COMP5721M: Programming for Data Science

Coursework 3: Data Analysis Project

Personality Analysis-Marketing Strategy

- Merve Turk Oydem, mm22mto@leeds.ac.uk
- Sulayman Birt, ee19sb@leeds.ac.uk
- Thamer Aljohani, ml21ta@leeds.ac.uk
- Wasif Shaukat, mm22ws@leeds.ac.uk

Project Plan

The Data

A thorough examination of a business's ideal customer can be done through customer personality analysis. It makes it simpler for businesses to adapt their products to the unique needs, behaviors, and concerns of various customers, and also helps them to understand their customers in detail. Consumer personality analysis assists a corporation in tailoring the stock levels or product to its target market from various customer categories. For instance, targeted marketing could be done based on different segments of customers and the campaign could then be specifically targeted to those customers. For this project, it was unanimously decided to look for data where consumer behavior can be analysed.

*The chosen Dataset is from the **Kaggle** online Data Science platform and provides a detailed breakdown of the characteristics of an imaginary company's customer base. It is intended to give insight into what makes an ideal customer, and how to gear different products to the needs and lifestyles of particular customers. The population of customers can be segmented into different personality groups, which allows for tailored marketing to each customer group by assessing the group with the highest chance of purchasing a particular product. For each individual customer, there are numerous attributes, including those that describe the customer themself and their lifestyles, such as age, education level and income, the amount they spend on different product groups, such as fruit, meat and fish, their acceptance of promotional offers in different campaigns, and their preferred location or method of making purchases, such as online, in-store, or through a catalogue.*

It is unclear whether this dataset is real or artificial, as there is no direct mention of a source or method of collection. However, we can assume this is an accurate dataset as the quality of data is high, with almost all columns containing only meaningful data and no extreme mean or standard deviation values. There are a few minor issues of null values and categorical errors which we will remove from the data, and only one column, of customer income, has significant outlier values. This dataset appears to accurately serve its purpose as a training dataset for investigating customer personalities and buying habits.

Below is a detailed breakdown of each of the different attributes for each customer.

- *People*
  - *ID: customer's unique identifier - (int/numerical)*
  - *Year_Birth: customer's birth year - (int)*
  - *Education: customer's education level - (object/categorical)*
  - *Marital_Status: customer's marital status - (object/categorical)*
  - *Income: customer's yearly household income - (float)*
  - *Kidhome: Number of children in customer's household - (int)*
  - *Teenhome: Number of teenagers in customer's household - (int)*
  - *Dt_customer: Date of customer's enrollment with the company - (object)*
  - *Recency: Number of days since customer's last purchase - (int)*
  - *Complain: 1 if the customer complained in the last 2 years, 0 otherwise - (int)*
- *Products*
  - *MntWines: Amount spent on wine in last 2 years - (int)*
  - *MntFruits: Amount spent on fruits in last 2 years - (int)*
  - *MntMeatProducts: Amount spent on meat in last 2 years - (int)*
  - *MntFishProducts: Amount spent on fish in last 2 years - (int)*
  - *MntSweetProducts: Amount spent on sweets in last 2 years - (int)*
  - *MntGoldProds: Amount spent on gold in last 2 years - (int)*
- *Promotion*
  - *NumDealsPurchases: Number of purchases made with a discount - (int)*
  - *AcceptedCmp1: 1 if customer accepted the offer in the 1st campaign, 0 otherwise - (int)*
  - *AcceptedCmp2: 1 if customer accepted the offer in the 2nd campaign, 0 otherwise - (int)*
  - *AcceptedCmp3: 1 if customer accepted the offer in the 3rd campaign, 0 otherwise - (int)*
  - *AcceptedCmp4: 1 if customer accepted the offer in the 4th campaign, 0*

otherwise - (int)
- *AcceptedCmp5: 1 if customer accepted the offer in the 5th campaign, 0 otherwise - (int)*
- *Response: 1 if customer accepted the offer in the last campaign, 0 otherwise - (int)*

- *Place*
  - *NumWebPurchases: Number of purchases made through the company's website - (int)*
  - *NumCatalogPurchases: Number of purchases made using a catalogue - (int)*
  - *NumStorePurchases: Number of purchases made directly in stores - (int)*
  - *NumWebVisitsMonth: Number of visits to company's website in the last month - (int)*

- ***The attributes can be classified as follows:***

  Nominal: *Education, Marital-Status*

  Ordinal: *ID, Year-Birth, Dt-Customer*

  Discrete: *Kidhome, Teenhome, Recency, NumDealsPurchases, NumWebPurchases, NumCatalogPurchases, NumStorePurchases, NumWebVisitsMonth, AcceptedCmp1, AcceptedCmp2, AcceptedCmp3, AcceptedCmp4, AcceptedCmp5, Complain, Response*

  Continuous: *Income, MntWines, MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts, MntGoldProds*

Project Aim and Objectives

The aim of this project is to map the buying patterns of the customers in our dataset and analyse whether individual categorised groups of customers are inclined to certain products or methods of shopping. For instance,

1. *Which age group should be targeted for an upcoming new lineup of wine?*
2. *Which product stocks are most critical to the business model?*
3. *Which mode of shopping is most used by the customers?*

There could be unlimited questions that can be raised from a given dataset but the objectives were limited to four objectives for a simple yet detailed analysis into some key avenues of interest.

First, the dataset will be cleaned, statistically analyzed, and described to identify outliers, errors, and useless data. After that, an advanced dataset consisting of only the data columns that will be needed will be created. Then, the project will focus on our 4 avenues of interest in customer buying patterns which aim to categorise the customer base according to classified characteristics and compare trends within each key objective.

Subsequently, the sub-dataset for each objective will be analysed and plotted, and the trends will be assessed between different groups to paint an overall picture of the customer base. Data will be categorised both using numerical analyses as well as through qualitative grouping of similar data. It will aim to measure if there are statistical correlations between different attributes and therefore if there is a real-world explanation for these patterns, feeding back into ideas regarding the optimal marketing strategy to sell products to their ideal customers. Furthermore, it will aim to display our data in the most appealing and simple manner to highlight trends and complement our analyses, whilst trying out innovative methods of multi-faceted data visualization that can capture several types of information simultaneously.

Specific Objective(s)

- **Objective 1:** *The effect of people's income on the products they buy*

For this objective, we wanted to investigate the effect of income on chosen products. We predict that having a greater or smaller income will lead to a difference in priority of purchasing, as some items like food products are more essential and should be purchased by all income groups equally, whilst products like wine and gold are more catered to having a disposable income and would be seen more at higher income levels, leading to a predicted overall higher spend on products for higher income

groups. This investigation will allow us to cater different products to the income group(s) that are making the most purchases.

- **Objective 2:** *The effect of people's relationship status and having children on the products they buy*

We wanted to investigate the effects of being in a relationship and having offspring on their choices in product purchasing. We believe that being in a relationship and having kids would both lead to an increase in the amount of products purchased as there would be a greater number of people to cater for, especially in regards to the food products. Also, we believe that the priorities of purchase would change as being single without children would perhaps lead to more disposable income that could be spent on luxury items, whilst you would have to change your priorities when cohabiting with others. We will then be able to determine which products are better marketed to singles, couples or families.

- **Objective 3:** *Which platform is most used by customers, in-store or online, and the impact of marital status on this choice*

We think that it would be beneficial to look at the trends in preferred purchasing platform as we move into a digital age where almost instantaneous ordering and delivery of goods is possible and would like to determine which products are better suited to promotion on which kind of purchasing platform, in order to maximize sales of each product. Furthermore, we would like to investigate whether being in a relationship will affect people's choices in purchasing platform, as we foresee that it could lead to different priorities in the means of purchasing products when considering the opinions of a partner.

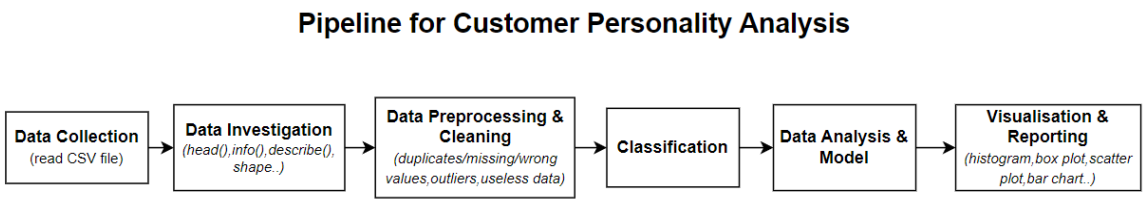
- **Objective 4:** *The effect of people's age on the products they buy*

For this objective, we wanted to see if people's choices in products purchased would change as they moved through different stages in life. We predict that the types off products purchased will change as people have different priorities in spending on essentials products versus luxury items and in consideration of possible family members when in youth, middle age and old age. This is related to our investigation into income, as we believe that with an increase in age, customers may have more disposable income and we can thereby infer which products to market to which age groups.

System Design

Architecture

In this project, the first phase of our pipeline begins with the collection of data from its source files so that analysis can be performed. The data will be extracted from the source in CSV format and read into the necessary subsequent functions. Then the data investigation phase will begin, in which the data type, data size and descriptive statistical information will be obtained for each column, in order to build an initial picture of the dataset. The next stage is data pre-processing and cleaning, in which data will be manipulated and altered according to business needs and the necessary requirements for this data analysis. After obtaining cleaned data, the classification process take place, with data classified in ways that allow each of the 4 objectives to be met. Then is the data analysis and model phase, where the necessary calculations for each objective are made to facilitate analysis and plotting. Here, the data goes through various processes in order to identify possible answers to the defined objectives. Finally, the project output is given in line with the analyses obtained in the last step, visualization, and reporting.



Processing Modules and Algorithms

The dataset that was finalised had a single CSV file and therefore no merging of files was required. When considering all the objectives, it was finalised that the main data frame should be enriched with all additional calculated columns before commencing the objective analysis to have a consistent methodology of analysis and avoid possible ambiguity later. This will help to ensure that the outcomes of different objectives are consistent with each other.

However, it was decided that for missing entries or illogical values, the values will be imputed using a suitable statistical method. In this case, where customer spending analysis is to be performed, the average values of a given cluster/group seem logical as the imputed values, whilst ensuring that this does not disturb the distribution of data in that particular cluster/group.

Once the main data frame is ready for analysis than various dictionaries and lists will be created for analysis and plotting. To list these steps

- *Cleaning data by imputing outliers and removing unnecessary columns*
- *Modifying data frame with additiona; calculated values*
- *Grouping data, for instance, according to age group or education for analysis*
- *Constructing dictionaries and lists for plotting and analysis*
- *Plotting the data to visualise results and deduce outcomes*

Program Code

Importing Libraries

The libraries to be used in the analysis process are imported as the first step.

In [1]:
```
# The library for data processing
import pandas as pd

# The library for linear algebra
import numpy as np

# The library for data vizualization
import matplotlib.pyplot as plt
from matplotlib.gridspec import GridSpec
from matplotlib import cm
from matplotlib.colors import ListedColormap, LinearSegmentedColormap
import matplotlib as mpl

# The library for data vizualization
import seaborn as sn

pd.options.mode.chained_assignment = None
```

Extracting Data from CSV File

The raw data was read from the `CSV file`, and `pandas` was used to convert the raw data to a `DataFrame`.

In [2]:
```
file_name = ( "marketing_campaign.csv") # defining file name
main_DF = pd.read_csv( file_name , sep='\t') # reading the data from CSV
```

Data Investigation Process

Quick Testing: *The first 5 rows of the dataset are shown for an overview to see what data the file contains by using `head()` function. The `shape` attribute is used to obtain the shape of a DataFrame.*

In [3]:
```
display(main_DF.head()) # Returns the first 5 rows to gain a general idea

main_DF.shape # enables us to obtain the shape of a DataFrame
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Custo
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2

5 rows × 29 columns

Out[3]:
```
(2240, 29)
```

The raw data consists of 29 columns and 2240 rows in total.

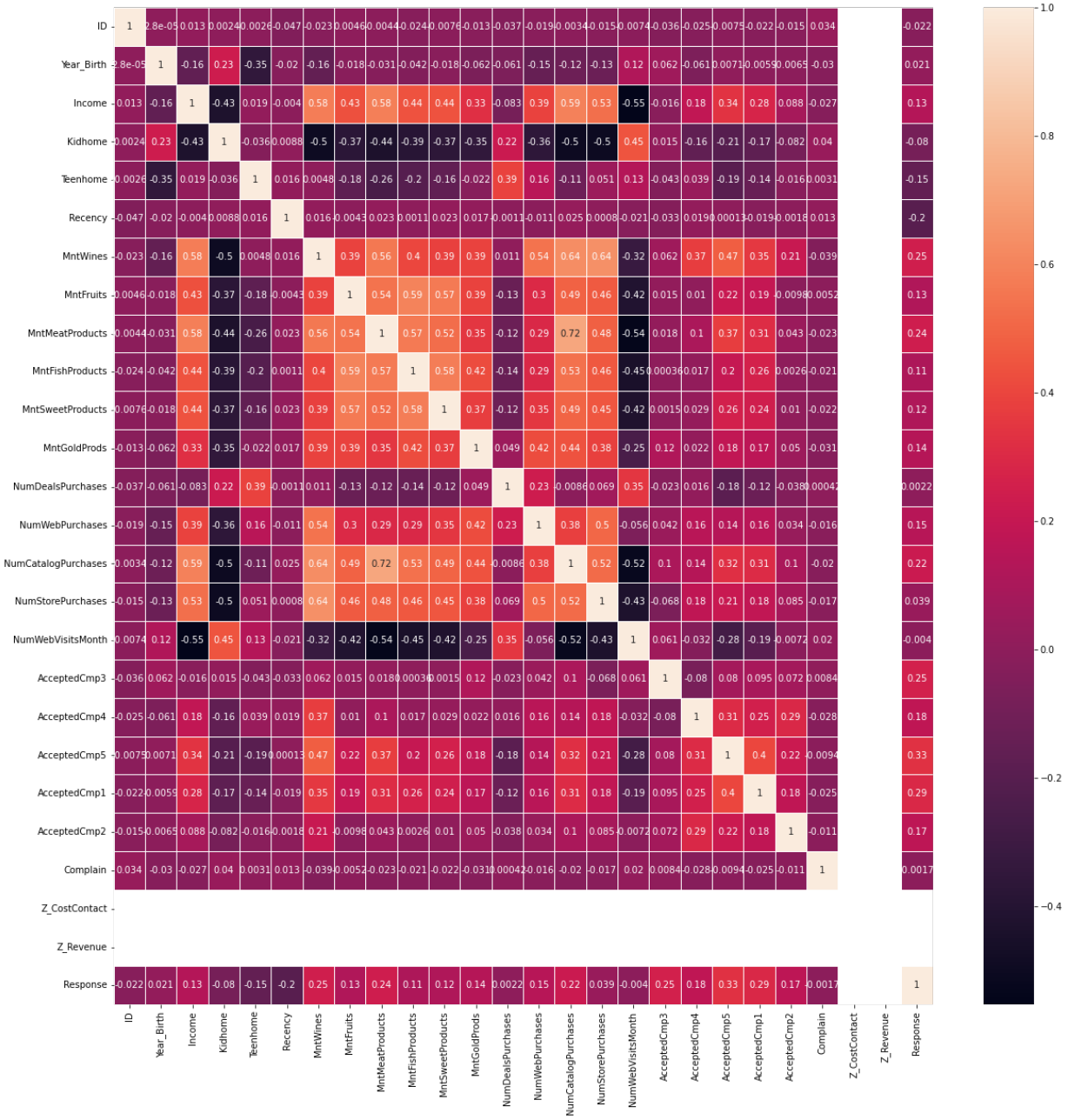
Checking Correlation

Before determining the objective, the correlation between the attributes was checked.

Correlation analysis reveals offsprings between different attributes, and addressing these offsprings can yield new insights. Therefore, it is important to check correlations before setting objectives.

In [4]:
```
fig, ax = plt.subplots(figsize=(20,20))
sn.heatmap(main_DF.corr(), annot=True, linewidths=.5, ax=ax) # for a grap
```

Out[4]:
```
<AxesSubplot:>
```



Identifying Column Types and See Description of the Data: *The `info()` method is used to see the data types of the columns. The `describe()` method is used to see the description of the data.*

In [5]:
```
main_DF.info() # prints information about the DataFrame
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 29 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   ID                   2240 non-null   int64  
 1   Year_Birth           2240 non-null   int64  
 2   Education            2240 non-null   object 
 3   Marital_Status       2240 non-null   object 
 4   Income               2216 non-null   float64
 5   Kidhome              2240 non-null   int64  
 6   Teenhome             2240 non-null   int64  
 7   Dt_Customer          2240 non-null   object 
 8   Recency              2240 non-null   int64  
 9   MntWines             2240 non-null   int64  
 10  MntFruits            2240 non-null   int64  
 11  MntMeatProducts      2240 non-null   int64  
 12  MntFishProducts      2240 non-null   int64  
 13  MntSweetProducts     2240 non-null   int64  
 14  MntGoldProds         2240 non-null   int64  
 15  NumDealsPurchases    2240 non-null   int64  
 16  NumWebPurchases      2240 non-null   int64  
 17  NumCatalogPurchases  2240 non-null   int64  
 18  NumStorePurchases    2240 non-null   int64  
 19  NumWebVisitsMonth    2240 non-null   int64  
 20  AcceptedCmp3         2240 non-null   int64  
 21  AcceptedCmp4         2240 non-null   int64  
 22  AcceptedCmp5         2240 non-null   int64  
 23  AcceptedCmp1         2240 non-null   int64  
 24  AcceptedCmp2         2240 non-null   int64  
 25  Complain             2240 non-null   int64  
 26  Z_CostContact        2240 non-null   int64  
 27  Z_Revenue            2240 non-null   int64  
 28  Response             2240 non-null   int64  
dtypes: float64(1), int64(25), object(3)
memory usage: 507.6+ KB
```

The raw data contains 1 float type, 25 integer type and 3 object type data.

In [6]:
```
main_DF.describe() #  returns description of the data in the DataFrame
```

Out[6]:

	ID	Year_Birth	Income	Kidhome	Teenhome	Recer
count	2240.000000	2240.000000	2216.000000	2240.000000	2240.000000	2240.0000
mean	5592.159821	1968.805804	52247.251354	0.444196	0.506250	49.1093
std	3246.662198	11.984069	25173.076661	0.538398	0.544538	28.9624
min	0.000000	1893.000000	1730.000000	0.000000	0.000000	0.0000
25%	2828.250000	1959.000000	35303.000000	0.000000	0.000000	24.0000
50%	5458.500000	1970.000000	51381.500000	0.000000	0.000000	49.0000
75%	8427.750000	1977.000000	68522.000000	1.000000	1.000000	74.0000
max	11191.000000	1996.000000	666666.000000	2.000000	2.000000	99.0000

8 rows × 26 columns

*From this table, we can see that the **Income** column has missing data. Comparing the 75% quartile and Maximum value for **Income**, we can see that the data may contain outliers.*

Data Cleaning Process & Preprocessing

Adding Calculated Columns

- *Adding `Age` column to use in our analysis.*

In [7]:
```
main_Year_Birth = main_DF["Year_Birth"].to_numpy()  # used to return a Nu
main_Age = 2022 - main_Year_Birth # calculating Age
if 'Age' not in main_DF: # check if Age column exists or not
      main_DF.insert(3, 'Age', main_Age.tolist()) # adding Age column in
main_DF.head()
```

Out[7]:

	ID	Year_Birth	Education	Age	Marital_Status	Income	Kidhome	Teenhome	Dt_
0	5524	1957	Graduation	65	Single	58138.0	0	0	04
1	2174	1954	Graduation	68	Single	46344.0	1	1	08
2	4141	1965	Graduation	57	Together	71613.0	0	0	21
3	6182	1984	Graduation	38	Together	26646.0	1	0	10
4	5324	1981	PhD	41	Married	58293.0	1	0	19

5 rows × 30 columns

- *Adding `Total_Spending` column to use in our analysis.*

In [8]:
```
if 'Total_Spending' not in main_DF: # check if Total_Spending column exis
    main_DF['Total_Spending'] = main_DF["MntWines"]+main_DF["MntFruits"]+
main_DF.head()
```

Out[8]:

	ID	Year_Birth	Education	Age	Marital_Status	Income	Kidhome	Teenhome	Dt_(
0	5524	1957	Graduation	65	Single	58138.0	0	0	04
1	2174	1954	Graduation	68	Single	46344.0	1	1	08
2	4141	1965	Graduation	57	Together	71613.0	0	0	21
3	6182	1984	Graduation	38	Together	26646.0	1	0	10
4	5324	1981	PhD	41	Married	58293.0	1	0	19

5 rows × 31 columns

- *Adding `Total_Purchases` column to use in our analysis.*

In [9]:
```
if 'Total_Purchases' not in main_DF: # check if Total_Purchases column ex
    main_DF['Total_Purchases'] = main_DF["NumWebPurchases"] + main_DF["Nu
main_DF.head()
```

Out[9]:

	ID	Year_Birth	Education	Age	Marital_Status	Income	Kidhome	Teenhome	Dt_(
0	5524	1957	Graduation	65	Single	58138.0	0	0	04
1	2174	1954	Graduation	68	Single	46344.0	1	1	08
2	4141	1965	Graduation	57	Together	71613.0	0	0	21
3	6182	1984	Graduation	38	Together	26646.0	1	0	10
4	5324	1981	PhD	41	Married	58293.0	1	0	19

5 rows × 32 columns

Checking for Duplicate Values: *The `duplicated()` method is used to checked if there are any duplicate values in the dataset.*

In [10]:
```
def remove_duplicates(dfName): # function to remove duplicates values fro

  duplicates = dfName.duplicated().sum() #returns number of duplicate val

  if duplicates > 0:
    dfName = dfName.drop_duplicates()
    print('"{}" duplicate values are removed from the DataFrame.'.format(
  else:
    print('This dataframe does not contain any duplicate values.\nDuplica

remove_duplicates(main_DF)
```

```
This dataframe does not contain any duplicate values.
Duplicate value: 0
```

It returned `0` as a result, meaning that the dataset does not contain any duplicate values.

In [11]:
```
main_DF.duplicated().value_counts() #returns number of duplicate values -
```

Out[11]:
```
False    2240
dtype: int64
```

It returned `False - 2240` as a result so it means that DataFrame has 2240 unique data points, so the dataset does not contain any duplicate values.

Checking for Missing Values: *The `isnull()` methods is used to detect missing values.*

In [12]:
```
def check_missing_values(dfName): # function to check missing values in t
    for i in dfName :
        missingValue = dfName[i].isnull().sum()
        if missingValue>0:
          print('"{}" column has {} missing values out of {}.'.format(i,m
    return

check_missing_values(main_DF)
```

```
"Income" column has 24 missing values out of 2240.
```

Dropping Useless Columns: *The `drop()` methods is used to remove the specified unnecessary columns.*

In [13]:
```
main_DF.drop(['ID','Z_CostContact','Z_Revenue','Recency','Dt_Customer'],
main_DF.shape
```

Out[13]:
```
(2240, 27)
```

The above columns will not be used in our analysis so they were deleted from the dataframe to reduce the data size.

Filling Missing Values: *Missing data in the `Income` column are filled in according to the `average` value of the `Education` level they belong to.*

By using the `isna()` function, we can see the rows with missing data.

In [14]:
```
main_DF[main_DF['Income'].isna()] # detect missing values
```

Out[14]:

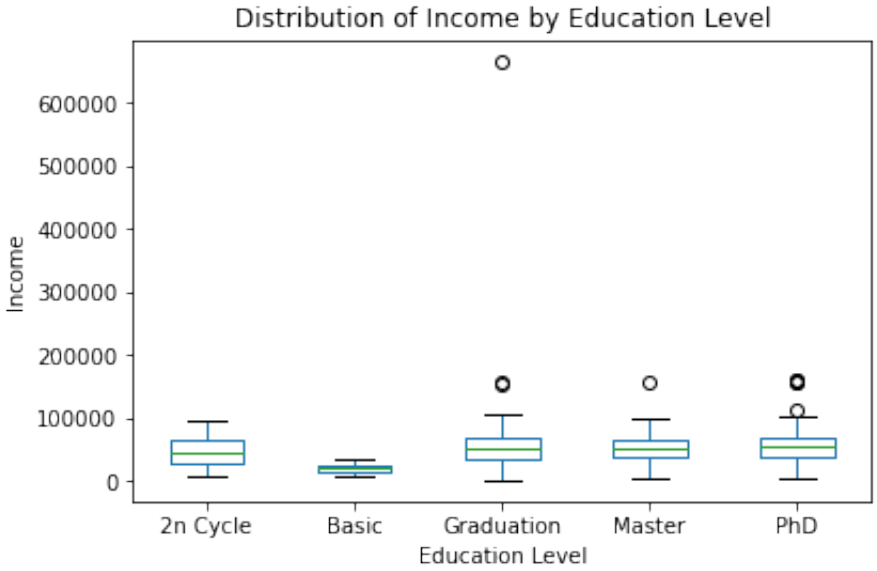
	Year_Birth	Education	Age	Marital_Status	Income	Kidhome	Teenhome	MntWin
10	1983	Graduation	39	Married	NaN	1	0	
27	1986	Graduation	36	Single	NaN	1	0	
43	1959	PhD	63	Single	NaN	0	0	
48	1951	Graduation	71	Single	NaN	2	1	
58	1982	Graduation	40	Single	NaN	1	0	
71	1973	2n Cycle	49	Married	NaN	1	0	
90	1957	PhD	65	Married	NaN	2	1	2
91	1957	Graduation	65	Single	NaN	1	1	
92	1973	Master	49	Together	NaN	0	0	4
128	1961	PhD	61	Married	NaN	0	1	3
133	1963	Graduation	59	Married	NaN	0	1	2
312	1989	Graduation	33	Married	NaN	0	0	8
319	1970	Graduation	52	Single	NaN	1	2	7
1379	1970	Master	52	Together	NaN	0	1	1
1382	1958	Graduation	64	Together	NaN	1	1	
1383	1964	2n Cycle	58	Single	NaN	1	1	
1386	1972	PhD	50	Together	NaN	1	0	
2059	1969	Master	53	Together	NaN	1	1	3
2061	1981	PhD	41	Single	NaN	1	0	
2078	1971	Graduation	51	Married	NaN	1	1	
2079	1954	Master	68	Together	NaN	0	1	1
2081	1955	Graduation	67	Single	NaN	0	1	2
2084	1943	Master	79	Widow	NaN	0	0	5
2228	1978	2n Cycle	44	Together	NaN	0	0	

24 rows × 27 columns

The distribution of Income according to Education level was controlled via box plot. The `boxplot()` method is used to draw a box plot.

If there is a normal distribution, using the mean value will not affect the overall distribution in the data set. Therefore, the missing data in the income column were filled with the average value of the education level they belonged to.

In [15]:
```
boxplot = main_DF.boxplot(column=['Income'], by=['Education'], grid=False

title_boxplot = 'Distribution of Income by Education Level'
plt.title( title_boxplot )
plt.suptitle('')
boxplot.set_xlabel("Education Level");
boxplot.set_ylabel("Income");
plt.show()
```



When we look at the box plot for each Education Level, we can see that the box plot is evenly distributed around the median. We can infer that the distribution is symmetric or normally distributed, so the mean value of each Education Level will be used to fill missing values.

In [16]:
```
mean_incomes=main_DF[['Education','Income']].groupby(['Education']).mean(
mean_incomes_dF=pd.DataFrame(mean_incomes)
mean_incomes_dF.reset_index(inplace=True)

for i in range(len(main_DF)):  # filling missing Income column with the a
    if not main_DF['Income'][i]>0:
        if  main_DF['Education'][i] == '2n Cycle':
            main_DF['Income'][i] =  mean_incomes_dF['Income'][mean_income
        elif main_DF['Education'][i] == 'Basic':
            main_DF['Income'][i] = mean_incomes_dF['Income'][mean_incomes
        elif main_DF['Education'][i] == 'Graduation':
            main_DF['Income'][i] = mean_incomes_dF['Income'][mean_incomes
        elif main_DF['Education'][i] == 'Master':
            main_DF['Income'][i] = mean_incomes_dF['Income'][mean_incomes
        elif main_DF['Education'][i] == 'PhD':
            main_DF['Income'][i] = mean_incomes_dF['Income'][mean_incomes
        else:
            pass
main_DF[main_DF['Income'].isna()]
```

Out[16]:

Year_Birth	Education	Age	Marital_Status	Income	Kidhome	Teenhome	MntWines	M

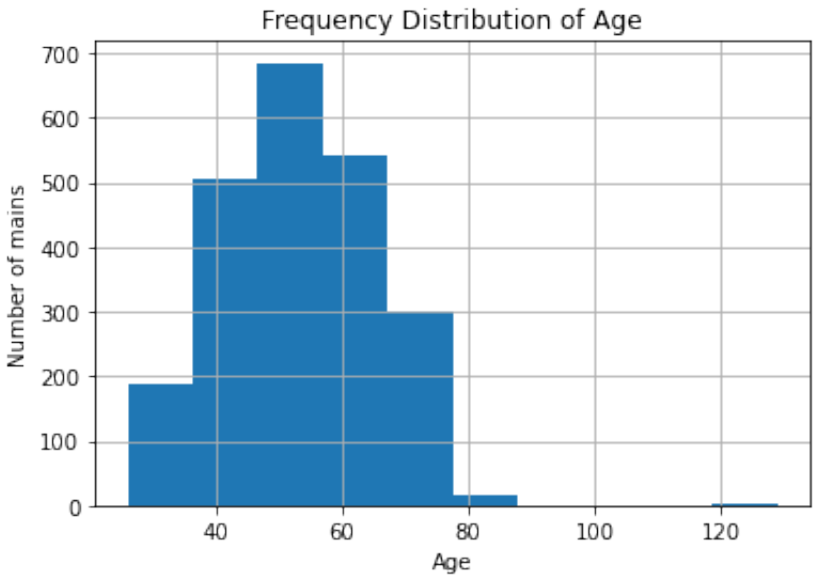
0 rows × 27 columns

Classification Process

Drawing Histogram to determine the range for 'Age' Column

To use in our analysis, `Age Groups` *are defined and this classification is added to the data set as a new column which is named* `Age–Group` *. The histogram was used to determine the age ranges. The* `hist()` *method is used to draw a histogram.*

In [17]:
```
main_DF.hist(column='Age')
title_histogram = 'Frequency Distribution of Age'
plt.title( title_histogram )
plt.suptitle('')
plt.xlabel("Age");
plt.ylabel("Number of mains");
plt.show()
```



Based on the frequencies that were observed, the data is grouped by age to gain an understanding of the different customer age groups

NOTE: The smallest Age in the data set is 26, so the first age bracket is defined to be starting from 25

Age Groups

- *Youngsters = 25-34*
- *Middle Aged = 35-50*
- *Elders = 50 - 70*
- *Main = 70+*

Adding New Column for 'Age Group'

In [18]:
```
age_group =[]
for age in main_DF['Age']:
    if (age>=25) & (age<=34):
        age_group.append("Youngsters")
    elif (age>34) & (age<=50):
        age_group.append("Middle_Aged")
    elif (age>50) & (age<=70):
        age_group.append("Old")
    elif (age>70):
        age_group.append("Elderly")
    else:
        age_group.append("Age_NAN")

if 'Age_Group' not in main_DF:
      main_DF.insert(loc=5,column='Age_Group',value=age_group)
```

- ***Additional Step:*** *Thanks to histogram, looking for Ages among the whole data, we found that there were wrong values or outliers. Therefore, we defined the Age limit as a maximum of 100 and removed any data above this from the dataset.*

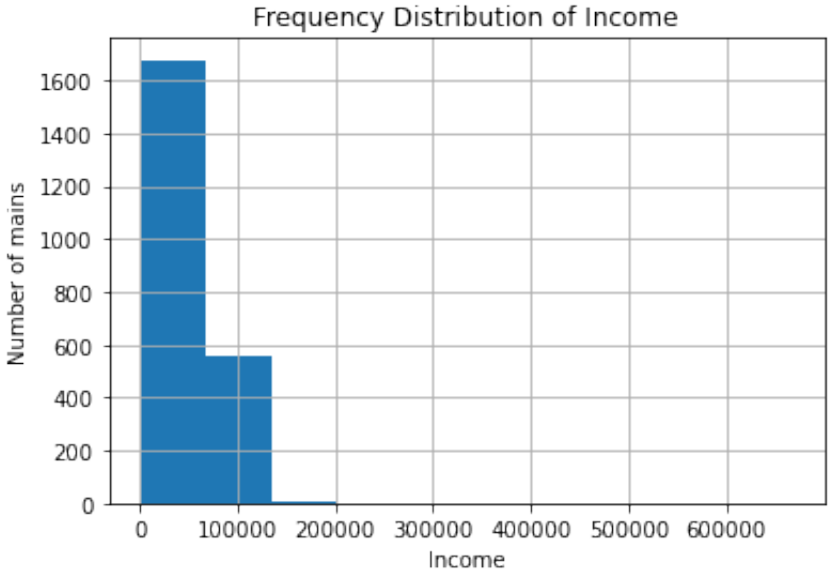
In [19]:
```
outliers_Age = main_DF[main_DF['Age'] > 100].index
outliers_Age
main_DF.drop(outliers_Age , inplace=True)
main_DF.shape
```

Out[19]:
```
(2237, 28)
```

Drawing Histogram to determine the range for 'Income' Column

To use in our analysis, `Income Groups` are defined and this classification is added to the data set as a new column which is named `Income-Group`. The histogram was used to determine the age ranges. The `hist()` method is used to draw a histogram.

In [20]:
```
main_DF.hist(column='Income')
title_histogram = 'Frequency Distribution of Income'
plt.title( title_histogram )
plt.suptitle('')
plt.xlabel("Income");
plt.ylabel("Number of mains");
plt.show()
```



- ***Additional Step:*** *As we can see from the previous boxplot and histogram, the `Income` column contains outlier data. We will set the upper limit for the `Income` column as 200000 and remove the data above this value from the dataset.*

In [21]:
```
outliers_Income = main_DF[main_DF['Income'] > 200000].index
outliers_Income
main_DF.drop(outliers_Income , inplace=True)
main_DF.shape
```

Out[21]:
```
(2236, 28)
```

According to the histogram, income ranges are determined as follows.

Income Groups:

- *I-0-30000 = 0-30000*
- *I-30001-60000 = 30001-60000*
- *I-60001-90000 = 60001-90000*
- *I-90000+ = 90000+*

Adding New Column for 'Income Group'

A column is added to the dataframe including the newly defined incomes groups.

In [22]:
```
income_group =[]

for income in main_DF['Income']:
    if (income>0) & (income<=30000):
        income_group.append("0-30000")
    elif (income>30000) & (income<=60000):
        income_group.append("30001-60000")
    elif (income>60000) & (income<=90000):
        income_group.append("60001-90000")
    elif (income>90000):
        income_group.append("90000+")
    else:
        income_group.append("Income_NAN")

if 'Income_Group' not in main_DF:
      main_DF.insert(loc=5,column='Income_Group',value=income_group)
main_DF.head()
```

Out[22]:

	Year_Birth	Education	Age	Marital_Status	Income	Income_Group	Age_Group	Kid
0	1957	Graduation	65	Single	58138.0	30001-60000	Old	
1	1954	Graduation	68	Single	46344.0	30001-60000	Old	
2	1965	Graduation	57	Together	71613.0	60001-90000	Old	
3	1984	Graduation	38	Together	26646.0	0-30000	Middle_Aged	
4	1981	PhD	41	Married	58293.0	30001-60000	Middle_Aged	

5 rows × 29 columns

Adding New Column for 'Relationship Status'

The marital status data contains some strange inputs such as "Absurd" and "YOLO" which are hard to define, along with some small groups such as "Widow" and "Alone". To simplify the data, each of these is recategorised into either "Relationship" or "No relationship", allowing for simplicity and consistency in subsequent analyses.

In [23]:
```
main_Marital_Status = main_DF["Marital_Status"].to_list()
print(main_DF['Marital_Status'].value_counts())

main_Relationship = []
for item in main_Marital_Status:
    if item in ["Married", "Together"]:
        main_Relationship.append("Relationship")
    else:
        main_Relationship.append("No relationship")

if 'Relationship' not in main_DF:
    main_DF.insert(4, 'Relationship', main_Relationship)

main_DF.head()
```

```
Married     864
Together    578
Single      479
Divorced    231
Widow        77
Alone         3
Absurd        2
YOLO          2
Name: Marital_Status, dtype: int64
```

Out[23]:

	Year_Birth	Education	Age	Marital_Status	Relationship	Income	Income_Group	A
0	1957	Graduation	65	Single	No relationship	58138.0	30001-60000	
1	1954	Graduation	68	Single	No relationship	46344.0	30001-60000	
2	1965	Graduation	57	Together	Relationship	71613.0	60001-90000	
3	1984	Graduation	38	Together	Relationship	26646.0	0-30000	Mid
4	1981	PhD	41	Married	Relationship	58293.0	30001-60000	Mid

5 rows × 30 columns

Adding New Column for 'Offspring Status'

Combining the number of kids or teenagers into the household into a a new column counting the number of offspring in a household. Also adding a column indicating the status of having a household with or without offspring.

In [24]:
```
main_Youthhome = main_DF["Kidhome"].to_numpy() + main_DF["Teenhome"].to_n
main_Offspring = []

for item in main_Youthhome:
    if item > 1:
        main_Offspring.append("Offspring")
    else:
        main_Offspring.append("No offspring")

if 'Offspring' not in main_DF:
    main_DF.insert(10, 'Offspring', main_Offspring)
main_DF.head()
```

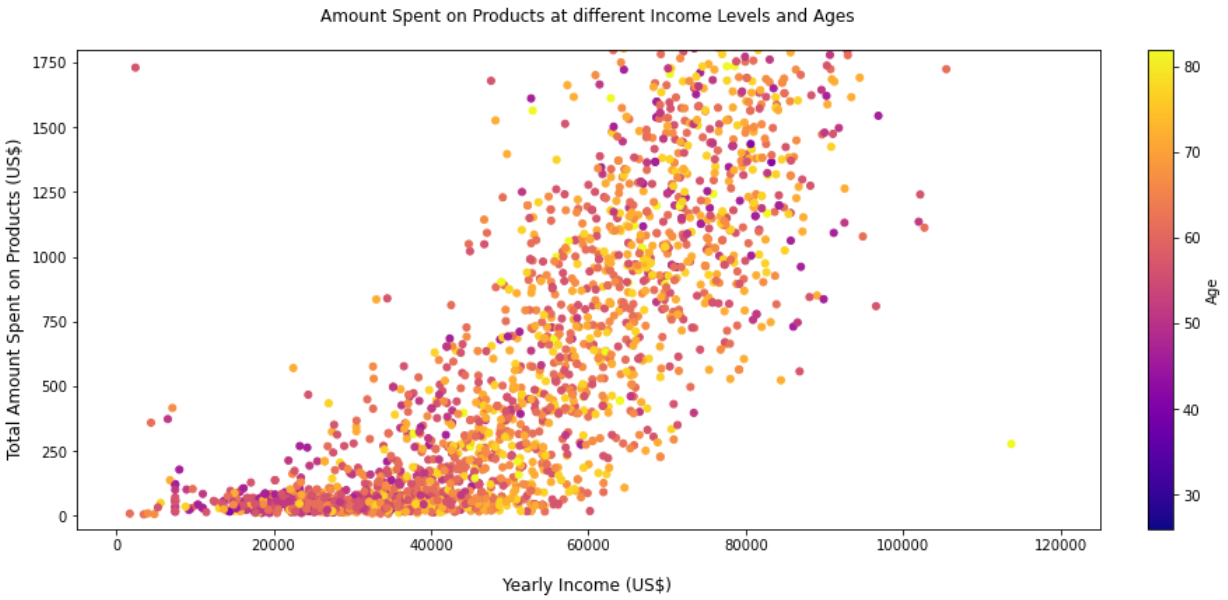
Out[24]:

	Year_Birth	Education	Age	Marital_Status	Relationship	Income	Income_Group	A
0	1957	Graduation	65	Single	No relationship	58138.0	30001-60000	
1	1954	Graduation	68	Single	No relationship	46344.0	30001-60000	
2	1965	Graduation	57	Together	Relationship	71613.0	60001-90000	
3	1984	Graduation	38	Together	Relationship	26646.0	0-30000	Mid
4	1981	PhD	41	Married	Relationship	58293.0	30001-60000	Mid

5 rows × 31 columns

Visualizing Our Data

In [25]:
```
viridis = cm.get_cmap('plasma', 12)

fig, axs = plt.subplots(1, 2, figsize=(12, 6), gridspec_kw={'width_ratios

axs[0].scatter(main_DF['Income'].to_numpy(), main_DF['Total_Spending'].to

axs[0].set_xlim(-5000, 125000)
axs[0].set_ylim(-50, 1800)

plt.suptitle("Amount Spent on Products at different Income Levels and Age
fig.supxlabel("Yearly Income (US$)")
fig.supylabel("Total Amount Spent on Products (US$)")

fig.tight_layout()

cmap = mpl.cm.plasma
norm = mpl.colors.Normalize(vmin=min(main_DF['Age'].to_numpy()), vmax=max

fig.colorbar(mpl.cm.ScalarMappable(norm=norm, cmap=cmap),
             cax=axs[1], orientation='vertical', label='Age')

plt.show()
```



The above plot gives a picture of our dataset of customers, particularly in their spending on products and how it changes with both income and age. There is a clear positive correlation between income and total amount spent on products, which supports the results of our heatmap analysis. Conversely, there appears to be no clear link between age and either income or total spending, but further quantitative analysis is necessary to verify this.

Data Analysis

Note: *In order not to increase the data size by keeping unnecessary data in the data set, we defined new dataframes for each objective containing only the necessary columns.*

Analysis for Objective 1

- *Objective 1 requires Income Group, MntWines, MntFruits, MntMeatProducts,MntFishProducts,MntSweetProducts and MntGoldProds columns.*

In [26]:
```
objective1_DF = main_DF.filter(['Income_Group','MntWines','MntFruits','Mn
objective1_DF.head()
```

Out[26]:

	Income_Group	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetPr
0	30001-60000	635	88	546	172	
1	30001-60000	11	1	6	2	
2	60001-90000	426	49	127	111	
3	0-30000	11	4	20	10	
4	30001-60000	173	43	118	46	

The new dataframe was created using the `filter()` method.

In [27]:
```
def objective1_Graph():
    plt.figure(1) # to create a first figure object
    product_values = objective1_DF.groupby(['Income_Group']).sum()  # cal
    income_GroupNames = ['I_0-30000','I_30001-60000','I_60001-90000','I_9

    plt.figure(figsize=(8, 8))
    # defining a list that contains the y-axis values
    mntWines = product_values['MntWines'].tolist()
    mntFruits = product_values['MntFruits'].tolist()
    mntMeatProducts = product_values['MntMeatProducts'].tolist()
    mntFishProducts = product_values['MntFishProducts'].tolist()
    mntSweetProducts = product_values['MntSweetProducts'].tolist()
    mntGoldProds = product_values['MntGoldProds'].tolist()


    X_axis = np.arange(len(income_GroupNames))

    plt.bar(X_axis, mntWines, 0.1, label = 'MntWines')
    plt.bar(X_axis - 0.1, mntFruits, 0.1, label = 'MntFruits')
    plt.bar(X_axis + 0.1, mntMeatProducts, 0.1, label = 'MntMeatProducts'
    plt.bar(X_axis + 0.2, mntFishProducts, 0.1, label = 'MntFishProducts'
    plt.bar(X_axis - 0.2, mntSweetProducts, 0.1, label = 'MntSweetProduct
    plt.bar(X_axis + 0.3, mntGoldProds, 0.1, label = 'MntGoldProds')


    plt.xticks(X_axis, income_GroupNames)
    plt.xlabel("Income Groups")
    plt.ylabel("Amount of Products")
    plt.title("Amount of Products by Income Group")
    plt.legend()
    plt.show()
    plt.close()

    plt.figure(2) # to create a second figure object
    plt.figure(figsize=(8, 8))

    # plot lines
    plt.plot(income_GroupNames, mntWines, label = "MntWines")
    plt.plot(income_GroupNames, mntMeatProducts, label = "MntMeatProducts
    plt.plot(income_GroupNames, mntFishProducts, label = "MntFishProducts
    plt.plot(income_GroupNames, mntSweetProducts, label = "MntSweetProduc
    plt.plot(income_GroupNames, mntGoldProds, label = "MntGoldProds")

    plt.xlabel("Income Groups")
    plt.ylabel("Amount of Products")
    plt.title("Amount of Products by Income Group")

    plt.legend()
    plt.show()

    return
```

Analysis for Objective 2

- *Objective 2 requires Relationship,Offspring, MntWines, MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts, MntGoldProds columns.*

In [28]:
```
objective2_DF = main_DF.filter(['Relationship','Offspring', 'MntWines', '
objective2_DF.head()
```

Out[28]:

	Relationship	Offspring	MntWines	MntFruits	MntMeatProducts	MntFishProducts	Mn
0	No relationship	No offspring	635	88	546	172	
1	No relationship	Offspring	11	1	6	2	
2	Relationship	No offspring	426	49	127	111	
3	Relationship	No offspring	11	4	20	10	
4	Relationship	No offspring	173	43	118	46	

The new dataframe was created using the `filter()` method.

- ***Analysis based on Relationship Status***

In [29]:
```
main_DF_Relationship = main_DF[main_DF.Relationship == "Relationship"]
main_DF_No_relationship = main_DF[main_DF.Relationship == "No relationshi
```

Two sub dataframes were created for Relationship Status to calculate the average amount of spendings on different products.

In [30]:
```
main_NP_MntWines_Relationship_average = sum(main_DF_Relationship["MntWine
main_NP_MntFruits_Relationship_average = sum(main_DF_Relationship["MntFru
main_NP_MntMeatProducts_Relationship_average = sum(main_DF_Relationship["
main_NP_MntFishProducts_Relationship_average = sum(main_DF_Relationship["
main_NP_MntSweetProducts_Relationship_average = sum(main_DF_Relationship[
main_NP_MntGmainProds_Relationship_average = sum(main_DF_Relationship["Mn
```

The average amounts of spending on Wines, Fruits, Meat, Fish, Sweets and Gold were calculated for people who are in a relationship.

In [31]:
```
main_NP_MntWines_No_relationship_average = sum(main_DF_No_relationship["M
main_NP_MntFruits_No_relationship_average = sum(main_DF_No_relationship["
main_NP_MntMeatProducts_No_relationship_average = sum(main_DF_No_relation
main_NP_MntFishProducts_No_relationship_average = sum(main_DF_No_relation
main_NP_MntSweetProducts_No_relationship_average = sum(main_DF_No_relatio
main_NP_MntGmainProds_No_relationship_average = sum(main_DF_No_relationsh
```

The average amounts of spending on Wines, Fruits, Meat, Fish, Sweets and Gold were calculated for people who are not in a relationship.

In [32]:
```
def objective2_relationship_Graph():
    fig, axs = plt.subplots(1, 2, figsize=(10, 5))

    axs[0].bar(0.6, main_NP_MntWines_Relationship_average, 0.2)
    axs[0].bar(0.8, main_NP_MntFruits_Relationship_average, 0.2)
    axs[0].bar(1, main_NP_MntMeatProducts_Relationship_average, 0.2)
    axs[0].bar(1.2, main_NP_MntFishProducts_Relationship_average, 0.2)
    axs[0].bar(1.4, main_NP_MntSweetProducts_Relationship_average, 0.2)
    axs[0].bar(1.6, main_NP_MntGmainProds_Relationship_average, 0.2)

    axs[1].bar(0.6, main_NP_MntWines_No_relationship_average, 0.2)
    axs[1].bar(0.8, main_NP_MntFruits_No_relationship_average, 0.2)
    axs[1].bar(1, main_NP_MntMeatProducts_No_relationship_average, 0.2)
    axs[1].bar(1.2, main_NP_MntFishProducts_No_relationship_average, 0.2)
    axs[1].bar(1.4, main_NP_MntSweetProducts_No_relationship_average, 0.2
    axs[1].bar(1.6, main_NP_MntGmainProds_No_relationship_average, 0.2)

    axs[0].set_ylim(0, 350)
    axs[1].set_ylim(0, 350)

    axs[0].set_xticks([0.6, 0.8, 1, 1.2, 1.4, 1.6], ["Wines", "Fruits", "
    axs[1].set_xticks([0.6, 0.8, 1, 1.2, 1.4, 1.6], ["Wines", "Fruits", "

    axs[0].set_xlabel("In a relationship")
    axs[1].set_xlabel("Not in a relationship")

    fig.suptitle("Amount Spent on Products, split by Relationship Status"
    fig.supxlabel("Product Type")
    fig.supylabel("Average Amount Spent on Products (US$)")

    plt.show()
    return
```

- ***Analysis based on having children***

In [33]:
```
main_DF_Offspring = main_DF[main_DF.Offspring == "Offspring"]
main_DF_No_offspring = main_DF[main_DF.Offspring == "No offspring"]
```

Two sub dataframes were created for offspring status to calculate the average amount of spending on different products.

In [34]:
```
main_NP_MntWines_Offspring_average = sum(main_DF_Offspring["MntWines"].to
main_NP_MntFruits_Offspring_average = sum(main_DF_Offspring["MntFruits"].
main_NP_MntMeatProducts_Offspring_average = sum(main_DF_Offspring["MntMea
main_NP_MntFishProducts_Offspring_average = sum(main_DF_Offspring["MntFis
main_NP_MntSweetProducts_Offspring_average = sum(main_DF_Offspring["MntSw
main_NP_MntGmainProds_Offspring_average = sum(main_DF_Offspring["MntGoldP
```

The average amounts of spending on Wines, Fruits, Meat, Fish, Sweets and Gold were calculated for people who have offspring.

In [35]:
```
main_NP_MntWines_No_offspring_average = sum(main_DF_No_offspring["MntWine
main_NP_MntFruits_No_offspring_average = sum(main_DF_No_offspring["MntFru
main_NP_MntMeatProducts_No_offspring_average = sum(main_DF_No_offspring["
main_NP_MntFishProducts_No_offspring_average = sum(main_DF_No_offspring["
main_NP_MntSweetProducts_No_offspring_average = sum(main_DF_No_offspring[
main_NP_MntGmainProds_No_offspring_average = sum(main_DF_No_offspring["Mn
```

The average amounts of spending on Wines, Fruits, Meat, Fish, Sweets and Gold were calculated for people who do not have offspring.

In [36]:
```
def objective2_offspring_Graph():
    fig, axs = plt.subplots(1, 2, figsize=(10, 5))

    axs[0].bar(0.6, main_NP_MntWines_Offspring_average, 0.2)
    axs[0].bar(0.8, main_NP_MntFruits_Offspring_average, 0.2)
    axs[0].bar(1, main_NP_MntMeatProducts_Offspring_average, 0.2)
    axs[0].bar(1.2, main_NP_MntFishProducts_Offspring_average, 0.2)
    axs[0].bar(1.4, main_NP_MntSweetProducts_Offspring_average, 0.2)
    axs[0].bar(1.6, main_NP_MntGmainProds_Offspring_average, 0.2)

    axs[1].bar(0.6, main_NP_MntWines_No_offspring_average, 0.2)
    axs[1].bar(0.8, main_NP_MntFruits_No_offspring_average, 0.2)
    axs[1].bar(1, main_NP_MntMeatProducts_No_offspring_average, 0.2)
    axs[1].bar(1.2, main_NP_MntFishProducts_No_offspring_average, 0.2)
    axs[1].bar(1.4, main_NP_MntSweetProducts_No_offspring_average, 0.2)
    axs[1].bar(1.6, main_NP_MntGmainProds_No_offspring_average, 0.2)

    axs[0].set_ylim(0, 350)
    axs[1].set_ylim(0, 350)

    axs[0].set_xticks([0.6, 0.8, 1, 1.2, 1.4, 1.6], ["Wines", "Fruits", "
    axs[1].set_xticks([0.6, 0.8, 1, 1.2, 1.4, 1.6], ["Wines", "Fruits", "

    axs[0].set_xlabel("Has offspring")
    axs[1].set_xlabel("No offspring")

    fig.suptitle("Amount Spent on Products, split by Offspring Status")
    fig.supxlabel("Product Type")
    fig.supylabel("Average Amount Spent on Products (US$)")

    plt.show()
    return
```

Analysis for Objective 3

- *Objective 3 requires only NumWebPurchases, NumStorePurchases, Total-Purchases and Relationship columns.*

Creating a new DataFrame containing four columns from the original DataFrame: NumWebPurchases, NumStorePurchases, Total-Purchases, and Relationship to plot these values to achieve the objectives.

The column named 'Total-Purchases' represents the total number of purchases, which is the sum of the two columns NumWebPurchases and NumStorePurchases.

In [64]:
```
objective3_DF = main_DF.filter(['NumWebPurchases','NumStorePurchases','To
objective3_DF.head()
```

Out[64]:

	NumWebPurchases	NumStorePurchases	Total_Purchases	Relationship
0	8	4	12	No relationship
1	1	2	3	No relationship
2	8	10	18	Relationship
3	2	4	6	Relationship
4	5	6	11	Relationship

The new dataframe was created using the `filter()` method.

- ***Analysis based on Shopping Platform***

In [65]:
```
def objective3_platform_Graph():
    fig, ax = plt.subplots(figsize=(12,6))

    #here defined a list that contains the x-axis value.
    platform = ['In Store', 'Website']

    #here defined a list that contains the y-axis value.
    total = [objective3_DF['NumStorePurchases'].sum(),objective3_DF['NumW

    #here defined a list that contains the color for each bar.
    bar_color = ['green','blue']
    #here defined a variable that represent the bar size.
    bar_width = 0.2

    # create a bar chart with x-axis the list defined above 'Platform' an
    ax.bar(platform,total, bar_width, color = bar_color)
    # write a label for x-axis
    ax.set_xlabel('Platforms', fontweight='bold',fontsize=16)
    # write a label for y-axis
    ax.set_ylabel('Totla Number Of Purchases', fontweight='bold',fontsize
    # write the value above each bar.
    ax.bar_label(ax.containers[0], label_type='edge')
    # write a title for the bar chart.
    ax.set_title("Which platform is mostly used? ", fontsize=16,  loc='ce

    # show the chart
    plt.show()
    return
```

- ***The effect of relationship status on chosen platform***

In this analysis, it is investigated whether relationship Status affects the chosen method of shopping. For example, do married customers prefer purchasing in-store?

Firstly, determine the total number of purchases for each platform, and then add the values to lists. The first list contains the total number of purchases in-store and the total number of purchases online in the 'InRelationship' dataframe and the second list contains the total number of purchases in-store and the total number of purchases online in the 'NotRelationship' dataframe.

In [39]:
```
# Create dataFrames
InRelationship_df = objective3_DF[objective3_DF['Relationship']== 'Relati
NotRelationship_df = objective3_DF[objective3_DF['Relationship']== 'No re
# find the total number of purchases in dataframe InRelationship_df
sum_in_R = InRelationship_df['NumStorePurchases'].sum()
sum_online_R = InRelationship_df['NumWebPurchases'].sum()

# find the total number of purchases in dataframe NotRelationship_df
sum_in_N = NotRelationship_df['NumStorePurchases'].sum()
sum_online_N = NotRelationship_df['NumWebPurchases'].sum()

# add the values into a lists
list_in_R = [sum_in_R,sum_online_R]
list_not_R = [sum_in_N,sum_online_N]
```

Creating a Bar chart which shows the relationship status and number of purchases for each platform (In-store and online).

In [40]:
```
def objective3_marital_Graph():
    fig, ax = plt.subplots(figsize=(12,6))
    # assing the x-axis values for variable x
    x = np.array([1,2])
    # size of bars
    bar_width = 0.4
    # create a bar chart with two diffrent values.  call the lists that d
    ax.bar(x-0.2, list_in_R, bar_width,label="In store")
    ax.bar(x+0.2, list_not_R, bar_width, label="Online")

    # write a title for the bar chart.
    ax.set_title("Effect of relationship status on chosen Shopping Platfo
    # replace the label of x-axis values,
    plt.xticks([1, 2], ['In Relationship','Not in Relationship'])
    # write the calues above each bar
    ax.bar_label(ax.containers[0], label_type='edge')
    ax.bar_label(ax.containers[1], label_type='edge')

    # create a legend.
    ax.legend(title='Platforms', fontsize=10, title_fontsize=30, loc='upp

    # write label for both axis
    ax.set_xlabel('Marital stutus',fontweight='bold',fontsize=16)
    ax.set_ylabel('Number of purchases',fontweight='bold', fontsize=16)
    plt.show()
    return
```

Analysis for Objective 4

- *Objective 4 requires only Age, Age-Group, MntWines, MntFruits,MntMeatProducts, MntFishProducts, MntSweetProducts, and MntGoldProds columns.*

In [41]:
```
spending_by_age = main_DF.filter(["Age","Age_Group","MntWines","MntFruits
spending_by_age.head()
```

Out[41]:

	Age	Age_Group	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSwee
0	65	Old	635	88	546	172	
1	68	Old	11	1	6	2	
2	57	Old	426	49	127	111	
3	38	Middle_Aged	11	4	20	10	
4	41	Middle_Aged	173	43	118	46	

Creating grouped dictionaries and required lists for onward analysis.

In [42]:
```
AgeGroup_sum= spending_by_age.groupby("Age_Group").sum()
AgeGroup_count = spending_by_age.groupby("Age_Group").count()
count= AgeGroup_count["MntWines"].to_numpy()
AgeGroup_sum
```

Out[42]:

Age_Group	Age	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSwee
Elderly	13179	73442	5577	38112	8634	
Middle_Aged	38605	211259	20499	130210	29608	
Old	63129	360030	28338	175976	40097	
Youngsters	3823	35298	4339	29077	5592	

Creating total and average arrays for each product.

In [60]:
```
Age_Group = ["Elderly", "Middle Aged", "Old", "Youngsters"]
wines_sum = AgeGroup_sum["MntWines"].to_numpy()
fruit_sum = AgeGroup_sum["MntFruits"].to_numpy()
meat_sum = AgeGroup_sum["MntMeatProducts"].to_numpy()
fish_sum = AgeGroup_sum["MntFishProducts"].to_numpy()
sweet_sum = AgeGroup_sum["MntSweetProducts"].to_numpy()
gold_sum = AgeGroup_sum["MntGoldProds"].to_numpy()

wines_avg = wines_sum / count
fruit_avg = fruit_sum / count
meat_avg = meat_sum / count
fish_avg = fish_sum / count
sweet_avg = sweet_sum / count
gold_avg = gold_sum / count
```

Total and Average Plots for each Product Group:

Creating a function to draw a plot for the Wines product group.

In [44]:
```
############################ WINES PLOT ###################################
def spending_wine():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Wine")

    # plot bars
    ax[0].bar(Age_Group, wines_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Wine")

    # plot bars
    ax[1].bar(Age_Group, wines_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Creating a function to draw a plot for the Fruit product group.

In [45]:
```
############################ FRUIT PLOT ###################################
def spending_fruit():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Fruits")

    # plot bars
    ax[0].bar(Age_Group, fruit_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Fruits")

    # plot bars
    ax[1].bar(Age_Group, fruit_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Creating a function to draw a plot for the Meat product group.

In [46]:
```
########################## MEAT PLOT ###################################
def spending_meat():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Meat Products")

    # plot bars
    ax[0].bar(Age_Group, fruit_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Meat Products")

    # plot bars
    ax[1].bar(Age_Group, fruit_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Creating a function to draw a plot for the Fish product group.

In [47]:
```
########################## FISH PLOT ###################################
def spending_fish():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Fish Products")

    # plot bars
    ax[0].bar(Age_Group, fish_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Fish Products")

    # plot bars
    ax[1].bar(Age_Group, fish_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Creating a function to draw a plot for the Sweet product group.

In [48]:
```
############################ SWEET PLOT ###################################
def spending_sweet():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Sweet Products")

    # plot bars
    ax[0].bar(Age_Group, sweet_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Sweet Products")

    # plot bars
    ax[1].bar(Age_Group, sweet_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Creating a function to draw a plot for the Gold product group.

In [49]:
```
############################ GOLD PLOT ###################################
def spending_gold():
    fig, ax = plt.subplots(1,2, figsize=(24,6))

    ###### TOTAL SPENDING PLOT############
    # fig, ax = plt.subplots()
    ax[0].set_title("Total Spending on Gold Products")

    # plot bars
    ax[0].bar(Age_Group, gold_sum)

    # axes labels and other attributes
    ax[0].set_xlabel("Age Groups")
    ax[0].set_ylabel('Total Spending in Pounds')

    ###### AVERAGE SPENDING PLOT############
    ax[1].set_title("Average Spending on Gold Products")

    # plot bars
    ax[1].bar(Age_Group, gold_avg)

    # axes labels and other attributes
    ax[1].set_xlabel("Age Groups")
    ax[1].set_ylabel('Total Spending in Pounds')
    return
```

Making necessary calculations for the Summary Table.

In [50]:
```
product_group = AgeGroup_sum.filter(["Age_Group","MntWines","MntFruits","
product_group = product_group.transpose()
product = ["MntWines","MntFruits","MntMeatProducts","MntFishProducts","Mn
product_group["Product"]= product
product_group

young_by_prod = dict(zip(product_group.Product,product_group.Youngsters))
middle_by_prod = dict(zip(product_group.Product,product_group.Middle_Aged
elders_by_prod = dict(zip(product_group.Product,product_group.Elderly))
old_by_prod = dict(zip(product_group.Product,product_group.Old))

wines_sum = AgeGroup_sum["MntWines"].to_numpy()
fruit_sum = AgeGroup_sum["MntFruits"].to_numpy()
meat_sum = AgeGroup_sum["MntMeatProducts"].to_numpy()
fish_sum = AgeGroup_sum["MntFishProducts"].to_numpy()
sweet_sum = AgeGroup_sum["MntSweetProducts"].to_numpy()
gold_sum = AgeGroup_sum["MntGoldProds"].to_numpy()
a= list(young_by_prod.values())
b= list(elders_by_prod.values())
c= list(middle_by_prod.values())
d= list(old_by_prod.values())
```

Creating a function to draw a plot for the Summary.

In [51]:
```
def summary_bar():

    x_tags = ["Wines","Fruits","MeatProducts","FishProducts","SweetProduc
    X_axis = np.arange(len(x_tags))
    plt.bar(X_axis -0.4,a,0.2, label = "Youngsters (25-34)")
    plt.bar(X_axis -0.2,b,0.2, label = "Old (50 - 70)")
    plt.bar(X_axis +0.0,c,0.2, label = "Middle Aged(35 - 50)")
    plt.bar(X_axis +0.2,d,0.2, label = "Elderly (70+)")
    plt.xticks(X_axis, x_tags)
    plt.xlabel("Products")
    plt.ylabel("Amount in GBP")
    plt.xticks(rotation=90)
    plt.legend()
    plt.tight_layout()
    plt.title("Spending of Each Age Group on Different Products")
    return
```

Creating a list for Revenues of each age group to use in the plot.

In [52]:
```
revenue = [sum(wines_sum),sum(meat_sum), sum(fruit_sum ), sum(fish_sum),
revenue_list = ["Wines", "Meats","Fruits", "Fish", "Sweets", "Gold"]
revenue_list
revenue_by_agegroup = AgeGroup_sum
revenue_by_agegroup["Total Spending"] = revenue_by_agegroup["MntWines"]+r
revenue_by_agegroup = revenue_by_agegroup.filter(["Total Spending"])
revenue_by_agegroup
revenue_by_agegroup_list= revenue_by_agegroup["Total Spending"].to_numpy(
revenue_by_agegroup_list
```

Out[52]:
```
array([140522, 447948, 682204,  84312])
```

In [53]:
```
#Now plotting the pie charts to see where the most revenue is coming from
def summary_pie():

    fig, ax = plt.subplots(1,2, figsize=(24,6))


    # ax[0].bar(revenue_by_agegroup_list,Age_Group)
    # ax[1].bar(revenue_by_agegroup_list,Age_Group)
    # ax[0].set_title("Total Spending on Gold Products")
    explode = [0.05,0,0,0]
    ax[0].pie(revenue_by_agegroup_list,labels =Age_Group,autopct="%1.1f%%
    ax[0].set_title("Total Spending by Each Age Group")
    ax[0].text(1.5,1, "Youngsters = Up to 34\nMiddle Aged = 35-50\nOld =
    ax[1].pie(revenue,labels =revenue_list, autopct="%1.1f%%", startangle
    ax[1].set_title("Total Spending by Product")
    # ax[1].text(1.5,1, "Youngsters = Up to 34\nMiddle Aged = 35-50\nElde
    return
```

Project Outcome

Generally speaking, it is common for one to make generalised conclusions for different matters that have most and least probable variables, like a cricket match between Team A (known to be ranked number 1) and Team B (a team which is ranked last). In this example, one would tend to believe that Team A will almost definitely win the match. Similarly, before analysing this data it was assumed that Meat products are the most significant product group for any market because it is widely used for daily meals and cooking. Also, it was thought that most spending will be done by younger customers, and that customers would prefer online shopping rather than going in person.

However, the results were not according to our expectations or assumptions, which highlights a key idea in data science that you should not make any assumptions prior to conducting an investigation as this can affect one's methodology and thereby the validity of one's results. It has been determined that people of all ages and income ranges in the sample prefer to buy wine more than any other product, contrary to initial assumptions. The findings of each objective are outlined below separately.

Overview of Results

As a general overview, one product stood out irrespective of age group, income group, number of offspring and relationship status, which was wine, which along with meat made up a significant proprtion of purchaes. Being in a relationship seemed when taking averages to have no effect on product purchasing, whilst having offspring seemed to increase average spending. Surprisingly, no relationship was found between shopping platform preference and relationship status. Different products stood out for different age groups when average(normalised) spending was compared with absolute or total spending.

Objective 1

Objective: *The effect of people's income on the products they buy*

Explanation of Results

This objective was set to see the impact of revenue on the types of products purchased. As you can see from the bar and line chart below, it has been observed that wine is the most purchased product, except for the group with the lowest income level. Even the lowest income group actually has very little difference between meat, the most popular product, and wines. Based on this, we can safely say that wine products are the most popular items in this customer base region, likely due to the nature of this product store being inclined to wine sales. When looking at the line chart, the priority of people from every income group in choosing a product is almost exactly the same. While the most preferred product group was wines except for the lowest income group, the least preferred group was fruits and sweet products, meaning that marketing for these products needs to be refocussed.

Based on these graphs, it can be said that each income group shopping at this market has almost the same product selection priority and the difference is only the number of purchases. It can be seen that the income group 60001-90000 purchases the greatest number of products, followed by the 30001-60000 income group, which also makes up a large proportion. As income increases, the amount of products purchased also increases, not counting the highest income group. We can attribute the decrease in purchasing in the highest income group to many reasons. For instance, people with a high income level may have stopped shopping themselves, or they may prefer to go to another vendor instead of this one.

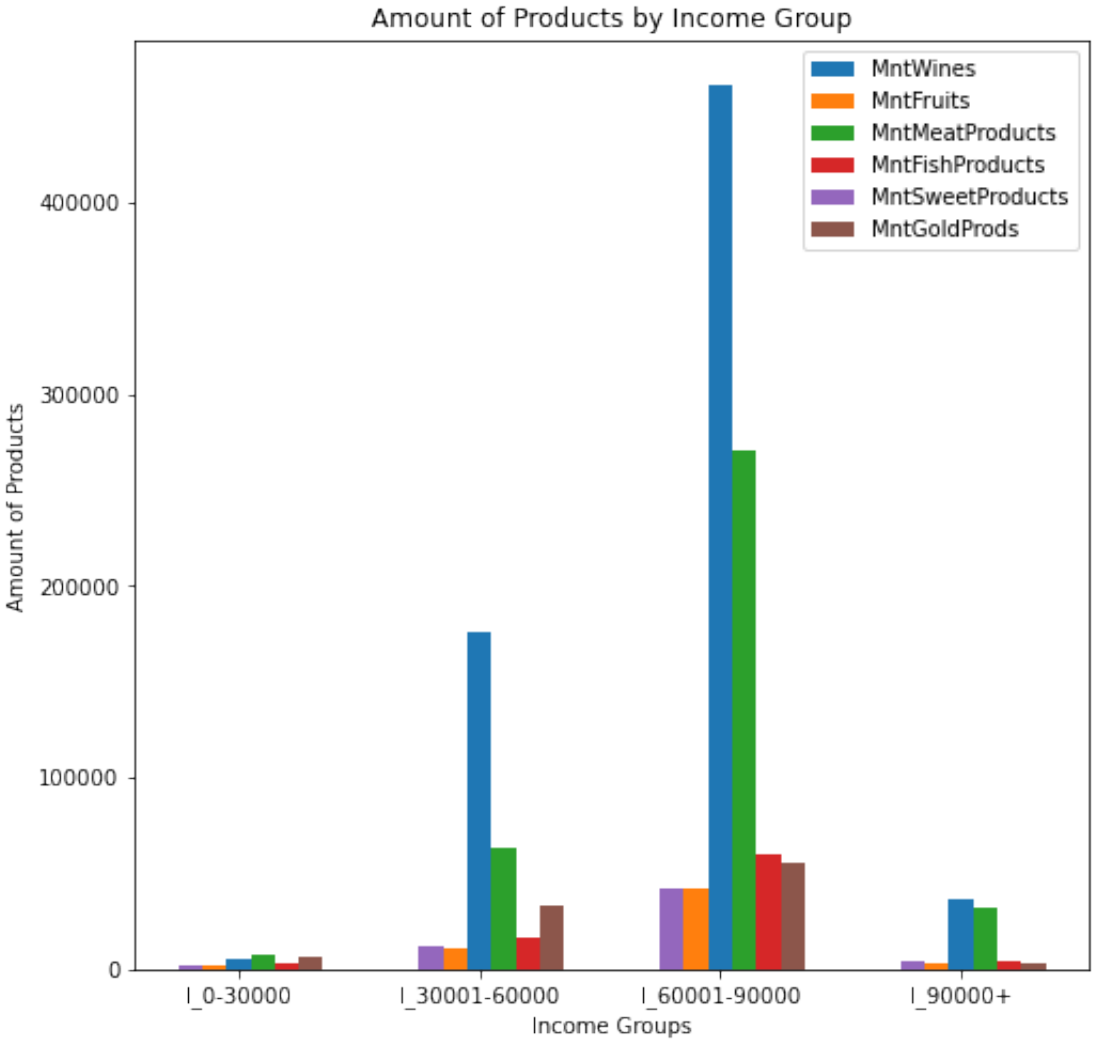
Income_Group	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProd
I_0-30000	5099	2126	7941	2964	2290
I_30001-60000	176282	10978	63360	16663	11723
I_60001-90000	461523	42621	270157	59755	42596
I_90000+	37125	3028	31917	4549	3943

As you can see from the table above, the number of products purchased is by far the Wine group, except for the lowest income level. When we look at the group whose Income range is 30001-60000, wines products purchase is almost 3 times higher than meat purchase. Also, this group has the highest ratio between purchasing wine and meat products.

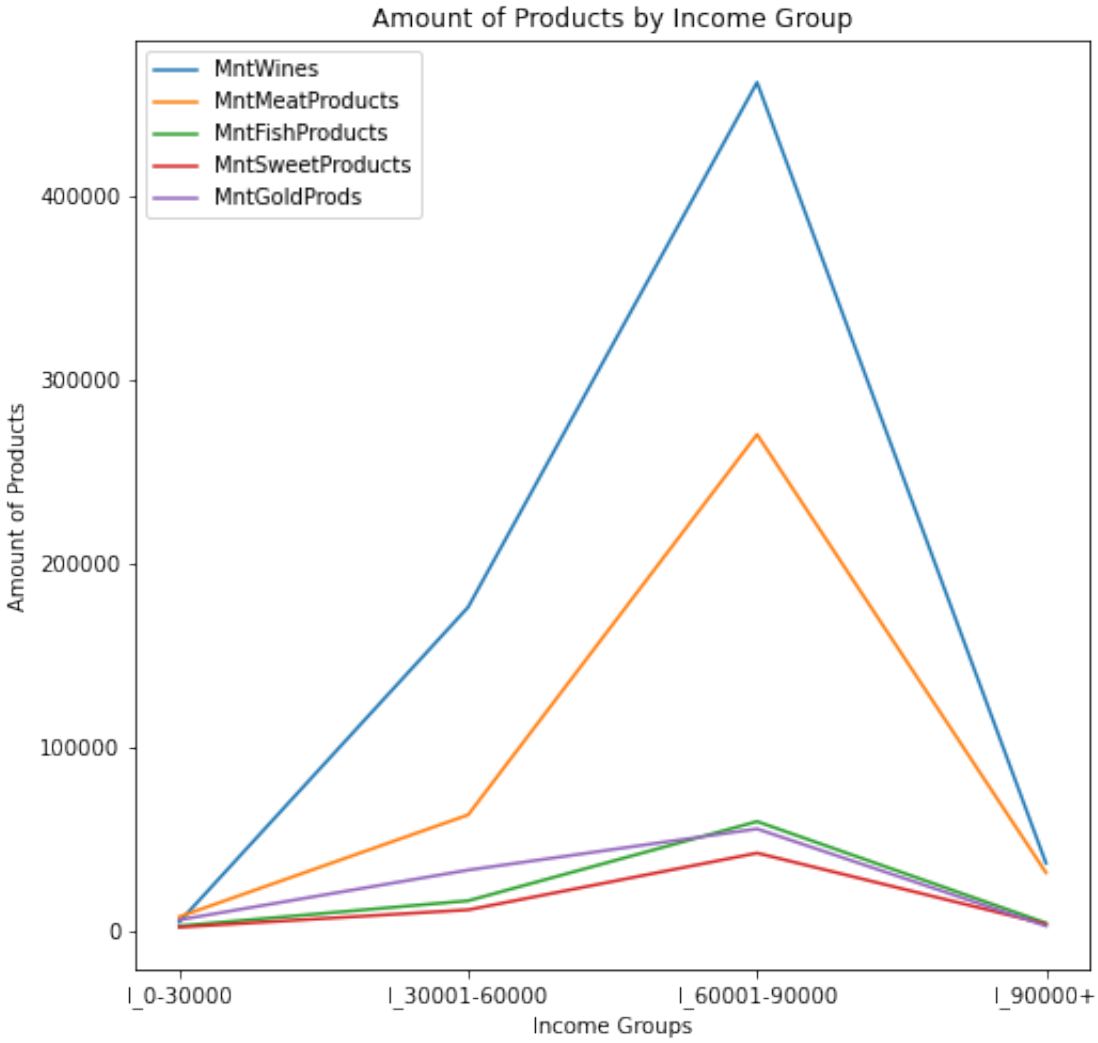
Visualisation

In [54]:
```
objective1_Graph()
```

```
<Figure size 432x288 with 0 Axes>
```




```
<Figure size 432x288 with 0 Axes>
```



The above bar chart illustrates information about how many products from which product group people buy according to their income ranges. Also, the line graph shows the change in the total amount sold for each product group according to the income ranges.

Objective 2

Objective: *The effect of people's relationship status and having children on the products they buy*

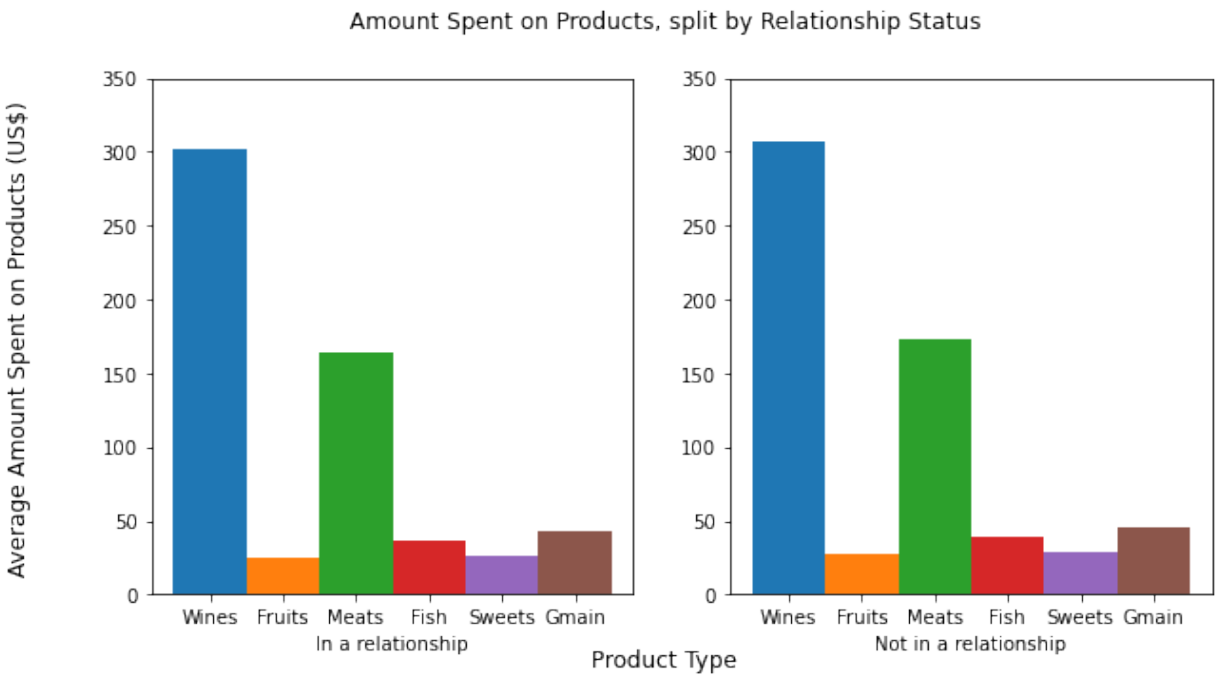
Explanation of Results

From these plots it appears that being in a relationship or not has no bearing on the amount that is spent on each product by customers on average, with very similar proportions across the board, which match those seen in income analysis where wine and meat were the most popular. Around $300 is spent on average by customers on wine and ~160-170 dollars on meat products, for both groups, making up around 50% and 27% of spending on average respectively. Further clustering analysis is needed to determine if other factors alongside relationship status have a bearing on product purchasing.

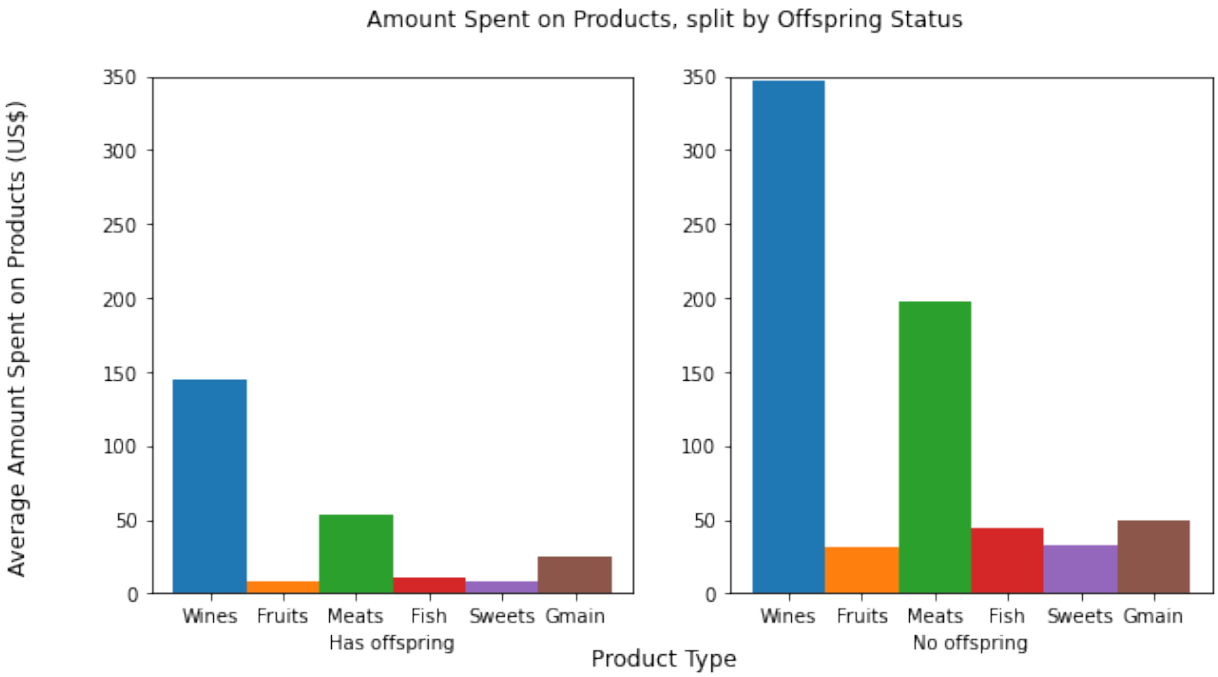
In these plots, we can see that there is a similar distribution of spending on products when having offspring or not, with wine and meat still the most popular categories. However, the distribution is much flatter when having offspring, and there is a clear reduction in the average amount spent on products even if the proportion of spending on products is similar. There is around 200 dollars less spent on wine and 150 dollars less spent on meat, although it must be noted that the no offspring group (1766 customers) is much larger than the offspring group (474 customers), which affects the results of this analysis. It could be assumed that having offspring leads to more purchases due to having to accomodate for more people cohabiting in a household, although further testing is needed due to the lack of data.

Visualisation

In [55]:
```
objective2_relationship_Graph()
```



In [56]:
```
objective2_offspring_Graph()
```



Objective 3

Objective: *Which platform is most used by customers, in-store or online, and the impact of marital status on this choice*

Explanation of Results

For any company the most used method that their customers use to purchase items is what the company must focus its efforts on order to accomplish its best sales and success, leading to this analysis in order to try and find the most popular platform.

As the first bar chart shows, the most platform used by customers is in-store purchases. 12970 purchases have been made in-store while 9150 purchases weremade through the company website. As a result of this analysis, the company must focus on in-store sales by providing more products in stock and hiring more employees to serve those customers. However, it must be kept in mind that the store will cost the company more money due to operation costs (electricity, man power and maintenance).

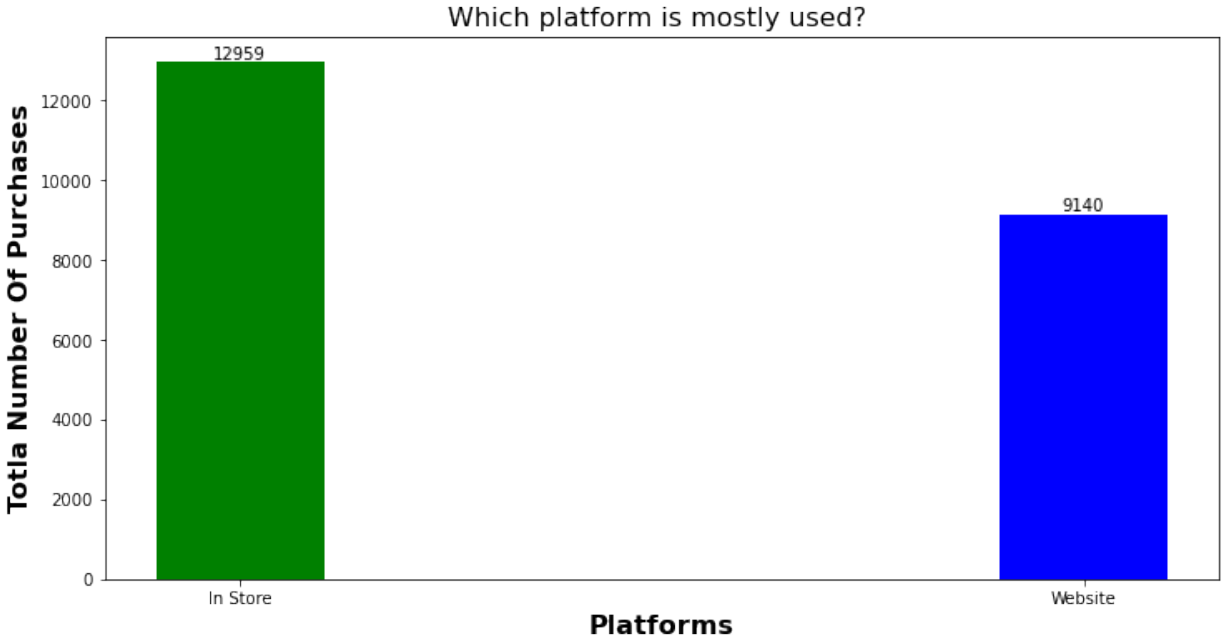
The second bar chart shows that in-store purchases are more popular than online purchases whether in a relationship or not. For both groups, in-store purchases are the most popular platform. Furthermore, the number of purchases from in-relationship customers is greater than the customers who are not in a relationship for both platforms. Therefore, the company can focus on this category of customers by targeting them with offers and focussed marketing because these are loyal customers and the largest group of customers that purchase from the company.

It is important to keep in mind that the relationship group (1444 customers) is almost twice as big as the non-relationship group (796 customers), meaning an exact comparison is not entirely possible when looking at the total amounts of spending.

Visualisation

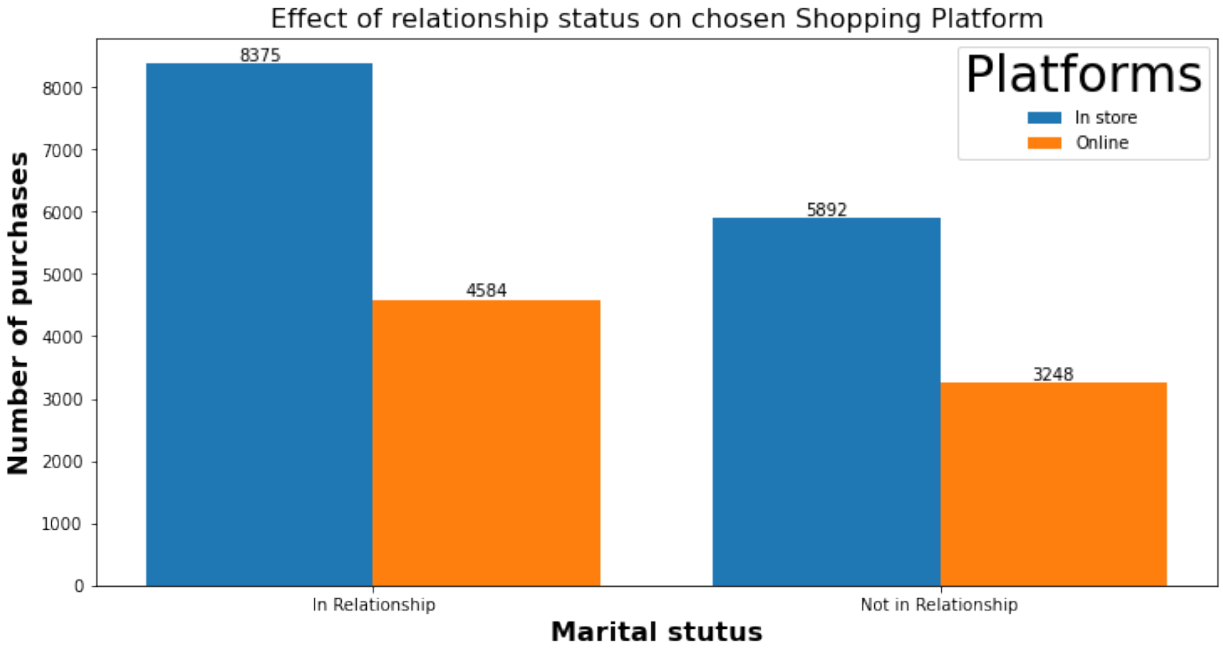
The below bar chart shows which platform is mostly used by customers is it in store purchases or purchases from the company website.

In [57]:
```
objective3_platform_Graph()
```



The below bar chart shows the impact of marital status.

In [58]:
```
objective3_marital_Graph()
```



Objective 4

Objective: *The effect of people's age on the products they buy*

Explanation of Results and Visualisation

It is always good to find out customers' spending trends, i.e. who is more prone to spending and on what products a given customer will spend. This can enable us to set priorities for products. For instance, we can never run out of product A since the product is being bought the most OR we should target our campaign for Product A on a certain age group to ensure maximum success. Knowing these insights will help us maximise the sale and ensure good revenues. Keeping this in mind, this objective will analyse two things

Which Product is being bought the most which may in turn enable us to for instance;

1. *Set priorities for different aspects like stock level, campaign priorities etc.?*
2. *Which age group is mostly doing the spending, and therefore what future targets can be set to meet different objectives?*

To find the effect of age on buying of products it is decided to be necessary to find out

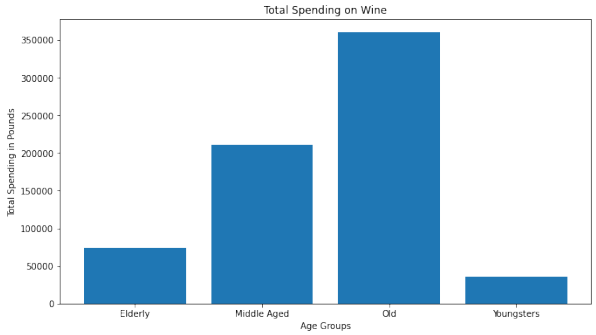
1. *Total and average spending by each group on different products*
2. *The Age group contributing the most to the spending, which in turn is a source of revenue for the store*

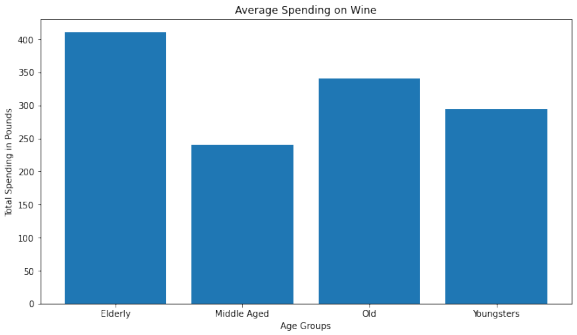
Total amount spent by each age group on each product and total number of purchasing by each age group are displayed in following table.

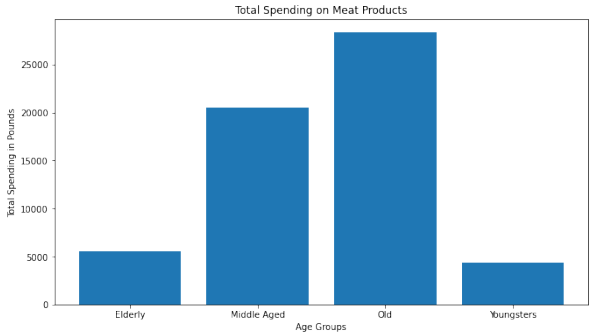
Age_Group	Number of Purchasing	MntWines	MntFruits	MntMeatProducts	MntFishProducts	Mn
Elderly	1058	360030	283338	175976	40097	287
Middle_Aged	879	211259	20499	130210	29608	219
Old	179	73442	5577	38112	8634	578
Youngsters	120	35298	4339	29077	5592	400

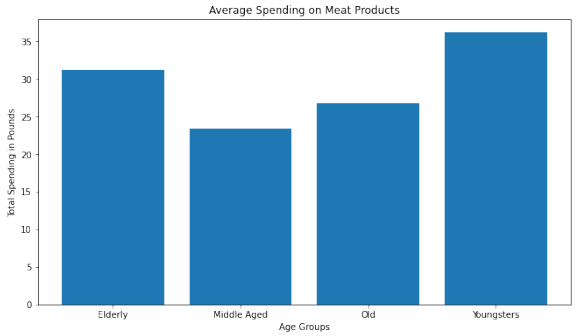
This table is evident that most of the shopping is done by old people since the number of purchasing came out to be the highest for this group. In other words, for a particular household, most of the shopping/groceries will be done by this group. This actually makes sense because they are the ones who are usually responsible for making important different decisions at homes or beingg the primary income earners. Subsequently, analysis will be made using the total and average plots.

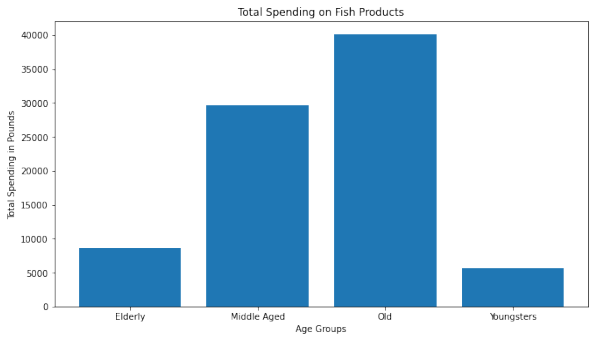
In [61]:
```
spending_wine()
spending_meat()
spending_fish()
spending_fruit()
spending_sweet()
spending_gold()
```

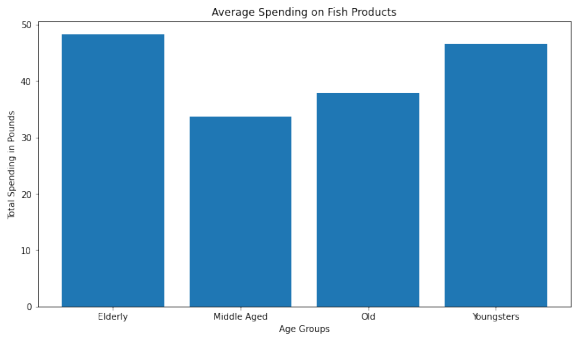


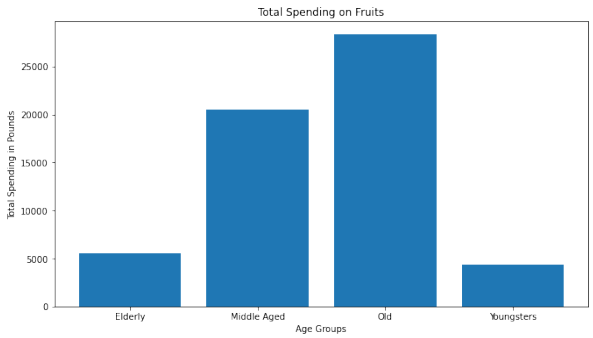


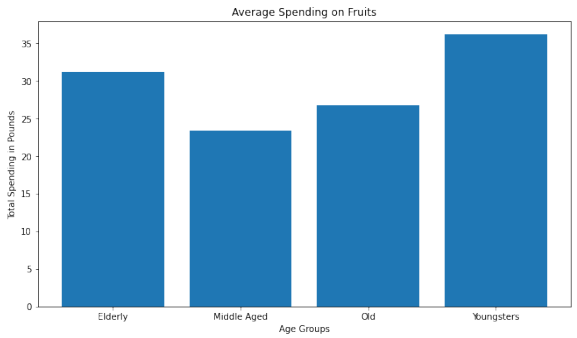


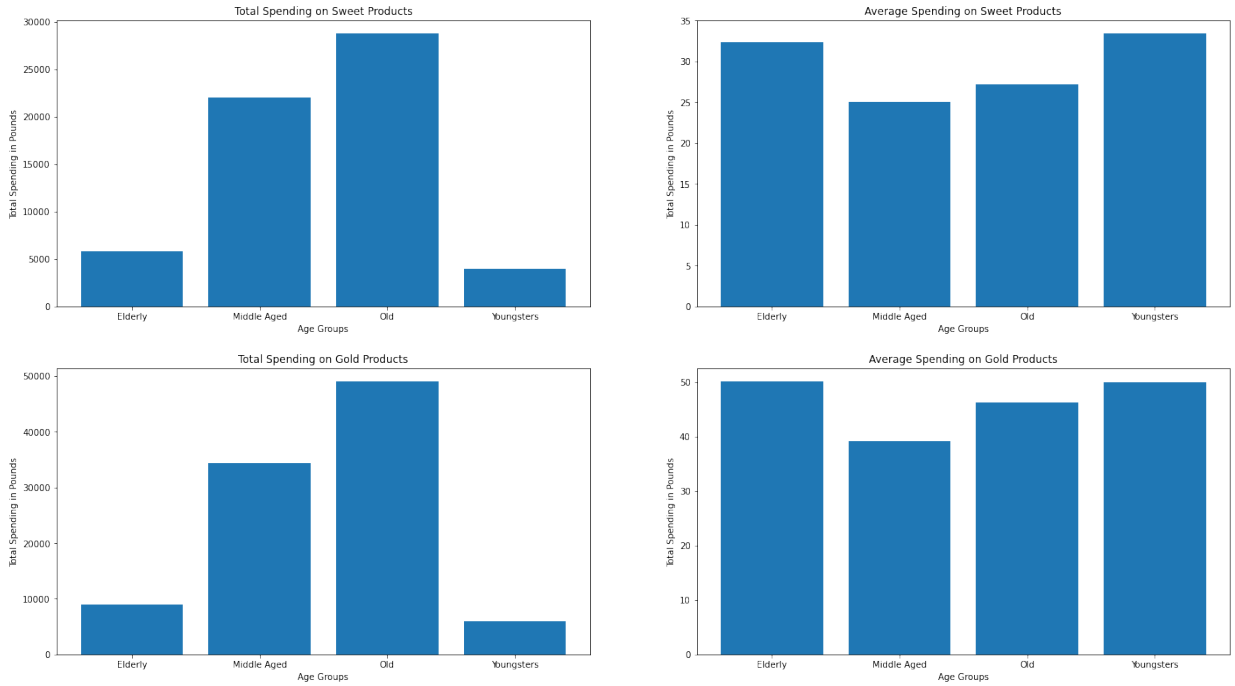












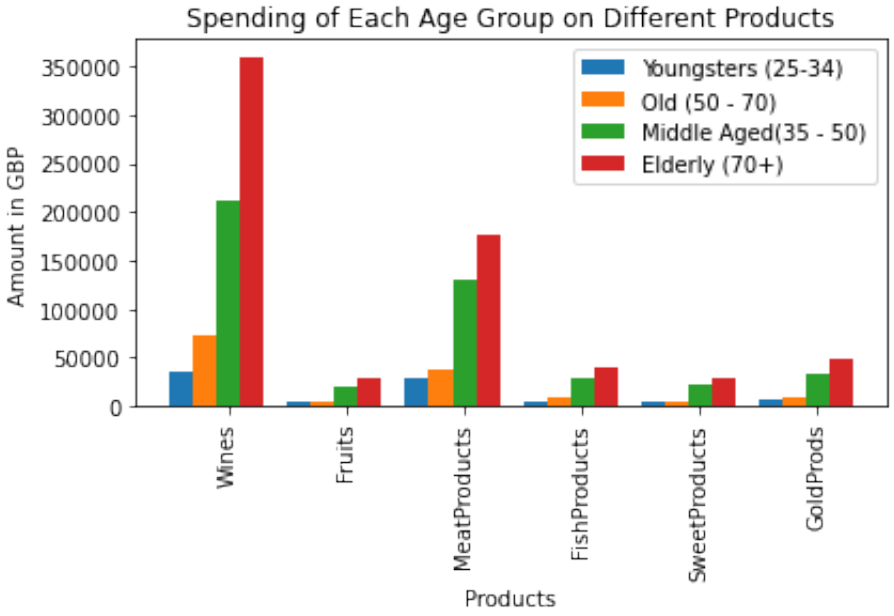
Since Old People (50 - 70 years old) are the ones that are shopping mostly, the total spending for all products remained greatest for this group. However, while looking at the average buying trends there were different results for different products which are

1. *Wines by Elderly*
2. *Fruit by Youngsters*
3. *Meat by Youngsters*
4. *Fish by Elderly*
5. *Sweets by Youngsters*
6. *Gold by Youngster and Elderly*

In other words if our subject fall under the age group of Old, then it is possible that they are is most likely to buy Wine Products, Fish Products or Gold Products, whereas a person in the Youngsters age group are more likely to buy Fruits and Sweet products. This is not surprising as Old People are mostly income earners for the family, so their spending must be balanced in all areas.

Lets try to visualise the data from a different angle. we are now going to focus on total spending on each producet rather than average spending on each product to find out where the most of the revenue is coming from.

In [62]:
```
summary_bar()
```

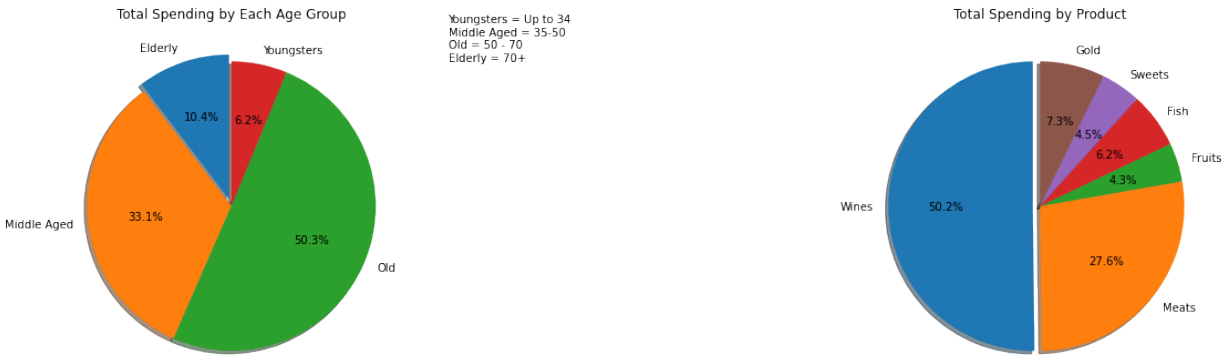


The above plot is a high level summary of spending trends and following points are deduced

1. *Most of the spending by all age groups is done on wine, which aligns with prior results*
2. *On all the products, most of the spending is done by old people*

In other words in terms of age group old people are doing the shopping most frequently and in terms of products, wines are being bought most frequently. This is a qualitative analysis. Following plots will quantify these findings helping us to understand how much each product and age group is contributing towards generating revenue.

In [63]:
```
summary_pie()
```



Based on the analysis and figures the summary of the results is

1. *Old People are doing the shopping most of the time*
2. *Wines are being bought most frequently by all age groups*
3. *Half of the revenue is coming from old people whereas approximately 1/4 is coming from the middle-aged group, making these the two most significant age groups*
4. *Half of the revenue is generated by wines whereas approximately 1/4 is generated by meat. This makes meat and wine the critical stock and the store should take measures to ensure that it never runs out of these products*

For the sake of targeted marketing

1. *Marketing for meat, sweet and fruit products should be targeted to youngsters (25-34 years old).*
2. *Marketing for wine is tricky because on average it is bought most by the elderly, however it is also the most bought product by all age groups.*
3. *Marketing for Gold should be targeted to the elderly and youngsters.*

Conclusion

Achievements

From the findings we obtained as a result of this study, it was determined that the shopping tendencies of people in this sample were inclined to wine products instead of meat and fish products, which we can define as important day-to-day foods as opposed to wine which is a luxury product. It can be predicted that the marketing of wine products in the customer base region will result in success, because people here are very prone to spend their income on this product group. Also, defining product groups by income level would also be meaningless, because, as can be seen from the analysis, all income levels prioritized product groups in the same way, except for people with the lowest income, who were only in the minority. Relationship status seems to have no impact on its own on purchasing trends, whilst having offspring tends to increase spending, so perhaps marketing should be adjusted to reflect this. In addition, despite the developing technology and widespread online shopping trends, the people in this sample have a higher tendency to shop by going to the store. Therefore, it is predicted that investments to be made in the field of online shopping may fail.

Limitations

One of the biggest limitations in this project was the small size of the dataset. Working with a larger data set to see the overall distribution will provide more accurate inferences. In addition, the absence of gender data affected the elaboration of the studies, as this parameter in conjunction with age, income and marital status would give a stronger picture of each individual customer. Furthermore, the scope of this study did not allow for a more complex personality analysis through the use of k-means clustering or similar methods to characterise groups of customers with similar buying patterns. The lack of distinct clustering analysis lead to some objectives not yielding useful results, as a simple customer personality analysis comparing only a few felds rather than many in conjunction does not allow for a deeper segmentation of customer groups.

Future Work

Future work may include conducting k-means clustering to segment the customer base into groups for which marketing and product design can be targeted. This could be broken down into separate analyses for each product type in order to determine their individual customer segmentations and incorporate into marketing accordingly. Moreover, it would be useful collect a large and diverse data to reach a more precise result and infer any stronger correlations between the parameters. Several new objectives could be defined in order to ask more specific questions about individual products and marketing strategies, using these deeper profiles of customers to link objectives and optimise methods of marketing products to customers.

In [ ]: