Project Report

On

Customer Personality Analysis

Submitted in partial fulfilment of the requirements for the award of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

(Artificial Intelligence & Machine Learning)

by

Ms. I SIVANI – 22WH1A6613

Ms. DIVYA SAAHITHYA N – 22WH1A6616

Ms. D LASYA– 22WH1A6617

Ms. K SREEJA – 22WH1A6657

Under the esteemed guidance of

Ms. A Naga Kalyani

Assistant Professor, CSE(AI&ML)

BVRIT HYDERABAD College of Engineering for Women

(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)

(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)

Bachupally, Hyderabad – 500090

2024-25

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with A Grade

Bachupally, Hyderabad – 500090

2024-25

CERTIFICATE

This is to certify that the major project entitled **"Customer Personality Analysis using python"** is a bonafide work carried out **by Ms. I Sivani (22wh1a6613), Ms. Divya Saahithya N(22wh1a6616), Ms. D Lasya(22wh5a6617), Ms. K Sreeja (22wh1a6657)** in partial fulfillment for the award of B. Tech degree in **Computer Science & Engineering (AI&ML), BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad,** affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor
Ms. A Naga Kalyani
Assistant Professor

Dept of CSE(AI&ML)

Head of the Department
Dr. B. Lakshmi Praveena
HOD & Professor

Dept of CSE(AI&ML)

External Examiner

DECLARATION

We hereby declare that the work presented in this project entitled **"Global earthquake prediction using python"** submitted towards completion of Project work in IV Year of B.Tech of CSE(AI&ML) at **BVRIT HYDERABAD College of Engineering for Women,** Hyderabad is an authentic record of our original work carried out under the guidance of **Ms. A Naga Kalyani, Assistant Professor, Department of CSE(AI&ML).**

Sign with Date:

I Sivani

(22wh1a6613)

Sign with Date:

Divya Saahithya N

(22wh1a6616)

Sign with Date:

D Lasya

(22wh1a6617)

Sign with Date:

K Sreeja

(22wh1a6657)

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. B. Lakshmi Praveena, Head of the Department, Department of CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide**, Ms. A Naga Kalyani, Assistant Professor, CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for her constant guidance and encouragement throughout the project.

Finally, we would like to thank our Major Project Coordinator, all Faculty and Staff of CSE(AI&ML) department who helped us directly or indirectly. Last but not least, we wish to acknowledge our **Parents and Friends** for giving moral strength and constant encouragement.

I Sivani (22wh1a6613)

Divya Saahithya N (22wh1a6616)

D Lasya(22wh1a6617)

K Sreeja (2wh1a6657)

ABSTRACT

The Customer Personality Analysis project utilizes a dataset of customer profiles to perform an in-depth analysis of their behavior and preferences. The dataset includes features such as income, marital status, education, number of purchases, and responses to marketing campaigns. The study involves data cleaning, preprocessing, and the creation of derived features to enhance the dataset's predictive potential.

Exploratory Data Analysis (EDA) uncovers key trends, such as correlations between income, expenses, and purchase frequency, and highlights the distribution of demographic and behavioral features. By identifying outliers and addressing missing data, we ensure robust and reliable insights.

The results reveal that customer spending is closely tied to income levels and that specific demographic groups exhibit distinct purchasing behaviors. Cluster analysis is used to segment customers into groups with shared characteristics, enabling businesses to implement targeted strategies.

This analysis provides actionable insights for businesses to personalize marketing efforts, optimize resource allocation, and build stronger relationships with their customer base.

PROBLEM STATEMENT

In the competitive landscape of modern business, understanding customer behavior and preferences is critical for personalizing services, optimizing marketing campaigns, and enhancing customer satisfaction. The Customer Personality Analysis aims to leverage customer data to uncover insights about spending habits, purchasing behaviors, and engagement patterns. These insights can help businesses segment customers, predict responses to campaigns, and tailor their strategies for improved customer retention and profitability.

This analysis focuses on identifying key customer characteristics such as income levels, education, marital status, age, spending patterns, and responses to marketing campaigns.

DATA SET

Customer Personality Analysis– Kaggle

https://www.kaggle.com/datasets/imakash3011/customer-personality-analysis

SOURCE CODE

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


df=pd.read_csv(r'/content/drive/MyDrive/marketing_campaign.csv',sep='\t')
df

new=df.copy()


df.sample(5)

df.info()

df.shape

df.describe()

df.isnull().sum()

df.dropna(inplace=True) #fill null values
df.isnull().sum()

df.duplicated().sum()

df['Z_CostContact'].value_counts()

df['Z_Revenue'].value_counts()

df.drop(columns = ['ID', 'Z_CostContact', 'Z_Revenue'], inplace=True)
```

```
df.head()

