Name: Inam Ul Haq Qureshi
CNIC: 8220356241979

Customer Segmentation

In this project, I will be performing an unsupervised clustering of data on the customer's records from a groceries firm's database. Customer segmentation is the practice of separating customers into groups that reflect similarities among customers in each cluster. I will divide customers into segments to optimize the significance of each customer to the business. To modify products according to distinct

TABLE OF CONTENTS

- 6. CLUSTERING
- 7. EVALUATING MODELS
- 8. PROFILING
- 9. CONCLUSION

IMPORTING LIBRARIES

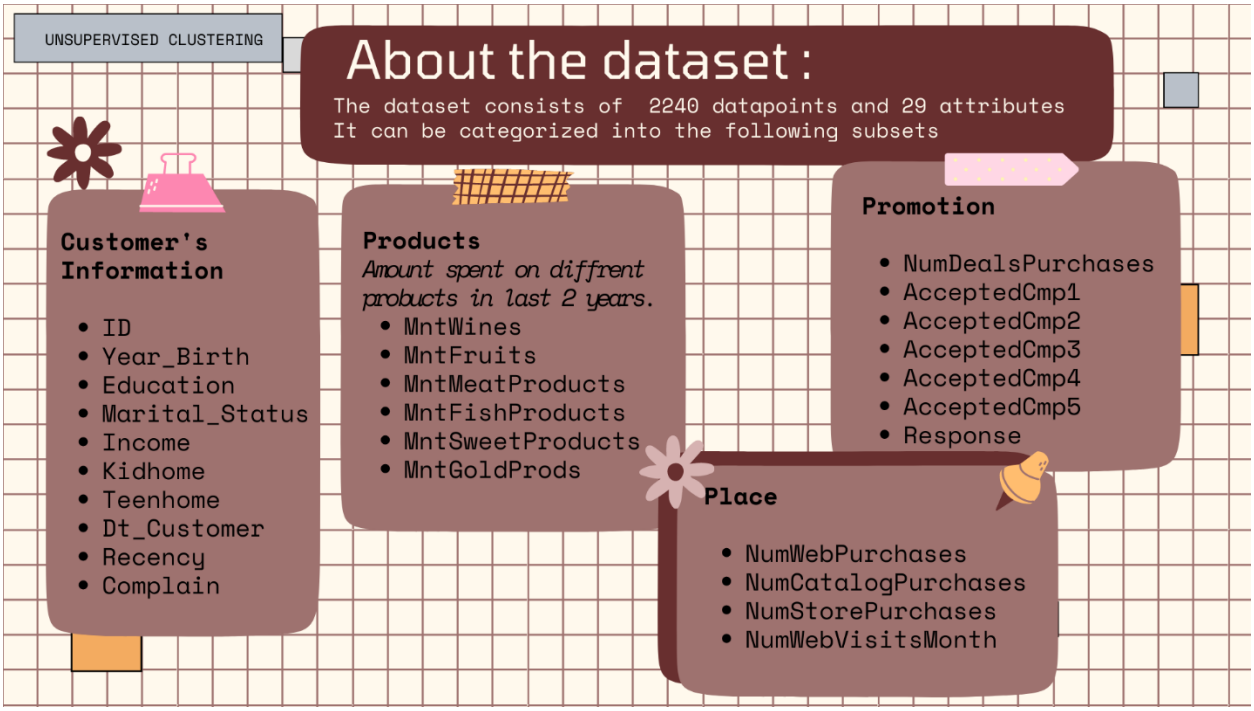
```
import numpy as np
import pandas as pd
import datetime
import matplotlib
import matplotlib.pyplot as plt
from matplotlib import colors
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from yellowbrick.cluster import KElbowVisualizer
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt, numpy as np
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import AgglomerativeClustering
from matplotlib.colors import ListedColormap
from sklearn import metrics
import warnings
import sys
if not sys.warnoptions:
    warnings.simplefilter("ignore")
np.random.seed(42)
```

LOADING DATA

data = pd.read_csv("../input/customer-personality-analysis/marketing_campaign.csv", sep="\t")

print("Number of datapoints:", len(data))

data.head()



DATA CLEANING

In this section

- Data Cleaning
- Feature Engineering

In order to, get a full grasp of what steps should I be taking to clean the dataset. Let us have a look at the information in data.

data.info()

#	Column	Non-Null Count	Dtype
0	ID	2240 non-null	int64
1	Year_Birth	2240 non-null	int64
2	Education	2240 non-null	object
3	Marital_Status	2240 non-null	object
4	Income	2216 non-null	float64
5	Kidhome	2240 non-null	int64
6	Teenhome	2240 non-null	int64
7	Dt_Customer	2240 non-null	object
8	Recency	2240 non-null	int64
9	MntWines	2240 non-null	int64
10	MntFruits	2240 non-null	int64
11	MntMeatProducts	2240 non-null	int64
12	MntFishProducts	2240 non-null	int64
13	MntSweetProducts	2240 non-null	int64
14	MntGoldProds	2240 non-null	int64
15	NumDealsPurchases	2240 non-null	int64
16	NumWebPurchases	2240 non-null	int64
17	NumCatalogPurchases	2240 non-null	int64
18	NumStorePurchases	2240 non-null	int64
19	NumWebVisitsMonth	2240 non-null	int64
20	AcceptedCmp3	2240 non-null	int64
21	AcceptedCmp4	2240 non-null	int64
22	AcceptedCmp5	2240 non-null	int64
23	AcceptedCmp1	2240 non-null	int64
24	AcceptedCmp2	2240 non-null	int64
25	Complain	2240 non-null	int64
26	Z_CostContact	2240 non-null	int64

27 Z_Revenue 2240 non-null int64

28 Response 2240 non-null int64

From the above output, we can conclude and note that:

- There are missing values in income
- Dt_Customer that indicates the date a customer joined the database is not parsed as DateTime
- There are some categorical features in our data frame; as there are some features in dtype: object). So we will need to encode them into numeric forms later.

First of all, for the missing values, I am simply going to drop the rows that have missing income values.

```
data = data.dropna()

print("The total number of data-points after removing the rows with missing values are:", len(data))
```

In the next step, I am going to create a feature out of "Dt_Customer" that indicates the number of days a customer is registered in the firm's database. However, in order to keep it simple, I am taking this value relative to the most recent customer in the record.

```
data["Dt_Customer"] = pd.to_datetime(data["Dt_Customer"])

dates = []

for i in data["Dt_Customer"]:

    i = i.date()

    dates.append(i)

print("The newest customer's enrolment date in the records:",max(dates))

print("The oldest customer's enrolment date in the records:",min(dates))
```

Creating a feature ("Customer_For") of the number of days the customers started to shop in the store relative to the last recorded date:

```
days = []

d1 = max(dates) #taking it to be the newest customer

for i in dates:

    delta = d1 - i

    days.append(delta)data["Customer_For"] = days
```

Now we will be exploring the unique values in the categorical features to get a clear idea of the data.

```
print("Total categories in the feature Marital_Status:\n", data["Marital_Status"].value_counts(), "\n")

print("Total categories in the feature Education:\n", data["Education"].value_counts())
```

Total categories in the feature Marital Status:

```
Married     857
Together    573
Single      471
Divorced    232
Widow        76
Alone         3
Absurd        2
YOLO          2
```

Total categories in the feature Education:

```
Graduation    1116
PhD            481
Master         365
2n Cycle       200
Basic           54
```

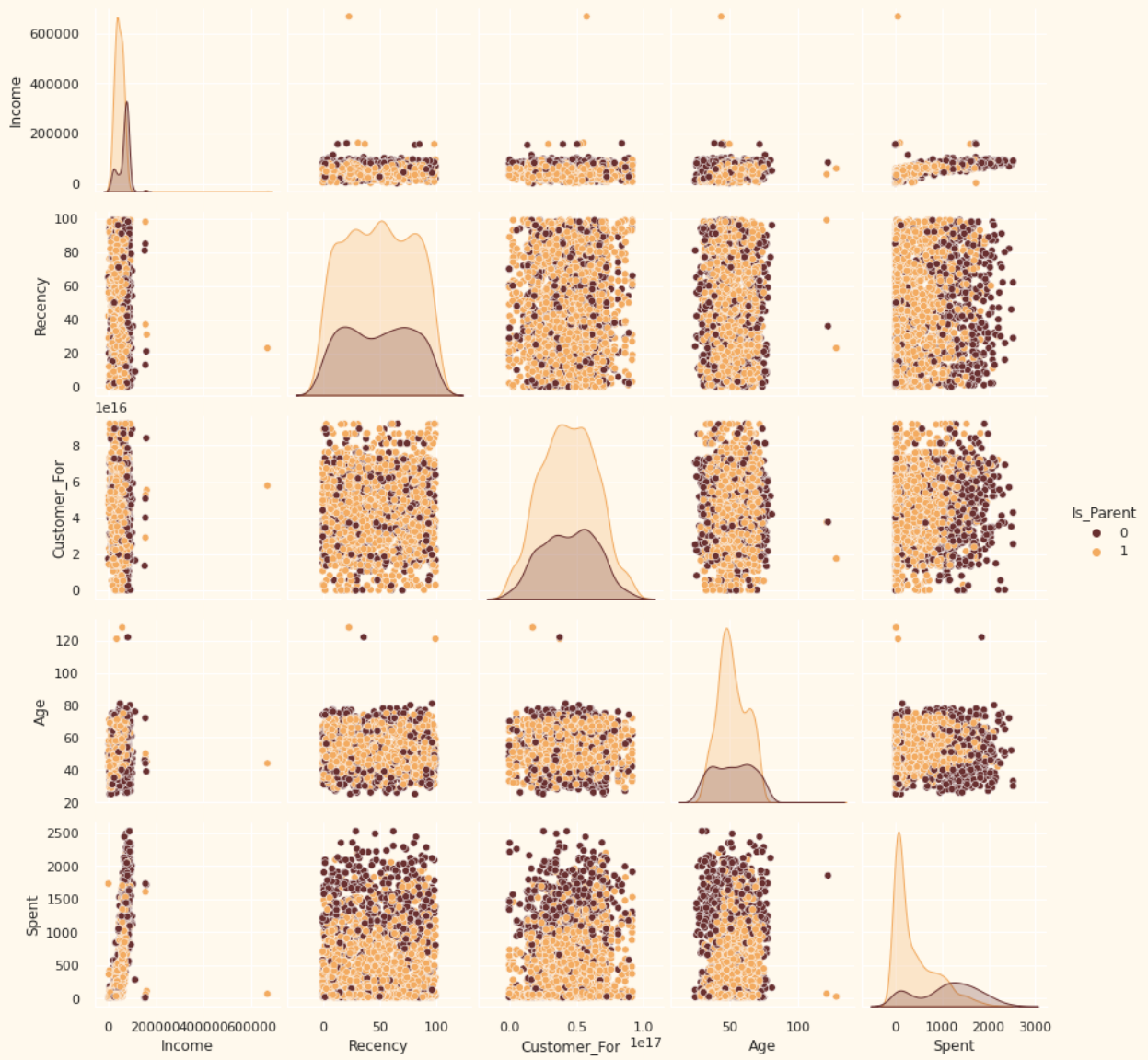
In the next bit, I will be performing the following steps to engineer some new features:

- Extract the "Age" of a customer by the "Year_Birth" indicating the birth year of the respective person.
- Create another feature "Spent" indicating the total amount spent by the customer in various categories over the span of two years.
- Create another feature "Living_With" out of "Marital_Status" to extract the living situation of couples.
- Create a feature "Children" to indicate total children in a household that is, kids and teenagers.
- To get further clarity of household, Creating feature indicating "Family_Size"

- Create a feature "Is_Parent" to indicate parenthood status
- Lastly, I will create three categories in the "Education" by simplifying its value counts.
- Dropping some of the redundant features

```
data["Age"] = 2021-data["Year_Birth"]

data["Spent"] = data["MntWines"]+ data["MntFruits"]+ data["MntMeatProducts"]+ data["MntFishProducts"]+ data["MntSweetProducts"]+ data["MntGoldProds"]

data["Living_With"]=data["Marital_Status"].replace({"Married":"Partner", "Together":"Partner", "Absurd":"Alone", "Widow":"Alone", "YOLO":"Alone", "Divorced":"Alone", "Single":"Alone",})

data["Children"]=data["Kidhome"]+data["Teenhome"]

data["Family_Size"] = data["Living_With"].replace({"Alone": 1, "Partner":2})+ data["Children"]

data["Is_Parent"] = np.where(data.Children> 0, 1, 0)

data["Education"]=data["Education"].replace({"Basic":"Undergraduate","2n Cycle":"Undergraduate", "Graduation":"Graduate", "Master":"Postgraduate", "PhD":"Postgraduate"})

data=data.rename(columns={"MntWines":
"Wines","MntFruits":"Fruits","MntMeatProducts":"Meat","MntFishProducts":"Fish","MntSweetProducts":"Sweets","MntGoldProds":"Gold"})

to_drop = ["Marital_Status", "Dt_Customer", "Z_CostContact", "Z_Revenue", "Year_Birth", "ID"]

data = data.drop(to_drop, axis=1)
```

The above stats show some discrepancies in mean Income and Age and max Income and age.

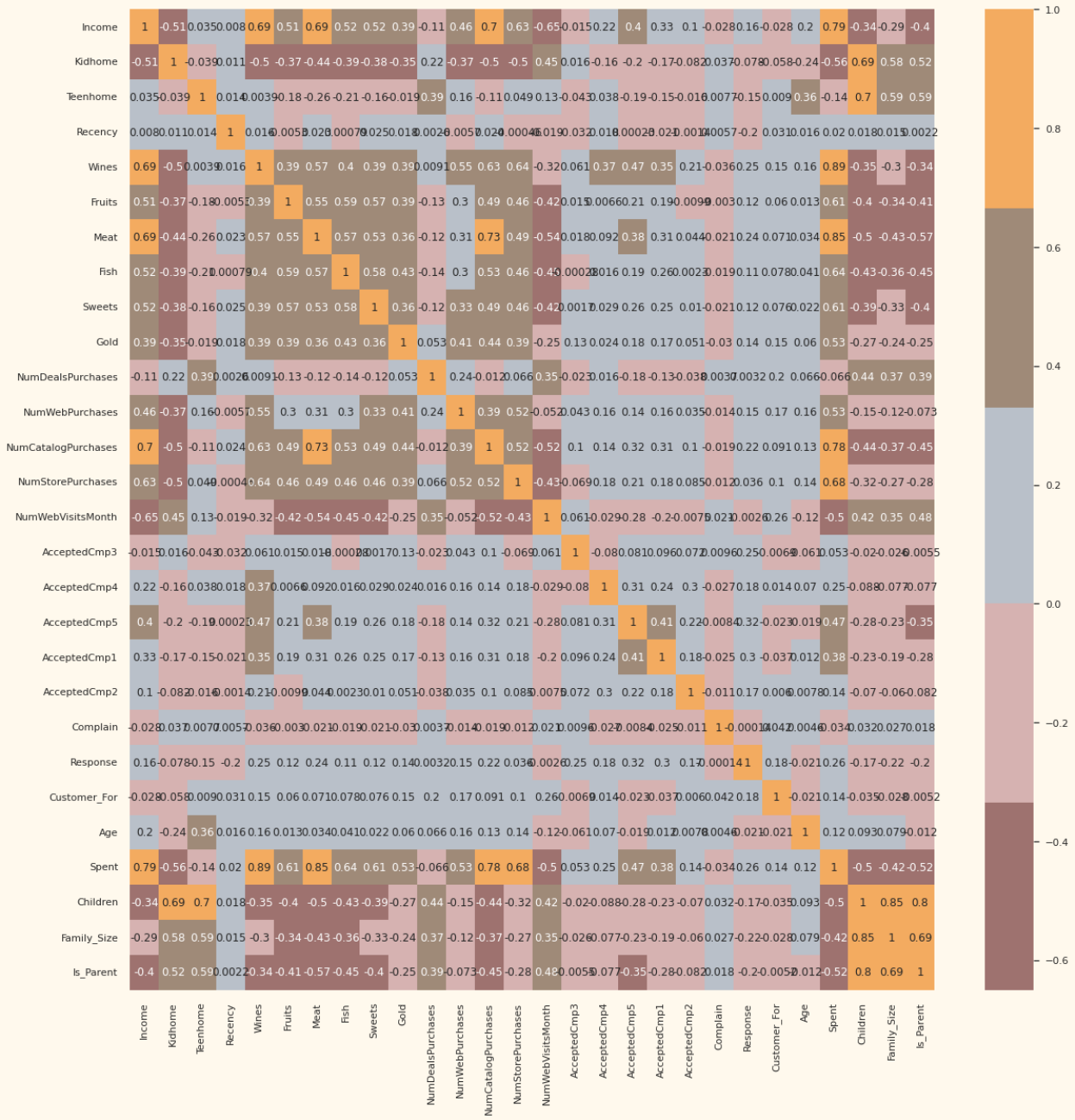


Clearly, there are a few outliers in the Income and Age features. I will be deleting the outliers in the data.

#Dropping the outliers by setting a cap on Age and income.

```
data = data[(data["Age"]<90)]
data = data[(data["Income"]<600000)]
print("The total number of data-points after removing the outliers are:", len(data))
```

The total number of data-points after removing the outliers are: 2212



The data is quite clean and the new features have been included. I will proceed to the next step. That is, preprocessing the data.

DATA PREPROCESSING

In this section, I will be preprocessing the data to perform clustering operations.

The following steps are applied to preprocess the data:

- Label encoding the categorical features

- Scaling the features using the standard scaler
- Creating a subset dataframe for dimensionality reduction

```
#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', 'Living_With']
```

```
#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
```

```
#Creating a copy of data
ds = data.copy()
# creating a subset of dataframe by dropping the features on deals accepted and promotions
cols_del = ['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1','AcceptedCmp2', 'Complain', 'Response']
ds = ds.drop(cols_del, axis=1)
#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
```

DIMENSIONALITY REDUCTION

In this problem, there are many factors on the basis of which the final classification will be done. These factors are basically attributes or features. The higher the number of features, the harder it is to work with it. Many of these features are correlated, and hence redundant. This is why I will be performing dimensionality reduction on the selected features before putting them through a classifier.
Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables.

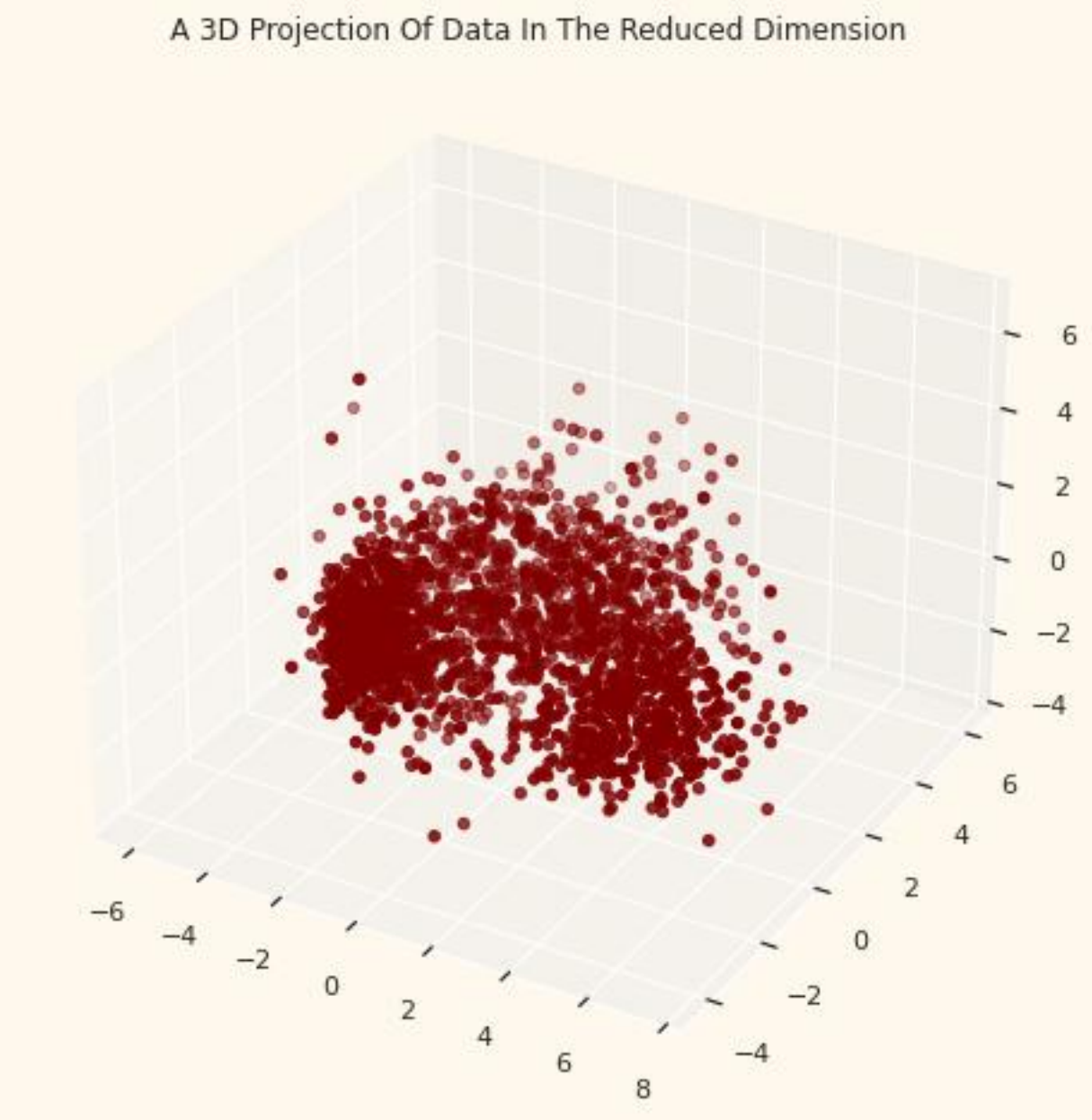
Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss.

Steps in this section:

- Dimensionality reduction with PCA
- Plotting the reduced dataframe

Dimensionality reduction with PCA

For this project, I will be reducing the dimensions to 3.

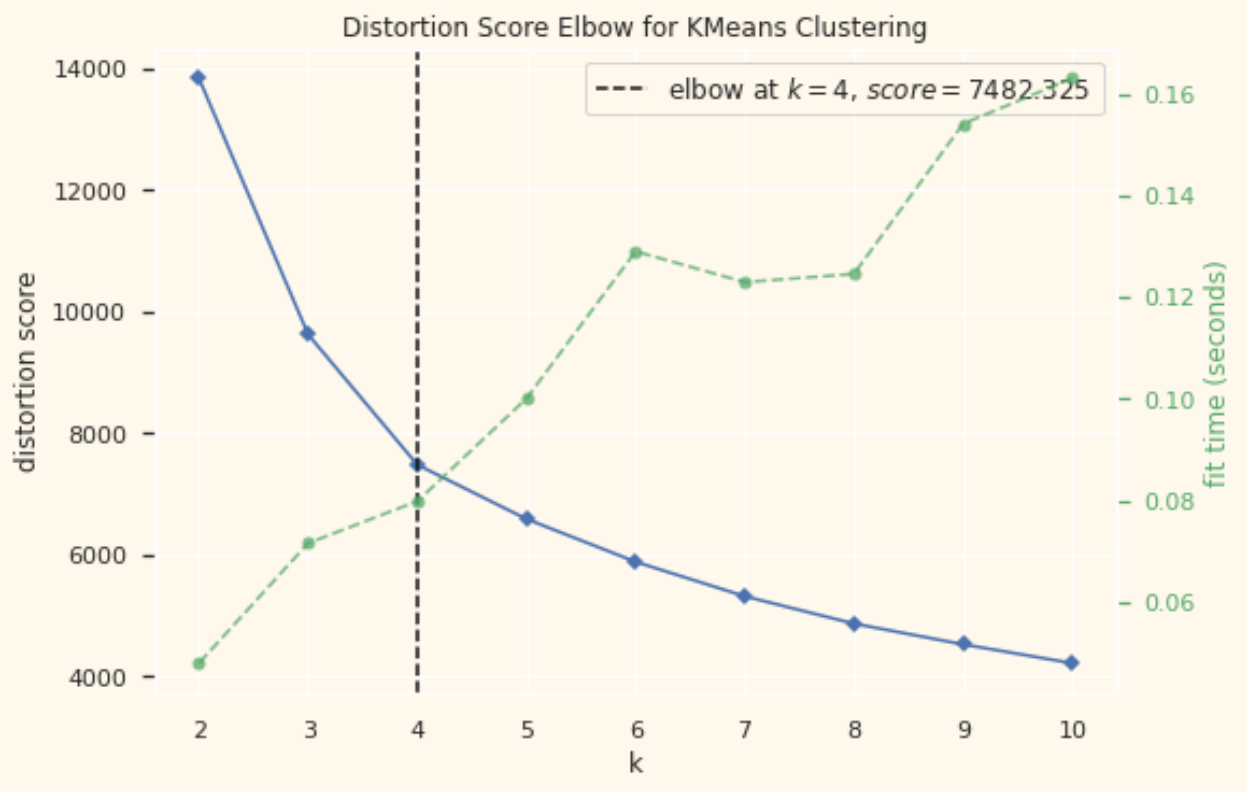


CLUSTERING

Now that I have reduced the attributes to three dimensions, I will be performing clustering via Agglomerative clustering. Agglomerative clustering is a hierarchical clustering method. It involves merging examples until the desired number of clusters is achieved.

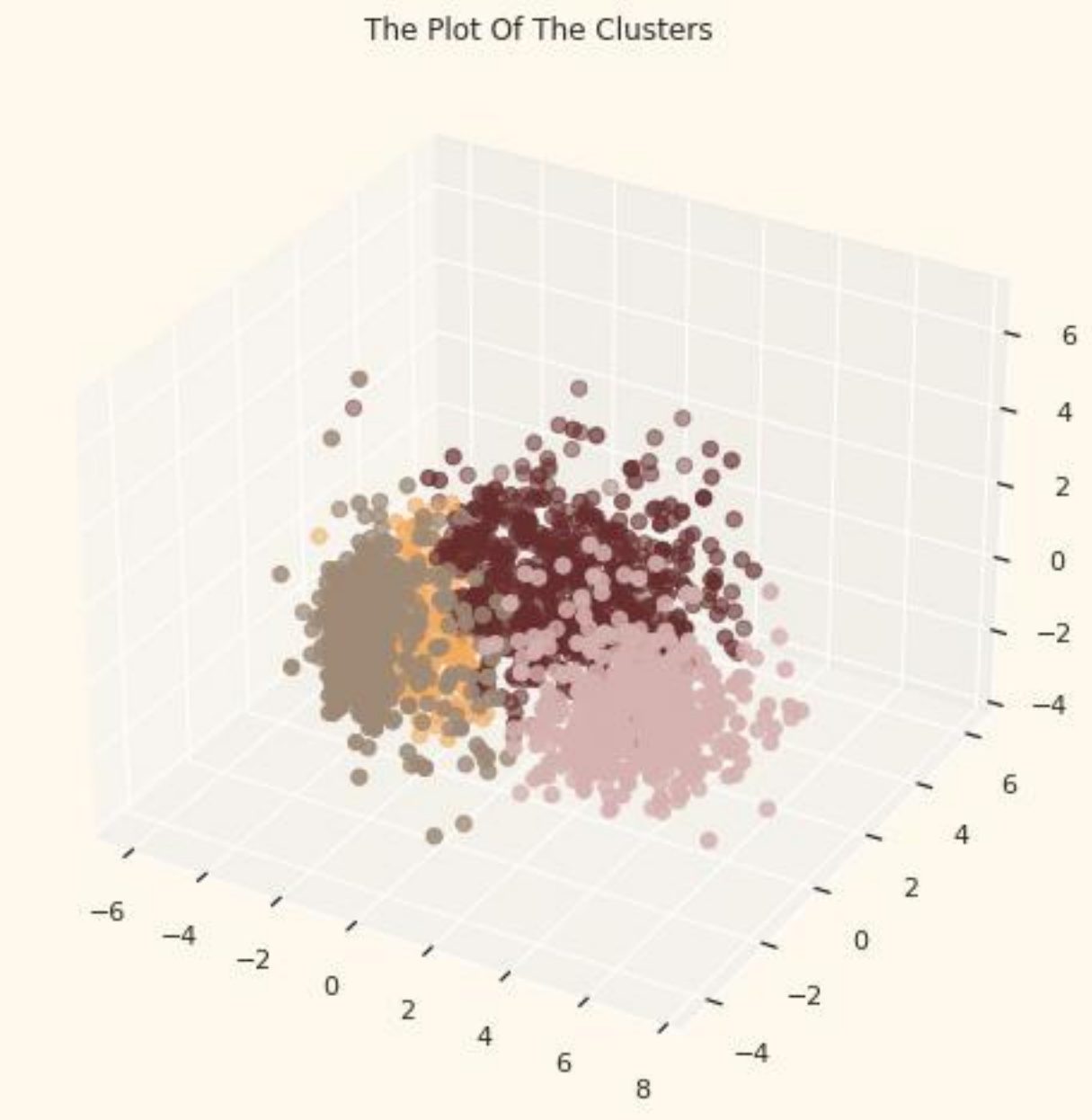
Steps involved in the Clustering:

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



#Plotting the clusters

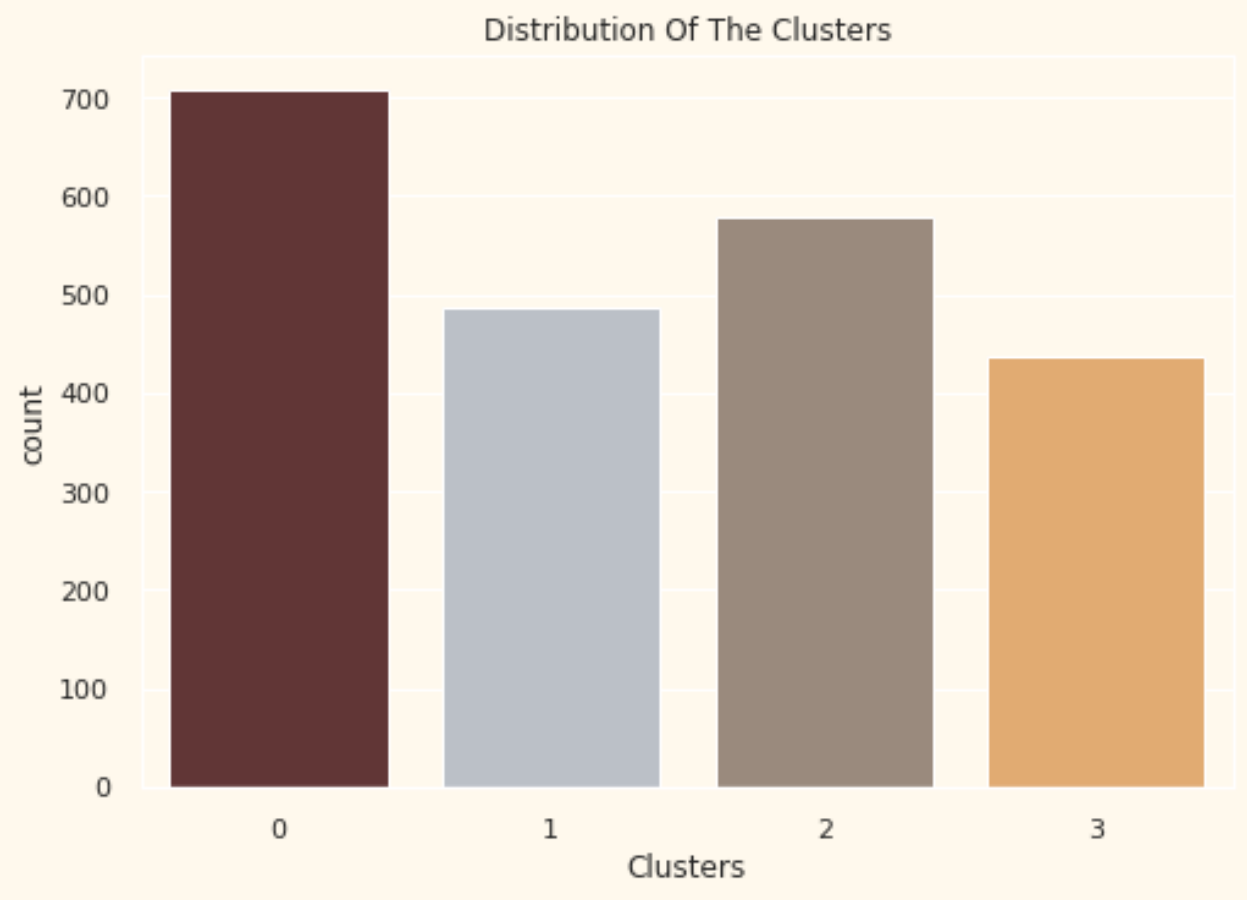
```
fig = plt.figure(figsize=(10,8))

ax = plt.subplot(111, projection='3d', label="bla")

ax.scatter(x, y, z, s=40, c=PCA_ds["Clusters"], marker='o', cmap = cmap )

ax.set_title("The Plot Of The Clusters")

plt.show()
```



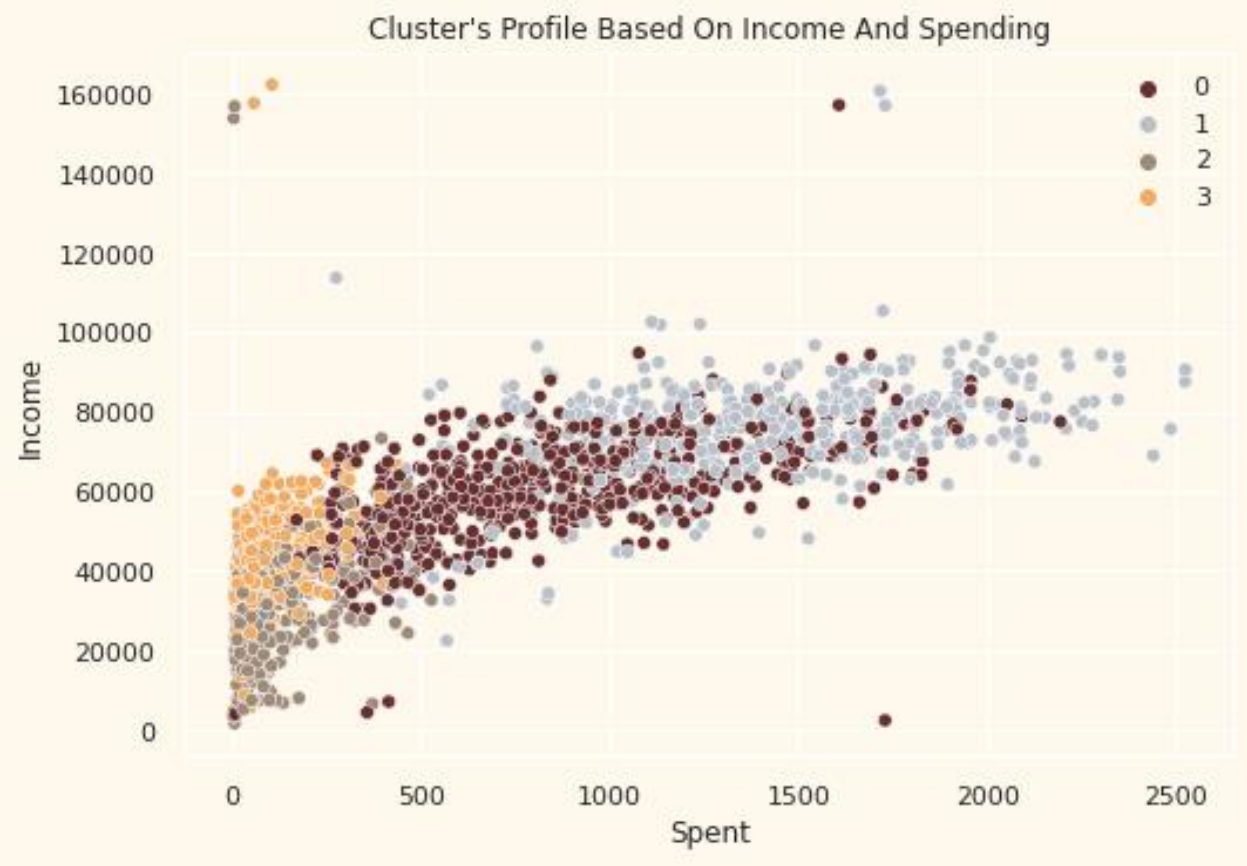
EVALUATING MODELS

Since this is an unsupervised clustering. We do not have a tagged feature to evaluate or score our model. The purpose of this section is to study the patterns in the clusters formed and determine the nature of the clusters' patterns.

For that, we will be having a look at the data in light of clusters via exploratory data analysis and drawing conclusions.



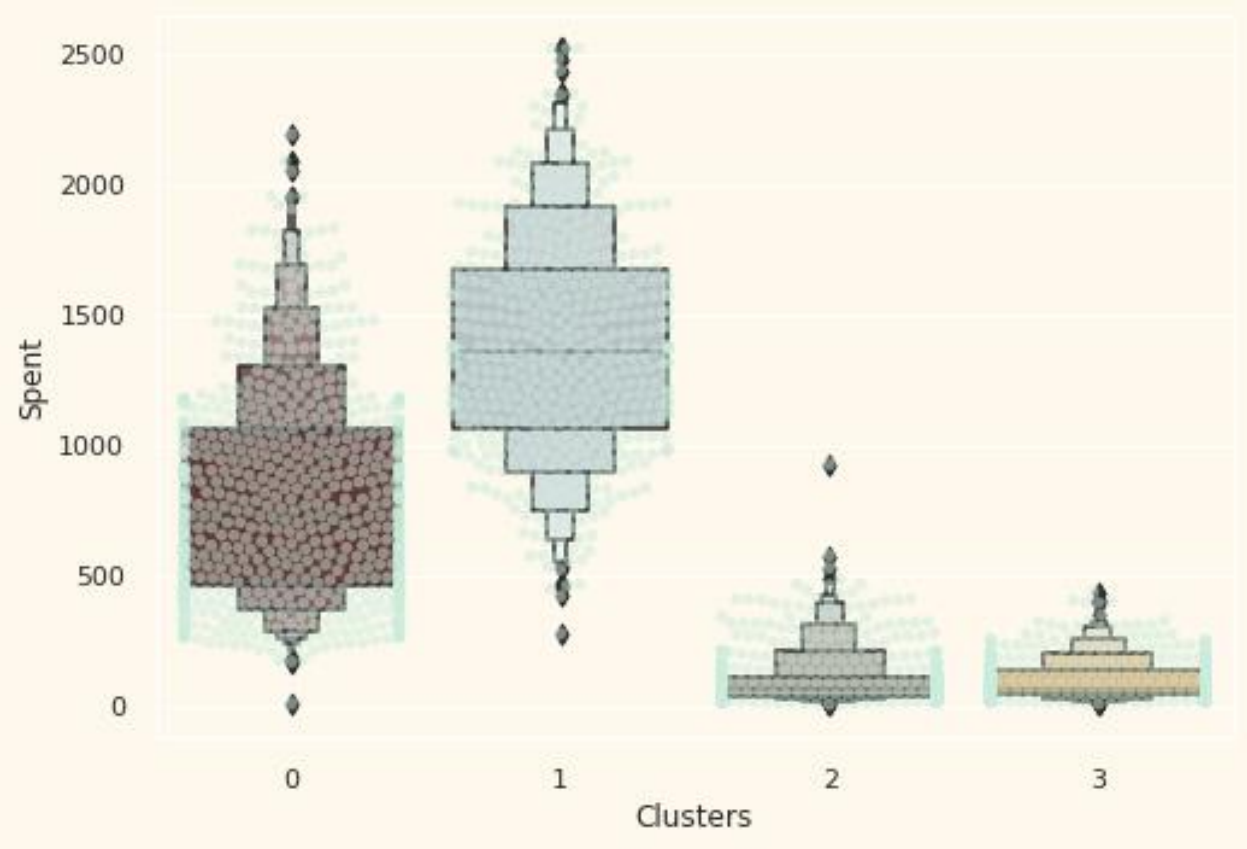
The clusters seem to be fairly distributed



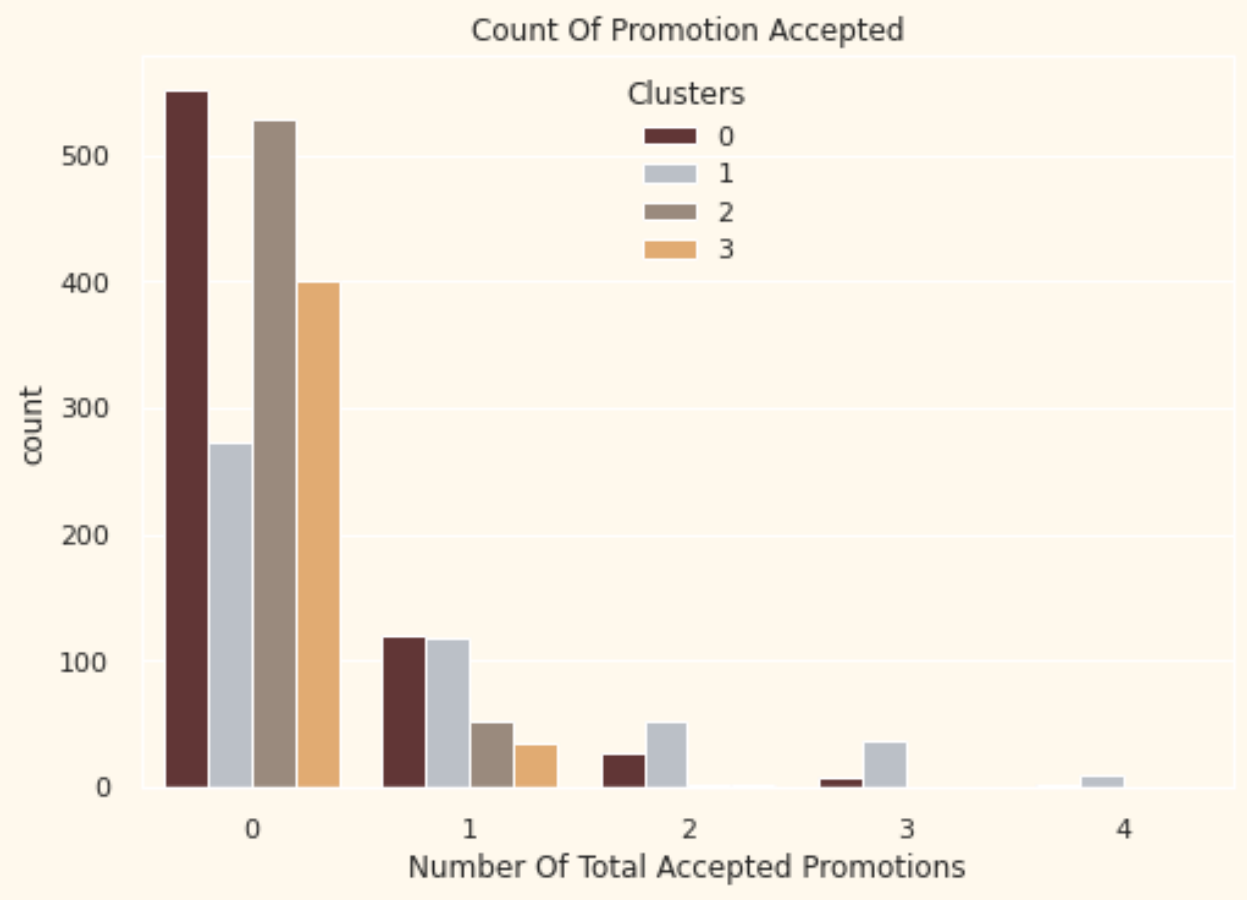
Income vs spending plot shows the clusters pattern

- group 0: high spending & average income
- group 1: high spending & high income
- group 2: low spending & low income
- group 3: high spending & low income

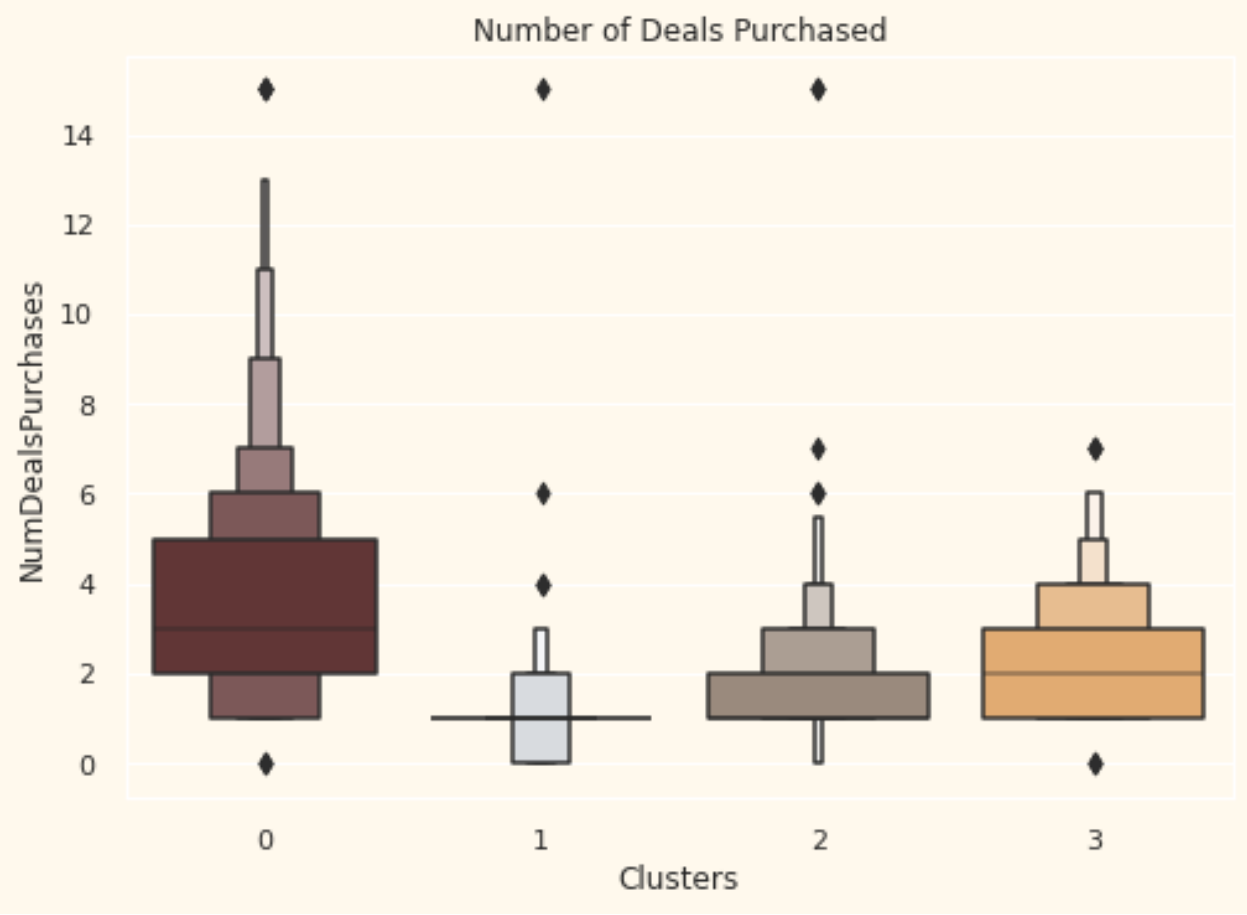
Next, I will be looking at the detailed distribution of clusters as per the various products in the data. Namely: Wines, Fruits, Meat, Fish, Sweets and Gold.



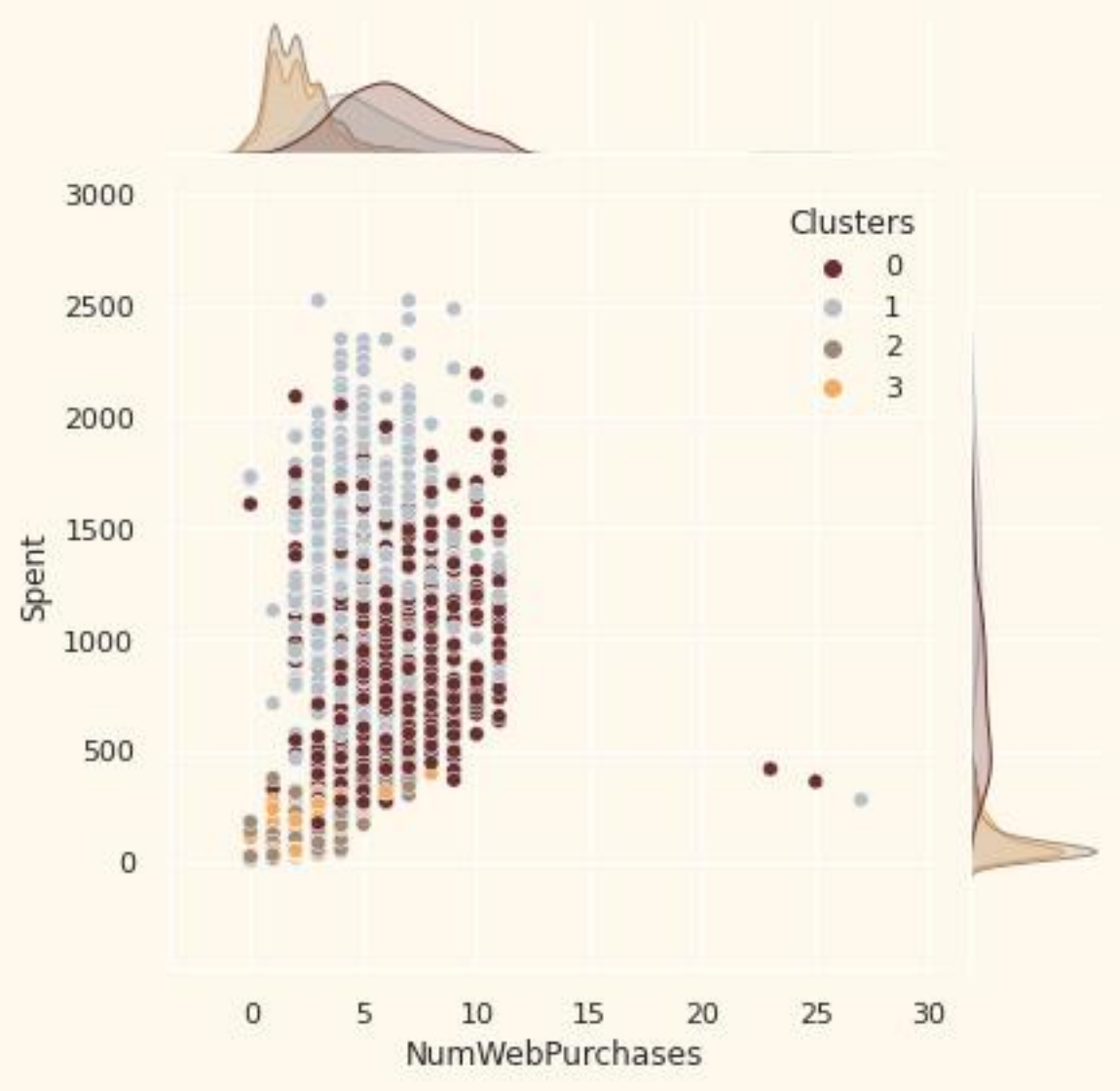
From the above plot, it can be clearly seen that cluster 1 is our biggest set of customers closely followed by cluster 0. We can explore what each cluster is spending on for the targeted marketing strategies.



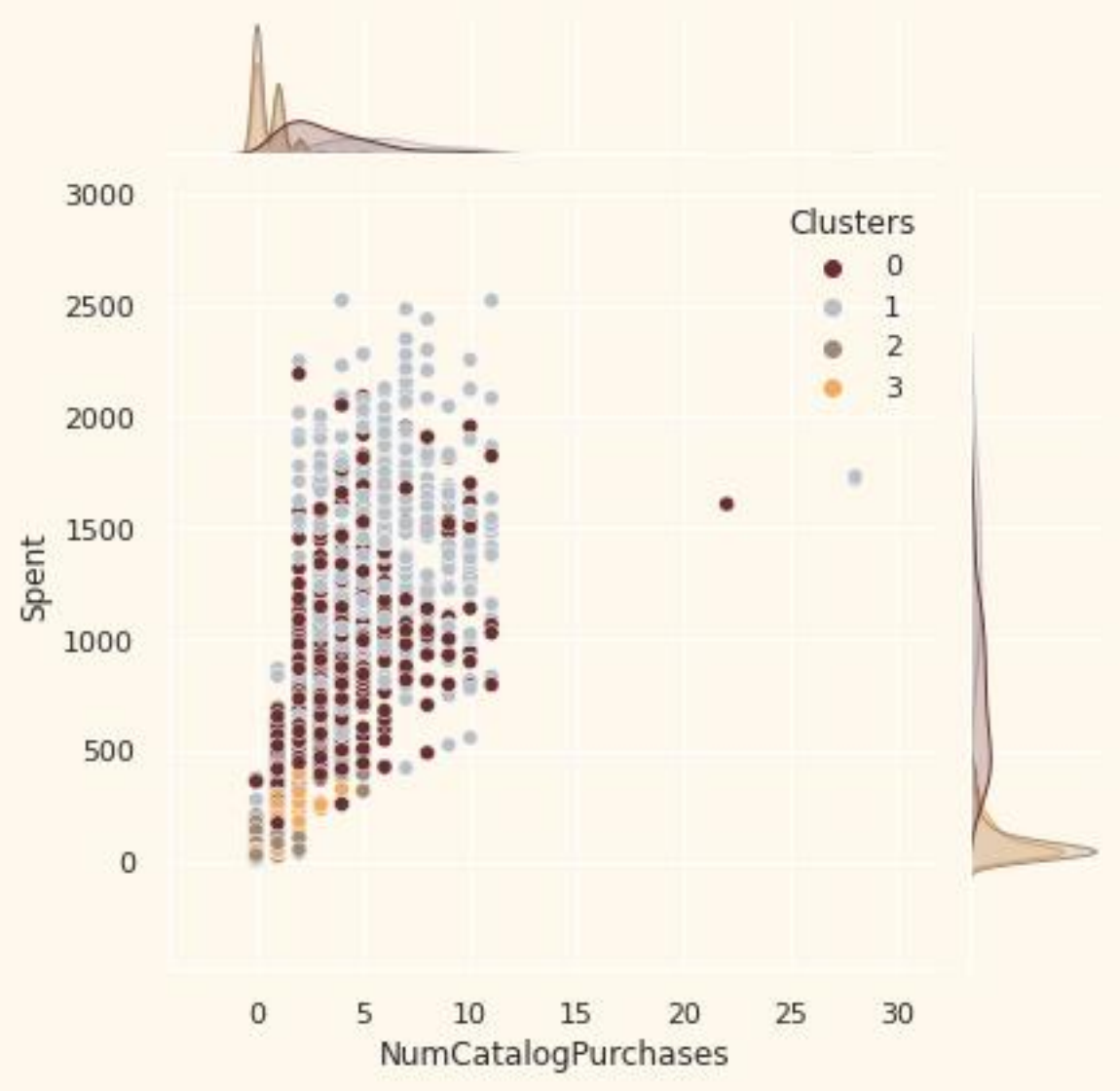
There has not been an overwhelming response to the campaigns so far. Very few participants overall. Moreover, no one part take in all 5 of them. Perhaps better-targeted and well-planned campaigns are required to boost sales.



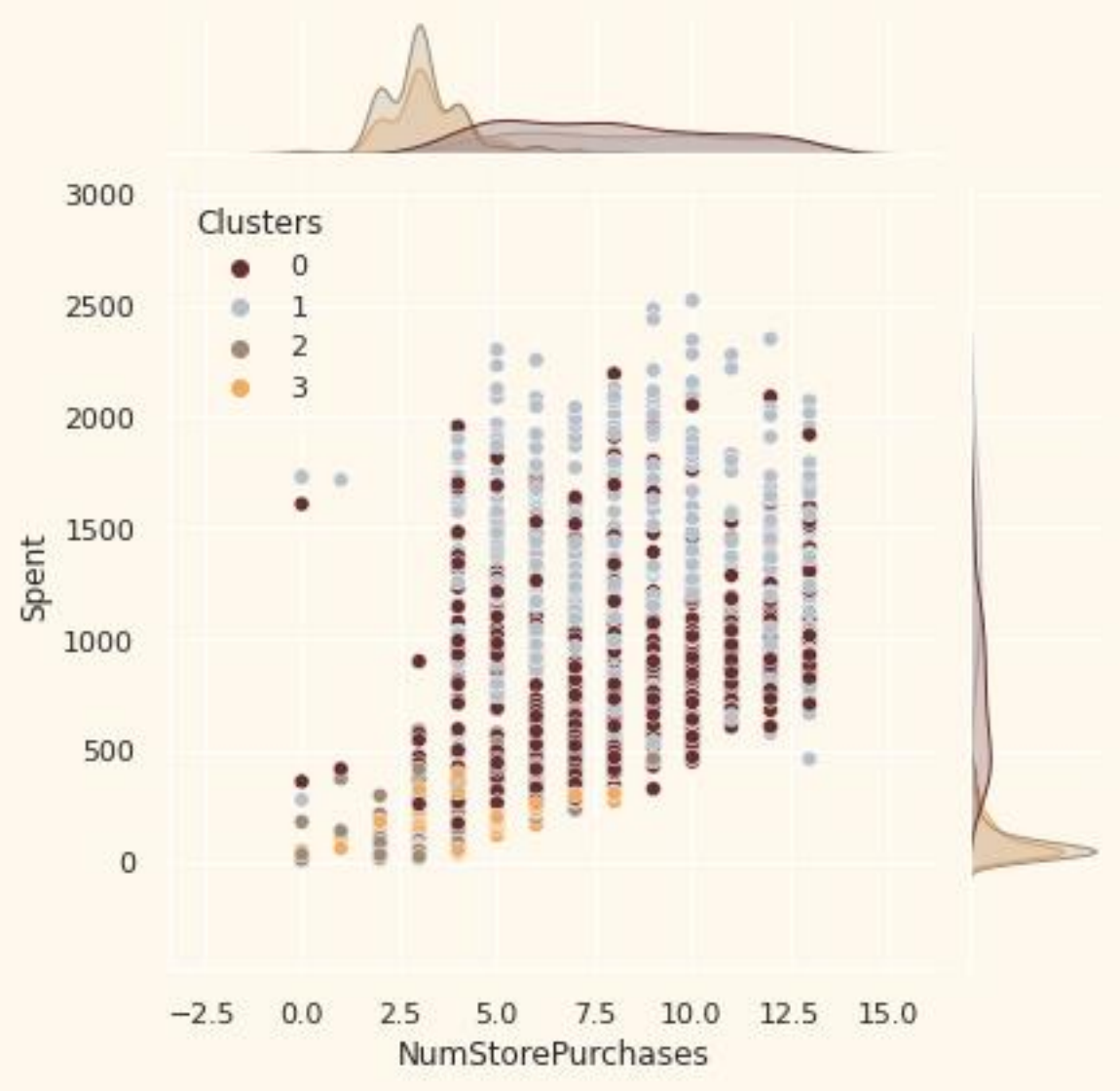
Unlike campaigns, the deals offered did well. It has best outcome with cluster 0 and cluster 3. However, our star customers cluster 1 are not much into the deals. Nothing seems to attract cluster 2 overwhelmingly



