Team Members

Role	ID	Name
Evaluation and Interpretation	22010323	أمنية محمد عبد الهادي أحمد زايد
Exploratory Data Analysis (EDA)	22011609	نور الدين سامر عبد الرزاق
Applying K-medoids Clustering	22011606	محمد عماد مبروك
Applying Hierarchical Clustering	22011589	محمد مصطفى محمد عبد المجيد
Dataset Selection & Data cleaning	22011547	سارة شريف حسن محمد

Overview:

This dataset is a detailed analysis of a company's ideal customers. It helps a business to better understand its customers and makes it easier for them to modify products according to the specific needs, behaviors and concerns of different types of customers.

Objective

The goal of this project is to segment customers based on their behavior to modify the company products to better suit the target segment of customers.

The Process

The dataset displays the data of over 2000 customers and a few attributes, such as their annual income, their age, their most recent purchase, their highest educational level, their marital status and the number of kids they have, and a few more attributes that are irrelevant to our task at hand. And now we'll walk you through the process step by step.

- Importing necessary libraries

```
[1]: import numpy as np
     import pandas as pd
     import datetime
     import matplotlib
     import matplotlib.pyplot as plt
     import seaborn as sns
     import sklearn_extra
     from scipy.cluster import hierarchy
     from scipy.cluster.hierarchy import linkage, dendrogram
     from sklearn_extra.cluster import KMedoids
     from sklearn.cluster import AgglomerativeClustering
     from sklearn.preprocessing import StandardScaler
```

- Importing csv file and displaying the data

```
[3]: df = pd.read_csv("C:\\Users\\pc\\Downloads\\marketing_campaign.csv", sep="\t")
     df.head()
```

[3]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth	AcceptedCmp3	AcceptedCm
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	...	7	0	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	...	5	0	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	...	4	0	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	...	6	0	
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	...	5	0	

5 rows × 29 columns

- Displaying data information

```
[3]: df.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
```

- There’re some unnecessary columns so we’ll drop them and drop duplicates as well

```
[4]: df.drop(df.columns[15:25], axis=1, inplace=True)
     df.drop(df.columns[16:19], axis=1, inplace=True)
```

```
[5]: df.drop_duplicates(inplace=True)
     df.dropna(inplace=True)

     df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2216 entries, 0 to 2239
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   ID                2216 non-null   int64
 1   Year_Birth        2216 non-null   int64
 2   Education         2216 non-null   object
 3   Marital_Status    2216 non-null   object
 4   Income            2216 non-null   float64
 5   Kidhome           2216 non-null   int64
 6   Teenhome          2216 non-null   int64
 7   Dt_Customer       2216 non-null   object
 8   Recency           2216 non-null   int64
 9   MntWines          2216 non-null   int64
 10  MntFruits         2216 non-null   int64
 11  MntMeatProducts   2216 non-null   int64
 12  MntFishProducts   2216 non-null   int64
 13  MntSweetProducts  2216 non-null   int64
 14  MntGoldProds      2216 non-null   int64
 15  Complain          2216 non-null   int64
dtypes: float64(1), int64(12), object(3)
memory usage: 294.3+ KB
```

- There are some inconsistent data types that we need to change such as `ID` and `Dt_Customer`

```
[6]: df["Dt_Customer"] = pd.to_datetime(df["Dt_Customer"], format='mixed')
     df['ID'] = df['ID'].astype(str)
```

- Add columns that could help us and take a look at summary statistics

```
[7]: df['Age']=2014-df['Year_Birth']
     df.drop('Year_Birth',axis=1, inplace=True)

     df['Num_Kids'] = df['Kidhome'] + df['Teenhome']
     df['Amount'] = df['MntWines'] + df['MntFruits'] + df['MntMeatProducts'] + df['MntFishProducts'] + df['MntSweetProducts'] + df[
```

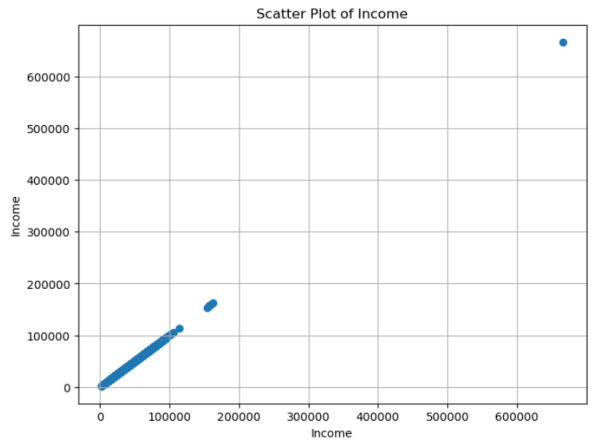
```
[8]: df.describe()
```

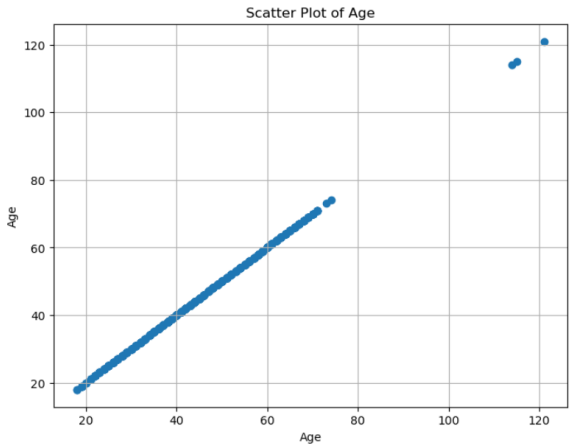
[8]:

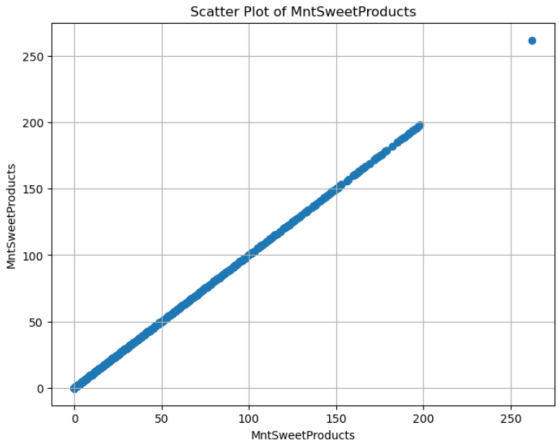
	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProd
count	2216.000000	2216.000000	2216.000000	2216	2216.000000	2216.000000	2216.000000	2216.000000	2216.00
mean	52247.251354	0.441787	0.505415	2013-07-11 23:50:54.151624704	49.012635	305.091606	26.356047	166.995939	37.63
min	1730.000000	0.000000	0.000000	2012-01-08 00:00:00	0.000000	0.000000	0.000000	0.000000	0.00
25%	35303.000000	0.000000	0.000000	2013-01-19 00:00:00	24.000000	24.000000	2.000000	16.000000	3.00
50%	51381.500000	0.000000	0.000000	2013-07-11 00:00:00	49.000000	174.500000	8.000000	68.000000	12.00
75%	68522.000000	1.000000	1.000000	2013-12-31 00:00:00	74.000000	505.000000	33.000000	232.250000	50.00
max	666666.000000	2.000000	2.000000	2014-12-06 00:00:00	99.000000	1493.000000	199.000000	1725.000000	259.00
std	25173.076661	0.536896	0.544181	NaN	28.948352	337.327920	39.793917	224.283273	54.75

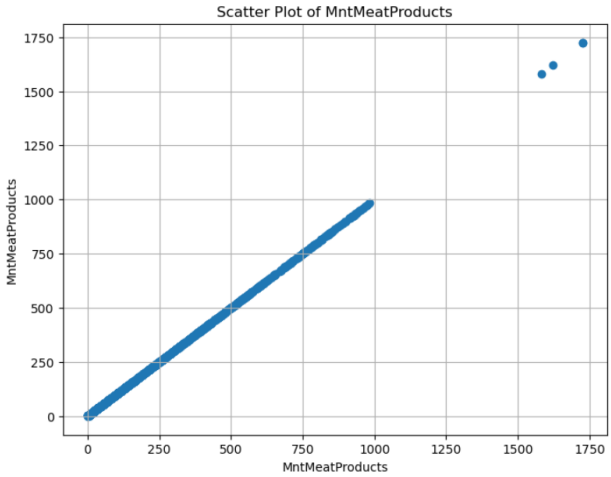
- There seems to be outliers and/or dummy data so we'll use scatterplots to see them more clearly

```
[9]: numerical_columns = df.select_dtypes(include=['int64', 'float64'])
for column in numerical_columns.columns:
    plt.figure(figsize=(8, 6))
    plt.scatter(df[column], df[column])
    plt.title(f'Scatter Plot of {column}')
    plt.xlabel(column)
    plt.ylabel(column)
    plt.grid(True)
    plt.show()
```







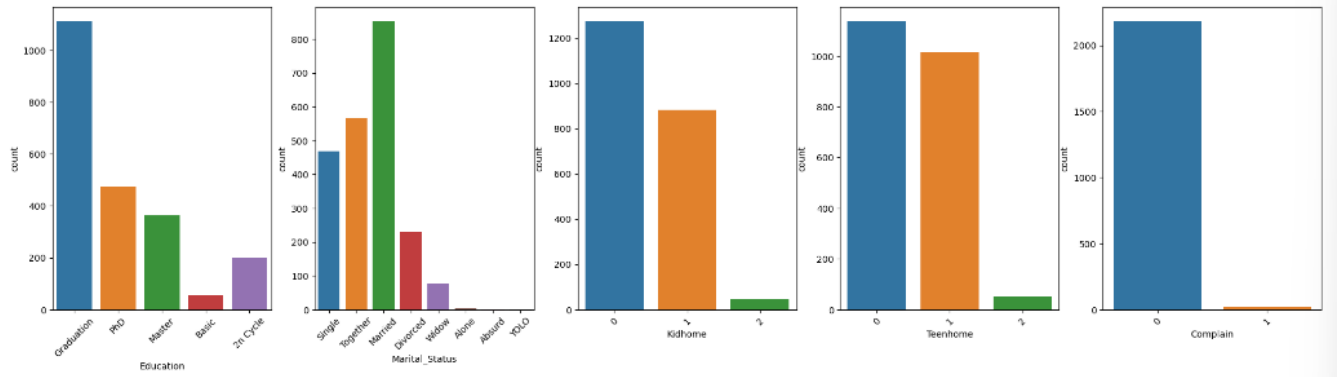


- Some values need to be removed for the data to be consistent

```
[10]: df = df[df["Age"]<90]
df = df[df["Income"]<150000]
df = df[df["MntMeatProducts"]<1500]
df = df[df["MntSweetProducts"]<250]
```

- Some barplots to get a better understanding of the data

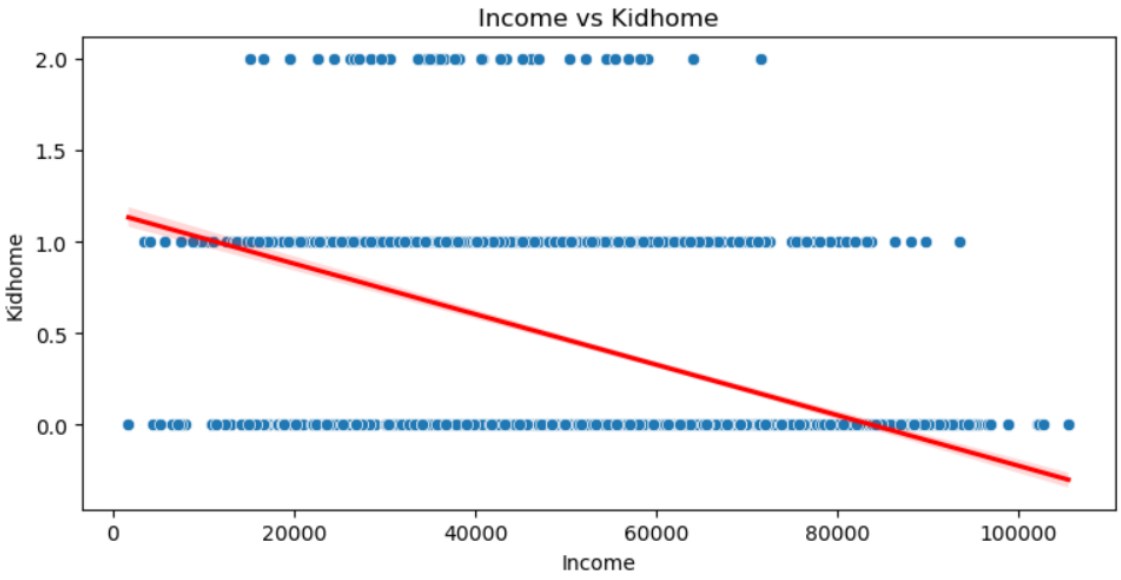
```
cal_cols = ['Education', 'Marital_Status', 'Kidhome', 'Teenhome', 'Complain']
fig, ax = plt.subplots(1, len(cal_cols), figsize=(25, 6))
for col in enumerate(cal_cols):
    sns.countplot(data=df, x=col[1], ax=ax[col[0]])
    # rotate the x-axis labels
    # ax[col[0]].xticks(rotation=45)
    ax[col[0]].tick_params(axis='x', rotation=45)
```

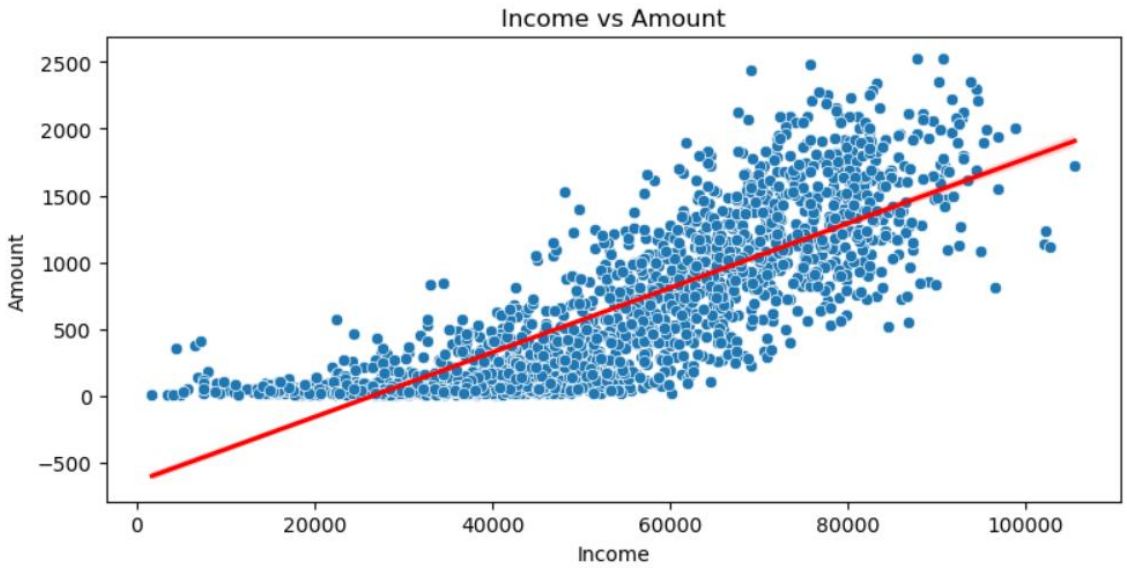


we can draw some conclusions from these barplots such as, most of these customers are college graduates, married with 0 kids and have not complained about the products in 3 years.

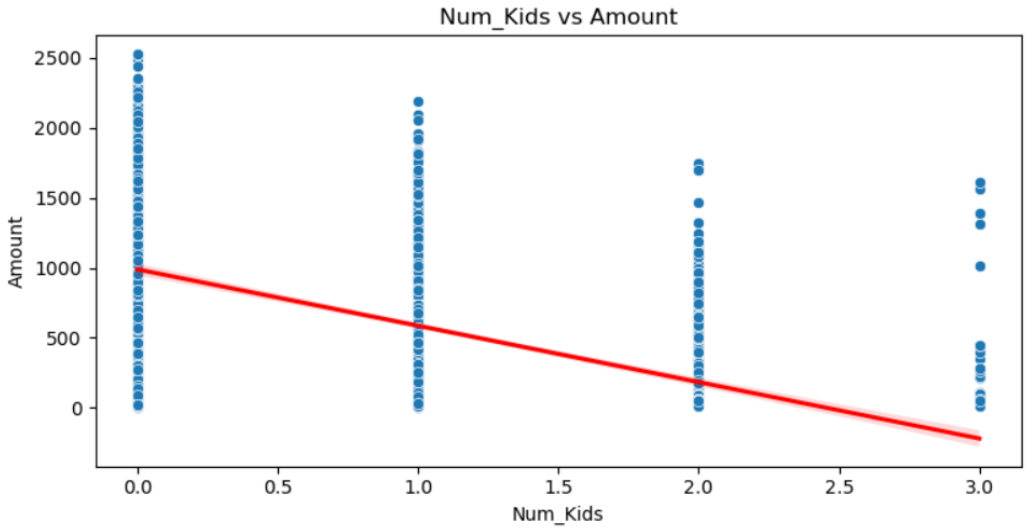
- Some scatterplots to identify interesting relationships between numeric variables

```
numerical_df = df.select_dtypes(include=['int64', 'float64']).drop(columns=['Income'])

fig, axes = plt.subplots(nrows=numerical_df.shape[1], ncols=1, figsize=(8, 4 * numerical_df.shape[1]))

for i, column in enumerate(numerical_df.columns):
    sns.scatterplot(x='Income', y=column, data=df, ax=axes[i])
    axes[i].set_title(f"Income vs {column}")
    sns.regplot(x='Income', y=column, data=df, scatter=False, ax=axes[i], color='r')

plt.tight_layout()
plt.show()
```






```
numerical_df = df.select_dtypes(include=['int64', 'float64']).drop(columns=['Num_Kids', 'Kidhome', 'Teenhome'])

# Create subplots for Income vs all other numerical features
fig, axes = plt.subplots(nrows=numerical_df.shape[1], ncols=1, figsize=(8, 4 * numerical_df.shape[1]))

for i, column in enumerate(numerical_df.columns):
    sns.scatterplot(x='Num_Kids', y=column, data=df, ax=axes[i])
    axes[i].set_title(f"Num_Kids vs {column}")
    sns.regplot(x='Num_Kids', y=column, data=df, scatter=False, ax=axes[i], color='r')

plt.tight_layout()
plt.show()
```



Interesting conclusions we can draw from these scatterplots

1. As the income increases, customers tend to have less children.

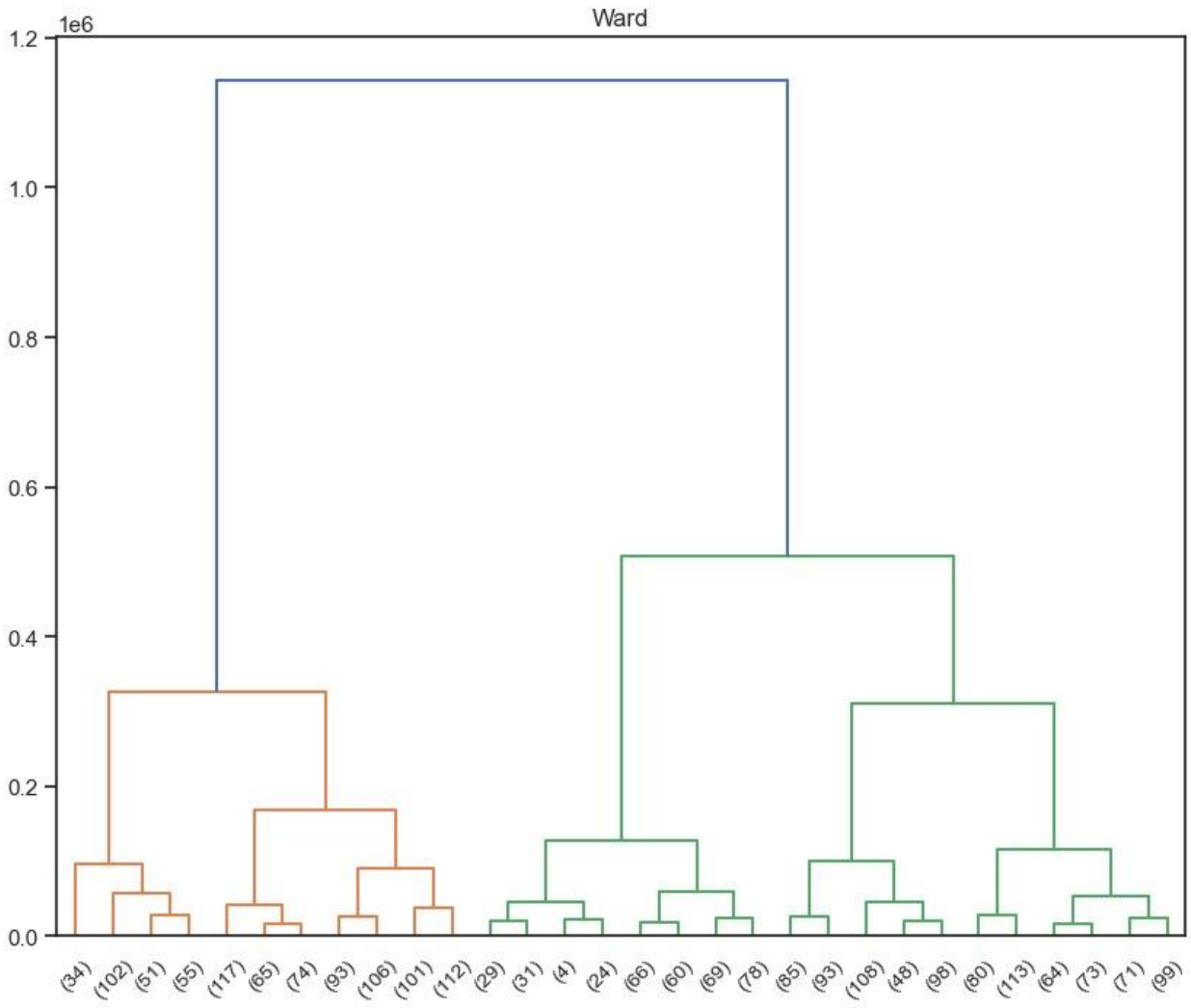
2. As the annual income increases, the total amount of money spent on products increases.

3. As the number of kids increases, the annual income decreases.

Clustering

A) Hierarchical Clustering

```
linkage_matrix = hierarchy.linkage(numerical_df, method='ward')
plt.figure(figsize=(10,8))
plt.title('Ward')
dendro = hierarchy.dendrogram(linkage_matrix, truncate_mode='lastp', p=20)
```



From this graph, we can infer that the number of clusters in this data is 3, which shortly after, we'll use in as K when performing Kmedoids clustering algorithm.

Adding a new index to the numerical data

```
numerical_columns['ID'] = df['ID']
numerical_columns.set_index('ID', inplace=True)
```

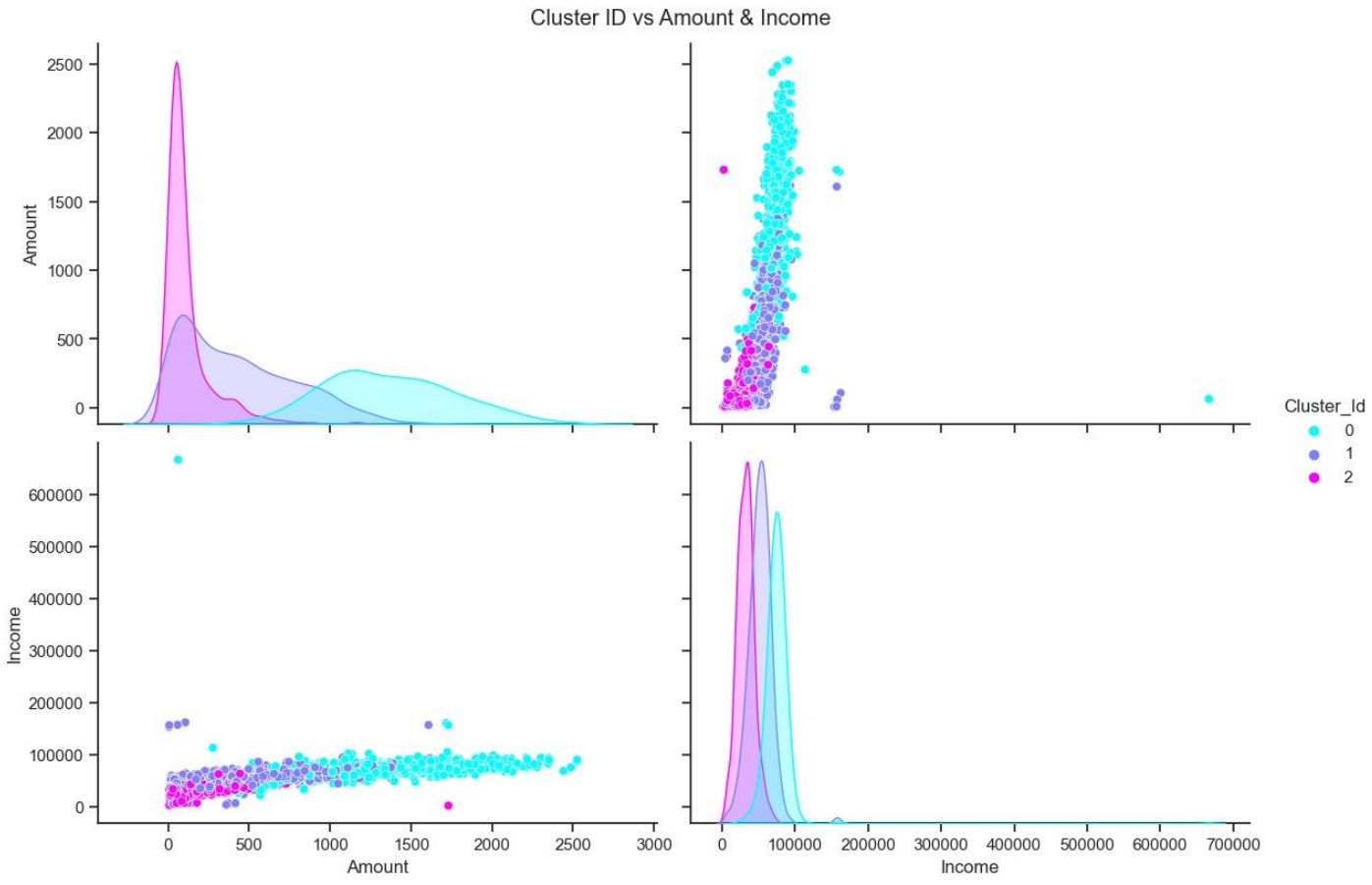
B) Kmedoids Clustering

```
scaler = StandardScaler()
scaled_data = scaler.fit_transform(numerical_columns)
k = 3
kmedoids = KMedoids(n_clusters=k, random_state=0)
kmedoids.fit(scaled_data)
numerical_columns['Cluster_Id'] = kmedoids.labels_
clusters = numerical_columns['Cluster_Id'].value_counts()
clusters
```

```
Cluster_Id
1    846
2    708
0    662
Name: count, dtype: int64
```

- Clusters characteristics through visualization

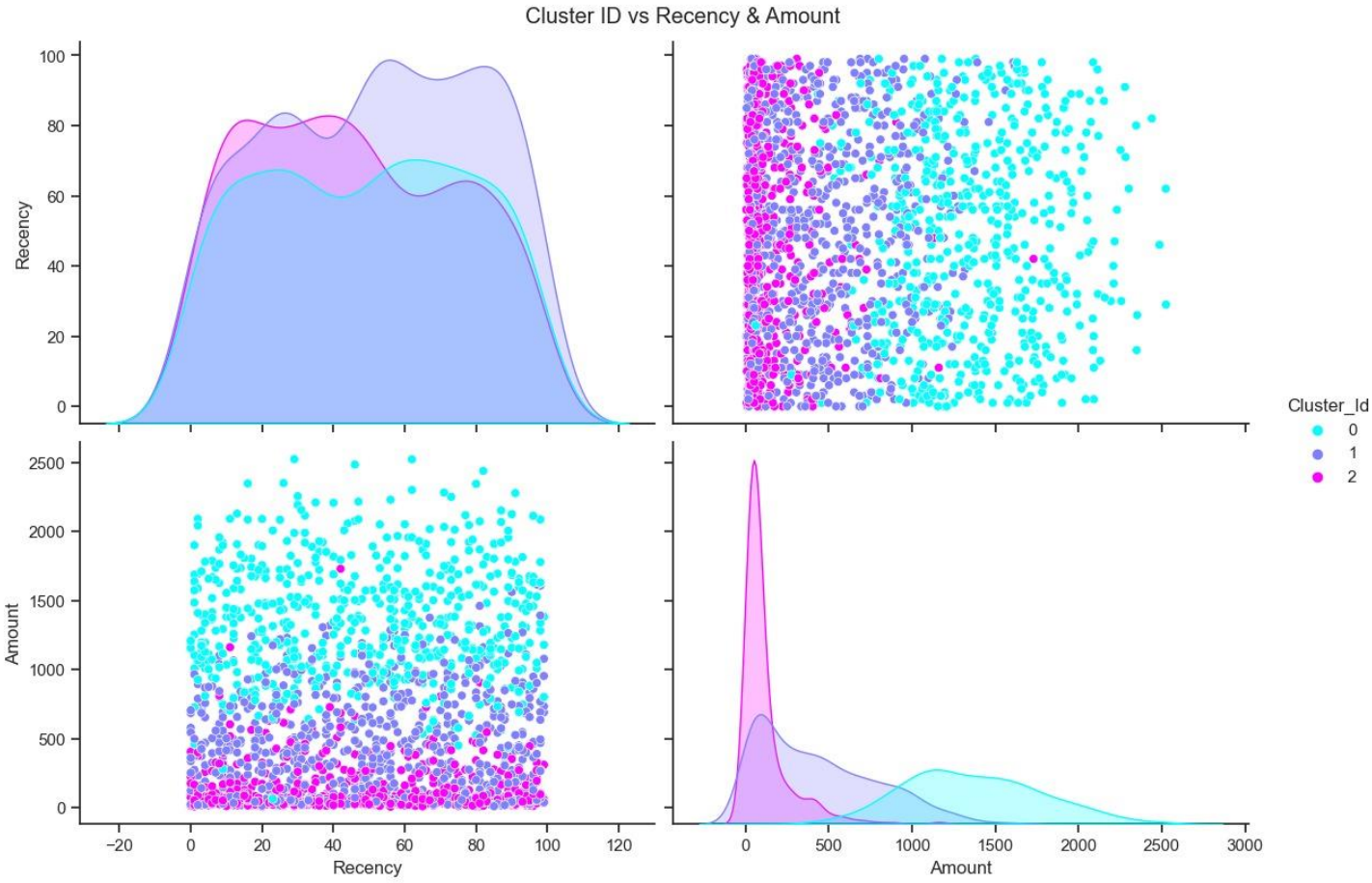
```
sns.set(style="ticks")
sns.pairplot(numerical_columns, hue='Cluster_Id', vars=['Amount', 'Income'], palette='cool', height=4, aspect=1.5)
plt.suptitle('Cluster ID vs Amount & Income', y=1.02)
plt.show()
```



- We can infer from this graph that cluster 0 have the highest income and spend the most on the products.
- Cluster 1 are middle income and don't spend as much as cluster 0 on the products.

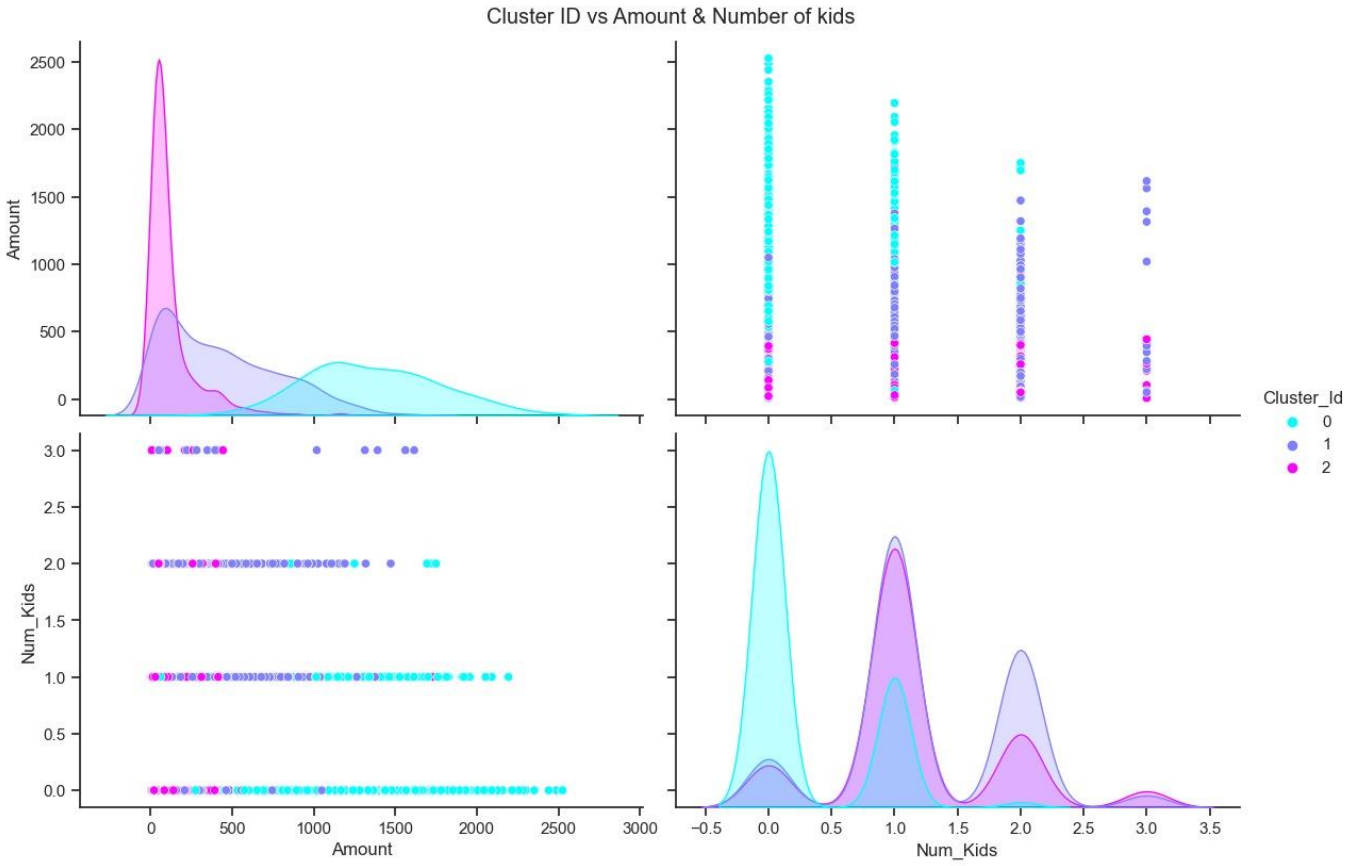
- Cluster 2 have the lowest income, and they are the lowest spenders.

```
sns.set(style="ticks")
sns.pairplot(numerical_columns, hue='Cluster_Id', vars=['Recency', 'Amount'], palette='cool', height=4, aspect=1.5)
plt.suptitle('Cluster ID vs Recency & Amount', y=1.02)
plt.show()
```



- Cluster 2 are the most recent buyers followed by cluster 1 then cluster 0.

```
sns.set(style="ticks")
sns.pairplot(numerical_columns, hue='Cluster_Id', vars=['Amount', 'Num_Kids'], palette='cool', height=4, aspect=1.5)
plt.suptitle('Cluster ID vs Amount & Number of kids', y=1.02)
plt.show()
```



- The high spenders (cluster 0) tend to always have no children.
- The middle spenders (cluster 1) tend to have 1 or 2 children.
- The low spenders (cluster 2) tend to have at least 1 child, so they have the most children out of the 3 clusters.

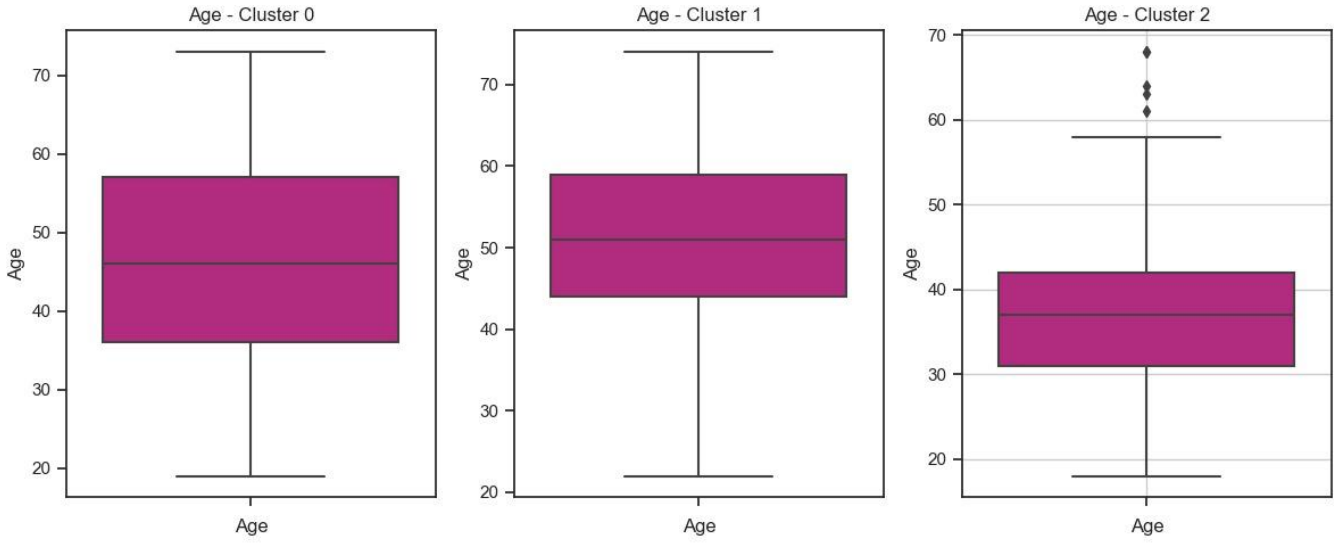
- Creating a data frame that has all the attributes for each cluster

```
cols = ['ID','Income', 'Kidhome', 'Teenhome', 'Recency', 'MntWines','MntFruits','MntMeatProducts',\
        'MntFishProducts','MntSweetProducts','MntGoldProds','Age', 'Num_Kids', 'Amount', 'Complain']
df = pd.merge(numerical_columns, df, on=cols , how='inner')
```

```
cluster0_df = df[df['Cluster_Id'] == 0]
cluster1_df = df[df['Cluster_Id'] == 1]
cluster2_df = df[df['Cluster_Id'] == 2]
```

- The distribution of age of each cluster

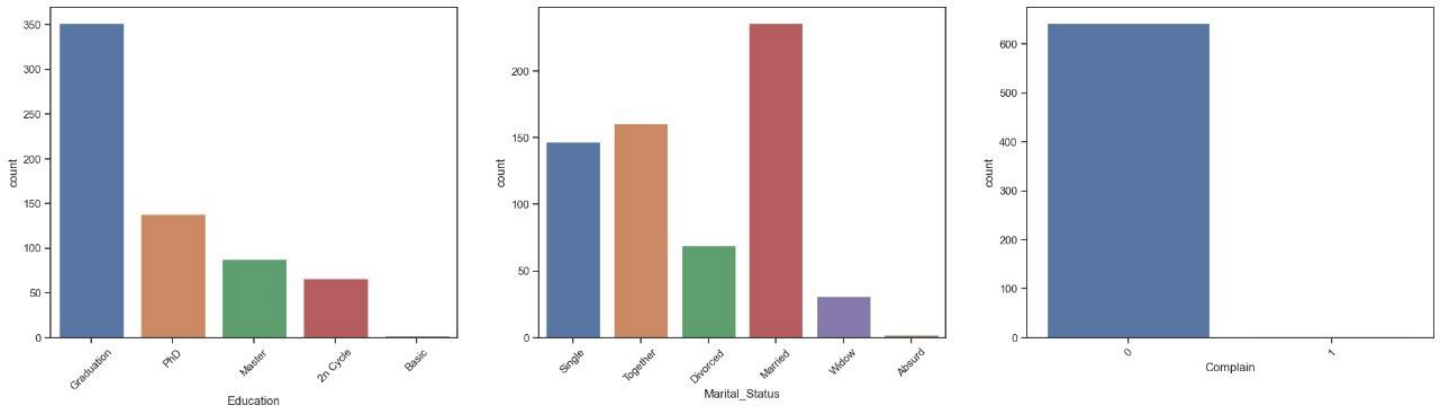
```
fig, axs = plt.subplots(1, 3, figsize=(12, 5))

sns.boxplot(data = cluster0_df, y='Age', color='mediumvioletred', ax=axs[0])
axs[0].set_xlabel("Age")
axs[0].set_title("Age - Cluster 0")

sns.boxplot(data = cluster1_df, y='Age', color='mediumvioletred', ax=axs[1])
axs[1].set_xlabel("Age")
axs[1].set_title("Age - Cluster 1")

sns.boxplot(data = cluster2_df, y='Age', color='mediumvioletred', ax=axs[2])
axs[2].set_xlabel("Age")
axs[2].set_title("Age - Cluster 2")

plt.grid(True)
plt.tight_layout()
```



- 50% of customers in cluster 0 are between 37 and 68 years old.
- 50% of customers in cluster 1 are between 43 and 69 years old.
- 50% of customers in cluster 2 are between 30 and 41 years old.

- To draw conclusions about cluster 0 we used bar plots to visualize categorical variables.

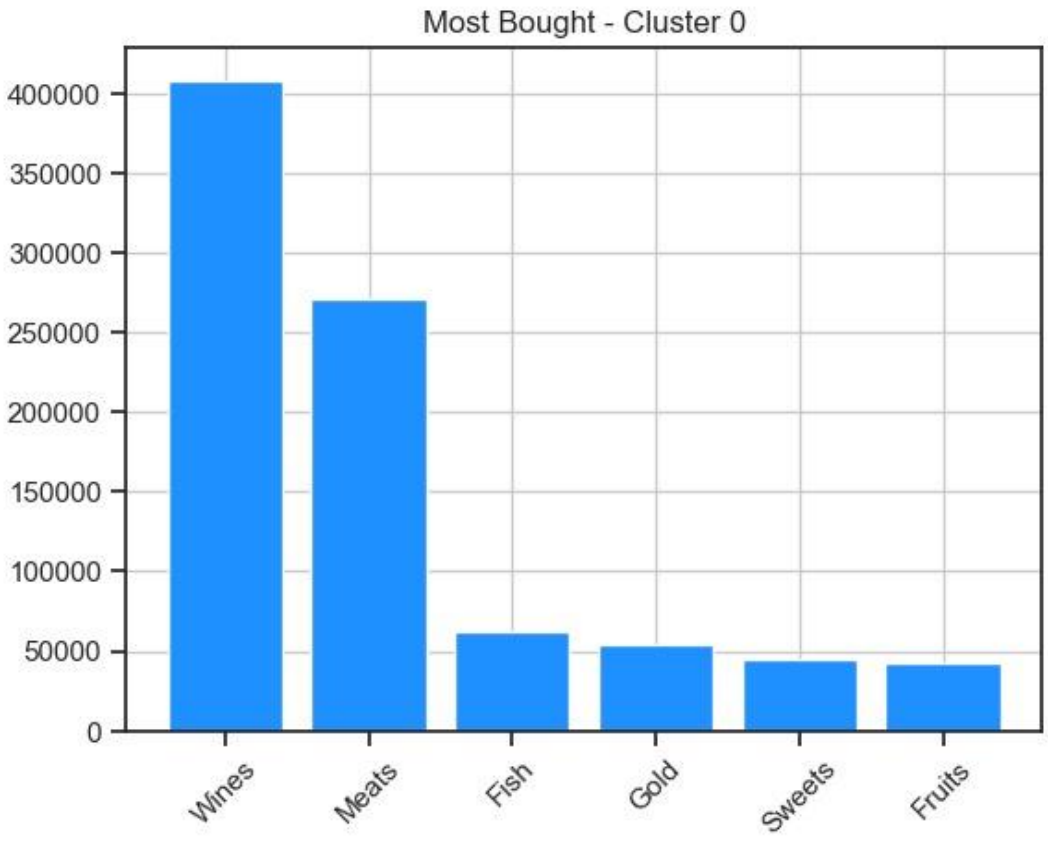
```
cal_cols = ['Education', 'Marital_Status','Complain']
fig, ax = plt.subplots(1, len(cal_cols), figsize=(25, 6))
for col in enumerate(cal_cols):
    sns.countplot(data=cluster0_df, x=col[1], ax=ax[col[0]])
    ax[col[0]].tick_params(axis='x', rotation=45)
```



- Most of cluster 0 are married, graduated with a college degree and have not complained in the last 3 years.

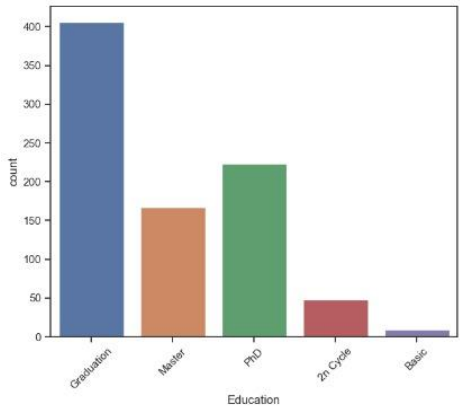
- We also used bar plots to get a sense of the products most bought by cluster 0

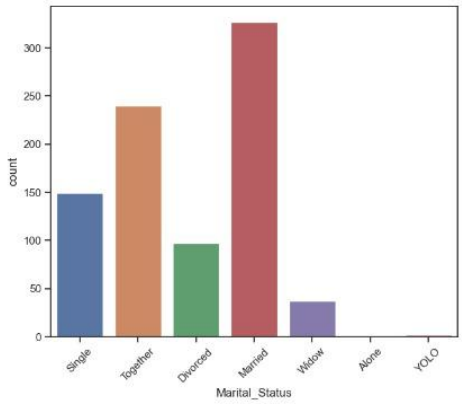
```
features=['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds']
col_sum = cluster0_df[features].sum(axis=0).sort_values(ascending=False)

plt.bar(col_sum.index, col_sum.values, color='dodgerblue')
plt.title("Most Bought - Cluster 0")
plt.xticks(col_sum.index,['Wines', 'Meats', 'Fish', 'Gold', 'Sweets', 'Fruits'],rotation=45)

plt.grid(True)
plt.show()
```

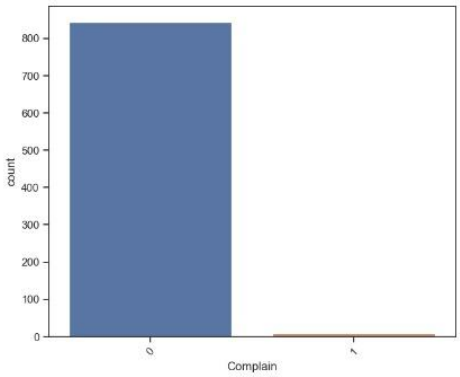


- We performed the same analysis for the other 2 clusters as well

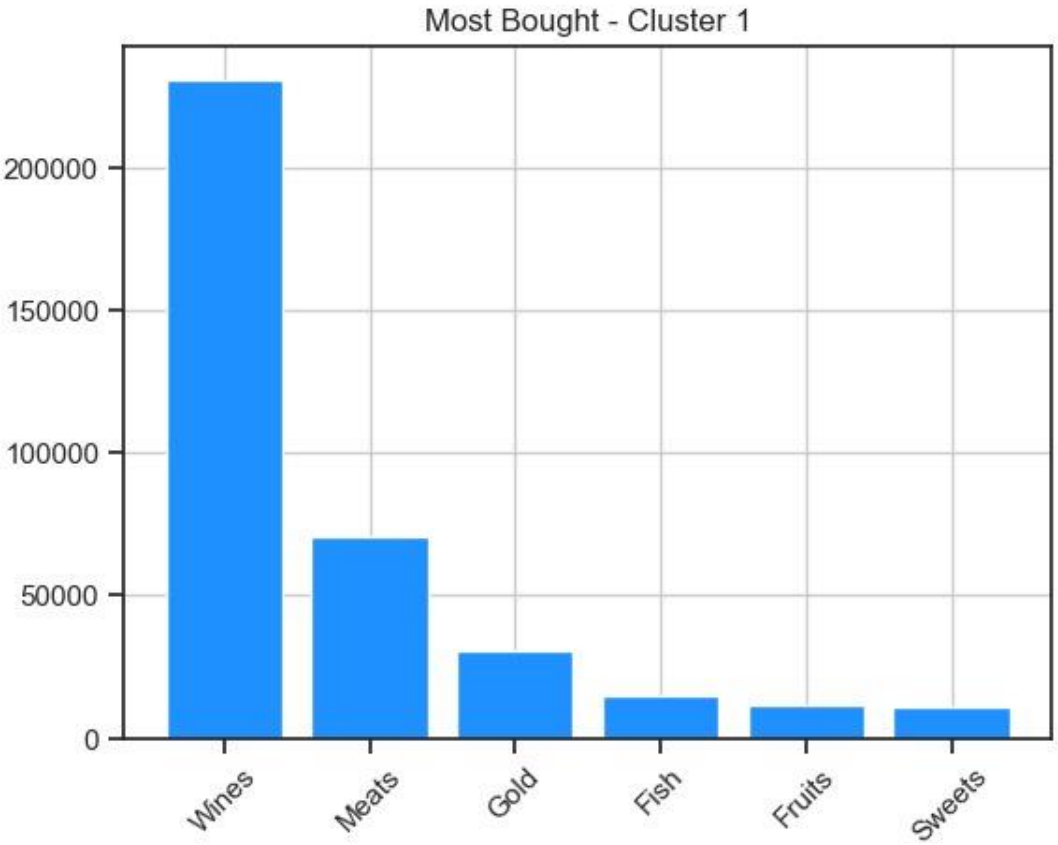
Cluster 1





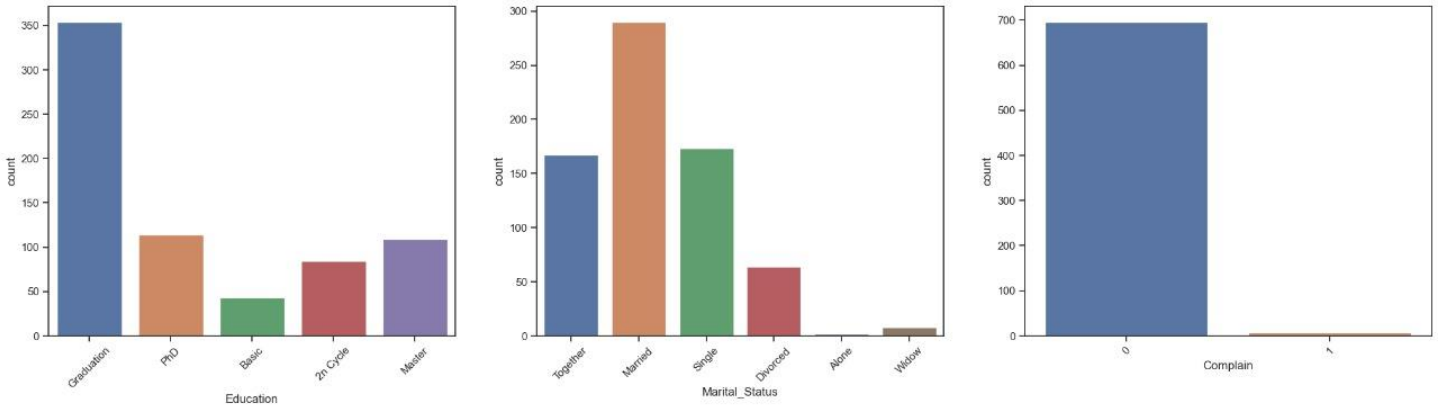


- Most of them are also college graduates but they have a higher tendency to have master or PhD degree among them.
- The highest percentage of them are married AND the second highest percentage are in a relationship.
- A very small percentage of them have complained in the last 3 years.

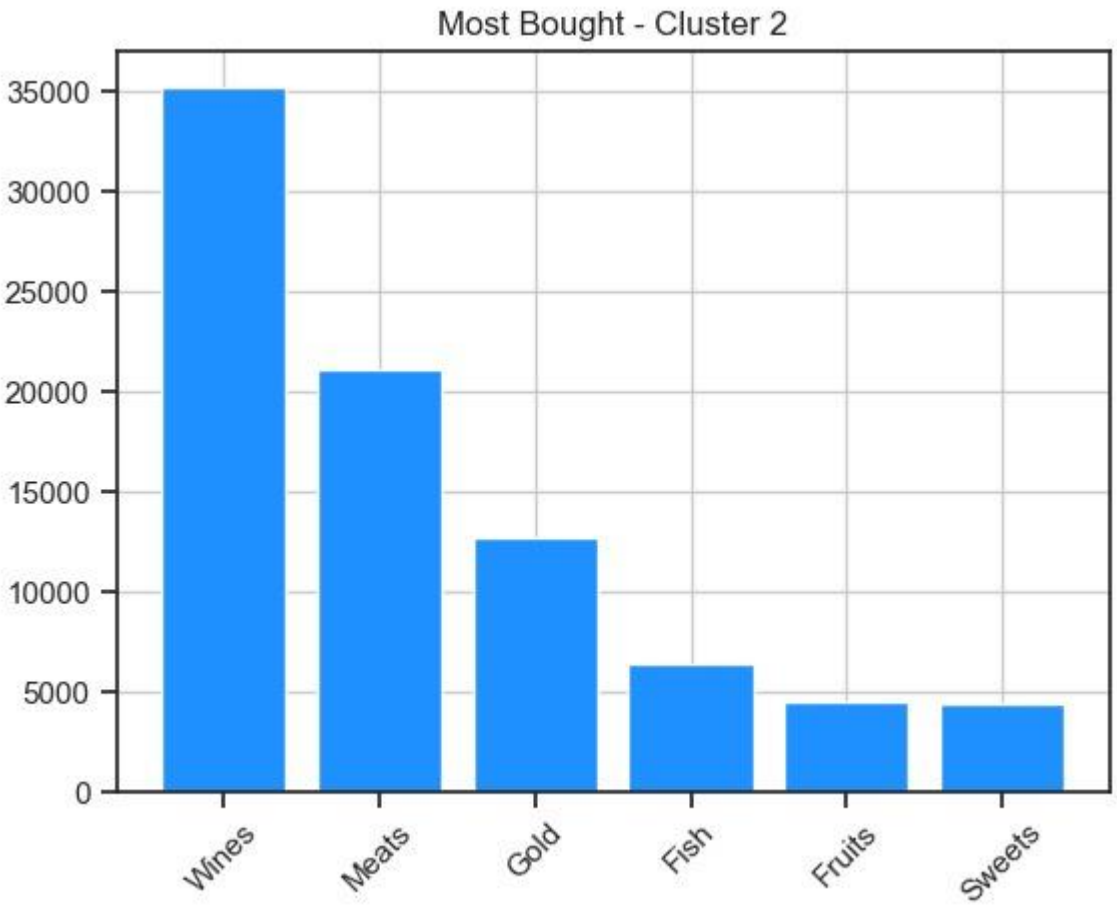


- Like the pervious cluster they also buy a lot of wine, but they buy way less meat.

Cluster 2



- Like the pervious cluster, they're college graduates and some of them have PhD and/or master's degree. But unlike the previous clusters, they have a higher percentage of single people among them.



- This cluster of customers buy more fish and way more gold than the other previous clusters.

Conclusion

Cluster 0:

Highest income, big spenders.

Middle aged with no kids.

Maximum wine buyers.

Cluster 1:

Middle income, middle spending habits.

Also, middle aged but tend to have 1 or 2 kids.

Cluster 2:

Low income, low spenders.

Relatively young, with at least one kid.

A relatively high amount of their money is spent on gold products.