df.head()
#Calculate the age from Birth Year and cutomer age in company from dt_Customer columns
# calculate customer edge
df['Dt_Customer'] = pd.to_datetime(df.Dt_Customer, format="%d-%m-%Y")
last_date = df['Dt_Customer'].max()
df['Days_is_client'] = (last_date - df['Dt_Customer']).dt.days

df['Cus_Age'] = (last_date.year - df['Year_Birth'])

df.head()

df.drop(columns = ['Year_Birth', 'Dt_Customer'], inplace = True)

df['Education'].value_counts()

df['Marital_Status'].value_counts()

df['Education'].replace({'PhD': 'Postgraduate',
'Master': 'Postgraduate',
'Graduation': 'Graduate',
'2n Cycle': 'Graduate',
'Basic': 'Undergraduate'
}, inplace=True)

df['Marital_Status'].replace({'Married': 'Partner',
'Together': 'Partner',
'Single': 'Single',
'Divorced': 'Single',
'Widow': 'Single',
'Alone': 'Single',
'Absurd': 'Single',
'YOLO': 'Single'
}, inplace=True)
```

Create a new features represeting a total amount spents

```
df['Expenses'] = df['MntWines'] + df['MntFruits'] + df['MntMeatProducts'] + df['MntFishProducts'] +
df['MntSweetProducts'] + df['MntGoldProds']
df.head()

# Combine a columns regarding a number of children, Campaigns accepted, and number of purchases
df['Kids'] = df['Kidhome'] + df['Teenhome']
```

```
df['TotalAcceptedCmp'] = df['AcceptedCmp1'] + df['AcceptedCmp2'] + df['AcceptedCmp3'] +
df['AcceptedCmp4'] + df['AcceptedCmp5']

df['TotalNumPurchases'] = df['NumWebPurchases'] + df['NumCatalogPurchases'] +
df['NumStorePurchases'] + df['NumDealsPurchases']

data = df.copy()
```

Select necessary columns

```
#Select necessary columns
necessary_columns = ['Education', 'Marital_Status', 'Income', 'Kids', 'Days_is_client', 'Recency', 'Expenses',
'Cus_Age',
'TotalNumPurchases', 'TotalAcceptedCmp', 'Complain', 'Response']

df = df[necessary_columns]
df.head()

df.shape

print('Dupliacted rows',df.duplicated().sum())
print('Null values',df.isnull().sum())
print('shape of data: ', df.shape)

# Categorize columns into three groups based on their data type

binary_columns = [col for col in df.columns if df[col].nunique() == 2]
categorical_columns = [col for col in df.columns if 2 < df[col].nunique() < 10]
numerical_columns = [col for col in df.select_dtypes(include=['number']).columns
if col not in binary_columns + categorical_columns]

import seaborn as sns
from scipy.stats import boxcox, zscore
from scipy.special import inv_boxcox

from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.model_selection import GridSearchCV

from sklearn.cluster import KMeans
```

```
# detecting outliers
z_scores = pd.DataFrame(zscore(df[numerical_columns]), columns=numerical_columns)
z_scores

# Identify rows where any of the z-scores exceed the threshold
outliers = z_scores[(np.abs(z_scores) > 3).any(axis=1)]
outliers

# Drop the rows containing outliers
df = df.drop(outliers.index)
```

EDA

```
x=1
plt.figure(figsize=(20,8))
for col in numerical_columns:
plt.subplot(len(numerical_columns)//2, len(numerical_columns)//2, x)
sns.histplot(data=df, x=col, kde=True, color='blue')
plt.title(f'Histogram of {col}', pad=10, fontweight='bold', fontsize=10)
plt.tight_layout()
x+=1

# Define the color palette
custom_palette = ["#327D7C", "#E2504A", "#F0C808"]

x=1
plt.figure(figsize=(20,15))
for col in categorical_columns + binary_columns:
plt.subplot(3,3,x)
ax=sns.countplot(data=df, x=col, palette=custom_palette)
plt.title(f"{col} Distribution",pad=10,fontweight="bold",fontsize=12)
ax.bar_label(ax.containers[0])
plt.tight_layout()
x+=1

colors = ["#327D7C", "#E2504A", "#F0C808"]

corr_matrix = df.select_dtypes(include='number').corr()
plt.figure(figsize=(10,10))
sns.heatmap(data=corr_matrix, annot=True, color=colors, cmap='coolwarm', fmt=".1g")
plt.show()
```

Key findings from the visualizations:

* Income Distribution: After removing outliers, income follows a normal distribution, suggesting that most customers earn around the average income, with fewer customers earning significantly more or less.
* Days with Client & Recency: Both features exhibit a fairly uniform distribution, indicating that the customers have been with the company for varying lengths of time and have recently interacted with the company across a wide range of time periods.

* Expenses Distribution: Expenses show an exponential distribution, which means a majority of customers have lower spending, with spending rapidly decreasing as the amount increases.
* Total Number of Purchases: This feature follows a binomial distribution, reflecting that there are common purchasing behaviors among customers, such as making a specific number of purchases.

* Total Number of Purchases: This feature follows a binomial distribution, reflecting that there are common purchasing behaviors among customers, such as making a specific number of purchases.
* Correlated Features: Income, expenses, and the total number of purchases are the most correlated features, suggesting that higher income is closely linked with higher spending and a greater number of purchases.

```
df_copy = df.copy()

categorical_columns = df_copy.select_dtypes(include='object').columns.tolist()

x_encoded = pd.get_dummies(df, columns=categorical_columns, drop_first=True, dtype=int)

x_encoded.info()

scale = StandardScaler()
x_scaled = scale.fit_transform(x_encoded)

x_scaled.shape
```

MODEL BUILDING

```
kmeans = KMeans(n_clusters=3, init='k-means++', n_init=10, max_iter=300, random_state=42)
kmeans.fit(x_scaled)
y_kmeans = kmeans.fit_predict(x_scaled)
inertia = kmeans.inertia_

kmeans_inertias = []

for n in np.arange(1,10,1):
kmeans = KMeans(n_clusters=n, init='k-means++', n_init=10, max_iter=300, random_state=42)
kmeans.fit(x_scaled)
inertia = kmeans.inertia_
kmeans_inertias.append(inertia)
```

```
plt.plot(np.arange(1,10,1), kmeans_inertias, marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()

from sklearn.metrics import silhouette_score

silhouette_scores = []
for k in range(2, 11):
kmeans = KMeans(n_clusters=k, init='k-means++', n_init=10, max_iter=300, random_state=42)
kmeans.fit(x_scaled)
score = silhouette_score(x_scaled, kmeans.labels_)
silhouette_scores.append(score)

# Plotting the silhouette scores
plt.plot(range(2, 11), silhouette_scores, marker='o')
plt.title('Silhouette Score Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Silhouette Score')
plt.show()

kmeans_final = KMeans(n_clusters=3, init='k-means++', n_init=10, max_iter=300, random_state=42)
kmeans_final.fit(x_scaled)
y_label = kmeans_final.labels_
y_kmeans = kmeans_final.fit_predict(x_scaled)
print(kmeans_final.inertia_)

x_original = scale.inverse_transform(x_scaled)

df_clusters = df.copy()
df_clusters['Cluster'] = y_kmeans
df_clusters.head()

columns_to_plot = ['Income', 'Kids', 'Days_is_client', 'Recency', 'Expenses', 'TotalNumPurchases',
'TotalAcceptedCmp']

x=1
plt.figure(figsize=(30,35))
for col in columns_to_plot:
plt.subplot(4,2,x)
sns.boxplot(data=df_clusters, x='Cluster', y=col, palette='Set2')
plt.title(f'Boxplot of {col} by Cluster', pad=10, fontweight='bold', fontsize=12)
plt.xlabel('Cluster', fontsize=16)
plt.ylabel(f"{col}" ,fontsize=16)
plt.tight_layout()
x+=1
```

```
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6)) # Adjust figure size as needed
plt.scatter(df_clusters['Income'], df_clusters['Expenses'], c=df_clusters['Cluster'], cmap='viridis')
plt.xlabel('Income')
plt.ylabel('Expenses')
plt.title('Scatter Plot of Income vs. Expenses, colored by Cluster')
plt.colorbar(label='Cluster')
plt.show()
```

Based on our analysis
1. Cluster 0:
* Less Income Group
* 1 to 2 kids
* less expenses
* less no of purchases
* Doesn't accept anything from promotions

2. Cluster 1:
* Medium Income Group
* 0 or 1 kids
* medium expenses
* average no of purchases
* Doesn't accept anything from promotions

1. Cluster 2:
* Higher Income Group
* 0 or 1 kids
* Highest Expenses
* Highest Number of Purchases
* Ocassionally accept from promotions

OUTPUT

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	...	7
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	...	5
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	...	4
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	...	6
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	...	5
...	...	...	...	...	...	...	...	...	...	...	...	...
2235	10870	1967	Graduation	Married	61223.0	0	1	13-06-2013	46	709	...	5
2236	4001	1946	PhD	Together	64014.0	2	1	10-06-2014	56	406	...	7
2237	7270	1981	Graduation	Divorced	56981.0	0	0	25-01-2014	91	908	...	6
2238	8235	1956	Master	Together	69245.0	0	1	24-01-2014	8	428	...	3
2239	9405	1954	PhD	Married	52869.0	1	1	15-10-2012	40	84	...	7

2240 rows × 29 columns

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth
1696	1890	1971	2n Cycle	Together	42033.0	1	1	19-09-2012	95	11	...	7
375	10703	1975	Master	Single	46098.0	1	1	18-08-2012	86	57	...	8
551	5371	1989	Graduation	Single	21474.0	1	0	08-04-2014	0	6	...	7
1706	1351	1956	Master	Together	58656.0	0	1	20-09-2012	25	962	...	6
346	8553	1965	Graduation	Married	44300.0	1	1	23-06-2013	65	30	...	6

5 rows × 29 columns

```
<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
```

	ID	Year_Birth	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts
count	2240.000000	2240.000000	2216.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000
mean	5592.159821	1968.805804	52247.251354	0.444196	0.506250	49.109375	303.935714	26.302232	166.950000
std	3246.662198	11.984069	25173.076661	0.538398	0.544538	28.962453	336.597393	39.773434	225.715373
min	0.000000	1893.000000	1730.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2828.250000	1959.000000	35303.000000	0.000000	0.000000	24.000000	23.750000	1.000000	16.000000
50%	5458.500000	1970.000000	51381.500000	0.000000	0.000000	49.000000	173.500000	8.000000	67.000000
75%	8427.750000	1977.000000	68522.000000	1.000000	1.000000	74.000000	504.250000	33.000000	232.000000
max	11191.000000	1996.000000	666666.000000	2.000000	2.000000	99.000000	1493.000000	199.000000	1725.000000

8 rows × 26 columns

	0
ID	0
Year_Birth	0
Education	0
Marital_Status	0
Income	24
Kidhome	0
Teenhome	0
Dt_Customer	0
Recency	0
MntWines	0
MntFruits	0
MntMeatProducts	0
MntFishProducts	0
MntSweetProducts	0
MntGoldProds	0
NumDealsPurchases	0
NumWebPurchases	0
NumCatalogPurchases	0

	count
Z_CostContact	
3	2216

dtype: int64

	count
Z_Revenue	
11	2216

dtype: int64

	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	...
0	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	88	...
1	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	1	...
2	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	49	...
3	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	4	...
4	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	43	...

5 rows × 26 columns

	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	...
0	1957	Graduation	Single	58138.0	0	0	2012-09-04	58	635	88	...
1	1954	Graduation	Single	46344.0	1	1	2014-03-08	38	11	1	...
2	1965	Graduation	Together	71613.0	0	0	2013-08-21	26	426	49	...
3	1984	Graduation	Together	26646.0	1	0	2014-02-10	26	11	4	...
4	1981	PhD	Married	58293.0	1	0	2014-01-19	94	173	43	...

5 rows × 28 columns

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

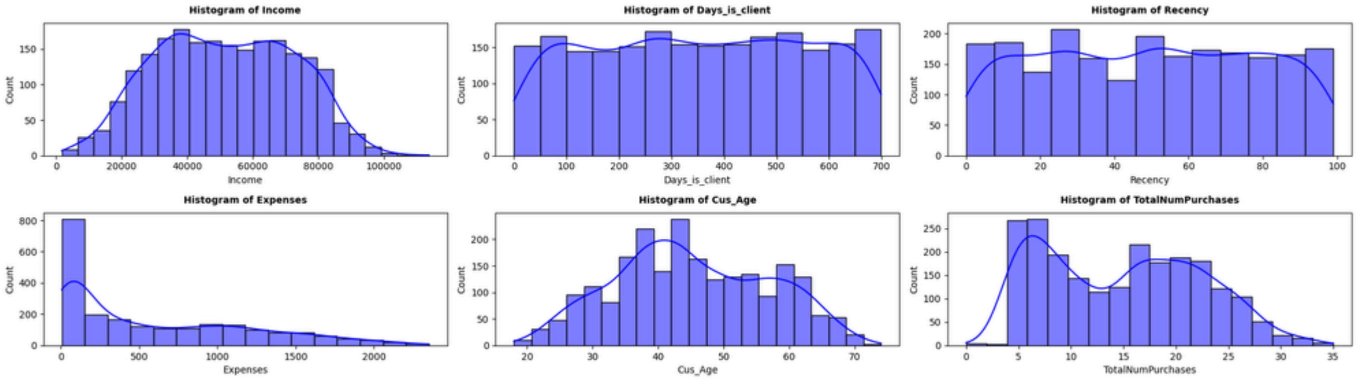
dtype: int64

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

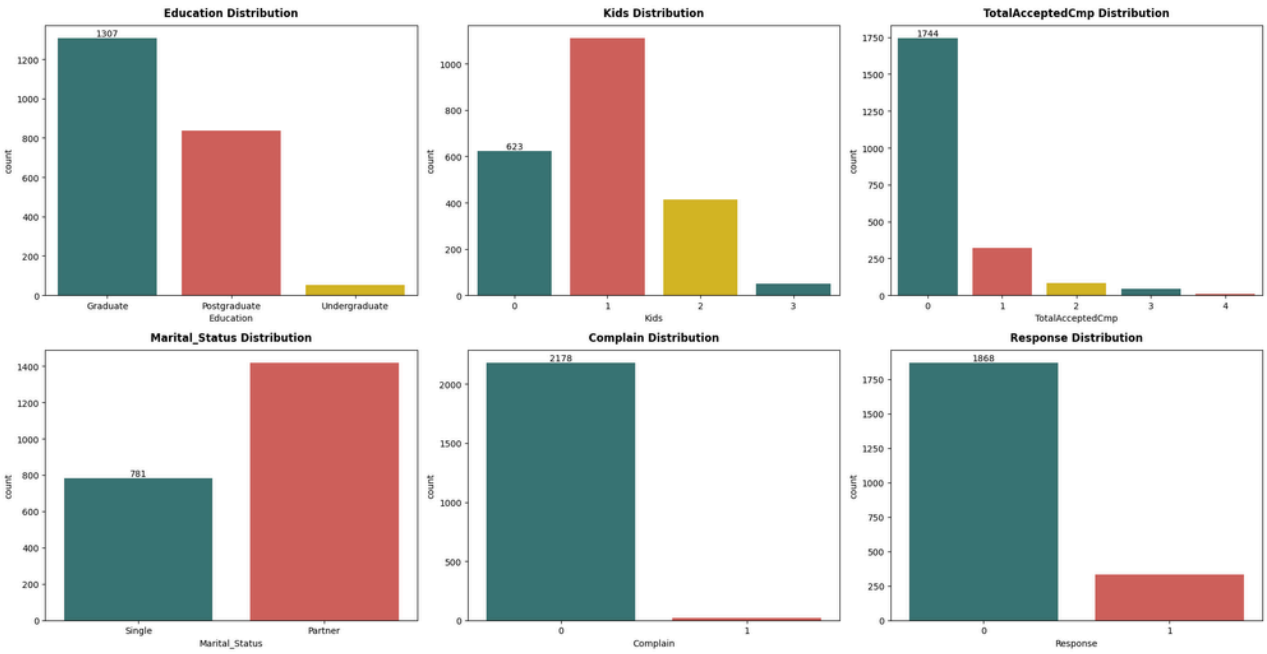
dtype: int64

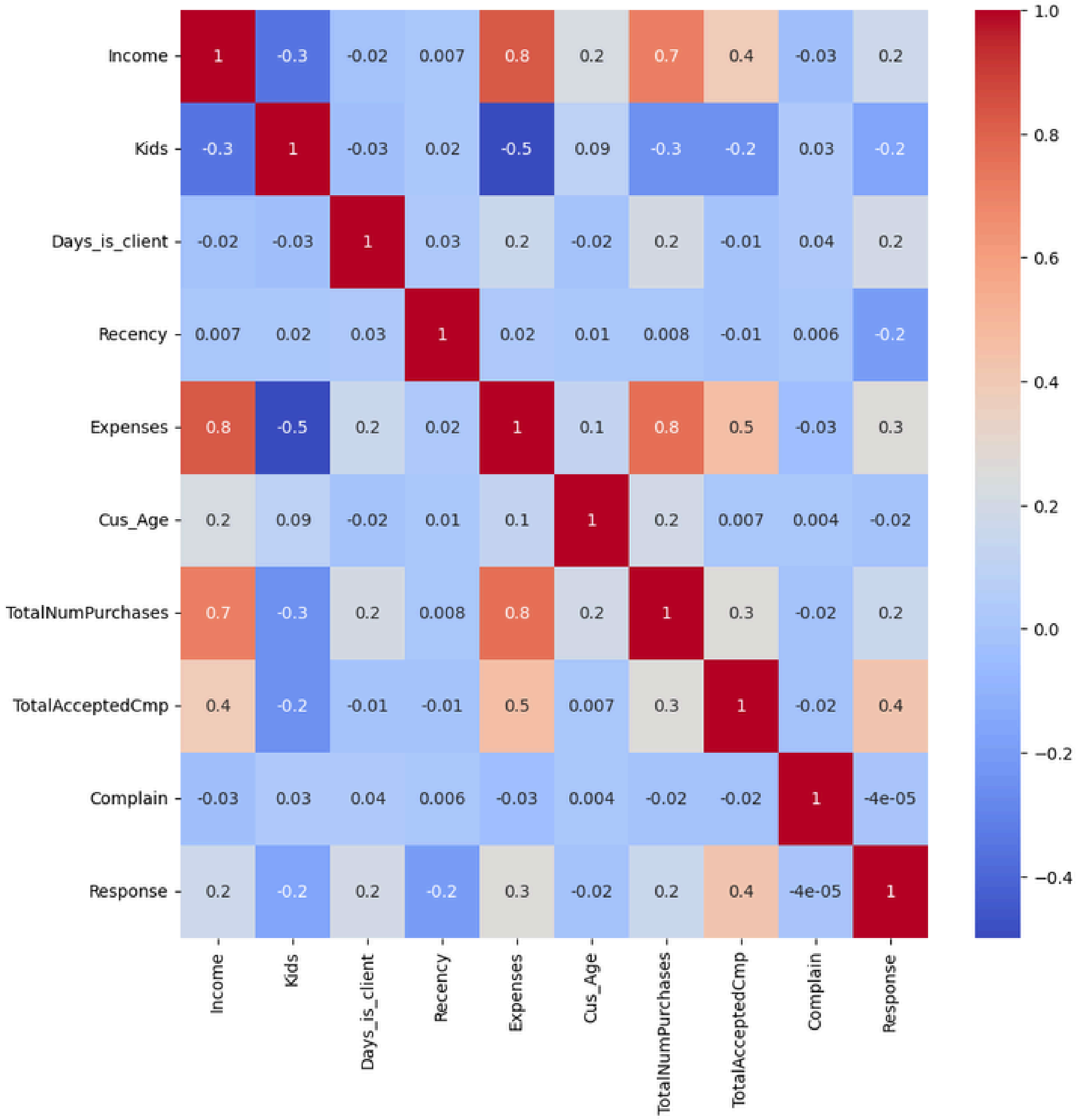
```
<ipython-input-75-4862c53ca570>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  df['Education'].replace({'PhD': 'Postgraduate',
<ipython-input-75-4862c53ca570>:8: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  df['Marital_Status'].replace({'Married': 'Partner',
```

```
Dupliacted rows 185
Null values Education
Marital_Status        0
Income                0
Kids                  0
Days_is_client        0
Recency               0
Expenses              0
Cus_Age               0
TotalNumPurchases     0
TotalAcceptedCmp      0
Complain              0
Response              0
dtype: int64
shape of data:  (2216, 12)
```

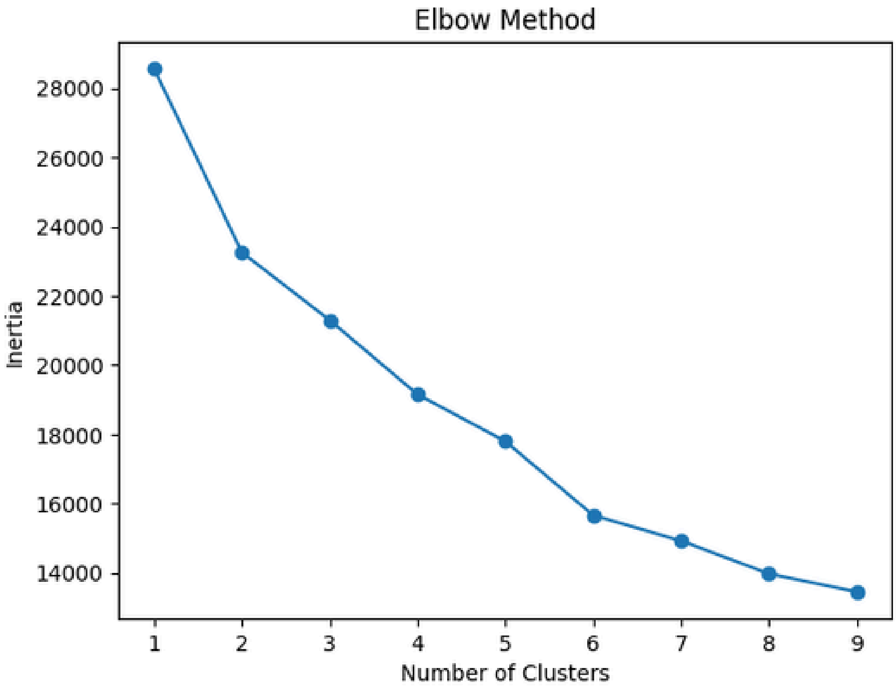


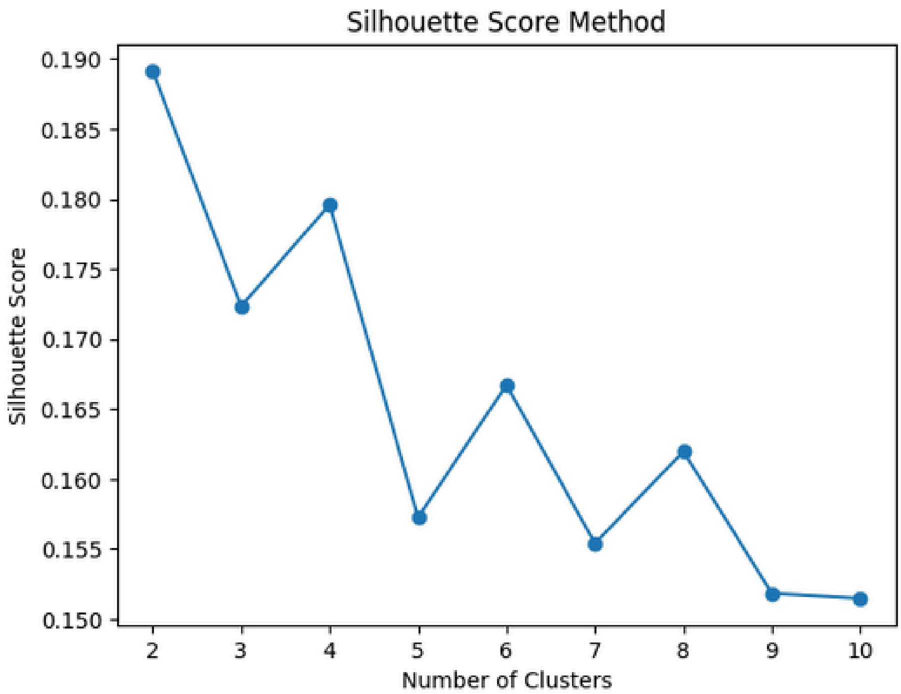

```
<ipython-input-85-2a60e7aa1c9b>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: UserWarning:
The palette list has fewer values (3) than needed (4) and will cycle, which may produce an uninterpretable plot.
  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: UserWarning:
The palette list has fewer values (3) than needed (5) and will cycle, which may produce an uninterpretable plot.
  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: UserWarning: The palette list has more values (3) than needed (2), which may not be intended.
  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  ax=sns.countplot(data=df, x=col, palette=custom_palette)
<ipython-input-85-2a60e7aa1c9b>:8: UserWarning: The palette list has more values (3) than needed (2), which may not be intended.
  ax=sns.countplot(data=df, x=col, palette=custom_palette)
```



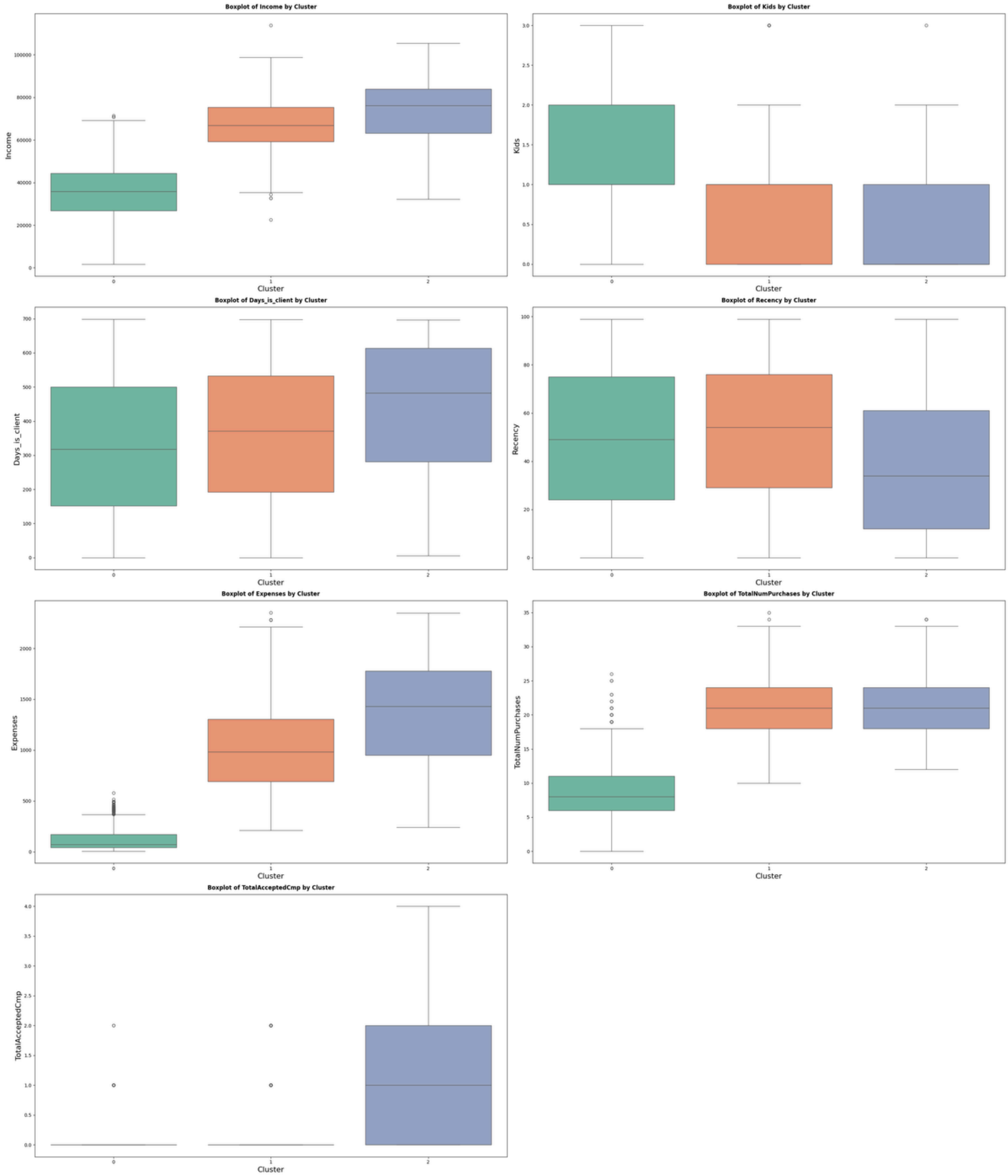


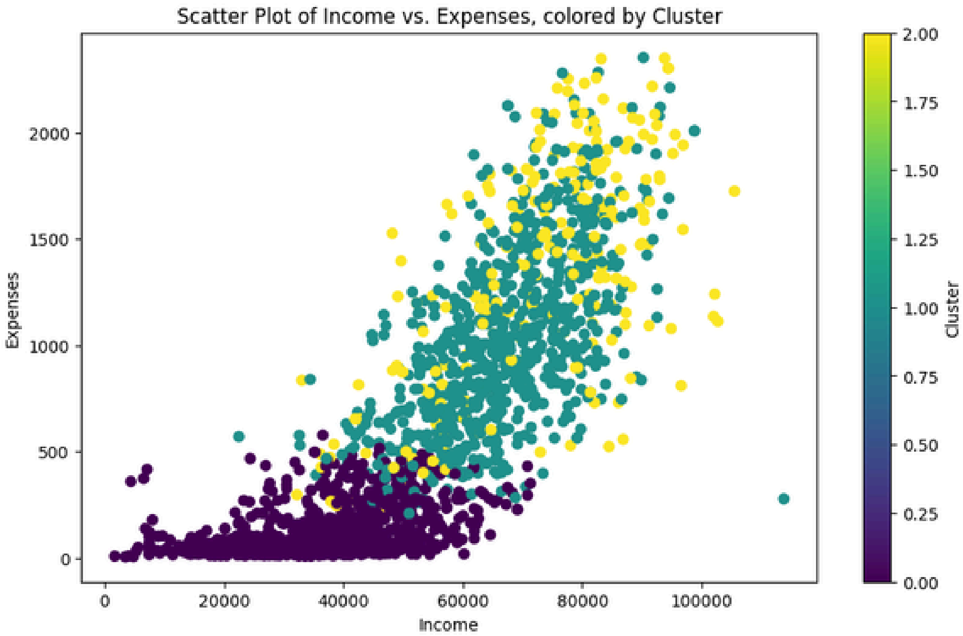

```
<class 'pandas.core.frame.DataFrame'>
Index: 2198 entries, 0 to 2239
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Income                   2198 non-null   float64
 1   Kids                     2198 non-null   int64  
 2   Days_is_client           2198 non-null   int64  
 3   Recency                  2198 non-null   int64  
 4   Expenses                 2198 non-null   int64  
 5   Cus_Age                  2198 non-null   int64  
 6   TotalNumPurchases        2198 non-null   int64  
 7   TotalAcceptedCmp         2198 non-null   int64  
 8   Complain                 2198 non-null   int64  
 9   Response                 2198 non-null   int64  
 10  Education_Postgraduate   2198 non-null   int64  
 11  Education_Undergraduate  2198 non-null   int64  
 12  Marital_Status_Single    2198 non-null   int64  
dtypes: float64(1), int64(12)
memory usage: 240.4 KB
```





	Education	Marital_Status	Income	Kids	Days_is_client	Recency	Expenses	Cus_Age	TotalNumPurchases	TotalAcc
0	Graduate	Single	58138.0	0	663	58	1617	57	25	
1	Graduate	Single	46344.0	2	113	38	27	60	6	
2	Graduate	Partner	71613.0	0	312	26	776	49	21	
3	Graduate	Partner	26646.0	1	139	26	53	30	8	
4	Postgraduate	Partner	58293.0	1	161	94	422	33	19	


Boxplot of Income by Cluster
Income
Cluster
Boxplot of Kids by Cluster
Kids
Cluster
Boxplot of Days_is_client by Cluster
Days_is_client
Cluster
Boxplot of Recency by Cluster
Recency
Cluster
Boxplot of Expenses by Cluster
Expenses
Cluster
Boxplot of TotalNumPurchases by Cluster
TotalNumPurchases
Cluster
Boxplot of TotalAcceptedCmp by Cluster
TotalAcceptedCmp
Cluster



Scatter Plot of Income vs. Expenses, colored by Cluster
Expenses
Income
Cluster
0
500
1000
1500
2000
20000
40000
60000
80000
100000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00