About Dataset

Problem Statement

Customer Personality Analysis is a detailed analysis of a company’s ideal customers. It helps a business to better understand its customers and makes it easier for them to modify products according to the specific needs, behaviors and concerns of different types of customers.

Customer personality analysis helps a business to modify its product based on its target customers from different types of customer segments. For example, instead of spending money to market a new product to every customer in the company’s database, a company can analyze which customer segment is most likely to buy the product and then market the product only on that particular segment.

Column Information

People

ID: Customer's unique identifier

Year_Birth: Customer's birth year

Education: Customer's education level

Marital_Status: Customer's marital status

Income: Customer's yearly household income

Kidhome: Number of children in customer's household

Teenhome: Number of teenagers in customer's household

Dt_Customer: Date of customer's enrollment with the company

Recency: Number of days since customer's last purchase

Complain: 1 if the customer complained in the last 2 years, 0 otherwise

Products

MntWines: Amount spent on wine in last 2 years

MntFruits: Amount spent on fruits in last 2 years

MntMeatProducts: Amount spent on meat in last 2 years

MntFishProducts: Amount spent on fish in last 2 years

MntSweetProducts: Amount spent on sweets in last 2 years

MntGoldProds: Amount spent on gold in last 2 years

Promotion

NumDealsPurchases: Number of purchases made with a discount

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

Place

NumWebPurchases: Number of purchases made through the company's website

NumCatalogPurchases: Number of purchases made using a catalogue

NumStorePurchases: Number of purchases made directly in stores

NumWebVisitsMonth: Number of visits to company's website in the last month

Target

Need to perform clustering to summarize customer segments.

```
import datetime
from datetime import date

import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt

import warnings
warnings.filterwarnings("ignore")

#READ THE DATASET...
df = pd.read_csv("/content/sample_data/marketing_campaign.csv", sep="\
t")

df.head()
```

{"type":"dataframe","variable_name":"df"}

```
df.columns
```

```
Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income',
'Kidhome',
       'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
```

```
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases',
'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue',
'Response'],
      dtype='object')
```

```
df.shape
```

```
(2240, 29)
```

```
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
```

```
df.describe().T
```


{"summary":"{\n \"name\": \"df\",\n \"rows\": 26,\n \"fields\": [\n   {\n   \"column\": \"count\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 4.706787243316417,\n   \"min\": 2216.0,\n   \"max\": 2240.0,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   2216.0,\n   2240.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"mean\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 10245.542508830891,\n   \"min\": 0.009375,\n   \"max\": 52247.25135379061,\n   \"num_unique_values\": 25,\n   \"samples\": [\n   166.95,\n   5.316517857142857\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"std\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 4945.885982814479,\n   \"min\": 0.0,\n   \"max\": 25173.076660901403,\n   \"num_unique_values\": 24,\n   \"samples\": [\n   225.7153725117536,\n   2.426645009547286\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"min\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 492.65479410419437,\n   \"min\": 0.0,\n   \"max\": 1893.0,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1893.0,\n   11.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"25%\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 6916.682939569983,\n   \"min\": 0.0,\n   \"max\": 35303.0,\n   \"num_unique_values\": 12,\n   \"samples\": [\n   2.0,\n   9.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"50%\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 10077.796284845519,\n   \"min\": 0.0,\n   \"max\": 51381.5,\n   \"num_unique_values\": 16,\n   \"samples\": [\n   5458.5,\n   1970.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"75%\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 13453.114094184804,\n   \"min\": 0.0,\n   \"max\": 68522.0,\n   \"num_unique_values\": 17,\n   \"samples\": [\n   8427.75,\n   1977.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"max\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 130623.70202867237,\n   \"min\": 1.0,\n   \"max\": 666666.0,\n   \"num_unique_values\": 19,\n   \"samples\": [\n   11191.0,\n   1493.0\n   ],\n   \"semantic_type\":

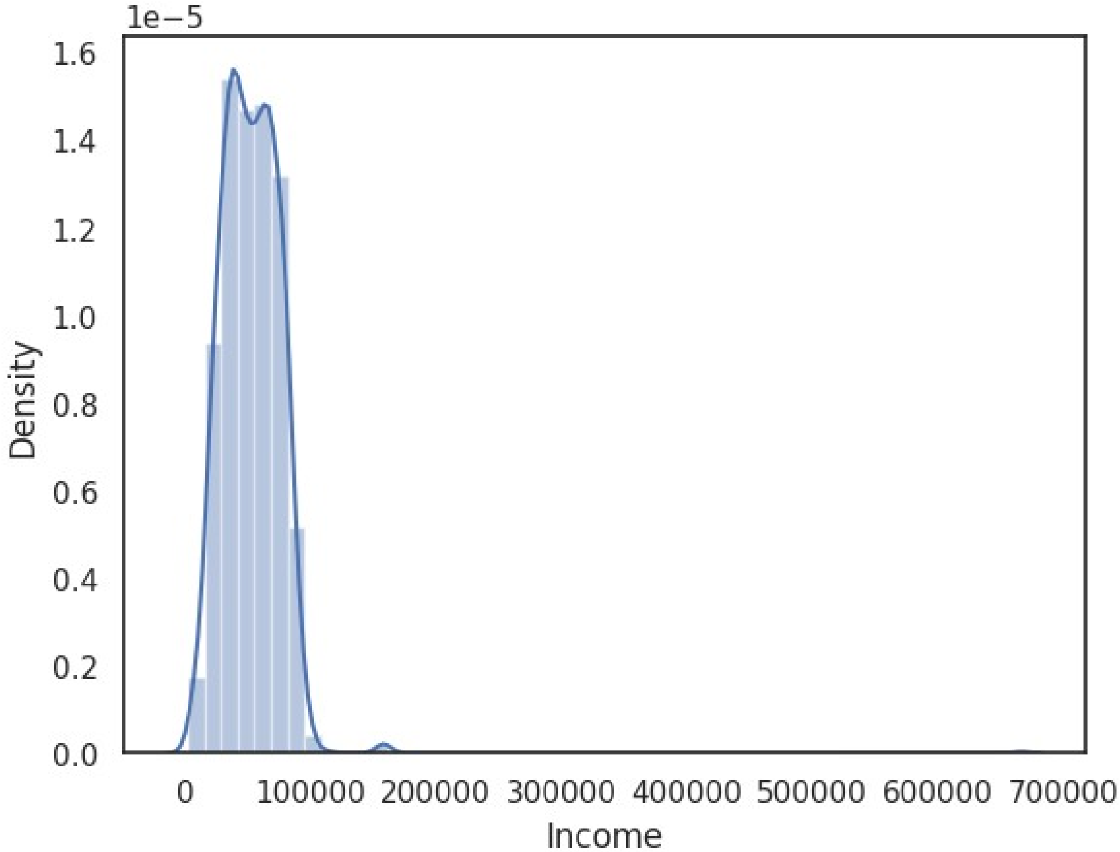
```
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```


```
df.isna().sum()
```

```
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
```

since there are some missing values in Income we will check that column and replace missing values with mean or median

```
sns.distplot(df['Income'])
plt.show()
```



since the data is left skewed we will replace the missing values with median

```
#FILL THE MISSING VALUES WITH THE MEDIAN VALUES..
df['Income']=df['Income'].fillna(df['Income'].median())

df[df.duplicated()]
```

{"type":"dataframe"}

```
#FINDING THE NUMBER OF UNIQUE VALUES PRESENT IN EACH COLUMN...
df.nunique()
```

```
ID                     2240
Year_Birth               59
Education                 5
Marital_Status            8
Income                 1975
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
```

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

```
df=df.drop(columns=["Z_CostContact", "Z_Revenue"],axis=1)
```

Univariate Analysis :-

1.Analysis on Year_Birth Variable.

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

Year_Birth
1976    89
1971    87
1975    83
1972    79
1978    77
1970    77
1973    74
1965    74
1969    71
1974    69
1956    55
1958    53
1979    53
1952    52
1977    52
1968    51
1959    51
1966    50
```

```
1954    50
1955    49
1960    49
1982    45
1963    45
1967    44
1962    44
1957    43
1951    43
1983    42
1986    42
1964    42
1980    39
1981    39
1984    38
1961    36
1953    35
1985    32
1989    30
1949    30
1950    29
1988    29
1987    27
1948    21
1990    18
1946    16
1947    16
1991    15
1992    13
1945     8
1943     7
1944     7
1993     5
1995     5
1994     3
1996     2
1899     1
1941     1
1893     1
1900     1
1940     1
Name: count, dtype: int64
```

Data points in year birth are uniformly distributed

2.Analysis On Education Variable.

```
df['Education'].unique()
```

```
array(['Graduation', 'PhD', 'Master', 'Basic', '2n Cycle'],
dtype=object)
```

```
#CHANGING CATEGORY INTO "UG" AND "PG" ONLY....
df['Education'] = df['Education'].replace(['PhD','2n
Cycle','Graduation', 'Master'],'Post Graduate')
df['Education'] = df['Education'].replace(['Basic'], 'Under Graduate')
```

```
df.Education.value_counts()
```

```
Education
Post Graduate     2186
Under Graduate      54
Name: count, dtype: int64
```

We observed that most of the data points here are post-Graduated

3.Analysis On Marital_Status Variable.

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

```
array(['Single', 'Together', 'Married', 'Divorced', 'Widow', 'Alone',
       'Absurd', 'YOLO'], dtype=object)
```

```
#REPLACING THE CONFLICT VALUES IN Marital_status..
df['Marital_Status'] = df['Marital_Status'].replace(['Married',
'Together'],'Relationship')
df['Marital_Status'] = df['Marital_Status'].replace(['Divorced',
'Widow', 'Alone', 'YOLO', 'Absurd'],'Single')
```

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

```
Marital_Status
Relationship    1444
Single           796
Name: count, dtype: int64
```

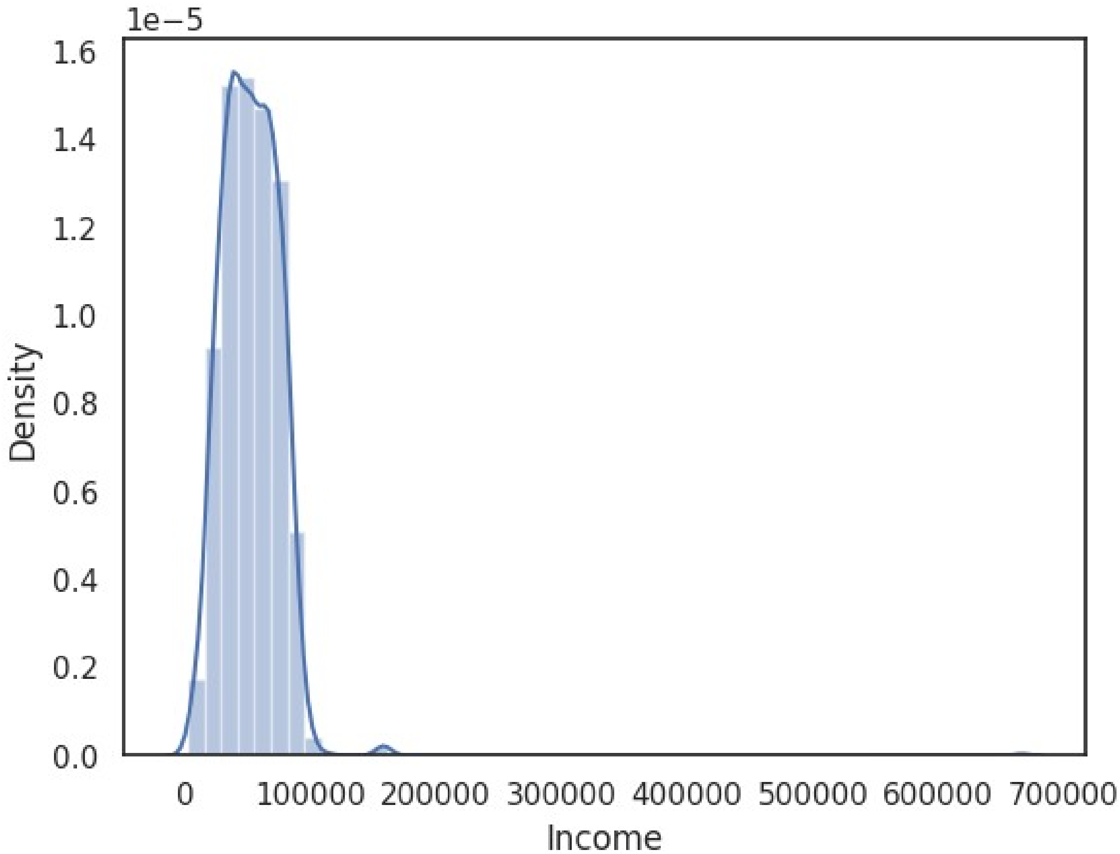
64.46% of Customers in the dataset are in "Relationship". 35.53% of Customers in the dataset are "Single".

4.Analysis On Income Variable.

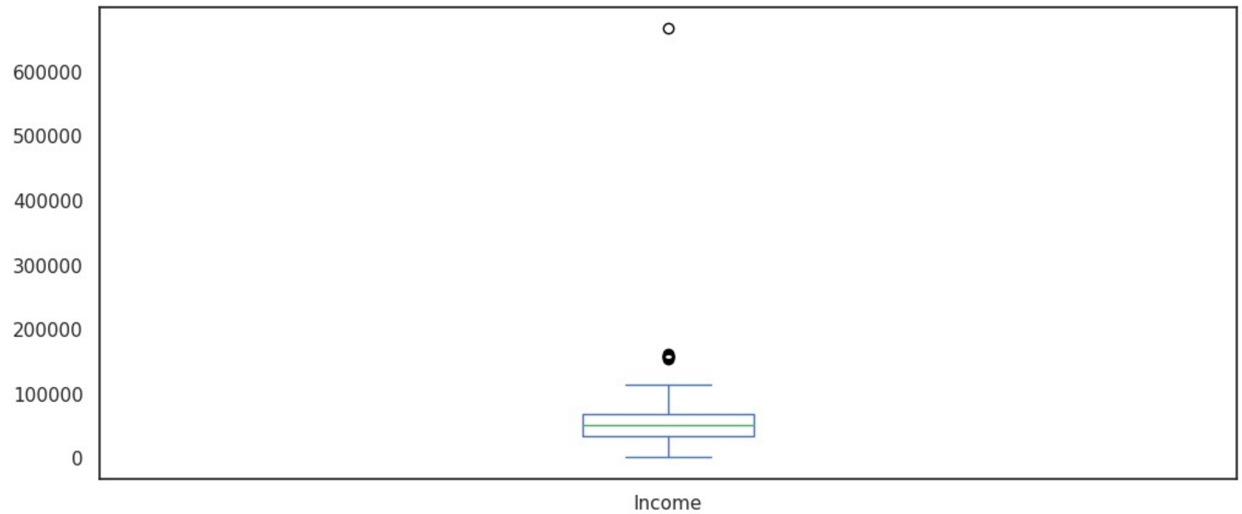
```
df['Income'].describe()
```

```
count     2240.000000
mean     52237.975446
std      25037.955891
min       1730.000000
25%      35538.750000
50%      51381.500000
75%      68289.750000
```

```
max      666666.000000
Name: Income, dtype: float64

sns.distplot(df["Income"])
plt.show()
```




```
df["Income"].plot.box(figsize=(12,5))
plt.show()
```



The income column is left skewed as we saw earrlier but it has some outliers that we will treat it in later stage while model building

5.Analysis On "Kidhome,Teenhome" Variable.

```
df['Teenhome'].unique()

array([0, 1, 2])

df['Kidhome'].unique()

array([0, 1, 2])

# Combining different dataframe into a single column to reduce the
number of dimension

df['Kids'] = df['Kidhome'] + df['Teenhome']

df.Kids.value_counts()

Kids
1    1128
0     638
2     421
3      53
Name: count, dtype: int64
```

50.35% of Customers in the dataset have 1 kid. 28.48% of Customers in the dataset have no kids. 18.79% of Customers in the dataset have 2 kids. 2.36% of Customers in the dataset have 3 kids.

6.Analysis On "MntWines,MntMeatProducts,MntFishProducts,MntSweetProducts,MntGoldProds" Variable.

```
df[['MntFruits','MntMeatProducts']].head()
```

{"summary":"{\n \"name\": \"df[['MntFruits','MntMeatProducts']]\",\n \"rows\": 5,\n \"fields\": [\n   {\n   \"column\": \"MntFruits\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 35,\n   \"min\": 1,\n   \"max\": 88,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1,\n   43,\n   49\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"MntMeatProducts\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 220,\n   \"min\": 6,\n   \"max\": 546,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   6,\n   118,\n   127\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df['MntFishProducts'].nunique()
```

```
182
```

```
df['MntFruits'].nunique()
```

```
158
```

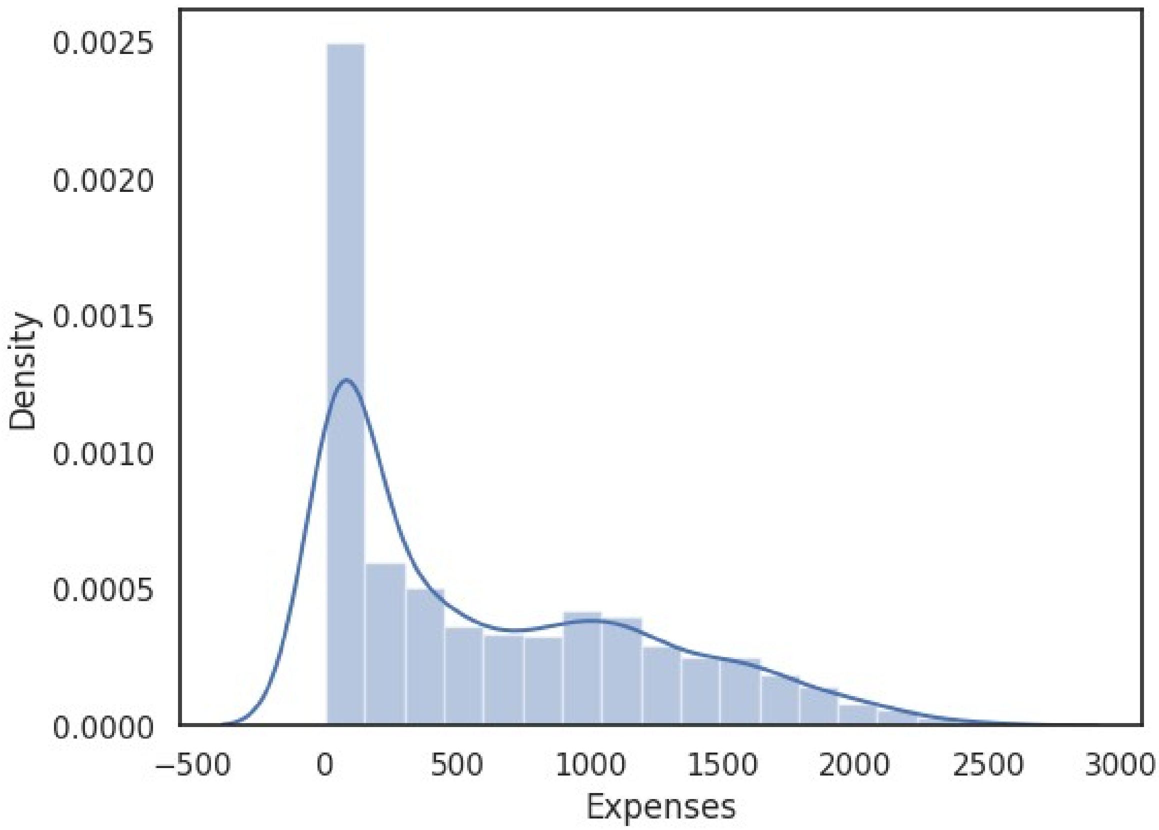
```
# Combining different dataframe into a single column to reduce the number of dimension

df['Expenses'] = df['MntWines'] + df['MntFruits'] + df['MntMeatProducts'] + df['MntFishProducts'] + df['MntSweetProducts'] + df['MntGoldProds']
df['Expenses'].head(10)
```

```
0    1617
1      27
2     776
3      53
4     422
5     716
6     590
7     169
8      46
9      49
Name: Expenses, dtype: int64
```

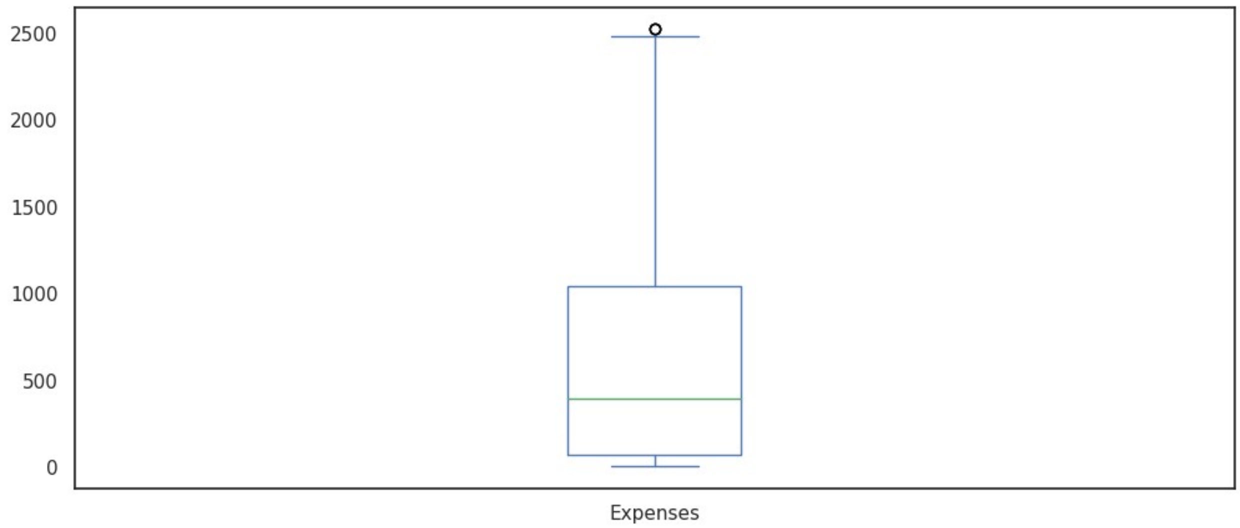
```
df['Expenses'].describe()
```

```
count    2240.000000
mean      605.798214
std       602.249288
min         5.000000
25%        68.750000
50%       396.000000
75%      1045.500000
```

```
max        2525.000000
Name: Expenses, dtype: float64

sns.distplot(df["Expenses"])
plt.show()
```




```
df["Expenses"].plot.box(figsize=(12,5))
plt.show()
```



The distribution of expenses is uniform

7.Analysis on "AcceptedCmp1,AcceptedCmp2,AcceptedCmp3,AcceptedCmp4,AcceptedCmp5" Variable.

```
df['AcceptedCmp1'].unique()
```

```
array([0, 1])
```

```
df['AcceptedCmp2'].unique()
```

```
array([0, 1])
```

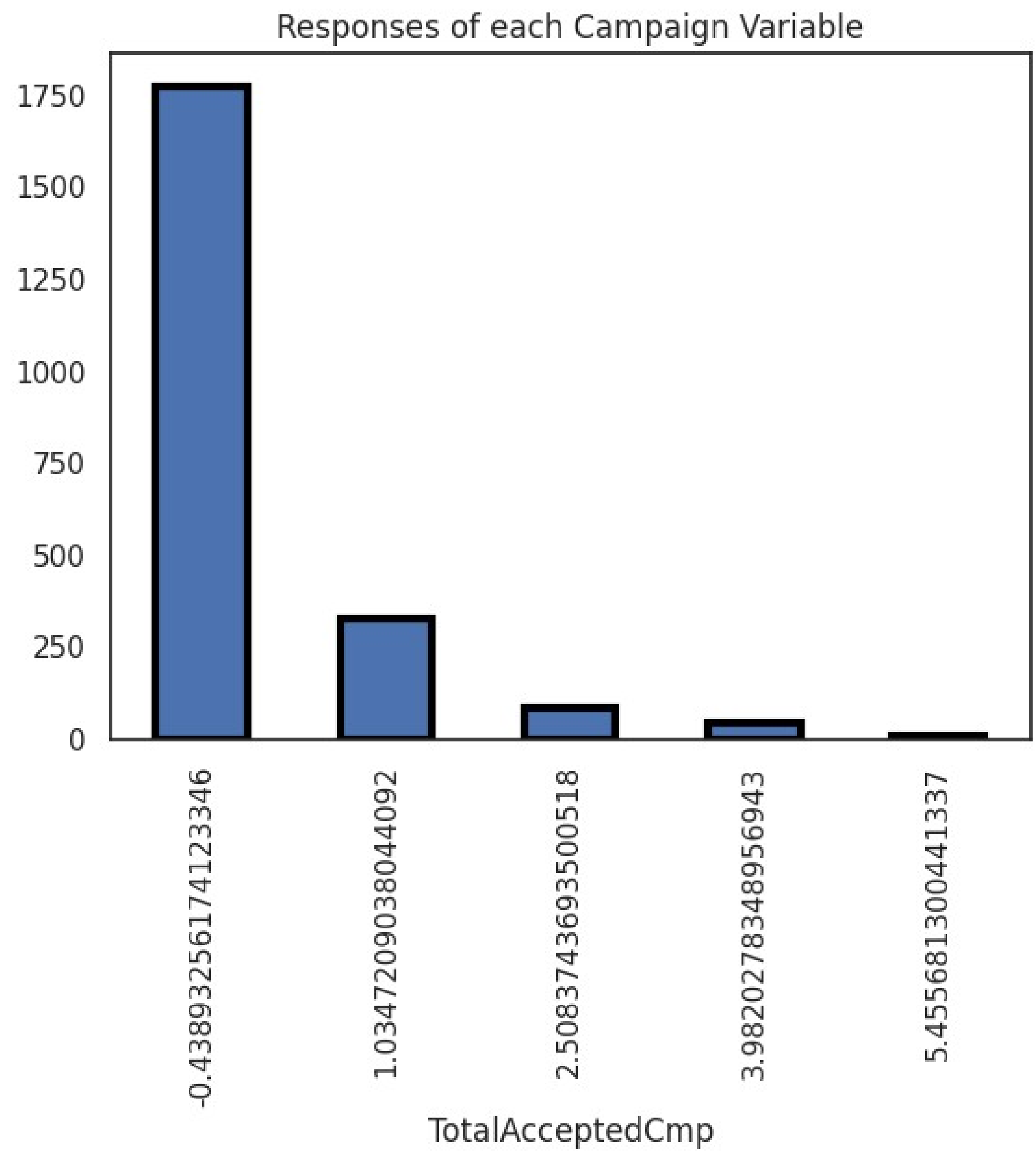
```
df['TotalAcceptedCmp'] = df['AcceptedCmp1'] + df['AcceptedCmp2'] +
df['AcceptedCmp3'] + df['AcceptedCmp4'] + df['AcceptedCmp5']
```

```
#CHECKING NUMBER OF UNIQUE CATEGORIES PRESENT IN THE
"TotalAcceptedCmp"
print("Unique categories present in the
TotalAcceptedCmp:",df['TotalAcceptedCmp'].value_counts())
print("\n")
```

```
Unique categories present in the TotalAcceptedCmp: TotalAcceptedCmp
0    1777
1     325
2      83
3      44
4      11
Name: count, dtype: int64
```

```
df['TotalAcceptedCmp'].value_counts().plot(kind='bar',edgecolor =
"black",linewidth = 3)
```

```
plt.title("Responses of each Campaign Variable")
plt.show()
```



79.33% of Customers accepted the offer in the campaign are "0". 14.50% of Customers accepted the offer in the campaign are "1". 3.70% of Customers accepted the offer in the campaign are "2". 1.96% of Customers accepted the offer in the campaign are "3". 0.49% of Customers accepted the offer in the campaign are "4".

8.Analysis on "NumWebPurchases,NumCatalogPurchases,NumStorePurchases,NumDealsPurchases" Variable.

```
df['NumWebPurchases'].unique()

array([ 8,  1,  2,  5,  6,  7,  4,  3, 11,  0, 27, 10,  9, 23, 25])

df['NumCatalogPurchases'].unique()

array([10,  1,  2,  0,  3,  4,  6, 28,  9,  5,  8,  7, 11, 22])

df['NumStorePurchases'].unique()

array([ 4,  2, 10,  6,  7,  0,  3,  8,  5, 12,  9, 13, 11,  1])

df['NumTotalPurchases'] = df['NumWebPurchases'] +
df['NumCatalogPurchases'] + df['NumStorePurchases'] +
df['NumDealsPurchases']
df['NumTotalPurchases'].unique()

array([25,  6, 21,  8, 19, 22, 10,  2,  4, 16, 15,  5, 26,  9, 13, 12,
43,
       17, 20, 14, 27, 11, 18, 28,  7, 24, 29, 23, 32, 30, 37, 31, 33,
35,
       39,  1, 34,  0, 44])

df[['NumTotalPurchases']]
```

{"summary":"{\n \"name\": \"df[['NumTotalPurchases']]\",\n \"rows\":
2240,\n \"fields\": [\n   {\n   \"column\":
\"NumTotalPurchases\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 7,\n   \"min\": 0,\n
\"max\": 44,\n   \"num_unique_values\": 39,\n   \"samples\":
[\n   35,\n   34,\n   19\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

```
df['NumTotalPurchases'].describe()

count    2240.000000
mean       14.862054
std         7.677173
min         0.000000
25%         8.000000
50%        15.000000
75%        21.000000
max        44.000000
Name: NumTotalPurchases, dtype: float64

df['NumTotalPurchases'].value_counts()

NumTotalPurchases
7     149
5     145
4     128
```

```
6     123
17    116
9     102
19    101
16    101
21     95
8      94
22     94
20     94
23     87
10     80
18     79
15     74
12     70
25     68
26     67
11     67
24     56
14     55
13     44
27     39
28     35
29     19
32     12
30     11
31     11
1       4
0       4
33      4
34      4
2       3
37      1
39      1
35      1
43      1
44      1
Name: count, dtype: int64
```

```
df.head()
```

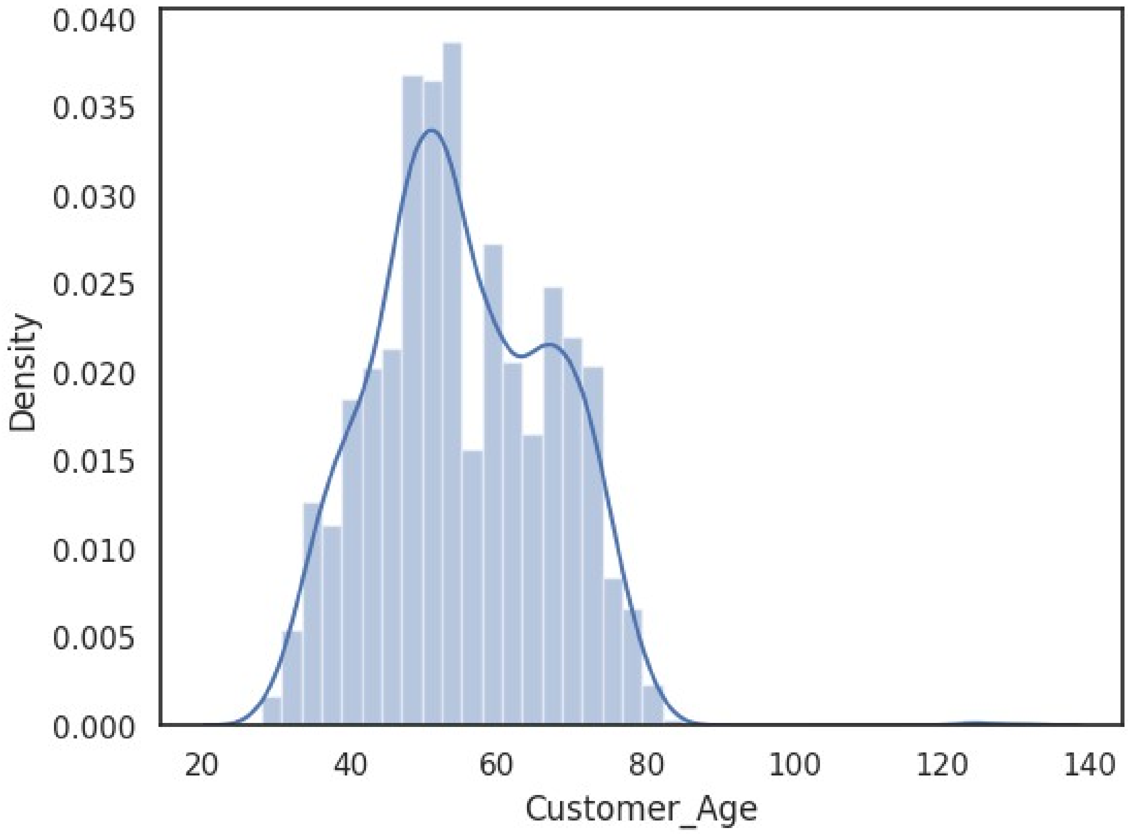
{"type":"dataframe","variable_name":"df"}

1. Converting the Year_Birth to customer_Age

```
#ADDING A COLUMN "customer_Age" IN THE DATAFRAME....
df['Customer_Age'] = (pd.Timestamp('now').year) - df['Year_Birth']
df.head()
```

{"type":"dataframe","variable_name":"df"}

```
sns.distplot(df["Customer_Age"])
plt.show()
```



Most of the cutomers we have are in middle age i.e between 35-55

```
# Deleting some column to reduce dimension and complexity of model

col_del = ["Year_Birth","ID","AcceptedCmp1" , "AcceptedCmp2",
"AcceptedCmp3" , "AcceptedCmp4","AcceptedCmp5","NumWebVisitsMonth",
"NumWebPurchases","NumCatalogPurchases","NumStorePurchases","NumDealsP
urchases" , "Kidhome", "Teenhome","MntWines", "MntFruits",
"MntMeatProducts", "MntFishProducts", "MntSweetProducts",
"MntGoldProds"]
df=df.drop(columns=col_del,axis=1)

df.head()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 2240,\n \"fields\":
[\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 2,\n
\"samples\": [\n   \"Under Graduate\",\n   \"Post
Graduate\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Marital_Status\",\n   \"properties\": {\n   \"dtype\":

\"category\",\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   \"Relationship\",\n   \"Single\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Income\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\": 25037.9558906219,\
n   \"min\": 1730.0,\n   \"max\": 666666.0,\n
\"num_unique_values\": 1975,\n   \"samples\": [\n
53154.0,\n   63211.0\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Dt_Customer\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 663,\n
\"samples\": [\n   \"23-04-2013\",\n   \"02-12-2012\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Recency\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
28,\n   \"min\": 0,\n   \"max\": 99,\n
\"num_unique_values\": 100,\n   \"samples\": [\n   78,\n
87\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Complain\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   1,\n   0\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Response\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   0,\n   1\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Kids\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 3,\n   \"num_unique_values\": 4,\n   \"samples\":
[\n   2,\n   3\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 602,\n   \"min\": 5,\n
\"max\": 2525,\n   \"num_unique_values\": 1054,\n
\"samples\": [\n   160,\n   1822\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0,\n   \"min\": 0,\n   \"max\": 4,\n
\"num_unique_values\": 5,\n   \"samples\": [\n   1,\n
4\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"NumTotalPurchases\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 7,\n   \"min\": 0,\n
\"max\": 44,\n   \"num_unique_values\": 39,\n   \"samples\":
[\n   35,\n   34\n   ],\n   \"semantic_type\":

```
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Customer_Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 11,\n        \"min\": 28,\n
\"max\": 131,\n        \"num_unique_values\": 59,\n
\"samples\": [\n          67,\n          57\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Education          2240 non-null   object
 1   Marital_Status     2240 non-null   object
 2   Income             2240 non-null   float64
 3   Dt_Customer        2240 non-null   object
 4   Recency            2240 non-null   int64
 5   Complain           2240 non-null   int64
 6   Response           2240 non-null   int64
 7   Kids               2240 non-null   int64
 8   Expenses           2240 non-null   int64
 9   TotalAcceptedCmp   2240 non-null   int64
 10  NumTotalPurchases  2240 non-null   int64
 11  Customer_Age       2240 non-null   int64
dtypes: float64(1), int64(8), object(3)
memory usage: 210.1+ KB
```

In the next step, I am going to create a feature out of "Dt_Customer" that indicates the number of days a customer is registered in the firm's database. However, in order to keep it simple, I am taking this value relative to the most recent customer in the record.

Thus to get the values I must check the newest and oldest recorded dates.

```
df["Dt_Customer"] = pd.to_datetime(df["Dt_Customer"], format="%d-%m-
%Y")

dates = []
for i in df["Dt_Customer"]:
    i = i.date()
    dates.append(i)
#Dates of the newest and oldest recorded customer
print("The newest customer's enrolment date in
therecords:",max(dates))
print("The oldest customer's enrolment date in the
records:",min(dates))
```

```
The newest customer's enrolment date in therecords: 2014-06-29
The oldest customer's enrolment date in the records: 2012-07-30
```

Creating a feature ("Customer_For") of the number of days the customers started to shop in the store relative to the last recorded date

```python
#Created a feature "Customer_For"
days = []
d1 = max(dates) #taking it to be the newest customer
for i in dates:
    delta = d1 - i
    days.append(delta)
df["Customer_For"] = days
df['Customer_For'] = df['Customer_For'].apply(lambda x:x.days)

df.head()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 2240,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Under Graduate\",\n   \"Post Graduate\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Relationship\",\n   \"Single\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 25037.9558906219,\n   \"min\": 1730.0,\n   \"max\": 666666.0,\n   \"num_unique_values\": 1975,\n   \"samples\": [\n   53154.0,\n   63211.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Dt_Customer\",\n   \"properties\": {\n   \"dtype\": \"date\",\n   \"min\": \"2012-07-30 00:00:00\",\n   \"max\": \"2014-06-29 00:00:00\",\n   \"num_unique_values\": 663,\n   \"samples\": [\n   \"2013-04-23 00:00:00\",\n   \"2012-12-02 00:00:00\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Recency\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 28,\n   \"min\": 0,\n   \"max\": 99,\n   \"num_unique_values\": 100,\n   \"samples\": [\n   78,\n   87\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Complain\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   1,\n   0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\

n   },\n   {\n   \"column\": \"Response\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   0,\n   1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 3,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   2,\n   3\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 602,\n   \"min\": 5,\n   \"max\": 2525,\n   \"num_unique_values\": 1054,\n   \"samples\": [\n   160,\n   1822\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 4,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1,\n   4\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"NumTotalPurchases\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 7,\n   \"min\": 0,\n   \"max\": 44,\n   \"num_unique_values\": 39,\n   \"samples\": [\n   35,\n   34\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_Age\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 11,\n   \"min\": 28,\n   \"max\": 131,\n   \"num_unique_values\": 59,\n   \"samples\": [\n   67,\n   57\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_For\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 202,\n   \"min\": 0,\n   \"max\": 699,\n   \"num_unique_values\": 663,\n   \"samples\": [\n   432,\n   574\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
df['Customer_For'].describe()
```

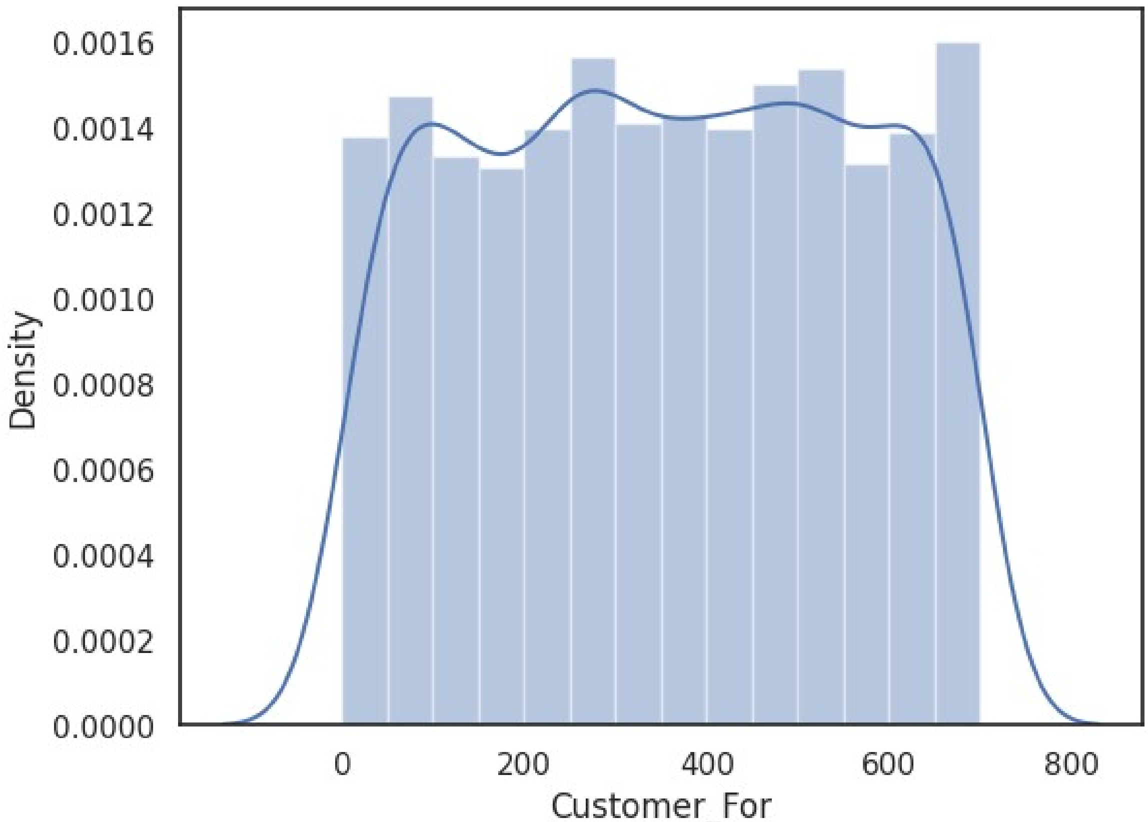
```
count    2240.000000
mean      353.582143
std       202.122512
min         0.000000
25%       180.750000
50%       355.500000
75%       529.000000
max       699.000000
Name: Customer_For, dtype: float64
```

```python
df.drop(['Dt_Customer','Recency','Complain','Response'],axis=1,inplace=True)

df.head()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 2240,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Under Graduate\",\n   \"Post Graduate\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Relationship\",\n   \"Single\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 25037.9558906219,\n   \"min\": 1730.0,\n   \"max\": 666666.0,\n   \"num_unique_values\": 1975,\n   \"samples\": [\n   53154.0,\n   63211.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 3,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   2,\n   3\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 602,\n   \"min\": 5,\n   \"max\": 2525,\n   \"num_unique_values\": 1054,\n   \"samples\": [\n   160,\n   1822\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 4,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1,\n   4\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"NumTotalPurchases\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 7,\n   \"min\": 0,\n   \"max\": 44,\n   \"num_unique_values\": 39,\n   \"samples\": [\n   35,\n   34\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_Age\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 11,\n   \"min\": 28,\n   \"max\": 131,\n   \"num_unique_values\": 59,\n   \"samples\": [\n   67,\n   57\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_For\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 202,\n   \"min\": 0,\n   \"max\": 699,\n

\"num_unique_values\": 663,\n   \"samples\": [\n   432,\n   574\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
sns.distplot(df["Customer_For"])
plt.show()
```



Most of the customers are regular to the campaign for 200-850 days

```
df.head()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 2240,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Under Graduate\",\n   \"Post Graduate\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Relationship\",\n   \"Single\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 25037.9558906219,\

```
n        \"min\": 1730.0,\n        \"max\": 666666.0,\n
\"num_unique_values\": 1975,\n        \"samples\": [\n
53154.0,\n          63211.0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Kids\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0,\n        \"min\": 0,\n
\"max\": 3,\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          2,\n          3\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Expenses\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 602,\n        \"min\": 5,\n
\"max\": 2525,\n        \"num_unique_values\": 1054,\n
\"samples\": [\n          160,\n          1822\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"TotalAcceptedCmp\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 4,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          1,\n
4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"NumTotalPurchases\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 7,\n        \"min\": 0,\n
\"max\": 44,\n        \"num_unique_values\": 39,\n        \"samples\":
[\n          35,\n          34\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Customer_Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 11,\n        \"min\": 28,\n
\"max\": 131,\n        \"num_unique_values\": 59,\n
\"samples\": [\n          67,\n          57\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Customer_For\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
202,\n        \"min\": 0,\n        \"max\": 699,\n
\"num_unique_values\": 663,\n        \"samples\": [\n          432,\n
574\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

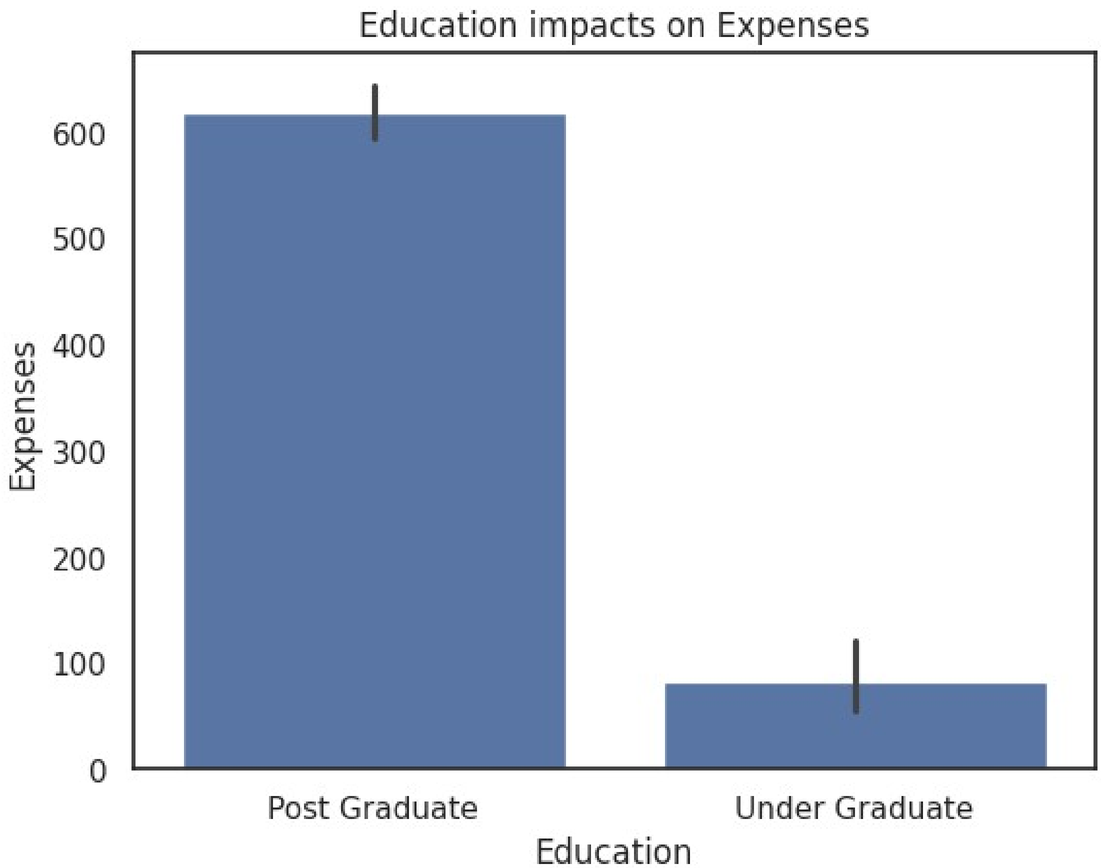

```
df.shape
```

```
(2240, 9)
```

Bivariate Analysis :-

1.Education vs Expenses

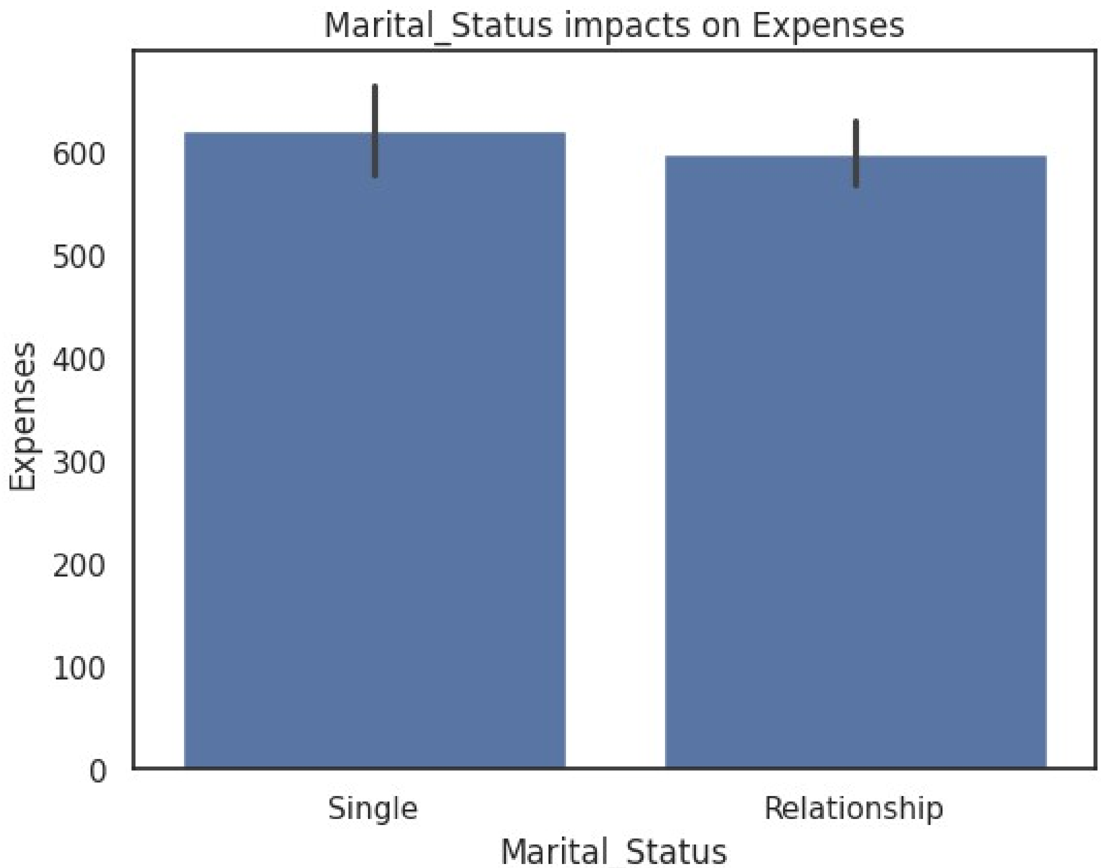
```
plt.title('Education impacts on Expenses')
ax = sns.barplot(x="Education", y="Expenses", data=df)
plt.show()
```



We observe that the post graduated people spends more than the UG people

2.Marital status vs Expenses

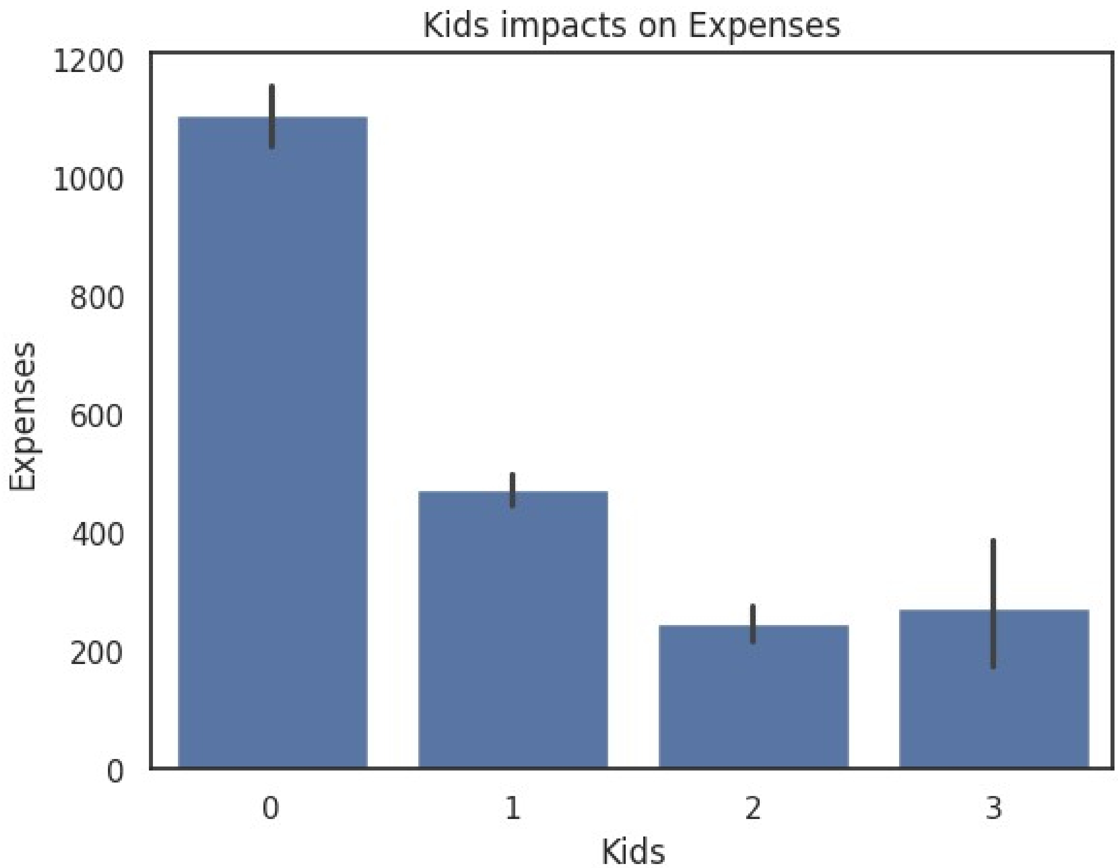
```
plt.title('Marital_Status impacts on Expenses')
ax = sns.barplot(x="Marital_Status", y="Expenses", data=df)
plt.show()
```



We observe that single and married people have the same spendings

3.Kids vs Expenses

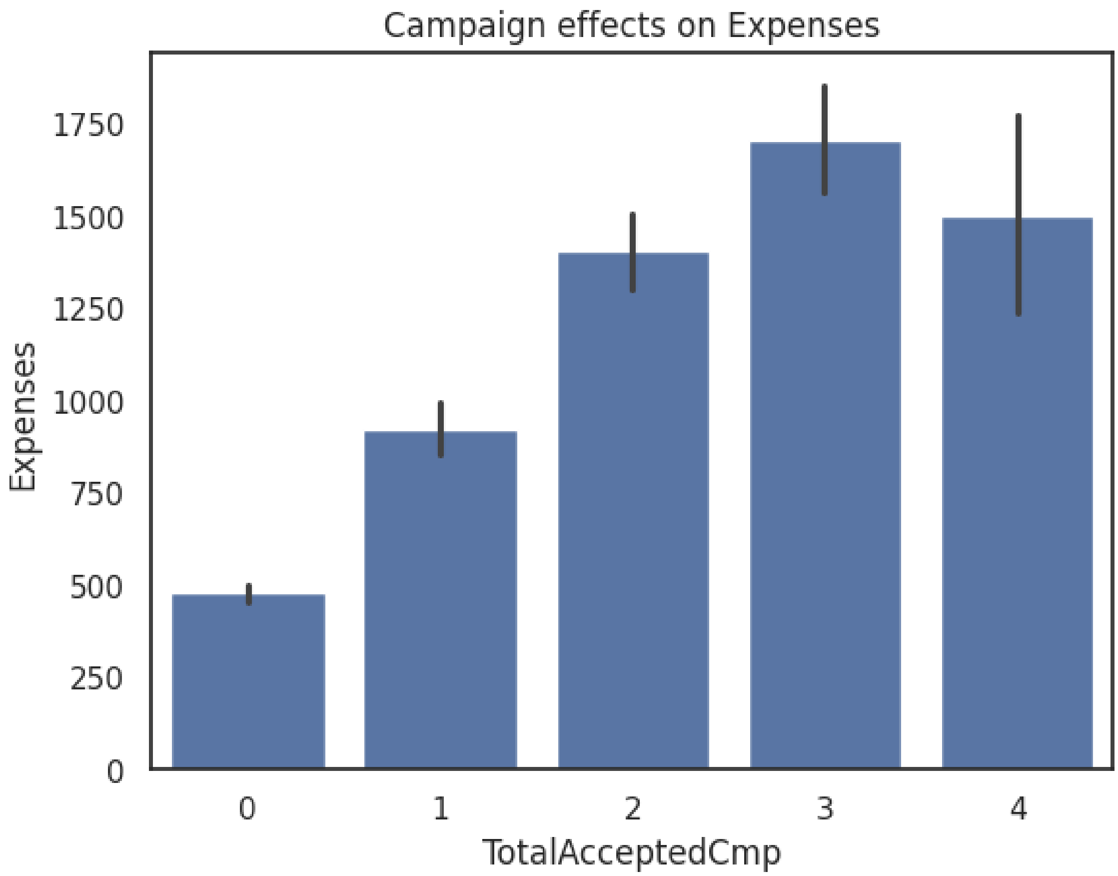
```
plt.title('Kids impacts on Expenses')
ax = sns.barplot(x="Kids", y="Expenses", data=df)
plt.show()
```



Here we observe some thing different that parents with 1 kid spends more than the parents who are having 2 or 3 kids

4.TotalAcceptedCmp vs Expenses

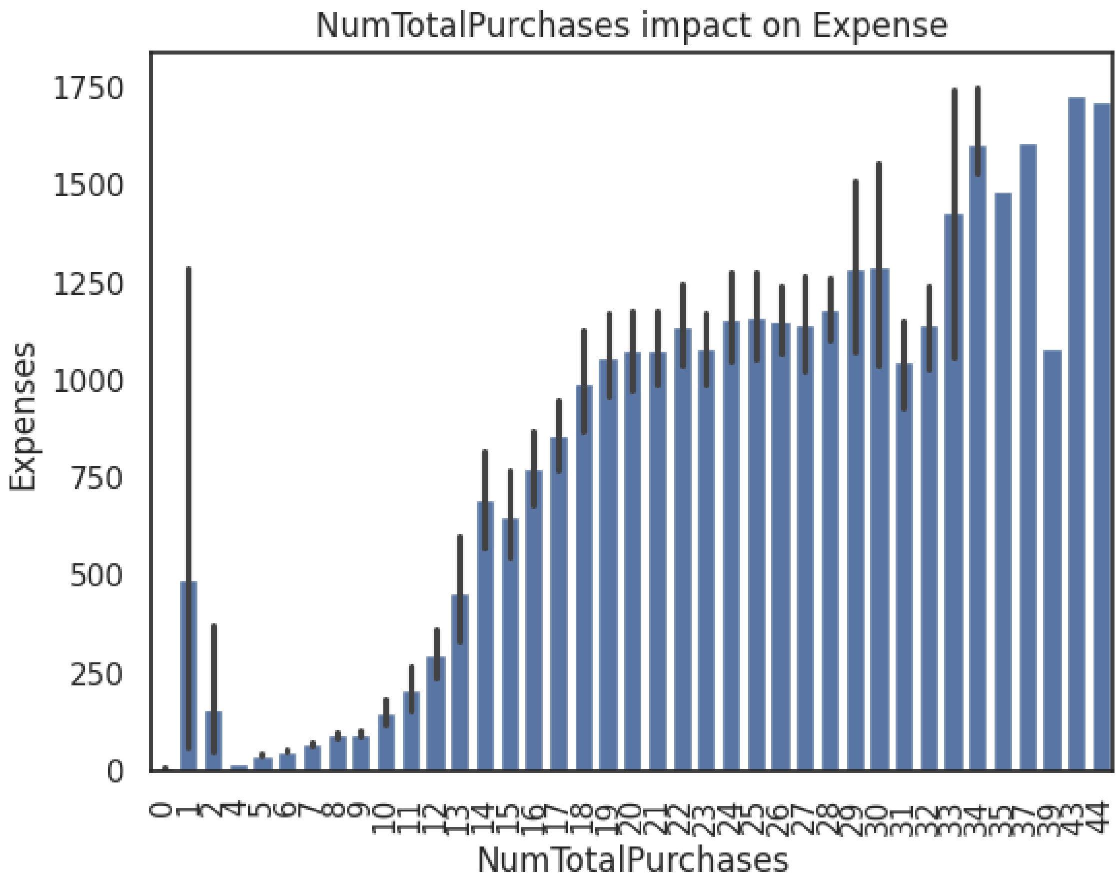
```
plt.title('Campaign effects on Expenses')
ax = sns.barplot(x="TotalAcceptedCmp", y="Expenses", data=df)
```



those who accepeted more campaign have more expenses

5.NumTotalPurchases vs Expenses

```
plt.title("NumTotalPurchases impact on Expense")
ax = sns.barplot(x="NumTotalPurchases", y="Expenses", data=df)
plt.xticks(rotation=90)
plt.show()
```



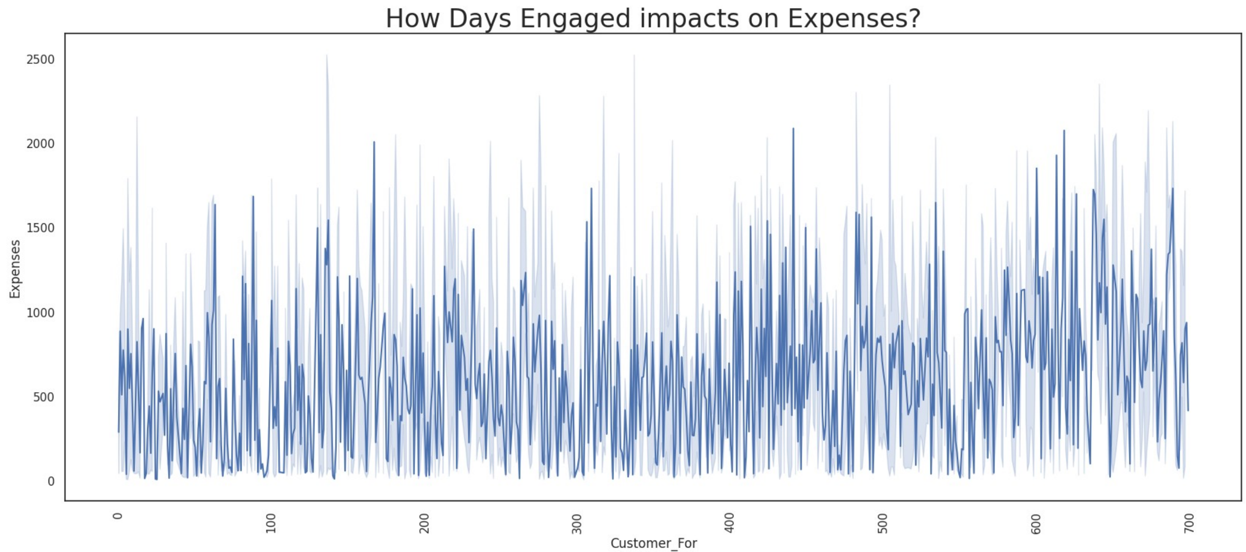
Those who have more purchases have more expenses

6.Day engaged vs Expenses

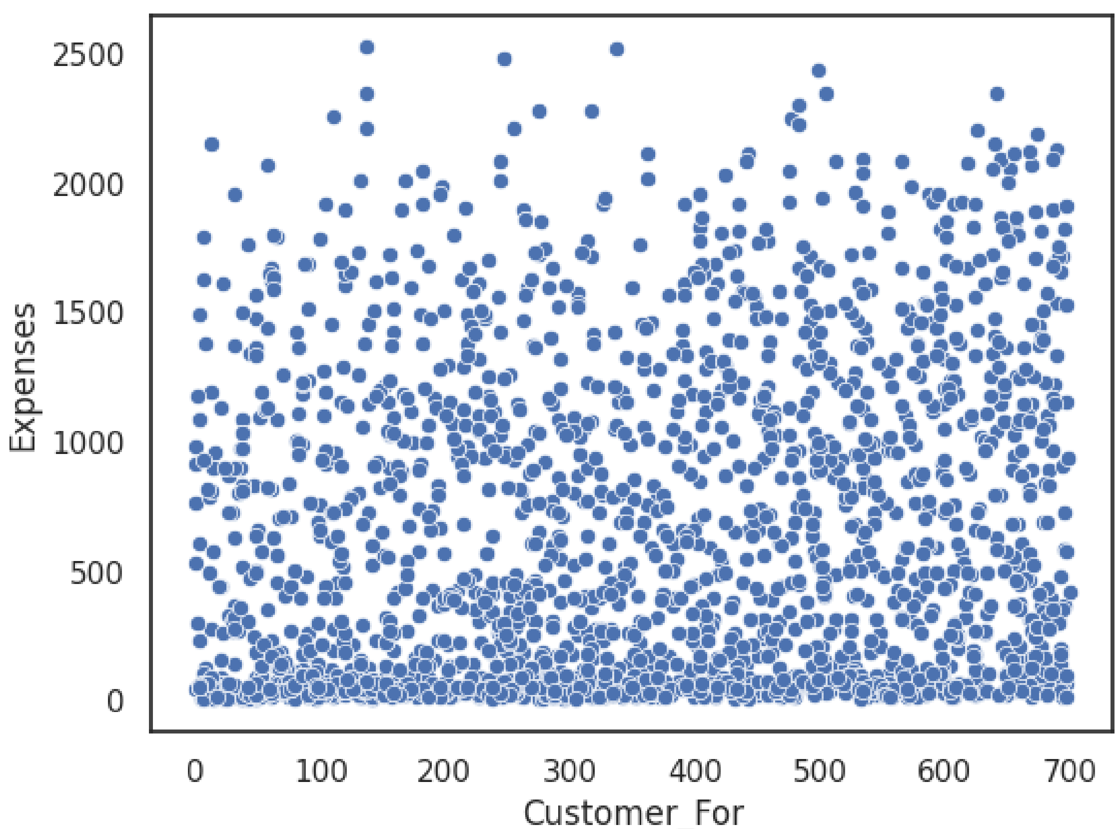
```
df.columns
```

```
Index(['Education', 'Marital_Status', 'Income', 'Kids', 'Expenses',
       'TotalAcceptedCmp', 'NumTotalPurchases', 'Customer_Age',
       'Customer_For'],
      dtype='object')
```

```
sns.set_theme(style="white")
plt.figure(figsize=(20,8))
plt.title("How Days Engaged impacts on Expenses?",fontsize=24)
plt.xticks(rotation=90)
ax = sns.lineplot(x="Customer_For", y="Expenses",
data=df,palette="rainbow")
```



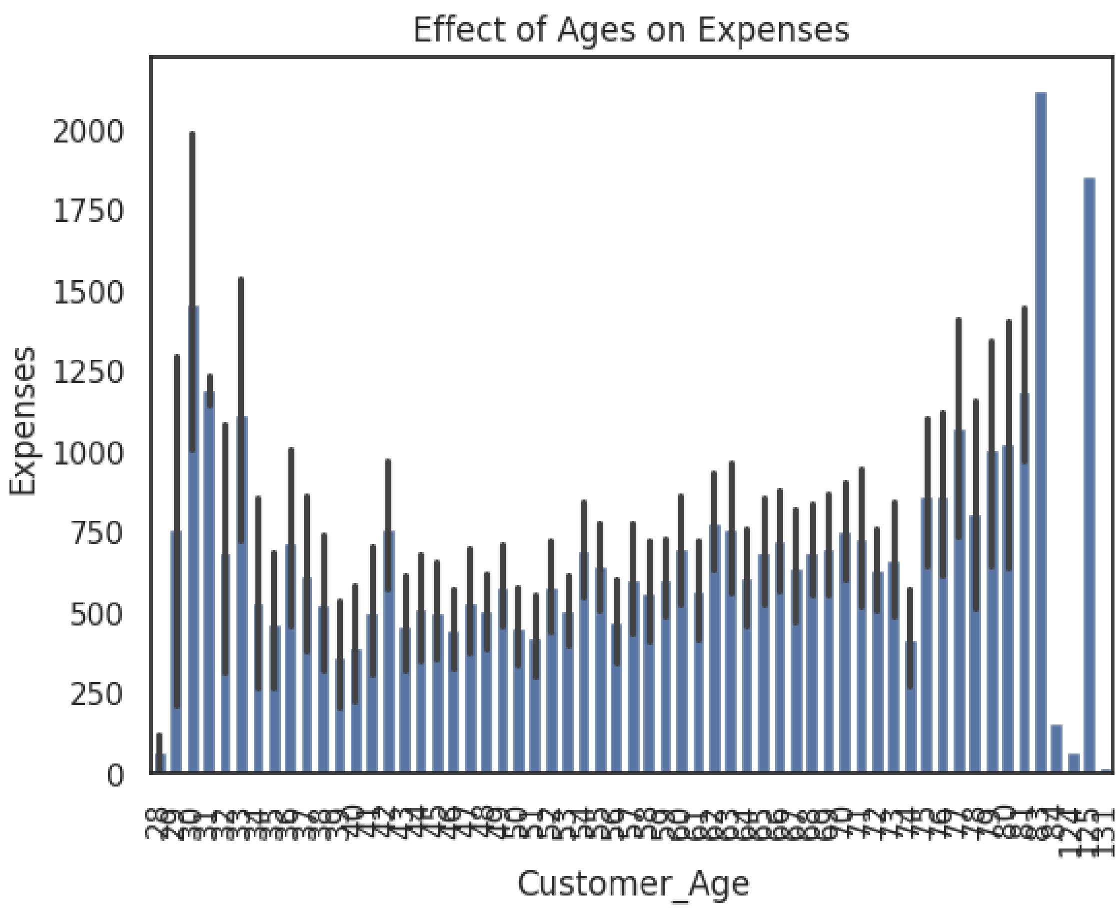

```
sns.scatterplot(x=df['Customer_For'],y=df['Expenses'])
plt.show()
```



No relationship between days enagaged vs expenses

7.Customer Age vs Expenses

```
plt.title('Effect of Ages on Expenses')
ax = sns.barplot(x="Customer_Age", y="Expenses", data=df)
plt.xticks(rotation=90)
plt.show()
```



People who are in middle age have less expenses than others

Remove some outliers present in age and income

```
df['Income'].describe()
```

```
count      2240.000000
mean      52237.975446
std       25037.955891
min        1730.000000
25%       35538.750000
50%       51381.500000
75%       68289.750000
max      666666.000000
Name: Income, dtype: float64
```

```
df['Customer_For'].describe()
```

```
count    2240.000000
mean      353.582143
std       202.122512
min         0.000000
25%       180.750000
50%       355.500000
75%       529.000000
max       699.000000
Name: Customer_For, dtype: float64
```

```
df.shape
```

```
(2240, 9)
```

```
df = df[df['Customer_Age'] < 90]
df = df[df['Income'] < 300000]
```

```
df.shape
```

```
(2236, 9)
```

```
df.head()
```

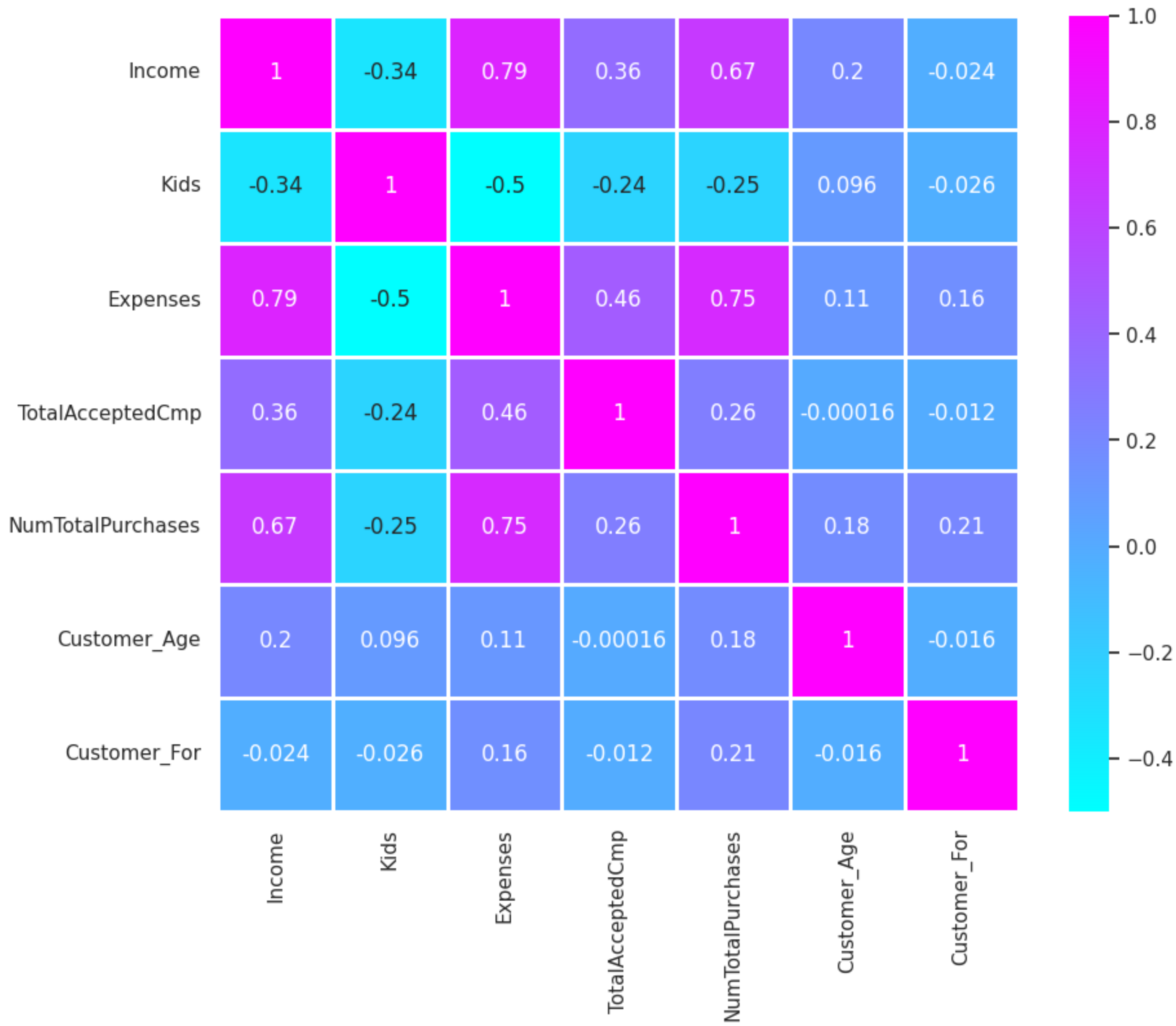
{"summary":"{\n \"name\": \"df\",\n \"rows\": 2236,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Under Graduate\",\n   \"Post Graduate\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Relationship\",\n   \"Single\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 21411.466850558867,\n   \"min\": 1730.0,\n   \"max\": 162397.0,\n   \"num_unique_values\": 1971,\n   \"samples\": [\n   82716.0,\n   30772.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 3,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   2,\n   3\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 601,\n   \"min\": 5,\n   \"max\": 2525,\n   \"num_unique_values\": 1054,\n   \"samples\": [\n   160,\n   1822\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\":

```
0,\n        \"min\": 0,\n        \"max\": 4,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          1,\n
4\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"NumTotalPurchases\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 7,\n        \"min\": 0,\n
\"max\": 44,\n        \"num_unique_values\": 39,\n        \"samples\":
[\n          35,\n          34\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Customer_Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 11,\n        \"min\": 28,\n
\"max\": 84,\n        \"num_unique_values\": 56,\n        \"samples\":
[\n          67,\n          57\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Customer_For\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 202,\n        \"min\": 0,\n
\"max\": 699,\n        \"num_unique_values\": 663,\n
\"samples\": [\n          683,\n          55\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


Finding the correlation:-

```
# Select only numeric columns
numeric_cols = df.select_dtypes(include=['number']).columns

# Create the heatmap
plt.figure(figsize=(10,8))
sns.heatmap(df[numeric_cols].corr(), annot=True,cmap =
'cool',linewidths=1)
```

```
<Axes: >
```



Income is more positively correlated to Expenses and Number of purchases

Expenses is positively correlated to Income and Number of pur chases and negitively correlated with Kids

```
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows
# how to understand word labels.
label_encoder = preprocessing.LabelEncoder()


df['Education'] = label_encoder.fit_transform(df['Education'])
df['Marital_Status'] =
label_encoder.fit_transform(df['Marital_Status'])

df.columns
```

```
Index(['Education', 'Marital_Status', 'Income', 'Kids', 'Expenses',
       'TotalAcceptedCmp', 'NumTotalPurchases', 'Customer_Age',
       'Customer_For'],
      dtype='object')
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
col_scale = ['Income', 'Kids', 'Expenses',
       'TotalAcceptedCmp', 'NumTotalPurchases', 'Customer_Age',
'Customer_For']

df[col_scale] = scaler.fit_transform(df[col_scale])

df.head()
```

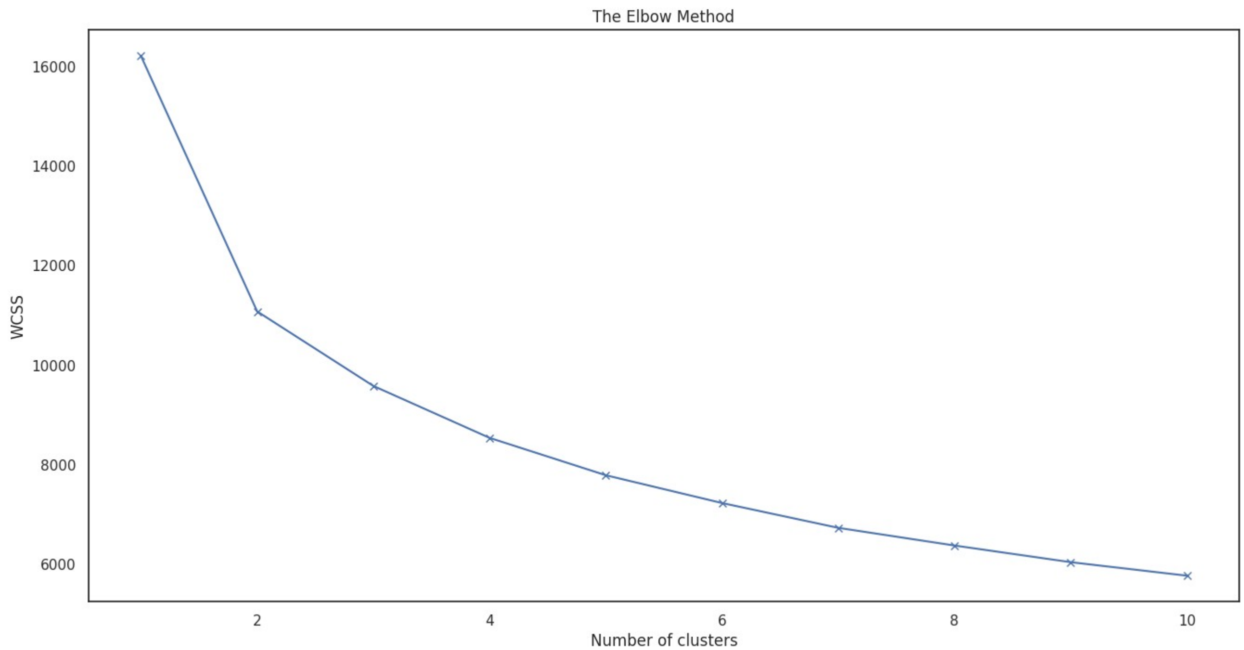
{"summary":"{\n \"name\": \"df\",\n \"rows\": 2236,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   1,\n   0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   0,\n   1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -2.3461189567805856,\n   \"max\": 5.159342509379425,\n   \"num_unique_values\": 1971,\n   \"samples\": [\n   1.4370929180813004,\n   -0.9894395397240711\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -1.2643075685916398,\n   \"max\": 2.7248623138696586,\n   \"num_unique_values\": 4,\n   \"samples\": [\n   1.3951390197158926,\n   2.7248623138696586\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -0.9987636117979262,\n   \"max\": 3.1891573371113737,\n   \"num_unique_values\": 1054,\n   \"samples\": [\n   -0.7411732359721557,\n   2.020860342237073\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -0.43893256174123346,\n   \"max\": 5.455681300441337,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1.0347209038044092,\n   5.455681300441337\n   ],\n

```
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"NumTotalPurchases\",\n      
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 
1.0002236886282312,\n        \"min\": -1.9374983150303555,\n        
\"max\": 3.7945369947259673,\n        \"num_unique_values\": 39,\n        
\"samples\": [\n          2.622075226821265,\n          
2.4918016970540755\n        ],\n        \"semantic_type\": \"\",\n        
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": 
\"Customer_Age\",\n      \"properties\": {\n        \"dtype\": 
\"number\",\n        \"std\": 1.0002236886282312,\n        \"min\": -
2.316276170624657,\n        \"max\": 2.4697772449023567,\n        
\"num_unique_values\": 56,\n        \"samples\": [\n          
1.0168681723316564,\n          0.16221577670183246\n        ],\n        
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Customer_For\",\n      
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 
1.0002236886282314,\n        \"min\": -1.750171428817049,\n        
\"max\": 1.7078905038986267,\n        \"num_unique_values\": 663,\n        
\"samples\": [\n          1.6287360104459074,\n          -
1.4780778575733262\n        ],\n        \"semantic_type\": \"\",\n        
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```


Model Building

K-Means

```
X_0 = df.copy()

from sklearn.cluster import KMeans

wcss=[]
for i in range (1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',random_state=42)
    kmeans.fit(X_0)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(16,8))
plt.plot(range(1,11),wcss, 'bx-')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



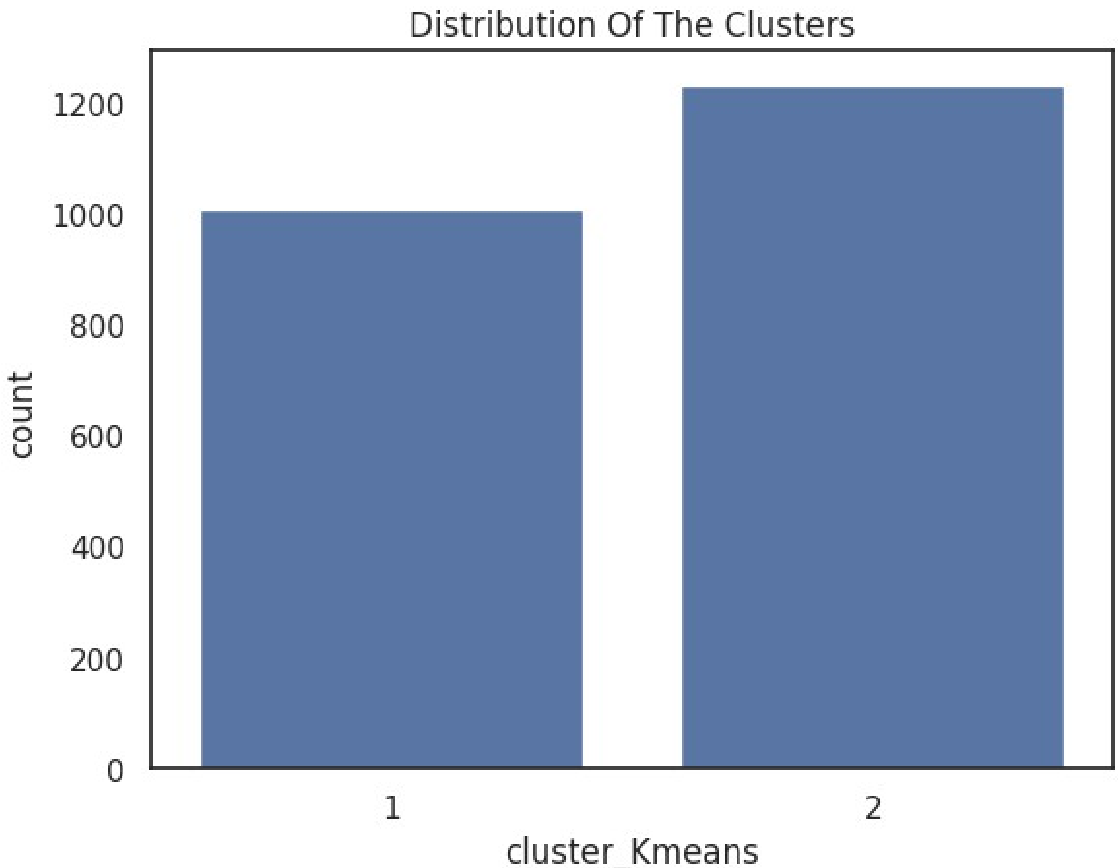
We can understand from the plot that cluster = 2 is the best...

```
# Training a predicting using K-Means Algorithm.

kmeans=KMeans(n_clusters=2, random_state=42).fit(X_0)
pred=kmeans.predict(X_0)


# Appending those cluster value into main dataframe (without standard-scalar)

X_0['cluster_Kmeans'] = pred + 1

X_0.head()
```

{"summary":"{\n \"name\": \"X_0\",\n \"rows\": 2236,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   1,\n   0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   0,\n   1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Income\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -2.3461189567805856,\n   \"max\": 5.159342509379425,\n   \"num_unique_values\": 1971,\n   \"samples\": [\n

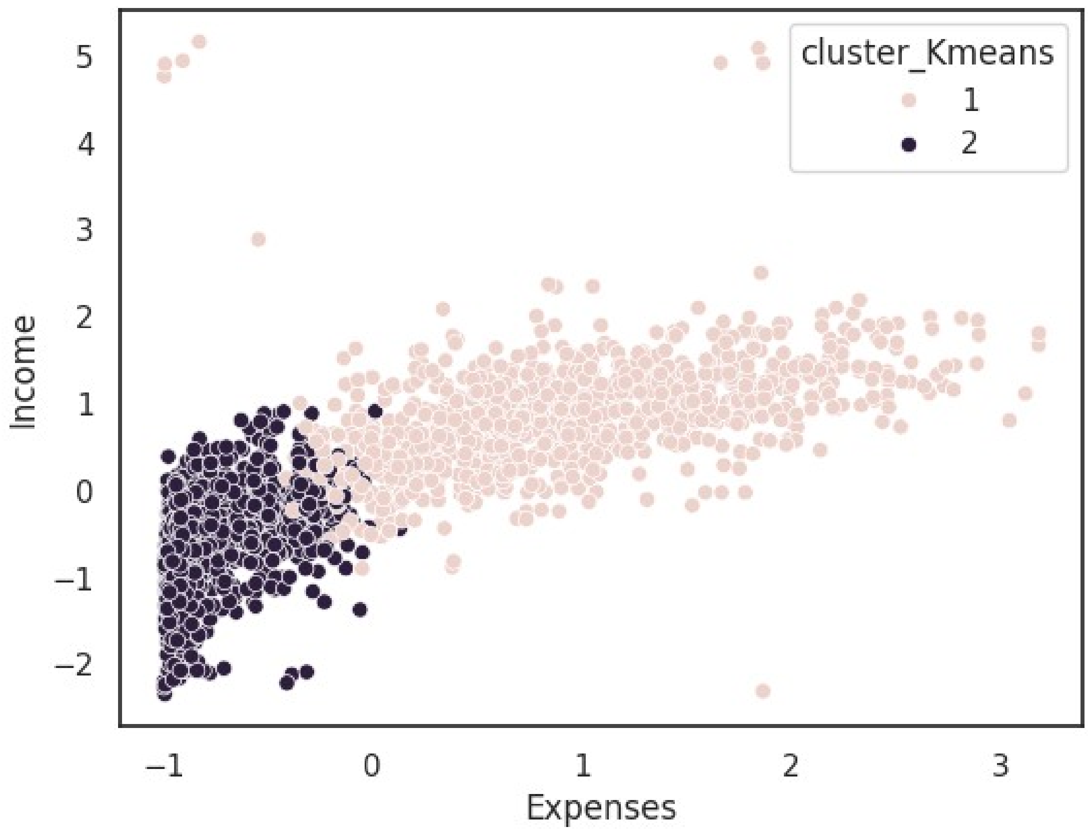
1.4370929180813004,\n   -0.9894395397240711\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n
\"min\": -1.2643075685916398,\n   \"max\": 2.724862313869658‌6,\n
\"num_unique_values\": 4,\n   \"samples\": [\n
1.3951390197158926,\n   2.7248623138696586\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\":
1.0002236886282312,\n   \"min\": -0.9987636117979262,\n
\"max\": 3.1891573371113737,\n   \"num_unique_values\": 1054,\n
\"samples\": [\n   -0.7411732359721557,\n
2.020860342237073\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -
0.43893256174123346,\n   \"max\": 5.455681300441337,\n
\"num_unique_values\": 5,\n   \"samples\": [\n
1.0347209038044092,\n   5.455681300441337\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"NumTotalPurchases\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
1.0002236886282312,\n   \"min\": -1.9374983150303555,\n
\"max\": 3.7945369947259673,\n   \"num_unique_values\": 39,\n
\"samples\": [\n   2.622075226821265,\n
2.4918016970540755\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Customer_Age\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -
2.316276170624657,\n   \"max\": 2.4697772449023567,\n
\"num_unique_values\": 56,\n   \"samples\": [\n
1.0168681723316564,\n   0.16221577670183246\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Customer_For\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
1.0002236886282314,\n   \"min\": -1.750171428817049,\n
\"max\": 1.7078905038986267,\n   \"num_unique_values\": 663,\n
\"samples\": [\n   1.6287360104459074,\n   -
1.4780778575733262\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"cluster_Kmeans\",\n   \"properties\": {\n   \"dtype\":
\"int32\",\n   \"num_unique_values\": 2,\n   \"samples\": [\
n   2,\n   1\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"X_0"}

```
sns.countplot(x=X_0["cluster_Kmeans"])
plt.title("Distribution Of The Clusters")
plt.show()
```




```
sns.scatterplot(x= X_0['Expenses'],y=
X_0['Income'],hue=X_0['cluster_Kmeans'])
```

```
<Axes: xlabel='Expenses', ylabel='Income'>
```



pca with Agglomerative clustering

```
df.head()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 2236,\n \"fields\":
[\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   1,\n   0\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Marital_Status\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   0,\n   1\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Income\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -
2.3461189567805856,\n   \"max\": 5.159342509379425,\n
\"num_unique_values\": 1971,\n   \"samples\": [\n
1.4370929180813004,\n   -0.9894395397240711\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Kids\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n
\"min\": -1.2643075685916398,\n   \"max\": 2.7248623138696586,\n

\"num_unique_values\": 4,\n   \"samples\": [\n   1.3951390197158926,\n   2.7248623138696586\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Expenses\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -0.9987636117979262,\n   \"max\": 3.1891573371113737,\n   \"num_unique_values\": 1054,\n   \"samples\": [\n   -0.7411732359721557,\n   2.020860342237073\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"TotalAcceptedCmp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -0.43893256174123346,\n   \"max\": 5.455681300441337,\n   \"num_unique_values\": 5,\n   \"samples\": [\n   1.0347209038044092,\n   5.455681300441337\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"NumTotalPurchases\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -1.9374983150303555,\n   \"max\": 3.7945369947259673,\n   \"num_unique_values\": 39,\n   \"samples\": [\n   2.622075226821265,\n   2.4918016970540755\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_Age\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282312,\n   \"min\": -2.316276170624657,\n   \"max\": 2.4697772449023567,\n   \"num_unique_values\": 56,\n   \"samples\": [\n   1.0168681723316564,\n   0.16221577670183246\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Customer_For\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1.0002236886282314,\n   \"min\": -1.750171428817049,\n   \"max\": 1.7078905038986267,\n   \"num_unique_values\": 663,\n   \"samples\": [\n   1.6287360104459074,\n   -1.4780778575733262\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
X_1 = df.copy()
```

```
X_1.head()
```

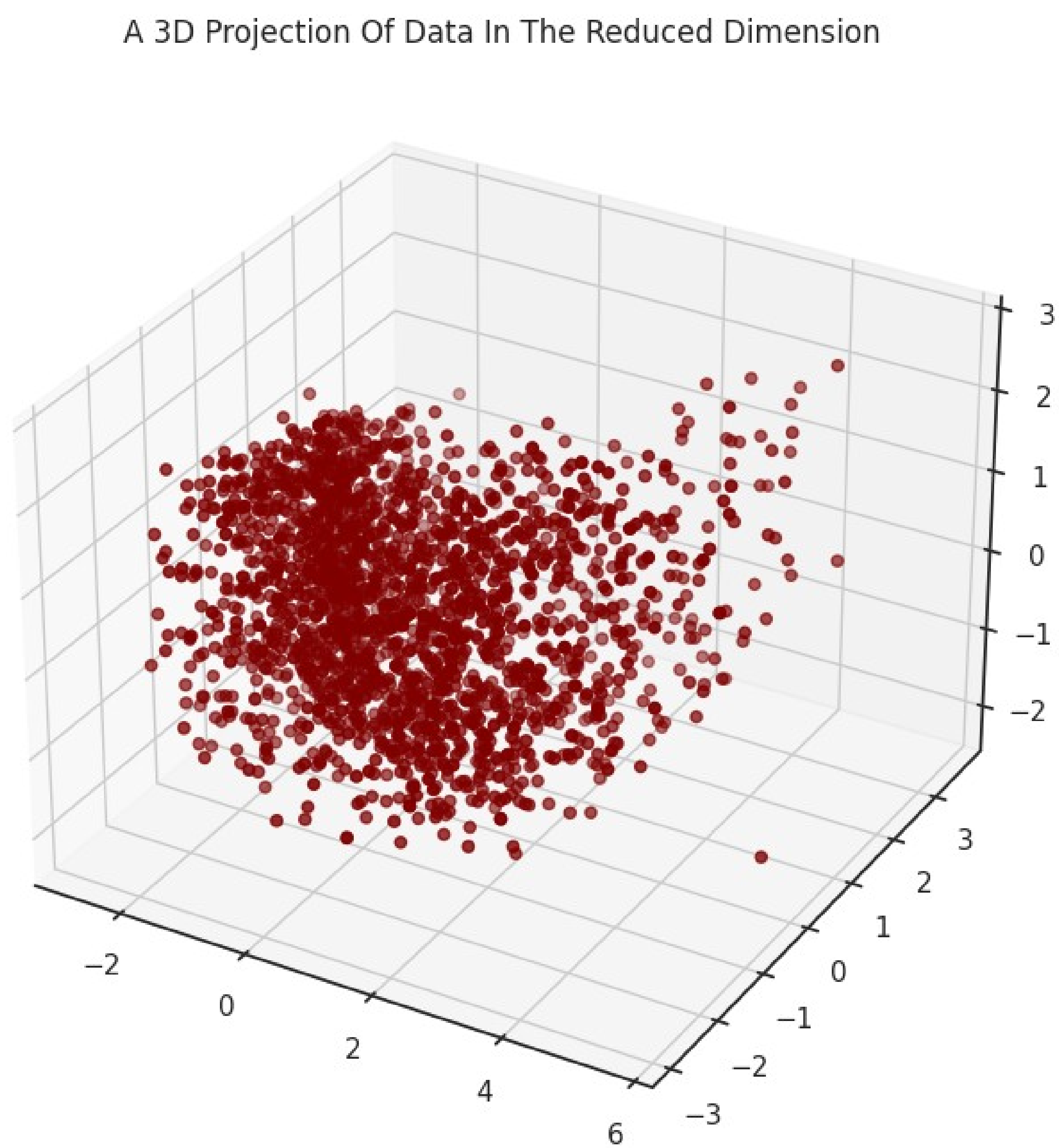
{"summary":"{\n \"name\": \"X_1\",\n \"rows\": 2236,\n \"fields\": [\n   {\n   \"column\": \"Education\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n   \"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\": [\n   1,\n   0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Marital_Status\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n

```
\"max\": 1,\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          0,\n          1\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Income\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.0002236886282312,\n        \"min\": -
2.3461189567805856,\n        \"max\": 5.159342509379425,\n
\"num_unique_values\": 1971,\n        \"samples\": [\n
1.4370929180813004,\n          -0.9894395397240711\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Kids\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1.0002236886282312,\n
\"min\": -1.2643075685916398,\n        \"max\": 2.7248623138696586,\n
\"num_unique_values\": 4,\n        \"samples\": [\n
1.3951390197158926,\n          2.7248623138696586\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Expenses\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
1.0002236886282312,\n        \"min\": -0.9987636117979262,\n
\"max\": 3.1891573371113737,\n        \"num_unique_values\": 1054,\n
\"samples\": [\n          -0.7411732359721557,\n
2.020860342237073\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"TotalAcceptedCmp\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.0002236886282312,\n        \"min\": -
0.43893256174123346,\n        \"max\": 5.455681300441337,\n
\"num_unique_values\": 5,\n        \"samples\": [\n
1.0347209038044092,\n          5.455681300441337\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"NumTotalPurchases\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.0002236886282312,\n        \"min\": -1.9374983150303555,\n
\"max\": 3.7945369947259673,\n        \"num_unique_values\": 39,\n
\"samples\": [\n          2.622075226821265,\n
2.4918016970540755\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Customer_Age\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.0002236886282312,\n        \"min\": -
2.316276170624657,\n        \"max\": 2.4697772449023567,\n
\"num_unique_values\": 56,\n        \"samples\": [\n
1.0168681723316564,\n          0.16221577670183246\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Customer_For\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.0002236886282314,\n        \"min\": -1.750171428817049,\n
\"max\": 1.7078905038986267,\n        \"num_unique_values\": 663,\n
\"samples\": [\n          1.6287360104459074,\n          -
1.4780778575733262\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"X_1"}
```

```
from sklearn.decomposition import PCA
#Initiating PCA to reduce dimentions aka features to 3
pca = PCA(n_components=3)
pca.fit(X_1)
PCA_ds = pd.DataFrame(pca.transform(X_1), columns=(["col1","col2",
"col3"]))
PCA_ds.describe().T
```

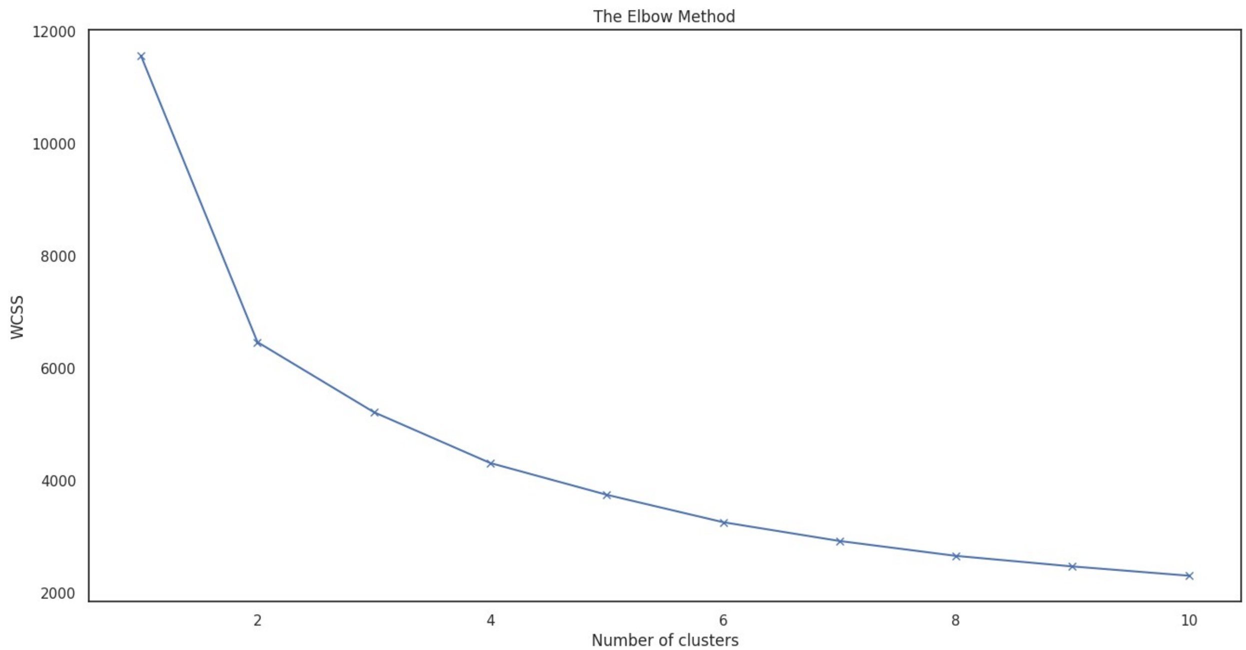
{"summary":"{\n \"name\": \"PCA_ds\",\n \"rows\": 3,\n \"fields\":
[\n   {\n   \"column\": \"count\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0.0,\n   \"min\":
2236.0,\n   \"max\": 2236.0,\n   \"num_unique_values\": 1,\n
\"samples\": [\n   2236.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"mean\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 2.411386389120482e-17,\n
\"min\": -6.037706609768293e-17,\n   \"max\": -
1.2710961283722723e-17,\n   \"num_unique_values\": 3,\n
\"samples\": [\n   -1.2710961283722723e-17\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"std\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0.39435149598200153,\n
\"min\": 1.0277624070027822,\n   \"max\": 1.7278754937921885,\n
\"num_unique_values\": 3,\n   \"samples\": [\n
1.7278754937921885\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"min\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0.36228969021854984,\n   \"min\": -2.8994522638110127,\n
\"max\": -2.221346769507373,\n   \"num_unique_values\": 3,\n
\"samples\": [\n   -2.781512978172441\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"25%\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0.45298342374099754,\n
\"min\": -1.6051333093452809,\n   \"max\": -0.7925354258706272,\n
\"num_unique_values\": 3,\n   \"samples\": [\n   -
1.6051333093452809\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"50%\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0.14741761178643784,\n   \"min\": -0.2561929913947927,\n
\"max\": 0.004862645674875934,\n   \"num_unique_values\": 3,\n
\"samples\": [\n   -0.2561929913947927\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"75%\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0.3533818927704634,\n
\"min\": 0.7325012881503232,\n   \"max\": 1.3836088928916606,\n
\"num_unique_values\": 3,\n   \"samples\": [\n
1.3836088928916606\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"max\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 1.4757189223342124,\n   \"min\": 2.7787218446211974,\n

```
\"max\": 5.653584200318344,\n        \"num_unique_values\": 3,\n
\"samples\": [\n          5.653584200318344\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}

#A 3D Projection Of Data In The Reduced Dimension
x =PCA_ds["col1"]
y =PCA_ds["col2"]
z =PCA_ds["col3"]

#To plot
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x,y,z, c="maroon", marker="o" )
ax.set_title("A 3D Projection Of Data In The Reduced Dimension")

plt.show()
```




```
from sklearn.cluster import AgglomerativeClustering
from sklearn.decomposition import PCA


wcss=[]
for i in range (1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',random_state=42)
    kmeans.fit(PCA_ds)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(16,8))
plt.plot(range(1,11),wcss, 'bx-')
```

```
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



WCSS is the sum of the squared distance between each point and the centroid in a cluster.

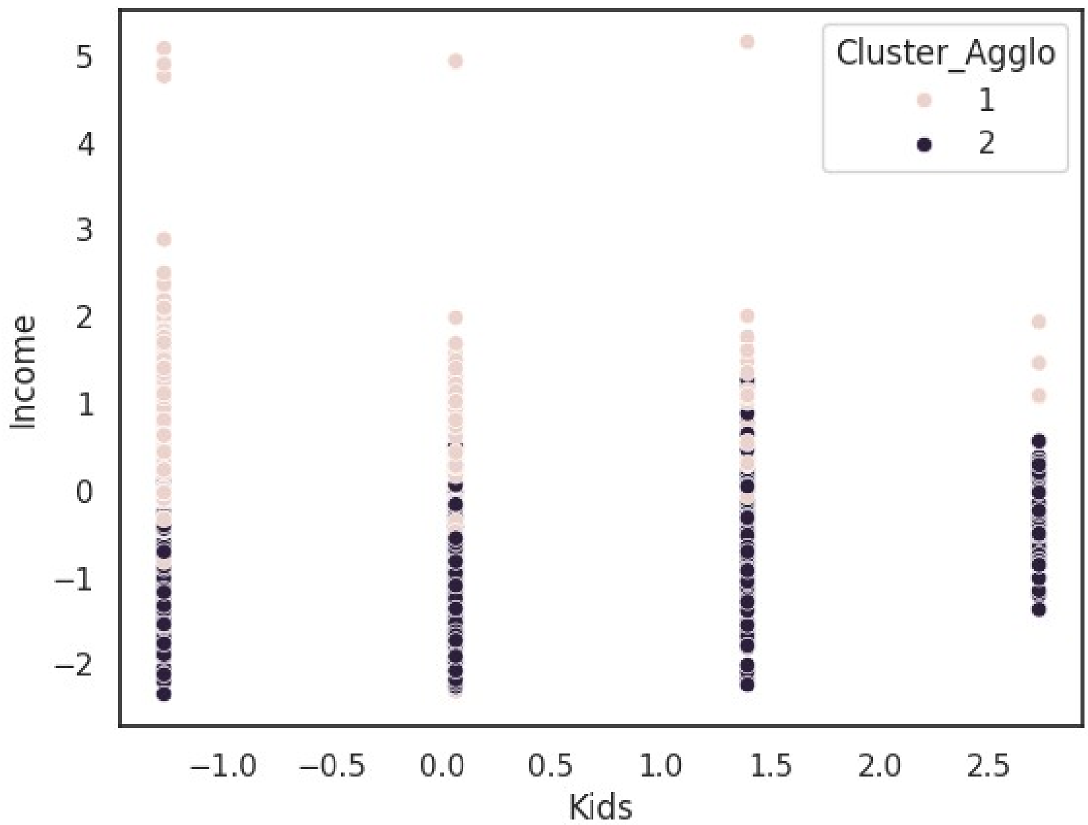
wcss values is more less for k=2 here...so we take k=2

```
#Initiating the Agglomerative Clustering model
AC = AgglomerativeClustering(n_clusters=2)

# fit model and predict clusters
yhat_AC = AC.fit_predict(PCA_ds)
PCA_ds["Clusters"] = yhat_AC

#Adding the Clusters feature to the orignal dataframe.
X_1["Cluster_Agglo"]= yhat_AC + 1

sns.scatterplot(x= X_1['Expenses'],y=
X_1['Income'],hue=X_1['Cluster_Agglo'])
```

```
<Axes: xlabel='Expenses', ylabel='Income'>
```

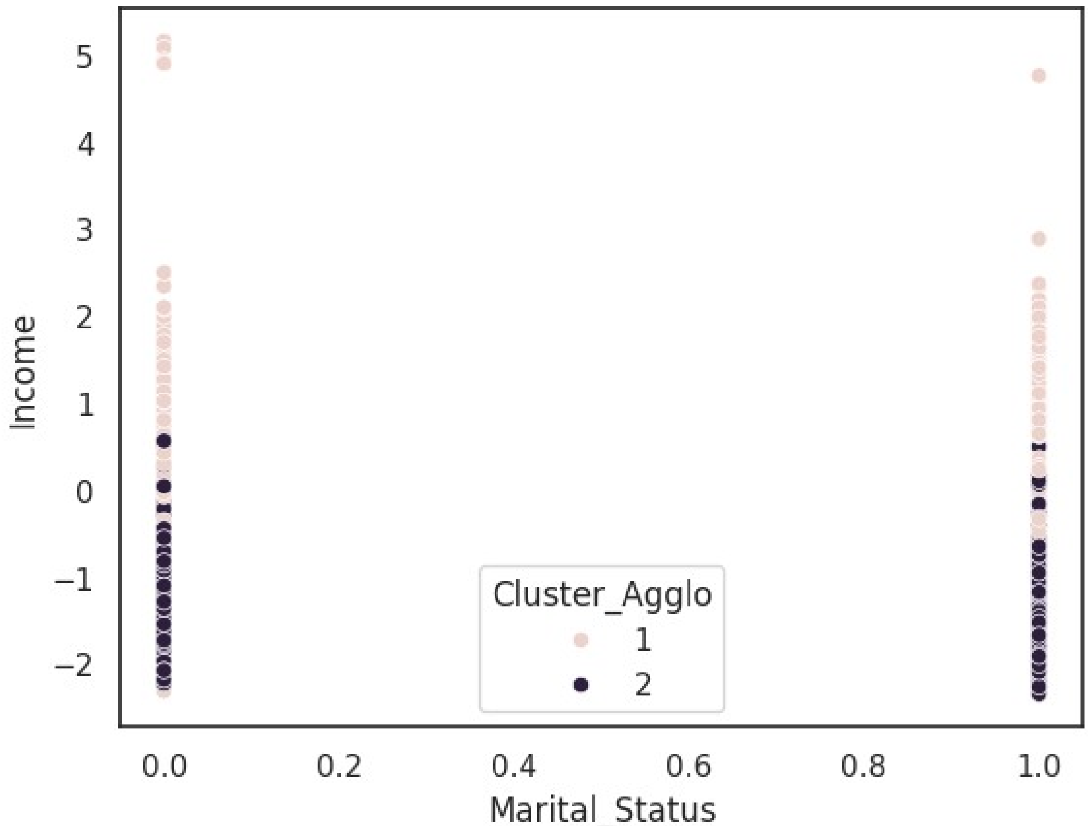

```
sns.scatterplot(data=X_1, x="Kids", y="Income", hue="Cluster_Agglo")
```

```
<Axes: xlabel='Kids', ylabel='Income'>
```



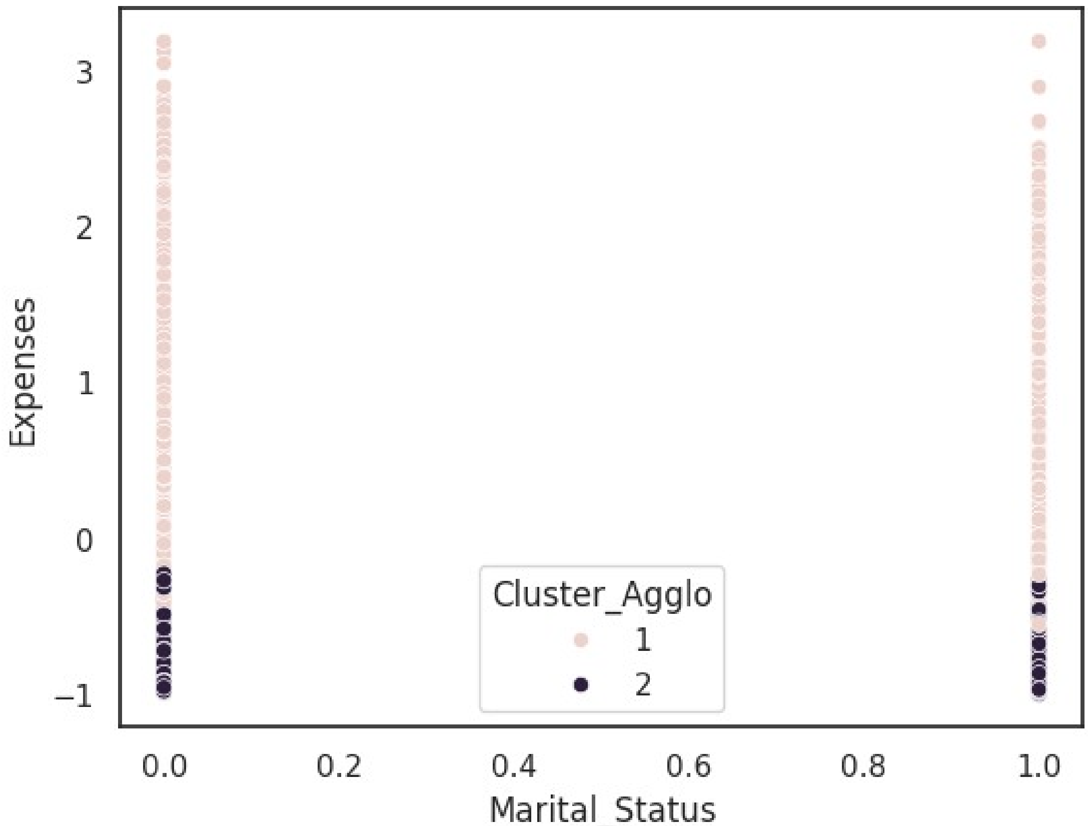

```
sns.scatterplot(data=X_1, x='Marital_Status', y='Income',
hue='Cluster_Agglo')
```

```
<Axes: xlabel='Marital_Status', ylabel='Income'>
```

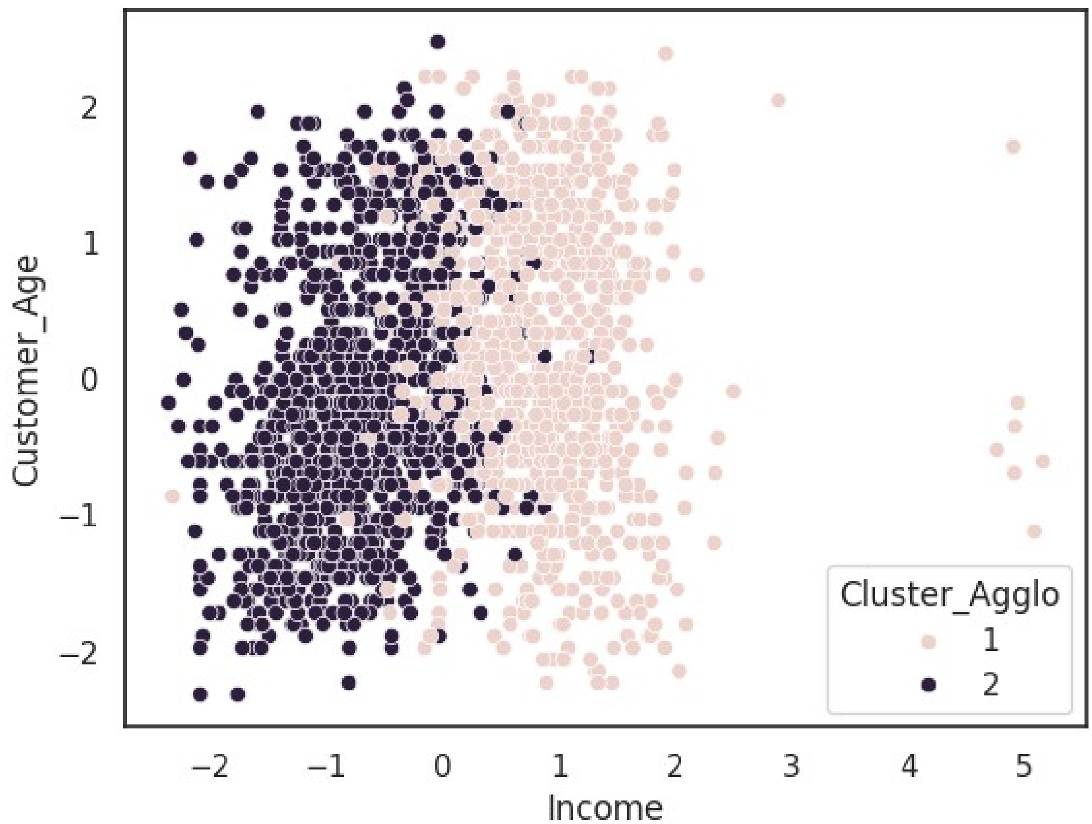



```
sns.scatterplot(x = X_1['Marital_Status'],y =
X_1['Expenses'],hue=X_1['Cluster_Agglo'])
```

```
<Axes: xlabel='Marital_Status', ylabel='Expenses'>
```

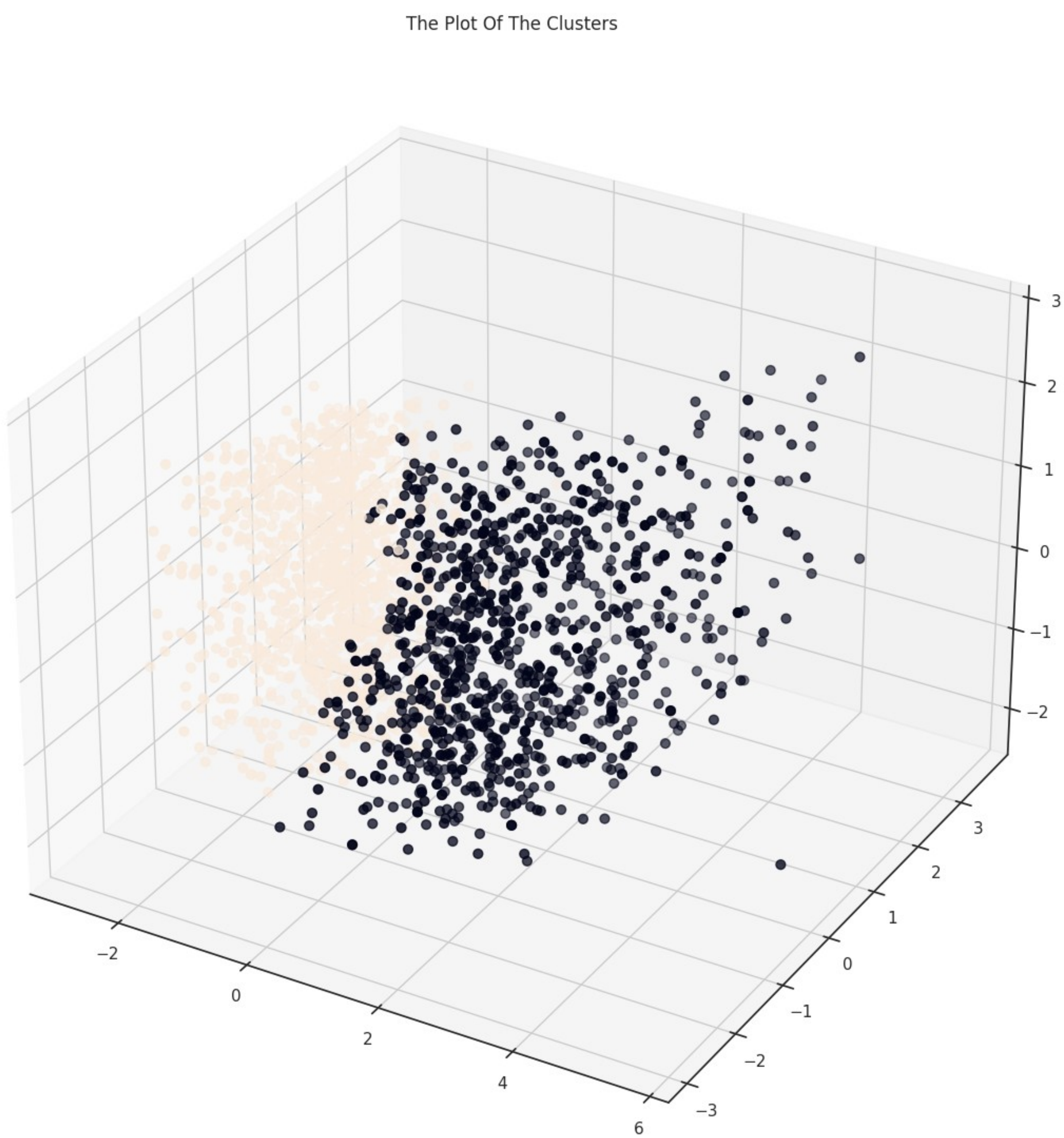



```
sns.scatterplot(x = X_1['Income'],y =
X_1['Customer_Age'],hue=X_1['Cluster_Agglo'])

<Axes: xlabel='Income', ylabel='Customer_Age'>
```




```
sns.countplot(x=X_1["Cluster_Agglo"])
plt.title("Distribution Of The Clusters")
plt.show()
```


```
#Plotting the clusters
fig = plt.figure(figsize=(16,14))
ax = plt.subplot(111, projection='3d', label="bla")

ax.scatter(x, y, z, s=40, c=PCA_ds["Clusters"], marker='o')
ax.set_title("The Plot Of The Clusters")

plt.show()
```



Conclusions:

Cluster 1:

People with less expenses

people who are married and parents of more than 3 kids

people which low income

Cluster 2:

people with more expenses

people who are single or parents who have less than 3 kids

people with high income

Age is not the criteria but it is observed to some extent that people who are older fall in this group

So, the customers falling in cluster 2 likes to spend more...so the Firm's can target people falling in cluster 2 for the sale of their Products....

Thanks you!!!!