Question 3

Clustering algorithm- part2: Use K-Means or K-Means++ on the data set you have to generate ahard clustering of the data using the following.

(a) Get K clusters where K is varied to have values 2, 5, 10, 50, and 100. In this case, each pattern belongs to only one of the K clusters. Note that AM is a matrix of size n × K here. It will have in its ith row a 1 in the jth column if the ith pattern belongs to the jth cluster. All the remaining K – 1 columns (other than the jth column) in the ith row will have zeros. (b) Use the K cluster centroids as the rows of the CRM matrix of size K × d. Here, the ith centroid will be the ith row of CRM. (c) Here also the product matrix P M = AM.CRM is of size n × d. Compute Error(K) for each value of K chosen here using the approximation DM ≈ PM corresponding to the hard clustering. (d) The error in the approximation is given by i=1 j=1 PM respectively. (e) Example: Consider nine three-dimensional patterns to be clustered into K = 3 clusters. ∑n ∑d (DMij –PMij)2 Error(K)= (f) Note that Error(3) = 12 in this example.

Answer : Program is run on the data points and the output seen as below.

Implemented -kmeans ++

Notation :

Labels - Cluster Index - Example:

labels [0 0 0 2 2 2 1 1 1]

DP 1 mapped to cluster 0 DP 2 mapped to cluster 1 DP 8 mapped to cluster 1

DM Matrix for Data Points

1 2 2

2 1 3

3 3 1

6 1 1

6 2 3

6 3 2

6 6 6

6 6 7

6 6 8

centroids = [[6. 6. 7.]

[2. 2. 2.]

[6. 2. 2.]]

labels [1 1 1 2 2 2 0 0 0]

CRM Matrix for Data Points

6.0 6.0 7.0

2.0 2.0 2.0

6.0 2.0 2.0

AM Matrix for Data Points

0.0 1.0 0.0

0.0 1.0 0.0

0.0 1.0 0.0

0.0 0.0 1.0

0.0 0.0 1.0

0.0 0.0 1.0

1.0 0.0 0.0

1.0 0.0 0.0

1.0 0.0 0.0

PM Matrix for Data Points

2.0 2.0 2.0

2.0 2.0 2.0

2.0 2.0 2.0

6.0 2.0 2.0

6.0 2.0 2.0

6.0 2.0 2.0

6.0 6.0 7.0

6.0 6.0 7.0

6.0 6.0 7.0

Error ==> 12.0 for Threshold 3

In [3]:
```
import pandas as pd
import numpy as np
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D  # noqa: F401 unused import
from matplotlib import cm
import seaborn as sns
from sklearn.model_selection import train_test_split

w = 4
h = 4
d = 70
```

In [2]:

```
cluster_list_pattern={}
Error_Threshold_List=[]

PATTERNS = 1000
FEATURES = 100

PATTERNS= 9
FEATURES = 3

cluster_leader_list= []

def eqclidean_dist(leader_features, test_features):
    sum = 0
    for i in range(len(leader_features) ):
        sum = sum + np.power( (leader_features[i] - test_features[i] ), 2)

    eqcli = "{:.2f}".format(np.sqrt(sum))

    #print(f"{name} is an {type_of_company} company.")
    #print(f"Eqclidean distance {eqcli }")
    return np.sqrt(sum)

def process_clustering(pl, thresh_hold):

    from sklearn.cluster import KMeans

    # Number of clusters
    kmeans = KMeans(n_clusters=3)
    # Fitting the input data
    kmeans = kmeans.fit(pl)
    # Getting the cluster labels
    labels = kmeans.predict(pl)
    # Centroid values
    centroids = kmeans.cluster_centers_

    print(f"centroids = {centroids} labels  {kmeans.labels_}")

    # Determine Assignment Matrix

    ROWS = len(pl)
    COLS = len(centroids)
    AM = np.zeros((ROWS,COLS))
```

```
    for k,cl in enumerate(labels):
        AM[k,cl] = 1


    # Determine C R Matrix
    CRM = centroids

    # Calcualte Multiplication
    CRM_DataFrame = pd.DataFrame(data=CRM)
    print("CRM Matrix for Data Points \n")
    print(f"{CRM_DataFrame.to_string(header=None,index=False)}")

    AM_DataFrame = pd.DataFrame(data=AM)

    print("AM Matrix for Data Points \n")
    print(f"{AM_DataFrame.to_string(header=None,index=False)}")

    #PM = CRM_DataFrame.dot(AM_DataFrame )
    PM = np.dot(AM,CRM)
    PM_DataFrame = pd.DataFrame(data=PM)

    print("PM Matrix for Data Points \n")
    print(f"{PM_DataFrame.to_string(header=None,index=False)}")

    # Calculate Error

    Error_Threshold = 0
    for i in range(PATTERNS):
        for j in range(FEATURES):
            #print(f"{pl.iloc[i,j] } { PM_DataFrame.iloc[i,j]}")
            Error_Threshold = Error_Threshold + np.power(( pl.iloc[i,j] - PM_DataFrame.iloc[i,j]),2)

    # print("Error_Threshold",np.sqrt(Error_Threshold))
    Error_Threshold_List.append(Error_Threshold)
    return (Error_Threshold)


data = [[1,1],[3,3],[2,2],[3,4]]
data = [[2,2],[1,1],[3,3],[3,4]]
data = [[1, 2, 2],[2, 1 ,3],[3, 3 ,1],[6 ,1 ,1],[6 ,2, 3],[6 ,3 ,2],[6, 6 ,6],[6 ,6 ,7],[6, 6, 8]]
```

```
DM = pd.DataFrame(data)

print('DM Matrix for Data Points ')
print(f"{DM.to_string(header=None,index=False)}")
print('\n')
Threshhold = 3
Error_Threshold = process_clustering(DM, Threshhold)
print( f"Error ==>  { Error_Threshold } for Threshold {Threshhold}")
```

```
DM Matrix for Data Points
 1  2  2
 2  1  3
 3  3  1
 6  1  1
 6  2  3
 6  3  2
 6  6  6
 6  6  7
 6  6  8


centroids = [[6. 6. 7.]
 [2. 2. 2.]
 [6. 2. 2.]] labels  [1 1 1 2 2 2 0 0 0]
CRM Matrix for Data Points

 6.0  6.0  7.0
 2.0  2.0  2.0
 6.0  2.0  2.0
AM Matrix for Data Points

 0.0  1.0  0.0
 0.0  1.0  0.0
 0.0  1.0  0.0
 0.0  0.0  1.0
 0.0  0.0  1.0
 0.0  0.0  1.0
 1.0  0.0  0.0
 1.0  0.0  0.0
 1.0  0.0  0.0
PM Matrix for Data Points

 2.0  2.0  2.0
 2.0  2.0  2.0
 2.0  2.0  2.0
 6.0  2.0  2.0
 6.0  2.0  2.0
 6.0  2.0  2.0
 6.0  6.0  7.0
 6.0  6.0  7.0
 6.0  6.0  7.0
Error ==>  12.0 for Threshold 3
```

Centroids is the mean in the clusters. Labels - Cluster Index - Example: labels [0 0 0 2 2 2 1 1 1]

DP 1 mapped to cluster 0 DP 2 mapped to cluster 1 DP 8 mapped to cluster 1

Run 1: centroids = [[6. 2. 2.] [6. 6. 7.] [2. 2. 2.]] labels [2 2 2 0 0 0 1 1 1] CRM Matrix for Data Points

6.0 2.0 2.0 6.0 6.0 7.0 2.0 2.0 2.0 AM Matrix for Data Points

0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 PM Matrix for Data Points

2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 6.0 2.0 2.0 6.0 2.0 2.0 6.0 2.0 2.0 6.0 6.0 7.0 6.0 6.0 7.0 6.0 6.0 7.0 Error ==> 12.0 for Threshold 3

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: