**Name: YATHIN K V**

**USN: 1PE17IS105**

**Class: ISE 7B**

**Machine Learning Laboratory**

Program 8

**Apply EM algorithm to cluster a set of data stored in a .CSV file. Use the same dataset**

**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.**

K-Means Algorithm :

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

**PROGRAM:**

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.cluster import KMeans

import pandas as pd

import numpy as np

from sklearn import preprocessing

#from sklearn.mixture import GMM # Used for older versions of sklearn

from sklearn.mixture import GaussianMixture

iris = datasets.load\_iris()

X = pd.DataFrame(iris.data)

X.columns = ['Sepal\_Length', 'Sepal\_Width', 'Petal\_Length', 'Petal\_Width']

X\_norm = preprocessing.normalize(X)

y = pd.DataFrame(iris.target)

y.columns = ['Targets']

# K-Means Model

model = KMeans(n\_clusters = 3)

model.fit(X\_norm)

# EM Model

#gmm = GMM(n\_components = 3) # Used for older versions of sklearn

gmm = GaussianMixture(n\_components = 3)

gmm.fit(X\_norm)

gmm\_y = gmm.predict(X\_norm)

plt.figure(figsize = (10, 10))

colormap = np.array(['red', 'lime', 'black'])

# Real Clusters

plt.subplot(2, 2, 1)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c = colormap[y.Targets], s = 300)

plt.title('Real Clusters')

plt.xlabel('Petal Lenght')

plt.ylabel('Petal Width')

# K-Means Output

plt.subplot(2, 2, 2)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c = colormap[model.labels\_], s = 300)

plt.title('K-Means Clustering')

plt.xlabel('Petal Length')

plt.ylabel('Petal Width')

# EM Output

plt.subplot(2, 2, 3)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c = colormap[gmm\_y], s = 300)

plt.title('GMM Clustering')

plt.xlabel('Petal Length')

plt.ylabel('Petal Width')

plt.show()

**OUTPUT:**

