David Luong
Professor Shilpa Lakhanpal
CPSC 483
December 9, 2019

# K means Project

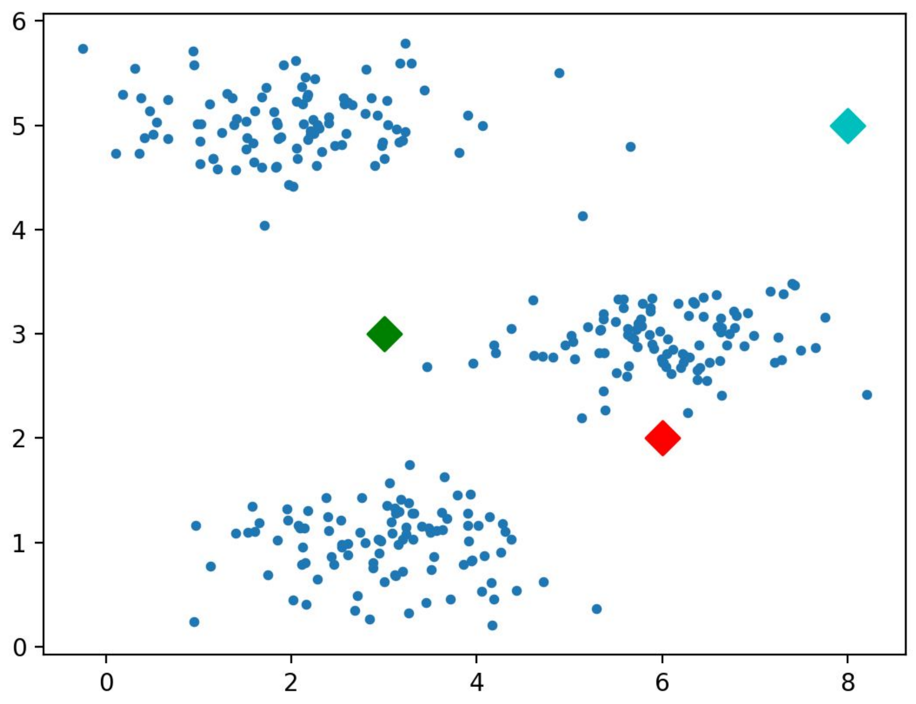
1. Code:

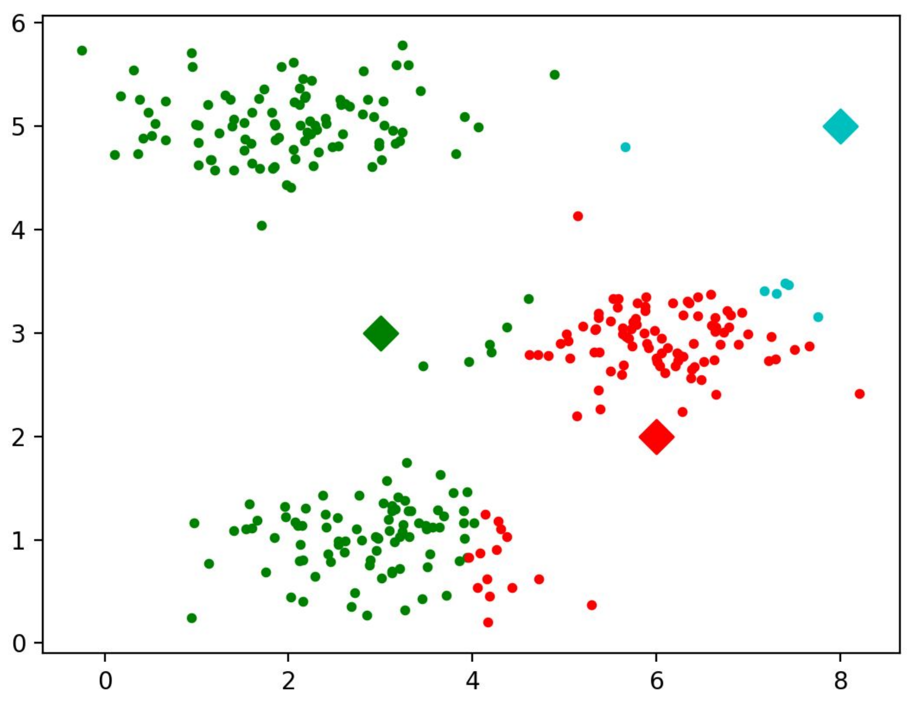
```
import numpy as np
import matplotlib.pyplot as plt
import math
import scipy.io

def dist(x1, x2):
   return math.sqrt((x1[0] - x2[0])**2 + (x1[1] - x2[1])**2)

def compare(cent, newCent): #Uses lists of two per point
   for i in range(len(cent)):
       if not (cent[i][0] == newCent[i][0] and cent[i][1] == newCent[i][1]):
           return False
   return True

def k_means (data, cent, graph):
   num_points = len(data)
   k = len(cent)
   matrix = np.zeros((k,num_points))
   smallest = np.zeros(num_points, dtype=np.int8)

   for i in range(num_points): #for each point
       for j in range(k): #for each Cent
           matrix[j][i] = dist(data[i], cent[j])
           if matrix[smallest[i]][i] > matrix[j][i]: #Keeps track of smallest
               smallest[i] = j

   if False: #Print
       print("Distance Matrix: ")
       print (matrix)

   k_count = np.zeros(k)
   centNew  = np.zeros((k, 2),dtype = float)
```

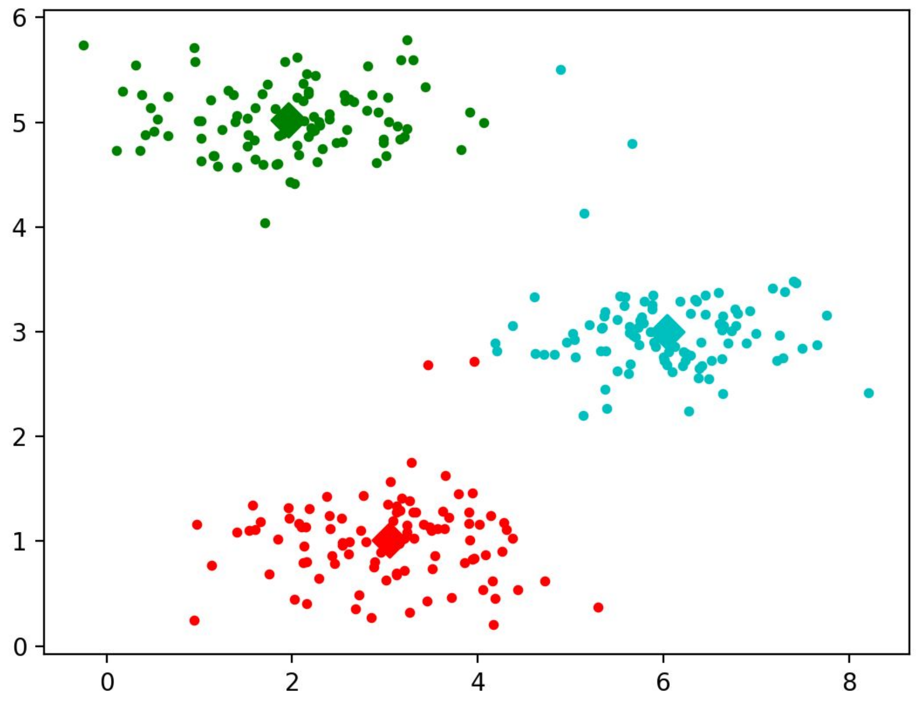
```
    for i in range(num_points): #add up all points for each smallest dist K
point
        centNew[smallest[i]] += data[i]
        k_count[smallest[i]] += 1.0

    for i in range(k): #Take avg
        centNew[i][0] /= k_count[i]
        centNew[i][1] /= k_count[i]

    if graph: #PLOT
        points_x = []
        points_y = []
        for i in range(k): #Create [[], [], []]
            points_x.append([])
            points_y.append([])

        for i in range(num_points): #Load points belonging to each cent
            points_x[smallest[i]].append(data[i][0])
            points_y[smallest[i]].append(data[i][1])

        colors = ['g', 'r', 'c', 'm', 'y', 'k', 'w']
        for i in range(k):#uses init cent not updated ones
            plt.scatter(points_x[i], points_y[i], color=colors[i], s = 10)
            plt.scatter(cent[i][0], cent[i][1], color = colors[i], marker='D',
s=100)

        plt.show()

    print("New cents")
    print(centNew)
    return(centNew)

#MAIN:
mat = scipy.io.loadmat('kmeansdata.mat')
data = mat['X']
if False:
    print("Data in X:")
    print(data)
```

```
   print("Dim of X: ", data.shape)
   print("Num points: ", len(data))
cent = np.array([[3, 3], [6,2], [8,5]], dtype=float)

#Plot 1
x,y = zip(*data)
plt.scatter(x, y, s = 10)
colors = ['g', 'r', 'c', 'm', 'y', 'k', 'w']
for i in range(len(cent)):
   plt.scatter(cent[i][0], cent[i][1], color = colors[i], marker='D', s=100)
plt.show()

#Special cases for plots 2, 3
cent = k_means(data, cent, True)
for i in range(1, 9):
   cent = k_means(data, cent, False)

cent = k_means(data, cent, True)
```

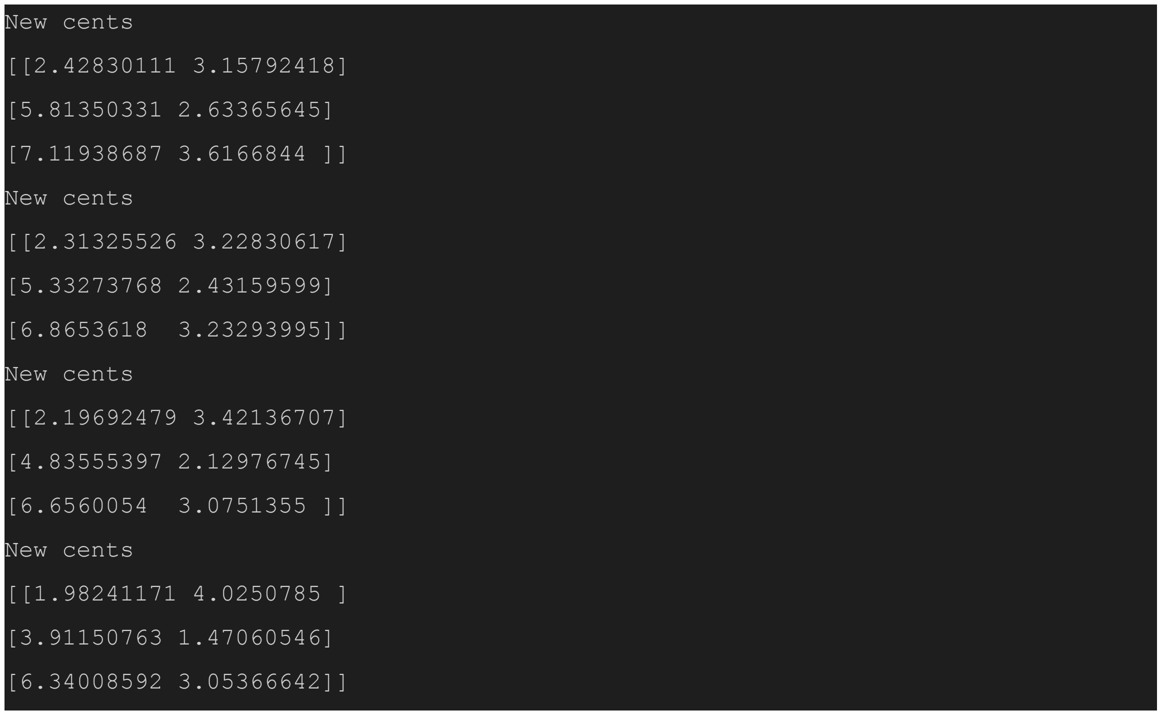
2. Plots:


0
1
2
3
4
5
6
0
2
4
6
8


0
1
2
3
4
5
6
0
2
4
6
8



3.



```
New cents
[[2.42830111 3.15792418]
[5.81350331 2.63365645]
[7.11938687 3.6166844 ]]
New cents
[[2.31325526 3.22830617]
[5.33273768 2.43159599]
[6.8653618  3.23293995]]
New cents
[[2.19692479 3.42136707]
[4.83555397 2.12976745]
[6.6560054  3.0751355 ]]
New cents
[[1.98241171 4.0250785 ]
[3.91150763 1.47060546]
[6.34008592 3.05366642]]
```

```
New cents
[[1.95399466 5.02557006]
[3.12663743 1.1121712 ]
[6.12919526 3.01606258]]
New cents
[[1.95399466 5.02557006]
[3.04367119 1.01541041]
[6.03366736 3.00052511]]
New cents
[[1.95399466 5.02557006]
[3.04367119 1.01541041]
[6.03366736 3.00052511]]
New cents
[[1.95399466 5.02557006]
[3.04367119 1.01541041]
[6.03366736 3.00052511]]
New cents
[[1.95399466 5.02557006]
[3.04367119 1.01541041]
[6.03366736 3.00052511]]
New cents
[[1.95399466 5.02557006]
[3.04367119 1.01541041]
[6.03366736 3.00052511]]
```