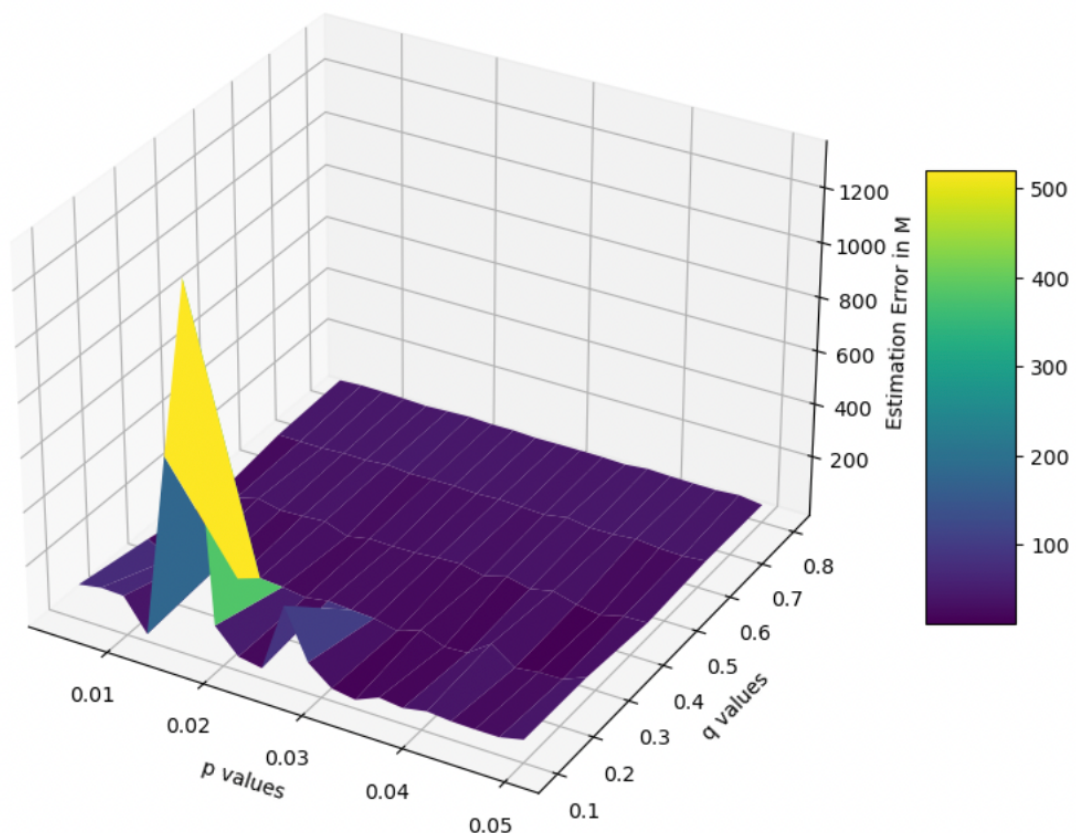
There's a positive correlation between the height of the adoption curve and both p and q , assuming the other parameter remains constant. Higher values of p or q lead to a larger initial number of adopters, increasing the curve's height. A combination of high p and q values contributes to a higher peak in adoption.

The values of p and q are directly linked to the slope of the adoption curve, representing the rate of increase in new adopters. A higher p or q results in a steeper slope, indicating a quicker adoption rate. The interplay between innovation and imitation influences the spread of the product, evident in the slope's steepness.

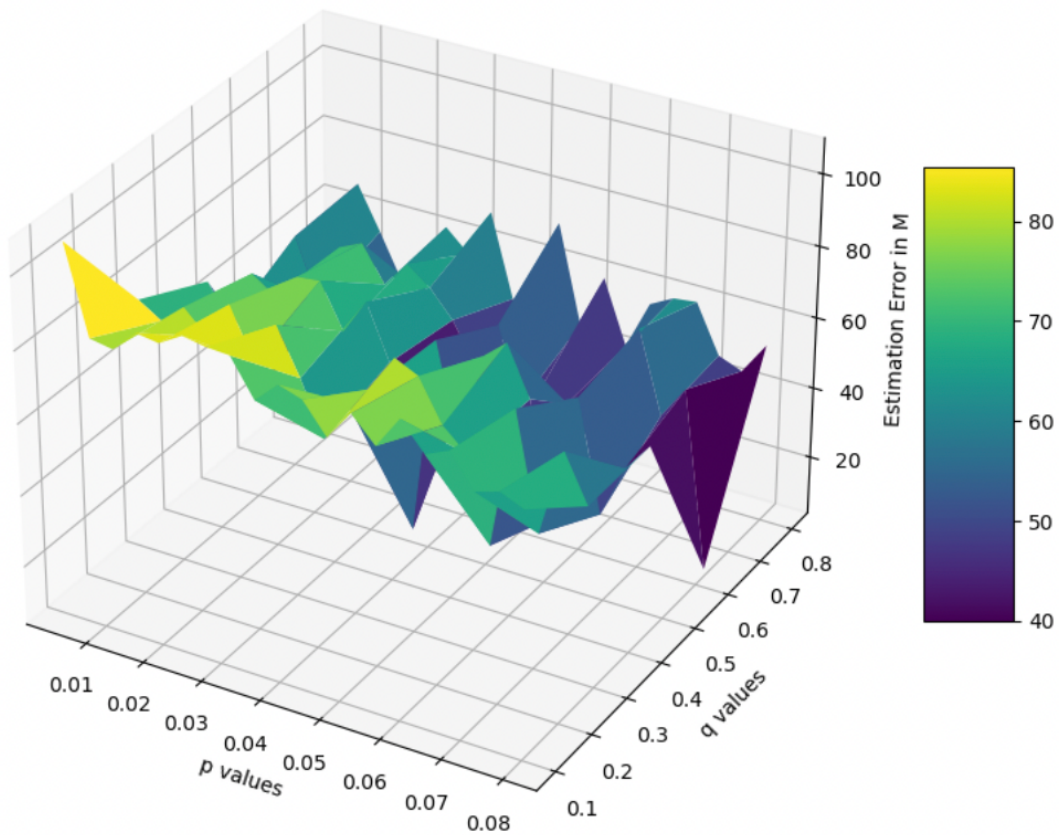
The value of q predominantly influences the decay phase of the curve, dictating how quickly the number of new users declines post-peak. A higher q signifies a faster decay, indicating a sharp decline in new users post-peak, potentially indicating early market saturation. Conversely, a smaller q leads to a longer decay phase, suggesting a more consistent rate of adoption over an extended period.

Optional Q3:

Estimation Error of M across different p and q values for 10% noise



Estimation Error of M across different p and q values 30 % noise



The increase in p , representing innovation, initially leads to a rise in the estimation error for M . This suggests that when innovation has a stronger impact, accurately gauging the total market size based solely on early adoption data becomes more challenging. Early adopters, influenced by innovation, might not fully reflect the subsequent impact of imitation, which is a key driver of market saturation.

The impact of q , representing imitation, shows varying trends. Lower values of p combined with changes in q result in a more significant estimation error for M . This indicates that when initial adoption is less influenced by innovators, the effect of imitation becomes more crucial for predicting total market potential. However, as p increases, the influence of q on the estimation error for M appears to diminish. This suggests that the initial adoption rate might overshadow the effects of imitation in early market estimation.

When either p or q is at an intermediate level, the estimation error tends to be lower. This suggests that maintaining a balance between innovation and imitation provides a more stable foundation for estimating M . If either parameter is too high, it could lead to overfitting to early trends, resulting in inaccurate forecasts.

When the noise level is relatively high, such as 10%, and the values of p and q are small (less than 0.05 and 0.5, respectively), it can indeed lead to highly inaccurate estimates for M .

Optional Q4

We have a non-linear equation at hand we have to reduce the no of parameters by one as we have fixed M.

In order to do that we will use the following equation :

$$\frac{N(t)}{R(t)} = p + q \frac{A(t)}{M}$$

$$R(t) = M - A(t)$$

Performing linear regression on the given function allows you to estimate the values of parameters p and q directly from the coefficients β_0 and β_1 in the regression model.

Covariance type:		nonrobust				
	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.0011	0.000	5.201	0.000	0.001	0.002
A	0.1935	0.009	20.441	0.000	0.173	0.214
Omnibus:		3.694	Durbin-Watson:			3.681
Prob(Omnibus):		0.158	Jarque-Bera (JB):			1.380
Skew:		0.321	Prob(JB):			0.501
Kurtosis:		1.602	Cond. No.			69.2

Values doesn't match exactly with Q1.4 but still it give approximate match of Q1.4 values, $p = 0.00114$, $q = 0.178$, $p' = 0.0011$ and $q' = 0.193$

Optional Q5 (these are ChatGpt 3.5)
Prompt and reply

You

```

whats wrong
for i in range(13,35):
    df['A'][i] = df['N(t)'][i-1]+df['A'][i-1].cumsum()
    df['A_square'][i] = df['A'][i]*df['A'][i]
    df['N(t)'][i] = model_test.predict(df['A'][i],df['A_square'][i])

print(df)

```

ChatGPT

There are several issues in your code. Let's address them:

- When updating values in the DataFrame using `df['A'][i]`, `df['A_square'][i]`, and `df['N(t)'][i]`, you need to use `.loc` to access and modify the values properly.
- In the `model_test.predict()` function call, you need to pass a DataFrame, not individual values.

Here's the corrected version of your code:

```

python
# Assuming 'model_test' and 'df' are defined earlier

for i in range(13, 35):
    df.loc[i, 'A'] = df.loc[i - 1, 'N(t)'] + df.loc[i - 1, 'A']
    df.loc[i, 'A_square'] = df.loc[i, 'A'] * df.loc[i, 'A']
    prediction_data = pd.DataFrame({'A': [df.loc[i, 'A']], 'A_square': [df.loc[i, 'A_square']]})
    df.loc[i, 'N(t)'] = model_test.predict(prediction_data)

print(df)

```

Message ChatGPT...

prompt

I am trying to add a new column with the following values update the code that is easier to read
`df['A'][0] = 0`
`for i in range(len(df['N(t)']):`
`df['A'][i+1] = df['A'][i] + df['N(t)'][i]`

reply

Your code has a small syntax error in the loop, and there's a more efficient way to achieve this using pandas. Here's the corrected code:

```

python
import pandas as pd

# Assuming df is already defined as described in your question

# Initialize the first value of 'A' to 0
df['A'] = 0

# Compute cumulative sum
df['A'] = df['N(t)'].cumsum()

# Shift the 'A' column by one position to the right
df['A'] = df['A'].shift(fill_value=0)

# You can also directly calculate 'A' without using a loop
# df['A'] = df['N(t)'].cumsum().shift(fill_value=0)

# Display the DataFrame
print(df)

```

This code will add a new column 'A' to your DataFrame 'df', with each row containing the cumulative sum of the corresponding row i column 'N(t)' plus the preceding value of 'A', and the first value of 'A' initialized to 0.