Customer Segmentation: Clustering

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import date
from datetime import datetime
import warnings
warnings.filterwarnings("ignore")
```

In [ ]:
```
data=pd.read_csv('marketing_campaign.csv',sep='\t')
```

Data Cleaning and Adding features

In [ ]:
```
data.head()
```

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

remove null values

In [ ]:
```
data=data.dropna()
```

converting the type of Dt_Customer column

In [ ]:
```
data['Dt_Customer']=pd.to_datetime(data['Dt_Customer'],format='mixed')
```

In [ ]:
```
dates = []
for i in data["Dt_Customer"]:
    i = i.date()
    dates.append(i)
```

adding column - cutomer for

to show how many days a customer has been in the databes

need to calculate oldest and newest record

In [ ]:
```
days = []
d1 = max(dates)
for i in dates:
    delta = d1 - i
    days.append(delta)
data["Customer_For"] = days
data["Customer_For"] = pd.to_numeric(data["Customer_For"], errors="coerce")
```

checking categories in the marriage and education section

In [ ]:
```
data['Marital_Status'].value_counts()
```

In [ ]:
```
data['Education'].value_counts()
```

adding new features

getting age of cutomer from year_birth

we consider the last date of record entry as reference

In [ ]:
```
max(data['Dt_Customer'])
```

In [ ]:
```
data['Age']=2014-data['Year_Birth']
```

Total Spending

In [ ]:
```
data['Spent']=data.iloc[:,9:15].sum(axis=1)
```

Deriving Living situation

In [ ]:
```
data["Living_With"]=data["Marital_Status"].replace({"Married":"Partner", "Toge
```

Total children in household

In [ ]:
```
data["Children"]=data["Kidhome"]+data["Teenhome"]
```

total members in the household

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

checking if parent

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

defining education level

In [ ]:
```
data["Education"]=data["Education"].replace({"Basic":"Undergraduate","2n Cycle
```

removing columns we dont need

In [ ]:
```
to_drop = ["Marital_Status", "Dt_Customer", "Z_CostContact", "Z_Revenue", "Yea
data = data.drop(to_drop, axis=1)
```

In [ ]:
```
to_dropp=data.iloc[:,16:21]
data=data.drop(to_dropp,axis=1)
```

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

Removing outliers

In [ ]:
```
data = data[(data["Age"]<90)]
data = data[(data["Income"]<600000)]
```

plotting a pairplot to visulaise data

In [ ]:
```
To_Plot = [ "Income", "Recency", "Customer_For", "Age", "Spent", "Is_Parent"]
plt.figure()
sns.pairplot(data[To_Plot], hue= "Is_Parent",palette='viridis')
plt.show()
```

viewing correlation

In [ ]:
```
numeric_data = data.select_dtypes(include=[np.number])
corrmat =numeric_data.corr()
plt.figure(figsize=(15,15))
sns.heatmap(corrmat,annot=True, cmap='Set3', center=0)
```

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

preprocessing data

coverting categorical variables to numeric form

In [ ]:
```
s = (data.dtypes == 'object')
object_cols = list(s[s].index)
object_cols
```

Label Encoding the object dtypes.

In [ ]:
```
from sklearn.preprocessing import LabelEncoder
LE=LabelEncoder()
for i in object_cols:
    data[i]=data[[i]].apply(LE.fit_transform)
```

scaling

In [ ]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(data)
scaled_data = pd.DataFrame(scaler.transform(data),columns= data.columns )
```

In [ ]:
```
scaled_data.head()
```

Dimensionality reduction

- Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables.

Principal component analysis (PCA) is a dimensionality reduction and machine learning method used to simplify a large data set into a smaller set while still maintaining significant patterns and trends.

In [ ]:
```
from sklearn.decomposition import PCA
```

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

3d projection of data in reduced dimension

In [ ]:
```
x =PCA_data["col1"]
y =PCA_data["col2"]
z =PCA_data["col3"]

fig = plt.figure(figsize=(6,5))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x,y,z )
ax.set_title("3d Projection Of Data In The Reduced Dimension")
plt.show()
```

Clustering

Agglomerative clustering

- Hierarchial clustering method
- It starts with each data point as its own cluster and then repeatedly merges the closest pairs of clusters

finding number of clusters

elbow method

- used to determine number of clusters to be formed
- By plotting the inertia (spread of data) against the number of clusters and looking for the point where the inertia reduction slows down significantly(elbow shape), you can identify the most appropriate number of clusters for your dataset.

In [ ]:
```
!pip install yellowbrick -q
from yellowbrick.cluster import KElbowVisualizer
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from sklearn import metrics
```

In [ ]:
```
sns.set(palette='deep',rc={"figure.figsize": (6, 4)})
Elbow_M = KElbowVisualizer(KMeans(), k=10)
Elbow_M.fit(PCA_data)
Elbow_M.show()
```

4 is the optimal number of clusters

agglomerative clustering model

In [ ]:
```
AC = AgglomerativeClustering(n_clusters=4)
yhat_AC = AC.fit_predict(PCA_data)
PCA_data["Clusters"] = yhat_AC
data["Clusters"]= yhat_AC
```

In [ ]:
```
x =PCA_data["col1"]
y =PCA_data["col2"]
z =PCA_data["col3"]

fig=plt.figure(figsize=(8,6))
ax=plt.subplot(111,projection='3d')
ax.scatter(x,y,z,s=20,c=PCA_data['Clusters'],cmap='Set3')
ax.set_title('plot of clusters')
plt.show()
```

distribution of clusters

In [ ]:
```
pl=sns.countplot(x=data['Clusters'],palette='deep',alpha=0.8)
pl.set_title('distribution of clusters')
plt.show()
```

clusters income and spending

In [ ]:
```
pl=sns.scatterplot(x=data['Spent'],y=data['Income'],hue=data['Clusters'],palet
pl.set_title('clusters profile based on income and spending')
plt.legend()
plt.show()
```

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

amount spent by clusters

In [ ]:
```
sns.boxenplot(x=data['Clusters'],y=data['Spent'])
plt.title('amount spent by clusters')
```

cluster 2 is the biggest spender

Number of deals Purchased

In [ ]:
```
sns.boxenplot(x=data['Clusters'],y=data['NumDealsPurchases'])
plt.title('Number of deals purchased')
```

the average/low income groups (0,3) took the most amount of deals offered

spending on products by clusters

In [ ]:
```
data2=data.iloc[:,5:11]
data2['Clusters']=data['Clusters']
```

In [ ]:
```
# creating clustered bar chart
grouped = data2.groupby('Clusters').sum().reset_index()
melted = pd.melt(grouped, id_vars='Clusters', var_name='product', value_name='
plt.figure(figsize=(10, 6))
sns.barplot(x='Clusters', y='spending', hue='product', data=melted, palette='d
plt.title('Spending on Products by Clusters')
plt.xlabel('Clusters')
plt.ylabel('Amount Spent')
plt.legend(title='Product')
plt.show()
```

Profiling

- we have formed clustered and looked at spending habits.
- we need to identify who is in the clusters and identify target customer

plotting clusters on basis of personalfeatures

kids (age less than 13) at home

In [ ]:
```
sns.jointplot(x=data['Kidhome'],y=data['Spent'],hue=data['Clusters'],palette='
```

teenagers at home

In [ ]:
```
sns.jointplot(x=data['Teenhome'],y=data['Spent'],hue=data['Clusters'],palette=
```

age

In [ ]:
```
sns.boxplot(x=data['Clusters'],y=data['Age'],palette='deep')
```

total children

In [ ]:
```
sns.stripplot(x=data['Children'], y=data["Spent"], hue =data["Clusters"],palet
```

family size

In [ ]:
```
sns.stripplot(x=data['Family_Size'], y=data["Spent"], hue =data["Clusters"],pa
```

parent

In [ ]:
```
sns.stripplot(x=data['Is_Parent'], y=data["Spent"], hue =data["Clusters"],pale
```

education

In [ ]:
```
sns.countplot(x=data['Clusters'], hue =data["Education"].astype(str),palette='
```

cluster information

cluster 0:

- definitely a parent
- have 1 or 2 kids
- mostly have a teenager
- maximum 4 members and atleast 2
- average age between 40 to 60
- high spending and average income

cluster 1:

- majority of them are parents
- have mostly one kid
- no teenagers
- maximum 3 people in the family
- average age between 30 to 40
- low spending low income

cluster 2:

- definitely not a parent
- no kids
- no teenagers
- maximum 2 people in the family
- average age between 35 to 60 (spans all ages)
- high spending high income

cluster 3:

- definitely a parent
- have 1 to 3 kids
- majority have 1 teenager
- family is between 2 to 5 people
- average age is between 42 to 55
- high spending average income

Conclusion

- group 0 and 2 are the people with maximum spending
- cluster 2 is a high income no kids group which can be an ideal target group for increased quality products
- cluster 3 and 0 are families who will respond to discounts,deals and bulk sales
- cluster 1 is mostly a couple and one kid

In [ ]:

```
```