

# CUSTOMER PERSONALITY ANALYSIS USING UNSUPERVISED LEARNING

Woodi Raghavendra Varun  
Graduate student  
Department of Data Science  
University of Massachusetts Dartmouth  
Dartmouth MA 02747  
Email: [wvarun@umassd.edu](mailto:wvarun@umassd.edu)

Pradyoth Singenahalli Prabhu  
Graduate student  
Department of Data Science  
University of Massachusetts Dartmouth  
Dartmouth MA 02747  
Email: [psingenahalliprabhu@umassd.edu](mailto:psingenahalliprabhu@umassd.edu)

## Abstract

Unsupervised learning is a type of machine learning algorithm that can be used to analyze customer data and identify common patterns and traits among a group of customers. By using clustering techniques to group similar customers together, companies can create customer segments with distinct personality characteristics, allowing them to tailor their marketing and sales efforts to better target each segment. Unsupervised learning algorithms can be trained on large amounts of data, making them powerful tools for companies looking to better understand and serve their customers.

## Introduction and Background

Customer personality analysis is the process of identifying and understanding the unique characteristics and traits that make up an individual customer's personality. This information can be used by companies to tailor their marketing and sales efforts to better target and serve each customer's specific needs and preferences.

Traditionally, customer personality analysis has been done manually by marketing and sales teams, who would use their expertise and experience to identify common patterns and trends among customers. However, with the advent of data mining and machine learning, it is now possible to automate this process using algorithms that can analyze large amounts of data and identify common patterns and traits among customers.

One type of machine learning algorithm that can be used for customer personality analysis is unsupervised learning. Unsupervised learning algorithms are trained on a large amount of data and can automatically detect patterns and similarities among customers without being explicitly told what to look for. This makes them particularly well suited for customer personality analysis, as they can uncover subtle differences and trends that might not be immediately apparent to humans.

The dataset for this project is a public dataset from Kaggle provided by Dr. Omar Romero-Hernandez. This dataset has 2,240 rows of observations and 28 columns of variables. Among the variables, there are 5-character variables and 23 numerical variables.

In this paper, we will discuss the use of unsupervised learning algorithms for customer personality analysis, and how they can be used by companies to better understand and serve their customers. We will also provide an overview of the different types of unsupervised learning algorithms and how they work, as well as discuss some of the challenges and limitations of using unsupervised learning for customer personality analysis.

## Methodology

### Data Preprocessing:

Data cleaning refers to identifying incomplete, incorrect, inaccurate, or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data. Data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database. Data cleansing can be done in batches using scripting or a data quality firewall, or it can be done interactively with data wrangling tools. Data cleaning is an essential step in the customer segmentation process, as it helps to ensure that the data used for analysis is accurate and reliable. To clean data for customer segmentation, you will need to identify and address any errors, inconsistencies, and missing values in your dataset. This can be done by manually reviewing the data, using visualizations, or applying statistical tests.

Steps taken for preprocessing:

1. The income column of the data frame has 24 missing values. The missing numbers are being dropped because it is fewer.
2. We are creating a feature that displays the length of time a customer has been in the company's database. This feature will be based on the "Dt Customer" data.
3. In order to improve the clarity and consistency of the data, I added a new column called "Living With" which has the same meaning as an existing column with a different name. The same approach was used for the "Education" data as well.
4. The "Is parent" feature has been implemented and it returns a value of 1 for parents and 0 for non-parents.
5. Add a new feature called "Spent" that displays the customer's overall spending across all categories over a two-year period.
6. Dropping some of the redundant features.
7. After plotting for "Income" and "Age", outliers are present which will be deleted
8. Doing label encoding for categorical features i.e., 'Education' and 'Living With'.

9. Dropping the columns of deals accepted and promotions, then scaling the remaining features using “Standard scaler”.

The data is quite clean, and the new features have been included.

PCA:

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

The final classification in this challenge will be based on a wide range of variables. These aspects or characteristics are essentially these factors. Working with it becomes more challenging the more features it has. The correlation between several of these features makes them unnecessary. For this reason, before running the features through a classifier, I shall reduce their dimensionality.

We are reducing the dimensions to 3. After this we will be plotting these data frame.

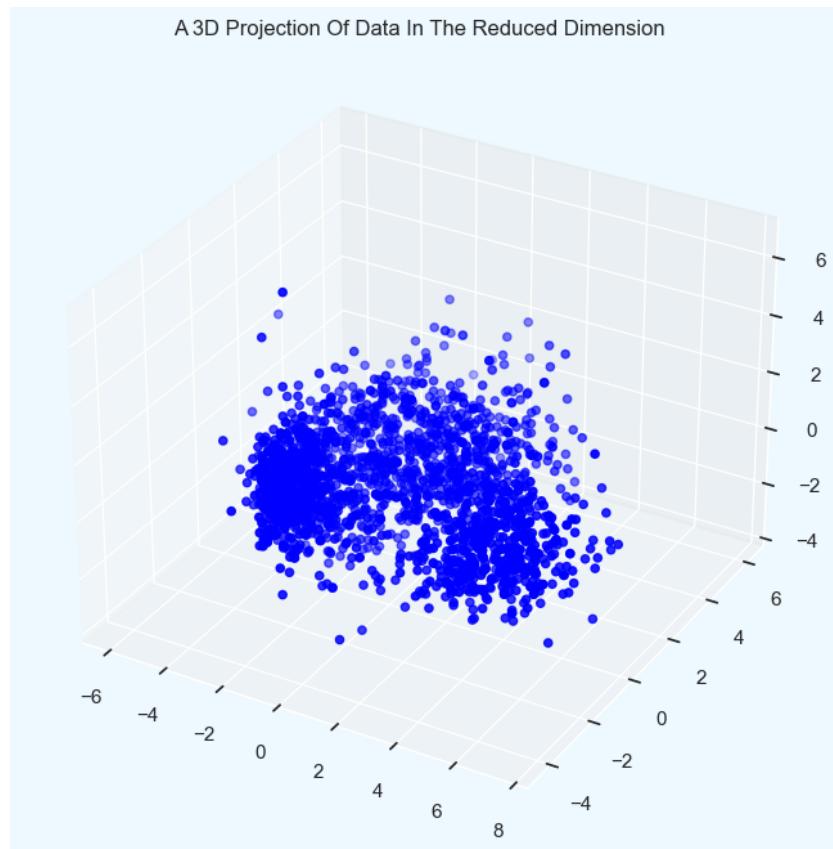


Figure 1: 3D plot of data after PCA

The plot above shows the data in three dimensions after it has been processed using principal component analysis (PCA). This projection allows us to visualize the data in a way that is easier to understand and interpret.

# Methods

## Hierarchical Clustering

We will be using hierarchical clustering in this project. The most typical hierarchical clustering method used to put objects in clusters based on their similarity is called agglomerative clustering. Another name for it is AGNES (Agglomerative Nesting). Each object is first treated as a singleton cluster by the algorithm. Once all clusters have been merged into a single large cluster containing all items, pairs of clusters are gradually combined. The outcome is a dendrogram, which is a tree-based representation of the objects. This would be useful for identifying distinct customer personality types and for targeted marketing and communication efforts to specific groups of customers.

## K-Means Clustering

K-means is a centroid-based clustering algorithm where each data point is assigned to a cluster based on the distance between it and a centroid. Finding the K number of groups in the dataset is the objective. The goal of K-means clustering, a vector quantization technique that originated in signal processing, is to divide n observations into k clusters, where each observation belongs to the cluster that has the closest mean, which acts as the cluster prototype. With the use of k-means we will be finding clusters. In total we will be selecting 4 clusters.

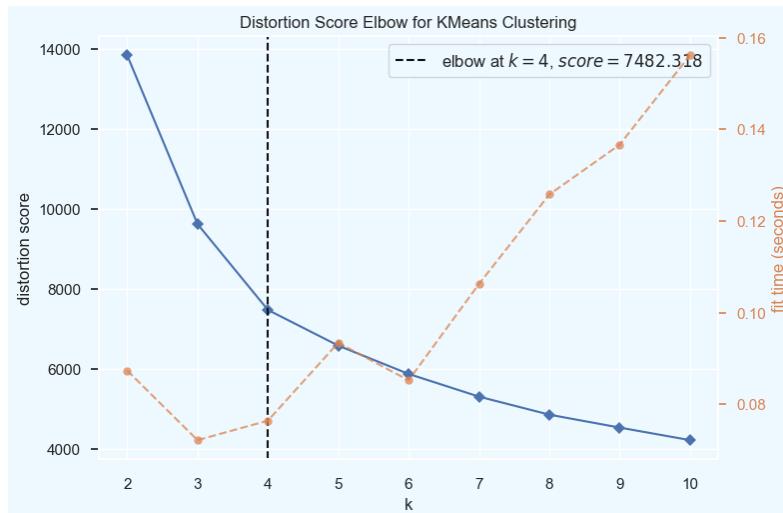


Figure 2: Elbow method visualization

After reducing the attributes to three dimensions, I will apply Agglomerative clustering to perform the clustering. A hierarchical clustering technique is agglomerative clustering. Up until the appropriate number of clusters is reached, examples are merged. First, we will be using elbow method to determine the number of clusters. Once we have got the clusters then we will apply hierarchical clustering based on the PCA dataset what we had created earlier.

Finally, clusters will be formed then we will plot the result.

# Results

Cluster formed using Hierarchical Clustering:

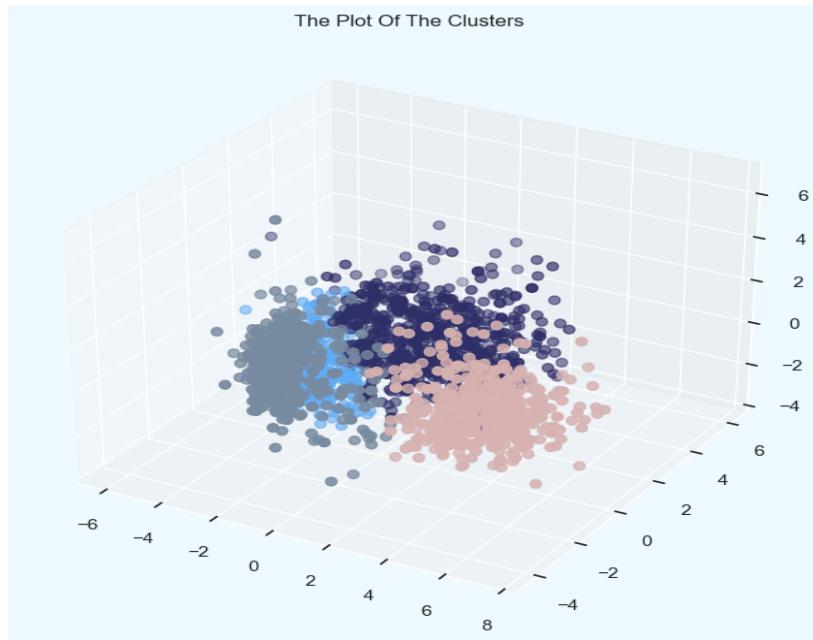


Figure 3: 4 cluster formed

The plot above demonstrates the results of applying hierarchical clustering to the data, which has resulted in the formation of four clusters.

Count of each cluster:

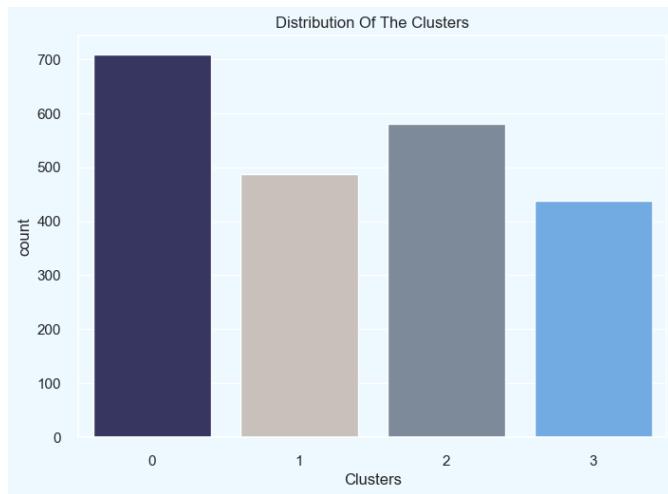


Figure 4: Count of each cluster

The plot above shows the distribution of data points within each cluster, as determined by a clustering algorithm.

Profile of a group of individuals based on their income and spending habits:

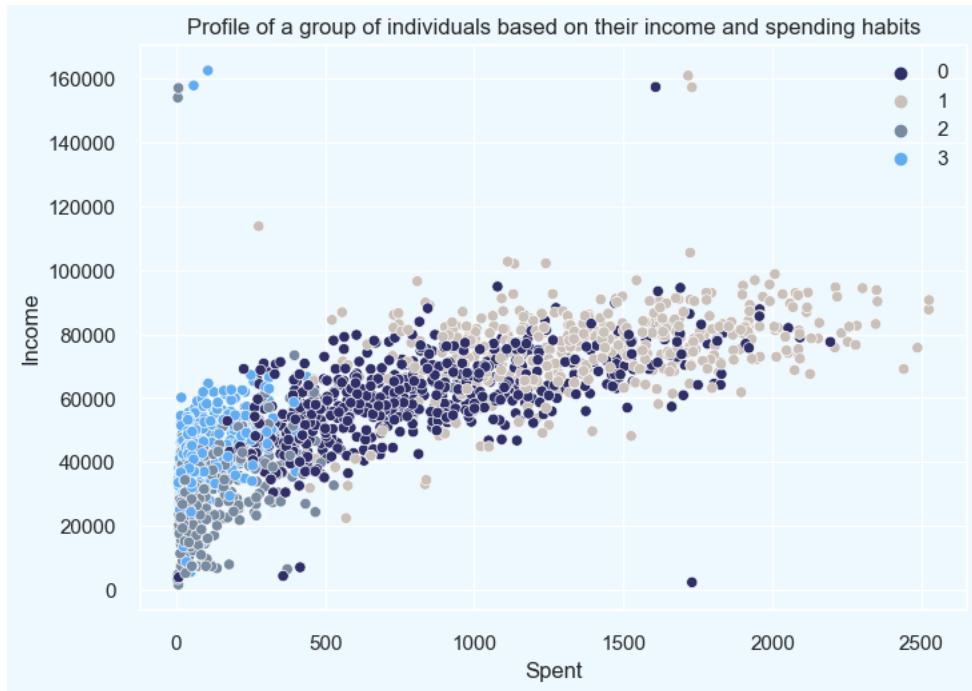
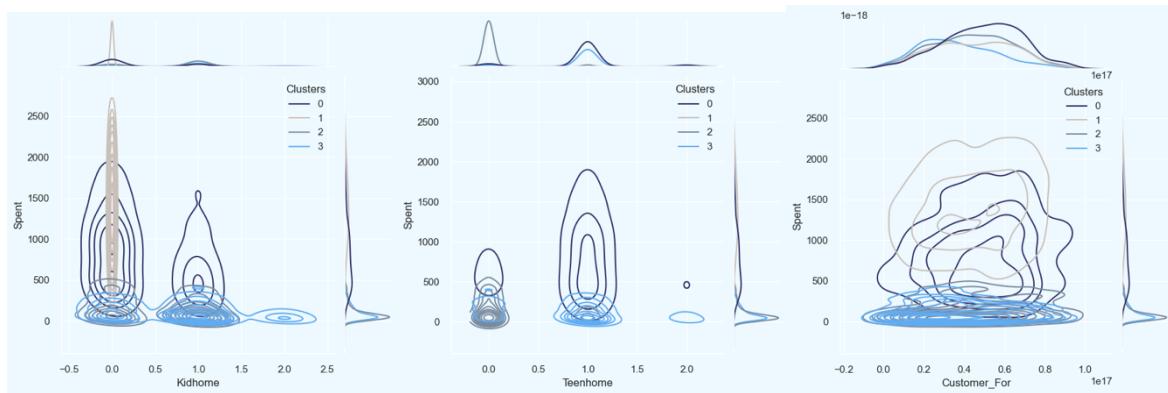


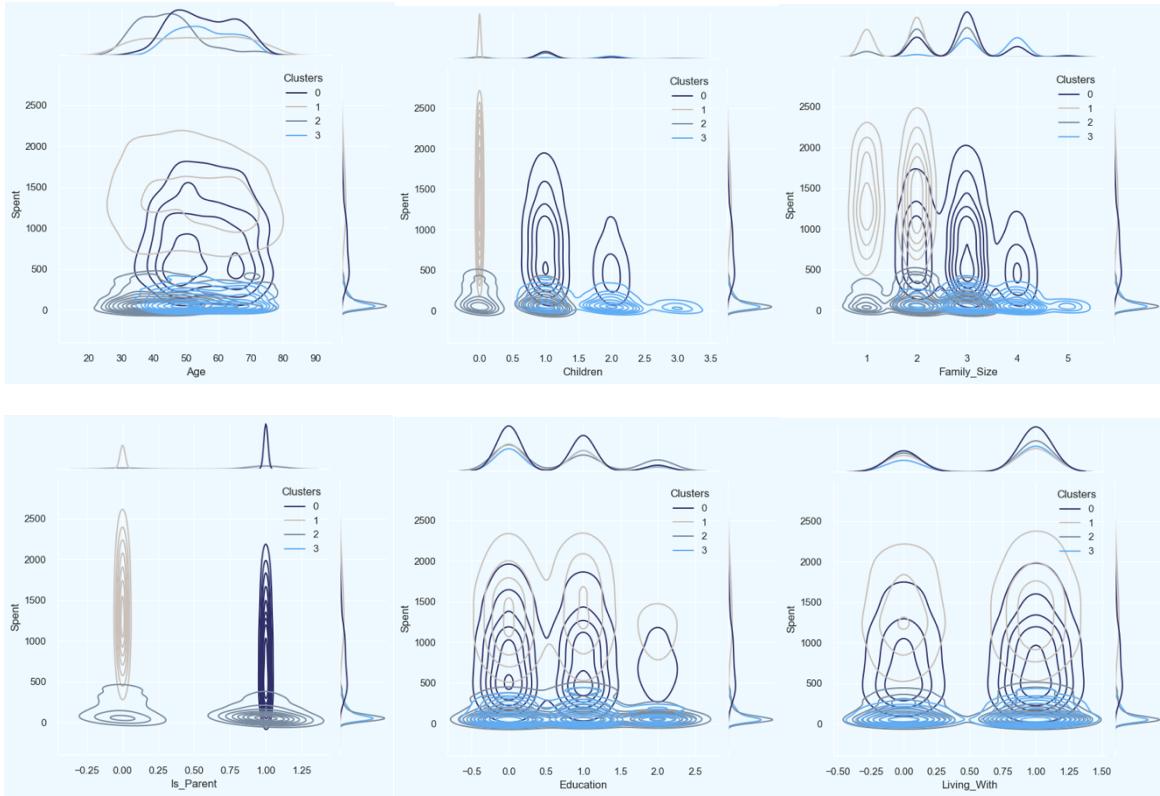
Figure 5: Income vs Spent

Income vs spending plot shows the clusters pattern:

- Group 0: those with average income and high spending
- Group 1: those with high income and high spending
- Group 2: those with low income and low spending
- Group 3: those with low income and high spending

Profiling Customers:





*Figure 6: Profiling Customers*

Below is the Analysis of Cluster Numbers 0-3

#### About Cluster Number: 0

As shown in the chart, it can be concluded that the group being discussed is a group of parents who are relatively old and have a family with a maximum of four members and at least two members. Most of the parents in this group have a teenager at home, and single parents are a subset of this group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

#### About Cluster Number: 1

According to the data in the charts, it can be concluded that the group being discussed is a group of non-parents who have a maximum of two members in their families. The majority of this group consists of couples, rather than single people. The members of this group span a wide range of ages and are part of a high-income group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

#### About Cluster Number: 2

The charts indicate, it can be concluded that the group being discussed is a group of parents who are relatively younger and have families with a maximum of three members. Most of these parents have one child, who is typically not a teenager. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

#### About Cluster Number: 3

The data in the charts suggests, it can be concluded that the group being discussed is a group of parents who are relatively older and have a family with a maximum of five members and at least two members. Most of these parents have a teenager at home, and they are part of a lower-income group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

## Discussion and Conclusion

The objective of a project on customer personality analysis using unsupervised learning would be to develop and implement a machine learning model that can uncover hidden insights and patterns in customer behavior data. The goal of the project would be to use this model to better understand customer personalities and preferences, and to use this information to improve the overall customer experience and drive business success. This could involve training the model on a large dataset of customer behavior data, using techniques like clustering or dimensionality reduction to identify patterns and trends in the data. The resulting model could then be used to analyze new data and make predictions about customer personalities. The project could also involve developing tools and processes for integrating the model into business operations, and for using the insights it generates to inform marketing and sales efforts.

In conclusion, unsupervised learning can be an effective method for customer personality analysis. By using clustering algorithms and dimensionality reduction techniques, you can identify patterns and relationships in customer data and predict customer personality traits based on their behavior and characteristics. This can provide valuable insights into your customer base and help you tailor your marketing and customer service strategies to better serve their needs and preferences. Potential application of unsupervised learning in customer personality analysis is in the use of virtual assistants and chatbots. By incorporating the results of customer personality analysis into their algorithms, these systems could provide more personalized and effective interactions with customers, potentially leading to improved customer satisfaction and loyalty.

There are many challenges associated with customer personality analysis. Data quality is an important factor. Accurate and reliable customer personality predictions require high-quality data that is free from errors and biases. Limitations of customer personality analysis are possible. Dependence on data is one of them. The accuracy and reliability of customer personality predictions using unsupervised learning algorithms are highly dependent on the quality and quantity of data available for analysis.

In some cases, the data may be limited or incomplete, which could limit the ability of the algorithm to make accurate predictions.

Overall, the future of customer personality analysis using unsupervised learning is likely to involve the development of increasingly sophisticated algorithms and techniques for uncovering hidden patterns and relationships in customer data, and the integration of these insights into a wide range of business applications.

## References:

Julien Ah-Pine (2018). *An Efficient and Effective Generic Agglomerative Hierarchical Clustering Approach*, 19(42):1–43, 2018. URL: <https://www.jmlr.org/papers/volume19/18-117/18-117.pdf>

Margareta Ackerman and Shai Ben-David (2016). *A characterization of linkage-based hierarchical clustering*. *Journal of Machine Learning Research*, 17:1–17, 2016. URL: <https://www.jmlr.org/papers/volume17/11-198/11-198.pdf>

D T Pham, S S Dimov, and C D Nguyen (2004). *Selection of K in K-means clustering*. URL: <https://www.ee.columbia.edu/~dpwe/papers/PhamDN05-kmeans.pdf>

### Other References:

- Python 3: <https://www.python.org/>
- PCA: <https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>
- Pandas: <https://pandas.pydata.org/>
- Numpy: <https://numpy.org/>
- Seaborn: <https://seaborn.pydata.org/>
- Matplotlib: <https://matplotlib.org/>
- Elbow: <https://www.scikit-yb.org/en/latest/>
- Label Encoding: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>
- Standard Scaler: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>
- K-Means Clustering: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>
- Hierarchical Clustering: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>
- Data Set: <https://www.kaggle.com/datasets/imakash3011/customer-personality-analysis?datasetId=1546318&sortBy=voteCount>

# Appendix

## Importing Libraries

```
In [1]: import sys
import warnings

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import colors
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from yellowbrick.cluster import KElbowVisualizer

if not sys.warnoptions:
    warnings.simplefilter("ignore")
np.random.seed(42)
```

## Importing Dataset

```
In [2]: #Loading the dataset
data = pd.read_csv("data/marketing_campaign.csv", sep="\t")
print("No. of datapoints:", len(data))
print("No. of columns:", len(data.columns))
data.head()
```

No. of datapoints: 2240

No. of columns: 29

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

5 rows × 29 columns

## Data Preprocessing

```
In [3]: data.info()
```

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

### Observations:

1. There are null values in the income column of our data frame, and the "Dt\_Customer" column is not formatted as a date.
2. Additionally, some of the columns in our data frame are categorical, and will need to be encoded into numeric form before they can be used in further analysis.

```
In [4]: # Removing null values
```

```
data = data.dropna()
print(f"After removing the null values data contains {len(data)}")
```

```
After removing the null values data contains 2216
```

In the next step, I will create a new feature "**Dt\_Customer**" that indicates the number of days a customer has been registered in the firm's database. To keep things simple, I will make this value relative to the most recent customer in the record.

```
In [5]: data["Dt_Customer"] = pd.to_datetime(data["Dt_Customer"])

dates = [i.date() for i in data["Dt_Customer"]]

#Dates of the newest and oldest recorded customer
print("The newest customer's enrolment date in the records:", max(dates))
print("The oldest customer's enrolment date in the records:", min(dates))
```

The newest customer's enrolment date in the records: 2014-12-06  
The oldest customer's enrolment date in the records: 2012-01-08

I will create a new feature called "**Customer\_For**" that indicates the number of days a customer has been shopping in the store, relative to the most recent date in the record

```
In [6]: #Created a feature "Customer_For"
dates_max = max(dates) #taking it to be the newest customer

days = [dates_max - i for i in dates]

data["Customer_For"] = days
data["Customer_For"] = pd.to_numeric(data["Customer_For"], errors="coerce")
```

```
In [7]: print("Total categories in the feature Marital_Status:\n", data["Marital_Status"])
print("Total categories in the feature Education:\n", data["Education"].v
```

Total categories in the feature Marital\_Status:  
Married 857  
Together 573  
Single 471  
Divorced 232  
Widow 76  
Alone 3  
Absurd 2  
YOLO 2  
Name: Marital\_Status, dtype: int64

Total categories in the feature Education:  
Graduation 1116  
PhD 481  
Master 365  
2n Cycle 200  
Basic 54  
Name: Education, dtype: int64

To extract the age of a customer, I will create a new feature called "**Age**" using the "**Year\_Birth**" column. This new feature will indicate the birth year of the customer, allowing me to easily calculate their age.

I will also create a feature called "**Spent**" that indicates the total amount of money that a customer has spent in various categories over the past two years.

In addition, I will create a new feature called "Living\_With" that uses the "**Marital\_Status**" column to determine whether couples are living together or not.

Another feature I will create is "**Children**", which indicates the total number of children and teenagers in a household. This will allow me to get a better understanding of the size and composition of each household.

To further clarify the makeup of each household, I will also create a new feature called "**Family\_Size**" that indicates the total number of people living in a household.

Additionally, I will create a feature called "**Is\_Parent**" that indicates whether a customer is a parent or not. This will allow me to better understand the family dynamics of each household.

Finally, I will create three categories in the "**Education**" column by simplifying its value counts. This will make it easier to analyze and compare the education levels of different customers.

After creating these new features, I will drop any redundant columns that are no longer needed.

```
In [8]: #Feature Engineering
#Age of customer today
data["Age"] = 2021 - data["Year_Birth"]

#Total spendings on various items
data["Spent"] = data["MntWines"] + data["MntFruits"] + data["MntMeatProdu"
    "MntSweetProducts"] + data["MntGoldProds"]

#Deriving living situation by marital status"Alone"
data["Living_With"] = data["Marital_Status"].replace(
    {"Married": "Partner", "Together": "Partner", "Absurd": "Alone", "Wid
    "Divorced": "Alone", "Single": "Alone", })

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

#Feature for total members in the household
data["Family_Size"] = data["Living_With"].replace({"Alone": 1, "Partner": 2, "Widow": 2, "Divorced": 2, "Together": 2, "Absurd": 2})

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

#Segmenting education levels in three groups
data["Education"] = data["Education"].replace(
    {"Basic": "Undergraduate", "2n Cycle": "Undergraduate", "Graduation": "Postsecondary School", "PhD": "Postgraduate"})

#For clarity
data = data.rename(
    columns={"MntWines": "Wines", "MntFruits": "Fruits", "MntMeatProducts": "MeatProducts", "MntSweetProducts": "Sweets", "MntGoldProds": "Gold"})

#Dropping some of the redundant features
to_drop = ["Marital_Status", "Dt_Customer", "Z_CostContact", "Z_Revenue", "MntProdArt", "MntProdCon", "MntProdElect", "MntProdHome", "MntProdLeisure", "MntProdOther"]
data = data.drop(to_drop, axis=1)
```

```
In [9]: data.describe()
```

	Income	Kidhome	Teenhome	Recency	Wines	Fruit
<b>count</b>	2216.000000	2216.000000	2216.000000	2216.000000	2216.000000	2216.000000
<b>mean</b>	52247.251354	0.441787	0.505415	49.012635	305.091606	26.35604
<b>std</b>	25173.076661	0.536896	0.544181	28.948352	337.327920	39.79391
<b>min</b>	1730.000000	0.000000	0.000000	0.000000	0.000000	0.00000
<b>25%</b>	35303.000000	0.000000	0.000000	24.000000	24.000000	2.00000
<b>50%</b>	51381.500000	0.000000	0.000000	49.000000	174.500000	8.00000
<b>75%</b>	68522.000000	1.000000	1.000000	74.000000	505.000000	33.00000
<b>max</b>	666666.000000	2.000000	2.000000	99.000000	1493.000000	199.00000

8 rows × 28 columns

**Observations:** The statistics above show some inconsistencies in the mean and maximum values for income and age. It is worth noting that the maximum age is 128 years, but this value is likely not accurate because it is based on the current year (2021) and the data is from an older time period. This means that the age values may not be accurate and should be interpreted with caution.

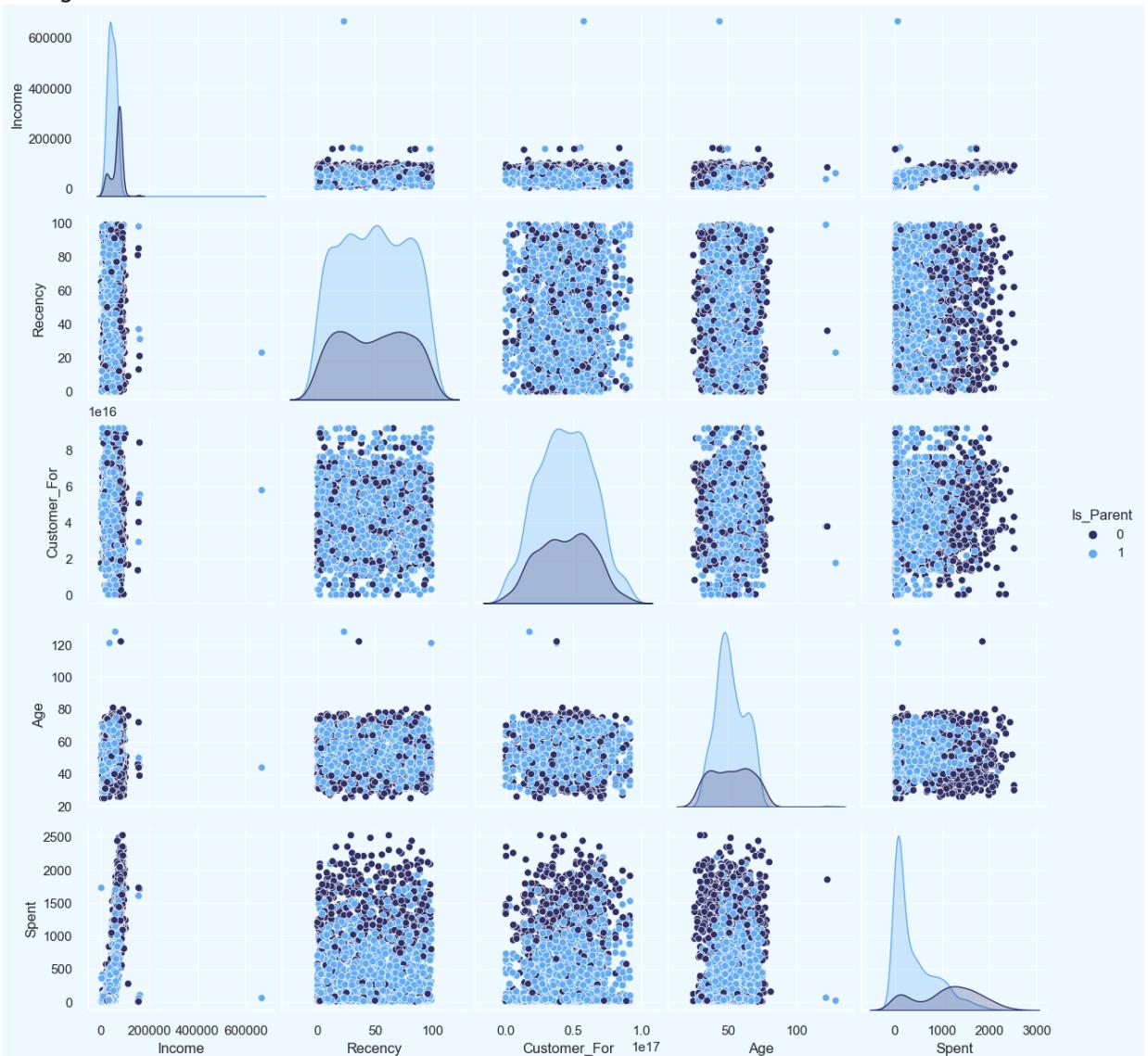
```
In [10]: # Initializing Color pallets
```

```
sns.set(rc={"axes.facecolor": "#edf9ff", "figure.facecolor": "#edf9ff"})
pallet = ["#2f2f68", "#6f729e", "#b1b2d6", "#c9c0b9", "#788a9f", "#60abf3"]
cmap = colors.ListedColormap([ "#2f2f68", "#6f729e", "#D6B2B1", "#b1b2d6",
pal = ["#2f2f68", "#c9c0b9", "#788a9f", "#60acf3"]]
```

```
In [11]: # Plotting following features
```

```
To_Plot = ["Income", "Recency", "Customer_For", "Age", "Spent", "Is_Paren
print("Relative Plot Of Some Selected Features: A Data Subset")
plt.figure()
sns.pairplot(data[To_Plot], hue="Is_Parent", palette=[ "#2f2f68", "#60abf3"])
plt.show()
```

Relative Plot Of Some Selected Features: A Data Subset  
<Figure size 800x550 with 0 Axes>



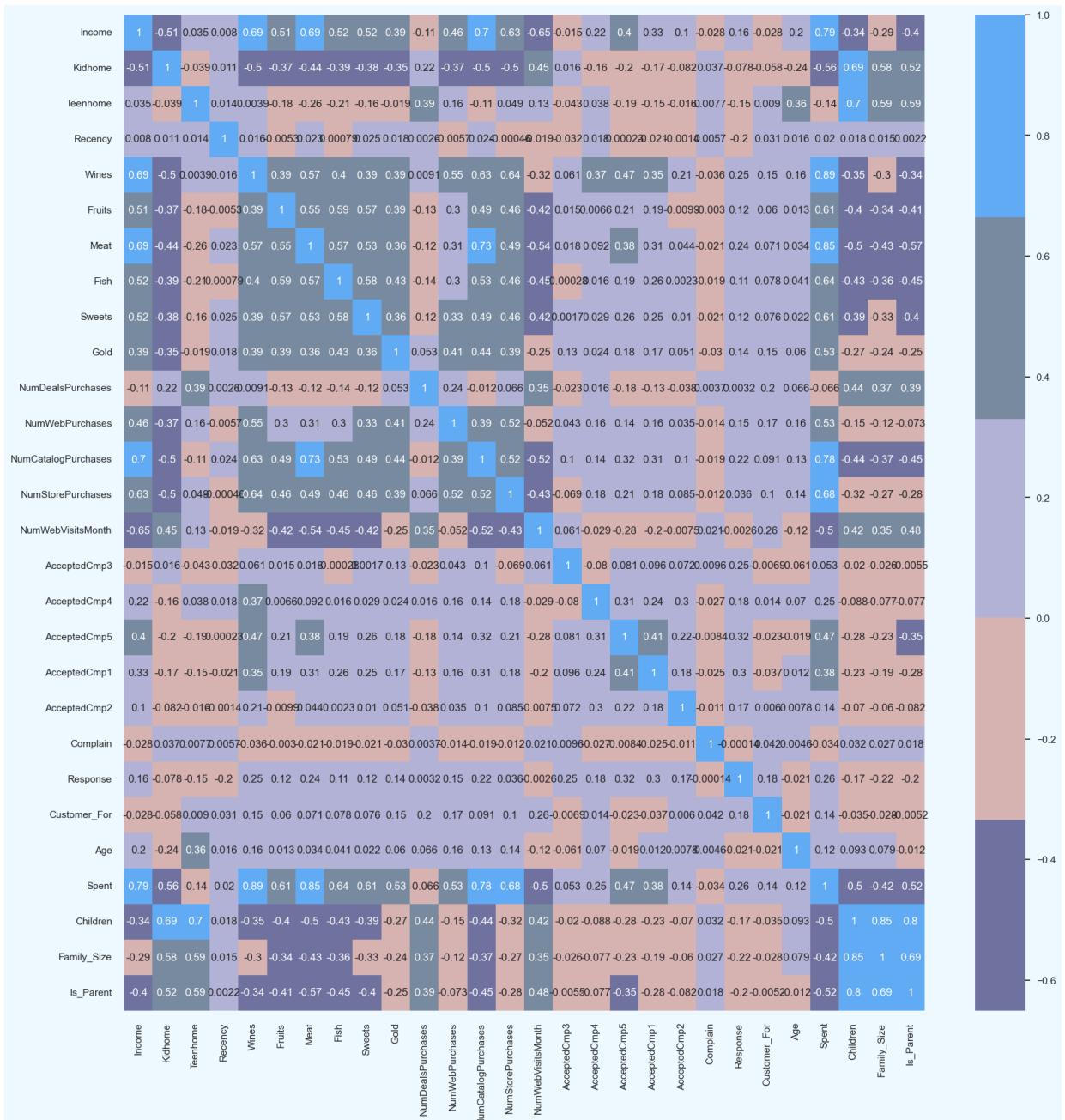
**Observations:** From the plots above, it is clear that there are some outliers in the income and age features. These outliers may have a negative impact on the accuracy of our analysis, so I will remove them from the data. This will help to ensure that our results are more accurate and reliable.

```
In [12]: #Dropping the outliers by setting a cap on Age and income.
data = data[(data["Age"] < 90)]
data = data[(data["Income"] < 600000)]
print(f"The total data-points after removing the outliers are: {len(data)}
```

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

```
In [13]: #correlation matrix
cor_mat = data.corr()
plt.figure(figsize=(20, 20))
sns.heatmap(cor_mat, annot=True, cmap=cmap, center=0)
```

Out [13]: <AxesSubplot: >



### Next steps:

1. Use label encoding to convert the categorical features into numeric form.
2. Scale the features using the standard scaler to ensure that they are on the same scale.
3. Create a subset dataframe for dimensionality reduction to reduce the number of features and improve the efficiency of our analysis.

```
In [14]: # 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']

```
In [15]: # Using Label Encoding to convert categorical data to numerical.
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 [16]: # Creating a copy of data
data_copy = data.copy()

# Creating a subset of dataframe by dropping the features on deals accepted
cols_del = ['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1']
data_copy = data_copy.drop(cols_del, axis=1)

# Scaling
scaler = StandardScaler()
scaler.fit(data_copy)
scaled_ds = pd.DataFrame(scaler.transform(data_copy), columns=data_copy.columns)
print("All features are now scaled using standard scaler")
```

All features are now scaled using standard scaler

## Dimensionality Reduction

In this problem, there are many factors or attributes that will be used to make the final classification. The more features there are, the harder it is to work with the data. Many of these features are correlated and redundant, so it is important to reduce the number of features in the data.

To accomplish this, I will perform dimensionality reduction on the selected features. Dimensionality reduction is the process of reducing the number of random variables by obtaining a set of principal variables. This will help to simplify the data and improve the efficiency of our analysis.

After reducing the dimensionality of the data, I will put the features through a classifier to make the final classification. This will help to identify any patterns or trends in the data and make more accurate predictions.

For this project, I will use principal component analysis (PCA) to perform dimensionality reduction on the selected features. PCA is a statistical technique that is used to reduce the number of dimensions in a dataset while preserving as much information as possible.

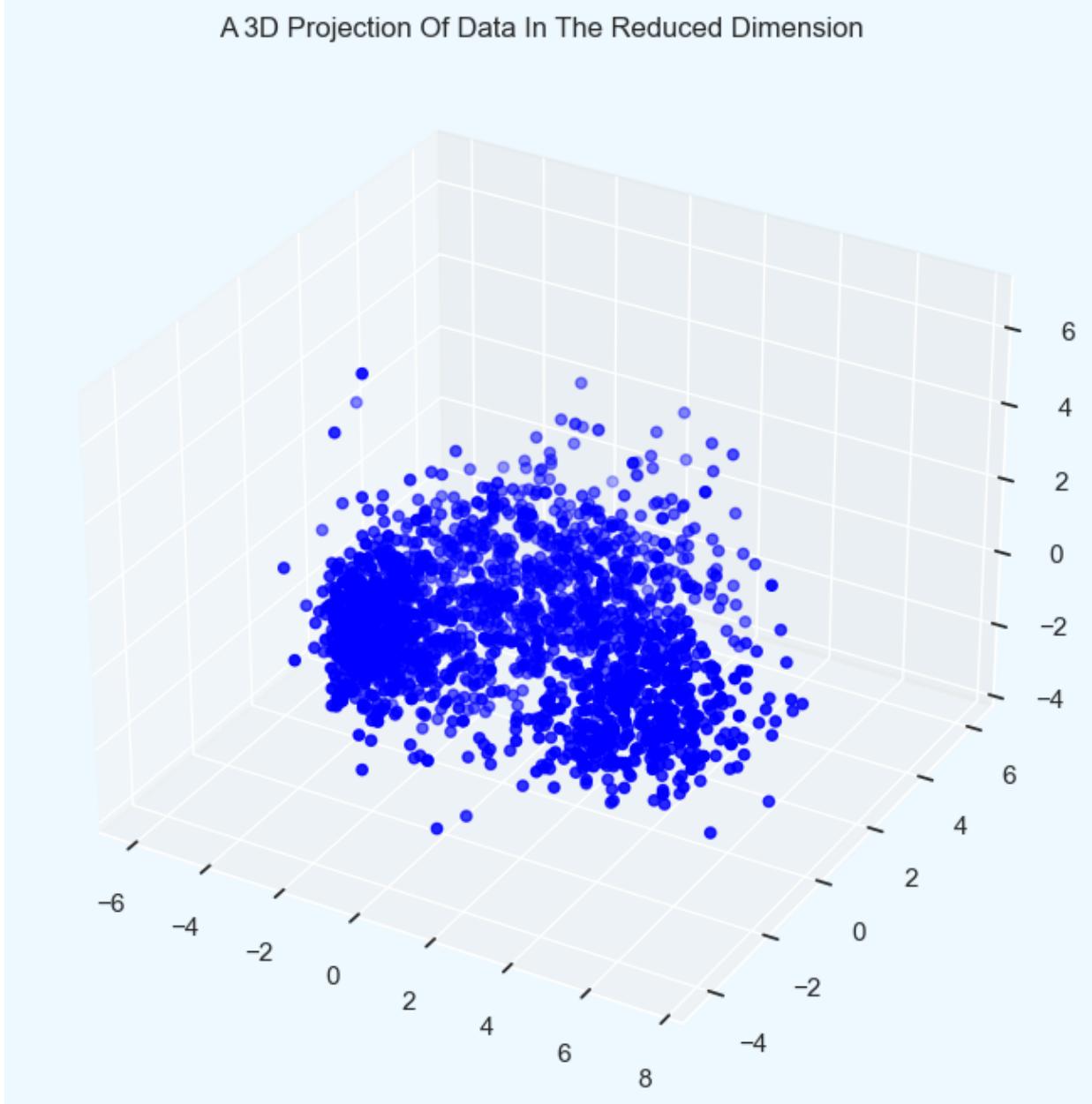
After applying PCA, I will reduce the dimensions to 3, which will make it easier to visualize the results and see how the different features are distributed. I will then plot the reduced dataframe to see the results of the dimensionality reduction and identify any patterns or trends in the data.

```
In [17]: #Initiating PCA to reduce dimentions aka features to 3
pca = PCA(n_components=3)
pca.fit(scaled_ds)
pca_ds = pd.DataFrame(pca.transform(scaled_ds), columns=["col1", "col2",
pca_ds.describe().T
```

	count	mean	std	min	25%	50%	75%	ma
col1	2212.0	-3.854662e-17	2.878377	-5.969394	-2.538494	-0.780421	2.383290	7.44430
col2	2212.0	-2.569775e-17	1.706839	-4.312196	-1.328316	-0.158123	1.242289	6.14272
col3	2212.0	5.701688e-17	1.221956	-3.530416	-0.829067	-0.022692	0.799895	6.61122

```
In [18]: # A 3D Projection Of Data In The Reduced Dimension
x = pca_ds["col1"]
y = pca_ds["col2"]
z = pca_ds["col3"]

# Plotting
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x, y, z, c="blue", marker="o")
ax.set_title("A 3D Projection Of Data In The Reduced Dimension")
plt.show()
```



## Clustering

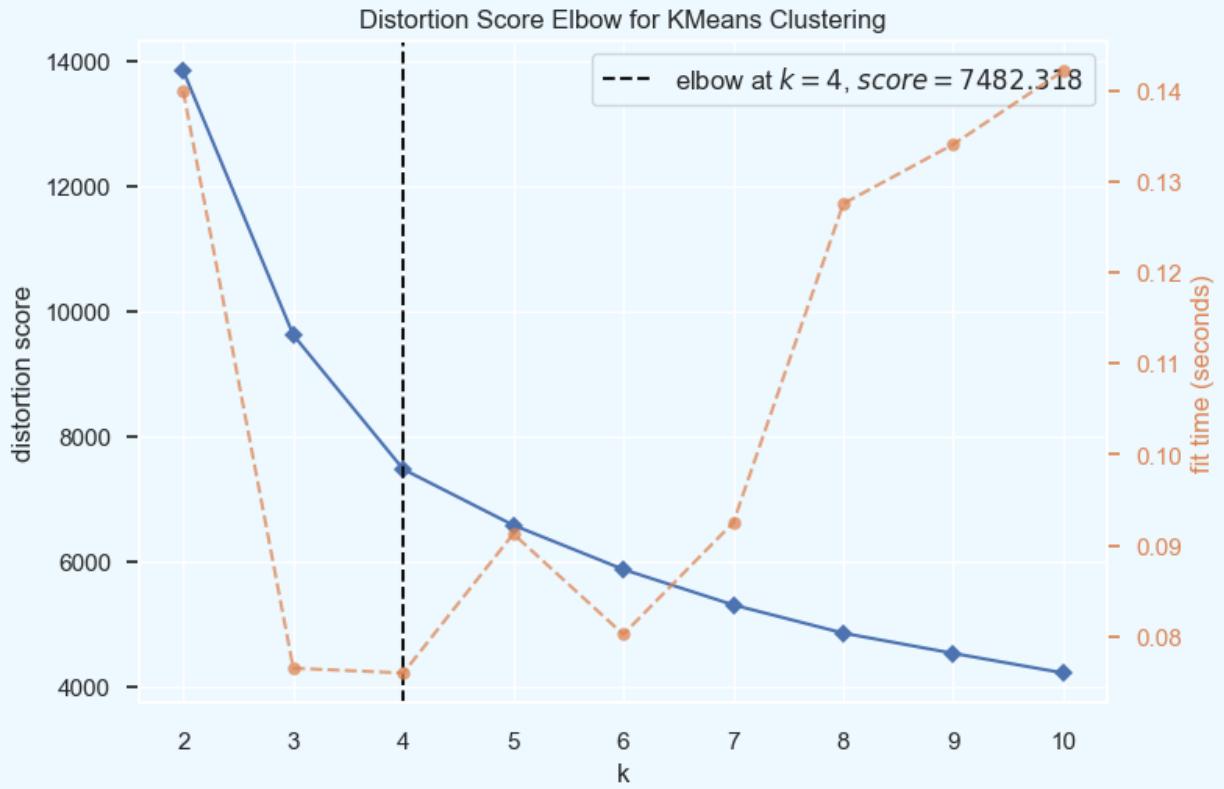
In order to perform clustering, I will use Agglomerative clustering on the reduced dataset, which has three dimensions. Agglomerative clustering is a hierarchical clustering method that involves merging individual examples into clusters until the desired number of clusters is reached.

The steps involved in the clustering process are:

1. Using the Elbow Method to determine the optimal number of clusters to form.
2. Performing clustering using Agglomerative Clustering.
3. Examining the resulting clusters using a scatter plot.

```
In [19]: # Quick examination of elbow method to find numbers of clusters to make.  
print('Elbow Method to determine the number of clusters to be formed: ')  
Elbow_M = KElbowVisualizer(KMeans(), k=10)  
Elbow_M.fit(pca_ds)  
Elbow_M.show()
```

Elbow Method to determine the number of clusters to be formed:

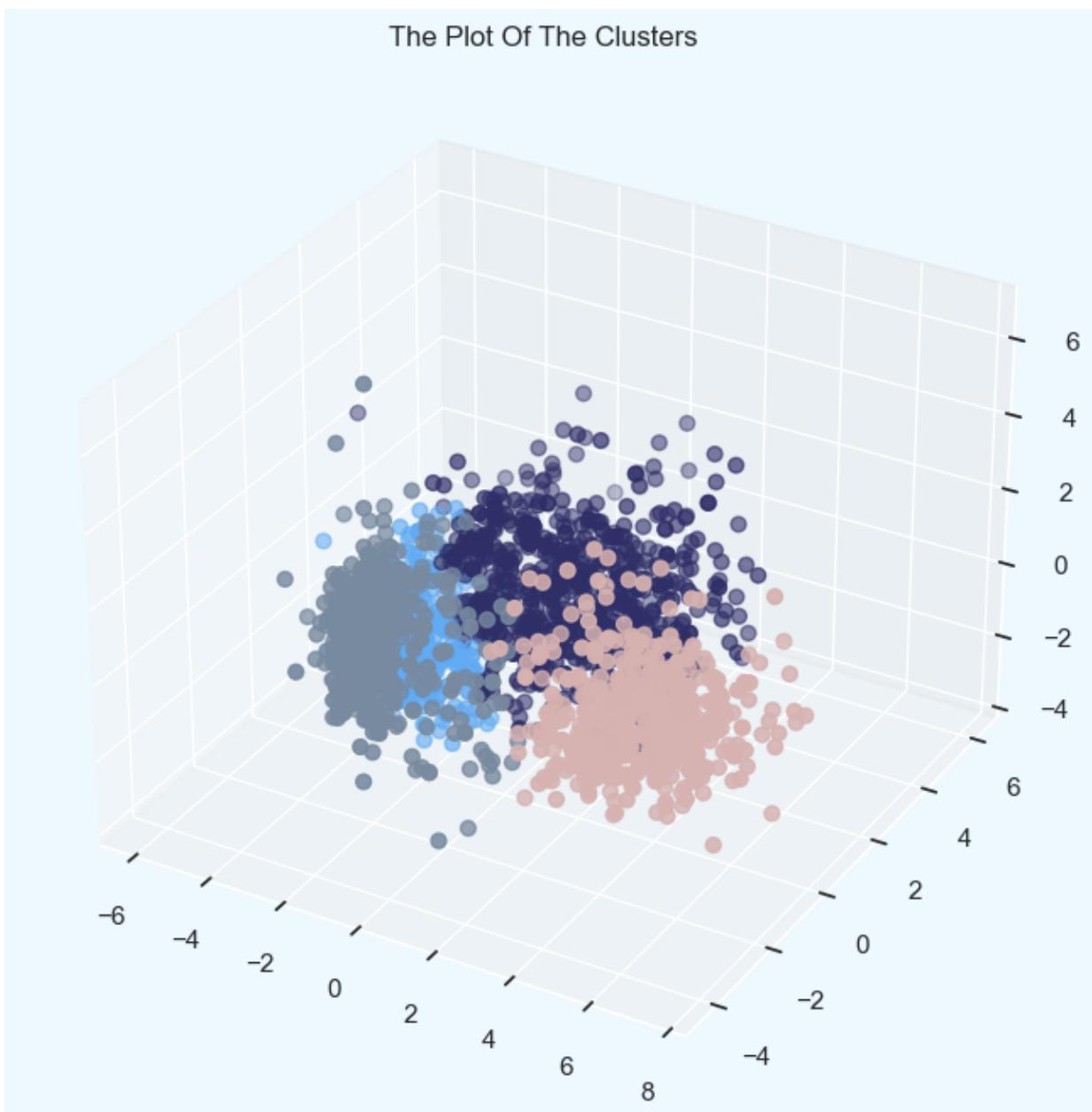


```
Out[19]: <AxesSubplot: title={'center': 'Distortion Score Elbow for KMeans Clustering'}, xlabel='k', ylabel='distortion score'>
```

**Observations:** The previous cell suggests that four clusters will be optimal for this dataset. Next, we will fit the Agglomerative Clustering Model to the data to obtain the final clusters.

```
In [20]: #Initiating the Agglomerative Clustering model  
AC = AgglomerativeClustering(n_clusters=4)  
# fit model and predict clusters  
yhat_AC = AC.fit_predict(pca_ds)  
pca_ds["Clusters"] = yhat_AC  
#Adding the Clusters feature to the original dataframe.  
data["Clusters"] = yhat_AC
```

```
In [21]: #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()
```

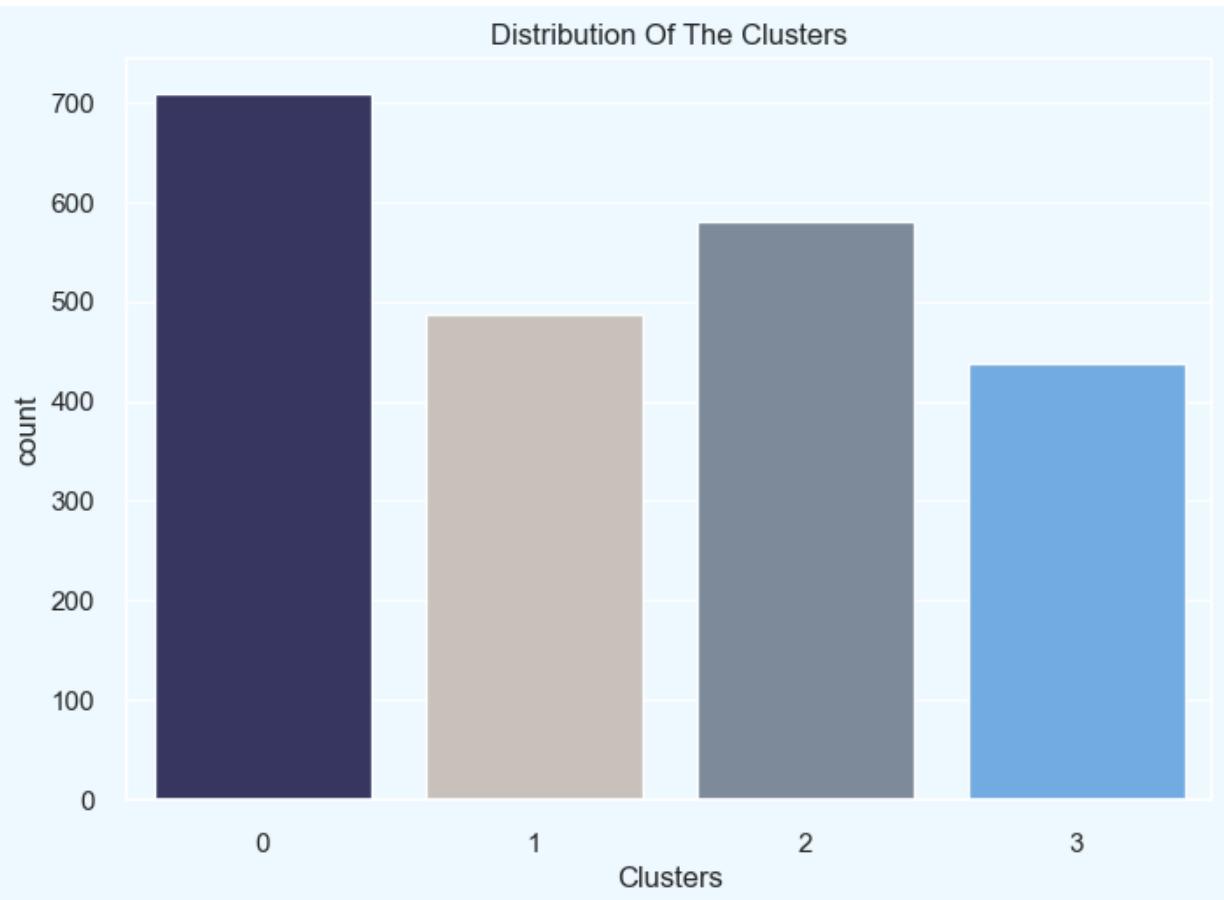


## Evaluating Model

Since this is an unsupervised clustering task, there is no labeled feature to use for evaluating or scoring the model. The purpose of this section is to study the patterns within the clusters that were formed and determine the nature of these patterns. To do this, we will use exploratory data analysis to examine the data in relation to the clusters and draw conclusions based on the resulting patterns.

Firstly, let us have a look at the group distribution of clustering

```
In [22]: fig = sns.countplot(x=data["Clusters"], palette=pal)
fig.set_title("Distribution Of The Clusters")
plt.show()
```



**Observations:** The clusters appear to be uniformly distributed.

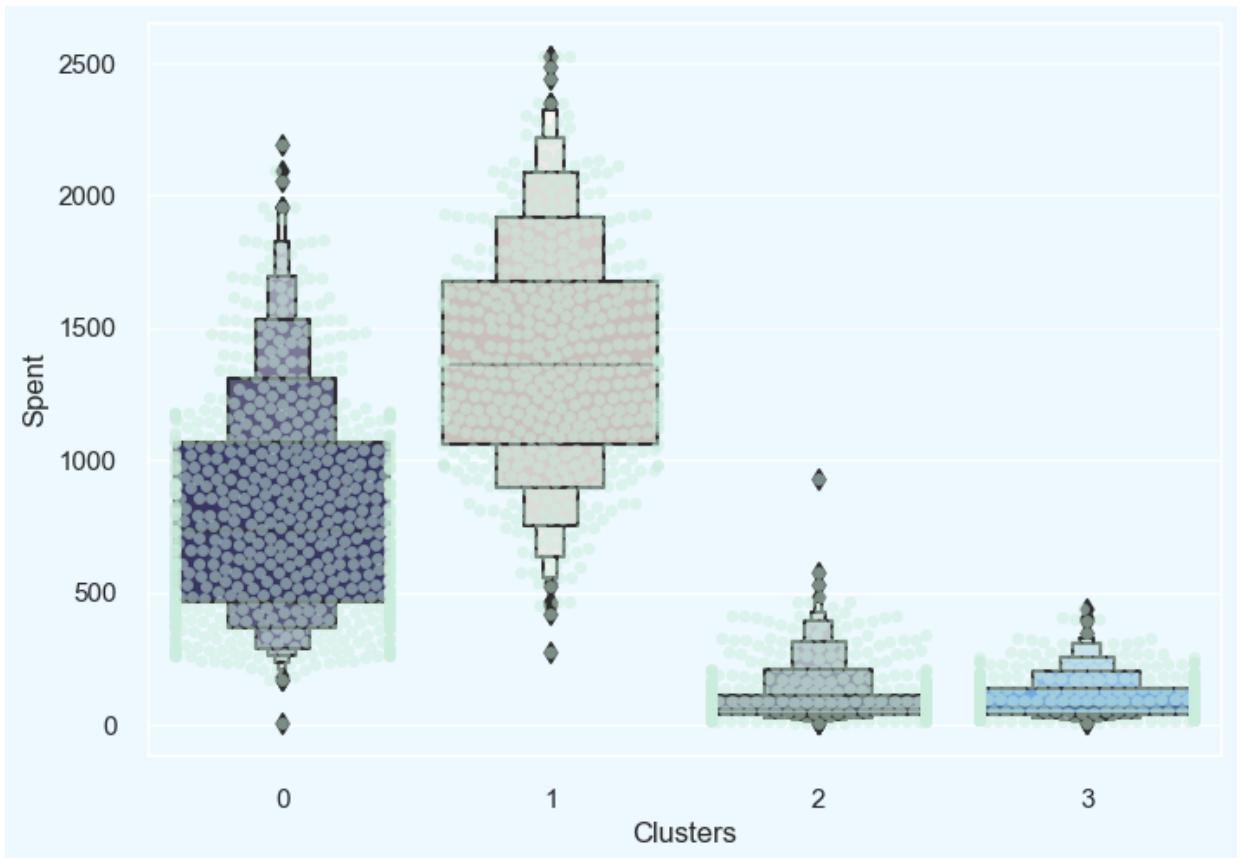
```
In [23]: fig = sns.scatterplot(data=data, x=data["Spent"], y=data["Income"], hue=d)
fig.set_title("Profile of a group of individuals based on their income an
plt.legend()
plt.show()
```



Income vs spending plot shows the clusters pattern:

- Group 0: those with average income and high spending
- Group 1: those with high income and high spending
- Group 2: those with low income and low spending
- Group 3: those with low income and high spending

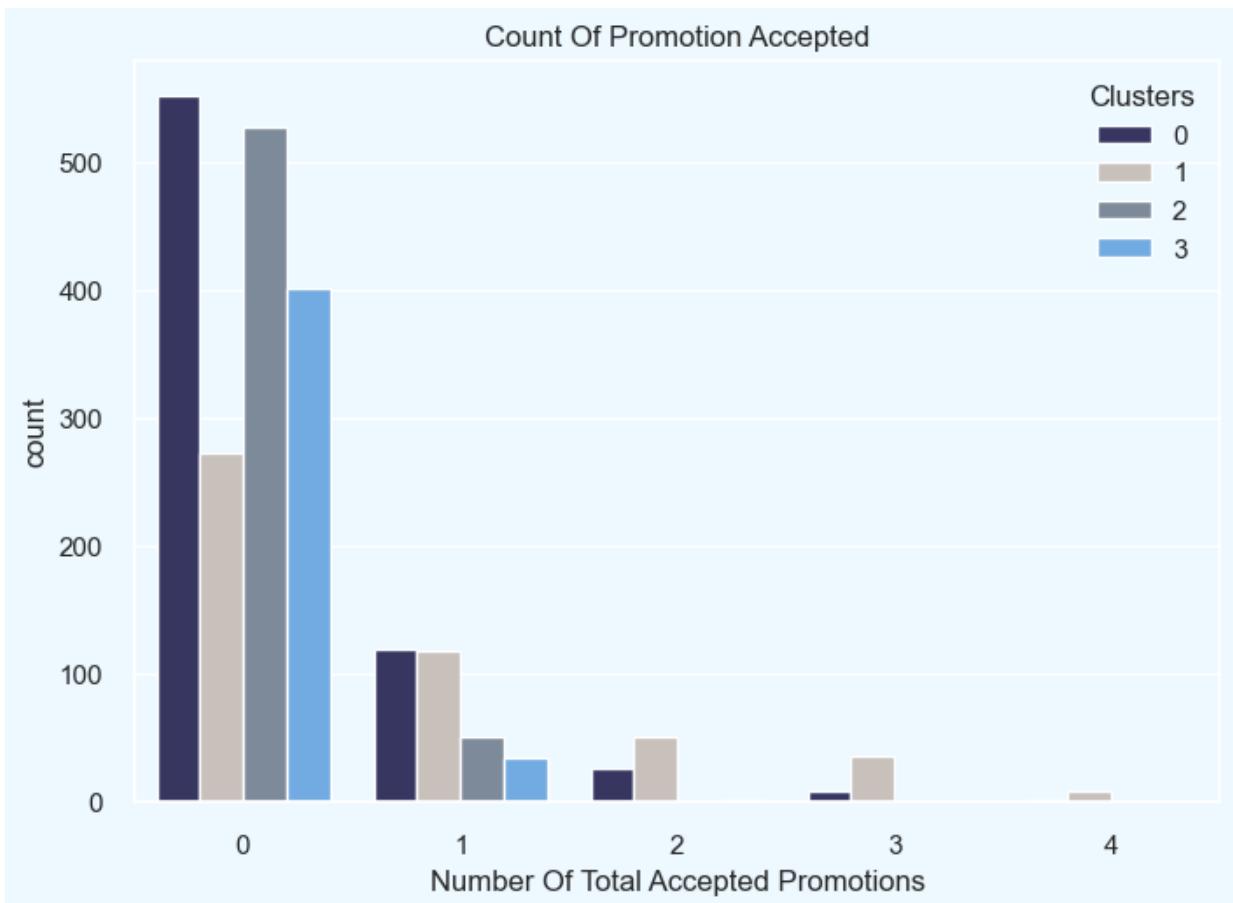
```
In [24]: plt.figure()
fig = sns.swarmplot(x=data["Clusters"], y=data["Spent"], color="#CBEDDD",
fig = sns.boxenplot(x=data["Clusters"], y=data["Spent"], palette=pal)
plt.show()
```



**Observations:** Based on the plot above, it appears that cluster 1 is the largest group of customers, followed closely by cluster 0. We can explore the spending habits of each cluster to inform targeted marketing strategies.

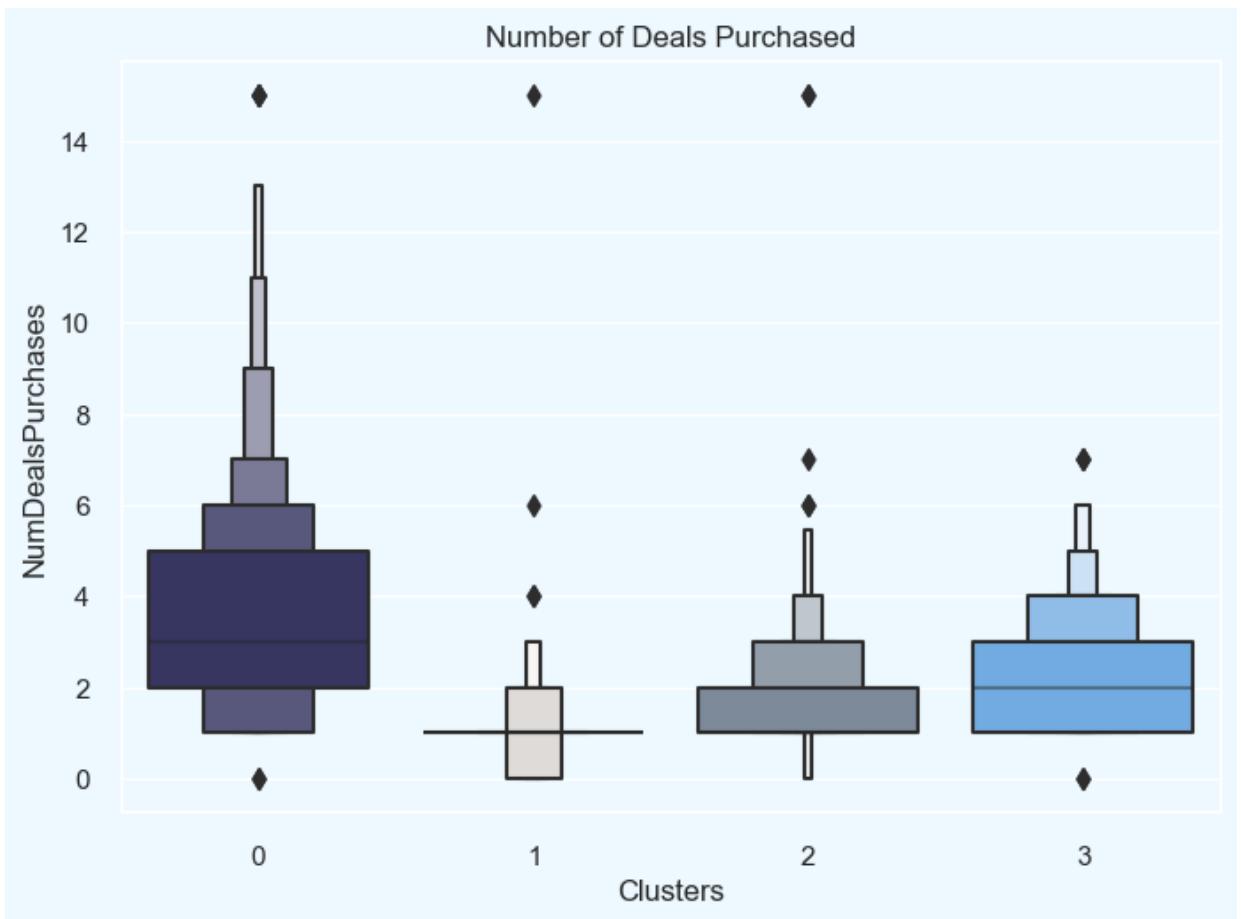
### Exploring past campaigns

```
In [25]: #Creating a feature to get a sum of accepted promotions
data["Total_Promos"] = data["AcceptedCmp1"] + data["AcceptedCmp2"] + data["AcceptedCmp5"]
#Plotting count of total campaign accepted.
plt.figure()
fig = sns.countplot(x=data["Total_Promos"], hue=data["Clusters"], palette="Set1")
fig.set_title("Count Of Promotion Accepted")
fig.set_xlabel("Number Of Total Accepted Promotions")
plt.show()
```



**Observations:** The response to the marketing campaigns has been relatively weak, with only a small number of participants overall. Furthermore, no one has participated in all five of the campaigns. This suggests that more targeted and well-planned campaigns may be necessary in order to boost sales.

```
In [26]: #Plotting the number of deals purchased
plt.figure()
pl = sns.boxenplot(y=data["NumDealsPurchases"], x=data["Clusters"], palette="Set1")
pl.set_title("Number of Deals Purchased")
plt.show()
```

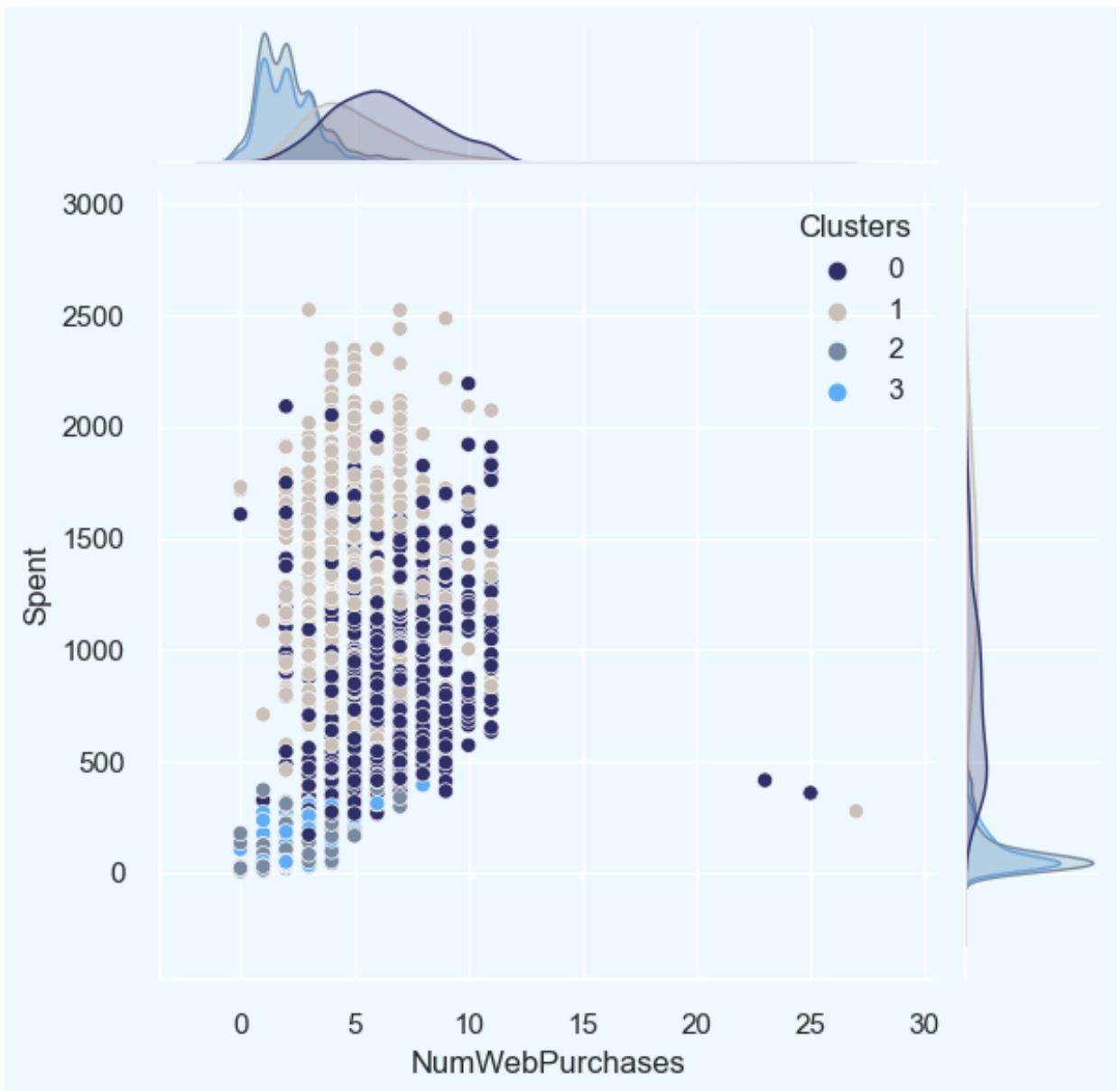


**Observations:** In contrast to the campaigns, the deals offered were more successful. They had the best results with cluster 0 and cluster 3. However, our most valuable customers in cluster 1 did not seem to be very interested in the deals. There does not appear to be anything that particularly attracts customers in cluster 2.

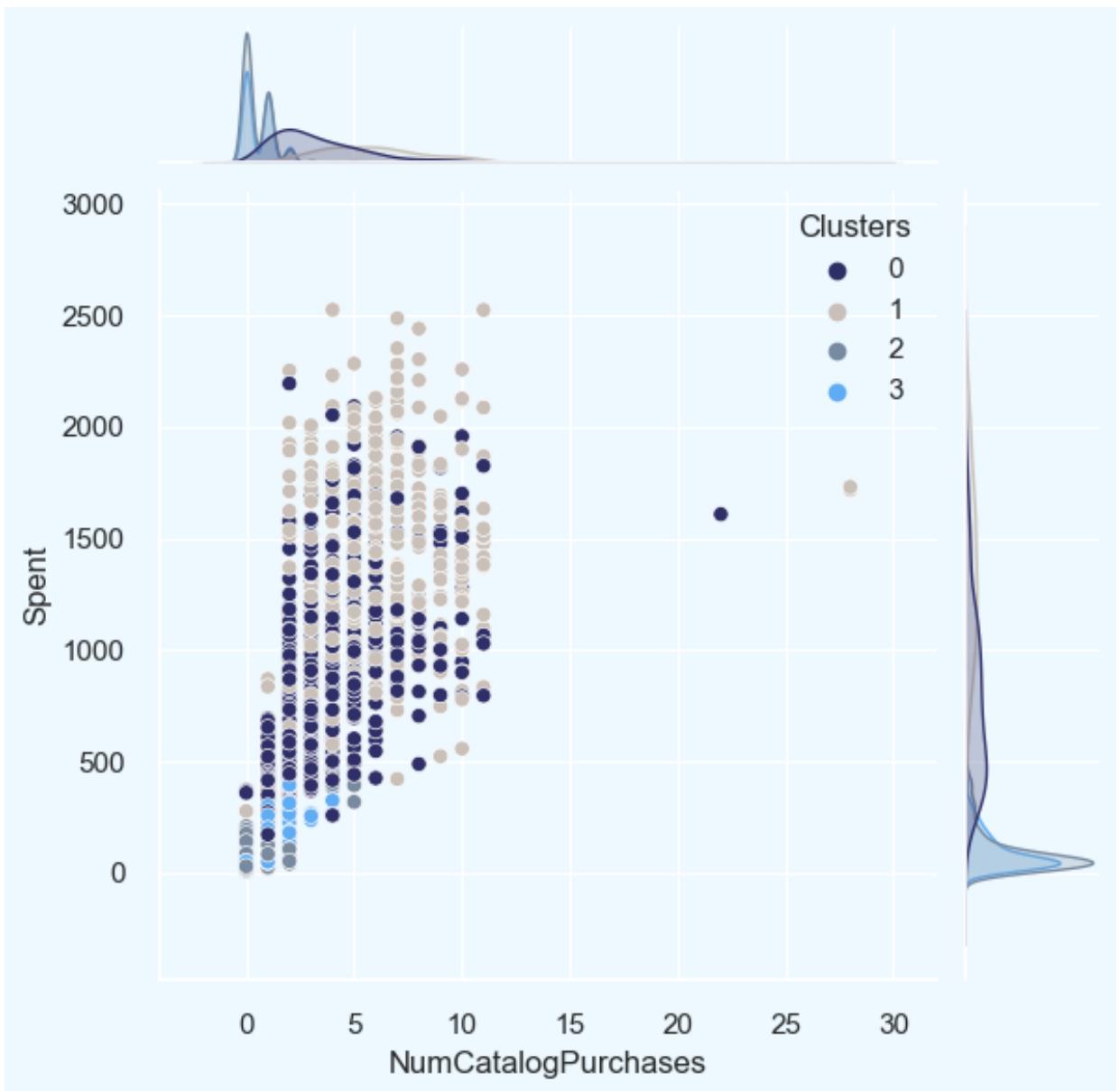
```
In [27]: # for more details on the purchasing style
purchases = ["NumWebPurchases", "NumCatalogPurchases", "NumStorePurchases"]

for i in purchases:
    plt.figure()
    sns.jointplot(x=data[i], y=data["Spent"], hue=data["Clusters"], palette=palettes[i])
    plt.show()
```

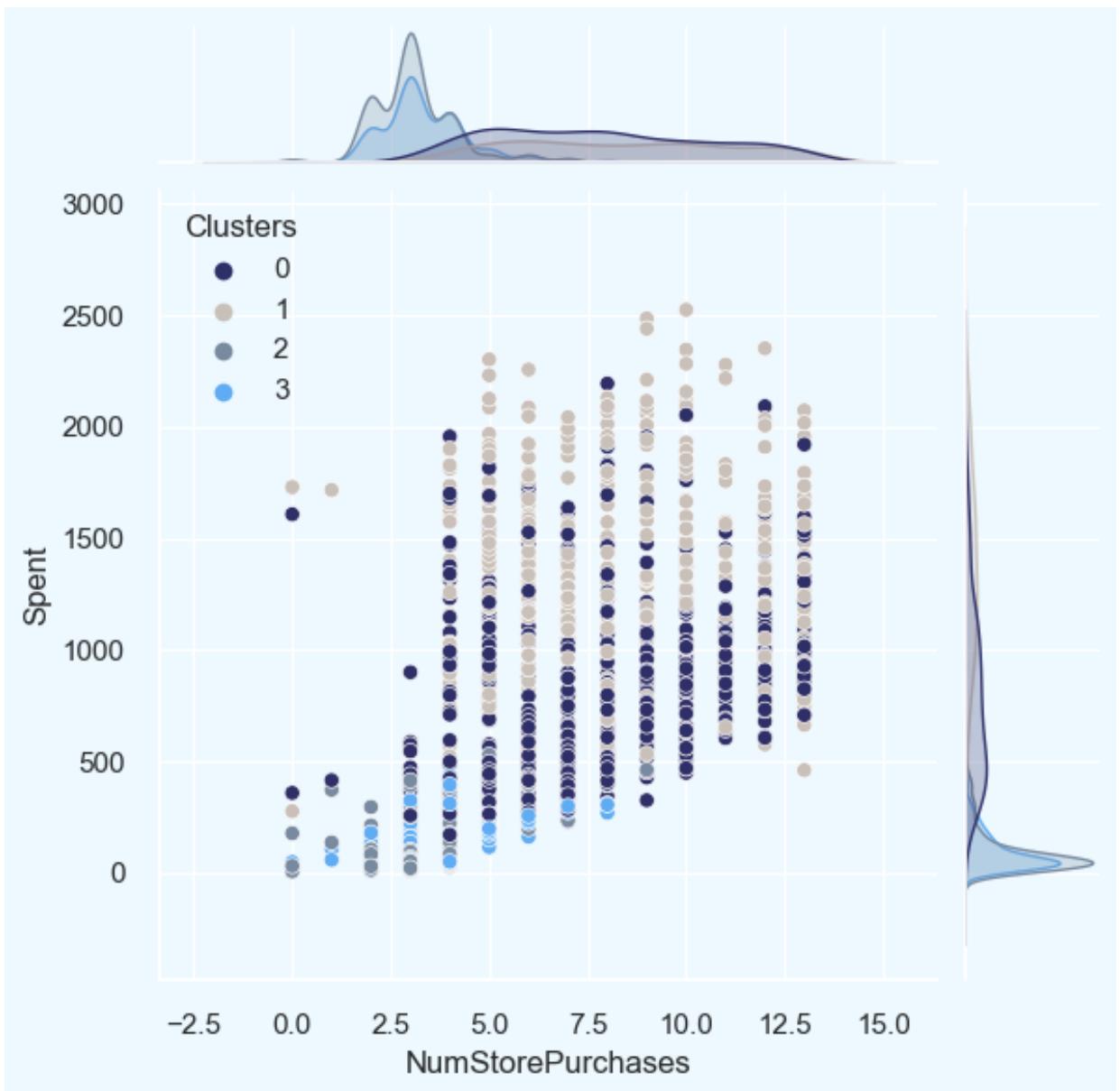
<Figure size 800x550 with 0 Axes>



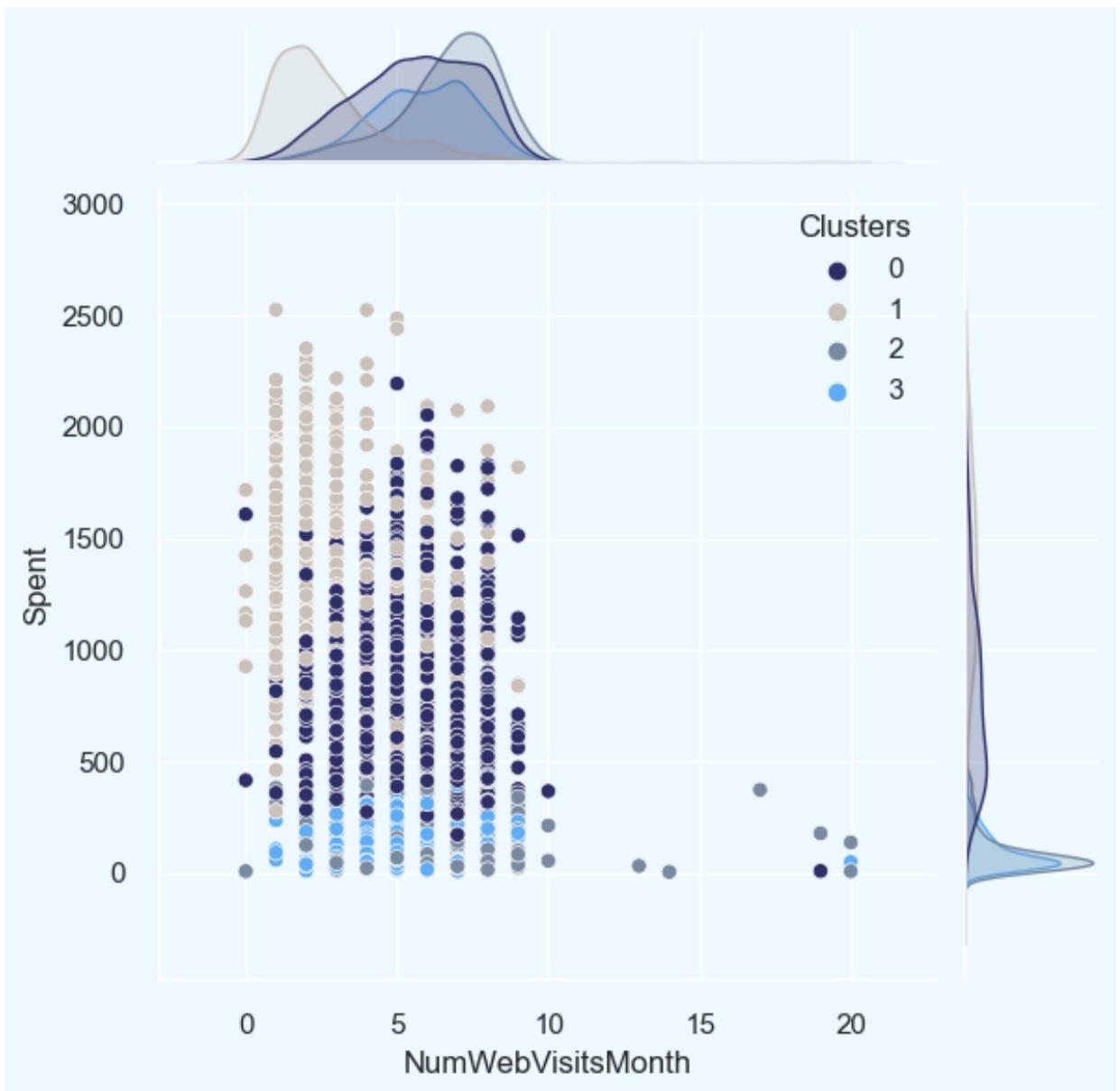
<Figure size 800x550 with 0 Axes>



<Figure size 800x550 with 0 Axes>



<Figure size 800x550 with 0 Axes>



## Profiling

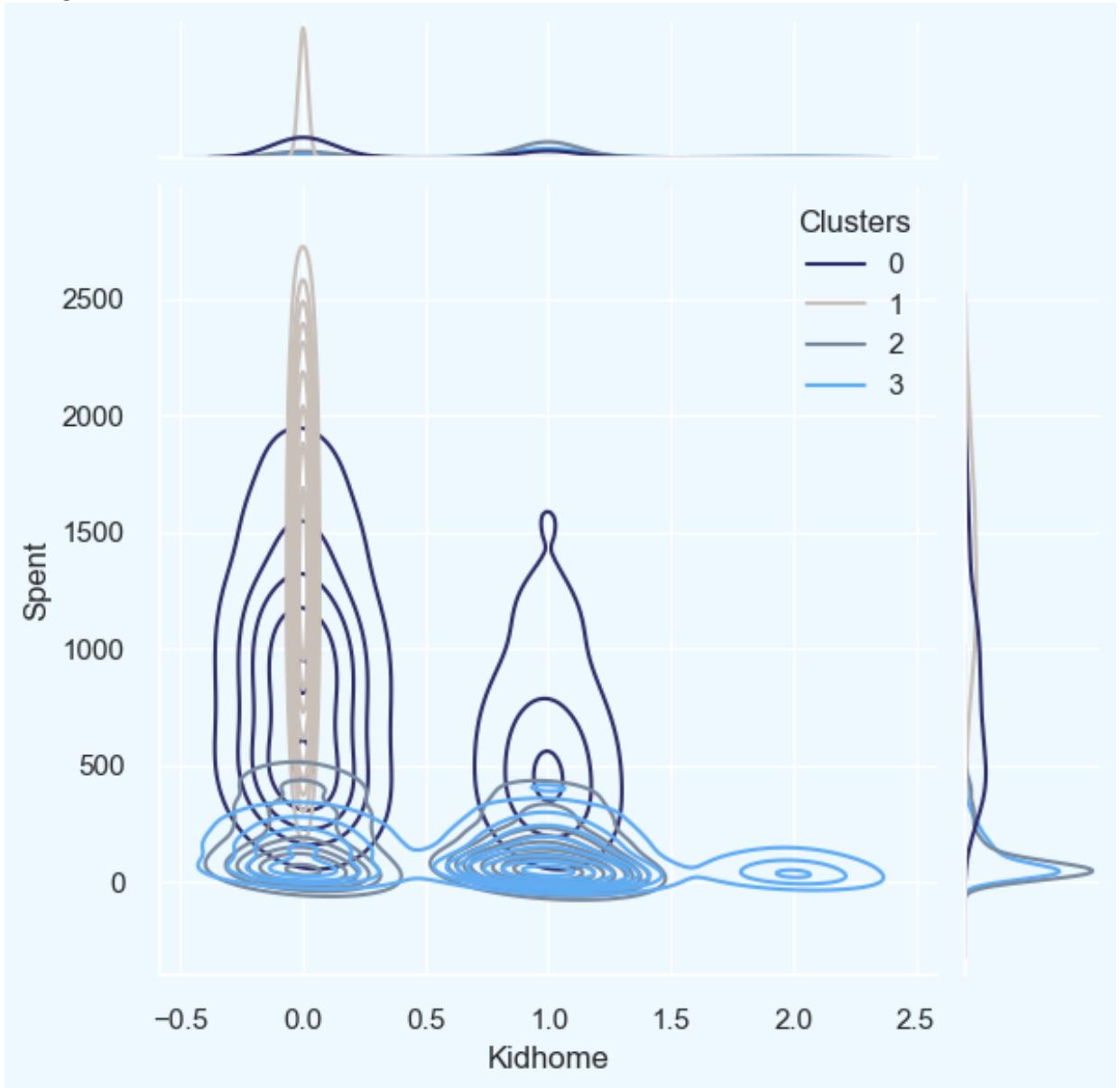
Now that we have formed the clusters and examined their purchasing habits, let us take a closer look at the individuals in each cluster. To do this, we will create profiles of the clusters and use this information to determine which customers are our most valuable and which ones require more attention from the retail store's marketing team.

To make this determination, I will plot some of the features that are indicative of a customer's personal traits in relation to the cluster they are in. Based on the results of this analysis, I will draw conclusions about the characteristics of each cluster.

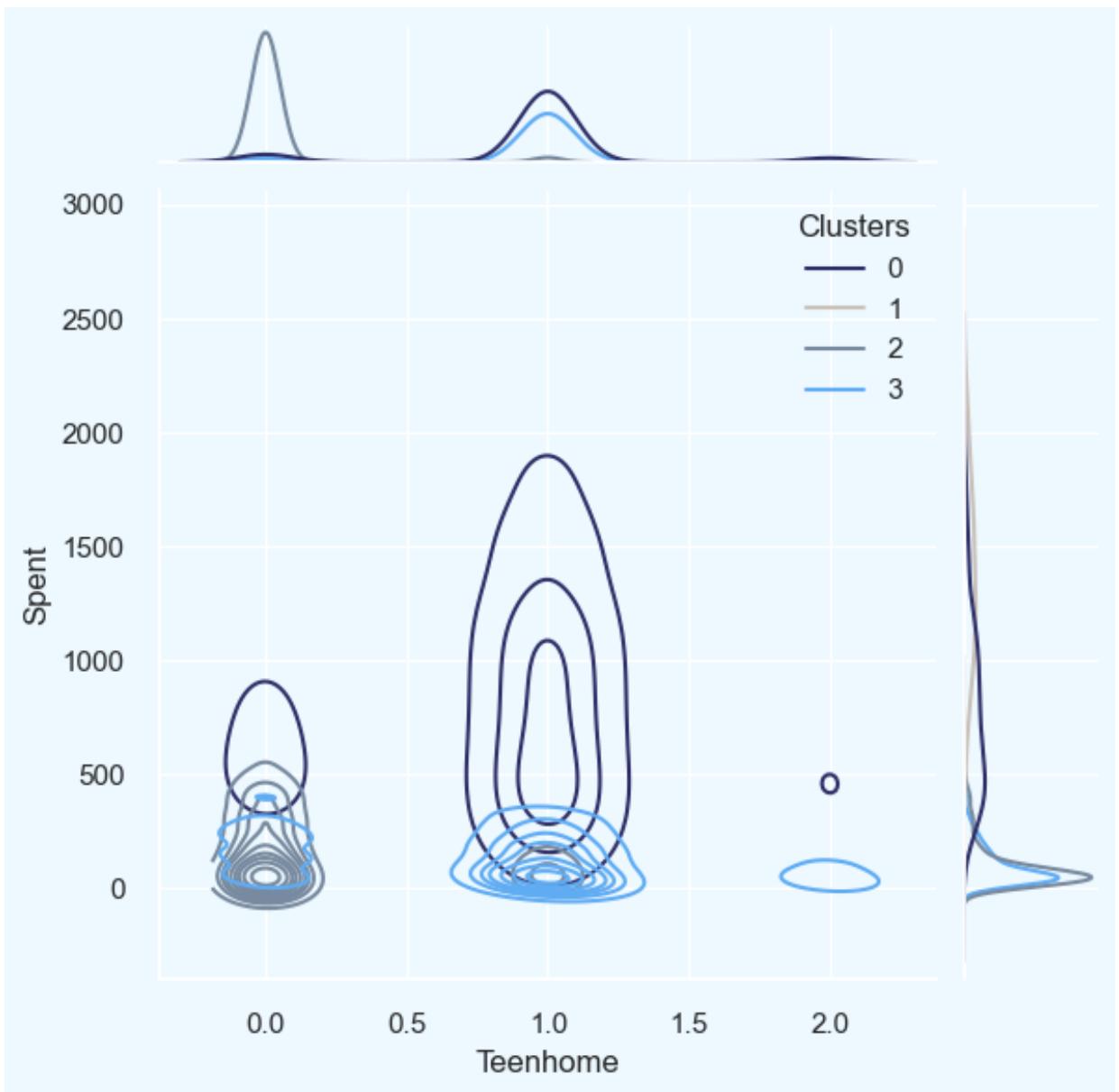
```
In [28]: personal_info = ["Kidhome", "Teenhome", "Customer_For", "Age", "Children"
                     "Living_With"]

for i in personal_info:
    plt.figure()
    sns.jointplot(x=data[i], y=data["Spent"], hue=data["Clusters"], kind=
    plt.show()
```

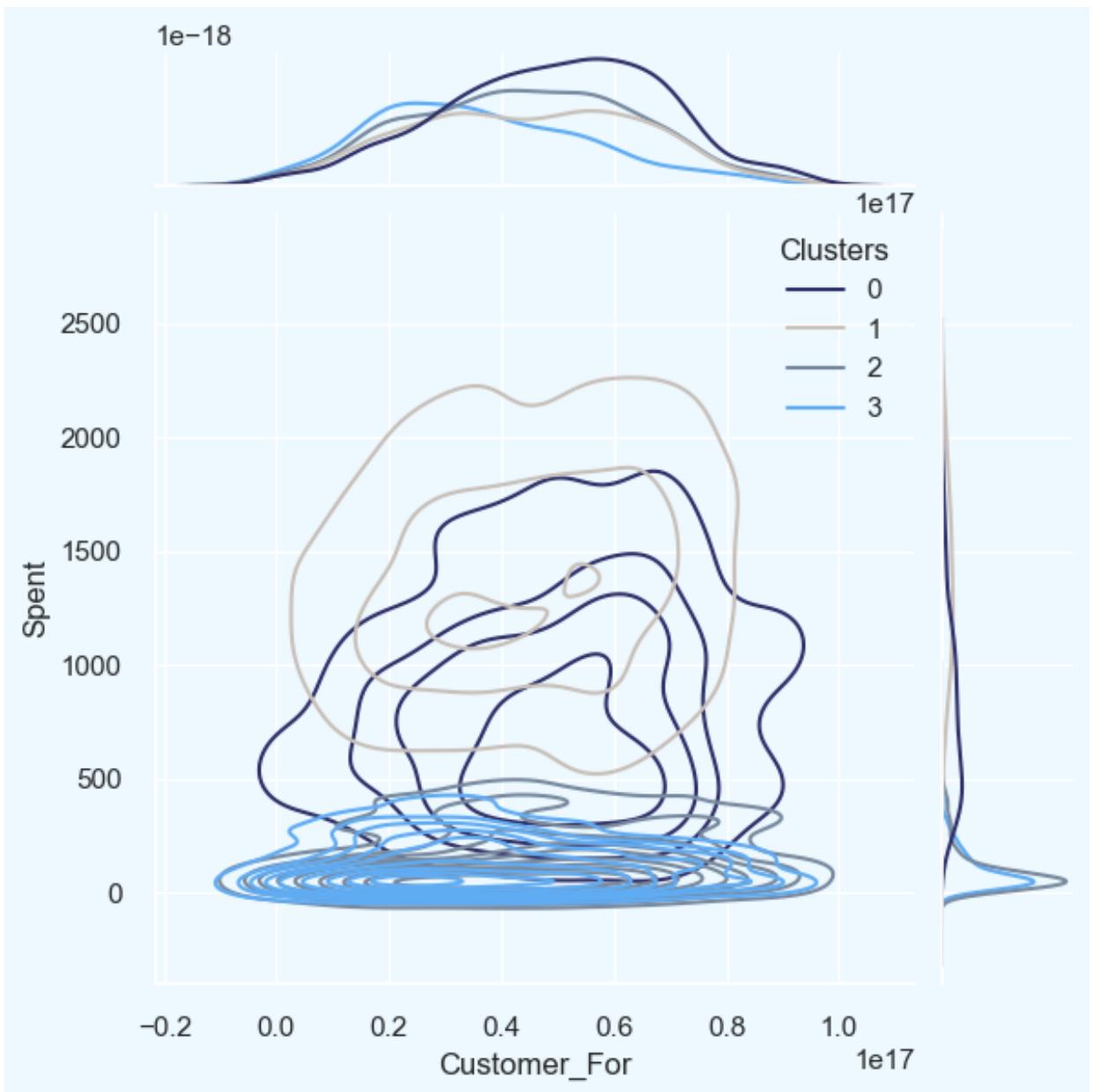
<Figure size 800x550 with 0 Axes>



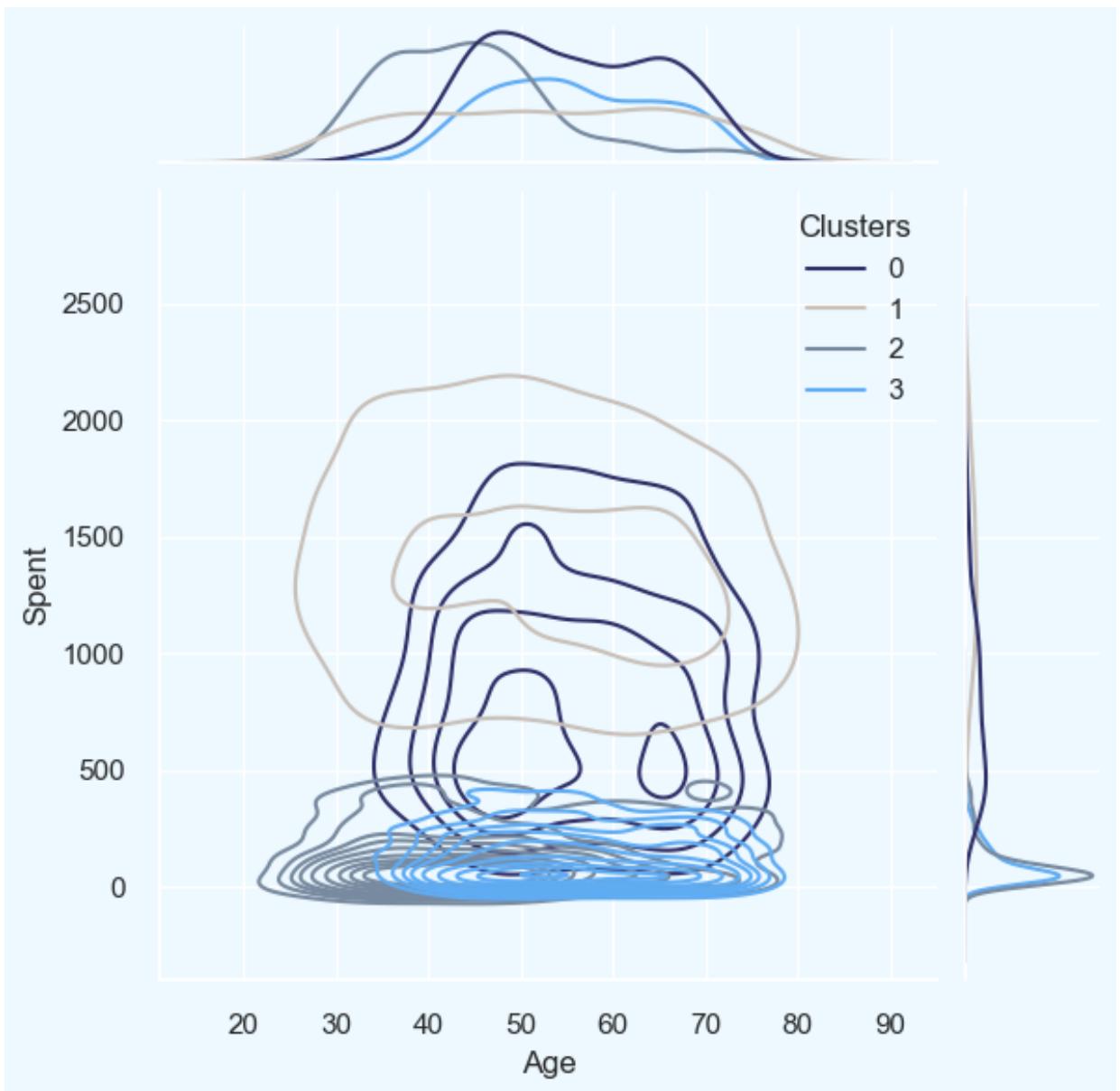
<Figure size 800x550 with 0 Axes>



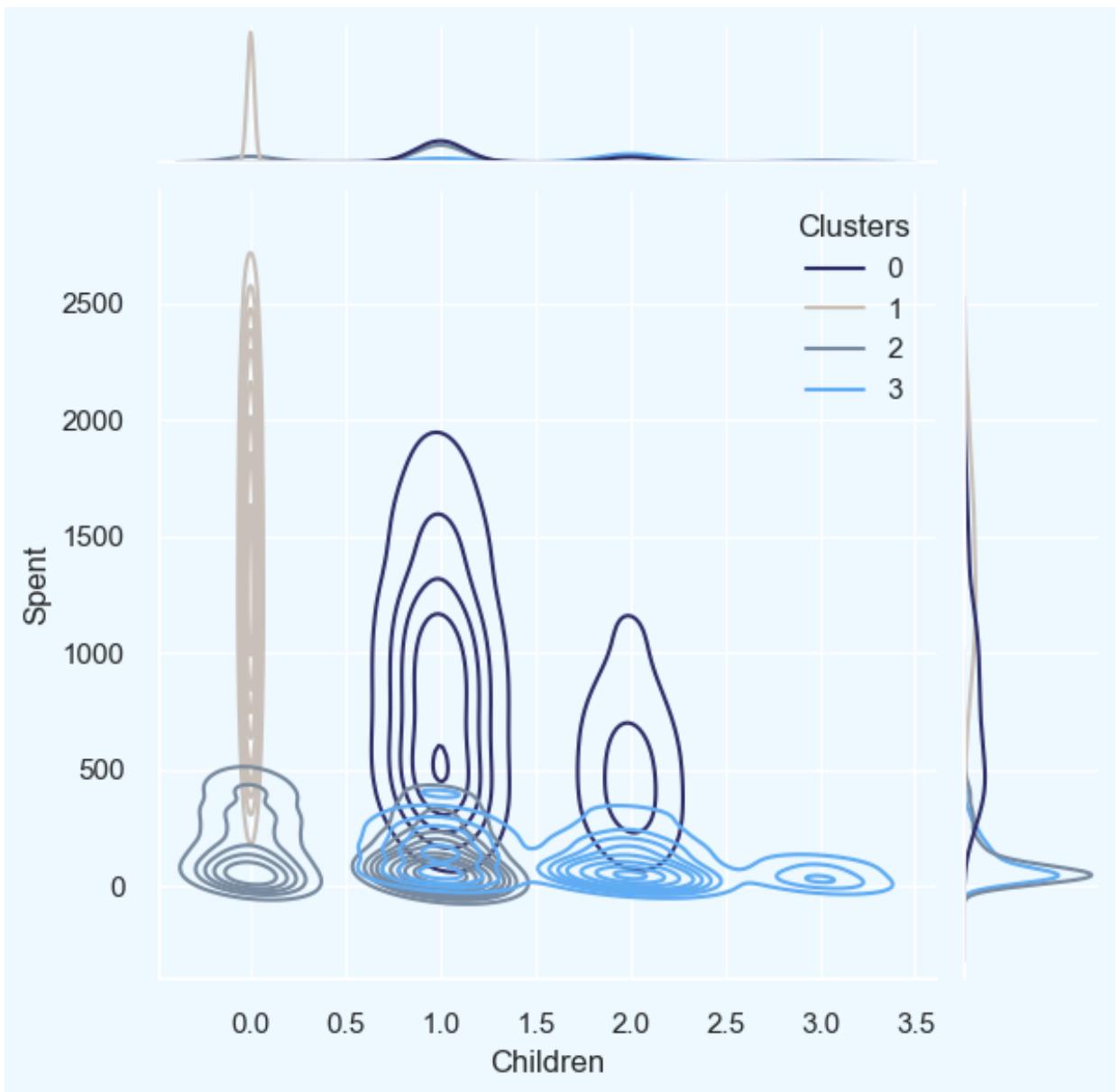
<Figure size 800x550 with 0 Axes>



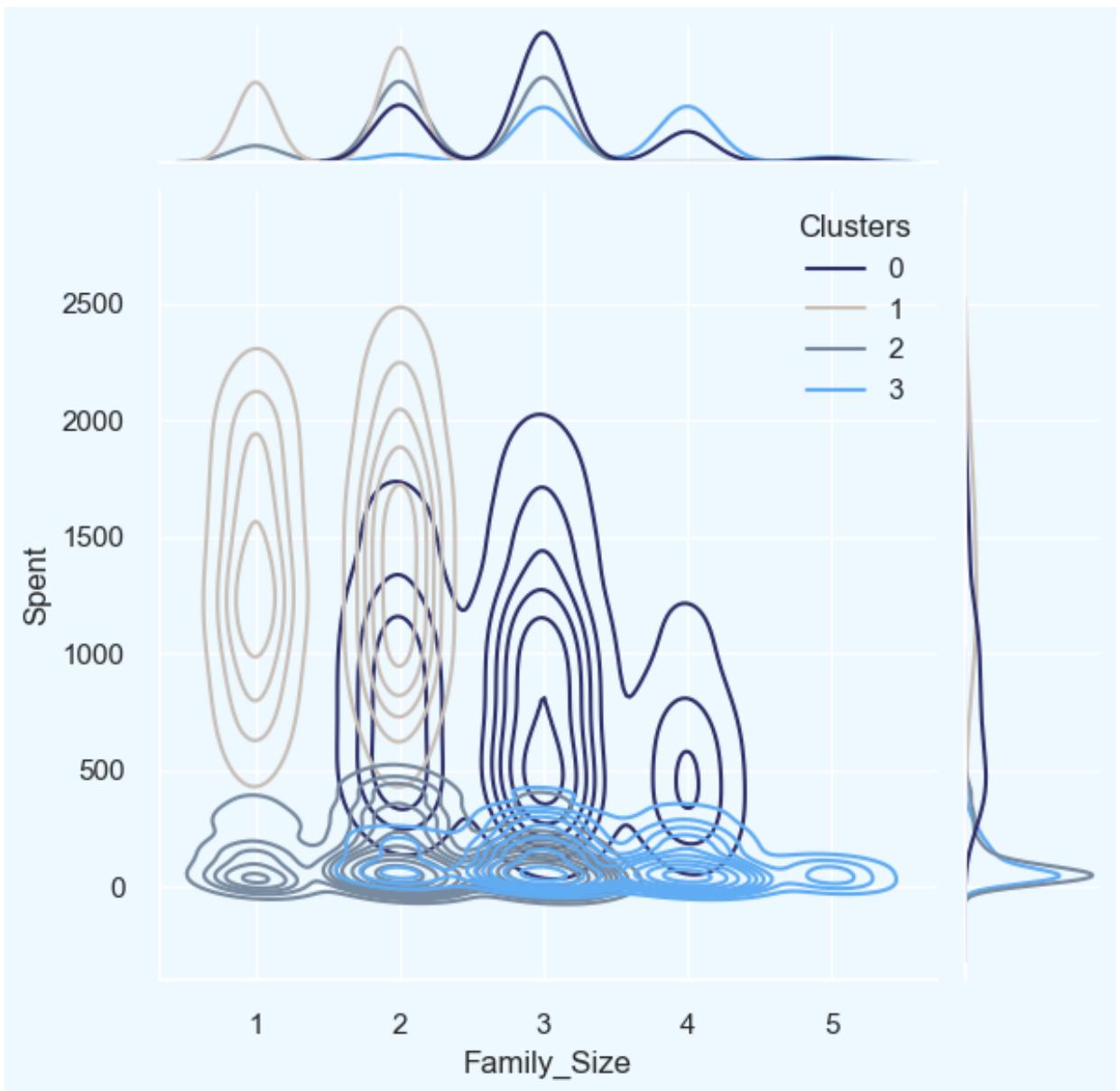
<Figure size 800x550 with 0 Axes>



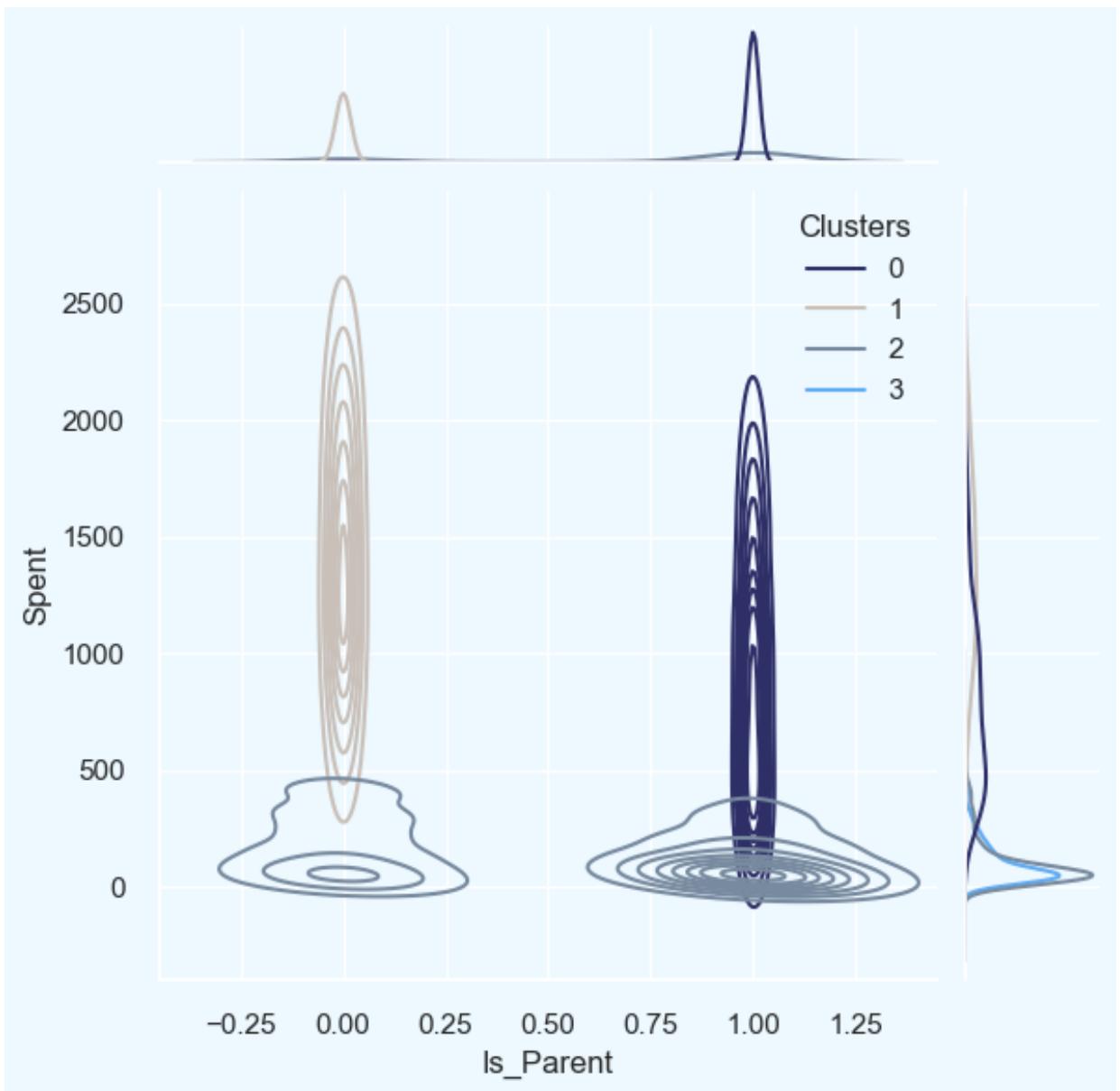
<Figure size 800x550 with 0 Axes>



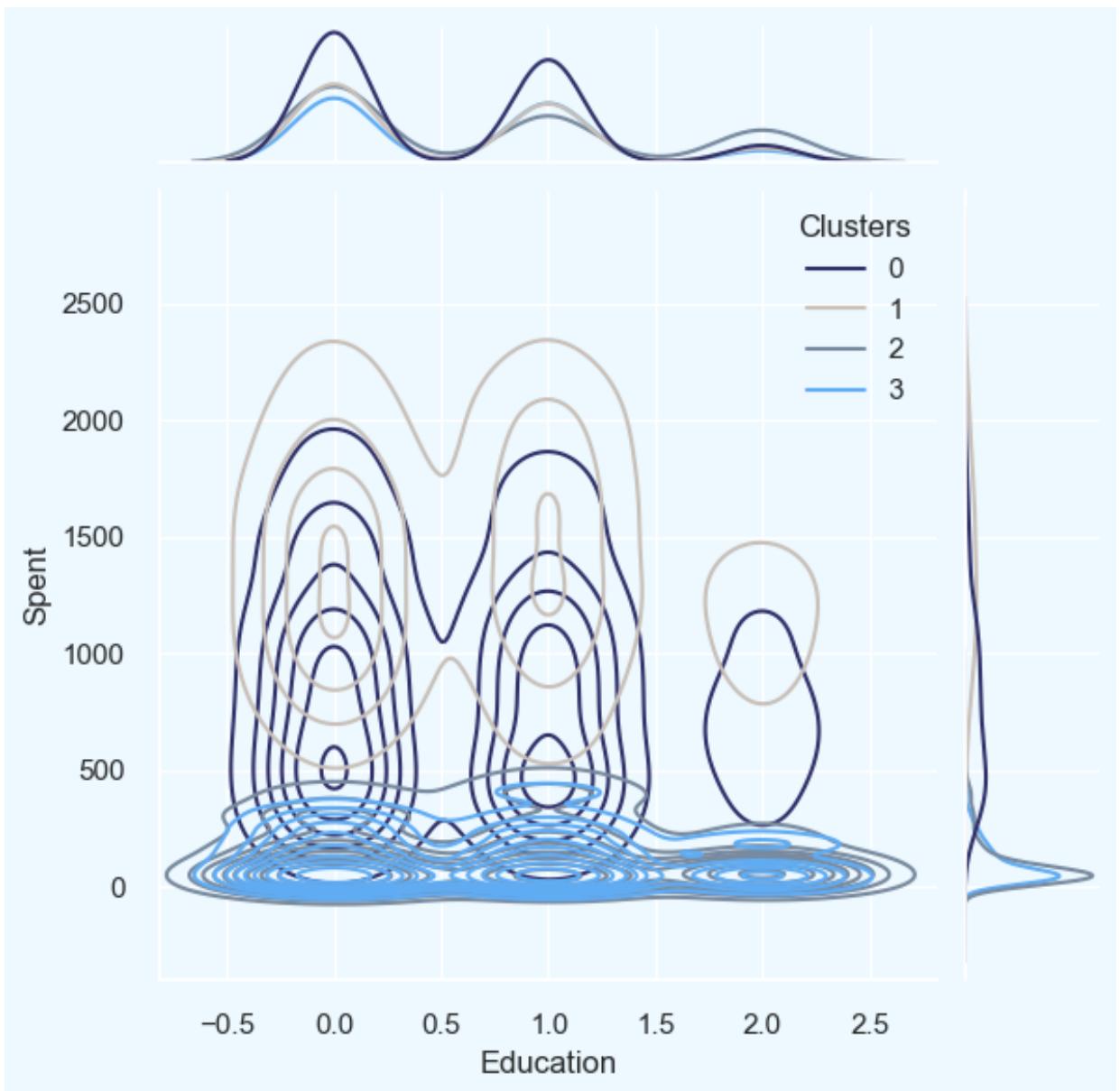
<Figure size 800x550 with 0 Axes>



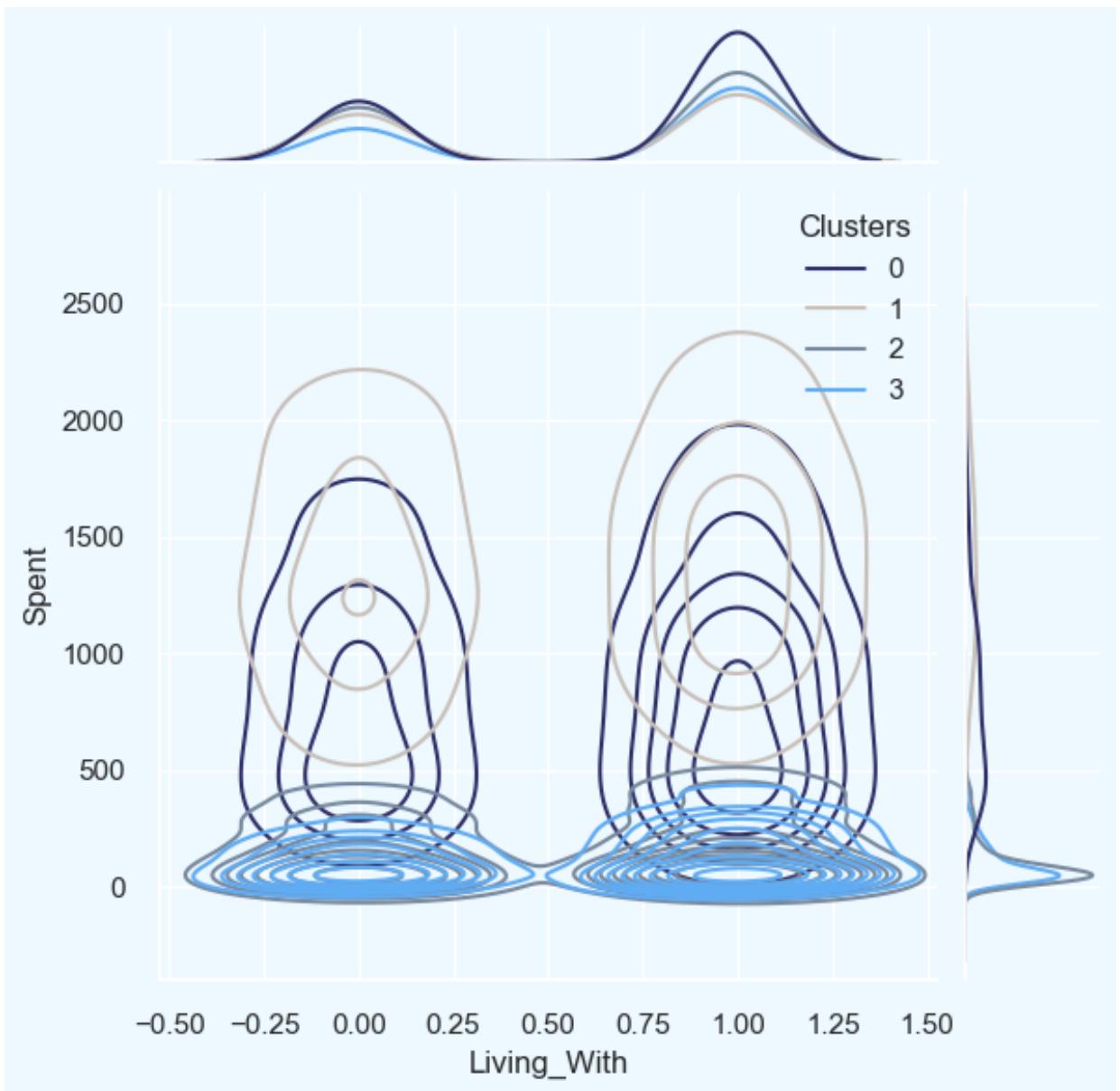
<Figure size 800x550 with 0 Axes>



<Figure size 800x550 with 0 Axes>



<Figure size 800x550 with 0 Axes>



## **Observations:**

**About Cluster Number: 0** As shown in the chart, it can be concluded that the group being discussed is a group of parents who are relatively old and have a family with a maximum of four members and at least two members. Most of the parents in this group have a teenager at home, and single parents are a subset of this group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

**About Cluster Number: 1** According to the data in the charts, it can be concluded that the group being discussed is a group of non-parents who have a maximum of two members in their families. The majority of this group consists of couples, rather than single people. The members of this group span a wide range of ages and are part of a high-income group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

**About Cluster Number: 2** The charts indicate, it can be concluded that the group being discussed is a group of parents who are relatively younger and have families with a maximum of three members. Most of these parents have one child, who is typically not a teenager. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

**About Cluster Number: 3** The data in the charts suggests, it can be concluded that the group being discussed is a group of parents who are relatively older and have a family with a maximum of five members and at least two members. Most of these parents have a teenager at home, and they are part of a lower-income group. However, please note that this is only a conclusion based on the information provided, and it may not be accurate in all cases. It is important to verify and confirm any information before making conclusions or decisions based on it.

In [ ]: