

Customer Analysis

Problem Statement: A well known company with numerous products needs to analyze their customer behaviour and classify them to know whether they will accept the campaigns held by the company.

In this project I will be doing an unsupervised clustering of data on customer records from the company's database. Customer clustering/segmentation is the practice of separating customers into groups that reflect similarities among customers in each cluster. It helps to modify products according to distinct needs and behaviours of the customers.

Importing libraries

```
In [2]: import numpy as np
import pandas as pd
import datetime
import matplotlib
import matplotlib.pyplot as plt
from matplotlib import colors
import seaborn as sns
```

```
In [3]: from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from sklearn import metrics
from sklearn.metrics import silhouette_score
```

Loading data

```
In [4]: data = pd.read_csv("market_train.csv")
data.head()
```

```
Out[4]:
```

| | Unnamed: 0.1 | Unnamed: 0 | ID | Year_Birth | Education | Marital_Status | Income | Kidhome |
|---|--------------|------------|------|------------|-----------|----------------|------------|---------|
| 0 | 0 | 0 | 5524 | 1957 | S1 | Lajang | 58138000.0 | 0 |
| 1 | 1 | 1 | 2174 | 1954 | S1 | Lajang | 46344000.0 | 1 |
| 2 | 2 | 2 | 4141 | 1965 | S1 | Bertunangan | 71613000.0 | 0 |
| 3 | 3 | 3 | 6182 | 1984 | S1 | Bertunangan | 26646000.0 | 1 |
| 4 | 4 | 4 | 5324 | 1981 | S3 | Menikah | 58293000.0 | 1 |

5 rows × 31 columns



```
In [5]: test = pd.read_csv("market_test.csv")  
test.shape
```

```
Out[5]: (559, 31)
```

```
In [6]: data.shape
```

```
Out[6]: (1680, 31)
```

The data is provided as two parts - one for training and the other for testing. The columns can be categorized in to 4 types - customer information, Products, Promotion, Places(details about purchase),

Exploratory Analysis and Data Cleaning

In [7]: `# information about the data and features`

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1680 entries, 0 to 1679
Data columns (total 31 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Unnamed: 0.1                          1680 non-null   int64
1   Unnamed: 0                             1680 non-null   int64
2   ID                                      1680 non-null   int64
3   Year_Birth                            1680 non-null   int64
4   Education                             1680 non-null   object
5   Marital_Status                        1680 non-null   object
6   Income                                1663 non-null   float64
7   Kidhome                               1680 non-null   int64
8   Teenhome                              1680 non-null   int64
9   Dt_Customer                           1680 non-null   object
10  Recency                               1680 non-null   int64
11  MntCoke                               1680 non-null   int64
12  MntFruits                             1680 non-null   int64
13  MntMeatProducts                       1680 non-null   int64
14  MntFishProducts                       1680 non-null   int64
15  MntSweetProducts                      1680 non-null   int64
16  MntGoldProds                          1680 non-null   int64
17  NumDealsPurchases                     1680 non-null   int64
18  NumWebPurchases                       1680 non-null   int64
19  NumCatalogPurchases                   1680 non-null   int64
20  NumStorePurchases                     1680 non-null   int64
21  NumWebVisitsMonth                     1680 non-null   int64
22  AcceptedCmp3                          1680 non-null   int64
23  AcceptedCmp4                          1680 non-null   int64
24  AcceptedCmp5                          1680 non-null   int64
25  AcceptedCmp1                          1680 non-null   int64
26  AcceptedCmp2                          1680 non-null   int64
27  Complain                               1680 non-null   int64
28  Z_CostContact                          1680 non-null   int64
29  Z_Revenue                             1680 non-null   int64
30  Response                              1680 non-null   int64
dtypes: float64(1), int64(27), object(3)
memory usage: 407.0+ KB
```

In [8]: `test.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 31 columns):
#   Column                                Non-Null Count  Dtype
---  ---                                -
0   Unnamed: 0.1                          559 non-null    int64
1   Unnamed: 0                            559 non-null    int64
2   ID                                    559 non-null    int64
3   Year_Birth                           559 non-null    int64
4   Education                             559 non-null    object
5   Marital_Status                       559 non-null    object
6   Income                               552 non-null    float64
7   Kidhome                              559 non-null    int64
8   Teenhome                             559 non-null    int64
9   Dt_Customer                          559 non-null    object
10  Recency                              559 non-null    int64
11  MntCoke                              559 non-null    int64
12  MntFruits                            559 non-null    int64
13  MntMeatProducts                      559 non-null    int64
14  MntFishProducts                      559 non-null    int64
15  MntSweetProducts                     559 non-null    int64
16  MntGoldProds                         559 non-null    int64
17  NumDealsPurchases                    559 non-null    int64
18  NumWebPurchases                      559 non-null    int64
19  NumCatalogPurchases                 559 non-null    int64
20  NumStorePurchases                   559 non-null    int64
21  NumWebVisitsMonth                   559 non-null    int64
22  AcceptedCmp3                        559 non-null    int64
23  AcceptedCmp4                        559 non-null    int64
24  AcceptedCmp5                        559 non-null    int64
25  AcceptedCmp1                        559 non-null    int64
26  AcceptedCmp2                        559 non-null    int64
27  Complain                             559 non-null    int64
28  Z_CostContact                        559 non-null    int64
29  Z_Revenue                           559 non-null    int64
30  Response                             559 non-null    int64
dtypes: float64(1), int64(27), object(3)
memory usage: 135.5+ KB
```

There are unwanted coulmsns, missing values, values that are objects. These need to be addressed

In [9]: `# dropping 2 unwanted columns`
`data = data.drop(["Unnamed: 0.1", "Unnamed: 0"], axis='columns')`

In [10]: `test = test.drop(["Unnamed: 0.1", "Unnamed: 0"], axis='columns')`

```
In [11]: # replacing null values
income_mean = data['Income'].mean()
data['Income'].fillna(value=income_mean, inplace = True)
```

```
In [12]: income_mean = test['Income'].mean()
test['Income'].fillna(value=income_mean, inplace = True)
```

```
In [13]: # converting Dt_Customer to datetime
data["Dt_Customer"] = pd.to_datetime(data["Dt_Customer"])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1680 entries, 0 to 1679
Data columns (total 29 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    1680 non-null   int64
1   Year_Birth            1680 non-null   int64
2   Education             1680 non-null   object
3   Marital_Status        1680 non-null   object
4   Income                1680 non-null   float64
5   Kidhome               1680 non-null   int64
6   Teenhome              1680 non-null   int64
7   Dt_Customer           1680 non-null   datetime64[ns]
8   Recency               1680 non-null   int64
9   MntCoke               1680 non-null   int64
10  MntFruits              1680 non-null   int64
11  MntMeatProducts        1680 non-null   int64
12  MntFishProducts        1680 non-null   int64
13  MntSweetProducts       1680 non-null   int64
14  MntGoldDried           1680 non-null   int64
```

```
In [14]: ▶ test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
```

```
g: Parsing '20-11-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-12-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-04-2014' in DD/MM/YYYY format. Provide format or specify in
```



```
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-11-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
```

```
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-02-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-02-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-11-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
```

```
g: Parsing '27-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-03-2014' in DD/MM/YYYY format. Provide format or specify in
```

```
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-12-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
```

```
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-01-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
```

```
g: Parsing '14-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '31-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-11-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-02-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-06-2013' in DD/MM/YYYY format. Provide format or specify in
```



```
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-12-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-12-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '13-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-02-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
```

```
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '29-04-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '29-11-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '30-11-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '17-08-2012' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '27-12-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '22-09-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '26-06-2014' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '18-05-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '19-10-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '25-12-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '27-03-2014' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '26-08-2012' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '18-10-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarning: Parsing '30-10-2013' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '15-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-09-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-05-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-11-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '17-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-02-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
    test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
```

```

g: Parsing '26-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-10-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-04-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '14-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-10-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '27-03-2013' in DD/MM/YYYY format. Provide format or specify in

```

```
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '23-12-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '28-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '29-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '22-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-03-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-02-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-04-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-08-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '30-07-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '20-12-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '19-06-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
```

```

test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '25-06-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '26-09-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '18-05-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '16-03-2013' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '21-08-2012' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])
C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3609367120.py:1: UserWarnin
g: Parsing '24-01-2014' in DD/MM/YYYY format. Provide format or specify in
fer_datetime_format=True for consistent parsing.
test["Dt_Customer"] = pd.to_datetime(test["Dt_Customer"])

```

```

In [15]: ▶ # handling categorical variables
print("Total categories in the feature Marital_Status:\n", data["Marital_St
print("Total categories in the feature Education:\n", data["Education"].val

```

Total categories in the feature Marital_Status:

| | |
|-------------|-----|
| Menikah | 650 |
| Bertunangan | 438 |
| Lajang | 360 |
| Cerai | 177 |
| Janda | 52 |
| Duda | 3 |

Name: Marital_Status, dtype: int64

Total categories in the feature Education:

| | |
|-----|-----|
| S1 | 834 |
| S3 | 373 |
| S2 | 279 |
| D3 | 159 |
| SMA | 35 |

Name: Education, dtype: int64

```
In [16]: # Replacing marital status with english synonyms
data["Marital_Status"] = data['Marital_Status'].replace('Menikah', 'Married')
data["Marital_Status"] = data['Marital_Status'].replace('Bertunangan', 'Enga
data["Marital_Status"] = data['Marital_Status'].replace('Lajang', 'Bachelor')
data["Marital_Status"] = data['Marital_Status'].replace('Ceraai', 'Divorced')

test["Marital_Status"] = test['Marital_Status'].replace('Menikah', 'Married')
test["Marital_Status"] = test['Marital_Status'].replace('Bertunangan', 'Enga
test["Marital_Status"] = test['Marital_Status'].replace('Lajang', 'Bachelor')
test["Marital_Status"] = test['Marital_Status'].replace('Ceraai', 'Divorced')

data["Marital_Status"].value_counts()
```

```
Out[16]: Married      650
Engaged      438
Bachelor     360
Divorced     177
Janda        52
Duda         3
Name: Marital_Status, dtype: int64
```

Feature Engineering

```
In [17]: #Creating some new features for grouping

#Age of customer today
data["Age"] = 2023-data["Year_Birth"]
test["Age"] = 2023-test["Year_Birth"]

#Total spendings on various items
data["Spent"] = data["MntCoke"]+ data["MntFruits"]+ data["MntMeatProducts"]
test["Spent"] = test["MntCoke"]+ test["MntFruits"]+ test["MntMeatProducts"]

#Feature indicating total children living in the household
data["Children"] = data["Kidhome"]+data["Teenhome"]
test["Children"] = test["Kidhome"]+test["Teenhome"]

#Feature pertaining parenthood
data["Is_Parent"] = np.where(data.Children> 0, 1, 0)
test["Is_Parent"] = np.where(test.Children> 0, 1, 0)

#Segmenting education levels in three groups
data["Education"] = data["Education"].replace({"SMA": "Undergraduate", "S1": "Gr
test["Education"] = test["Education"].replace({"SMA": "Undergraduate", "S1": "Gr
```

```
In [18]: #Dropping some of the redundant features
to_drop = ["Z_CostContact", "Z_Revenue", "Year_Birth", "ID"]
data = data.drop(to_drop, axis=1)
test = test.drop(to_drop, axis=1)
```

```
In [19]: # converting income to thousands
data["Income"] = data["Income"]/1000
test["Income"] = test["Income"]/1000

data.describe()
```

```
Out[19]:
```

| | Income | Kidhome | Teenhome | Recency | MntCoke | MntFruits |
|--------------|---------------|-------------|-------------|-------------|--------------|---------------|
| count | 1680.000000 | 1680.000000 | 1680.000000 | 1680.000000 | 1.680000e+03 | 1680.000000 |
| mean | 52014.343355 | 0.452381 | 0.500000 | 49.083333 | 3.048994e+05 | 25918.452381 |
| std | 21373.445420 | 0.546901 | 0.550055 | 28.930637 | 3.387051e+05 | 39532.059109 |
| min | 1730.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000e+00 | 0.000000 |
| 25% | 35790.750000 | 0.000000 | 0.000000 | 24.000000 | 2.400000e+04 | 1000.000000 |
| 50% | 51445.500000 | 0.000000 | 0.000000 | 50.000000 | 1.730000e+05 | 8000.000000 |
| 75% | 67897.500000 | 1.000000 | 1.000000 | 74.000000 | 4.942500e+05 | 32000.000000 |
| max | 162397.000000 | 2.000000 | 2.000000 | 99.000000 | 1.492000e+06 | 199000.000000 |

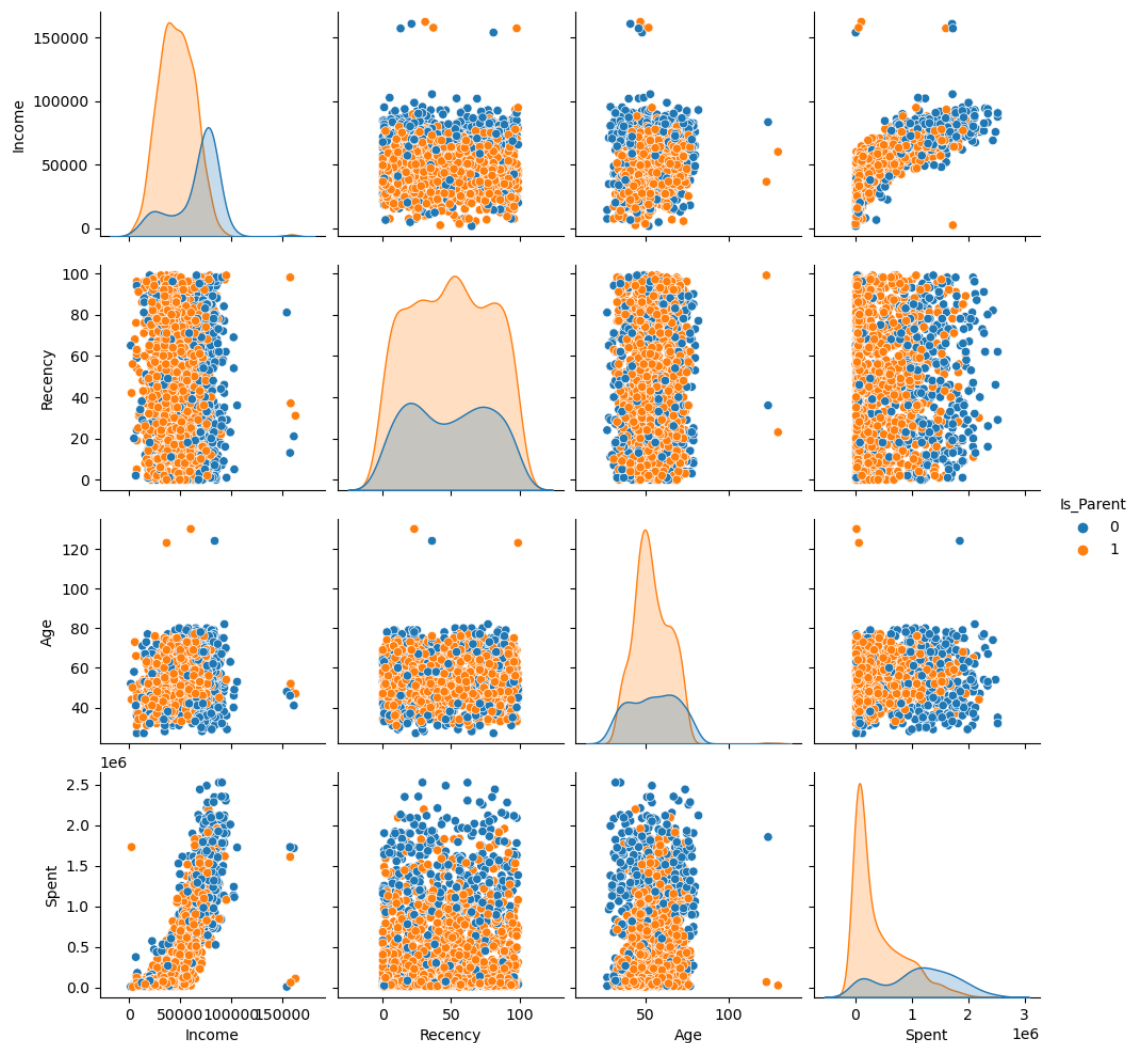
8 rows × 26 columns




```
In [20]: #To plot some selected features
To_Plot = [ "Income", "Recency", "Age", "Spent", "Is_Parent"]
print("Relativig Plot Of Some Selected Features")
plt.figure()
sns.pairplot(data[To_Plot], hue= "Is_Parent")
#Taking hue
plt.show()
```

Relativig Plot Of Some Selected Features

<Figure size 640x480 with 0 Axes>



```
In [21]: #Dropping the outliers on Age and income.
data = data[(data["Age"]<90)]
data = data[(data["Income"]<600000)]

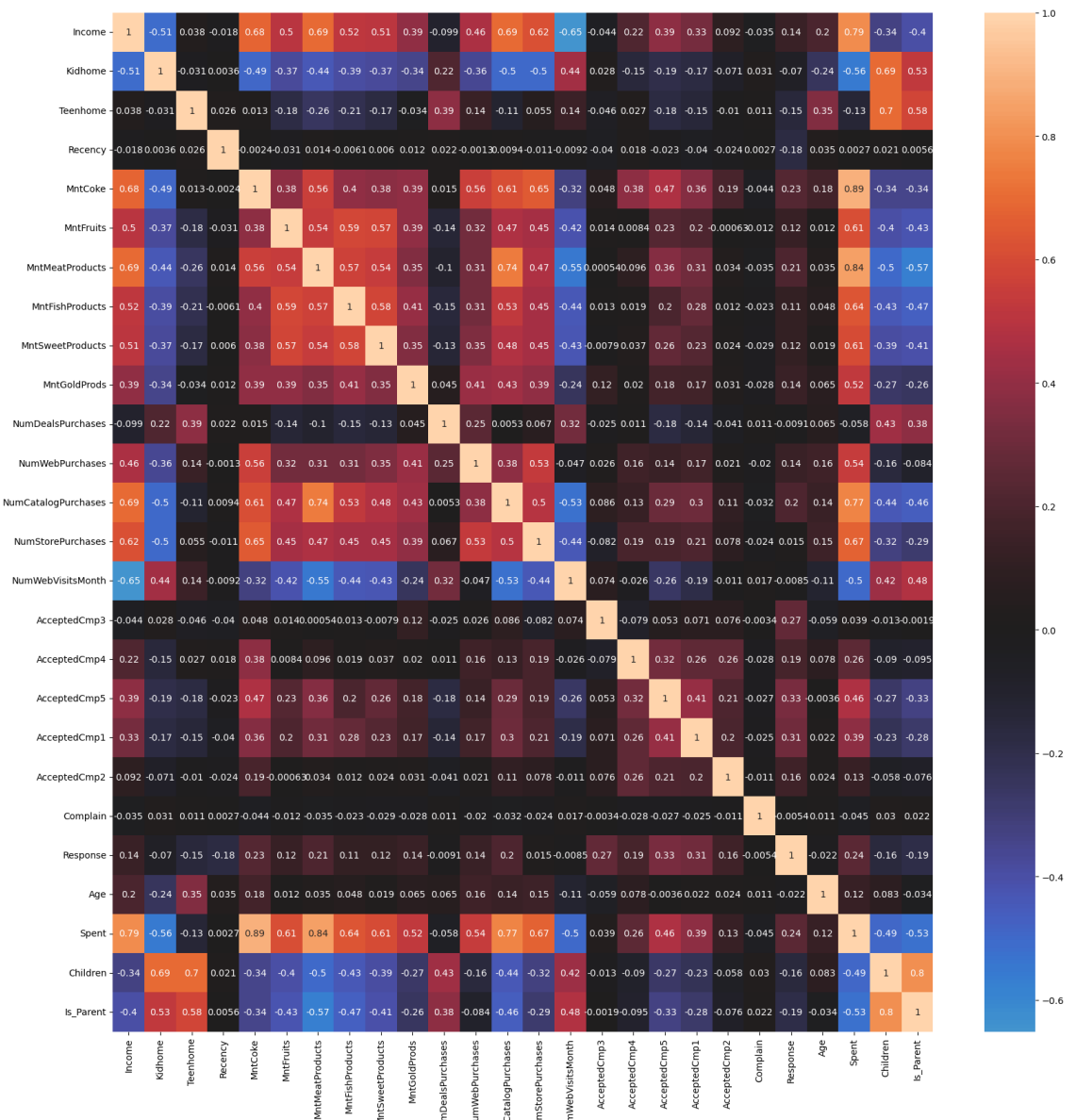
test = test[(test["Age"]<90)]
test = test[(test["Income"]<60000)]

len(data), len(test)
```

Out[21]: (1677, 343)

```
In [22]: #correlation matrix
corrmat= data.corr()
plt.figure(figsize=(20,20))
sns.heatmap(corrmat,annot=True, center=0)
```

Out[22]: <AxesSubplot:>



Data Preprocessing

```
In [23]: #Get List of categorical variables
s = (data.dtypes == 'object')
object_cols = list(s[s].index)

print("Categorical variables in the dataset:", object_cols)
```

Categorical variables in the dataset: ['Education', 'Marital_Status']

```
In [24]: #Label Encoding the object dtypes.
LE=LabelEncoder()
for i in object_cols:
    data[i]=data[[i]].apply(LE.fit_transform)

print("All features are now numerical")
```

All features are now numerical

```
In [25]: LE=LabelEncoder()
for i in object_cols:
    test[i]=test[[i]].apply(LE.fit_transform)
```

```
In [26]: #Creating a copy of data
ds = data.copy()
# creating a subset of dataframe by dropping the features on deals accepted
cols_del = ['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
ds = ds.drop(cols_del, axis=1)
```

```
In [27]: ts = test.copy()
ts = ts.drop(cols_del, axis=1)
```

```
In [28]: #Scaling
scaler = StandardScaler()
scaler.fit(ds)
scaled_ds = pd.DataFrame(scaler.transform(ds),columns= ds.columns )
print("All features are now scaled")
```

All features are now scaled

```
In [29]: scaler = StandardScaler()
scaler.fit(ts)
scaled_ts = pd.DataFrame(scaler.transform(ts),columns= ts.columns )
```

```
In [30]: #Scaled data to be used for reducing the dimensionality
print("Dataframe to be used for further modelling:")
scaled_ds.head()
```

Dataframe to be used for further modelling:

```
Out[30]:
```

| | Education | Marital_Status | Income | Kidhome | Teenhome | Recency | MntCoke | MntFruits |
|---|-----------|----------------|-----------|-----------|-----------|-----------|-----------|-----------|
| 0 | -0.802765 | -1.489737 | 0.287258 | -0.827652 | -0.909671 | 0.308614 | 0.974724 | 1.575077 |
| 1 | -0.802765 | -1.489737 | -0.264686 | 1.001034 | 0.908587 | -0.383107 | -0.868284 | -0.630678 |
| 2 | -0.802765 | 0.024072 | 0.917871 | -0.827652 | -0.909671 | -0.798139 | 0.357434 | 0.586290 |
| 3 | -0.802765 | 0.024072 | -1.186528 | 1.001034 | -0.909671 | -0.798139 | -0.868284 | -0.554617 |
| 4 | 1.064415 | 1.033277 | 0.294512 | 1.001034 | -0.909671 | 1.553710 | -0.389811 | 0.434169 |

5 rows × 21 columns

Clustering

The clustering will be performed by **Agglomerative clustering**. Agglomerative clustering is a hierarchical clustering method. It involves merging examples until the desired number of clusters is achieved.

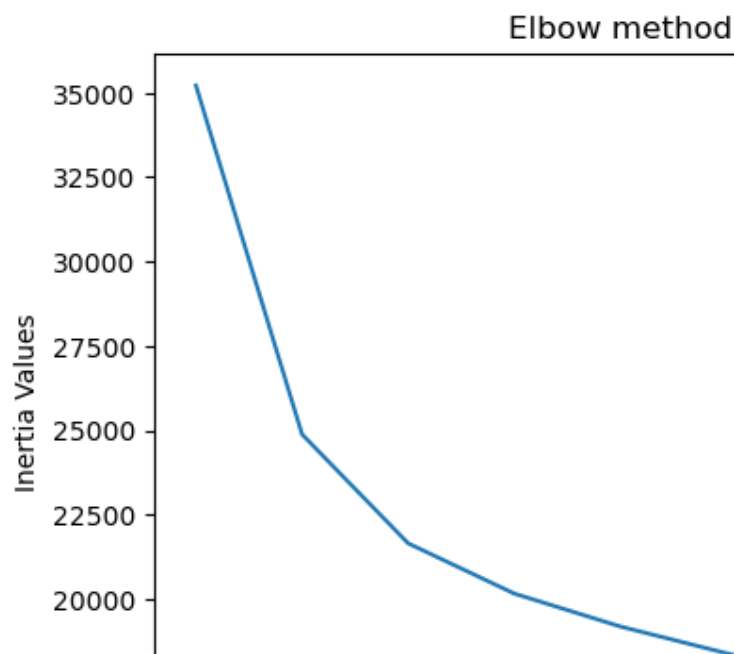
Steps involved in the Clustering:

Elbow Method to determine the number of clusters to be formed
Clustering via Agglomerative Clustering
Examining the clusters formed via scatter plot

```
In [31]: ▶ inertia_values=[]  
k = range(1,10)  
for i in k:  
    model = KMeans(n_clusters=i)  
    model.fit(scaled_ds)  
    inertia_values.append(model.inertia_)
```

C:\Users\HP\anaconda3\lib\site-packages\sklearn\cluster_kmeans.py:1036: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=7.
warnings.warn(

```
In [32]: ▶ plt.plot(k,inertia_values)  
plt.title("Elbow method")  
plt.xlabel("K Values")  
plt.ylabel("Inertia Values")  
  
plt.show()
```



```
In [33]: ► score = silhouette_score(scaled_ds, model.labels_, metric='euclidean')
score
```

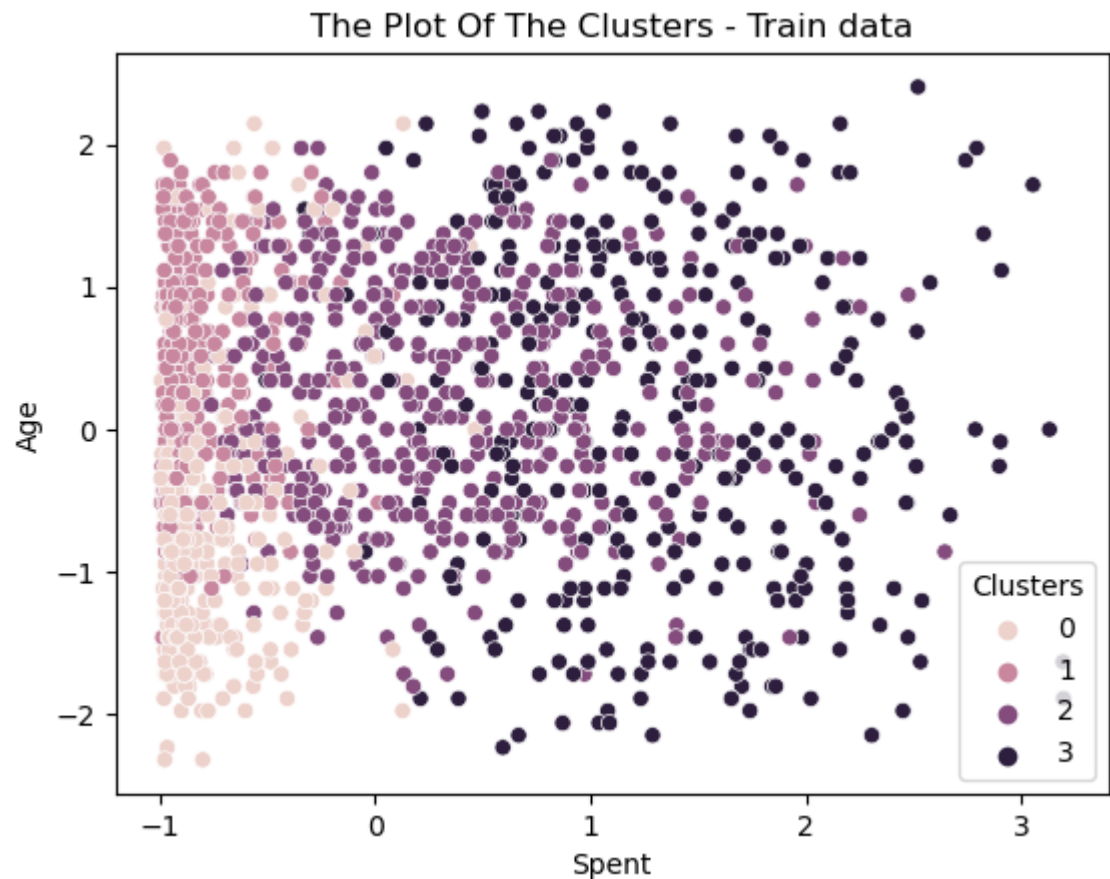
```
Out[33]: 0.14676921361934298
```

```
In [34]: ► #Initiating the Agglomerative Clustering model with k=4
AC = AgglomerativeClustering(n_clusters=4)
# fit model and predict clusters
yhat_AC = AC.fit_predict(scaled_ds)
scaled_ds["Clusters"] = yhat_AC
#Adding the Clusters feature to the original dataframe.
data["Clusters"] = yhat_AC
```

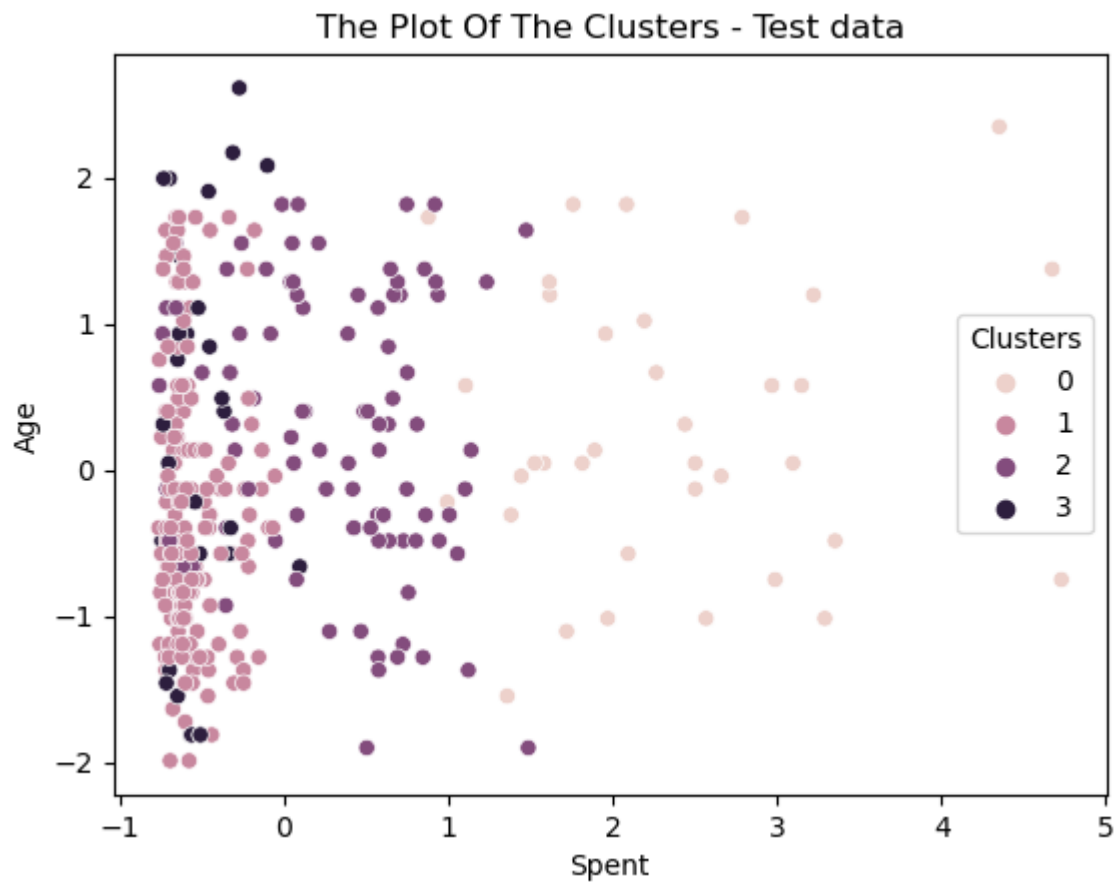
```
In [35]: ► test_AC = AC.fit_predict(scaled_ts)
scaled_ts["Clusters"] = test_AC
test["Clusters"] = test_AC
```

```
In [42]: ► #Plotting the clusters for train data

sns.scatterplot(data = scaled_ds, x=scaled_ds["Spent"], y=scaled_ds["Age"], h
plt.title("The Plot Of The Clusters - Train data")
plt.show()
```



```
In [43]: #Plotting the clusters for test data  
sns.scatterplot(data = scaled_ts,x=scaled_ts["Spent"], y=scaled_ts["Age"],h  
plt.title("The Plot Of The Clusters - Test data")  
plt.show()
```



Evaluating Models

The purpose of this section is to study the patterns in the clusters formed and determine the nature of the clusters' patterns.

```
In [38]: ▶ pl = sns.countplot(x=test["Clusters"])  
          pl.set_title("Distribution Of The Clusters for test data")
```

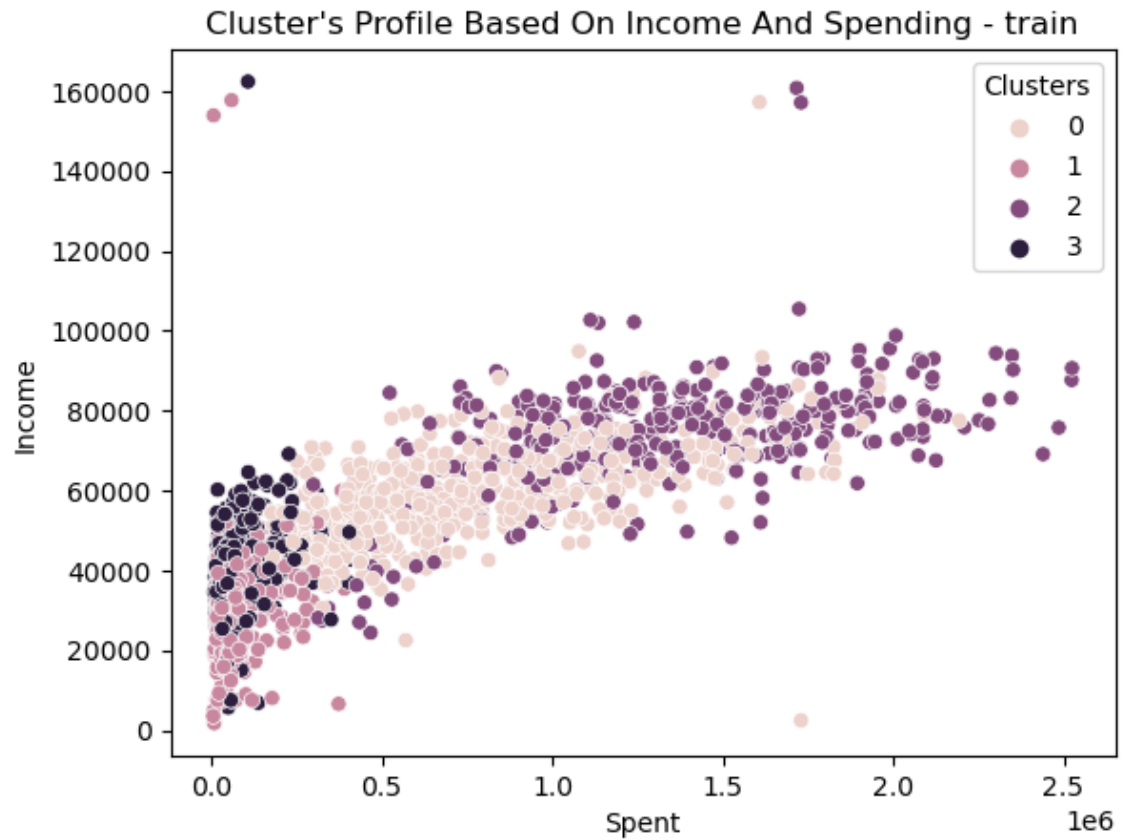
```
Out[38]: Text(0.5, 1.0, 'Distribution Of The Clusters for test data')
```



The clusters seem to be fairly distributed

```
In [39]: ▶ pl = sns.scatterplot(data = data,x=data["Spent"], y=data["Income"],hue=data["Cluster"],  
pl.set_title("Cluster's Profile Based On Income And Spending - train")
```

```
Out[39]: Text(0.5, 1.0, "Cluster's Profile Based On Income And Spending - train")
```

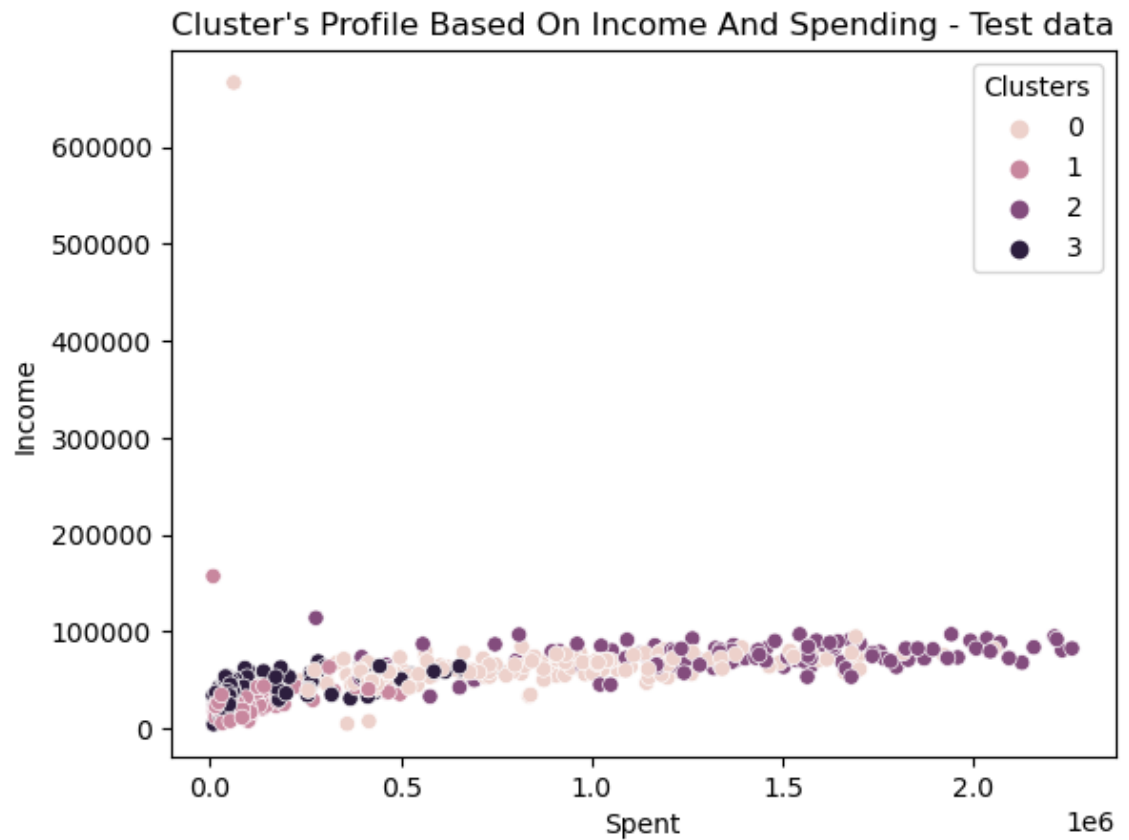


Classification

- Group 0: Low income and low spending
- Group 1: High income and high spending
- Group 2: High income and low spending
- Group 3: Average income and average spending


```
In [40]: ▶ pl = sns.scatterplot(data = test,x=test["Spent"], y=test["Income"],hue=test["Cluster"],  
pl.set_title("Cluster's Profile Based On Income And Spending - Test data"))
```

```
Out[40]: Text(0.5, 1.0, "Cluster's Profile Based On Income And Spending - Test data")
```



Classification

Group 0: High income and average spending

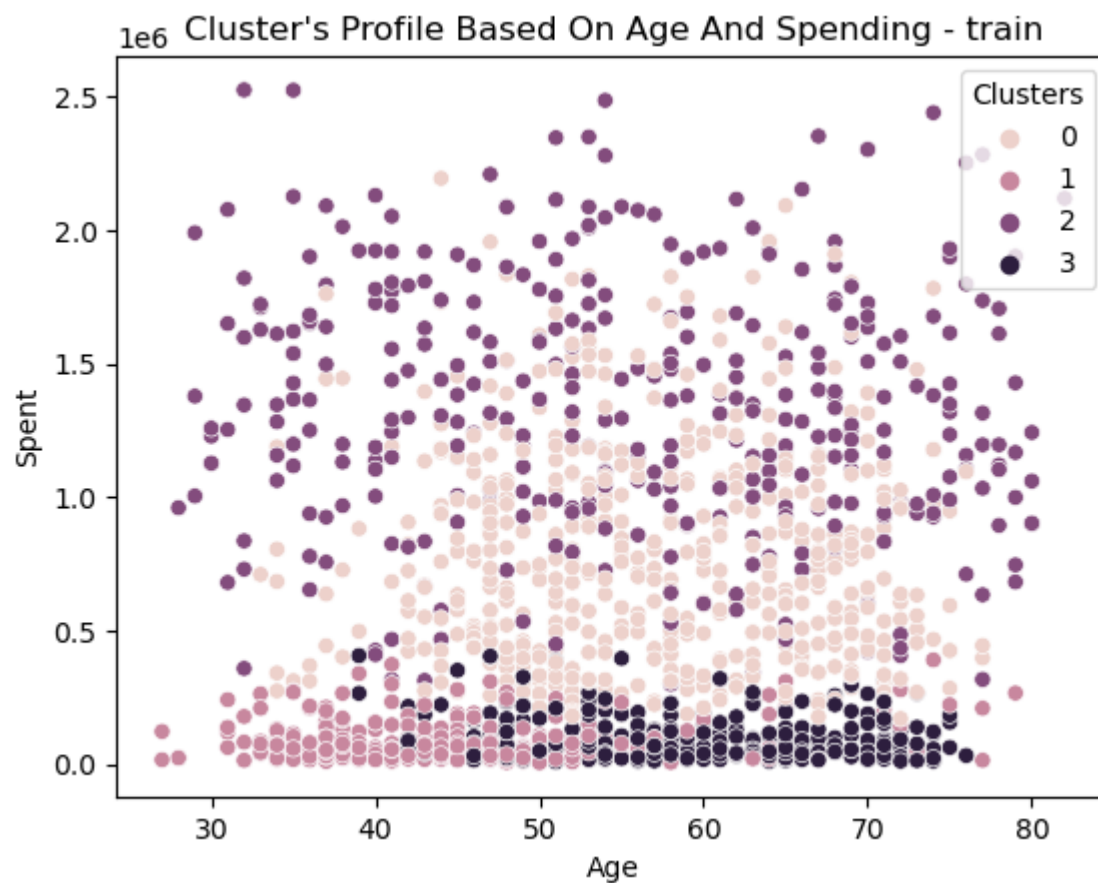
Group 1: Low income and low spending

Group 2: Low income high spending

Group 3: Average income and low spending

```
In [41]: ▶ pl = sns.scatterplot(data = data,x=data["Age"], y=data["Spent"],hue=data["C"]  
pl.set_title("Cluster's Profile Based On Age And Spending - train")
```

```
Out[41]: Text(0.5, 1.0, "Cluster's Profile Based On Age And Spending - train")
```



```
In [42]: ▶ pl = sns.scatterplot(data = test,x=test["Age"], y=test["Spent"],hue=test["C"]  
pl.set_title("Cluster's Profile Based On Age And Spending - test")
```

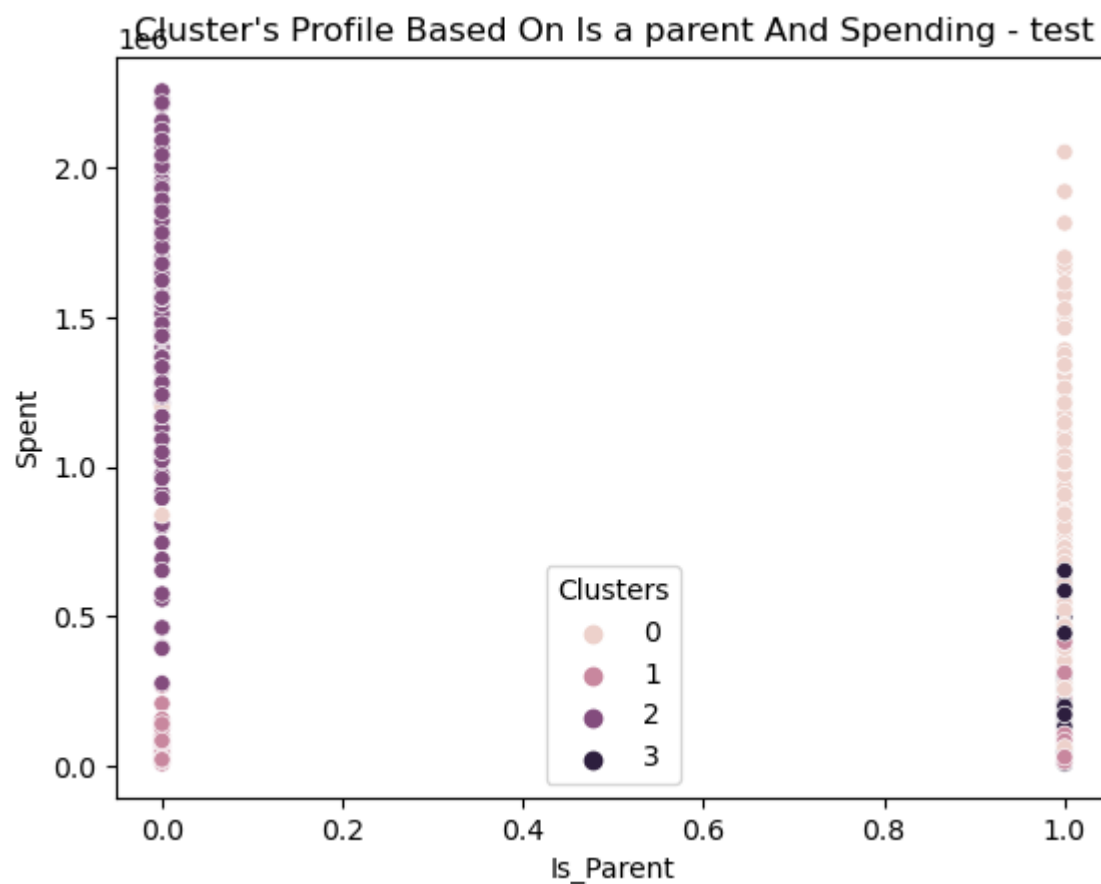
```
Out[42]: Text(0.5, 1.0, "Cluster's Profile Based On Age And Spending - test")
```



Group 0: Age between 40 & 80 with average to high spending
Group 1: Age between 30 & 80 with average spending
Group 2: Age between 25 & 80 with high spending
Group 3: Age between 40 & 70 low spending

```
In [43]: ▶ pl = sns.scatterplot(data = test,x=test["Is_Parent"], y=test["Spent"],hue=t  
pl.set_title("Cluster's Profile Based On Is a parent And Spending - test")
```

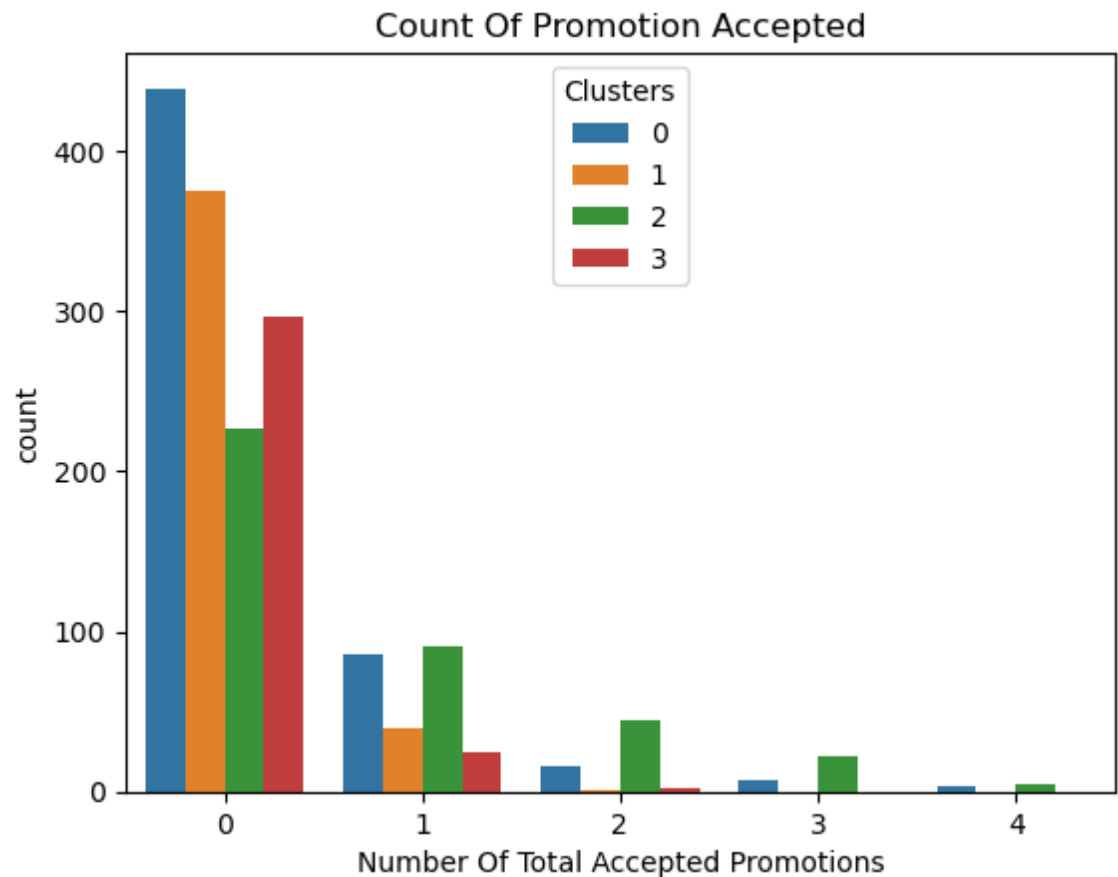
```
Out[43]: Text(0.5, 1.0, "Cluster's Profile Based On Is a parent And Spending - tes  
t")
```



Most spending comes from customers who are not parents

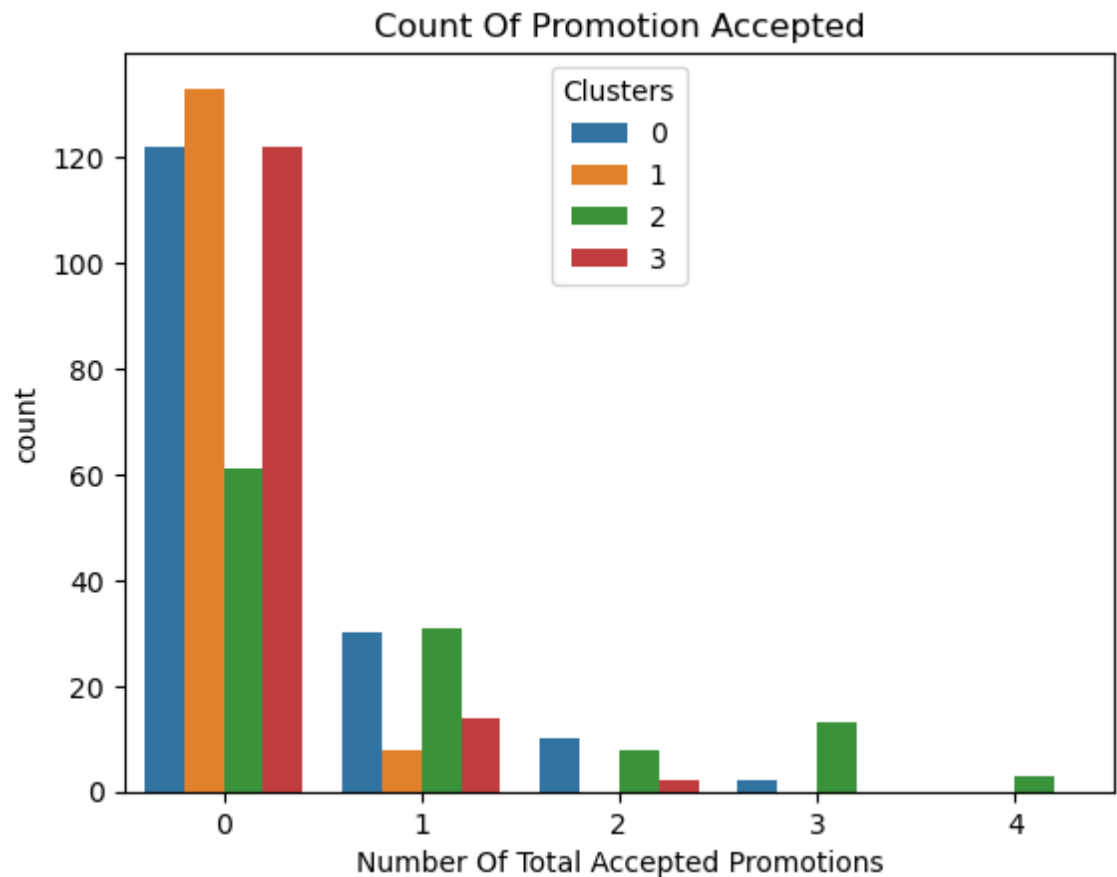
```
In [44]: #Creating a feature to get a sum of accepted promotions for train data  
data["Total_Promos"] = data["AcceptedCmp1"]+ data["AcceptedCmp2"]+ data["Ac  
#Plotting count of total campaign accepted.  
pl = sns.countplot(x=data["Total_Promos"],hue=data["Clusters"])  
pl.set_title("Count Of Promotion Accepted")  
pl.set_xlabel("Number Of Total Accepted Promotions")
```

Out[44]: Text(0.5, 0, 'Number Of Total Accepted Promotions')



```
In [45]: #Creating a feature to get a sum of accepted promotions for test data
test["Total_Promos"] = test["AcceptedCmp1"]+ test["AcceptedCmp2"]+ test["Ac
#Plotting count of total campaign accepted.
pl = sns.countplot(x=test["Total_Promos"],hue=test["Clusters"])
pl.set_title("Count Of Promotion Accepted")
pl.set_xlabel("Number Of Total Accepted Promotions")
```

Out[45]: Text(0.5, 0, 'Number Of Total Accepted Promotions')



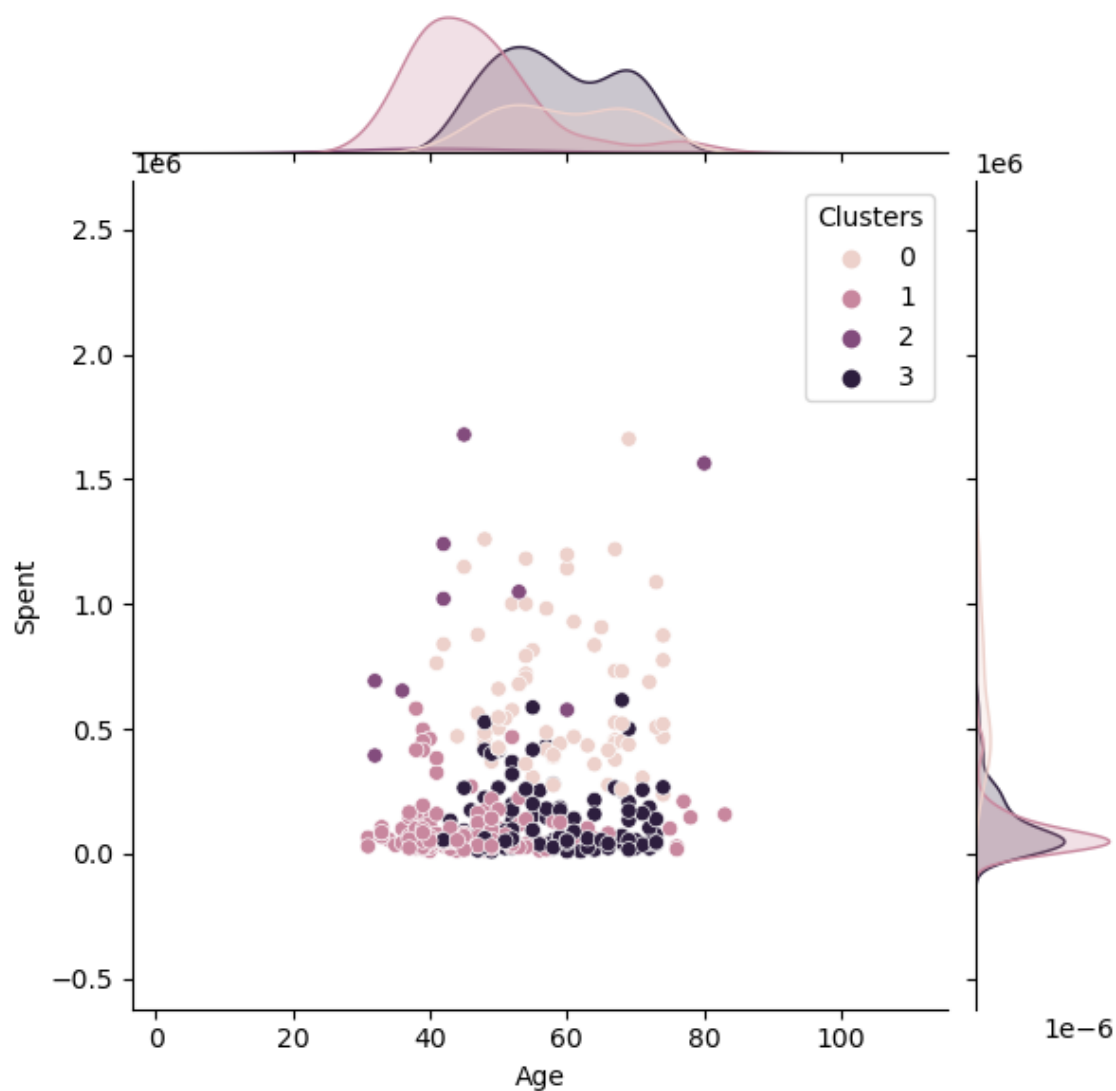
There isnt any overwhelming response to the campaigns so far. There are only a few participants overall. Moreover, no one has taken part in all 5 of them. Perhaps better-targeted and well-planned campaigns are required to boost sales.

Profiling the clusters

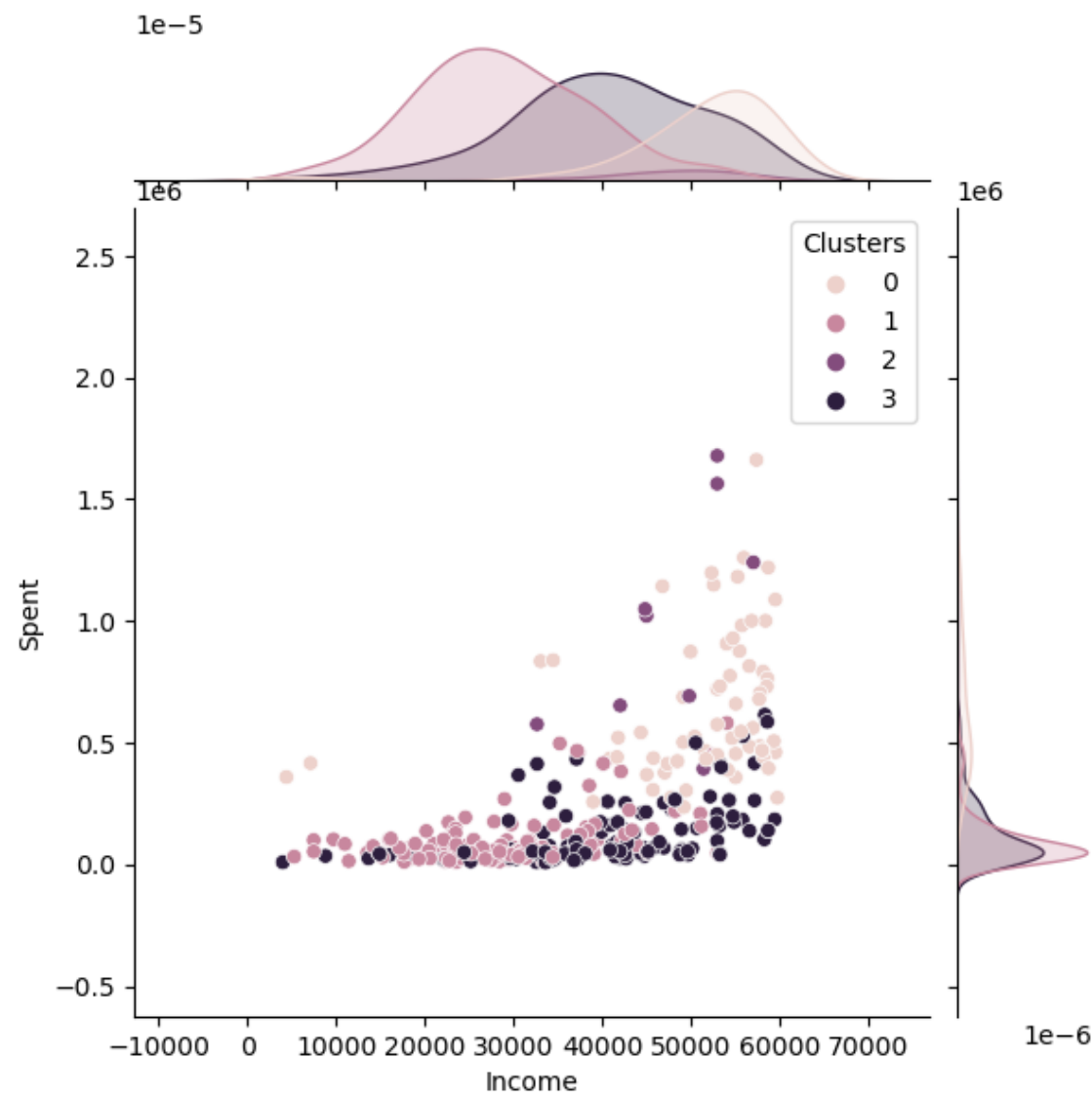
This is done to get an idea about who are valuable customer and who needs more attention from the retail store's marketing team. Plotting some of the features that indicates customer's personal traits based on the cluster they are in.

```
In [59]: Features = ["Age", "Income", "Marital_Status", "Teenhome", "Is_Parent", "Edu"]  
  
for i in Features:  
    plt.figure()  
    sns.jointplot(x=test[i], y=test["Spent"], hue=test["Clusters"], kind="scatter")  
    plt.show()
```

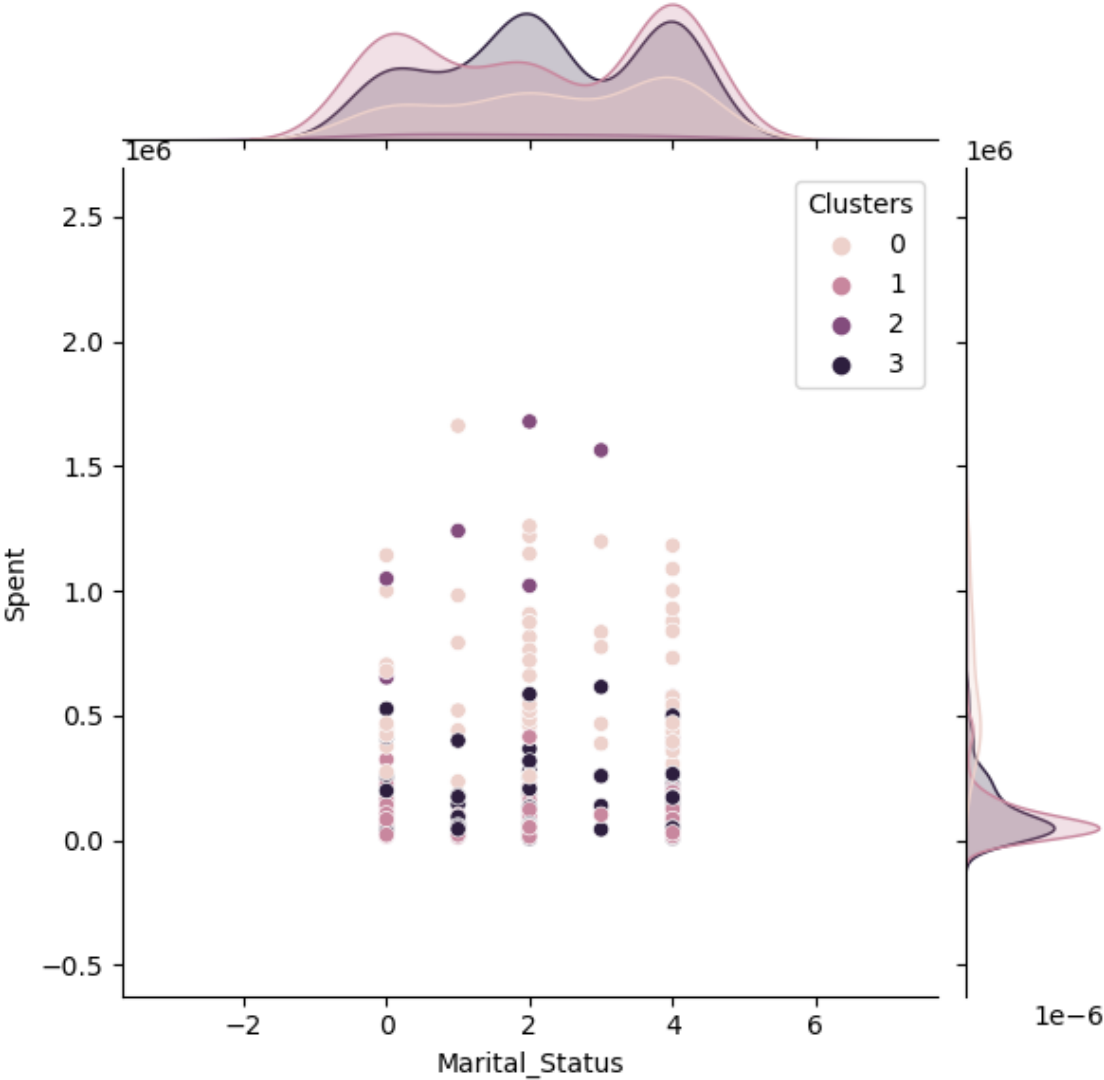
<Figure size 640x480 with 0 Axes>



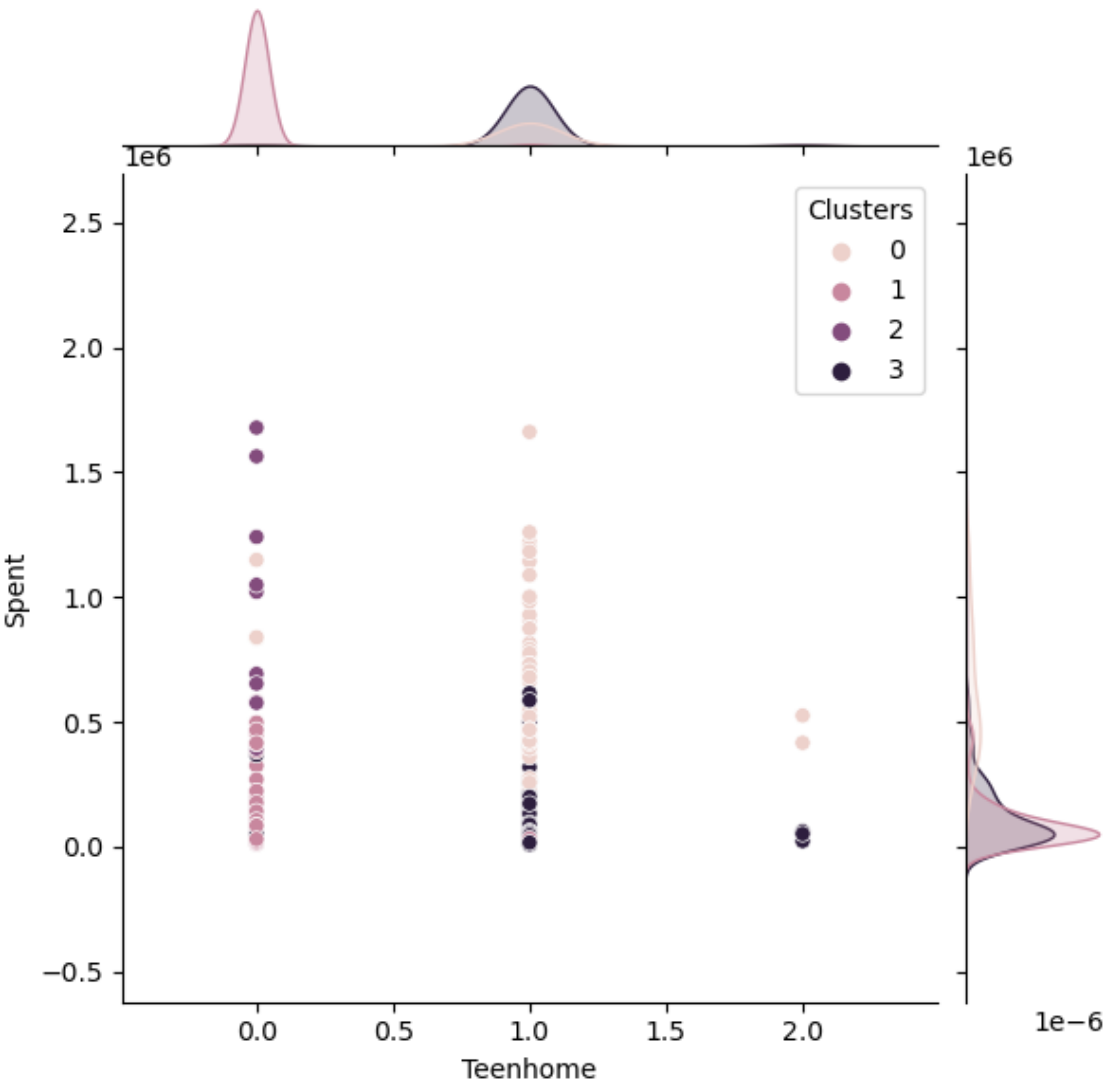
<Figure size 640x480 with 0 Axes>



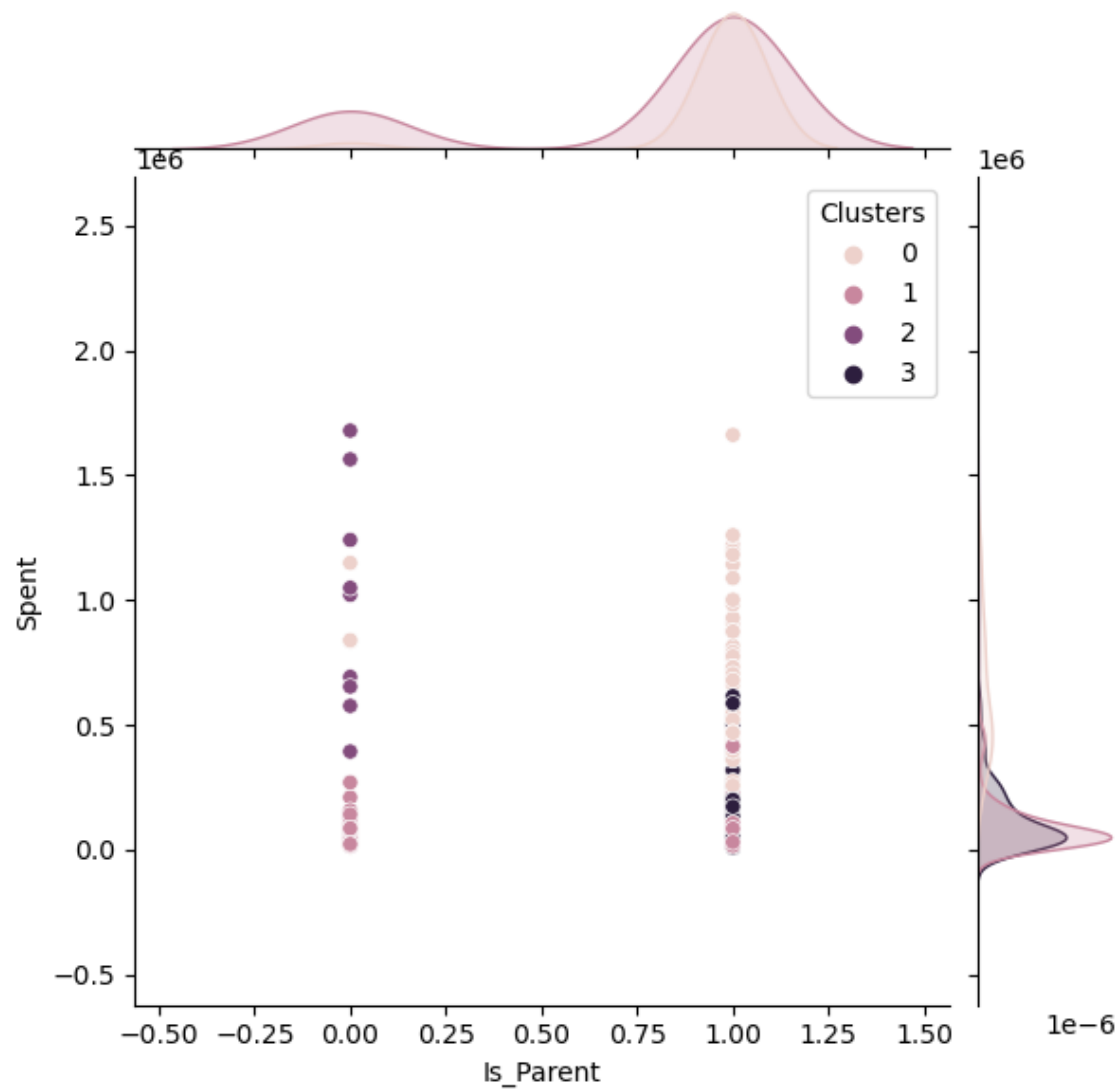
<Figure size 640x480 with 0 Axes>



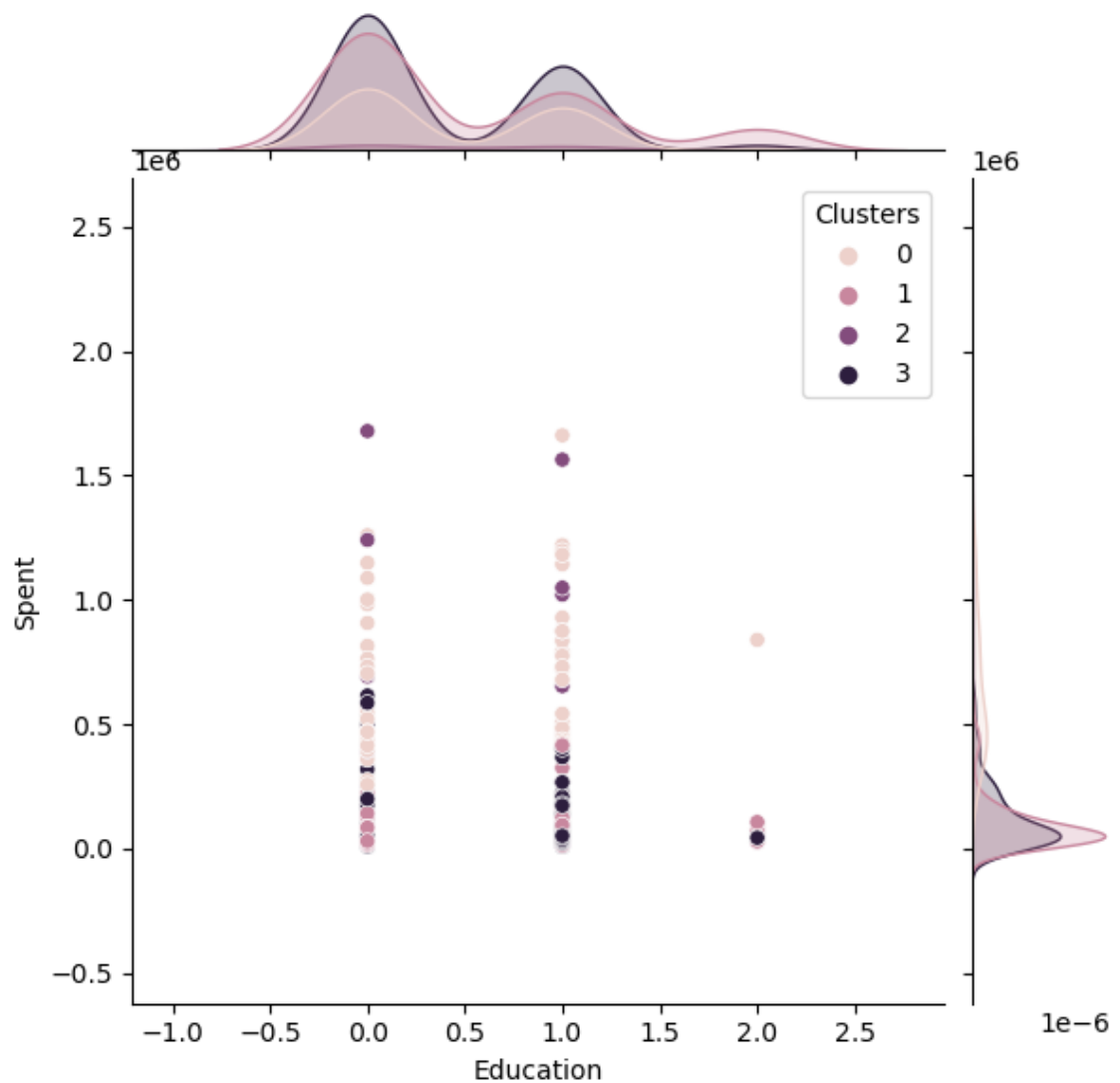
<Figure size 640x480 with 0 Axes>



<Figure size 640x480 with 0 Axes>



<Figure size 640x480 with 0 Axes>



The following inferences are obtained from the analysis:

Cluster 0:

Between age 40 & 80 with average spending with high income. Mostly parents with teens at home. Mostly graduates

Cluster 1:

Between age 20 & 90 with low spending and low-average income. Mostly parents with all education types

Cluster 2:

Between age 40 & 80 with high spending and average to high income. Includes people in all marital status and all education types

Cluster 3:

Between age 40 & 80 low spending and mostly high income. There are mostly parents with teens.

Classification

Trying the classification using Random Forest Classifier

In [44]: `data.head()`

Out[44]:

| | Education | Marital_Status | Income | Kidhome | Teenhome | Dt_Customer | Recency | MntCoke |
|---|-----------|----------------|---------|---------|----------|-------------|---------|---------|
| 0 | 0 | 0 | 58138.0 | 0 | 0 | 2012-04-09 | 58 | 635000 |
| 1 | 0 | 0 | 46344.0 | 1 | 1 | 2014-08-03 | 38 | 11000 |
| 2 | 0 | 3 | 71613.0 | 0 | 0 | 2013-08-21 | 26 | 426000 |
| 3 | 0 | 3 | 26646.0 | 1 | 0 | 2014-10-02 | 26 | 11000 |
| 4 | 1 | 5 | 58293.0 | 1 | 0 | 2014-01-19 | 94 | 173000 |

5 rows × 30 columns

In [51]: `target = ["Clusters"]
x_train = scaled_ds.drop(target, axis=1)
y_train = scaled_ds[target]`

In [52]: `x_test = scaled_ts.drop(target,axis=1)
y_test = scaled_ts["Clusters"]`

In [53]: `from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(x_train, y_train)`

C:\Users\HP\AppData\Local\Temp\ipykernel_16920\3814242233.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
model.fit(x_train, y_train)

Out[53]: RandomForestClassifier()

In [62]: `model.score(x_train,y_train)`

Out[62]: 1.0

In [69]: `model.score(x_test,y_test)`

Out[69]: 0.42565597667638483

```
In [61]: model.predict([[-0.768549, -0.138043, 0.025063, 0.764156, 0.823622, 1.271718, -0.
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning:  
X does not have valid feature names, but DecisionTreeClassifier was fitted  
with feature names  
warnings.warn(  

```

```
Out[61]: array([2], dtype=int64)
```

```
In [68]: y_test.values[2]
```

```
Out[68]: 1
```

```
In [67]: y_test.iloc[0]
```

```
Out[67]: 1
```

The cluster is predicted correctly for the given input and the accuracy score for test data is 0.42

Conclusion

The dataset is analysed and segmented using unsupervised clustering. The whole customers are categorized in to 4 clusters based on their income, age and family background. This can be made use by the company to plan their products and promotions.

Thank you!!

```
In [ ]: 
```