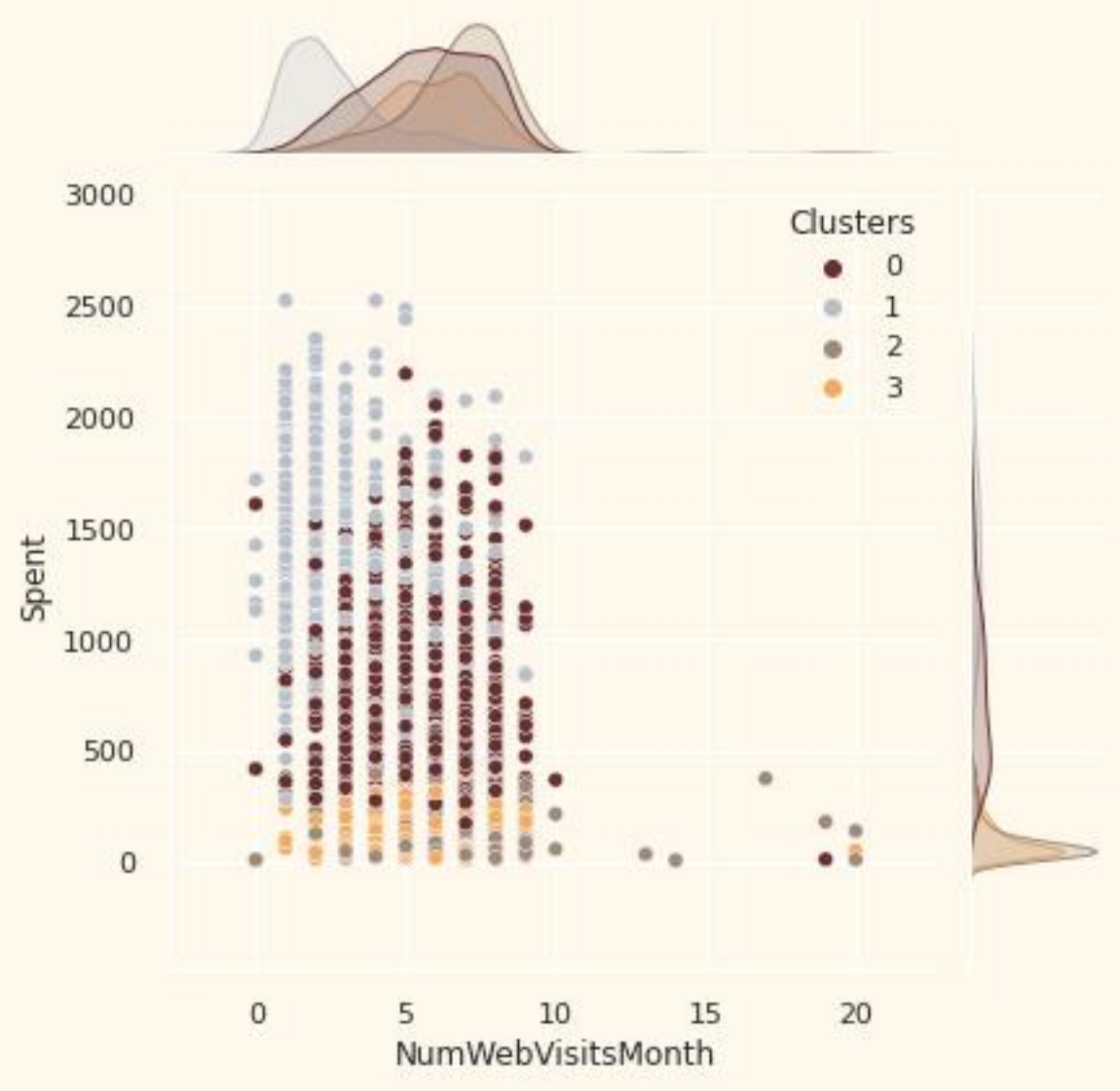
<Figure size 576x396 with 0 Axes>



<Figure size 576x396 with 0 Axes>



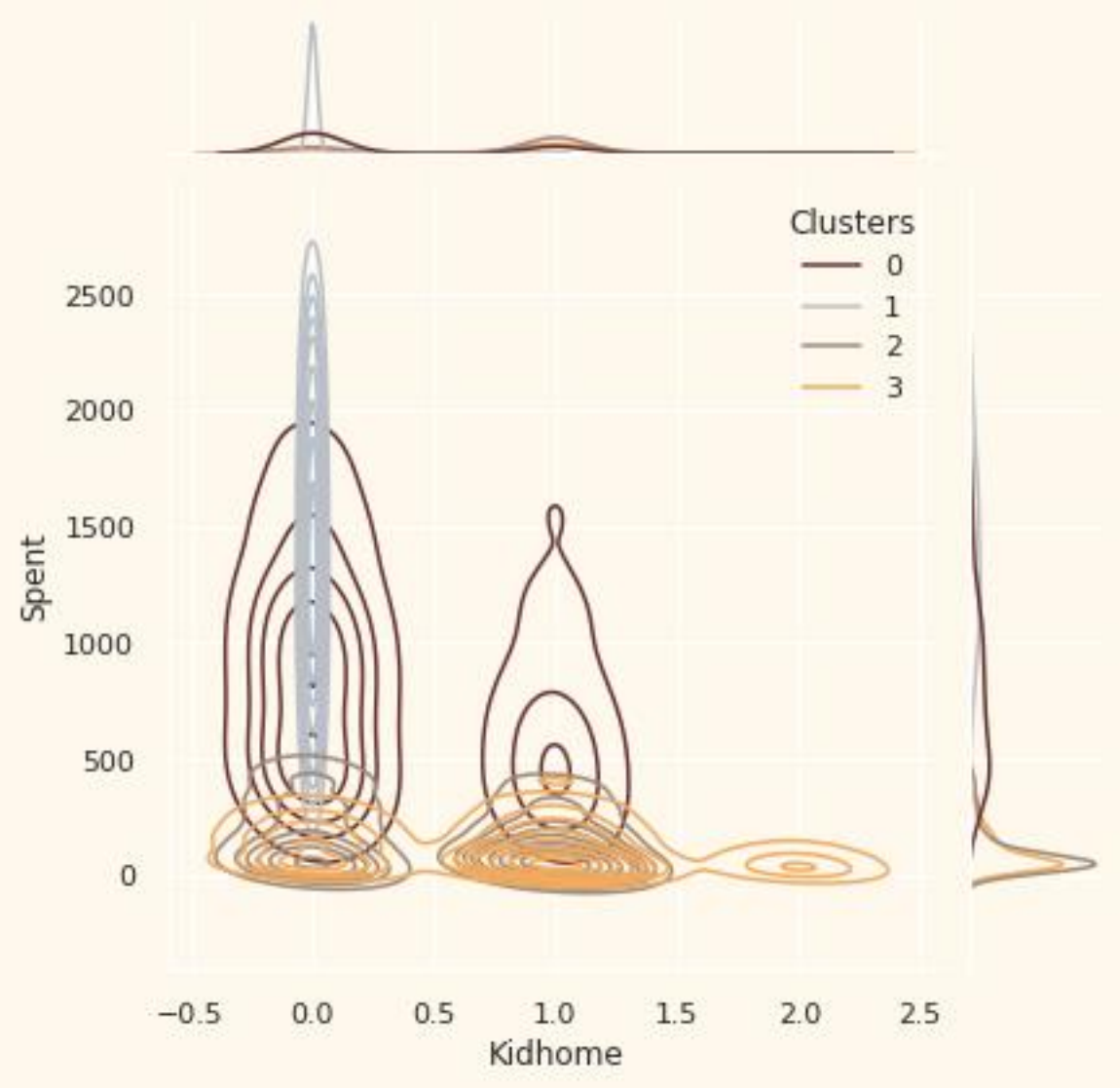
<Figure size 576x396 with 0 Axes>



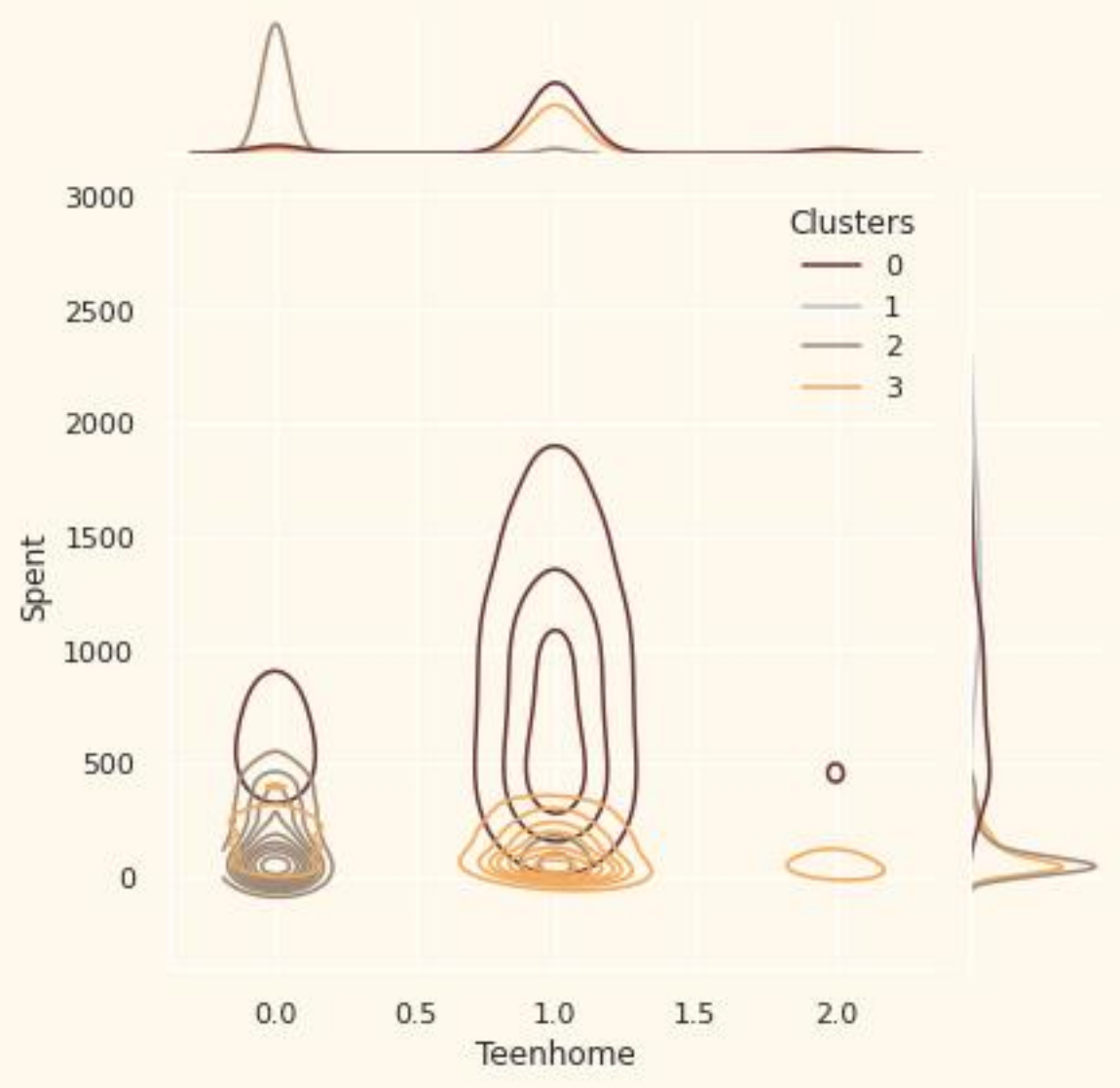
PROFILING

Now that we have formed the clusters and looked at their purchasing habits. Let us see who all are there in these clusters. For that, we will be profiling the clusters formed and come to a conclusion about who is our star customer and who needs more attention from the retail store's marketing team.

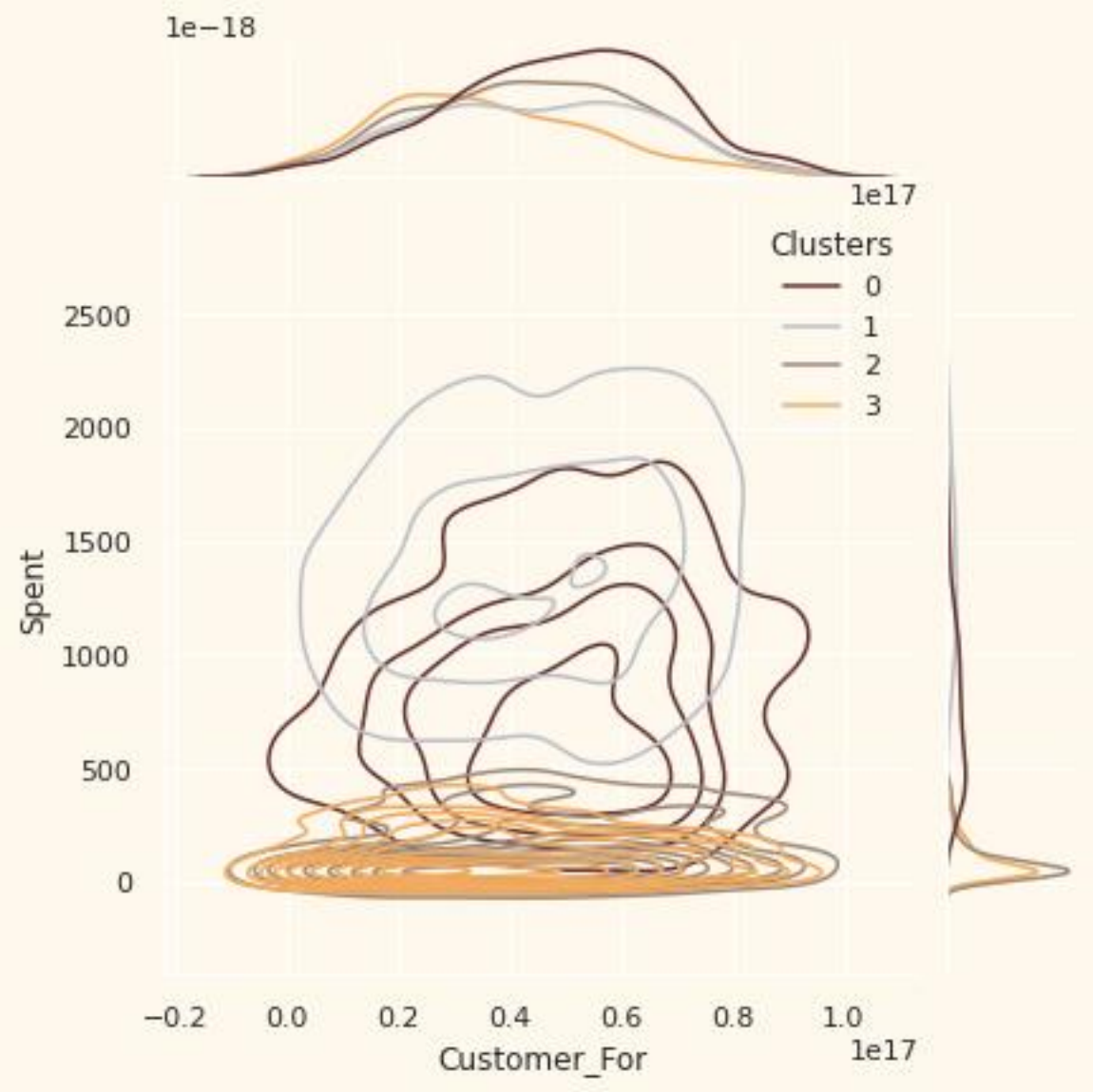
To decide that I will be plotting some of the features that are indicative of the customer's personal traits in light of the cluster they are in. On the basis of the outcomes, I will be arriving at the conclusions.



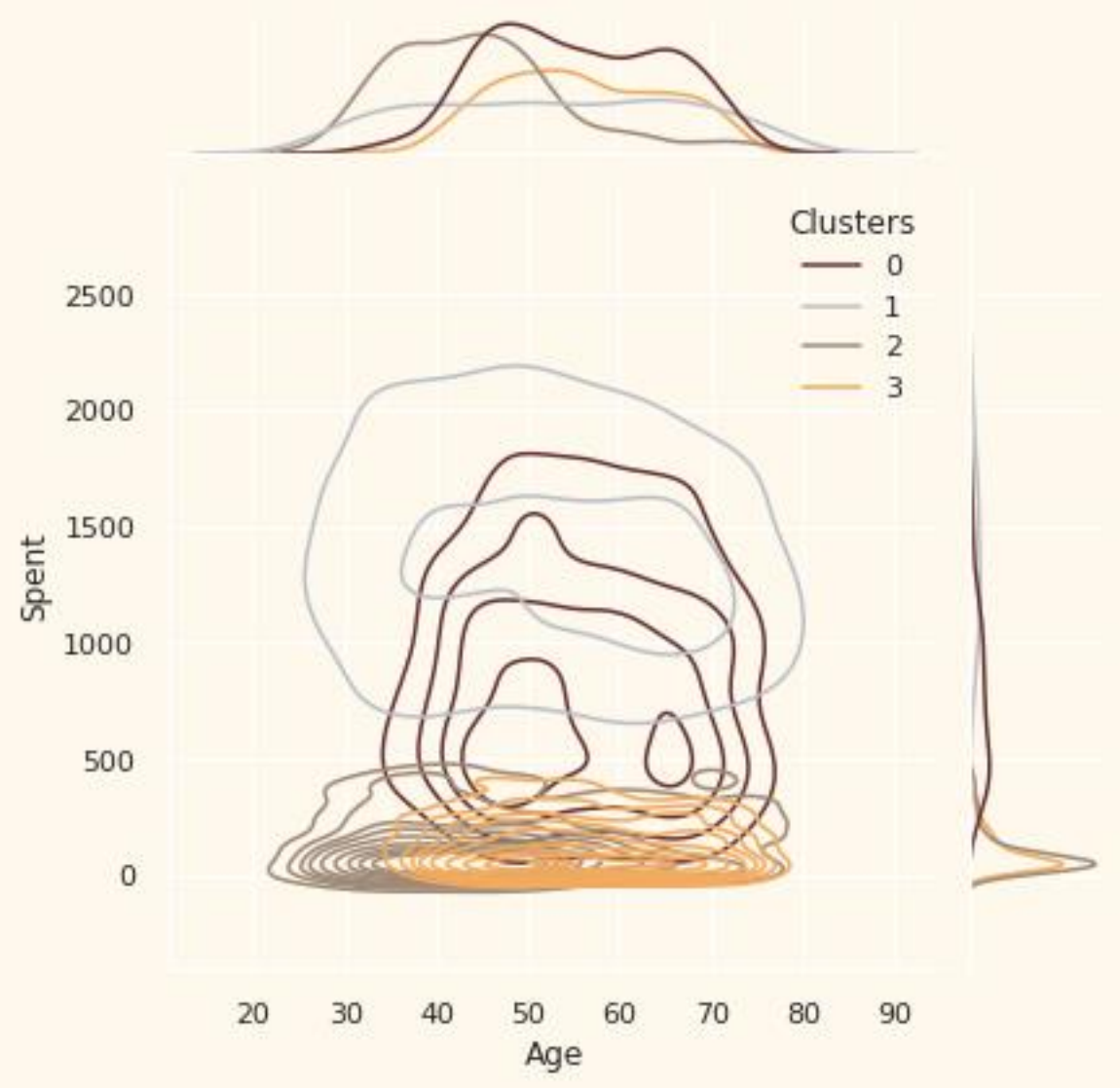
<Figure size 576x396 with 0 Axes>



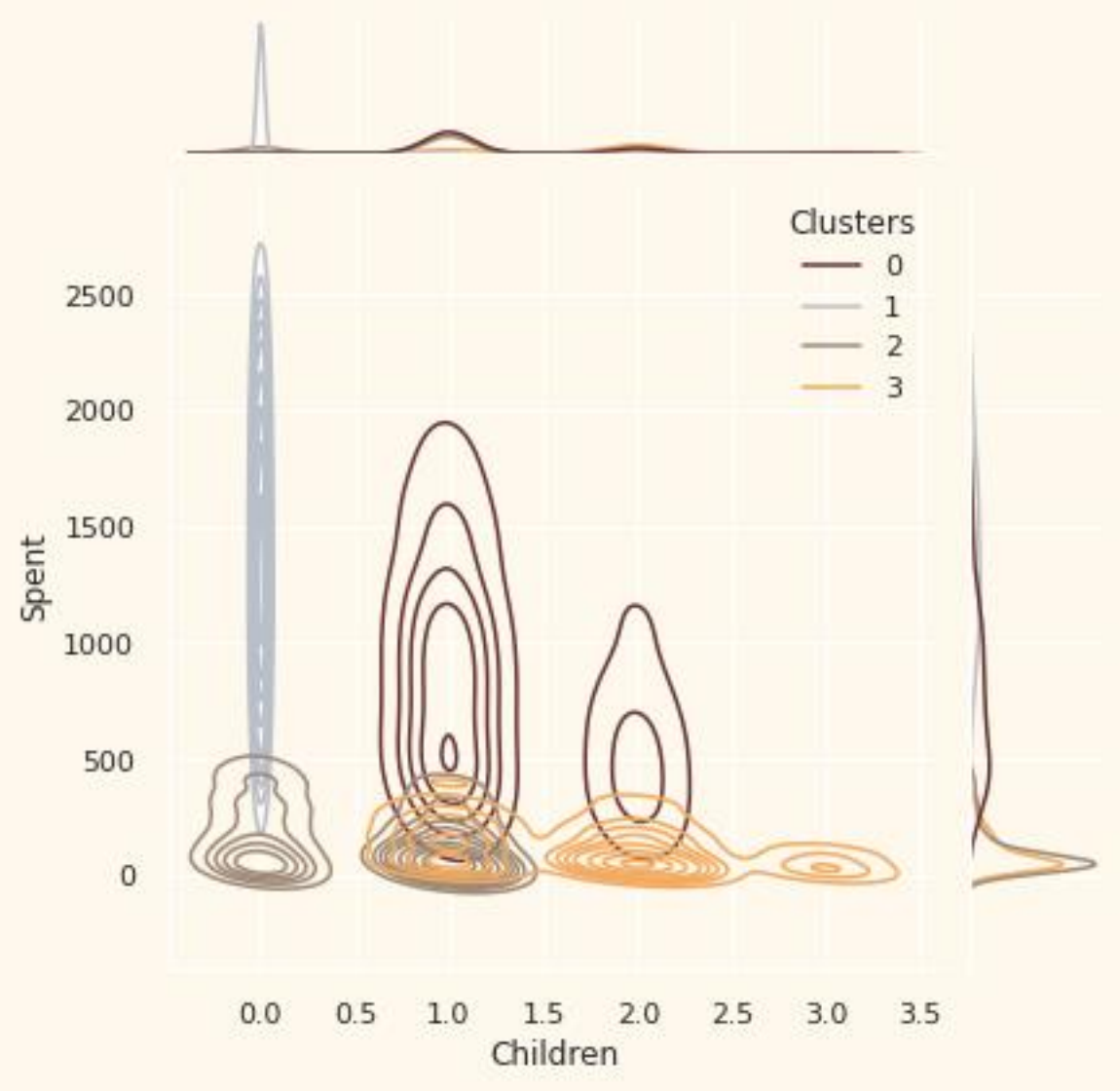
<Figure size 576x396 with 0 Axes>



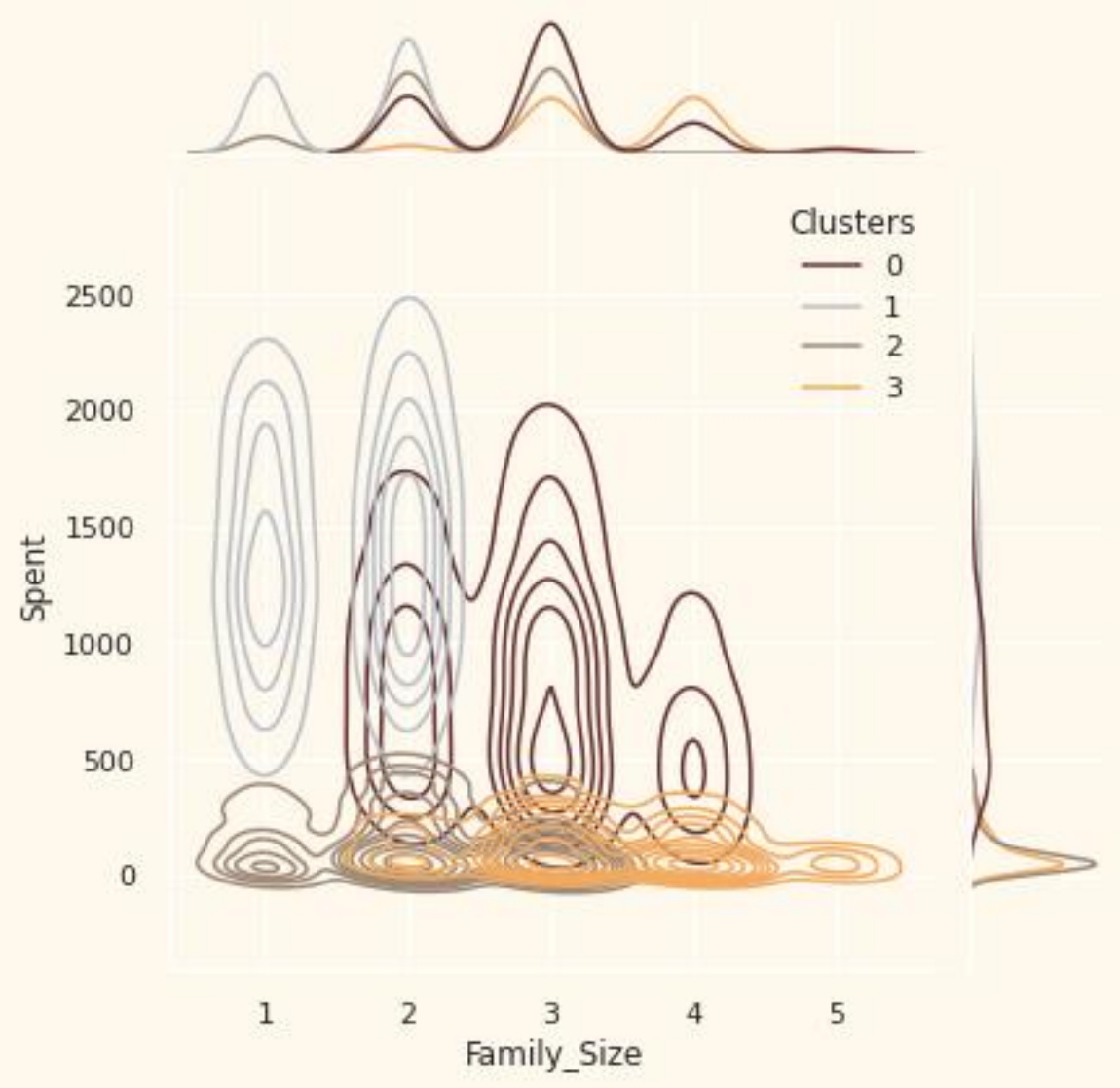
<Figure size 576x396 with 0 Axes>



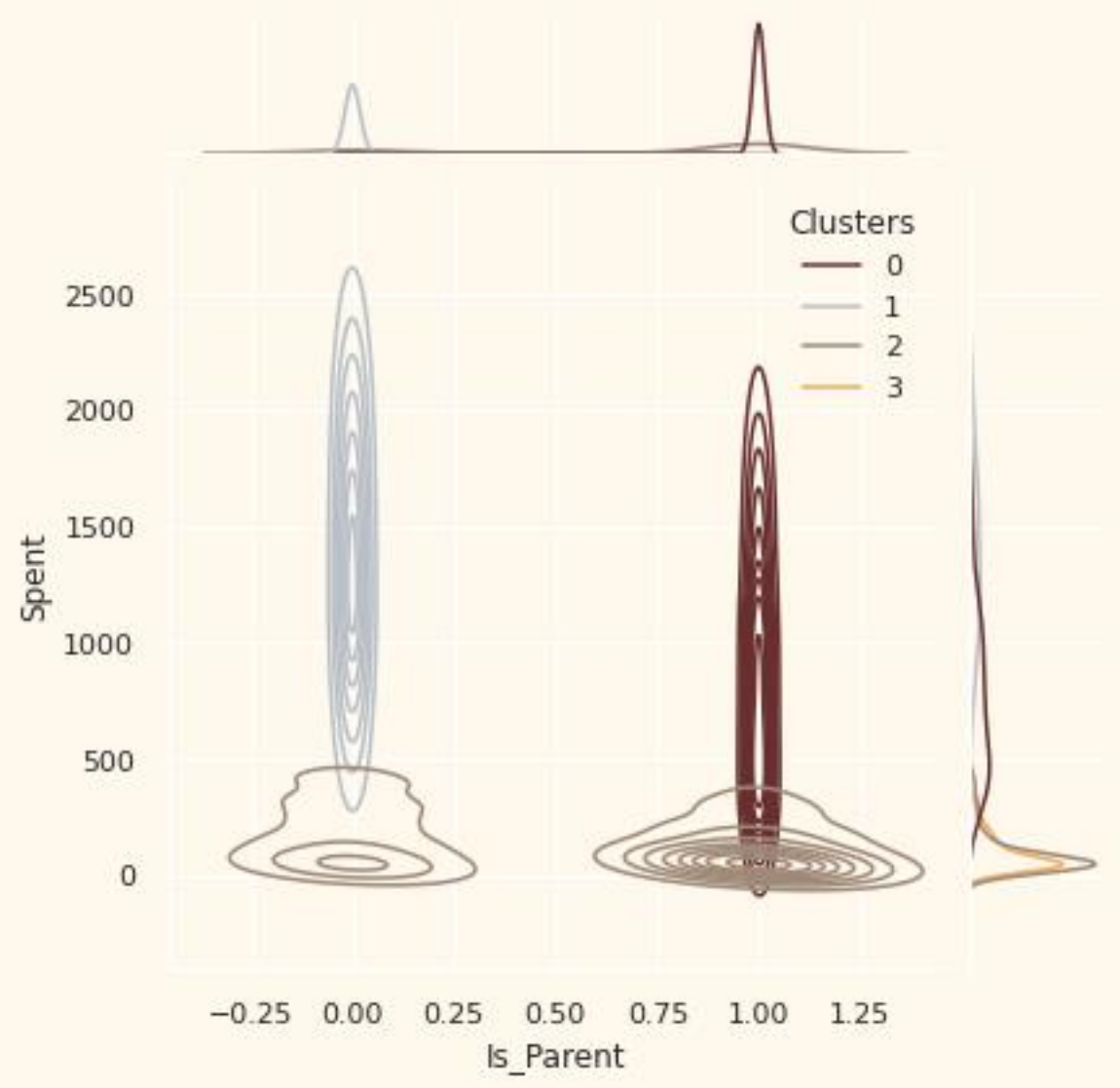
<Figure size 576x396 with 0 Axes>



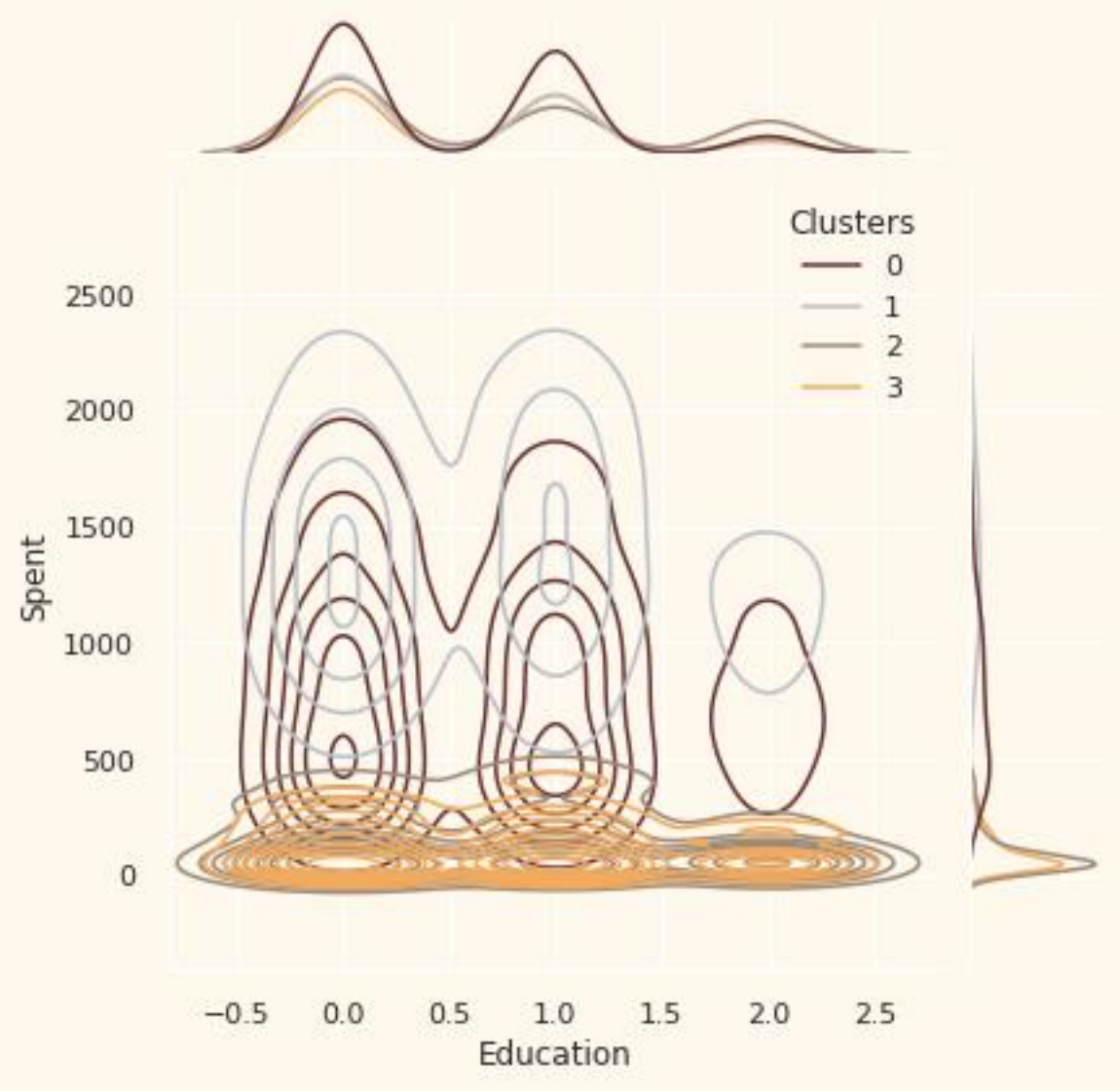
<Figure size 576x396 with 0 Axes>



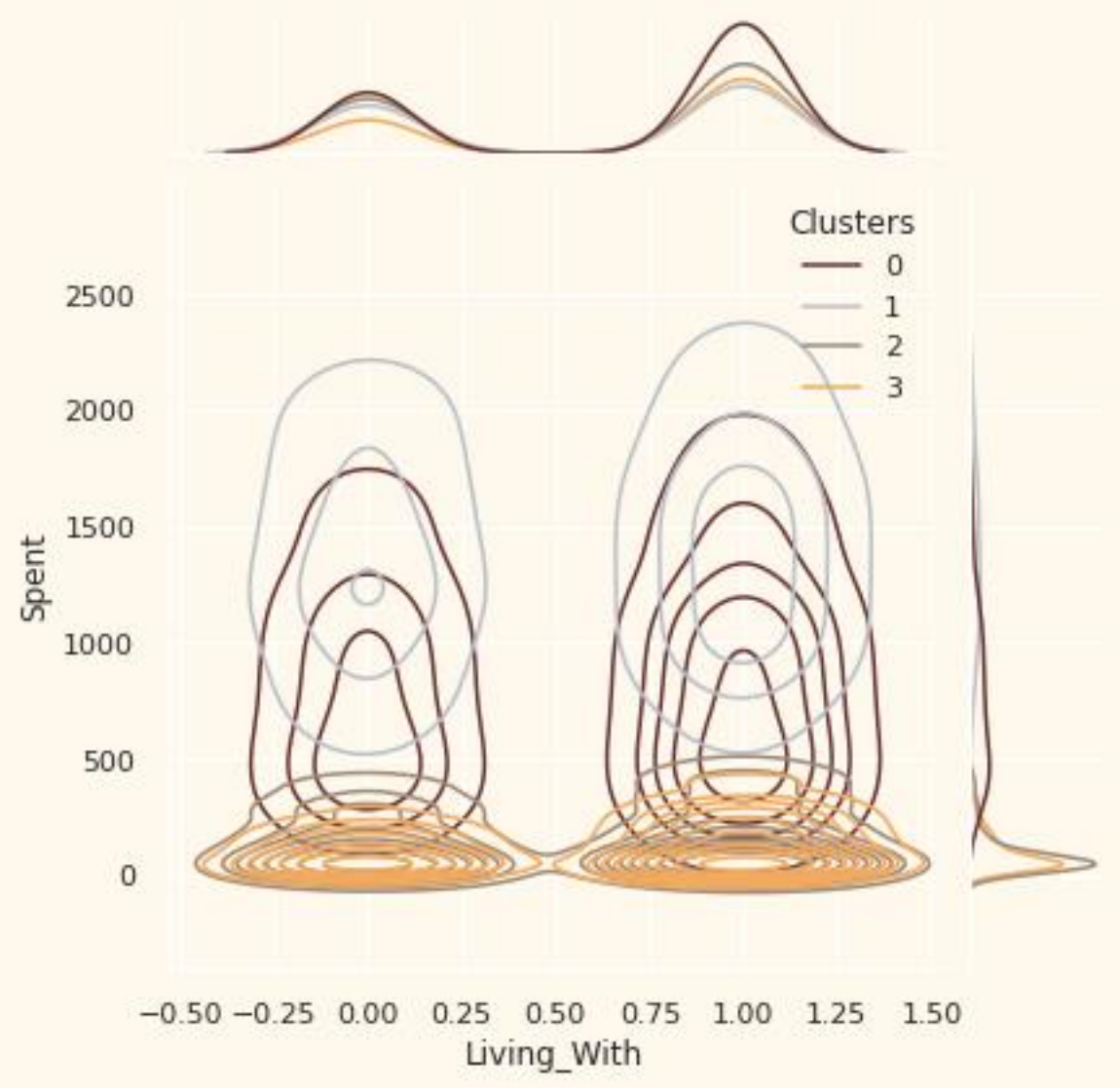
<Figure size 576x396 with 0 Axes>



<Figure size 576x396 with 0 Axes>

