# Project Title - Customer Personality Analysis

The Customer Personality data Analysis is one of the best analysis to collect a information from the customer and in which data maximum information is given. We analysis some informations to get important data like customer in which product to money investing. and we learn how to analysis data with the help of pandas numeric calculation numpy visualization etc.

## Description about a data:

Here's a brief version of the data description file.

### People

ID: Customer's unique identifier

Year_Birth: Customer's birth year

Education: Customer's education level

Marital_Status: Customer's marital status

Income: Customer's yearly household income

Kidhome: Number of children in customer's household

Teenhome: Number of teenagers in customer's household

Dt_Customer: Date of customer's enrollment with the company

Recency: Number of days since customer's last purchase

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

### Products

MntWines: Amount spent on wine in last 2 years

MntFruits: Amount spent on fruits in last 2 years

MntMeatProducts: Amount spent on meat in last 2 years

MntFishProducts: Amount spent on fish in last 2 years

MntSweetProducts: Amount spent on sweets in last 2 years

MntGoldProds: Amount spent on gold in last 2 years

### Promotion

NumDealsPurchases: Number of purchases made with a discount

AcceptedCmp1: 1 if customer accepted the offer in the 1st campaign, 0 otherwise

AcceptedCmp2: 1 if customer accepted the offer in the 2nd campaign, 0 otherwise

AcceptedCmp3: 1 if customer accepted the offer in the 3rd campaign, 0 otherwise

AcceptedCmp4: 1 if customer accepted the offer in the 4th campaign, 0 otherwise

AcceptedCmp5: 1 if customer accepted the offer in the 5th campaign, 0 otherwise

Response: 1 if customer accepted the offer in the last campaign, 0 otherwise

### Place

NumWebPurchases: Number of purchases made through the company’s web site

NumCatalogPurchases: Number of purchases made using a catalogue

NumStorePurchases: Number of purchases made directly in stores

NumWebVisitsMonth: Number of visits to company’s web site in the last month

Code the following,

import pandas as pd. pandas is used for buidling dataframes.

import numpy as np. numpy is imported with pandas

df = pd.read_csv('marketing_campaign (1)new (version 1).xlsb.csv')

In [23]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # visualizing data
%matplotlib inline
import seaborn as sns
```

Let's begin by downloading the data, and listing the files within the dataset.

In [4]:
```
df = pd.read_csv(r'marketing_campaign (1)new (version 1).xlsb.csv', encoding= 'unicode_escape')
```

Understand the data - Inspect the data View and inspect summary information about the dataframe by coding the following:

In [5]:
```
df.columns
```

Out[5]:
```
Index(['ID', 'Year', 'Birth', 'Education', 'Marital', 'Status', 'Income',
       'Kidhome', 'Teenhome', 'Dt', 'Customer', 'Recency', 'MntWines',
       'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z', 'CostContact', 'Z.1', 'Revenue',
       'Response'],
      dtype='object')
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 34 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   ID                   2240 non-null   int64
 1   Year                 2240 non-null   int64
 2   Birth                2240 non-null   object
 3   Education            2240 non-null   object
 4   Marital              2216 non-null   float64
 5   Status               2240 non-null   int64
 6   Income               2240 non-null   int64
 7   Kidhome              2240 non-null   object
 8   Teenhome             2240 non-null   int64
 9   Dt                   2240 non-null   int64
 10  Customer             2240 non-null   int64
 11  Recency              2240 non-null   int64
 12  MntWines             2240 non-null   int64
 13  MntFruits            2240 non-null   int64
 14  MntMeatProducts      2240 non-null   int64
 15  MntFishProducts      2240 non-null   int64
 16  MntSweetProducts     2240 non-null   int64
 17  MntGoldProds         2240 non-null   int64
 18  NumDealsPurchases    2240 non-null   int64
 19  NumWebPurchases      2240 non-null   int64
 20  NumCatalogPurchases  2240 non-null   int64
 21  NumStorePurchases    2240 non-null   int64
 22  NumWebVisitsMonth    2240 non-null   int64
 23  AcceptedCmp3         2240 non-null   int64
 24  AcceptedCmp4         2240 non-null   int64
 25  AcceptedCmp5         2240 non-null   int64
 26  AcceptedCmp1         2240 non-null   int64
 27  AcceptedCmp2         2240 non-null   int64
 28  Complain             2240 non-null   int64
 29  Z                    0 non-null      float64
 30  CostContact          0 non-null      float64
 31  Z.1                  0 non-null      float64
 32  Revenue              0 non-null      float64
 33  Response             0 non-null      float64
dtypes: float64(6), int64(25), object(3)
memory usage: 595.1+ KB
```

In [7]:
```
#Check for missing data
pd.isnull(df).sum()
```

Out[7]:
```
ID                      0
Year                    0
Birth                   0
Education               0
Marital                24
Status                  0
Income                  0
Kidhome                 0
Teenhome                0
Dt                      0
Customer                0
Recency                 0
MntWines                0
MntFruits               0
MntMeatProducts         0
MntFishProducts         0
MntSweetProducts        0
MntGoldProds            0
NumDealsPurchases       0
NumWebPurchases         0
NumCatalogPurchases     0
NumStorePurchases       0
NumWebVisitsMonth       0
AcceptedCmp3            0
AcceptedCmp4            0
AcceptedCmp5            0
AcceptedCmp1            0
AcceptedCmp2            0
Complain                0
Z                    2240
CostContact          2240
Z.1                  2240
Revenue              2240
Response             2240
dtype: int64
```

In [8]:
```
#drop all null values
df.dropna(inplace=True)
```

In [8]:
```
df.columns
```

Out[8]:
```
Index(['ID', 'Year', 'Birth', 'Education', 'Marital', 'Status', 'Income',
       'Kidhome', 'Teenhome', 'Dt', 'Customer', 'Recency', 'MntWines',
       'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z', 'CostContact', 'Z.1', 'Revenue',
       'Response'],
      dtype='object')
```

In [9]:
```
df.tail(5)
```

Out[9]:

| | ID | Year | Birth | Education | Marital | Status | Income | Kidhome | Teenhome | Dt | ... | AcceptedCmp4 | AcceptedCmp5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **2235** | 10870 | 1967 | Graduation | Married | 61223.0 | 0 | 1 | 13-06-2013 | 46 | 709 | ... | 0 | 0 |
| **2236** | 4001 | 1946 | PhD | Together | 64014.0 | 2 | 1 | 10-06-2014 | 56 | 406 | ... | 0 | 0 |
| **2237** | 7270 | 1981 | Graduation | Divorced | 56981.0 | 0 | 0 | 25-01-2014 | 91 | 908 | ... | 0 | 0 |
| **2238** | 8235 | 1956 | Master | Together | 69245.0 | 0 | 1 | 24-01-2014 | 8 | 428 | ... | 0 | 0 |
| **2239** | 9405 | 1954 | PhD | Married | 52869.0 | 1 | 1 | 15-10-2012 | 40 | 84 | ... | 0 | 0 |

5 rows × 34 columns

In [11]:
```
#Some Summary statistics
df.describe()
```

Out[11]:

| | ID\tYear_Birth\tEducation\tMarital_Status\tIncome\tKidhome\tTeenhome\tDt_Customer\tRecency\tMntWines\tMntFruits\tMntMeatPro |
|---|---|
| count | |
| unique | |
| top | |
| freq | |

In [10]:
```
df.shape
```

Out[10]:
```
(2240, 34)
```

In [11]:
```
df.duplicated().sum()
```

Out[11]:
```
0
```

In [12]:
```
df.nunique()
```

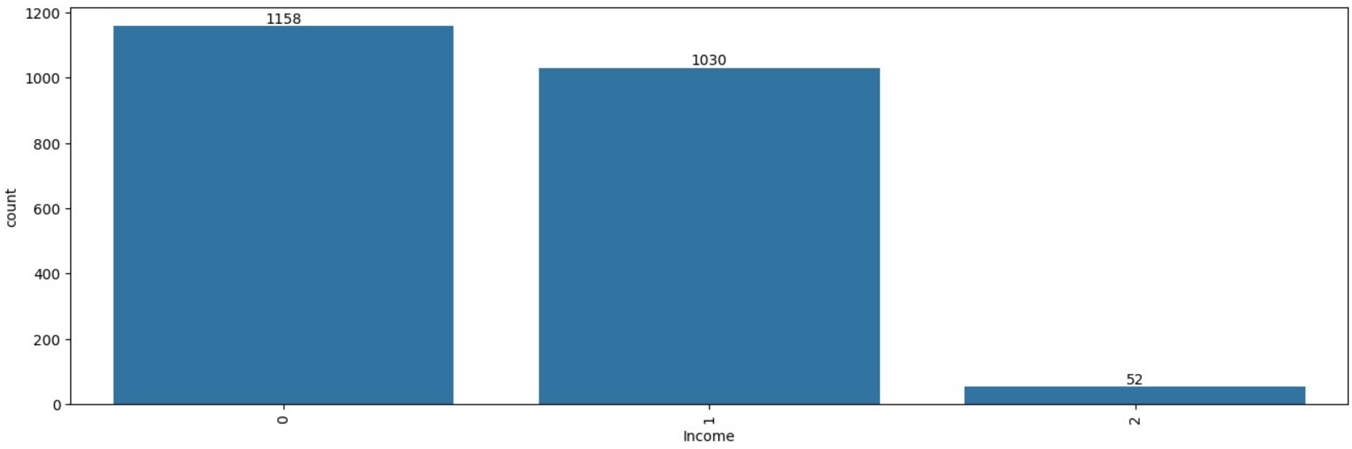
Out[12]:
```
ID                     2240
Year                     59
Birth                     5
Education                 8
Marital                1974
Status                    3
Income                    3
Kidhome                 663
Teenhome                100
Dt                      776
Customer                158
Recency                 558
MntWines                182
MntFruits               177
MntMeatProducts         213
MntFishProducts          15
MntSweetProducts         15
MntGoldProds             14
NumDealsPurchases        14
NumWebPurchases          16
NumCatalogPurchases       2
NumStorePurchases         2
NumWebVisitsMonth         2
AcceptedCmp3              2
AcceptedCmp4              2
AcceptedCmp5              2
AcceptedCmp1              1
AcceptedCmp2              1
Complain                  2
Z                         0
CostContact               0
Z.1                       0
Revenue                   0
Response                  0
dtype: int64
```

## Exploratory Analysis and Visualization

**Custumer Personality Analysis** - In this we visualization some columns 'Year of birth' , 'Income' , 'Kidhome', 'Teenhome' etc.

> - Compute the mean, sum, range and other interesting statistics for numeric columns
> - Explore distributions of numeric columns using histograms etc.
> - Explore relationship between columns using scatter plots, bar charts etc.
> - Make a note of interesting insights from the exploratory analysis

In [61]:
```
plt.figure(figsize=(16,5))
ax=sns.countplot(x=df['Income'])
for bars in ax.containers:
    ax.bar_label(bars)
    plt.xticks(rotation=90)
    plt.show()
```




The count plot visualizes the distribution of income levels within the dataset. Here's a brief analysis:

Income Distribution: The plot shows the frequency of different income levels, with each bar representing a specific income range or value.

Bar Labels: Each bar is labeled with its count, providing a clear numerical indication of how many data points fall into each income category.

X-Axis Labels: The x-axis labels are rotated 90 degrees to ensure readability, especially important if the income categories are numerous or have lengthy labels.

Overall, the plot provides a clear and detailed view of how income levels are distributed in the dataset, with labeled bars making it easy to identify the most and least common income ranges.

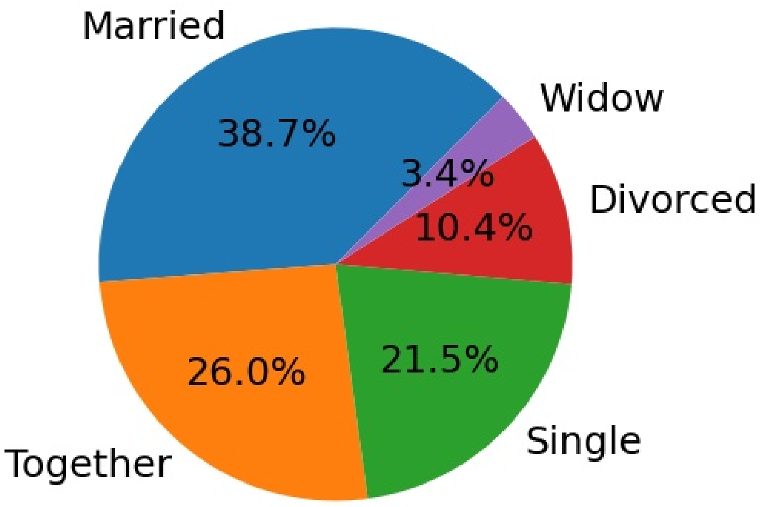
In [52]:
```
pd.crosstab(df['Teenhome'],df['Income'])
```

Out[52]:

| Income | 0 | 1 | 2 |
|---|---|---|---|
| **Teenhome** | | | |
| **0** | 11 | 17 | 0 |
| **1** | 14 | 10 | 0 |
| **2** | 14 | 13 | 1 |
| **3** | 10 | 18 | 1 |
| **4** | 11 | 16 | 0 |
| **...** | ... | ... | ... |
| **95** | 10 | 9 | 0 |
| **96** | 13 | 12 | 0 |
| **97** | 8 | 10 | 2 |
| **98** | 11 | 10 | 1 |
| **99** | 7 | 9 | 1 |

100 rows × 3 columns

In [18]:
```
plt.figure(figsize=(5,5))
d=(df['Education'].value_counts(normalize=True)*100).head()
keys=df['Education'].value_counts().head().index
colourz=['#B5DF00','#AD1FFF','#FFC93F','#5FB1FF','BF1B00']
exploda=(0.02,0.02,0.02,0.4,0.02)
plt.pie(d,labels=keys,autopct='%1.1f%%',startangle=45,textprops={'fontsize':18})
plt.savefig("audiencepie.png")
```




The pie chart visualizes the distribution of the top five education levels within the dataset. Here's a brief analysis:

1. **Proportional Representation**: The pie chart shows the percentage share of each education level among the top five categories, allowing for easy comparison.
2. **Visual Enhancements**: The use of distinct colors and the explosion effect on one slice (with a larger explode value for emphasis) highlights key segments and enhances visual appeal.
3. **Readable Labels**: The chart includes labels with percentage values and text properties set for better readability, providing clear and immediate insights into the distribution.

Overall, the pie chart effectively conveys the relative proportions of different education levels in the dataset, making it easy to understand the dominant categories at a glance.

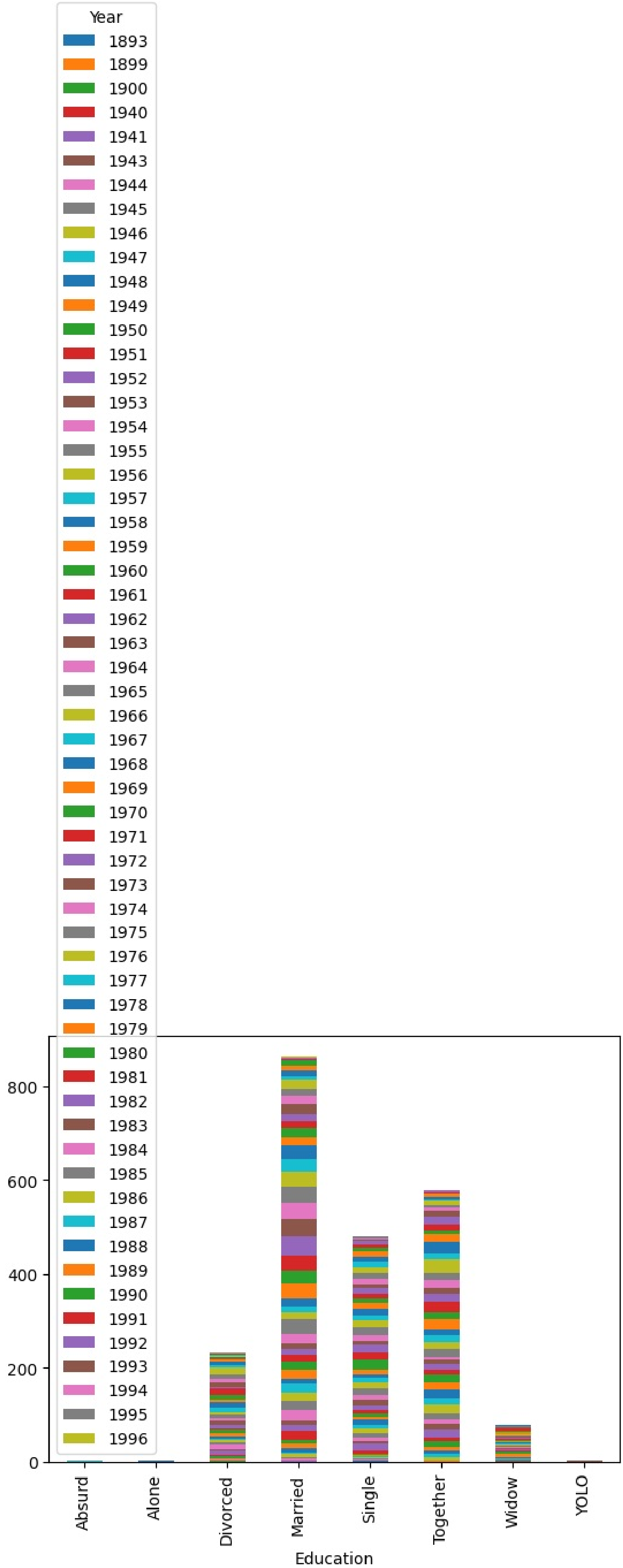
In [29]:
```
pd.crosstab(df['Education'],df['Year'])
```

Out[29]:

| Year | 1893 | 1899 | 1900 | 1940 | 1941 | 1943 | 1944 | 1945 | 1946 | 1947 | ... | 1987 | 1988 | 1989 | 1990 | 1991 | 1992 | 1993 | 1994 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Education** | | | | | | | | | | | | | | | | | | | | |
| **Absurd** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| **Alone** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **Divorced** | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | ... | 3 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | |
| **Married** | 0 | 0 | 0 | 0 | 1 | 2 | 4 | 3 | 7 | 3 | ... | 7 | 12 | 11 | 11 | 2 | 4 | 0 | 0 | |
| **Single** | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 3 | 2 | ... | 13 | 10 | 11 | 7 | 9 | 6 | 4 | 1 | |
| **Together** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 6 | 9 | ... | 4 | 5 | 6 | 0 | 4 | 3 | 0 | 2 | |
| **Widow** | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 1 | 0 | 2 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| **YOLO** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

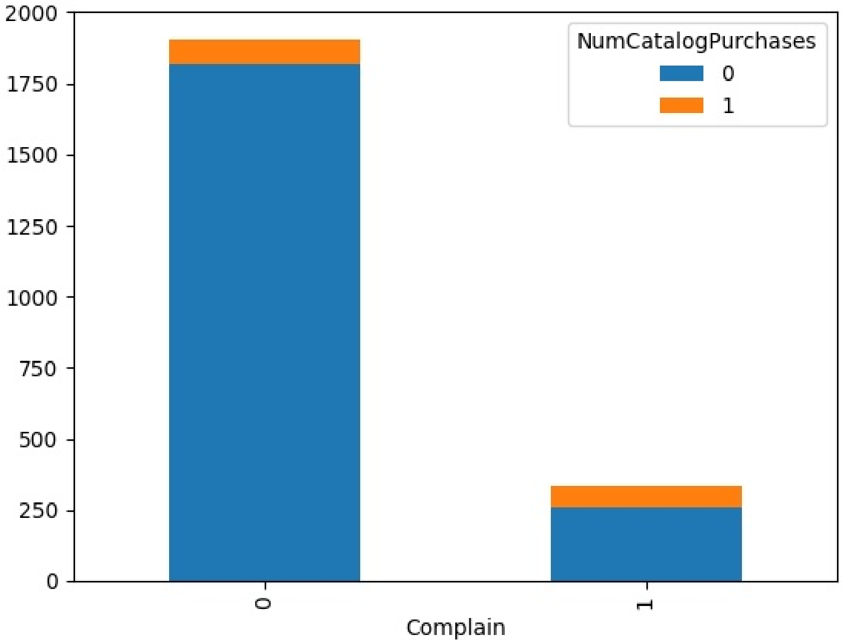
8 rows × 59 columns

In [30]:
```
pd.crosstab(df['Education'],df['Year']).plot(kind='bar',stacked='true')
plt.show()
```



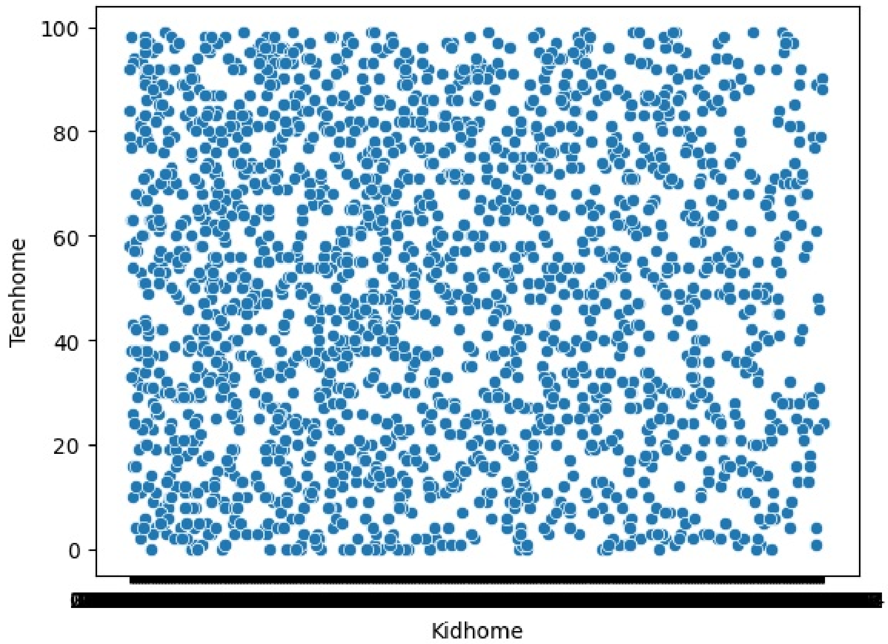

The stacked bar plot created from the crosstab function displays the distribution of different education levels across various years. Each bar represents a specific year, with segments within the bar showing the count of each education level. This visualization helps to understand how the composition of education levels changes over time.

In [31]:
```
pd.crosstab(df['Complain'],df['NumCatalogPurchases']).plot(kind='bar',stacked='true')
plt.show()
```




In [51]:
```
sns.scatterplot(x=df['Kidhome'],y=df['Teenhome'])
```

Out[51]: <Axes: xlabel='Kidhome', ylabel='Teenhome'>




The scatter plot visualizes the relationship between the number of children at home ("Kidhome") and the number of teenagers at home ("Teenhome"). Here's a brief analysis:

Data Distribution: Each point represents a data entry, with its position determined by the number of children and teenagers in a household. The distribution of points reveals how these two variables correlate.

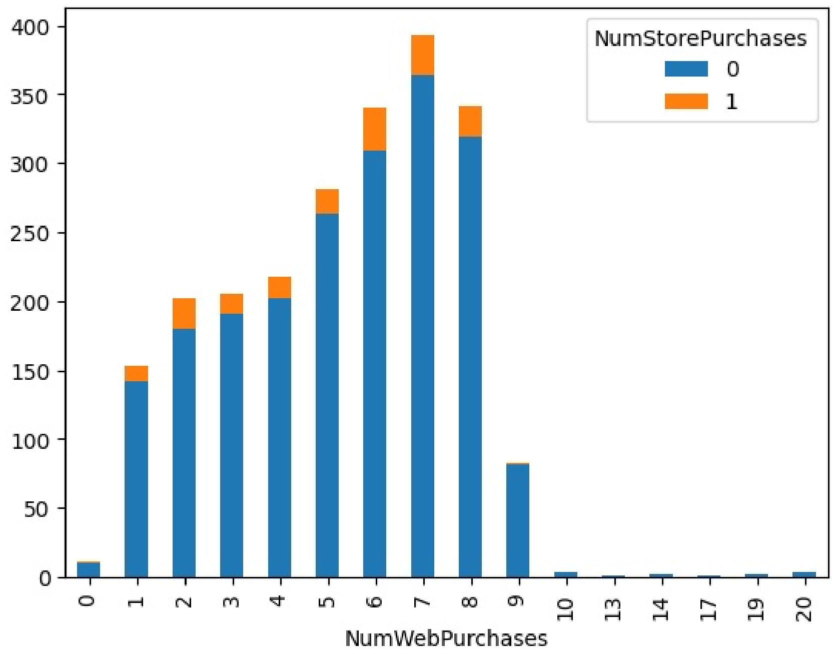
Correlation Insight: The plot helps identify if there's any correlation between having children and teenagers at home. For instance, clusters of points or a linear trend might indicate a relationship.

Pattern Identification: Any noticeable patterns, such as clustering of points along certain values, can provide insights into common household compositions within the dataset.

Overall, the scatter plot effectively displays the relationship between the number of children and teenagers in a household, helping to identify trends or correlations between these variables.

In [38]:
```
plt.figure(figsize=(14,5))
pd.crosstab(df['NumWebPurchases'],df['NumStorePurchases']).plot(kind='bar',stacked='true')
plt.show()
```

<Figure size 1400x500 with 0 Axes>




Comparison of Purchases: The plot compares the frequency of different combinations of web and store purchases. Each bar represents a unique count of web purchases, while the segments within each bar correspond to the number of store purchases.

Stacked Bar Plot: The bars are stacked, showing the cumulative distribution of store purchases for each level of web purchases. This helps in understanding how the web and store purchases are distributed together.

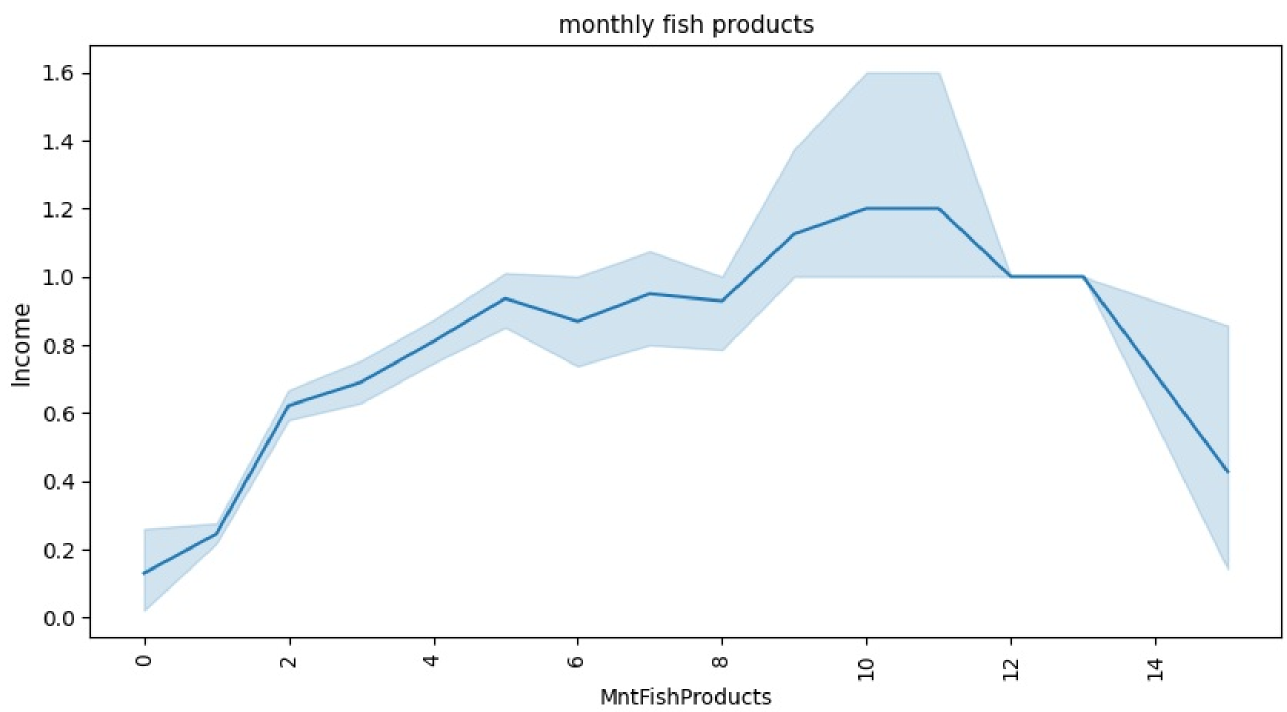
Insights: By examining the heights of the bars and their segments, one can infer patterns such as whether customers who make a high number of web purchases also tend to make more store purchases, or if there's a tendency to prefer one channel over the other.

Visualization Clarity: The stacked nature of the plot aids in visualizing the combined effect, though it can sometimes be challenging to interpret the exact values of each segment without labels.

Overall, the plot is useful for understanding the relationship between web and store purchases, showing how frequently different combinations occur within the dataset.

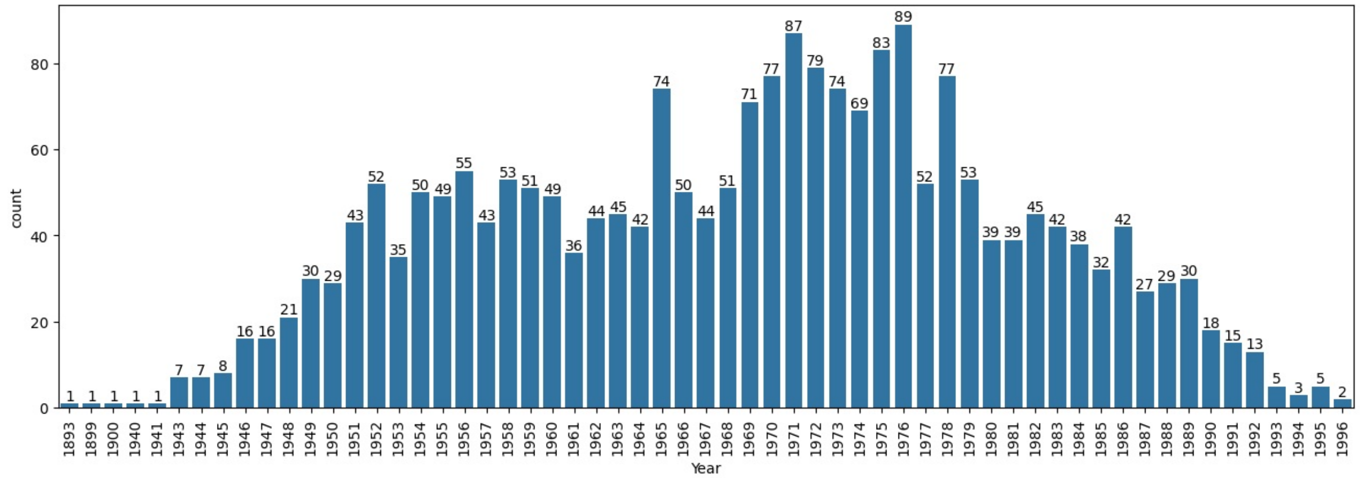
In [41]:
```
plt.figure(figsize=(10,5))
sns.lineplot(x="MntFishProducts",y="Income",data=df)
plt.title("monthly fish products",fontsize=11)

plt.ylabel("Income",fontsize=11)
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.show()
```




The line plot depicts the relationship between the amount spent on fish products per month ("MntFishProducts") and income. The overall trend suggests a positive correlation, indicating that higher spending on fish products is associated with higher income levels. This relationship implies that as people earn more, they tend to spend more on fish products, which may reflect higher disposable income or a preference for a healthier diet among higher-income groups. The plot is clearly labeled, and the rotation of x-axis labels improves readability.

In [49]:
```python
plt.figure(figsize=(16,5))
ax=sns.countplot(x=df['Year'])
for bars in ax.containers:
    ax.bar_label(bars)
    plt.xticks(rotation=90)
    plt.show()
```




The count plot visualizes the distribution of data points across different years. Here's an analysis of the plot:

1. **Distribution of Data Points**: The plot shows how many data entries are associated with each year, giving an overview of the dataset's temporal distribution.
2. **Bar Labels**: Each bar is labeled with its count, providing a clear numerical indication of the number of entries per year.
3. **X-Axis Labels**: The x-axis labels are rotated 90 degrees, enhancing readability, especially if the year labels are lengthy or if there are many bars.

Overall, the plot efficiently conveys the frequency of entries per year, and the additional bar labels help quickly identify the count values for each year.

# Inferences and Conclusion

## Customer Personality Analysis

you show that in above we analysis a data in which some interesting data collected.

> - We firstly data to check in which any null values.
> - we prepare data and clearing some missing values of rows or insert in Nan.
> - we added a new column for income .
> - we visualize some graphs show aboves related to number of years birth
> - also other graphs is shows in which interesting data collected.

We getting some important data related to customer. we analysis in some data and collect in some interesting data as per get a result.

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js