Marketing Campaign Analysis

Analyze marketing data and address some important business problems/questions.

Context

Marketing Analytics broadly refers to the practice of using analytical methods and techniques to understand the effectiveness of various marketing activities ar
driven decisions to optimize for ROI on conversion rates. It typically involves analyzing various metrics around customer engagement with various marketing a
and costs associated with various marketing channels. These can generate valuable insights that can help an organization form better marketing strategies, o
and achieve overall growth.

Problem Statement

Company 'All You Need' has hired you as a Data Scientist and you've been told by the Chief Marketing Officer that recent marketing campaigns have not been as effective as they were expected to be and the conversion rate is very low. Your task is to analyze the related data, understand the problem, and identify key insights and recommendations for the CMO to potentially implement.

The data set marketing_data.csv consists of 2,240 customers of All You Need company with data on:

- Campaign successes/failures
- Product preferences
- Channel performances
- Customer profiles based on the spending habits

Data Dictionary

- ID : Unique ID of each customer
- Year_Birth : Age of the customer
- Education : Customer's level of education
- Marital_Status : Customer's marital status
- Kidhome : Number of small children in customer's household
- Teenhome : Number of teenagers in customer's household
- Income : Customer's yearly household income
- Recency : Number of days since the last purchase
- MntFishProducts : The amount spent on fish products in the last 2 years
- MntMeatProducts : The amount spent on meat products in the last 2 years
- MntFruits : The amount spent on fruits products in the last 2 years
- MntSweetProducts : Amount spent on sweet products in the last 2 years
- MntWines : The amount spent on wine products in the last 2 years
- MntGoldProds : The amount spent on gold products in the last 2 years
- NumDealsPurchases : Number of purchases made with discount
- NumCatalogPurchases : Number of purchases made using catalog (buying goods to be shipped through the mail)
- NumStorePurchases : Number of purchases made directly in stores
- NumWebPurchases : Number of purchases made through the company's website
- NumWebVisitsMonth : Number of visits to company's website in the last month
- AcceptedCmp1 : 1 if customer accepted the offer in the first campaign, 0 otherwise
- AcceptedCmp2 : 1 if customer accepted the offer in the second campaign, 0 otherwise
- AcceptedCmp3 : 1 if customer accepted the offer in the third campaign, 0 otherwise
- AcceptedCmp4 : 1 if customer accepted the offer in the fourth campaign, 0 otherwise
- AcceptedCmp5 : 1 if customer accepted the offer in the fifth campaign, 0 otherwise
- AcceptedCmp6 : 1 if customer accepted the offer in the last campaign, 0 otherwise
- Complain : 1 If the customer complained in the last 2 years, 0 otherwise
- Country: Country customer belongs to

Importing libraries and overview of the dataset

In [2]:

```
# Library to supress warnings or deprecation notes
import warnings
warnings.filterwarnings('ignore')

# Libraries to help with reading and manipulating data
import numpy as np
import pandas as pd

# Libraries to help with data visualization
```

```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [3]:

```
# from google.colab import files
# uploaded = files.upload()
```

Load the dataset

In [6]:

```
# loading the datset

df = pd.read_csv("C:/Users/ND/Downloads/Marketing+data (1).csv")
df.head()
```

Out[6]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	...	NumStorePurchases	NumWebVisitsMonth
0	1826	1970	Graduation	Divorced	84835.0	0	0	0	189	104	...	6	1
1	1	1961	Graduation	Single	57091.0	0	0	0	464	5	...	7	5
2	10476	1958	Graduation	Married	67267.0	0	1	0	134	11	...	5	2
3	1386	1967	Graduation	Together	32474.0	1	1	0	10	0	...	2	7
4	5371	1989	Graduation	Single	21474.0	1	0	0	6	16	...	2	7

5 rows × 27 columns

Check info of the dataset

In [7]:

```
#Checking the info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 27 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   Recency              2240 non-null   int64  
 8   MntWines             2240 non-null   int64  
 9   MntFruits            2240 non-null   int64  
 10  MntMeatProducts      2240 non-null   int64  
 11  MntFishProducts      2240 non-null   int64  
 12  MntSweetProducts     2240 non-null   int64  
 13  MntGoldProds         2240 non-null   int64  
 14  NumDealsPurchases    2240 non-null   int64  
 15  NumWebPurchases      2240 non-null   int64  
 16  NumCatalogPurchases  2240 non-null   int64  
 17  NumStorePurchases    2240 non-null   int64  
 18  NumWebVisitsMonth    2240 non-null   int64  
 19  AcceptedCmp1         2240 non-null   int64  
 20  AcceptedCmp2         2240 non-null   int64  
 21  AcceptedCmp3         2240 non-null   int64  
 22  AcceptedCmp4         2240 non-null   int64  
 23  AcceptedCmp5         2240 non-null   int64  
 24  AcceptedCmp6         2240 non-null   int64  
 25  Complain             2240 non-null   int64  
 26  Country              2240 non-null   object 
dtypes: float64(1), int64(23), object(3)
memory usage: 472.6+ KB
```

Observations:

- There are a total of 27 columns and 2,240 observations in the dataset
- We can see that the Income column has less than 2,240 non-null values i.e. column has missing values. We'll explore this further

Check the percentage of missing values for the Income column.

In [8]:

```
# % Null values in the Income column

(df.isnull().sum()/df.shape[0]*100)['Income']
```

Out[8]:

```
1.0714285714285714
```

Observations:

- Income has ~1.07% missing values.

Create a list for numerical columns in the dataset and check the summary statistics

Summary statistics for numerical columns

In [9]:

```
# num_cols contain numerical varibales
num_cols=['Year_Birth','Income','Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth','Kidhome',
       'Teenhome']
```

In [10]:

```
# printing descriptive statistics of numerical columns

#Uncomment the following code and fill in the blanks
df[num_cols].describe().T
```

Out[10]:

	count	mean	std	min	25%	50%	75%	max
Year_Birth	2240.0	1968.805804	11.984069	1893.0	1959.00	1970.0	1977.00	1996.0
Income	2216.0	52247.251354	25173.076661	1730.0	35303.00	51381.5	68522.00	666666.0
Recency	2240.0	49.109375	28.962453	0.0	24.00	49.0	74.00	99.0
MntWines	2240.0	303.935714	336.597393	0.0	23.75	173.5	504.25	1493.0
MntFruits	2240.0	26.302232	39.773434	0.0	1.00	8.0	33.00	199.0
MntMeatProducts	2240.0	166.950000	225.715373	0.0	16.00	67.0	232.00	1725.0
MntFishProducts	2240.0	37.525446	54.628979	0.0	3.00	12.0	50.00	259.0
MntSweetProducts	2240.0	27.062946	41.280498	0.0	1.00	8.0	33.00	263.0
MntGoldProds	2240.0	44.021875	52.167439	0.0	9.00	24.0	56.00	362.0
NumDealsPurchases	2240.0	2.325000	1.932238	0.0	1.00	2.0	3.00	15.0
NumWebPurchases	2240.0	4.084821	2.778714	0.0	2.00	4.0	6.00	27.0
NumCatalogPurchases	2240.0	2.662054	2.923101	0.0	0.00	2.0	4.00	28.0
NumStorePurchases	2240.0	5.790179	3.250958	0.0	3.00	5.0	8.00	13.0
NumWebVisitsMonth	2240.0	5.316518	2.426645	0.0	3.00	6.0	7.00	20.0
Kidhome	2240.0	0.444196	0.538398	0.0	0.00	0.0	1.00	2.0
Teenhome	2240.0	0.506250	0.544538	0.0	0.00	0.0	1.00	2.0

**Observations:

    * There is a huge difference between the 3rd quartile and the maximum value for all products purchased in the last   2 years indicating there might be outliers to the right in these variables

    * Each Customer had a maximum of two children either kidhome or teenhome

    * it is likely customers income was spent more on wine

    * There seems to be a huge value for income outlier to the right

Create a list for categorical columns in the dataset and check the count of each category

In [11]:

```
#cat_cols contain categorical variables
cat_cols=['Education', 'Marital_Status', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'AcceptedCmp6', 'Complain', 'Country']
```

In [12]:

```
# Printing the count of each unique value in each column

for column in cat_cols:
    print(df[column].value_counts(normalize=True))
    print("-" * 40)
```

```
Graduation    0.503125
PhD           0.216964
Master        0.165179
2n Cycle      0.090625
Basic         0.024107
Name: Education, dtype: float64
----------------------------------------
Married     0.385714
Together    0.258929
Single      0.214286
Divorced    0.103571
Widow       0.034375
Alone       0.001339
YOLO        0.000893
Absurd      0.000893
Name: Marital_Status, dtype: float64
----------------------------------------
0    0.927232
1    0.072768
Name: AcceptedCmp3, dtype: float64
----------------------------------------
0    0.935714
1    0.064286
Name: AcceptedCmp4, dtype: float64
----------------------------------------
0    0.925446
1    0.074554
Name: AcceptedCmp5, dtype: float64
----------------------------------------
0    0.927232
1    0.072768
Name: AcceptedCmp1, dtype: float64
----------------------------------------
0    0.850893
1    0.149107
Name: AcceptedCmp2, dtype: float64
----------------------------------------
0    0.986607
1    0.013393
Name: AcceptedCmp6, dtype: float64
----------------------------------------
0    0.990625
1    0.009375
Name: Complain, dtype: float64
----------------------------------------
SP     0.488839
SA     0.150446
CA     0.119643
AUS    0.071429
IND    0.066071
GER    0.053571
US     0.048661
ME     0.001339
Name: Country, dtype: float64
----------------------------------------
```

Observations:

- In education, 2n cycle and Master means the same thing. We can combine these two categories.
- There are many categories in marital status. We can combine the category 'Alone' with 'Single'.
- It is not clear from the data that what do the terms 'Absurd', and 'YOLO' actually mean. We can combine these categories to make a new category - 'Others'.
- There are only 21 customers who complained in the last two years.
- The majority of the customers belong to Spain and least to Mexico.
- The most common educational status is Graduation
- The most common marital status is Married

Data Preprocessing and Exploratory Data Analysis

- Fixing the categories
- Creating new columns as the total amount spent, total purchase made, total kids at home, and total accepted campaigns
- Dealing with missing values and outliers
- Extract key insights from the data

Replacing the "2n Cycle" category with "Master" in Education and "Alone" with "Single" in Marital_Status and "Absurd" and "YOLO" categories with "Others" in Marital_Status

In [13]:

```
# Replacing 2n Cycle with Master

df["Education"].replace("2n Cycle", "Master", inplace=True)
```

In [14]:

```
# Replacing Alone with Single

df["Marital_Status"].replace(["Alone",], "Single", inplace=True)
```

In [15]:

```
# Replacing YOLO, Absurd with Others

df['Marital_Status'].replace(["Absurd", "YOLO"], "Others", inplace=True)
```

We have fixed the categories in the Marital_Status. Check distribution count in different categories for marital status.

In [16]:

```
df.Marital_Status.value_counts()
```

Out[16]:

```
Married     864
Together    580
Single      483
Divorced    232
Widow        77
Others        4
Name: Marital_Status, dtype: int64
```

Observation:

- The majority of customer belong to married category and the other category have only 4 observations.

Creating new features from the existing features

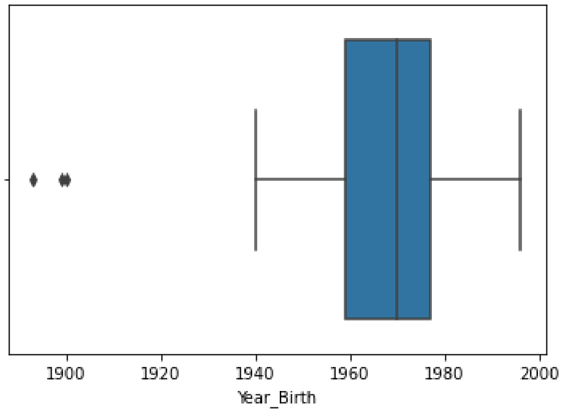
In [17]:

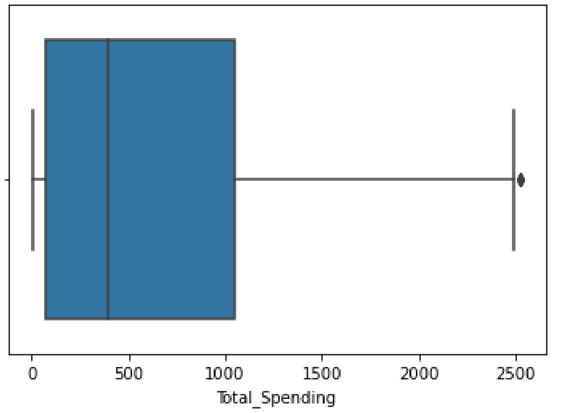
```
# creating new features to get overall picture of a customer, how much he/she has spend,
#how many children he/she has, total campaigns accepted, etc.


# total spending by a customer
spending_col = [col for col in df.columns if 'Mnt' in col]
df['Total_Spending'] = df[spending_col].sum(axis = 1)

#total purchases made by a customer
platform_col = [col for col in df.columns if 'Purchases' in col]
df['Total_Purchase'] = df[platform_col].sum(axis = 1)

#total no. of childern
df['NumberofChildren'] = df['Kidhome'] + df['Teenhome']

# Total no. of campaign accepted by a customer
campaigns_cols = [col for col in df.columns if 'Cmp' in col]
df['TotalCampaignsAcc'] = df[campaigns_cols].sum(axis=1)
```

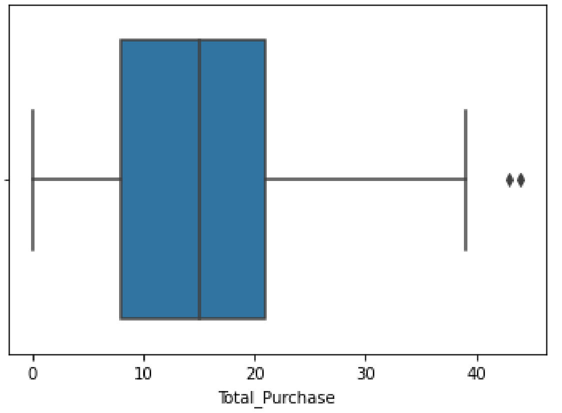
Check outliers for new variables - Total_Spending, Total_Purchase and also analyze the Year_Birth column as we observed above that it had a minimum value of 1893.

In [18]:

```
# Plotting boxplot for Year_Birth, Total_Spending, Total_Purchase

cols=['Year_Birth','Total_Spending','Total_Purchase']
for i in cols:
    sns.boxplot(x=df[i])
    plt.show()
```







Observations:

- The birth year is reported as <=1900 for some users, while the current year is 2021. it's very unlikely that the person is alive. it may be a reporting error.
- There are some outliers in total spending and total purchase.
- The observations marked as outliers are very closed to the upper whisker and some extreme points can be expected for variables like total spending. We can leave these outliers untreated.

Check the number of observations for which year birth is less than 1900.

In [19]:

```
df[df['Year_Birth'] < 1900]
```

Out[19]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	...	AcceptedCmp3	AcceptedCmp4	Accepte
513	11004	1893	Master	Single	60182.0	0	1	23	8	0	...	0	0	
827	1150	1899	PhD	Together	83532.0	0	0	36	755	144	...	1	0	

2 rows × 31 columns

Observation:

- There are only 2 observations for which birth year is less than 1900. We can drop these observations.

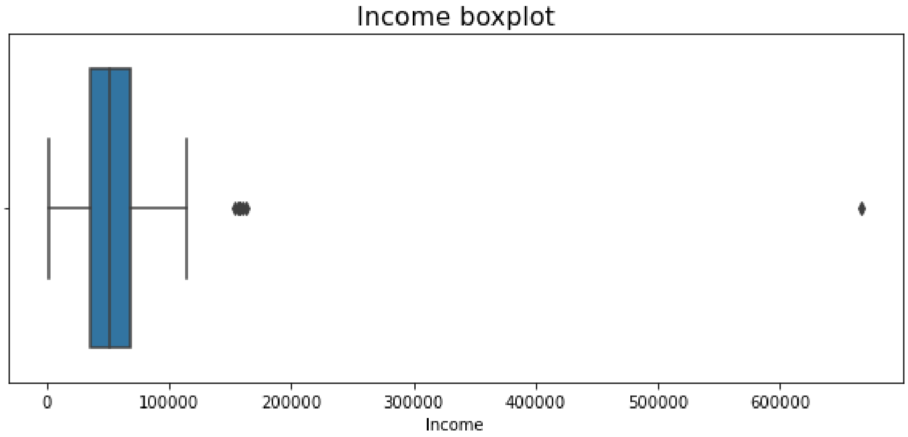
In [20]:

```
#keeping data for customers having birth year >1900

df = df[df['Year_Birth'] > 1900]
```

Check the outliers and impute the missing values for the Income variable

In [21]:

```
#plotting Boxplot for income

plt.figure(figsize=(10,4))
sns.boxplot(df['Income'])
plt.title('Income boxplot', size=16)
plt.show()
```



Observations:

- We can see from the boxplot that there are some outliers in the income variable.
- Find the value at upper whisker to check how many observations are marked as outliers.

In [22]:

```
#Calculating the upper whisker for the Income variable

Q1 = df.quantile(q=0.25) #First quartile
Q3 = df.quantile(q=0.75) #Third quartile
IQR = Q3 - Q1            #Inter Quartile Range

upper_whisker = (Q3 + 1.5*IQR)['Income']   #Upper Whisker
print(upper_whisker)
```

```
118348.5
```

In [23]:

```
#Checking the observations marked as outliers
df[df.Income>upper_whisker]
```

Out[23]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	...	AcceptedCmp3	AcceptedCmp4	Accep
325	4931	1977	Graduation	Together	157146.0	0	0	13	1	0	...	0	0	
497	1501	1982	PhD	Married	160803.0	0	0	21	55	16	...	0	0	
527	9432	1977	Graduation	Together	666666.0	1	0	23	9	14	...	0	0	
731	1503	1976	PhD	Together	162397.0	1	1	31	85	1	...	0	0	
853	5336	1971	Master	Together	157733.0	1	0	37	39	1	...	0	0	
1826	5555	1975	Graduation	Divorced	153924.0	0	0	81	1	1	...	0	0	
1925	11181	1949	PhD	Married	156924.0	0	0	85	2	1	...	0	0	
2204	8475	1973	PhD	Married	157243.0	0	1	98	20	2	...	0	0	

8 rows × 31 columns

Observations:

- We have only 8 observations with an income greater than the upper whisker.
- Only 3 observations (ID- 4931, 1501, 8475) out of 8 outliers have purchased more than 11 times in the last 2 years.
- Other 5 observations have very less amount of total spending.

Compare the summary statistics for these observations with observations on the other side of the upper whisker.

In [24]:

```
#Checking the summary statistics for observations marked as outliers
df[df.Income>upper_whisker].describe().T
```

Out[24]:

	count	mean	std	min	25%	50%	75%	max
ID	8.0	5989.250	3525.251308	1501.0	4074.00	5445.5	8714.25	11181.0
Year_Birth	8.0	1972.500	10.028531	1949.0	1972.50	1975.5	1977.00	1982.0
Income	8.0	221604.500	179850.404431	153924.0	157090.50	157488.0	161201.50	666666.0
Kidhome	8.0	0.375	0.517549	0.0	0.00	0.0	1.00	1.0
Teenhome	8.0	0.250	0.462910	0.0	0.00	0.0	0.25	1.0
Recency	8.0	48.625	33.687376	13.0	22.50	34.0	82.00	98.0
MntWines	8.0	26.500	30.798887	1.0	1.75	14.5	43.00	85.0
MntFruits	8.0	4.500	6.524678	0.0	1.00	1.0	5.00	16.0
MntMeatProducts	8.0	621.875	846.511402	1.0	7.25	17.0	1592.00	1725.0
MntFishProducts	8.0	4.250	5.650537	1.0	1.00	2.0	3.50	17.0
MntSweetProducts	8.0	1.250	0.886405	0.0	1.00	1.0	1.25	3.0
MntGoldProds	8.0	3.750	4.131759	1.0	1.00	1.5	5.00	12.0
NumDealsPurchases	8.0	4.250	6.777062	0.0	0.00	0.0	6.75	15.0
NumWebPurchases	8.0	0.500	1.069045	0.0	0.00	0.0	0.25	3.0
NumCatalogPurchases	8.0	9.875	13.484780	0.0	0.00	0.5	23.50	28.0
NumStorePurchases	8.0	0.750	1.035098	0.0	0.00	0.5	1.00	3.0
NumWebVisitsMonth	8.0	1.125	2.031010	0.0	0.00	0.5	1.00	6.0
AcceptedCmp1	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
AcceptedCmp2	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
AcceptedCmp3	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
AcceptedCmp4	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
AcceptedCmp5	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
AcceptedCmp6	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
Complain	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0
Total_Spending	8.0	662.125	848.380884	6.0	46.25	84.5	1635.25	1730.0
Total_Purchase	8.0	15.375	18.220377	0.0	0.75	6.5	30.25	44.0
NumberofChildren	8.0	0.625	0.744024	0.0	0.00	0.5	1.00	2.0
TotalCampaignsAcc	8.0	0.000	0.000000	0.0	0.00	0.0	0.00	0.0

In [25]:

```
#Checking the summary statistics for observations not marked as outliers
df[df.Income<upper_whisker].describe().T
```

Out[25]:

	count	mean	std	min	25%	50%	75%	max
ID	2205.0	5585.439456	3247.546423	0.0	2815.0	5455.0	8418.0	11191.0
Year_Birth	2205.0	1968.904308	11.705801	1940.0	1959.0	1970.0	1977.0	1996.0
Income	2205.0	51622.094785	20713.063826	1730.0	35196.0	51287.0	68281.0	113734.0
Kidhome	2205.0	0.442177	0.537132	0.0	0.0	0.0	1.0	2.0
Teenhome	2205.0	0.506576	0.544380	0.0	0.0	0.0	1.0	2.0
Recency	2205.0	49.009070	28.932111	0.0	24.0	49.0	74.0	99.0
MntWines	2205.0	306.164626	337.493839	0.0	24.0	178.0	507.0	1493.0
MntFruits	2205.0	26.403175	39.784484	0.0	2.0	8.0	33.0	199.0
MntMeatProducts	2205.0	165.312018	217.784507	0.0	16.0	68.0	232.0	1725.0
MntFishProducts	2205.0	37.756463	54.824635	0.0	3.0	12.0	50.0	259.0
MntSweetProducts	2205.0	27.128345	41.130468	0.0	1.0	8.0	34.0	262.0
MntGoldProds	2205.0	44.057143	51.736211	0.0	9.0	25.0	56.0	321.0
NumDealsPurchases	2205.0	2.318367	1.886107	0.0	1.0	2.0	3.0	15.0
NumWebPurchases	2205.0	4.100680	2.737424	0.0	2.0	4.0	6.0	27.0
NumCatalogPurchases	2205.0	2.645351	2.798647	0.0	0.0	2.0	4.0	28.0
NumStorePurchases	2205.0	5.823583	3.241796	0.0	3.0	5.0	8.0	13.0
NumWebVisitsMonth	2205.0	5.336961	2.413535	0.0	3.0	6.0	7.0	20.0
AcceptedCmp1	2205.0	0.073923	0.261705	0.0	0.0	0.0	0.0	1.0
AcceptedCmp2	2205.0	0.151020	0.358150	0.0	0.0	0.0	0.0	1.0
AcceptedCmp3	2205.0	0.073016	0.260222	0.0	0.0	0.0	0.0	1.0
AcceptedCmp4	2205.0	0.064399	0.245518	0.0	0.0	0.0	0.0	1.0
AcceptedCmp5	2205.0	0.074376	0.262442	0.0	0.0	0.0	0.0	1.0
AcceptedCmp6	2205.0	0.013605	0.115872	0.0	0.0	0.0	0.0	1.0
Complain	2205.0	0.009070	0.094827	0.0	0.0	0.0	0.0	1.0
Total_Spending	2205.0	606.821769	601.675284	5.0	69.0	397.0	1047.0	2525.0
Total_Purchase	2205.0	14.887982	7.615277	0.0	8.0	15.0	21.0	43.0
NumberofChildren	2205.0	0.948753	0.749231	0.0	0.0	1.0	1.0	3.0
TotalCampaignsAcc	2205.0	0.450340	0.894075	0.0	0.0	0.0	1.0	5.0

Observations:

- None of the outliers have accepted any of the campaigns or have submitted any complaints in the last 2 years.
- We can see that customers who are outliers have lower mean expenditure per customer for all the products except meat products.
- The outliers have a higher number of catalog purchases on average and very low number of web purchases.
- We can drop the 5 observations at indices [527, 731, 853, 1826, 1925] as they would not add value to our analysis.

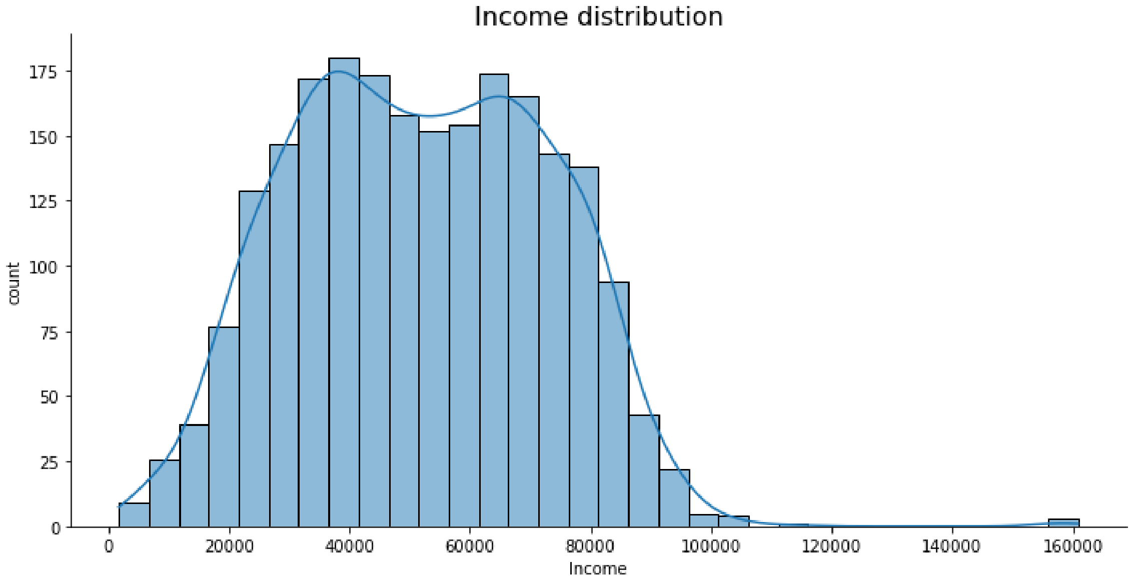
In [26]:

```
#Dropping 5 observations at indices 527, 731, 853, 1826, 1925
df.drop(index=[527, 731, 853, 1826, 1925], inplace=True)
```

Check the distribution for Income

In [27]:

```
#plotting displot for income

sns.displot(df['Income'], kde=True, height=5, aspect=2)
plt.title('Income distribution', size=16, )
plt.ylabel('count');
```



Observations:

- After treating outliers, the distribution for the income variable is close to normal distribution with very few extreme observations to the right.
- Replace the missing values for the income variable with the median, and not mean, as the variable is slightly skewed to the right

In [28]:

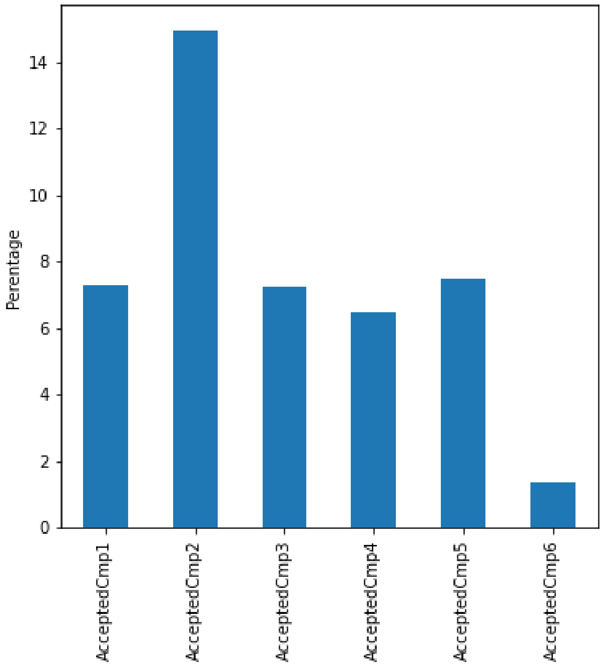
```
#filling null values with median

df['Income'].fillna(df.Income.median(), inplace=True)
```

Analyzing all the campaigns

find out what is the acceptance rate for each campaign?

In [29]:

```
# PLotting the % acceptance for every campaign

Camp_cols=['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp6']

success_campaign=(df[Camp_cols].sum()/df.shape[0])*100

# plot
success_campaign.plot(kind='bar', figsize=(6,6))
plt.ylabel("Perentage")
plt.show()
```



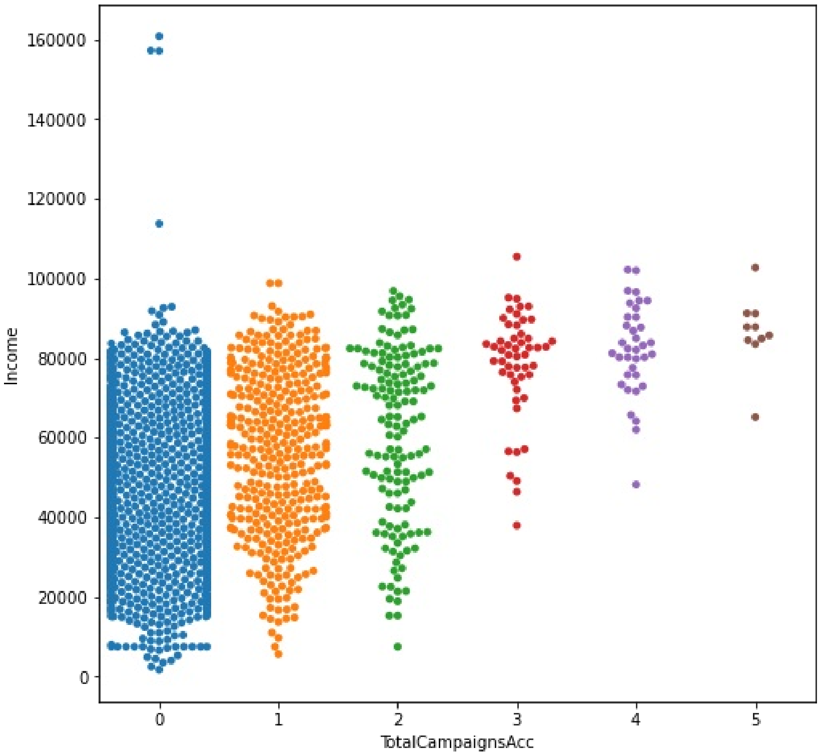
**Observations:

* 6th Campaigne had the lowest Acceptance rate while 2nd Campaigne had the highest Acceptance rate compaired to other campaigns
* 1st, 3rd, 4th and 5th Campaign had close range acceptance rate

Analyze what kind of customer are accepting campaigns?

In [30]:

```
plt.figure(figsize=(8,8))
sns.swarmplot(x='TotalCampaignsAcc', y='Income', data=df)
plt.show()
```



Observations:

- Higher the income higher the number of campaigns accepted.

In [31]:

```
# Let's see the mean income of customers
df.Income.mean()
```

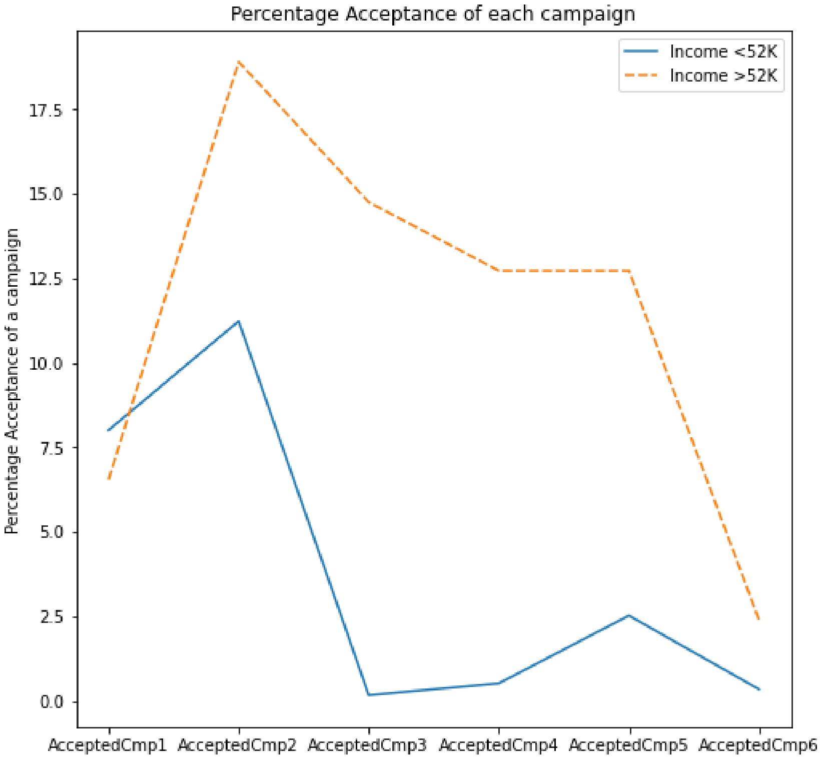
Out[31]:

```
51762.59811827957
```

The mean income of customers is close to 52K. Let's divide the income into 2 segments of income>52k and income<52k and see the acceptance rate in each segment.

In [32]:

```
# making dataframes of customers having income <52k and >52K
df1=df[df.Income<52000]
df2=df[df.Income>52000]

Camp_cols=['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp6']

#Calculating success rate of each campaign for both segments
success_campaign1=pd.DataFrame((df1[Camp_cols].sum()/df1.shape[0])*100, columns=['Income <52K'])

success_campaign2=pd.DataFrame((df2[Camp_cols].sum()/df2.shape[0])*100, columns=['Income >52K'])

new_df=pd.concat([success_campaign1, success_campaign2], axis=1)

# plot
plt.figure(figsize=(8,8))
sns.lineplot(data=new_df)
plt.title("Percentage Acceptance of each campaign")
plt.ylabel("Percentage Acceptance of a campaign")
plt.show()
```



**Observations:

*For Campaigne 1, customers income below 52k had more campaigne acceptance rate compared to customers with income above 52k

*For campaigne 2 to 6, the customers who had income lower than 52k had lower percentage acceptance of campaign compared to customers earning above 52k who had a higher percentage on Acceptance of campaigne.

Find out who has accepted the last campaign and what could be the reason

In [33]:

```
df[df['AcceptedCmp6']==1].shape
```

Out[33]:

```
(30, 31)
```

- There are only 30 customers who have accepted the last campaign.
- Let's check if these customers are new or they have accepted previous campaigns as well.

In [34]:

```
grouped2=df.groupby('AcceptedCmp6').mean()['TotalCampaignsAcc']
grouped2
```

Out[34]:

```
AcceptedCmp6
0    0.404632
1    3.633333
Name: TotalCampaignsAcc, dtype: float64
```

Observations:

- We know that the maximum number of campaigns any customer has accepted is 5.
- We can observe that the value for TotalCampaignsAcc is ~3.6 for customers who have accepted the last campaign.
- This implies that these 30 customers are those loyal customers who have been accepting most of the campaigns.

It could be that different campaigns are focussed on different set of products. Check if the product preference for those who accepted the campaigns is different from those who didn't - using amount spent and number of purchases

Define a function which will take the column name for the product as input and will generate the barplot for every campaign and average amount spent on a product

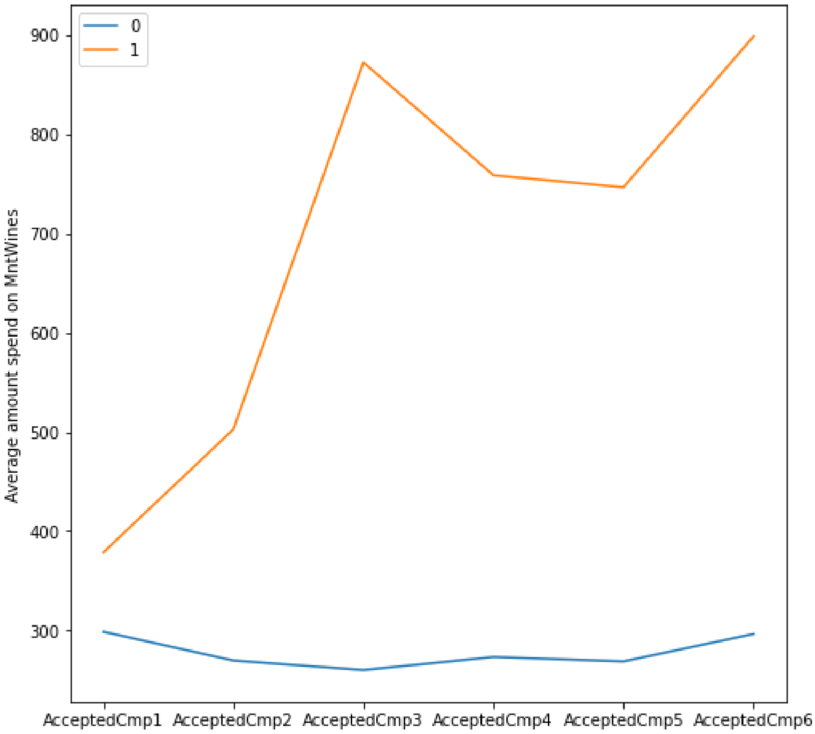
In [35]:

```
def amount_per_campaign(columns_name):
    p1=pd.DataFrame(df.groupby(['AcceptedCmp1']).mean()[columns_name]).T
    p2=pd.DataFrame(df.groupby(['AcceptedCmp2']).mean()[columns_name]).T
    p3=pd.DataFrame(df.groupby(['AcceptedCmp3']).mean()[columns_name]).T
    p4=pd.DataFrame(df.groupby(['AcceptedCmp4']).mean()[columns_name]).T
    p5=pd.DataFrame(df.groupby(['AcceptedCmp5']).mean()[columns_name]).T
    p6=pd.DataFrame(df.groupby(['AcceptedCmp6']).mean()[columns_name]).T
    pd.concat([p1,p2,p3,p4,p5,p6],axis=0).set_index([Camp_cols]).plot(kind='line', figsize=(8,8))
    plt.ylabel('Average amount spend on' + ' ' + columns_name)
    plt.show()
```

Use the function defined above to generate barplots for different purchasing Products

In [36]:

```
#here is an example showing how to use this function on the column MntWines
amount_per_campaign('MntWines')
```

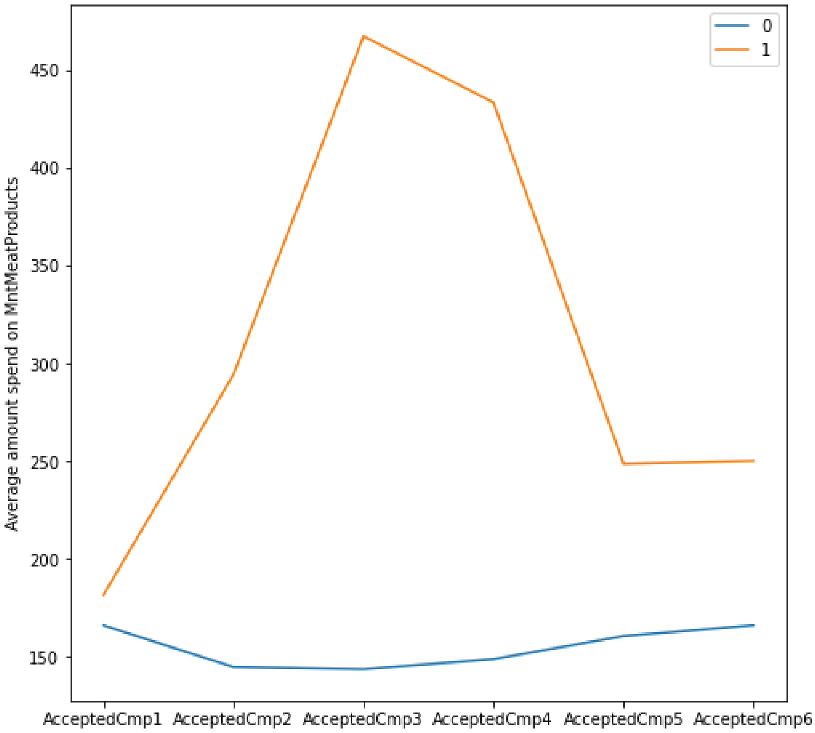


Observations:

- For the customers accepting campaign 3, 4, 5, and 6 the average amount spent on wine is quite high.

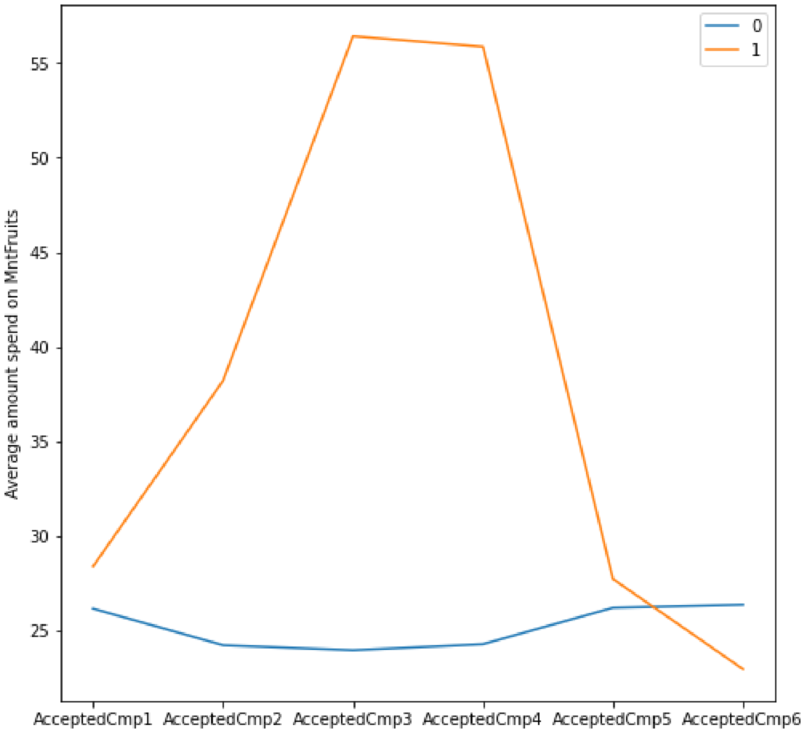
In [37]:

```
#meat products

#call the function amount_per_campaign for MntMeatProducts
amount_per_campaign('MntMeatProducts')
```



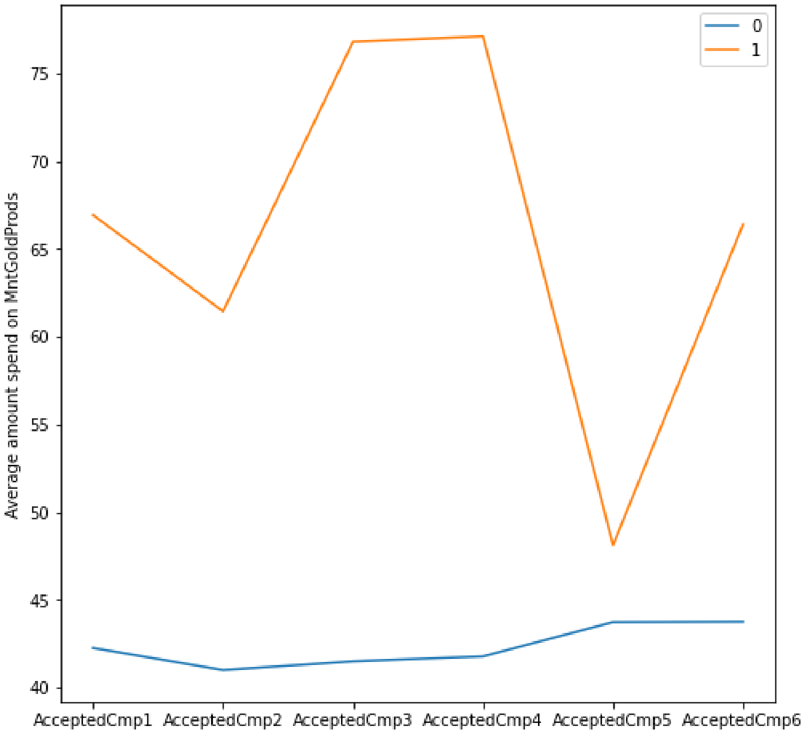
In [38]:

```
# Fruit products

#call the function amount_per_campaign for MntFruits
amount_per_campaign('MntFruits')
```



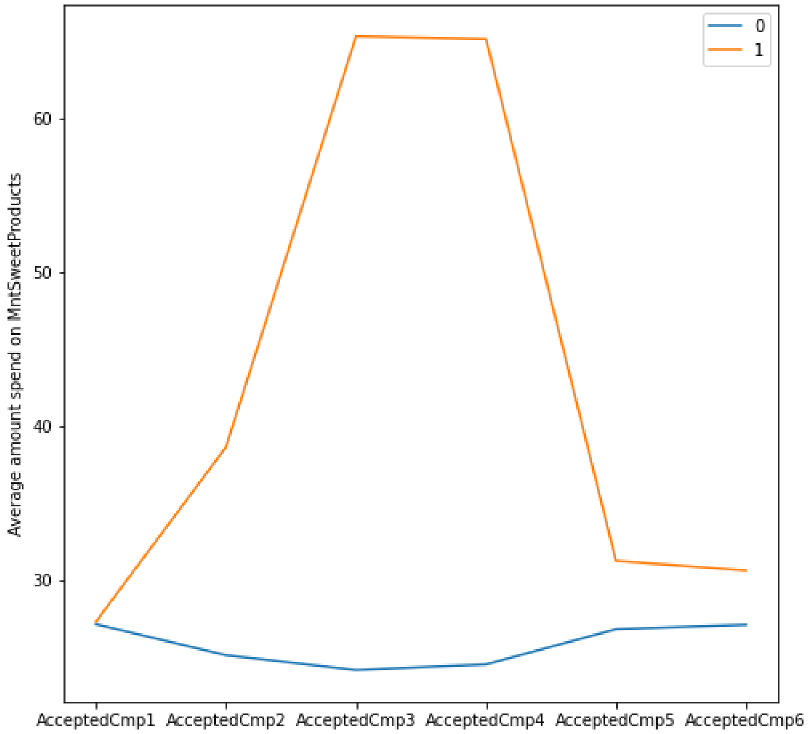
In [39]:

```
# gold products

#call the function amount_per_campaign for MntGoldProds
amount_per_campaign('MntGoldProds')
```



In [40]:

```
#sweet products

#call the function amount_per_campaign for MntSweetProducts
amount_per_campaign('MntSweetProducts')
```



**Observations- For MEAT products- for Customers that accepted campaign 2,3,4 the average amount spent on Meat products is high than 5 and 6.

For FRUIT products- campaign 6 had higher average for customers that didn't accept campaign than those that accepted campaign; accepted campaign 2,3,4 have higher average amount spent on fruit than customers who didn't accept campaign.

For GOLD products- the difference in average amount spent by customers who accepted campaign and those that didn't accept is high for campaign 1,2,3,4,6 and is low for 5

For SWEET products- Campaign 1 have almost same average amount spent both for accepted and reject campaign. There was an increase in average amount spent for accepted campaign on 2,3,4 and a sharp decrease for 5 on accepted campaign.

There is generally a low average amount spent across the four products for customers that didn't accept campaign compared to customers that accepted campaign.

Check the relationship of campaigns with different purchasing channels.

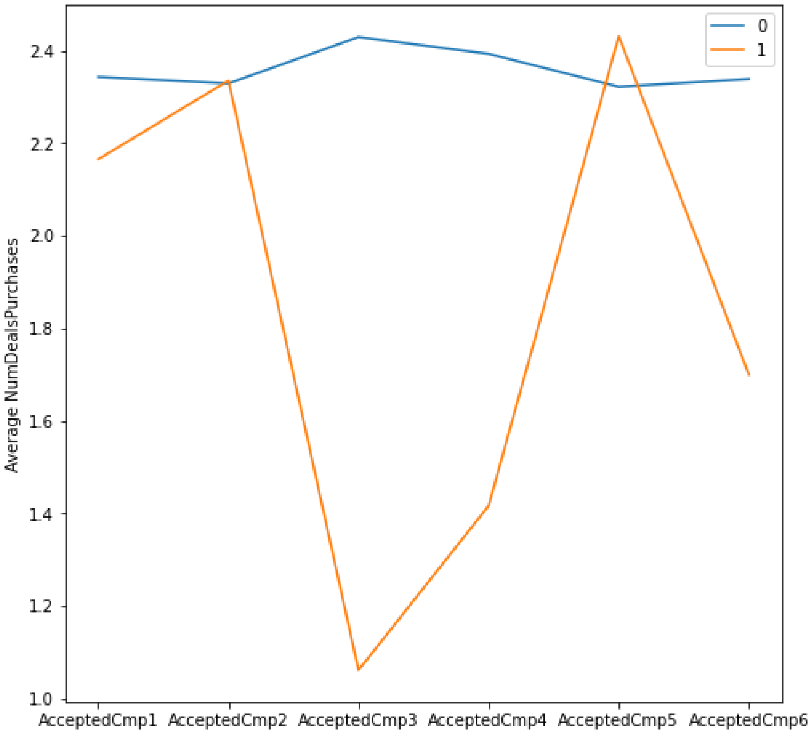
We have a defined a function which will take the column name of the channel name as input and will generate the barplot for every campaign and average purchase made through that channel if the campaign is accepted

In [41]:

```
def Purchases_per_campaign(columns_name):
    dp1=pd.DataFrame(df.groupby(['AcceptedCmp1']).mean()[columns_name]).T
    dp2=pd.DataFrame(df.groupby(['AcceptedCmp2']).mean()[columns_name]).T
    dp3=pd.DataFrame(df.groupby(['AcceptedCmp3']).mean()[columns_name]).T
    dp4=pd.DataFrame(df.groupby(['AcceptedCmp4']).mean()[columns_name]).T
    dp5=pd.DataFrame(df.groupby(['AcceptedCmp5']).mean()[columns_name]).T
    dp6=pd.DataFrame(df.groupby(['AcceptedCmp6']).mean()[columns_name]).T
    pd.concat([dp1,dp2,dp3,dp4,dp5,dp6],axis=0).set_index([Camp_cols]).plot(kind='line', figsize=(8,8))
    plt.ylabel('Average' + ' ' + columns_name)
    plt.show()
```

In [42]:

```
#here is an example showing how to use this function on the column NumDealsPurchases
Purchases_per_campaign('NumDealsPurchases')
```

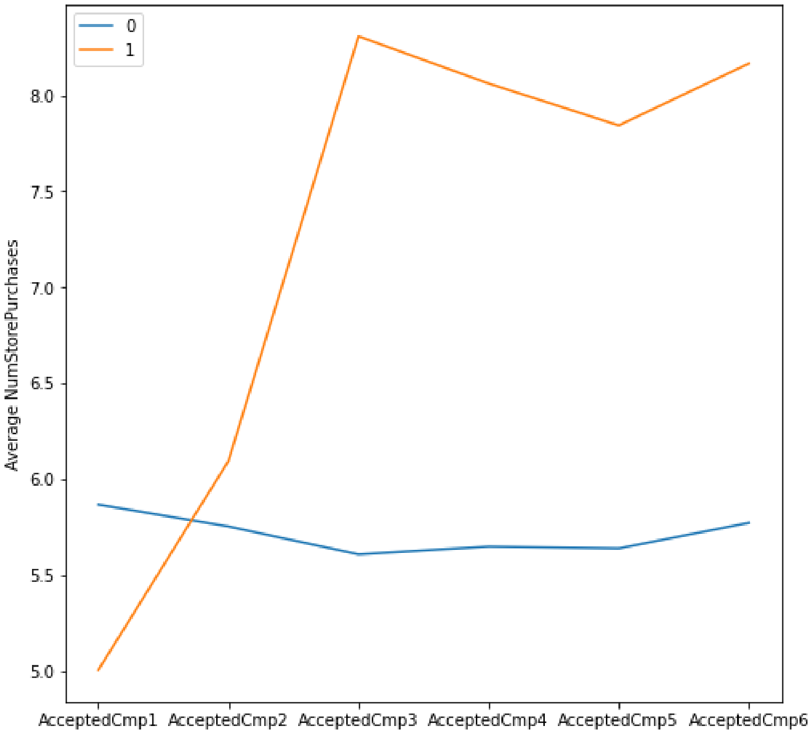


Observations:

- For the customers accepting campaign 3, 4, and 6 the average deals purchase is quite low.

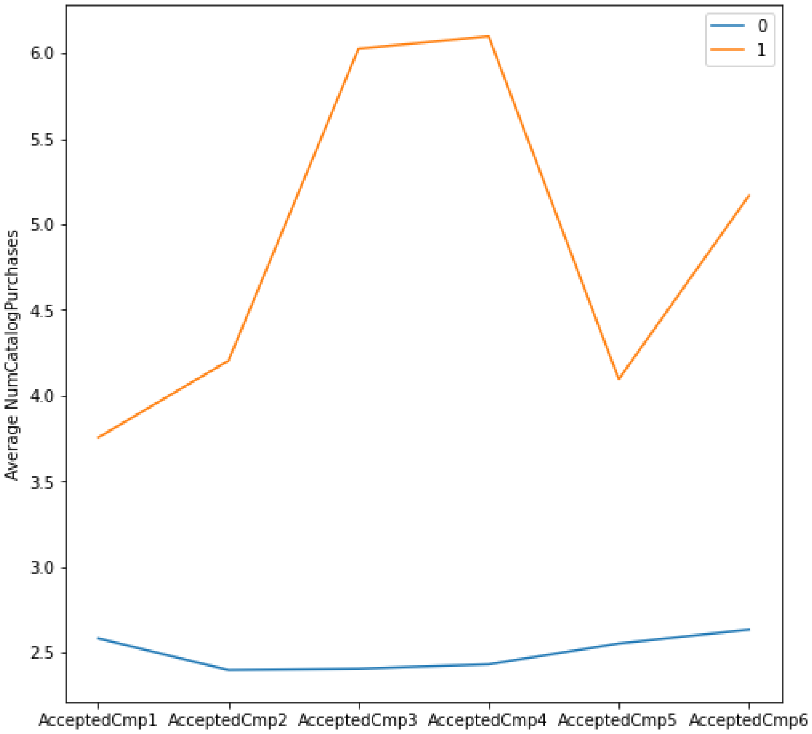
In [64]:

```
# store purchase
Purchases_per_campaign('NumStorePurchases')
#call the function Purchases_per_campaign for NumStorePurchases
```



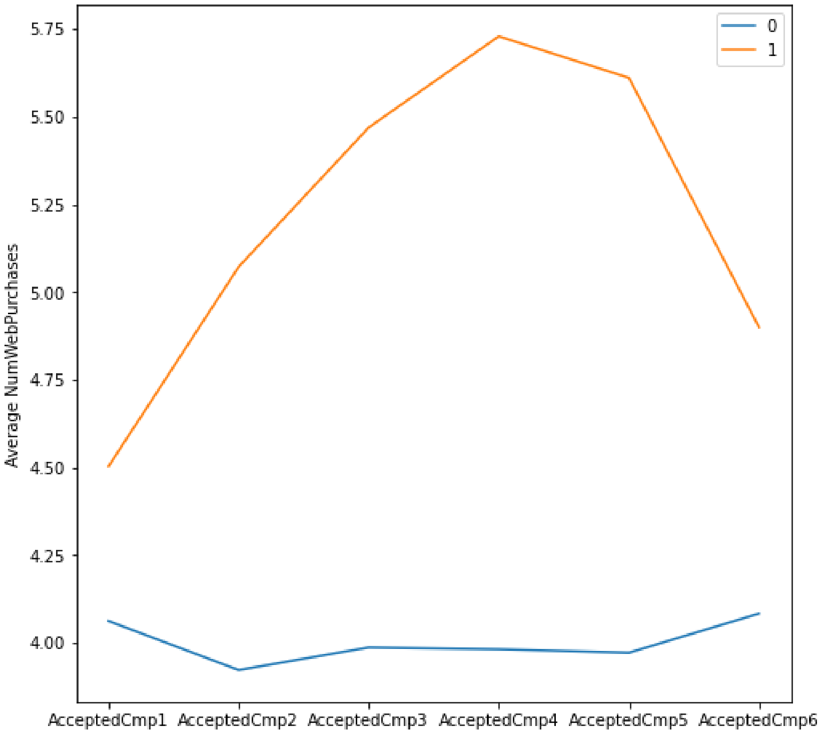
In [65]:

```
#Catalog purchase
Purchases_per_campaign('NumCatalogPurchases')
#call the function Purchases_per_campaign for NumCatalogPurchases
```



In [67]:

```
#Web purchases
Purchases_per_campaign('NumWebPurchases')
#call the function Purchases_per_campaign for NumWebPurchases
```



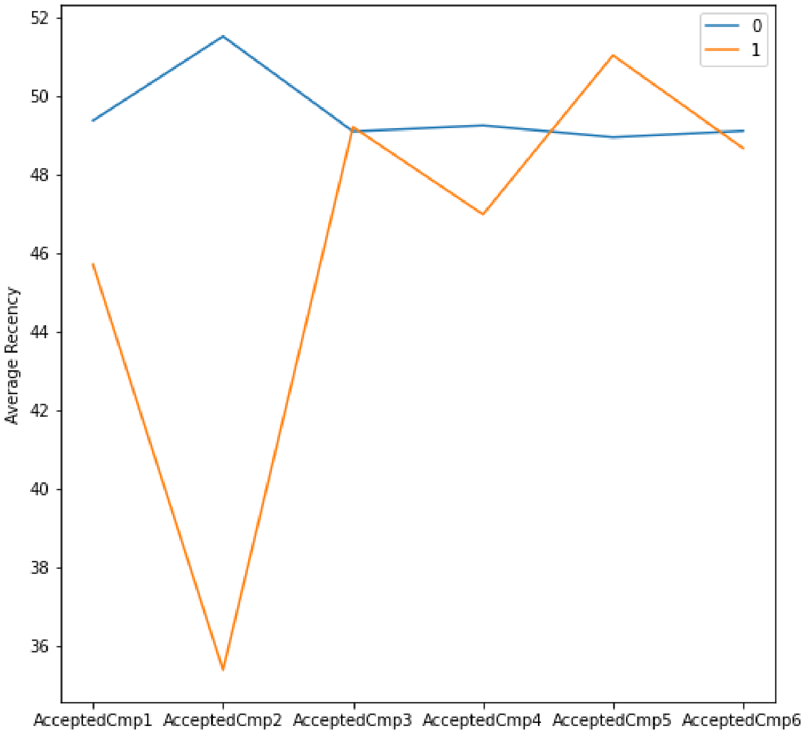
**Observations: NumStorePurchases- Unlike campaign 2,3,4,5,6; customers accepting campaign 1 have averagely lower number of store purchases than customers who rejected campaign 1.

NumCatalogPurchases- Though the 6 rejected campaigns are low compaired to accepted campaigns, the Accepted campaign for 3 and 4 have higher average number of catalog purchases than 1,2,5 and 6

NumWebPurchases- There is a rise of average number of web purchase from accepted campaign 1 to 4 then a gradual drop from accepted campaign 5 to 6

In [68]:

```
#Recency

Purchases_per_campaign('Recency')
```



Observations:

- Average recency of the customers who accepted campaign 2 is quite low which implies that campaign 2 was accepted by the customers who recently purchased an item.

Check see the relationship of campaigns with different categorical variables

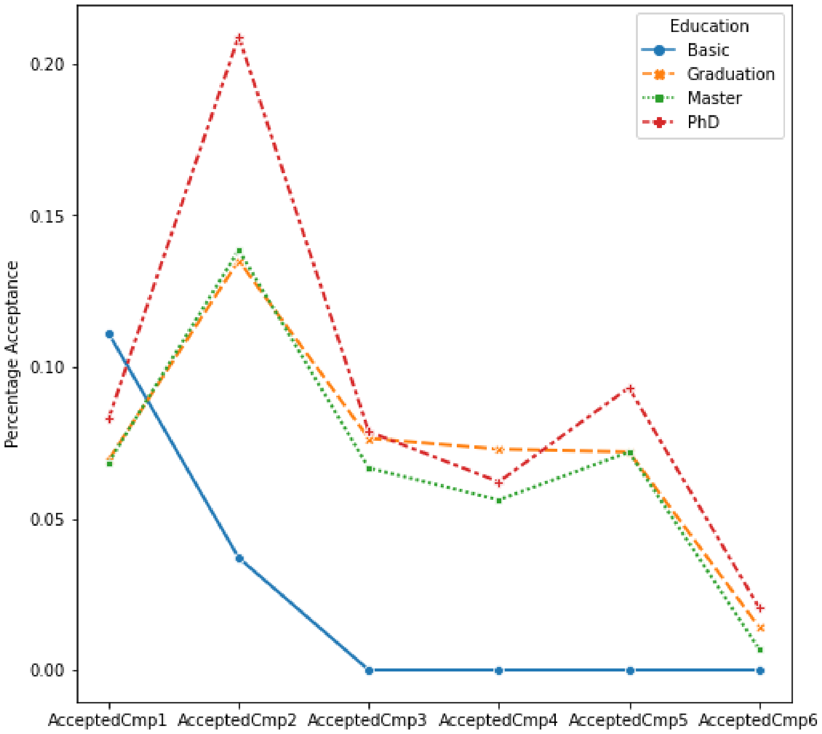
Check the percentage acceptance of each campaign with respect to each category in the categorical variable. The percentage acceptance is calculated as number of customers who have accepted the campaign to the total number of customers.

In [69]:

```
def Cat_Campaign_Relation(df, column_name):
    e1=(df.groupby([column_name]).sum()['AcceptedCmp1']/df.groupby([column_name]).count()['AcceptedCmp1'])
    e2=(df.groupby([column_name]).sum()['AcceptedCmp2']/df.groupby([column_name]).count()['AcceptedCmp2'])
    e3=(df.groupby([column_name]).sum()['AcceptedCmp3']/df.groupby([column_name]).count()['AcceptedCmp3'])
    e4=(df.groupby([column_name]).sum()['AcceptedCmp4']/df.groupby([column_name]).count()['AcceptedCmp4'])
    e5=(df.groupby([column_name]).sum()['AcceptedCmp5']/df.groupby([column_name]).count()['AcceptedCmp5'])
    e6=(df.groupby([column_name]).sum()['AcceptedCmp6']/df.groupby([column_name]).count()['AcceptedCmp6'])
    df_new=pd.concat([e1,e2,e3,e4,e5,e6],axis=1).T
    plt.figure(figsize=(8,8))
    sns.lineplot(data=df_new, markers=True, linewidth=2)
    plt.ylabel('Percentage Acceptance')
    plt.show()
```

In [70]:

```
#here is an example showing how to use this function on the column Education
Cat_Campaign_Relation(df, 'Education')
```

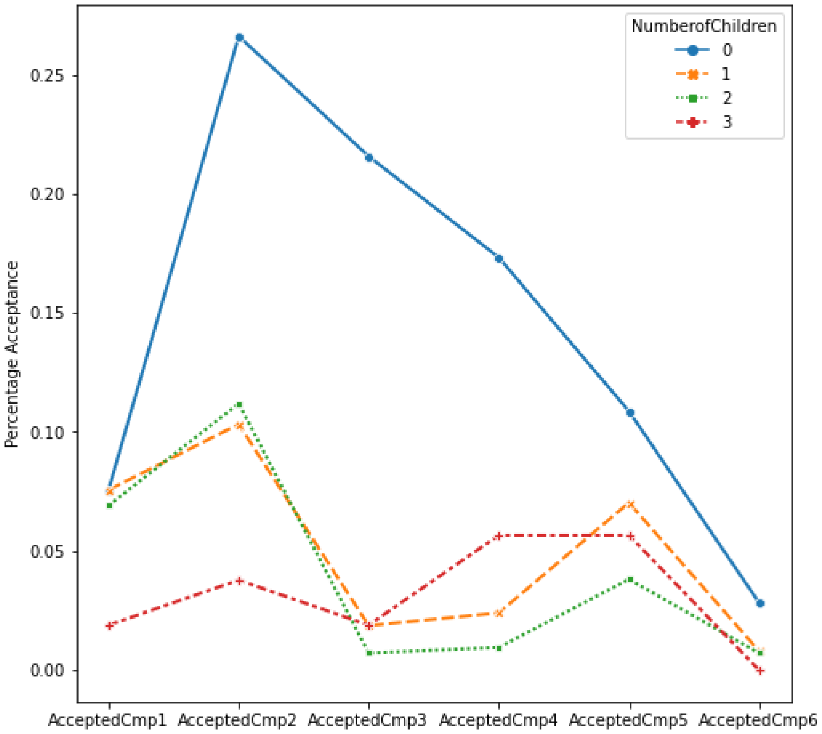


Observations:

- More than 20% of the customers with Ph.D have accepted campaign 2.
- Customers with basic education have only accepted campaign 1 and 2.
- Except customers with basic education level, all education levels follow the same trend.

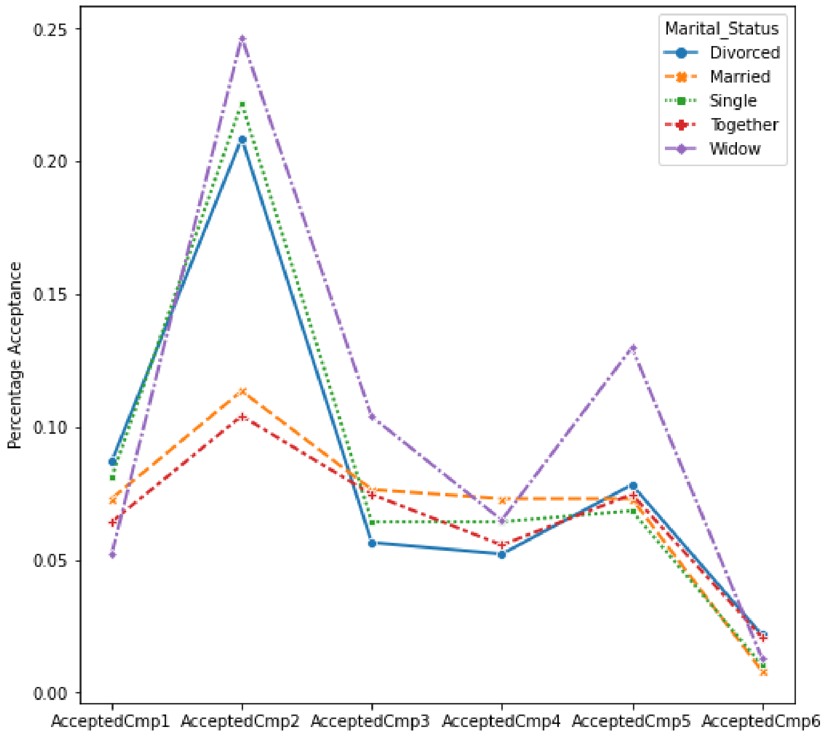
In [71]:

```
#NumberofChildren

#call the function Cat_Campaign_Relation for NumberofChildren
Cat_Campaign_Relation(df, 'NumberofChildren')
```



In [72]:

```
#Let's filter the observations with 'Others' category as they are only 4 such observations
df_rest=df[df.Marital_Status!='Others']

#call the function Cat_Campaign_Relation for Marital_Status with dataframe df_rest
Cat_Campaign_Relation(df_rest, 'Marital_Status')
```

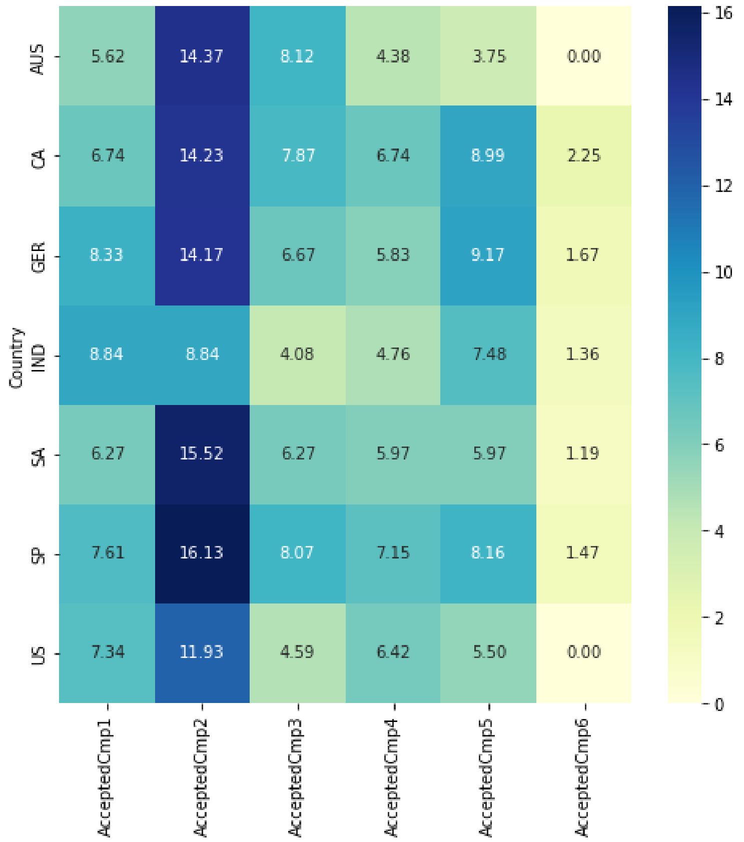


In [73]:

```
#Let's filter the observations for 'ME' country as they are only 3 such observations
df_not_mexico=df[df.Country!='ME']

#Plot
plt.figure(figsize=(8,8))
sns.heatmap((df_not_mexico.groupby('Country').sum()[Camp_cols]/df_not_mexico.groupby('Country').count()[Camp_cols])*100, annot=True, fmt='0.2f', cmap
```

Out[73]:

```
<AxesSubplot:ylabel='Country'>
```



**Observation: Number of Children- over 25% of customers who accepted campaign2 don't have children.. For campaign 1, customers with 0 to 2 children have very close range of percentage accepted campaign.. Except for customer without children, customers with 1 to 3 children somewhat have similar trend for percentage acceptance..

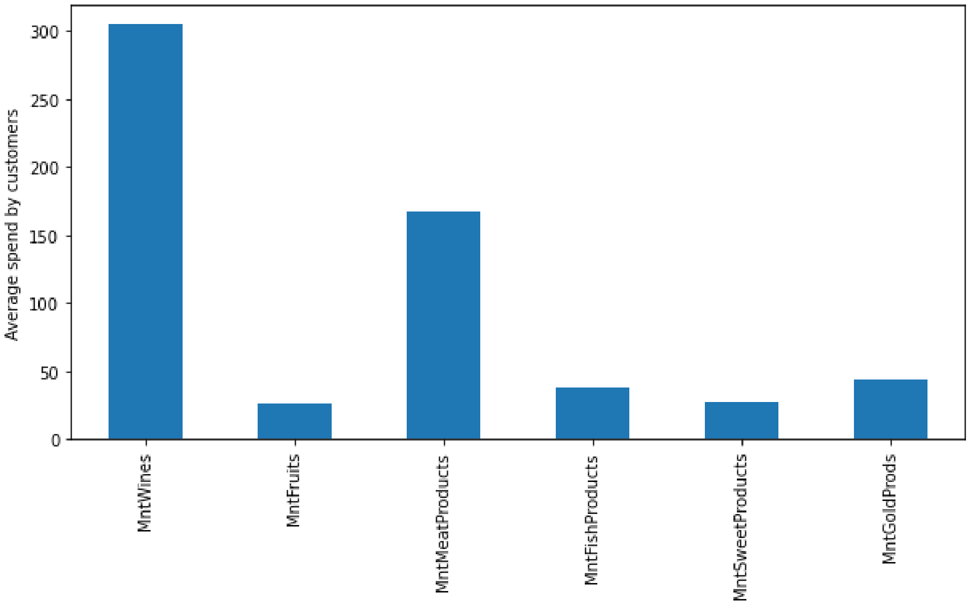
Marital Status- over 20% of customers who are widows, divorced or single accepted campaign 2.. Percentage of customers who accepted campaign 6 are very low across the 5 marital status.. All campaigns have similar trends of percentage acceptance

Country- Except for India which is lesser, the customers in other countries accepted more of campaign 2.. For all country customers in the heatmap, Accepted campaign 6 have the lowest acceptance percentage across the 6 campaigns

Check the product preferences by customers

In [74]:

```
#creating a list which contains name of all products

mnt_cols = [col for col in df.columns if 'Mnt' in col]

spending=df[mnt_cols].mean(axis=0)
spending.plot(kind='bar', figsize=(10,5))
plt.ylabel("Average spend by customers")
plt.show()
```



Observations:

- The mean amount spent by customers in the last 2 years is highest for wines followed by meat products.

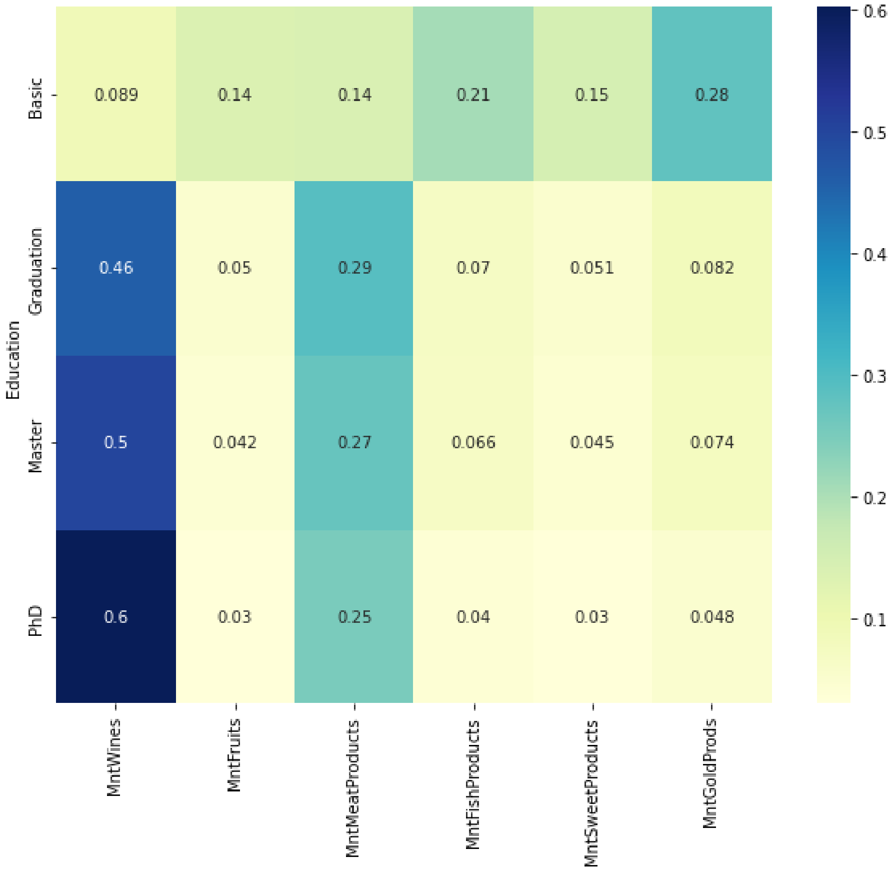
Check if the product preferences are similar for different types of customers. Then calculate the percentage amount spent by customers on a product for each category with respect to the total spending by customers belonging to that category.

In [75]:

```
def amount_per_category(df, column_name):
    df_new1=((df.groupby([column_name]).sum()[mnt_cols].T)/df.groupby([column_name]).sum()['Total_Spending'])
    plt.figure(figsize=(10,8))
    sns.heatmap(df_new1.T, annot=True, cmap="YlGnBu")
    plt.show()
```

In [76]:

```
# plot showing the percentage of total spending of different products by a group of customers having the same education level

amount_per_category(df, 'Education')
```

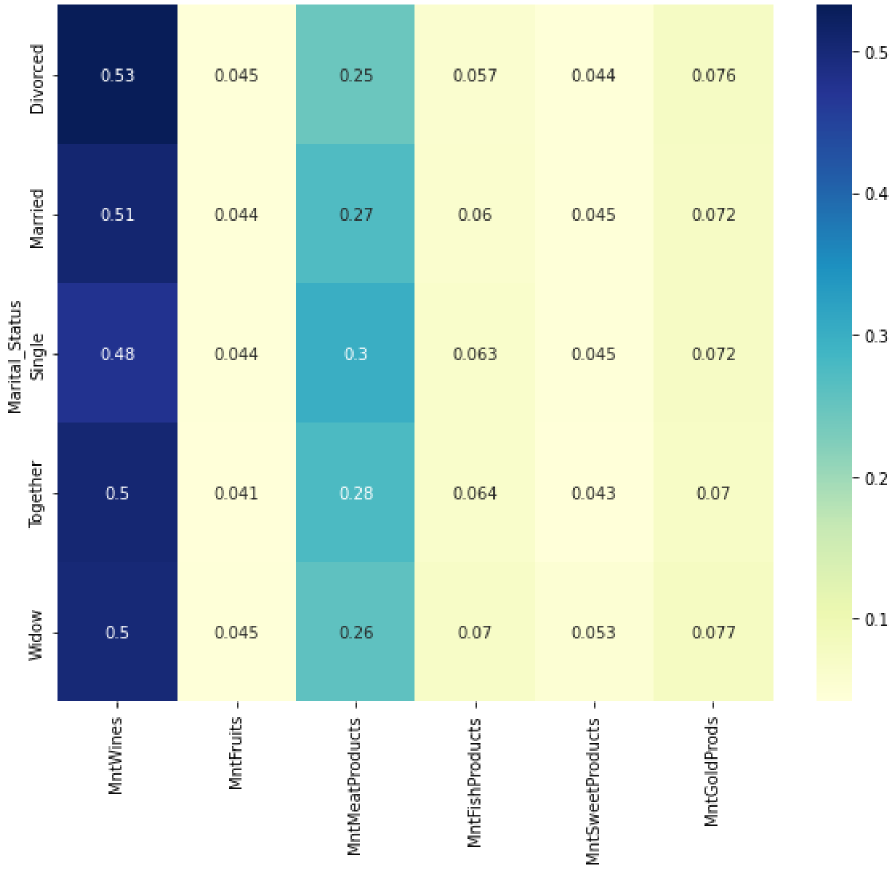


Observations:

- Customers with PhD spend ~60% of their total spending on wines.
- Customers with Graduation and Master's spend ~45-50% of their total spending on wines.
- Customers with Graduation and Master's spend ~27-29% of their total spending on meat.
- Customers with PhD spend ~25% of their total spending on meat.
- Customers having education level Master or PhD spend ~80% on meat and wines.
- Customers with basic education spend more on Fruits, Fish, Sweet, and Gold products.

In [77]:

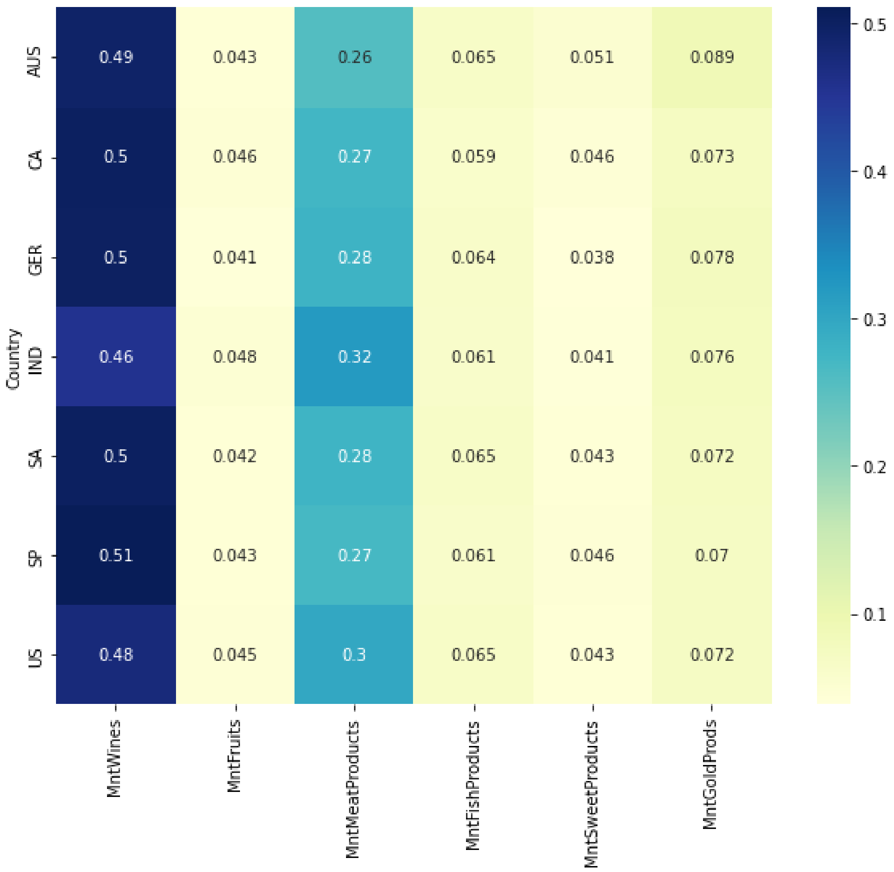
```
#call the function amount_per_category for Marital_Status with dataframe df_rest
amount_per_category( df_rest, 'Marital_Status')
```



In [78]:

```
#call the function amount_per_category for Country with dataframe df_not_mexico
```

```
amount_per_category( df_not_mexico, 'Country')
```



**Observations: Marital Status- Irrespective of all marital status, customers spent the most on wine products than other products of between 48% to 53% of their salary, followed by Meat Products.. Customers spent the least on sweet products compared to others in the last 2 years.

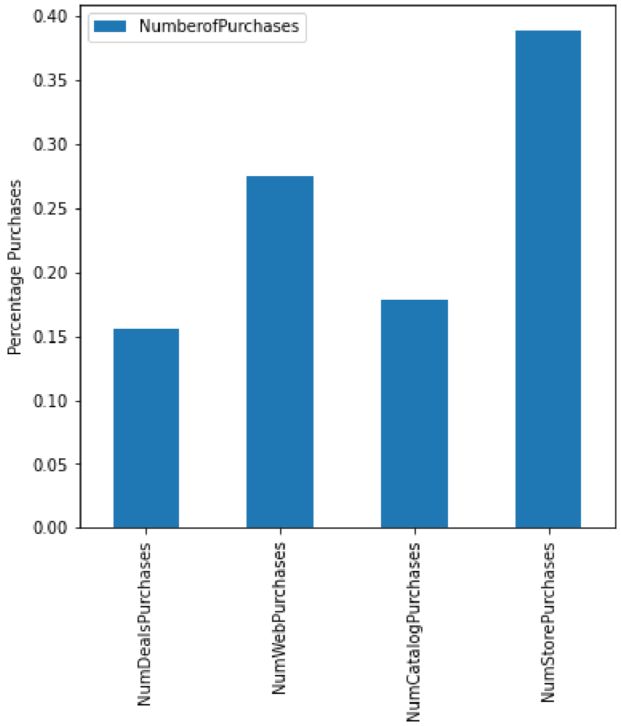
Country- Spainish Customers spent more on wine than other products compared to other countries. Generally Customers did more spending on wine across the 7 countries and spent less on Sweet products than other products for all 7 countries.

Check different channel performances

Calculate the percentage of purchases for all the channels.

In [79]:

```
# list of cols for channels

channel_cols = [col for col in df.columns if 'Purchases' in col]

#making dataframe of columns having purchase and taking sum of them.
channels = pd.DataFrame(df[channel_cols].sum()/df.Total_Purchase.sum(), columns=['NumberofPurchases'])

# plot
channels.plot(kind='bar', figsize=(6,6))
plt.ylabel("Percentage Purchases")
plt.show()
```



Observations:

- We can see that the most purchases are from the stores followed by web purchases.
- Number of deal purchases and catalog purchases are low.

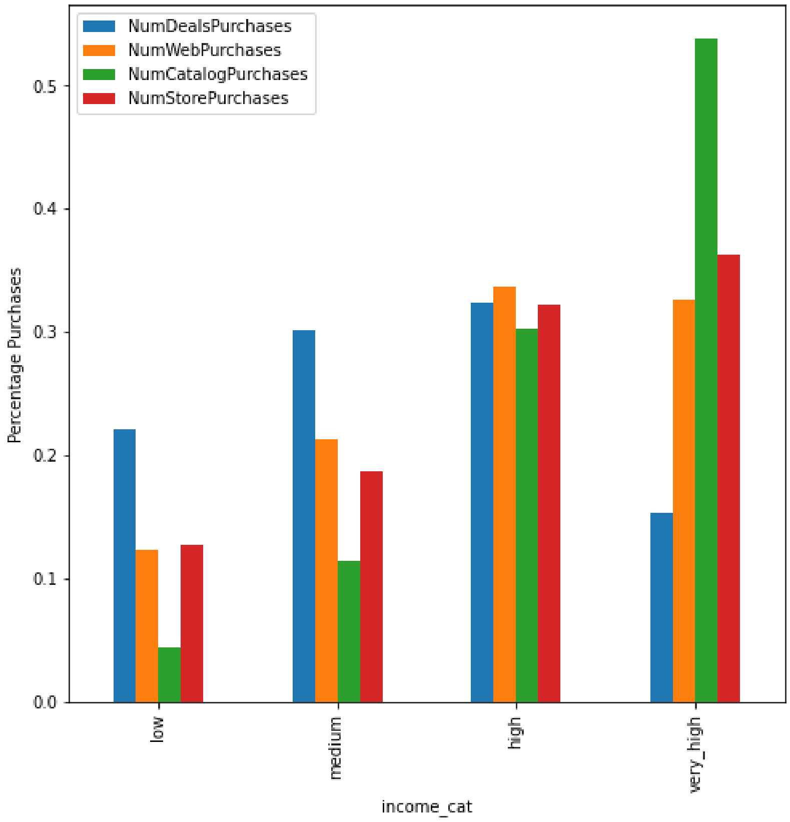
Check how number of purchases via different channels varies for different income bins.

In [58]:

```
#Binning the income column
df['income_cat']=pd.qcut(df.Income, q=[0, 0.25, 0.50, 0.75, 1], labels=['low', 'medium', 'high', 'very_high'])
```

In [59]:

```
group=df.groupby('income_cat').sum()[channel_cols]
(group/group.sum()).plot(kind='bar', figsize=(8,8))
plt.ylabel("Percentage Purchases")
plt.show()
```



**Observations: Customers having very high income had the highest percentage purchases compared to other channels used. Low income earners had the lowest percentage purchases across all channels. High Income earners had a close range of percentage of purchases for all channels.

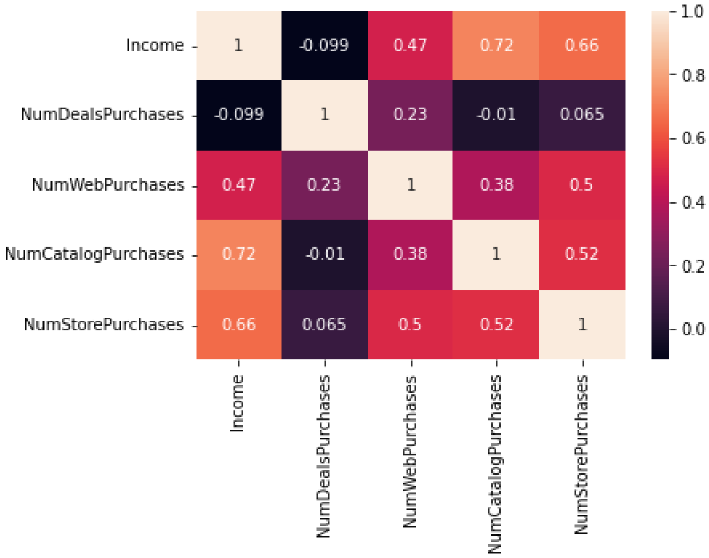
Visualize the correlation by purchases from different channels and income of the customer.

In [60]:

```
corr=df[['Income', 'NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases' ]].corr()

#Write your code here
sns.heatmap(corr, annot = True)
```

Out[60]:

```
<AxesSubplot:>
```



**Observations: The above plot shows the highest correlation 59% moderate correlation between 'Income' and 'NumCatalogPurchase', followed by correlation between 'Income' and 'NumStorePurchases' having 53%.

There is a high negative correlation between 'Income' and 'NumDealsPurchases'

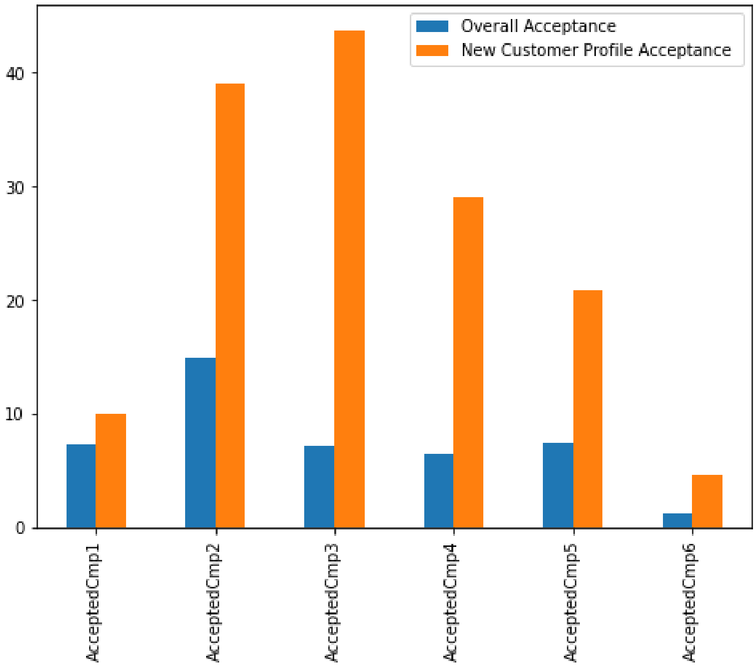
As we know from our analysis done so far that customers with income, number of children, and amount spending on wines are the important factors. Try to come up with new customer profile on the basis of these 3 attributes and check what would be the acceptance rate for that customer profile.

In [61]:

```
df3=df[df.Income>52000]
df4=df3[df3.MntWines>df3.MntWines.mean()]
new_profile=df4[df4.NumberofChildren==0]
```

In [62]:

```
#Calculating success rate of each campaign for both segments
success_campaign3=pd.DataFrame(success_campaign, columns=['Overall Acceptance'])

success_campaign4=pd.DataFrame((new_profile[Camp_cols].sum()/new_profile.shape[0])*100, columns=['New Customer Profile Acceptance '])

# plot
pd.concat([success_campaign3, success_campaign4], axis=1).plot(kind='bar', figsize=(8,6))
plt.title("")
plt.ylabel("")
plt.show()
```



Observations:

- Orange bars in the plot indicates that acceptance rate would have been high for new customer profile i.e. income greater than the mean income, no kid at home, amount spent of wines is greater than the mean amount spent on wines.

Conclusion and Recommendations

Conclusions

- Generally Customers across different countries spend more on wine than any other product.
- From the data, Customers accepted campaign 2 more than any other campaign, while campaign 6 had the least acceptance rate. Campaign strategy was most effective for Campaign 2 and least effective for Campaign 6.
- We need to find out why customers didn't accept many offers in the last Campaign, so that improvements can be made. Also Campaign 2 strategy needs to be investigated to see if it can be applied to other campaigns to effectively increase acceptance rate for 1,3,4,5,6.
- There seems to be an anomaly; the acceptance rate for Campaign 1 was higher for customers earning below 52k than customers who earned above 52k. Such anomaly needs to be investigated to know why lower income earners accepted more offers on the first trial compared to other campaigns.
- Campaign 5 for all products had a low difference between average amount spent on various products for accepted and rejected campaigns for average amount spent on various products. This Campaign should be investigated so improvements can be made or the campaign be eliminated.
- Because Campaign 3 and 4 strategy was able to get a high average amount spent for all products, this strategy may be used on Campaign 1,2, 5 and 6.
- Customers who don't have children accepted more campaign offers than those who had at least one child; so campaign strategies need to be put in place to increase campaign acceptance rate for customers who have at least one child and above.
- Generally Campaign 2 strategy seems to be effective for both customers who are not married/living together and customers from different countries. Campaign 6 strategy seems to be least effective based on both country and marital status.
- A negative correlation between income and Number of purchases made with discount means customers with lower income tend to purchase more with discounts channel and vice versa.
- For positive correlation associated with income and number of purchases made with catalog, it means customers with higher income tend to use the catalog channel which will allow them purchase goods that will be shipped through mail

Recommendations

- Generally there is a trend in increased purchases from Campaign 1 to 4 across all channels. CMO needs to acquire Data on what influenced this increase for further analysis.
- CMO needs to pay close attention to Campaign 6 and also campaign 1 Strategies.
- CMO needs to find effective Campaign strategies to increase customer purchase of sweet, gold, and fish products without affecting other products negatively irrespective of customers country or marital status.
- Also the CMO needs to pay attention to the good correlation trend for customers without children and their high rate of accepting campaign offers compared to those with at least one child.

In [ ]: