```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the Excel data
file_path = '/content/marketing_analysis.xlsx'  # Adjust with actual
path
data = pd.ExcelFile(file_path)
df = pd.read_excel(data, sheet_name='Campaign_Data')  # Replace
'Sheet1' with the actual sheet name if needed
```

Data Handling:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 13 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Campaign_ID         2000 non-null   object
 1   Campaign_Name       2000 non-null   object
 2   Marketing_Channel   2000 non-null   object
 3   Start_Date          2000 non-null   datetime64[ns]
 4   End_Date            2000 non-null   datetime64[ns]
 5   Impressions         2000 non-null   float64
 6   Clicks              2000 non-null   float64
 7   Conversions         2000 non-null   float64
 8   Total_Spend         2000 non-null   float64
 9   Revenue_Generated   2000 non-null   float64
 10  Location            2000 non-null   object
 11  Age_Group           2000 non-null   object
 12  Gender              2000 non-null   object
dtypes: datetime64[ns](2), float64(5), object(6)
memory usage: 203.2+ KB
```

```
print(df.shape)
df.head()
```

```
(2000, 13)
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 2000,\n  \"fields\":
[\n    {\n      \"column\": \"Campaign_ID\",\n      \"properties\": {\
n        \"dtype\": \"string\",\n        \"num_unique_values\": 2000,\
n        \"samples\": [\n          \"CMP1861\",\n
\"CMP354\",\n          \"CMP1334\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Campaign_Name\",\n
```

```
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 6,\n        \"samples\": [\n
\"Seasonal Steals\",\n          \"Cart to Couch\",\n          \"Flash
Sale Frenzy\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Marketing_Channel\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          \"Paid Ads\",\n          \"Social Media\",\n
\"Influencer Marketing\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Start_Date\",\n      \"properties\": {\n
\"dtype\": \"date\",\n        \"min\": \"2022-05-11 00:00:00\",\n
\"max\": \"2023-03-05 00:00:00\",\n        \"num_unique_values\":
299,\n        \"samples\": [\n          \"2022-06-20 00:00:00\",\n
\"2023-01-05 00:00:00\",\n          \"2022-07-03 00:00:00\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"End_Date\",\n      \"properties\": {\n        \"dtype\": \"date\",\n
\"min\": \"2024-01-06 00:00:00\",\n        \"max\": \"2024-11-23
00:00:00\",\n        \"num_unique_values\": 319,\n        \"samples\":
[\n          \"2024-05-05 00:00:00\",\n          \"2024-05-29
00:00:00\",\n          \"2024-08-30 00:00:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Impressions\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
22628.77722458958,\n        \"min\": 1619.2,\n        \"max\":
79894.40000000001,\n        \"num_unique_values\": 1969,\n
\"samples\": [\n          33043.200000000004,\n          36104.0,\n
79273.6\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Clicks\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 17314.42426601064,\n        \"min\": 169.60000000000002,\n
\"max\": 77414.40000000001,\n        \"num_unique_values\": 1938,\n
\"samples\": [\n          6836.8,\n          22785.600000000002,\n
21403.2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Conversions\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 12024.111326176884,\n        \"min\":
3.2,\n        \"max\": 70790.40000000001,\n
\"num_unique_values\": 1793,\n        \"samples\": [\n
4412.8,\n          3371.2000000000003,\n          42812.8\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Total_Spend\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
2245.111135150541,\n        \"min\": 161.04000000000002,\n
\"max\": 7999.664000000001,\n        \"num_unique_values\": 1993,\n
\"samples\": [\n          7851.024000000001,\n          1078.432,\n
4931.024\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"Revenue_Generated\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 6922.119092849307,\n        \"min\":
237.72800000000004,\n        \"max\": 31791.344,\n
\"num_unique_values\": 1998,\n        \"samples\": [\n
9466.032000000001,\n          7770.0,\n          17724.624\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Location\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"Gala\\u021bi\",\n          \"Suceava\",\n
\"Bac\\u0103u\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Age_Group\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"45-54\",\n          \"35-44\",\n          \"18-24\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Gender\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
2,\n        \"samples\": [\n          \"Female\",\n          \"Male\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```


```
df.describe()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Start_Date\",\n      \"properties\": {\n
\"dtype\": \"date\",\n        \"min\": \"1970-01-01
00:00:00.000002\",\n        \"max\": \"2023-03-05 00:00:00\",\n
\"num_unique_values\": 7,\n        \"samples\": [\n
\"2000\",\n          \"2022-10-06 18:20:09.600000\",\n
\"2022-12-22 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"End_Date\",\n      \"properties\": {\n        \"dtype\":
\"date\",\n        \"min\": \"1970-01-01 00:00:00.000002\",\n
\"max\": \"2024-11-23 00:00:00\",\n        \"num_unique_values\": 7,\n
\"samples\": [\n          \"2000\",\n          \"2024-06-13
12:41:02.400000\",\n          \"2024-08-28 00:00:00\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Impressions\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
27301.594053631063,\n        \"min\": 1619.2,\n        \"max\":
79894.40000000001,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          40096.0672,\n          59463.200000000004,\n
2000.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Clicks\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 24959.883184369413,\n        \"min\": 169.60000000000002,\n
\"max\": 77414.40000000001,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          20140.6688,\n          30935.2,\n
```

```
2000.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Conversions\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 23329.629696772416,\n        \"min\":
3.2,\n        \"max\": 70790.40000000001,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
10127.8264,\n          14076.400000000001,\n          2000.0\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total_Spend\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 2510.1679480706644,\n        \"min\":
161.04000000000002,\n        \"max\": 7999.664000000001,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
4078.5991280000003,\n          6016.0,\n          2000.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Revenue_Generated\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
9969.477743424713,\n        \"min\": 237.72800000000004,\n
\"max\": 31791.344,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          10610.86952,\n          15062.004,\n
2000.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```


```
df.isnull().sum().sort_values(ascending=False).head() #no nan values
```

```
Campaign_ID          0
Campaign_Name        0
Marketing_Channel    0
Start_Date           0
End_Date             0
dtype: int64
```

```
df.duplicated().sum() #no duplicated values
```

```
0
```

Calculate Metrics

```
# According to the Excel file columns: 'Clicks', 'Conversions',
'Total_Spend', 'Revenue_Generated', and 'Impressions'
df['CVR'] = (df['Conversions'] / df['Clicks']) * 100  # Conversion
Rate in percentage

df['CPC'] = df['Total_Spend'] / df['Clicks']  # Cost per Click

df['CPA'] = df['Total_Spend'] / df['Conversions']  # Cost per
Conversion
```

```
df['ROAS'] = df['Revenue_Generated'] / df['Total_Spend']  # Return on Ad Spend

df.head(2)
```

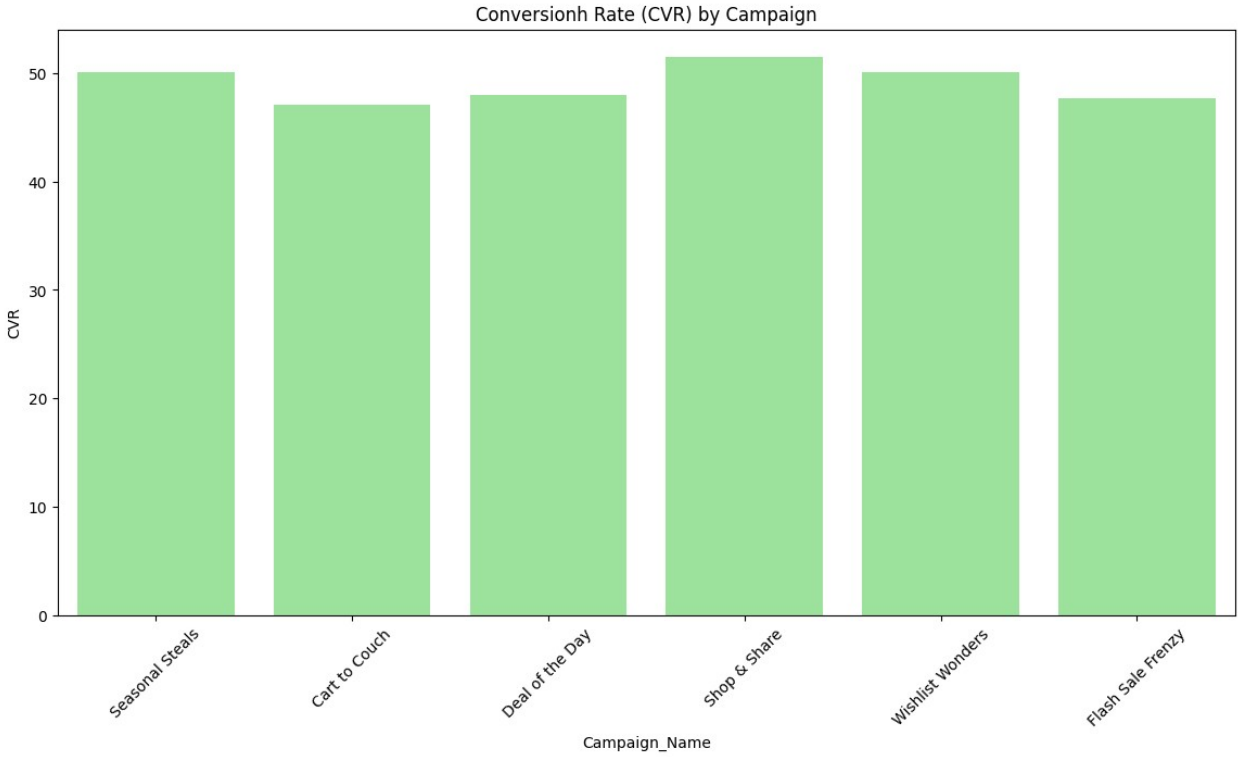

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 2000,\n  \"fields\": [\n    {\n      \"column\": \"Campaign_ID\",\n      \"properties\": {\n        \"dtype\": \"string\",\n        \"num_unique_values\": 2000,\n        \"samples\": [\n          \"CMP1861\",\n          \"CMP354\",\n          \"CMP1334\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Campaign_Name\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 6,\n        \"samples\": [\n          \"Seasonal Steals\",\n          \"Cart to Couch\",\n          \"Flash Sale Frenzy\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Marketing_Channel\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 4,\n        \"samples\": [\n          \"Paid Ads\",\n          \"Social Media\",\n          \"Influencer Marketing\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Start_Date\",\n      \"properties\": {\n        \"dtype\": \"date\",\n        \"min\": \"2022-05-11 00:00:00\",\n        \"max\": \"2023-03-05 00:00:00\",\n        \"num_unique_values\": 299,\n        \"samples\": [\n          \"2022-06-20 00:00:00\",\n          \"2023-01-05 00:00:00\",\n          \"2022-07-03 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"End_Date\",\n      \"properties\": {\n        \"dtype\": \"date\",\n        \"min\": \"2024-01-06 00:00:00\",\n        \"max\": \"2024-11-23 00:00:00\",\n        \"num_unique_values\": 319,\n        \"samples\": [\n          \"2024-05-05 00:00:00\",\n          \"2024-05-29 00:00:00\",\n          \"2024-08-30 00:00:00\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Impressions\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 22628.77722458958,\n        \"min\": 1619.2,\n        \"max\": 79894.40000000001,\n        \"num_unique_values\": 1969,\n        \"samples\": [\n          33043.200000000004,\n          36104.0,\n          79273.6\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Clicks\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 17314.42426601064,\n        \"min\": 169.60000000000002,\n        \"max\": 77414.40000000001,\n        \"num_unique_values\": 1938,\n        \"samples\": [\n          6836.8,\n          22785.600000000002,\n          21403.2\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Conversions\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 12024.111326176884,\n        \"min\":
```

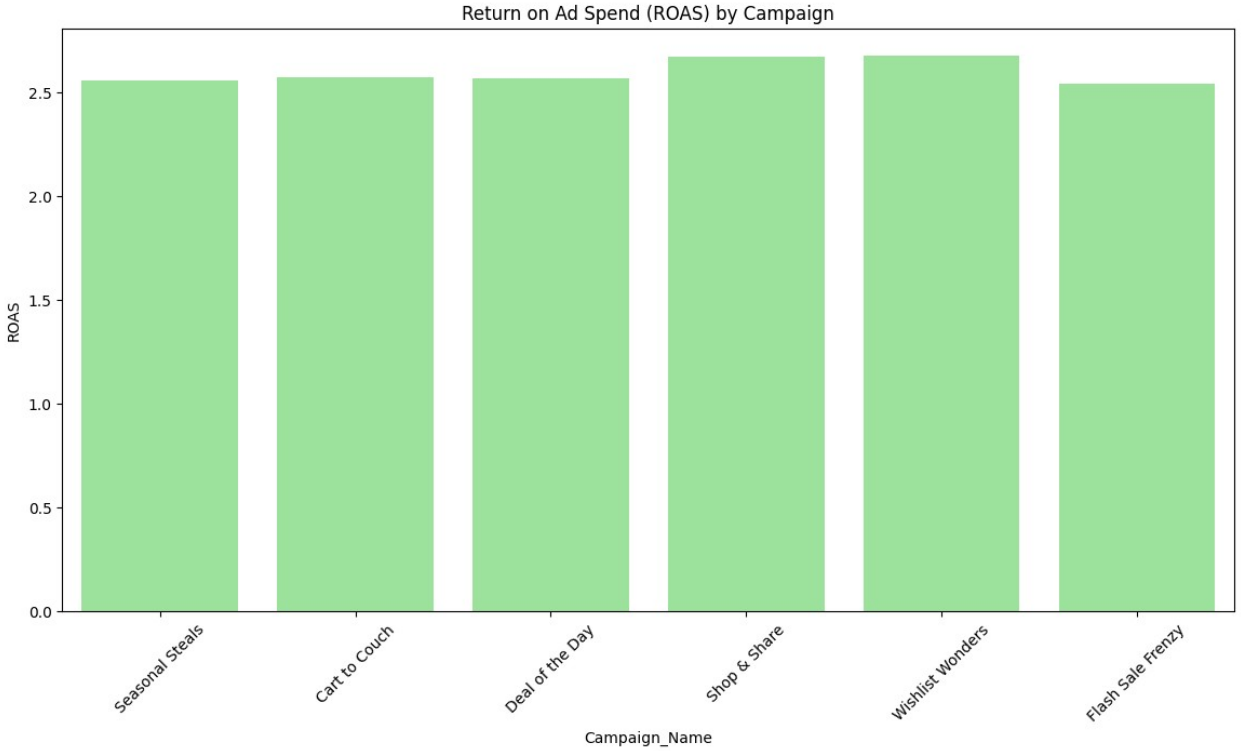
```
3.2,\n        \"max\": 70790.40000000001,\n
\"num_unique_values\": 1793,\n        \"samples\": [\n
4412.8,\n          3371.2000000000003,\n          42812.8\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Total_Spend\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
2245.111135150541,\n        \"min\": 161.04000000000002,\n
\"max\": 7999.664000000001,\n        \"num_unique_values\": 1993,\n
\"samples\": [\n          7851.024000000001,\n          1078.432,\n
4931.024\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Revenue_Generated\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 6922.119092849307,\n        \"min\":
237.72800000000004,\n        \"max\": 31791.344,\n
\"num_unique_values\": 1998,\n        \"samples\": [\n
9466.032000000001,\n          7770.0,\n          17724.624\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Location\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"Gala\\u021bi\",\n          \"Suceava\",\n
\"Bac\\u0103u\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Age_Group\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 5,\n        \"samples\":
[\n          \"45-54\",\n          \"35-44\",\n          \"18-24\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Gender\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
2,\n        \"samples\": [\n          \"Female\",\n          \"Male\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CVR\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 28.782435850904527,\n        \"min\": 0.02978850163836759,\n
\"max\": 99.98504337421478,\n        \"num_unique_values\": 1998,\n
\"samples\": [\n          60.44113739038002,\n
63.21639458145189\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CPC\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 2.249974759369632,\n        \"min\": 0.004079337569805507,\n
\"max\": 37.08241379310345,\n        \"num_unique_values\": 2000,\n
\"samples\": [\n          0.537286411716843,\n
1.5053296891014694\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"CPA\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 82.66535202879494,\n        \"min\": 0.004979824182158812,\n
\"max\": 2499.22,\n        \"num_unique_values\": 2000,\n
\"samples\": [\n          18.342361111111114,\n
7.368060200668897\n        ],\n        \"semantic_type\": \"\",\n
```

```
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"ROAS\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0.8012533158742741,\n        \"min\": 1.2001578531965271,\n
\"max\": 3.997877281092016,\n        \"num_unique_values\": 2000,\n
\"samples\": [\n          1.8965736569113691,\n
3.8161753024216423\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```


Visualize Campaign Performance

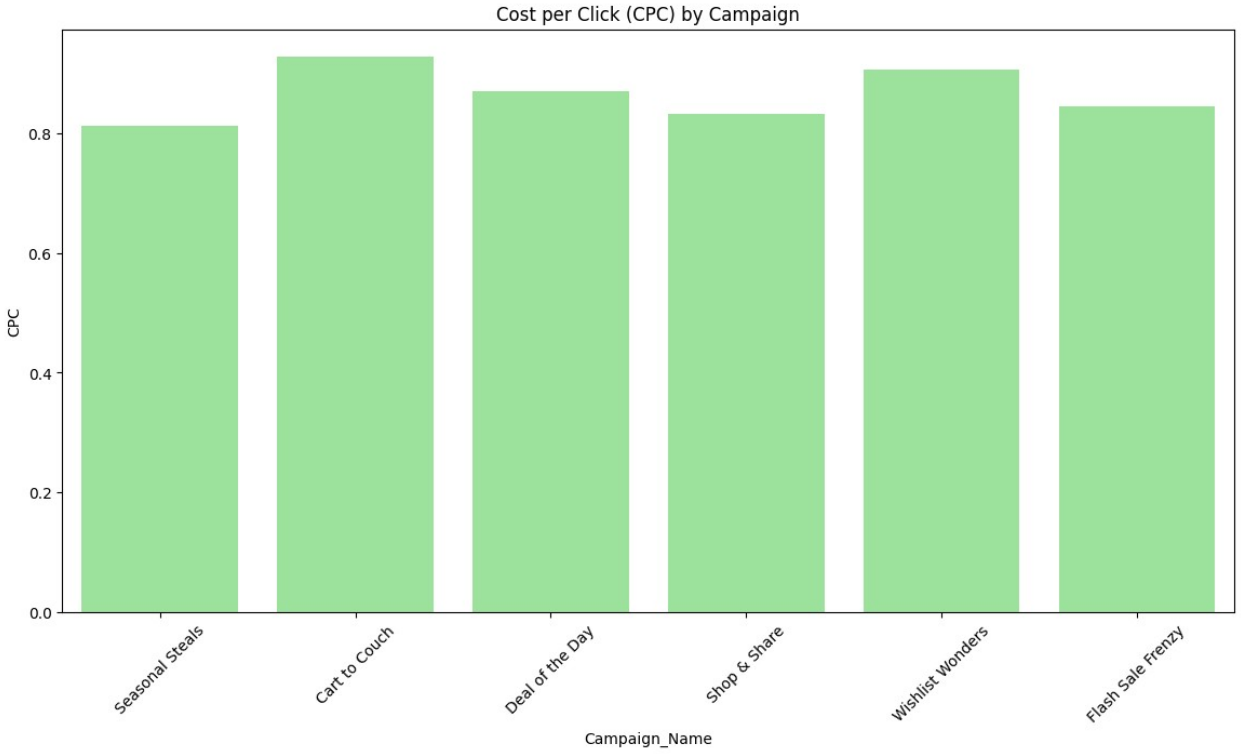
```
# Bar chart for CVR and ROAS across campaigns
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign_Name', y='CVR', data=df,
color='lightgreen',errorbar=None)
plt.title('Conversionh Rate (CVR) by Campaign')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign_Name', y='ROAS', data=df,
color='lightgreen',errorbar=None)
plt.title('Return on Ad Spend (ROAS) by Campaign')
plt.xticks(rotation=45)
plt.show()
```

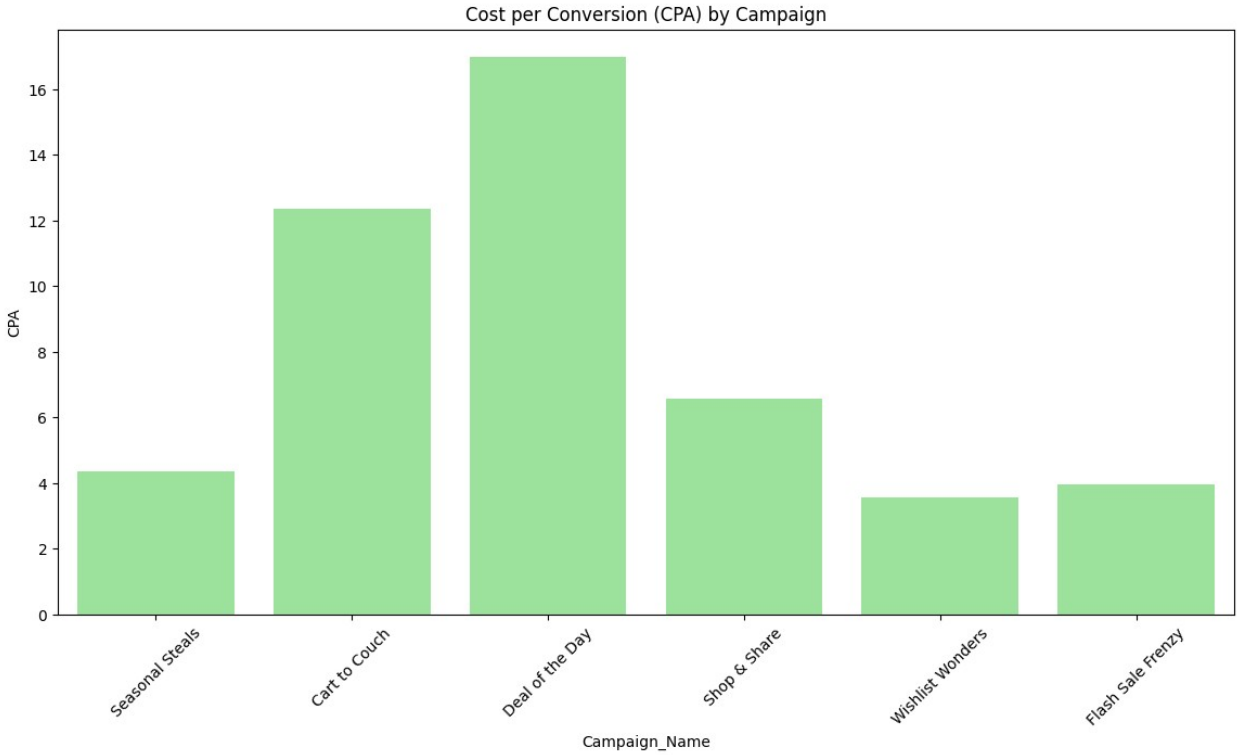




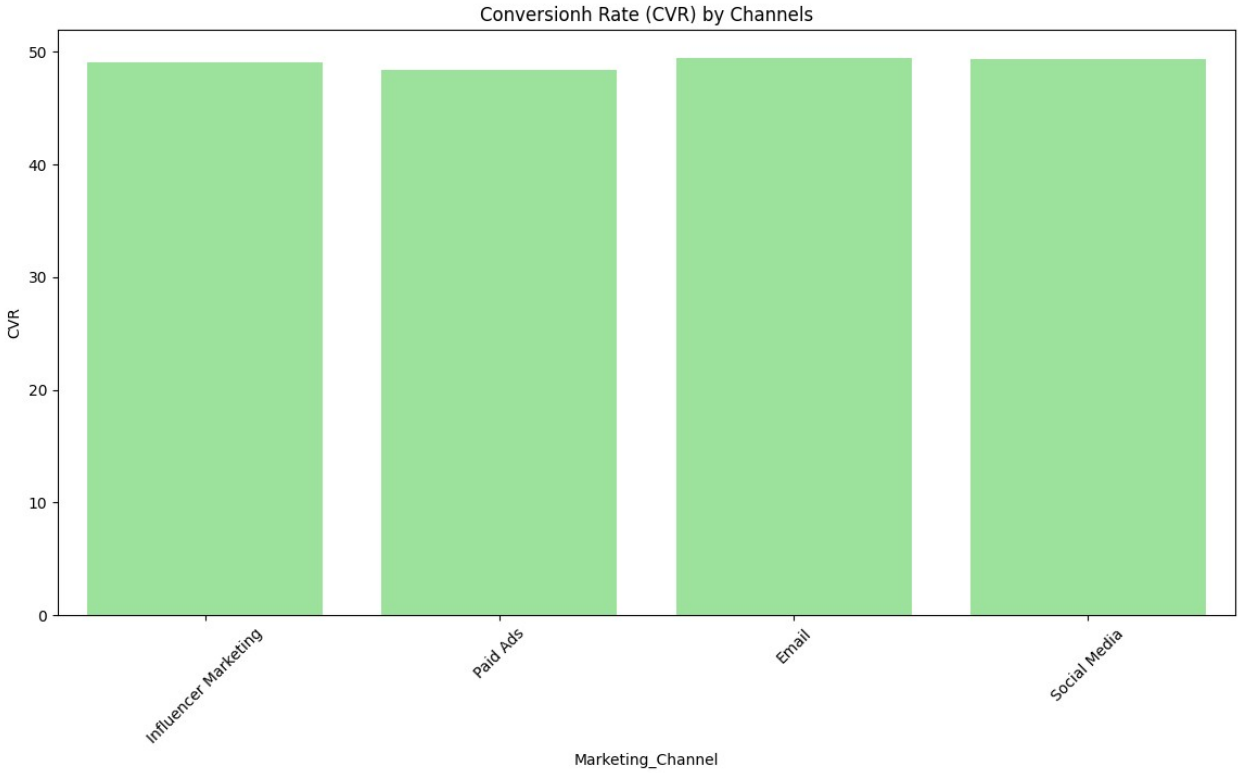

```
# Bar chart for CPC and CPA across campaigns
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign_Name', y='CPC', data=df,
```

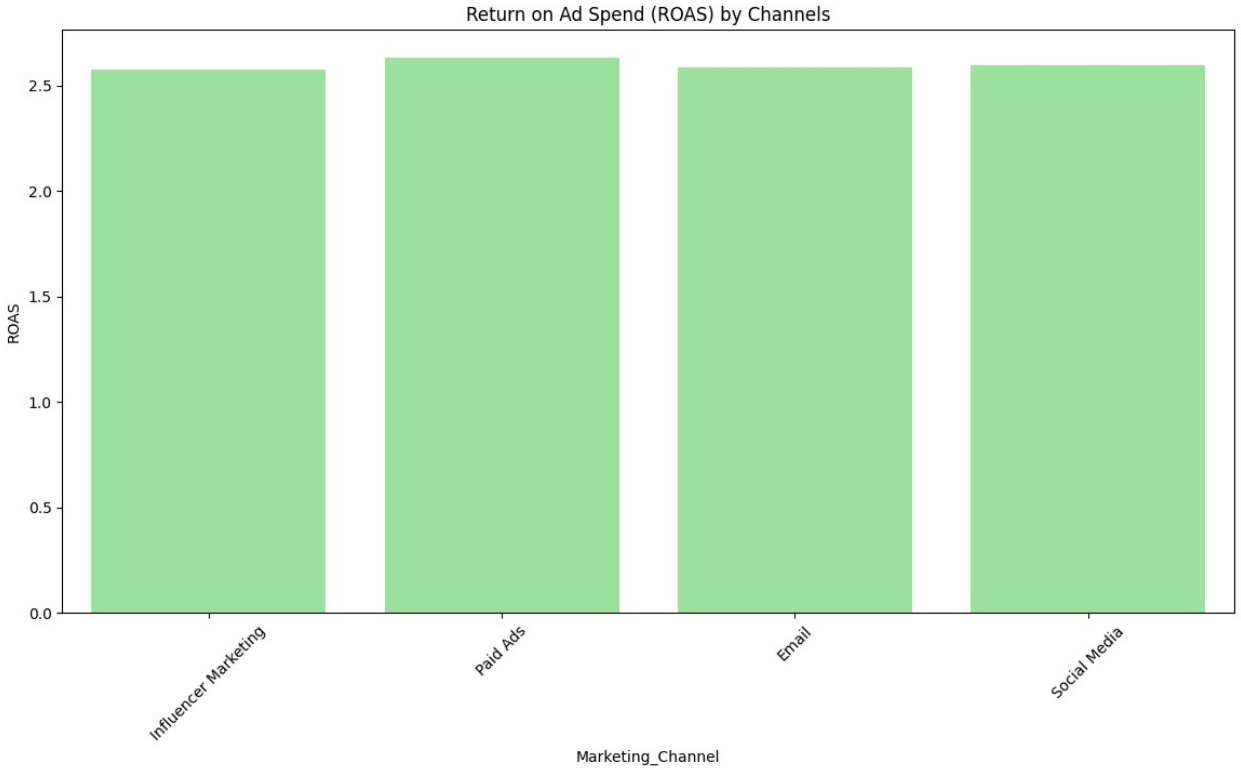
```
color='lightgreen',errorbar=None)
plt.title('Cost per Click (CPC) by Campaign')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign_Name', y='CPA', data=df,
color='lightgreen',errorbar=None)
plt.title('Cost per Conversion (CPA) by Campaign')
plt.xticks(rotation=45)
plt.show()
```






```
# Bar chart for CVR and ROAS across Channels
plt.figure(figsize=(14, 7))
sns.barplot(x='Marketing_Channel', y='CVR', data=df,
color='lightgreen',errorbar=None)
plt.title('Conversionh Rate (CVR) by Channels')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(14, 7))
sns.barplot(x='Marketing_Channel', y='ROAS', data=df,
color='lightgreen',errorbar=None)
plt.title('Return on Ad Spend (ROAS) by Channels')
plt.xticks(rotation=45)
plt.show()
```



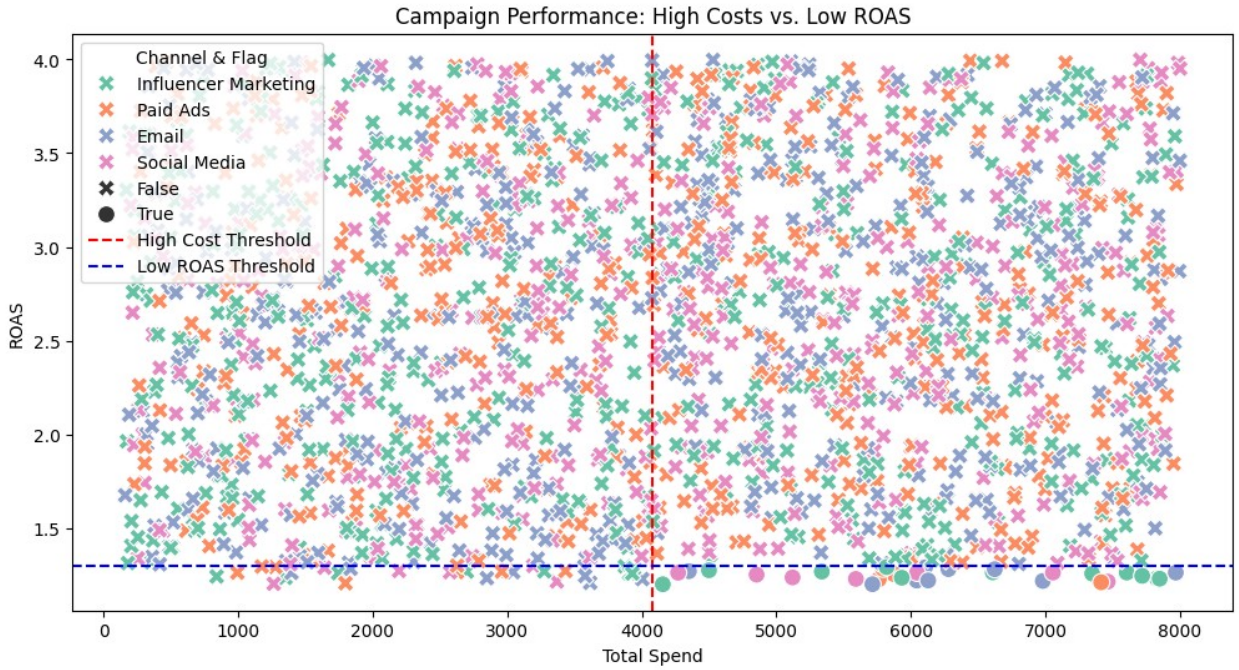



```
# Define thresholds for high cost and low ROAS
high_cost_threshold = df['Total_Spend'].mean()  # average spend
```

```
low_roas_threshold = df['ROAS'].mean()/2  # ROAS below this threshold
is unprofitable
print(low_roas_threshold)
# Filter campaigns with high costs and low ROAS
high_cost_low_roas = df[(df['Total_Spend'] > high_cost_threshold) &
(df['ROAS'] < low_roas_threshold)]
```

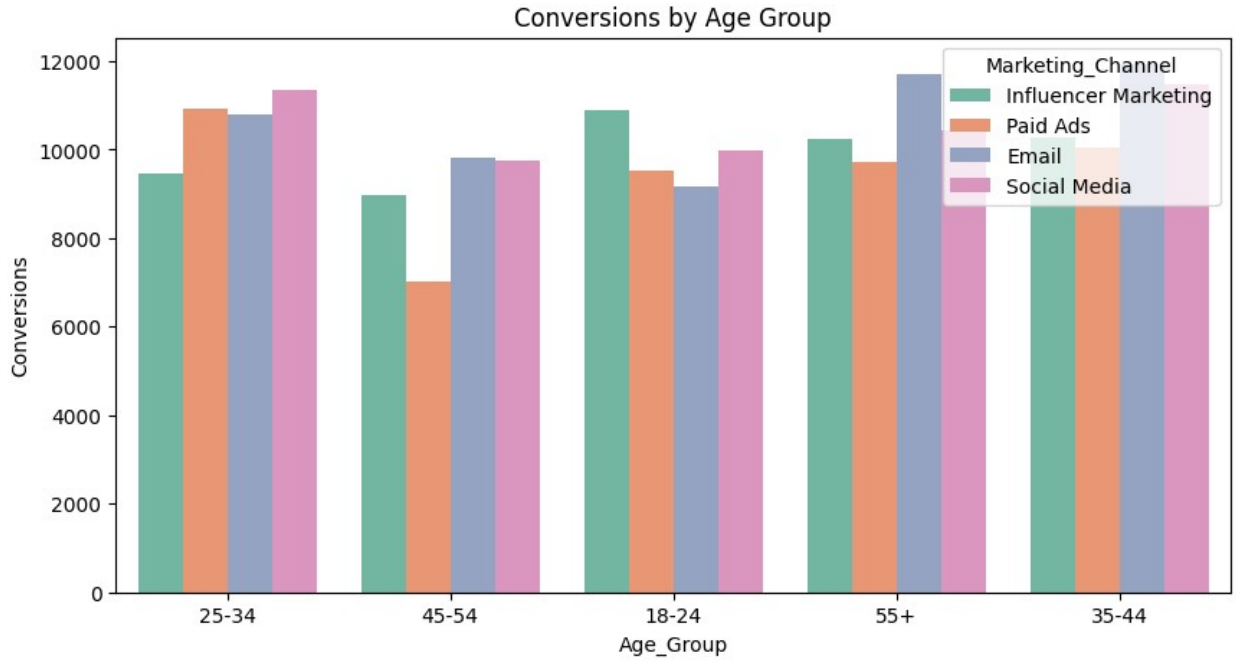
```
1.2982090836921296
```

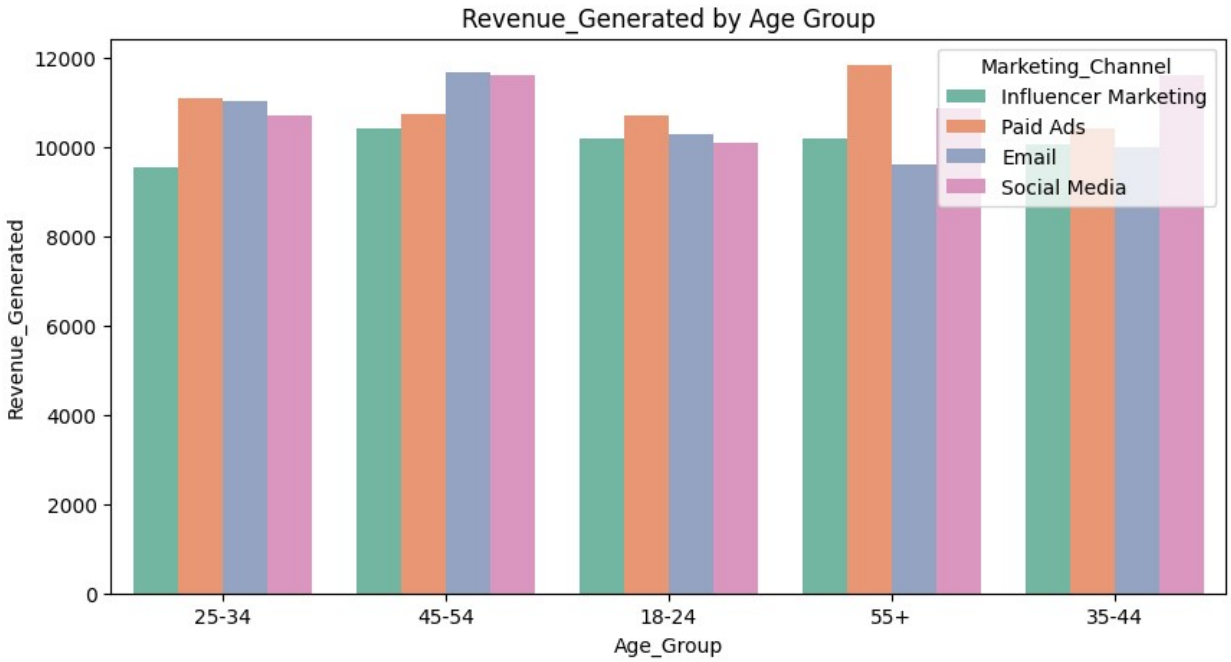
```
# Plot the data
plt.figure(figsize=(12, 6))
sns.scatterplot(data=df,x='Total_Spend',
y='ROAS',hue='Marketing_Channel',
    style=df.index.isin(high_cost_low_roas.index),
    markers={True: 'o', False: 'X'},palette='Set2',s=100)
# Highlight campaigns with high costs and low ROAS
plt.axvline(high_cost_threshold, color='red', linestyle='--',
label='High Cost Threshold')
plt.axhline(low_roas_threshold, color='blue', linestyle='--',
label='Low ROAS Threshold')
# Add labels and title
plt.title('Campaign Performance: High Costs vs. Low ROAS')
plt.xlabel('Total Spend')
plt.ylabel('ROAS')
plt.legend(title='Channel & Flag')
plt.show()
```

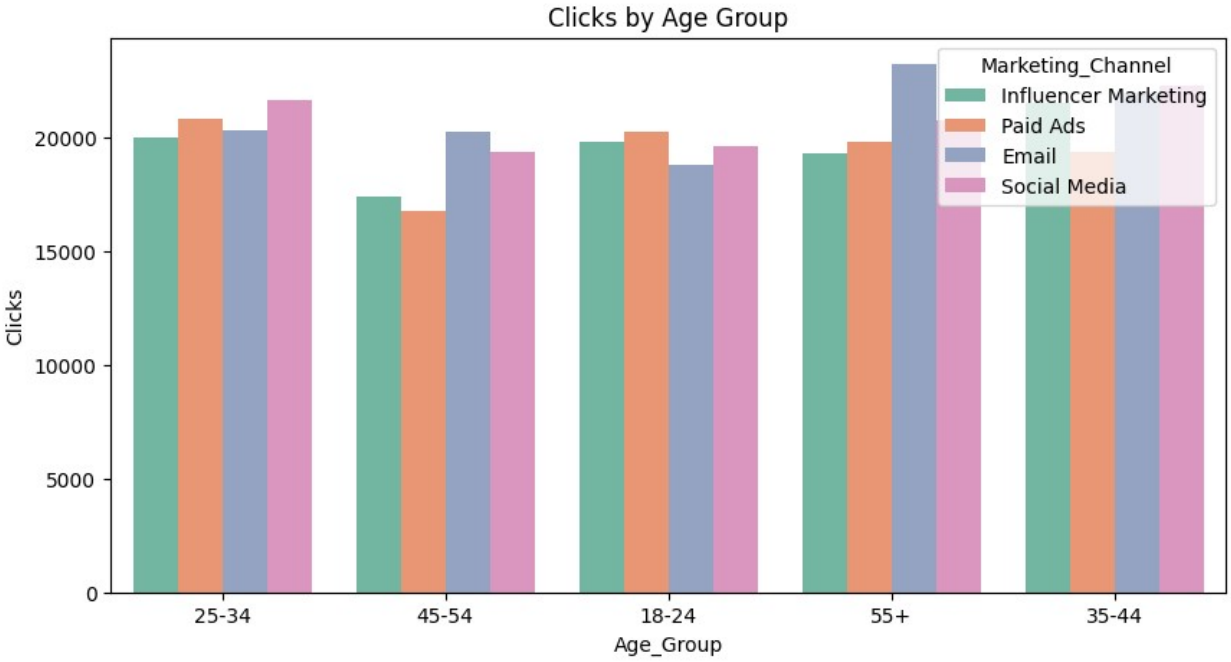


Demographic Segmentation

```
# Group by demographics and analyze performance
# 1- Age:
plt.figure(figsize=(10, 5))
sns.barplot(x='Age_Group', y='Conversions',
data=df,palette='Set2',hue='Marketing_Channel',errorbar=None)
plt.title('Conversions by Age Group')
plt.show()

plt.figure(figsize=(10, 5))
sns.barplot(x='Age_Group', y='Revenue_Generated',
data=df,palette='Set2', hue='Marketing_Channel',errorbar=None)
plt.title('Revenue_Generated by Age Group')
plt.show()

plt.figure(figsize=(10, 5))
sns.barplot(x='Age_Group', y='Clicks',
data=df,palette='Set2',hue='Marketing_Channel',errorbar=None)
plt.title('Clicks by Age Group')
plt.show()
```

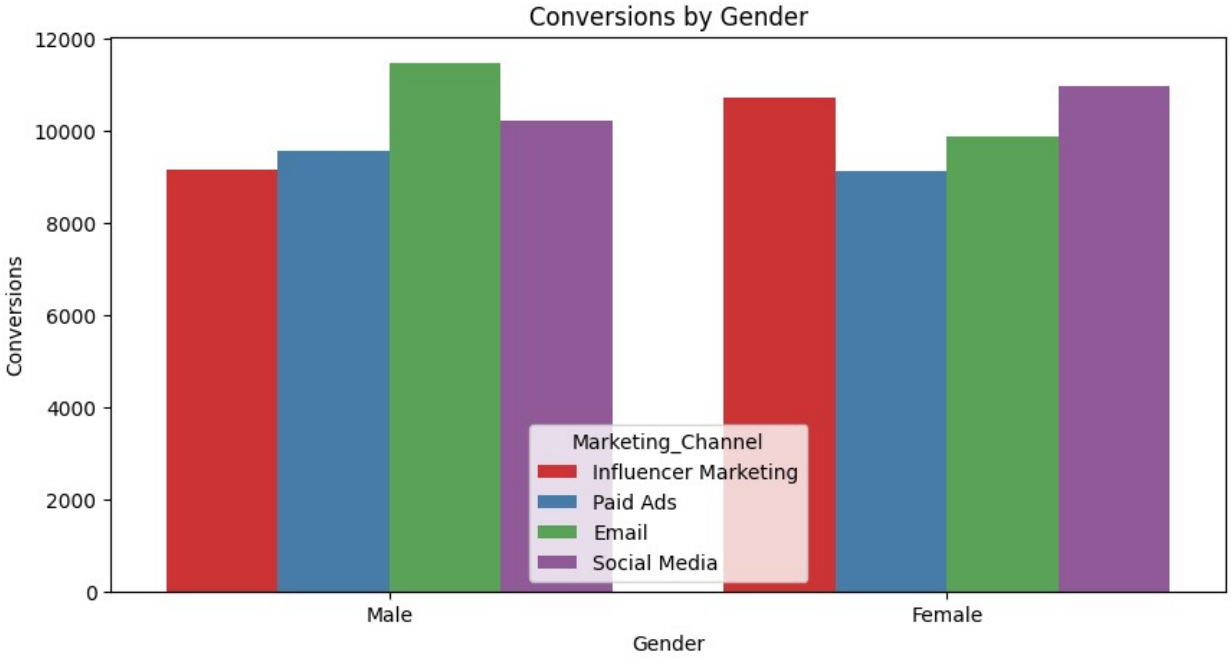


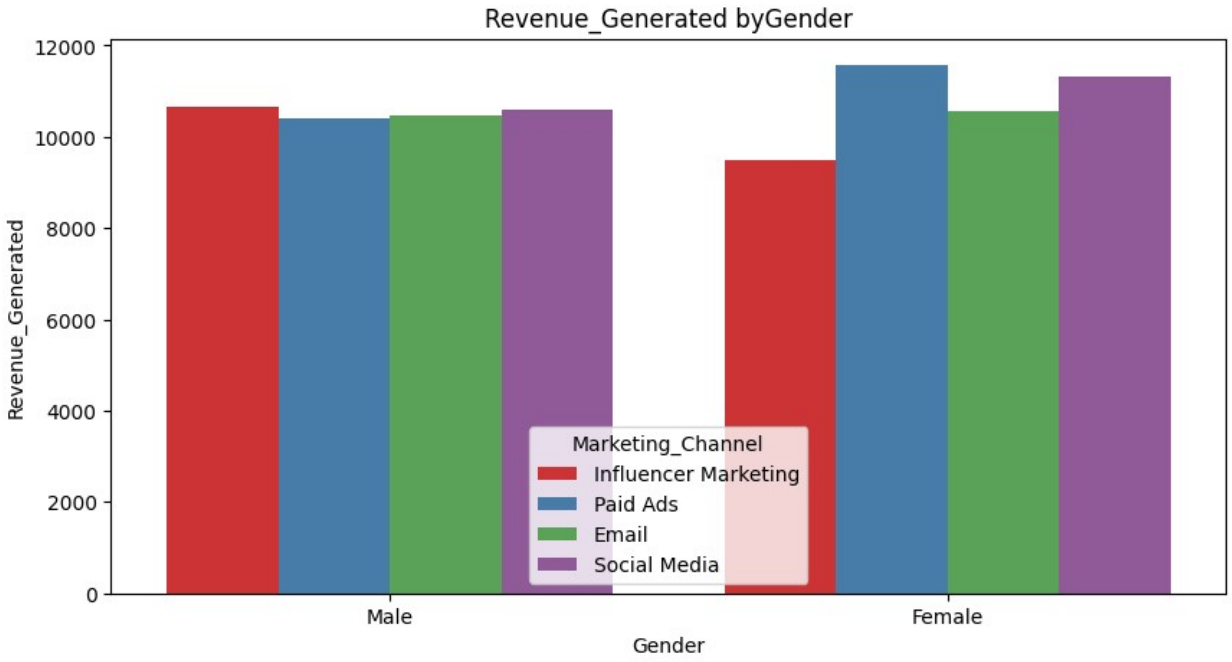


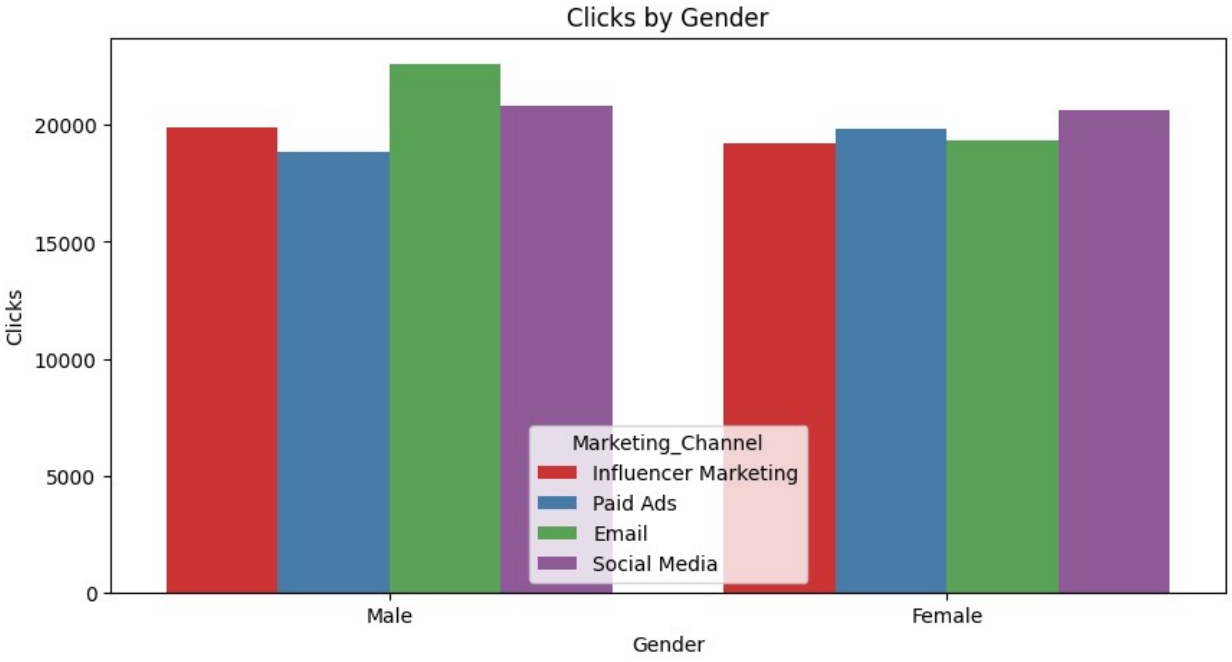



```
# Group by demographics and analyze performance
# 2- Gender:
plt.figure(figsize=(10, 5))
sns.barplot(x='Gender', y='Conversions',
data=df,palette='Set1',hue='Marketing_Channel',errorbar=None)
plt.title('Conversions by Gender')
plt.show()
```

```
plt.figure(figsize=(10, 5))
sns.barplot(x='Gender', y='Revenue_Generated', data=df,palette='Set1',
hue='Marketing_Channel',errorbar=None)
plt.title('Revenue_Generated byGender')
plt.show()

plt.figure(figsize=(10, 5))
sns.barplot(x='Gender', y='Clicks',
data=df,palette='Set1',hue='Marketing_Channel',errorbar=None)
plt.title('Clicks by Gender')
plt.show()
```

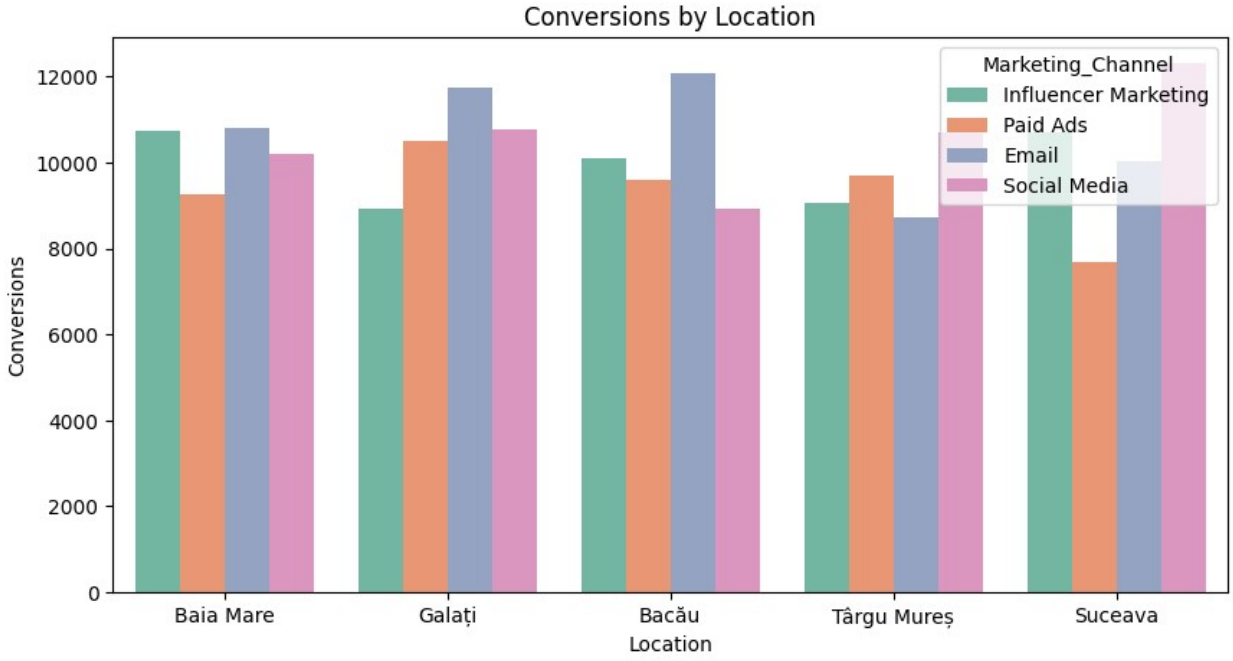


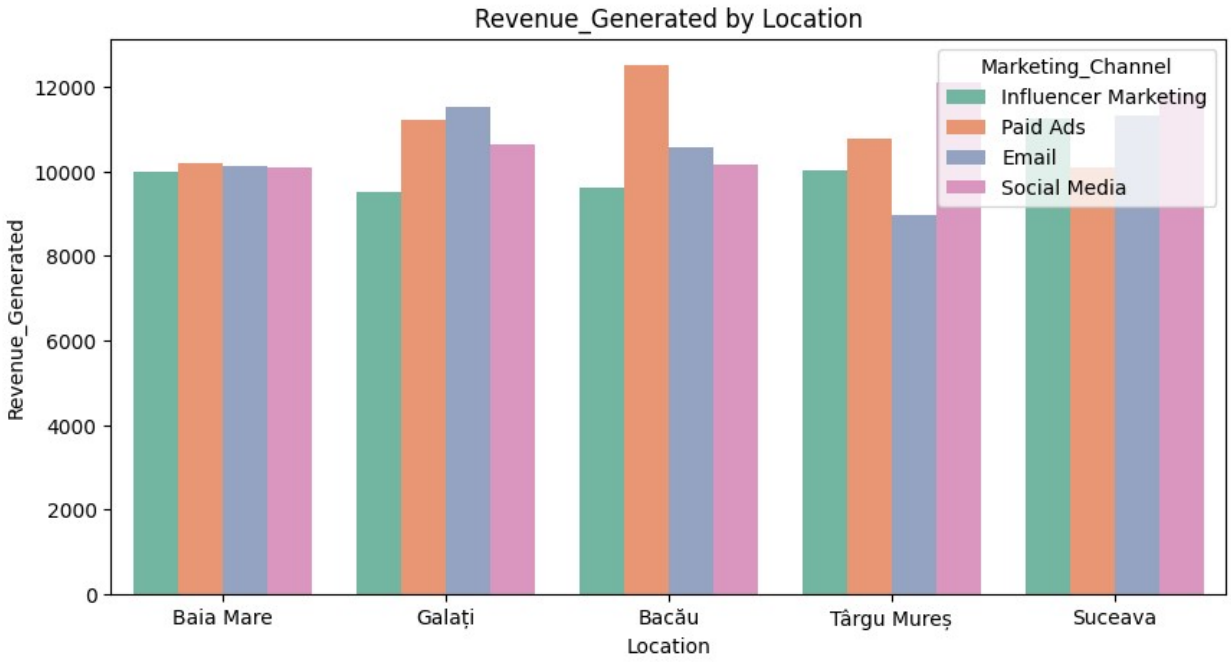


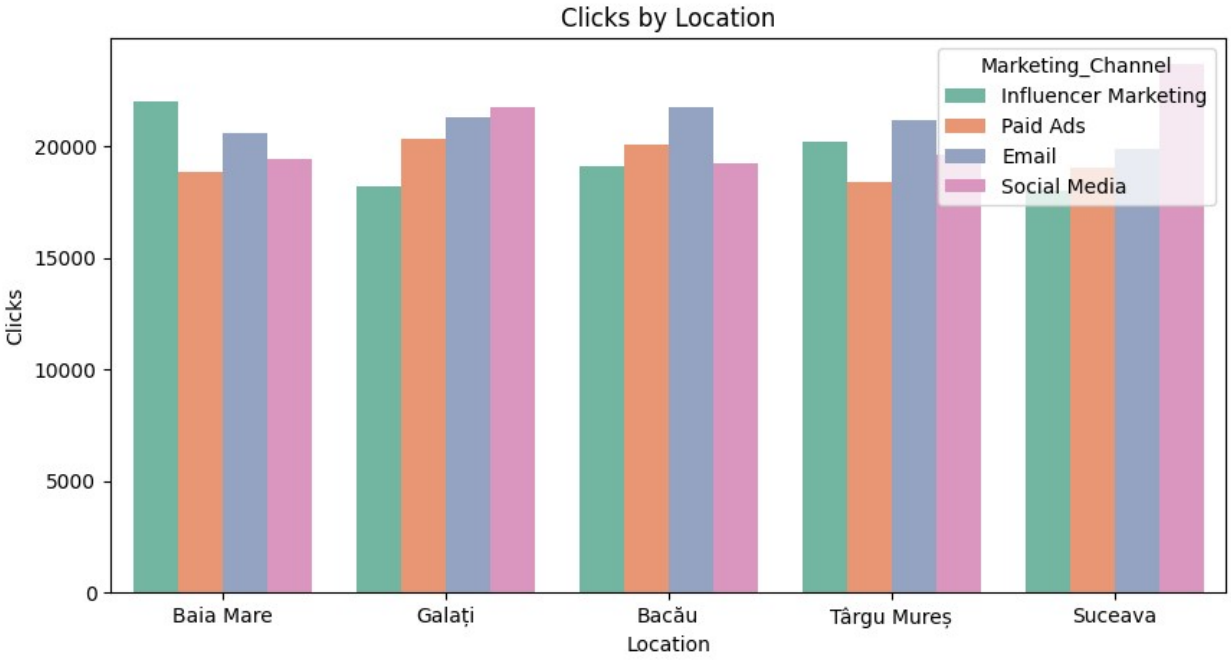



```
# Group by demographics and analyze performance
# 3- Location:
plt.figure(figsize=(10, 5))
sns.barplot(x='Location', y='Conversions',
data=df,palette='Set2',hue='Marketing_Channel',errorbar=None)
plt.title('Conversions by Location')
plt.show()
```

```
plt.figure(figsize=(10, 5))
sns.barplot(x='Location', y='Revenue_Generated',
data=df,palette='Set2', hue='Marketing_Channel',errorbar=None)
plt.title('Revenue_Generated by Location')
plt.show()

plt.figure(figsize=(10, 5))
sns.barplot(x='Location', y='Clicks',
data=df,palette='Set2',hue='Marketing_Channel',errorbar=None)
plt.title('Clicks by Location')
plt.show()
```







4. Time-Based Analysis

```
# Sample data
campaign_data = pd.DataFrame({
    'Campaign_ID': df['Campaign_ID'],
    'Start_Date': df['Start_Date'],
    'End_Date': df['End_Date'],
```

```
    'Total_Spend': df['Total_Spend'],
    'Revenue_Generated': df['Revenue_Generated'],
    'ROAS': df['ROAS']
})

# Convert start and end dates to datetime
campaign_data['Start_Date'] = pd.to_datetime(df['Start_Date'])
campaign_data['End_Date'] = pd.to_datetime(df['End_Date'])

# Expand the data to include all days within the campaign period
expanded_data = []
for index, row in campaign_data.iterrows():
    date_range = pd.date_range(row['Start_Date'], row['End_Date'],
freq='D')
    for date in date_range:
        expanded_data.append({
            'Campaign_ID': row['Campaign_ID'],
            'Date': date,
            'Total_Spend': row['Total_Spend'],
            'Revenue_Generated': row['Revenue_Generated'],
            'ROAS': row['ROAS']
        })

# Convert the list of dicts to a DataFrame
expanded_campaign_data = pd.DataFrame(expanded_data)

# Display the expanded data
print(expanded_campaign_data.head())
```

```
  Campaign_ID       Date  Total_Spend  Revenue_Generated      ROAS
0        CMP1 2022-08-01     2316.656           8879.568  3.832925
1        CMP1 2022-08-02     2316.656           8879.568  3.832925
2        CMP1 2022-08-03     2316.656           8879.568  3.832925
3        CMP1 2022-08-04     2316.656           8879.568  3.832925
4        CMP1 2022-08-05     2316.656           8879.568  3.832925
```

```
expanded_campaign_data.shape
```

```
(1233529, 5)
```

```
expanded_campaign_data['Month'] =
expanded_campaign_data['Date'].dt.to_period('M')

# Aggregate by month
monthly_performance = expanded_campaign_data.groupby('Month').agg(
    total_spend=('Total_Spend', 'sum'),
    total_revenue=('Revenue_Generated', 'sum'),
    average_roas=('ROAS', 'mean')
).reset_index()
```

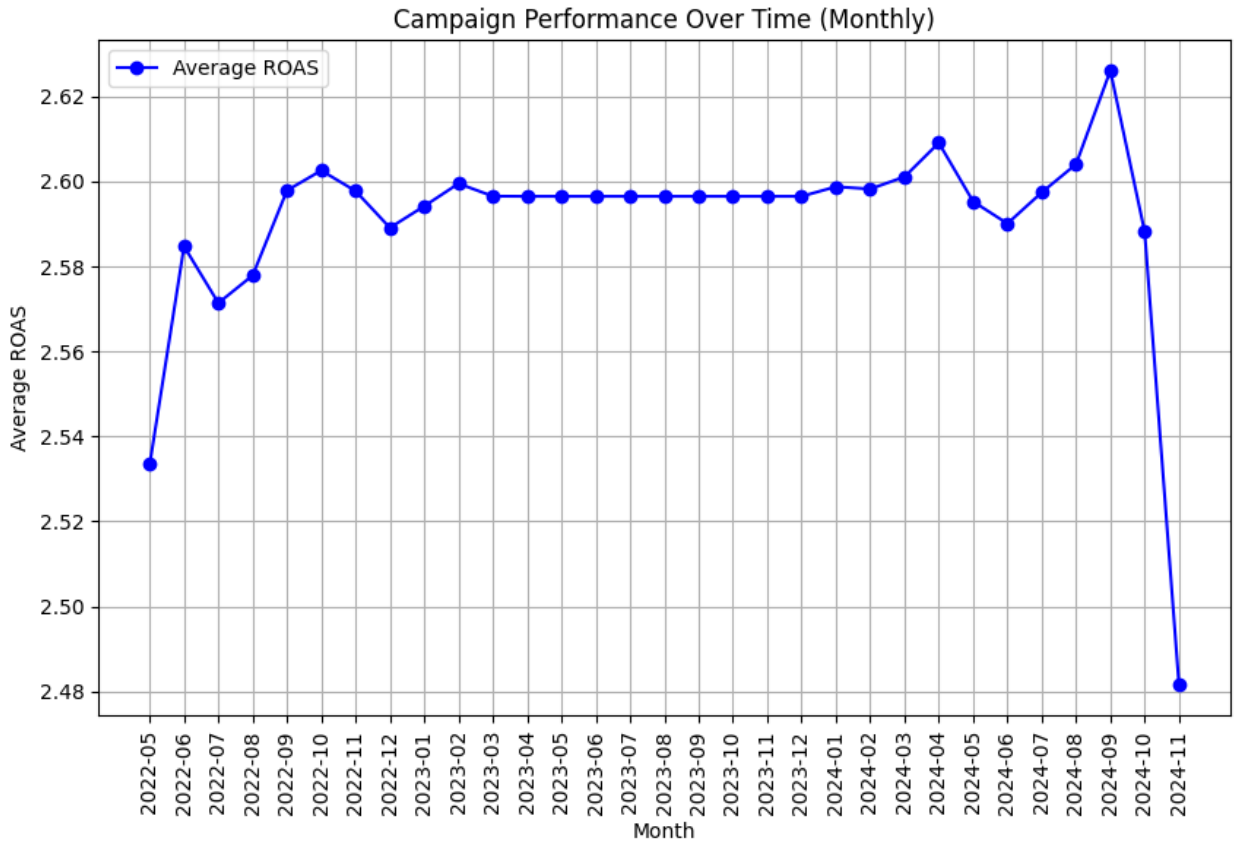
```
# Display monthly performance
print(monthly_performance)
```

```
      Month   total_spend  total_revenue  average_roas
0   2022-05  7.712339e+06   1.893981e+07      2.533406
1   2022-06  3.193872e+07   8.128609e+07      2.584662
2   2022-07  5.818045e+07   1.487166e+08      2.571333
3   2022-08  8.348672e+07   2.148975e+08      2.577832
4   2022-09  1.050509e+08   2.728940e+08      2.597805
5   2022-10  1.348181e+08   3.514376e+08      2.602499
6   2022-11  1.543156e+08   4.014395e+08      2.597671
7   2022-12  1.860768e+08   4.820888e+08      2.589033
8   2023-01  2.117642e+08   5.502171e+08      2.594114
9   2023-02  2.135906e+08   5.566541e+08      2.599423
10  2023-03  2.526142e+08   6.572150e+08      2.596456
11  2023-04  2.447159e+08   6.366522e+08      2.596418
12  2023-05  2.528731e+08   6.578739e+08      2.596418
13  2023-06  2.447159e+08   6.366522e+08      2.596418
14  2023-07  2.528731e+08   6.578739e+08      2.596418
15  2023-08  2.528731e+08   6.578739e+08      2.596418
16  2023-09  2.447159e+08   6.366522e+08      2.596418
17  2023-10  2.528731e+08   6.578739e+08      2.596418
18  2023-11  2.447159e+08   6.366522e+08      2.596418
19  2023-12  2.528731e+08   6.578739e+08      2.596418
20  2024-01  2.481847e+08   6.466558e+08      2.598685
21  2024-02  2.115052e+08   5.519540e+08      2.598154
22  2024-03  2.014628e+08   5.262317e+08      2.601013
23  2024-04  1.703864e+08   4.459984e+08      2.609004
24  2024-05  1.511590e+08   3.933354e+08      2.595207
25  2024-06  1.212572e+08   3.146664e+08      2.590021
26  2024-07  9.958845e+07   2.589825e+08      2.597345
27  2024-08  7.384422e+07   1.933684e+08      2.603977
28  2024-09  4.593344e+07   1.209331e+08      2.625985
29  2024-10  2.195824e+07   5.665973e+07      2.588113
30  2024-11  2.674296e+06   6.566547e+06      2.481600
```

```
expanded_campaign_data['Week'] =
expanded_campaign_data['Date'].dt.to_period('W')

# Aggregate by week
weekly_performance = expanded_campaign_data.groupby('Week').agg(
    total_spend=('Total_Spend', 'sum'),
    total_revenue=('Revenue_Generated', 'sum'),
    average_roas=('ROAS', 'mean')
).reset_index()

# Display weekly performance
print(weekly_performance)
```

```
                      Week  total_spend  total_revenue  average_roas
0    2022-05-09/2022-05-15   509877.904   1.149440e+06      2.344461
1    2022-05-16/2022-05-22  2066453.136   5.110970e+06      2.539954
2    2022-05-23/2022-05-29  3797949.760   9.320064e+06      2.534341
3    2022-05-30/2022-06-05  5112127.872   1.296184e+07      2.606186
4    2022-06-06/2022-06-12  6352492.880   1.629576e+07      2.610781
..                     ...          ...            ...           ...
128  2024-10-21/2024-10-27  3621914.624   9.350427e+06      2.583368
129  2024-10-28/2024-11-03  2028459.904   5.164541e+06      2.550211
130  2024-11-04/2024-11-10  1297697.024   3.219462e+06      2.505174
131  2024-11-11/2024-11-17   553090.064   1.285523e+06      2.375944
132  2024-11-18/2024-11-24    78806.160   1.949870e+05      2.475624

[133 rows x 4 columns]
```

```
#Plotting monthly performance
plt.figure(figsize=(10, 6))
plt.plot(monthly_performance['Month'].astype(str),
monthly_performance['average_roas'], marker='o', color='b',
label='Average ROAS')

plt.title('Campaign Performance Over Time (Monthly)')
plt.xlabel('Month')
plt.ylabel('Average ROAS')
plt.xticks(rotation=90)
plt.grid(True)
plt.legend()

plt.show()
```


Campaign Performance Over Time (Monthly)
Average ROAS
Average ROAS
2.62
2.60
2.58
2.56
2.54
2.52
2.50
2.48
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
2022-11
2022-12
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
2023-08
2023-09
2023-10
2023-11
2023-12
2024-01
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
2024-11
Month