JULY 15, 2024

Interview Exercise Analysis on Company XYZ

Brian Witarsa

Roadmap

Import Data

Data Cleaning

Establishing KPIs

Data Analysis

Interpret Results

Step 1: Import Libraries to Load Data

- Use **pandas** to insert data

View of data when first loaded in

	Company Name (ID)	Campaign Name	Week Start Date	Advertising Spend	People Reached	Customers Acquired by Campaign	Campaign Conversions	Ads Delivered	Campaign Revenue
0	Brand XYZ	Campaign A	7/26/2020	$57	7,786	2	2	10,007	$203
1	Brand XYZ	Campaign A	8/2/2020	$1,140	124,987	28	29	199,975	$7,976
2	Brand XYZ	Campaign A	8/2/2020	$924	95,252	7	7	162,167	$1,130
3	Brand XYZ	Campaign A	8/9/2020	$853	87,318	56	58	149,655	$9,401
4	Brand XYZ	Campaign A	8/9/2020	$915	97,319	16	16	160,593	$2,276

Step 2: Data Cleaning (1/3)

Data types loaded in as strings…

```
Company Name (ID)                  object
Campaign Name                      object
Week Start Date                    object
Advertising Spend                  object
People Reached                     object
Customers Acquired by Campaign     object
Campaign Conversions               object
Ads Delivered                      object
Campaign Revenue                   object
dtype: object
```

…Data types converted to floats and date

```
Company Name (ID)                          object
Campaign Name                              object
Week Start Date                    datetime64[ns]
Advertising Spend                         float64
People Reached                            float64
Customers Acquired by Campaign            float64
Campaign Conversions                      float64
Ads Delivered                             float64
Campaign Revenue                          float64
dtype: object
```

- Commas and $ were removed from strings and converted into numeric floats

Step 2: Data Cleaning (2/3)

View of data after data type transformation

	Company Name (ID)	Campaign Name	Week Start Date	Advertising Spend	People Reached	Customers Acquired by Campaign	Campaign Conversions	Ads Delivered	Campaign Revenue
0	Brand XYZ	Campaign A	2020-07-26	57.0	7786.0	2.0	2.0	10007.0	203.0
1	Brand XYZ	Campaign A	2020-08-02	1140.0	124987.0	28.0	29.0	199975.0	7976.0
2	Brand XYZ	Campaign A	2020-08-02	924.0	95252.0	7.0	7.0	162167.0	1130.0
3	Brand XYZ	Campaign A	2020-08-09	853.0	87318.0	56.0	58.0	149655.0	9401.0
4	Brand XYZ	Campaign A	2020-08-09	915.0	97319.0	16.0	16.0	160593.0	2276.0

Step 2: Data Cleaning (3/3)

Searched for the following:

- Invalid Values
- Missing Values
- Outliers

Identified outliers as values greater than 10 standard deviations above the mean

Found outlier in Campaign Revenue

```
Outliers:
   Company Name (ID) Campaign Name Week Start Date  Advertising Spend  \
45         Brand XYZ    Campaign D      2020-04-05             1365.0

    People Reached  Customers Acquired by Campaign  Campaign Conversions  \
45         34890.0                            25.0                  26.0

    Ads Delivered  Campaign Revenue
45       227517.0      1.030402e+12
```

Step 3: KPI and Additional Metrics (1/2)

1. **Cost per Acquisition** → $\frac{\text{Advertising Spend}}{\text{Customers Acquired by Campaign}}$

2. **Profit** → Campaign Revenue x 0.05

3. **Conversion Rate** → $\frac{\text{Campaign Conversions}}{\text{People Reached}}$

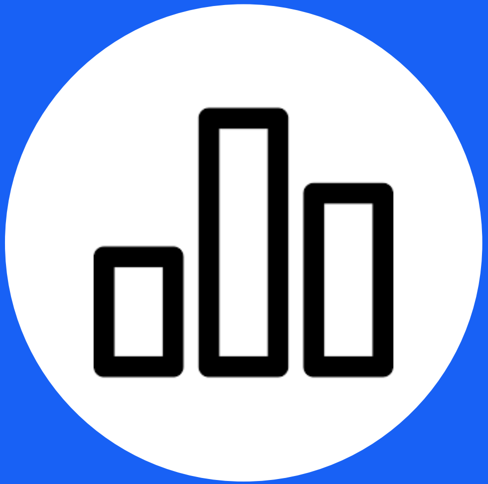
4. **Return on Advertising Spend** → $\frac{\text{Campaign Revenue}}{\text{Advertising Spend}}$

Step 3: KPI and Additional Metrics (2/2)

View of data after additional metrics included

	Company Name (ID)	Campaign Name	Week Start Date	Advertising Spend	People Reached	Customers Acquired by Campaign	Campaign Conversions	Ads Delivered	Campaign Revenue	Profit	Conversion Rate	Cost per Acquisition	ROAS
0	Brand XYZ	Campaign A	2020-07-26	57.0	7786.0	2.0	2.0	10007.0	203.0	10.15	0.000257	28.500000	3.561404
1	Brand XYZ	Campaign A	2020-08-02	1140.0	124987.0	28.0	29.0	199975.0	7976.0	398.80	0.000232	40.714286	6.996491
2	Brand XYZ	Campaign A	2020-08-02	924.0	95252.0	7.0	7.0	162167.0	1130.0	56.50	0.000073	132.000000	1.222944
3	Brand XYZ	Campaign A	2020-08-09	853.0	87318.0	56.0	58.0	149655.0	9401.0	470.05	0.000664	15.232143	11.021102
4	Brand XYZ	Campaign A	2020-08-09	915.0	97319.0	16.0	16.0	160593.0	2276.0	113.80	0.000164	57.187500	2.487432

Step 4: Data Analysis



Summary Statistics

ANOVA Test

Tukey's HSD Test

Regression Analysis

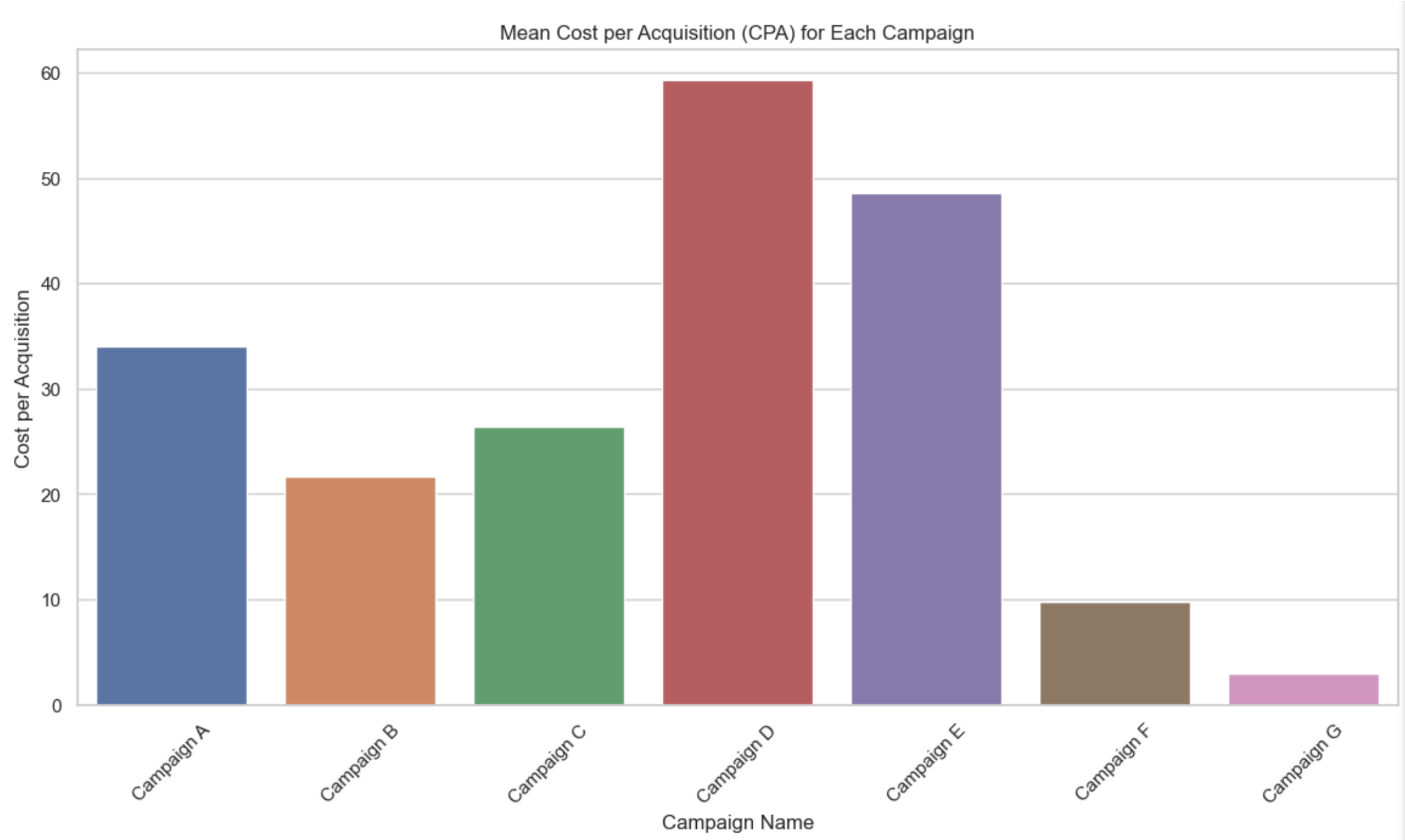
Step 4: Data Analysis Summary Statistics

General comparison of means

- Provides preliminary overview of campaign performance across metrics
- Calculate means of metrics across campaigns, visualized using bar plots

Step 5: Interpreting Results Summary Statistics (1/4)

Mean CPA across campaigns

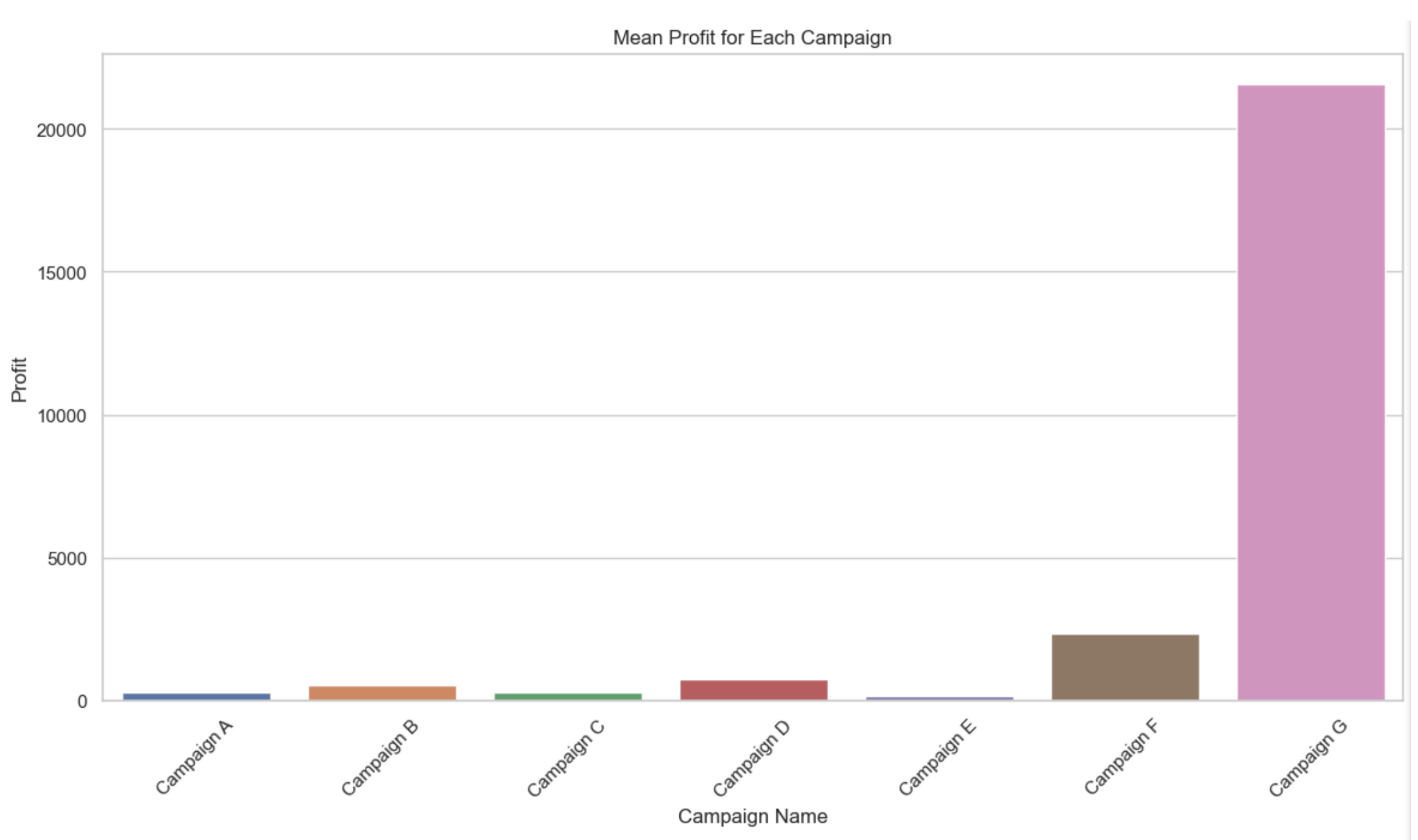


Key Takeaways

- Campaign G significantly outperforms all other campaigns
- Campaign F is a definitive second
- Campaign D and E have the highest CPA

Step 5: Interpreting Results Summary Statistics (2/4)

Mean Profit across campaigns

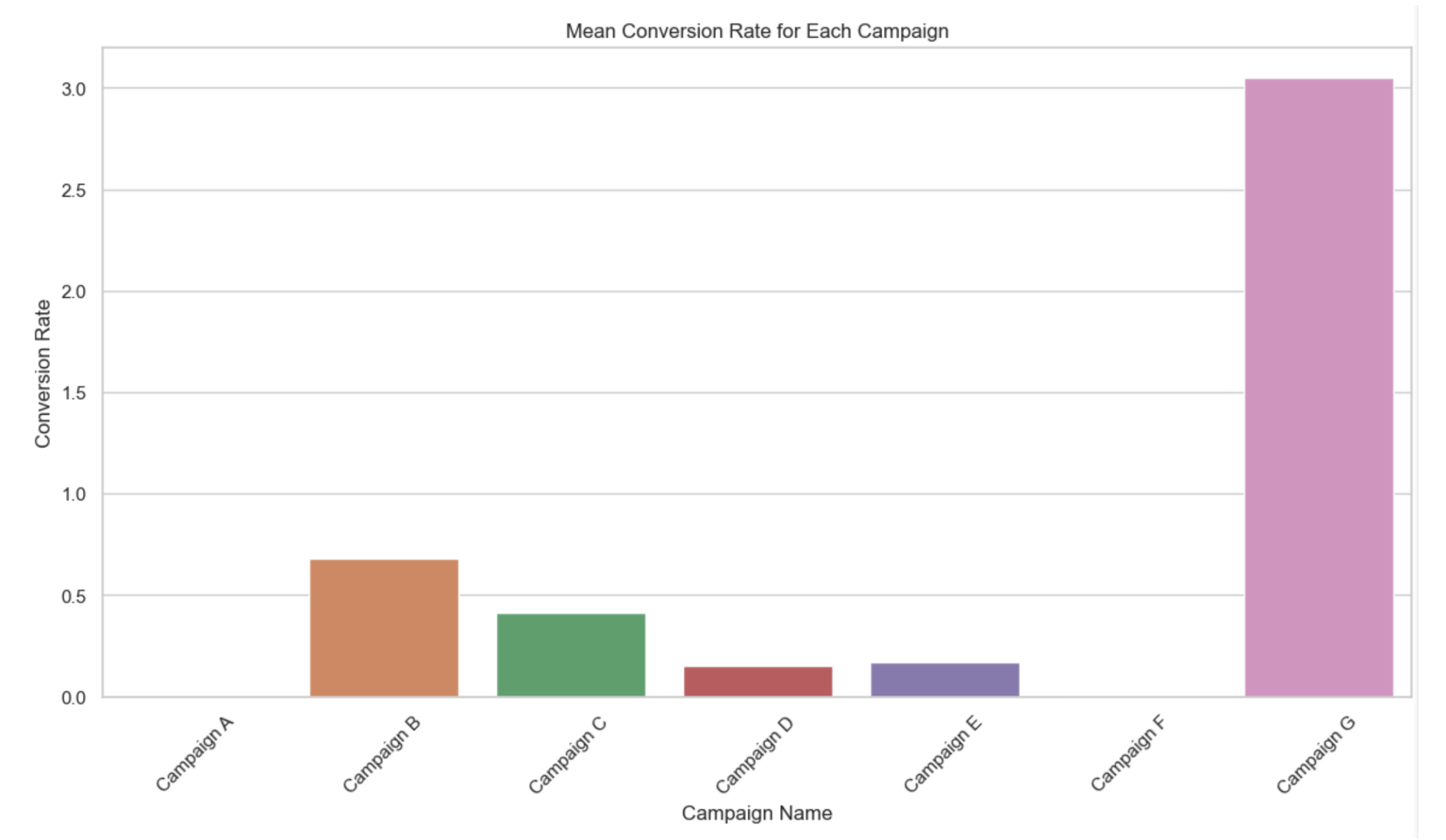


Key Takeaways

- Campaign G significantly outperforms all other campaigns
- Campaign F is a definitive second

Step 5: Interpreting Results Summary Statistics (3/4)

Mean Conversion Rate across Campaigns

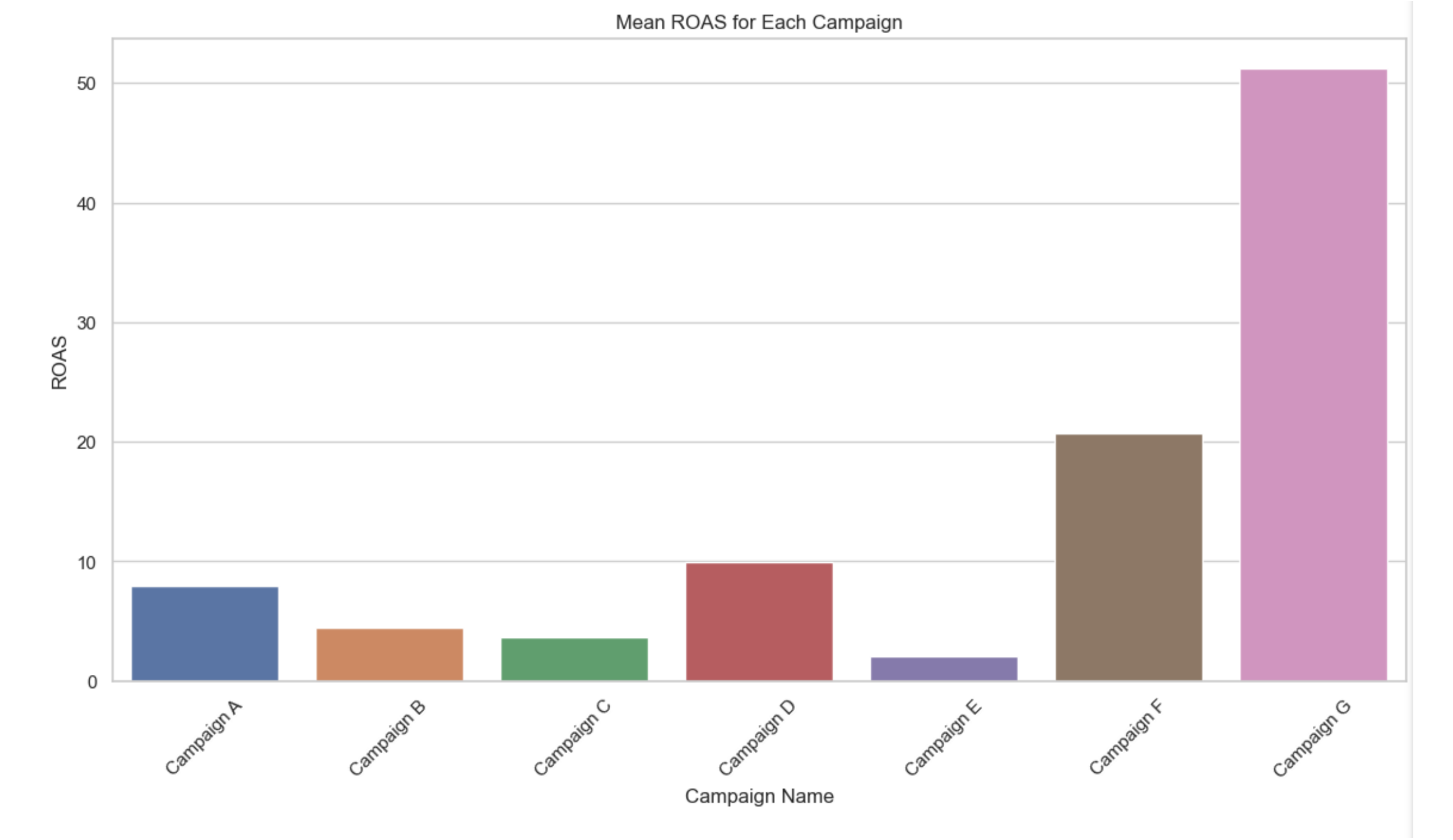


Key Takeaways

- Campaign G has the definitive highest conversion rate. Campaign B second
- Not many takeaways can be extracted from observing Conversion Rate

Step 5: Interpreting Results Summary Statistics (4/4)

Mean ROAS across campaigns



Key Takeaways

- Campaign G outperforms other campaigns in terms of ROAS
- Campaign F definitive second
- Seemingly negligible differences amongst others

Epsilon®

Step 4: Data Analysis ANOVA Test Overview

Determines if there are statistically significant differences in performance across campaigns.

- F-statistic: Indicator of variability. Ratio of variance between groups to the variance within groups
- p-value: Probability of observing the data if null hypothesis is true. Low p-value (< 0.05) indicates strong evidence to reject the null hypothesis, which means a significant difference between group means
- Null Hypothesis: Assumption all group means are equal (no significant difference between group means)

Step 5: Interpret Results ANOVA Test

CPA

F-statistic: 3.864

P-value: 0.00161

- p-value > 0.05 means that there is a statistically significant difference between campaigns. Some campaigns are more cost-effective than others

Profit

F-statistic: 11.645

P-value: 6.058e-10

- F-statistic is high and p-value is much less than 0.05, which indicates variation in profits across campaigns.

Conversion Rate

F-statistic: 0.248

P-value: 0.959

- F-statistic is very low which indicates low variation in rates across campaigns. P-value is much greater than 0.05, which indicates there are no significant differences for conversion rates across campaigns

ROAS

F-statistic: 16.521

P-value: 2.701e-13

- F-statistic is very high and P-value is much less than 0.05, indicating significant variation across campaigns

Post-Hoc Analysis

Tukey's HSD Test

- The ANOVA test does **not** tell us how the campaigns perform compared to one another
- Pairwise comparisons to identify which groups are significantly different from one another
- P-values < 0.05 and Confidence Intervals that do **not** include 0 indicate significant difference
- Negative Mean Difference value indicates first group's mean > second group's mean

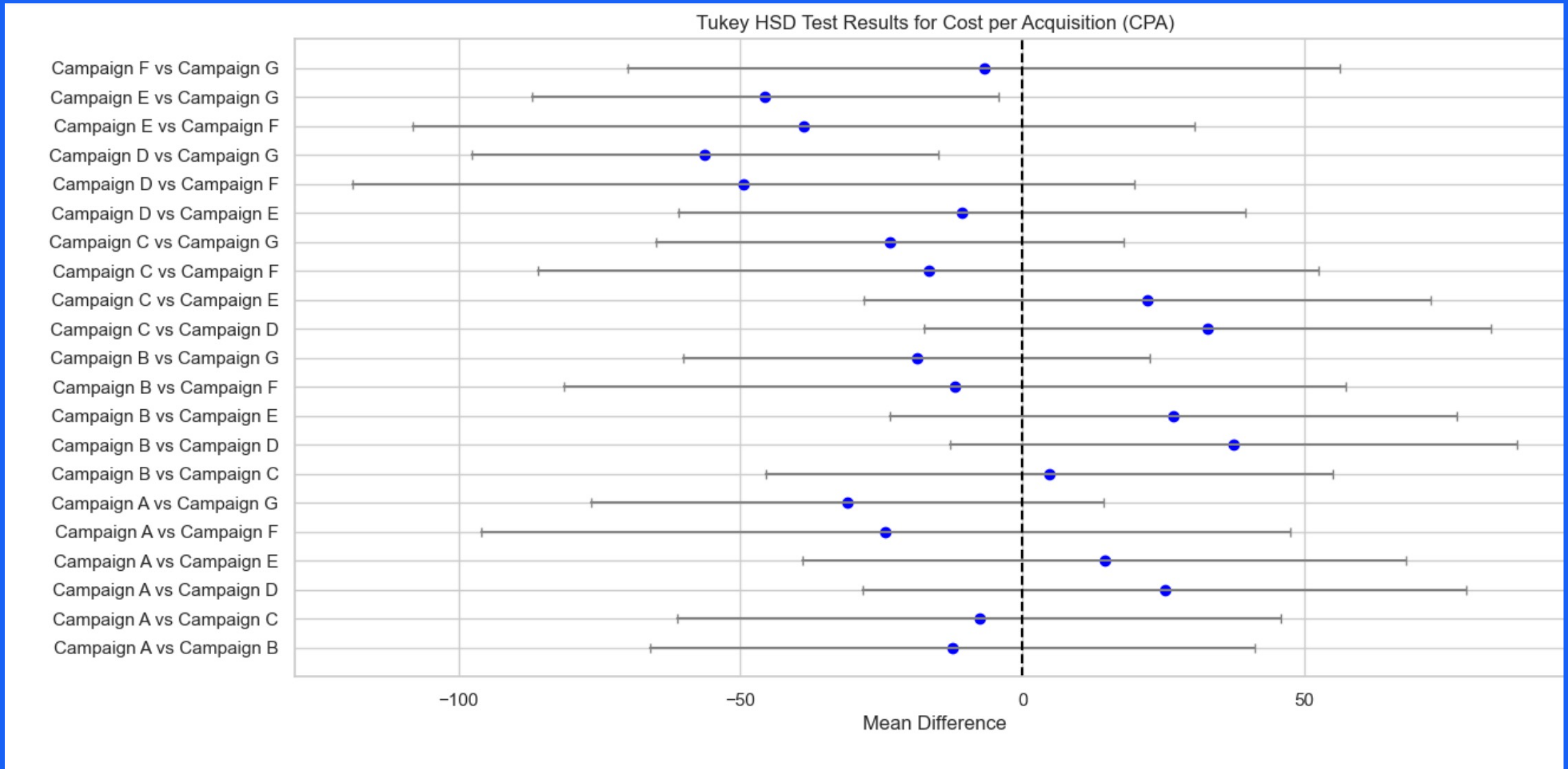
Tukey HSD results for CPA

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
Campaign A	Campaign B	-12.3289	0.9928	-65.9193	41.2615	False
Campaign A	Campaign C	-7.6407	0.9995	-61.2311	45.9497	False
Campaign A	Campaign D	25.2206	0.7926	-28.3698	78.811	False
Campaign A	Campaign E	14.5283	0.9829	-39.0621	68.1187	False
Campaign A	Campaign F	-24.2911	0.9488	-96.0302	47.448	False
Campaign A	Campaign G	-31.0741	0.3856	-76.4822	14.3341	False
Campaign B	Campaign C	4.6882	1.0	-45.5841	54.9604	False
Campaign B	Campaign D	37.5495	0.2807	-12.7227	87.8218	False
Campaign B	Campaign E	26.8572	0.6786	-23.415	77.1294	False
Campaign B	Campaign F	-11.9622	0.9985	-81.2577	57.3333	False
Campaign B	Campaign G	-18.7452	0.8214	-60.1851	22.6947	False
Campaign C	Campaign D	32.8614	0.4427	-17.4109	83.1336	False
Campaign C	Campaign E	22.169	0.8383	-28.1032	72.4413	False
Campaign C	Campaign F	-16.6504	0.9909	-85.9459	52.6451	False
Campaign C	Campaign G	-23.4334	0.6174	-64.8733	18.0066	False
Campaign D	Campaign E	-10.6923	0.9953	-60.9646	39.5799	False
Campaign D	Campaign F	-49.5117	0.3328	-118.8072	19.7838	False
Campaign D	Campaign G	-56.2947	0.0016	-97.7346	-14.8548	True
Campaign E	Campaign F	-38.8194	0.6279	-108.1149	30.4761	False
Campaign E	Campaign G	-45.6024	0.0212	-87.0423	-4.1625	True
Campaign F	Campaign G	-6.783	0.9999	-69.964	56.398	False

Step 5: Interpreting Results Tukey HSD Test (1/2)

Tukey HSD Test results for CPA visualized



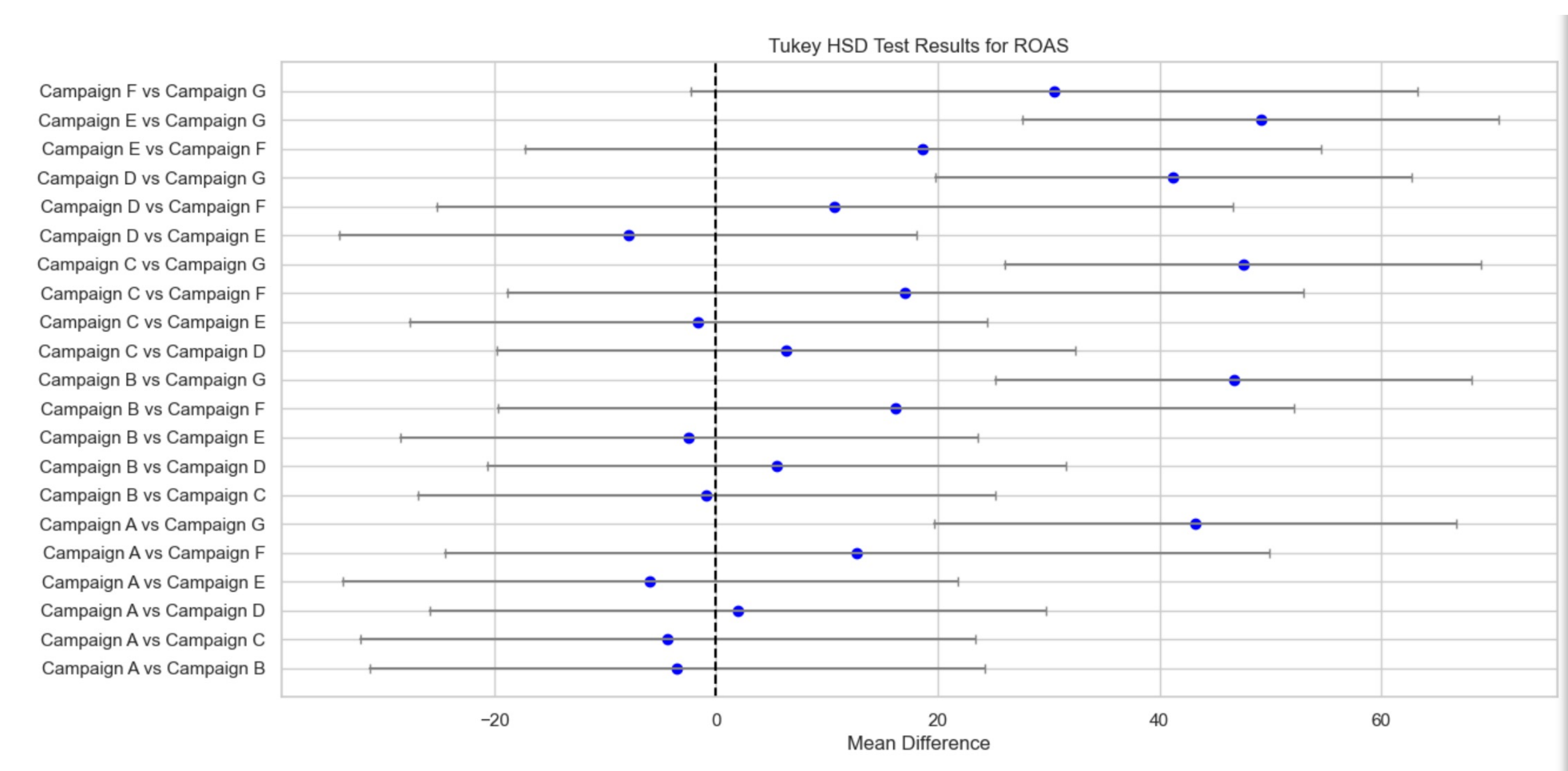
Key Takeaways

- Campaign E vs Campaign G
  - 0 not in Confidence Interval
  - Negative mean difference
- Campaign D vs Campaign G
  - 0 not in Confidence Interval
  - Negative mean difference

Campaign G significantly outperforms Campaigns D and E in CPA

Step 5: Interpreting Results Tukey HSD Test (2/2)

Tukey HSD Test results for ROAS visualized



Key Takeaways

- All Campaigns **except** Campaign F when compared to Campaign G:
  - 0 not in Confidence Interval
  - Positive mean difference

Campaign G significantly outperforms all other campaigns in ROAS apart from Campaign F

Epsilon

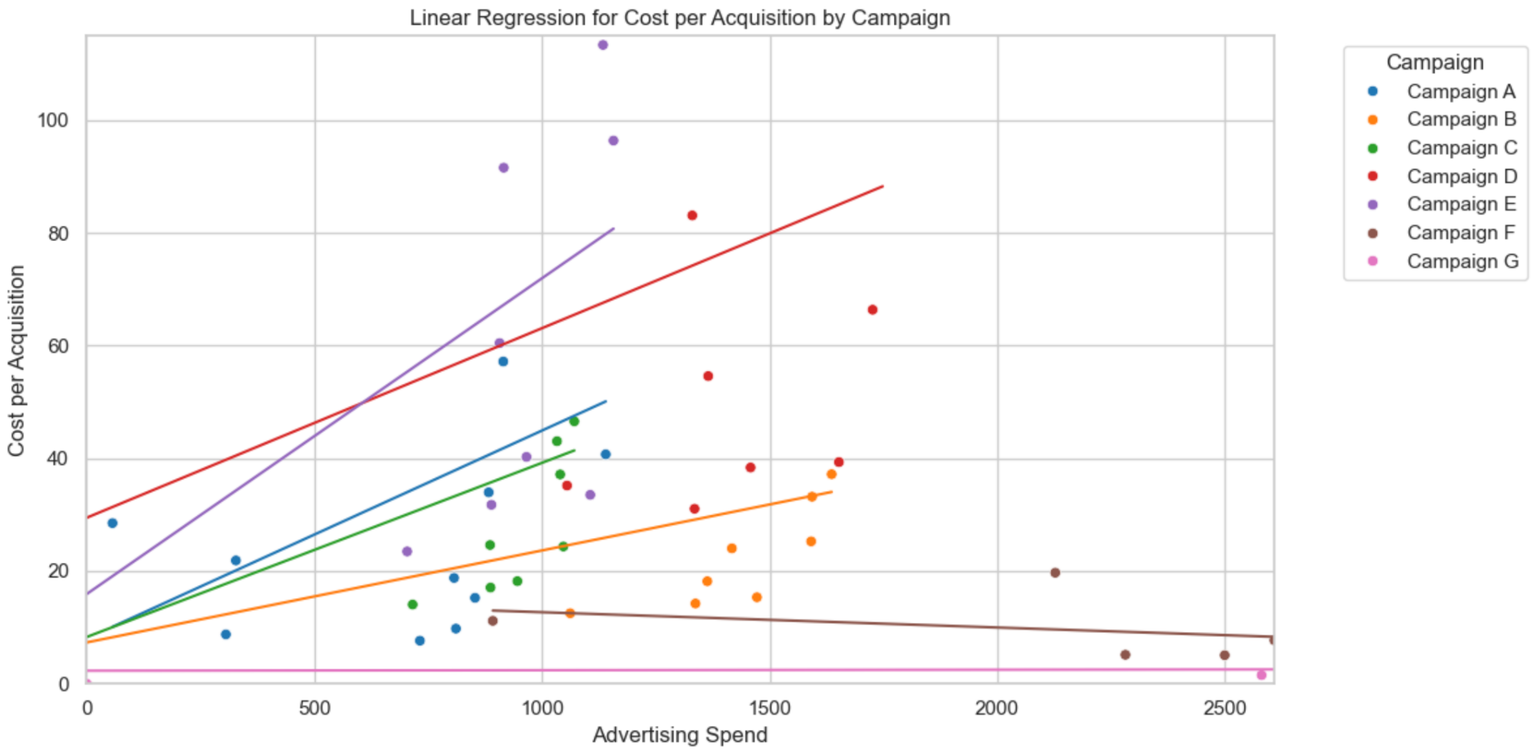
Step 4: Data Analysis Linear Regression Analysis

Analyze relationship between an independent variable and a dependent variable

- To predict performance of various campaigns across several metrics with increased advertising spend

Step 5: Interpreting Results Linear Regression Analysis (1/4)

CPA with increased Advertising Spend

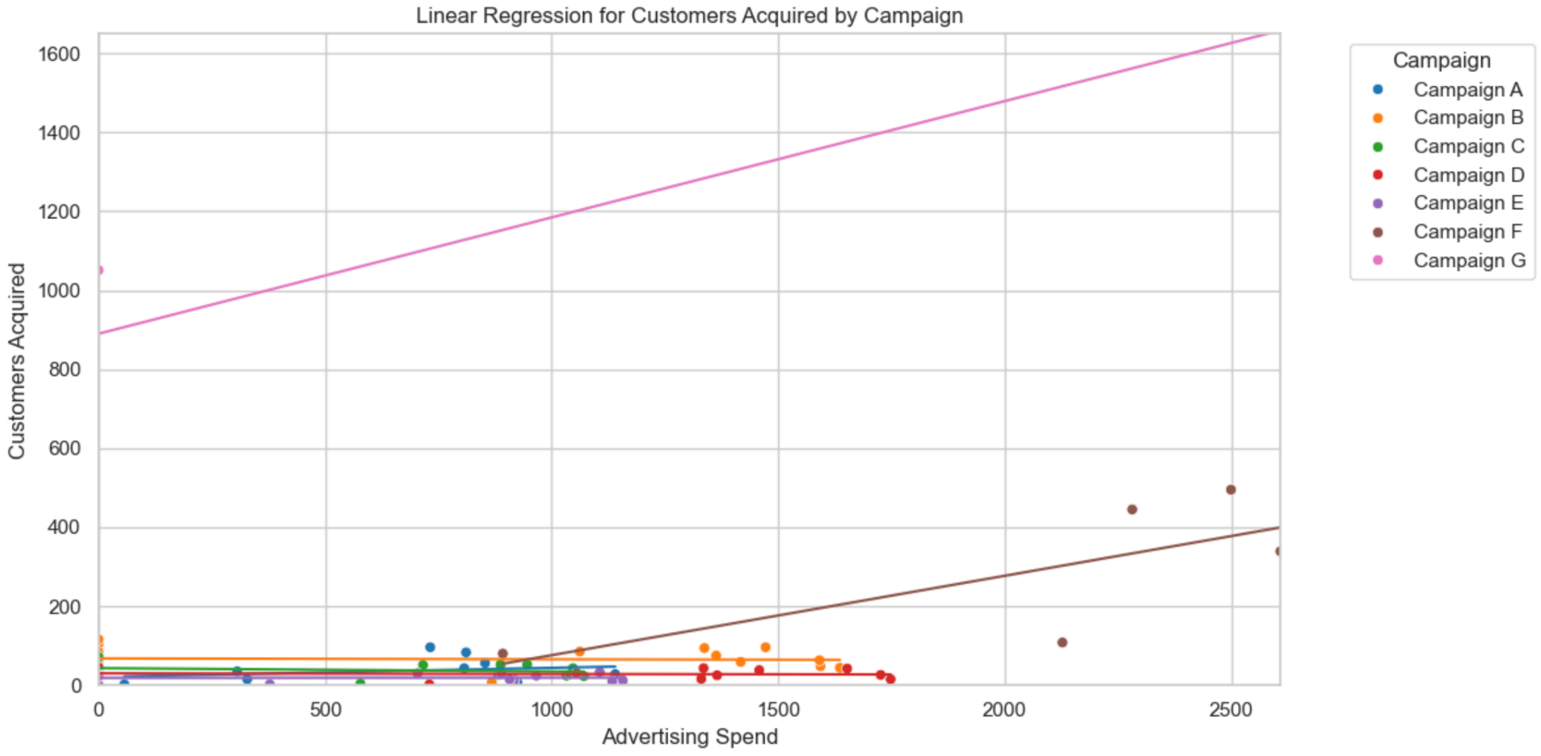


Key Takeaways

- Campaign G with least CPA
- Campaign F definitive second
- Campaigns D and E continue to show poor performance
- Campaigns A, B, C displaying similar performance

Step 5: Interpreting Results
Linear Regression Analysis (2/4)

Customers Acquired with increased Advertising Spend

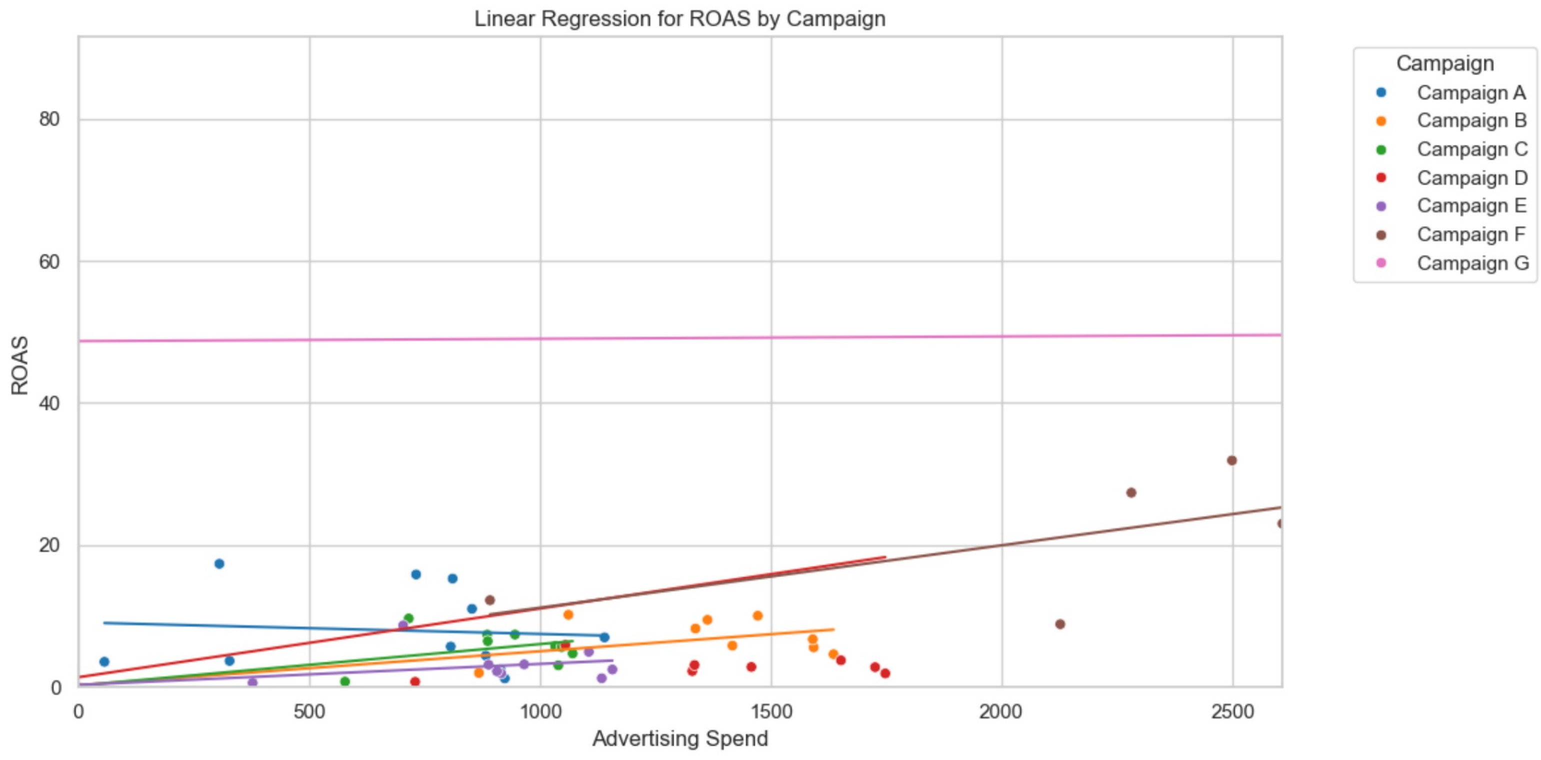


Key Takeaways

- Campaign G attracts most customers by far
- Campaign F a distant second
- Negligible difference amongst remaining campaigns

Step 5: Interpreting Results
Linear Regression Analysis (3/4)

ROAS with increased Advertising Spend

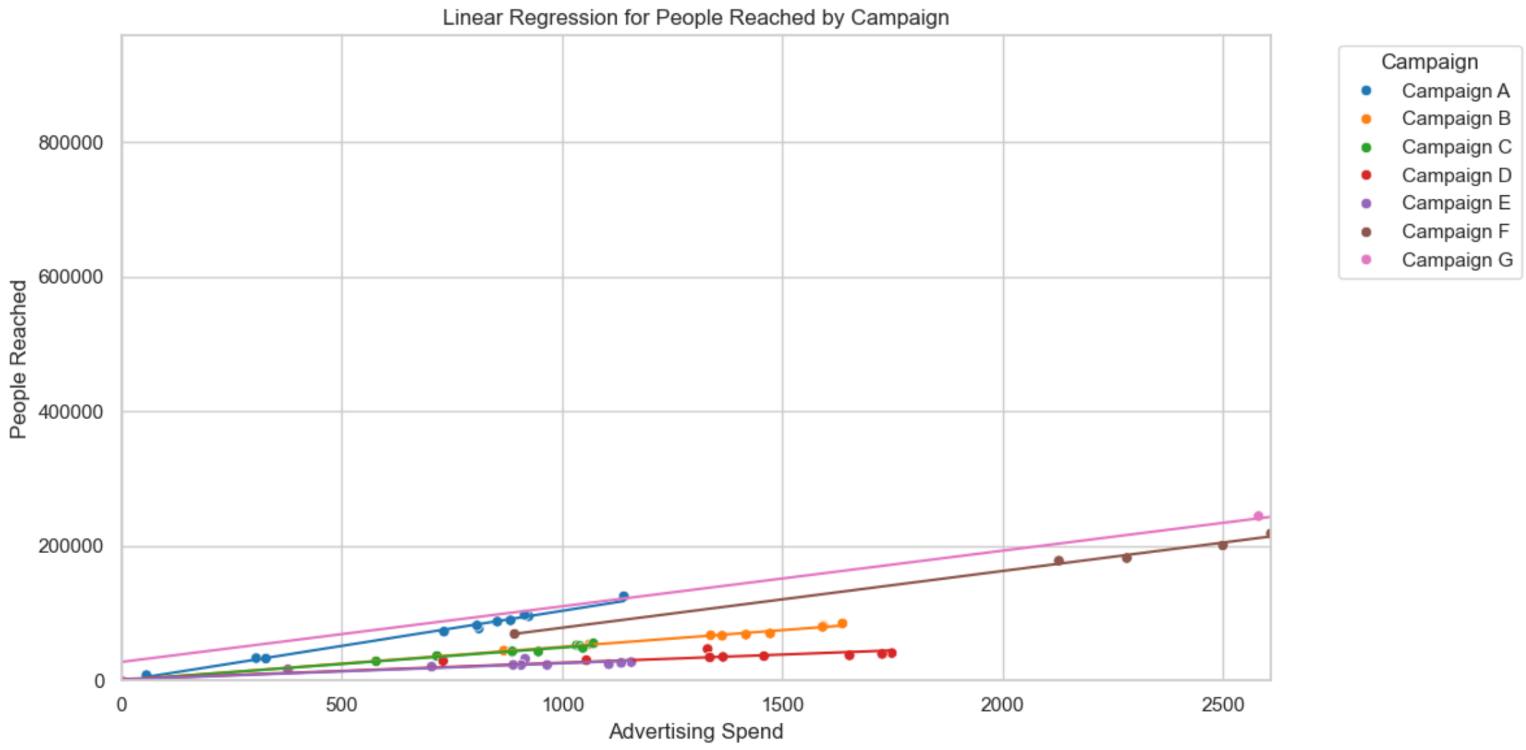


Key Takeaways

- Campaign G with greatest return
- Campaign F and D with similar projections
- Remaining campaigns with negligible differences

Step 5: Interpreting Results Linear Regression Analysis (4/4)

People Reached with increased Advertising Spend



Key Takeaways

- Campaigns G, F, and A with highest projections
- Remaining campaigns have negligible differences

Conclusion

Assessment of Campaigns and Recommendations

Conclusion | Top Performers

1

Campaign G

- Campaign G the top performer in all tests
- Led in Summary Statistics, excelling in CPA, Profit, Conversion Rate, and ROAS
- Outperformed campaigns D and E significantly in Tukey's HSD Test
- Exhibited superior projections in Linear Regression for CPA, Customers Acquired, ROAS, and People Reached with increased Advertising Spend

2

Campaign F

- Consistently ranks second in most tests conducted
- Second best in Summary Statistics for CPA, Profit, and ROAS
- Second best in Tukey's HSD Test. **No** significant performance difference from Campaign G in CPA and ROAS
- Second Best performer in Linear Regression Analysis, strong projections across CPA and Customers Acquired with increased Advertising Spend

Conclusion | Mid Tier Performers

These 3 campaigns performed similarly, with negligible differences on most tests.

3

Campaign A

4

Campaign C

5

Campaign B

Conclusion | Worst Performers

6 Campaign D

7 Campaign E

These two campaigns were consistently the worst performers across all tests

- Ranked low on all metrics in the Summary Statistic Analysis
  - Notably highest CPA across all campaigns
- Both campaigns significantly underperformed compared to Campaign G in the Tukey's HSD Test for CPA and ROAS
- Consistent poor performers across most metrics in Linear Regression Analysis
- Slight edge given to Campaign D for its performance on the ROAS metric

Appendix

Python Code (1/7)

Importing libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from scipy import stats
from statsmodels.stats.multicomp import pairwise_tukeyhsd
from itertools import combinations
```

Loading in data

```
#loading data in
df = pd.read_csv('Sample_Data_Company_XYZ.csv')
```

Data cleaning and transformation

```
#verifying data integrity
missing_val = df.isnull().sum()
print(missing_val)

duplicates = df.duplicated().sum()
print(f'Duplicates: {duplicates}')

df['Week Start Date'] = pd.to_datetime(df['Week Start Date'])
df['Advertising Spend'] = df['Advertising Spend'].str.replace(r'[$,]', '', regex=True).astype(float)
df['Campaign Revenue'] = df['Campaign Revenue'].str.replace(r'[$,]', '', regex=True).astype(float)
df['Customers Acquired by Campaign'] = df['Customers Acquired by Campaign'].str.replace(',', '', regex=True).astype(float)
df['People Reached'] = df['People Reached'].str.replace(',', '', regex=True).astype(float)
df['Campaign Conversions'] = df['Campaign Conversions'].str.replace(',', '', regex=True).astype(float)
df['Ads Delivered'] = df['Ads Delivered'].str.replace(',', '', regex=True).astype(float)
```

Python Code (2/7)

Find and remove outliers

```
mean_revenue = df['Campaign Revenue'].mean()
std_revenue = df['Campaign Revenue'].std()

#threshold for outliers
threshold = mean_revenue + 10 * std_revenue

#find outliers
outliers = df[df['Campaign Revenue'] > threshold]
print("Outliers:")
print(outliers)

#remove outliers
non_outliers_mean = df[df['Campaign Revenue'] <= threshold]['Campaign Revenue'].mean()
df.loc[df['Campaign Revenue'] > threshold, 'Campaign Revenue'] = non_outliers_mean
```

Establish KPIs and replace invalid values with 0

```
#establish KPIs
df['Profit'] = df['Campaign Revenue']*0.05
df['Conversion Rate'] = df['Campaign Conversions'] / df['People Reached']
df['Cost per Acquisition'] = df['Advertising Spend']/df['Customers Acquired by Campaign']
df['ROAS'] = df['Campaign Revenue'] / df['Advertising Spend']

# Replace invalid values with 0
df.fillna(0, inplace=True)
df.replace([np.inf, -np.inf], 0, inplace=True)
```

Python Code (3/7)

Calculate and plot means for Summary Statistic Analysis

```python
# calculate mean by campaign
mean_profit = df.groupby('Campaign Name')['Profit'].mean().reset_index()
mean_cpa = df.groupby('Campaign Name')['Cost per Acquisition'].mean().reset_index()
mean_conversion_rate = df.groupby('Campaign Name')['Conversion Rate'].mean().reset_index()
mean_roas = df.groupby('Campaign Name')['ROAS'].mean().reset_index()

print(mean_conversion_rate)
# mean Profit for Each Campaign
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign Name', y='Profit', hue='Campaign Name', data=mean_profit)
plt.title('Mean Profit for Each Campaign')
plt.xticks(rotation=45)
plt.show()

# mean CPA for Each Campaign
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign Name', y='Cost per Acquisition', hue='Campaign Name', data=mean_cpa)
plt.title('Mean Cost per Acquisition (CPA) for Each Campaign')
plt.xticks(rotation=45)
plt.show()

# mean Conversion Rate for each campaign
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign Name', y='Conversion Rate', hue='Campaign Name', data=mean_conversion_rate)
plt.title('Mean Conversion Rate for Each Campaign')
plt.xticks(rotation=45)
plt.show()

# mean ROAS for each campaign
plt.figure(figsize=(14, 7))
sns.barplot(x='Campaign Name', y='ROAS', hue='Campaign Name', data=mean_roas)
plt.title('Mean ROAS for Each Campaign')
plt.xticks(rotation=45)
plt.show()
```

Python Code (4/7)

ANOVA Test Calculation

```
campaigns = df['Campaign Name'].unique()

#prepare data for ANOVA
cpa_data = [df[df['Campaign Name'] == campaign]['Cost per Acquisition'].dropna() for campaign in campaigns]
profit_data = [df[df['Campaign Name'] == campaign]['Profit'].dropna() for campaign in campaigns]
conversion_rate_data = [df[df['Campaign Name'] == campaign]['Conversion Rate'].dropna() for campaign in campaigns]
roas_data = [df[df['Campaign Name'] == campaign]['ROAS'].dropna() for campaign in campaigns]

#ANOVA test for CPA
anova_cpa = stats.f_oneway(*cpa_data)
print('ANOVA result for Cost per Acquisition by Campaign:', anova_cpa)

#Profit
anova_profit = stats.f_oneway(*profit_data)
print('ANOVA result for Profit by Campaign:', anova_profit)

#Conversion Rate
anova_conversion_rate = stats.f_oneway(*conversion_rate_data)
print('ANOVA result for Conversion Rate by Campaign:', anova_conversion_rate)

#ROAS
anova_roas = stats.f_oneway(*roas_data)
print('ANOVA result for ROAS by Campaign:', anova_roas)
```

Python Code (5/7)

Tukey's HSD Test

```
#Tukey HSD test for CPA
tukey_cpa = pairwise_tukeyhsd(endog=df['Cost per Acquisition'], groups=df['Campaign Name'], alpha=0.05)
print(tukey_cpa)

#Profit
tukey_profit = pairwise_tukeyhsd(endog=df['Profit'], groups=df['Campaign Name'], alpha=0.05)
print(tukey_profit)

#Conversion Rate
tukey_conversion_rate = pairwise_tukeyhsd(endog=df['Conversion Rate'], groups=df['Campaign Name'], alpha=0.05)
print(tukey_conversion_rate)

#ROAS
tukey_roas = pairwise_tukeyhsd(endog=df['ROAS'], groups=df['Campaign Name'], alpha=0.05)
print(tukey_roas)
```

Plot results

```
def plot_tukey_hsd(tukey_result, title):
    tukey_df = pd.DataFrame(data=tukey_result._results_table.data[1:], columns=tukey_result._results_table.data[0])

    plt.figure(figsize=(14, 7))
    for i, row in tukey_df.iterrows():
        plt.plot([row['lower'], row['upper']], [i, i], color='grey', marker='|')
        plt.scatter(row['meandiff'], i, color='red' if row['reject'] == 'True' else 'blue')

    plt.yticks(range(len(tukey_df)), tukey_df['group1'] + ' vs ' + tukey_df['group2'])
    plt.axvline(x=0, color='black', linestyle='--')
    plt.xlabel('Mean Difference')
    plt.title(title)
    plt.grid(True)
    plt.show()

#plot Tukey HSD results
plot_tukey_hsd(tukey_cpa, 'Tukey HSD Test Results for Cost per Acquisition (CPA)')
plot_tukey_hsd(tukey_profit, 'Tukey HSD Test Results for Profit')
plot_tukey_hsd(tukey_conversion_rate, 'Tukey HSD Test Results for Conversion Rate')
plot_tukey_hsd(tukey_roas, 'Tukey HSD Test Results for ROAS')
```

Python Code (6/7)

Linear Regression

```python
#fit and predict using linear regression for each campaign
def fit_predict_linear_regression(df, x_col, y_col):
    predictions = pd.DataFrame()

    for campaign in df['Campaign Name'].unique():
        df_campaign = df[df['Campaign Name'] == campaign].copy()
        X = df_campaign[[x_col]]
        y = df_campaign[y_col]

        model = LinearRegression()
        model.fit(X, y)

        df_campaign['Prediction'] = model.predict(X)
        predictions = pd.concat([predictions, df_campaign], axis=0)

    return predictions


predictions_customers = fit_predict_linear_regression(df, 'Advertising Spend', 'Customers Acquired by Campaign')
predictions_cpa = fit_predict_linear_regression(df, 'Advertising Spend', 'Cost per Acquisition')
predictions_roas = fit_predict_linear_regression(df, 'Advertising Spend', 'ROAS')
predictions_people_reached = fit_predict_linear_regression(df, 'Advertising Spend', 'People Reached')

predictions_customers.rename(columns={'Prediction': 'Predicted Customers Acquired'}, inplace=True)
predictions_cpa.rename(columns={'Prediction': 'Predicted Cost per Acquisition'}, inplace=True)
predictions_roas.rename(columns={'Prediction': 'Predicted ROAS'}, inplace=True)
predictions_people_reached.rename(columns={'Prediction': 'Predicted People Reached'}, inplace=True)
```

Set x and y ranges

```python
#set xlim range
max_spend_per_campaign = df.groupby('Campaign Name')['Advertising Spend'].max()
second_longest_cutoff = max_spend_per_campaign.sort_values(ascending=False).iloc[1]

#ylim ranges
ylim_range_customers = (predictions_customers['Customers Acquired by Campaign'].min(), predictions_customers['Customers Acquired by Campaig
ylim_range_cpa = (predictions_cpa['Cost per Acquisition'].min(), predictions_cpa['Cost per Acquisition'].quantile(0.95))
ylim_range_roas = (predictions_roas['ROAS'].min(), predictions_roas['ROAS'].quantile(0.95))
ylim_range_people_reached = (predictions_people_reached['People Reached'].min(), predictions_people_reached['People Reached'].quantile(0.95
```

Python Code (7/7)

Plot results using scatterplots with regression line

```
#Plot
#Customers Acquired
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Advertising Spend', y='Customers Acquired by Campaign', hue='Campaign Name', data=predictions_customers, palette='tab10'
sns.lineplot(x='Advertising Spend', y='Predicted Customers Acquired', hue='Campaign Name', data=predictions_customers, palette='tab10', leg
plt.title('Linear Regression for Customers Acquired by Campaign')
plt.xlabel('Advertising Spend')
plt.ylabel('Customers Acquired')
plt.xlim(0, second_longest_cutoff)
plt.ylim(ylim_range_customers)
plt.legend(title='Campaign', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()

# CPA
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Advertising Spend', y='Cost per Acquisition', hue='Campaign Name', data=predictions_cpa, palette='tab10')
sns.lineplot(x='Advertising Spend', y='Predicted Cost per Acquisition', hue='Campaign Name', data=predictions_cpa, palette='tab10', legend=
plt.title('Linear Regression for Cost per Acquisition by Campaign')
plt.xlabel('Advertising Spend')
plt.ylabel('Cost per Acquisition')
plt.xlim(0, second_longest_cutoff)
plt.ylim(ylim_range_cpa)
plt.legend(title='Campaign', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()

# ROAS
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Advertising Spend', y='ROAS', hue='Campaign Name', data=predictions_roas, palette='tab10')
sns.lineplot(x='Advertising Spend', y='Predicted ROAS', hue='Campaign Name', data=predictions_roas, palette='tab10', legend=False)
plt.title('Linear Regression for ROAS by Campaign')
plt.xlabel('Advertising Spend')
plt.ylabel('ROAS')
plt.xlim(0, second_longest_cutoff)
plt.ylim(ylim_range_roas)
plt.legend(title='Campaign', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

Thank You

Thank you

Q&A