```
import numpy as np
import pandas as pd
import datetime
from datetime import date
import datetime as dt
import matplotlib
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.graph_objects as go
from sklearn.preprocessing import StandardScaler, normalize
from sklearn import metrics
from sklearn.mixture import GaussianMixture
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import warnings
warnings.filterwarnings('ignore')
import scipy.cluster.hierarchy as shc
from sklearn.cluster import AgglomerativeClustering
```

```
data=pd.read_csv("/content/customer_data (1).csv")
```

```
data.head()
```

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

5 rows × 29 columns

```
data.describe()
```

	ID	Year_Birth	Income	Kidhome	Teenhome	Recency
count	2240.000000	2240.000000	2216.000000	2240.000000	2240.000000	2240.000000
mean	5592.159821	1968.805804	52247.251354	0.444196	0.506250	49.109375
std	3246.662198	11.984069	25173.076661	0.538398	0.544538	28.962453
min	0.000000	1893.000000	1730.000000	0.000000	0.000000	0.000000
25%	2828.250000	1959.000000	35303.000000	0.000000	0.000000	24.000000
50%	5458.500000	1970.000000	51381.500000	0.000000	0.000000	49.000000
75%	8427.750000	1977.000000	68522.000000	1.000000	1.000000	74.000000
max	11191.000000	1996.000000	666666.000000	2.000000	2.000000	99.000000

8 rows × 26 columns

```
#Now I will create some new features in the dataset to define the customer personalities as a part of data preparation:
```

```
data.isnull().sum()
```

```
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
NumStorePurchases       0
NumWebVisitsMonth       0
AcceptedCmp3            0
AcceptedCmp4            0
AcceptedCmp5            0
AcceptedCmp1            0
AcceptedCmp2            0
Complain                0
Z_CostContact           0
Z_Revenue               0
Response                0
dtype: int64
```

```
df=data.fillna(data["Income"].mean())
```

```
#drop un unusual columns
df=df.drop("Z_CostContact",axis=1)
```

```
df=df.drop("Z_Revenue",axis=1)
```

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

```
Married     864
Together    580
Single      480
Divorced    232
Widow        77
Alone         3
Absurd        2
YOLO          2
Name: Marital_Status, dtype: int64
```

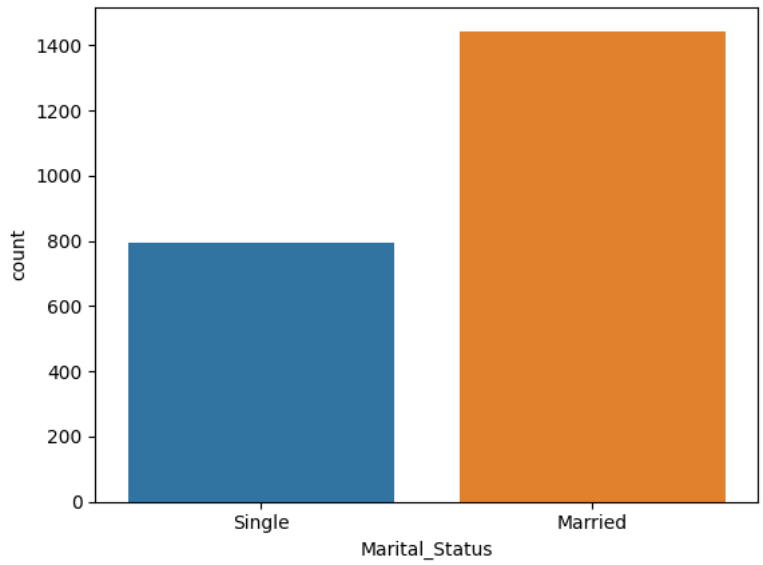
```
df.Marital_Status.replace({"Alone":"Single","Widow":"Single","Absurd":"Single","YOLO":"Single","Divorced":"Single","Together":"Married"},
```

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

```
Married    1444
Single      796
Name: Marital_Status, dtype: int64
```

```
sns.countplot(x="Marital_Status",data=df)
```

```
<Axes: xlabel='Marital_Status', ylabel='count'>
```




```
df["TotalChildren"]=df["Kidhome"]+df["Teenhome"]
```

```
df2=df.drop(columns=["Kidhome","Teenhome"])
df=df2
df.head()
```

df.head()

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	...	NumStorePur
0	5524	1957	Graduation	Single	58138.0	04/09/12	58	635	88	546	...	
1	2174	1954	Graduation	Single	46344.0	08/03/14	38	11	1	6	...	
2	4141	1965	Graduation	Married	71613.0	21/08/13	26	426	49	127	...	
3	6182	1984	Graduation	Married	26646.0	10/02/14	26	11	4	20	...	
4	5324	1981	PhD	Married	58293.0	19/01/14	94	173	43	118	...	

5 rows × 26 columns

```
df.Marital_Status.replace({"Single":0,"Married":1},inplace=True)
```

```
df.head()
```

	ID	Year_Birth	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	...	NumStorePur
0	5524	1957	Graduation	0	58138.0	04/09/12	58	635	88	546	...	
1	2174	1954	Graduation	0	46344.0	08/03/14	38	11	1	6	...	
2	4141	1965	Graduation	1	71613.0	21/08/13	26	426	49	127	...	
3	6182	1984	Graduation	1	26646.0	10/02/14	26	11	4	20	...	
4	5324	1981	PhD	1	58293.0	19/01/14	94	173	43	118	...	

5 rows × 26 columns

```
df['Age']=2023-df['Year_Birth']
```

```
df2=df.drop(columns=["Year_Birth"])
```

```
df2.Education.replace({'Basic':'Undergraduate','2n Cycle':'Undergraduate','Graduation':'Postgraduate','Master':'Postgraduate','PhD':'Post
```

```
df2.Education.replace({"Undergraduate":0,"Postgraduate":1},inplace=True)
```

```
df2['Education'].value_counts()
```

```
1    1983
0     257
Name: Education, dtype: int64
```

```
sns.boxplot(df2["Income"])
plt.show()
```

```
data=df2
```

```
data=data.dropna(subset=['Income'])
data=data[data['Income']<600000]
```

```
data.head()
```

	ID	Education	Marital_Status	Income	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	...	NumWeb
0	5524	1	0	58138.0	04/09/12	58	635	88	546	172	...	
1	2174	1	0	46344.0	08/03/14	38	11	1	6	2	...	
2	4141	1	1	71613.0	21/08/13	26	426	49	127	111	...	
3	6182	1	1	26646.0	10/02/14	26	11	4	20	10	...	
4	5324	1	1	58293.0	19/01/14	94	173	43	118	46	...	

5 rows × 26 columns

```
data.isnull().sum()
```

```
ID                   0
Education            0
Marital_Status       0
Income               0
Dt_Customer          0
Recency              0
MntWines             0
MntFruits            0
MntMeatProducts      0
MntFishProducts      0
MntSweetProducts     0
MntGoldProds         0
NumDealsPurchases    0
NumWebPurchases      0
NumCatalogPurchases  0
NumStorePurchases    0
NumWebVisitsMonth    0
AcceptedCmp3         0
AcceptedCmp4         0
AcceptedCmp5         0
AcceptedCmp1         0
AcceptedCmp2         0
Complain             0
Response             0
TotalChildren        0
Age                  0
dtype: int64
```

```
sns.boxplot(data["Income"])
```


```
data['Spending']=data['MntWines']+data['MntFruits']+data['MntMeatProducts']+data['MntFishProducts']+data['MntSweetProducts']+data['MntGol
```

```
data=data.rename(columns={'MntWines': "Wines",'MntFruits':'Fruits','MntMeatProducts':'Meat','MntFishProducts':'Fish','MntSweetProducts':'
data.head()
```

	ID	Education	Marital_Status	Income	Dt_Customer	Recency	Wines	Fruits	Meat	Fish	...	AcceptedCmp3	AcceptedCmp4	Accept
0	5524	1	0	58138.0	04/09/12	58	635	88	546	172	...	0	0	
1	2174	1	0	46344.0	08/03/14	38	11	1	6	2	...	0	0	
2	4141	1	1	71613.0	21/08/13	26	426	49	127	111	...	0	0	
3	6182	1	1	26646.0	10/02/14	26	11	4	20	10	...	0	0	
4	5324	1	1	58293.0	19/01/14	94	173	43	118	46	...	0	0	

5 rows × 27 columns

```
##Seniority variable creation
last_date = date(2023,4,4)
```

```
data['Seniority']=pd.to_datetime(data['Dt_Customer'])
```

```
data['Seniority']=pd.to_datetime(data['Seniority'], dayfirst=True,format = '%Y-%m-%d')
```

```
data['Seniority'] = pd.to_numeric(data['Seniority'].dt.date.apply(lambda x: (last_date - x)).dt.days, downcast='integer')/30
```

```
data['Seniority']
```

```
0       133.733333
1       105.533333
2       117.100000
3       103.533333
4       112.066667
           ...
2235    119.400000
2236    103.400000
2237    111.866667
2238    111.900000
2239    127.433333
Name: Seniority, Length: 2239, dtype: float64
```

```
data.head()
```

	ID	Education	Marital_Status	Income	Dt_Customer	Recency	Wines	Fruits	Meat	Fish	...	AcceptedCmp4	AcceptedCmp5	Accept
0	5524	1	0	58138.0	04/09/12	58	635	88	546	172	...	0	0	
1	2174	1	0	46344.0	08/03/14	38	11	1	6	2	...	0	0	
2	4141	1	1	71613.0	21/08/13	26	426	49	127	111	...	0	0	
3	6182	1	1	26646.0	10/02/14	26	11	4	20	10	...	0	0	
4	5324	1	1	58293.0	19/01/14	94	173	43	118	46	...	0	0	

5 rows × 28 columns

```
df2=data.drop(columns=["ID","Dt_Customer"])
data=df2
data.head()
```

	Education	Marital Status	Income	Recency	Wines	Fruits	Meat	Fish	Sweets	Gold	...	AcceptedCmp4	AcceptedCmp5	AcceptedCmp
2	1	1	71613.0	26	426	49	127	111	21	42	...	0	0	

```
scaler=StandardScaler()
dataset_temp=data[['Income','Seniority','Spending']]
X_std=scaler.fit_transform(dataset_temp)
X = normalize(X_std,norm='l2')
```

