Problem Statement

We have given a collection of 8 point

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 belong to?
2. What is the population of cluster around m2?
3. What is updated value of m1 and m2?

K Means Clustering Key Points

- Unsupervised Machine Learning Algorithm
- Partition Clustering
- k = number of classes, k>2
- A cluster is a collection of data points aggregated together because of certain similarities.
- in the beginning the centroids are taken randomly
- ends when
    - centroids have stabilized
    - iteration has exceeded the limit
- k can be determined using elbow method
    - Plot a graph of k vs Distortion
    - Identify the Elbow of the curve
- Advantages
    - Relatively simple to implement.
    - Scales to large data sets.
    - Guarantees convergence.
- Disadvantages
    - Choosing k manually
    - Clustering data of varying sizes and density.

- Clustering outliers.
- Curse of Dimensionality
- Dependant on initial cluster center values

K Means Clustering Algorithm

Algorithm

1. for each datapoint in dataset

   1.1 for each center in centers

   - 1.1.1. get distance between datapoint and center

   1.2 select center closest to datapoint

   1.3 assign cluster based on closest center

2. for each cluster

   2.1 assign new center as centroid of datapoints in cluster

3. if new centers = old centers then return new centers else go to step 1

```
#importing libraries
import numpy as np
import numpy.matlib
import matplotlib.pyplot as plt
import pandas as pd
```

```
data = [
    [0.1, 0.6],
    [0.15, 0.71],
    [0.08, 0.9],
    [0.16, 0.85],
    [0.2, 0.3],
    [0.25, 0.5],
    [0.24, 0.1],
    [0.3, 0.2]
]

data = pd.DataFrame(data, columns = ['x', 'y'])

centroids = [
    [0.1, 0.6],
    [0.3, 0.2]
]

k = 2
```

```
#calculate distance between 2 points
def calc_distance(x1, x2):
    return (sum((x1 - x2)**2))**0.5
```

```
#check the point is closer to which centroid
def assign_clusters(centroids, data):
    clusters = []
    for i in range(data.shape[0]):
        distances = []
        for centroid in centroids:
            distances.append(calc_distance(centroid, data.iloc[i]))
        print(distances)
        cluster = [z for z, val in enumerate(distances) if val==min(distances)]
        clusters.append(cluster[0])

    return clusters
```

```
#new centroid = mean of all the points belonging to that cluster
def calc_centroids(clusters, data):
    new_centroids = []
    cluster_df = pd.concat([pd.DataFrame(data), pd.DataFrame(clusters, columns=['c
    for c in set(cluster_df['cluster']):
        current_cluster = cluster_df[cluster_df['cluster']==c][cluster_df.columns[
        cluster_mean = current_cluster.mean(axis=0)
        new_centroids.append(cluster_mean)
    return new_centroids
```

```
clusters = assign_clusters(centroids, data)
print(clusters)
```

```
[0.0, 0.44721359549995787]
[0.12083045973594571, 0.5316013544000805]
[0.3006659275674582, 0.7337574531137656]
[0.2570992026436488, 0.6649060083951716]
[0.31622776601683794, 0.14142135623730948]
[0.18027756377319945, 0.30413812651491096]
[0.5192301994298868, 0.11661903789690602]
[0.44721359549995787, 0.0]
[0, 0, 0, 0, 1, 0, 1, 1]
```

```
centroids = calc_centroids(clusters, data)
print(centroids)
```

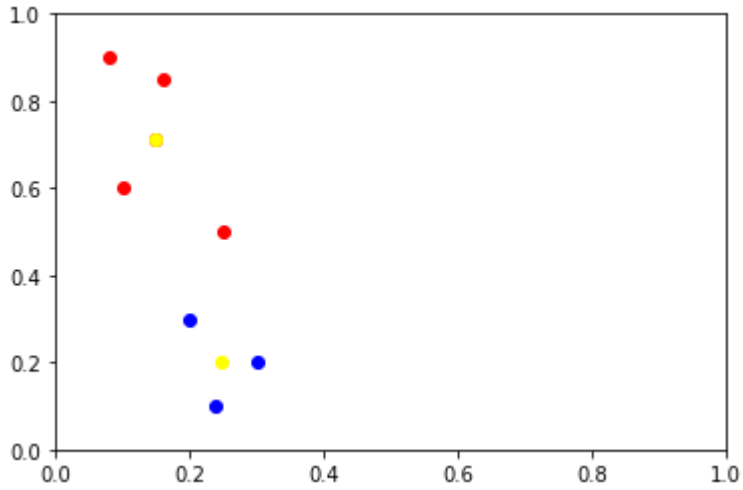
```
[x    0.148
y    0.712
dtype: float64, x    0.246667
y    0.200000
dtype: float64]
```

```
plt.plot()
colors = ['red', 'blue']

for i in range(data.shape[0]):
    plt.scatter(data.iloc[i]['x'], data.iloc[i]['y'], c = colors[clusters[i]])

for i in centroids:
    plt.scatter(i[0], i[1], c = 'yellow')

plt.axis([0, 1, 0, 1])
plt.show()
```




```
print(f'P6 belongs to cluster C{clusters[5]+1} coloured in {colors[clusters[5]]}')
```

```
P6 belongs to cluster C1 coloured in red
```

```
print(f'Population of cluster around P8 is {clusters.count(clusters[7])} (Cluster
```

```
Population of cluster around P8 is 3 (Cluster is shown in blue colour)
```

```
print(f'Updated values of centroids are ({centroids[0][0]}, {centroids[0][1]}) and
```

```
Updated values of centroids are (0.148, 0.712) and (0.24666666666666667, 0.20000
```

https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1

```
from sklearn.cluster import KMeans
Kmean = KMeans(n_clusters=2)
Kmean.fit(X)
```