o9gr05fv6

November 26, 2023

0.0.1 ANALYSIS OF THE MAVEN MARKETING CAMPAIGN DATASET

[ ]:
```
import pandas as pd
```

[ ]:
```
df = pd.read_csv("Marketing+Data/marketing_data.csv")
df.head()
```

[ ]:
```
      ID  Year_Birth   Education Marital_Status   Income   Kidhome  Teenhome  \
0   1826        1970  Graduation       Divorced  84835.0         0         0
1      1        1961  Graduation         Single  57091.0         0         0
2  10476        1958  Graduation        Married  67267.0         0         1
3   1386        1967  Graduation       Together  32474.0         1         1
4   5371        1989  Graduation         Single  21474.0         1         0

  Dt_Customer  Recency  MntWines  …  NumStorePurchases  NumWebVisitsMonth  \
0  2014-06-16        0       189  …                  6                  1
1  2014-06-15        0       464  …                  7                  5
2  2014-05-13        0       134  …                  5                  2
3  2014-05-11        0        10  …                  2                  7
4  2014-04-08        0         6  …                  2                  7

   AcceptedCmp3  AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  \
0             0             0             0             0             0
1             0             0             0             0             1
2             0             0             0             0             0
3             0             0             0             0             0
4             1             0             0             0             0

   Response  Complain    Country
0         1         0      Spain
1         1         0     Canada
2         0         0        USA
3         0         0  Australia
4         1         0      Spain

[5 rows x 28 columns]
```

[3]:
```
df.describe()
```

[3]:

	ID	Year_Birth	Income	Kidhome	Teenhome	\
count	2240.000000	2240.000000	2216.000000	2240.000000	2240.000000	
mean	5592.159821	1968.805804	52247.251354	0.444196	0.506250	
std	3246.662198	11.984069	25173.076661	0.538398	0.544538	
min	0.000000	1893.000000	1730.000000	0.000000	0.000000	
25%	2828.250000	1959.000000	35303.000000	0.000000	0.000000	
50%	5458.500000	1970.000000	51381.500000	0.000000	0.000000	
75%	8427.750000	1977.000000	68522.000000	1.000000	1.000000	
max	11191.000000	1996.000000	666666.000000	2.000000	2.000000	

	Recency	MntWines	MntFruits	MntMeatProducts	\
count	2240.000000	2240.000000	2240.000000	2240.000000	
mean	49.109375	303.935714	26.302232	166.950000	
std	28.962453	336.597393	39.773434	225.715373	
min	0.000000	0.000000	0.000000	0.000000	
25%	24.000000	23.750000	1.000000	16.000000	
50%	49.000000	173.500000	8.000000	67.000000	
75%	74.000000	504.250000	33.000000	232.000000	
max	99.000000	1493.000000	199.000000	1725.000000	

	MntFishProducts	…	NumCatalogPurchases	NumStorePurchases	\
count	2240.000000	…	2240.000000	2240.000000	
mean	37.525446	…	2.662054	5.790179	
std	54.628979	…	2.923101	3.250958	
min	0.000000	…	0.000000	0.000000	
25%	3.000000	…	0.000000	3.000000	
50%	12.000000	…	2.000000	5.000000	
75%	50.000000	…	4.000000	8.000000	
max	259.000000	…	28.000000	13.000000	

	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5	\
count	2240.000000	2240.000000	2240.000000	2240.000000	
mean	5.316518	0.072768	0.074554	0.072768	
std	2.426645	0.259813	0.262728	0.259813	
min	0.000000	0.000000	0.000000	0.000000	
25%	3.000000	0.000000	0.000000	0.000000	
50%	6.000000	0.000000	0.000000	0.000000	
75%	7.000000	0.000000	0.000000	0.000000	
max	20.000000	1.000000	1.000000	1.000000	

	AcceptedCmp1	AcceptedCmp2	Response	Complain
count	2240.000000	2240.000000	2240.000000	2240.000000
mean	0.064286	0.013393	0.149107	0.009375
std	0.245316	0.114976	0.356274	0.096391
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000

```
75%        0.000000      0.000000     0.000000     0.000000
max        1.000000      1.000000     1.000000     1.000000

[8 rows x 24 columns]
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 28 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  Response             2240 non-null   int64
 26  Complain             2240 non-null   int64
 27  Country              2240 non-null   object
dtypes: float64(1), int64(23), object(4)
memory usage: 490.1+ KB
```

Answering the featured questions on the mavens analytics website

Are there any null values or outliers? How will you handle them?

What factors are significantly related to the number of web purchases?

Which marketing campaign was the most successful?

What does the average customer look like?

Which products are performing best?

Which channels are underperforming?

Are there any null values or outliers? How will you handle them?

[5]:
```python
# The income column has leading spaces so we rename to get rid of the spaces
df.rename(columns = {" Income ":"Income"},inplace = True)
```

[6]:
```python
df.isna().sum()
```

[6]:
```
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
Response                0
Complain                0
Country                 0
dtype: int64
```

[7]:
```
#here we will make a copy of the dataset and work with the copy
df1 = df.copy()
```

[8]:
```
# we can see that only the Income field has empty values so we fill with␣
↪average value
df1["Income"].fillna(df1.Income.mean(), inplace = True)
df1.info()
```

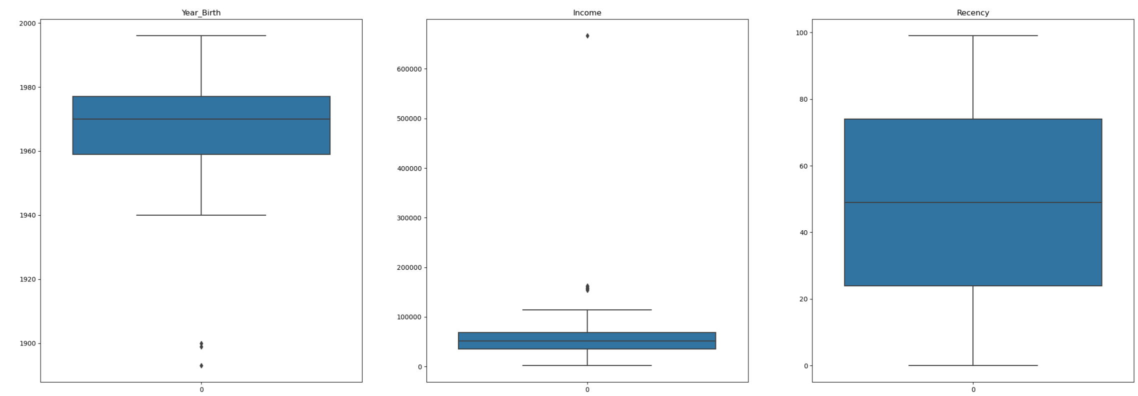
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 28 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   ID                   2240 non-null   int64
 1   Year_Birth           2240 non-null   int64
 2   Education            2240 non-null   object
 3   Marital_Status       2240 non-null   object
 4   Income               2240 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  Response             2240 non-null   int64
 26  Complain             2240 non-null   int64
 27  Country              2240 non-null   object
dtypes: float64(1), int64(23), object(4)
memory usage: 490.1+ KB
```

[9]:
```
df1.isna().sum()
```

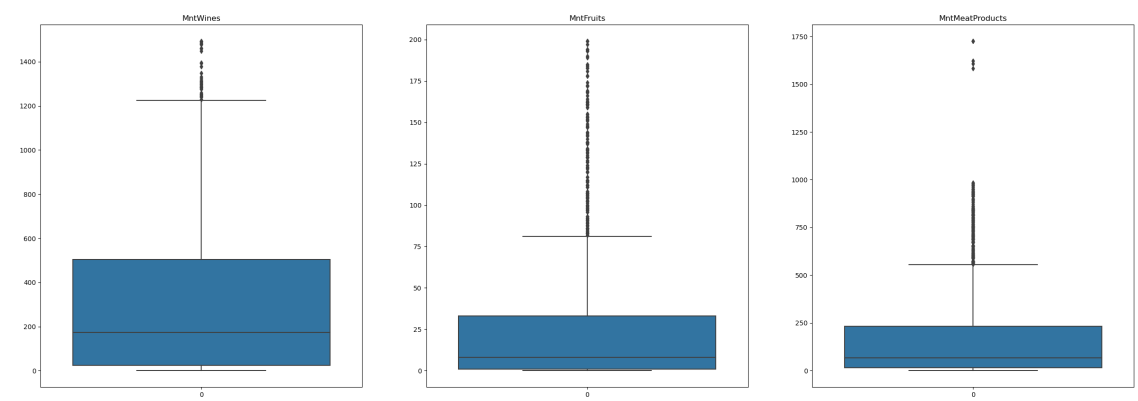
[9]:
```
ID                     0
Year_Birth             0
Education              0
Marital_Status         0
Income                 0
Kidhome                0
Teenhome               0
Dt_Customer            0
Recency                0
MntWines               0
MntFruits              0
MntMeatProducts        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
Response               0
Complain               0
Country                0
dtype: int64
```

[10]:
```
import matplotlib.pyplot as plt
import seaborn as sb
```

[11]:
```
plt.figure(figsize=(30,10))
plt.subplot(1,3,1)
plt.title("Year_Birth")
sb.boxplot(data=df1["Year_Birth"])

plt.subplot(1,3,2)
plt.title("Income")
sb.boxplot(data=df1["Income"])

plt.subplot(1,3,3)
plt.title("Recency")
sb.boxplot(data=df1["Recency"])

plt.show()
```

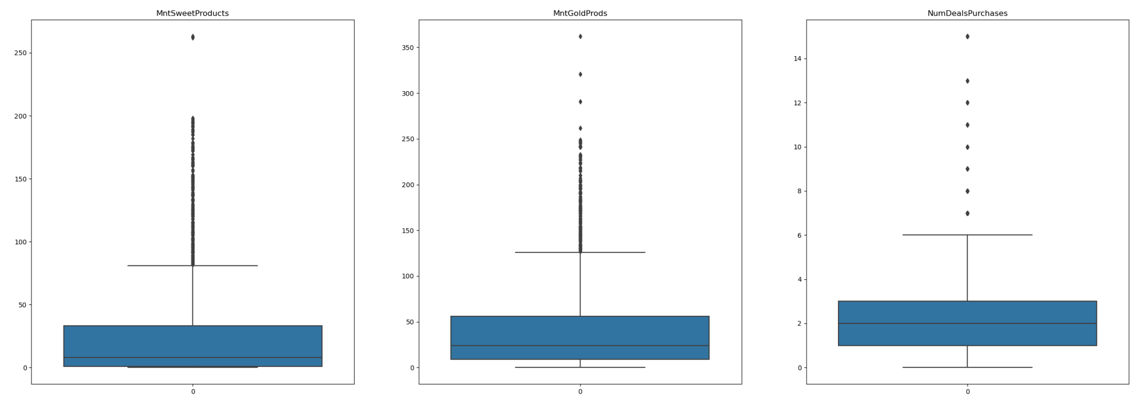


[12]:
```
plt.figure(figsize=(30,10))
plt.subplot(1,3,1)
plt.title("MntWines")
sb.boxplot(data=df1["MntWines"])

plt.subplot(1,3,2)
plt.title("MntFruits")
sb.boxplot(data=df1["MntFruits"])

plt.subplot(1,3,3)
plt.title("MntMeatProducts")
sb.boxplot(data=df1["MntMeatProducts"])

plt.show()
```

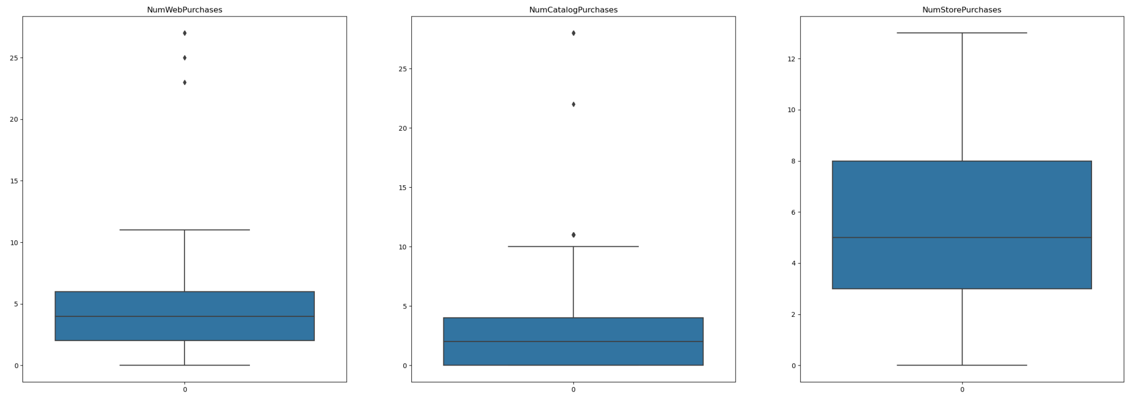


[13]:
```
plt.figure(figsize=(30,10))
plt.subplot(1,3,1)
plt.title("MntSweetProducts")
sb.boxplot(data=df1["MntSweetProducts"])
```

```
plt.subplot(1,3,2)
plt.title("MntGoldProds")
sb.boxplot(data=df1["MntGoldProds"])

plt.subplot(1,3,3)
plt.title("NumDealsPurchases")
sb.boxplot(data=df1["NumDealsPurchases"])

plt.show()
```

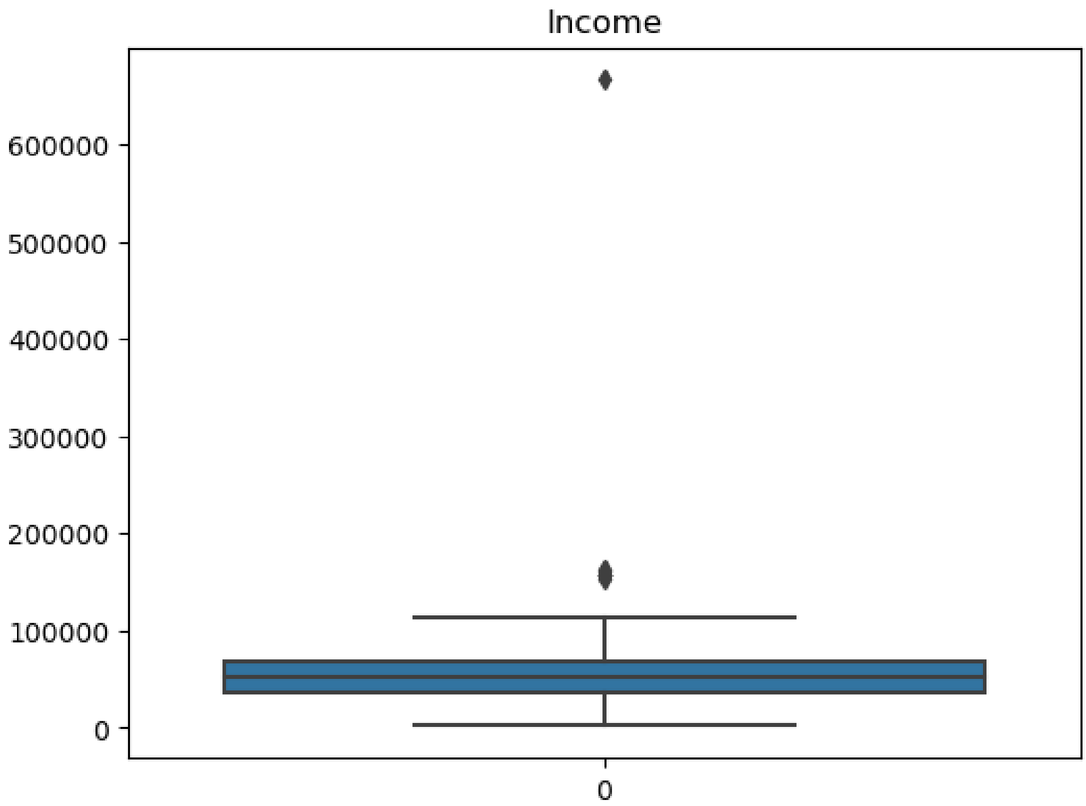


[14]:
```
plt.figure(figsize=(30,10))
plt.subplot(1,3,1)
plt.title("NumWebPurchases")
sb.boxplot(data=df1["NumWebPurchases"])

plt.subplot(1,3,2)
plt.title("NumCatalogPurchases")
sb.boxplot(data=df1["NumCatalogPurchases"])

plt.subplot(1,3,3)
plt.title("NumStorePurchases")
sb.boxplot(data=df1["NumStorePurchases"])

plt.show()
```



[15]:
```
plt.title("Income")
sb.boxplot(data=df1["Income"])
```

[15]: <AxesSubplot:title={'center':'Income'}>



[16]:
```
numerical_attributes = ['Year_Birth', 'Income', 'Recency', 'MntWines',␣
 ↪'MntFruits', 'MntMeatProducts',
```

```
                        'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',␣
↪'NumDealsPurchases',
                        'NumWebPurchases', 'NumCatalogPurchases',␣
↪'NumStorePurchases', 'NumWebVisitsMonth']
```

[17]:
```
def iqr(data,column):
    #find the iqr by subtracting 25-quantile from the 75-quantile
    IQR = data[column].quantile(0.75) - data[column].quantile(0.25)
    return IQR
```

[18]:
```
def outliers(data,column_list):
    myDict = {}
    for column_name in column_list:
        IQR = data[column_name].quantile(0.75) - data[column_name].quantile(0.
↪25)
        lower_outlier = data[column_name].quantile(0.25) - (1.5 *IQR)
        higher_outlier = data[column_name].quantile(0.75) + (1.5 *IQR)
        myDict[column_name] = [lower_outlier,higher_outlier]
    return myDict
```

[19]:
```
limits = outliers(df1,numerical_attributes)
limits
```

[19]:
```
{'Year_Birth': [1932.0, 2004.0],
 'Income': [-13587.75, 117416.25],
 'Recency': [-51.0, 149.0],
 'MntWines': [-697.0, 1225.0],
 'MntFruits': [-47.0, 81.0],
 'MntMeatProducts': [-308.0, 556.0],
 'MntFishProducts': [-67.5, 120.5],
 'MntSweetProducts': [-47.0, 81.0],
 'MntGoldProds': [-61.5, 126.5],
 'NumDealsPurchases': [-2.0, 6.0],
 'NumWebPurchases': [-4.0, 12.0],
 'NumCatalogPurchases': [-6.0, 10.0],
 'NumStorePurchases': [-4.5, 15.5],
 'NumWebVisitsMonth': [-3.0, 13.0]}
```

[20]:
```
Year_Birth_outlier = df1[df1["Year_Birth"] < (limits['Year_Birth'][0])]
len(Year_Birth_outlier)
```

[20]:
```
3
```

[21]:
```
def count_outliers(data,col_list):
    Dict = {}
    for col in col_list:
         #consider lower limit for Year and upper limits for the rest
```

```
        if col == "Year_Birth":
            numOfOutliers = data[data["Year_Birth"] < (limits['Year_Birth'][0])]
            Dict["Year_Birth"] = len(numOfOutliers)
        else:
            numOfOutliers = data[data[col] > limits[col][1]]
            Dict[col] = len(numOfOutliers)
    return Dict
```

[22]:
```
count_outliers(df1,numerical_attributes)
```

[22]:
```
{'Year_Birth': 3,
 'Income': 8,
 'Recency': 0,
 'MntWines': 35,
 'MntFruits': 227,
 'MntMeatProducts': 175,
 'MntFishProducts': 223,
 'MntSweetProducts': 248,
 'MntGoldProds': 207,
 'NumDealsPurchases': 86,
 'NumWebPurchases': 4,
 'NumCatalogPurchases': 23,
 'NumStorePurchases': 0,
 'NumWebVisitsMonth': 8}
```

[23]:
```
import numpy as np
```

[24]:
```
upper_limit_inc = df["Year_Birth"].quantile(1)
lower_limit_inc = df["Year_Birth"].quantile(0.05)
lower_limit_inc
```

[24]:
```
1950.0
```

[25]:
```
upper_limit_fr = df["MntFruits"].quantile(0.89)
lower_limit_fr = df["MntFruits"].quantile(0.05)
upper_limit_fr
```

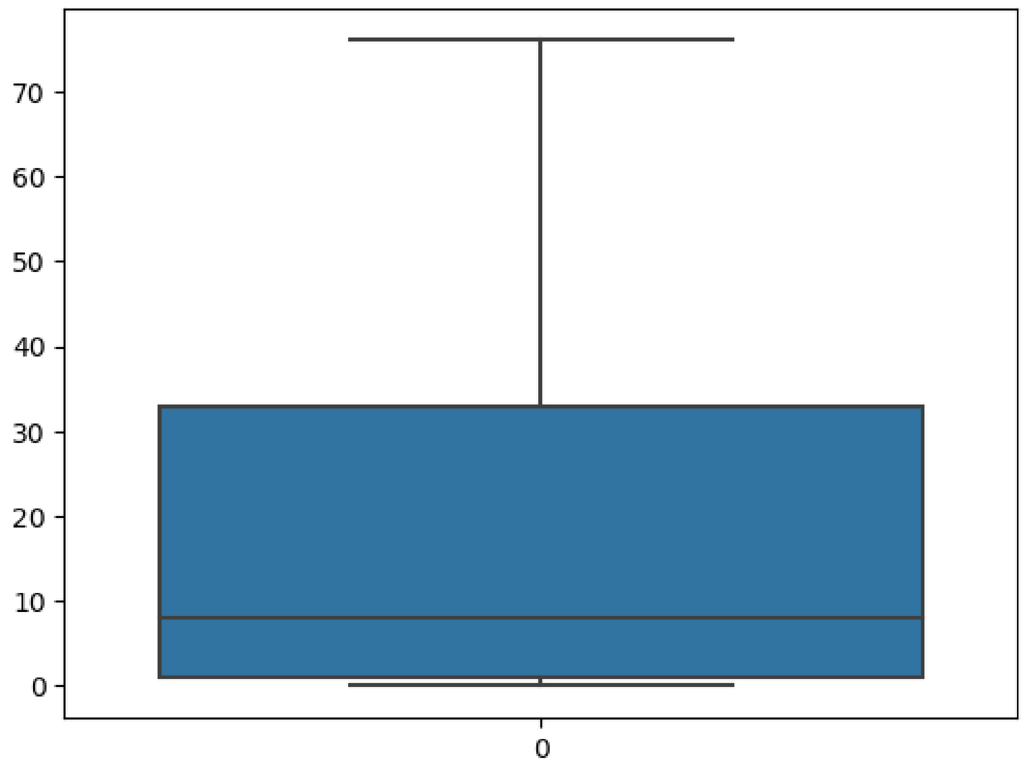
[25]:
```
80.0
```

[26]:
```
# Testing Capping --> Windsorization on MntFruits Column
df1["MntFruits"] = np.where(df1["MntFruits"] >= upper_limit_fr,
            upper_limit_fr,
            np.where(df1["MntFruits"] <= lower_limit_fr,
            lower_limit_fr,
            df1["MntFruits"]))
```

[27]:
```
upper_limit_swt = df["MntSweetProducts"].quantile(0.88)
lower_limit_swt = df["MntSweetProducts"].quantile(0.05)
upper_limit_swt
# Windsorization on MntSweetProducts Column
df1["MntSweetProducts"] = np.where(df1["MntSweetProducts"] >= upper_limit_swt,
            upper_limit_swt,
            np.where(df1["MntSweetProducts"] <= lower_limit_swt,
            lower_limit_swt,
            df1["MntSweetProducts"]))
```

[28]:
```
sb.boxplot(data=df1["MntSweetProducts"])
```

[28]: <AxesSubplot:>



[29]:
```
# A function that winsorizes the remaining numerical attributes
def winsor_all(col_list):
    for col in col_list:
        upper_limit = df[col].quantile(0.90)
        lower_limit = df[col].quantile(0.05)
        #winsorize the column
        df1[col] = np.where(df1[col] >= upper_limit,
            upper_limit,
```

```
            np.where(df1[col] <= lower_limit,
            lower_limit,
            df1[col]))
    return "Winsorization Successful"
```

The winsorization function was significant on only a few columns because the data of each column is distributed differently so I check for columns which where outliers were not significantly impacted to winsorize them individually using thresholds tailored for each column.

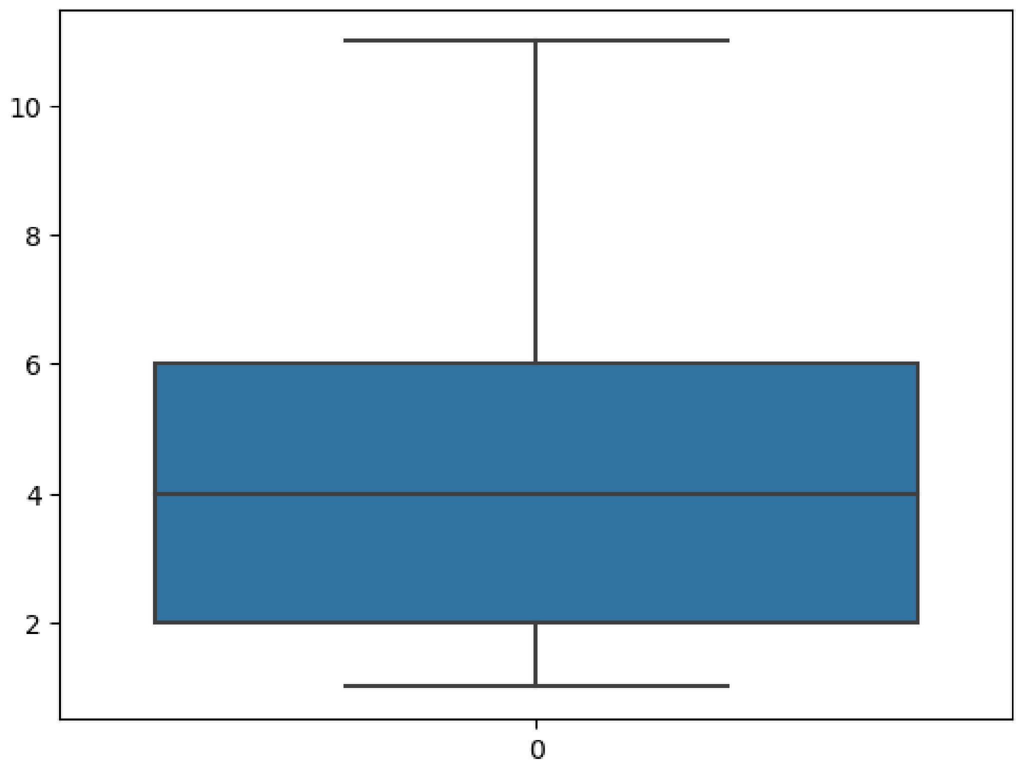
[30]:
```
upper_limit_gold = df["MntGoldProds"].quantile(0.88)
lower_limit_gold = df["MntGoldProds"].quantile(0.05)
upper_limit_gold
```

[30]: 108.0

[31]:
```
upper_limit_wp = df["NumWebPurchases"].quantile(0.98)
lower_limit_wp = df["NumWebPurchases"].quantile(0.05)
upper_limit_wp

# Windsorization on MntSweetProducts Column
df1["NumWebPurchases"] = np.where(df1["NumWebPurchases"] >= upper_limit_wp,
            upper_limit_wp,
            np.where(df1["NumWebPurchases"] <= lower_limit_wp,
            lower_limit_wp,
            df1["NumWebPurchases"]))
```

[32]:
```
sb.boxplot(data=df1["NumWebPurchases"])
```

[32]: <AxesSubplot:>



Answer the Question : Are there any null values or outliers? How will you handle them?

Only the Income column had null values. These values were imputed using the mean. Most of the columns had outliers and this outliers were detected using box plot and handled using the winsorization technique

Q2. What factors are significantly related to the number of web purchases?

[33]:
```
def significant_corr(corr_col,data,col_list):
    myDict = {}
    for col in col_list:
        corr_num = data[corr_col].corr(data[col])
        corr_num = np.around(corr_num, 1)
        myDict[col] = corr_num
    return myDict
```

[34]:
```
web_purchase = significant_corr("NumWebPurchases",df1,numerical_attributes)
web_purchase = pd.DataFrame(list(web_purchase.items()))
web_purchase
```

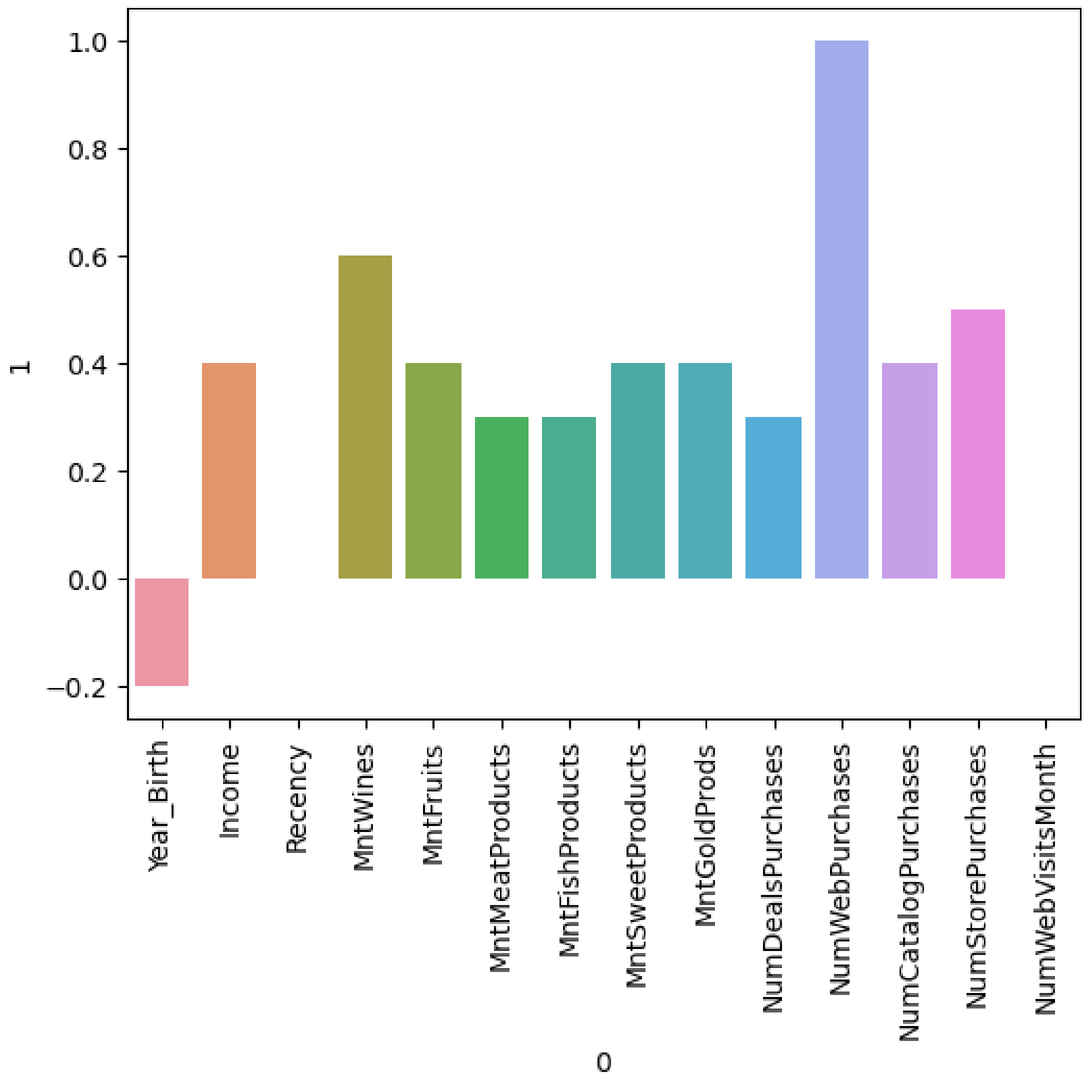
[34]:
```
                       0    1
0             Year_Birth -0.2
1                 Income  0.4
2                Recency -0.0
3               MntWines  0.6
4              MntFruits  0.4
5        MntMeatProducts  0.3
6        MntFishProducts  0.3
7       MntSweetProducts  0.4
8           MntGoldProds  0.4
9      NumDealsPurchases  0.3
10       NumWebPurchases  1.0
11   NumCatalogPurchases  0.4
12     NumStorePurchases  0.5
13     NumWebVisitsMonth -0.0
```

[35]:
```
web_purchase.shape
```

[35]:
```
(14, 2)
```

[36]:
```
sb.barplot(x=web_purchase[0],y=web_purchase[1])
plt.xticks(rotation=90)
```

[36]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13]),
 [Text(0, 0, 'Year_Birth'),
  Text(1, 0, 'Income'),
  Text(2, 0, 'Recency'),
  Text(3, 0, 'MntWines'),
  Text(4, 0, 'MntFruits'),
  Text(5, 0, 'MntMeatProducts'),
  Text(6, 0, 'MntFishProducts'),
  Text(7, 0, 'MntSweetProducts'),
  Text(8, 0, 'MntGoldProds'),
  Text(9, 0, 'NumDealsPurchases'),
  Text(10, 0, 'NumWebPurchases'),
  Text(11, 0, 'NumCatalogPurchases'),
  Text(12, 0, 'NumStorePurchases'),
  Text(13, 0, 'NumWebVisitsMonth')])
```



Answer to Q2

From the graph above it can be seen that the the num of web purchases is strongly correlated to the Income of customers.

Which marketing campaign was the most successful?

[37]:
```
marketing_df = df1.copy()
```

[38]:
```
marketing_df = marketing_df.
 ↪drop(columns=["ID","Income",'MntWines',"Year_Birth","Education","Kidhome","Teenhome","Dt_Cu
                                "Recency",␣
 ↪'Marital_Status','MntFruits','MntMeatProducts', 'MntFishProducts',
```

```
                                'MntSweetProducts','MntGoldProds','Complain',␣
  ↪'Country'], axis=1)
marketing_df
```

[38]:
```
      NumDealsPurchases  NumWebPurchases  NumCatalogPurchases  \
0                     1              4.0                    4
1                     1              7.0                    3
2                     1              3.0                    2
3                     1              1.0                    0
4                     2              3.0                    1
…                   …                …                    …
2235                  2              5.0                    2
2236                  1              1.0                    0
2237                  2              6.0                    1
2238                  1              5.0                    4
2239                  1              8.0                    5

      NumStorePurchases  NumWebVisitsMonth  AcceptedCmp3  AcceptedCmp4  \
0                     6                  1             0             0
1                     7                  5             0             0
2                     5                  2             0             0
3                     2                  7             0             0
4                     2                  7             1             0
…                   …                …           …           …
2235                 11                  4             0             0
2236                  3                  8             0             0
2237                  5                  8             0             0
2238                 10                  3             0             0
2239                  4                  7             0             1

      AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response
0                0             0             0         1
1                0             0             1         1
2                0             0             0         0
3                0             0             0         0
4                0             0             0         1
…              …           …           …       …
2235             0             0             0         0
2236             0             0             0         0
2237             0             0             0         0
2238             0             0             0         0
2239             1             0             0         1

[2240 rows x 11 columns]
```

[39]:
```
campaign_acceptance = ["AcceptedCmp1","AcceptedCmp2","AcceptedCmp3",
                       "AcceptedCmp4","AcceptedCmp5","Response"]
```

```python
def sum_accepted(data,accepted_list):
    myDict = {}
    for accepted in accepted_list:
        total = data[accepted].sum()
        myDict[accepted] = total
    return myDict
```

[40]:
```python
campaign_data = sum_accepted(marketing_df,campaign_acceptance)
campaign_data
```

[40]:
```
{'AcceptedCmp1': 144,
 'AcceptedCmp2': 30,
 'AcceptedCmp3': 163,
 'AcceptedCmp4': 167,
 'AcceptedCmp5': 163,
 'Response': 334}
```

[41]:
```python
campaign_df = pd.DataFrame(campaign_data.items())
campaign_df
```

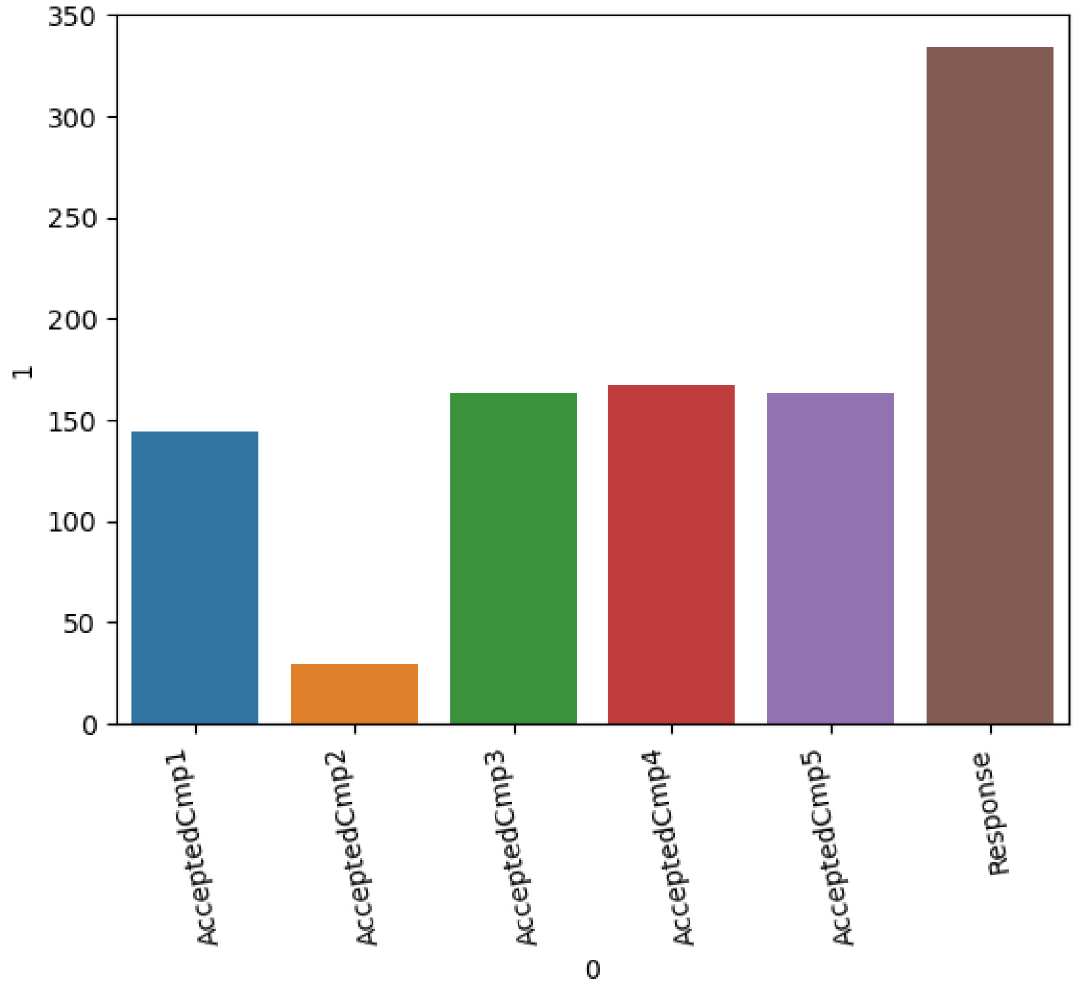
[41]:

	0	1
0	AcceptedCmp1	144
1	AcceptedCmp2	30
2	AcceptedCmp3	163
3	AcceptedCmp4	167
4	AcceptedCmp5	163
5	Response	334

[42]:
```python
sb.barplot(campaign_df[0],campaign_df[1])
plt.xticks(rotation=100)
```

```
/Users/pc/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[42]:
```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'AcceptedCmp1'),
  Text(1, 0, 'AcceptedCmp2'),
  Text(2, 0, 'AcceptedCmp3'),
  Text(3, 0, 'AcceptedCmp4'),
  Text(4, 0, 'AcceptedCmp5'),
  Text(5, 0, 'Response')])
```



Answer Q3

From the bar graph above, It can be seen that the best performing marketing campaign was the last marketing campaign and the worst performing marketing campaign was the second campaign.

Digging deeper into the last marketing campain

[43]:
```
last_campaign_df = df1[df1["Response"] ==␣
↪1][["MntWines","MntFruits",'MntMeatProducts','MntFishProducts',
'MntSweetProducts','MntGoldProds','NumWebPurchases','NumCatalogPurchases',
'NumStorePurchases',"Country"]]
```

What products were bought more on the last marketing campaign

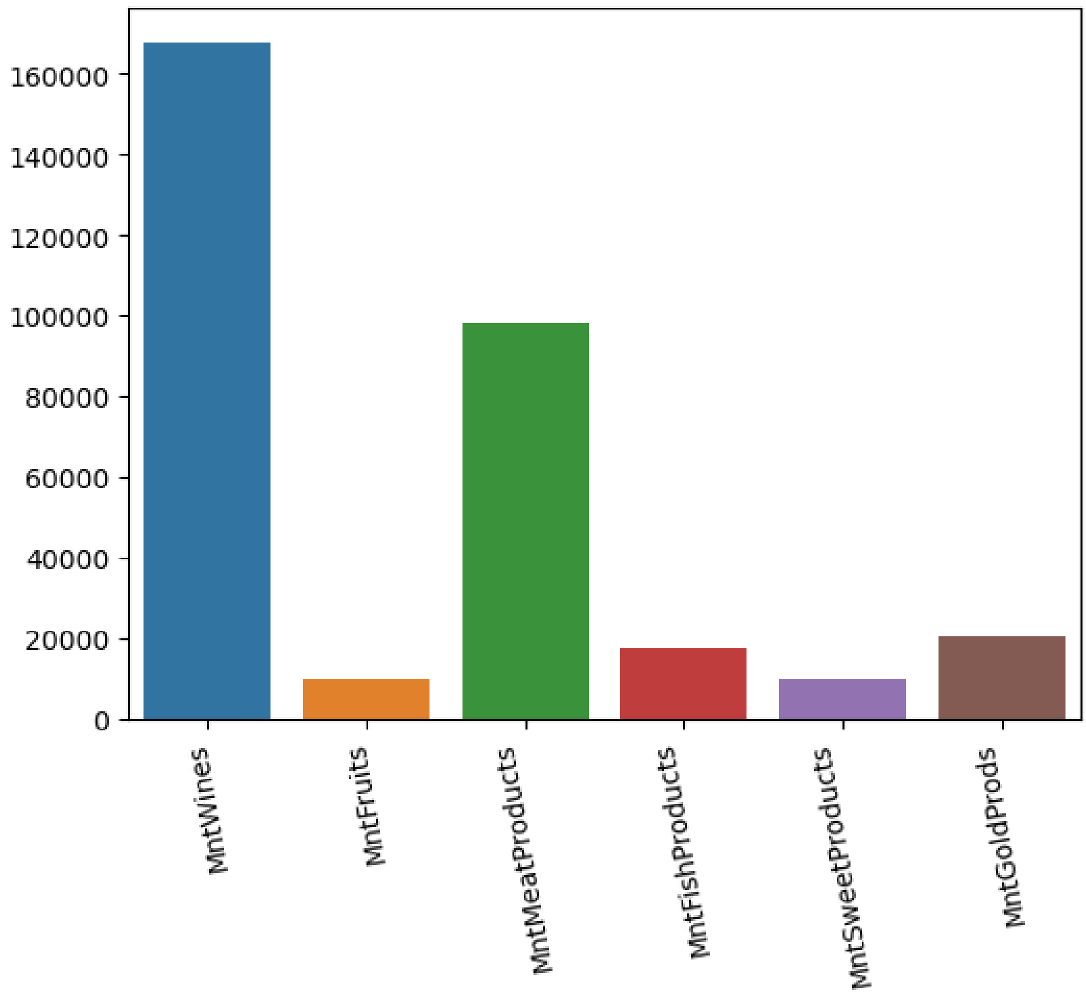
[44]:
```
last_campaign_df[["MntWines","MntFruits",'MntMeatProducts','MntFishProducts',
'MntSweetProducts','MntGoldProds']].sum()
```

[44]:
```
MntWines            167903.0
MntFruits            10142.0
MntMeatProducts      98314.0
MntFishProducts      17385.0
MntSweetProducts     10048.0
MntGoldProds         20523.0
dtype: float64
```

[45]:
```
last_campaign_prd = ["MntWines","MntFruits","MntMeatProducts","MntFishProducts",
                     "MntSweetProducts","MntGoldProds"]
last_campaign_totals = [167903.0,10142.0,98314.0,17385.0,10048.0,20523.0]
```

[46]:
```
sb.barplot(x=last_campaign_prd, y=last_campaign_totals)
plt.xticks(rotation=100)
```

[46]:
```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'MntWines'),
  Text(1, 0, 'MntFruits'),
  Text(2, 0, 'MntMeatProducts'),
  Text(3, 0, 'MntFishProducts'),
  Text(4, 0, 'MntSweetProducts'),
  Text(5, 0, 'MntGoldProds')])
```



The above bar chat shows that in the last campaign, Wine, Meat and Gold products were the top 3 performing products respectively

Which sales channel was used by the the customers who responded to the last campaign?

[47]:
```
last_campaign_channel = 
 ↪last_campaign_df[['NumWebPurchases','NumCatalogPurchases',
 'NumStorePurchases']]
last_campaign_channel.sum()
```

[47]:
```
NumWebPurchases        1695.0
NumCatalogPurchases    1404.0
NumStorePurchases      2036.0
dtype: float64
```

What does the average customer look like?

```
[48]: df1["Age"] = 2023 - df["Year_Birth"]
      df1["Children"] = df["Kidhome"] + df["Teenhome"]
      df1
```

```
[48]:          ID  Year_Birth   Education Marital_Status   Income  Kidhome  \
0      1826        1970  Graduation       Divorced  84835.0        0
1         1        1961  Graduation         Single  57091.0        0
2     10476        1958  Graduation        Married  67267.0        0
3      1386        1967  Graduation       Together  32474.0        1
4      5371        1989  Graduation         Single  21474.0        1
…       …           …         …              …       …        …
2235  10142        1976         PhD       Divorced  66476.0        0
2236   5263        1977    2n Cycle        Married  31056.0        1
2237     22        1976  Graduation       Divorced  46310.0        1
2238    528        1978  Graduation        Married  65819.0        0
2239   4070        1969         PhD        Married  94871.0        0

      Teenhome Dt_Customer  Recency  MntWines  …  AcceptedCmp3  \
0            0  2014-06-16        0       189  …             0
1            0  2014-06-15        0       464  …             0
2            1  2014-05-13        0       134  …             0
3            1  2014-05-11        0        10  …             0
4            0  2014-04-08        0         6  …             1
…          …         …        …        …  …           …
2235         1  2013-03-07       99       372  …             0
2236         0  2013-01-22       99         5  …             0
2237         0  2012-12-03       99       185  …             0
2238         0  2012-11-29       99       267  …             0
2239         2  2012-09-01       99       169  …             0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
0                0             0             0             0         1
1                0             0             0             1         1
2                0             0             0             0         0
3                0             0             0             0         0
4                0             0             0             0         1
…              …             …             …             …        …
2235             0             0             0             0         0
2236             0             0             0             0         0
2237             0             0             0             0         0
2238             0             0             0             0         0
2239             1             1             0             0         1

      Complain    Country  Age  Children
0            0      Spain   53         0
1            0     Canada   62         0
```

```
2             0        USA   65          1
3             0  Australia   56          2
4             0      Spain   34          1
…         …         …   …      …
2235          0        USA   47          1
2236          0      Spain   46          1
2237          0      Spain   47          1
2238          0      India   45          0
2239          0     Canada   54          2

[2240 rows x 30 columns]
```

[ ]:

[49]:
```
df1[numerical_attributes].mean()
```

[49]:
```
Year_Birth              1968.805804
Income                 52247.251354
Recency                   49.109375
MntWines                 303.935714
MntFruits                 21.575000
MntMeatProducts          166.950000
MntFishProducts           37.525446
MntSweetProducts          21.358036
MntGoldProds              44.021875
NumDealsPurchases          2.325000
NumWebPurchases            4.080804
NumCatalogPurchases        2.662054
NumStorePurchases          5.790179
NumWebVisitsMonth          5.316518
dtype: float64
```

[50]:
```
numerical_attributes.append("Age")
```

[51]:
```
numerical_attributes.append("Children")
```

[52]:
```
numerical_attributes
```

[52]:
```
['Year_Birth',
 'Income',
 'Recency',
 'MntWines',
 'MntFruits',
 'MntMeatProducts',
 'MntFishProducts',
 'MntSweetProducts',
 'MntGoldProds',
```

```
 'NumDealsPurchases',
 'NumWebPurchases',
 'NumCatalogPurchases',
 'NumStorePurchases',
 'NumWebVisitsMonth',
 'Age',
 'Children']
```

[53]:
```
numerical_attributes.remove("Year_Birth")
```

[54]:
```
df1[numerical_attributes].mean()
```

[54]:
```
Income                 52247.251354
Recency                   49.109375
MntWines                 303.935714
MntFruits                 21.575000
MntMeatProducts          166.950000
MntFishProducts           37.525446
MntSweetProducts          21.358036
MntGoldProds              44.021875
NumDealsPurchases          2.325000
NumWebPurchases            4.080804
NumCatalogPurchases        2.662054
NumStorePurchases          5.790179
NumWebVisitsMonth          5.316518
Age                       54.194196
Children                   0.950446
dtype: float64
```

[55]:
```
df1.columns
```

[55]:
```
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', 'Response', 'Complain', 'Country', 'Age', 'Children'],
      dtype='object')
```

[56]:
```
categorical_attr = ['Education', 'Marital_Status','Country']
df[categorical_attr].mode()
```

[56]:

	Education	Marital_Status	Country
0	Graduation	Married	Spain

Answer Q3 #### The average customer has: ###### Income 51447.697559 ###### Recency 48.704018 ###### MntWines 281.795268 ###### MntFruits 21.575000 ###### MntMeatProducts 145.846429 ###### MntFishProducts 32.066071 ###### MntSweet-Products 21.358036 ###### MntGoldProds 39.077232 ###### NumDealsPurchases 2.170536 ###### NumWebPurchases 3.944643 ###### NumCatalogPurchases 2.466964 ###### NumStorePurchases 5.685714 ###### NumWebVisitsMonth 5.244196 ###### Age 54.194196 ###### Children 0.950446

Q5. Which products are performing best?

[57]:
```
df1
```

[57]:
```
         ID  Year_Birth   Education Marital_Status   Income  Kidhome  \
0      1826        1970  Graduation       Divorced  84835.0        0
1         1        1961  Graduation         Single  57091.0        0
2     10476        1958  Graduation        Married  67267.0        0
3      1386        1967  Graduation       Together  32474.0        1
4      5371        1989  Graduation         Single  21474.0        1
…       …          …          …             …        …        …
2235  10142        1976         PhD       Divorced  66476.0        0
2236   5263        1977    2n Cycle        Married  31056.0        1
2237     22        1976  Graduation       Divorced  46310.0        1
2238    528        1978  Graduation        Married  65819.0        0
2239   4070        1969         PhD        Married  94871.0        0

      Teenhome Dt_Customer  Recency  MntWines  …  AcceptedCmp3  \
0            0  2014-06-16        0       189  …             0
1            0  2014-06-15        0       464  …             0
2            1  2014-05-13        0       134  …             0
3            1  2014-05-11        0        10  …             0
4            0  2014-04-08        0         6  …             1
…          …          …       …         …  …           …
2235         1  2013-03-07       99       372  …             0
2236         0  2013-01-22       99         5  …             0
2237         0  2012-12-03       99       185  …             0
2238         0  2012-11-29       99       267  …             0
2239         2  2012-09-01       99       169  …             0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
0                0             0             0             0         1
1                0             0             0             1         1
2                0             0             0             0         0
3                0             0             0             0         0
4                0             0             0             0         1
…              …             …             …             …       …
2235             0             0             0             0         0
2236             0             0             0             0         0
2237             0             0             0             0         0
```

```
2238              0             0             0             0          0
2239              1             1             0             0          1

      Complain    Country  Age  Children
0            0      Spain   53         0
1            0     Canada   62         0
2            0        USA   65         1
3            0  Australia   56         2
4            0      Spain   34         1
…          …         …   …        …
2235         0        USA   47         1
2236         0      Spain   46         1
2237         0      Spain   47         1
2238         0      India   45         0
2239         0     Canada   54         2

[2240 rows x 30 columns]
```

[58]:
```python
df1.columns
```

[58]:
```
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', 'Response', 'Complain', 'Country', 'Age', 'Children'],
      dtype='object')
```

[59]:
```python
product_df = df1[["MntWines",'MntFruits','MntMeatProducts', 'MntFishProducts',␣
↪'MntSweetProducts',
                  'MntGoldProds']]
product_df
```

[59]:
```
      MntWines  MntFruits  MntMeatProducts  MntFishProducts  MntSweetProducts  \
0          189       80.0              379              111              76.0
1          464        5.0               64                7               0.0
2          134       11.0               59               15               2.0
3           10        0.0                1                0               0.0
4            6       16.0               24               11               0.0
…          …         …              …              …                …
2235       372       18.0              126               47              48.0
2236         5       10.0               13                3               8.0
2237       185        2.0               88               15               5.0
2238       267       38.0              701              149              76.0
2239       169       24.0              553              188               0.0
```

```
      MntGoldProds
0              218
1               37
2               30
3                0
4               34
…             …
2235            78
2236            16
2237            14
2238            63
2239           144

[2240 rows x 6 columns]
```

[60]:
```python
product_list = ["MntWines",'MntFruits','MntMeatProducts', 'MntFishProducts',␣
 ↪'MntSweetProducts',
                  'MntGoldProds']
def sum_items(df,col_list):
    myDict = {}
    for prod in col_list:
        total = df[prod].sum()
        myDict[prod] = total
    return myDict
```

[61]:
```python
product_performance = sum_items(product_df,product_list)
product_performance
```

[61]:
```
{'MntWines': 680816,
 'MntFruits': 48328.0,
 'MntMeatProducts': 373968,
 'MntFishProducts': 84057,
 'MntSweetProducts': 47842.0,
 'MntGoldProds': 98609}
```

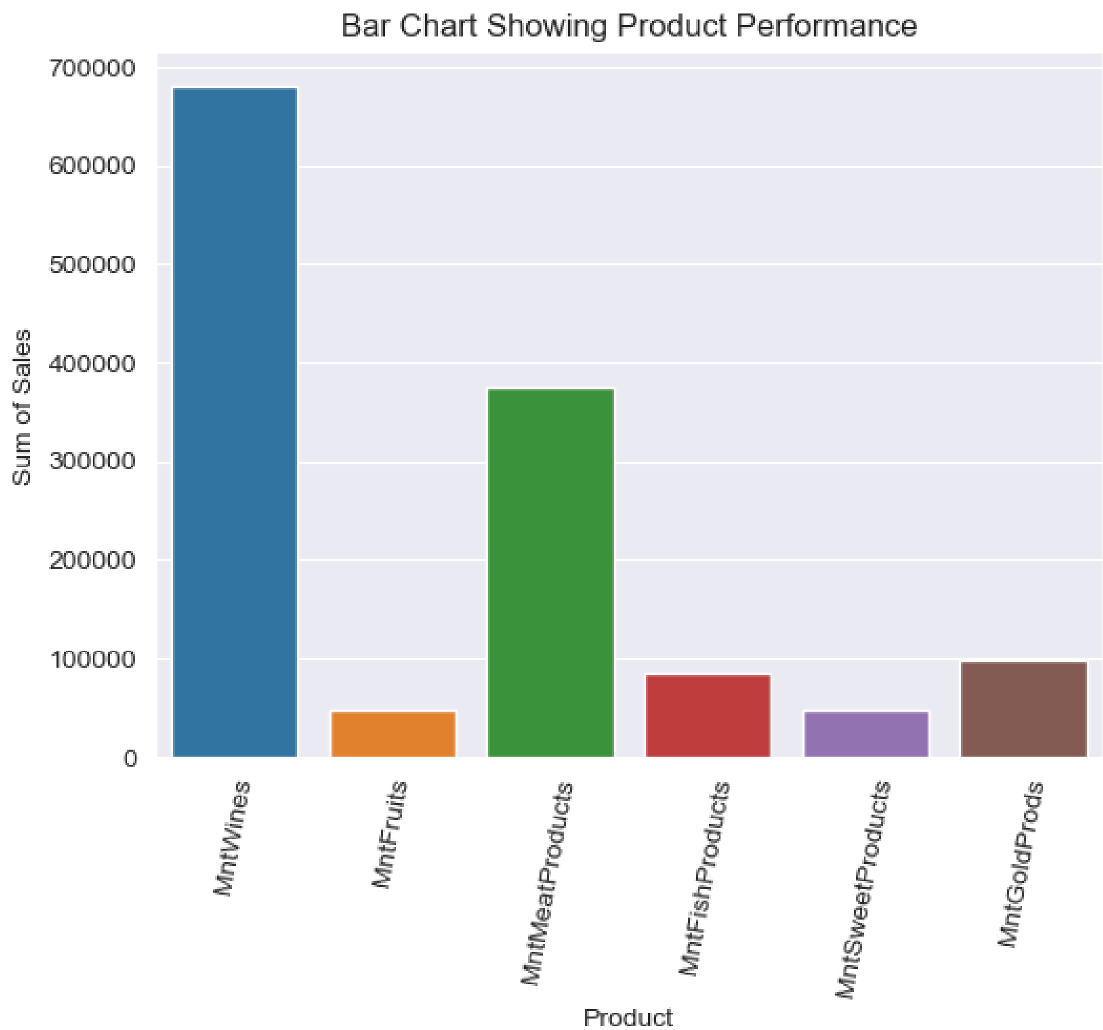
[62]:
```python
product_performance_df = pd.DataFrame(list(product_performance.items()),
                                      columns=["Product","Sum of Sales"])
product_performance_df
```

[62]:

	Product	Sum of Sales
0	MntWines	680816.0
1	MntFruits	48328.0
2	MntMeatProducts	373968.0
3	MntFishProducts	84057.0
4	MntSweetProducts	47842.0
5	MntGoldProds	98609.0

[63]:
```
sb.set_style("darkgrid")
sb.barplot(x=product_performance_df["Product"],y=product_performance_df["Sum of␣
↪Sales"])
plt.title("Bar Chart Showing Product Performance")
plt.xticks(rotation=80)
```

[63]:
```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'MntWines'),
  Text(1, 0, 'MntFruits'),
  Text(2, 0, 'MntMeatProducts'),
  Text(3, 0, 'MntFishProducts'),
  Text(4, 0, 'MntSweetProducts'),
  Text(5, 0, 'MntGoldProds')])
```



Answer to Q5

From the graph above, It can be seen that wine is the product that performed really well with sum of sales exceeding 600,000. Meat was the next perfoming products with sum of sales exceeding 300,000. The remaining products did not really perform as well as the Meat and Wine products. Given the the high value of gold products, it is commendable to see that gold products sold more that fish, fruit and sweet products.

Which channels are underperforming?

[64]:
```
df1.columns
```

[64]:
```
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', 'Response', 'Complain', 'Country', 'Age', 'Children'],
      dtype='object')
```

[65]:
```
channels_df = df1[['NumWebPurchases','NumCatalogPurchases','NumStorePurchases']]
channels_df
```

[65]:

	NumWebPurchases	NumCatalogPurchases	NumStorePurchases
0	4.0	4	6
1	7.0	3	7
2	3.0	2	5
3	1.0	0	2
4	3.0	1	2
…	…	…	…
2235	5.0	2	11
2236	1.0	0	3
2237	6.0	1	5
2238	5.0	4	10
2239	8.0	5	4

[2240 rows x 3 columns]

[66]:
```
sale_channels = sum_items(channels_df,
                                                   ␣
 ↪['NumWebPurchases','NumCatalogPurchases','NumStorePurchases'])
sale_channels
```

[66]:
```
{'NumWebPurchases': 9141.0,
 'NumCatalogPurchases': 5963,
 'NumStorePurchases': 12970}
```

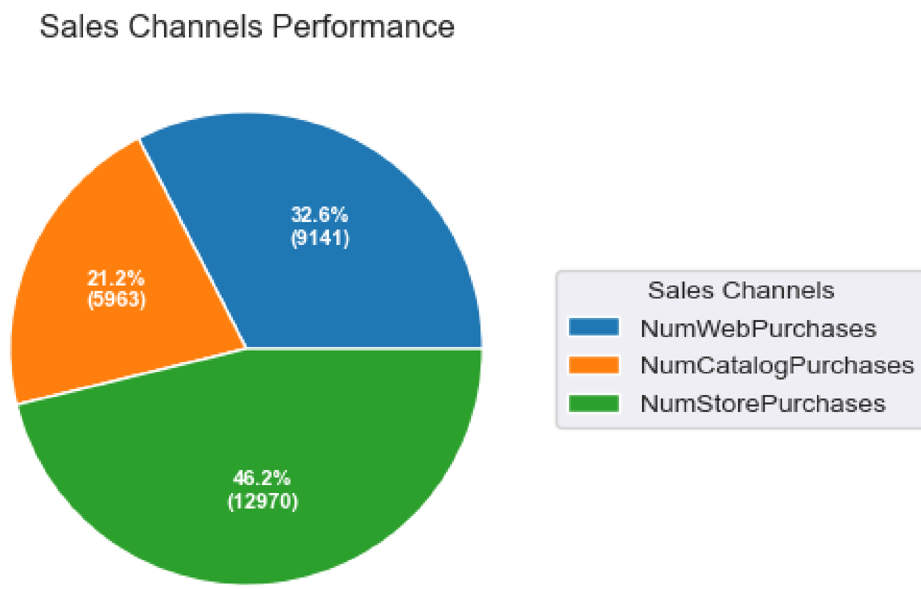
[67]:
```python
sale_channels_df = pd.DataFrame(sale_channels.items(),
                               columns=["Channel","Total Num of Sales"])
sale_channels_df
```

[67]:

	Channel	Total Num of Sales
0	NumWebPurchases	9141.0
1	NumCatalogPurchases	5963.0
2	NumStorePurchases	12970.0

[68]:
```python
channel_data = [num for num in sale_channels_df["Total Num of Sales"]]
channel_labels = [label for label in sale_channels_df["Channel"]]
```

[69]:
```python
fig, ax = plt.subplots(figsize=(6, 4), subplot_kw=dict(aspect="equal"))

def func(pct, allvals):
    absolute = int(np.round(pct/100.*np.sum(allvals)))
    return f"{pct:.1f}%\n({absolute:d})"


wedges, texts, autotexts = ax.pie(channel_data, autopct=lambda pct: func(pct,␣
 ↪channel_data),
                                  textprops=dict(color="w"))

ax.legend(wedges, channel_labels,
          title="Sales Channels",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Sales Channels Performance")

plt.show()
```



Answer to Q6

From the pie chart above, it is clear that the catalog purchase is the underperforming sales channel when compared to the other two channels. The Store purchase channel is the most effective one generating a total number of sales of 12,970 which is 46.2% of the total number of sales from all channels.

[96]:
```
df1
```

[96]:
```
         ID  Year_Birth   Education Marital_Status   Income  Kidhome  \
0      1826        1970  Graduation       Divorced  84835.0        0
1         1        1961  Graduation         Single  57091.0        0
2     10476        1958  Graduation        Married  67267.0        0
3      1386        1967  Graduation       Together  32474.0        1
4      5371        1989  Graduation         Single  21474.0        1
…       …           …         …           …        …
2235  10142        1976         PhD       Divorced  66476.0        0
2236   5263        1977    2n Cycle        Married  31056.0        1
2237     22        1976  Graduation       Divorced  46310.0        1
2238    528        1978  Graduation        Married  65819.0        0
2239   4070        1969         PhD        Married  94871.0        0

      Teenhome Dt_Customer  Recency  MntWines  …  AcceptedCmp3  \
0            0  2014-06-16        0       189  …             0
1            0  2014-06-15        0       464  …             0
```

```
2             1  2014-05-13        0       134  ...            0
3             1  2014-05-11        0        10  ...            0
4             0  2014-04-08        0         6  ...            0
...         ...         ...      ...       ... ...          ...
2235          1  2013-03-07       99       372  ...            0
2236          0  2013-01-22       99         5  ...            0
2237          0  2012-12-03       99       185  ...            0
2238          0  2012-11-29       99       267  ...            0
2239          2  2012-09-01       99       169  ...            0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
0                0             0             0             0         1
1                0             0             0             1         1
2                0             0             0             0         0
3                0             0             0             0         0
4                0             0             0             0         1
...            ...           ...           ...           ...       ...
2235             0             0             0             0         0
2236             0             0             0             0         0
2237             0             0             0             0         0
2238             0             0             0             0         0
2239             1             1             0             0         1

      Complain    Country  Age  Children
0            0      Spain   53         0
1            0     Canada   62         0
2            0        USA   65         1
3            0  Australia   56         2
4            0      Spain   34         1
...        ...        ...  ...       ...
2235         0        USA   47         1
2236         0      Spain   46         1
2237         0      Spain   47         1
2238         0      India   45         0
2239         0     Canada   54         2

[2240 rows x 30 columns]
```

[97]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 30 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               2240 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   float64
 11  MntMeatProducts      2240 non-null   int64
 12  MntFishProducts      2240 non-null   int64
 13  MntSweetProducts     2240 non-null   float64
 14  MntGoldProds         2240 non-null   int64
 15  NumDealsPurchases    2240 non-null   int64
 16  NumWebPurchases      2240 non-null   float64
 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  Response             2240 non-null   int64
 26  Complain             2240 non-null   int64
 27  Country              2240 non-null   object
 28  Age                  2240 non-null   int64
 29  Children             2240 non-null   int64
dtypes: float64(4), int64(22), object(4)
memory usage: 525.1+ KB
```

ANALYSING THE PURCHASE HABITS BY INCOME AND AGE GROUP

[72]:
```
#Splitting the datasets into 4 age groups namely children, adolescents, adult␣
↪and senior adult
children = [0,12]
adolescent = [13,18]
adult = [19,59]
senior = [60]
```

[73]:
```
children_df = df1[(df1["Age"] > children[0]) & (df1["Age"] <= children[1])]
children_df
```

[73]:
```
Empty DataFrame
Columns: [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, Response, Complain,
Country, Age, Children]
Index: []

[0 rows x 30 columns]
```

[74]:
```
adolescent_df = df1[(df1["Age"] >= adolescent[0]) & (df1["Age"] <=␣
↪adolescent[1])]
adolescent_df
```

[74]:
```
Empty DataFrame
Columns: [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, Response, Complain,
Country, Age, Children]
Index: []

[0 rows x 30 columns]
```

[75]:
```
adult_df = df1[(df1["Age"] >= adult[0]) & (df1["Age"] <= adult[1])]
adult_df
```

[75]:
```
         ID  Year_Birth   Education Marital_Status   Income  Kidhome  \
0      1826        1970  Graduation       Divorced  84835.0        0
3      1386        1967  Graduation       Together  32474.0        1
4      5371        1989  Graduation         Single  21474.0        1
7      1991        1967  Graduation       Together  44931.0        0
11     5642        1979      Master       Together  62499.0        1
…       …           …         …            …        …
2235  10142        1976         PhD       Divorced  66476.0        0
2236   5263        1977    2n Cycle        Married  31056.0        1
2237     22        1976  Graduation       Divorced  46310.0        1
2238    528        1978  Graduation        Married  65819.0        0
2239   4070        1969         PhD        Married  94871.0        0

      Teenhome Dt_Customer  Recency  MntWines  …  AcceptedCmp3  \
0            0  2014-06-16        0       189  …             0
3            1  2014-05-11        0        10  …             0
4            0  2014-04-08        0         6  …             1
7            1  2014-01-18        0        78  …             0
11           0  2013-12-09        0       140  …             0
…         …           …        …  …            …
2235         1  2013-03-07       99       372  …             0
2236         0  2013-01-22       99         5  …             0
2237         0  2012-12-03       99       185  …             0
```

```
2238          0  2012-11-29       99       267  …              0
2239          2  2012-09-01       99       169  …              0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
0                0             0             0             0         1
3                0             0             0             0         0
4                0             0             0             0         1
7                0             0             0             0         0
11               0             0             0             0         0
…              …            …            …            …        …
2235             0             0             0             0         0
2236             0             0             0             0         0
2237             0             0             0             0         0
2238             0             0             0             0         0
2239             1             1             0             0         1

      Complain    Country  Age  Children
0            0      Spain   53         0
3            0  Australia   56         2
4            0      Spain   34         1
7            0      Spain   56         1
11           0      Spain   44         1
…          …        …  …       …
2235         0        USA   47         1
2236         0      Spain   46         1
2237         0      Spain   47         1
2238         0      India   45         0
2239         0     Canada   54         2

[1496 rows x 30 columns]
```

[76]:

```
senior_df = df1[df1["Age"] >=␣
↪senior[0]][["Marital_Status","Income",'MntWines','MntFruits',
      'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
      'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
      'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth','Age',␣
↪'Children']]
senior_df
```

[76]:

```
     Marital_Status   Income  MntWines  MntFruits  MntMeatProducts  \
1            Single  57091.0       464        5.0               64
2           Married  67267.0       134       11.0               59
5            Single  71691.0       336       80.0              411
6           Married  63564.0       769       80.0              252
8           Married  65324.0       384        0.0              102
…              …      …     …       …               …
2202         Single  23091.0        35        0.0               11
```

```
2216      Divorced  50611.0       459       0.0                24
2217      Divorced  50611.0       459       0.0                24
2227      Together  62568.0       362      17.0               398
2233      Divorced  36640.0        15       6.0                 8

      MntFishProducts  MntSweetProducts  MntGoldProds  NumDealsPurchases  \
1                   7               0.0            37                  1
2                  15               2.0            30                  1
5                 240              32.0            43                  1
6                  15              34.0            65                  1
8                  21              32.0             5                  3
…                 …                …            …                 …
2202                0               0.0             2                  4
2216                6               0.0             4                  6
2217                6               0.0             4                  6
2227               80              35.0            61                  3
2233                7               4.0            25                  1

      NumWebPurchases  NumCatalogPurchases  NumStorePurchases  \
1                 7.0                    3                  7
2                 3.0                    2                  5
5                 4.0                    7                  5
6                10.0                   10                  7
8                 6.0                    2                  9
…                 …                   …                 …
2202              2.0                    1                  3
2216              4.0                    5                  7
2217              4.0                    5                  7
2227              5.0                    3                  5
2233              2.0                    1                  2

      NumWebVisitsMonth  Age  Children
1                     5   62         0
2                     2   65         1
5                     2   65         0
6                     6   69         0
8                     4   69         1
…                   …  …       …
2202                  7   60         2
2216                  6   63         1
2217                  6   63         1
2227                  4   61         1
2233                  5  123         1

[744 rows x 15 columns]
```

[100]:
```
print(adult_df["Income"].max())
print (adult_df["Income"].mean())
```

```
666666.0
49733.0852399419
```

[101]:
```
print(senior_df["Income"].max())
print (senior_df["Income"].mean())
```

```
156924.0
57302.617625723
```

Analysing the purchase habits for each age group exploration will be done on the basis of:

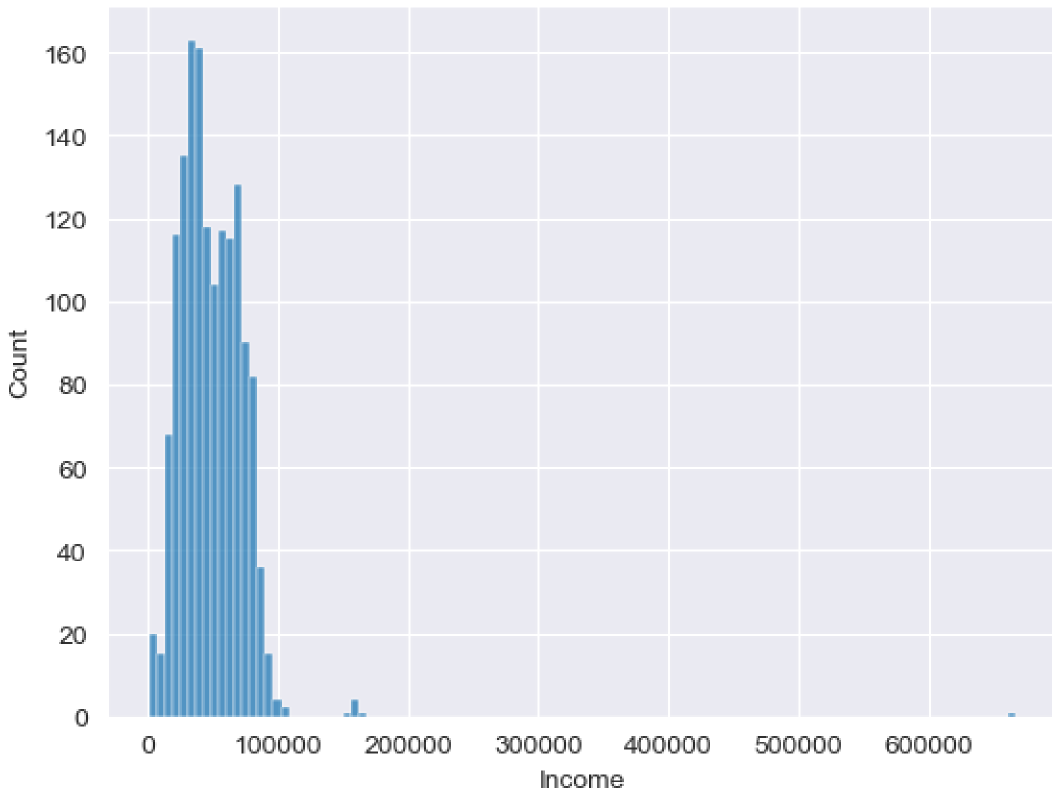
1. **Income levels of the age group**

2. **Type of products consumed or purchased**

3. **Type of sales channels used**

[77]:
```
sb.histplot(data=adult_df["Income"])
```

[77]: <AxesSubplot:xlabel='Income', ylabel='Count'>



[78]:
```
adult_df["Income"].max()
```

[78]: 666666.0

The adult age group has a few individuals with relatively higher income so I will separate this individuals to study their spending habits

[79]:
```
#individuals with higher income
adult_df_high = adult_df.loc[adult_df["Income"] > 130000]
adult_df_high
```

[79]:
```
        ID  Year_Birth   Education Marital_Status    Income  Kidhome  \
325   4931        1977  Graduation       Together  157146.0        0
497   1501        1982         PhD        Married  160803.0        0
527   9432        1977  Graduation       Together  666666.0        1
731   1503        1976         PhD       Together  162397.0        1
853   5336        1971      Master       Together  157733.0        1
1826  5555        1975  Graduation       Divorced  153924.0        0
2204  8475        1973         PhD        Married  157243.0        0

      Teenhome Dt_Customer  Recency  MntWines  …  AcceptedCmp3  \
```

```
325           0  2013-04-29       13         1  ...             0
497           0  2012-08-04       21        55  ...             0
527           0  2013-06-02       23         9  ...             0
731           1  2013-06-03       31        85  ...             0
853           0  2013-06-04       37        39  ...             0
1826          0  2014-02-07       81         1  ...             0
2204          1  2014-03-01       98        20  ...             0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
325              0             0             0             0         0
497              0             0             0             0         0
527              0             0             0             0         0
731              0             0             0             0         0
853              0             0             0             0         0
1826             0             0             0             0         0
2204             0             0             0             0         0

      Complain       Country  Age  Children
325          0  Saudi Arabia   46         0
497          0           USA   41         0
527          0  Saudi Arabia   46         1
731          0         Spain   47         2
853          0         Spain   52         1
1826         0         Spain   48         0
2204         0         India   50         1

[7 rows x 30 columns]
```

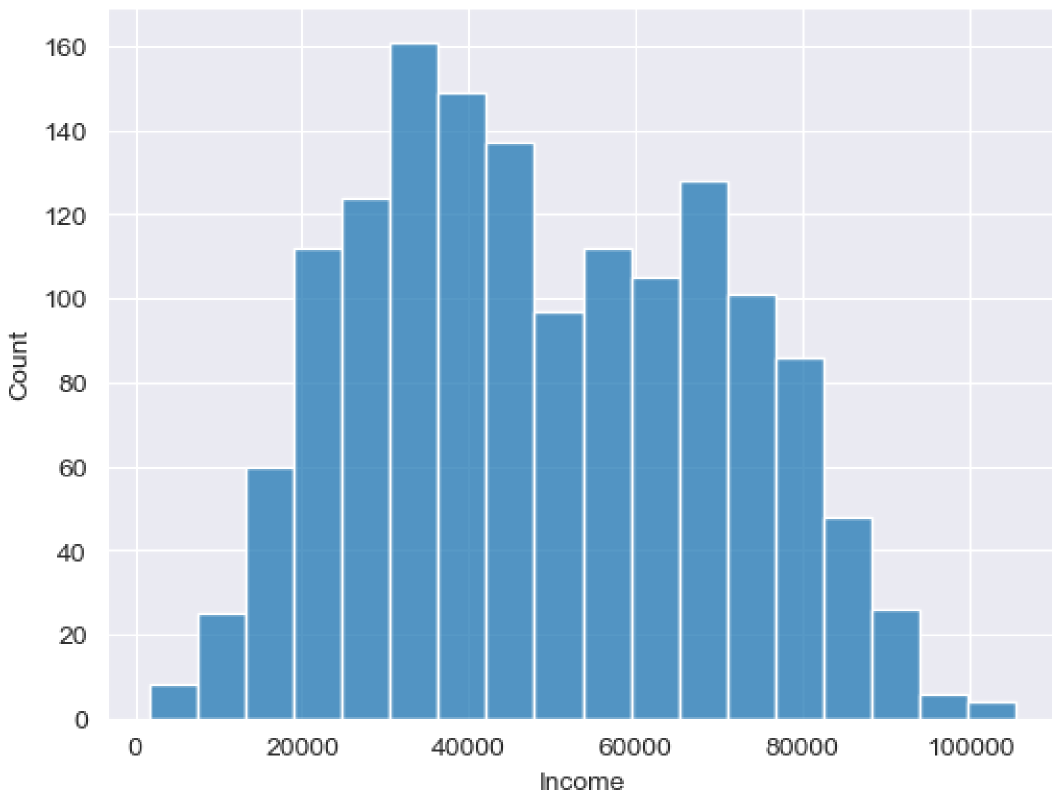
[80]:
```
#Adults with not very high income (ie income follws normal distribution)
adult_df_normal = adult_df.loc[adult_df["Income"] < 130000]
adult_df_normal
```

[80]:
```
         ID  Year_Birth   Education Marital_Status   Income  Kidhome  \
0      1826        1970  Graduation       Divorced  84835.0        0
3      1386        1967  Graduation       Together  32474.0        1
4      5371        1989  Graduation         Single  21474.0        1
7      1991        1967  Graduation       Together  44931.0        0
11     5642        1979      Master       Together  62499.0        1
...     ...         ...         ...            ...      ...      ...
2235  10142        1976         PhD       Divorced  66476.0        0
2236   5263        1977    2n Cycle        Married  31056.0        1
2237     22        1976  Graduation       Divorced  46310.0        1
2238    528        1978  Graduation        Married  65819.0        0
2239   4070        1969         PhD        Married  94871.0        0

      Teenhome Dt_Customer  Recency  MntWines  ...  AcceptedCmp3  \
0            0  2014-06-16        0       189  ...             0
```

```
3              1  2014-05-11         0         10  ...              0
4              0  2014-04-08         0          6  ...              1
7              1  2014-01-18         0         78  ...              0
11             0  2013-12-09         0        140  ...              0
...          ...         ...       ...        ...  ...            ...
2235           1  2013-03-07        99        372  ...              0
2236           0  2013-01-22        99          5  ...              0
2237           0  2012-12-03        99        185  ...              0
2238           0  2012-11-29        99        267  ...              0
2239           2  2012-09-01        99        169  ...              0

      AcceptedCmp4  AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Response  \
0                0             0             0             0         1
3                0             0             0             0         0
4                0             0             0             0         1
7                0             0             0             0         0
11               0             0             0             0         0
...            ...           ...           ...           ...       ...
2235             0             0             0             0         0
2236             0             0             0             0         0
2237             0             0             0             0         0
2238             0             0             0             0         0
2239             1             1             0             0         1

      Complain    Country  Age  Children
0            0      Spain   53         0
3            0  Australia   56         2
4            0      Spain   34         1
7            0      Spain   56         1
11           0      Spain   44         1
...        ...        ...  ...       ...
2235         0        USA   47         1
2236         0      Spain   46         1
2237         0      Spain   47         1
2238         0      India   45         0
2239         0     Canada   54         2

[1489 rows x 30 columns]
```

[81]:
```
sb.histplot(data=adult_df_normal["Income"])
```

[81]:
```
<AxesSubplot:xlabel='Income', ylabel='Count'>
```



[82]:
```
adult_normal_df =␣
↪adult_df_normal[["Marital_Status","Income","MntWines","MntFruits","MntMeatProducts",
                                                                                                         ␣
↪"MntFishProducts","MntSweetProducts","MntGoldProds","NumDealsPurchases",
                                                                                                         ␣
↪"NumWebPurchases","NumCatalogPurchases","NumStorePurchases","NumWebVisitsMonth",
                                        "Age","Children"]]
adult_normal_df
```

[82]:

```
     Marital_Status   Income  MntWines  MntFruits  MntMeatProducts  \
0          Divorced  84835.0       189       80.0              379
3          Together  32474.0        10        0.0                1
4            Single  21474.0         6       16.0               24
7          Together  44931.0        78        0.0               11
11         Together  62499.0       140        4.0               61
…                 …        …       …         …                …
2235       Divorced  66476.0       372       18.0              126
2236        Married  31056.0         5       10.0               13
2237       Divorced  46310.0       185        2.0               88
2238        Married  65819.0       267       38.0              701
2239        Married  94871.0       169       24.0              553
```

```
      MntFishProducts  MntSweetProducts  MntGoldProds  NumDealsPurchases  \
0                 111              76.0           218                  1
3                   0               0.0             0                  1
4                  11               0.0            34                  2
7                   0               0.0             7                  1
11                  0              13.0             4                  2
…                 …                 …            …                  …
2235               47              48.0            78                  2
2236                3               8.0            16                  1
2237               15               5.0            14                  2
2238              149              76.0            63                  1
2239              188               0.0           144                  1

      NumWebPurchases  NumCatalogPurchases  NumStorePurchases  \
0                 4.0                    4                  6
3                 1.0                    0                  2
4                 3.0                    1                  2
7                 2.0                    1                  3
11                3.0                    1                  6
…                 …                   …                 …
2235              5.0                    2                 11
2236              1.0                    0                  3
2237              6.0                    1                  5
2238              5.0                    4                 10
2239              8.0                    5                  4

      NumWebVisitsMonth  Age  Children
0                     1   53         0
3                     7   56         2
4                     7   34         1
7                     5   56         1
11                    4   44         1
…                   …  …       …
2235                  4   47         1
2236                  8   46         1
2237                  8   47         1
2238                  3   45         0
2239                  7   54         2

[1489 rows x 15 columns]
```

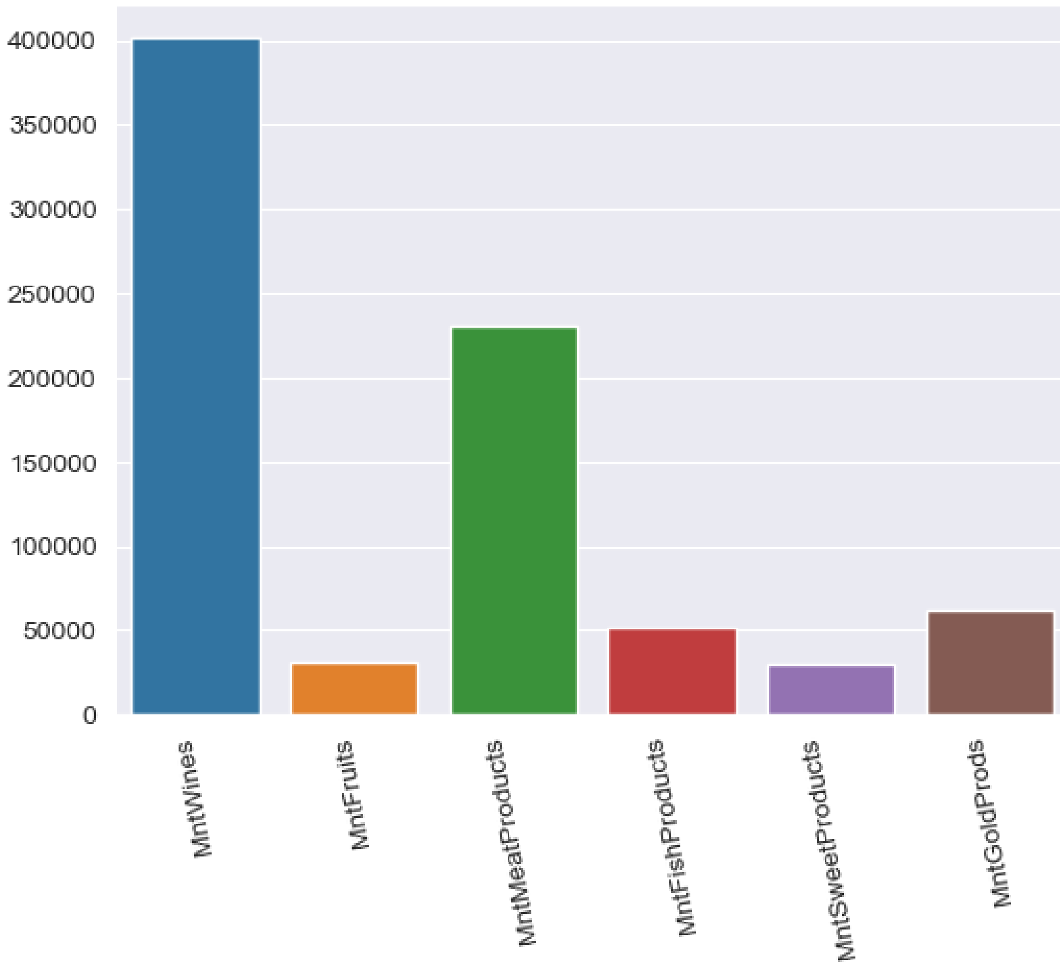
[83]:
```
adult_normal_prod = adult_normal_df[["MntWines","MntFruits","MntMeatProducts",
                "MntFishProducts","MntSweetProducts","MntGoldProds"]].sum()
adult_normal_prod
```

[83]:
```
MntWines            401343.0
MntFruits            30883.0
MntMeatProducts     230648.0
MntFishProducts      52205.0
MntSweetProducts     30480.0
MntGoldProds         61527.0
dtype: float64
```

[84]:
```
adult_prd =␣
↪["MntWines","MntFruits","MntMeatProducts","MntFishProducts","MntSweetProducts",
            "MntGoldProds"]

adult_prd_sum = [401343.0,30883.0,230648.0,52205.0,30480.0,61527.0]
```

[85]:
```
sb.barplot(x=adult_prd,y=adult_prd_sum)
plt.xticks(rotation=100)
```

[85]:
```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'MntWines'),
  Text(1, 0, 'MntFruits'),
  Text(2, 0, 'MntMeatProducts'),
  Text(3, 0, 'MntFishProducts'),
  Text(4, 0, 'MntSweetProducts'),
  Text(5, 0, 'MntGoldProds')])
```



In tune with the general population, the adult age group with the low income consume wine and meat products the most. They also spend on gold products

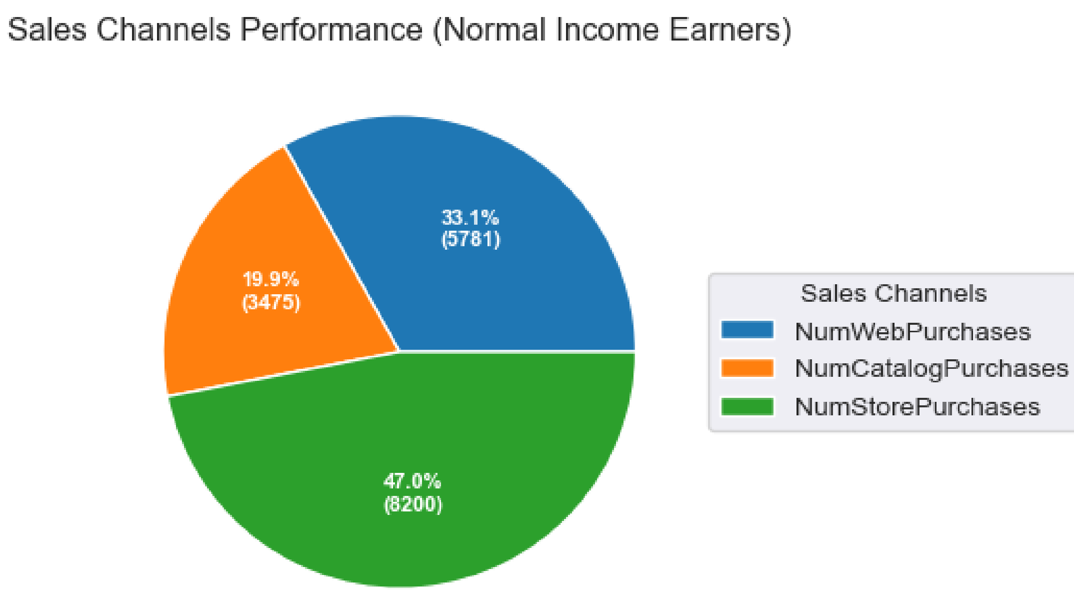
[86]:
```
adult_normal_channel = adult_normal_df[["NumWebPurchases","NumCatalogPurchases",
                                         "NumStorePurchases"]].sum()
adult_normal_channel
```

[86]:
```
NumWebPurchases        5781.0
NumCatalogPurchases    3475.0
NumStorePurchases      8200.0
dtype: float64
```

[87]:
```
adult_normal_channel_data = [5781.0,3475.0,8200.0]
adult_normal_channel_label =␣
 ↪["NumWebPurchases","NumCatalogPurchases","NumStorePurchases"]
```

[88]:

```python
fig, ax = plt.subplots(figsize=(6, 4), subplot_kw=dict(aspect="equal"))

def func(pct, allvals):
    absolute = int(np.round(pct/100.*np.sum(allvals)))
    return f"{pct:.1f}%\n({absolute:d})"


wedges, texts, autotexts = ax.pie(adult_normal_channel_data, autopct=lambda pct:
 ↪ func(pct, adult_normal_channel_data),
                                  textprops=dict(color="w"))

ax.legend(wedges, adult_normal_channel_label,
          title="Sales Channels",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Sales Channels Performance (Normal Income Earners)")

plt.show()
```



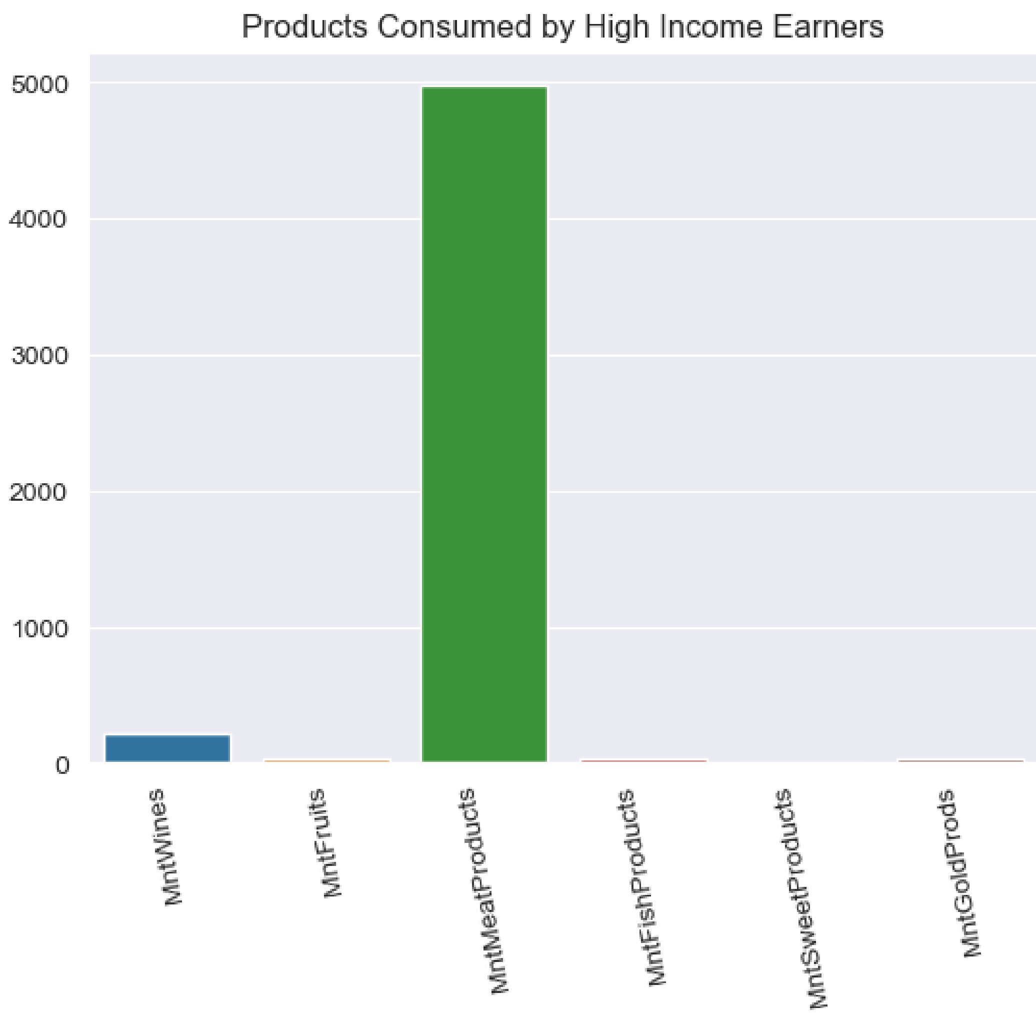
As with the the general population yet again, normal income earners prefer to use the store for purchases as shown above

[89]:
```python
# analysing the purchase habits of high income earners now
adult_df_high_prod = adult_df_high[["MntWines","MntFruits","MntMeatProducts",
                "MntFishProducts","MntSweetProducts","MntGoldProds"]].sum()
adult_df_high_prod
```

[89]:
```
MntWines              210.0
MntFruits              35.0
MntMeatProducts      4973.0
MntFishProducts        33.0
MntSweetProducts        9.0
MntGoldProds           29.0
dtype: float64
```

[90]:
```python
adult_high_prd =␣
 ↪["MntWines","MntFruits","MntMeatProducts","MntFishProducts","MntSweetProducts",
            "MntGoldProds"]

adult_high_prd_sum = [210.0,35.0,4973.0,33.0,9.0,29.0]
```

[91]:
```python
sb.barplot(x = adult_high_prd, y = adult_high_prd_sum)
plt.xticks(rotation=100)
plt.title("Products Consumed by High Income Earners")
plt.show()
```



From the bar graph above it can be seen that high income earners are spending more on Meat products. This is different from the spending habits of the general population

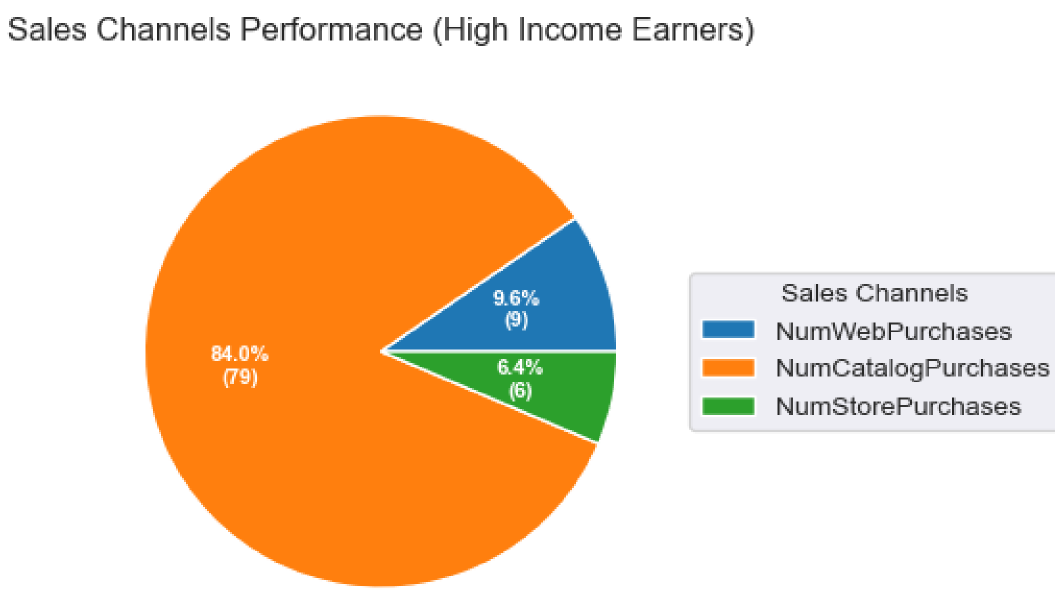
[92]:
```
adult_high_channel = adult_df_high[["NumWebPurchases","NumCatalogPurchases",
                                     "NumStorePurchases"]].sum()
adult_high_channel
```

[92]:
```
NumWebPurchases         9.0
NumCatalogPurchases    79.0
NumStorePurchases       6.0
dtype: float64
```

[93]:
```
adult_high_channel_data = [9.0,79.0,6.0]
adult_high_channel_label =␣
 ↪["NumWebPurchases","NumCatalogPurchases","NumStorePurchases"]
```

[94]:

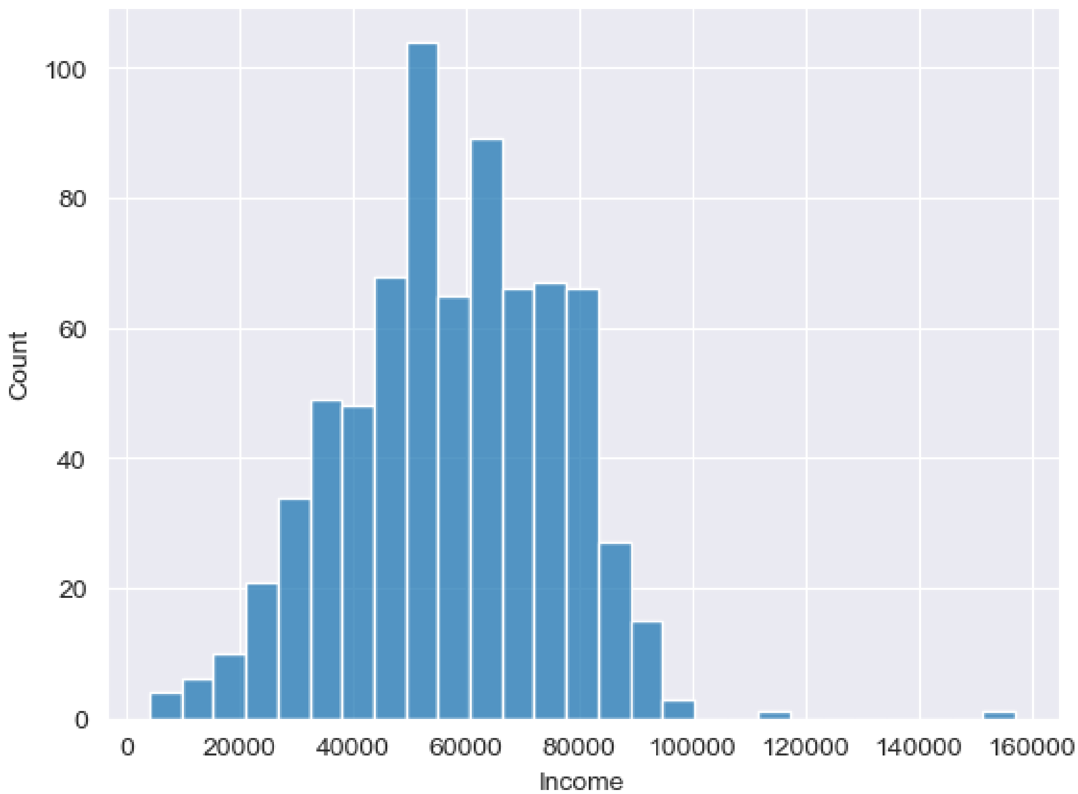
```
fig, ax = plt.subplots(figsize=(6, 4), subplot_kw=dict(aspect="equal"))

def func(pct, allvals):
    absolute = int(np.round(pct/100.*np.sum(allvals)))
    return f"{pct:.1f}%\n({absolute:d})"


wedges, texts, autotexts = ax.pie(adult_high_channel_data, autopct=lambda pct:␣
 ↪func(pct, adult_high_channel_data),
                                  textprops=dict(color="w"))

ax.legend(wedges, adult_high_channel_label,
          title="Sales Channels",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Sales Channels Performance (High Income Earners)")

plt.show()
```



Again as opposed to the general and low income earners, High Income earners generally tend to purchase products from products catalogs as can be seen in the pie charts above

[98]:
```python
sb.histplot(data=senior_df["Income"])
```

[98]: <AxesSubplot:xlabel='Income', ylabel='Count'>



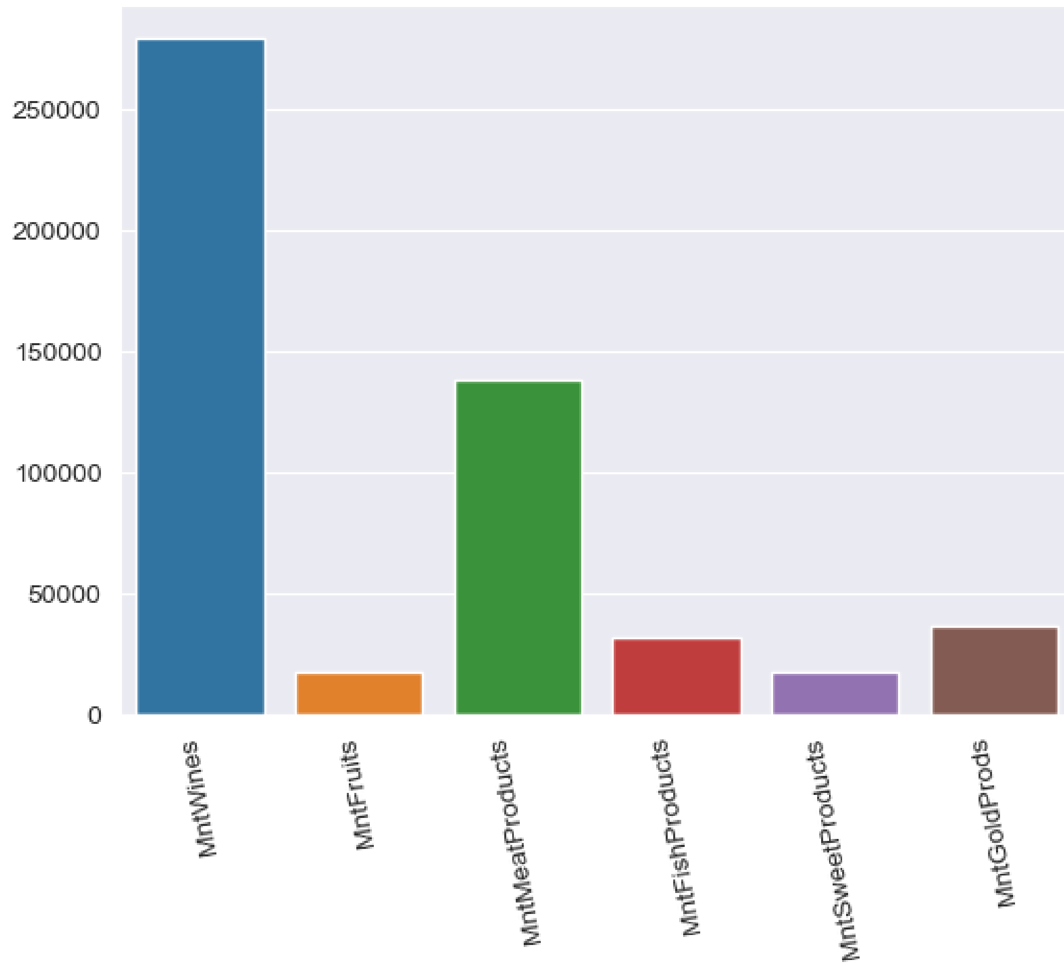
[102]:
```python
senior_prod = senior_df[["MntWines","MntFruits","MntMeatProducts",
                "MntFishProducts","MntSweetProducts","MntGoldProds"]].sum()
senior_prod
```

[102]:
```
MntWines            279263.0
MntFruits            17410.0
MntMeatProducts     138347.0
MntFishProducts      31819.0
MntSweetProducts     17353.0
MntGoldProds         37053.0
dtype: float64
```

[105]:
```python
senior_prd =␣
 ↪["MntWines","MntFruits","MntMeatProducts","MntFishProducts","MntSweetProducts",
           "MntGoldProds"]
senior_prd_sum = [279263.0,17410.0,138347.0,31819.0,17353.0,37053.0]
```

[106]:
```
sb.barplot(x=senior_prd,y=senior_prd_sum)
plt.xticks(rotation=100)
```

[106]:
```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'MntWines'),
  Text(1, 0, 'MntFruits'),
  Text(2, 0, 'MntMeatProducts'),
  Text(3, 0, 'MntFishProducts'),
  Text(4, 0, 'MntSweetProducts'),
  Text(5, 0, 'MntGoldProds')])
```



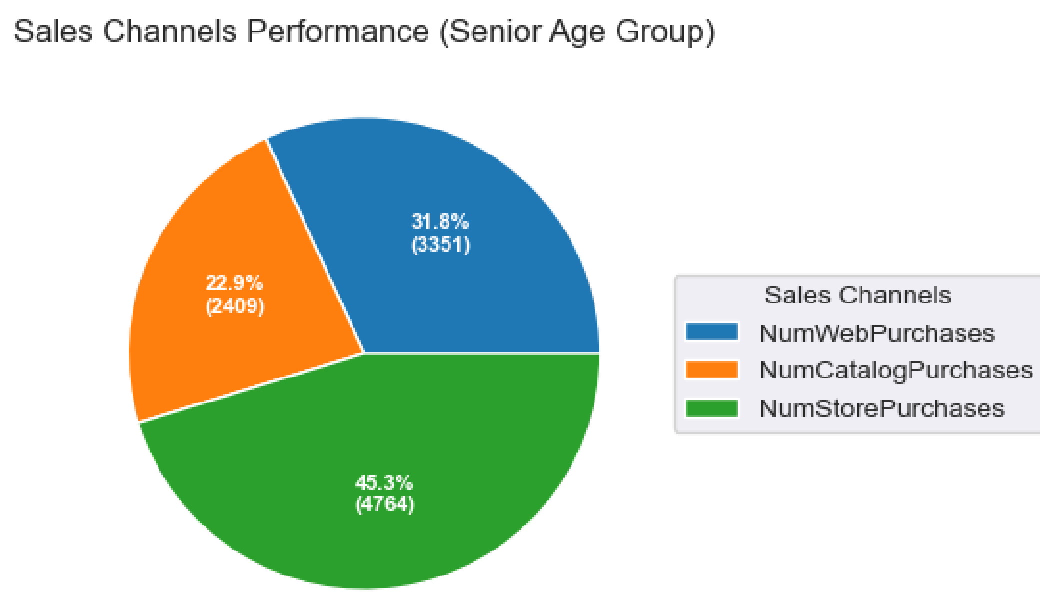
In tune with the general population, the senior age group consume wine and meat products the most. They also spend on gold products

[107]:
```
senior_channel = senior_df[["NumWebPurchases","NumCatalogPurchases",
                                         "NumStorePurchases"]].sum()
senior_channel
```

[107]:
```
NumWebPurchases        3351.0
NumCatalogPurchases    2409.0
NumStorePurchases      4764.0
dtype: float64
```

[108]:
```
senior_channel_data = [3351.0,2409.0,4764.0]
senior_channel_label =␣
↪["NumWebPurchases","NumCatalogPurchases","NumStorePurchases"]
```

[109]:
```
fig, ax = plt.subplots(figsize=(6, 4), subplot_kw=dict(aspect="equal"))

def func(pct, allvals):
    absolute = int(np.round(pct/100.*np.sum(allvals)))
    return f"{pct:.1f}%\n({absolute:d})"


wedges, texts, autotexts = ax.pie(senior_channel_data, autopct=lambda pct:␣
↪func(pct, senior_channel_data),
                                  textprops=dict(color="w"))

ax.legend(wedges, senior_channel_label,
          title="Sales Channels",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Sales Channels Performance (Senior Age Group)")

plt.show()
```



As with the the general population yet again, the senior age group prefer to use the store for purchases as shown above

```
[ ]:
```

0.0.2 INSIGHTS FROM DATASET

FACTORS SIGNIFICANTLY RELATED TO WEB PURCHASES - The data shows that the the num of web purchases is strongly correlated to the Income of customers.

MARKETING CAMPAIGN SUCCESS - The data also shows that the best performing marketing campaign was the last marketing campaign and the worst performing marketing campaign was the second campaign.

- In the last marketing campaign which was the most accepted by clients, Wine, Meat and Gold products were the top 3 performing products respectively. Customers who accepted the last camapign generally used the store purchase channel.

DESCRIPTION OF AVERAGE CUSTOMER - The average customer has income of 51447.697559, Recency of 48.704018, spent 281.795268, 21.575000, 145.846429, 32.066071, 21.358036, 39.077232 on wine, fruit, meat, fish, sweets and gold respectively, aged 54 and has 1 or more children.

PRODUCT PERFORMANCE - The data also shows that wine is the product that performed really well with sum of sales exceeding 600,000. Meat was the next perfoming products with sum of sales exceeding 300,000. The remaining products did not really perform as well as the Meat and Wine products. Given the the high value of gold products, it is commendable to see that gold products sold more than fish, fruit and sweet products.

SALES CHANNEL PERFORMANCE - Exploring and analysing the whole data shows that the catalog purchase is the underperforming sales channel when compared to the other two channels. The Store purchase channel is the most effective one generating a total number of sales of 12,970 which is 46.2% of the total number of sales from all channels.

AGE GROUPS EXPLORATION - There were no transactions involving children (Age 1 to 12) and adolescents (Age 13 to 18). All transactions were associated with adults (Age 19 to 59) and seniors (Age 60 above).

- On average seniors earn much more than than adults with average income of 57302.6 and 49733.0 respectively.

SPENDING HABITS BY AGE GROUP (ADULT AGE GROUP) - The adult age (Age 19 to 59) group has a few individuals with relatively higher income was split into two (normal income and very high income levels) to study their spending habits.

- In tune with the general population, the adult age group with the normal income consume wine and meat products the most and they also spend on gold products.

- As with the the general population yet again, normal income earners prefer to use the store for purchases as their channel of purchase.

- High income earners are spending more on Meat products. This is different from the spending habits of the general population

- Again as opposed to the normal income earners, High Income earners generally tend to purchase products from product catalogs.

SPENDING HABITS BY AGE GROUP (SENIOR AGE GROUP) - Again in tune with the general population, the senior age group consume wine and meat products the most. They also spend on gold products.

- In alignment with the general population, the senior age group consume wine and meat products the most. They also spend on gold products

- Also aligning with the general population, the senior age group consume wine and meat products the most. They also spend on gold products

[ ]:

```

```