NAME: Kshitija Nalawade

ROLL NO: 4202041

BATCH: BB2

**Problem Statement:** We have given a collection of 8 points. P1=[0.1,0.6] P2=[0.15,0.71] P3=[0.08,0.9] P4=[0.16, 0.85] P5=[0.2,0.3] P6=[0.25,0.5] P7=[0.24,0.1] P8=[0.3,0.2]. Perform the k-mean clustering with initial centroids as m1=P1 =Cluster#1=C1 and m2=P8=cluster#2=C2. Answer the following

1] Which cluster does P6 belongs to?

2] What is the population of cluster around m2?

3] What is updated value of m1 and m2?

**CODE:**

In [1]:

**from** **copy** **import** deepcopy

**import** **numpy** **as** **np**

**import** **pandas** **as** **pd**

**from** **matplotlib** **import** pyplot **as** plt

**import** **math**

In [2]:

df = pd.read\_csv('datas.csv')

df

Out[2]:

|  | **X** | **Y** |
| --- | --- | --- |
| **0** | 0.10 | 0.60 |
| **1** | 0.15 | 0.71 |
| **2** | 0.08 | 0.90 |
| **3** | 0.16 | 0.85 |
| **4** | 0.20 | 0.30 |
| **5** | 0.25 | 0.50 |
| **6** | 0.24 | 0.10 |
| **7** | 0.30 | 0.20 |

In [3]:

X = np.array(df)

In [4]:

c\_x = np.array([0.1,0.3])

c\_y = np.array([0.6,0.2])

centroids = np.array(list(zip(c\_x,c\_y)))

centroids

Out[4]:

array([[0.1, 0.6],

[0.3, 0.2]])

In [5]:

**class** **K\_Means**:

**def** \_\_init\_\_(self, k=2, tol=0.001, max\_iter=300):

self.k = k

self.tol = tol

self.max\_iter = max\_iter

**def** fit(self,data,centroids):

self.centroids = {}

**for** i **in** range(self.k):

self.centroids[i] = centroids[i]

**for** i **in** range(self.max\_iter):

self.classifications = {}

**for** i **in** range(self.k):

self.classifications[i] = []

**for** featureset **in** data:

distances = [np.linalg.norm(featureset-self.centroids[centroid]) **for** centroid **in** self.centroids]

classification = distances.index(min(distances))

self.classifications[classification].append(featureset)

prev\_centroids = dict(self.centroids)

**for** classification **in** self.classifications:

self.centroids[classification] = np.average(self.classifications[classification],axis=0)

optimized = **True**

**for** c **in** self.centroids:

original\_centroid = prev\_centroids[c]

current\_centroid = self.centroids[c]

**if** np.sum((current\_centroid-original\_centroid)/original\_centroid\*100.0) > self.tol:

print(np.sum((current\_centroid-original\_centroid)/original\_centroid\*100.0))

optimized = **False**

**if** optimized:

**break**

**def** predict(self,data):

distances = [np.linalg.norm(data-self.centroids[centroid]) **for** centroid **in** self.centroids]

classification = distances.index(min(distances))

**return** classification

In [6]:

model = K\_Means()

model.fit(X, centroids)

66.66666666666666

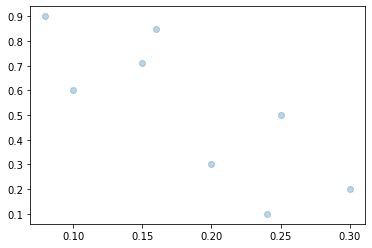
In [7]:

*## Initial Data Points*

plt.figure()

plt.scatter(X[:,0],X[:,1],alpha=0.3)

plt.show()



In [8]:

*# data points with 2 cluster centroids*

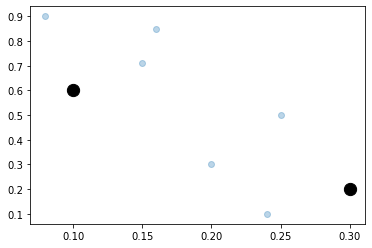
plt.figure()

plt.scatter(X[:,0],X[:,1],alpha=0.3)

plt.scatter(c\_x,c\_y, marker='o', c='black', s=150)

Out[8]:

<matplotlib.collections.PathCollection at 0x24fce7b72b0>



In [9]:

*#Result after kmeans clustering*

colors = ['r','b']

**for** centroid **in** model.centroids:

plt.scatter(model.centroids[centroid][0], model.centroids[centroid][1],

marker="o", color="k", s=150, linewidths=5)

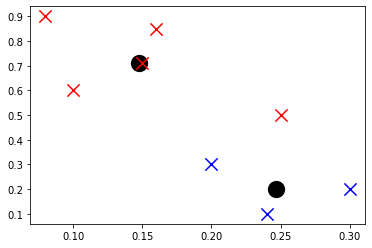
**for** classification **in** model.classifications:

color = colors[classification]

**for** featureset **in** model.classifications[classification]:

plt.scatter(featureset[0], featureset[1], marker="x", color=color, s=150, linewidths=5)

plt.show()



In [10]:

print("Point P6 belongs to cluster", model.predict([0.25,0.5]))

Point P6 belongs to cluster 0

In [11]:

print("Population of cluster2 is", len(model.classifications[1]))

Population of cluster2 is 3

In [12]:

print("Initial values of cluster centroids m1 and m2")

print("m1=",centroids[0])

print("m2=",centroids[1])

print("**\n**Updated value of cluster centroids m1 and m2")

print("m1=",model.centroids[0])

print("m2=",model.centroids[1])

Initial values of cluster centroids m1 and m2

m1= [0.1 0.6]

m2= [0.3 0.2]

Updated value of cluster centroids m1 and m2

m1= [0.148 0.712]

m2= [0.24666667 0.2 ]

In [ ]: