import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make\_blobs

X,y = make\_blobs(n\_samples = 500,n\_features = 2,centers = 6,random\_state = 23)

fig = plt.figure(0)

plt.grid(True)

plt.scatter(X[:,0],X[:,1])

plt.show()

k = 6

clusters = {}

np.random.seed(23)

for idx in range(k):

center = 2\*(2\*np.random.random((X.shape[1],))-1)

points = []

cluster = {

'center' : center,

'points' : []

}

clusters[idx] = cluster

clusters

plt.scatter(X[:,0],X[:,1])

plt.grid(True)

for i in clusters:

center = clusters[i]['center']

plt.scatter(center[0],center[1],marker = '\*',c = 'red')

plt.show()

plt.scatter(X[:,0],X[:,1])

plt.grid(True)

for i in clusters:

center = clusters[i]['center']

plt.scatter(center[0],center[1],marker = '\*',c = 'red')

plt.show()

def distance(p1,p2):

return np.sqrt(np.sum((p1-p2)\*\*2))

#Implementing E step

def assign\_clusters(X, clusters):

for idx in range(X.shape[0]):

dist = []

curr\_x = X[idx]

for i in range(k):

dis = distance(curr\_x,clusters[i]['center'])

dist.append(dis)

curr\_cluster = np.argmin(dist)

clusters[curr\_cluster]['points'].append(curr\_x)

return clusters

#Implementing the M-Step

def update\_clusters(X, clusters):

for i in range(k):

points = np.array(clusters[i]['points'])

if points.shape[0] > 0:

new\_center = points.mean(axis =0)

clusters[i]['center'] = new\_center

clusters[i]['points'] = []

return clusters

def pred\_cluster(X, clusters):

pred = []

for i in range(X.shape[0]):

dist = []

for j in range(k):

dist.append(distance(X[i],clusters[j]['center']))

pred.append(np.argmin(dist))

return pred

clusters = assign\_clusters(X,clusters)

clusters = update\_clusters(X,clusters)

pred = pred\_cluster(X,clusters)

plt.scatter(X[:,0],X[:,1],c = pred)

for i in clusters:

center = clusters[i]['center']

plt.scatter(center[0],center[1],marker = '^',c = 'red')

plt.show()