Sardar Patel Institute of Technology
Department of Electronics and Telecommunication Engineering
B.E. Sem-VII (2021.22)
EC433 - Machine Learning and AI

Experiment: Clustering using K mean and EM algorithm

Name:Anshal Padole **Roll No. 2019120033 Batch: C**

Objective: Apply EM algorithm to cluster a set of data stored in a .CSV file. Use the same data set for clustering using k-Means algorithm. Compare the results of these two algorithms and comment on the quality of clustering. You can add Java/Python ML library classes/API in the program.

Outcomes:

1. Find the clustering using Estimation and maximization algorithm.
2. Apply K mean and EM algorithm for clustering.
3. Compare the results of these two algorithms
4. Find accuracy, precision, recall of the model for data set.

System Requirements: Linux OS with Python and libraries or R or windows with MATLAB

Theory: EM algorithm

In applications of the EM algorithm in statistics, Y is a random vector taking values in RM and governed by the probability density function (pdf) or probability function (pf) fY (y|θtrue). The θ true is a parameter, or vector of parameters, to be estimated; the set Θ is the collection of all potential values of θ true. We have one realization, y, of Y , and we will estimate θtrue by maximizing the likelihood function of θ, given by L(θ) = fY (y|θ), over θ ∈ Θ, to get θ ML, a maximum-likelihood estimate of θ true. In the EM approach it is postulated that there is a second random vector, X, taking values in R N , such that, had we obtained an instance x of X, maximizing the function Lx(θ) = fX(x|θ) would have been computationally simpler than maximizing L(θ) = fY (y|θ). Clearly, maximizing Lx(θ) is equivalent to maximizing LLx(θ) = log fX(x|θ). In most discussions of the EM algorithm the vector y is called the “incomplete” data, while the x is the “complete” data and the situation is described by saying that there is “missing” data. In many applications of the EM algorithm this is suitable terminology. However, any data that we do not have but wish that we did have can be called “missing”. I will call the vector y the “observed” data and the x the “preferred” data. It would be reasonable to estimate x, using the current estimate θ k−1 and the data y, and then to use this estimate of x to get the next estimate θ k . Since it is LLx(θ) that we want to maximize, we estimate log fX(x|θ), rather than x itself. The EM algorithm estimates LLx(θ) as

E(log fX(X|θ)|y, θk−1) = Z fX|Y (x|y, θk−1) log fX(x|θ)dx, (2.1)
the conditional expected value of the random function log fX(X|θ), conditioned on the data y and the current estimate θ k−1 . This is the so-called E-step of the EM algorithm. It is convenient to define

Q(θ|θ k−1) .= Z fX|Y (x|y, θk−1) log fX(x|θ)dx. (2.2)

The M-step is to maximize Q(θ|θ k−1) to get θ k . For the case of probability functions we replace the integral with summation. An EM algorithm generates a sequence {θ k} of estimates of θtrue.

K mean clustering:

k-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move any more. Finally, this algorithm aims at minimizing an objective function know as squared error function given by:

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Procedure:

K-Means Algorithm Steps

1. Load data set
2. Clusters the data into k groups where k is predefined.
3. Select k points at random as cluster centers.
4. Assign objects to their closest cluster center according to the Euclidean distance function.
5. Calculate the centroid or mean of all objects in each cluster.

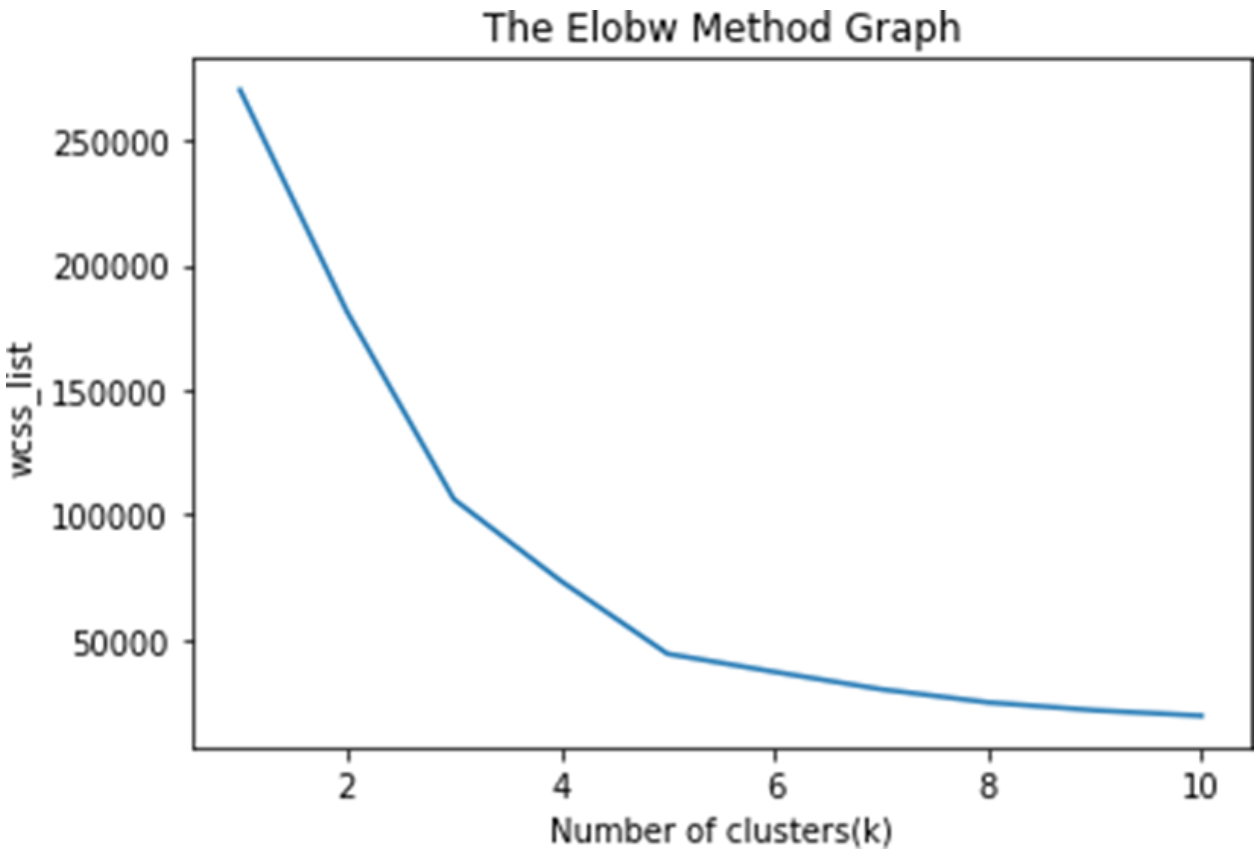
6. Repeat steps 3, 4 and 5 until the same points are assigned to each cluster in consecutive rounds.

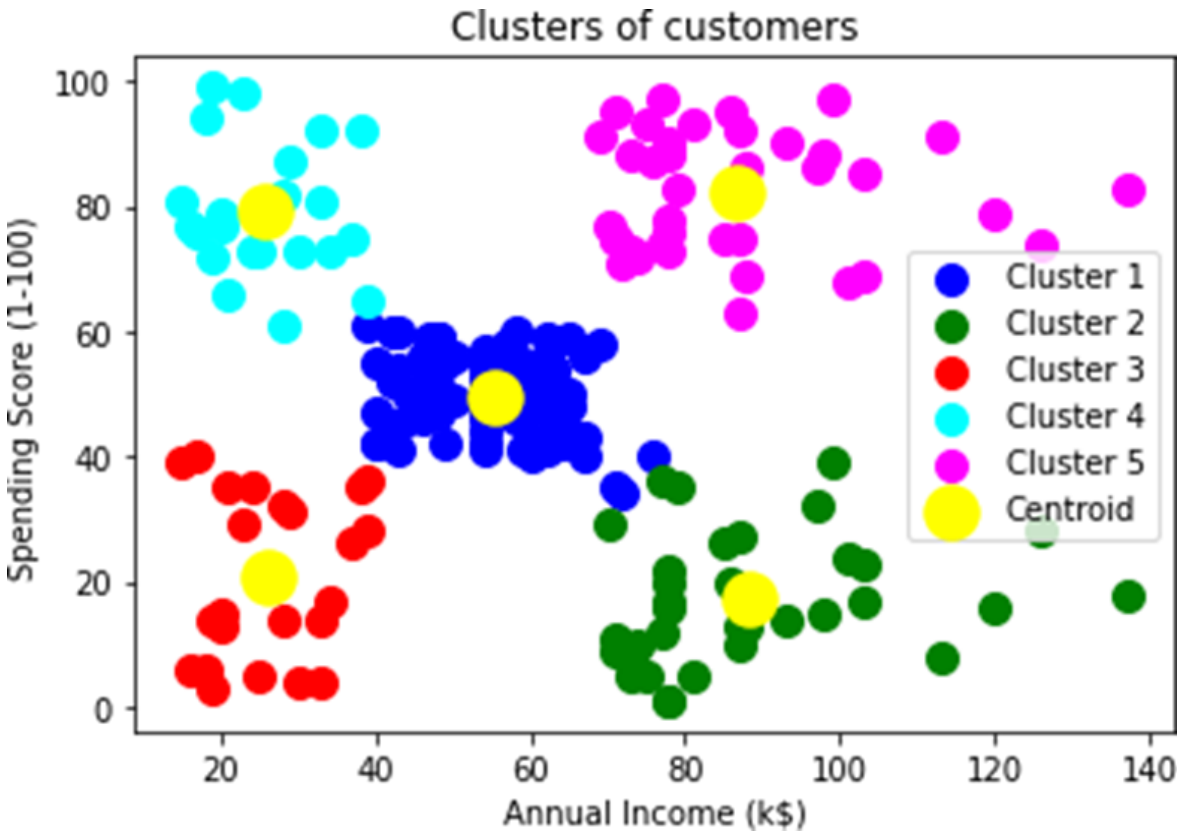
Code:

```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
# Importing the dataset
dataset = pd.read_csv('Mall_Customers.csv') x =
dataset.iloc[:, [3, 4]].values
#finding optimal number of clusters using the elbow method from
sklearn.cluster import KMeans

wcss_list= [] #Initializing the list for the values of WCSS



#Using for loop for iterations from 1 to 10. for i in
range(1, 11):
        kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
        kmeans.fit(x)
        wcss_list.append(kmeans.inertia_)
mtp.plot(range(1, 11), wcss_list)
mtp.title('The Elobw Method Graph')
mtp.xlabel('Number of clusters(k)')
mtp.ylabel('wcss_list')

mtp.show()

#training the K-means model on a dataset

kmeans = KMeans(n_clusters=5, init='k-means++', random_state= 42) y_predict=
kmeans.fit_predict(x)

#visulaizing the clusters

mtp.scatter(x[y_predict == 0, 0], x[y_predict == 0, 1], s = 100, c = 'bl ue', label = 'Cluster 1')
#for first cluster
mtp.scatter(x[y_predict == 1, 0], x[y_predict == 1, 1], s = 100, c = 'gr een', label = 'Cluster
2') #for second cluster
```

```
mtp.scatter(x[y_predict== 2, 0], x[y_predict == 2, 1], s = 100, c = 'red ', label = 'Cluster 3')
#for third cluster
mtp.scatter(x[y_predict == 3, 0], x[y_predict == 3, 1], s = 100, c = 'cy an', label = 'Cluster 4')
#for fourth cluster
mtp.scatter(x[y_predict == 4, 0], x[y_predict == 4, 1], s = 100, c = 'ma genta', label = 'Cluster
5') #for fifth cluster mtp.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s
= 300, c = 'yellow', label = 'Centroid')
mtp.title('Clusters of customers')
mtp.xlabel('Annual Income (k$)')
mtp.ylabel('Spending Score (1-100)')
mtp.legend()

mtp.show()
```

Output:





Conclusion:

(1) Through this experiment, we have successfully performed clustering of given input dataset using K-mean and EM algorithm.
(2) From the above elbow method graph, the elbow point is at 5. This means that the number of clusters formed of above dataset is 5.
(3) These clusters are shown in next output graph through clusters of five different colours.
(4) The clusters are formed between two parameters of the dataset; Annual income of customer and Spending. Thus, above clusters can be understood as follows,

- Cluster1 shows the customers with average salary and average spending so we can categorize these customers as ideal.
- Cluster2 shows the customer has a high income but low spending, so we can categorize them as careful.
- Cluster3 shows the low income and also low spending so they can be categorized as sensible.
- Cluster4 shows the customers with low income with very high spending so they can be categorized as careless.
- Cluster5 shows the customers with high income and high spending so they can be categorized as target, and these customers can be the most profitable customers for the mall owner.