**Pgm 8:** 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.

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.cluster import KMeans

import sklearn.metrics as sm

import pandas as pd

import numpy as np

iris = datasets.load\_iris()

X = pd.DataFrame(iris.data)

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

from sklearn import datasets

from sklearn.cluster import KMeans

import sklearn.metrics as sm

import pandas as pd

import numpy as np

#%matplotlib inline

l1 = [0,1,2]

def rename(s):

l2 = []

for i in s:

if i not in l2:

l2.append(i)

for i in range(len(s)):

pos = l2.index(s[i])

s[i] = l1[pos]

return s

iris = datasets.load\_iris()

print("\n IRIS TARGET NAMES:\n",iris.target\_names)

X = pd.DataFrame(iris.data)

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

y = pd.DataFrame(iris.target)

y.columns = ['Targets']

plt.figure(figsize=(14,7))

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

plt.subplot(1, 2, 1)

plt.scatter(X.Sepal\_Length,X.Sepal\_Width, c=colormap[y.Targets], s=40)

plt.title('Sepal')

plt.subplot(1, 2, 2)

plt.scatter(X.Petal\_Length,X.Petal\_Width, c=colormap[y.Targets], s=40)

plt.title('Petal')

plt.show()

print("Actual Target is:\n", iris.target)

model = KMeans(n\_clusters=3)

model.fit(X)

plt.figure(figsize=(14,7))

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

plt.subplot(1, 2, 1)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c=colormap[y.Targets], s=40)

plt.title('Real Classification')

plt.subplot(1, 2, 2)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c=colormap[model.labels\_], s=40)

plt.title('K Mean Classification')

plt.show()

km = rename(model.labels\_)

print("\nWhat KMeans thought: \n", km)

print("Accuracy of KMeans is ",sm.accuracy\_score(y, km))

print("Confusion Matrix for KMeans is \n",sm.confusion\_matrix(y, km))

from sklearn import preprocessing

scaler = preprocessing.StandardScaler()

scaler.fit(X)

xsa = scaler.transform(X)

xs = pd.DataFrame(xsa, columns = X.columns)

print("\n",xs.sample(5))

from sklearn.mixture import GaussianMixture

gmm = GaussianMixture(n\_components=3)

gmm.fit(xs)

y\_cluster\_gmm = gmm.predict(xs)

plt.subplot(1, 2, 1)

plt.scatter(X.Petal\_Length, X.Petal\_Width, c=colormap[y\_cluster\_gmm], s=40)

plt.title('GMM Classification')

plt.show()

em = rename(y\_cluster\_gmm)

print("\nWhat EM thought: \n", em)

print("Accuracy of EM is ",sm.accuracy\_score(y, em))

print("Confusion Matrix for EM is \n", sm.confusion\_matrix(y, em))

**Output**

IRIS TARGET NAMES:

['setosa' 'versicolor' 'virginica']

Actual Target is:

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2

2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

2 2]

What KMeans thought:

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 2 2 1 2 2 2 2

2 2 1 1 2 2 2 2 1 2 1 2 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2 2 1 2

2 1]

Accuracy of KMeans is 0.8933333333333333

Confusion Matrix for KMeans is

[[50 0 0]

[ 0 48 2]

[ 0 14 36]]

Sepal\_Length Sepal\_Width Petal\_Length Petal\_Width

100 0.553333 0.558611 1.274295 1.712096

19 -0.900681 1.709595 -1.283389 -1.183812

50 1.401508 0.328414 0.535409 0.264142

53 -0.416010 -1.743357 0.137547 0.132510

21 -0.900681 1.479398 -1.283389 -1.052180

What EM thought:

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 2 1

1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2

2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

2 2]

Accuracy of EM is 0.9666666666666667

Confusion Matrix for EM is

[[50 0 0]

[ 0 45 5]

[ 0 0 50]]