CASE STUDY

CAMPAIGN PERFORMANCE ANALYSIS

Understanding Consumer Behavior Through EDA
For Effective Advertising Strategies

Developed by:
Fania Rafalina Fadli

Tools Used : python™ Visual Studio Code

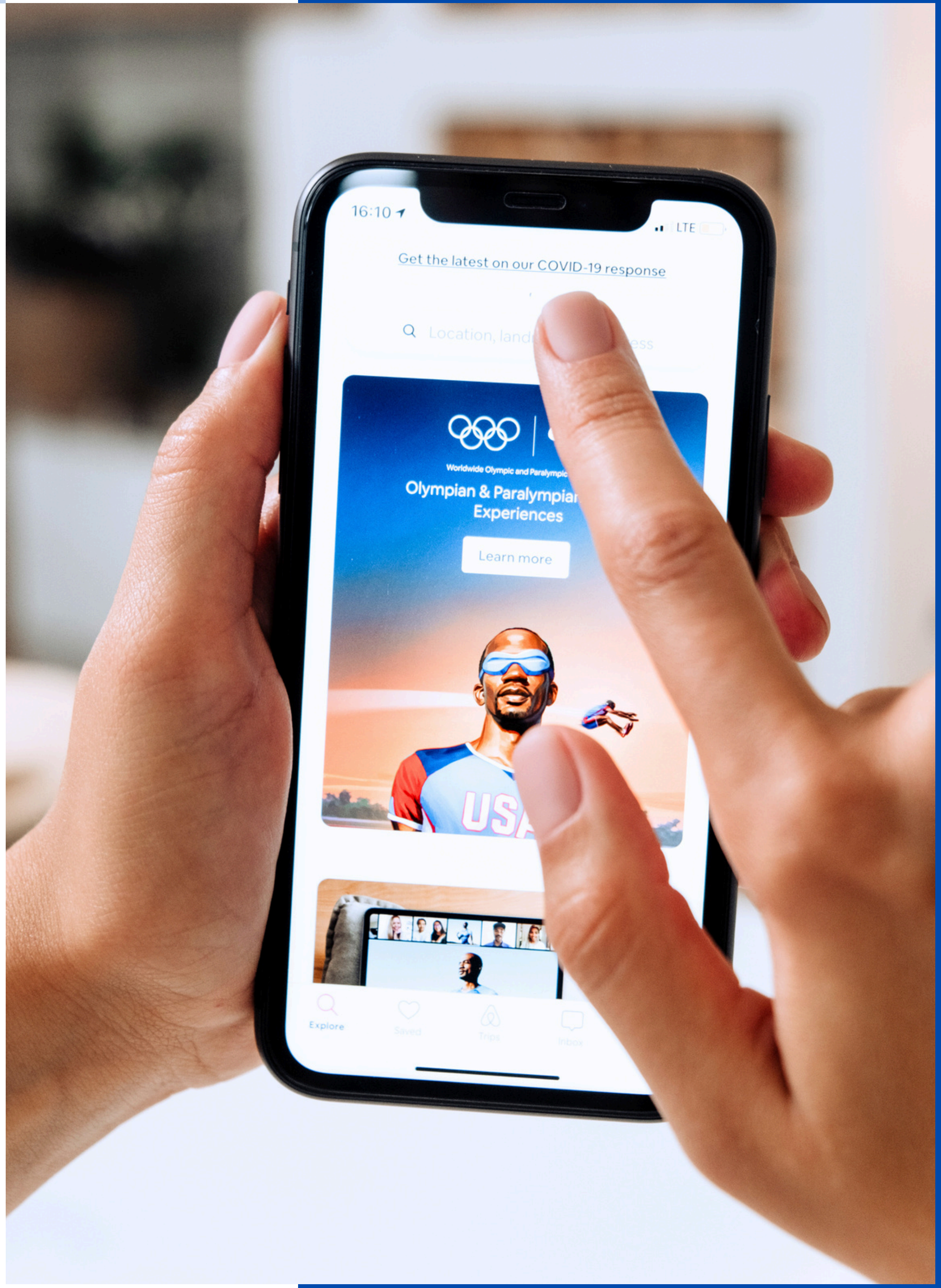
TABLE OF CONTENT

OVERVIEW

GOOGLE ADS SALES DATASET

The dataset contains one year of advertising performance data related to promotional campaigns for Data Analysis course. It was collected from Google Ads, an online advertising platform that enables individuals and organizations to promote websites, products, or services through paid advertisements. The dataset includes various performance metrics, allowing business owners to conduct analysis and develop high-performing advertising strategies. However, there are several issues that need to be addressed to ensure the reliability of insights before processing the data.

- Spelling Errors
- Inconsistent Format
- Duplicate Data
- Conflicting Symbols
- Mixed Casing
- Missing Values



UNDERSTANDING DATA

```python
# Read CSV file
import pandas as pd
df = pd.read_csv("C:/Users/FR/Downloads/google_ads.csv")
```

	Ad_ID	Campaign_Name	Clicks	Impressions	Cost	Leads	Conversions	Conversion Rate	Sale_Amount	Ad_Date	Location	Device	Keyword
0	A1000	DataAnalyticsCourse	104.0	4498.0	$231.88	14.0	7.0	0.058	$1,892	11/16/2024	hyderabad	desktop	learn data analytics
1	A1001	DataAnalyticsCourse	173.0	5107.0	$216.84	10.0	8.0	0.046	$1,679	20-11-2024	hyderabad	mobile	data analytics course
2	A1002	Data Anlytics Corse	90.0	4544.0	$203.66	26.0	9.0	NaN	$1,624	11/16/2024	hyderabad	Desktop	data analitics online
3	A1003	Data Analytcis Course	142.0	3185.0	$237.66	17.0	6.0	NaN	$1,225	11/26/2024	HYDERABAD	tablet	data anaytics training
4	A1004	Data Analytics Corse	156.0	3361.0	$195.90	30.0	8.0	NaN	$1,091	11/22/2024	hyderabad	desktop	online data analytic
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2595	A3595	DataAnalyticsCourse	88.0	5344.0	$242.07	17.0	9.0	0.054	$1,418	29-11-2024	HYDERABAD	MOBILE	online data analytic
2596	A3596	DataAnalyticsCourse	154.0	3211.0	$248.28	14.0	6.0	0.039	$1,950	11/28/2024	hyderabad	TABLET	data analitics online
2597	A3597	Data Anlytics Corse	113.0	3808.0	$233.25	18.0	4.0	0.035	$1,085	11/2/2024	Hyderbad	desktop	data anaytics training
2598	A3598	Data Analytics Corse	196.0	5853.0	$220.13	16.0	7.0	0.036	$1,558	11/8/2024	hydrebad	Tablet	data anaytics training
2599	A3598	Data Analytics Corse	196.0	5853.0	$220.13	16.0	7.0	0.036	$1,558	11/8/2024	hydrebad	Tablet	data anaytics training

2600 rows × 13 columns

```
df.info()

Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Ad_ID            2600 non-null   object
 1   Campaign_Name    2600 non-null   object
 2   Clicks           2489 non-null   float64
 3   Impressions      2546 non-null   float64
 4   Cost             2504 non-null   object
 5   Leads            2552 non-null   float64
 6   Conversions      2526 non-null   float64
 7   Conversion Rate  1975 non-null   float64
 8   Sale_Amount      2461 non-null   object
 9   Ad_Date          2600 non-null   object
 10  Location         2600 non-null   object
 11  Device           2600 non-null   object
 12  Keyword          2600 non-null   object
dtypes: float64(5), object(8)
```

Overview On The Columns

- Ad_ID: Unique campaign identifier
- Campaign_Name: Advertised campaign name
- Clicks: Number of clicks received
- Impressions: Number of times the ad was shown
- Cost: Total advertising cost
- Leads: Post-click actions (e.g., sign-up)
- Conversions: Final actions (e.g., form submission)
- Conversion Rate: Ratio of conversions to clicks
- Sale_Amount: Revenue from conversions
- Ad_Date: Scheduled ad date
- Location: Targeted location
- Device: Targeted device
- Keyword: Trigger keyword for the ad

Dataset Structure

- The dataset contains 2600 entries of promotional campaign.
- There are 13 different columns, including 5 numerical and 8 categorical columns, with some missing values.

DATA STANDARDIZATION

01 Remove currency symbols

```
df["Cost"] = pd.to_numeric(df["Cost"].replace("[$,]", "", regex=True), errors='coerce')
df["Sale_Amount"] = pd.to_numeric(df["Sale_Amount"].replace("[$,]", "", regex=True), errors='coerce')
```

Cost	Sale_Amount
231.88	1892.0
216.84	1679.0
203.66	1624.0
237.66	1225.0
195.90	1091.0
...	...

Some columns, such as **Cost** and **Sales_Amount**, contained currency symbols. By removing these symbols, the columns were successfully converted into numeric data types, enabling further analysis and insight generation.

02 Standardize date format

```
from datetime import datetime

## Function to standardize date format
def standardize_ad_date(date_str):
    # Convert DD-MM-YYYY to MM/DD/YYYY
    if '-' in date_str:
        try:
            dt = datetime.strptime(date_str, '%d-%m-%Y')
            return dt.strftime('%m/%d/%Y')
        except ValueError:
            return date_str
    return date_str

## Apply function to data frame
df['Ad_Date'] = df['Ad_Date'].apply(standardize_ad_date)
```

Ad_Date
11/16/2024
11/20/2024
11/16/2024
11/26/2024
11/22/2024
...
11/29/2024
11/28/2024
11/2/2024
11/8/2024
11/8/2024

Dates in the **Ad_Date** column appeared in inconsistent formats, using different separators (e.g., '-' and '/') and varying day-month order. The datetime library was used to standardize all dates into the MM/DD/YYYY format.

03 Finding & Fixing Typos

```python
string_columns = ['Campaign_Name', 'Location', 'Device', 'Keyword']

val_counts_all = []
for col in string_columns:
    counts = df[col].value_counts()
    val_counts_all.append((col, counts))

for col_name, counts in val_counts_all:
    print(f"\nValue counts for column: {col_name}")
    print(counts)
```

```
Value counts for column: Campaign_Name
Campaign_Name
Data Analytcis Course    680
Data Analytics Corse     647
DataAnalyticsCourse      637
Data Anlytics Corse      636
Name: count, dtype: int64

Value counts for column: Location
Location
HYDERABAD    660
Hyderbad     656
hyderabad    650
hydrebad     634
Name: count, dtype: int64
```

```
Value counts for column: Device
Device
MOBILE     311
tablet     305
Desktop    305
desktop    304
Mobile     291
TABLET     279
DESKTOP    278
mobile     276
Tablet     251
Name: count, dtype: int64
```

```
Value counts for column: Keyword
Keyword
online data analytic      453
learn data analytics      444
data analytics course     440
analytics for data        428
data analitics online     420
data anaytics training    415
Name: count, dtype: int64
```

```python
## Campaign_Name
df["Campaign_Name"] = "Data Analytics Course"

## Location
df['Location'] = "Hyderabad"

## Keyword
keyword_map = {
    'online data analytic': 'online data analytics',
    'data analitics online': 'data analytics online',
    'data anaytics training': 'data analytics training',
}
df['Keyword'] = df['Keyword'].replace(keyword_map, regex=True)


# Capitalize values
df["Device"] = df["Device"].str.capitalize()
df["Keyword"] = df["Keyword"].str.capitalize()
```

Step 1: Investigate all string columns

All string columns were checked for unique values along with their frequencies to identify potential inconsistencies.

- **Campaign_Name, Location**: Spelling variations were found despite referring to the same entity.
- **Device**: Entries referring to the same device appeared in different casing formats (e.g., "Mobile" vs "mobile").
- **Keyword**: Some typos were detected, indicating the need of consistency improvement for future optimization.

Step 2: Handle typos for each column

- **Standardization**: Replaced all variations in the Campaign_Name and Location with the same correct value.
- **Normalization**: Capitalized all values in Device column to eliminate differences.
- **Correction**: Corrected values with typo in Keyword column to their proper spelling to improve data accuracy while preserving distinct keyword entries.

MISSING VALUES & DUPLICATES

01 Handling Missing Values

Step 1: Investigate

```
df.isnull().sum()
Ad_ID                0
Campaign_Name        0
Clicks             111
Impressions         54
Cost                96
Leads               48
Conversions         74
Conversion Rate    625
Sale_Amount        139
Ad_Date              0
Location             0
Device               0
Keyword              0
dtype: int64
```

Step 2: Statistical summary

```
df.describe()
```

	Clicks	Impressions	Cost	Leads	Conversions	Conversion Rate	Sale_Amount
count	2489.000000	2546.000000	2504.000000	2552.000000	2526.000000	1975.000000	2461.000000
mean	138.979912	4523.437942	215.092636	20.005486	6.519794	0.048973	1498.804145
std	34.631298	870.131982	20.285794	6.030756	2.272392	0.019984	287.034407
min	80.000000	3000.000000	180.010000	10.000000	3.000000	0.015000	1000.000000
25%	110.000000	3764.000000	197.540000	15.000000	5.000000	0.035000	1248.000000
50%	139.000000	4518.500000	215.580000	20.000000	7.000000	0.046000	1505.000000
75%	169.000000	5279.500000	232.980000	25.000000	9.000000	0.058000	1742.000000
max	199.000000	5999.000000	249.890000	30.000000	10.000000	0.123000	2000.000000

Step 3: Fill in missing values

```
normal_dist_cols = ['Clicks', 'Cost', 'Leads', 'Conversion_Rate']
skewed_cols = ['Impressions', 'Conversions', 'Sale_Amount']

for column in df.columns:
    if df[column].dtype == 'object':
        df[column].fillna(df[column].mode()[0], inplace=True)
    elif column in normal_dist_cols:
        df[column].fillna(df[column].mean(), inplace=True)
    elif column in skewed_cols:
        df[column].fillna(df[column].median(), inplace=True)
    else:
        df[column].fillna(df[column].mean(), inplace=True)
```

Investigation shows that there were found 7 numerical columns with missing values. By comparing the value of mean and median, we'll be able to determine the distribution of each column and how to handle the missing values.

- **Normal distribution**: mean ≥ median; missing values were filled in with mean
- **Skewed distribution**: mean < median; missing values were filled with median

MISSING VALUES & DUPLICATES

02 Handling Duplicates

Step 1: Check for duplicates

```
check_duplicate = df["Ad_ID"].duplicated().sum()
print(check_duplicate)
```

Step 2: Drop & Re-check

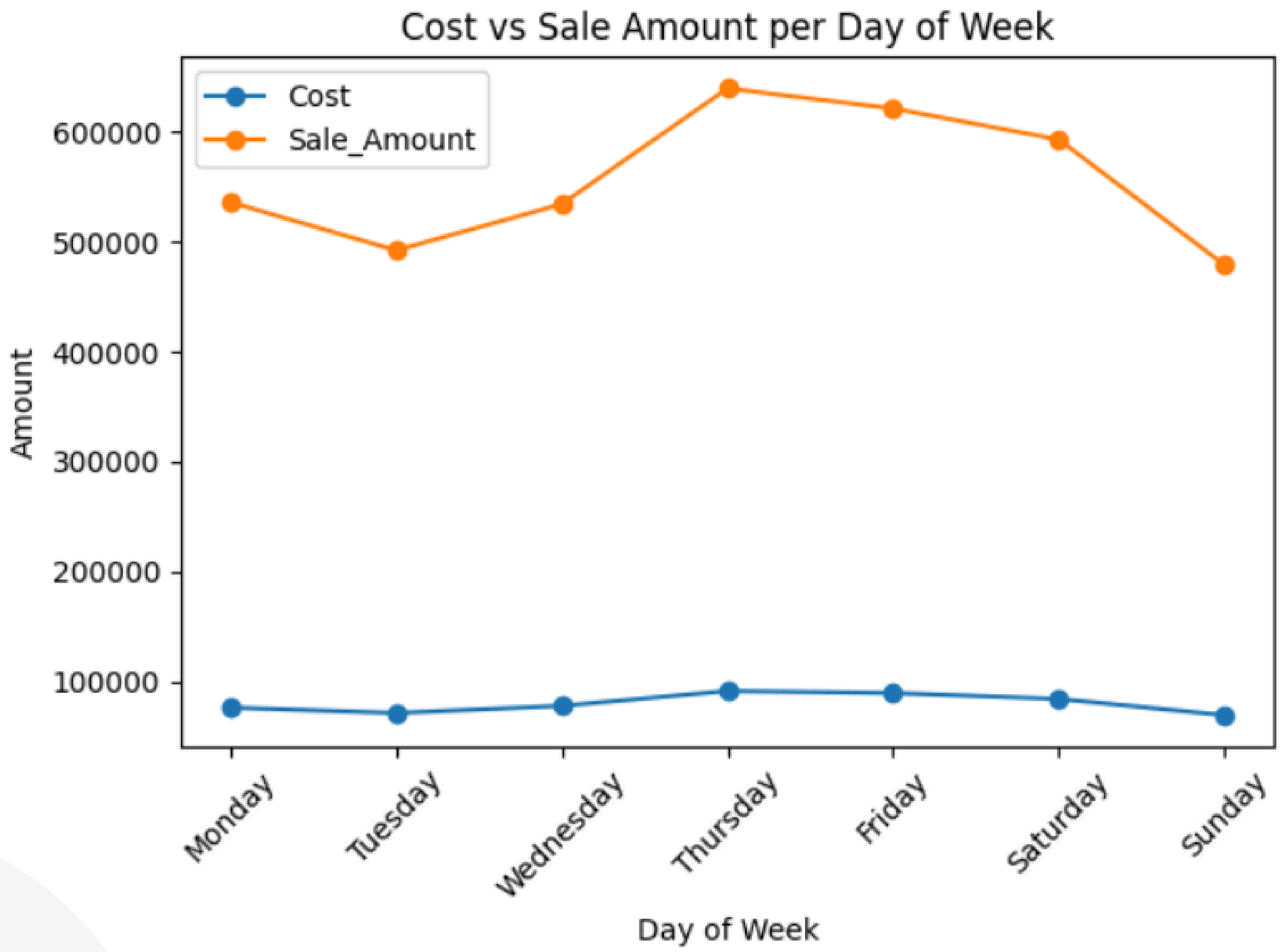
```
df = df.drop_duplicates()

# Recheck for duplicate Ad ID after elimination
recheck_duplicate = df["Ad_ID"].duplicated().sum()
print(recheck_duplicate)
```

Ad_ID was used to identify duplicates, as it serves as a unique identifier for each campaign. Among the 2,600 entries, **one duplicate** Ad_ID was found.

Duplicates are essential to be removed in order to prevent bias and ensure accuracy in analysis by avoiding double-counting. To remove the duplicate, df.drop_duplicates() was applied. A recheck **confirmed that there is no duplicates left** and all remaining Ad_IDs are unique.

DATA VISUALIZATIONS & INSIGHTS

1 - AVERAGE DAILY COST & SALES



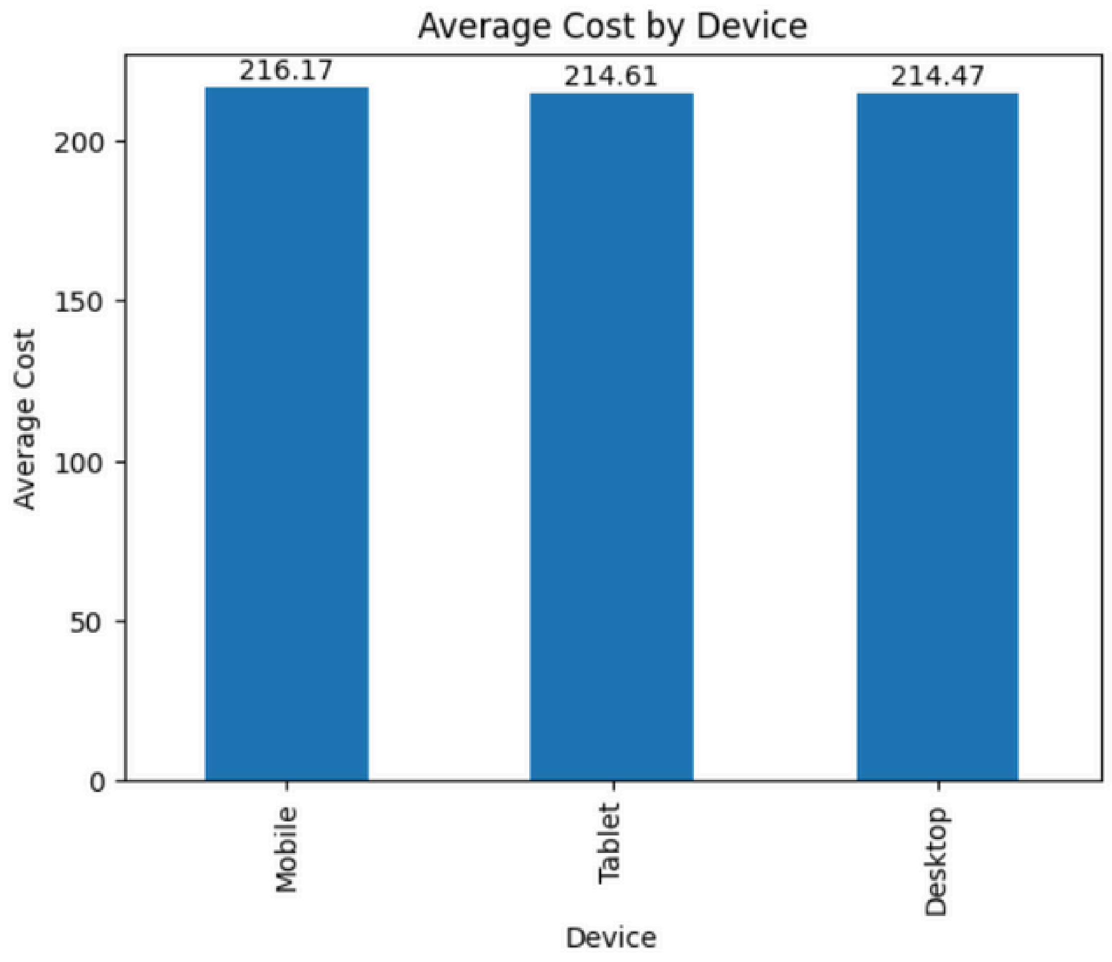
Return On Ad Spend (ROAS) Pattern

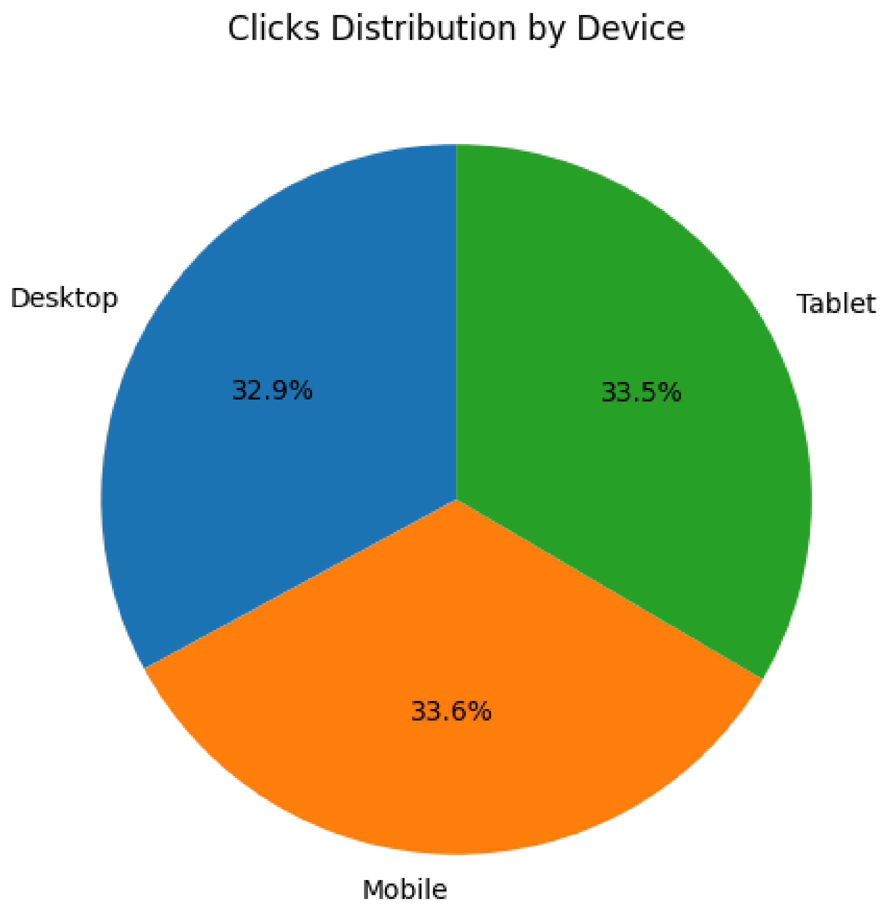
The graph suggests that Google Ads' cost policy remains consistent throughout the week. However, sales performance varies significantly. Thursday and Friday yield the highest returns on ad spend, indicating stronger customer engagement on those days.

Key Takeaway

Ad cost is stable, but customer behavior isn't. Aligning ad strategies with days of higher return can improve campaign effectiveness.

2 - COST & CLICKS PER DEVICE





Slight Difference in Average Cost

The graph shows that Google Ads tends to incur slightly higher rates for ads displayed on mobile devices, while advertising costs on tablet and desktop are relatively lower.

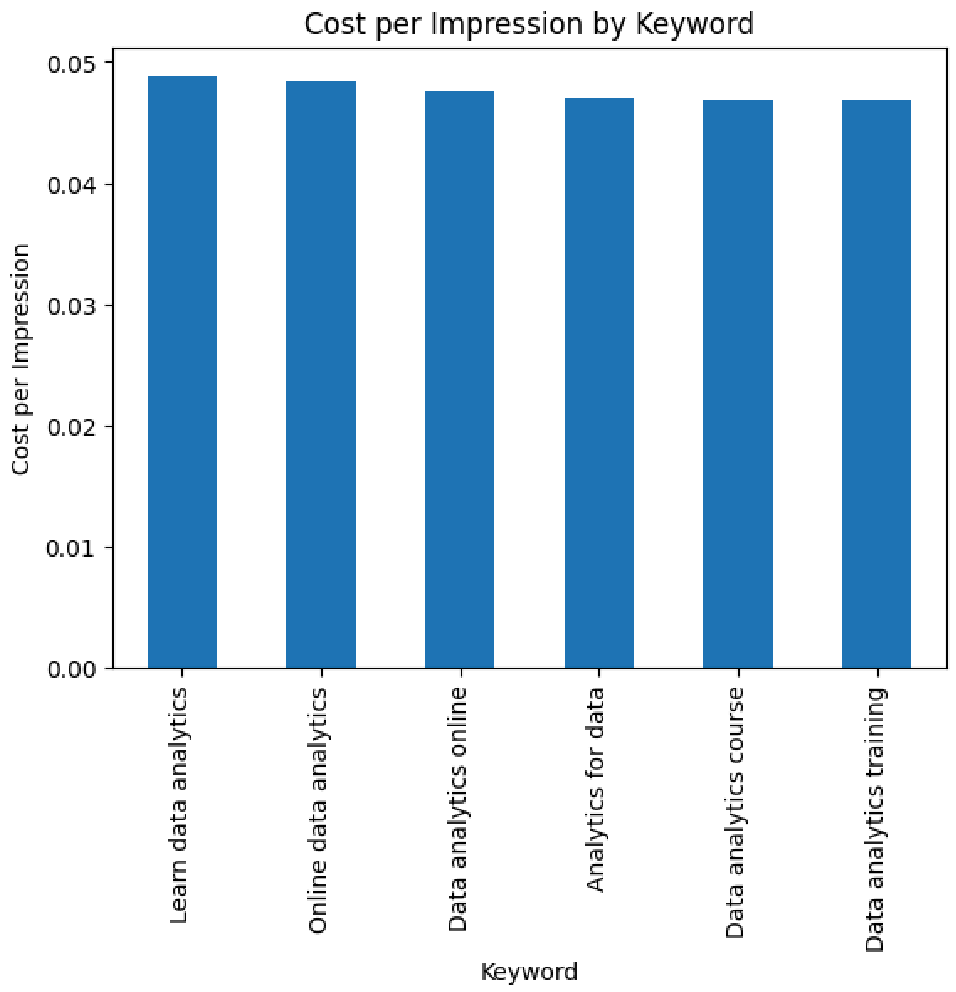
Even Clicks Distribution

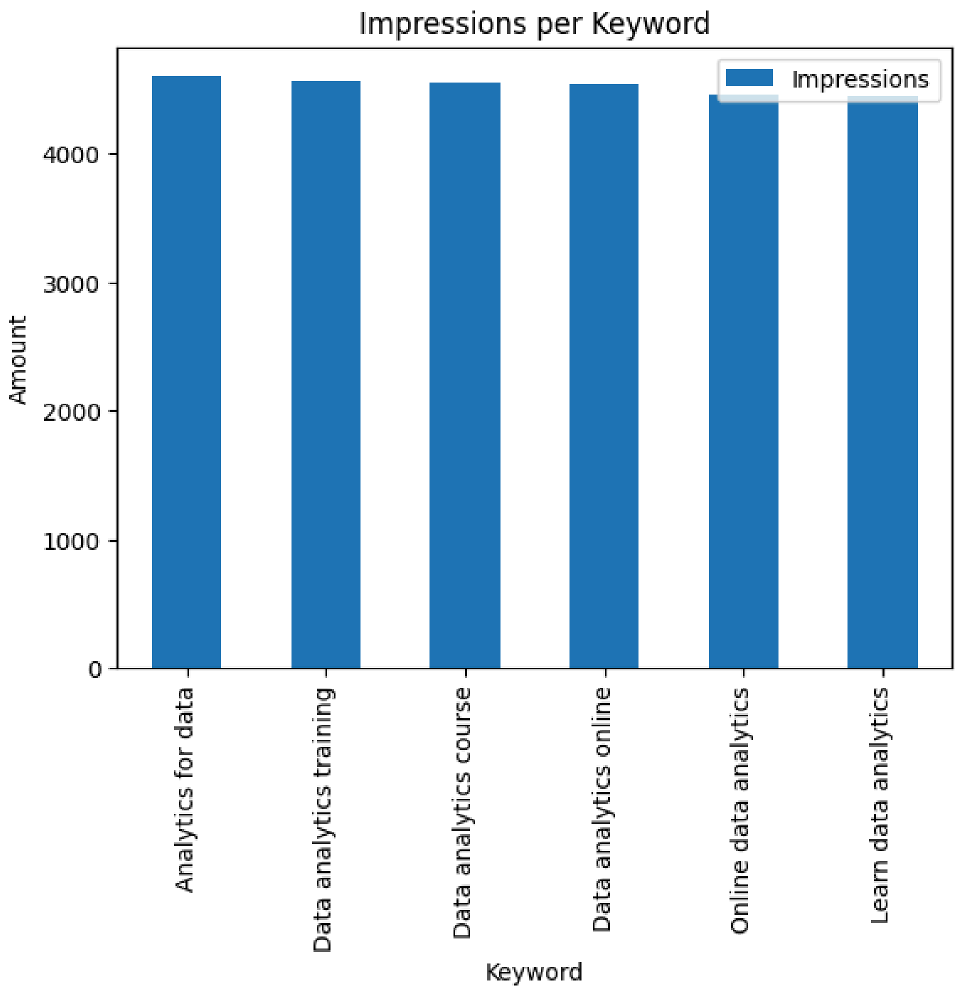
Among the data entries, campaigns receive a fairly even amount of average clicks regardless of the device type, suggesting consistent engagement.

Key Takeaway

Users with different device types interact with ads on similar rate. Additionally, campaign targetting mobile device users might require slightly larger budget allocation due to higher advertising cost.

3 - KEYWORD PERFORMANCE





Higher performing and cost-efficient keywords

The left graph reveals that keywords like “Analytics for data”, “Data analytics course”, and “Data analytics training” demonstrate lower cost per impression. Moreover, the descendingly sorted graph on the right shows that they also generate the highest number of impressions, despite having slight difference. This indicates that these keywords are not only more effective in reaching potential audience, but also more cost-efficient.

Key Takeaway

Keyword prioritization improves cost-efficiency and boosts visibility, ultimately increasing potential of conversion.

RECOMMENDATIONS

BASED ON KEY TAKEAWAYS ON VISUALIZATION INSIGHTS

Ads Scheduling

Prioritize ad placement on Thursdays and Fridays, when return on ad spend tends to be the highest.

Device Targetting

Since cost and click rates are fairly consistent across devices, targetting can **remain broad without requiring device-specific strategies**.

Keyword Selection

Consider combining discount strategies with cost-efficient keywords (e.g., Analytics for data) to boost reach and optimize spending.

FIND MORE ABOUT THE PROJECT ON GITHUB.

GitHub Repository

LET'S CONNECT!

faniarafalina@gmail.com

linkedin.com/fania-rafalina-fadli