#Lab 4 : CLUSTERING Part 1

In this Lab you will have to write code for 2 clustering algorithms based on the mathematical theory :

1. K-means Clustering
2. Gaussian Mixture Model

You will then have to use these algorithms on a pratical dataset and compare the results with the inbuilt algorithms present in scikit learn toolkit

Please use plots wherever possible to demonstrate the results

```
import numpy as np
import matplotlib.pyplot as plt
```

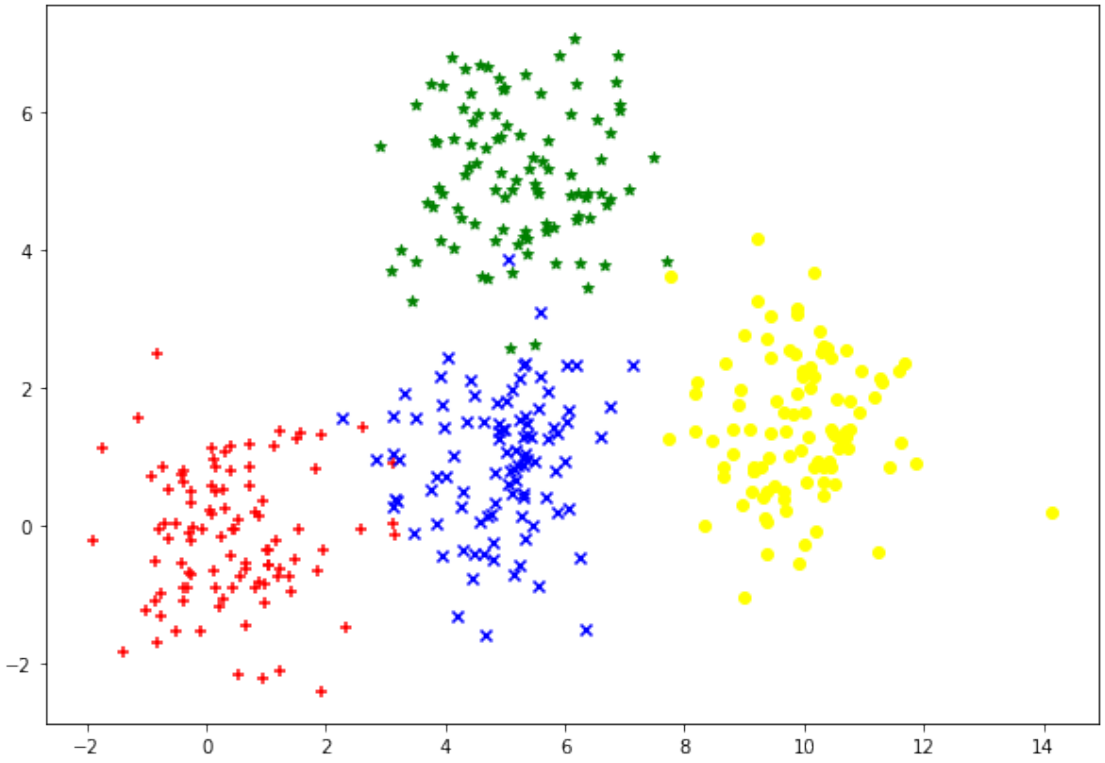
K-means Clustering

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided.

Step 1 : Data Generation

Generate 2D gaussian data of 4 types each having 100 points, by taking appropriate mean and varince (example: mean :(0.5 0) (5 5) (5 1) (10 1.5), variance : Identity matrix)

```
# write your code here
K = 4

data = []
means = [(0.5, 0), (5, 5), (5, 1), (10, 1.5)]
color_style = [('red', '+'), ('green', '*'), ('blue', 'x'), ('yellow',
'o')]

plt.figure(figsize=(10, 7))

for i in range(K):
    d = np.random.multivariate_normal(mean=means[i],
cov=np.identity(2), size=100)
    plt.scatter(d[:, 0], d[:, 1], color=color_style[i%4][0],
marker=color_style[i%4][1])
    data.append(d)
```

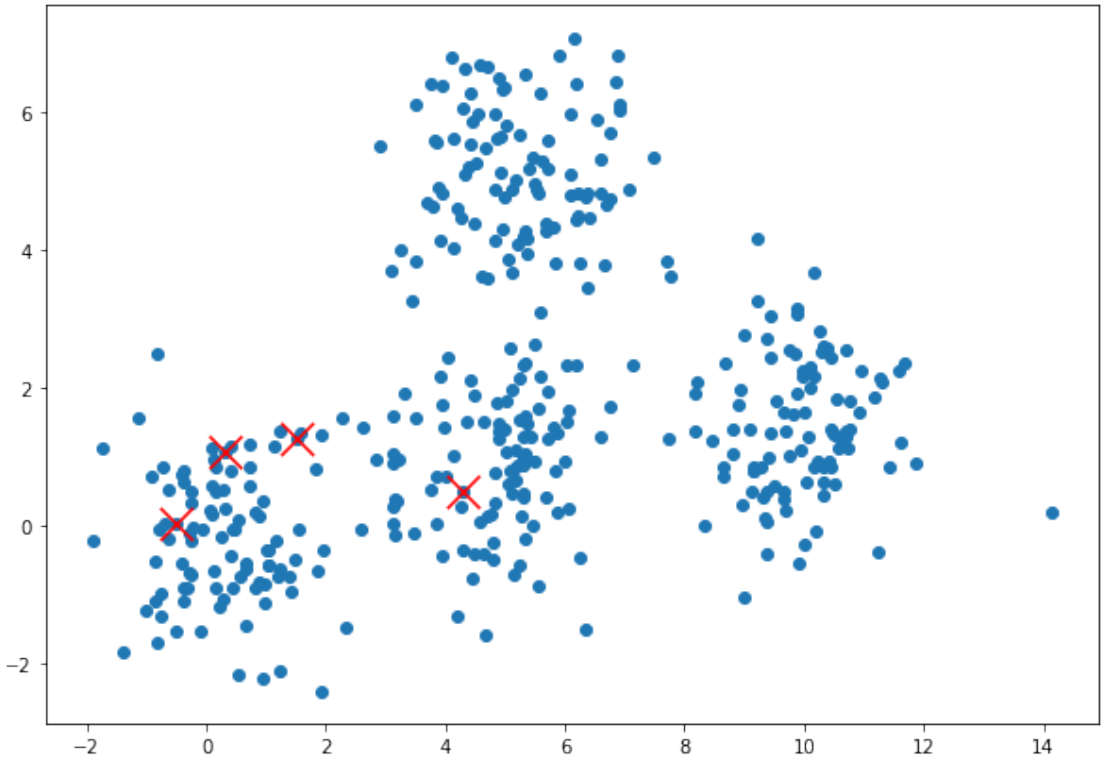


Step 2 : Cluster Initialisation

Initialse K number of Clusters (Here, K=4)

```
data_points = np.concatenate(data, axis=0)
np.random.shuffle(data_points)


plt.figure(figsize=(10, 7))

mean_vectors = data_points[:K, :]

print("Mean Vectors:", mean_vectors)

plt.scatter(data_points[:, 0], data_points[:, 1])
plt.scatter(mean_vectors[:, 0], mean_vectors[:, 1], color='red',
marker='x', s=300)
```

```
Mean Vectors: [[ 4.30122479  0.49284635]
 [-0.5101086   0.042674  ]
 [ 1.5151311   1.26933342]
 [ 0.32761835  1.07878382]]

<matplotlib.collections.PathCollection at 0x1d1ffc23340>
```



Step 3 : Cluster assignment and re-estimation Stage

a) Using initial/estimated cluster centers (mean μ_i) perform cluster assignment.

b) Assigned cluster for each feature vector (X_j) can be written as:

$$arg\min_i ||C_i - X_j||_2, 1 \le i \le K, 1 \le j \le N$$

c) Re-estimation: After cluster assignment, the mean vector is recomputed as,

$$\mu_i = \frac{1}{N_i} \sum_{j \in i^{th} cluster} X_j$$

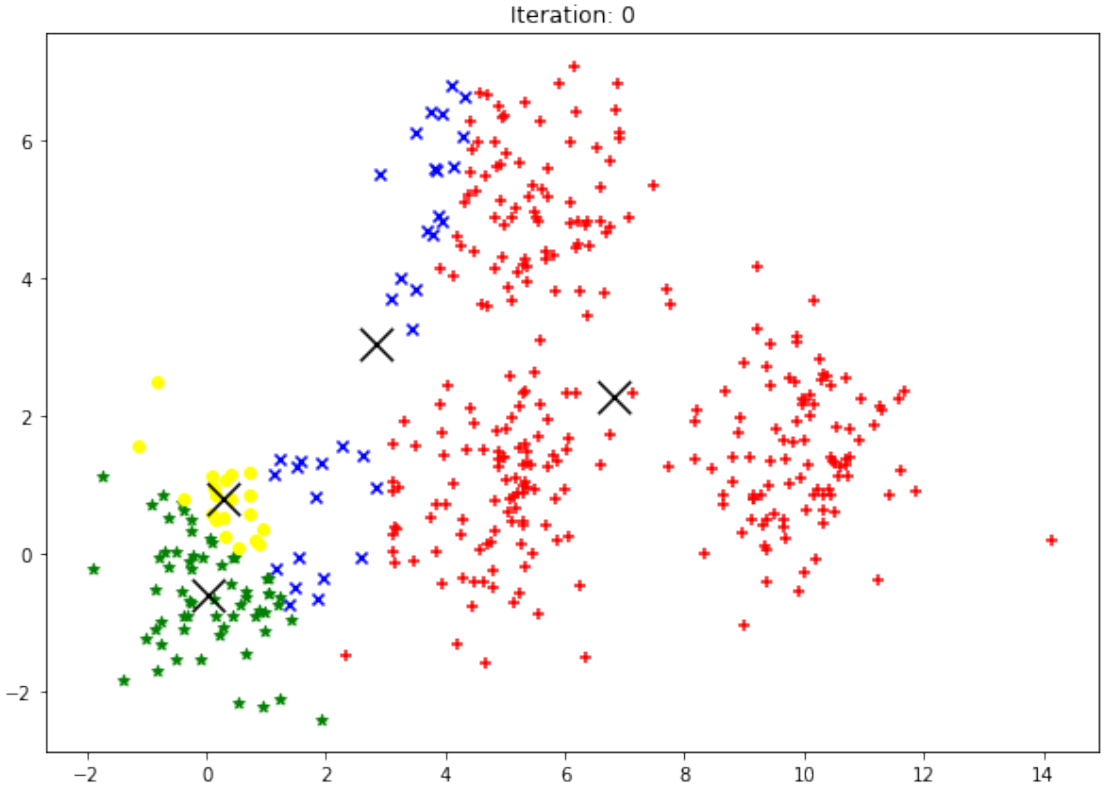
where N_i represents the number of datapoints in the i^{th} cluster.

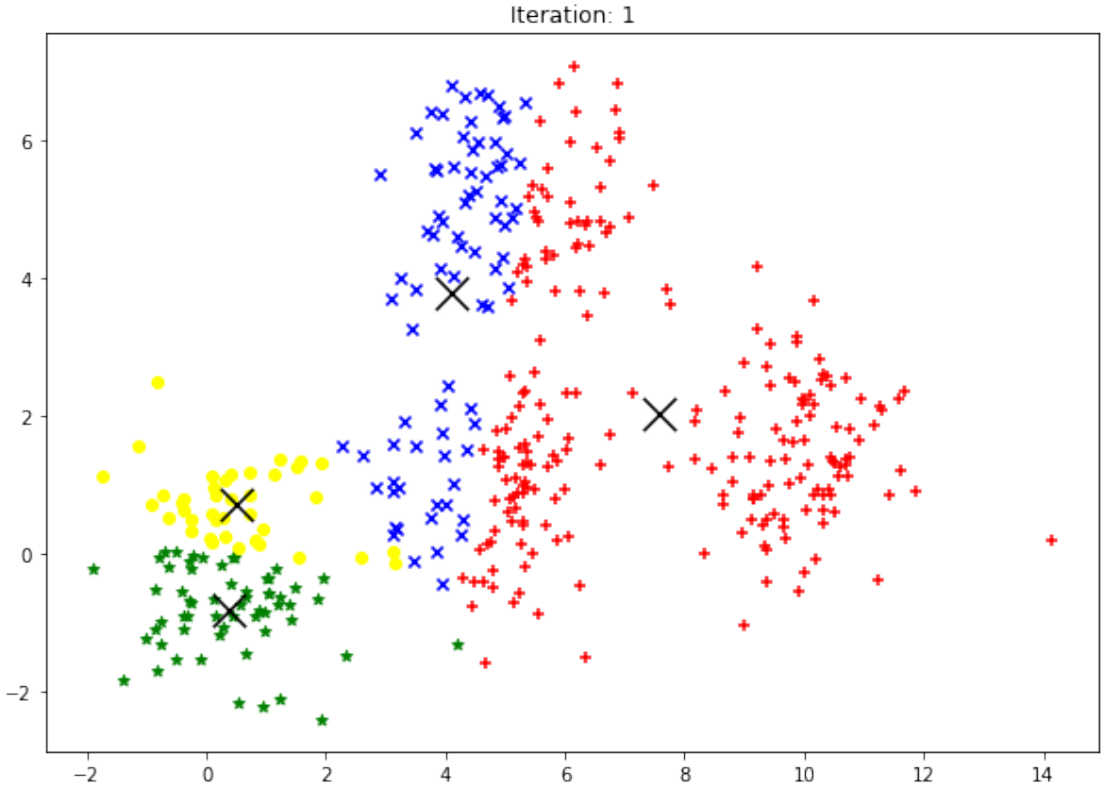
d) Objective function (to be minimized):

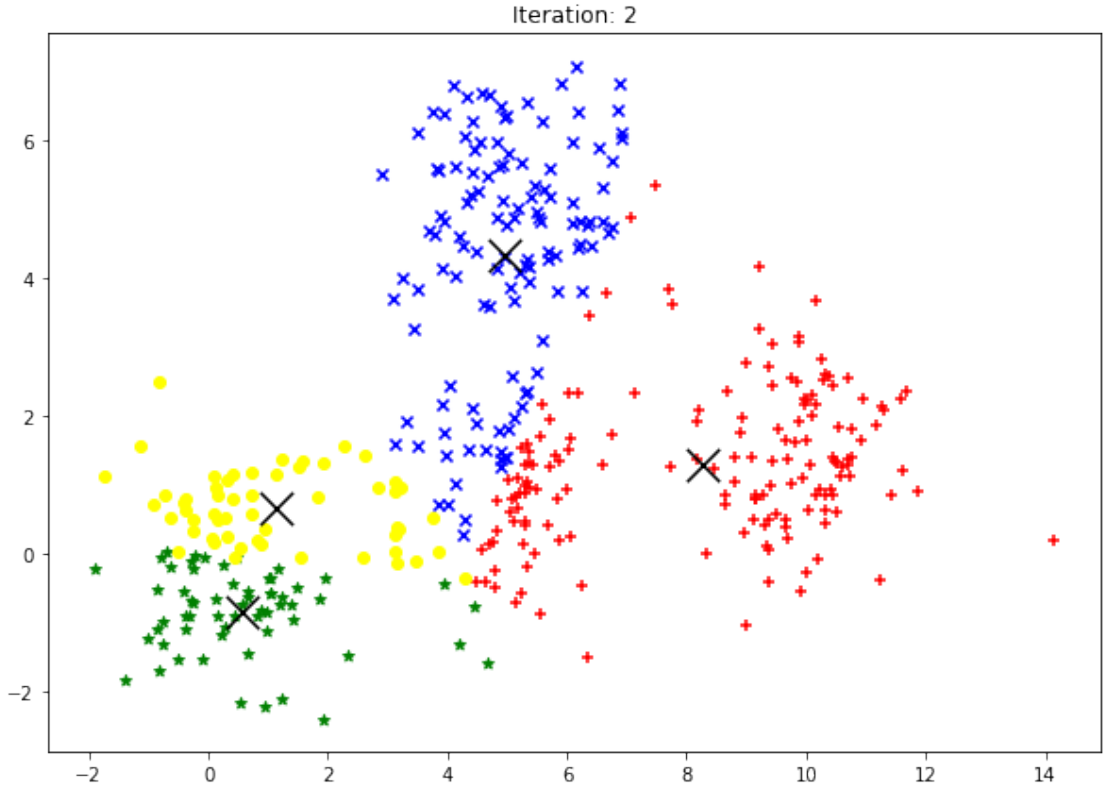
$$Error(\mu) = \frac{1}{N} \sum_{i=1}^{K} \sum_{j \in i^{th} cluster} ||C_i - X_j||_2$$

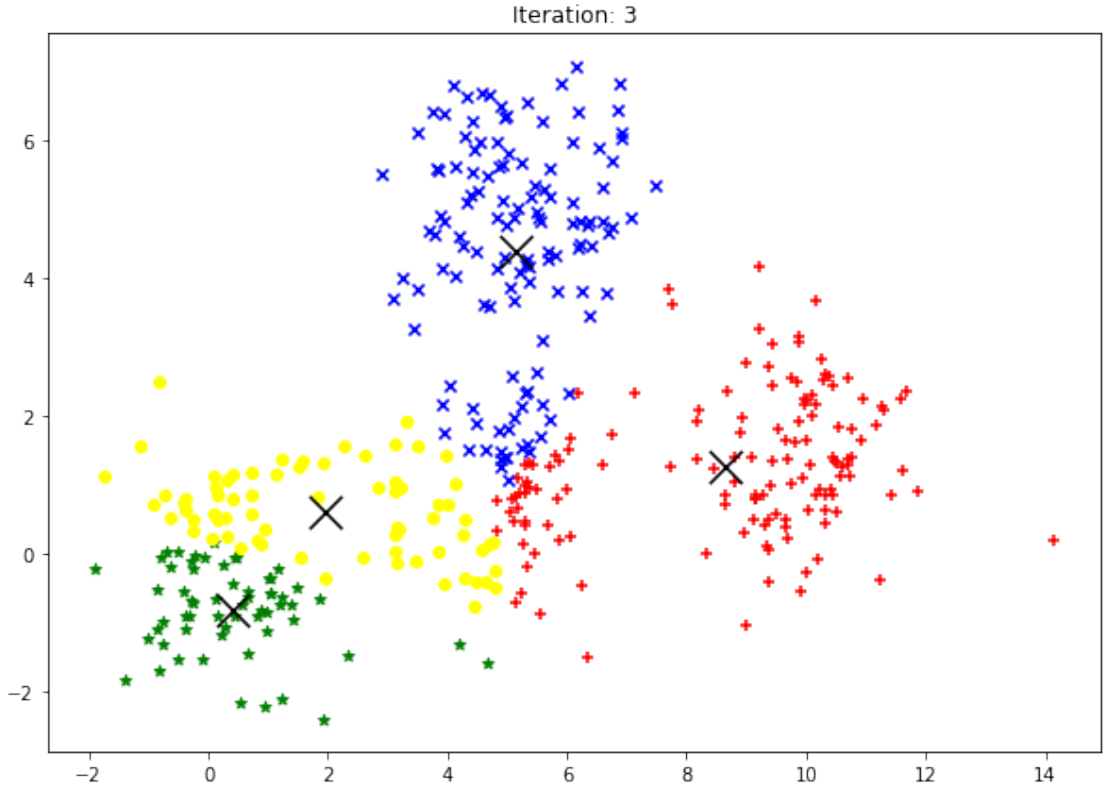
```
tol = 1e-5

delta = float('inf')
prev_error = float('inf')

errors = []
iter = 0
```

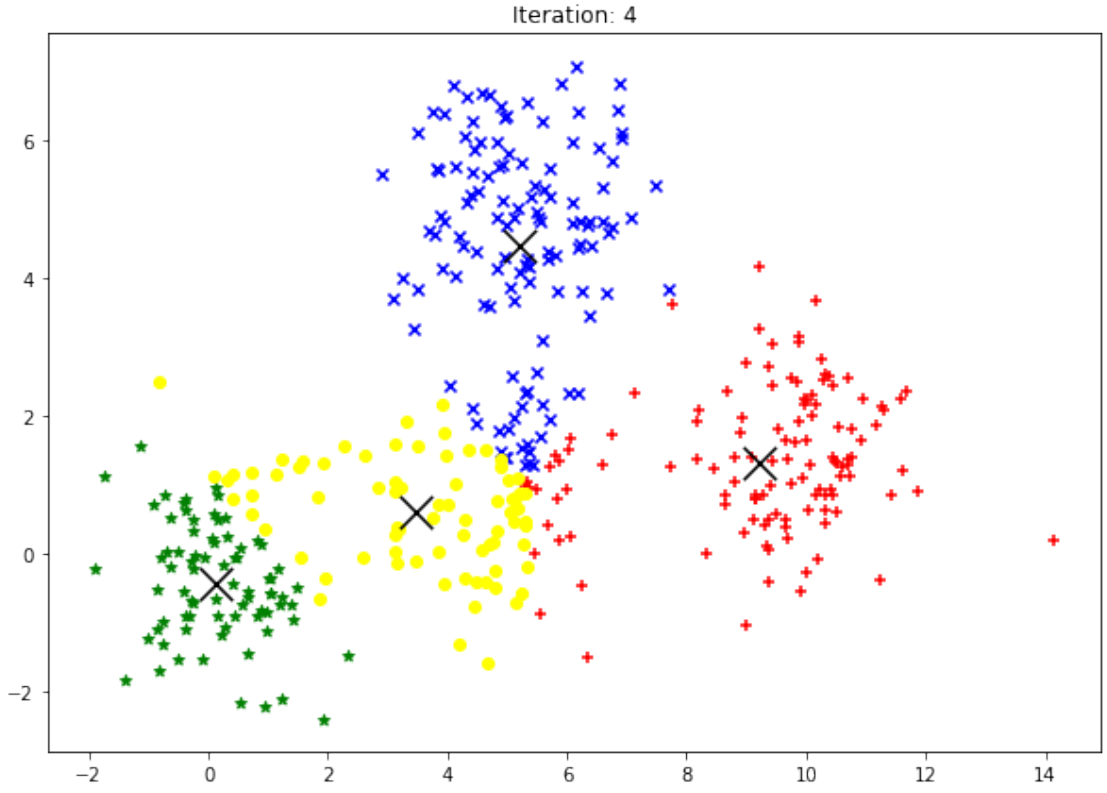
```
while delta>tol:
    assigned = [[] for i in range(K)]
    for i in range(data_points.shape[0]):
        min_index = np.argmin(np.linalg.norm(data_points[i] -
mean_vectors, axis=1))
        assigned[min_index].append(i)

    assigned_clusters = []
    for i in range(K):
assigned_clusters.append(data_points[assigned[i]])

    mean_vectors = np.array([cluster.mean(axis=0) for cluster in
assigned_clusters])

    plt.figure(figsize=(10, 7))

    for i in range(K):
        plt.scatter(assigned_clusters[i][:, 0], assigned_clusters[i]
[:, 1], color=color_style[i%4][0], marker=color_style[i%4][1])

    plt.scatter(mean_vectors[:, 0], mean_vectors[:, 1], color='black',
marker='x', s=300)
    plt.title(f'Iteration: {iter}')

    sum_error = 0

    for i in range(K): sum_error +=
np.sum(np.linalg.norm(assigned_clusters[i]-mean_vectors[i], axis=1))

    sum_error
    delta = abs(prev_error - sum_error/data_points.shape[0])
    prev_error = sum_error/data_points.shape[0]

    prev_error
    errors.append(prev_error)
    iter+=1

plt.figure(figsize=(10, 7))
plt.plot(range(len(errors)), errors)


[<matplotlib.lines.Line2D at 0x1d151e1aa70>]
```

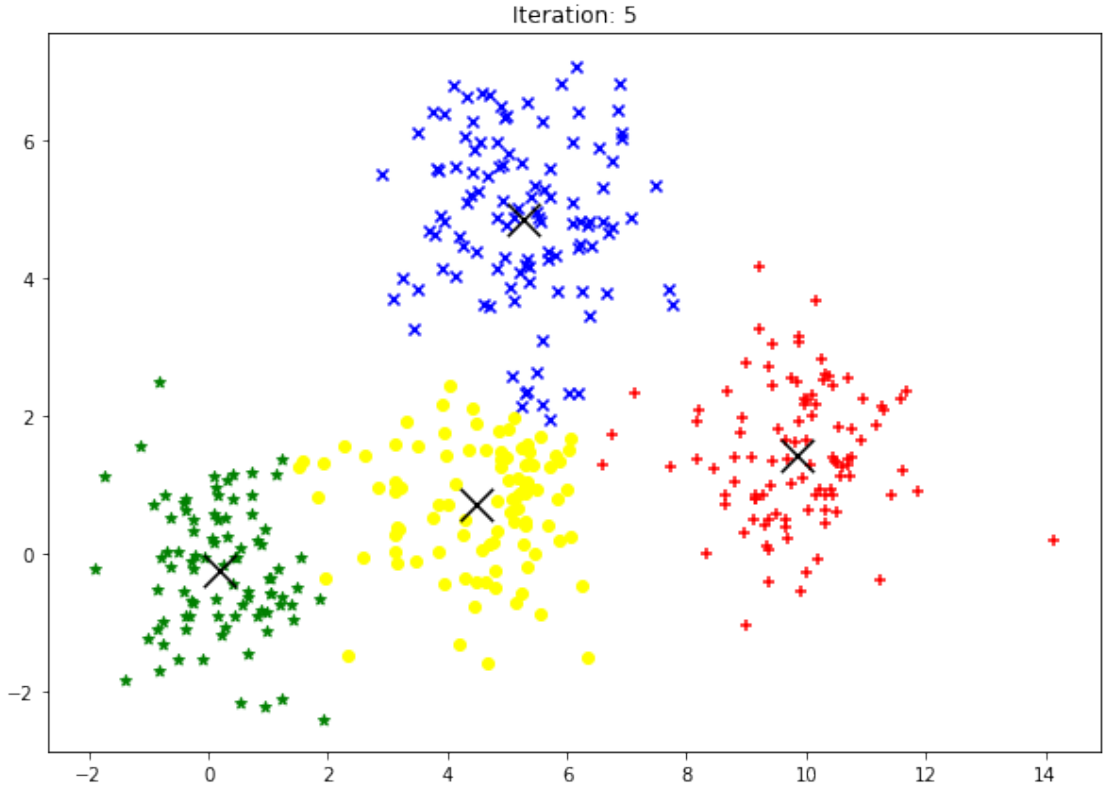

Iteration: 0
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

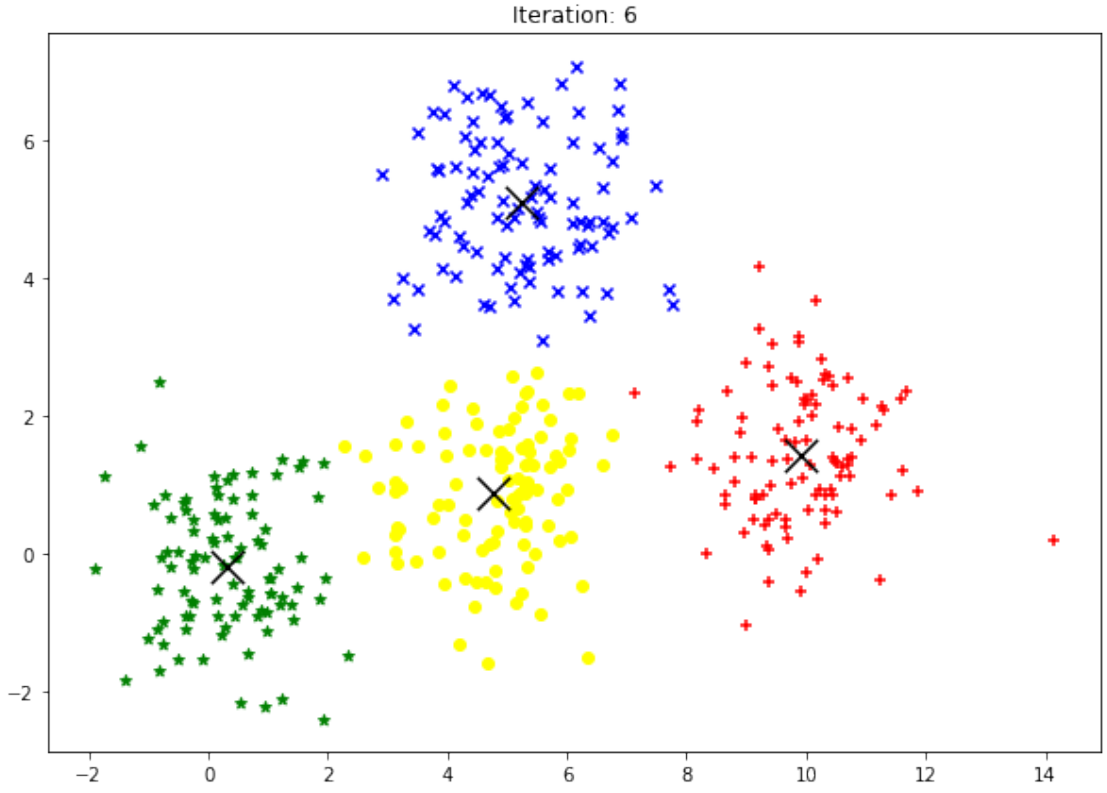

Iteration: 1
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

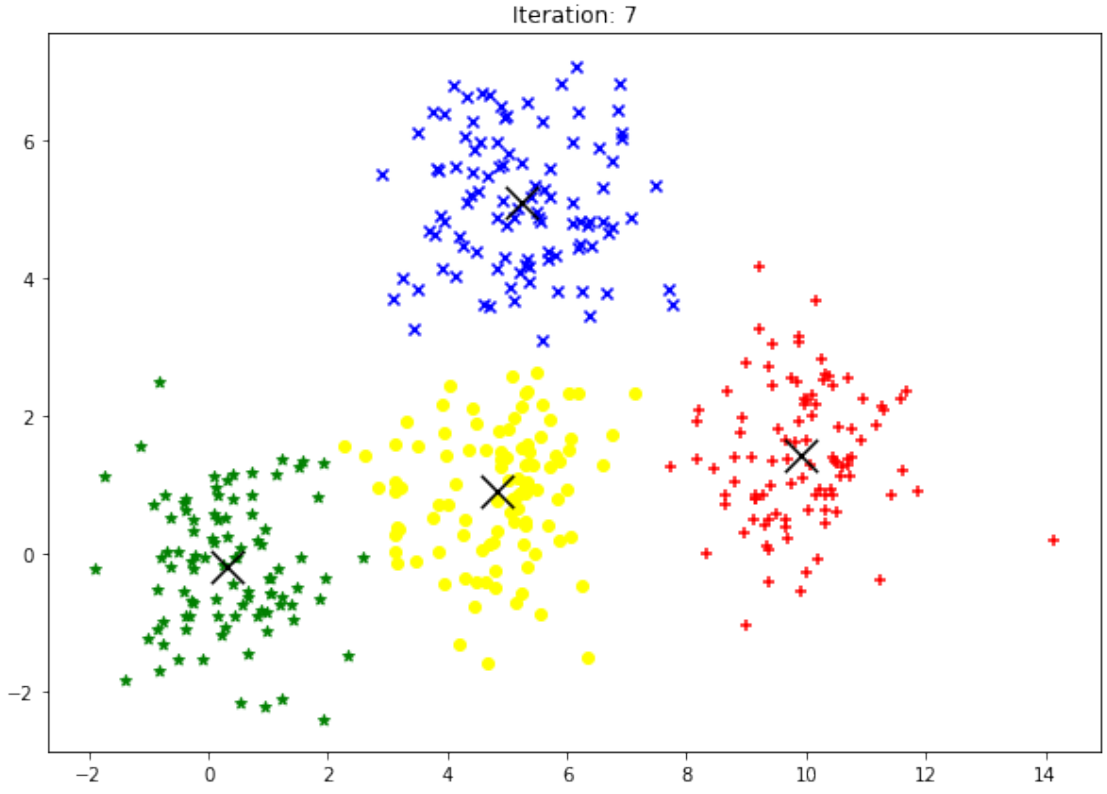

Iteration: 2
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

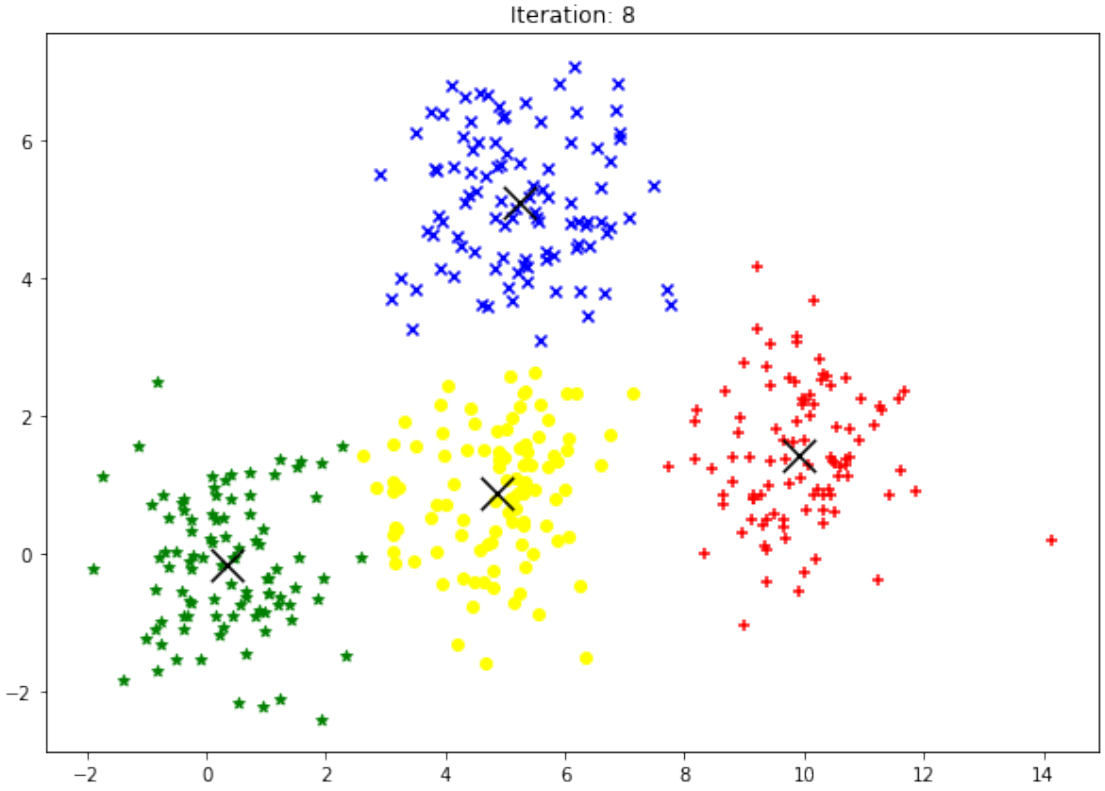

Iteration: 3
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

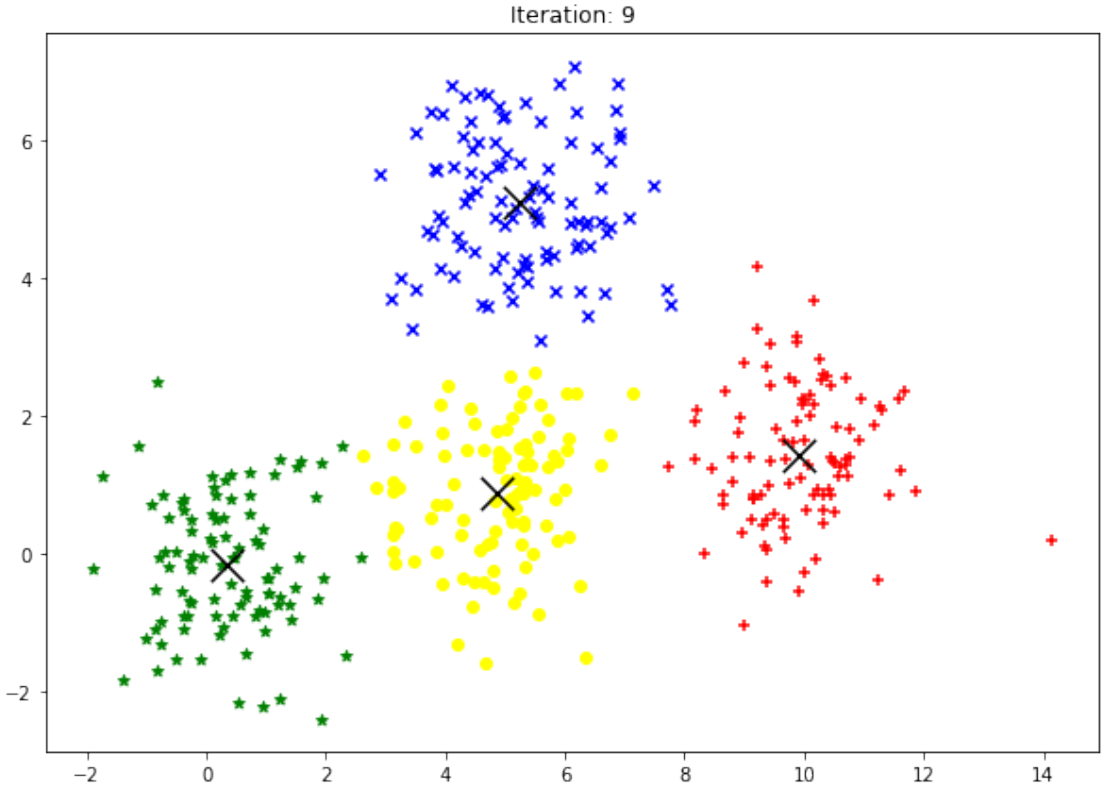

Iteration: 4
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

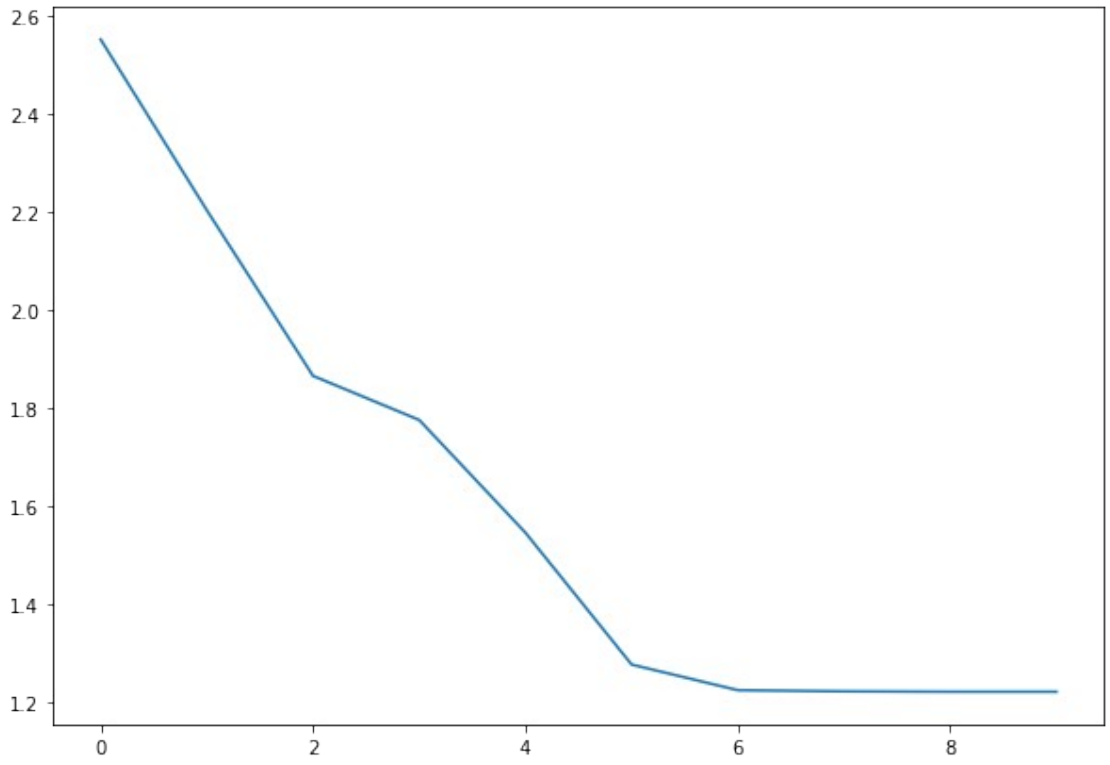

Iteration: 5
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration: 6
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration: 7
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration: 8
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration: 9
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14



Step 4 : Performance metric

Compute Homogeneity score and Silhouette coefficient using the information given below

Homogeneity score : A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. This metric is independent of the absolute values of the labels: a permutation of the class or cluster label values won't change the score value in any way.

Silhouette coeeficient :

$a(x)$: Average distance of x to all other vectors in same cluster

$b(x)$: Average distance of x to the vectors in other clusters. Find minimum among the clusters

$$s(x) = \frac{b(x) - a(x)}{max(a(x), b(x))}$$

Silhouette coefficient (SC) :

$$SC = \frac{1}{N} \sum_{i=1}^{N} s(x)$$

```
from sklearn.metrics.cluster import homogeneity_score,
silhouette_score
```

```
# Homogeneity Score
labels_true = []
labels_predicted = []

for i in range(K):
    for j in range(len(data[i])):
        labels_true.append(i)
        index = np.argmin(np.linalg.norm(data[i][j] - mean_vectors,
axis=1))
        labels_predicted.append(index)

h_score = homogeneity_score(labels_true, labels_predicted)

print(f'Homogeneity score for the classification is: {h_score}')

# Silhouette coefficient
all_points = np.concatenate(assigned_clusters, axis=0)
N = len(all_points)

pairwise = [[0 for i in range(N)] for j in range(N)]
for i in range(N):
    for j in range(N):
        pairwise[i][j] = np.linalg.norm(np.subtract(all_points[i],
all_points[j]))

labels=[]
for id, cluster in enumerate(assigned_clusters):
    labels += [id for i in range(len(cluster))]

sc_score = silhouette_score(pairwise, labels)
print(f'Silhouette coefficient for the classification is: {sc_score}')
```

```
Homogeneity score for the classification is: 0.9140113134183985
Silhouette coefficient for the classification is: 0.6085015371637745
```

Gaussian Mixture Models Clustering

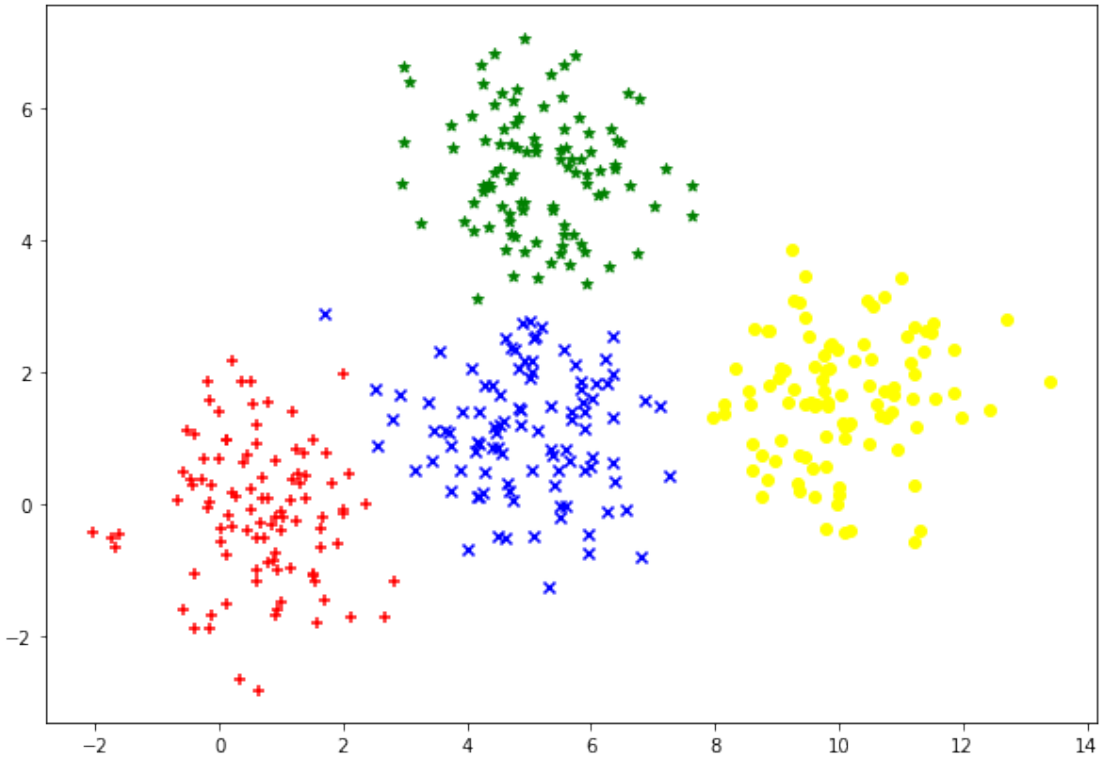
Gaussian mixture model is an unsupervised machine learning method. It summarizes a multivariate probability density function with a mixture of Gaussian probability distributions as its name suggests. It can be used for data clustering and data mining. In this lab, GMM is used for clustering.

Step 1: Data generation

a) Follow the same steps as in K-means Clustering to generate the data

```
# write your code here
K = 4
```

```
data_sep = []
means = [(0.5, 0), (5, 5), (5, 1), (10, 1.5)]
color_style = [('red', '+'), ('green', '*'), ('blue', 'x'), ('yellow',
'o')]

plt.figure(figsize=(10, 7))

for i in range(K):
    d = np.random.multivariate_normal(mean=means[i],
cov=np.identity(2), size=100)
    plt.scatter(d[:, 0], d[:, 1], color=color_style[i%4][0],
marker=color_style[i%4][1])
    data_sep.append(d)

data = np.concatenate(data_sep, axis=0)
```



Step 2. Initialization

a) Mean vector (randomly any from the given data points) (μ_k)

b) Coveriance (initialize with (identity matrix)*max(data)) (Σ_k)

c) Weights (uniformly) (w_k), with constraint: $\sum_{k=1}^{K} w_k = 1$

```
def initialization(data,K):
```

```
  dim = data.shape[1]
  indices = np.random.randint(0, len(data)-1, K)
  mean_vectors = data[indices, :]
  cov_matrix = np.zeros((dim, dim, K))
  for k in range(K):
    cov_matrix[:, :, k] = np.identity(dim)*np.max(data)
  w = np.array([1/K for i in range(K)])

  theta = [mean_vectors, cov_matrix, w]

  return theta
```

Step 3: Expectation stage

$$\gamma_{ik}=\frac{w_k P\left(x_i \vee \Phi_k\right)}{\sum_{k=1}^{K} w_k P\left(x_i \vee \Phi_k\right)}$$

where,

$$\Phi_k=\{\mu_k, \Sigma_k\}$$

$$\theta_k=\{\Phi_k, w_k\}$$

$$w_k=\frac{N_k}{N}$$

$$N_k=\sum_{i=1}^{N} \gamma_{ik}$$

$$P\left(x_i \vee \Phi_k\right)=\frac{1}{\left(2\pi\right)^{d/2}\left|\Sigma_k\right|^{1/2}} e^{-\left(x_i-\mu_k\right)^T \Sigma_k^{-1}\left(x_i-\mu_k\right)}$$

```
# E-Step GMM
from scipy.stats import multivariate_normal

def E_Step_GMM(data,K,theta):

    # write your code here
    mean_vectors = theta[0]
    cov_matrix = theta[1]
    w = theta[2]

    responsibility = np.zeros((len(data), K))
    for i in range(len(data)):
        for k in range(K):
            p_ik = multivariate_normal.pdf(data[i],
mean=mean_vectors[k], cov=cov_matrix[:, :, k])
            numerator_ik = w[k] * p_ik
```

```
            denominator_ik = 0

            for k2 in range(K):
                p_ik_temp = multivariate_normal.pdf(data[i],
mean=mean_vectors[k2], cov=cov_matrix[:, :, k2])
                numerator_ik_temp = w[k2] * p_ik_temp
                denominator_ik += numerator_ik_temp

            responsibility[i][k] = numerator_ik/denominator_ik

    return responsibility
```

Step 4: Maximization stage

a) $w_k = \frac{N_k}{N}$, where $N_k = \sum_{i=1}^{N} \gamma_{ik}$

b) $\mu_k = \frac{\sum_{i=1}^{N} \gamma_{ik} x_i}{N_k}$

c) $\Sigma_k = \frac{\sum_{i=1}^{N} \gamma_{ik} (x_i - \mu_k)(x_i - \mu_k)^T}{N_k}$

Objective function(maximized through iteration):

$$L(\theta) = \sum_{i=1}^{N} log \sum_{k=1}^{K} w_k P(x_i \vee \Phi_k)$$

```
# M-STEP GMM

def M_Step_GMM(data,responsibility):
    N = len(data)
    dim = data.shape[1]
    K = responsibility.shape[1]

    N_k = np.sum(responsibility, axis=0)
    N_k = np.reshape(N_k, (4, 1))
    w = N_k/N

    num_k = np.transpose(responsibility) @ data
    mean_vectors = np.divide(num_k, N_k)

    cov_matrix = np.zeros((dim, dim, K))
    for k in range(K):
        for i in range(N):
            diff = np.reshape((data[i]-mean_vectors[k]), (2, 1))
```

```
            cov_matrix[:, :, k] += responsibility[i][k] * ( diff @
np.transpose(diff) )
        cov_matrix[:, :, k]/=N_k[k]

    theta = [mean_vectors, cov_matrix, w]

    log_likelihood = 0

    for k in range(K):
        inner = 0
        for i in range(N):
            inner += w[k] * multivariate_normal.pdf(data[i],
mean=mean_vectors[k], cov=cov_matrix[:, :, k])
        log_likelihood += np.log(inner)

    return theta, log_likelihood
```

Step 5: Final run (EM algorithm)

a) Initialization

b)Iterate E-M untill $L(\theta_n) - L(\theta_{n-1}) \leq th$

c) Plot and see the cluster allocation at each iteration

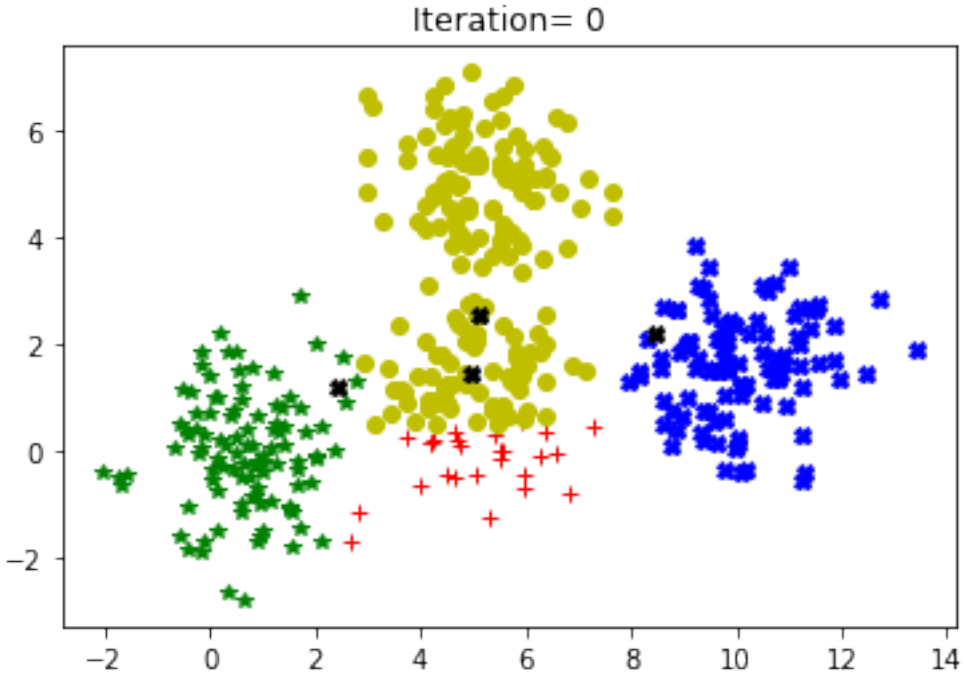
```
log_l=[]
Itr=50
eps=10**(-14)  # for threshold
clr=['r','g','b','y','k','m','c']
mrk=['+','*','X','o','.','<','p']


K = 4   # no. of clusters

theta=initialization(data,K)

for n in range(Itr):

  responsibility=E_Step_GMM(data,K,theta)

  cluster_label=np.argmax(responsibility,axis=1) #Label Points

  theta,log_likhd=M_Step_GMM(data,responsibility)


  log_l.append(log_likhd)

  plt.figure()
  for l in range(K):
    id=np.where(cluster_label==l)
```

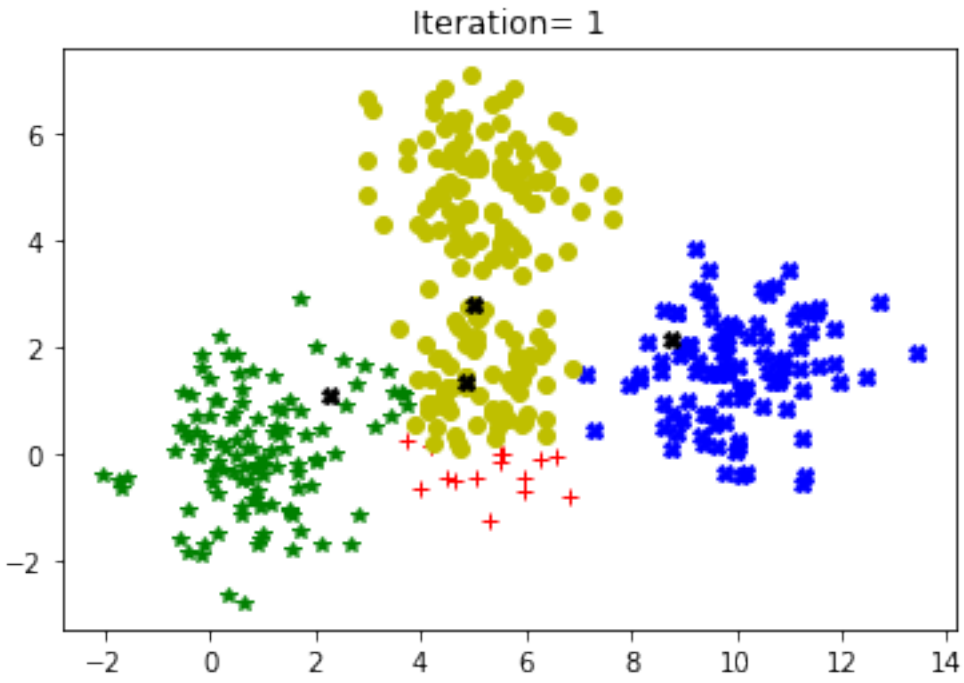
```
    plt.plot(data[id,0],data[id,1],'.',color=clr[l],marker=mrk[l])
  Cents=theta[0]
  plt.plot(Cents[:,0],Cents[:,1],'X',color='k')
  plt.title('Iteration= %d' % (n))

  if n>2:
    if abs(log_l[n]-log_l[n-1])<eps:
      break
```

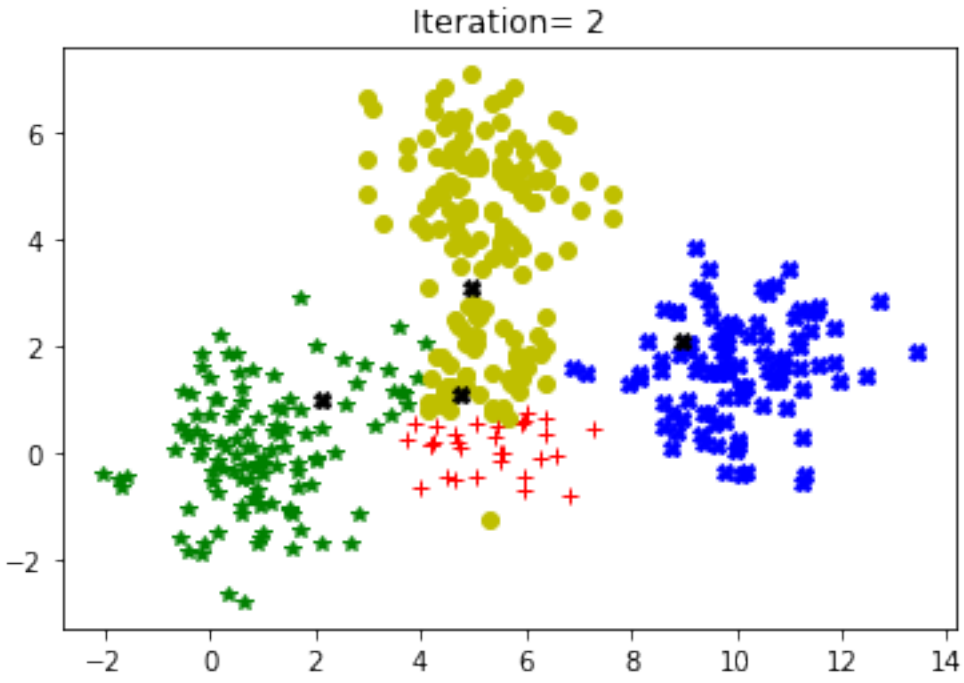
```
plt.figure()
plt.plot(log_l)
```

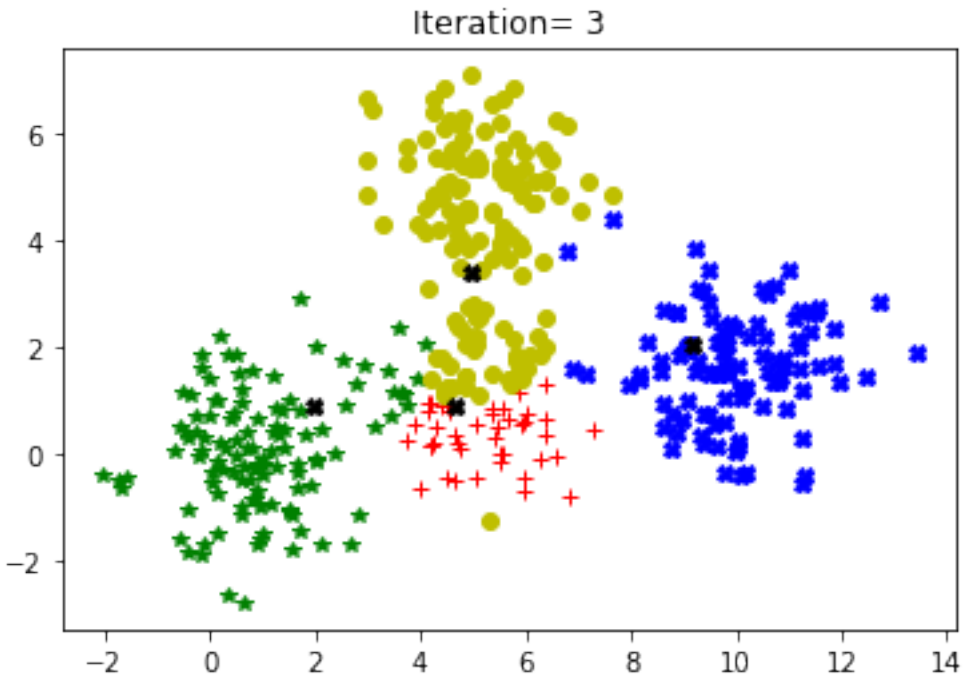
```
C:\Users\Shashank\AppData\Local\Temp\ipykernel_12660\3052499598.py:26:
UserWarning: marker is redundantly defined by the 'marker' keyword
argument and the fmt string "." (-> marker='.'). The keyword argument
will take precedence.
  plt.plot(data[id,0],data[id,1],'.',color=clr[l],marker=mrk[l])
C:\Users\Shashank\AppData\Local\Temp\ipykernel_12660\3052499598.py:23:
RuntimeWarning: More than 20 figures have been opened. Figures created
through the pyplot interface (`matplotlib.pyplot.figure`) are retained
until explicitly closed and may consume too much memory. (To control
this warning, see the rcParam `figure.max_open_warning`).
  plt.figure()
```

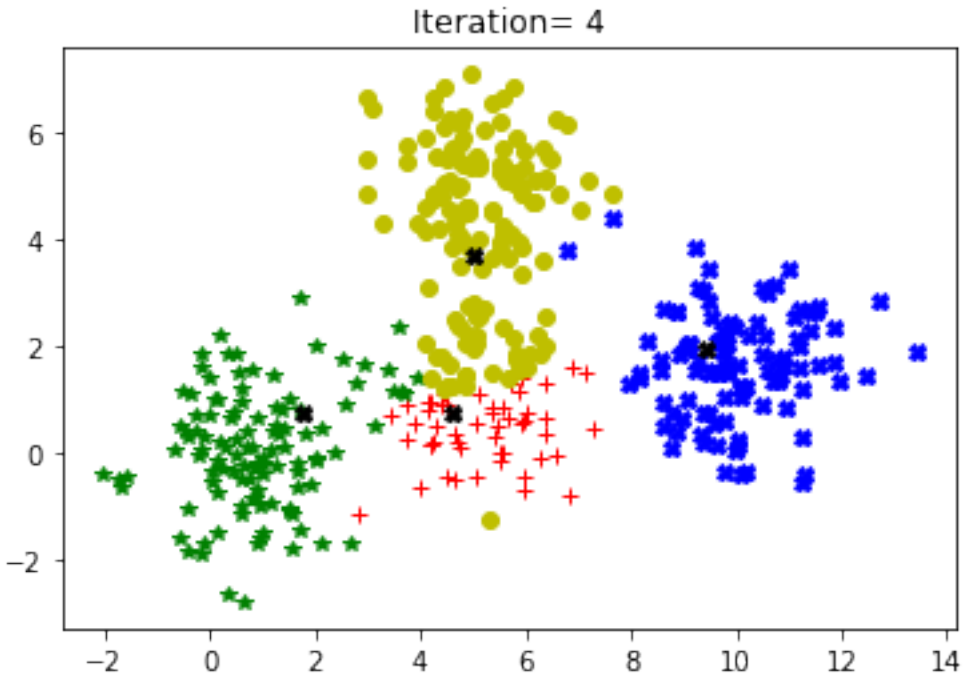
```
[<matplotlib.lines.Line2D at 0x1d12c348e50>]
```

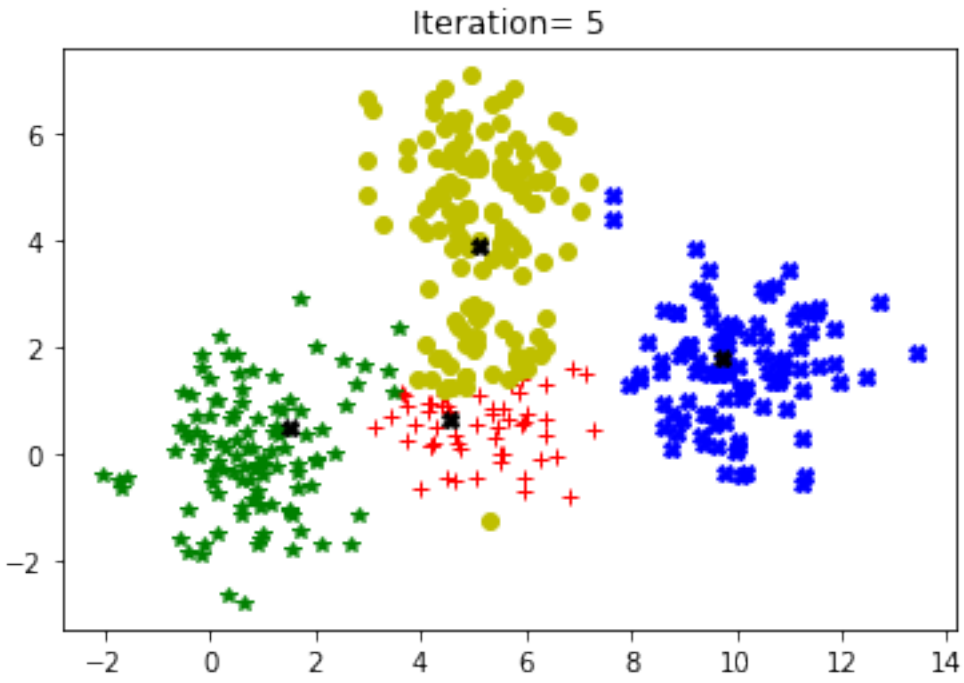


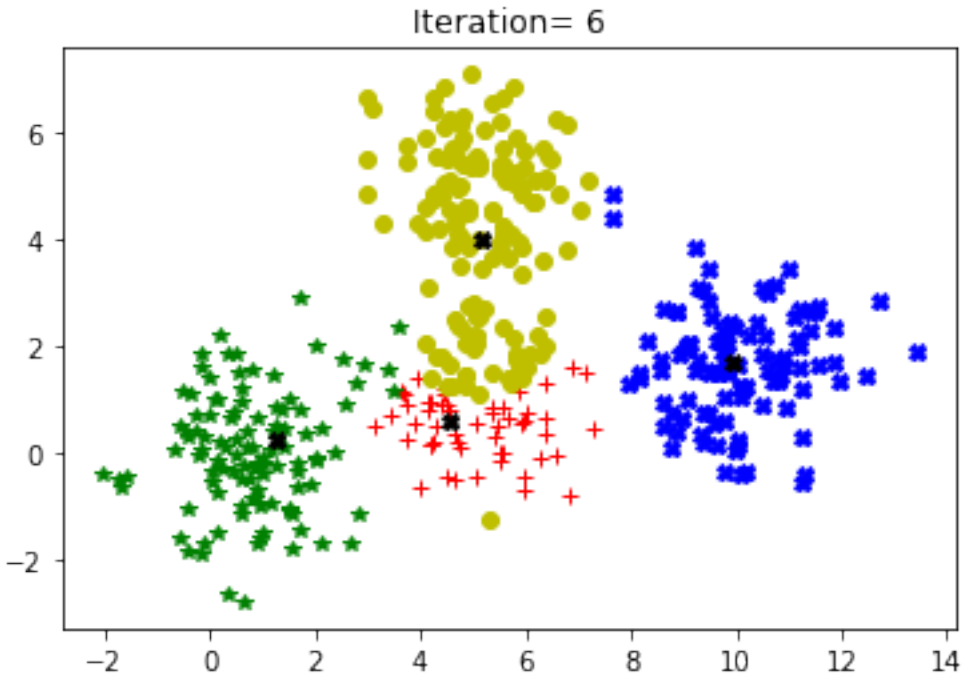

Iteration= 1
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

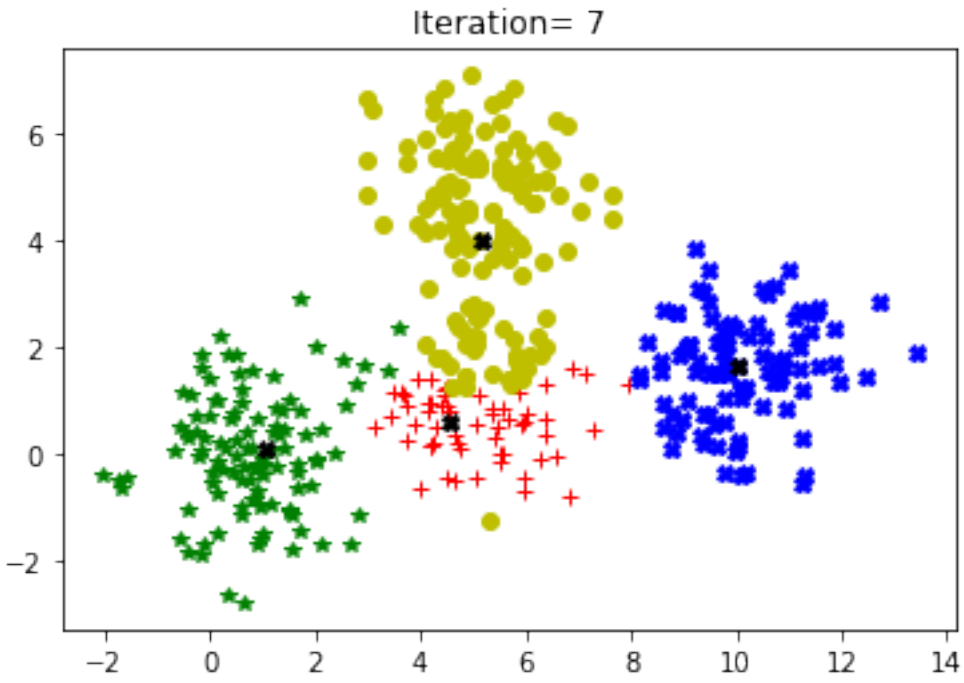

Iteration= 2
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

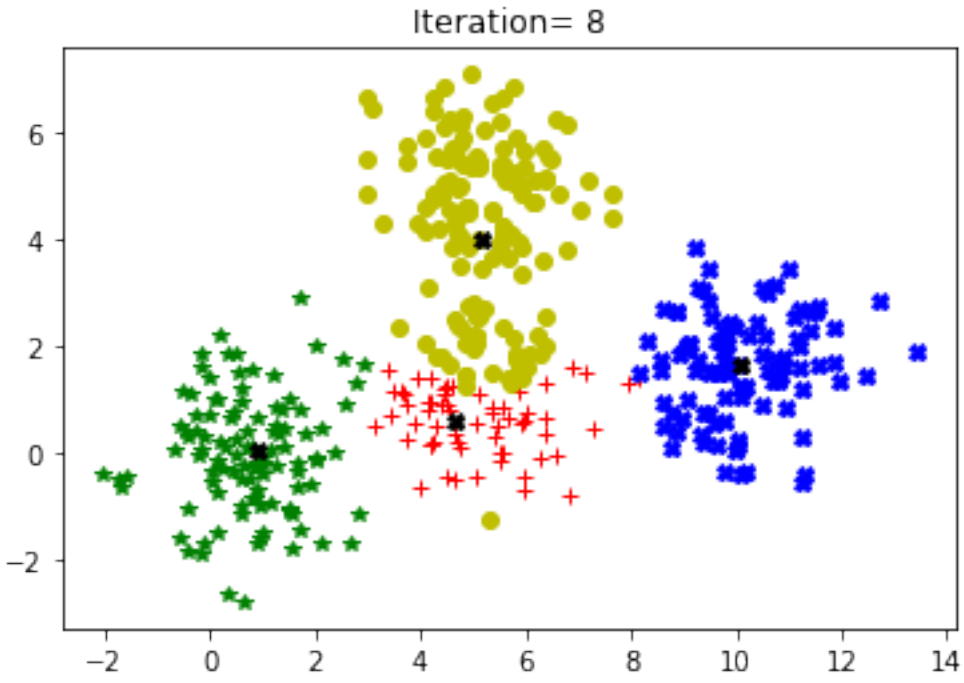

Iteration= 3
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

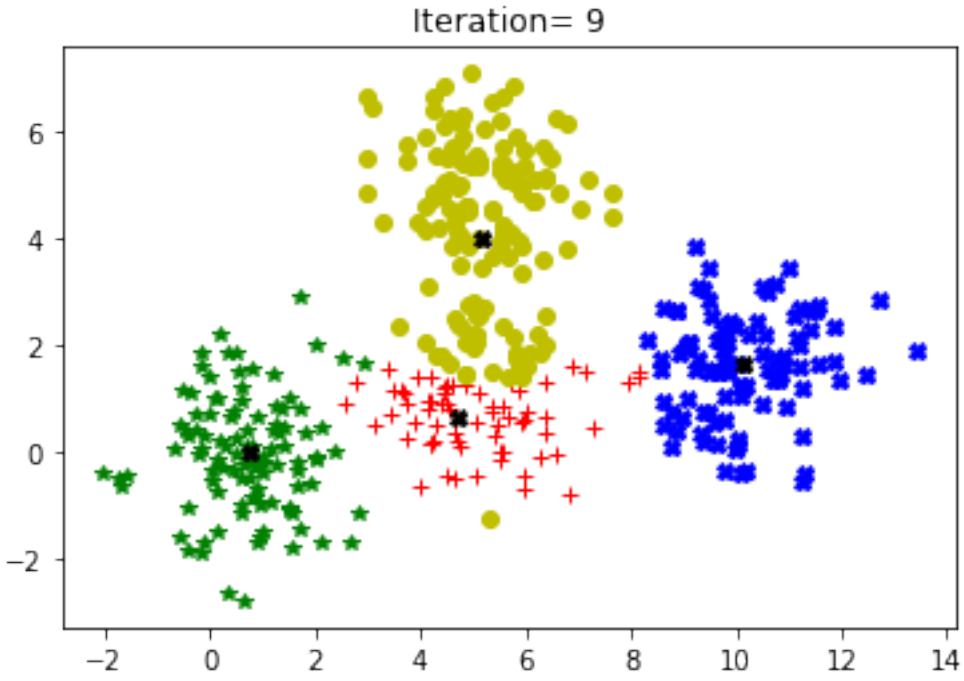

Iteration= 4
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

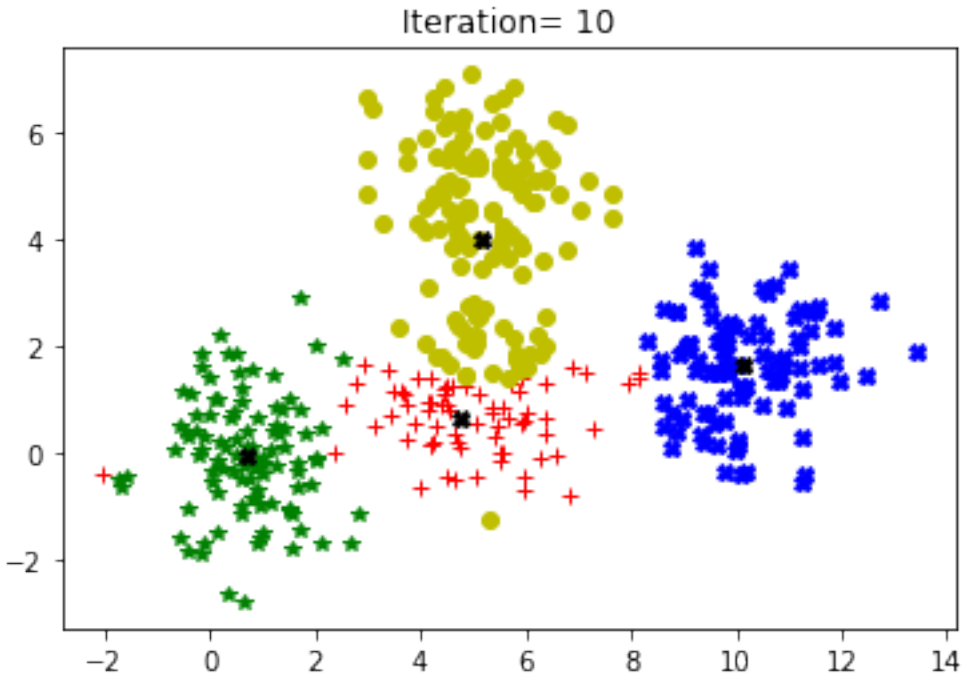

Iteration= 5
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

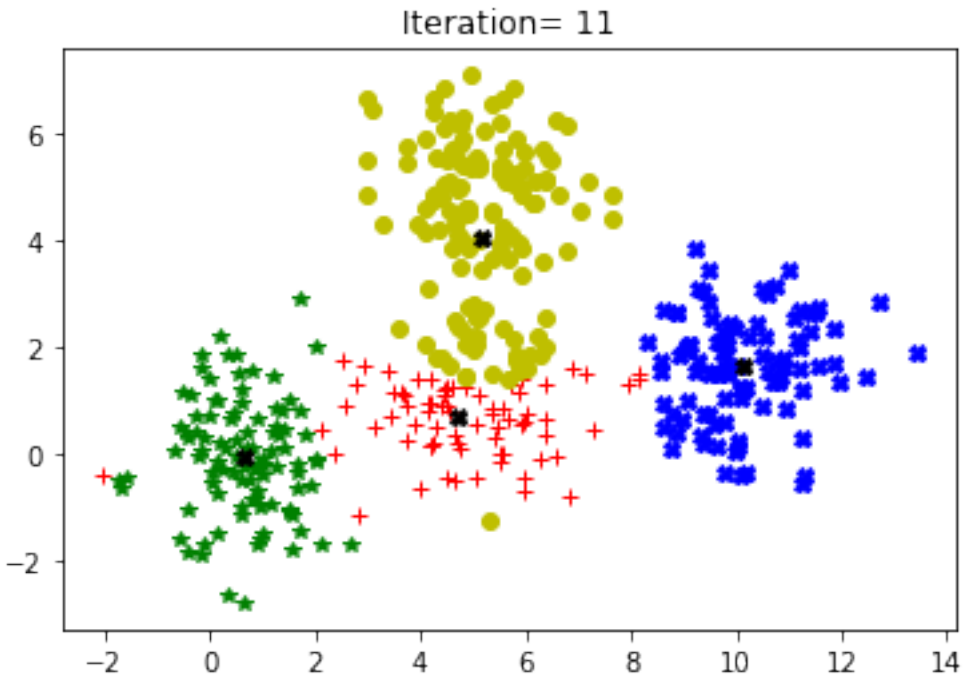

Iteration= 6
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

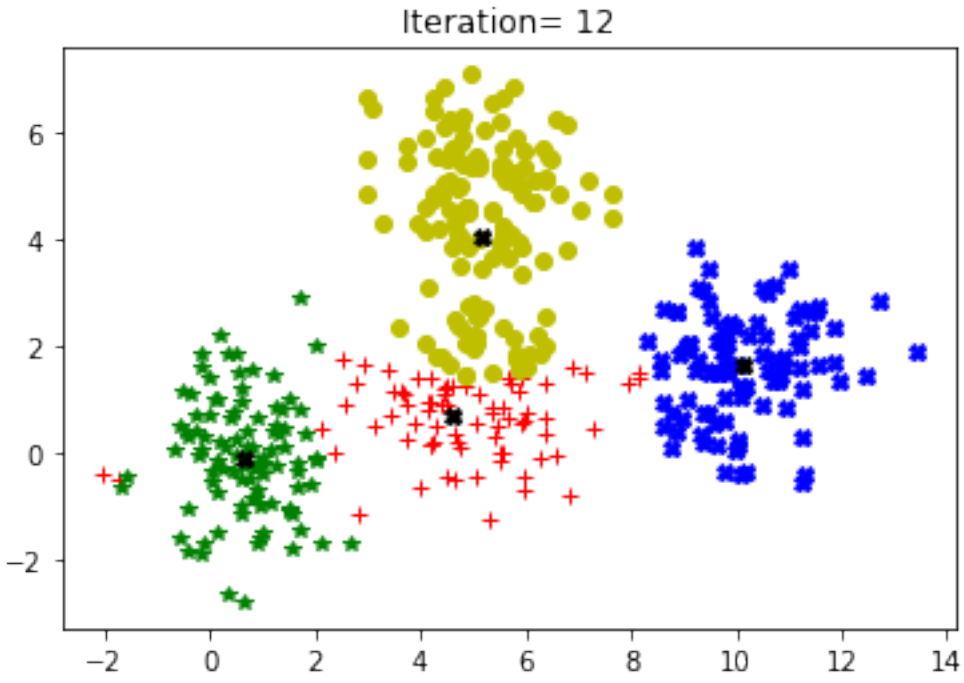

Iteration= 7
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

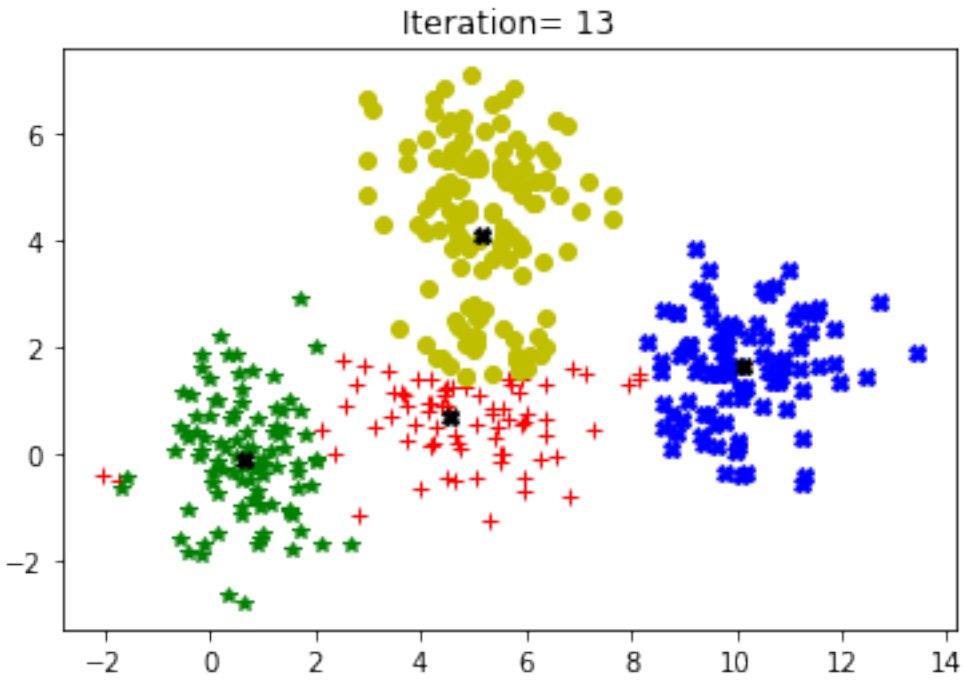

Iteration= 8
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

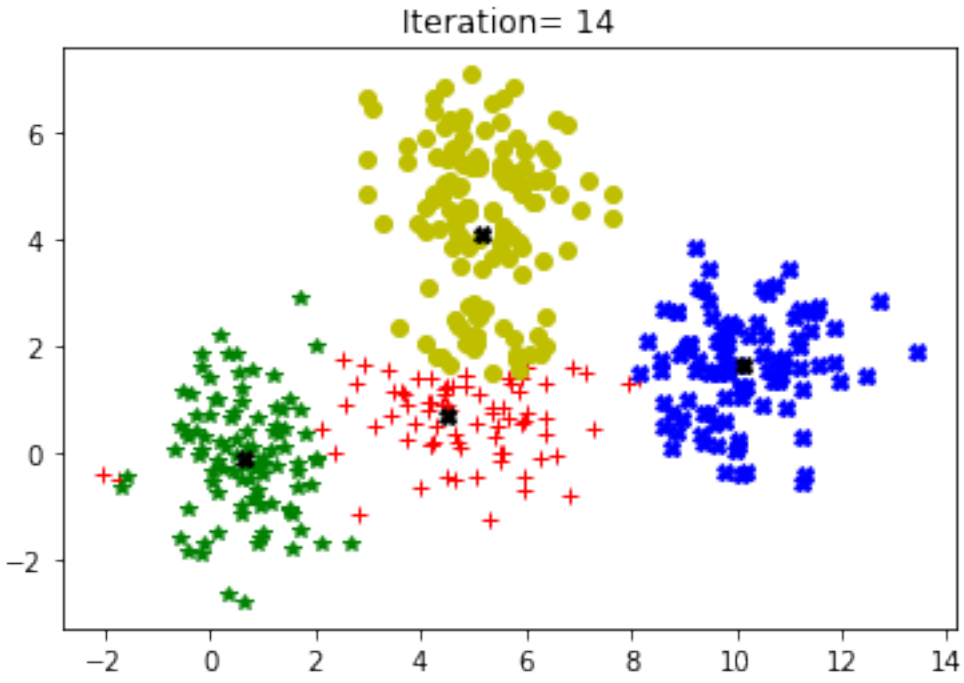

Iteration= 9
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

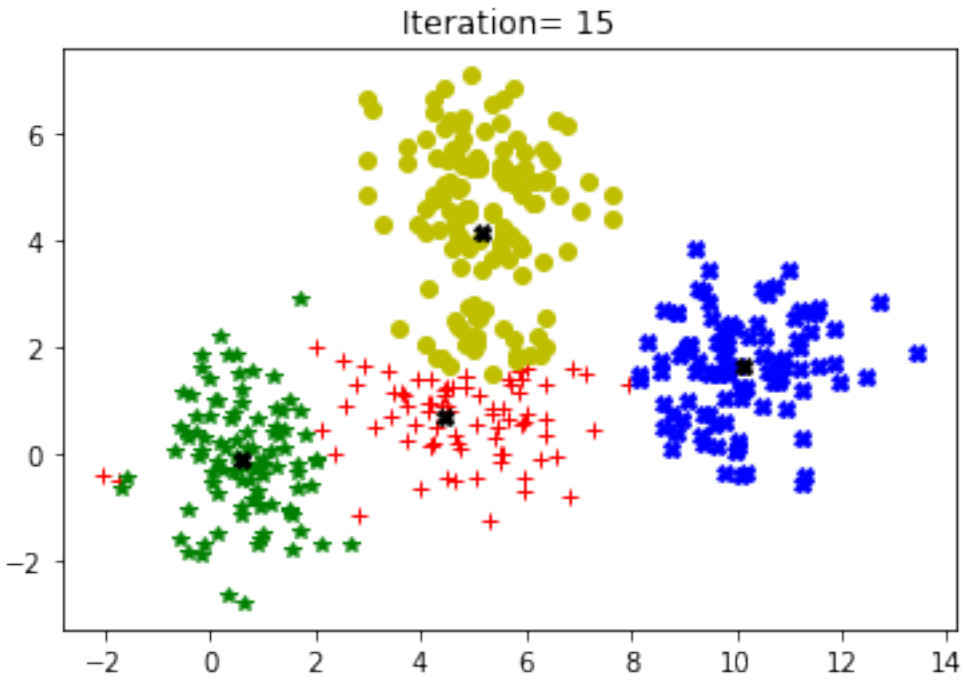

Iteration= 10
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

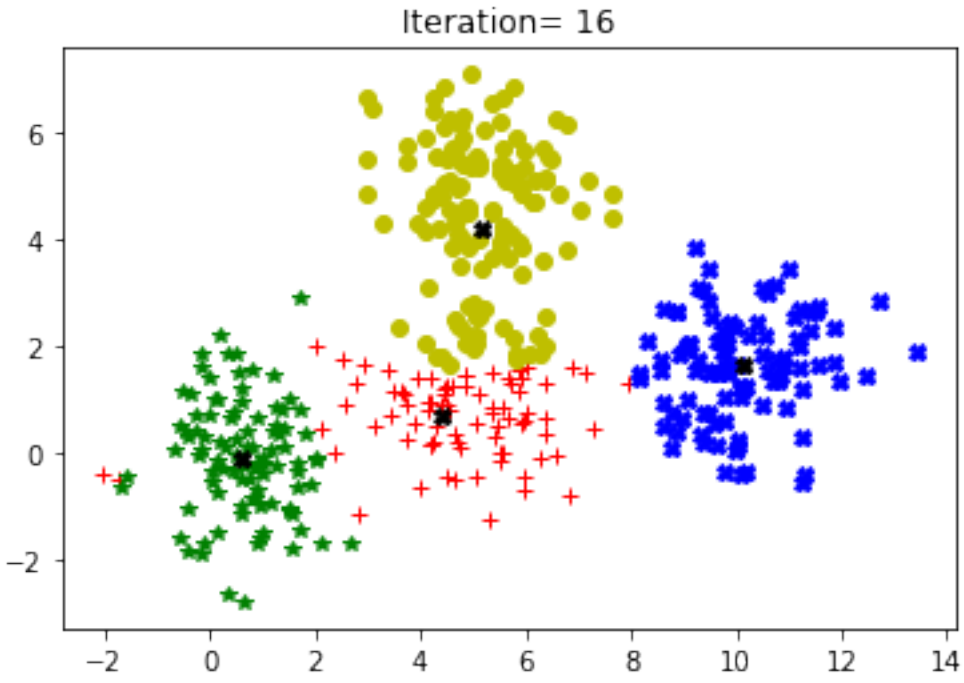

Iteration= 11
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

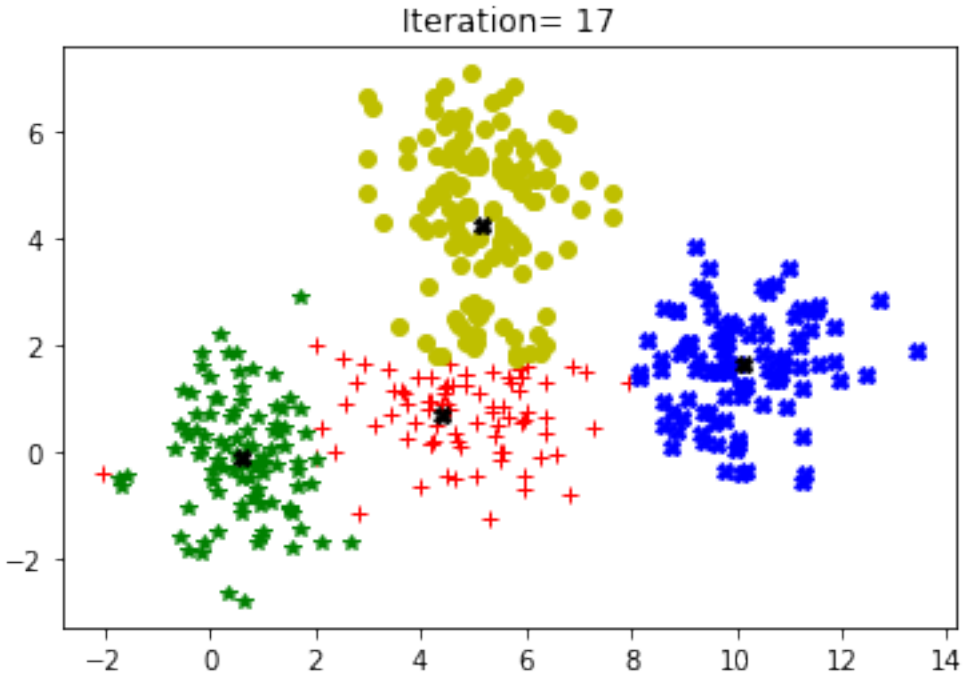

Iteration= 12
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

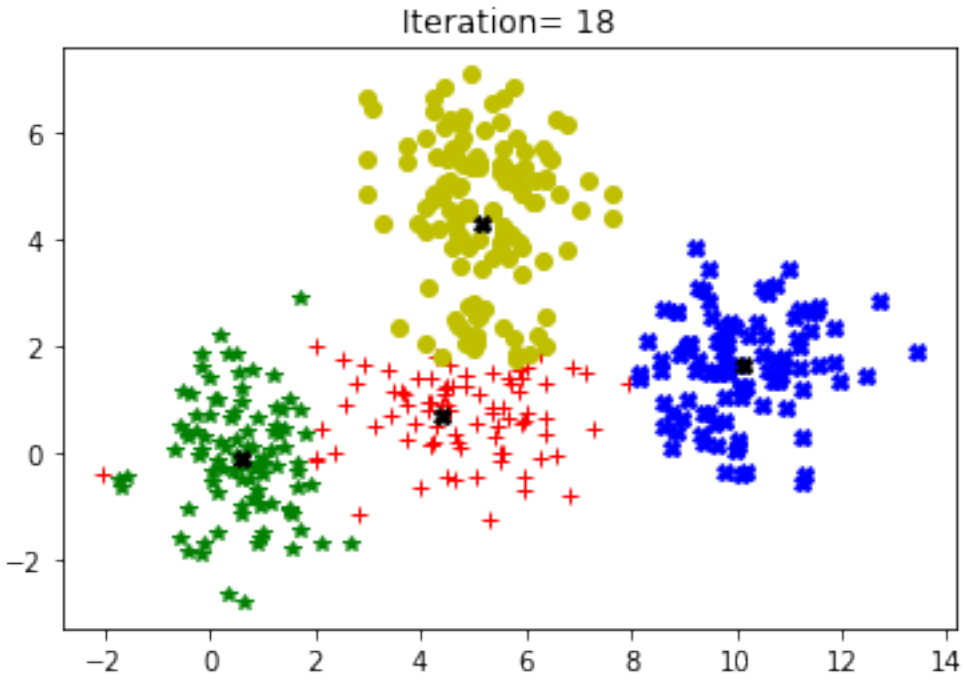

Iteration= 13
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

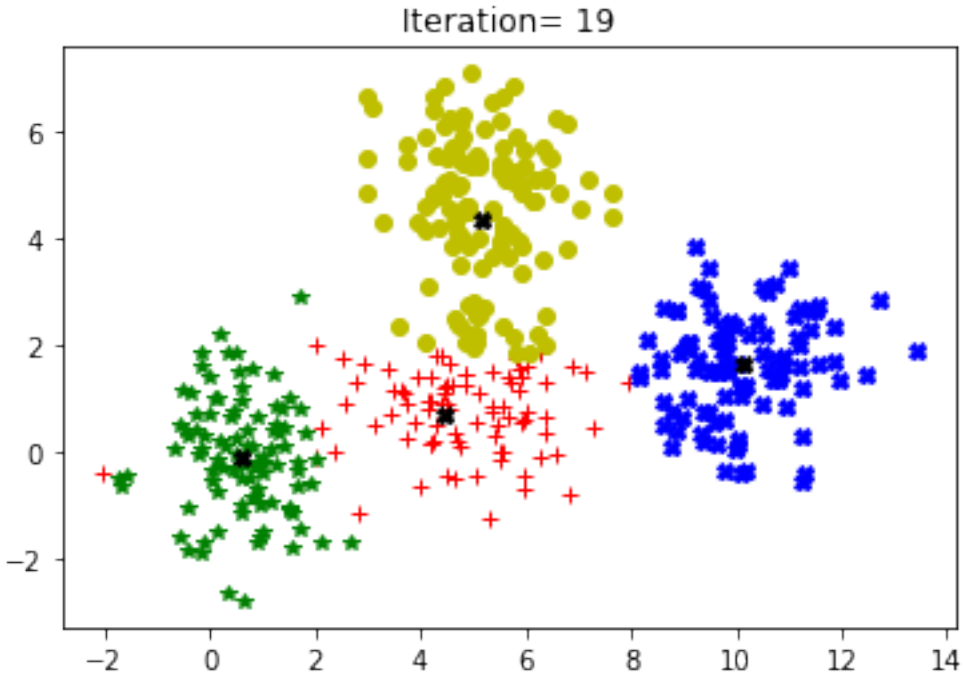

Iteration= 14
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

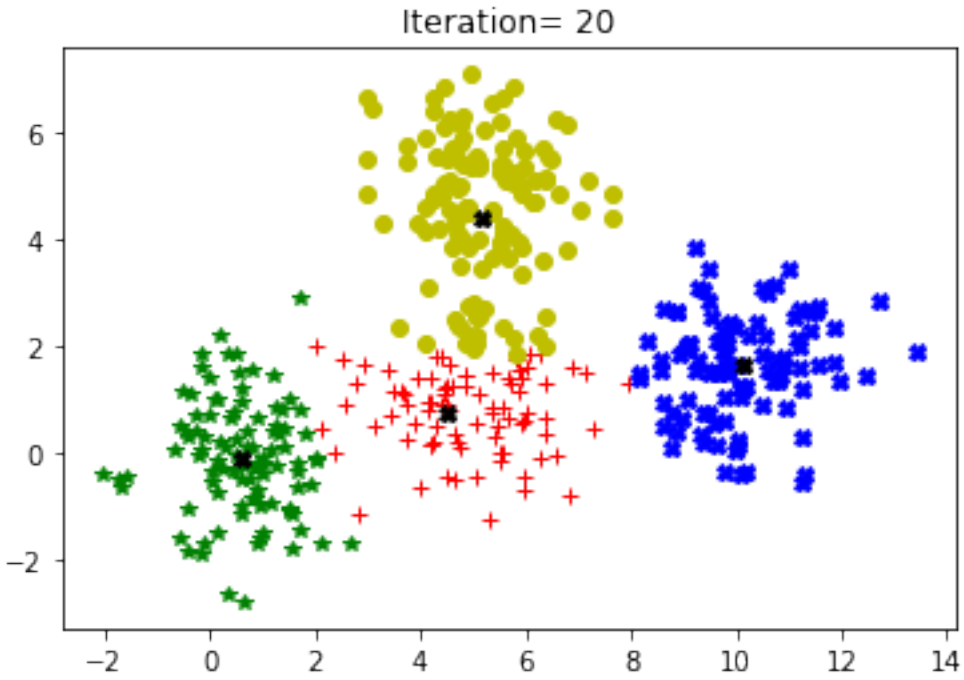

Iteration= 15
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

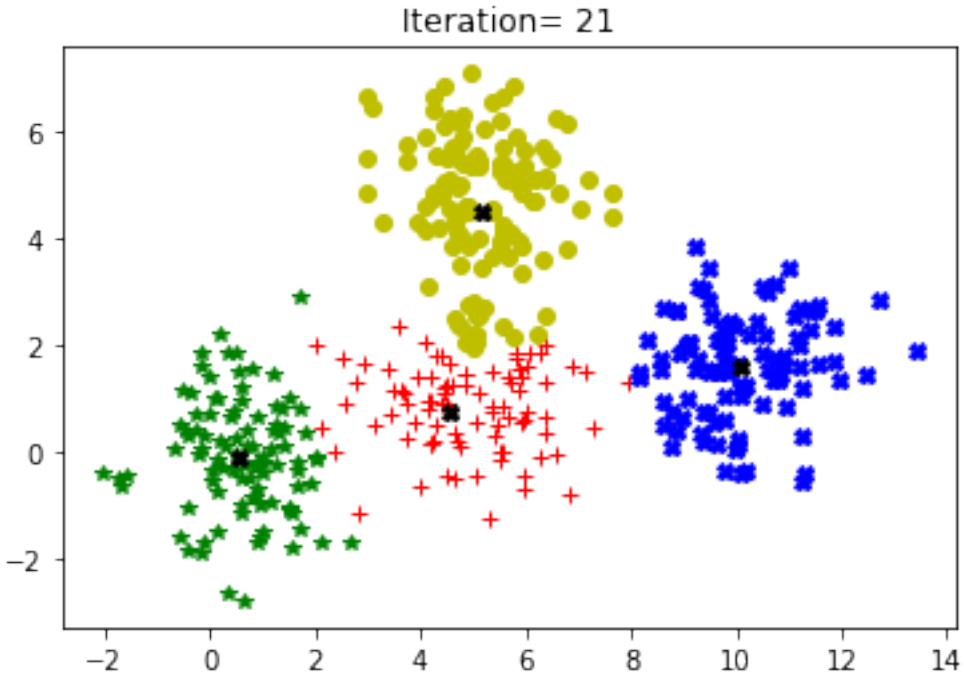

Iteration= 16
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

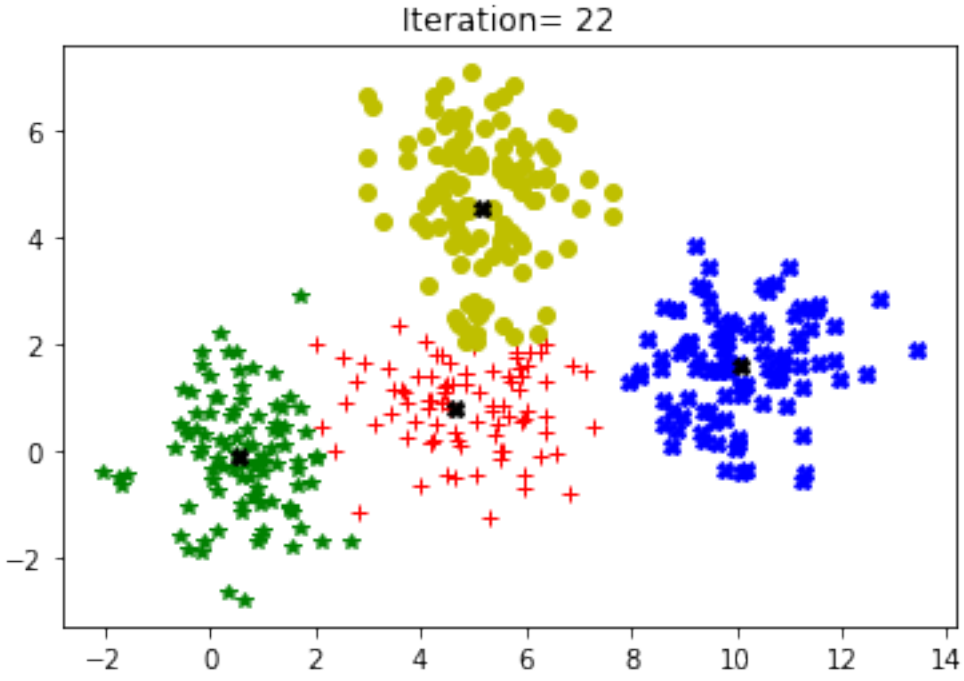

Iteration= 17
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

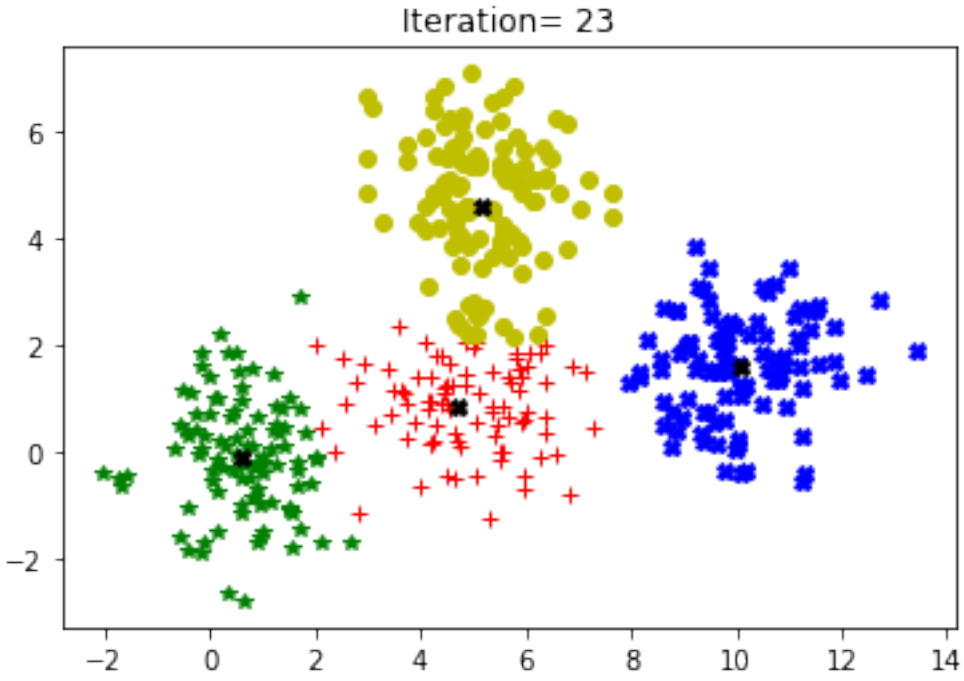

Iteration= 18
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

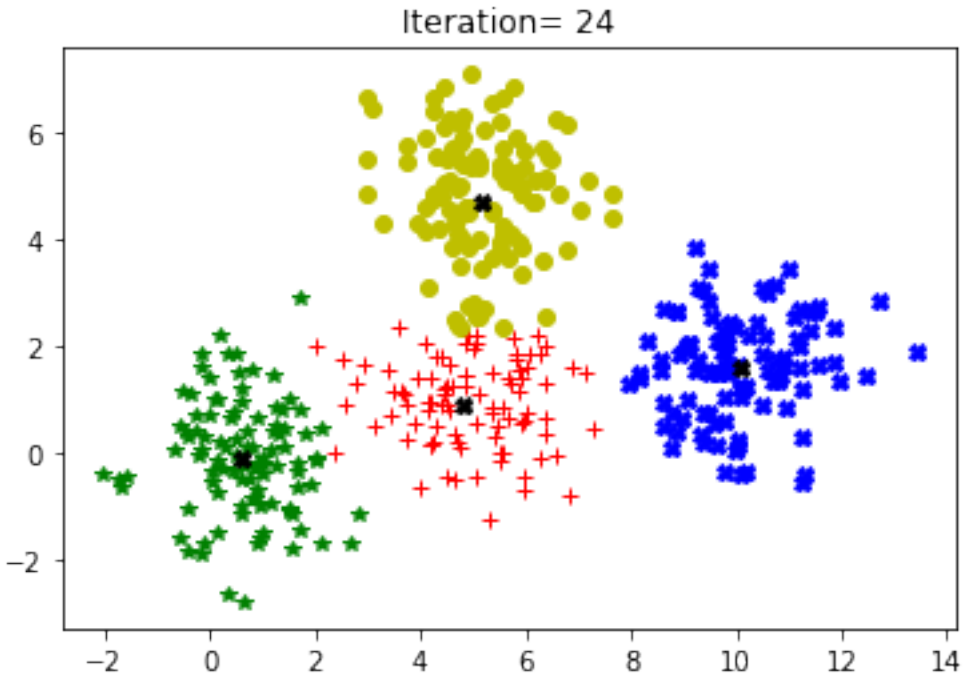

Iteration= 19
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

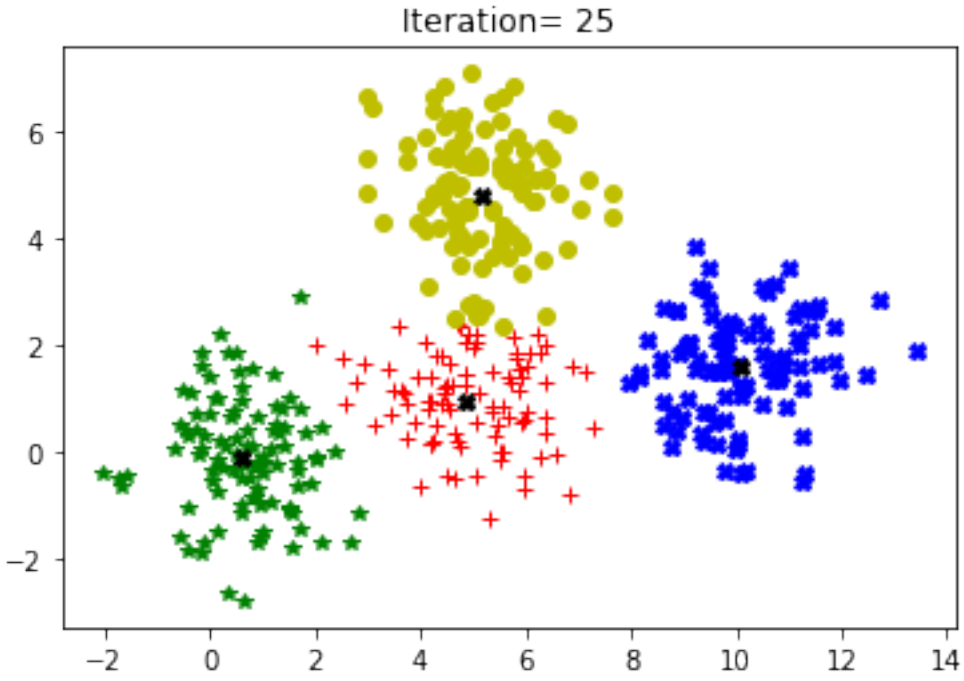

Iteration= 20
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

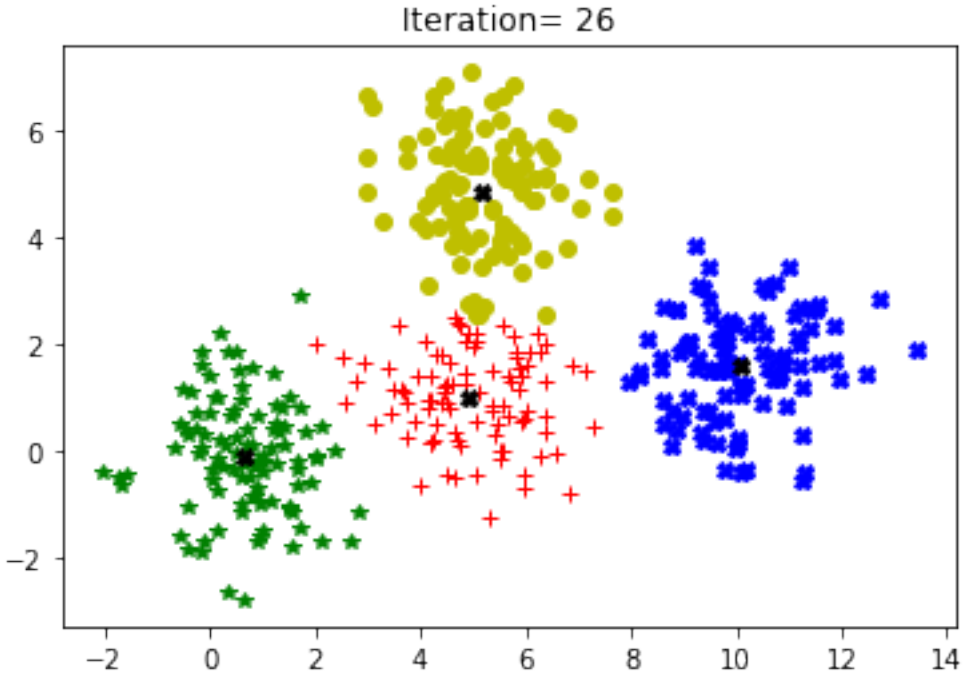

Iteration= 21
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

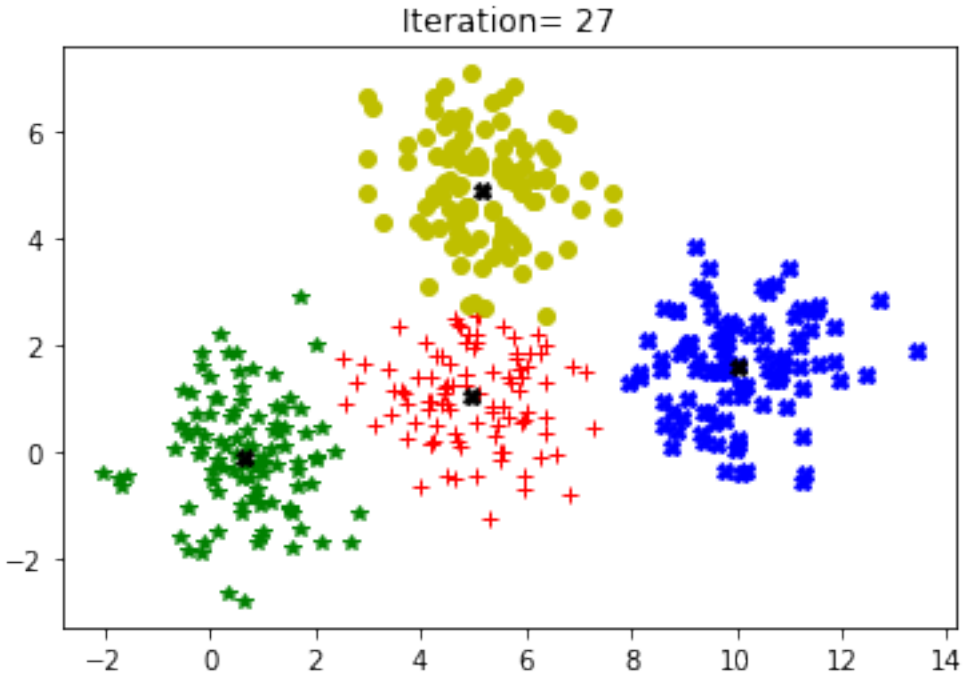

Iteration= 22
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

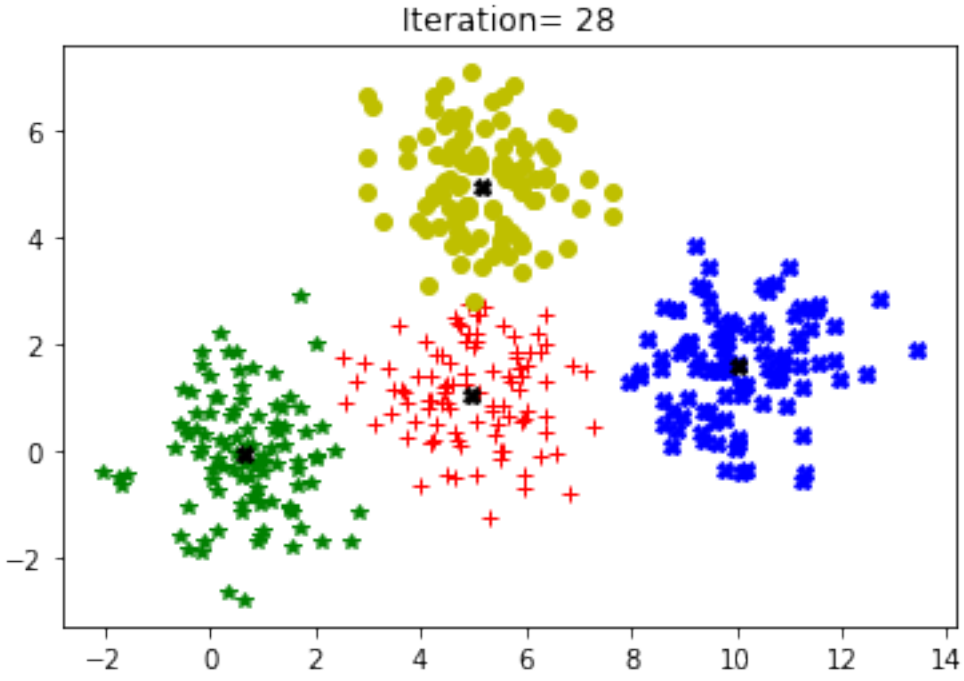

Iteration= 23
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

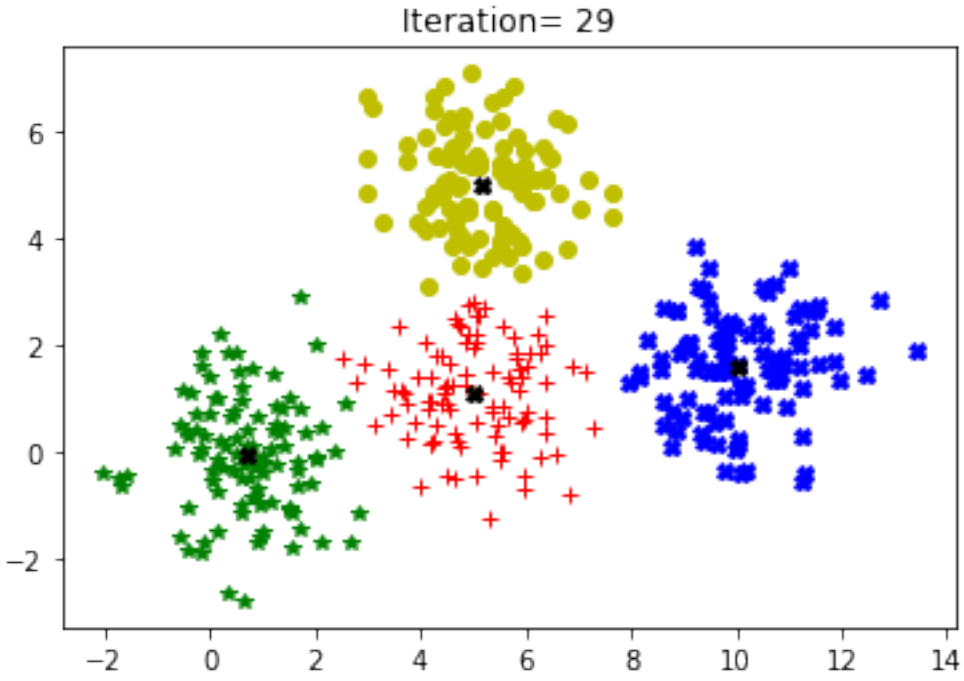

Iteration= 24
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

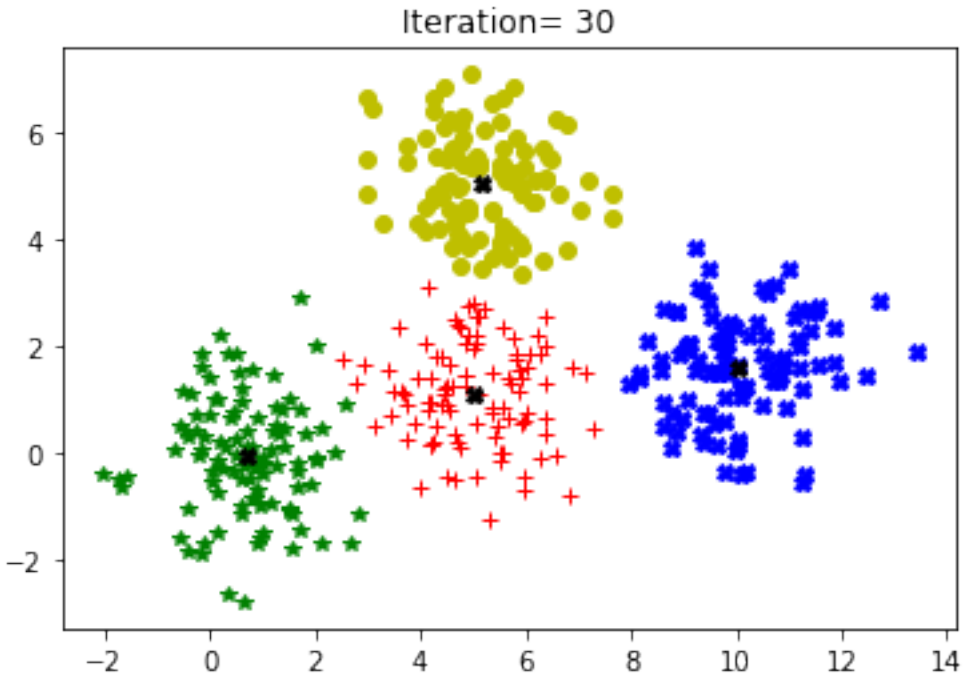

Iteration= 25
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

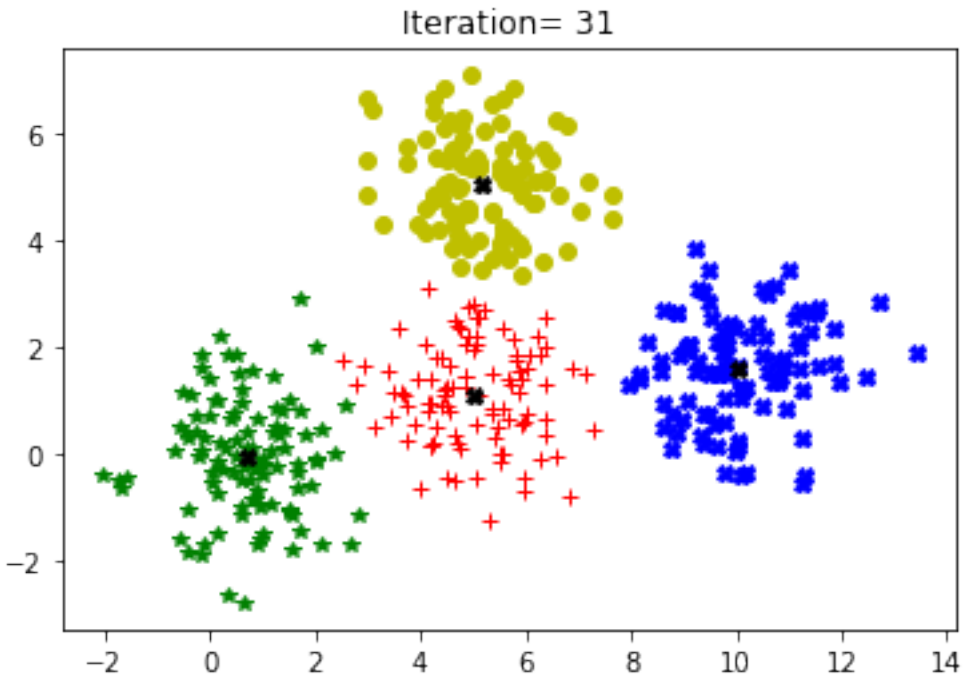

Iteration= 26
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

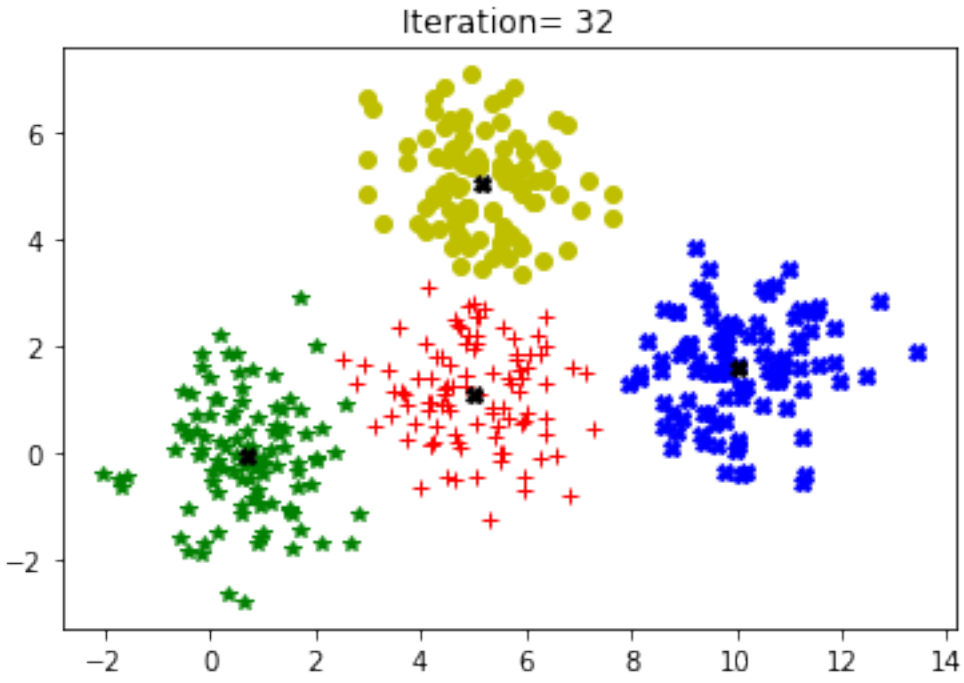

Iteration= 27
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

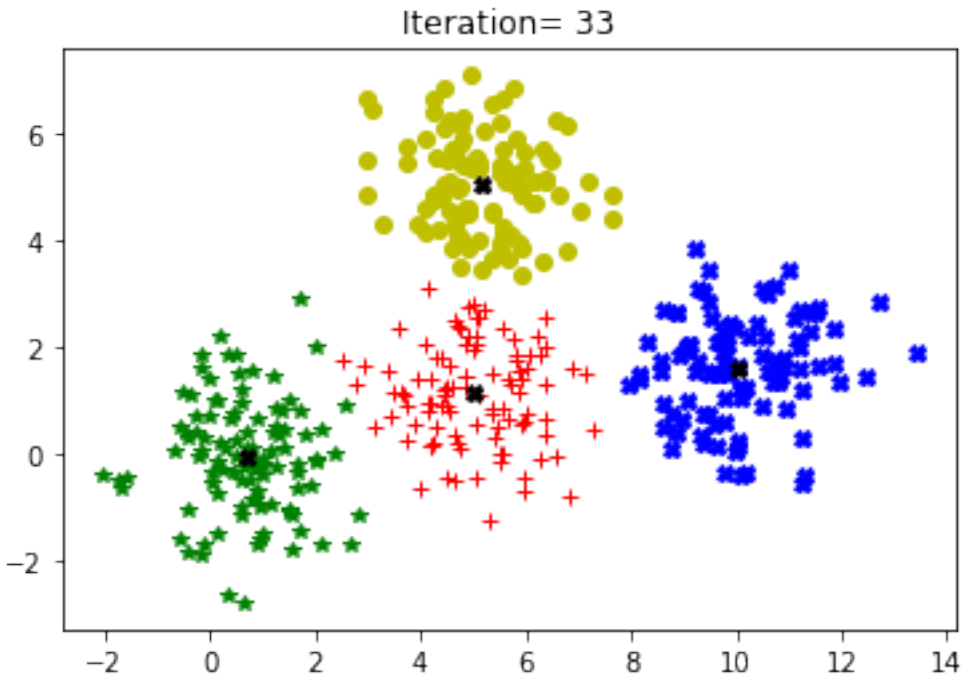

Iteration= 28
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

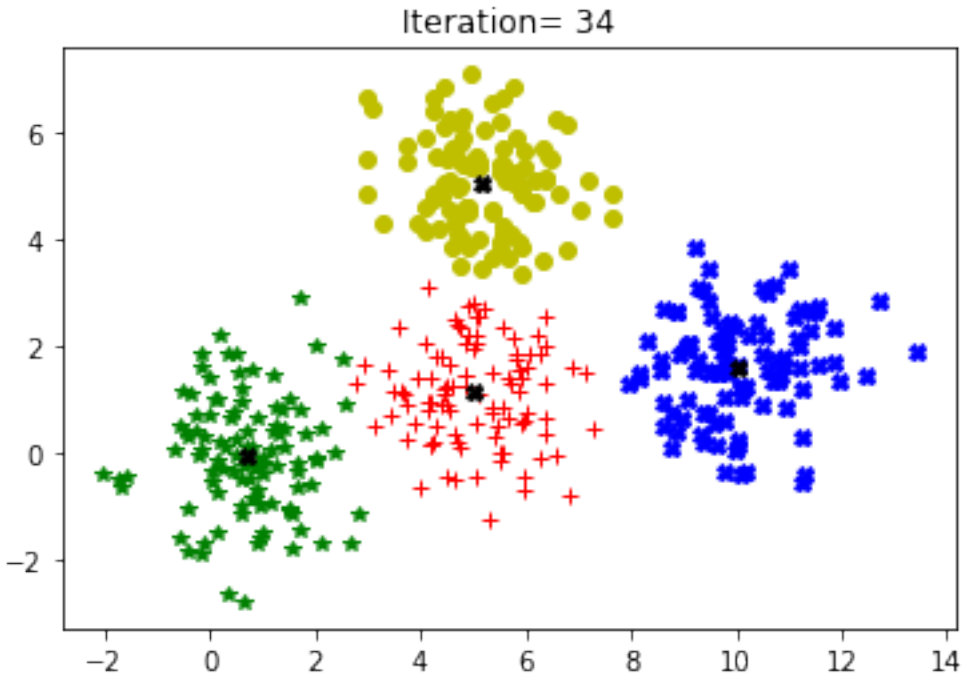

Iteration= 29
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

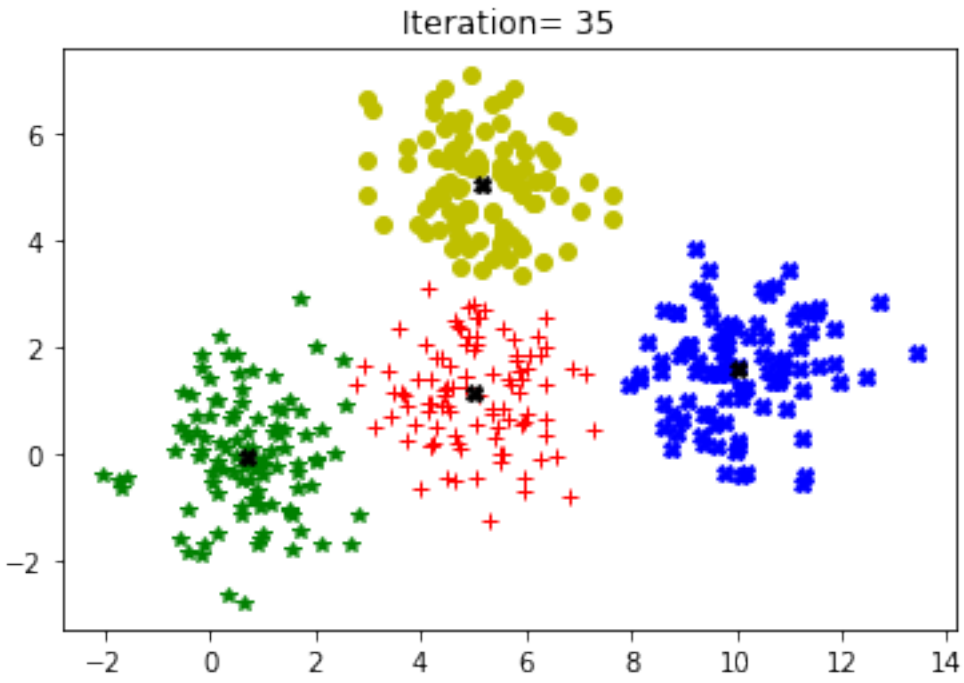

Iteration= 30
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

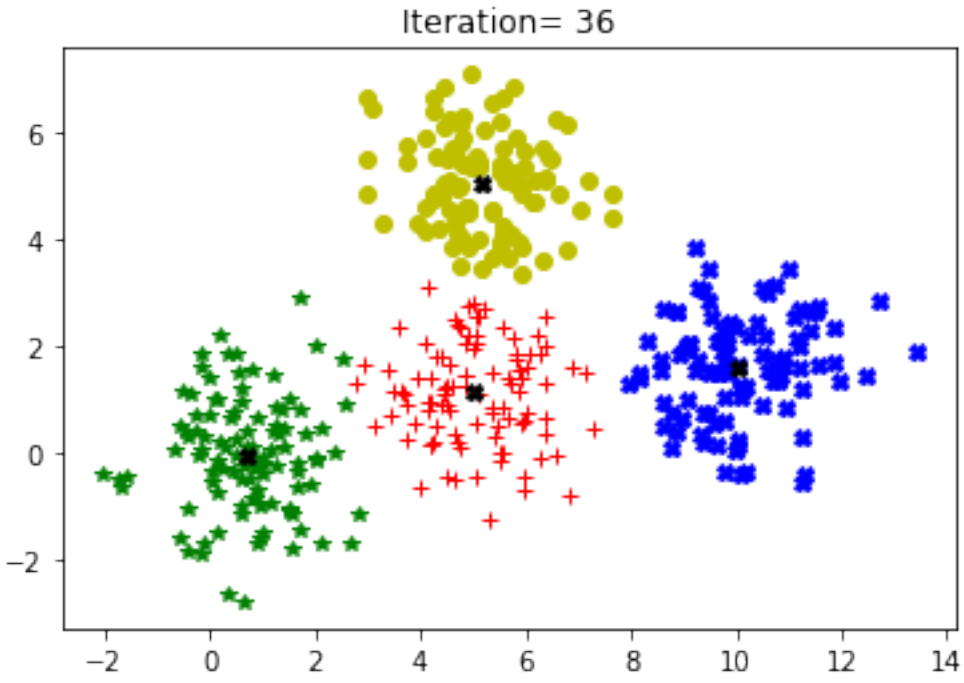

Iteration= 31
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

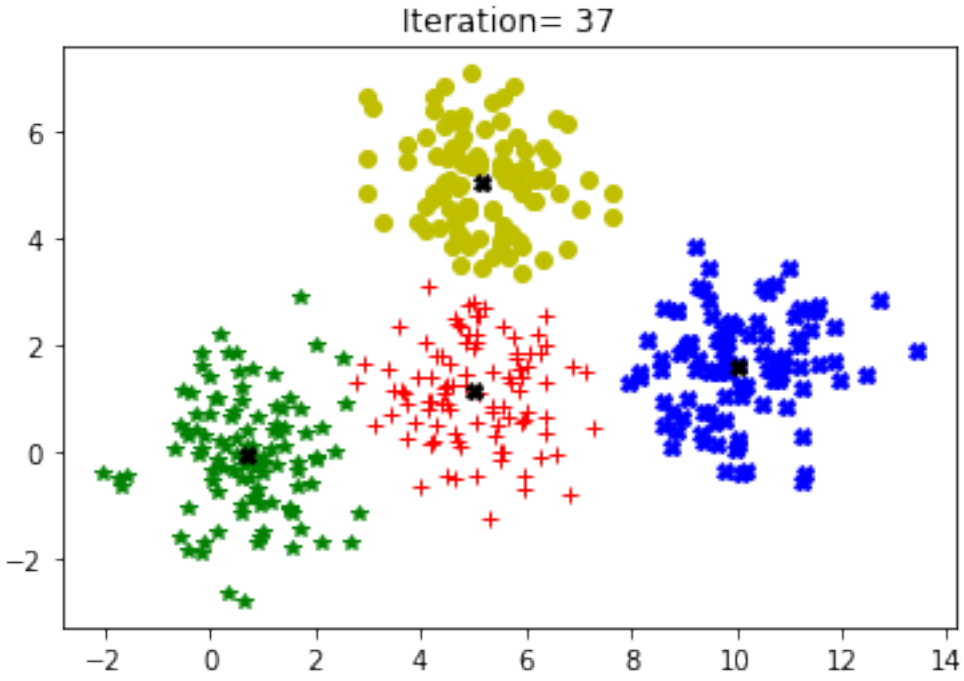

Iteration= 32
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

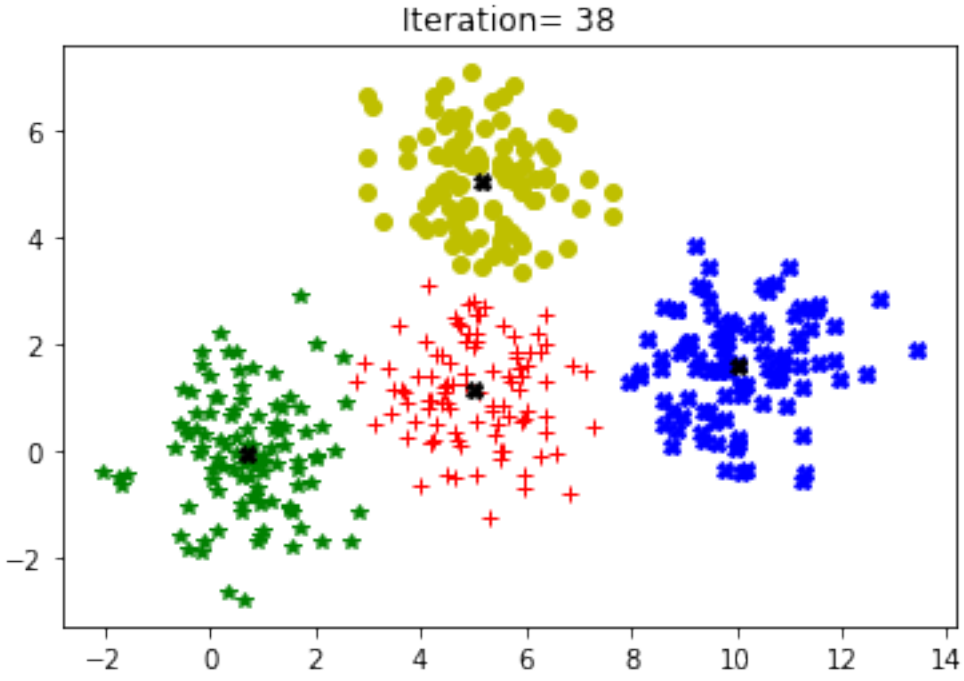

Iteration= 33
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

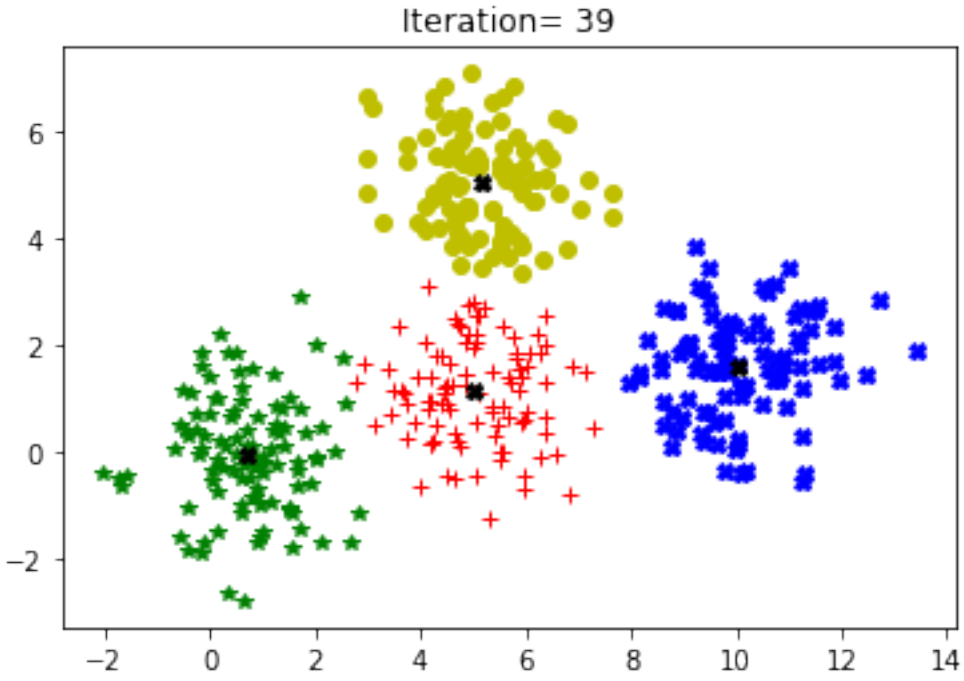

Iteration= 34
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

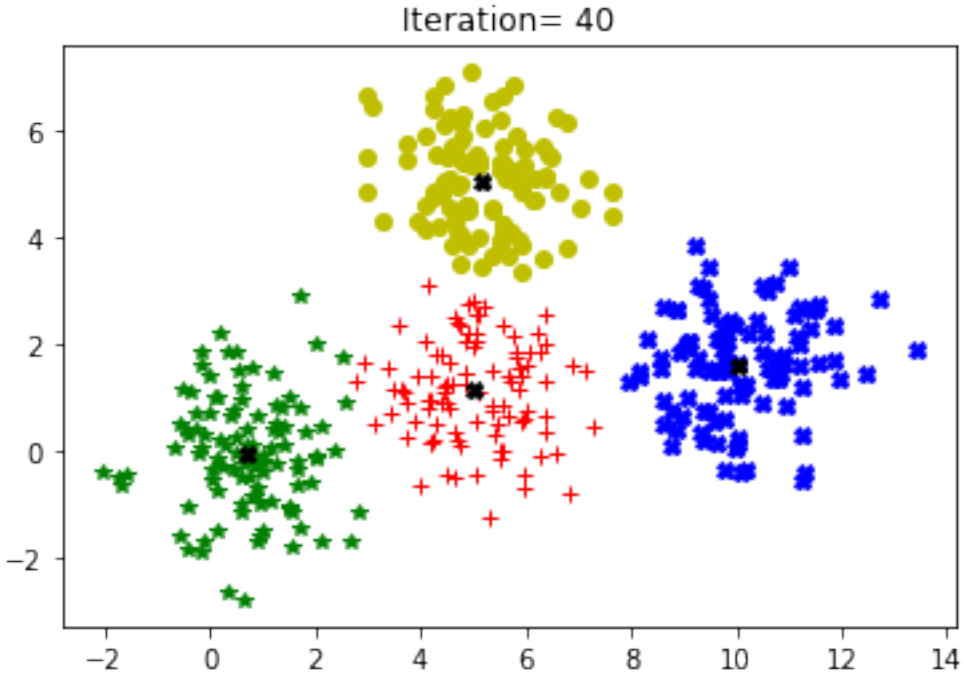

Iteration= 35
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

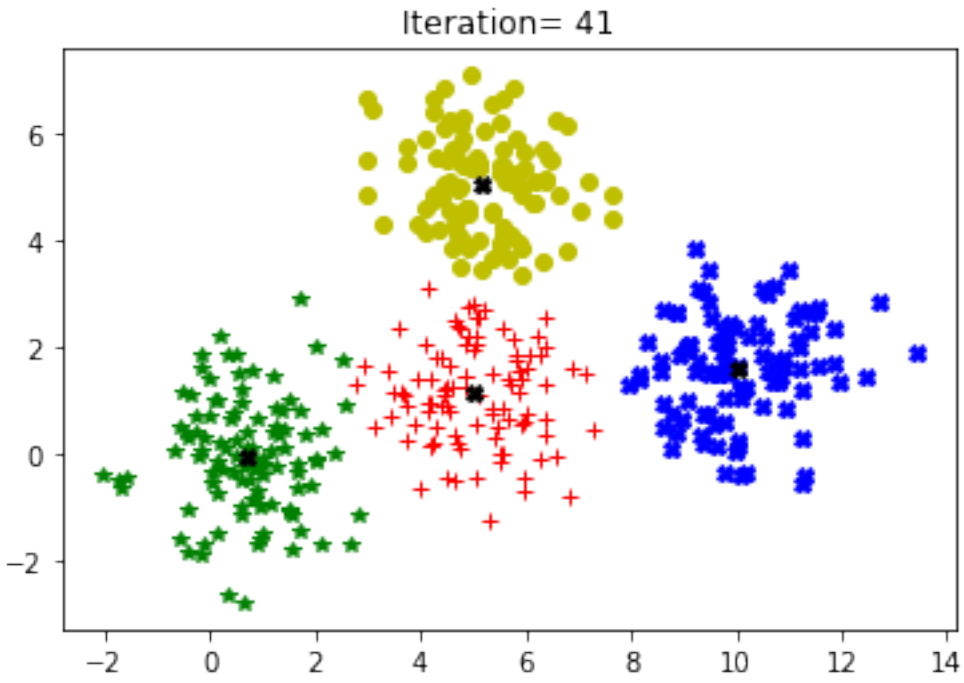

Iteration= 36
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

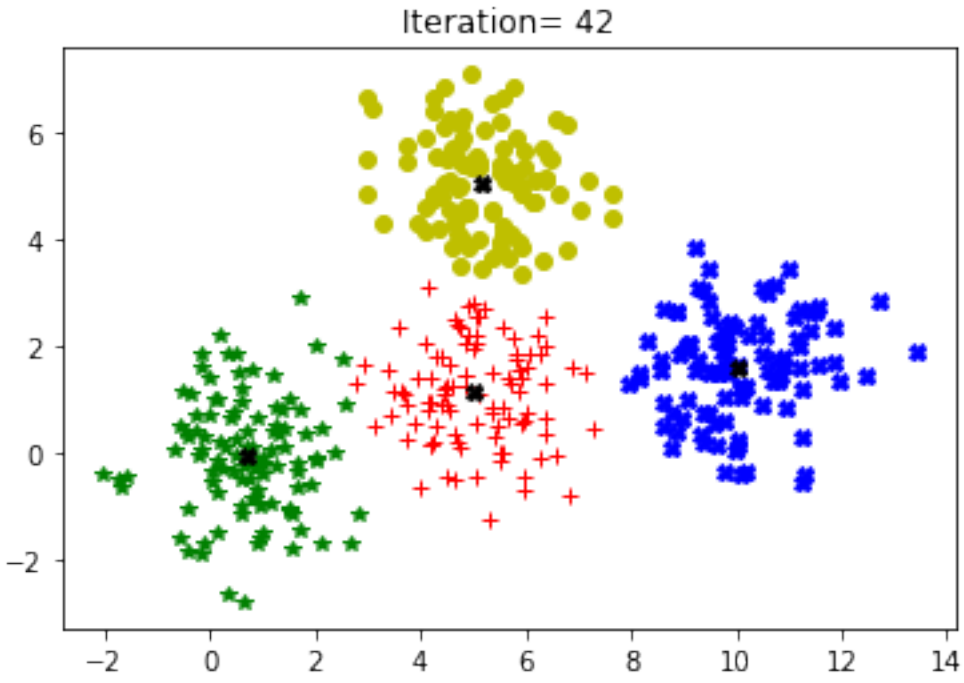

Iteration= 37
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

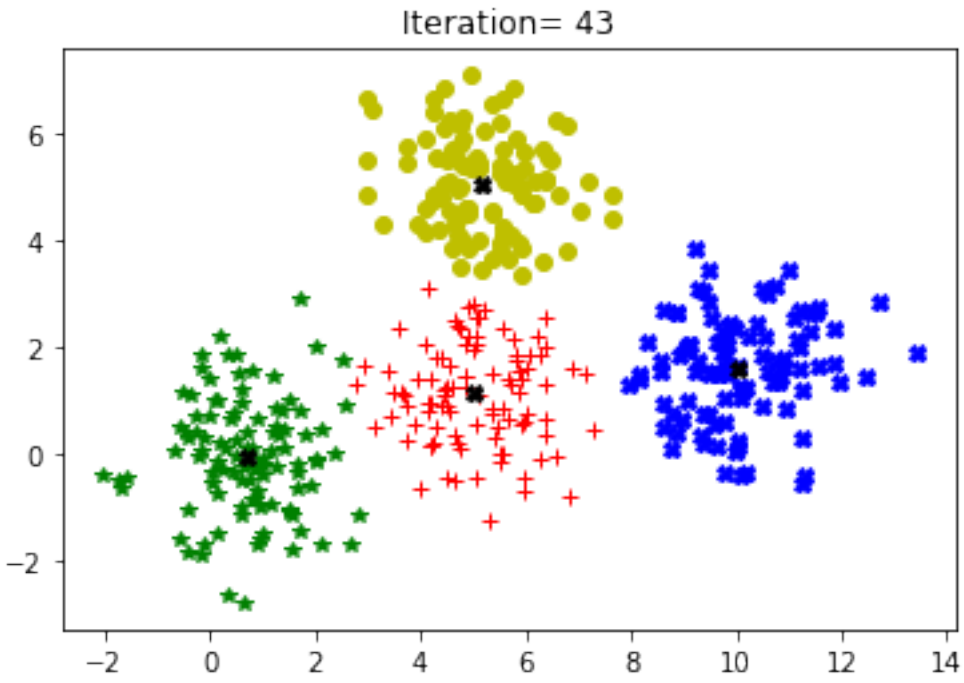

Iteration= 38
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

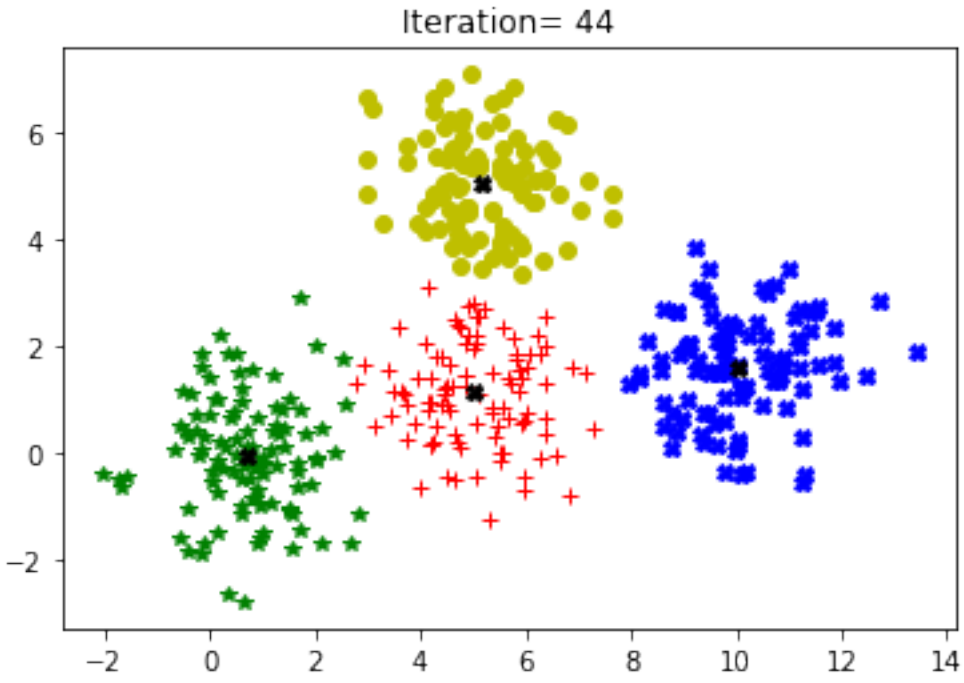

Iteration= 39
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

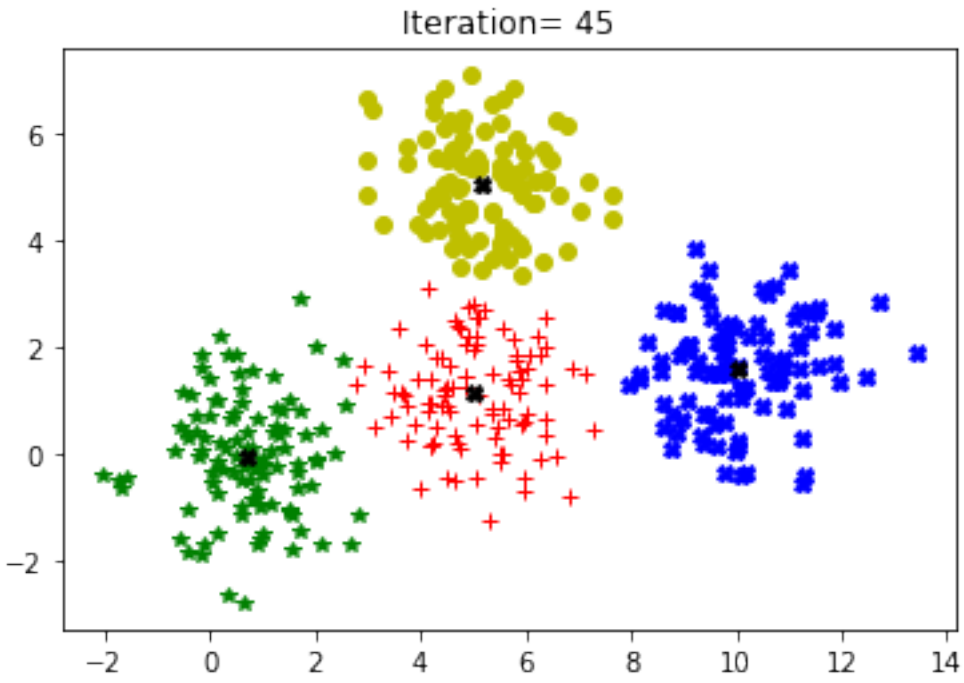

Iteration= 40
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

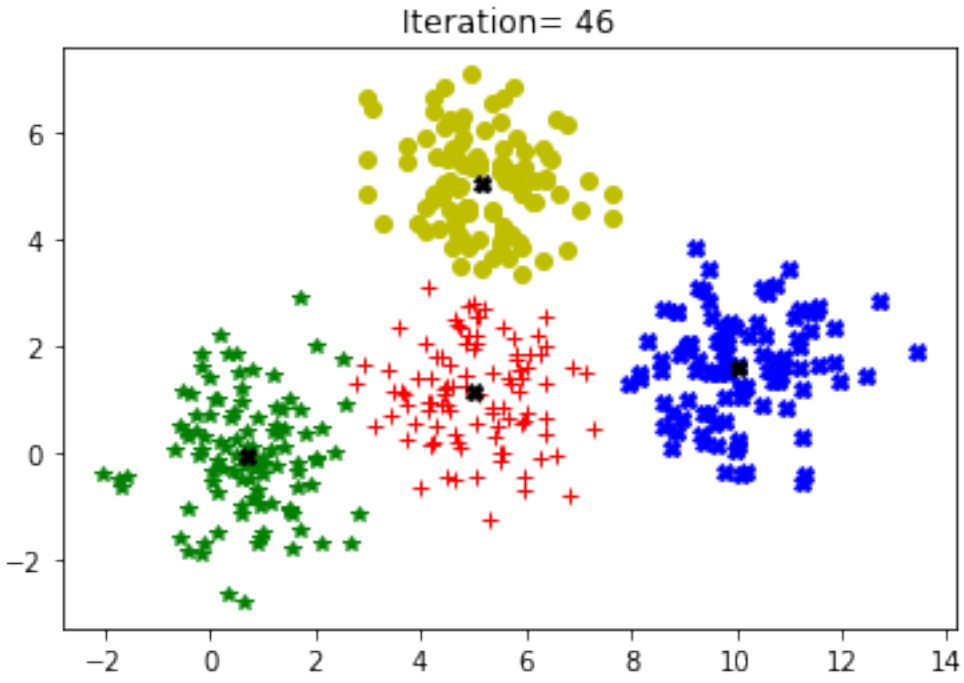

Iteration= 41
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

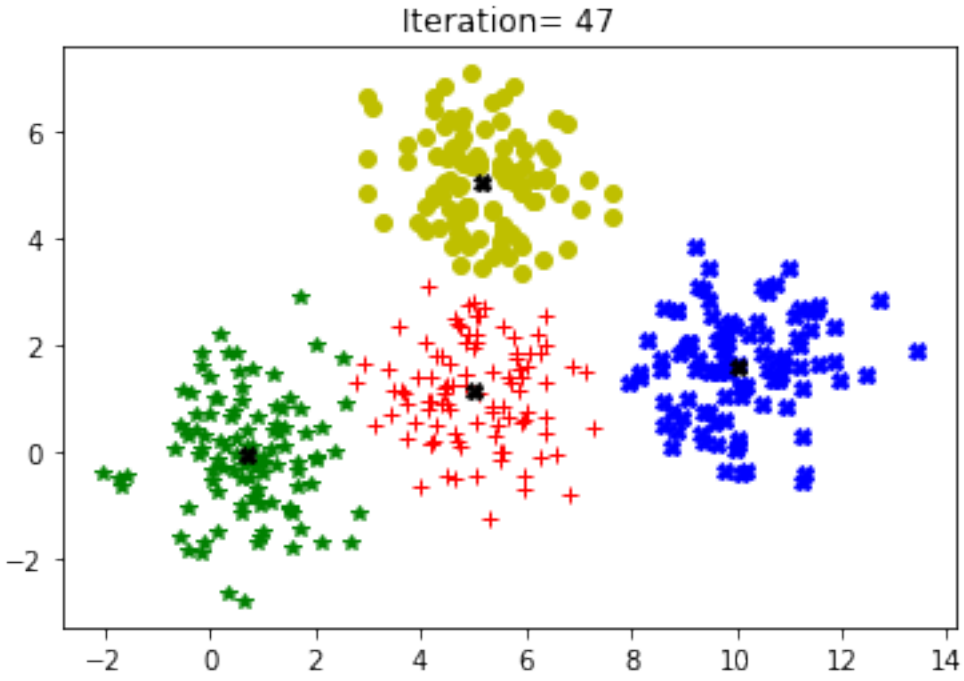

Iteration= 42
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

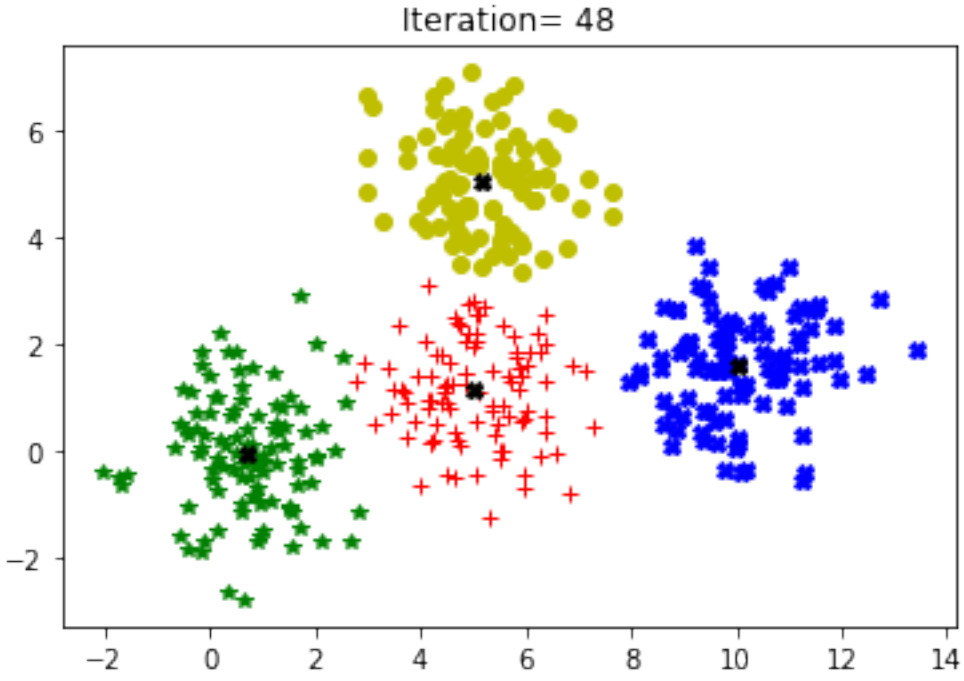

Iteration= 43
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

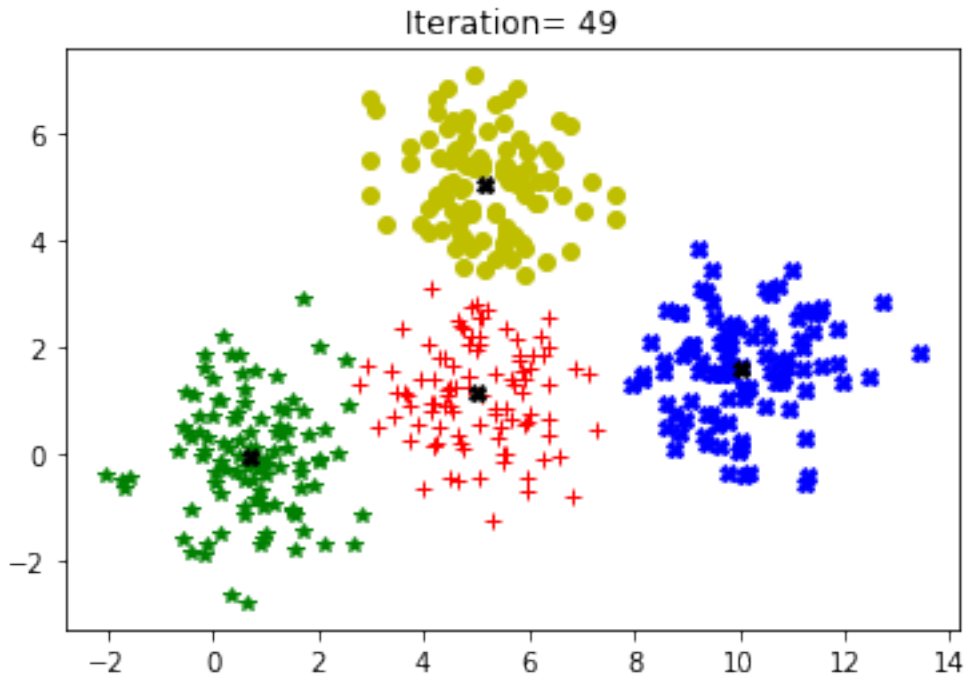

Iteration= 44
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14

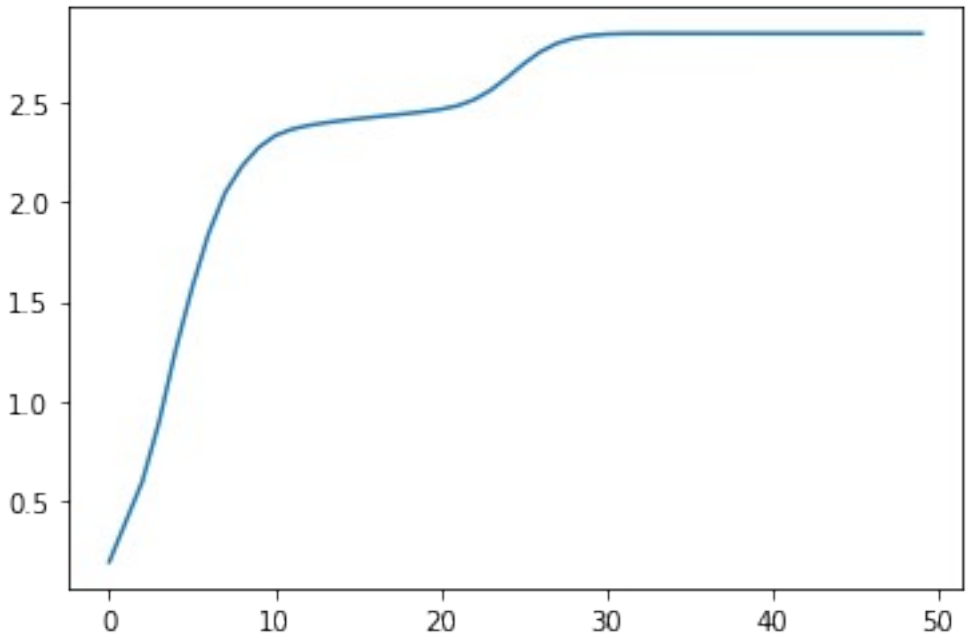

Iteration= 45
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration= 46
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration= 47
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14


Iteration= 48
6
4
2
0
−2
−2
0
2
4
6
8
10
12
14





Step 6 : Performance metric

Compute Homogeneity score and Silhouette coefficient using the information given below

Homogeneity score : A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. This metric is independent of the absolute values of the labels: a permutation of the class or cluster label values won't change the score value in any way.

Silhouette coeeficient :

$a(x)$: Average distance of x to all other vectors in same cluster

$b(x)$: Average distance of x to the vectors in other clusters. Find minimum among the clusters

$$s(x) = \frac{b(x) - a(x)}{max(a(x), b(x))}$$

Silhouette coefficient (SC) :

$$SC = \frac{1}{N} \sum_{i=1}^{N} s(x)$$

```
# write your code here
from sklearn.metrics.cluster import homogeneity_score,
silhouette_score

# Homogeneity Score
labels_true = []
labels_predicted = []
mean_vectors = Cents

for i in range(K):
    for j in range(len(data_sep[i])):
        labels_true.append(i)
        index = np.argmin(np.linalg.norm(data_sep[i][j] -
mean_vectors, axis=1))
        labels_predicted.append(index)

h_score = homogeneity_score(labels_true, labels_predicted)

print(f'Homogeneity score for the classification is: {h_score}')

# Silhouette coefficient
all_points = data
N = len(all_points)

pairwise = [[0 for i in range(N)] for j in range(N)]
for i in range(N):
    for j in range(N):
        pairwise[i][j] = np.linalg.norm(np.subtract(all_points[i],
all_points[j]))

labels=[]
for id, cluster in enumerate(assigned_clusters):
    labels += [id for i in range(len(cluster))]
```

```
sc_score = silhouette_score(pairwise, labels)
print(f'Silhouette coefficient for the classification is: {sc_score}')
```

```
Homogeneity score for the classification is: 0.9721668667897367
Silhouette coefficient for the classification is: 0.5582009349375389
```

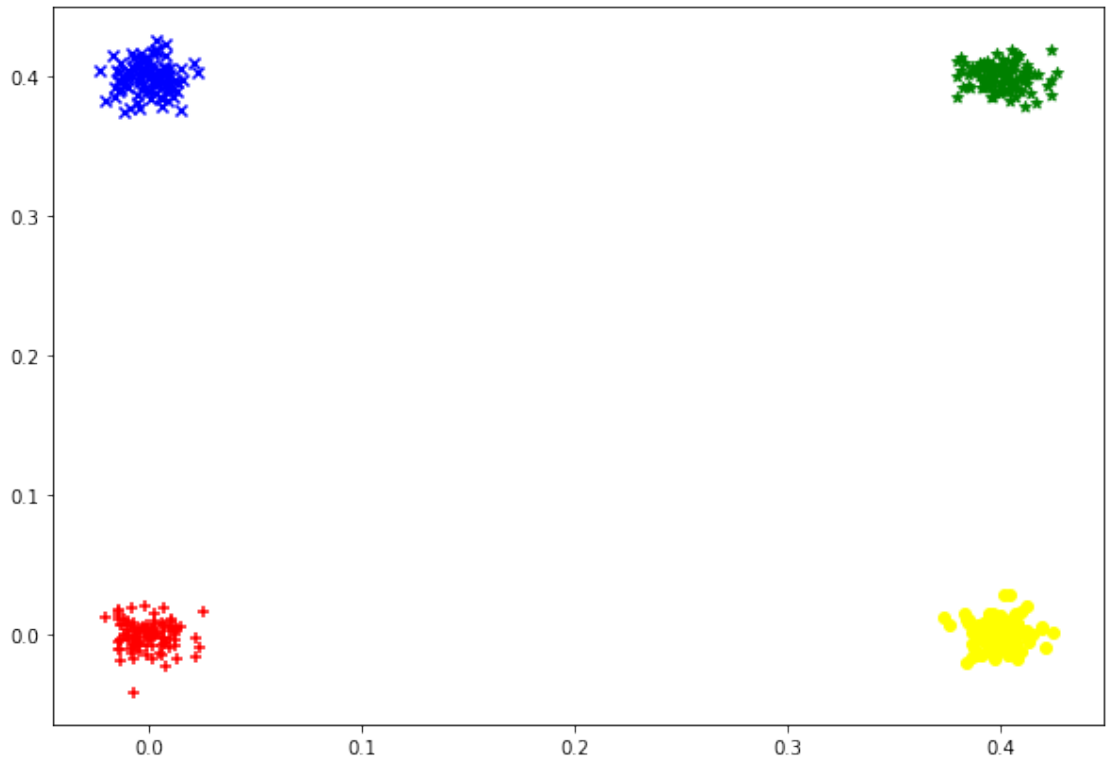
GMM v/s K-means

(a) Generate Data to show shortcomings of Kmeans and advantage of GMM over it

(b) Perform GMM on the same data and justify how it is better than K-means in that particular case

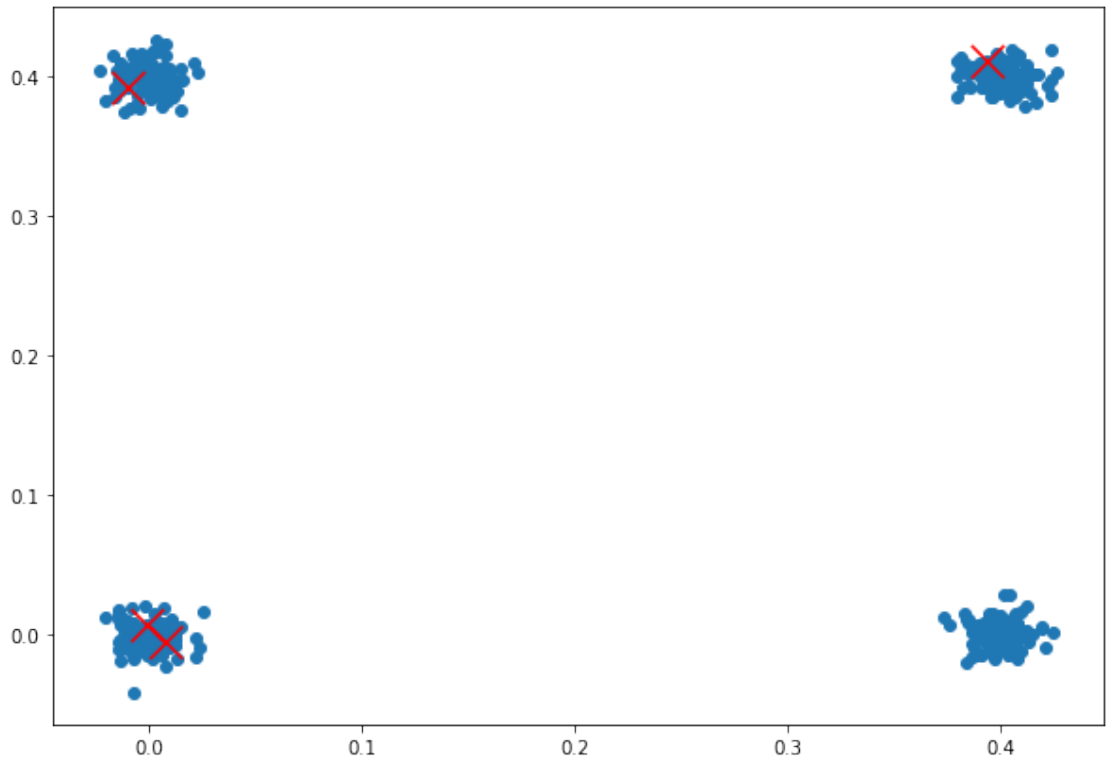
(c) Verify the same using performance metrics

```
# write your code here
K = 4

data = []
means = [(0, 0), (0.4, 0.4), (0, 0.4), (0.4, 0)]
color_style = [('red', '+'), ('green', '*'), ('blue', 'x'), ('yellow',
'o')]

plt.figure(figsize=(10, 7))

for i in range(K):
    d = np.random.multivariate_normal(mean=means[i],
cov=np.identity(2)*0.0001, size=100)
    plt.scatter(d[:, 0], d[:, 1], color=color_style[i%4][0],
marker=color_style[i%4][1])
    data.append(d)
```




```
data_points = np.concatenate(data, axis=0)
np.random.shuffle(data_points)


plt.figure(figsize=(10, 7))

mean_vectors = data_points[:K, :]

print("Mean Vectors:", mean_vectors)

plt.scatter(data_points[:, 0], data_points[:, 1])
plt.scatter(mean_vectors[:, 0], mean_vectors[:, 1], color='red',
marker='x', s=300)
```

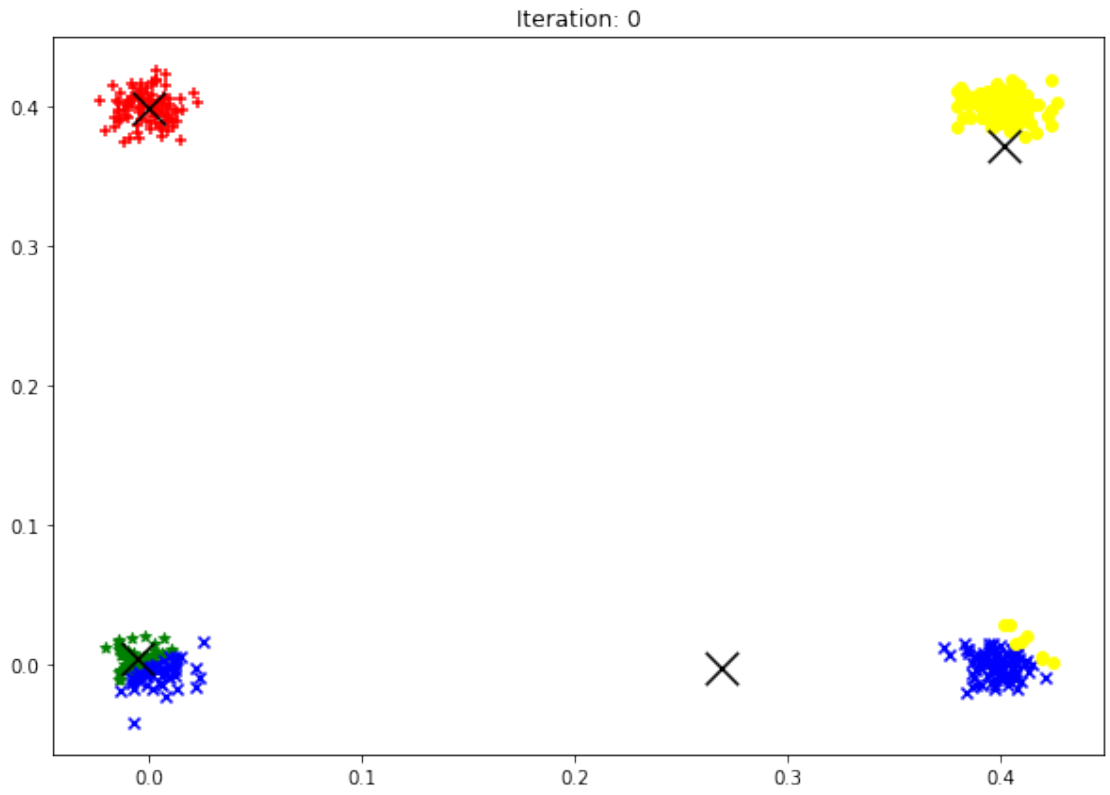
```
Mean Vectors: [[-0.00976398  0.39177267]
 [-0.00126376  0.00734528]
 [ 0.00824136 -0.0049231 ]
 [ 0.39335206  0.40968358]]

<matplotlib.collections.PathCollection at 0x1d121bd4d30>
```

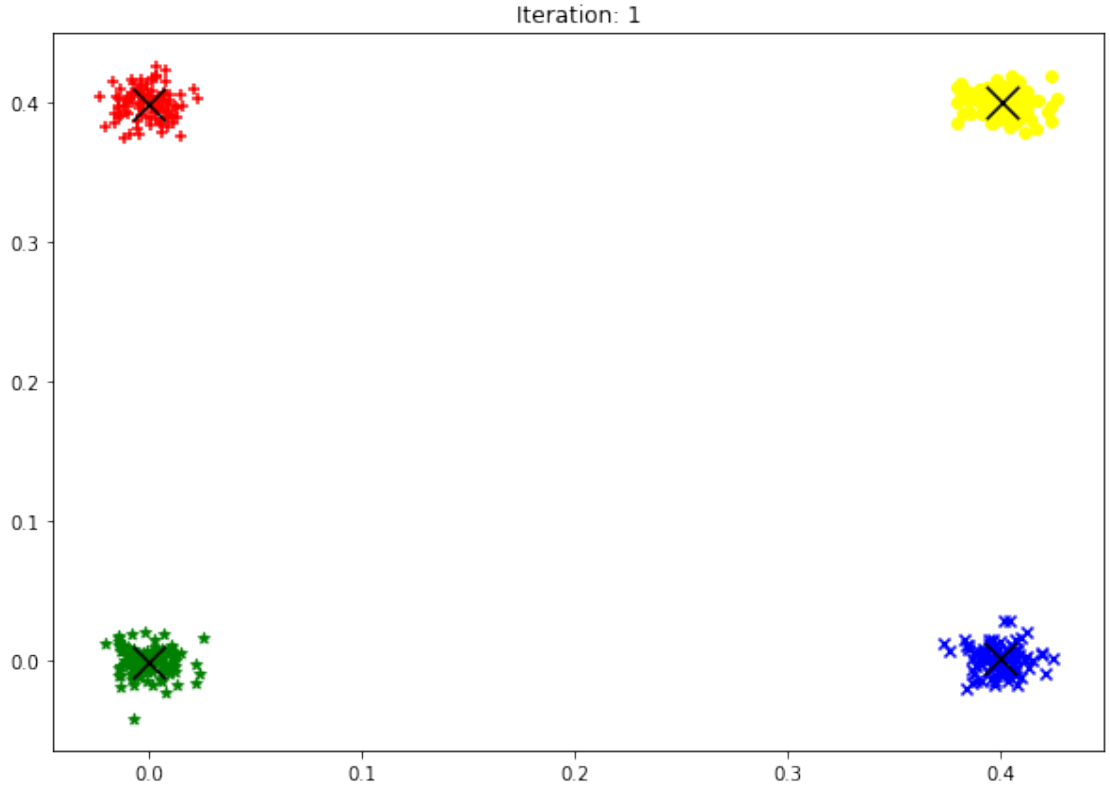


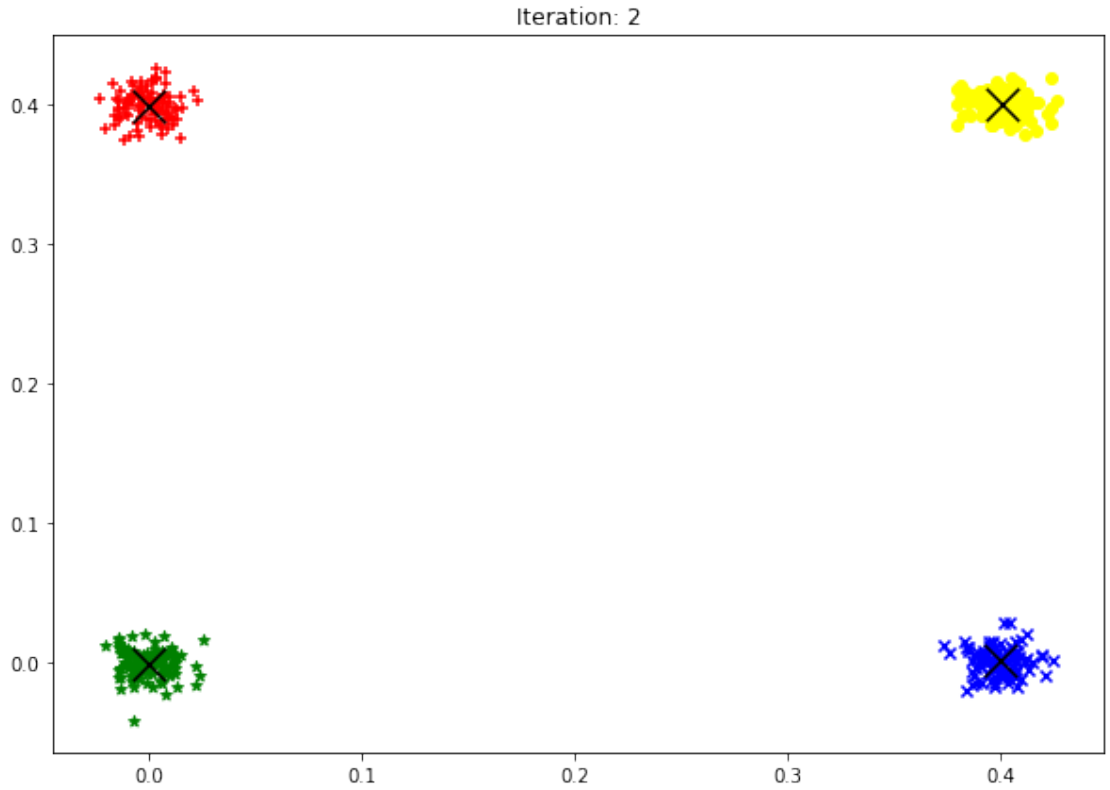

```
tol = 1e-5

delta = float('inf')
prev_error = float('inf')

errors = []
iter = 0

while delta>tol:
    assigned = [[] for i in range(K)]
    for i in range(data_points.shape[0]):
        min_index = np.argmin(np.linalg.norm(data_points[i] -
mean_vectors, axis=1))
        assigned[min_index].append(i)

    assigned_clusters = []
    for i in range(K):
assigned_clusters.append(data_points[assigned[i]])

    mean_vectors = np.array([cluster.mean(axis=0) for cluster in
assigned_clusters])

    plt.figure(figsize=(10, 7))

    for i in range(K):
        plt.scatter(assigned_clusters[i][:, 0], assigned_clusters[i]
```

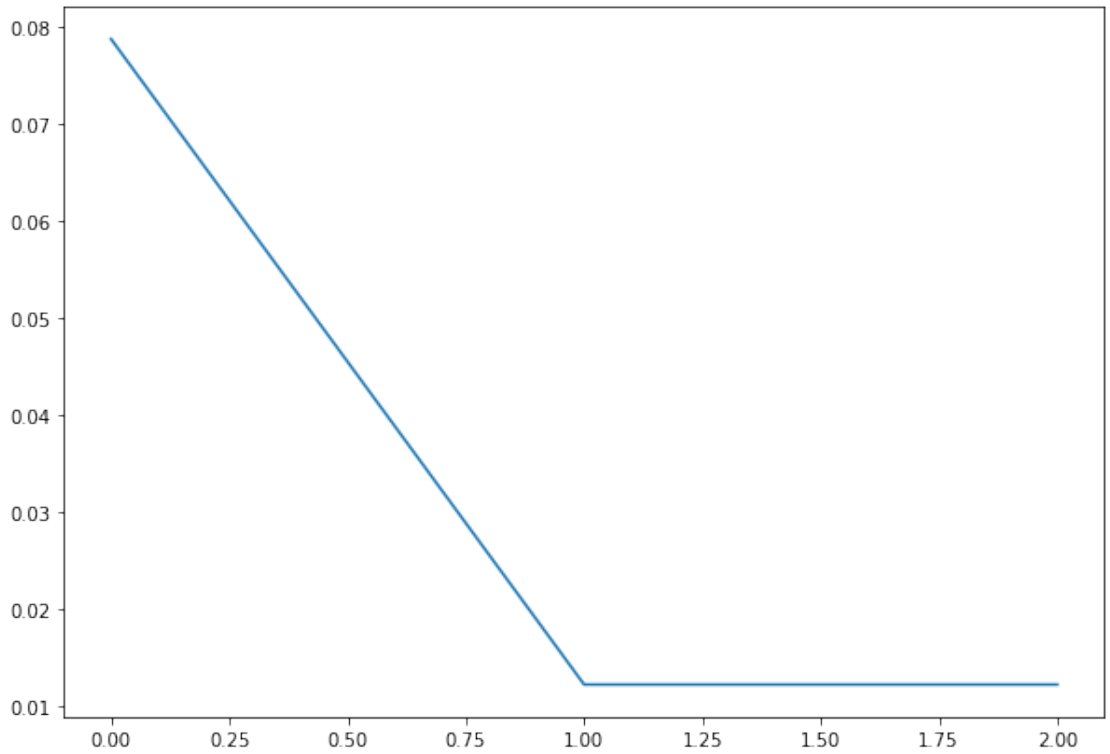
```
[:, 1], color=color_style[i%4][0], marker=color_style[i%4][1])

    plt.scatter(mean_vectors[:, 0], mean_vectors[:, 1], color='black',
marker='x', s=300)
    plt.title(f'Iteration: {iter}')

    sum_error = 0

    for i in range(K): sum_error +=
np.sum(np.linalg.norm(assigned_clusters[i]-mean_vectors[i], axis=1))

    sum_error
    delta = abs(prev_error - sum_error/data_points.shape[0])
    prev_error = sum_error/data_points.shape[0]

    prev_error
    errors.append(prev_error)
    iter+=1

plt.figure(figsize=(10, 7))
plt.plot(range(len(errors)), errors)
```

```
[<matplotlib.lines.Line2D at 0x1d12d9f1210>]
```




Iteration: 1
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4


Iteration: 2
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

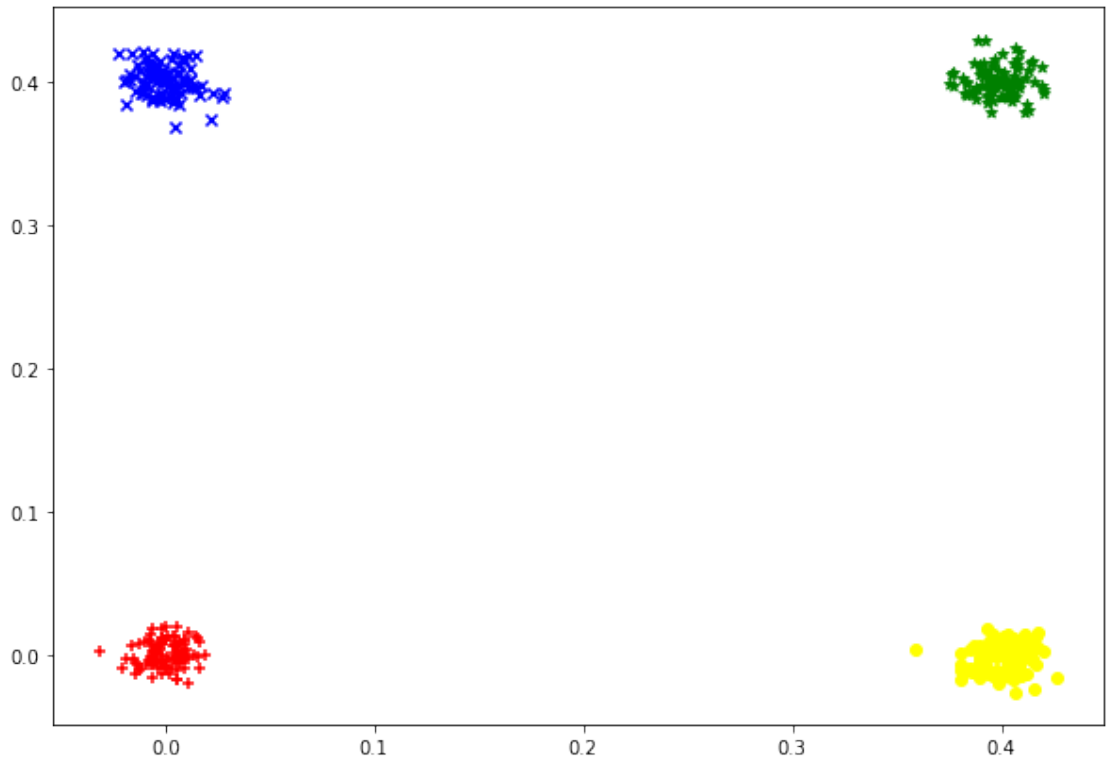



```
from sklearn.metrics.cluster import homogeneity_score,
silhouette_score

# Homogeneity Score
labels_true = []
labels_predicted = []

for i in range(K):
    for j in range(len(data[i])):
        labels_true.append(i)
        index = np.argmin(np.linalg.norm(data[i][j] - mean_vectors,
axis=1))
        labels_predicted.append(index)

h_score = homogeneity_score(labels_true, labels_predicted)

print(f'Homogeneity score for the classification is: {h_score}')

# Silhouette coefficient
all_points = np.concatenate(assigned_clusters, axis=0)
N = len(all_points)

pairwise = [[0 for i in range(N)] for j in range(N)]
for i in range(N):
    for j in range(N):
        pairwise[i][j] = np.linalg.norm(np.subtract(all_points[i],
```

```
all_points[j]))

labels=[]
for id, cluster in enumerate(assigned_clusters):
    labels += [id for i in range(len(cluster))]

sc_score = silhouette_score(pairwise, labels)
print(f'Silhouette coefficient for the classification is: {sc_score}')
```

```
Homogeneity score for the classification is: 1.0
Silhouette coefficient for the classification is: 0.9596128809111198
```

```
# write your code here
K = 4

data_sep = []
means = [(0, 0), (0.4, 0.4), (0, 0.4), (0.4, 0)]
color_style = [('red', '+'), ('green', '*'), ('blue', 'x'), ('yellow',
'o')]

plt.figure(figsize=(10, 7))

for i in range(K):
    d = np.random.multivariate_normal(mean=means[i],
cov=np.identity(2)*0.0001, size=100)
    plt.scatter(d[:, 0], d[:, 1], color=color_style[i%4][0],
marker=color_style[i%4][1])
    data_sep.append(d)

data = np.concatenate(data_sep, axis=0)
```

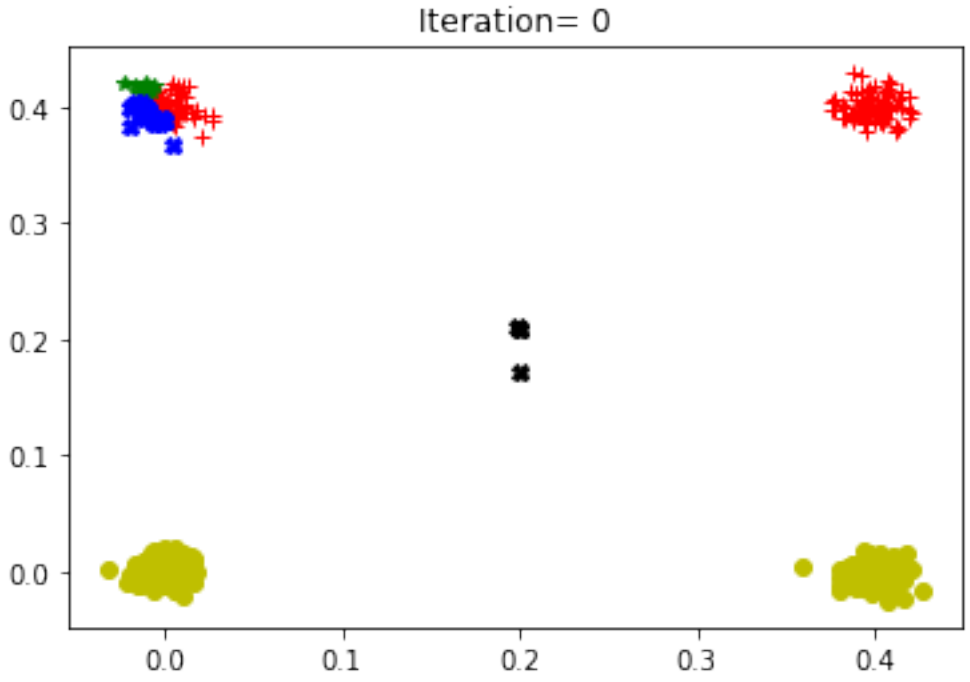


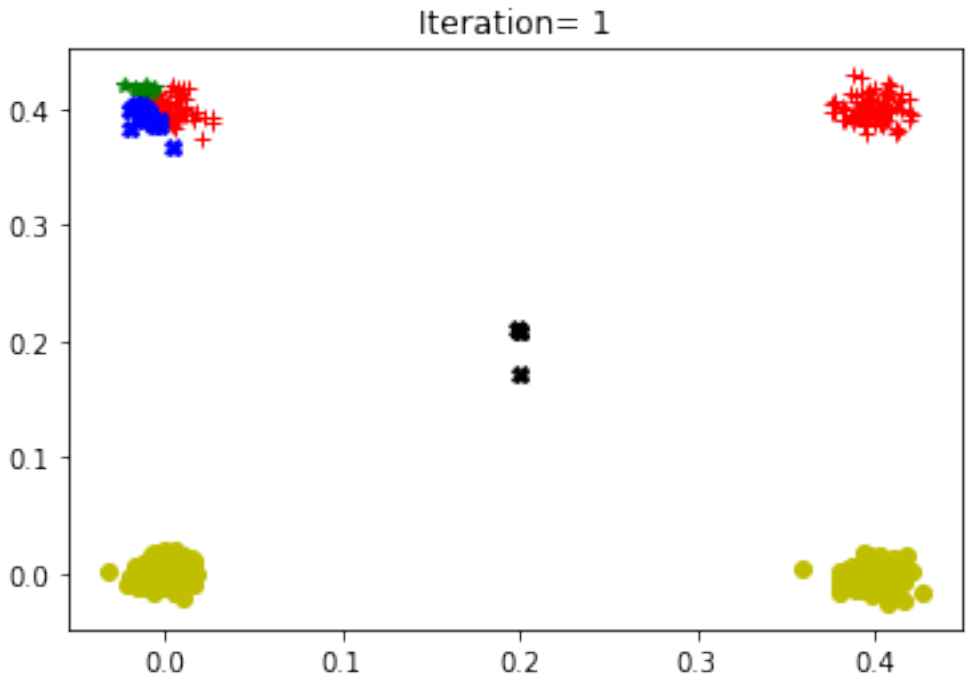

```
log_l=[]
Itr=50
eps=10**(-14)  # for threshold
clr=['r','g','b','y','k','m','c']
mrk=['+','*','X','o','.','<','p']


K = 4   # no. of clusters

theta=initialization(data,K)

for n in range(Itr):

  responsibility=E_Step_GMM(data,K,theta)

  cluster_label=np.argmax(responsibility,axis=1) #Label Points

  theta,log_likhd=M_Step_GMM(data,responsibility)


  log_l.append(log_likhd)

  plt.figure()
  for l in range(K):
    id=np.where(cluster_label==l)
    plt.plot(data[id,0],data[id,1],'.',color=clr[l],marker=mrk[l])
```

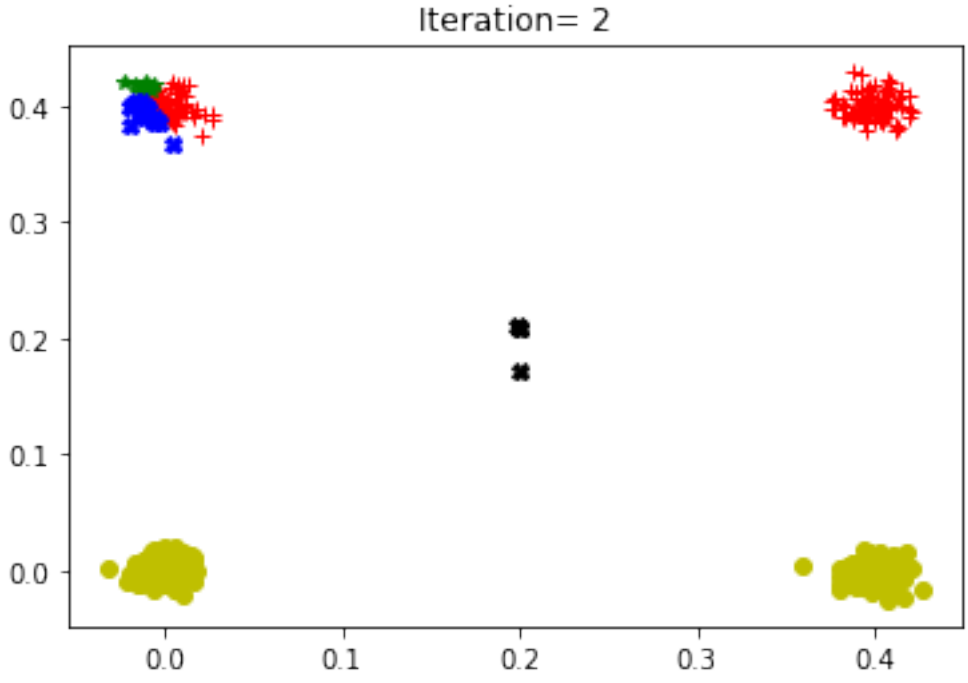
```
  Cents=theta[0]
  plt.plot(Cents[:,0],Cents[:,1],'X',color='k')
  plt.title('Iteration= %d' % (n))

  if n>2:
    if abs(log_l[n]-log_l[n-1])<eps:
      break


plt.figure()
plt.plot(log_l)
```

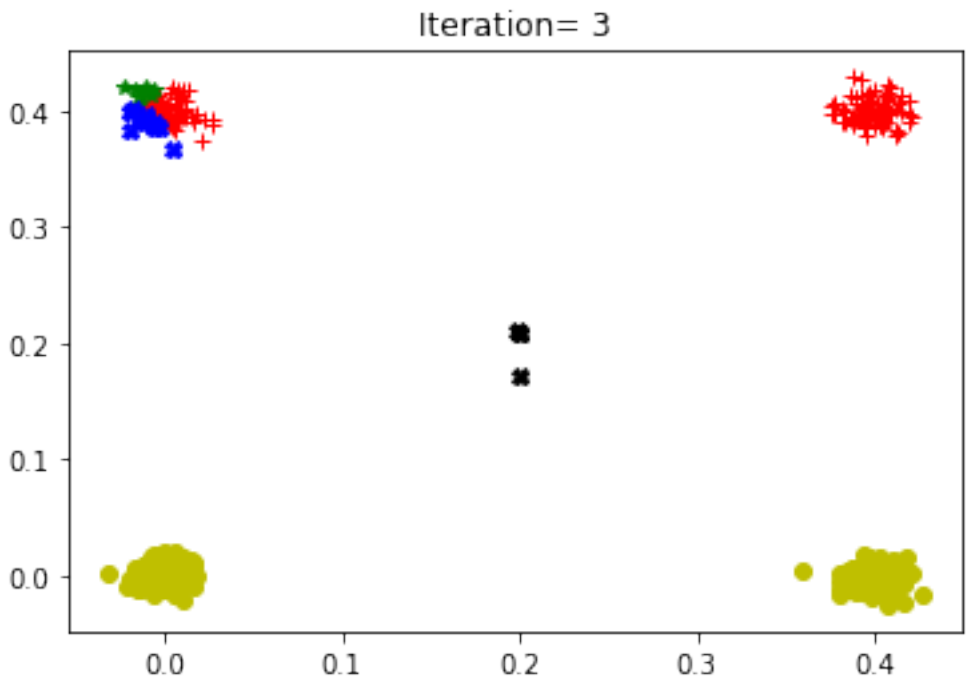
```
C:\Users\Shashank\AppData\Local\Temp\ipykernel_12660\3052499598.py:26:
UserWarning: marker is redundantly defined by the 'marker' keyword
argument and the fmt string "." (-> marker='.'). The keyword argument
will take precedence.
  plt.plot(data[id,0],data[id,1],'.',color=clr[l],marker=mrk[l])
C:\Users\Shashank\AppData\Local\Temp\ipykernel_12660\3052499598.py:23:
RuntimeWarning: More than 20 figures have been opened. Figures created
through the pyplot interface (`matplotlib.pyplot.figure`) are retained
until explicitly closed and may consume too much memory. (To control
this warning, see the rcParam `figure.max_open_warning`).
  plt.figure()
```

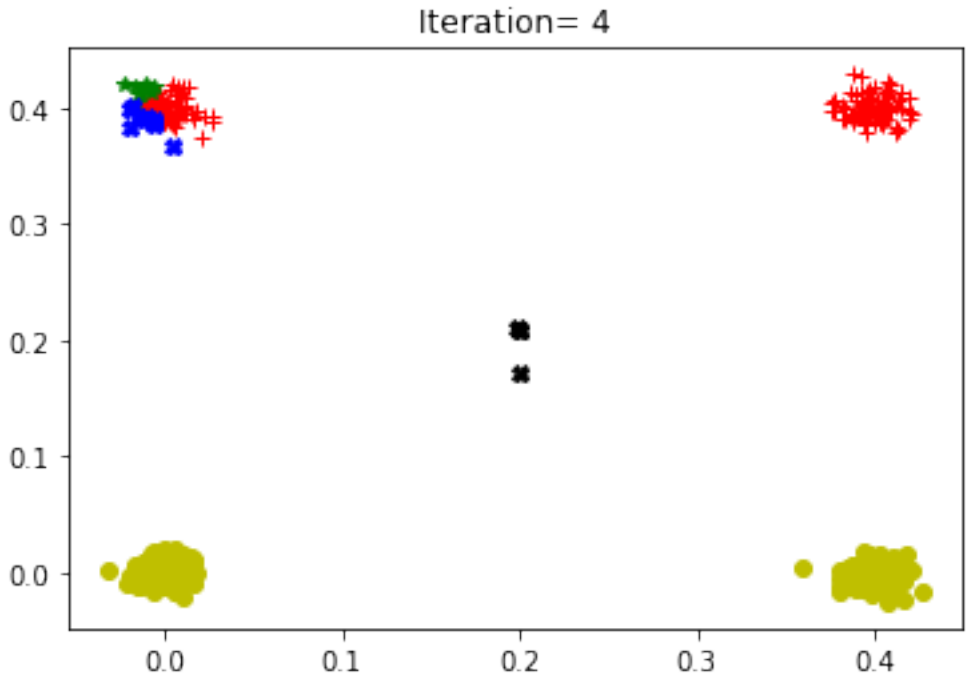
```
[<matplotlib.lines.Line2D at 0x1d1f3226920>]
```

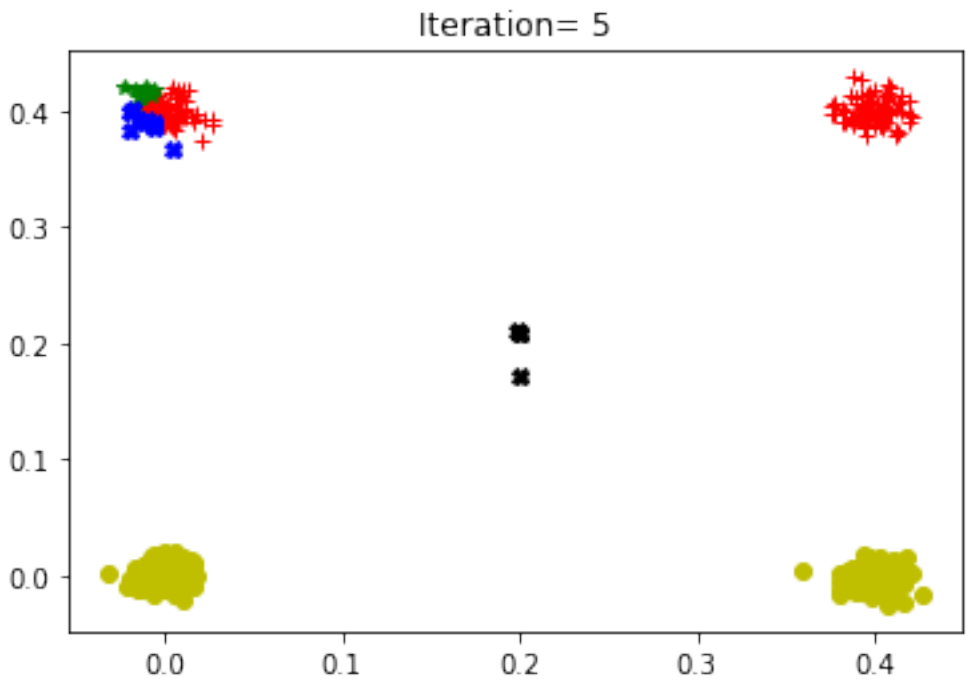


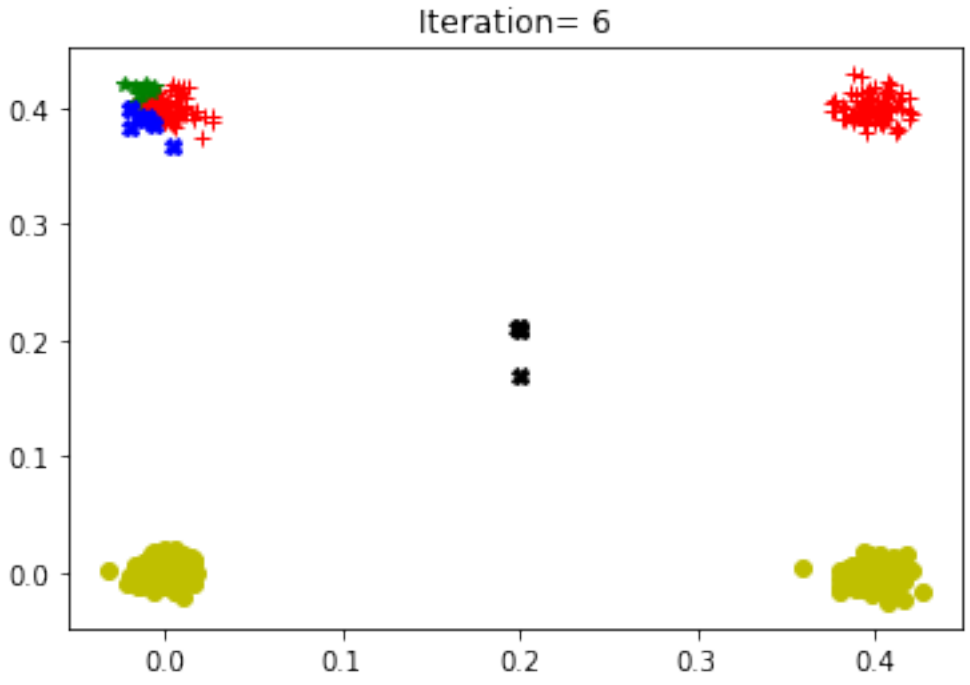

Iteration= 1
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

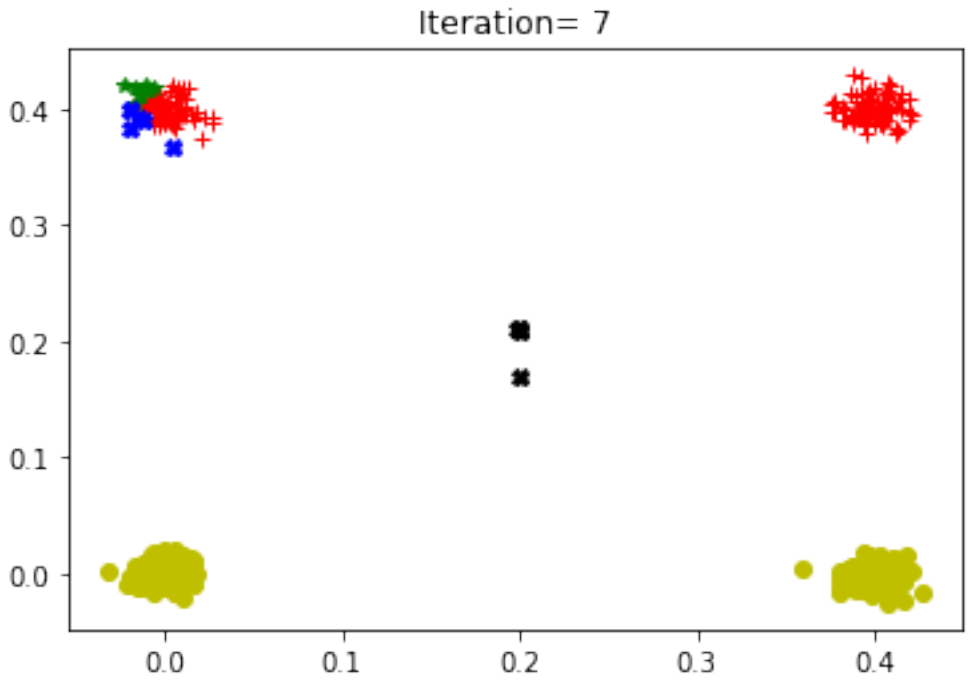

Iteration= 2
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

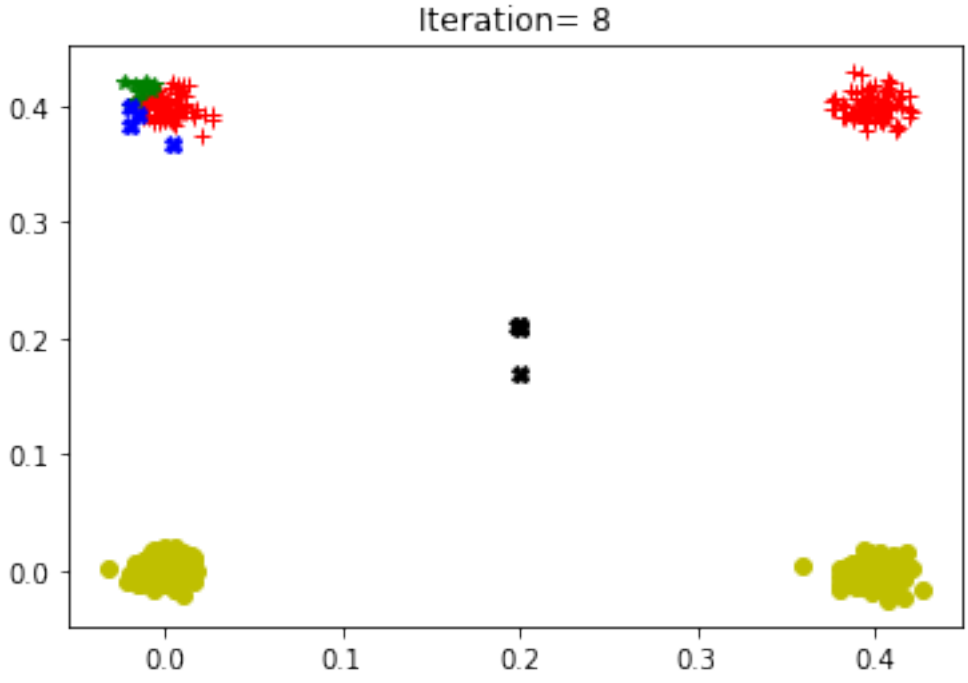

Iteration= 3
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

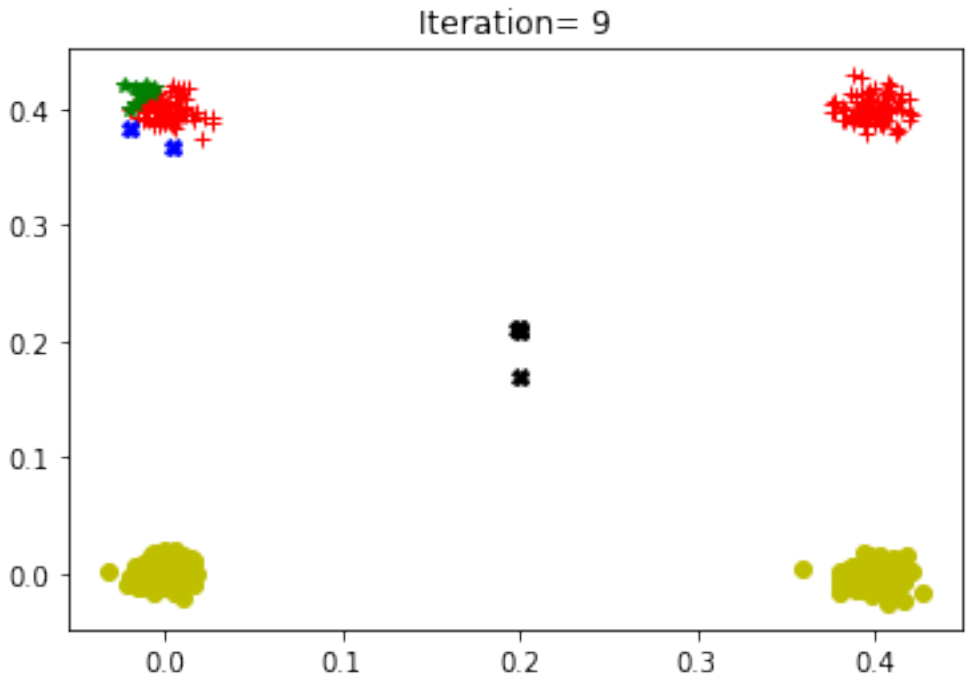

Iteration= 4
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

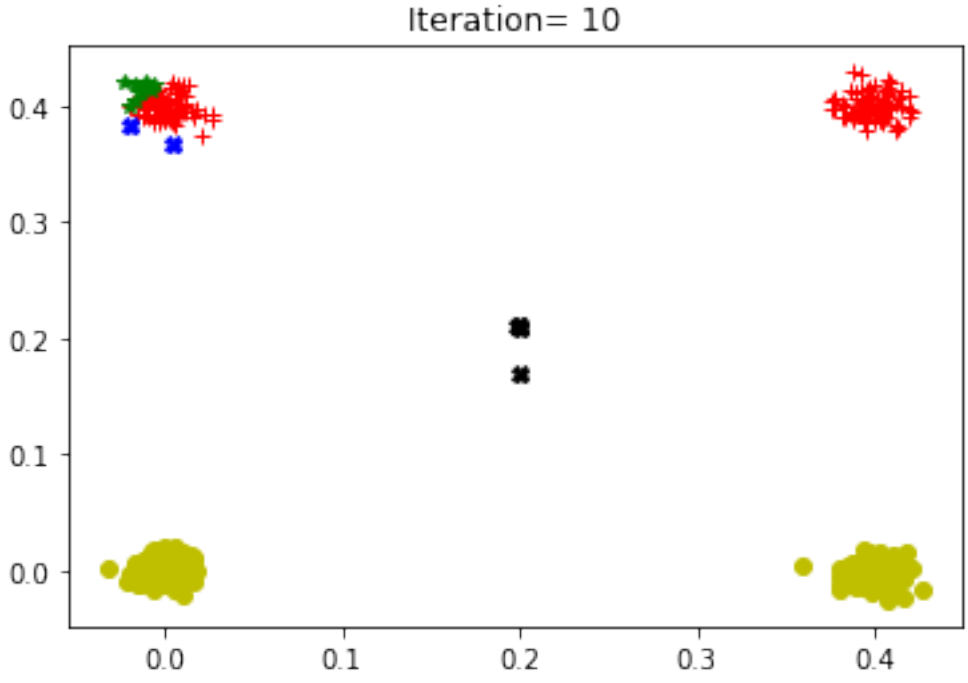

Iteration= 5
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

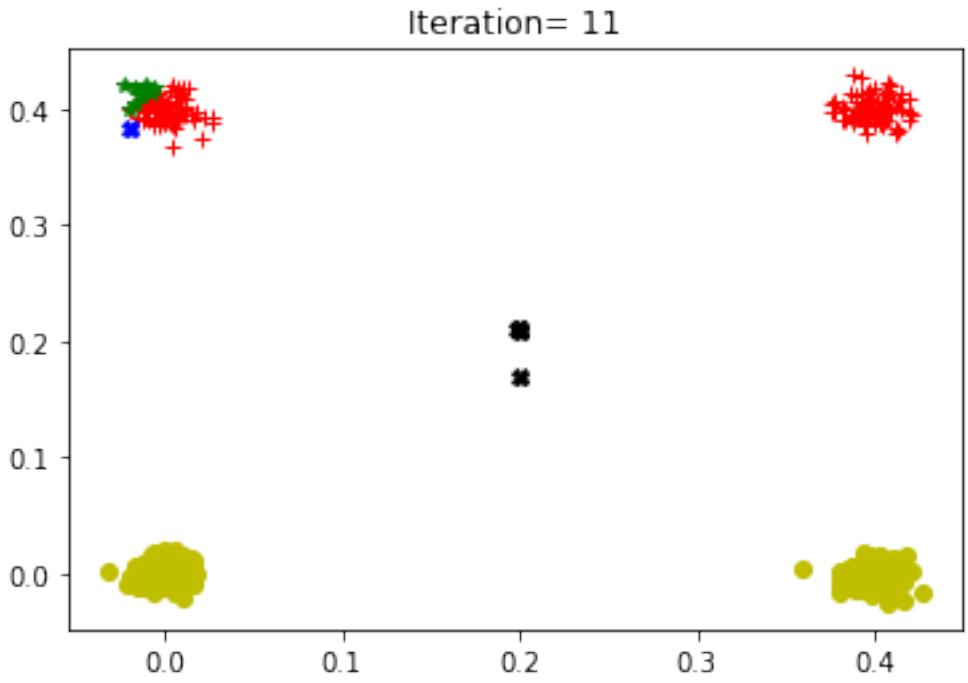

Iteration= 6
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

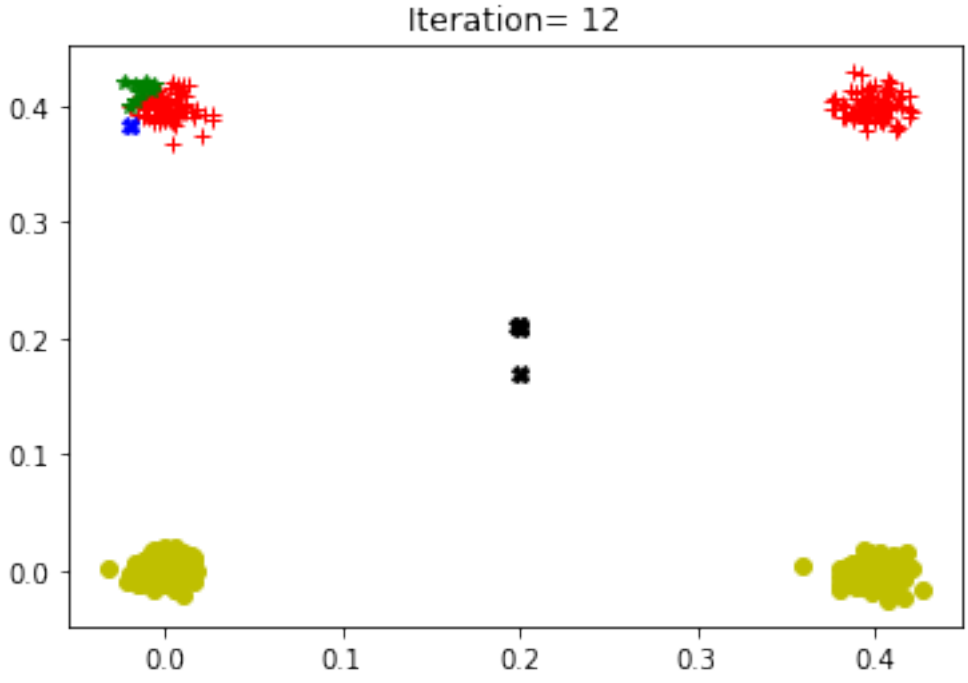

Iteration= 7
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

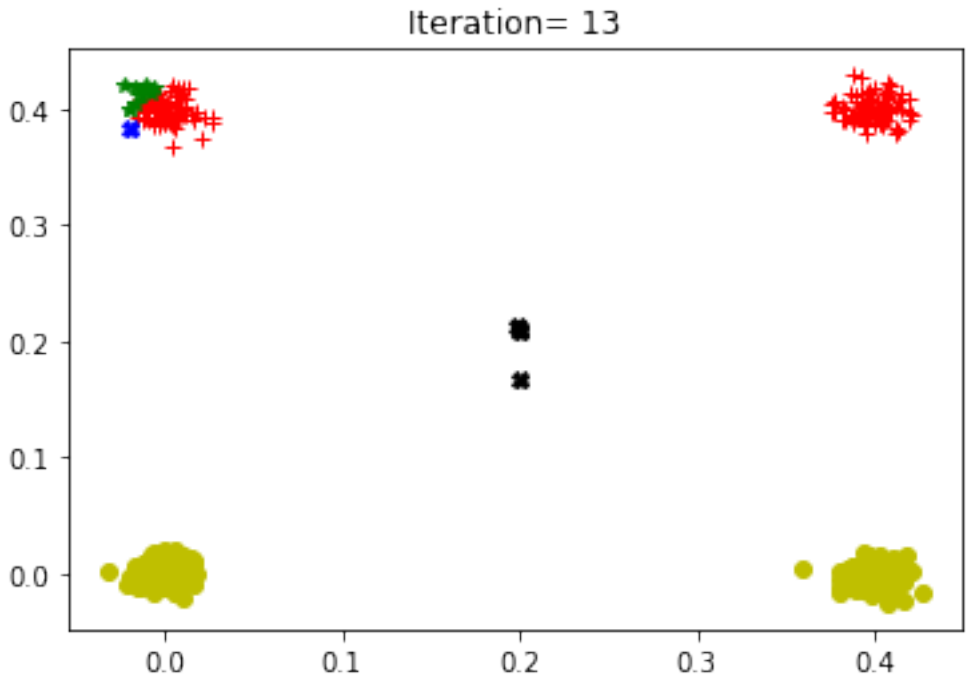

Iteration= 8
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

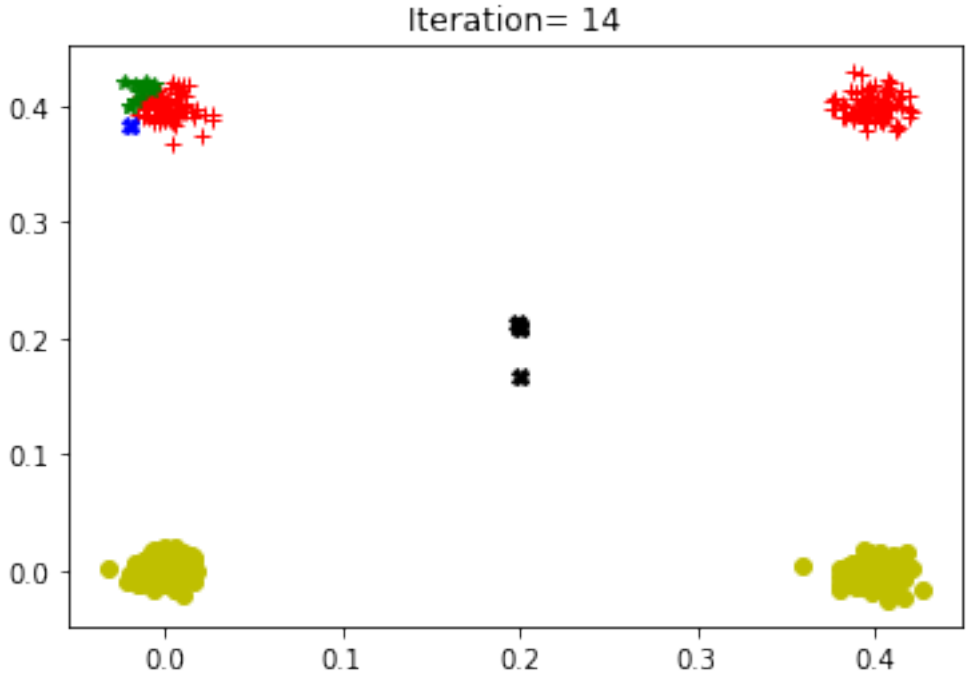

Iteration= 9
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

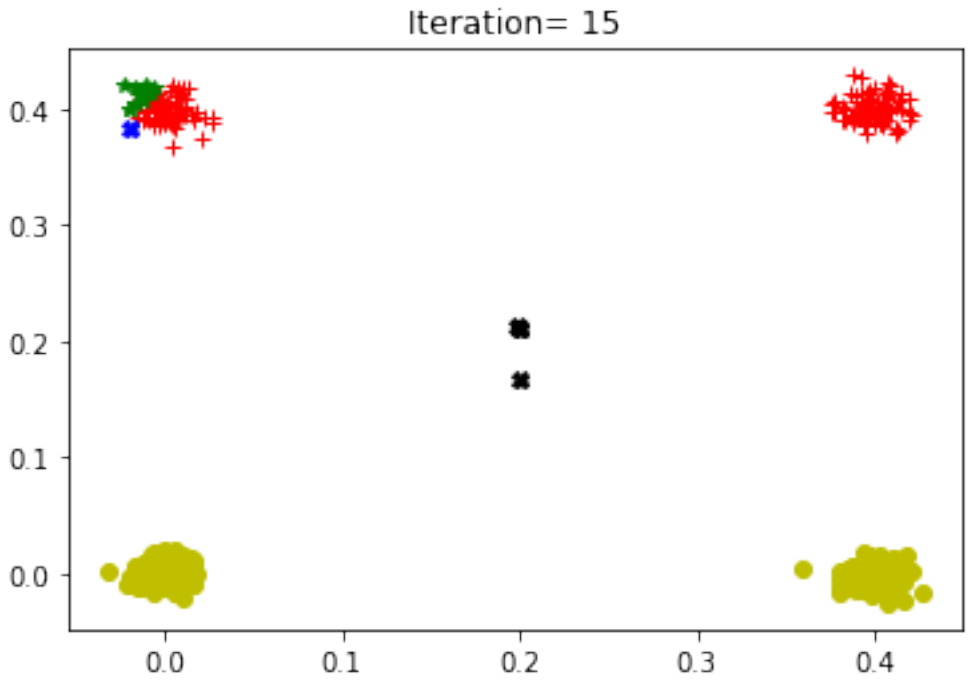

Iteration= 10
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

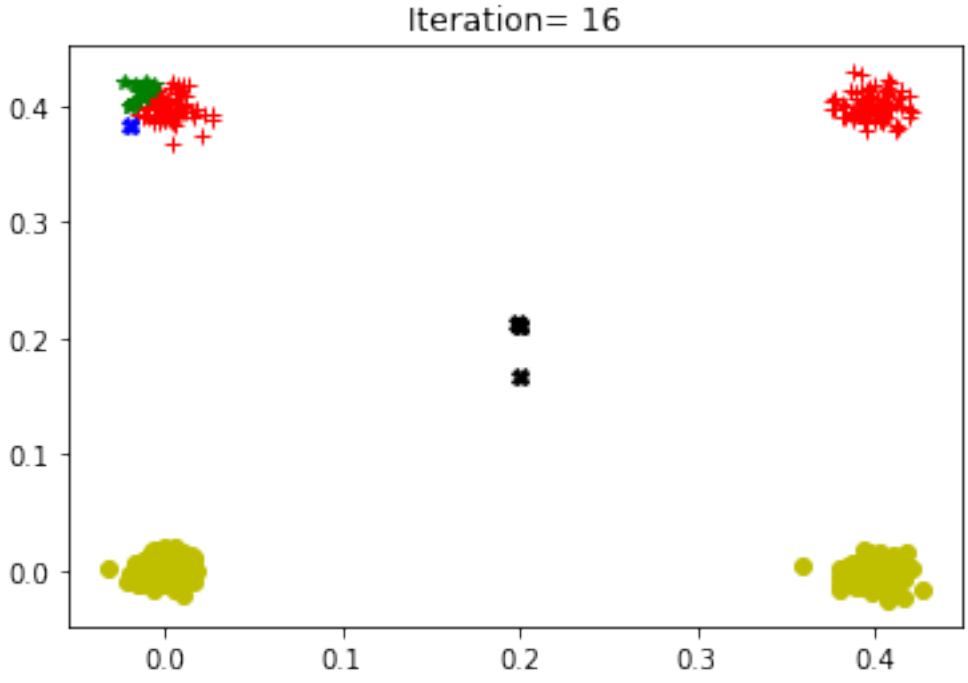

Iteration= 11
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

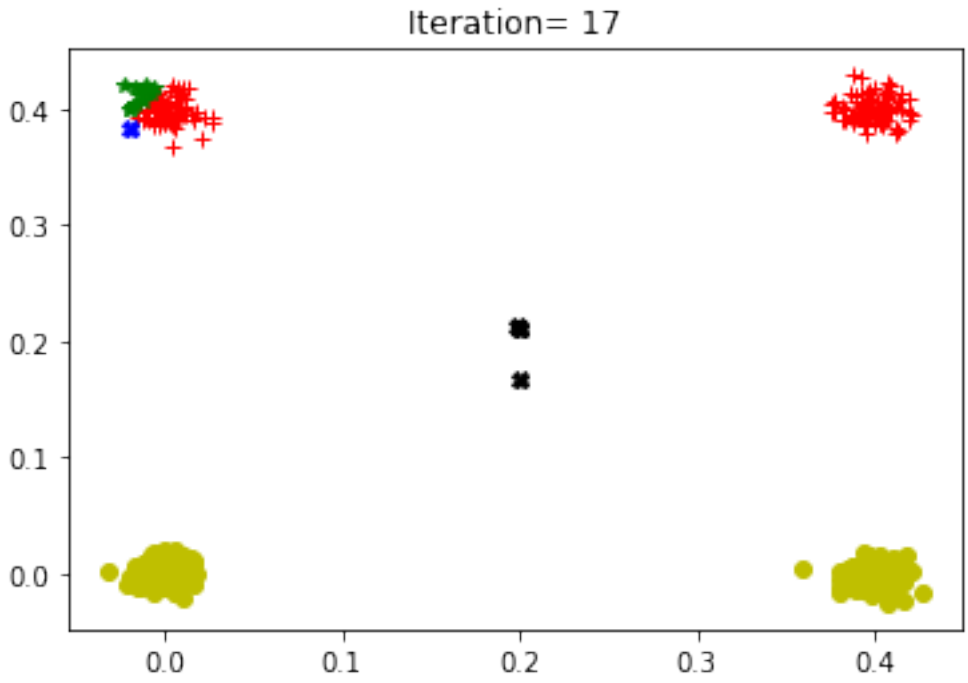

Iteration= 12
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

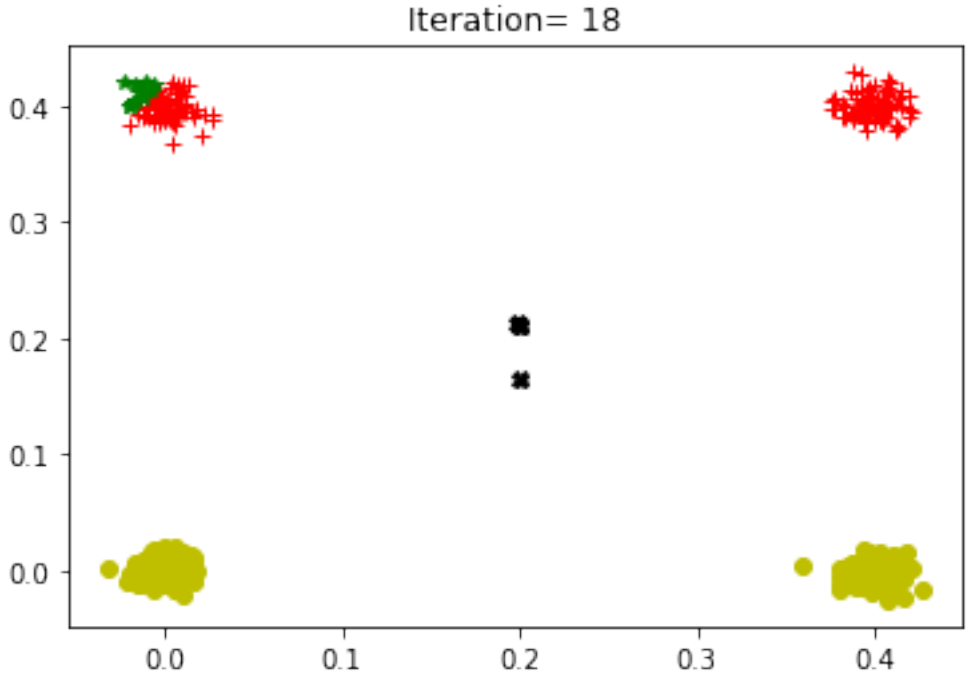

Iteration= 13
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

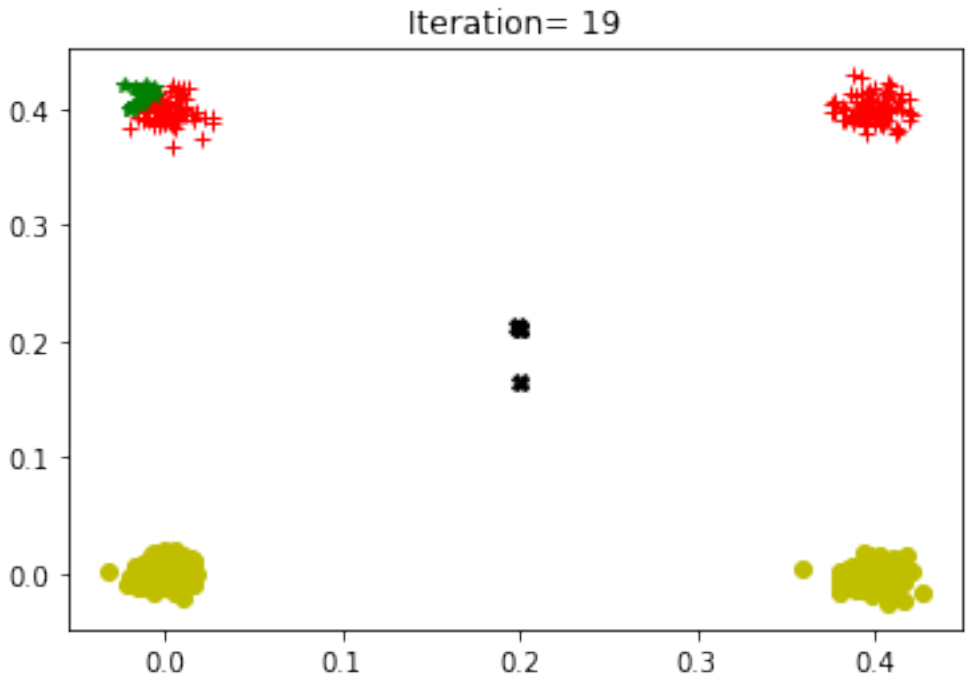

Iteration= 14
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

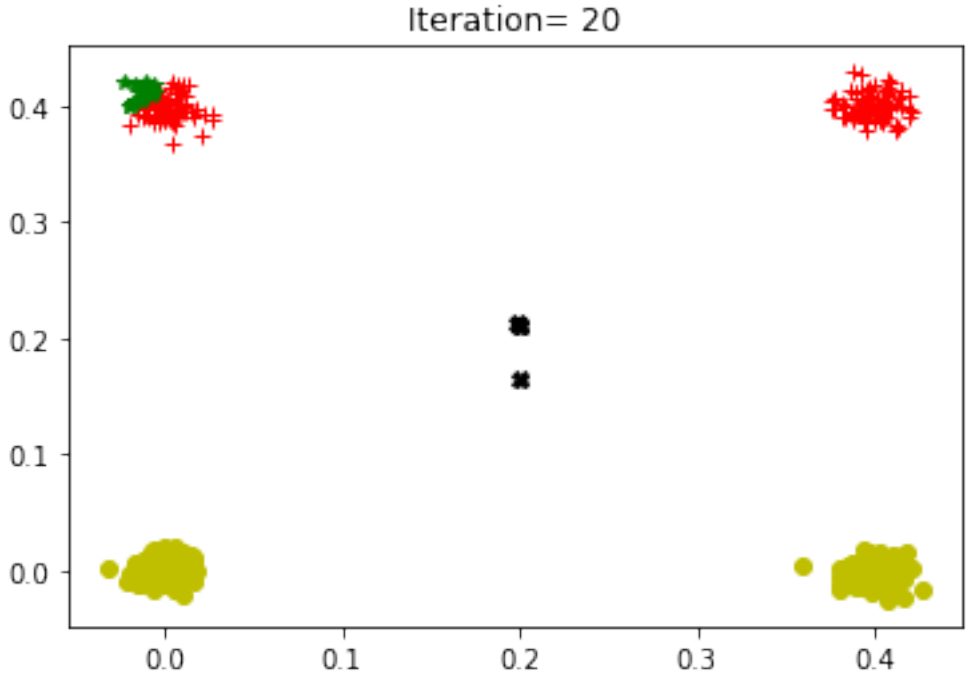

Iteration= 15
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

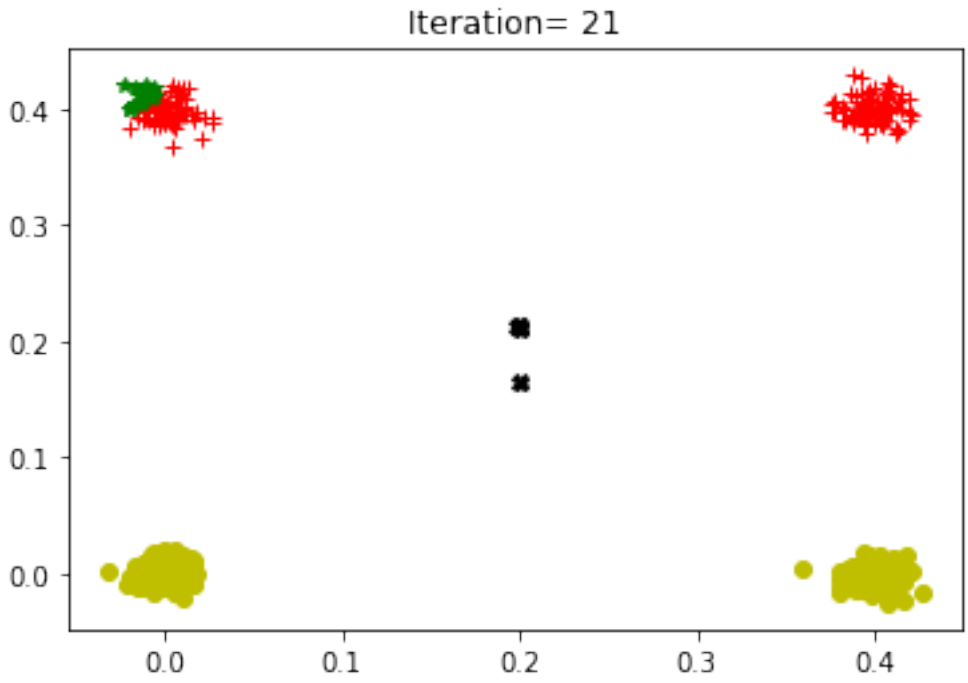

Iteration= 16
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

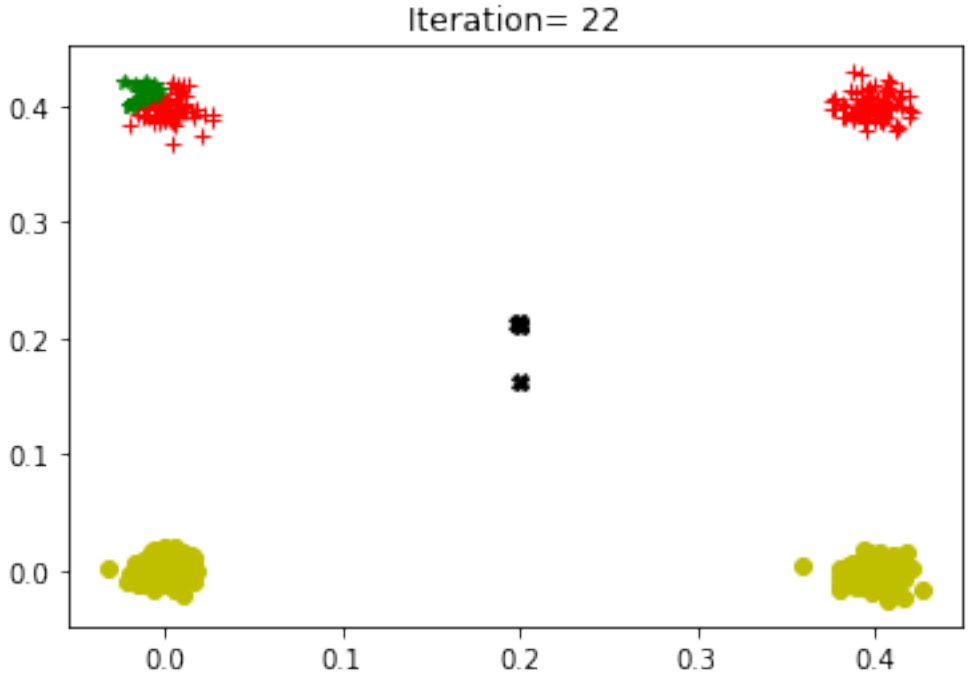

Iteration= 17
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

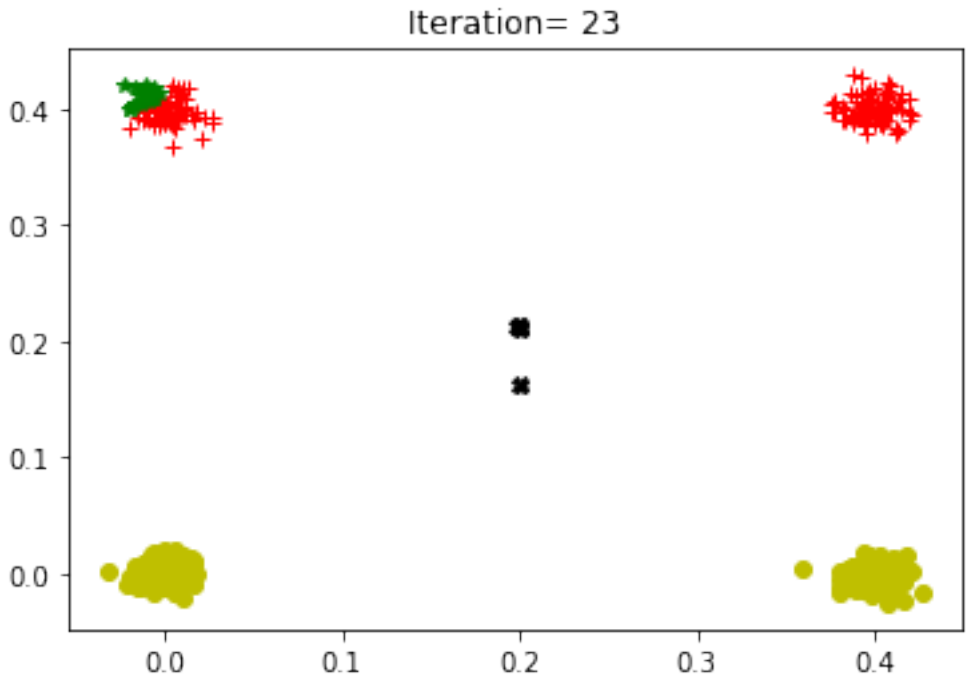

Iteration= 18
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

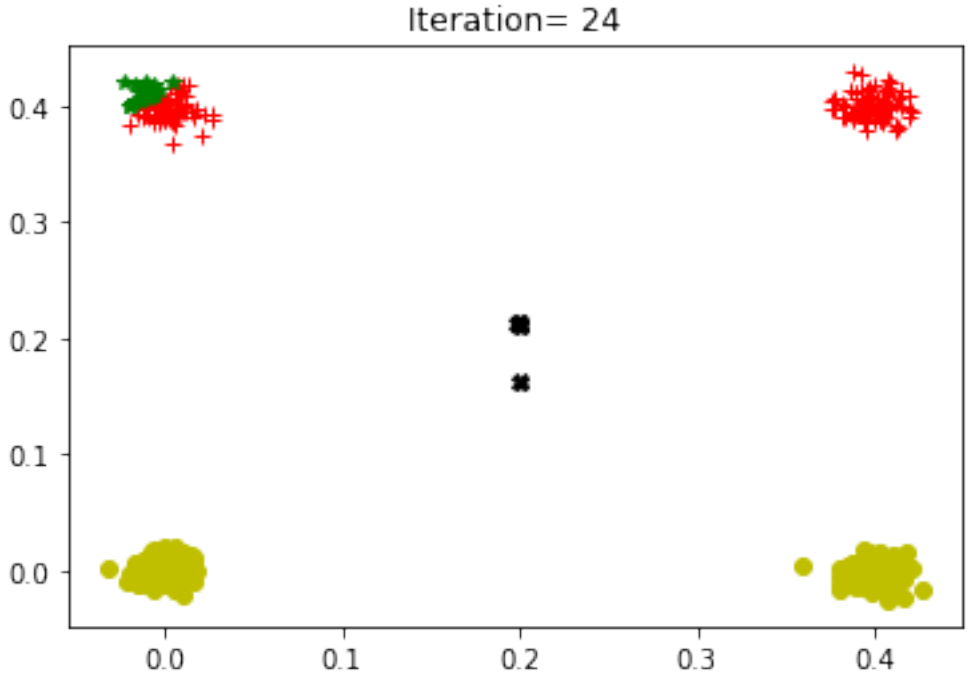

Iteration= 19
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

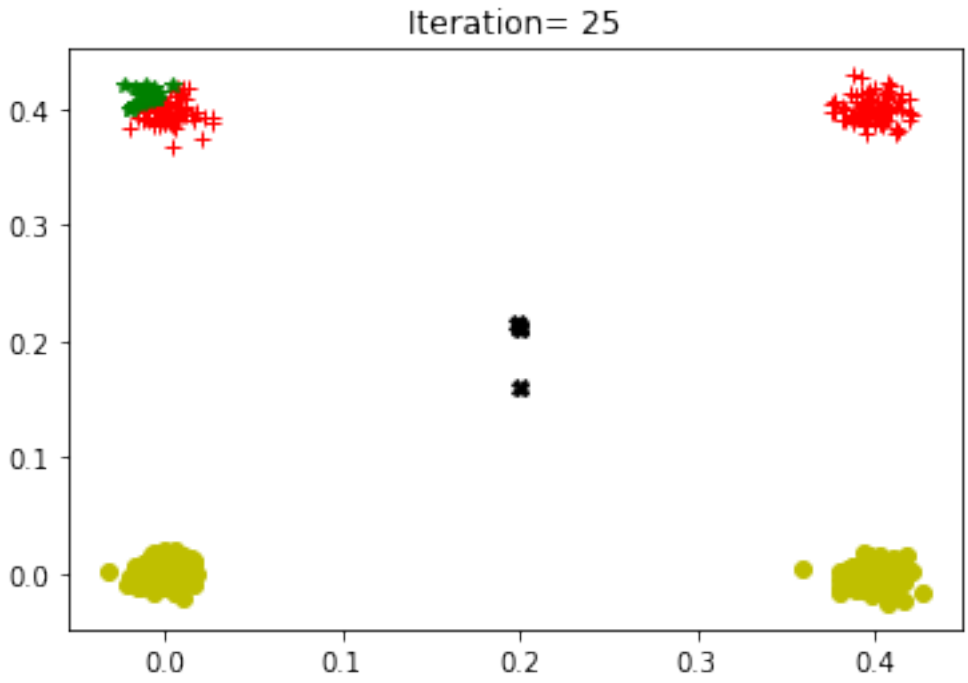

Iteration= 20
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

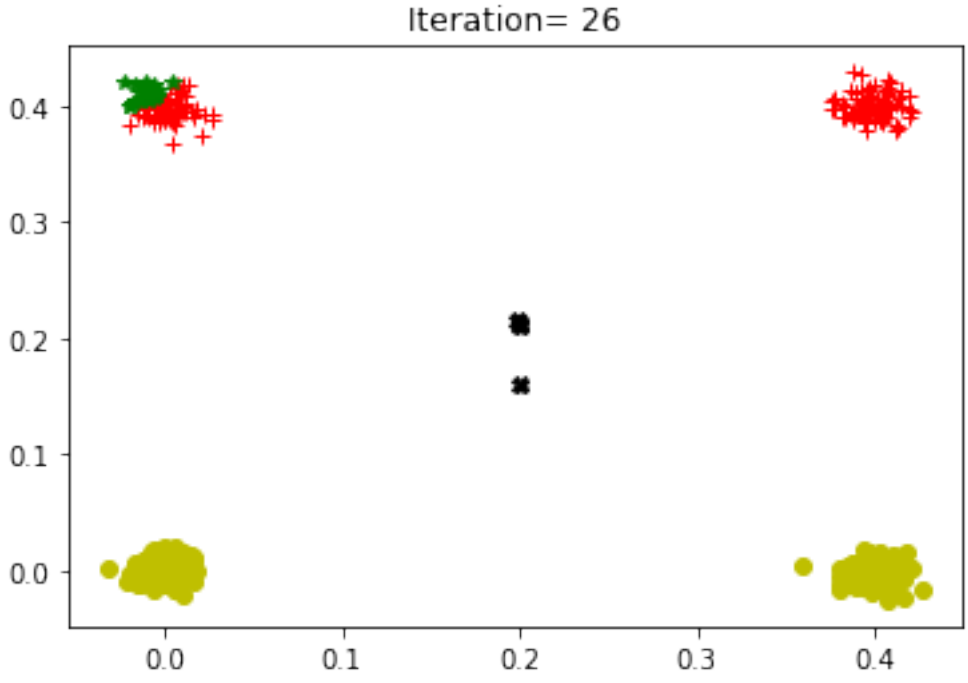

Iteration= 21
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

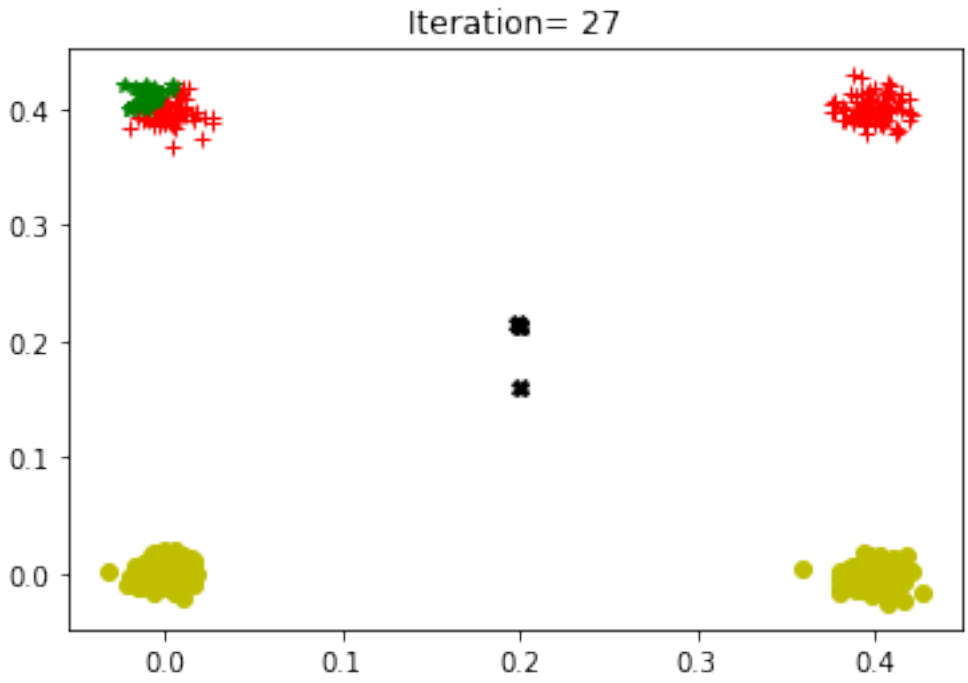

Iteration= 22
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

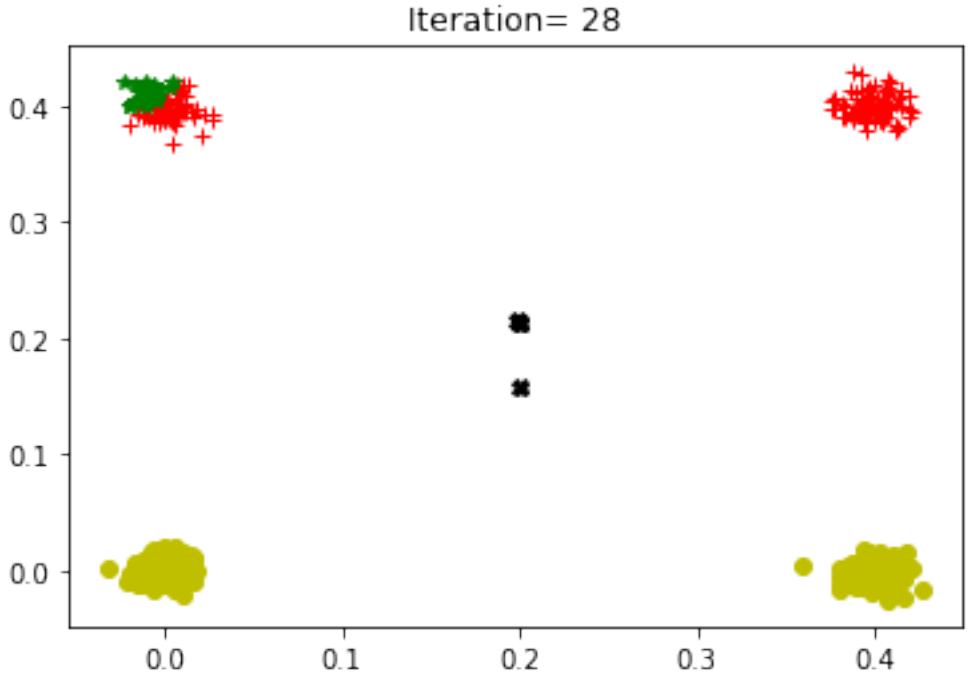

Iteration= 23
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

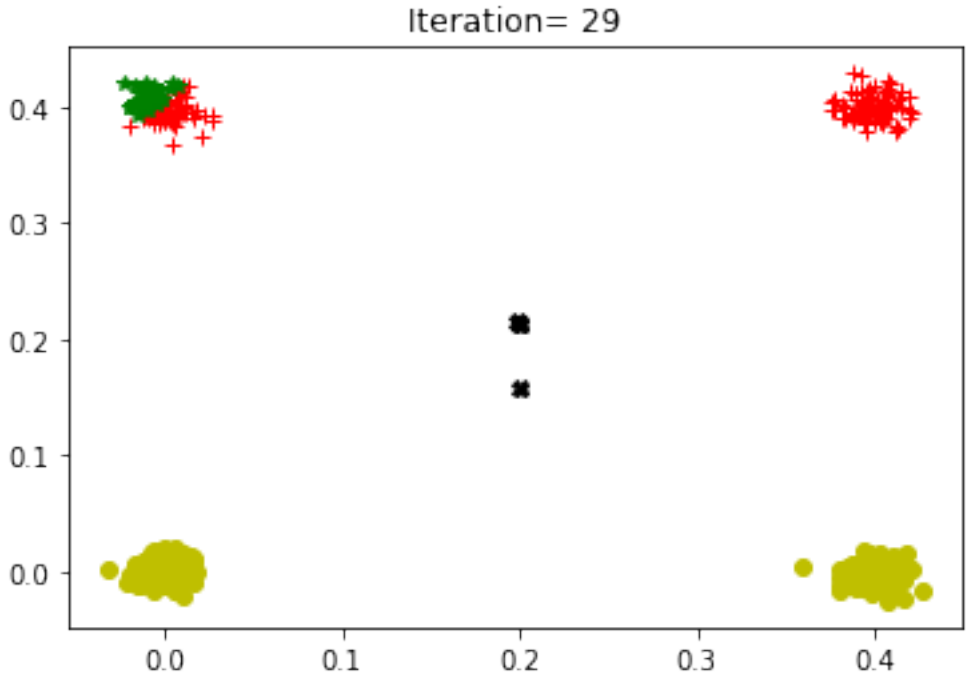

Iteration= 24
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

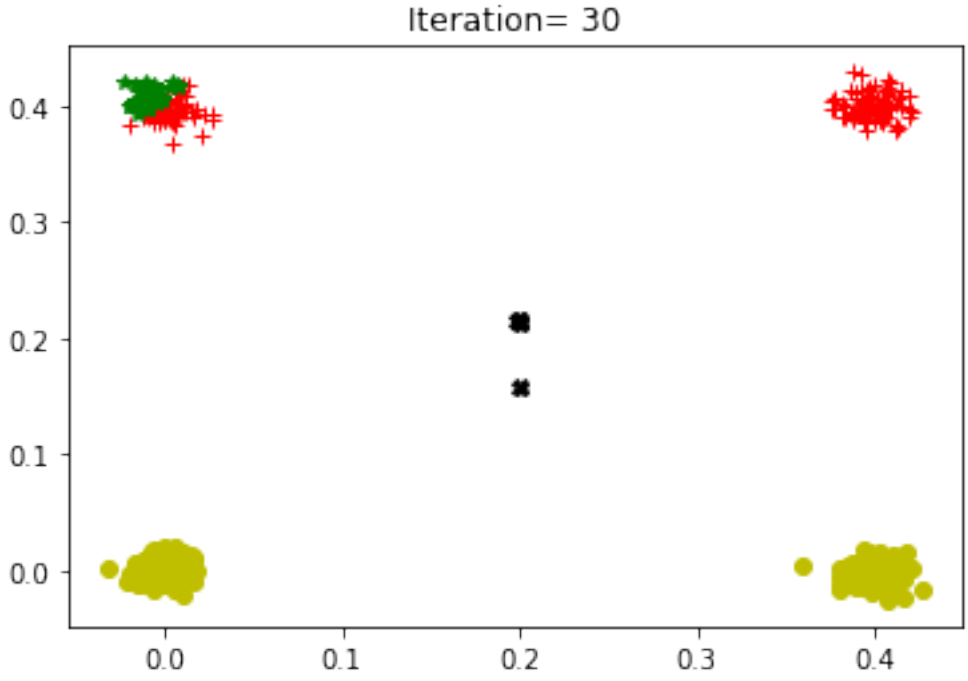

Iteration= 25
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

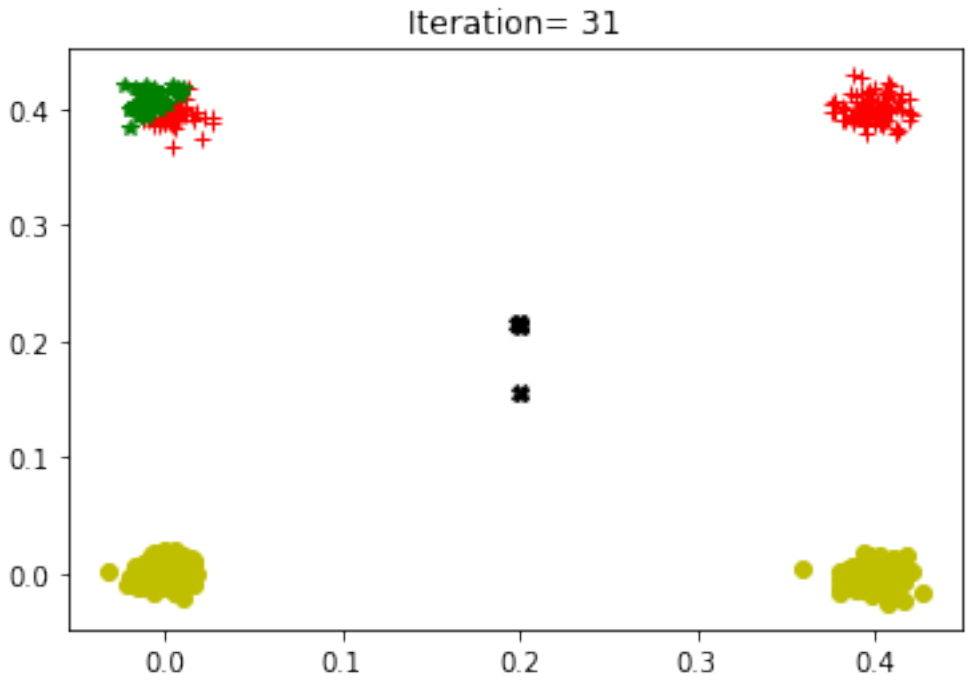

Iteration= 26
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

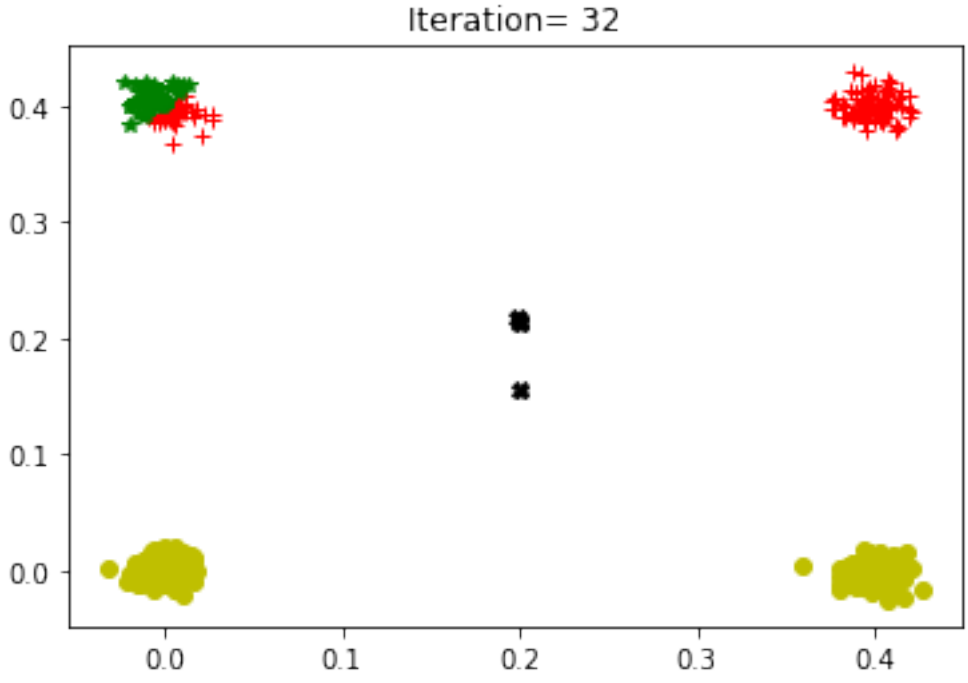

Iteration= 27
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

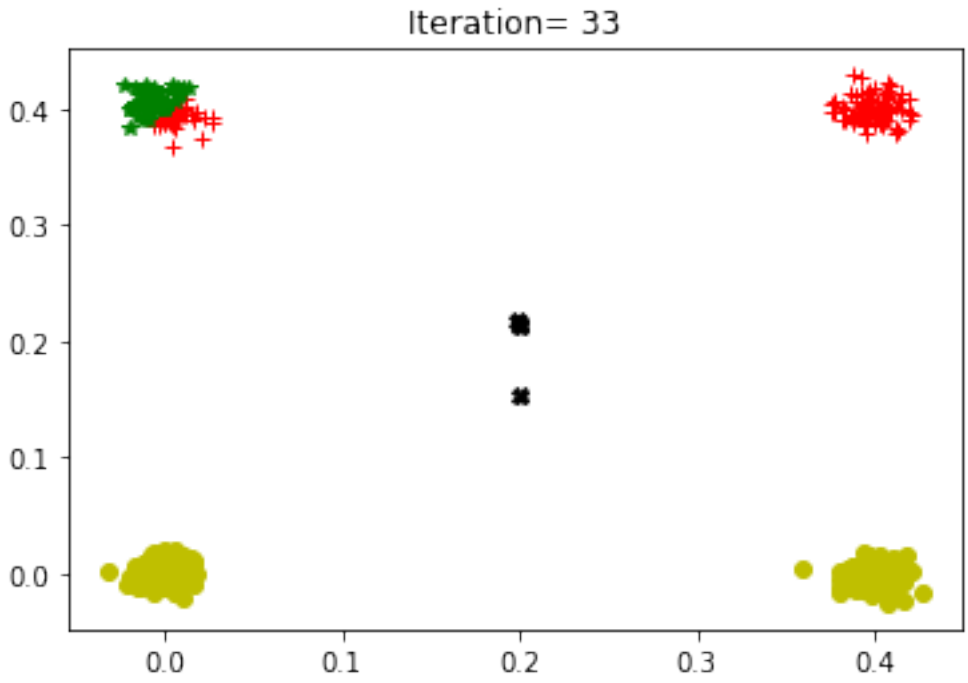

Iteration= 28
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

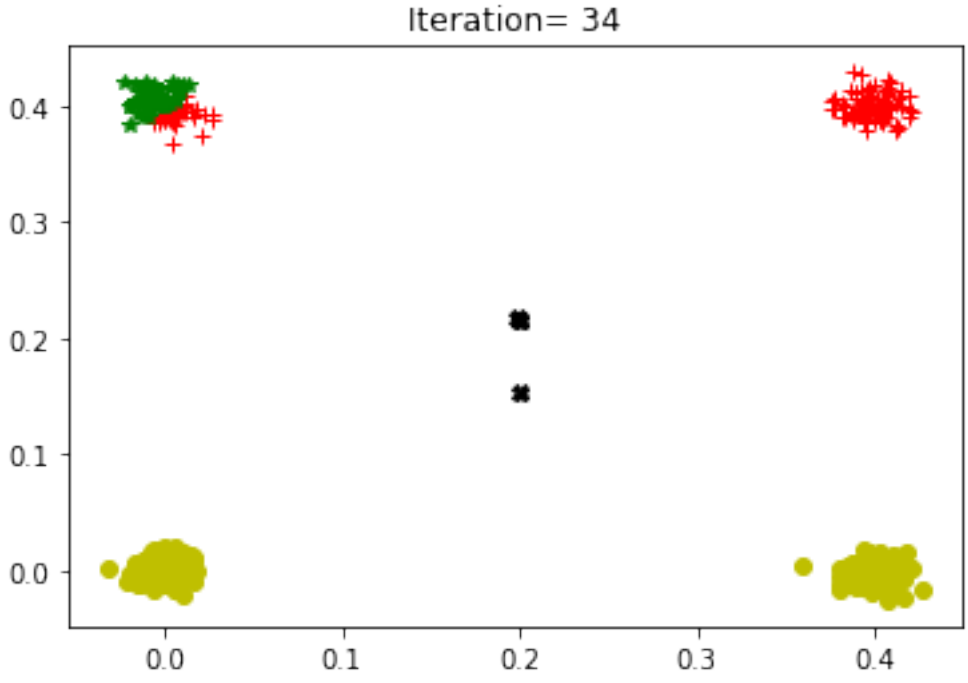

Iteration= 29
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

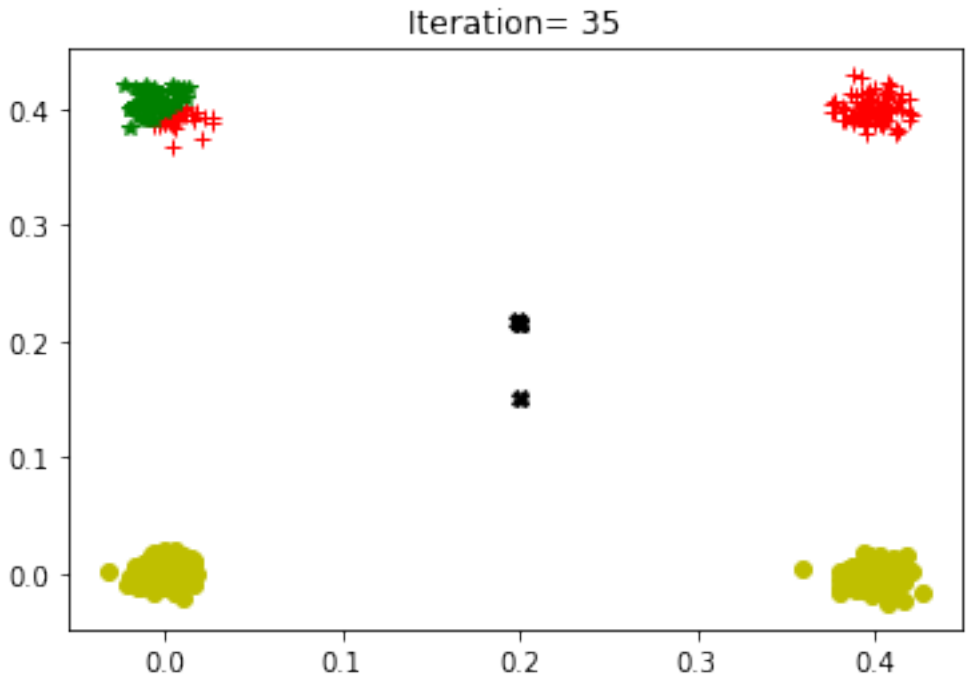

Iteration= 30
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

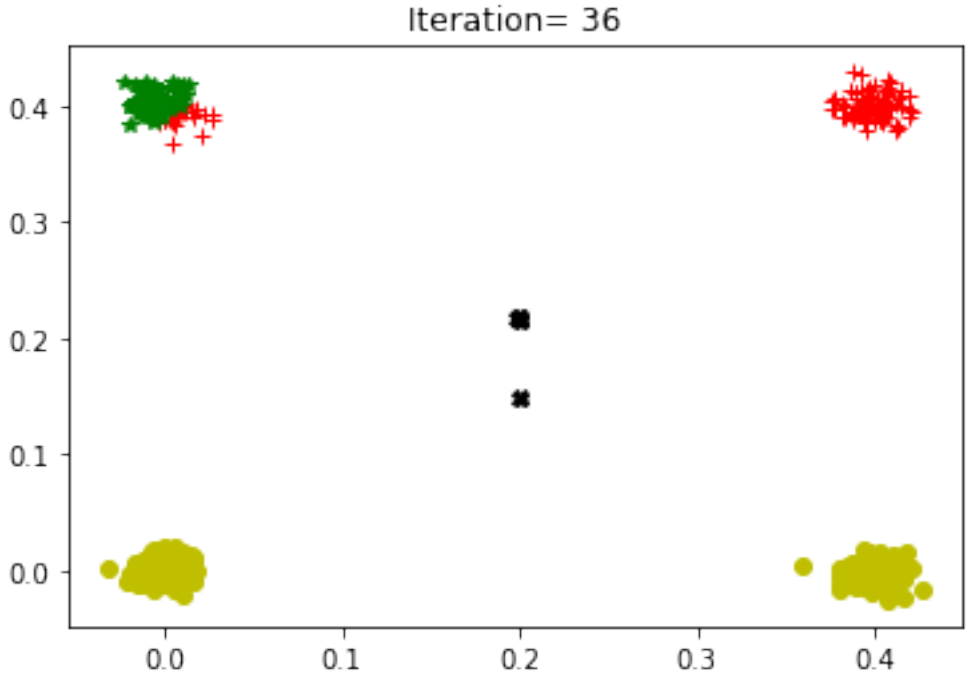

Iteration= 31
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

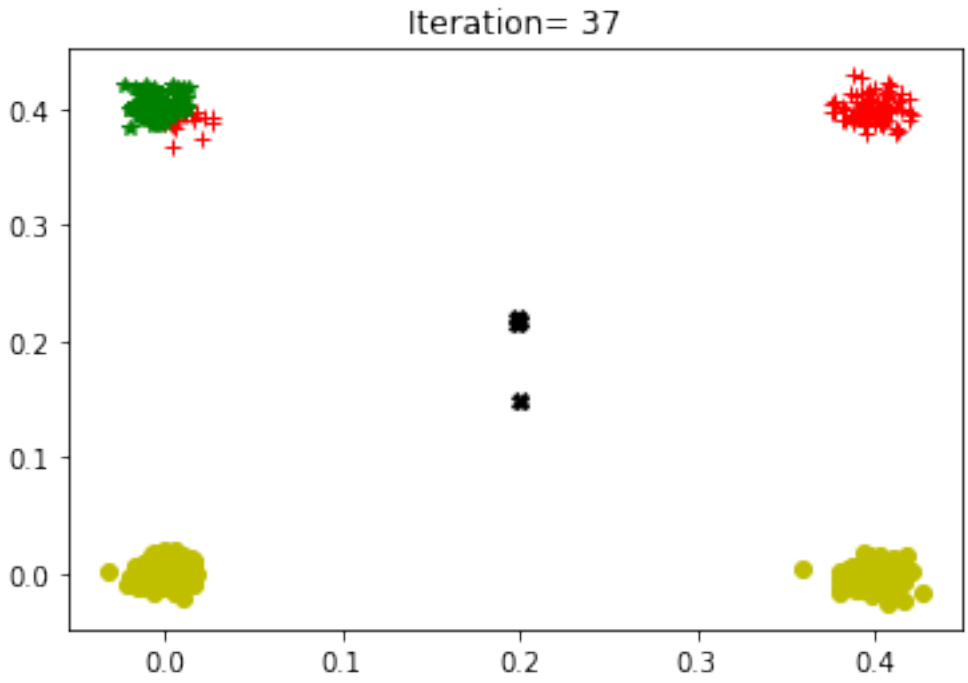

Iteration= 32
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

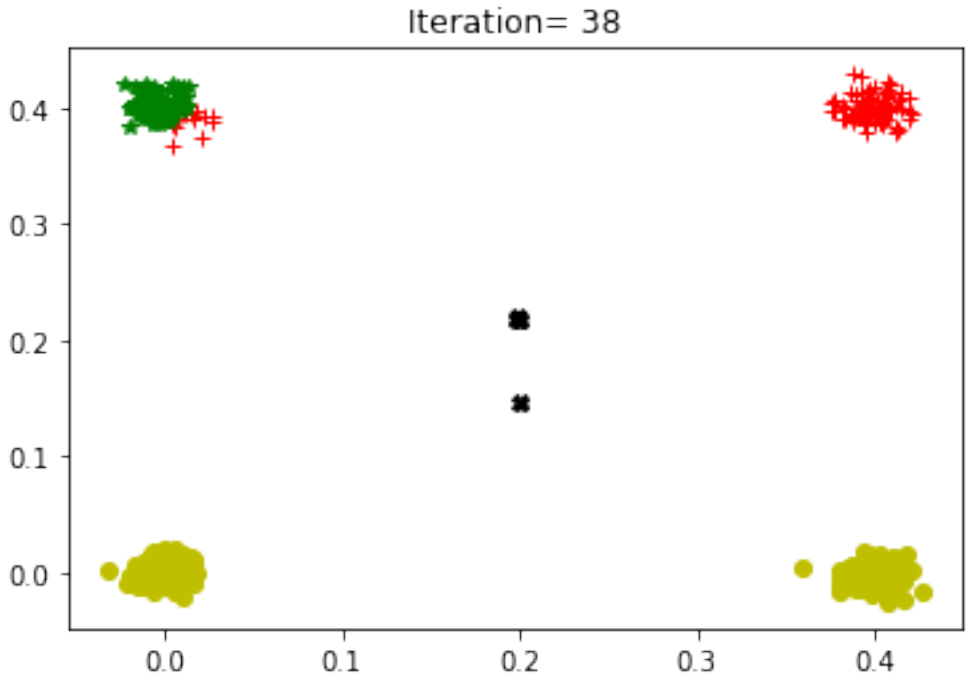

Iteration= 33
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

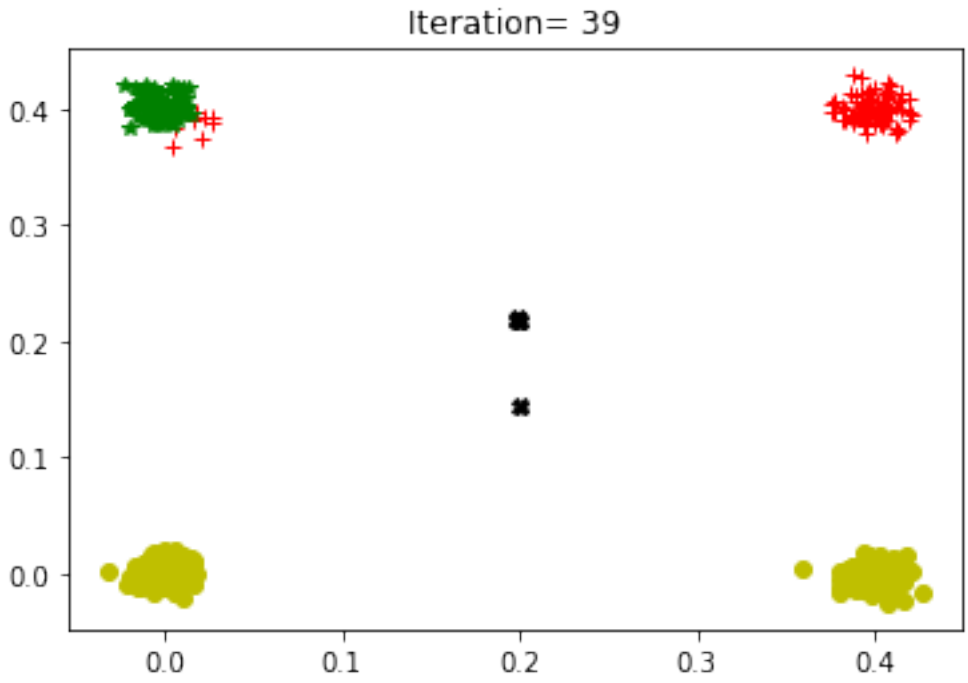

Iteration= 34
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

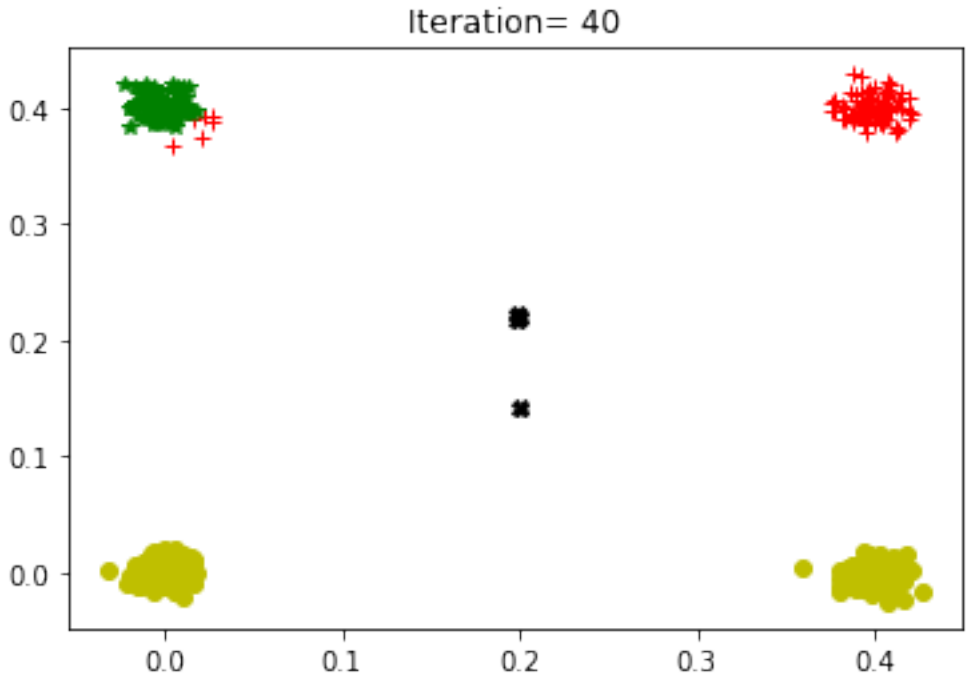

Iteration= 35
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

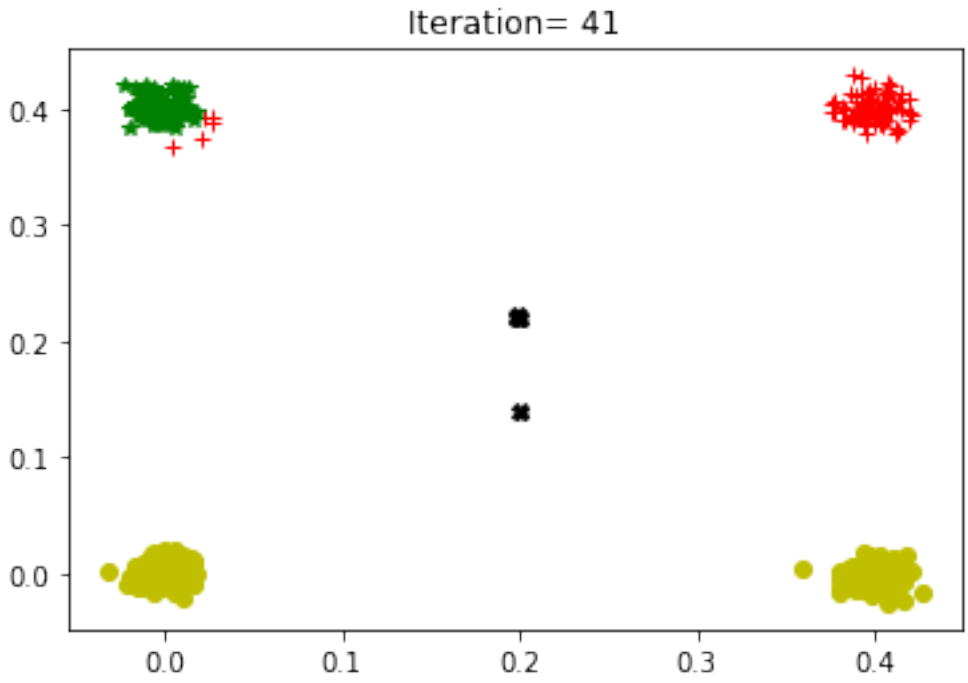

Iteration= 36
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

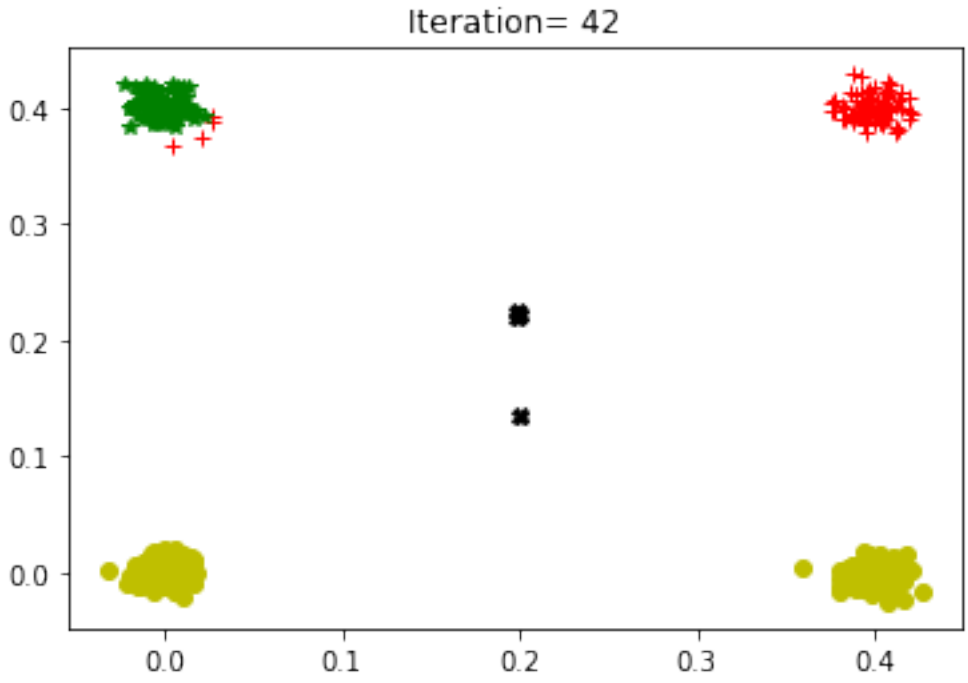

Iteration= 37
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

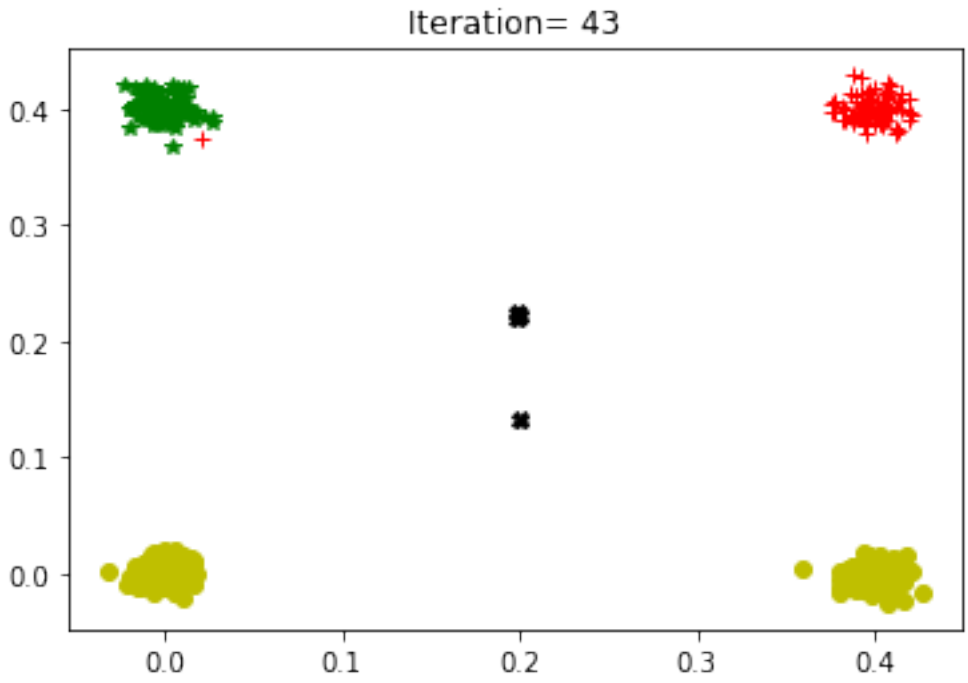

Iteration= 38
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

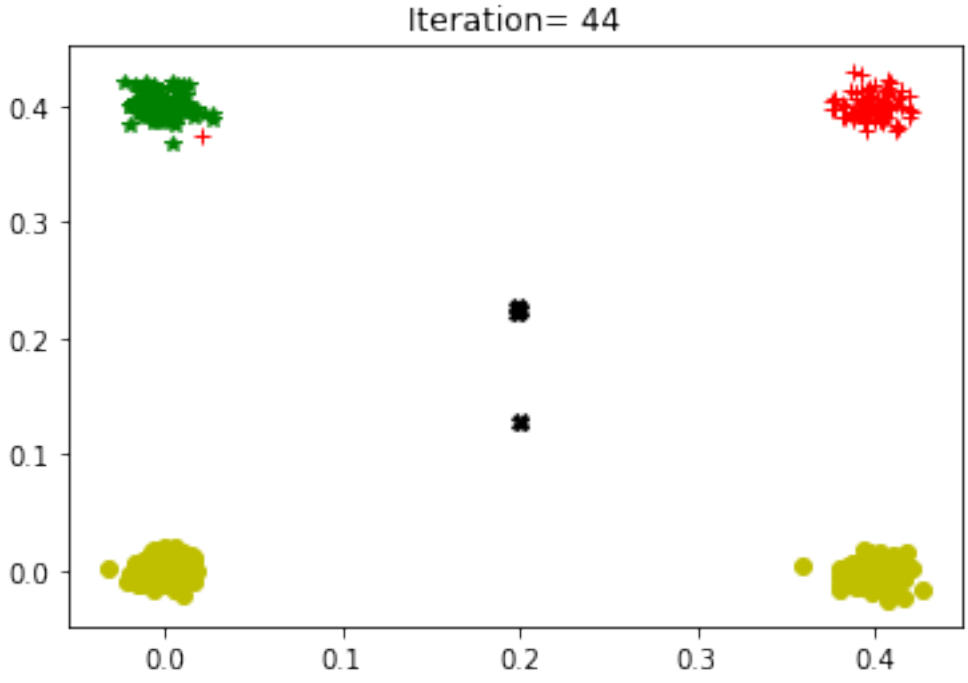

Iteration= 39
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

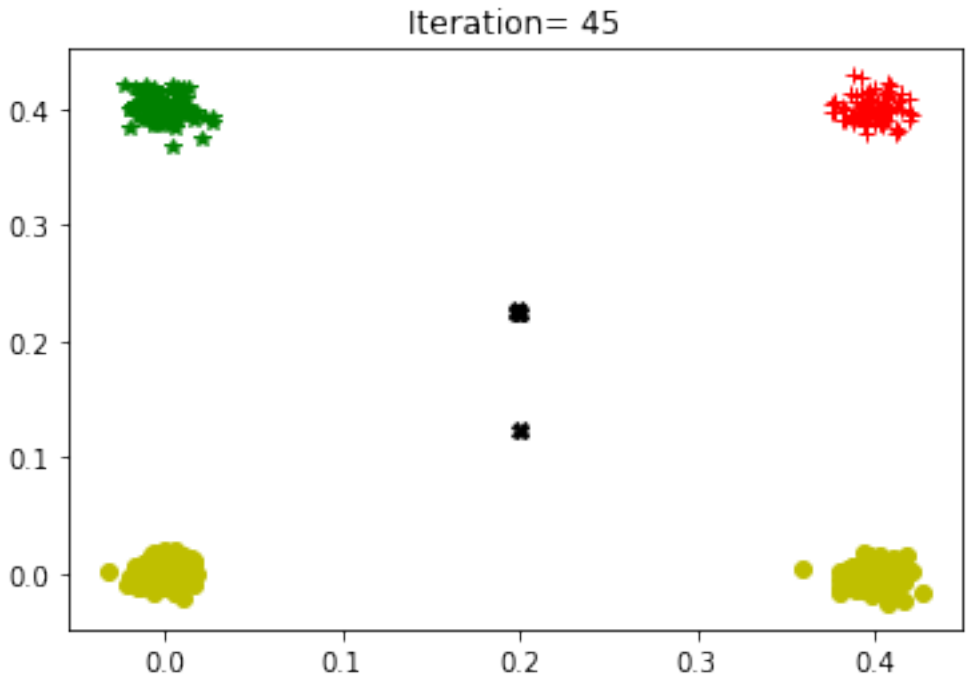

Iteration= 40
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

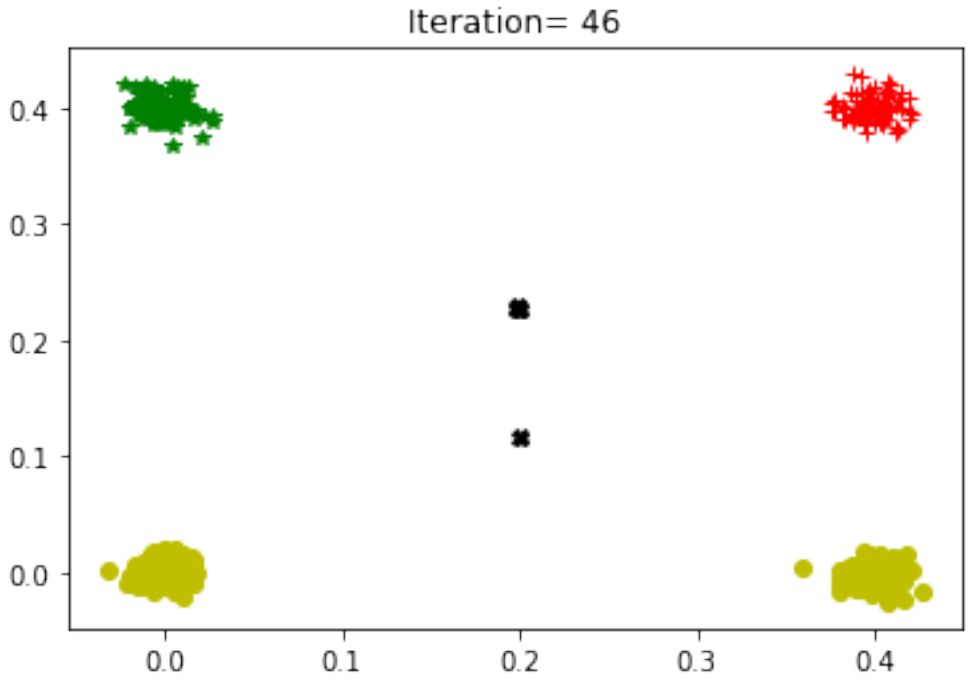

Iteration= 41
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

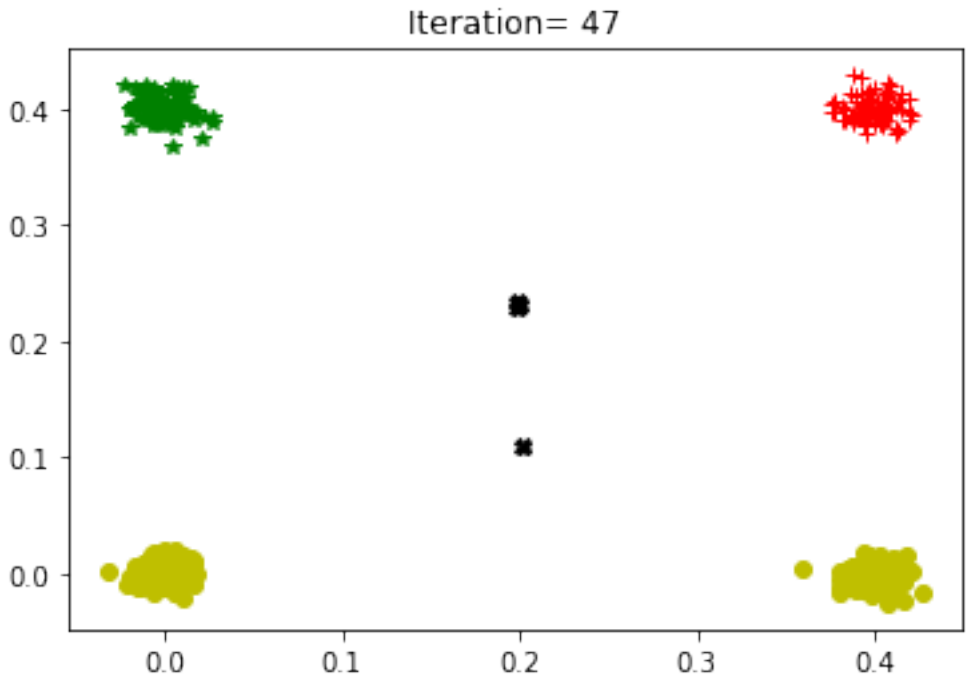

Iteration= 42
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

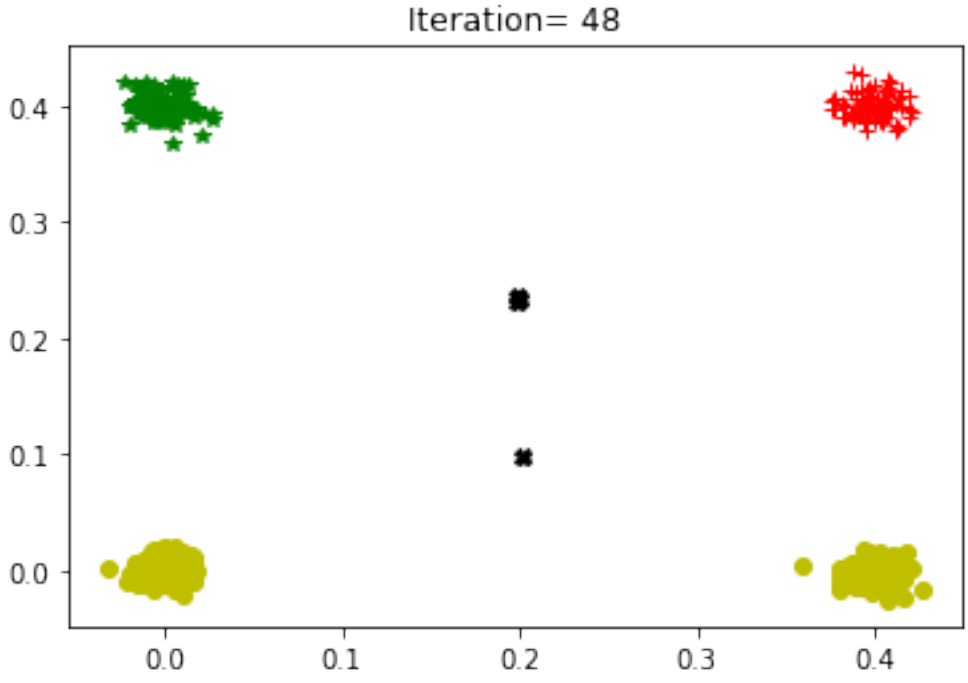

Iteration= 43
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

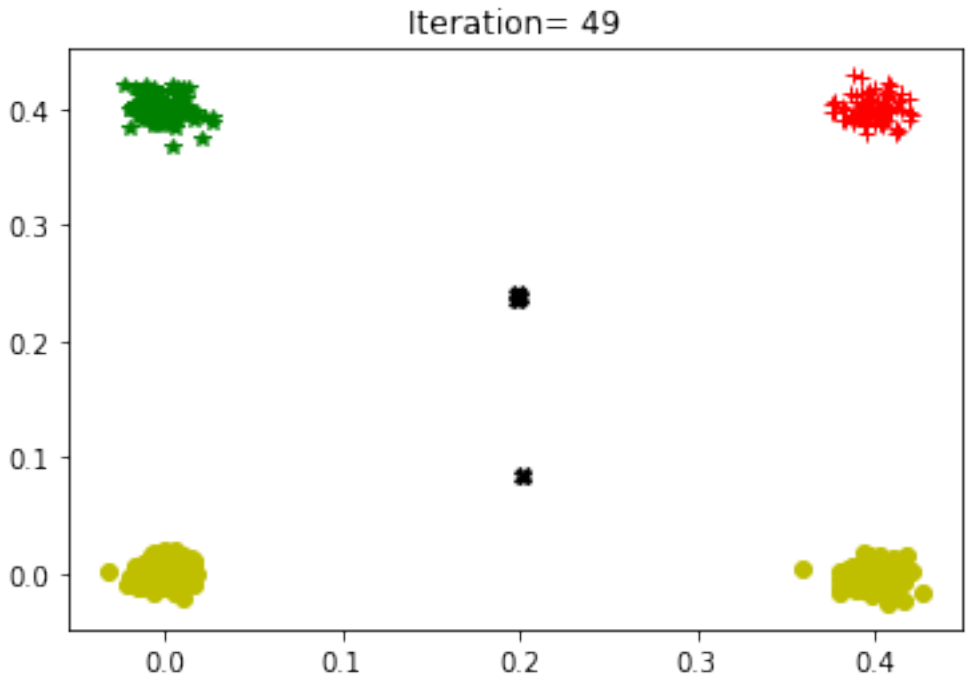

Iteration= 44
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4

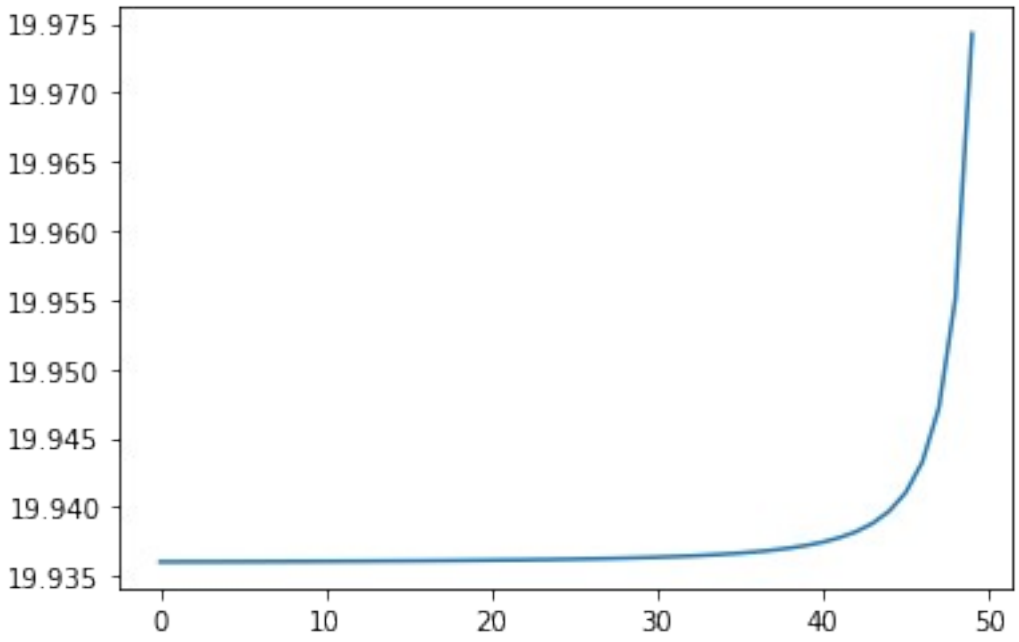

Iteration= 45
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4


Iteration= 46
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4


Iteration= 47
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4


Iteration= 48
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4






```
# write your code here
from sklearn.metrics.cluster import homogeneity_score,
silhouette_score

# Homogeneity Score
labels_true = []
labels_predicted = []
mean_vectors = Cents
```

```
for i in range(K):
    for j in range(len(data_sep[i])):
        labels_true.append(i)
        index = np.argmin(np.linalg.norm(data_sep[i][j] -
mean_vectors, axis=1))
        labels_predicted.append(index)

h_score = homogeneity_score(labels_true, labels_predicted)

print(f'Homogeneity score for the classification is: {h_score}')

# Silhouette coefficient
all_points = data
N = len(all_points)

pairwise = [[0 for i in range(N)] for j in range(N)]
for i in range(N):
    for j in range(N):
        pairwise[i][j] = np.linalg.norm(np.subtract(all_points[i],
all_points[j]))

labels=[]
for id, cluster in enumerate(assigned_clusters):
    labels += [id for i in range(len(cluster))]

sc_score = silhouette_score(pairwise, labels)
print(f'Silhouette coefficient for the classification is: {sc_score}')
```

```
Homogeneity score for the classification is: 0.500000000000001
Silhouette coefficient for the classification is: 0.959237946395165
```

Practical Use Case : K-means Clustering

For this exercise we will be using the **IRIS FLOWER DATASET** and explore how K-means clustering is performing

IRIS Dataset consists of 50 samples from each of the three species of Iris flower (Iris Setosa, Iris Viriginca and Iris Versicolor)

Four features were measured from each sample : Length of Sepals, Width of sepals, Length of Petals, Width of Sepals all in centimeters. Based on the combinations of these 4 features each flower was categorized into one of the 3 species

Steps :

(a) Convert the given iris.csv file into a Pandas Dataframe, then extract both feature vector and target vector

(b) Perform analysis of Dataset, Plot the following features : (Sepal Length vs Sepal Width), (Petal Length vs Petal Width)

(c) Next group the data points into 3 clusters using the above K-means Clustering algorithm and compare the performance against the true labels obtained by the target vector, Also explain the results using a Confusion matrix

(d) Next use scikit learn tool to perform K-means Clustering and compare the performance against the true labels obtained by the target vector, Also explain the results using a Confusion matrix

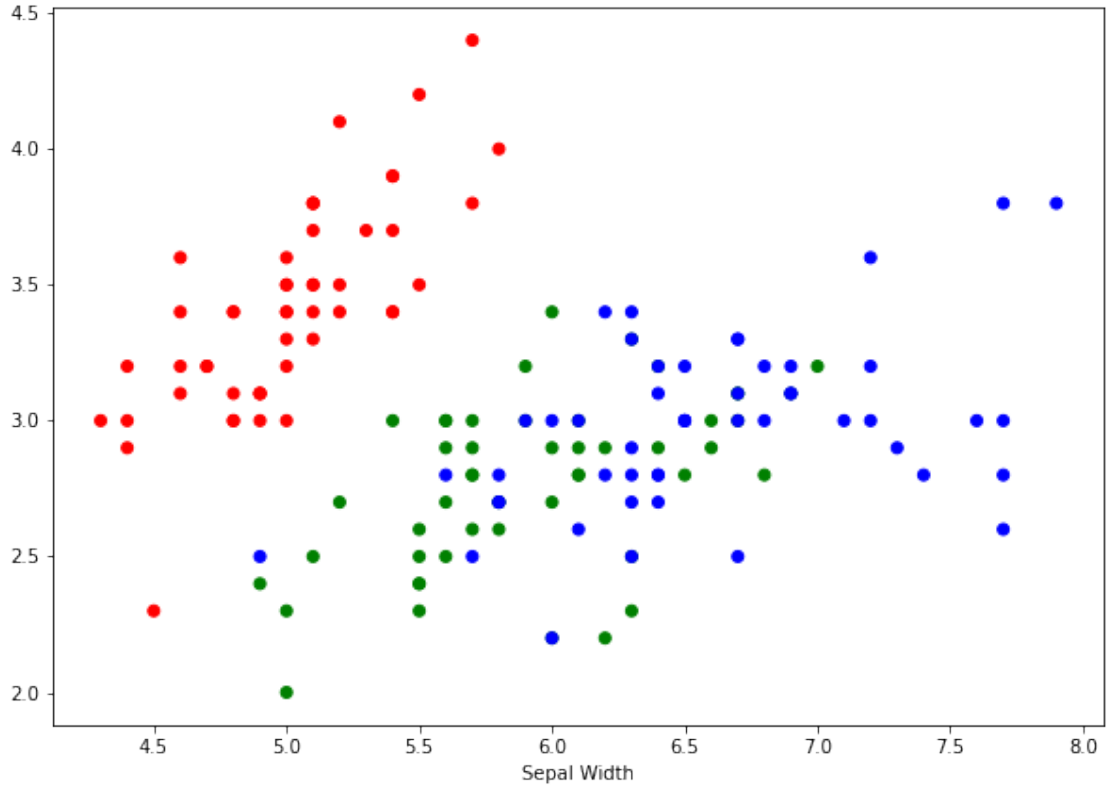
(e) Vary the Number of Clusters (K) and run K-means algorithm from 1-10 and find the optimal number of clusters

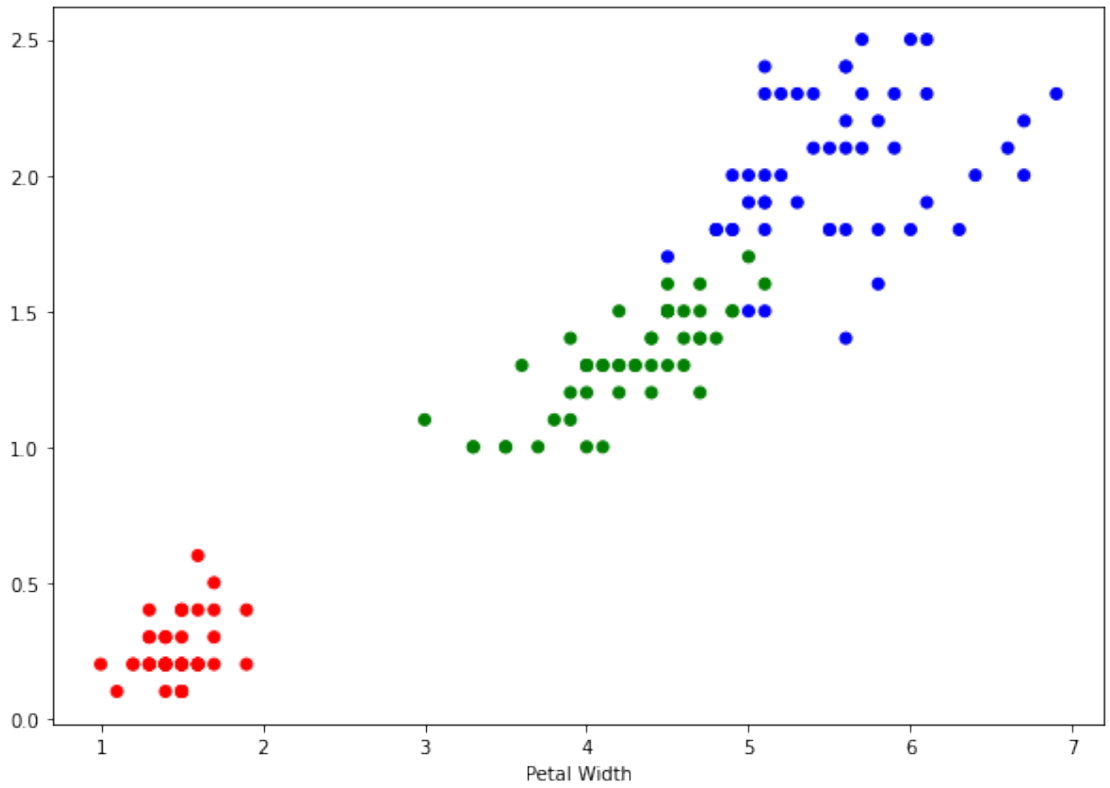
```
import pandas as pd

data = pd.read_csv('Iris.csv')

data[['target']] = data[['species']].apply(lambda
col:pd.Categorical(col).codes)

x =
np.asarray(data[['sepal_length','sepal_width','petal_length','petal_wi
dth']].values.tolist())
y = (data[['species']].apply(lambda
col:pd.Categorical(col).codes)).values.tolist()
target = np.asarray([item for sublist in y for item in sublist])

print(x.shape)
print(target)
```

```
(150, 4)
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

```
colors = np.array(['red', 'green', 'blue'])

plt.figure(figsize=(10, 7))

plt.scatter(x[:, 0], x[:, 1], c=colors[data['target']])
plt.xlabel("Sepal Length")
plt.xlabel("Sepal Width")
```

```
plt.figure(figsize=(10, 7))

plt.scatter(x[:, 2], x[:, 3], c=colors[data['target']])
plt.xlabel("Petal Length")
plt.xlabel("Petal Width")

Text(0.5, 0, 'Petal Width')
```






```
mean_vectors = x[:K, :]

print("Mean Vectors:", mean_vectors)


tol = 1e-5
K = 3
delta = float('inf')
prev_error = float('inf')

errors = []
iter = 0


while delta>tol:
    assigned = [[] for i in range(K)]
    for i in range(x.shape[0]):
        min_index = np.argmin(np.linalg.norm(x[i] - mean_vectors,
axis=1))
        assigned[min_index].append(i)

    assigned_clusters = []
    for i in range(K): assigned_clusters.append(x[assigned[i]])

    mean_vectors = np.array([cluster.mean(axis=0) for cluster in
```

```
assigned_clusters])
    sum_error = 0

    for i in range(K): sum_error +=
np.sum(np.linalg.norm(assigned_clusters[i]-mean_vectors[i], axis=1))

    sum_error
    delta = abs(prev_error - sum_error/x.shape[0])
    prev_error = sum_error/x.shape[0]

    prev_error
    errors.append(prev_error)
    iter+=1

print("Final Mean Vector: ", mean_vectors)
```

```
Mean Vectors: [[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]]
Final Mean Vector:  [[6.85384615 3.07692308 5.71538462 2.05384615]
 [5.88360656 2.74098361 4.38852459 1.43442623]
 [5.006      3.418      1.464      0.244     ]]
```

```
from sklearn.metrics.cluster import homogeneity_score,
silhouette_score

# Homogeneity Score
labels_true = target
labels_predicted = []

for j in range(len(x)):
    index = np.argmin(np.linalg.norm(x[i] - mean_vectors, axis=1))
    labels_predicted.append(index)

h_score = homogeneity_score(labels_true, labels_predicted)

print(f'Homogeneity score for the classification is: {h_score}')

# Silhouette coefficient
all_points = np.concatenate(assigned_clusters, axis=0)
N = len(all_points)

pairwise = [[0 for i in range(N)] for j in range(N)]
for i in range(N):
    for j in range(N):
        pairwise[i][j] = np.linalg.norm(np.subtract(all_points[i],
all_points[j]))

labels=[]
for id, cluster in enumerate(assigned_clusters):
```

```
    labels += [id for i in range(len(cluster))]

sc_score = silhouette_score(pairwise, labels)
print(f'Silhouette coefficient for the classification is: {sc_score}')
```

```
Homogeneity score for the classification is: 0.0
Silhouette coefficient for the classification is: 0.6167390883488784
```

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3,n_init=100,max_iter=100,verbose=1)
kmeans.fit(x)
```

```
Initialization complete
Iteration 0, inertia 173.64000000000004.