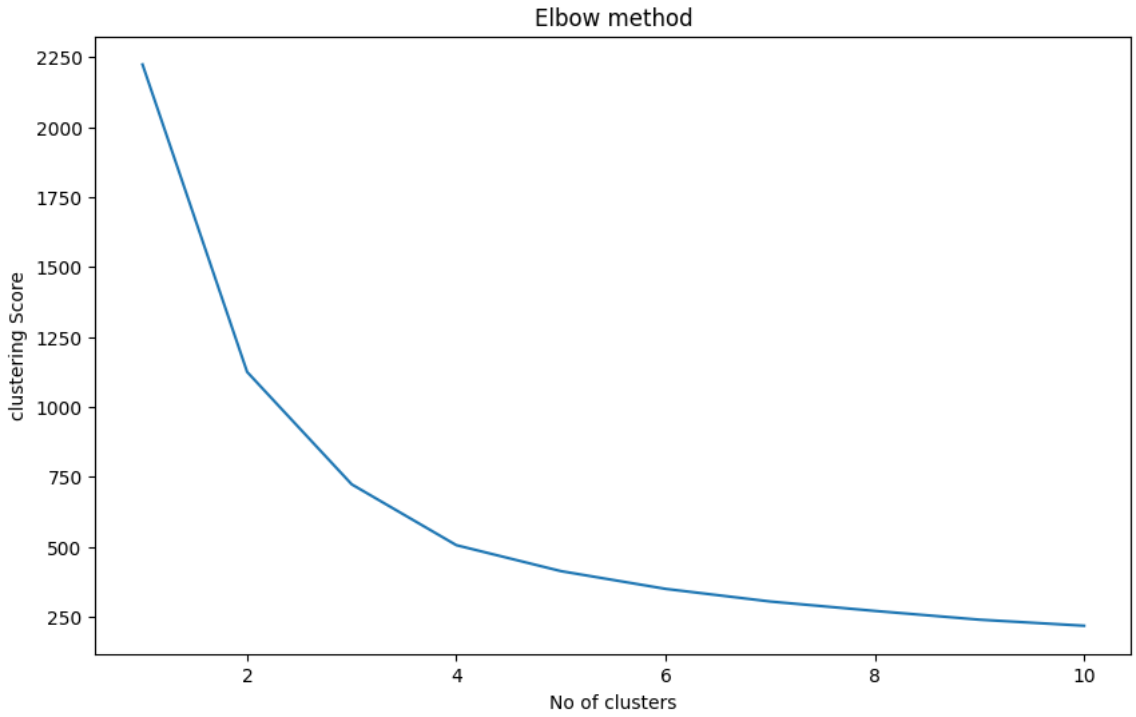
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans


from sklearn.metrics import silhouette_score
silhouette_score_lst=[]
for i in range(2,11):
  silhouette_score_lst.append(silhouette_score(X,(KMeans(n_clusters=i).fit_predict(X))))



#elbow method
clustering_score=[]
for i in range(1,11):
  kmeans=KMeans(n_clusters=i,init='random',random_state=23)
  kmeans.fit(X)
  clustering_score.append(kmeans.inertia_)


plt.figure(figsize=(10,6))
plt.plot(range(1,11),clustering_score)
plt.xlabel('No of clusters')
plt.ylabel('clustering Score')
plt.title("Elbow method")
plt.show()
```




```
## set up a model
kmeans=KMeans(n_clusters=3,random_state=23)
#fit model
kmeans.fit(X)
#predict
pred=kmeans.predict(X)


len(pred)
```

```
2239
```

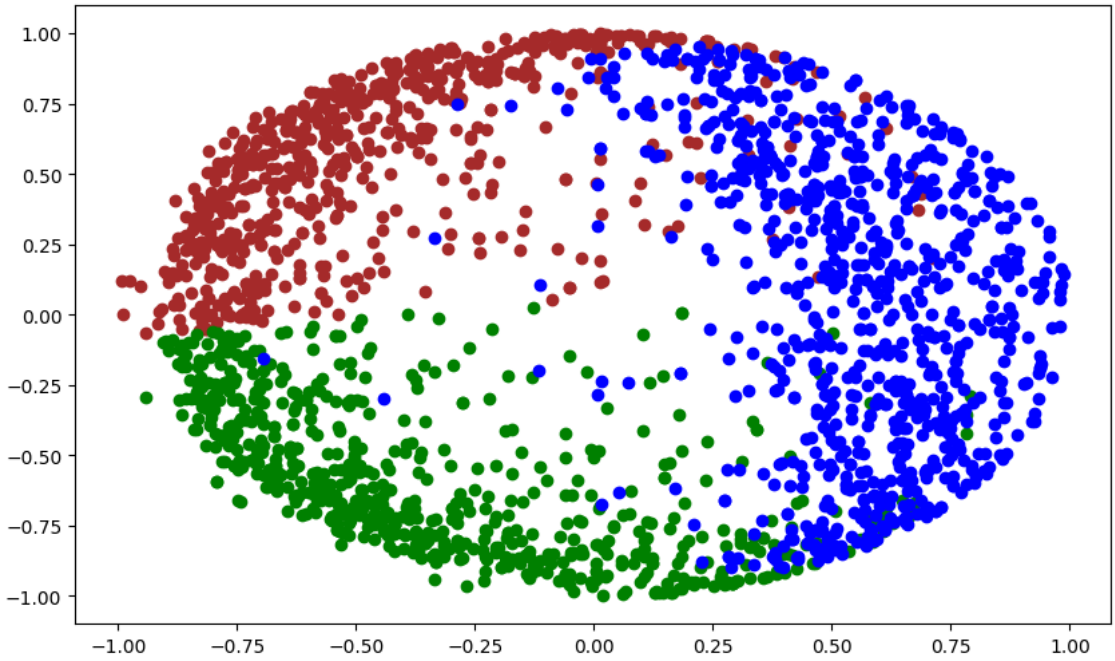
```
data['cluster'] = pd.DataFrame(pred,columns = ['cluster'])
data.head(10)
```

	Education	Marital_Status	Income	Recency	Wines	Fruits	Meat	Fish	Sweets	Gold	...	AcceptedCmp5	AcceptedCmp1	AcceptedCmp
0	1	0	58138.0	58	635	88	546	172	88	88	...	0	0	
1	1	0	46344.0	38	11	1	6	2	1	6	...	0	0	
2	1	1	71613.0	26	426	49	127	111	21	42	...	0	0	
3	1	1	26646.0	26	11	4	20	10	3	5	...	0	0	
4	1	1	58293.0	94	173	43	118	46	27	15	...	0	0	
5	1	1	62513.0	16	520	42	98	0	42	14	...	0	0	
6	1	0	55635.0	34	235	65	164	50	49	27	...	0	0	
7	1	1	33454.0	32	76	10	56	3	1	23	...	0	0	
8	1	1	30351.0	19	14	0	24	3	3	2	...	0	0	
9	1	1	5648.0	68	28	0	6	1	1	13	...	0	0	

10 rows × 27 columns

```
#Now let's plot this data to have a look at the clustering of customers:
plt.figure(figsize = (10,6))

plt.scatter(X[pred==0.0,0],X[pred==0.0,1],c = 'brown',label = 'cluster 0')
plt.scatter(X[pred==1.0,0],X[pred==1.0,1],c = 'green',label = 'cluster 1')
plt.scatter(X[pred==2.0,0],X[pred==2.0,1],c = 'blue',label = 'cluster 2')
plt.show()
```




```
#Now I will prepare the data for the Apriori algorithm. Here I will be defining three segments of the customers according to the age, inc


#Create Age segment
cut_labels_Age = ['Young', 'Adult', 'Mature', 'Senior']
cut_bins = [0, 30, 45, 65, 120]
data['Age_group'] = pd.cut(data['Age'], bins=cut_bins, labels=cut_labels_Age)
#Create Income segment
cut_labels_Income = ['Low income', 'Low to medium income', 'Medium to high income', 'High income']
data['Income_group'] = pd.qcut(data['Income'], q=4, labels=cut_labels_Income)
#Create Seniority segment
cut_labels_Seniority = ['New customers', 'Discovering customers', 'Experienced customers', 'Old customers']
data['Seniority_group'] = pd.qcut(data['Seniority'], q=4, labels=cut_labels_Seniority)
data=data.drop(columns=['Age','Income','Seniority'])
data.head()
```

	Education	Marital_Status	Recency	Wines	Fruits	Meat	Fish	Sweets	Gold	NumDealsPurchases	...	AcceptedCmp1	AcceptedCmp2	C
0	1	0	58	635	88	546	172	88	88	3	...	0	0	
1	1	0	38	11	1	6	2	1	6	2	...	0	0	
2	1	1	26	426	49	127	111	21	42	1	...	0	0	
3	1	1	26	11	4	20	10	3	5	2	...	0	0	
4	1	1	94	173	43	118	46	27	15	5	...	0	0	

```
#Now I will define new segments according to the spending of customers on each product which will be based on:
cut_labels = ['Low consumer', 'Frequent consumer', 'Biggest consumer']
data['Wines_segment'] = pd.qcut(data['Wines'][data['Wines']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data['Fruits_segment'] = pd.qcut(data['Fruits'][data['Fruits']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data['Meat_segment'] = pd.qcut(data['Meat'][data['Meat']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data['Fish_segment'] = pd.qcut(data['Fish'][data['Fish']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data['Sweets_segment'] = pd.qcut(data['Sweets'][data['Sweets']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data['Gold_segment'] = pd.qcut(data['Gold'][data['Gold']>0],q=[0, .25, .75, 1], labels=cut_labels).astype("object")
data.replace(np.nan, "Non consumer",inplace=True)
data.drop(columns=['Spending','Wines','Fruits','Meat','Fish','Sweets','Gold'],inplace=True)
data = data.astype(object)
```

```
data.head()
```

ases	NumCatalogPurchases	NumStorePurchases	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	...	cluster	Age_group	Income_group	S
8	10	4	7	0	0	...	2.0	Senior	Medium to high income	
1	1	2	5	0	0	...	1.0	Senior	Low to medium income	
8	2	10	4	0	0	...	2.0	Mature	High income	
2	0	4	6	0	0	...	1.0	Adult	Low income	
5	3	6	5	0	0	...	1.0	Adult	Medium to high income	

Seniority_group Income_group	Discovering customers	Experienced customers	New customers	Old customers
High income	138	138	154	130
Low income	131	142	140	147
Low to medium income	146	140	140	134
Medium to high income	150	134	126	149

```
pd.crosstab(data["Income_group"],data["Wines_segment"])
```

Wines_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	337	218	5	0
Low income	0	138	409	13
Low to medium income	19	411	130	0
Medium to high income	200	345	14	0

```
pd.crosstab(data["Income_group"],data["Fish_segment"])
```

Fish_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	310	213	9	28
Low income	7	216	228	109
Low to medium income	29	192	188	151
Medium to high income	116	285	62	96

```
pd.crosstab(data["Income_group"],data["Meat_segment"])
```

Meat_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	425	130	5	0
Low income	2	191	366	1
Low to medium income	7	372	181	0
Medium to high income	124	396	39	0

```
pd.crosstab(data["Income_group"],data["Fruits_segment"])
```

Fruits_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	290	235	8	27
Low income	7	180	258	115
Low to medium income	21	208	180	151
Medium to high income	135	261	56	107

```
pd.crosstab(data["Income_group"],data["Sweets_segment"])
```

Sweets_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	290	227	7	36
Low income	5	202	240	113
Low to medium income	23	204	175	158

```
pd.crosstab(data["Income_group"],data["Gold_segment"])
```

Gold_segment Income_group	Biggest consumer	Frequent consumer	Low consumer	Non consumer
High income	247	277	21	15
Low income	18	243	285	14
Low to medium income	75	267	201	17

```
#will use this algorithm to identify the biggest customer of wines
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
pd.set_option('display.max_colwidth', 999)
pd.options.display.float_format = "{:.3f}".format
association=data.copy()
df = pd.get_dummies(association)
min_support = 0.08
max_len = 10
frequent_items = apriori(df, use_colnames=True, min_support=min_support, max_len=max_len + 1)
rules = association_rules(frequent_items, metric='lift', min_threshold=1)

product='Wines'
segment='Biggest consumer'
target = '{\'%s_segment_%s\'}' %(product,segment)
results_personnal_care = rules[rules['consequents'].astype(str).str.contains(target, na=False)].sort_values(by='confidence', ascending=Fa
results_personnal_care.head()
```

```
#So according to the output and overall analysis conducted on this data science project on customer personality analysis with Python, we
```

Colab paid products - Cancel contracts here

✓ 0s completed at 21:45

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.