K- MEANS CLUSTERING USING PCA

Manohar Akula
Ira A. Fulton Schools of Engineering
Arizona State University
Tempe, USA
makula@asu.edu

Abstract*— Clustering is employed in a variety of fields, including Engineering, Biology, Psychology, Economics, and Recommendation systems. Because the number of clusters or model parameters is seldom known and must be determined before clustering, the fundamental issue of cluster analysis is that the number of clusters or model parameters is rarely known and must be determined before clustering. A clustering method with several clusters has been presented. The k-means approach is a basic and quick clustering technique among them. We use a k-means strategy to solve the problem of cluster number selection. End users might be asked to supply several clusters ahead of time.

INTRODUCTION

"Clustering is the process of dividing the datasets into groups, consisting of similar data-points". Clustering obeys the following data: Points in the same group are as similar as possible, and Points in different group are as dissimilar as possible. Examples of the Clustering: Items arranged in a mall, Books arranged in the library, etc.

Types of Clustering:

i. Exclusive Clustering
ii. Overlapping Clustering
iii. Hierarchical Clustering

Exclusive Clustering:

It is a Hard clustering process, and the Data point or Item belongs exclusively to one
Example:
K- Means Clustering

Fig.1: *The figure shows the Non-overlapping of clusters (K-Means)* **[1]**

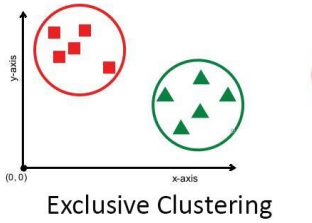

Fig.1

K-Means Clustering:

The objective of the K-Means Clustering algorithm is to group similar elements or data points into a cluster.
It is a simple clustering algorithm & "K'' in K-Means represent the number of clusters.

Dimensionality Reduction:

What if there are more features in a dataset than there are features? It becomes more difficult to picture and then work on the training set. There's a probability that most of these characteristics are connected and so redundant. In these situations, dimensionality reduction methods are used. Dimensionality reduction is the process of lowering the number of random variables under consideration by producing a set of primary variables.

The objective is to keep as much information as feasible while reducing the dataset's dimensionality. This is accomplished by feature selection and feature extraction, in which the feature space is reduced by removing features, then creating new independent features based on old independent variables, and finally removing the least important variables.

Principal Components Analysis (PCA):

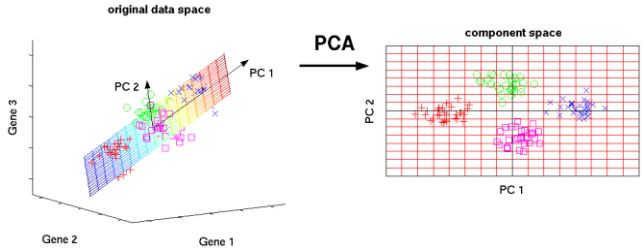

Fig.2

Fig.2: *The figure shows the transformation of a high dimensional data (3 dimension) to low dimensional data (2 dimension) using PCA* **[2]**

PCA is a method for transforming a dataset's columns into a new collection of characteristics known as Principal Components. This successfully compresses a huge portion of the data throughout the whole dataset into fewer feature columns. This allows for dimensionality reduction and the visualization of any class or cluster separation.

PROCEDURE

The k-means clustering technique is one of the most often used clustering methods. It creates k locations at random as initial centroids, with k being a user-specified number. The cluster with the closest centroid is then allocated to each location. Then, by calculating the mean of each cluster's data points, the centroid of each cluster is updated. Some data points may be transferred from one cluster to another. We construct fresh centroids and assign data points to the appropriate clusters once more. We repeat the assignment and update the centroids until the convergence conditions are fulfilled, i.e., no point switches clusters, or the centroids remain constant.

The distance between data points and centroids is generally calculated using Euclidean distance in this technique. This method describes pseudocode for the k-means clustering algorithm. The following equation describes the Euclidean distance between two multi-dimensional data values X = (x1, x2, x3... xm) and Y = (y1, y2, y3... ym):

$$d(X,Y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+..+(x_m-y_m)^2}$$

Fig.3

***Fig 3**: The figure shows the generalized distance formula for points given by Cartesian coordinates in n-dimensional Euclidean space.*

The important stages in K-Means Clustering:

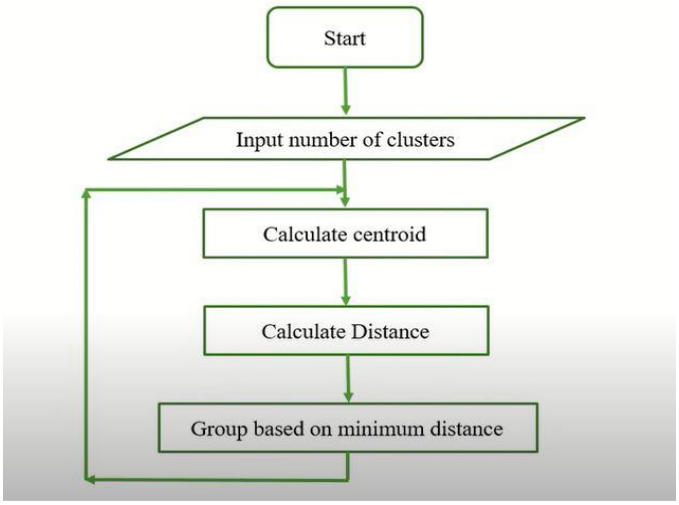

Fig.4

***Fig 4**: The figure shows the Flow-chart of K-Means Clustering.*

1. Creating centroids from scratch – It is accomplished by selecting K spots at random. These points can be found in the dataset, or they can be generated at random.

Drawback: The K-means method has the drawback of being sensitive to the initialization of the centroids or mean points. As a result, if a centroid is set to be a "far-off" point.

Solution: To overcome the above problem, we are using K mean++ algorithm. This approach guarantees that the centroids are correctly initialized and that the clustering quality is improved. The rest of the process is identical to the regular K-means algorithm, except for initialization.

2. Assigning clusters – After determining the distance from the centroid and assigning it to the centroid with the shortest distance, the clusters are allocated to each point in the dataset.

3. Re-calculating the centroids – The centroid of each cluster formed by us is calculated to update the centroids.

Steps involved in PCA:

1. Standardizing each column: Standardization is required prior to running PCA since it is extremely sensitive to deviations. As a result, the purpose of standardization is to have mean = 0 and variance = 1 in all feature spaces.

2. Computation of the Covariance Matrix: Covariance is used to identify how two variables are connected to one another, or whether they are going in the same direction with regard to one another. As covariance is positive, it signifies that when one variable rises, the other rises with it. When covariance is negative, the inverse is true. The covariance matrix is used to determine the covariance of all possible column combinations that result in a square matrix.

3. Computing Eigen values and Eigen Vectors: Eigen values and Eigen vectors are used to explain the amount of variation. It's also utilized to figure out how the columns relate to one another. Eigenvectors have a magnitude of one.

4. Identifying the Features of the Principal Components: The dot product of the standardized columns and the eigen vector yields the Principal Component features.

RESULT

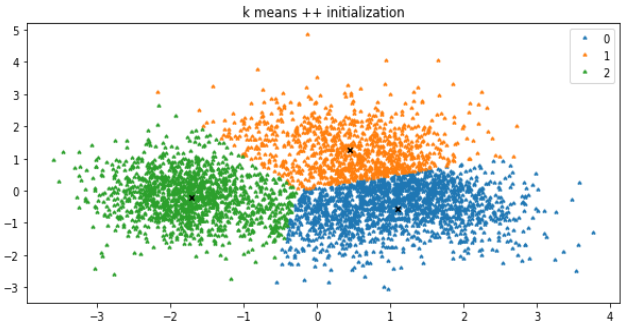

Fig.5

***Fig 5**: The figure shows the plot of K means++ clustering using PCA for K =3.*

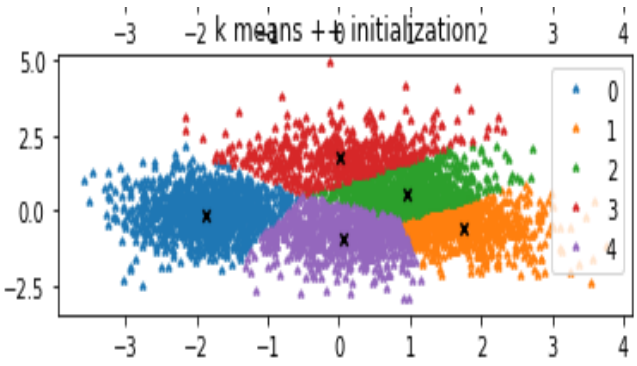

Fig.6

***Fig 6**: The figure shows the plot of K means++ clustering using PCA for K =5.*

REFERENCES

[1]. https://medium.com/swlh/a-noobs-guide-to-k-means-215a600bd1eb

[2]. https://www.analyticsvidhya.com/blog/2016/03/pca-practical-guide-principal-component-analysis-python/

Appendix :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
plt.figure(figsize=(10,10))
df = pd.read_csv("yourdata.csv", header = None)
x = df.to_numpy()
scaler = StandardScaler()
X = scaler.fit_transform(x)
pca = PCA(2)
data = pca.fit_transform(X)
for k in [3,5]:
    model = KMeans(n_clusters = k, init = "random")
    label = model.fit_predict(data)
    centers = np.array(model.cluster_centers_)
    print(centers)
    axis1 = plt.subplot(2, 1, 1)
    uniq = np.unique(label)
    for i in uniq:
        axis1.scatter(data[label == i , 0] , data[label == i , 1] ,s=10, label = i ,   marker= 10)
    axis1.scatter(centers[:,0], centers[:,1], marker="x",s=20, color='k')
    axis1.legend()
    axis1.set_title("random initialization")
    model = KMeans(n_clusters = k, init = "k-means++")
    label = model.fit_predict(data)
    centers = np.array(model.cluster_centers_)
    print(centers)
    axis2 = plt.subplot(2, 1,  2)
    uniq = np.unique(label)
    for i in uniq:
        axis2.scatter(data[label == i , 0] , data[label == i , 1] ,s=10, label = i ,  marker= 10)
    axis2.scatter(centers[:,0], centers[:,1], marker="x",s=20, color='k')
    axis2.legend()
    axis2.set_title("k means ++ initialization")
    plt.show()
```