Iteration 1, inertia 83.31655597065274.
Iteration 2, inertia 81.49584509677022.
Iteration 3, inertia 80.65491717301568.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 133.21.
Iteration 1, inertia 80.61848028575855.
Iteration 2, inertia 78.999456736457.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 123.67.
Iteration 1, inertia 86.54068732290581.
Iteration 2, inertia 84.58133138509844.
Iteration 3, inertia 83.66851394574321.
Iteration 4, inertia 82.8164109307298.
Iteration 5, inertia 81.63300278471789.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 134.86999999999995.
Iteration 1, inertia 88.16592440707144.
Iteration 2, inertia 80.53582884704555.
Iteration 3, inertia 79.2865426397778.
Iteration 4, inertia 78.94084142614601.
Converged at iteration 4: strict convergence.
Initialization complete
Iteration 0, inertia 138.43000000000004.
Iteration 1, inertia 85.04157943238867.
```

```
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 141.4.
Iteration 1, inertia 80.65140777858467.
Iteration 2, inertia 79.13274230660844.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 92.42000000000002.
Iteration 1, inertia 81.41746489357332.
Iteration 2, inertia 80.14882662005542.
Iteration 3, inertia 79.66525726935402.
Iteration 4, inertia 79.0868989564323.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 104.38.
Iteration 1, inertia 78.99502847222222.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 147.59000000000003.
Iteration 1, inertia 79.6143442279959.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 123.87000000000002.
Iteration 1, inertia 93.51034920634919.
Iteration 2, inertia 85.29856152165407.
Iteration 3, inertia 79.95338439527467.
Iteration 4, inertia 79.2865426397778.
Iteration 5, inertia 78.94084142614601.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 121.49.
Iteration 1, inertia 84.21032527000251.
Iteration 2, inertia 83.11956068208126.
Iteration 3, inertia 82.09154727068666.
Iteration 4, inertia 81.17033414092327.
Iteration 5, inertia 79.96297983461304.
Iteration 6, inertia 79.43376414532675.
Iteration 7, inertia 79.01070972222222.
```

```
Iteration 8, inertia 78.9450658259773.
Converged at iteration 8: strict convergence.
Initialization complete
Iteration 0, inertia 115.41999999999999.
Iteration 1, inertia 80.76854667526317.
Iteration 2, inertia 79.2865426397778.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 100.97.
Iteration 1, inertia 79.10369343504297.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 136.93999999999997.
Iteration 1, inertia 87.00356562736208.
Iteration 2, inertia 84.80172984452898.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 98.86.
Iteration 1, inertia 79.00734589639167.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 129.12000000000003.
Iteration 1, inertia 81.05694170108309.
Iteration 2, inertia 79.2865426397778.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 145.81999999999996.
Iteration 1, inertia 83.64930208333335.
Iteration 2, inertia 82.56108683945021.
Iteration 3, inertia 81.63300278471789.
Iteration 4, inertia 80.895776.
Iteration 5, inertia 79.96297983461304.
Iteration 6, inertia 79.43376414532675.
Iteration 7, inertia 79.01070972222222.
Iteration 8, inertia 78.9450658259773.
Converged at iteration 8: strict convergence.
Initialization complete
Iteration 0, inertia 117.00000000000001.
```

```
Iteration 1, inertia 83.4796326388889.
Iteration 2, inertia 82.09358738904194.
Iteration 3, inertia 81.17033414092327.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 101.52999999999996.
Iteration 1, inertia 79.00734589639167.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 113.68.
Iteration 1, inertia 79.30141304731167.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 137.27999999999997.
Iteration 1, inertia 89.32479987736613.
Iteration 2, inertia 81.99626654702085.
Iteration 3, inertia 79.48372885317471.
Iteration 4, inertia 78.999456736457.
Iteration 5, inertia 78.94084142614601.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 185.65000000000003.
Iteration 1, inertia 80.65491717301568.
Iteration 2, inertia 79.96297983461304.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 90.86.
Iteration 1, inertia 79.03209779286928.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 114.45.
Iteration 1, inertia 79.68515887735795.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 122.31000000000002.
Iteration 1, inertia 80.65491717301568.
Iteration 2, inertia 79.96297983461304.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
```

```
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 102.32.
Iteration 1, inertia 85.04157943238867.
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 118.65000000000005.
Iteration 1, inertia 80.70170616810312.
Iteration 2, inertia 79.2865426397778.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 186.86999999999998.
Iteration 1, inertia 80.990244638936.
Iteration 2, inertia 79.96297983461304.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 170.24999999999997.
Iteration 1, inertia 80.42974927070999.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 117.35000000000002.
Iteration 1, inertia 82.33137226813592.
Iteration 2, inertia 81.17033414092327.
Iteration 3, inertia 79.96297983461304.
Iteration 4, inertia 79.43376414532675.
Iteration 5, inertia 79.01070972222222.
Iteration 6, inertia 78.9450658259773.
Converged at iteration 6: strict convergence.
Initialization complete
Iteration 0, inertia 108.43999999999998.
Iteration 1, inertia 79.08880264145107.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 143.1.
Iteration 1, inertia 82.90561443518715.
```

```
Iteration 2, inertia 80.36666314924393.
Iteration 3, inertia 79.66525726935402.
Iteration 4, inertia 79.0868989564323.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 108.28000000000002.
Iteration 1, inertia 79.48423911916838.
Iteration 2, inertia 79.00734589639167.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 122.52999999999997.
Iteration 1, inertia 81.42891206577869.
Iteration 2, inertia 79.81797453321089.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 113.96.
Iteration 1, inertia 79.56835259340714.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 136.28.
Iteration 1, inertia 87.80883128271402.
Iteration 2, inertia 85.04157943238867.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 103.41999999999999.
Iteration 1, inertia 83.90447868073713.
Iteration 2, inertia 83.13638186876973.
Iteration 3, inertia 81.83900206772623.
Iteration 4, inertia 80.895776.
Iteration 5, inertia 79.96297983461304.
Iteration 6, inertia 79.43376414532675.
Iteration 7, inertia 79.01070972222222.
Iteration 8, inertia 78.9450658259773.
Converged at iteration 8: strict convergence.
Initialization complete
Iteration 0, inertia 168.68.
```

```
Iteration 1, inertia 87.00356562736208.
Iteration 2, inertia 84.80172984452898.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 128.19999999999996.
Iteration 1, inertia 82.52199859016184.
Iteration 2, inertia 79.13199166666668.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 143.70000000000005.
Iteration 1, inertia 82.50620892591817.
Iteration 2, inertia 80.895776.
Iteration 3, inertia 79.96297983461304.
Iteration 4, inertia 79.43376414532675.
Iteration 5, inertia 79.01070972222222.
Iteration 6, inertia 78.9450658259773.
Converged at iteration 6: strict convergence.
Initialization complete
Iteration 0, inertia 117.25.
Iteration 1, inertia 84.15229446479023.
Iteration 2, inertia 83.4796326388889.
Iteration 3, inertia 82.09358738904194.
Iteration 4, inertia 81.17033414092327.
Iteration 5, inertia 79.96297983461304.
Iteration 6, inertia 79.43376414532675.
Iteration 7, inertia 79.01070972222222.
Iteration 8, inertia 78.9450658259773.
Converged at iteration 8: strict convergence.
Initialization complete
Iteration 0, inertia 182.73999999999998.
Iteration 1, inertia 89.7510719985899.
Iteration 2, inertia 81.52124049823634.
Iteration 3, inertia 79.2865426397778.
Iteration 4, inertia 78.94084142614601.
Converged at iteration 4: strict convergence.
Initialization complete
Iteration 0, inertia 129.41.
Iteration 1, inertia 80.30846682568715.
Iteration 2, inertia 79.00734589639167.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
```

```
Initialization complete
Iteration 0, inertia 132.32.
Iteration 1, inertia 88.92035772737768.
Iteration 2, inertia 85.04157943238867.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 99.55000000000001.
Iteration 1, inertia 79.29384108199494.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 128.71.
Iteration 1, inertia 84.7939391377853.
Iteration 2, inertia 83.90447868073713.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 128.64999999999998.
Iteration 1, inertia 99.830986185238.
Iteration 2, inertia 87.34292218465293.
Iteration 3, inertia 80.23054939305555.
Iteration 4, inertia 79.2865426397778.
Iteration 5, inertia 78.94084142614601.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 195.60000000000002.
Iteration 1, inertia 91.51603224363818.
Iteration 2, inertia 79.00283107220375.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 114.58000000000001.
Iteration 1, inertia 79.48423911916838.
Iteration 2, inertia 79.00734589639167.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
```

```
Initialization complete
Iteration 0, inertia 114.84.
Iteration 1, inertia 82.9589329256554.
Iteration 2, inertia 81.8500876628782.
Iteration 3, inertia 80.895776.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 102.24000000000001.
Iteration 1, inertia 85.3459166319745.
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 169.15000000000006.
Iteration 1, inertia 87.32761271218051.
Iteration 2, inertia 84.80172984452898.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 120.61999999999999.
Iteration 1, inertia 79.6405430468882.
Iteration 2, inertia 79.01070972222222.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 100.98000000000002.
Iteration 1, inertia 79.2390602412587.
Iteration 2, inertia 79.01070972222222.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 151.74.
Iteration 1, inertia 100.07219912656417.
```

```
Iteration 2, inertia 87.07155320600982.
Iteration 3, inertia 80.23054939305555.
Iteration 4, inertia 79.2865426397778.
Iteration 5, inertia 78.94084142614601.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 120.71999999999998.
Iteration 1, inertia 83.21784080254825.
Iteration 2, inertia 82.09358738904194.
Iteration 3, inertia 81.17033414092327.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 133.29999999999998.
Iteration 1, inertia 83.34831944444446.
Iteration 2, inertia 82.34796145640075.
Iteration 3, inertia 81.41746489357332.
Iteration 4, inertia 80.14882662005542.
Iteration 5, inertia 79.66525726935402.
Iteration 6, inertia 79.0868989564323.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 136.22.
Iteration 1, inertia 84.57151680772571.
Iteration 2, inertia 83.77629490216783.
Iteration 3, inertia 82.884594526007.
Iteration 4, inertia 81.63300278471789.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 131.56.
Iteration 1, inertia 79.14467083333334.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 113.97.
Iteration 1, inertia 80.84446414666706.
Iteration 2, inertia 79.96297983461304.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
```

```
Initialization complete
Iteration 0, inertia 104.66999999999999.
Iteration 1, inertia 79.14036196719832.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 123.25000000000003.
Iteration 1, inertia 80.57862575295974.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 122.98000000000002.
Iteration 1, inertia 84.5383907924567.
Iteration 2, inertia 80.23054939305555.
Iteration 3, inertia 79.2865426397778.
Iteration 4, inertia 78.94084142614601.
Converged at iteration 4: strict convergence.
Initialization complete
Iteration 0, inertia 149.84.
Iteration 1, inertia 86.22167225130993.
Iteration 2, inertia 79.30141304731167.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 93.31000000000002.
Iteration 1, inertia 83.2673450767069.
Iteration 2, inertia 81.83900206772623.
Iteration 3, inertia 80.895776.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 115.98000000000003.
Iteration 1, inertia 81.18989049248029.
Iteration 2, inertia 79.8163185389246.
Iteration 3, inertia 79.43376414532675.
Iteration 4, inertia 79.01070972222222.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 190.56.
Iteration 1, inertia 79.01070972222222.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 116.65.
Iteration 1, inertia 90.78355774369352.
Iteration 2, inertia 82.54351817810458.
```

```
Iteration 3, inertia 79.70512102298426.
Iteration 4, inertia 79.1014489607446.
Iteration 5, inertia 78.94084142614601.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 106.22.
Iteration 1, inertia 79.75896556399564.
Iteration 2, inertia 79.1014489607446.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 134.60000000000002.
Iteration 1, inertia 79.19135670490881.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 139.28.
Iteration 1, inertia 86.68952871174562.
Iteration 2, inertia 84.35467300347223.
Iteration 3, inertia 83.4796326388889.
Iteration 4, inertia 82.09358738904194.
Iteration 5, inertia 81.17033414092327.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 146.0.
Iteration 1, inertia 86.13373786848074.
Iteration 2, inertia 84.58133138509844.
Iteration 3, inertia 83.66851394574321.
Iteration 4, inertia 82.8164109307298.
Iteration 5, inertia 81.63300278471789.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 185.69000000000005.
Iteration 1, inertia 80.8028785718144.
Iteration 2, inertia 79.13199166666668.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 101.29.
Iteration 1, inertia 79.05429864127426.
Iteration 2, inertia 78.94084142614601.
```

```
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 113.48.
Iteration 1, inertia 79.2390602412587.
Iteration 2, inertia 79.01070972222222.
Iteration 3, inertia 78.9450658259773.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 138.47.
Iteration 1, inertia 79.1127507562901.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 135.88.
Iteration 1, inertia 83.01836676022768.
Iteration 2, inertia 81.83900206772623.
Iteration 3, inertia 80.895776.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 129.07999999999998.
Iteration 1, inertia 79.79932603066868.
Iteration 2, inertia 79.00734589639167.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 172.10000000000002.
Iteration 1, inertia 145.53460361967402.
Iteration 2, inertia 144.57196757030434.
Iteration 3, inertia 144.07926881970437.
Iteration 4, inertia 143.65091370831087.
Iteration 5, inertia 143.50951798420766.
Iteration 6, inertia 143.45373548406212.
Converged at iteration 6: strict convergence.
Initialization complete
Iteration 0, inertia 186.10000000000002.
Iteration 1, inertia 84.58133138509844.
Iteration 2, inertia 83.66851394574321.
Iteration 3, inertia 82.8164109307298.
Iteration 4, inertia 81.63300278471789.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
```

```
Iteration 0, inertia 110.31000000000002.
Iteration 1, inertia 79.35994788069075.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 101.55999999999997.
Iteration 1, inertia 79.204003515625.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 169.36.
Iteration 1, inertia 89.53482320132917.
Iteration 2, inertia 85.04157943238867.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 106.16.
Iteration 1, inertia 79.26569443300406.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 91.96000000000002.
Iteration 1, inertia 82.38656100417441.
Iteration 2, inertia 81.17033414092327.
Iteration 3, inertia 79.96297983461304.
Iteration 4, inertia 79.43376414532675.
Iteration 5, inertia 79.01070972222222.
Iteration 6, inertia 78.9450658259773.
Converged at iteration 6: strict convergence.
Initialization complete
Iteration 0, inertia 107.97999999999999.
Iteration 1, inertia 84.80172984452898.
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 114.62.
```

```
Iteration 1, inertia 82.9589329256554.
Iteration 2, inertia 81.8500876628782.
Iteration 3, inertia 80.895776.
Iteration 4, inertia 79.96297983461304.
Iteration 5, inertia 79.43376414532675.
Iteration 6, inertia 79.01070972222222.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 111.67999999999999.
Iteration 1, inertia 79.70512102298426.
Iteration 2, inertia 79.1014489607446.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 117.92.
Iteration 1, inertia 80.22491376639681.
Iteration 2, inertia 79.66525726935402.
Iteration 3, inertia 79.0868989564323.
Iteration 4, inertia 78.9450658259773.
Converged at iteration 4: strict convergence.
Initialization complete
Iteration 0, inertia 138.32.
Iteration 1, inertia 80.61053151927437.
Iteration 2, inertia 79.2865426397778.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 109.70000000000006.
Iteration 1, inertia 79.16756949488614.
Iteration 2, inertia 78.94084142614601.
Converged at iteration 2: strict convergence.
Initialization complete
Iteration 0, inertia 128.92000000000004.
Iteration 1, inertia 85.3459166319745.
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
Initialization complete
Iteration 0, inertia 120.58000000000001.
Iteration 1, inertia 79.51834650631778.
Iteration 2, inertia 78.9450658259773.
Converged at iteration 2: strict convergence.
Initialization complete
```

```
Iteration 0, inertia 144.48000000000005.
Iteration 1, inertia 89.57389806356952.
Iteration 2, inertia 85.04157943238867.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
Iteration 0, inertia 93.34.
Iteration 1, inertia 83.34831944444446.
Iteration 2, inertia 82.34796145640075.
Iteration 3, inertia 81.41746489357332.
Iteration 4, inertia 80.14882662005542.
Iteration 5, inertia 79.66525726935402.
Iteration 6, inertia 79.0868989564323.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 99.30999999999999.
Iteration 1, inertia 81.40595271007962.
Iteration 2, inertia 80.14882662005542.
Iteration 3, inertia 79.66525726935402.
Iteration 4, inertia 79.0868989564323.
Iteration 5, inertia 78.9450658259773.
Converged at iteration 5: strict convergence.
Initialization complete
Iteration 0, inertia 111.10999999999997.
Iteration 1, inertia 79.3164909120747.
Iteration 2, inertia 79.00734589639167.
Iteration 3, inertia 78.94084142614601.
Converged at iteration 3: strict convergence.
Initialization complete
Iteration 0, inertia 118.80000000000001.
Iteration 1, inertia 87.00356562736208.
Iteration 2, inertia 84.80172984452898.
Iteration 3, inertia 84.1021788886515.
Iteration 4, inertia 83.13638186876973.
Iteration 5, inertia 81.83900206772623.
Iteration 6, inertia 80.895776.
Iteration 7, inertia 79.96297983461304.
Iteration 8, inertia 79.43376414532675.
Iteration 9, inertia 79.01070972222222.
Iteration 10, inertia 78.9450658259773.
Converged at iteration 10: strict convergence.
Initialization complete
```

```
Iteration 0, inertia 141.50000000000006.
Iteration 1, inertia 85.0895257573184.
Iteration 2, inertia 82.46622823832014.
Iteration 3, inertia 81.41746489357332.
Iteration 4, inertia 80.14882662005542.
Iteration 5, inertia 79.66525726935402.
Iteration 6, inertia 79.0868989564323.
Iteration 7, inertia 78.9450658259773.
Converged at iteration 7: strict convergence.
Initialization complete
Iteration 0, inertia 118.97999999999996.
Iteration 1, inertia 84.80172984452898.
Iteration 2, inertia 84.1021788886515.
Iteration 3, inertia 83.13638186876973.
Iteration 4, inertia 81.83900206772623.
Iteration 5, inertia 80.895776.
Iteration 6, inertia 79.96297983461304.
Iteration 7, inertia 79.43376414532675.
Iteration 8, inertia 79.01070972222222.
Iteration 9, inertia 78.9450658259773.
Converged at iteration 9: strict convergence.
```

```
KMeans(max_iter=100, n_clusters=3, n_init=100, verbose=1)
```

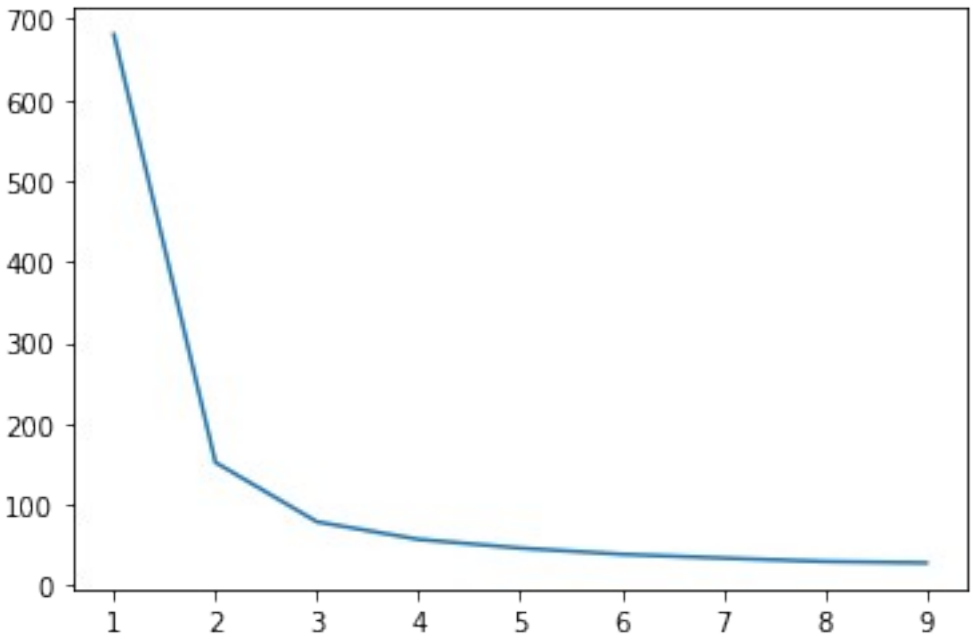
```
from sklearn import metrics

pred=kmeans.predict(x)
print('Homogeneity Score: ',metrics.homogeneity_score(pred,target))

import sklearn.metrics as sm

sm.confusion_matrix(pred, target)
```

```
Homogeneity Score:  0.7649861514489815
```

```
array([[50,  0,  0],
       [ 0, 48, 14],
       [ 0,  2, 36]], dtype=int64)
```

```
dist = []
K = range(1,10)
for no_of_clusters in K:
  k_model = KMeans(n_clusters=no_of_clusters)
  k_model.fit(x)
  dist.append(k_model.inertia_)
plt.plot(K, dist)
```

```
[<matplotlib.lines.Line2D at 0x1d121c0a8f0>]
```



Practical Use Case : GMM

Steps :

(a) Convert the given iris.csv file into a Pandas Dataframe, then extract both feature vector and target vector

(b) Next group the data points into 3 clusters using the above GMM Clustering algorithm and compare the performance against the true labels obtained by the target vector, Also explain the results using a Confusion matrix

(c) Next use scikit learn tool to perform GMM Clustering and compare the performance against the true labels obtained by the target vector, Also explain the results using a Confusion matrix

```
# write your code here
from sklearn.mixture import GaussianMixture
gmm=GaussianMixture(n_components=3,
init_params='kmeans',covariance_type='diag')
gmm.fit(x)
gmm.get_params(deep=True)
print("Means: ", gmm.means_)
print("Variance", gmm.covariances_)
print("Weights: ", gmm.weights_)

from sklearn import metrics

pred=gmm.predict(x)
print('performance=',metrics.homogeneity_score(pred,target))
```

```
import sklearn.metrics as sm
sm.confusion_matrix(pred, target)
```

```
Means:  [[5.92793543 2.75046463 4.40762592 1.41444826]
 [5.006      3.418      1.464      0.244     ]
 [6.81209583 3.07212929 5.72666022 2.10669076]]
Variance [[0.23174382 0.08726933 0.27664585 0.06948396]
 [0.121765   0.142277   0.029505   0.011265  ]
 [0.28321015 0.08204184 0.24785895 0.06020239]]
Weights:  [0.41477831 0.33333333 0.25188836]
performance= 0.8156456882407057
```

```
array([[ 0, 50, 14],
       [50,  0,  0],
       [ 0,  0, 36]], dtype=int64)
```