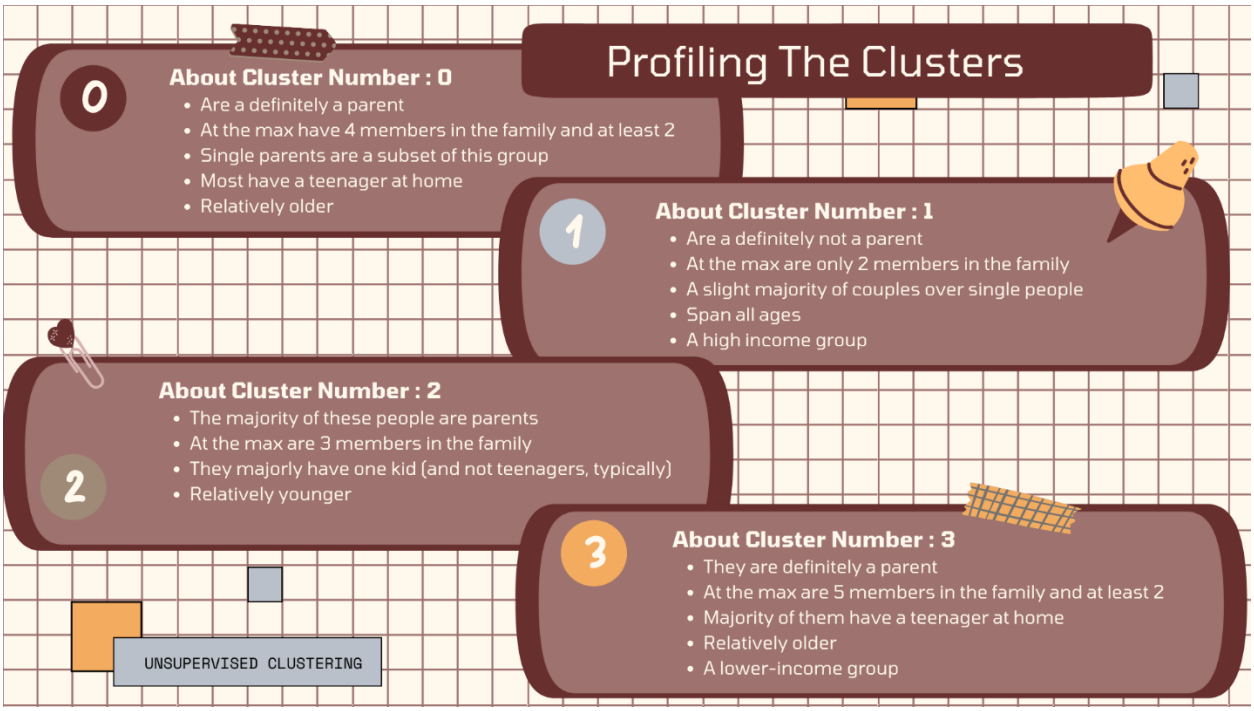


<Figure size 576x396 with 0 Axes>



Points to be noted:

The following information can be deduced about the customers in different clusters.



CONCLUSION

In this project, I performed unsupervised clustering. I did use dimensionality reduction followed by agglomerative clustering. I came up with 4 clusters and further used them in profiling customers in clusters according to their family structures and income/spending. This can be used in planning better marketing strategies.