**Project: Marketing Campaign Analysis**

**Marks:60**

Welcome to the project on Foundations for Data Science. In this project, we aim to 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 and deploy data-driven decisions to optimize for ROI on conversion rates. It typically involves analyzing various metrics around customer engagement with various marketing activities including but not limited to, ATL (above the line) marketing activities, BTL (below the line) campaigns, targeting personalized offers. Typically the variables of interest are customer profile, campaign conversion rates, and costs associated with various marketing channels. These can generate valuable insights that can help an organization form better marketing strategies, optimize/innovate on delivery, 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 [ ]:

*# 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 [ ]:

*# from google.colab import files*

*# uploaded = files.upload()*

**Load the dataset**

In [ ]:

*# loading the datset*

df = pd.read\_csv('Marketing data.csv')

df.head()

Out[ ]:

|  | **ID** | **Year\_Birth** | **Education** | **Marital\_Status** | **Income** | **Kidhome** | **Teenhome** | **Recency** | **MntWines** | **MntFruits** | **...** | **NumStorePurchases** | **NumWebVisitsMonth** | **AcceptedCmp1** | **AcceptedCmp2** | **AcceptedCmp3** | **AcceptedCmp4** | **AcceptedCmp5** | **AcceptedCmp6** | **Complain** | **Country** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 1826 | 1970 | Graduation | Divorced | 84835.0 | 0 | 0 | 0 | 189 | 104 | ... | 6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | SP |
| **1** | 1 | 1961 | Graduation | Single | 57091.0 | 0 | 0 | 0 | 464 | 5 | ... | 7 | 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | CA |
| **2** | 10476 | 1958 | Graduation | Married | 67267.0 | 0 | 1 | 0 | 134 | 11 | ... | 5 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | US |
| **3** | 1386 | 1967 | Graduation | Together | 32474.0 | 1 | 1 | 0 | 10 | 0 | ... | 2 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | AUS |
| **4** | 5371 | 1989 | Graduation | Single | 21474.0 | 1 | 0 | 0 | 6 | 16 | ... | 2 | 7 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | SP |

5 rows × 27 columns

**Check info of the dataset**

In [ ]:

*#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

**Let's check the percentage of missing values for the Income column.**

In [ ]:

*# % Null values in the Income column*

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

Out[ ]:

1.0714285714285714

**Observations:**

* Income has ~1.07% missing values.

**Let's create a list for numerical columns in the dataset and check the summary statistics**

**Question 1: Find the summary statistics for numerical columns and write your observations. (use describe function). - 4 Marks**

In [ ]:

*# 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 [ ]:

*# printing descriptive statistics of numerical columns*

*#Uncomment the following code and fill in the blanks*

*#df[\_\_\_].\_\_\_\_\_.T*

**Observations:*\_\_***

**Let's create a list for categorical columns in the dataset and check the count of each category**

In [ ]:

*#cat\_cols contain categorical variables*

cat\_cols=['Education', 'Marital\_Status', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',

'AcceptedCmp2', 'AcceptedCmp6', 'Complain', 'Country']

In [ ]:

*# 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

Absurd 0.000893

YOLO 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**

In this section, we will first prepare our dataset for 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", "Absurd" and "YOLO" categories with "Others" in Marital\_Status**

In [ ]:

*# Replacing 2n Cycle with Master*

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

In [ ]:

*# Replacing YOLO, Alone, Absurd with Single*

df["Marital\_Status"].replace(["Alone",], "Single", inplace=**True**)

In [ ]:

df['Marital\_Status'].replace(["Absurd", "YOLO"], "Others", inplace=**True**)

We have fixed the categories in the Marital\_Status. Now, let's see the distribution count in different categories for marital status.

In [ ]:

df.Marital\_Status.value\_counts()

Out[ ]:

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 [ ]:

*# 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)

**Let's check outliers for new variables - Total\_Spending, Total\_Purchase. Also, let's analyze the Year\_Birth column as we observed above that it had a minimum value of 1893.**

In [ ]:

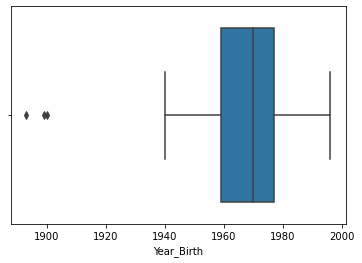
*# Plotting boxplot for Year\_Birth, Total\_Spending, Total\_Purchase*

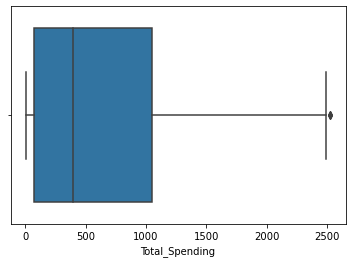
cols=['Year\_Birth','Total\_Spending','Total\_Purchase']

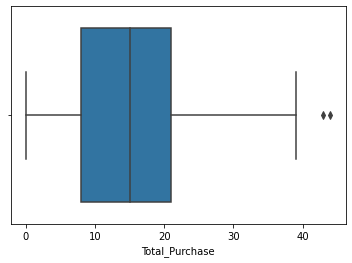
**for** i **in** cols:

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.

Let's check the number of observations for which year birth is less than 1900.

In [ ]:

df[df['Year\_Birth'] < 1900]

Out[ ]:

|  | **ID** | **Year\_Birth** | **Education** | **Marital\_Status** | **Income** | **Kidhome** | **Teenhome** | **Recency** | **MntWines** | **MntFruits** | **...** | **AcceptedCmp3** | **AcceptedCmp4** | **AcceptedCmp5** | **AcceptedCmp6** | **Complain** | **Country** | **Total\_Spending** | **Total\_Purchase** | **NumberofChildren** | **TotalCampaignsAcc** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **513** | 11004 | 1893 | Master | Single | 60182.0 | 0 | 1 | 23 | 8 | 0 | ... | 0 | 0 | 0 | 0 | 0 | SA | 22 | 4 | 1 | 0 |
| **827** | 1150 | 1899 | PhD | Together | 83532.0 | 0 | 0 | 36 | 755 | 144 | ... | 1 | 0 | 0 | 0 | 0 | SP | 1853 | 15 | 0 | 1 |

2 rows × 31 columns

**Observation**:

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

In [ ]:

*#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 [ ]:

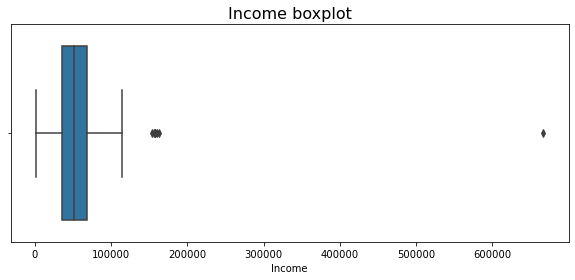
*#plotting Boxplot for income*

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.
* Let's find the value at upper whisker to check how many observations are marked as outliers.

In [ ]:

*#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 [ ]:

*#Checking the observations marked as outliers*

df[df.Income>upper\_whisker]

Out[ ]:

|  | **ID** | **Year\_Birth** | **Education** | **Marital\_Status** | **Income** | **Kidhome** | **Teenhome** | **Recency** | **MntWines** | **MntFruits** | **...** | **AcceptedCmp3** | **AcceptedCmp4** | **AcceptedCmp5** | **AcceptedCmp6** | **Complain** | **Country** | **Total\_Spending** | **Total\_Purchase** | **NumberofChildren** | **TotalCampaignsAcc** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **325** | 4931 | 1977 | Graduation | Together | 157146.0 | 0 | 0 | 13 | 1 | 0 | ... | 0 | 0 | 0 | 0 | 0 | SA | 1730 | 28 | 0 | 0 |
| **497** | 1501 | 1982 | PhD | Married | 160803.0 | 0 | 0 | 21 | 55 | 16 | ... | 0 | 0 | 0 | 0 | 0 | US | 1717 | 44 | 0 | 0 |
| **527** | 9432 | 1977 | Graduation | Together | 666666.0 | 1 | 0 | 23 | 9 | 14 | ... | 0 | 0 | 0 | 0 | 0 | SA | 62 | 11 | 1 | 0 |
| **731** | 1503 | 1976 | PhD | Together | 162397.0 | 1 | 1 | 31 | 85 | 1 | ... | 0 | 0 | 0 | 0 | 0 | SP | 107 | 1 | 2 | 0 |
| **853** | 5336 | 1971 | Master | Together | 157733.0 | 1 | 0 | 37 | 39 | 1 | ... | 0 | 0 | 0 | 0 | 0 | SP | 59 | 2 | 1 | 0 |
| **1826** | 5555 | 1975 | Graduation | Divorced | 153924.0 | 0 | 0 | 81 | 1 | 1 | ... | 0 | 0 | 0 | 0 | 0 | SP | 6 | 0 | 0 | 0 |
| **1925** | 11181 | 1949 | PhD | Married | 156924.0 | 0 | 0 | 85 | 2 | 1 | ... | 0 | 0 | 0 | 0 | 0 | CA | 8 | 0 | 0 | 0 |
| **2204** | 8475 | 1973 | PhD | Married | 157243.0 | 0 | 1 | 98 | 20 | 2 | ... | 0 | 0 | 0 | 0 | 0 | IND | 1608 | 37 | 1 | 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.

**Let's compare the summary statistics for these observations with observations on the other side of the upper whisker.**

In [ ]:

*#Checking the summary statistics for observations marked as outliers*

df[df.Income>upper\_whisker].describe().T

Out[ ]:

|  | **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 [ ]:

*#Checking the summary statistics for observations not marked as outliers*

df[df.Income<upper\_whisker].describe().T

Out[ ]:

|  | **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 [ ]:

*#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**

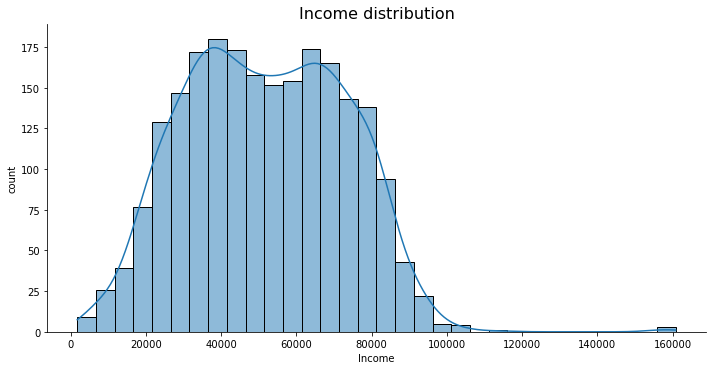
In [ ]:

*#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.
* We will replace the missing values for the income variable with the median, and not mean, as the variable is slightly skewed to the right

In [ ]:

*#filling null values with median*

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

**Analyzing all the campaigns**

**Question 2: Write your observations on acceptance rate for each campaign given in the below plot. - 4 Marks**

**Let's find out what is the acceptance rate for each campaign?**

In [ ]:

*# PLotting the % acceptance for every campaign*

Camp\_cols=['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp6']

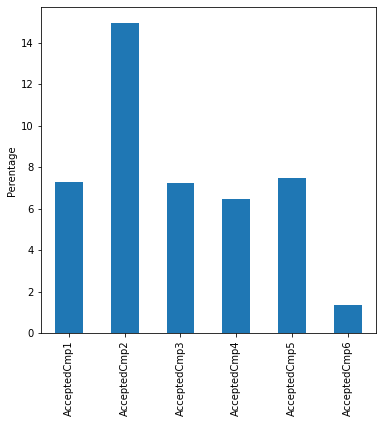
success\_campaign=(df[Camp\_cols].sum()/df.shape[0])\*100

*# plot*

success\_campaign.plot(kind='bar', figsize=(6,6))

plt.ylabel("Perentage")

plt.show()



**Observations:\_\_\_**

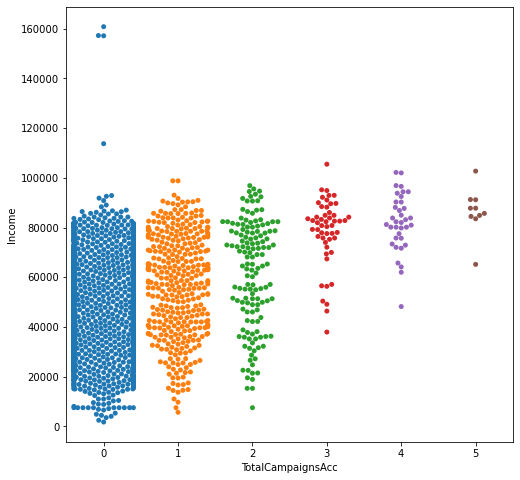
**Let's analyze what kind of customer are accepting campaigns?**

In [ ]:

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 [ ]:

*# Let's see the mean income of customers*

df.Income.mean()

Out[ ]:

51762.59811827957

**Question 3: Write your observations on acceptance rate for each campaign according to the income level. - 7 Marks**

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 [ ]:

*# 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*

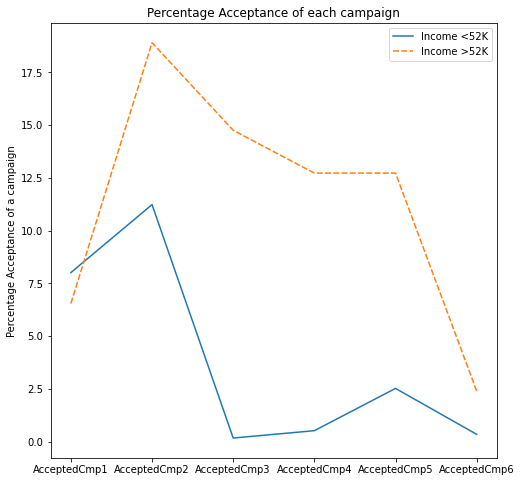
plt.figure(figsize=(8,8))

sns.lineplot(data=new\_df)

plt.title("Percentage Acceptance of each campaign")

plt.ylabel("Percentage Acceptance of a campaign")

plt.show()



**Observations:\_\_**

Let's find out who has accepted the last campaign and what could be the reason?

In [ ]:

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

Out[ ]:

(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 [ ]:

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

grouped2

Out[ ]:

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. Let's 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**

Let's 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 [ ]:

**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)

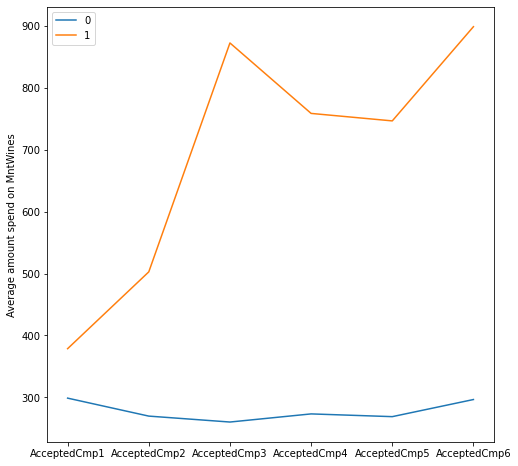
plt.show()

**Use the function defined above to generate line plots for different purchasing Products**

In [ ]:

*#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.

**Question 4: Write the code and your observations on average amount spent on different products across all campaigns. - 7 Marks**

In [ ]:

*#meat products*

*#call the function amount\_per\_campaign for MntMeatProducts*

In [ ]:

*# Fruit products*

*#call the function amount\_per\_campaign for MntFruits*

In [ ]:

*# gold products*

*#call the function amount\_per\_campaign for MntGoldProds*

In [ ]:

*#sweet products*

*#call the function amount\_per\_campaign for MntSweetProducts*

**Observations:\_\_**

**We have analyzed the relationship between campaigns and different products. Now, let's see 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 [ ]:

**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))

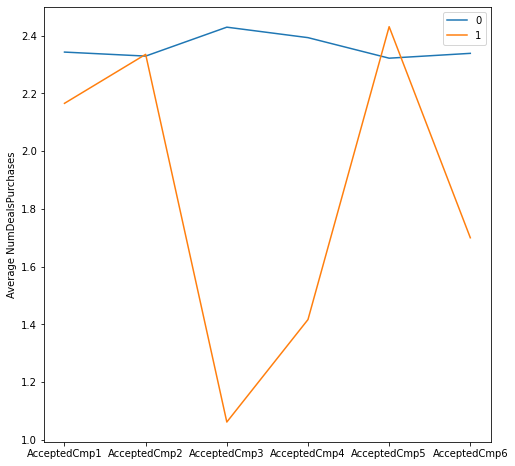
plt.ylabel('Average' + ' ' + columns\_name)

plt.show()

In [ ]:

*#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.

**Question 5: Write the code and your observations on average number of purchases from different channels across all campaigns. - 7 Marks**

In [ ]:

*# store purchase*

*#call the function Purchases\_per\_campaign for NumStorePurchases*

In [ ]:

*#Catalog purchase*

*#call the function Purchases\_per\_campaign for NumCatalogPurchases*

In [ ]:

*#Web purchases*

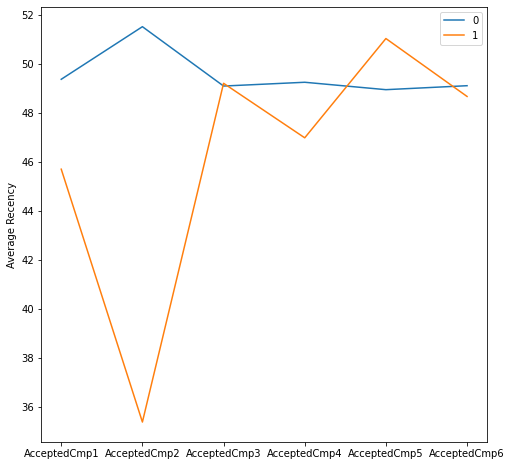
*#call the function Purchases\_per\_campaign for NumWebPurchases*

**Observations:\_\_\_**

In [ ]:

*#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.

**We have analyzed the relationship between campaigns and numerical variables. Let's see the relationship of campaigns with different categorical variables**

We will 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 [ ]:

**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)

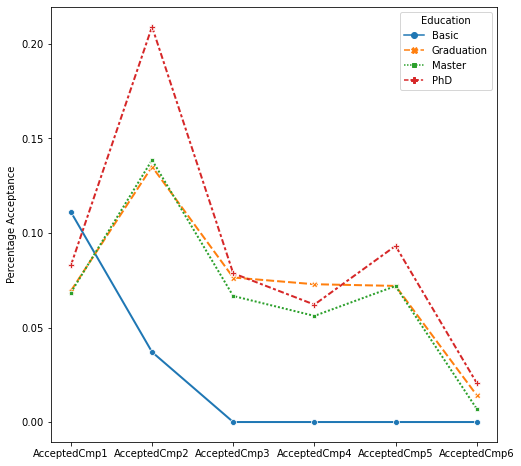
plt.ylabel('Percentage Acceptance')

plt.show()

In [ ]:

*#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.

**Question 6: Write the code and your observations on percentage acceptance for different categorical variables across all campaigns. - 7 Marks**

In [ ]:

*#NumberofChildren*

*#call the function Cat\_Campaign\_Relation for NumberofChildren*

In [ ]:

*#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*

In [ ]:

*#Let's filter the observations for 'ME' country as they are only 3 such observations*

df\_not\_mexico=df[df.Country!='ME']

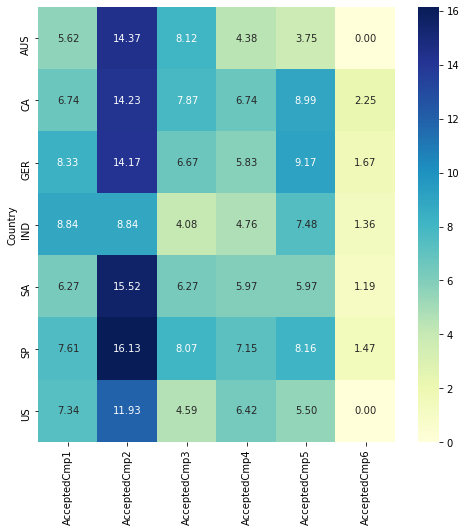
*#Plot*

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="YlGnBu")

Out[ ]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x1a4a88f5370>



**Observation:\_\_\_**

**Check the product preferences by customers**

In [ ]:

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

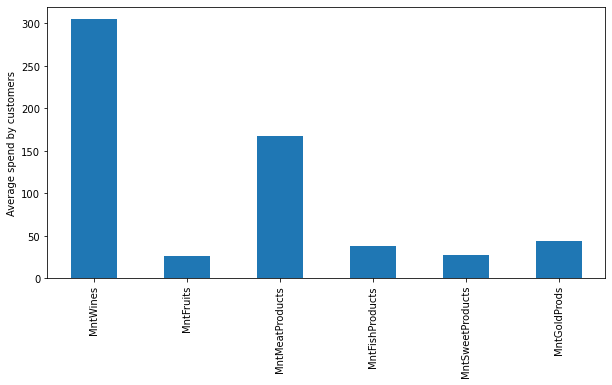
mnt\_cols = [col **for** col **in** df.columns **if** 'Mnt' **in** col]

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.

Let's check if the product preferences are similar for different types of customers. We will 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 [ ]:

**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))

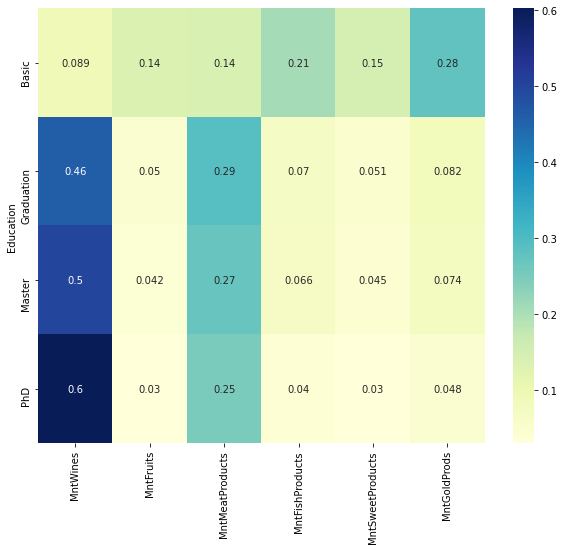
sns.heatmap(df\_new1.T, annot=**True**, cmap="YlGnBu")

plt.show()

In [ ]:

*# 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.

**Question 7: Write the code and your observations on percentage amount spent on different products for each category of the mentioned categorical variables. - 7 Marks**

In [ ]:

*#call the function amount\_per\_category for Marital\_Status with dataframe df\_rest*

In [ ]:

*#call the function amount\_per\_category for Country with dataframe df\_not\_mexico*

**Observations:\_\_\_**

**Check different channel performances**

Let's calculate the percentage of purchases for all the channels.

In [ ]:

*# 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.*

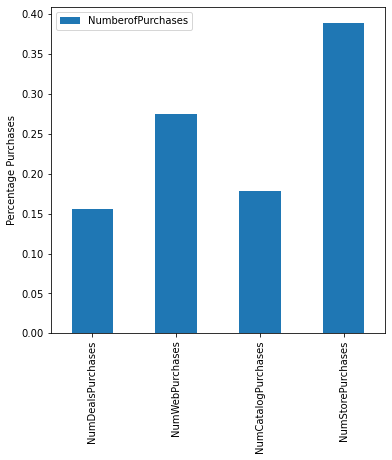
channels = pd.DataFrame(df[channel\_cols].sum()/df.Total\_Purchase.sum(), columns=['NumberofPurchases'])

*# 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.

**Question 8: Write your observations on percentage purchases from different channels for different categories of the income\_cat column. - 4 Marks**

Let's check how number of purchases via different channels varies for different income bins.

In [ ]:

*#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'])

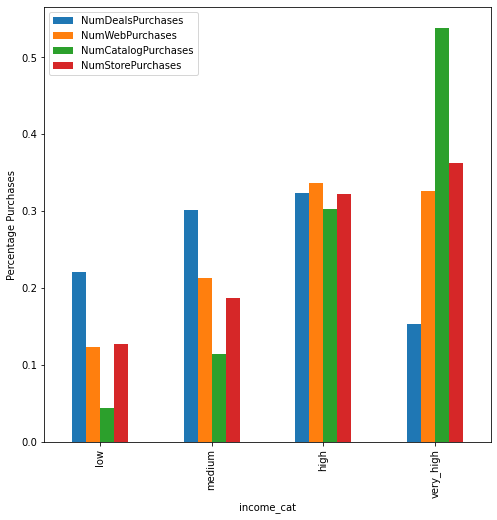
In [ ]:

group=df.groupby('income\_cat').sum()[channel\_cols]

(group/group.sum()).plot(kind='bar', figsize=(8,8))

plt.ylabel("Percentage Purchases")

plt.show()



**Observations:\_\_**

**We can also visualize the correlation by purchases from different channels and income of the customer.**

**Question 9: Find the correlation matrix for the columns mentioned below and visualize the same using heatmap. - 3 Marks**

In [ ]:

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

*#Write your code here*

**Observations:*\_\_***

As we know from our analysis we have done so far that customers with income, number of children, and amount spending on wines are the important factors. Let's 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 [ ]:

df3=df[df.Income>52000]

df4=df3[df3.MntWines>df3.MntWines.mean()]

new\_profile=df4[df4.NumberofChildren==0]

In [ ]:

*#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 '])

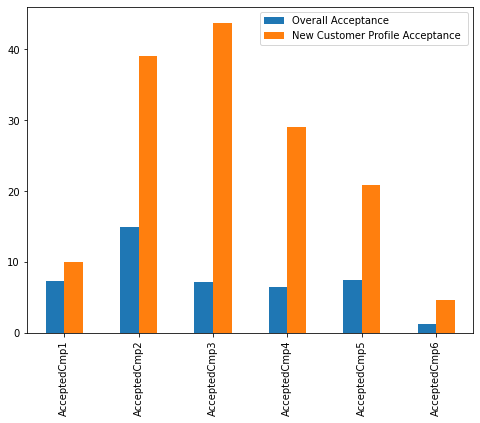
*# plot*

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.

**Question 10: Based on your analysis, write the conclusions and recommendations for the CMO to help make the next marketing campaign strategy. - 10 Marks**

**Conclusion and Recommendations**[**¶**](https://glnbviewer.oe5d8cpd7j99e.ap-south-1.cs.amazonlightsail.com/urls/lms-uploads.s3.ap-southeast-1.amazonaws.com/account_1/attachments/4597776/Learner_Notebook_Project_Marketing_Campaign_Analysis.ipynb/%3Fresponse-content-disposition%3Dattachment%253B%2520filename%253D%2522Learner_Notebook_Project_Marketing_Campaign_Analysis.ipynb%2522%253B%2520filename%252A%253DUTF-8%2527%2527Learner%25255FNotebook%25255FProject%25255FMarketing%25255FCampaign%25255FAnalysis.ipynb%26X-Amz-Algorithm%3DAWS4-HMAC-SHA256%26X-Amz-Credential%3DAKIA434J2A76JCS5LYOQ%252F20220407%252Fap-southeast-1%252Fs3%252Faws4_request%26X-Amz-Date%3D20220407T140142Z%26X-Amz-Expires%3D86400%26X-Amz-SignedHeaders%3Dhost%26X-Amz-Signature%3D650e9438b181e419b53d47232dfc2f12bd278030a3134b3346a42c13905eb798#Conclusion-and-Recommendations)

**Write your conclusions here**

**Write your recommendations here**