Report IDS

Students ID:

Khaled Saleh : 2232890

Mohanad Assaf : 2233114

Doctor :

Zaher Ibrahim Saleh Salah.

Data set used :

All.

Date:

5/17/2024

Import Packages

```
[1]: import pandas as pd
     import numpy as np

     import matplotlib.pyplot as plt
     import seaborn as sns

     from docx import Document # conda install conda-forge::python-docx

     from sklearn.preprocessing import LabelEncoder
     from sklearn.preprocessing import MinMaxScaler
     from sklearn.preprocessing import StandardScaler
```

Import several libraries used for data analysis (Pandas and Numpy), data visualization (matplotlib and seaborn), document creation (docx), and data preprocessing (LabelEncoder, MinMaxScalar, and StandardScalar).

```
[2]: df1 = pd.read_csv('Scoring-Dataset-1.csv')
     df2 = pd.read_csv('Scoring-Dataset-2.csv')
     df3 = pd.read_csv('Scoring-Dataset-3.csv')
     df4 = pd.read_csv('Scoring-Dataset-4.csv')
     df5 = pd.read_csv('Scoring-Dataset-5.csv')
     df6 = pd.read_csv('Scoring-Dataset-6.csv')
     df7 = pd.read_csv('Scoring-Dataset-7.csv')
     df8 = pd.read_csv('Scoring-Dataset-8.csv')
     df9 = pd.read_csv('Scoring-Dataset-9.csv')
     df10 = pd.read_csv('Scoring-Dataset-10.csv')
     df11 = pd.read_csv('Scoring-Dataset-11.csv')
     df12 = pd.read_csv('Scoring-Dataset-12.csv')
     df13 = pd.read_csv('Scoring-Dataset-13.csv')
     df14 = pd.read_csv('Scoring-Dataset-14.csv')
     df15 = pd.read_csv('Scoring-Dataset-15.csv')
     df16 = pd.read_csv('Scoring-Dataset-16.csv')
     df17 = pd.read_csv('Scoring-Dataset-17.csv')
     df18 = pd.read_csv('Scoring-Dataset-18.csv')
     df19 = pd.read_csv('Scoring-Dataset-19.csv')
     df20 = pd.read_csv('Scoring-Dataset-20.csv')
     df21 = pd.read_csv('Scoring-Dataset-21.csv')
     df22 = pd.read_csv('Scoring-Dataset-22.csv')
     df23 = pd.read_csv('Scoring-Dataset-23.csv')
     df24 = pd.read_csv('Scoring-Dataset-24.csv')
     df25 = pd.read_csv('Scoring-Dataset-25.csv')
     df26 = pd.read_csv('Scoring-Dataset-26.csv')
     df27 = pd.read_csv('Scoring-Dataset-27.csv')
     df28 = pd.read_csv('Scoring-Dataset-28.csv')
     df29 = pd.read_csv('Scoring-Dataset-29.csv')
     df30 = pd.read_csv('Scoring-Dataset-30.csv')

     # concat the whole datasets in one dataset because it is easier to deal with :)
     df = pd.concat([df1,df2,df3,df4,df5,df6,df7,df8,df9,df10,df11,df12,df13,df14,df15,df16,
             df17,df18,df19,df20,df21,df22,df23,df24,df25,df26,df27,df28,df29,df30], axis=0)
```

We read all the provided CSV files and merge them into a single data frame. This allows for easier data manipulation and function application.

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

<class 'pandas.core.frame.DataFrame'>
Index: 9000 entries, 0 to 299
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   User_ID                  9000 non-null   int64
 1   Gender                   9000 non-null   object
 2   Age                      9000 non-null   int64
 3   Marital_Status           9000 non-null   object
 4   Website_Activity         9000 non-null   object
 5   Browsed_Electronics_12Mo 9000 non-null   object
 6   Bought_Electronics_12Mo  9000 non-null   object
 7   Bought_Digital_Media_18Mo 9000 non-null  object
 8   Bought_Digital_Books     9000 non-null   object
 9   Payment_Method           9000 non-null   object
dtypes: int64(2), object(8)
memory usage: 773.4+ KB
```

```
[5]: df.isnull().sum()

[5]: User_ID                     0
     Gender                      0
     Age                         0
     Marital_Status              0
     Website_Activity            0
     Browsed_Electronics_12Mo    0
     Bought_Electronics_12Mo     0
     Bought_Digital_Media_18Mo   0
     Bought_Digital_Books        0
     Payment_Method              0
     dtype: int64
```

We took a general overview of the data, examined the datatypes of each column to know what methods should be followed to deal with this column, and then checked for the presence of any null values.

A1. Identify the type of each attribute (nominal, ordinal, interval or ratio):

- nominal: User_ID ,Gender, Marital_Status, Browsed_Electronics_12Mo, Bought_Electronics_12Mo, Bought_Digital_Media_18Mo, Bought_Digital_Books, Payment_Method
- ordinal: Website_Activity
- interval: None
- ratio: Age

```
[6]: doc = Document()
doc.add_heading('Data Types', 0)

doc.add_heading('Nominal:', level=1)
doc.add_paragraph().add_run('User_ID, Gender, Marital_Status,Browsed_Electronics_12Mo,\
Bought_Electronics_12Mo, Bought_Digital_Media_18Mo,Bought_Digital_Books,\
Payment_Method').bold = True

doc.add_heading('Ordinal:', level=1)
doc.add_paragraph().add_run('Website_Activity').bold = True

doc.add_heading('Interval:', level=1)
doc.add_paragraph().add_run('None').bold = True

doc.add_heading('Ratio:', level=1)
doc.add_paragraph().add_run('Age').bold = True

doc.save('A1.docx')
```

A1: We completed the task as instructed in the assignment, classifying the type of each feature, and then stored the results in a file named A1.docx.

`Website_Activity` is an ordinal datatype because there is a specific order to the data. 'Seldom' indicates less activity than 'regular', and 'regular' indicates less activity than 'frequent'.

`Age` is a ratio datatype because it has a true zero point. A value of zero represents the absence of age. And there are no negative ages in the world.

The rest of the features are nominal because they have no order and are not numeric, with the exception of `User_ID`. Although `User_ID` is numeric, it is also considered nominal because no calculations can be performed on it.

A2:

```
Attribute: Gender
Gender
M    4782
F    4218
Name: count, dtype: int64
--------------------------------
Attribute: Marital_Status
Marital_Status
M    4615
S    4385
Name: count, dtype: int64
--------------------------------
Attribute: Website_Activity
Website_Activity
Seldom      5434
Regular     2845
Frequent     721
Name: count, dtype: int64
```

```
Attribute: Browsed_Electronics_12Mo
Browsed_Electronics_12Mo
Yes    8600
No      400
Name: count, dtype: int64
--------------------------------
Attribute: Bought_Electronics_12Mo
Bought_Electronics_12Mo
No     4731
Yes    4269
Name: count, dtype: int64
--------------------------------
Attribute: Bought_Digital_Media_18Mo
Bought_Digital_Media_18Mo
Yes    7191
No     1809
Name: count, dtype: int64
```

```
Attribute: Bought_Digital_Books
Bought_Digital_Books
No     5149
Yes    3851
Name: count, dtype: int64
--------------------------------
Attribute: Payment_Method
Payment_Method
'Website Account'    3781
'Bank Transfer'      2902
'Credit Card'        1184
'Monthly Billing'    1133
Name: count, dtype: int64
```

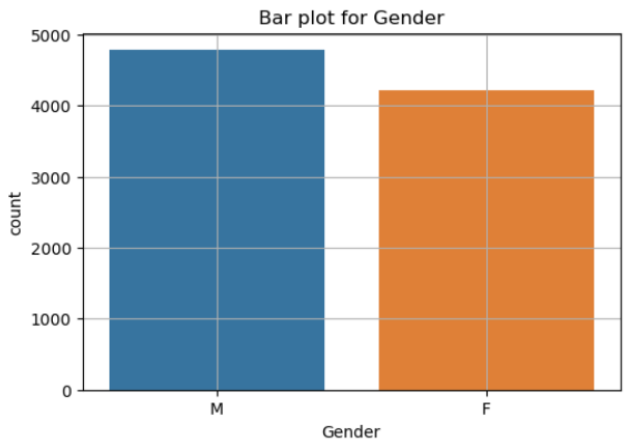
We identified the frequencies of values for object datatypes. For the numeric datatype Age (we will not perform any calculations on User_ID because it is merely an ID and it doesn't make any sense to perform calculations on it),

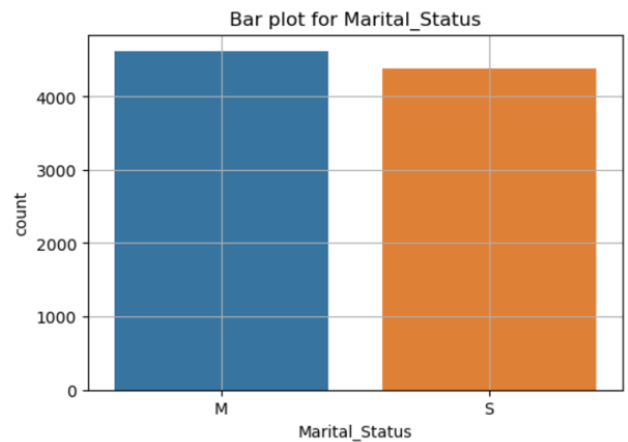
```
# 2. Location measures
print(f"Mean of Age: {df['Age'].mean()}")
print(f"Median of Age: {df['Age'].median()}")

# 3. Spread measures
print(f"Variance of Age: {df['Age'].var()}")
print(f"Standard deviation of Age: {df['Age'].std()}")
print(f"Range of Age: {df['Age'].min()} - {df['Age'].max()}")
print(f"(Q1) 25th Percentile of Age: {df['Age'].quantile(0.25)}")
print(f"(Q2) 50th Percentile of Age: {df['Age'].quantile(0.50)}")
print(f"(Q3) 75th Percentile of Age: {df['Age'].quantile(0.75)}")
print(f"IQR of Age: {df['Age'].quantile(0.75) - df['Age'].quantile(0.25)}")
#******************************************************************************
#******************************************************************************
# 2. Location measures
doc.add_heading('Location measures', 0)
doc.add_paragraph(f"Mean of Age: {df['Age'].mean()}")
doc.add_paragraph(f"Median of Age: {df['Age'].median()}")

# 3. Spread measures
doc.add_heading('Spread measures', 0)
doc.add_paragraph(f"Variance of Age: {df['Age'].var()}")
doc.add_paragraph(f"Standard deviation of Age: {df['Age'].std()}")
doc.add_paragraph(f"Range of Age: {df['Age'].min()} - {df['Age'].max()}")
doc.add_paragraph(f"(Q1) 25th Percentile of Age: {df['Age'].quantile(0.25)}")
doc.add_paragraph(f"(Q2) 50th Percentile of Age: {df['Age'].quantile(0.50)}")
doc.add_paragraph(f"(Q3) 75th Percentile of Age: {df['Age'].quantile(0.75)}")
doc.add_paragraph(f"IQR of Age: {df['Age'].quantile(0.75) - df['Age'].quantile(0.25)}")

doc.save("A2.docx")
```

```
Mean of Age: 45.894
Median of Age: 47.0
Variance of Age: 178.04654694966808
Standard deviation of Age: 13.343408370789978
Range of Age: 17 - 70
(Q1) 25th Percentile of Age: 35.0
(Q2) 50th Percentile of Age: 47.0
(Q3) 75th Percentile of Age: 56.0
IQR of Age: 21.0
```

we calculated location measures (median, mean) and spread measures (variance, standard deviation, range, Q1, Q2, Q3, IQR). After completing these calculations, we stored the results in a docx file named A2.docx.

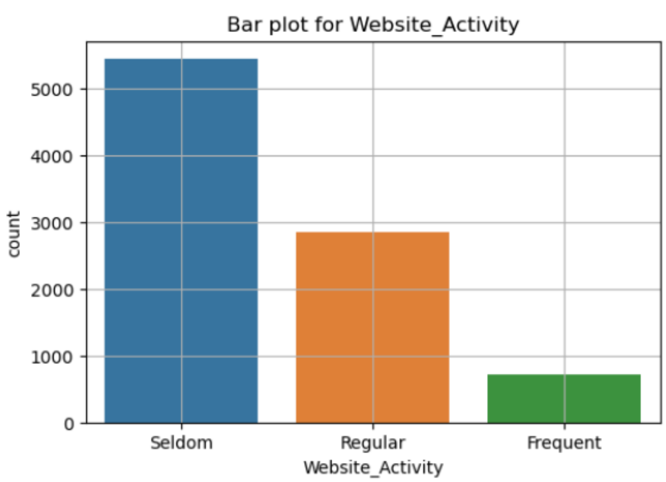
We created bar plots for the object features to visualize the data distribution.

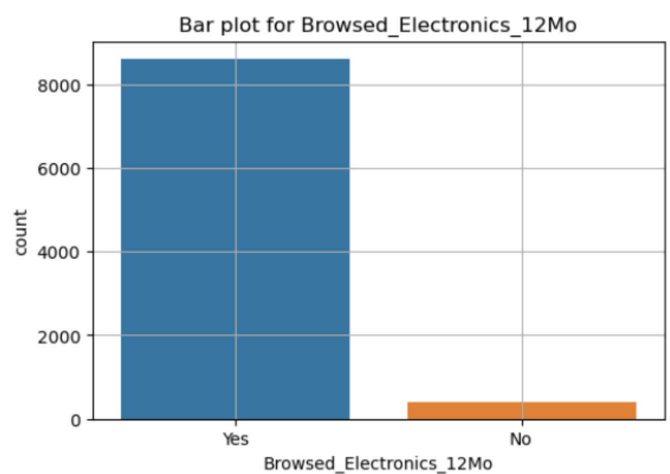




We plot the gender, as we can see that the males are higher than females.

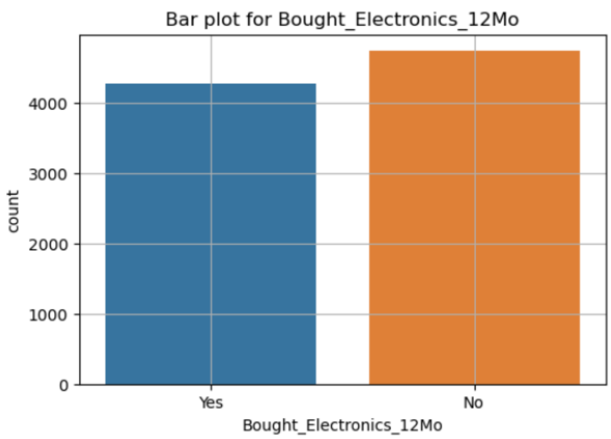
And in the second plot the married people and single people are very high. So, you should be careful when you do digital marketing for each one of them.

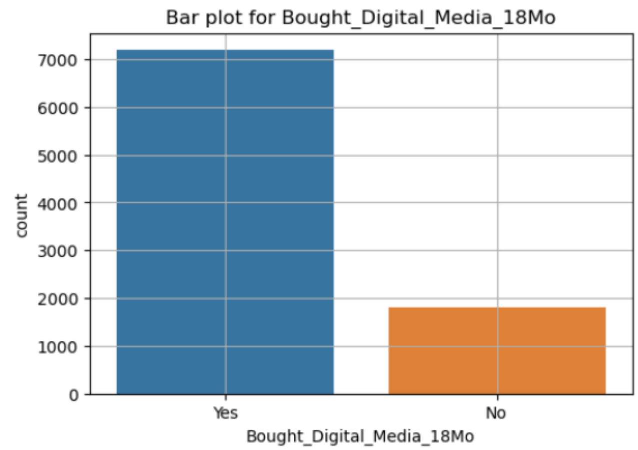




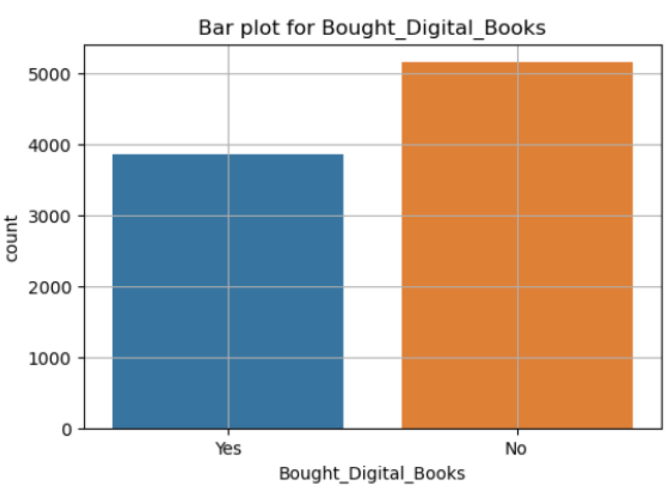
The activity on the website, most of the people are seldom, this is a negative thing. You should do something intereseting to fix that problem.

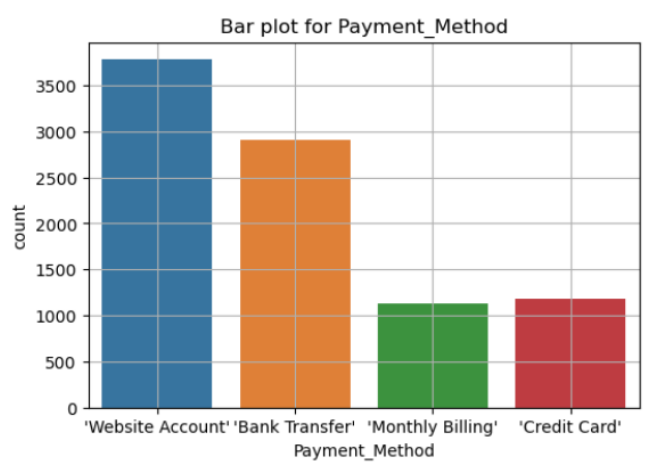
As we can see the browsed electronics most of then are yes, so you should do some good offers for this to make the website activity frequenet or regular visited.





As we can see the bought of the electronics, there are a lot of people did not buy anything of them. So, as I said previously you should do offers on the electronics. In the second plot this is improve for my words before, we say most of the people bought digital thing because it's cheaper, and he don't have to wait for it, immediately will reach the user.





Most of the people on our website don't read books at all, but most of them as we said browse the electronics. Most of the people use the website account and bank transactions to pay for their goods in general.

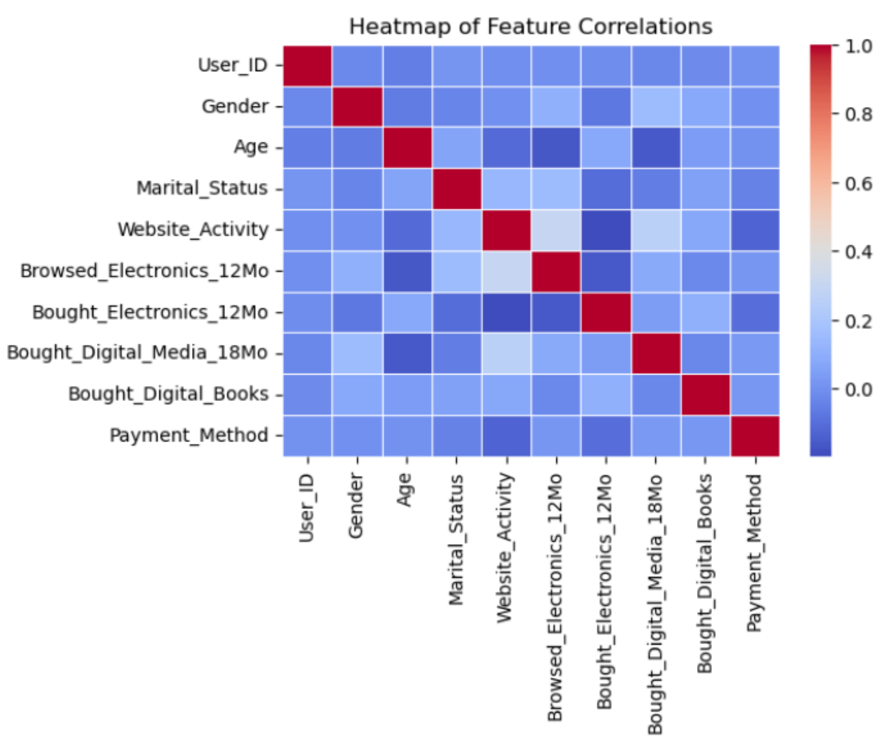
A3:

```python
# Convert to numeric because I want to make a heatmap
le = LabelEncoder()
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = le.fit_transform(df[col])
```

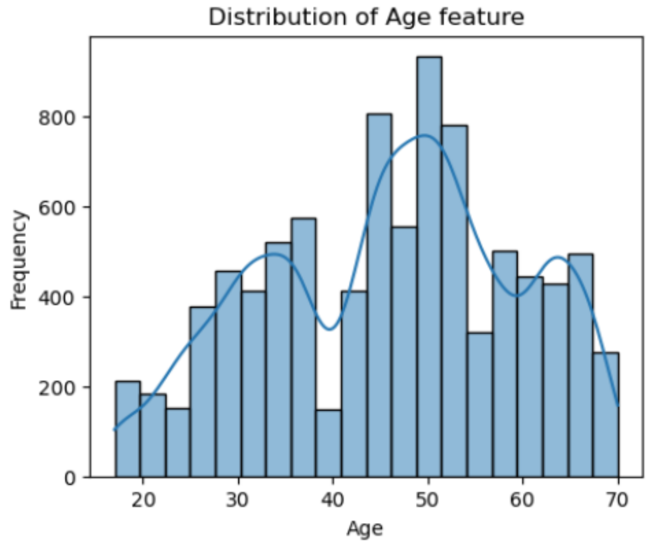
```python
df.dtypes
```

```
User_ID                       int64
Gender                        int32
Age                           int64
Marital_Status                int32
Website_Activity              int32
Browsed_Electronics_12Mo      int32
Bought_Electronics_12Mo       int32
Bought_Digital_Media_18Mo     int32
Bought_Digital_Books          int32
Payment_Method                int32
dtype: object
```

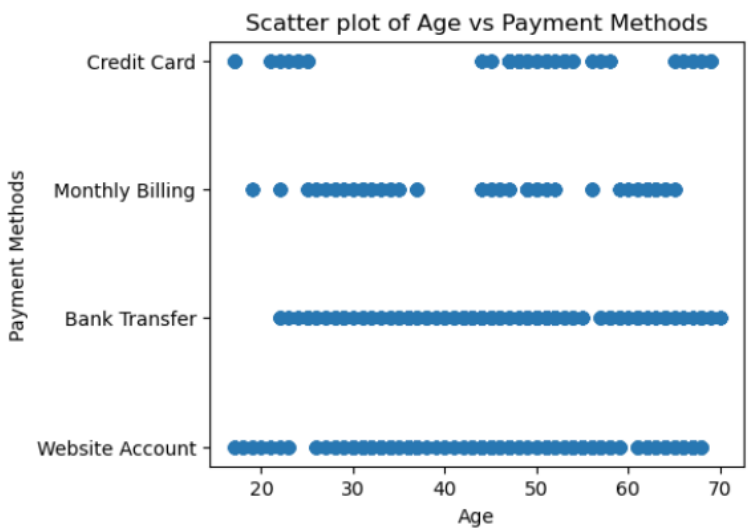
First, I used Label Encoder to convert the nominal features into numeric features, primarily to generate a heatmap and observe any correlations between the features.



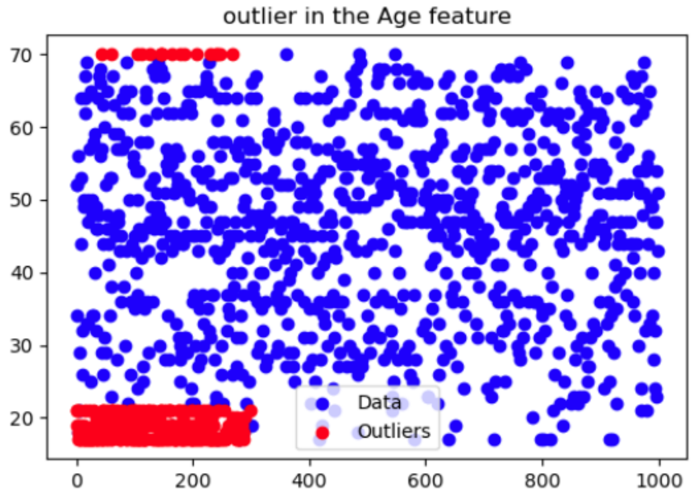
However, as we can see from the heatmap, there are no strong correlations between the features, unfortunately.



I plotted the 'Age' feature to identify the most common life stage among our website users, as our marketing approach differs for each life stage.



We also created a scatter plot comparing ages and payment methods to understand how different age groups prefer to pay. We plotted this before but this another improves to the results.



We used scatter plots and box plots to identify outliers. The outliers the ages that higher than 70 and the ages that approximately lower than 20.

B1: We used the preprocessing technique to minimize the range of the `Age` values (normalize) to make the data balance and easy to understand.

B2: *We categorize the Age features into 5 features as you will see in the csv file.*Teenager 1-16, Young 17-35, Mid_Age 36-55, Mature 56-70, Old 70+

B3: We did this step in the Label Encoder step. 😁

All of these steps stored in an csv file.