# Introduction

```
from google.colab import drive
drive.mount('/content/drive')
import os
os.chdir("/content/drive/My Drive/Spring 25/UGBA 167")
import pandas as pd
import mba263
import matplotlib.pyplot as plt
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

## Load Data

```
data = pd.read_csv('tuscan_rfm.csv')
print(data.head())
```

```
   numords  totdol  last buyer  dollars  rfm1  rfm2
0        7     493   207    no        0   222   211
1        4     423   625    no        0   421   422
2        4     246    28    no        0   134   122
3        3     271   778    no        0   523   532
4        2     148   396    no        0   343   343
```

## Question #1

```
data['buyer'] = data['buyer'].apply(lambda x: 1 if x == 'yes' else 0)

response_rate = data['buyer'].mean() * 100
print(f"The response rate is {response_rate:.2f}%")
```

```
The response rate is 2.46%
```

## Question #2

```
data = pd.read_csv('tuscan_rfm.csv')
data['buyer'] = data['buyer'].astype(str).str.lower().str.strip()
buyers = data[data['buyer'] == 'yes']
if not buyers.empty:
    average_purchase = buyers['dollars'].mean()
    print(f"The average purchase amount from those who bought is ${average_purchase:.2f}")
else:
    print("No purchases were recorded in this dataset.")
```

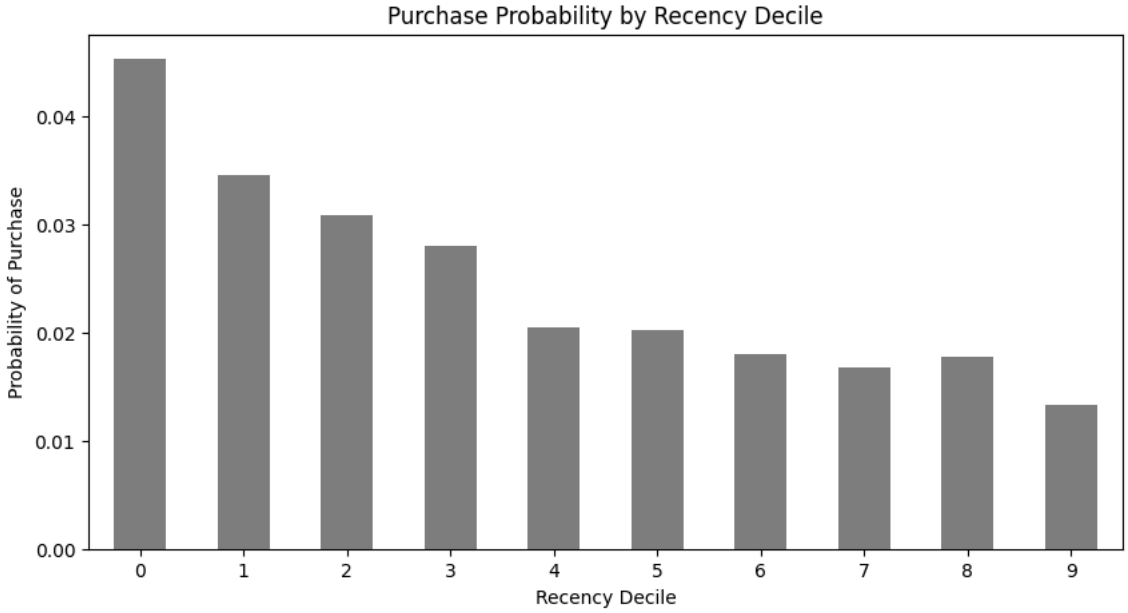
```
The average purchase amount from those who bought is $104.24
```

## Question #3

```
data['rec_dec'] = mba263.ntile(data['last'], 10)
data['buy_dummy'] = (data['buyer'].str.lower() == 'yes').astype(int)
purchase_prob_by_decile = data.groupby('rec_dec')['buy_dummy'].mean()
print("Purchase Probabilities by Recency Decile:")
print(purchase_prob_by_decile)

import matplotlib.pyplot as plt
plt.figure(figsize=(10, 5))
purchase_prob_by_decile.plot(kind='bar', color='grey')
plt.title('Purchase Probability by Recency Decile')
plt.xlabel('Recency Decile')
plt.ylabel('Probability of Purchase')
plt.xticks(rotation=0)
plt.show()
```

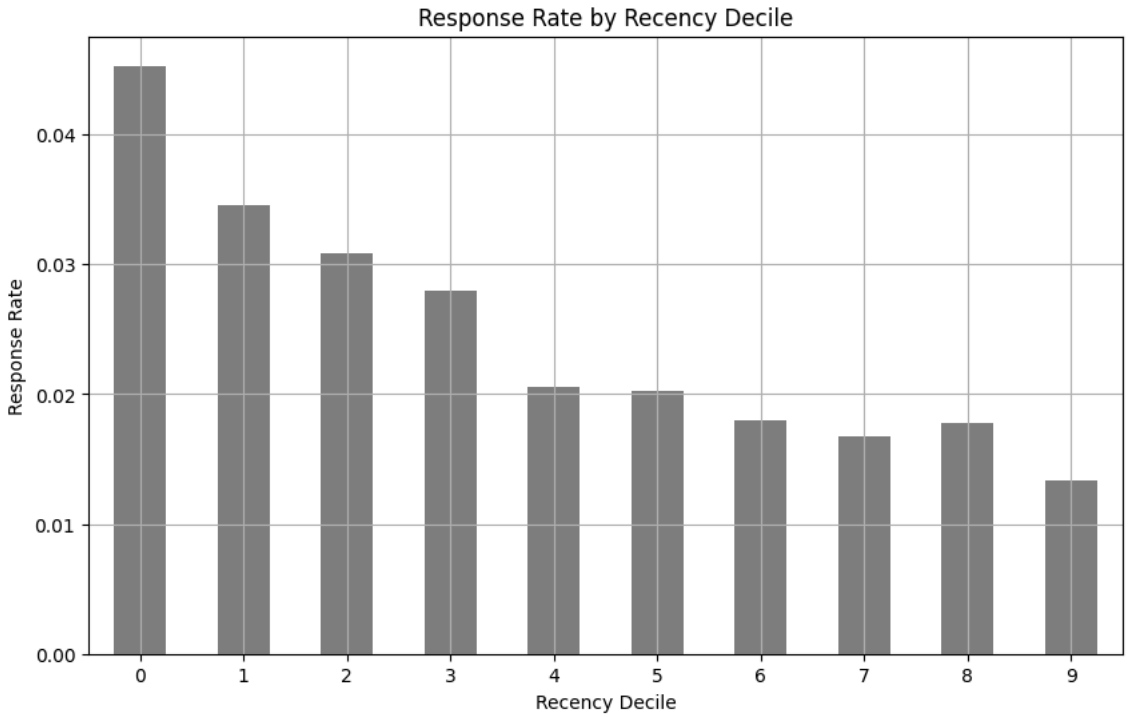
```
plt.show()
```

