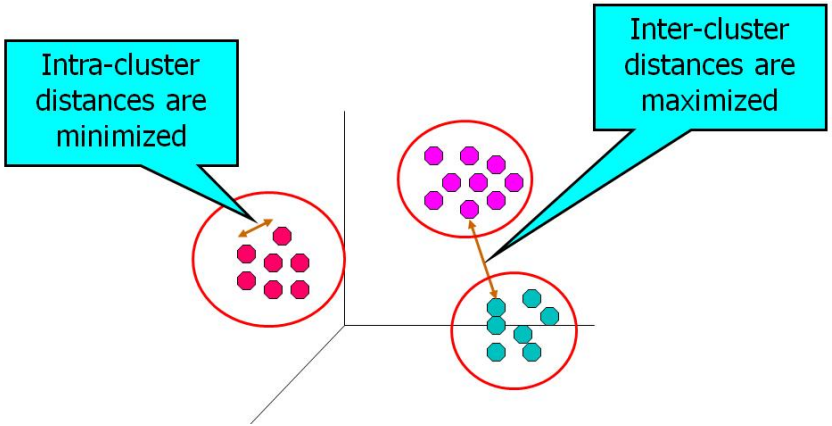
Non-Hierarchical Clustering
K-Means Method

Contents

Cluster Analysis

Cluster analysis is a class of statistical techniques that can be used to classify objects or cases into groups called **Clusters.**

- A cluster is a group of relatively homogeneous cases or observations.
- The observations are dissimilar to objects outside the cluster, particularly objects in other clusters.

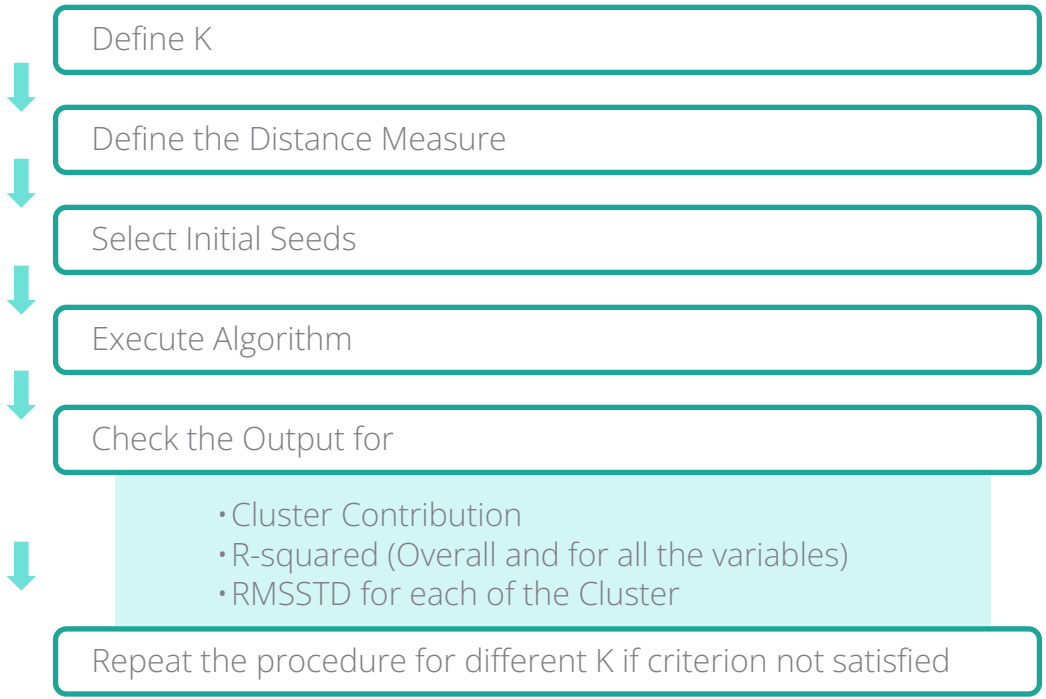


- Cluster Analysis is one of the unsupervised learning method.

K-Means Clustering

- K-Means Clustering is one of the most popular non-hierarchical clustering methods
- K –Means method is suitable for large data sets and widely used for customer segmentation in BFSI or retail domains
- The number of clusters (k) must be known a priori
  (Though in reality this may not be the case)
- Alternatively, cluster solutions can be observed for different k and evaluated to get the best possible cluster solution

K-Means Clustering – Steps



Case Study

Background

- A FMCG company has recorded information of customers based on their buying behaviour for a period of 1 year and would like to implement strategies by segmenting these customers into tiers.

Objective

- To create segment of customers.

Available Information

- Sample size is 1158.
- Variables : Custid, nsv, n_brands , n_bills, growth, region

Data Snapshot

RETAILERS DATA

Variables

Custid	nsv	n_brands	n_bills	growth	region
1001	2119456	7	14	-1.79	Mumbai
1002	1460163	12	42	-1.73	Mumbai
1003	147976	4	6	2.81	Mumbai
1018	2213576	14	14	5.69	Delhi
1019	2433971	11	25	3.71	Delhi

Columns	Description	Type	Measurement	Possible values
Custid	Unique customer ID	numeric	-	-
nsv	Net Sales Value	numeric	Rs.	positive values
n_brands	Number of unique brands purchased	numeric	-	positive values
n_bills	Number of bills generated	numeric	-	positive values
growth	Growth in net sales value	numeric	-	Positive & negative values
region	City of Customer	character	Delhi, Kolkata, Mumbai, Nagpur	4

K-Means Method in Python

```
# Importing Data
import pandas as pd
custsales = pd.read_csv("RETAILERS DATA.csv")
custsales_cl = custsales
custsales_cl = custsales_cl.drop(["Custid", "region"], axis = 1)
```

- **read_csv()** is used to import csv file. Our data is saved as an object named custsales. The subset of numeric variables is created.

```
# Scale (standardize) all variables.(subtract mean and divide by
# standard deviation)
import sklearn.preprocessing
custsales_cl2 = sklearn.preprocessing.scale(custsales_cl)
custsales_cl2

# Output
```

```
array([[ 1.3415212 , -0.57045618, -0.27177323, -1.40755324],
       [ 0.57976648,  0.03183071,  1.2604777 , -1.39462362],
       [-0.93634948, -0.93182832, -0.70955921, -0.41628239],
       ...,
       [ 1.04097324,  0.03183071,  0.43962899, -1.08862262],
       [-0.76570907, -0.57045618, -0.76428245, -0.19647886],
       [ 1.55237455,  2.8023504 ,  2.13604966,  1.37016007]])
```

K-Means Method in Python

```
# K means clustering
from sklearn.cluster import KMeans
CL = KMeans(n_clusters=4)
CL.fit(custsales_cl2)

# Compute centroids
centroids = CL.cluster_centers_
centroids
```

- **KMeans()** performs kmeans clustering on data matrix. The function requires data object and number of clusters to be formed.

```
# Output
array([[-0.83216038, -0.84148529, -0.72147912, -0.53311303],
       [ 1.18689039, -0.02445287,  0.30461312, -0.62608288],
       [ 1.0594337 ,  1.50599957,  1.62269348,  1.6235293 ],
       [-0.5018609 ,  0.09301541, -0.37791895,  0.05653806]])
```

K-Means Method in Python

```
# Create Segments

segment = pd.DataFrame(CL.labels_)
custsales = custsales.assign(segment = segment)
custsales.head()
```

```
# Output
```

	Custid	nsv	n_brands	n_bills	growth	region	segment
0	1001	2119456	7	14	-1.79	Mumbai	1
1	1002	1460163	12	42	-1.73	Mumbai	1
2	1003	147976	4	6	2.81	Mumbai	0
3	1004	1350474	13	30	-0.99	Delhi	1
4	1005	1414461	15	29	13.56	Delhi	2

K-Means Method in Python : Summarize Clusters Using Original Variables

```
# Aggregating data based on segments

nsv = custsales.groupby('segment')['nsv'].mean()
n_brands = custsales.groupby('segment')['n_brands'].mean()
n_bills = custsales.groupby('segment')['n_bills'].mean()
growth = custsales.groupby('segment')['growth'].mean()
pd.concat([nsv,n_brands,n_bills,growth],axis=1)

# Output
```

segment	nsv	n_brands	n_bills	growth
0	2.381509e+05	4.750000	5.782178	2.267847
1	1.985624e+06	11.532751	24.532751	1.836419
2	1.875311e+06	24.238095	48.619048	12.275762
3	5.240226e+05	12.507937	12.060317	5.004127

Interpretation :

- Cluster 3 is group of ‘Platinum’ clusters.
- Cluster 1 is a group of ‘non-performers’

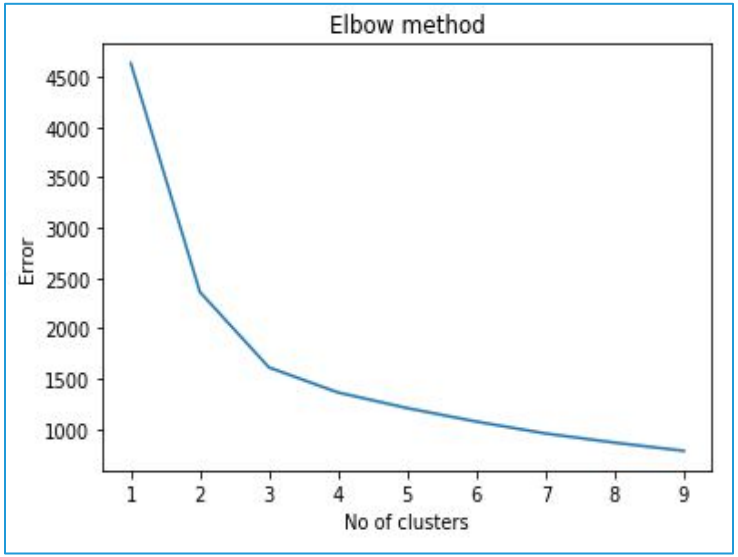
K-Means Method in Python
Elbow Method

```
# Optimum value of K by Elbow method
Error =[]
import matplotlib.pyplot as plt

for i in range(1, 10):
    CL = KMeans(n_clusters = i).fit(custsales_cl2)
    Error.append(CL.inertia_)

plt.plot(range(1, 10), Error)
plt.title('Elbow method')
plt.xlabel('No of clusters')
plt.ylabel('Error')
plt.show()
```

- We use Elbow method to find optimum value of K in which the sum of square distances from each point to its assigned center is calculated. (Within Sum of Squares)
- **CL.inertia_** gives Within Sum of Squares

K-Means Method in Python
Elbow Method

```
# Output
```



Interpretation :

- The location of a bend in the plot is generally considered as an indicator of the appropriate number of clusters.
- Here K= 3 or 4 is a good solution.
- The method is termed as Elbow Method.

Quick Recap

K-Means Clustering in R	• **`kMeans()`** function in Python performs K-Means Clustering • The KMeans function requires data object and value of K
Elbow Method	• Elbow method is used to determine optimum value of K