**K-Means Clustering**

In [1]: import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

**Creating DataFrame for analysis**

In [2]: #-------------------------------Creating Data Frame-------------

----------------------

df = pd.DataFrame({

'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53

, 55, 61, 64, 69, 72,51,12,22,36,45,65,15,19,31,81,26],

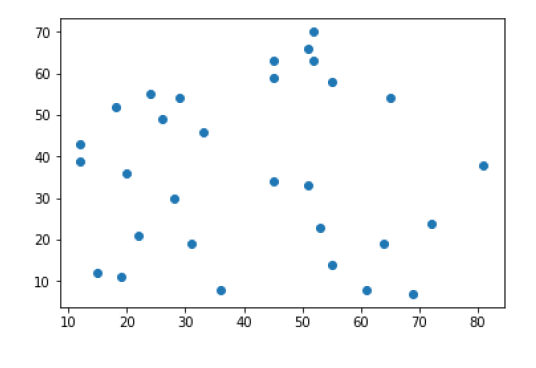
'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23

, 14, 8, 19, 7, 24,33,43,21,8,34,54,12,11,19,38,49]

})

plt.scatter(df['x'],df['y'])

Out[2]: <matplotlib.collections.PathCollection at 0xe465882128>



**Creating Centroid**

In [4]: #--------------------------Creating & Assigning Centroid--------

----------------------

np.random.seed(200)

k = 3

# centroids[i] = [x, y]

centroids = {

i+1: [np.random.randint(0, 80), np.random.randint(0, 80)]

for i in range(k)

}

#SELF

print(centroids)

#list1=[]

#list1=centroids[1]

#list2=centroids[2]

#list3=centroids[3]

#plt.figure()

#a=list1[0]

#b=list1[1]

#plt.plot(a,b,'\*r',0.6)

#plt.figure()

#a=list2[0]

#b=list2[1]

#plt.plot(a,b,'\*m',0.6)

#plt.figure()

#a=list3[0]

#b=list3[1]

#plt.plot(a,b,'\*c',0.6)

{1: [26, 16], 2: [68, 42], 3: [55, 76]}

**Plotting of points(seed)**

In [32]: fig = plt.figure(figsize=(5, 5))

plt.scatter(df['x'], df['y'], color='k')

colmap = {1: 'r', 2: 'g', 3: 'b'}

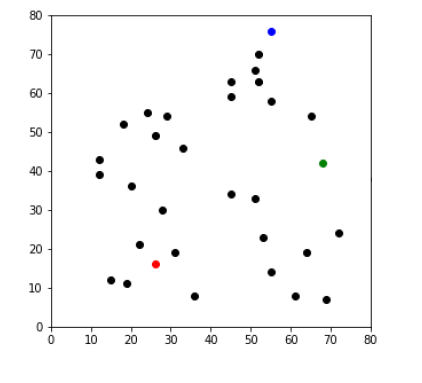
for i in centroids.keys():

plt.scatter(\*centroids[i], color=colmap[i])

plt.xlim(0, 80)

plt.ylim(0, 80)

plt.show()



**Centroid Assignment Stage(Euclidian Distance)**

In [33]: def assignment(df, centroids):

for i in centroids.keys():

# sqrt((x1 - x2)^2 - (y1 - y2)^2)

df['distance\_from\_{}'.format(i)] = (

np.sqrt(

(df['x'] - centroids[i][0]) \*\* 2

+ (df['y'] - centroids[i][1]) \*\* 2

)

)

centroid\_distance\_cols = ['distance\_from\_{}'.format(i) for i

in centroids.keys()]

df['closest'] = df.loc[:, centroid\_distance\_cols].idxmin(axi

s=1)

df['closest'] = df['closest'].map(lambda x: int(x.lstrip('di

stance\_from\_')))

df['color'] = df['closest'].map(lambda x: colmap[x])

return df

df = assignment(df, centroids)

print(df.head(40))

x y distance\_from\_1 distance\_from\_2 distance\_from\_3

closest color

0 12 39 26.925824 56.080300 56.727418

1 r

1 20 36 20.880613 48.373546 53.150729

1 r

2 28 30 14.142136 41.761226 53.338541

1 r

3 18 52 36.878178 50.990195 44.102154

1 r

4 29 54 38.118237 40.804412 34.058773

3 b

5 33 46 30.805844 35.227830 37.202150

1 r

6 24 55 39.051248 45.880279 37.443290

3 b

7 45 59 47.010637 28.600699 19.723083

3 b

8 45 63 50.695167 31.144823 16.401219

3 b

9 52 70 59.933296 32.249031 6.708204

3 b

10 51 66 55.901699 29.410882 10.770330

3 b

11 52 63 53.712196 26.400758 13.341664

3 b

12 55 58 51.039201 20.615528 18.000000

3 b

13 53 23 27.892651 24.207437 53.037722

2 g

14 55 14 29.068884 30.870698 62.000000

1 r

15 61 8 35.902646 34.713110 68.264193

2 g

16 64 19 38.118237 23.345235 57.706152

2 g

17 69 7 43.931765 35.014283 70.405966

2 g

18 72 24 46.690470 18.439089 54.708317

2 g

19 51 33 30.232433 19.235384 43.185646

2 g

20 12 43 30.413813 56.008928 54.203321

1 r

21 22 21 6.403124 50.566788 64.140471

1 r

22 36 8 12.806248 46.690470 70.604532

1 r

23 45 34 26.172505 24.351591 43.174066

2 g

24 65 54 54.451814 12.369317 24.166092

2 g

25 15 12 11.704700 60.901560 75.471849

1 r

26 19 11 8.602325 57.982756 74.303432

1 r

27 31 19 5.830952 43.566042 61.846584

1 r

28 81 38 59.236813 13.601471 46.043458

2 g

29 26 49 33.000000 42.579338 39.623226

1 r

**Plotting Seeds with Centroid**

In [34]: fig = plt.figure(figsize=(5, 5))

plt.scatter(df['x'], df['y'], color=df['color'], alpha=0.5, edge

color='k')

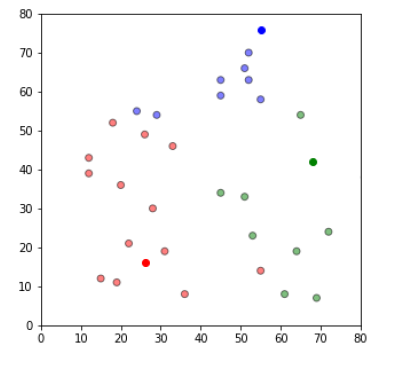
for i in centroids.keys():

plt.scatter(\*centroids[i], color=colmap[i])

plt.xlim(0, 80)

plt.ylim(0, 80)

plt.show()



**Update Stage**

In [35]: import copy

old\_centroids = copy.deepcopy(centroids)

def update(k):

for i in centroids.keys():

centroids[i][0] = np.mean(df[df['closest'] == i]['x'])

centroids[i][1] = np.mean(df[df['closest'] == i]['y'])

return k

centroids = update(centroids)

**Plot updated seeds in cluster form**

In [36]: fig = plt.figure(figsize=(5, 5))

ax = plt.axes()

plt.scatter(df['x'], df['y'], color=df['color'], alpha=0.5, edge

color='k')

for i in centroids.keys():

plt.scatter(\*centroids[i], color=colmap[i])

plt.xlim(0, 80)

plt.ylim(0, 80)

for i in old\_centroids.keys():

old\_x = old\_centroids[i][0]

old\_y = old\_centroids[i][1]

dx = (centroids[i][0] - old\_centroids[i][0]) \* 0.75

dy = (centroids[i][1] - old\_centroids[i][1]) \* 0.75

ax.arrow(old\_x, old\_y, dx, dy, head\_width=2, head\_length=3,

fc=colmap[i], ec=colmap[i])

plt.show()