```
Purchase Probabilities by Recency Decile:
rec_dec
0    0.045305
1    0.034589
2    0.030829
3    0.027971
4    0.020534
5    0.020221
6    0.017969
7    0.016733
8    0.017809
9    0.013354
Name: buy_dummy, dtype: float64
```




## Question #4

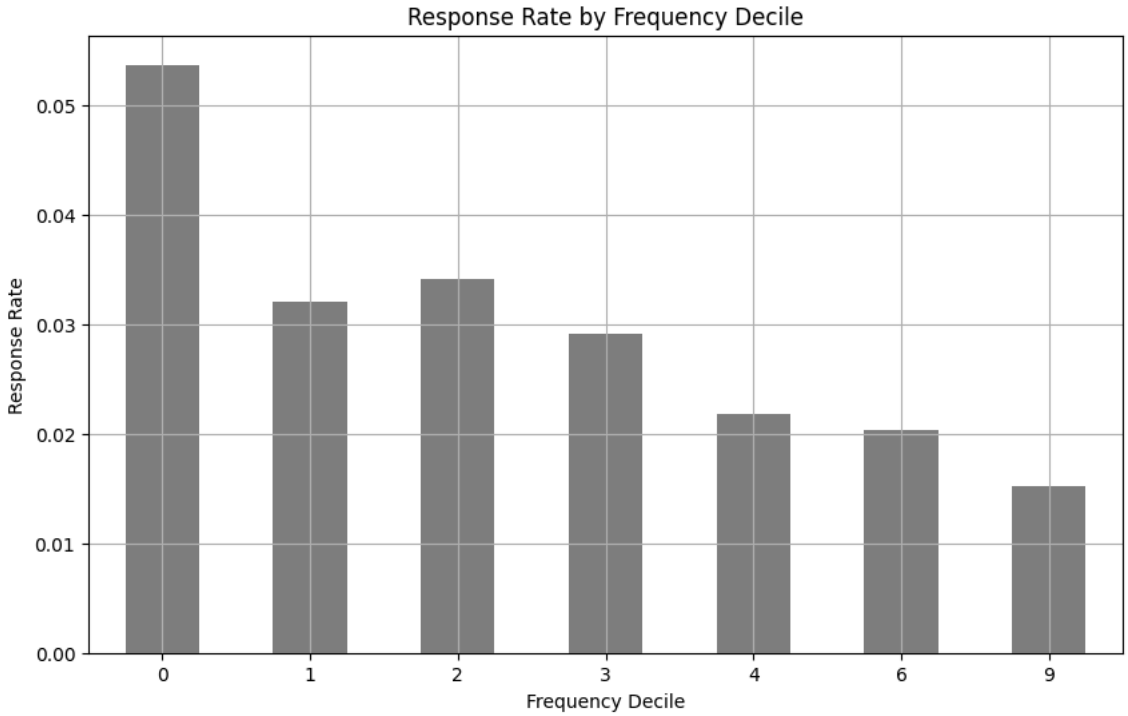
```
data = pd.read_csv('tuscan_rfm.csv')
data['buyer_dummy'] = data['buyer'].apply(lambda x: 1 if x.lower() == 'yes' else 0)
data['rec_dec'] = mba263.ntile(data['last'], 10)
response_rate_by_rec_decile = data.groupby('rec_dec')['buyer_dummy'].mean()

plt.figure(figsize=(10, 6))
response_rate_by_rec_decile.plot(kind='bar', color='grey')
plt.title('Response Rate by Recency Decile')
plt.xlabel('Recency Decile')
plt.ylabel('Response Rate')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()
```




## Question #5

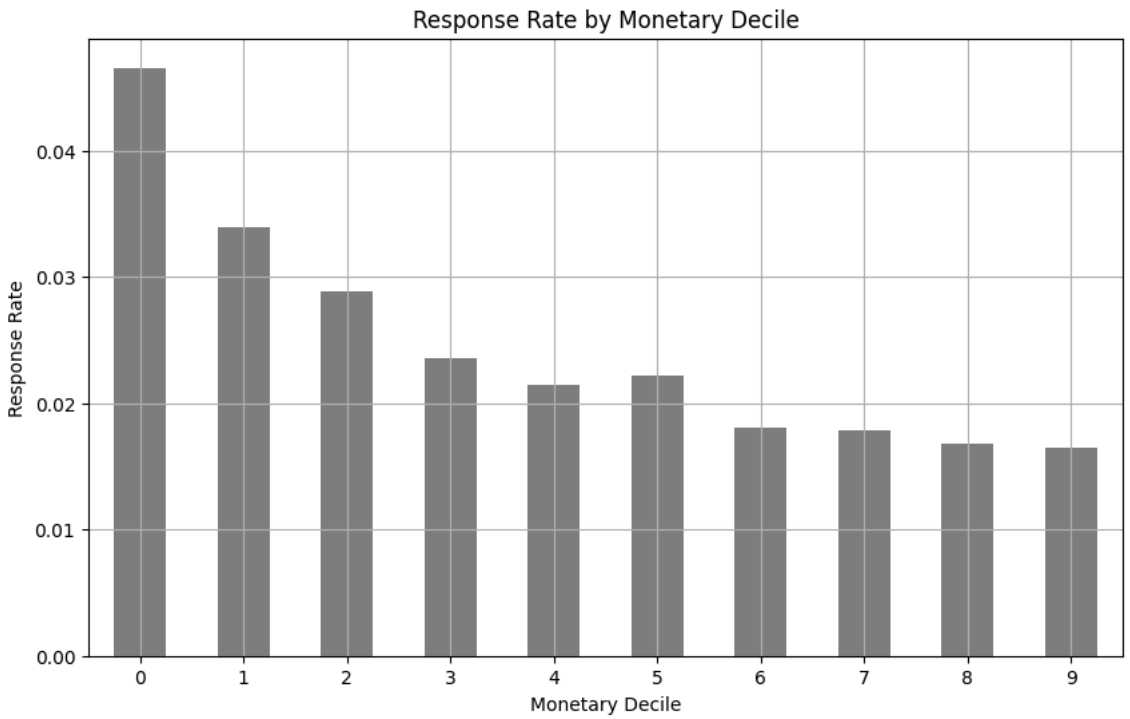
```
data = pd.read_csv('tuscan_rfm.csv')
data['buyer_dummy'] = data['buyer'].apply(lambda x: 1 if x.lower() == 'yes' else 0)
data['freq_dec'] = 9 - mba263.ntile(data['numords'], 10)
response_rate_by_freq_decile = data.groupby('freq_dec')['buyer_dummy'].mean()
print("Response Rates by Frequency Decile:")
print(response_rate_by_freq_decile)

plt.figure(figsize=(10, 6))
response_rate_by_freq_decile.plot(kind='bar', color='grey')
plt.title('Response Rate by Frequency Decile')
plt.xlabel('Frequency Decile')
plt.ylabel('Response Rate')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()
```

```
Response Rates by Frequency Decile:
freq_dec
0    0.053746
1    0.032147
2    0.034144
3    0.029147
4    0.021796
6    0.020365
9    0.015266
Name: buyer_dummy, dtype: float64
```



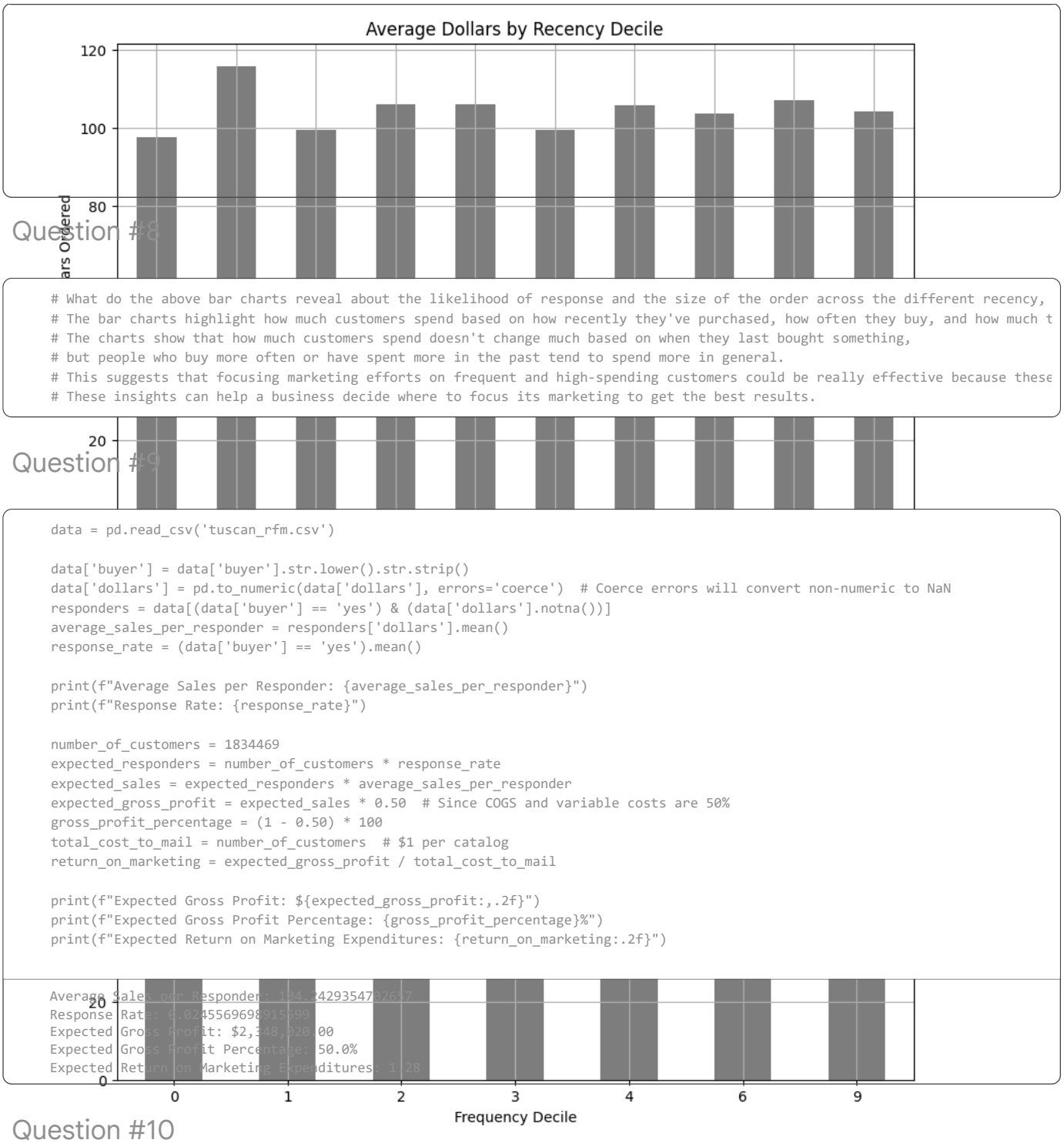

## Question #6

```
data['buyer_dummy'] = data['buyer'].apply(lambda x: 1 if x.lower() == 'yes' else 0)
data['monetary_dec'] = 9 - mba263.ntile(data['totdol'], 10)

response_rate_by_monetary_decile = data.groupby('monetary_dec')['buyer_dummy'].mean()

plt.figure(figsize=(10, 6))
response_rate_by_monetary_decile.plot(kind='bar', color='Grey')
plt.title('Response Rate by Monetary Decile')
plt.xlabel('Monetary Decile')
plt.ylabel('Response Rate')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()
```




## Question #7

```
data = pd.read_csv('tuscan_rfm.csv')

data['rec_dec'] = mba263.ntile(data['last'], 10)
data['freq_dec'] = 9- mba263.ntile(data['numords'], 10)
data['monetary_dec'] = 9- mba263.ntile(data['totdol'], 10)

data_ordered = data[data['buyer'].str.lower() == 'yes']
data_ordered['dollars'] = pd.to_numeric(data_ordered['dollars'], errors='coerce')
bins = range(0, 101, 10)

plt.figure(figsize=(10, 6))
data_ordered.groupby('rec_dec')['dollars'].mean().plot(kind='bar', color='grey')
plt.title('Average Dollars by Recency Decile')
plt.xlabel('Recency Decile')
plt.ylabel('Average Dollars Ordered')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()

plt.figure(figsize=(10, 6))
data_ordered.groupby('freq_dec')['dollars'].mean().plot(kind='bar', color='grey')
plt.title('Average Dollars by Frequency Decile')
plt.xlabel('Frequency Decile')
plt.ylabel('Average Dollars Ordered')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()

plt.figure(figsize=(10, 6))
data_ordered.groupby('monetary_dec')['dollars'].mean().plot(kind='bar', color='grey')
plt.title('Average Dollars by Monetary Decile')
plt.xlabel('Monetary Decile')
plt.ylabel('Average Dollars Ordered')
plt.xticks(rotation=0)
plt.grid(True)
plt.show()
```




## Question #8

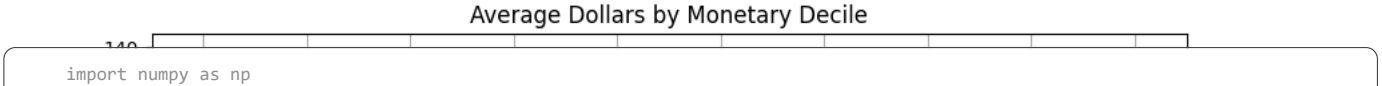
```
# What do the above bar charts reveal about the likelihood of response and the size of the order across the different recency,
# The bar charts highlight how much customers spend based on how recently they've purchased, how often they buy, and how much t
# The charts show that how much customers spend doesn't change much based on when they last bought something,
# but people who buy more often or have spent more in the past tend to spend more in general.
# This suggests that focusing marketing efforts on frequent and high-spending customers could be really effective because these
# These insights can help a business decide where to focus its marketing to get the best results.
```

## Question #9

```
data = pd.read_csv('tuscan_rfm.csv')

data['buyer'] = data['buyer'].str.lower().str.strip()
data['dollars'] = pd.to_numeric(data['dollars'], errors='coerce')  # Coerce errors will convert non-numeric to NaN
responders = data[(data['buyer'] == 'yes') & (data['dollars'].notna())]
average_sales_per_responder = responders['dollars'].mean()
response_rate = (data['buyer'] == 'yes').mean()

print(f"Average Sales per Responder: {average_sales_per_responder}")
print(f"Response Rate: {response_rate}")

number_of_customers = 1834469
expected_responders = number_of_customers * response_rate
expected_sales = expected_responders * average_sales_per_responder
expected_gross_profit = expected_sales * 0.50  # Since COGS and variable costs are 50%
gross_profit_percentage = (1 - 0.50) * 100
total_cost_to_mail = number_of_customers  # $1 per catalog
return_on_marketing = expected_gross_profit / total_cost_to_mail

print(f"Expected Gross Profit: ${expected_gross_profit:,.2f}")
print(f"Expected Gross Profit Percentage: {gross_profit_percentage}%")
print(f"Expected Return on Marketing Expenditures: {return_on_marketing:.2f}")
```

```
Average Sales per Responder: 104.24293547026837
Response Rate: 0.024556969891599
Expected Gross Profit: $2,348,020.00
Expected Gross Profit Percentage: 50.0%
Expected Return on Marketing Expenditures: 1.28
```

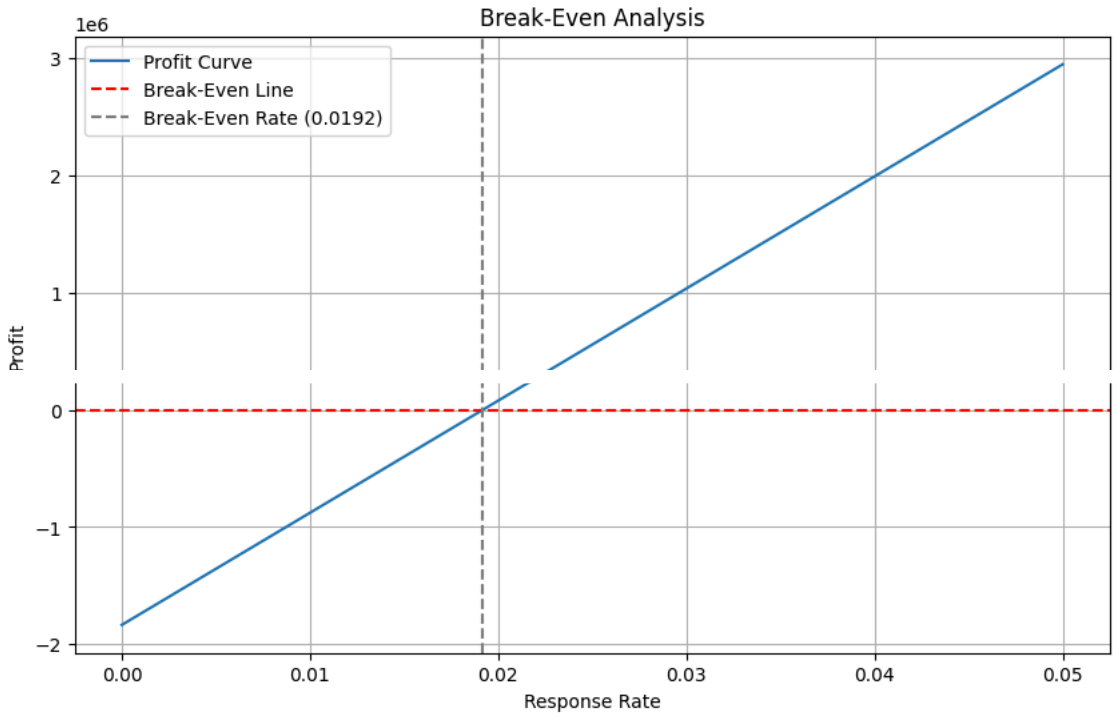
## Question #10




```
import numpy as np

number_of_customers = 1834469
cost_to_mail = 1
total_mailing_cost = number_of_customers * cost_to_mail
average_order_value = 104.24
cogs_rate = 0.50
break_even_response_rate = total_mailing_cost / (number_of_customers * average_order_value * (1 - cogs_rate))
print(f"Break-Even Response Rate: {break_even_response_rate:.4f}")

response_rates = np.linspace(0, 0.05, 100)
profits = [(rate * number_of_customers * average_order_value * (1 - cogs_rate) - total_mailing_cost) for rate in response_rates

plt.figure(figsize=(10, 6))
plt.plot(response_rates, profits, label='Profit Curve')
plt.axhline(0, color='red', linestyle='--', label='Break-Even Line')
```

```
plt.axvline(break_even_response_rate, color='grey', linestyle='--', label=f'Break-Even Rate ({break_even_response_rate:.4f})')
plt.title('Break-Even Analysis')
plt.xlabel('Response Rate')
plt.ylabel('Profit')
plt.legend()
plt.grid(True)
plt.show()
```

```
Break-Even Response Rate: 0.0192
```




## Question #10

```
data = pd.read_csv('tuscan_rfm.csv')
data['purchased'] = data['buyer'].apply(lambda x: 1 if x == 'yes' else 0)
rfm_analysis = data.groupby('rfm1').agg(total_purchases=('purchased', 'sum'), total_customers=('rfm1', 'count'))
rfm_analysis['response_rate'] = rfm_analysis['total_purchases'] / rfm_analysis['total_customers']

print(rfm_analysis[['response_rate']])
```

```
      response_rate
rfm1
111        0.087622
112        0.077886
113        0.065460
114        0.065551
115        0.060858
...             ...
551        0.007177
552        0.013973
553        0.010222
554        0.007849
555        0.013558

[115 rows x 1 columns]
```

## Question #11

```
breakeven_response_rate = 0.0192
profitable_segments = rfm_analysis[rfm_analysis['response_rate'] >= breakeven_response_rate]
total_customers_profitable = profitable_segments['total_customers'].sum()
```

```
total_expected_buyers = profitable_segments.apply(lambda x: x['total_customers'] * x['response_rate'], axis=1).sum()
average_order_value = 104.24
cogs_rate = 0.50

data['buyer'] = data['buyer'].apply(lambda x: 1 if x == 'yes' else 0)

response_rate = data['buyer'].mean() * 100
expected_gross_sales = total_expected_buyers * average_order_value
expected_gross_profit = expected_gross_sales * (1 - cogs_rate) - total_customers_profitable
cost_to_mail_profitable = total_customers_profitable
expected_gross_profit_percentage = (expected_gross_profit / expected_gross_sales) * 100
expected_rom_expenditures = expected_gross_profit / cost_to_mail_profitable

print(f"Total Customers in Profitable Segments: {total_customers_profitable}")
print(f"Total Expected Buyers: {total_expected_buyers}")
print(f"Expected Gross Sales: ${expected_gross_sales:,.2f}")
print(f"Expected Gross Profit: ${expected_gross_profit:,.2f}")
print(f"Expected Gross Profit Percentage: {expected_gross_profit_percentage:.2f}%")
print(f"Expected Return on Marketing Expenditures: {expected_rom_expenditures:.2f}")
```

```
Total Customers in Profitable Segments: 52289
Total Expected Buyers: 1757.0
Expected Gross Sales: $183,149.68
Expected Gross Profit: $39,285.84
Expected Gross Profit Percentage: 21.45%
Expected Return on Marketing Expenditures: 0.75
```

## Question #12

```
data = pd.read_csv('tuscan_rfm.csv')
print(data[['rfm1', 'rfm2']].head(20))
```

```
    rfm1  rfm2
```