Malnad College of Engineering

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

Hassan – 573202

Customer Data Using K-Means Clustering

Submitted by

Lavanya H R(4MC19IS025) **Kruthika V (4MC19IS023)**

Nikitha C S(4MC18IS024) **Ranju P S R(4MC19IS042)**

under the guidance of

Name of the Guide
Designation
D.R Balaji Prabu B V

Department of Information Science & Engineering
Malnad College of Engineering
Hassan - 573 202

Tel.: 08172-245093 Fax: 08172-245683 URL: www.mcehassan.ac.in

2021-22

Introduction:

Customer Data using K-Means Clustering: Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. You'll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the centre of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'means' in the K-means refers to averaging of the data; that is, finding the centroid.

Problem statement:

To identify different segments of customers on the basis of their shopping behaviour to run targeted marketing campaign.Cluster analysis helps design better customer acquisition strategies and helps in business growth.

Advantages:

1. Develop customized marketing campaigns.
2. Design an optimal distribution strategy.
3. Choose specific product information and features of deployment.
4. Determine appropriate product pricing.

Disadvantages:

1. With the different representation of the data, the results achieved are also different.
2. Euclidean distance can unequally weigh the factors
3. It gives the local optima of the squared error function.
4. Sometimes choosing the centroids randomly cannot give fruitful results.

Libraries used:

Numpy:NumPy is a Python library used for working with arrays.

Pandas: pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation toll built on top of the Python programming language.

Matplotlib:Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations.

Seaborn: Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

plotly.graph_objs: The plotly.graph_objs module is the most important module that contains all of the class definitions for the objects that make up the plots you see.

from sklearn.cluster import KMeans: Scikit-learn is a free Python machine learning software, sometimes known as sklearn. K-means, is imported from it.

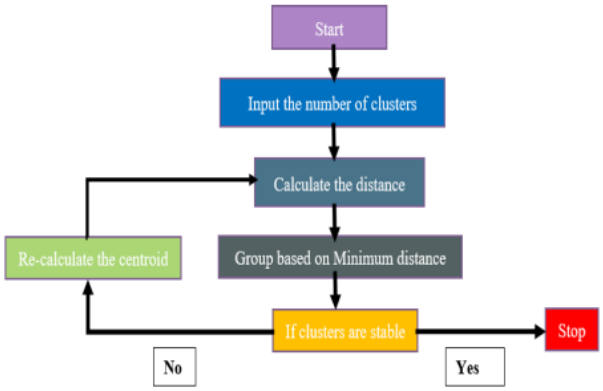
Dataset Details:

Customer	Gender	Age	Annual In	Spending	cluster
1	Male	19	15	39	4
2	Male	21	15	81	3
3	Female	20	16	6	4
4	Female	23	16	77	3
5	Female	31	17	40	4
6	Female	22	17	76	3
7	Female	35	18	6	4
8	Female	23	18	94	3
9	Male	64	19	3	4
10	Female	30	19	72	3

We have some basic data about the customers like Customer ID, age, gender, annual income and spending score. Spending Score is something you assign to the customer based on your defined parameters like customer behaviour and purchasing data.

K-Means Algorithm:

1. Randomly select centroids (centre of cluster) for each cluster.
2. Calculate the distance of all data points to the centroids.
3. Assign data points to the closest cluster.
4. Find the new centroids of each cluster by taking the mean of all data points in the cluster.
5. Repeat steps 2, 3 and 4 until all points converge and cluster centres stop moving.
6. Go to step-4 if there is a reassignment ,otherwise ,go o finish
7. The model is now complete.



Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
importplotly as py
import plotly.graph_objs as go
from sklearn.cluster import KMeans
import warnings
warnings.filterwarnings('ignore')
df = pd.read_csv('segmented_customers.csv')
df.head()
df.columns
df.info()
df.describe()
df.isnull().sum()
plt.figure(1 , figsize = (15 , 6))
n = 0
```

```
for x in ['Age' , 'Annual Income (k$)' , 'Spending Score (1-100)']:
plt.subplot(1 , 3 , n)
plt.subplots_adjust(hspace = 0.5 , wspace = 0.5)
sns.distplot(df[x] , bins = 15)
plt.title('Distplot of {}'.format(x))
plt.show()
sns.pairplot(df, vars = ['Spending Score (1-100)', 'Annual Income (k$)', 'Age'], hue = "Gender")
plt.figure(1 , figsize = (15 , 7))
plt.title('Scatter plot of Age v/s Spending Score', fontsize = 20)
plt.xlabel('Age')
plt.ylabel('Spending Score')
plt.scatter( x = 'Age', y = 'Spending Score (1-100)', data = df, s = 100
plt.show()
X1 = df[['Age' , 'Spending Score (1-100)']].iloc[: , :].values
inertia = []
for n in range(1 , 15):
algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
inertia.append(algorithm.inertia_)
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 15) , inertia , 'o')
plt.plot(np.arange(1 , 15) , inertia , '-' , alpha = 0.5)
```

```
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
algorithm = (KMeans(n_clusters = 4 ,init='k-means++', n_init = 10 ,max_iter=300,
tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X1)
labels1 = algorithm.labels_
centroids1 = algorithm.cluster_centers_
h = 0.02
x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
extent=(xx.min(), xx.max(), yy.min(), yy.max()),
cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'Age', y = 'Spending Score (1-100)', data = df, c = labels1, s = 100)
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Spending Score (1-100)') , plt.xlabel('Age')
```

```
plt.show()
algorithm = (KMeans(n_clusters = 5, init='k-means++', n_init = 10,
max_iter=300,
    tol=0.0001, random_state= 111 , algorithm='elkan'))
    algorithm.fit(X1)
    labels1 = algorithm.labels_
    centroids1 = algorithm.cluster_centers_
    h = 0.02
    x_min, x_max = X1[:, 0].min() - 1, X1[:, 0].max() + 1
    y_min, y_max = X1[:, 1].min() - 1, X1[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min,
    y_max, h))
    Z = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
    plt.figure(1 , figsize = (15 , 7) )
    plt.clf()
    Z = Z.reshape(xx.shape)
    plt.imshow(Z , interpolation='nearest',
    extent=(xx.min(), xx.max(), yy.min(), yy.max()),
    cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

    plt.scatter( x = 'Age', y = 'Spending Score (1-100)', data = df, c =
    labels1, s = 100)
    plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c =
    'red' , alpha = 0.5)
    plt.ylabel('Spending Score (1-100)') , plt.xlabel('Age')
    plt.show()
```

```
X2 = df[['Annual Income (k$)' , 'Spending Score (1-100)']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X2)
inertia.append(algorithm.inertia_)
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
plt.xlabel('Number of Clusters'), plt.ylabel('Inertia')
plt.show()
algorithm = (KMeans(n_clusters = 5 ,init='k-means++', n_init = 10 ,max_iter=300,
tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X2)
labels2 = algorithm.labels_
centroids2 = algorithm.cluster_centers_
h = 0.02
x_min, x_max = X2[:, 0].min() - 1, X2[:, 0].max() + 1
y_min, y_max = X2[:, 1].min() - 1, X2[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z2 = algorithm.predict(np.c_[xx.ravel(), yy.ravel()])
```

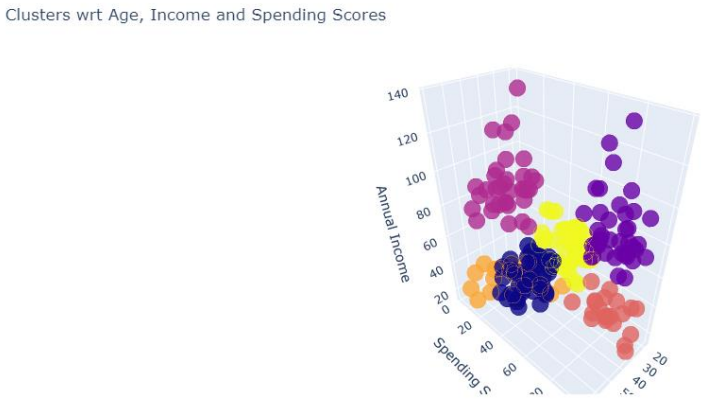
```
plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z2 = Z2.reshape(xx.shape)
plt.imshow(Z2 , interpolation='nearest',
extent=(xx.min(), xx.max(), yy.min(), yy.max()),
cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')


plt.scatter( x = 'Annual Income (k$)' ,y = 'Spending Score (1-100)' , data = df , c = labels2 ,
        s = 100 )
plt.scatter(x = centroids2[: , 0] , y =  centroids2[: , 1] , s = 300 , c = 'red' , alpha = 0.5)
plt.ylabel('Spending Score (1-100)') , plt.xlabel('Annual Income (k$)')
plt.show()
X3 = df[['Age' , 'Annual Income (k$)' ,'Spending Score (1-100)']].iloc[: , :].values
inertia = []
for n in range(1 , 11):
algorithm = (KMeans(n_clusters = n, init='k-means++', n_init = 10, max_iter=300,
tol=0.0001, random_state= 111, algorithm='elkan'))
algorithm.fit(X3)
inertia.append(algorithm.inertia_)
plt.figure(1 , figsize = (15 ,6))
plt.plot(np.arange(1 , 11) , inertia , 'o')
plt.plot(np.arange(1 , 11) , inertia , '-' , alpha = 0.5)
```

```
plt.xlabel('Number of Clusters') , plt.ylabel('Inertia')
plt.show()
algorithm = (KMeans(n_clusters = 6 ,init='k-means++', n_init = 10
,max_iter=300,
tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm.fit(X3)
labels3 = algorithm.labels_
centroids3 = algorithm.cluster_centers_

y_kmeans = algorithm.fit_predict(X3)
df['cluster'] = pd.DataFrame(y_kmeans)
df.head()
```

Result:



Conclusions:

This study demonstrates that client segmentation in shopping malls is achievable despite the fact that this form of machine learning application is highly useful in the market, a manager can concentrate all of his or her attention on each cluster that has been discovered and meet all of their requirements. Mall managers must be able to understand what customers require and, more importantly, how to meet those needs. analyze their purchasing habits, and establish frequent encounters with customers that make them feel comfortable in order to satisfy their demands.