BCIS 5110- Basic Programming in Business Analytics

Group-8
Social Media - performance marketing data and Analysis

S No	Name	Email	ID No
1	Anudeep Kumar Polagoni	anudeepkumarpolagoni@my.unt.edu	11594945
2	Rishitha Polavarapu	RishithaPolavarapu@my.unt.edu	11566008

Social Media Performance Marketing Campaigns and data Analysis:

Executive Summary:

What is Performance Marketing?

- Performance Marketing is the digital marketing strategy that is driven by results to track the performance of ad campaigns with certain metrics.
- This strategy is ideal for the companies that are looking to scale up in digital space, through brand awareness, Reach, Traffic, App Installs, Sales.

Observed, Targeted and Derived metrics:

- Observed metrics in social performance marketing campaigns are impressions, link clicks, reach, app install, sales, spends, revenue, transactions.
- Targeted metrics is the userbase we are targeting to showcase an ad basis their age, gender, user device, location, like and interest, socially observed interest.
- Derived metrics are the calculated results from observed metrics like CPM, CTR, CVR, ROAS, AOV, CPI.

Objective Summary of the project:

The data is performance results of Facebook social -marketing campaigns for an e-commerce startup based in India. Here, the data has the metrics on day-on-day level, with target based on gender and age. Objective is to analyze the data on day-on-day level, campaign level, gender level, age level and understand what worked and what not worked. Basis that the strategic actions can be taken on the campaigns that could increase the performance of campaigns, which results in higher revenue on spends and increase in conversion rate through link clicks to sales.

Project motivation/background:

The data set is driven from E-commerce startup based in India. I, Anudeep worked as Business analyst at merkle sokrati at Pune, India. As I worked in the field of performance marketing, I got the data from one of my friend's startup real time data of e-commerce business startup. My motivation is to analyze the data in all possible ways on campaign, gender, age and on day level to understand what worked to drive the performance, Basis that driving the insights to do strategic moves which could result in increased performance.

Key Questions:

1. Calculate the derived metrics from observed metrics on day-on-day level.
2. Calculate the derived metrics from observed metrics on month-on-month level.
3. Calculate the derived metrics from observed metrics on Campaign level
4. Calculate the derived metrics from observed metrics on Age Group level
5. Calculate the derived metrics from observed metrics on Gender level.
6. Calculate the derived metrics from observed metrics on Campaigns on Gender level.
7. Calculate ROAS, AOV on Gender level
8. Represent all the insights in 2D.
9. Predict the revenue using linear regression and forest regression model
10. Forecasting which model is better basis the accuracy

Data Source:

Real time data driven E-commerce startup from my known associate's business in India. (Not from any site). It is a data from Facebook Business Manager

Data Description:

- This is social – marketing campaign data set of a startup in E-commerce Industry.
- The data set have the volume of 2944 rows and 11 columns.
- The 11 columns are Campaigns name, impressions, link clicks, sessions, add to cart, transactions, revenue, spends, gender, age and date.
- Impressions- this is an incident captured, when a user see an add on screen.
- Link clicks- number of times, link clicked by the user
- Sessions – number of times, the user opened the website after click on the ad
- Add to cart- number of times, the user added the product to cart
- Transaction/Purchase – number of purchases made by the user.
- Revenue – the total revenue generated by the user on buying a service or product.
- Spends – the amount spends on the campaign to be active in digital space.
- Gender- targeting audience on gender base- Male and Female
- Age – Targeting the audience on age base 20-29,30-39,40-49,50-59
- Date – The data is on campaign/ day-on-day level for 3 months March, April and May
- Date, campaign name, age and gender are categorial variable
- Impression, Link clicks, spends, revenue, transactions/Purchases, sessions, add to cart are numerical variables.

Data Transformation/Exploratory Data Analysis:

- Exploratory data analysis is done on our dataset to understand the depth in the data and learn the different characteristics.
- We have done the EDA in the following steps:
  1. Data collection: here we have loaded the data in to our system

	Date	Campaign	Age	Gender	Amount Spent	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Revenue
0	2022-01-03 00:00:00	Campaign A	20-29	Female	2163.385064	18909.488890	220.541274	117.125263	21.090037	2	1495.861037
1	2022-01-03 00:00:00	Campaign A	30-39	Female	8110.415767	82701.843650	2281.995042	928.639982	275.269924	16	18746.794440
2	2022-01-03 00:00:00	Campaign A	40-49	Female	1558.589609	11849.907370	616.785586	337.026129	86.666189	1	1163.291941
3	2022-01-03 00:00:00	Campaign A	50-59	Female	2503.404864	18316.510180	245.022709	123.951641	30.589329	1	937.495158
4	2022-01-03 00:00:00	Campaign A	20-29	Male	7980.083393	109587.911600	8001.253998	4735.728220	871.807829	44	64590.288480
...	...	...	...	...	...	...	...	...	...	...	...
2939	31-05-2022	Campaign D	50-59	Female	2899.238863	60249.615000	2595.246913	1440.220551	393.559848	10	9042.607544
2940	31-05-2022	Campaign D	20-29	Male	1462.063713	27823.308280	1621.711950	774.613475	174.229844	14	16110.514790
2941	31-05-2022	Campaign D	30-39	Male	388.830119	3004.304679	102.183333	46.995709	11.145817	0	532.385418
2942	31-05-2022	Campaign D	40-49	Male	1875.316301	39072.414600	2079.213481	840.209212	119.124388	8	8974.965820
2943	31-05-2022	Campaign D	50-59	Male	773.042785	10060.020180	547.768661	325.665101	47.520579	2	2573.324878

2944 rows × 11 columns

  2. Information regarding the data.

```
In [24]: social_campaign_performance.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2944 entries, 0 to 2943
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          2944 non-null   datetime64[ns]
 1   Campaign      2944 non-null   object
 2   Age           2944 non-null   object
 3   Gender        2944 non-null   object
 4   Amount Spent  2944 non-null   float64
 5   Impressions   2944 non-null   float64
 6   Link Clicks   2944 non-null   float64
 7   Sessions      2944 non-null   float64
 8   Adds to Cart  2944 non-null   float64
 9   Transactions  2944 non-null   int64
 10  Revenue       2944 non-null   float64
dtypes: datetime64[ns](1), float64(6), int64(1), object(3)
memory usage: 218.6+ KB
```

  3. Data Cleaning
- Used the following libraries:
  - Import pandas as pd
  - Import numpy as np

- From matplotlib import pyplot as plt
- Import seaborn as sns

- We have checked if there are any null values present in the dataset.

```
social_campaign_performance.isnull().sum()
```

```
Date            0
Campaign        0
Age             0
Gender          0
Amount Spent    0
Impressions     0
Link Clicks     0
Sessions        0
Adds to Cart    0
Transactions    0
Revenue         0
dtype: int64
```

- ***As per the above output there are no null values***

- We have checked if there are any missing values present in the dataset

```
social_campaign_performance.isna().sum()
```

```
Date            0
Campaign        0
Age             0
Gender          0
Amount Spent    0
Impressions     0
Link Clicks     0
Sessions        0
Adds to Cart    0
Transactions    0
Revenue         0
dtype: int64
```

- ***As per the above output there are no missing values***
- The cleaning of data is done – Cleaning date column to usable format

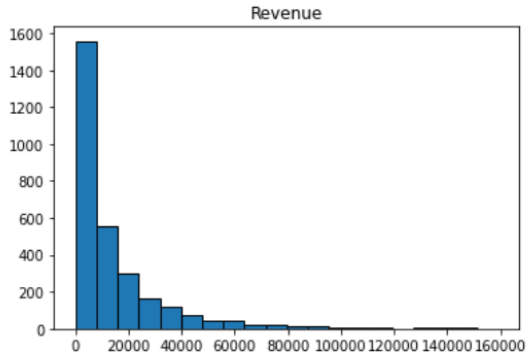
Descriptive Analysis:

- The transformation of unstructured data into a format that is simple to comprehend and interpret, i.e., the rearranging, sorting, and manipulation of data to produce insightful information about the supplied data.

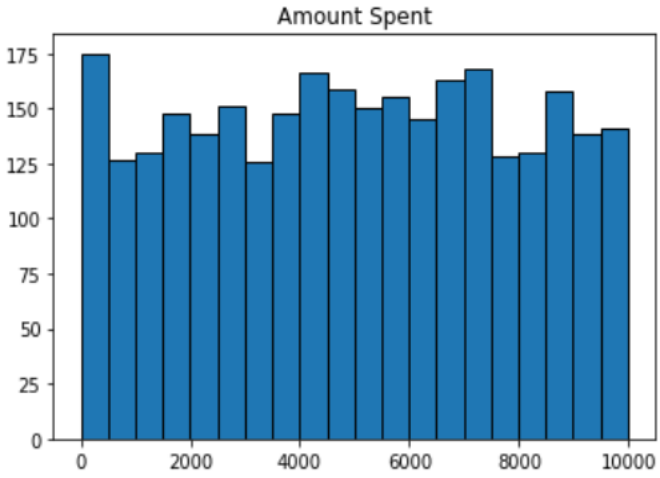
- Descriptive analysis is a sort of data analysis that aids in accurately describing, displaying, or summarizing data points so that patterns may appear that satisfy all of the data's requirements.

- **Univariate analysis, bi variate analysis and calculation of some important values is also done**
- In this section the patterns of how the particular variable is affected is shown in the graphs
- We have used the different graphs like histograms, crosstabs, count plots …
- Univariate and bivariate analysis for few factors is shown below:

Revenue:



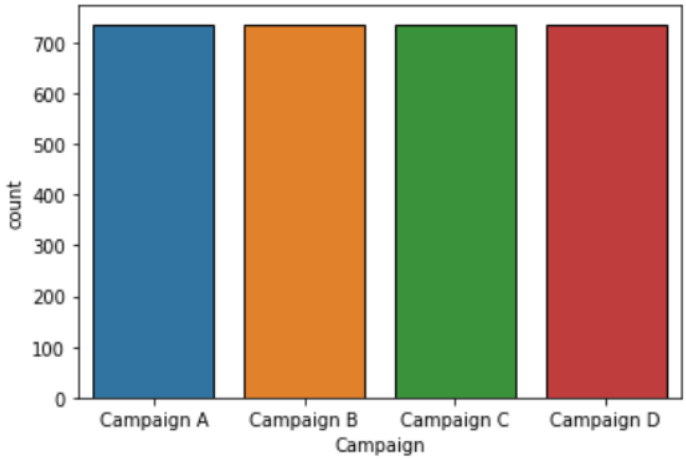
Amount Spent

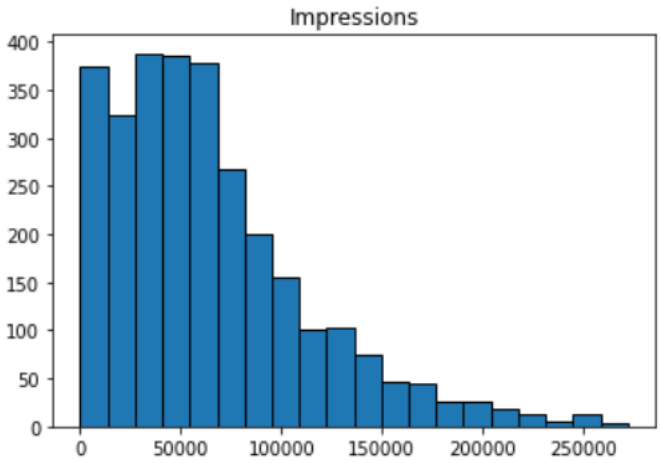


Age

col_0	Frequency
Age	
20-29	736
30-39	736
40-49	736
50-59	736

Campaign





The some of the important factors are calculated which helps in the effective analysis:

This function calculates some of the important metrics helpful to analyze the performance of an ad campaign which includes

CPM - Cost Per Mile - Amount spent for every thousand impressions

CTR - Click Through Rate - Number of link clicks per impression

CPS - Cost Per Sale - Amount spent on advertising for each session

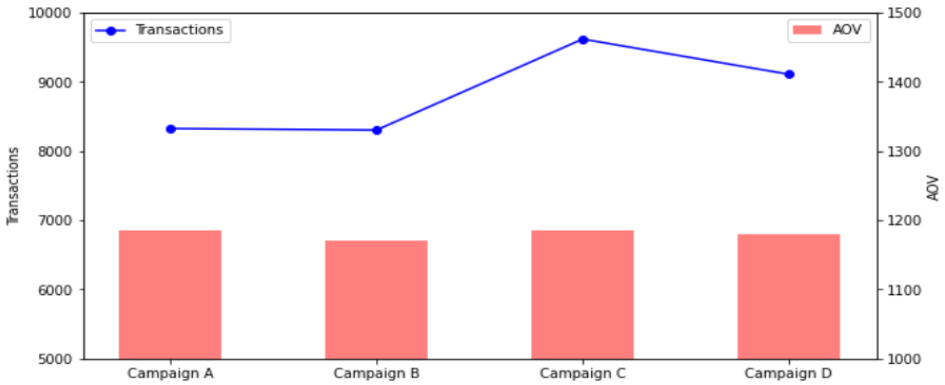
CPT - Cost Per Transaction - Amount spent on advertising per transaction

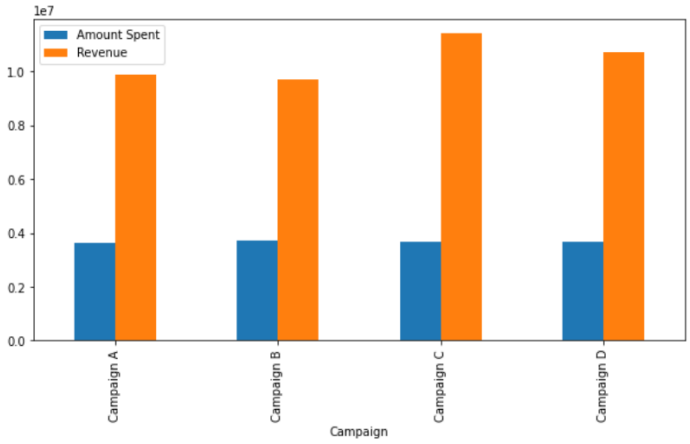
CVR - Conversion Rate - Percentage of link clicks actually got converted to transactions

AOV - Average Order Value - Per order value on an average

ROAS - Return on ad spend - Return on revenue for each buck spent on advertising

Adds to cart CVR - Number of added to carts got actually converted to transactions Based on them the day level campaign level and the age, gender level analysis is done.





Models and Analysis:

Since we have the day level revenue numbers along with the feature set, it is possible to predict the sales using machine learning models for a different set of features.

So, we can consider the day level analysis data frame to make the model upon and predict the revenue. As per use case we consider the revenue as the dependent or target variable. From the above analysis we can infer that the features like impressions, link clicks, sessions, adds to cart, Transactions, Amount spent are almost in a linear relationship with the revenue.

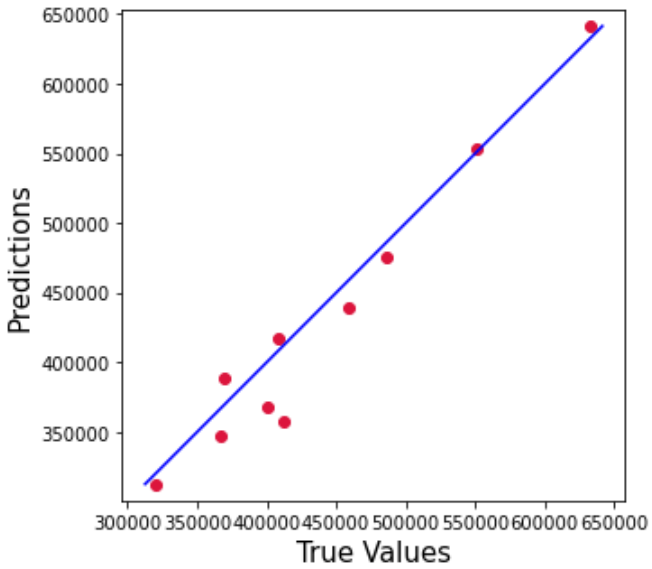
This is clearly a regression problem since the target variable is a continuous variable. So as the other features are in linear relationship, Linear regression would be the best model to apply.

Linear Regression attempts to fit a linear equation to observable data in order to model the relationship between two or more variables. The first variable is regarded as an explanatory variable, whereas the second is regarded as a dependent variable.

A linear regression line has an equation of the form $\boldsymbol{Y = a + bX}$, where $\boldsymbol{X}$ is the explanatory variable and $\boldsymbol{Y}$ is the dependent variable. The slope of the line is $\boldsymbol{b}$, and $\boldsymbol{a}$ is the intercept (the value of $\boldsymbol{y}$ when $\boldsymbol{x} = 0$).

The dataset consists of around twenty thousand data points and out of which we split the data points into training and testing sets in 90% and 10% proportions. The 9:1 ratio is because of the lesser amount of data, and this allows the model to learn more accurately on a larger set of data and perform accordingly. The linear regression function is imported from the scikit learn python module and is being trained with the training data set. The model is being fit with the training data.

After the model is trained with the training dataset, the model is tested towards the testing dataset, now for the predict function we give a dataset that contains all the necessary features and based on the things and trends learnt by the model, it predicts the prices of the revenue accordingly.



The above scatter plot shows the original datapoints and the line predicted by the linear regressor. It shows that the model is performing better.

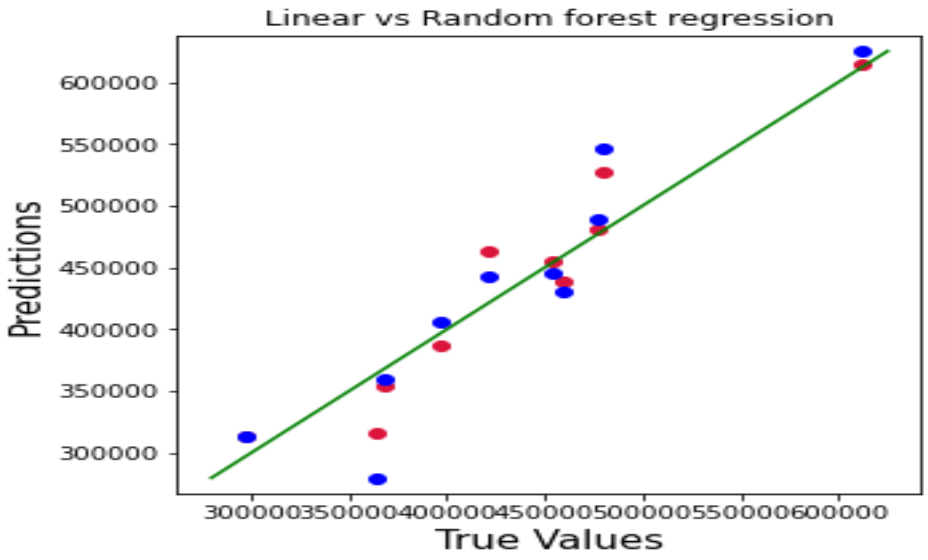
The model is then evaluated on MAPE i.e Mean Absolute Percentage Error, which means the percentage of absolute deviation between the actual and predicted values.

For the predictions on test dataset, the mape came around 0.044, which means that the accuracy i s 1-MAPE i.e 1-0.044. ~0.9556 = 95.56%

Forest Regression Model:

With the aid of several decision trees and a method known as Bootstrap and Aggregation, also ref erred to as bagging, Random Forest is an ensemble methodology capable of handling both regres sion and classification tasks. This method's fundamental principle is to integrate several decision t rees to get the final result rather than depending solely on one decision tree.

For the prediction on test data set, we mapped True values with prediction with linear vs random forest regression. Mape for Random Forest regression model is ~0.064. and which mean the accuracy is 93.54%



<u>Result :</u> This shows that for the available data linear regression model is giving an accuracy of nearly 95% whereas the random forest model gave only 93.5% accuracy. The above scatter plot shows the actual deviation between the data pints across both linear and random forest regression.

Findings and Managerial Implications:

1. Out of all the months, it is observed that march, April and may month have better performance with more revenue with respect to spends.
2. In campaign level analysis, it is observed that campaign C performed well with better conversion rate of 0.48% and AOV of 1,186.00 which is 0.67% higher than rest of the average campaigns AOV, that resulted in highest ROAS of 3.10
3. Despite of highest spends observed in Campaign B, it did not work well resulting in low AOV of 1,169.00 which is 1.15% lesser than the rest of the average AOV, that resulted in least ROAS of 2.62
4. It is observed that the conversion rate of 40-49 age group is higher that rest of all age groups but due to less AOV of 1,178.97 resulted in second place in case of ROAS.
5. 30-39 age group worked well even with lesser conversion rate and higher AOV of 1,185.96 resulted in increased revenue on spends with 3.03
6. For 20-29 age group, with increase in CPM and decrease in link clicks resulted in decrease conversion rate along with overall performance.
7. In Gender base analysis, on overall level performance female campaigns have higher average order value of the products than males but observed that due to increase in Click through rate of male resulted in increase in conversion rate than females.
8. Campaign B did not work for females with CVR of 0.44% resulted in ROAS of 2.48 and Campaign A did not work for males with least conversion rate observed with 0.42% and ROAS 2.70.

Conclusions on Insights Observation and prediction analysis:

- On campaign level performance, the spends on campaign C can be increased by 20% on average total spends by decreasing spends on campaign B. As in campaign B, AOV of the campaign is low so It is better to upgrade the product set with high AOV products, which might help to increase ROAS.
- The spends on the age group of 30-39,40-49 should be increased by decreasing spends on 20-29 age group. As the CTR is low on the age group 20-29, which impacted the performance instead we can concentrate on well performing age group.
- Maintaining spends on female and male targeted campaigns, include the lower AOV products to male catalogue which could result in increase conversion rate.
- On overall level female-campaign B spends should be decrease and instead spend it on campaign C.
- Decreasing spends on Male-Campaign A and split the reduced amount in two part in campaign C and D as they are performing better.
- Why is it important to decrease spends on average and low performing campaigns?
- By decreasing the spends, the ad reaches out to more quality userbase, which could result in increasing clickthrough rate, that can lead to hike in conversion rate and ROAS.

- We also predicted the revenue and ,that shows that for the available data linear regression model is giving an accuracy of nearly 95% where as the random forest model gave only 93.5% accuracy. The above scatter plot shows the actual deviation between the data pints across both linear and random forest regression

In [1]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns

from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_percentage_error
```

In [7]:
```
#EXPLORATORY DATA ANALYSYS
```

In [3]:
```
social_campaign_performance = pd.read_excel(r'C:\Users\HP\Downloads\PROJECT.xlsx', sheet_name = 'Social campaig
```

In [4]:
```
social_campaign_performance.head()
```

Out[4]:

	Date	Campaign	Age	Gender	Amount Spent	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Revenue
0	2022-01-03 00:00:00	Campaign A	20-29	Female	2163.385064	18909.48889	220.541274	117.125263	21.090037	2	1495.861037
1	2022-01-03 00:00:00	Campaign A	30-39	Female	8110.415767	82701.84365	2281.995042	928.639982	275.269924	16	18746.794440
2	2022-01-03 00:00:00	Campaign A	40-49	Female	1558.589609	11849.90737	616.785586	337.026129	86.666189	1	1163.291941
3	2022-01-03 00:00:00	Campaign A	50-59	Female	2503.404864	18316.51018	245.022709	123.951641	30.589329	1	937.495158
4	2022-01-03 00:00:00	Campaign A	20-29	Male	7980.083393	109587.91160	8001.253998	4735.728220	871.807829	44	64590.288480

In [5]:
```
social_campaign_performance.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2944 entries, 0 to 2943
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          2944 non-null   object
 1   Campaign      2944 non-null   object
 2   Age           2944 non-null   object
 3   Gender        2944 non-null   object
 4   Amount Spent  2944 non-null   float64
 5   Impressions   2944 non-null   float64
 6   Link Clicks   2944 non-null   float64
 7   Sessions      2944 non-null   float64
 8   Adds to Cart  2944 non-null   float64
 9   Transactions  2944 non-null   int64
 10  Revenue       2944 non-null   float64
dtypes: float64(6), int64(1), object(4)
memory usage: 253.1+ KB
```

In [6]:
```
social_campaign_performance.describe()
```

Out[6]:

	Amount Spent	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Revenue
count	2944.000000	2944.000000	2944.000000	2944.000000	2944.000000	2944.000000	2944.000000
mean	4990.458580	64275.642044	2587.754287	1289.170920	300.862118	12.010190	14175.300532
std	2860.639174	48991.529997	2697.839226	1358.015867	331.564745	16.035296	19582.131894
min	0.137628	1.362272	0.031589	0.016248	0.002947	0.000000	0.185997
25%	2542.733119	28681.226615	634.202360	312.108640	71.132158	2.000000	1952.430008
50%	5005.308461	54524.323665	1757.670582	878.850807	194.252941	6.000000	7083.053321
75%	7358.553318	87217.859465	3633.639519	1773.312529	409.277480	16.000000	17813.315970
max	9983.109419	272251.486400	21109.974880	11357.164460	2720.942888	143.000000	159231.040800

In [30]:
```
# Data cleaning - Cleaning date column to usable format
```

In [31]:
```
social_campaign_performance['Date'] = social_campaign_performance['Date'].astype('datetime64')
social_campaign_performance = social_campaign_performance.round(decimals = 2)
social_campaign_performance['Link Clicks'] = social_campaign_performance['Link Clicks'].round(decimals = 0)
```

DESCRIPTIVE DATA ANALYSYS

In [32]:
```
# Data distribution across ages and genders
```

In [11]:
```
count_across_ages=pd.crosstab(social_campaign_performance.Age, columns='Frequency')
```

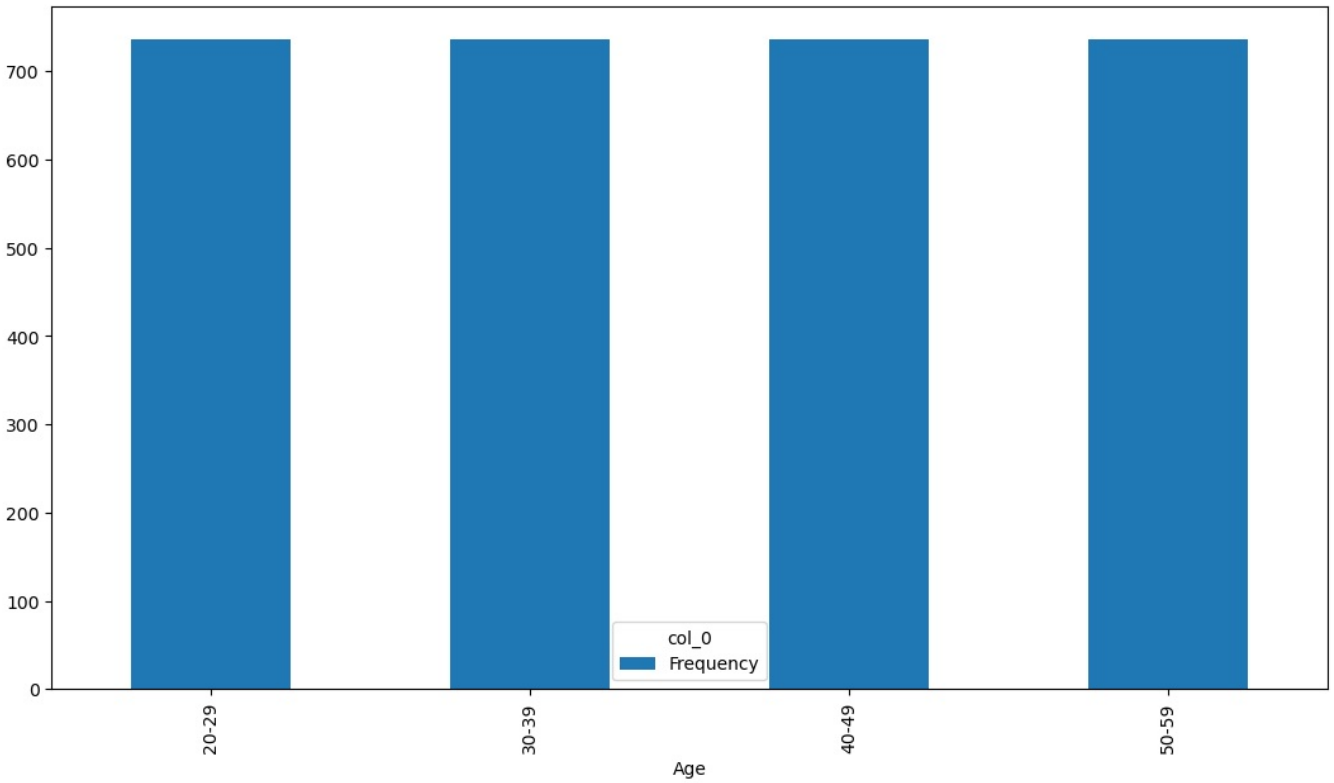
```
count_across_ages
```

Out[11]:

Age	col_0 Frequency
20-29	736
30-39	736
40-49	736
50-59	736

In [13]:
```
count_across_ages.plot(kind='bar',figsize=(13,7))
```

Out[13]:
```
<AxesSubplot:xlabel='Age'>
```



In [15]:
```
social_campaign_performance['Age'].describe()
```

Out[15]:
```
count      2944
unique        4
top       20-29
freq        736
Name: Age, dtype: object
```

In [ ]:
```
#From the above chart the frequency of people is same across different age groups
```

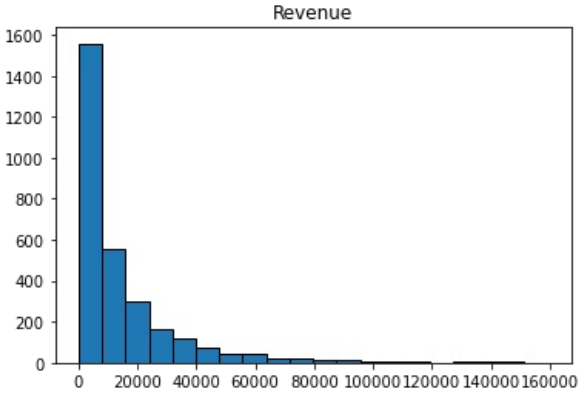
In [34]:
```
count_across_genders=pd.crosstab(social_campaign_performance.Gender, columns='Frequency')
count_across_genders
```

Out[34]:

Gender	col_0 Frequency
Female	1472
Male	1472

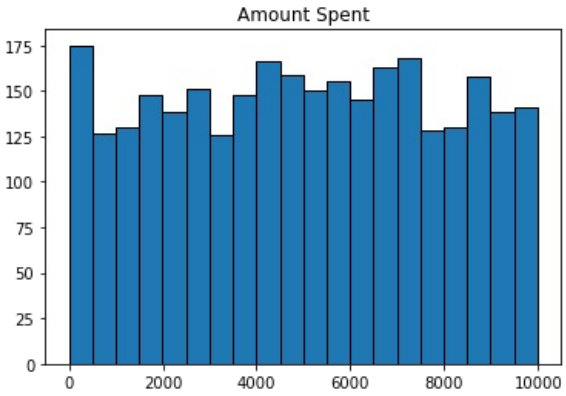
In [35]:
```
social_campaign_performance.hist(column="Revenue",grid=False,figsize=(6,4),bins=20,edgecolor='black')
```

Out[35]:
```
array([[<AxesSubplot:title={'center':'Revenue'}>]], dtype=object)
```



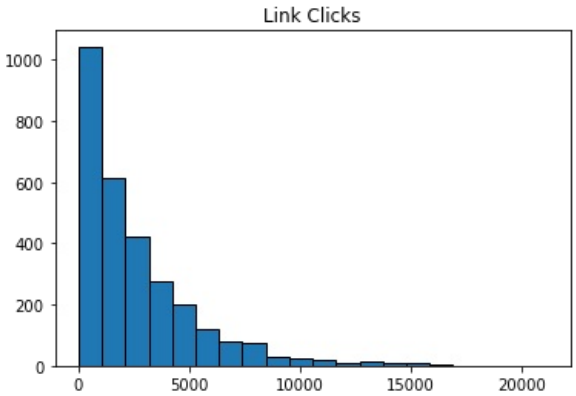
In [36]:
```
social_campaign_performance.hist(column="Amount Spent",grid=False,figsize=(6,4),bins=20,edgecolor='black')
```

Out[36]:
```
array([[<AxesSubplot:title={'center':'Amount Spent'}>]], dtype=object)
```



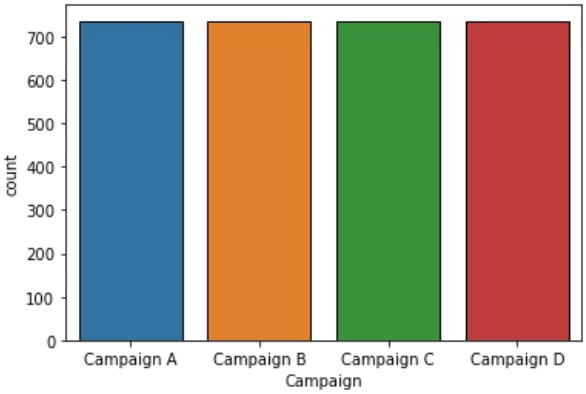
In [37]:
```
social_campaign_performance.hist(column="Link Clicks",grid=False,figsize=(6,4),bins=20,edgecolor='black')
```

Out[37]:
```
array([[<AxesSubplot:title={'center':'Link Clicks'}>]], dtype=object)
```



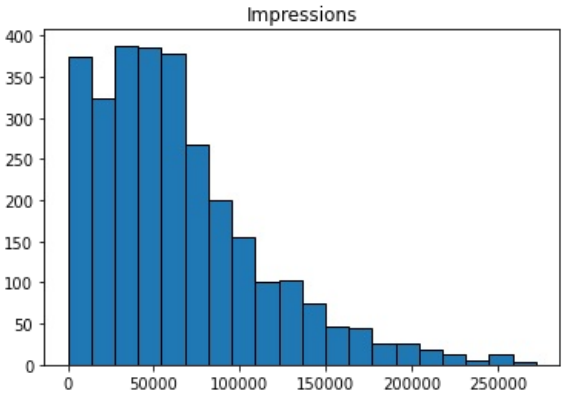
In [38]:
```
sns.countplot(x="Campaign",data=social_campaign_performance,edgecolor='Black')
```

Out[38]:
```
<AxesSubplot:xlabel='Campaign', ylabel='count'>
```



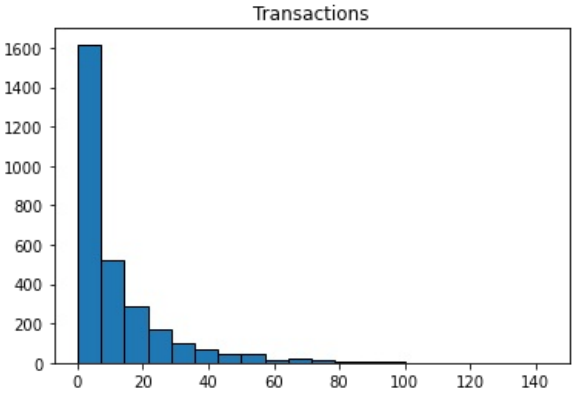
In [39]:
```
social_campaign_performance.hist(column="Impressions",grid=False,figsize=(6,4),bins=20,edgecolor='black')
```

Out[39]:
```
array([[<AxesSubplot:title={'center':'Impressions'}>]], dtype=object)
```



In [40]:
```
social_campaign_performance.hist(column="Transactions",grid=False,figsize=(6,4),bins=20,edgecolor='black')
```

Out[40]:
```
array([[<AxesSubplot:title={'center':'Transactions'}>]], dtype=object)
```



In [41]:
```
def metric_calculator(df):
    """
    This function calculates some of the important merics helpful to
    analyse the performance of an ad campaign which includes

    CPM - Cost Per Mile - Amount spent for every thousand impressions
    CTR - Click Through Rate - Number of link clicks per impression
    CPS - Cost Per Sale - Amount spent on advertising for each session
    CPT - Cost Per Transaction - Amount spent on advertising per transaction
    CVR - Conversion Rate - Percentage of link clicks actually got converted to transactions
    AOV - Average Order Value - Per order value on an average
    ROAS - Return on ad spend - Return on revenue for each buck spent on advertising
    Adds to cart CVR - Number of added to carts got actually converted to transactions
    """

    df['CPM'] = (df['Amount Spent']/df['Impressions'])*1000
    df['CTR (%)'] = (df['Link Clicks']/df['Impressions'])*100
```

```
    df['CPS'] = (df['Amount Spent']/df['Sessions'])
    df['CPT'] = (df['Amount Spent']/df['Transactions'])
    df['CVR (%)'] = (df['Transactions']/df['Link Clicks'])*100
    df['AOV'] = (df['Revenue']/df['Transactions'])
    df['ROAS'] = (df['Revenue']/df['Amount Spent'])
    df['Adds to cart CVR'] = (df['Adds to Cart']/df['Transactions'])
    df = df.round(decimals = 2)

    return df
```

In [42]:
```
# Consolidating the required data at various levels
day_level_aggregated_df = social_campaign_performance.groupby('Date').agg({
    'Impressions':'sum',
    'Link Clicks':'sum',
    'Sessions':'sum',
    'Adds to Cart':'sum',
    'Transactions':'sum',
    'Amount Spent':'sum',
    'Revenue':'sum'
}).reset_index()

campaign_level_aggregated_df = social_campaign_performance.groupby('Campaign').agg({
    'Impressions':'sum',
    'Link Clicks':'sum',
    'Sessions':'sum',
    'Adds to Cart':'sum',
    'Transactions':'sum',
    'Amount Spent':'sum',
    'Revenue':'sum'
}).reset_index()

age_group_level_aggregated_df = social_campaign_performance.groupby('Age').agg({
    'Impressions':'sum',
    'Link Clicks':'sum',
    'Sessions':'sum',
    'Adds to Cart':'sum',
    'Transactions':'sum',
    'Amount Spent':'sum',
    'Revenue':'sum'
}).reset_index()

gender_level_aggregated_df = social_campaign_performance.groupby('Gender').agg({
    'Impressions':'sum',
    'Link Clicks':'sum',
    'Sessions':'sum',
    'Adds to Cart':'sum',
    'Transactions':'sum',
    'Amount Spent':'sum',
    'Revenue':'sum'
}).reset_index()
```

In [43]:
```
day_level_analysis = metric_calculator(day_level_aggregated_df)
day_level_analysis['Day'] = day_level_analysis['Date'].dt.day_name()
day_level_analysis['Month'] = day_level_analysis['Date'].dt.strftime('%B')
```

In [44]:
```
day_level_analysis
```

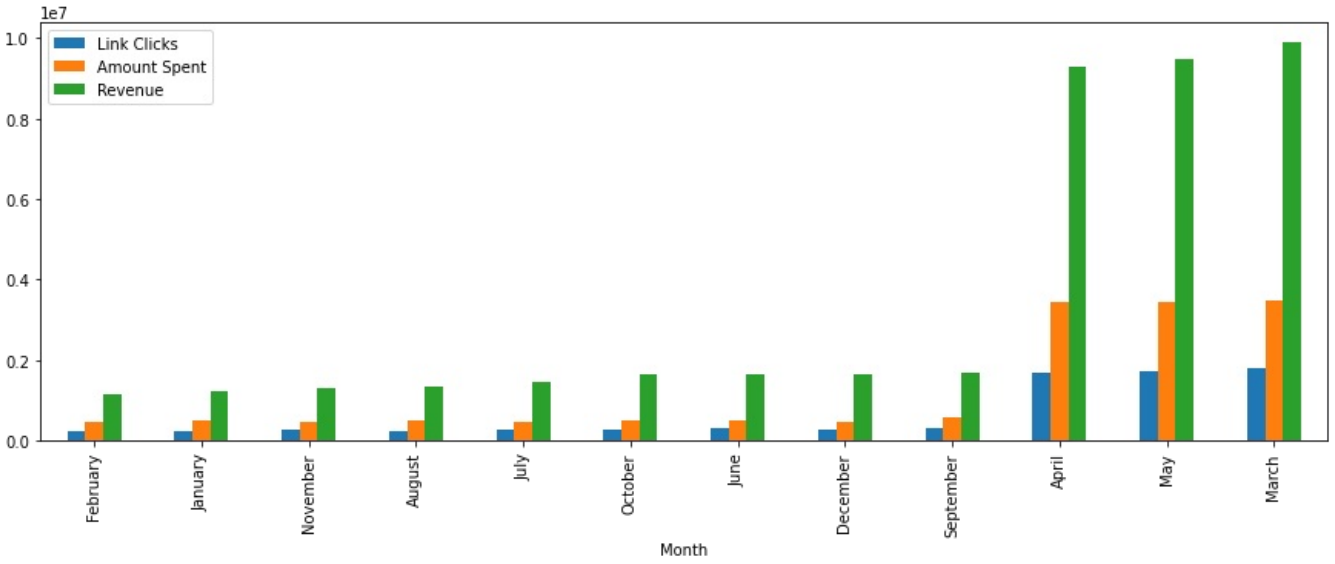
Out[44]:

	Date	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Amount Spent	Revenue	CPM	CTR (%)	CPS	CPT	CVR (%)	AOV	ROAS
0	2022-01-03	2141638.64	88983.0	44838.73	9554.51	511	171487.75	611340.95	80.07	4.15	3.82	335.59	0.57	1196.36	3.56
1	2022-01-04	1781969.25	68575.0	34366.71	7921.77	271	149242.04	297062.76	83.75	3.85	4.34	550.71	0.40	1096.17	1.99
2	2022-01-05	2035075.35	73533.0	35403.04	7801.24	294	163283.37	332567.95	80.23	3.61	4.61	555.39	0.40	1131.18	2.04
3	2022-02-03	1772391.72	83189.0	40817.79	9565.11	458	154478.58	551567.72	87.16	4.69	3.78	337.29	0.55	1204.30	3.57
4	2022-02-04	1993248.30	76264.0	39261.36	8876.08	321	162675.06	363740.12	81.61	3.83	4.14	506.78	0.42	1133.15	2.24
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
87	2022-11-04	1896531.36	85209.0	42928.36	9053.65	354	148153.70	408795.47	78.12	4.49	3.45	418.51	0.42	1154.79	2.76
88	2022-11-05	1767327.14	64587.0	32373.88	8345.79	312	120631.25	326058.81	68.26	3.65	3.73	386.64	0.48	1045.06	2.70
89	2022-12-03	2551220.27	116939.0	56387.92	13712.75	596	175881.62	760761.36	68.94	4.58	3.12	295.10	0.51	1276.45	4.33
90	2022-12-04	1791844.04	81614.0	40346.02	9919.63	350	148855.74	403703.74	83.07	4.55	3.69	425.30	0.43	1153.44	2.71
91	2022-12-05	1995998.04	86546.0	45084.67	10624.58	400	151838.81	485893.09	76.07	4.34	3.37	379.60	0.46	1214.73	3.20

92 rows × 18 columns

In [45]:
```
day_level_analysis.groupby('Month').agg({'Link Clicks':'sum',
                                         'Amount Spent':'sum',
                                         'Revenue':'sum'}).reset_index().sort_values('Revenue').plot(x='Month',
```

Out[45]:
```
<AxesSubplot:xlabel='Month'>
```



In [46]:
```
campaign_level_analysis =  metric_calculator(campaign_level_aggregated_df)
```

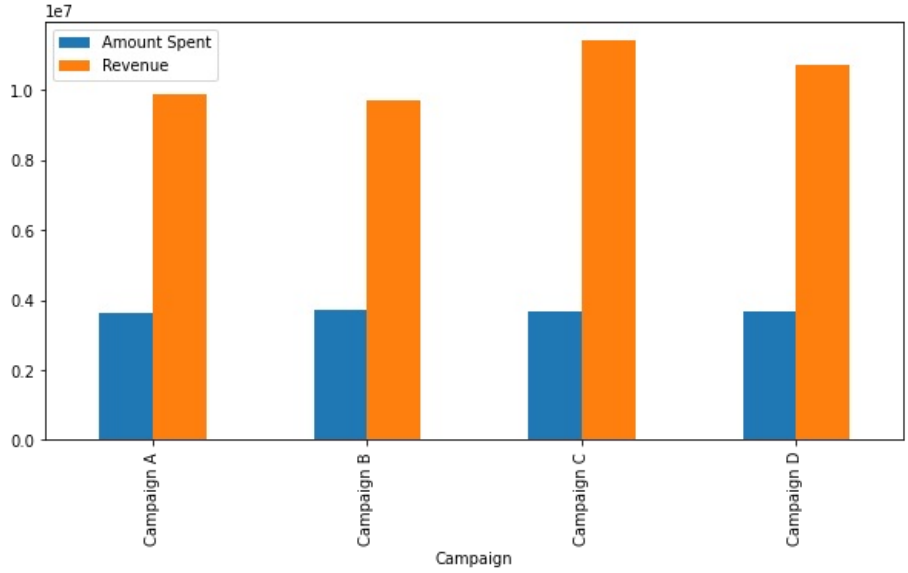
In [47]:
```
campaign_level_analysis
```

Out[47]:

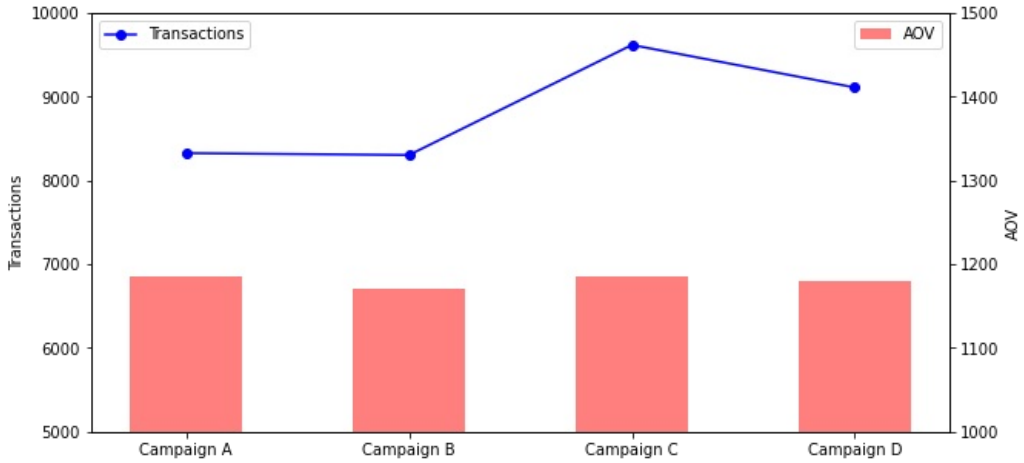
	Campaign	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Amount Spent	Revenue	CPM	CTR (%)	CPS	CPT	CVR (%)	AOV
0	Campaign A	47147434.97	1896524.0	943866.46	222933.87	8326	3628393.80	9868565.01	76.96	4.02	3.84	435.79	0.44	1185.27
1	Campaign B	47135435.18	1806173.0	899325.46	206898.00	8304	3714710.01	9715617.13	78.81	3.83	4.13	447.34	0.46	1169.99
2	Campaign C	47120252.39	2005894.0	991262.39	232836.38	9618	3684723.73	11406955.75	78.20	4.26	3.72	383.11	0.48	1186.00
3	Campaign D	47824367.60	1909754.0	960864.86	223070.08	9110	3664082.55	10740946.92	76.62	3.99	3.81	402.20	0.48	1179.03

In [48]:
```
campaign_level_analysis.plot(x='Campaign', y = ['Amount Spent', 'Revenue'], kind = 'bar', figsize = (10,5))
```

Out[48]: <AxesSubplot:xlabel='Campaign'>

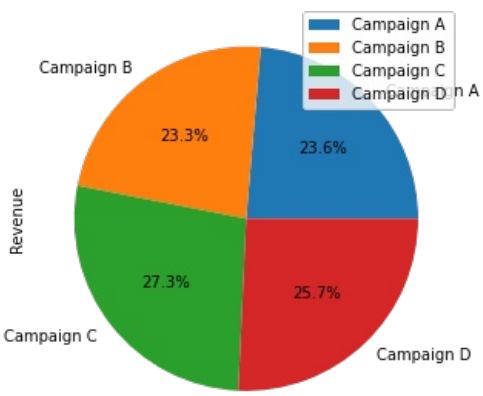


In [49]:
```
fig, ax1 = plt.subplots(figsize=(10,5))
# plot line chart on axis #1
ax1.plot(campaign_level_analysis['Campaign'], campaign_level_analysis['Transactions'], marker = 'o', color= 'bl
ax1.set_ylabel('Transactions')
ax1.set_ylim(5000, 10000)
ax1.legend(['Transactions'], loc="upper left")
# set up the 2nd axis
ax2 = ax1.twinx()
# plot bar chart on axis #2
ax2.bar(campaign_level_analysis['Campaign'],campaign_level_analysis['AOV'] , width=0.5, alpha=0.5, color='red')
ax2.grid(False) # turn off grid #2
ax2.set_ylabel('AOV')
ax2.set_ylim(1000, 1500)
ax2.legend(['AOV'], loc="upper right")
plt.show()
```



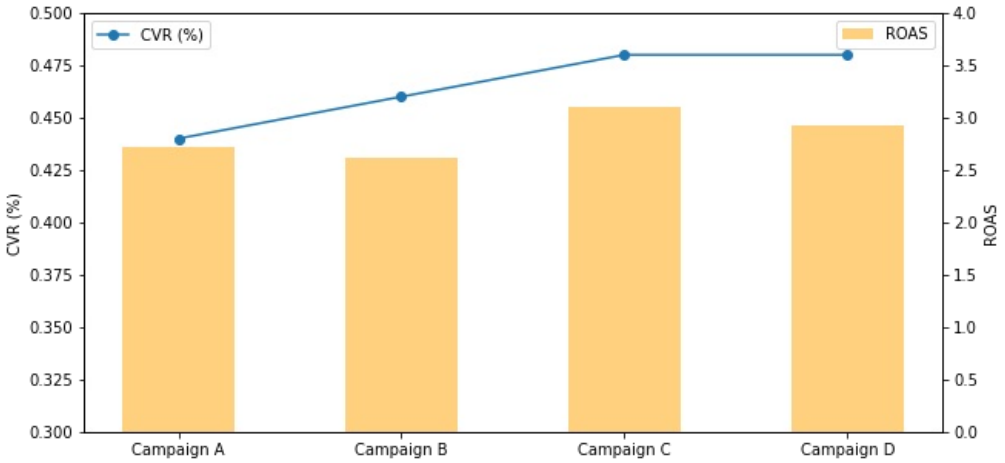
In [50]:
```
# Revenue break up from campaigns
campaign_level_analysis.set_index(keys = 'Campaign').plot.pie(y = 'Revenue', autopct='%1.1f%%', figsize = (5,10
```

Out[50]: <AxesSubplot:ylabel='Revenue'>



In [51]:
```
fig, ax1 = plt.subplots(figsize=(10,5))
# plot line chart on axis #1
```

```
ax1.plot(campaign_level_analysis['Campaign'], campaign_level_analysis['CVR (%)'], marker = 'o')
ax1.set_ylabel('CVR (%)')
ax1.set_ylim(0.3, 0.5)
ax1.legend(['CVR (%)'], loc="upper left")
# set up the 2nd axis
ax2 = ax1.twinx()
# plot bar chart on axis #2
ax2.bar(campaign_level_analysis['Campaign'], campaign_level_analysis['ROAS'], width=0.5, alpha=0.5, color='oran
ax2.grid(False) # turn off grid #2
ax2.set_ylabel('ROAS')
ax2.set_ylim(0, 4)
ax2.legend(['ROAS'], loc="upper right")
plt.show()
```

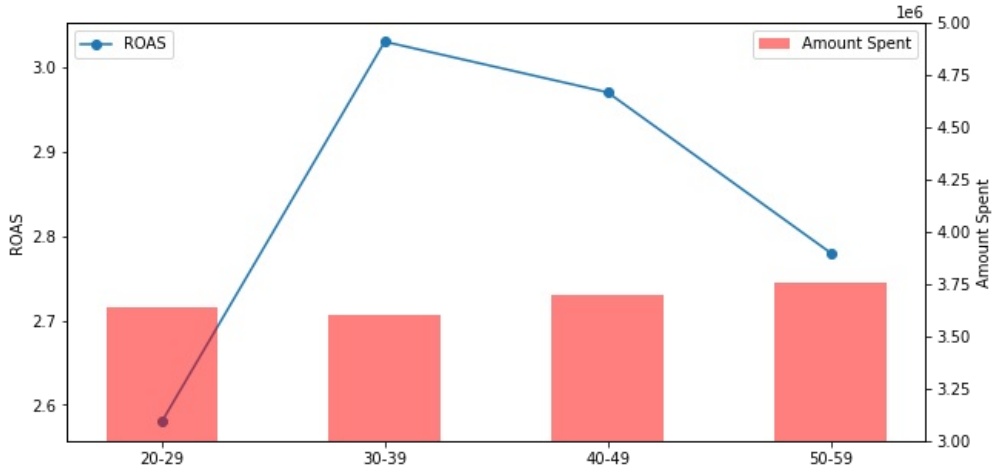


In [52]:
```
age_grp_lvl_analysis = metric_calculator(age_group_level_aggregated_df)
age_grp_lvl_analysis
```

Out[52]:

	Age	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Amount Spent	Revenue	CPM	CTR (%)	CPS	CPT	CVR (%)	AOV	ROA
0	20-29	46194082.80	1815938.0	911814.28	209094.29	7988	3639046.11	9400517.00	78.78	3.93	3.99	455.56	0.44	1176.83	2.5
1	30-39	46848078.66	1911859.0	950305.59	221846.23	9198	3598792.64	10908488.97	76.82	4.08	3.79	391.26	0.48	1185.96	3.0
2	40-49	48566468.32	1916957.0	954171.43	225676.32	9329	3699028.69	10998646.14	76.16	3.95	3.88	396.51	0.49	1178.97	2.9
3	50-59	47618860.36	1973591.0	979027.87	229121.49	8843	3755042.65	10424432.70	78.86	4.14	3.84	424.63	0.45	1178.83	2.7

In [53]:
```
fig, ax1 = plt.subplots(figsize=(10,5))
# plot line chart on axis #1
ax1.plot(age_grp_lvl_analysis['Age'], age_grp_lvl_analysis['ROAS'], marker = 'o')
ax1.set_ylabel('ROAS')
# ax1.set_ylim(0, 3.5)
ax1.legend(['ROAS'], loc="upper left")
# set up the 2nd axis
ax2 = ax1.twinx()
# plot bar chart on axis #2
ax2.bar(age_grp_lvl_analysis['Age'],age_grp_lvl_analysis['Amount Spent'] , width=0.5, alpha=0.5, color='red')
ax2.grid(False) # turn off grid #2
ax2.set_ylabel('Amount Spent')
ax2.set_ylim(3000000, 5000000)
ax2.legend(['Amount Spent'], loc="upper right")
plt.show()
```



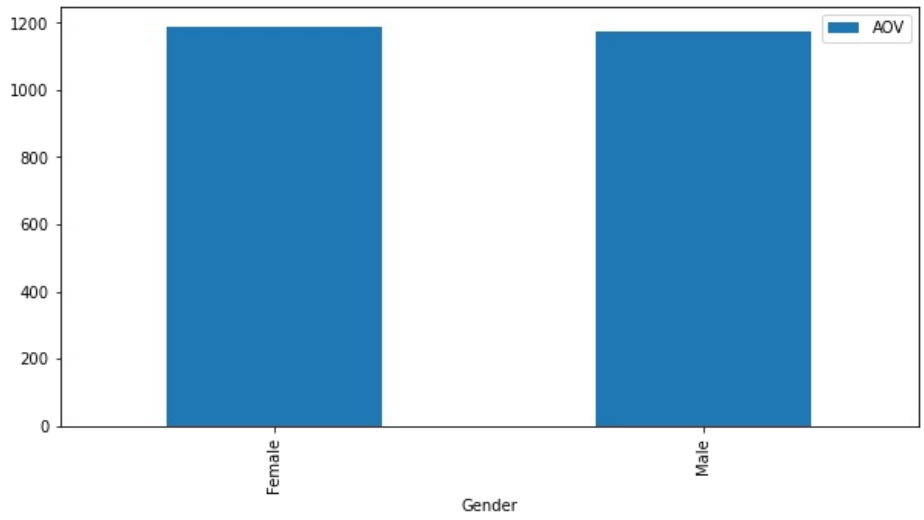
In [54]:
```
gender_lvl_analysis = metric_calculator(gender_level_aggregated_df)
gender_lvl_analysis
```

Out[54]:

	Gender	Impressions	Link Clicks	Sessions	Adds to Cart	Transactions	Amount Spent	Revenue	CPM	CTR (%)	CPS	CPT	CVR (%)	AOV
0	Female	94945526.04	3745542.0	1869486.51	435418.36	17317	7347648.16	20570535.06	77.39	3.94	3.93	424.30	0.46	1187.88
1	Male	94281964.10	3872803.0	1925832.66	450319.97	18041	7344261.93	21161549.75	77.90	4.11	3.81	407.09	0.47	1172.97

In [55]:
```
gender_lvl_analysis.plot(x='Gender', y = 'AOV' , kind = 'bar', figsize = (10,5))
```

Out[55]:
```
<AxesSubplot:xlabel='Gender'>
```



Predictive anaysis

In [56]:
```
## lets take the day level analysis df to make a model to predict the revenues based on given features

## As per use case, let us consider the target variable or dependent variable is the revenue.
## This is clearly a regression problem because the target variable is continuous.
## Since most of the features like impressions, link clicks etc are linearly correlated to the overall revenue,
```

In [66]:
```
# Taking out the dependent variable and splitting data into train and test
day_lvl_data = day_level_analysis[["Impressions", "Link Clicks", "Sessions", "Adds to Cart", "Transactions", "A
y = day_lvl_data['Revenue']
x = day_lvl_data.drop(['Revenue'], axis = 1)
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.1)
```

In [67]:
```
regressor = LinearRegression()
regressor.fit(x_train, y_train)
y_pred = regressor.predict(x_test)
pred_vs_actual = pd.DataFrame({'Actual': y_test.squeeze(), 'Predicted': y_pred.squeeze()})
pred_vs_actual = pred_vs_actual.round(3)
```

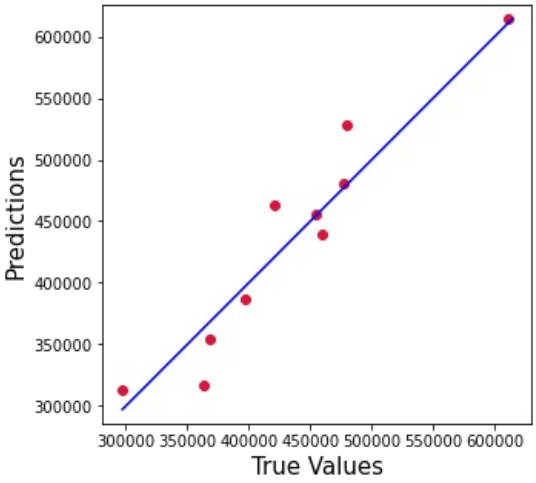
In [68]:
```
pred_vs_actual
```

Out[68]:

	Actual	Predicted
51	421248.38	462711.261
85	459066.70	439000.220
0	611340.95	614390.394
74	453952.52	455651.685
69	363797.31	316460.964
41	367798.02	354397.612
34	477672.64	481088.887
1	297062.76	313493.011
28	480059.40	527621.356
11	397232.45	387337.769

In [69]:
```
plt.figure(figsize=(5,5))
plt.scatter(pred_vs_actual['Actual'], pred_vs_actual['Predicted'], c='crimson')

p1 = max(max(pred_vs_actual['Predicted']), max(pred_vs_actual['Actual']))
p2 = min(min(pred_vs_actual['Predicted']), min(pred_vs_actual['Actual']))
plt.plot([p1, p2], [p1, p2], 'b-')
plt.xlabel('True Values', fontsize=15)
plt.ylabel('Predictions', fontsize=15)
plt.axis('equal')
plt.show()
```



Evaluating the linear regression model using MAPE

MAPE Stands for Mean Absolute percentage Error, which means the percentage of absolute deviation between the actual and predicted values

In [70]:
```
MAPE_linear_regression = mean_absolute_percentage_error(pred_vs_actual['Actual'], pred_vs_actual['Predicted'])
print('MAPE for linear regression model is:', MAPE_linear_regression)
accuracy = round((1-MAPE_linear_regression)*100,2)
print('Model accuracy percentage on test dataset:{}%'.format(accuracy))
```

```
MAPE for linear regression model is: 0.05038678055365429
Model accuracy percentage on test dataset:94.96%
```

Predictive analysis using random forest regression

In [81]:
```
rf_regressor = RandomForestRegressor(n_estimators = 100, random_state = 0)
rf_regressor.fit(x_train, y_train)
y_pred_rf = rf_regressor.predict(x_test)
pred_vs_actual_rf = pd.DataFrame({'Actual': y_test.squeeze(), 'Predicted': y_pred_rf.squeeze()})
pred_vs_actual_rf = pred_vs_actual_rf.round(3)
pred_vs_actual_rf
```

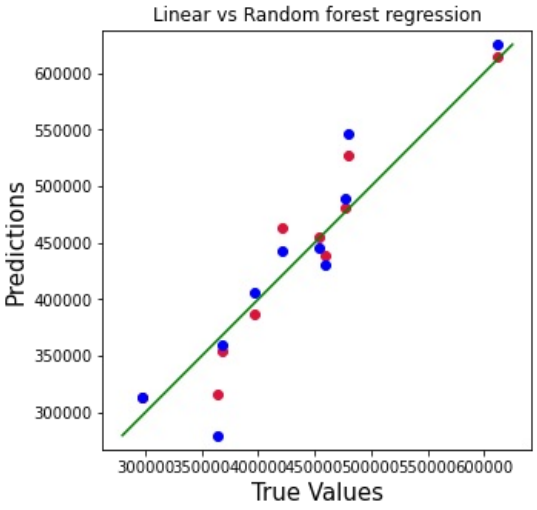
Out[81]:

	Actual	Predicted
51	421248.38	442810.817
85	459066.70	430917.112
0	611340.95	625225.890
74	453952.52	445803.804
69	363797.31	279861.565
41	367798.02	359981.853
34	477672.64	489741.082
1	297062.76	313944.422
28	480059.40	545903.337
11	397232.45	405835.182

In [91]:
```
plt.figure(figsize=(5,5))
plt.scatter(pred_vs_actual['Actual'], pred_vs_actual['Predicted'], c='crimson', label = 'Linear Regression')

p1 = max(max(pred_vs_actual_rf['Predicted']), max(pred_vs_actual_rf['Actual']))
p2 = min(min(pred_vs_actual_rf['Predicted']), min(pred_vs_actual_rf['Actual']))
plt.plot([p3, p4], [p3, p4], 'b-', color= 'green', label = 'RF Regression Line')

plt.scatter(pred_vs_actual_rf['Actual'], pred_vs_actual_rf['Predicted'], c='blue', label = 'Random Forest Regre
plt.xlabel('True Values', fontsize=15)
plt.ylabel('Predictions', fontsize=15)
plt.title('Linear vs Random forest regression')
plt.axis('equal')
plt.show()
```



In [92]:
```
MAPE_rf_regression = mean_absolute_percentage_error(pred_vs_actual_rf['Actual'], pred_vs_actual_rf['Predicted']
print('MAPE for random forest regression model is:', MAPE_lasso_regression)
accuracy_lasso = round((1-MAPE_rf_regression)*100,2)
print('Model accuracy percentage on test dataset:{}%'.format(accuracy_lasso))
```

```
MAPE for random forest regression model is: 0.06460497065400952
Model accuracy percentage on test dataset:93.54%
```

This shows that for the available data linear regression model is giving an accuracy of nearly 95% where as the random forest model gave only 93.5% accuracy. The above scatter plot shows the actual deviation between the data pints across both linear and random forest regression.

In [ ]: