Consumer Behavior and Personality Profiling

AUTHOR: MOHAN KRISHNA KOLA

In this project, we analyze customer demographics, behavior, and purchasing patterns to create detailed profiles. The goal is to provide businesses with actionable insights to improve marketing strategies and boost customer retention.

Imporintg important Libraries

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:
```
# Loading the data
main_df = pd.read_csv('marketing_campaign.csv', sep='\t')
df = main_df.copy()
df.head(10)
```

Out[2]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	
5	7446	1967	Master	Together	62513.0	0	1	09-09-2013	
6	965	1971	Graduation	Divorced	55635.0	0	1	13-11-2012	
7	6177	1985	PhD	Married	33454.0	1	0	08-05-2013	
8	4855	1974	PhD	Together	30351.0	1	0	06-06-2013	
9	5899	1950	PhD	Together	5648.0	1	1	13-03-2014	

10 rows × 29 columns

Exploratory Data Analysis

In [3]:
```
#shape of the dataset
df.shape
```

Out[3]:
```
(2240, 29)
```

In [4]:
```
# basic information of dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 29 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   ID                   2240 non-null   int64  
 1   Year_Birth           2240 non-null   int64  
 2   Education            2240 non-null   object 
 3   Marital_Status       2240 non-null   object 
 4   Income               2216 non-null   float64
 5   Kidhome              2240 non-null   int64  
 6   Teenhome             2240 non-null   int64  
 7   Dt_Customer          2240 non-null   object 
 8   Recency              2240 non-null   int64  
 9   MntWines             2240 non-null   int64  
 10  MntFruits            2240 non-null   int64  
 11  MntMeatProducts      2240 non-null   int64  
 12  MntFishProducts      2240 non-null   int64  
 13  MntSweetProducts     2240 non-null   int64  
 14  MntGoldProds         2240 non-null   int64  
 15  NumDealsPurchases    2240 non-null   int64  
 16  NumWebPurchases      2240 non-null   int64  
 17  NumCatalogPurchases  2240 non-null   int64  
 18  NumStorePurchases    2240 non-null   int64  
 19  NumWebVisitsMonth    2240 non-null   int64  
 20  AcceptedCmp3         2240 non-null   int64  
 21  AcceptedCmp4         2240 non-null   int64  
 22  AcceptedCmp5         2240 non-null   int64  
 23  AcceptedCmp1         2240 non-null   int64  
 24  AcceptedCmp2         2240 non-null   int64  
 25  Complain             2240 non-null   int64  
 26  Z_CostContact        2240 non-null   int64  
 27  Z_Revenue            2240 non-null   int64  
 28  Response             2240 non-null   int64  
dtypes: float64(1), int64(25), object(3)
memory usage: 507.6+ KB
```

- Here we have only 3 object type datatype and rest are numerical

In [5]:
```
# Finding the number of unique values present in each column
df.nunique()
```

Out[5]:
```
ID                   2240
Year_Birth             59
Education               5
Marital_Status          8
Income               1974
Kidhome                 3
Teenhome                3
Dt_Customer           663
Recency               100
MntWines              776
MntFruits             158
MntMeatProducts       558
MntFishProducts       182
MntSweetProducts      177
MntGoldProds          213
NumDealsPurchases      15
NumWebPurchases        15
NumCatalogPurchases    14
NumStorePurchases      14
NumWebVisitsMonth      16
AcceptedCmp3            2
AcceptedCmp4            2
AcceptedCmp5            2
AcceptedCmp1            2
AcceptedCmp2            2
Complain                2
Z_CostContact           1
Z_Revenue               1
Response                2
dtype: int64
```

- In above cell "Z_CostContact" and "Z_Revenue" have some value in all the rows that's why they are not going to contribute anything in the model building. So we can drop them

In [6]:
```
# Checking if ny NaN is present in column or not
df.isna().any()
```

Out[6]:
```
ID                   False
Year_Birth           False
Education            False
Marital_Status       False
Income                True
Kidhome              False
Teenhome             False
Dt_Customer          False
Recency              False
MntWines             False
MntFruits            False
MntMeatProducts      False
MntFishProducts      False
MntSweetProducts     False
MntGoldProds         False
NumDealsPurchases    False
NumWebPurchases      False
NumCatalogPurchases  False
NumStorePurchases    False
NumWebVisitsMonth    False
AcceptedCmp3         False
AcceptedCmp4         False
AcceptedCmp5         False
AcceptedCmp1         False
AcceptedCmp2         False
Complain             False
Z_CostContact        False
Z_Revenue            False
Response             False
dtype: bool
```

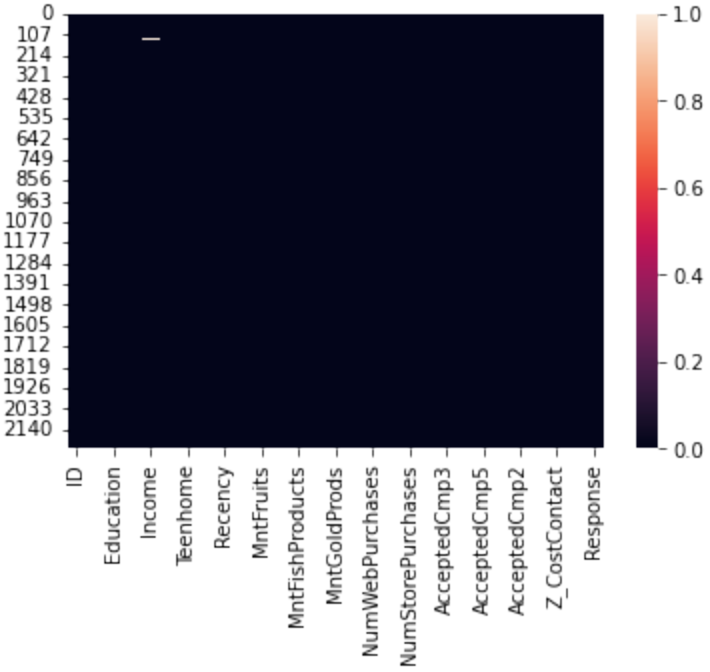
In [7]:
```
# Checking number of null values
df.isnull().sum()
```

Out[7]:
```
ID                     0
Year_Birth             0
Education              0
Marital_Status         0
Income                24
Kidhome                0
Teenhome               0
Dt_Customer            0
Recency                0
MntWines               0
MntFruits              0
MntMeatProducts        0
MntFishProducts        0
MntSweetProducts       0
MntGoldProds           0
NumDealsPurchases      0
NumWebPurchases        0
NumCatalogPurchases    0
NumStorePurchases      0
NumWebVisitsMonth      0
AcceptedCmp3           0
AcceptedCmp4           0
AcceptedCmp5           0
AcceptedCmp1           0
AcceptedCmp2           0
Complain               0
Z_CostContact          0
Z_Revenue              0
Response               0
dtype: int64
```

- Income column have some missing value in it so we will need to fill it by either maean or median.

In [8]:
```
# Checking for null value using heatmap
sns.heatmap(df.isnull())
```

Out[8]: <AxesSubplot:>



In [9]:
```
# Dropping the column beacause they will not contribute in model building
df = df.drop(columns=["Z_CostContact", "Z_Revenue"], axis=1)
df.head(10)
```

Out[9]:

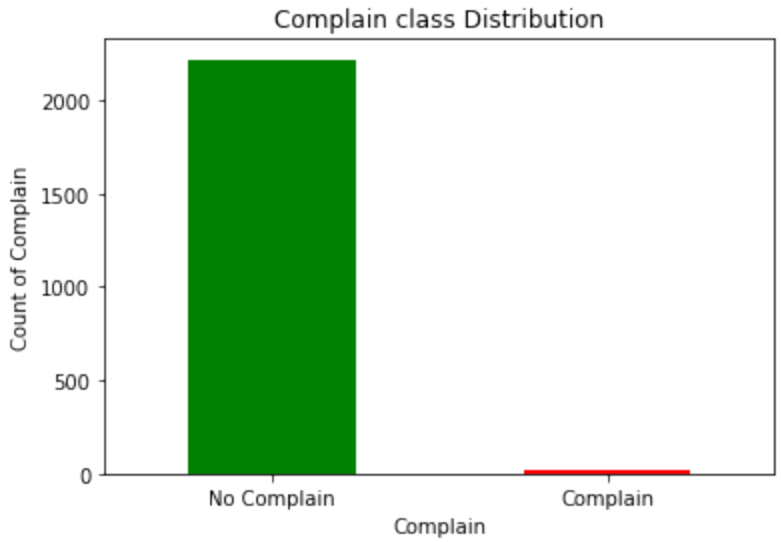
	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	
5	7446	1967	Master	Together	62513.0	0	1	09-09-2013	
6	965	1971	Graduation	Divorced	55635.0	0	1	13-11-2012	
7	6177	1985	PhD	Married	33454.0	1	0	08-05-2013	
8	4855	1974	PhD	Together	30351.0	1	0	06-06-2013	
9	5899	1950	PhD	Together	5648.0	1	1	13-03-2014	

10 rows × 27 columns

- Let's figure out the number of complains complained by customer and number of responses are positive or negative in last 2 years.

In [10]:
```
# Complain: 1 if customer complained in the last 2 years, 0 otherwise
label_complain = ["No Complain","Complain"]

count_complain = pd.value_counts(df['Complain'], sort=True)
count_complain.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("Complain class Distribution")
plt.xticks(range(2),label_complain)
plt.xlabel("Complain")
plt.ylabel("Count of Complain")
```

Out[10]:
```
Text(0, 0.5, 'Count of Complain')
```



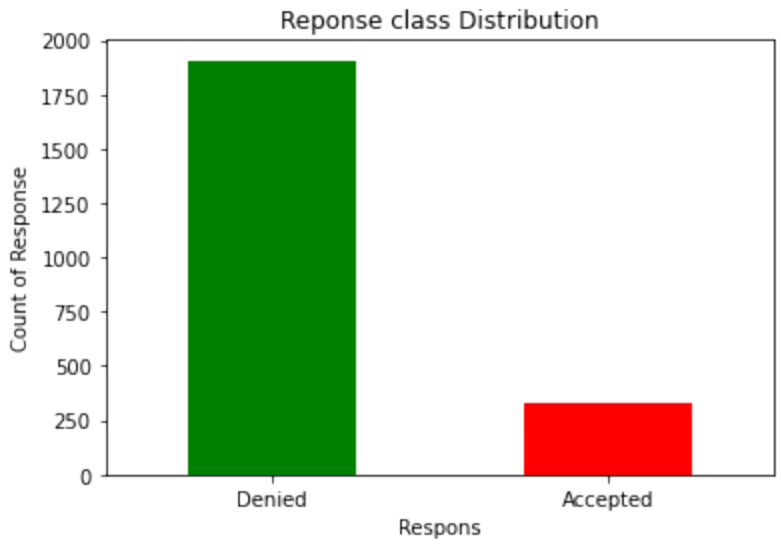
In [11]:
```
df['Complain'].value_counts()
# 1 if customer complained in the last 2 years, 0 otherwise
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[11]:
```
0    2219
1      21
Name: Complain, dtype: int64
```

- From above image we can say that there are not much complains by customers.

In [12]:
```
# Let's checl about response

# Response: 1 if customer accepted the offer in the last 2 campaign, 0 otherwis
label_response = ["Denied","Accepted"]

count_response = pd.value_counts(df['Response'], sort=True)
count_response.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("Reponse class Distribution")
plt.xticks(range(2),label_response)
plt.xlabel("Respons")
plt.ylabel("Count of Response")
```

Out[12]:
```
Text(0, 0.5, 'Count of Response')
```



In [13]:
```
df['Response'].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[13]:
```
0    1906
1     334
Name: Response, dtype: int64
```

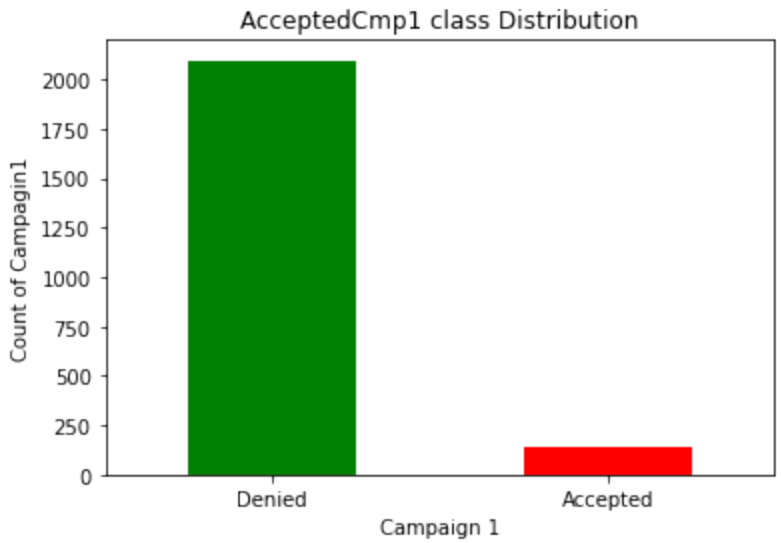
- This graph shows that last the offer has been denied by most of the customers

Let's check out all campaign offers

- AcceptedCmp1: 1 if customer accepted the offer in the 1st campaign, 0 otherwise
- AcceptedCmp2: 1 if customer accepted the offer in the 2nd campaign, 0 otherwise
- AcceptedCmp3: 1 if customer accepted the offer in the 3rd campaign, 0 otherwise
- AcceptedCmp4: 1 if customer accepted the offer in the 4th campaign, 0 otherwise
- AcceptedCmp5: 1 if customer accepted the offer in the 5th campaign, 0 otherwise
- Response: 1 if customer accepted the offer in the last campaign, 0 otherwise

In [14]:
```
#Campagin 1

labels_c1 = ["Denied", "Accepted"]

count_c1 = pd.value_counts(df['AcceptedCmp1'], sort=True)
count_c1.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("AcceptedCmp1 class Distribution")
plt.xticks(range(2),labels_c1)
plt.xlabel("Campaign 1")
plt.ylabel("Count of Campagin1")
```

Out[14]:
```
Text(0, 0.5, 'Count of Campagin1')
```

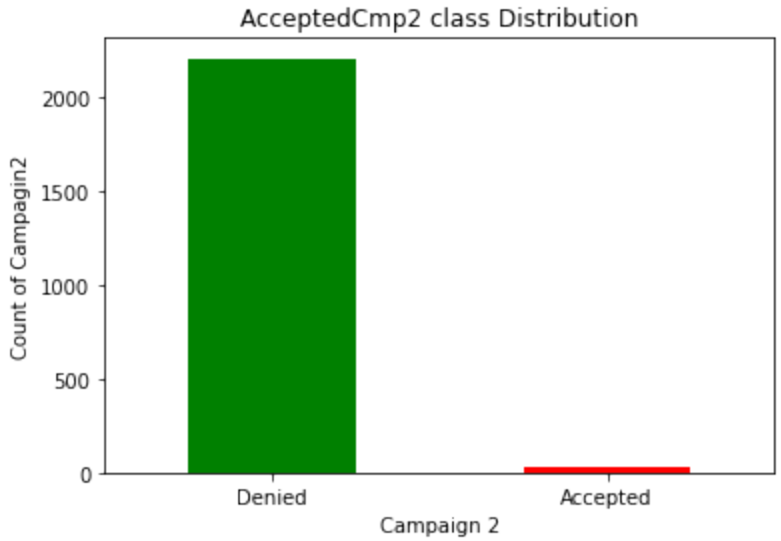


In [15]:
```
df['AcceptedCmp1'].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[15]:
```
0    2096
1     144
Name: AcceptedCmp1, dtype: int64
```

In [16]:
```
#Campagin 2

labels_c2 = ["Denied", "Accepted"]

count_c2 = pd.value_counts(df['AcceptedCmp2'], sort=True)
count_c2.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("AcceptedCmp2 class Distribution")
plt.xticks(range(2),labels_c2)
plt.xlabel("Campaign 2")
plt.ylabel("Count of Campagin2")
```

Out[16]:
```
Text(0, 0.5, 'Count of Campagin2')
```

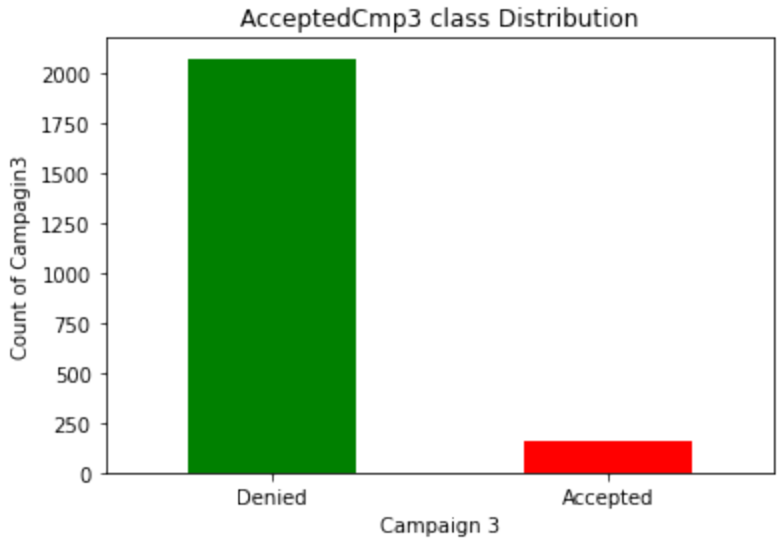


In [17]:
```
df["AcceptedCmp2"].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[17]:
```
0    2210
1      30
Name: AcceptedCmp2, dtype: int64
```

In [18]:
```
#Campagin 3

labels_c3 = ["Denied", "Accepted"]

count_c3 = pd.value_counts(df['AcceptedCmp3'], sort=True)
count_c3.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("AcceptedCmp3 class Distribution")
plt.xticks(range(2),labels_c3)
plt.xlabel("Campaign 3")
plt.ylabel("Count of Campagin3")
```

Out[18]:
```
Text(0, 0.5, 'Count of Campagin3')
```

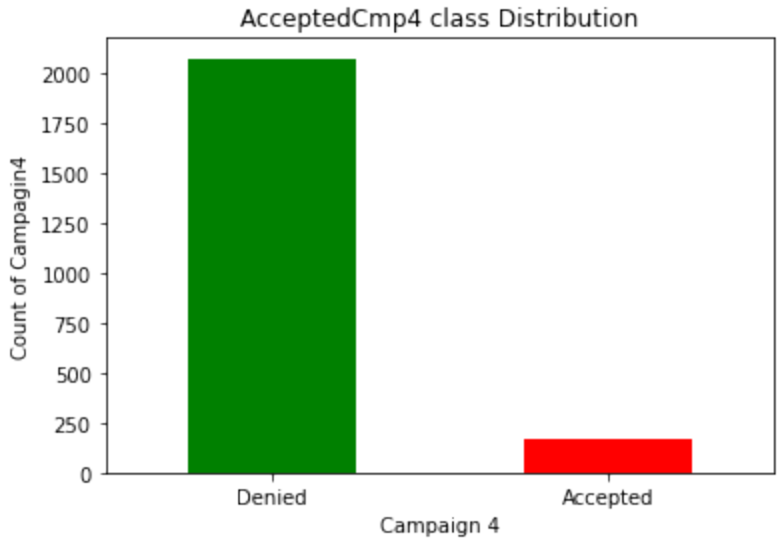


In [19]:
```
df["AcceptedCmp3"].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[19]:
```
0    2077
1     163
Name: AcceptedCmp3, dtype: int64
```

In [20]:
```
#Campagin 4

labels_c4 = ["Denied", "Accepted"]

count_c4 = pd.value_counts(df['AcceptedCmp4'], sort=True)
count_c4.plot(kind='bar', rot=0,color=['Green','Red'])
plt.title("AcceptedCmp4 class Distribution")
plt.xticks(range(2),labels_c4)
plt.xlabel("Campaign 4")
plt.ylabel("Count of Campagin4")
```

Out[20]:
```
Text(0, 0.5, 'Count of Campagin4')
```

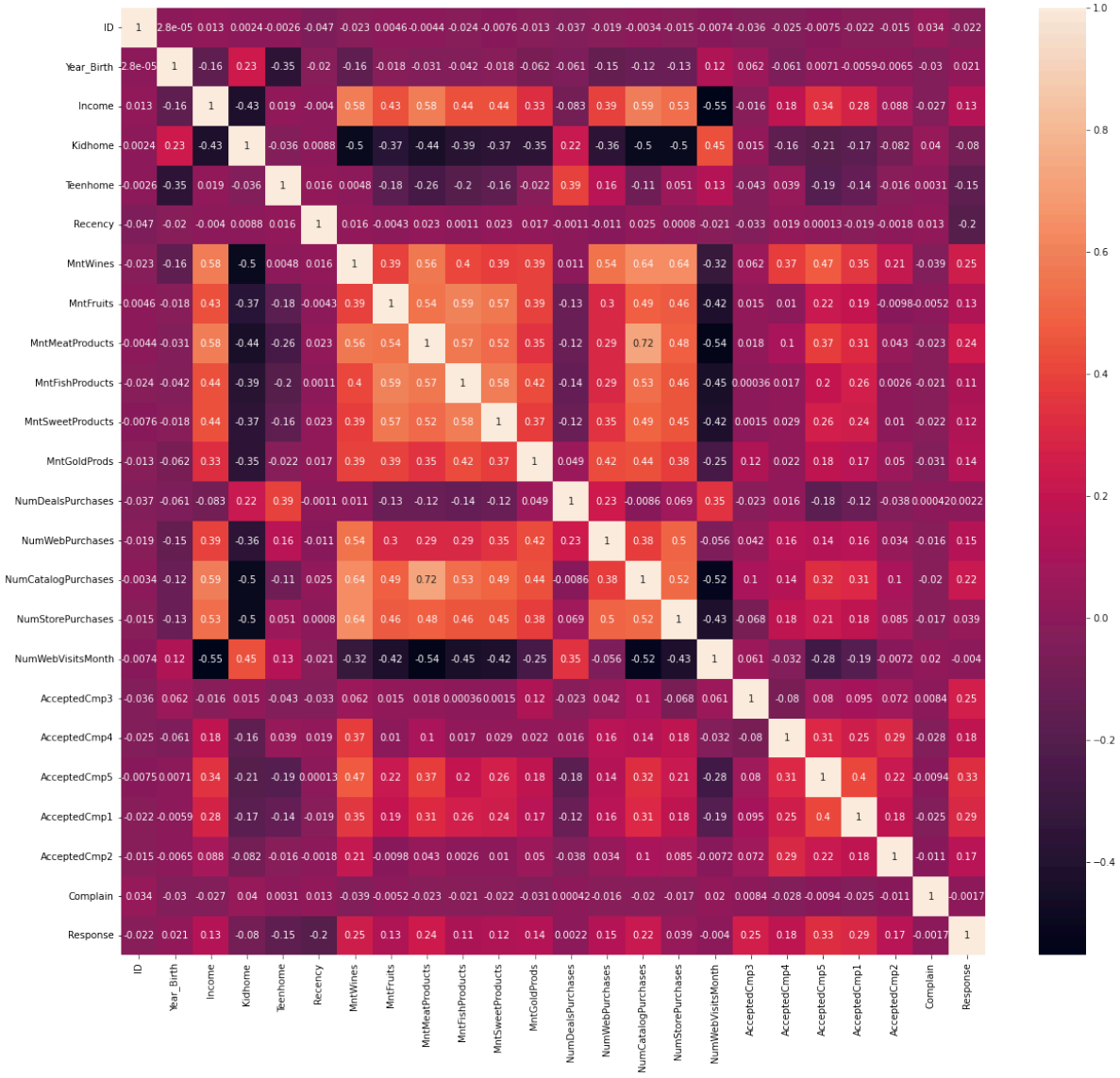


In [21]:
```
df["AcceptedCmp4"].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[21]:
```
0    2073
1     167
Name: AcceptedCmp4, dtype: int64
```

- From the above figures we clearly see that most of offers has been denied by customers in all campagins.
- But Campaign 4 had better amount of acceptance.
- Campaign 4 > Campaign 3 > Campaign 1 > Campagin 2: comparison of acceptance in campaigns.

In [22]:
```
#Finding the correlation between the feature column

plt.figure(figsize=(20,18))
sns.heatmap(df.corr(), annot=True)
plt.show()
```



- No two columns are too much correlated with each other so we can't drop any columns

Data Preprocssing

In [23]:
```
# Filling the missing value in the income by mean
df['Income'] = df['Income'].fillna(df['Income'].mean())
df.isnull().sum()
```

Out[23]:
```
ID                   0
Year_Birth           0
Education            0
Marital_Status       0
Income               0
Kidhome              0
Teenhome             0
Dt_Customer          0
Recency              0
MntWines             0
MntFruits            0
MntMeatProducts      0
MntFishProducts      0
MntSweetProducts     0
MntGoldProds         0
NumDealsPurchases    0
NumWebPurchases      0
NumCatalogPurchases  0
NumStorePurchases    0
NumWebVisitsMonth    0
AcceptedCmp3         0
AcceptedCmp4         0
AcceptedCmp5         0
AcceptedCmp1         0
AcceptedCmp2         0
Complain             0
Response             0
dtype: int64
```

- No null value in the dataset

In [24]:
```
df.head()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[24]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	

5 rows × 27 columns

In [25]:
```
#Checking the number of unique categories present in the "Marital_Status"

df['Marital_Status'].value_counts()
```

Out[25]:
```
Married     864
Together    580
Single      480
Divorced    232
Widow        77
Alone         3
Absurd        2
YOLO          2
Name: Marital_Status, dtype: int64
```

In [26]:
```
df['Marital_Status'] = df['Marital_Status'].replace(['Married', 'Together'],'r
df['Marital_Status'] = df['Marital_Status'].replace(['Divorced', 'Widow', 'Alo
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

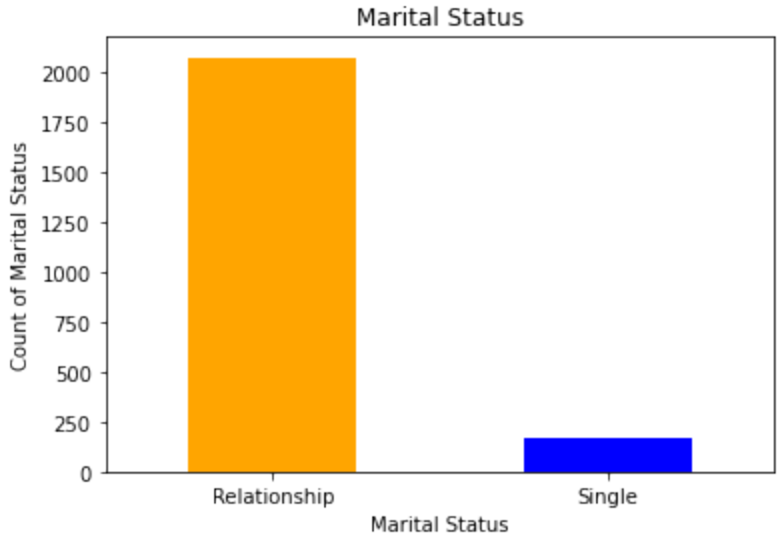
- In the above cell we are grouping 'Married', 'Together' as "relationship"
- Whereas 'Divorced', 'Widow', 'Alone', 'YOLO', 'Absurd' as "Single"

In [27]:
```
df['Marital_Status'].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[27]:
```
relationship    1444
Single           796
Name: Marital_Status, dtype: int64
```

In [28]:
```
# Relationship vs Single

labels_status = ["Relationship", "Single"]

count_status = pd.value_counts(df['Marital_Status'], sort=True)
count_c4.plot(kind='bar', rot=0,color=['Orange','Blue'])
plt.title("Marital Status")
plt.xticks(range(2),labels_status)
plt.xlabel("Marital Status")
plt.ylabel("Count of Marital Status")
```

Out[28]:
```
Text(0, 0.5, 'Count of Marital Status')
```



In [29]:
```
# Combining different dataframes into a single column to reduce the number of c
```

In [30]:
```
df['Kids'] = df['Kidhome'] + df['Teenhome']
df['Expenses'] = df['MntWines'] + df['MntFruits'] + df['MntMeatProducts'] + df
df['TotalAcceptedCmp'] = df['AcceptedCmp1'] + df['AcceptedCmp2'] + df['Accepte
df['NumTotalPurchases'] = df['NumWebPurchases'] + df['NumCatalogPurchases'] + 
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [31]:
```
#saving the data for tableau
df.to_csv('data_visuals.csv')
```

In [32]:
```
# Deleting some column to reduce dimension and complexity of model

col_del = ["AcceptedCmp1" , "AcceptedCmp2", "AcceptedCmp3" , "AcceptedCmp4","A
df=df.drop(columns=col_del,axis=1)
df.head()
```

Out[32]:

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	Complain	Kids
0	5524	1957	Graduation	Single	58138.0	04-09-2012	58	0	0
1	2174	1954	Graduation	Single	46344.0	08-03-2014	38	0	2
2	4141	1965	Graduation	relationship	71613.0	21-08-2013	26	0	0
3	6182	1984	Graduation	relationship	26646.0	10-02-2014	26	0	1
4	5324	1981	PhD	relationship	58293.0	19-01-2014	94	0	1

In [33]:
```
# Adding 'Age' column

df['Age'] = 2015 - df['Year_Birth']
```

In [34]:
```
df['Education'].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[34]:
```
Graduation    1127
PhD            486
Master         370
2n Cycle       203
Basic           54
Name: Education, dtype: int64
```

In [35]:
```
# Changing category into UG and PG only

df['Education'] = df['Education'].replace(['PhD','2n Cycle','Graduation', 'Mas
df['Education'] = df['Education'].replace(['Basic'], 'UG')
```

In [36]:
```
# Number of days a customer was engaged with company

# Changing bt_customer into timestamp format
df['Dt_Customer'] = pd.to_datetime(df.Dt_Customer)
df['first_day'] = '01-01-2015'
df['first_day'] = pd.to_datetime(df.first_day)
df['day_engaged'] = (df['first_day'] - df['Dt_Customer']).dt.days
```

In [37]:
```
df=df.drop(columns=["ID", "Dt_Customer", "first_day", "Year_Birth", "Dt_Custom
df.shape
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[37]: (2240, 9)

In [38]:
```
df.head()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[38]:

	Education	Marital_Status	Income	Kids	Expenses	TotalAcceptedCmp	NumTotalPurchases	A
0	PG	Single	58138.0	0	1617	1	25	
1	PG	Single	46344.0	2	27	0	6	
2	PG	relationship	71613.0	0	776	0	21	
3	PG	relationship	26646.0	1	53	0	8	
4	PG	relationship	58293.0	1	422	0	19	

Data Visualization

In [39]:
```
fig = px.bar(df, x='Marital_Status', y='Expenses', color='Education')
fig.show()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [40]:
```
fig = px.bar(df, x='Marital_Status', y='Expenses', color="Marital_Status")
fig.show()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [41]:
```
# Less number of single customer
fig = px.histogram (df, x = "Expenses",  facet_row = "Marital_Status",  templa
fig.show ()
```

In [42]:
```
fig = px.histogram (df, x = "Expenses",  facet_row = "Education",  template =
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [43]:
```
fig = px.histogram (df, x = "NumTotalPurchases",  facet_row = "Education",  te
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [44]:
```
fig = px.histogram (df, x = "Age",  facet_row = "Marital_Status",  template =
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [45]:
```
fig = px.histogram (df, x = "Income",  facet_row = "Marital_Status",  template
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [46]:
```
fig =  px.pie (df, names = "Marital_Status", hole = 0.4, template = "gridon")
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

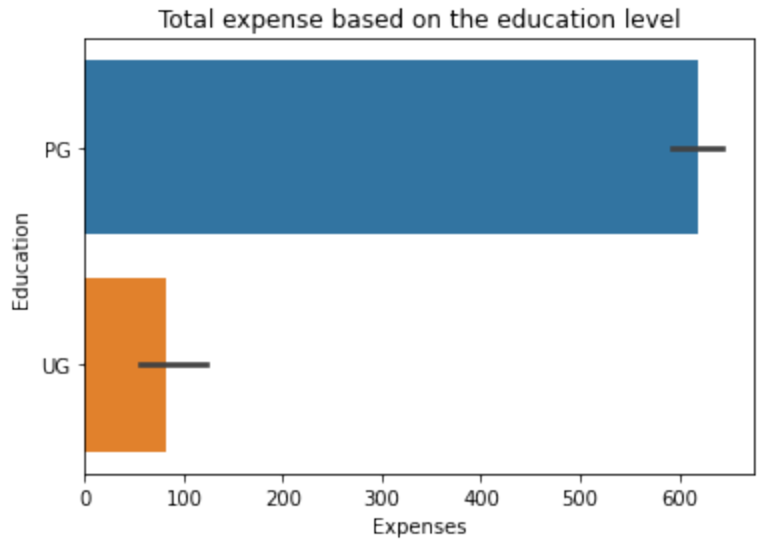
- 35% of the customer are single whereas more 64% are in relationship.

In [47]:
```
fig =  px.pie (df, names = "Education", hole = 0.4, template = "plotly_dark")
fig.show ()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

- More than 97% customer are from PG background. and Approx. 2% are from UG.

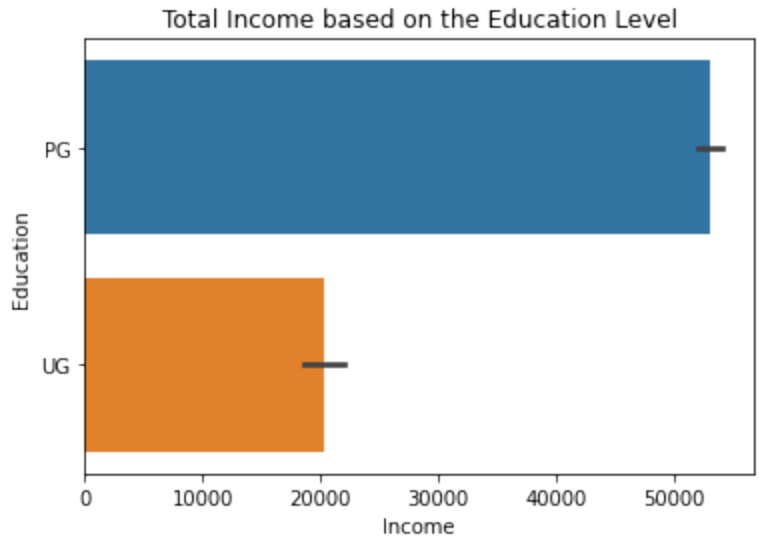
In [48]:
```
sns.barplot(x=df['Expenses'], y=df['Education'])
plt.title('Total expense based on the education level')
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[48]:
```
Text(0.5, 1.0, 'Total expense based on the education level')
```



In [49]:
```
sns.barplot(x=df['Income'], y=df['Education'])
plt.title('Total Income based on the Education Level')
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[49]:
```
Text(0.5, 1.0, 'Total Income based on the Education Level')
```



In [50]:
```
df.describe()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[50]:

	Income	Kids	Expenses	TotalAcceptedCmp	NumTotalPurchases	
count	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.00
mean	52247.251354	0.950446	605.798214	0.446875	14.862054	46.19
std	25037.797168	0.751803	602.249288	0.890543	7.677173	11.98
min	1730.000000	0.000000	5.000000	0.000000	0.000000	19.00
25%	35538.750000	0.000000	68.750000	0.000000	8.000000	38.00
50%	51741.500000	1.000000	396.000000	0.000000	15.000000	45.00
75%	68289.750000	1.000000	1045.500000	1.000000	21.000000	56.00
max	666666.000000	3.000000	2525.000000	5.000000	44.000000	122.00

In [51]:
```
sns.heatmap(df.corr(),annot=True)
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[51]: <AxesSubplot:>

In [52]:
```
obj = []
for i in df.columns:
    if(df[i].dtypes=="object"):
        obj.append(i)

print(obj)
```

```
['Education', 'Marital_Status']
```

In [53]:
```
# Label Encoding
from sklearn.preprocessing import LabelEncoder
```

In [54]:
```
df['Marital_Status'].value_counts()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[54]:
```
relationship    1444
Single           796
Name: Marital_Status, dtype: int64
```

In [55]:
```
lbl_encode = LabelEncoder()
for i in obj:
    df[i] = df[[i]].apply(lbl_encode.fit_transform)
```

In [56]:
```
df1 = df.copy()
df1.head()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[56]:

	Education	Marital_Status	Income	Kids	Expenses	TotalAcceptedCmp	NumTotalPurchases	A
0	0	0	58138.0	0	1617	1	25	
1	0	0	46344.0	2	27	0	6	
2	0	1	71613.0	0	776	0	21	
3	0	1	26646.0	1	53	0	8	
4	0	1	58293.0	1	422	0	19	

Standardization

In [57]:
```
from sklearn.preprocessing import StandardScaler
```

In [58]:
```
scaled_features = StandardScaler().fit_transform(df1.values)
scaled_features_df = pd.DataFrame(scaled_features, index=df1.index, columns=df
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

In [59]:
```
scaled_features_df.head()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[59]:

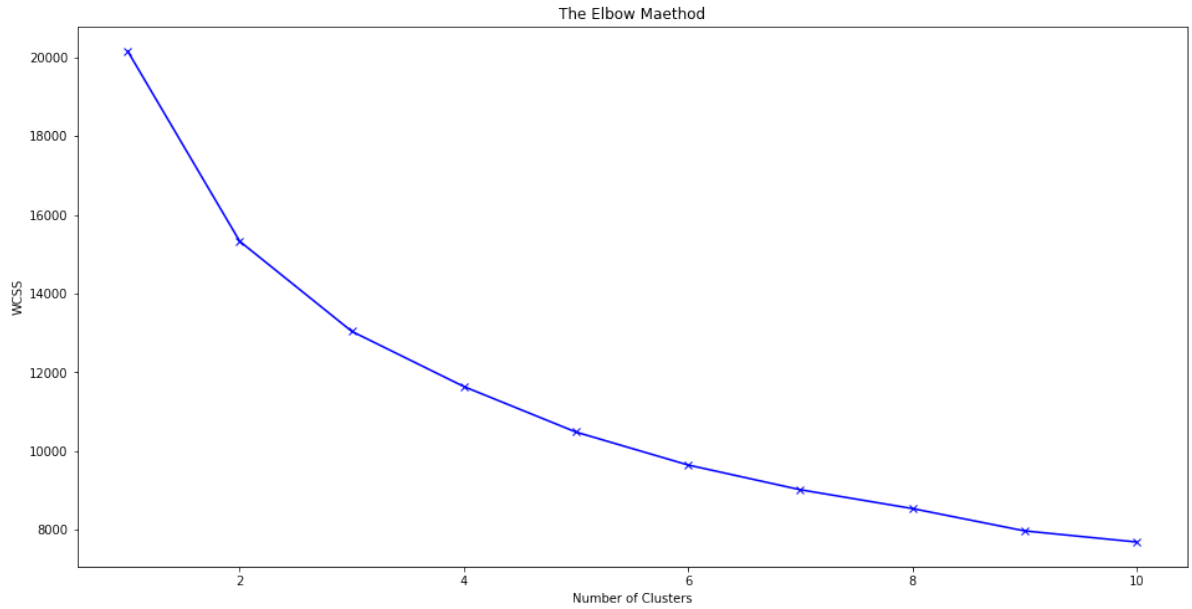
	Education	Marital_Status	Income	Kids	Expenses	TotalAcceptedCmp	NumTotalPurcha
0	-0.157171	-1.346874	0.235327	-1.264505	1.679417	0.621248	1.320
1	-0.157171	-1.346874	-0.235826	1.396361	-0.961275	-0.501912	-1.154
2	-0.157171	0.742460	0.773633	-1.264505	0.282673	-0.501912	0.799
3	-0.157171	0.742460	-1.022732	0.065928	-0.918094	-0.501912	-0.894
4	-0.157171	0.742460	0.241519	0.065928	-0.305254	-0.501912	0.539

In [60]:
```
# scaled_features_df.describe()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Elbow Method

In [61]:
```
from sklearn.cluster import KMeans
```

In [62]:
```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(scaled_features_df)
    wcss.append(kmeans.inertia_)
    # inetia_: Sum of squared distances of samples to their closest cluster cer
plt.figure(figsize=(16,8))
plt.plot(range(1,11), wcss, 'bx-')
plt.title('The Elbow Maethod')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```

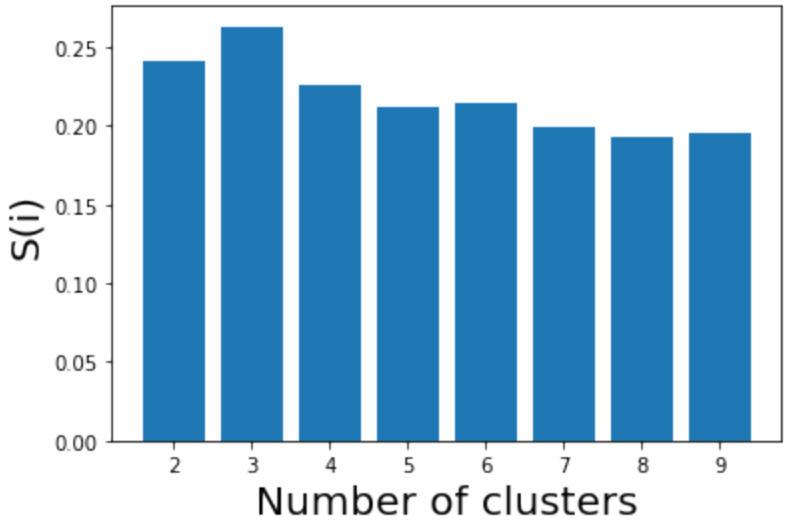


As it is not very clear from the elbow method that which value of K to choose

- Silhouette Score

In [63]:
```
from sklearn.metrics import silhouette_score
```

In [64]:
```
silhouette_scores = []
for i in range(2,10):
    m1 = KMeans(n_clusters=i, random_state=42)
    c = m1.fit_predict(scaled_features_df)
    silhouette_scores.append(silhouette_score(scaled_features_df, m1.fit_predi

plt.bar(range(2,10), silhouette_scores)
plt.xlabel('Number of clusters', fontsize=20)
plt.ylabel('S(i)', fontsize=20)
plt.show()
```



In [65]:
```
# Now we are using Silhouette score to measure the value of K
silhouette_scores
```

Out[65]:
```
[0.24145101432627075,
 0.2630066765900862,
 0.22547869857815794,
 0.2112495373878677,
 0.2149228429852001,
 0.1997135405176978,
 0.19301680336746188,
 0.19495794809915995]
```

In [66]:
```
# Getting the maximum value of silhouette score and adding 2 in index beacure

sc = max(silhouette_scores)
num_of_clusters = silhouette_scores.index(sc)+2
print("Number of Cluster Required is: ", num_of_clusters)
```

```
Number of Cluster Required is:  3
```

Model Building

In [67]:
```
# Training a prediction using K-Means Algorithm.

kmeans = KMeans(n_clusters = num_of_clusters, random_state=42).fit(scaled_featu
pred = kmeans.predict(scaled_features_df)
```

In [68]:
```
pred
```

Out[68]:
```
array([1, 0, 1, ..., 1, 1, 0])
```

In [69]:
```
# Appending those cluster value into the main dataframe (without standardizati
df['cluster'] = pred + 1
```

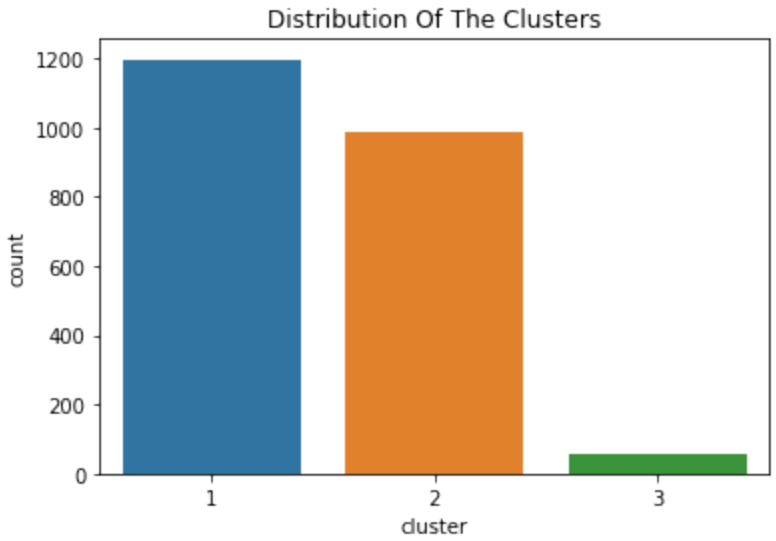
In [70]:
```
df.head()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```

Out[70]:

	Education	Marital_Status	Income	Kids	Expenses	TotalAcceptedCmp	NumTotalPurchases	A
0	0	0	58138.0	0	1617	1	25	
1	0	0	46344.0	2	27	0	6	
2	0	1	71613.0	0	776	0	21	
3	0	1	26646.0	1	53	0	8	
4	0	1	58293.0	1	422	0	19	

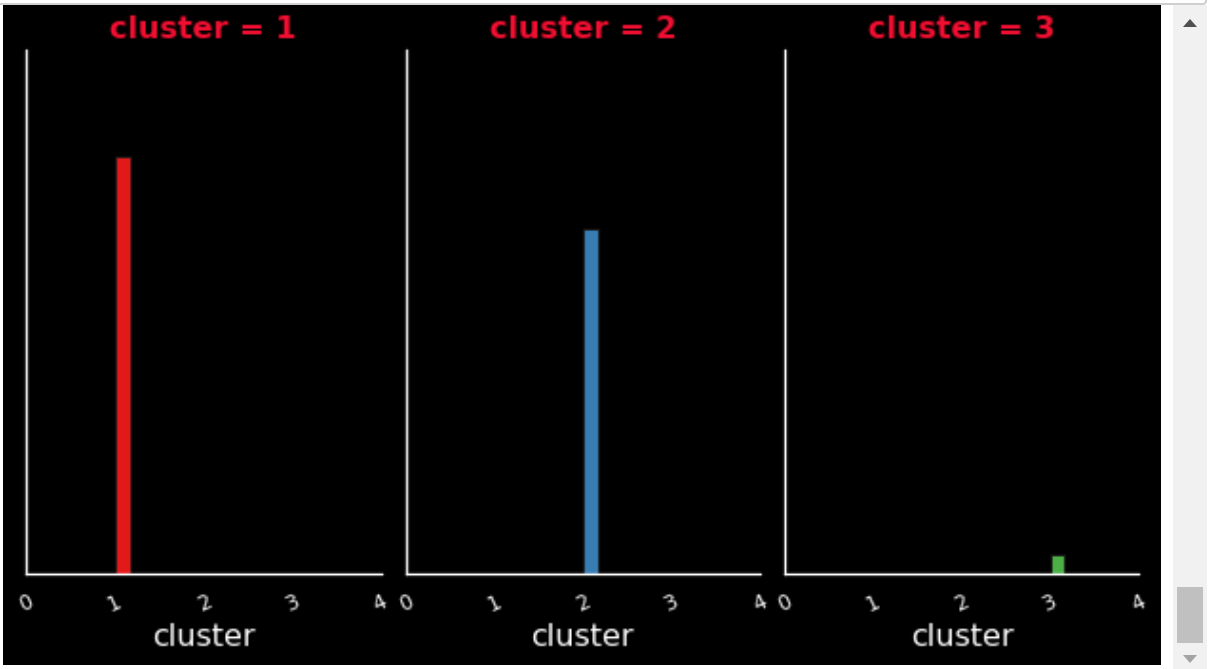
In [71]:
```
# saving clustering csv for Tableau
df.to_csv('data_visuals2.csv')
```

In [72]:
```
pl = sns.countplot(x=df["cluster"])
pl.set_title("Distribution Of The Clusters")
plt.show()
# Visualizing distribution of Age
df['Age'] = 2021 - df['Year_Birth']
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=20, kde=True, color='purple')
plt.title('Distribution of Customer Age')
plt.show()

# Distribution of Income
plt.figure(figsize=(10,6))
sns.histplot(df['Income'], bins=30, kde=True, color='blue')
plt.title('Income Distribution')
plt.show()

# Marital Status Distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='Marital_Status', palette='Set2')
plt.title('Marital Status Distribution')
plt.show()
```



As we can see here that weightage of customer are more in cluster 1 as compare to other.

In [73]:
```
sns.set(rc={'axes.facecolor':'black', 'figure.facecolor':'black', 'axes.grid'

for i in df:
    diag = sns.FacetGrid(df, col = "cluster", hue = "cluster", palette = "Set1
    diag.map(plt.hist, i, bins=6, ec="k")
    diag.set_xticklabels(rotation=25, color = 'white')
    diag.set_yticklabels(color = 'white')
    diag.set_xlabels(size=16, color = 'white')
    diag.set_titles(size=16, color = '#f01132', fontweight="bold")
    diag.fig.set_figheight(6)
```



Report

Based on above information we can divide customer into 3 parts:-

1. **Highly Active Customer**: These customers beloing to cluster one.
2. **Moderately Active Customer** :- These customers belong to cluster two.
3. **Least Active Customer** :- These customers belong to cluster third.

Characteristics of Highly Active Customer

- **In terms of Education**
  - Highly Active Customer are from PG background
- **In terms of Marital_status**
  - Number of people in relationship are approx. two times of single people
- **In terms of Income**
  - Income of Highly active customer are little less as compare to Moderately active customer.
- **In terms of Kids**

    - Highly active customer have more number of children as compare to other customer (avg. of 1 child).
- **In terms of Expenses**
    - Expenses of Highly Active customer are less as compare to moderate.
    - These customer spent avg. of approx. 100-200 unit money.
- **In terms of Age**
    - Age of these customer are between 25 to 75.
    - Maximum customer age are between 40 to 50.
- **In terms of day_engaged**
    - Highly Active customer are more loyal as they engaged with company for longer period of time.

Characteristics of Moderately Active Customer

- **In terms of Education**
    - Moderately Active Customer are also from PG backgroud
- **In terms of Marital_status**
    - Number of people in relationship are slightly more as compare to single people.
- **In terms of Income**
    - Income of Moderately active customer are higher as compare to other customer.
- **In terms of Kids**
    - Moderately active customer have less number of childern as compare to highly active customer (Max. customer has no child).
- **In terms of Expenses**
    - Expenses of Moderately Active customer are more as compare to Active.
    - These customer spent avg. of approx. 500-2000 unit money.
- **In terms of Age**
    - Age of these customer are between 25 to 75.
    - Maximum customer age are between 35 to 60.
- **In terms of day_engaged**
    - Moderately Active customer are slightly less engaged with company as compare to Highly Active Customer.

In [ ]: