Untitled1

June 19, 2021

[2]:
```
import numpy as np
import pandas as pd
import random
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets.samples_generator import make_blobs
```

[3]:
```
x1 = [-4.9, -3.5, 0, -4.5, -3, -1, -1.2, -4.5, -1.5, -4.5, -1, -2, -2.5, -2, -1.
 ↪5, 4, 1.8, 2, 2.5, 3, 4, 2.25, 1, 0, 1, 2.5, 5, 2.8, 2, 2]
x2 = [-3.5, -4, -3.5, -3, -2.9, -3, -2.6, -2.1, 0, -0.5, -0.8, -0.8, -1.5, -1.
 ↪75, -1.75, 0, 0.8, 0.9, 1, 1, 1, 1.75, 2, 2.5, 2.5, 2.5, 2.5, 3, 6, 6.5]
print('Datapoints defined!')
```

```
Datapoints defined!
```

[4]:
```
colors_map=np.array(['b','r'])

def assign_members(x1, x2, centers):
    compare_to_first_center = np.sqrt(np.square(np.array(x1) - centers[0][0]) +␣
 ↪np.square(np.array(x2) - centers[0][1]))
    compare_to_second_center = np.sqrt(np.square(np.array(x1) - centers[1][0])␣
 ↪+ np.square(np.array(x2) - centers[1][1]))
    class_of_points = compare_to_first_center > compare_to_second_center
    colors = colors_map[class_of_points+1-1]
    return colors, class_of_points

print('assign_members function defined!')
```

```
assign_members function defined!
```

[5]:
```
def update_centers(x1,x2,class_of_points):
    center1=[np.mean(np.array(x1)[~class_of_points]),np.mean(np.
 ↪array(x2)[~class_of_points])]
    center2=[np.mean(np.array(x1)[class_of_points]),np.mean(np.
 ↪array(x2)[class_of_points])]
    return(center1,center2)
```

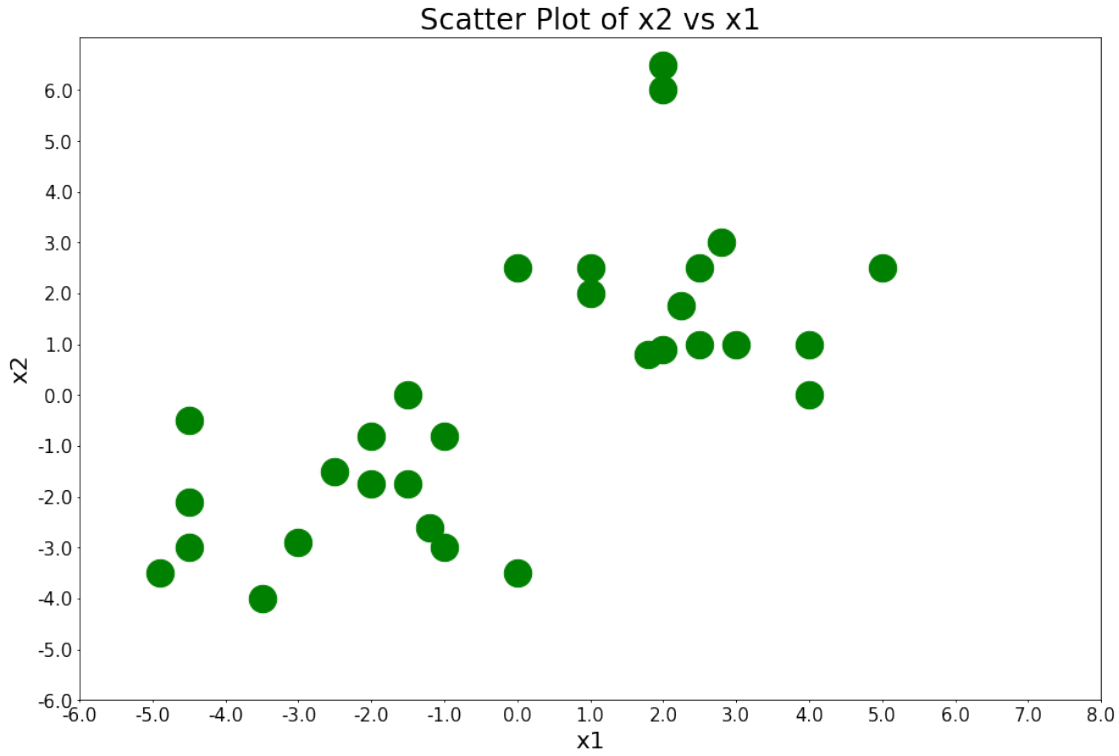
```
print('assign_members function defined!')
```

```
assign_members function defined!
```

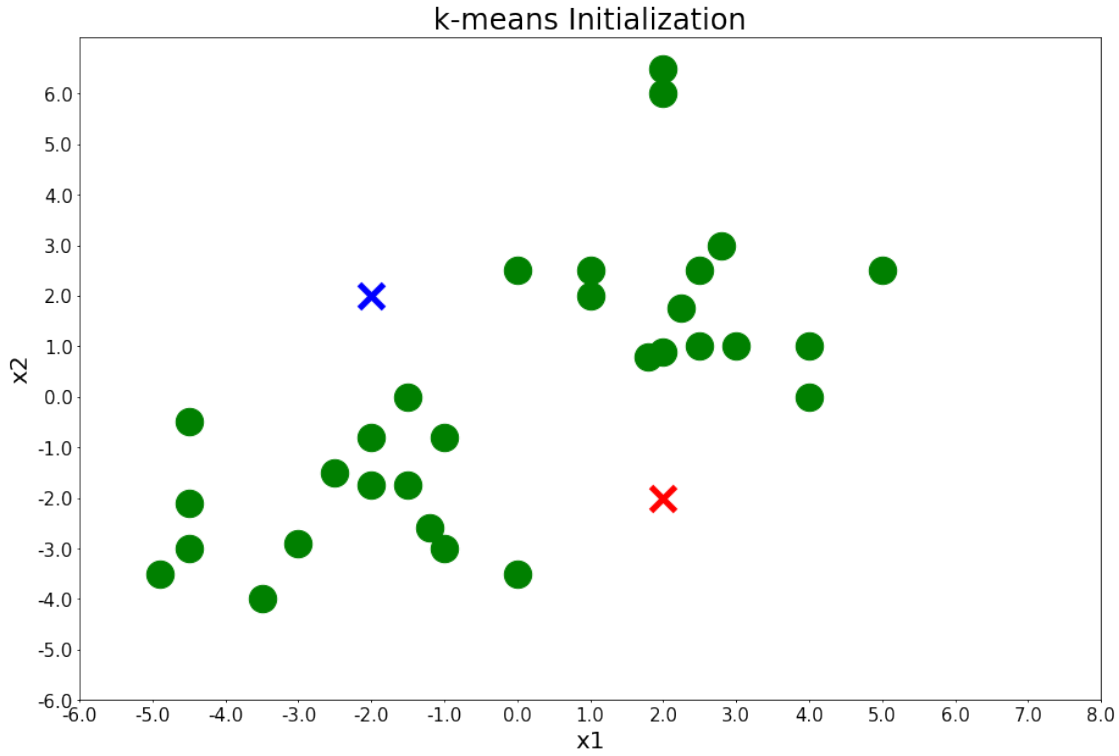
[6]:
```
def plot_points(centroids=None,colors='g',figure_title=None):
    fig=plt.figure(figsize=(15,10))
    ax=fig.add_subplot(1,1,1)


    centroid_colors=['bx','rx']
    if centroids:
        for(i,centroid) in enumerate(centroids):
            ax.plot(centroid[0],centroid[1],␣
 ↪centroid_colors[i],markeredgewidth=5,markersize=20)
    plt.scatter(x1,x2,s=500,c=colors)

    xticks=np.linspace(-6,8,15,endpoint=True)
    yticks=np.linspace(-6,6,13,endpoint=True)

    ax.set_xticks(xticks)
    ax.set_yticks(yticks)

    xlabels=xticks
    ax.set_xticklabels(xlabels)
    ylabels=yticks
    ax.set_yticklabels(ylabels)


    ax.xaxis.set_ticks_position('bottom')
    ax.yaxis.set_ticks_position('left')
    ax.tick_params('both',length=2,width=1,which='major',labelsize=15)

    ax.set_xlabel('x1',fontsize=20)
    ax.set_ylabel('x2',fontsize=20)

    ax.set_title(figure_title,fontsize=24)

    plt.show()

print('plot_points function defined')
```

```
plot_points function defined
```

[7]:
```
plot_points(figure_title='Scatter Plot of x2 vs x1')
```

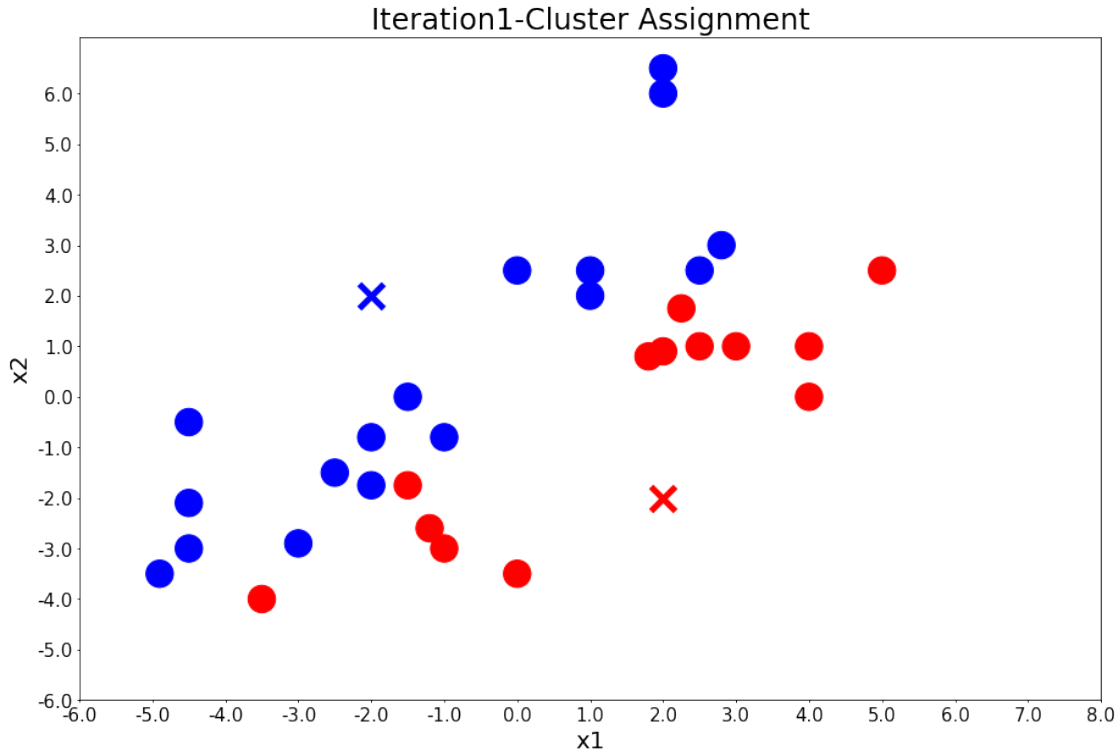


[8]:
```
centers = [[-2, 2], [2, -2]]
plot_points(centers, figure_title='k-means Initialization')
```

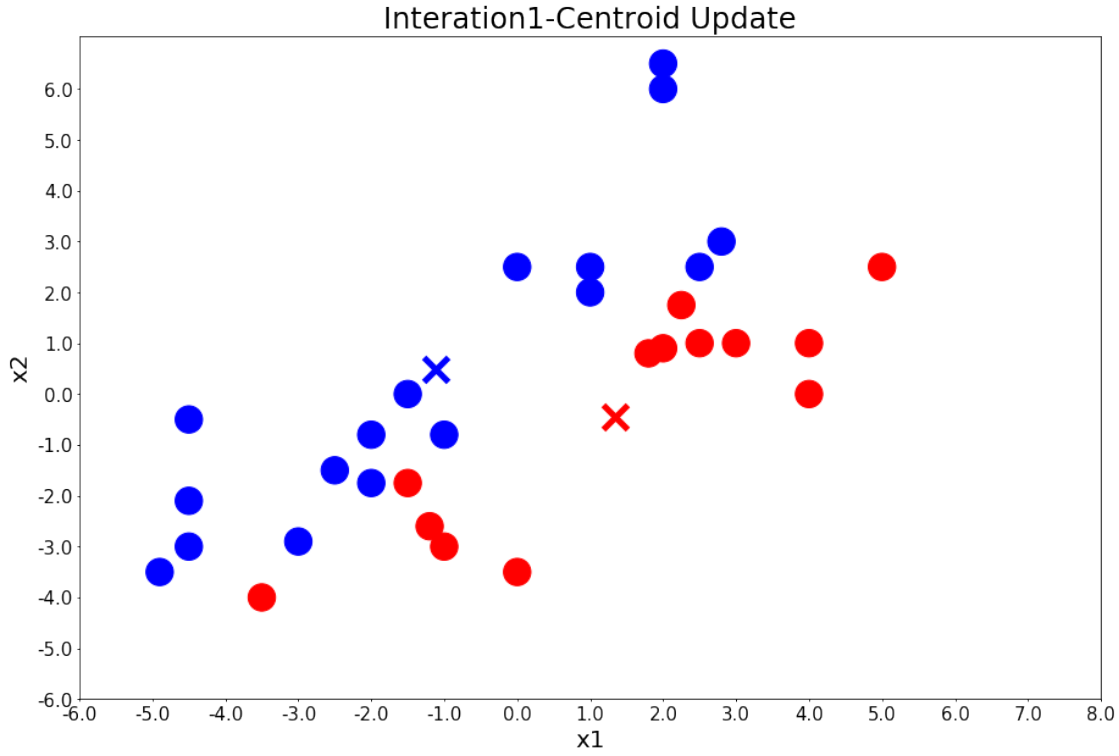


[9]:
```
number_of_iterations=4
for i in range (number_of_iterations):
    input('Iteration{}-Press enter to update the member of each cluster'.
 ↪format(i+1))
    colors,class_of_points=assign_members(x1,x2,centers)
    title='Iteration{}-Cluster Assignment'.format(i+1)
    plot_points(centers,colors,figure_title=title)
    input('Iteration{}-Press enter to update the centers'.format(i+1))
    centers=update_centers(x1,x2,class_of_points)
    title='Interation{}-Centroid Update'.format(i+1)
    plot_points(centers,colors,figure_title=title)
```

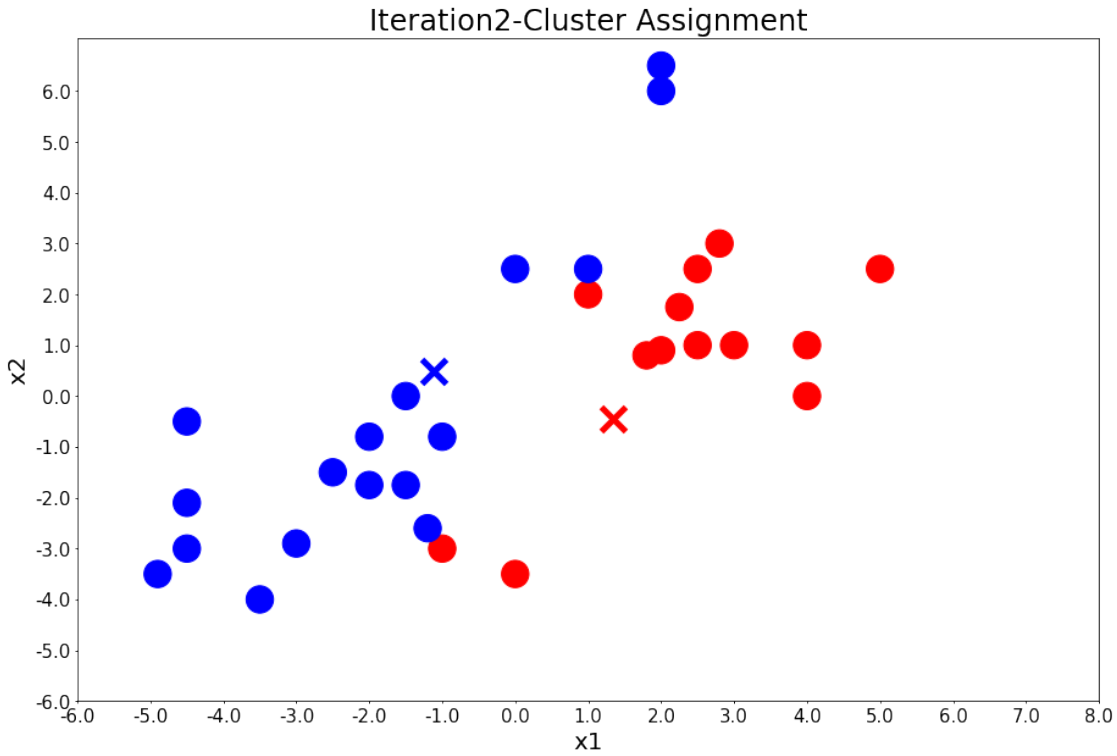
```
Iteration1-Press enter to update the member of each cluster
```



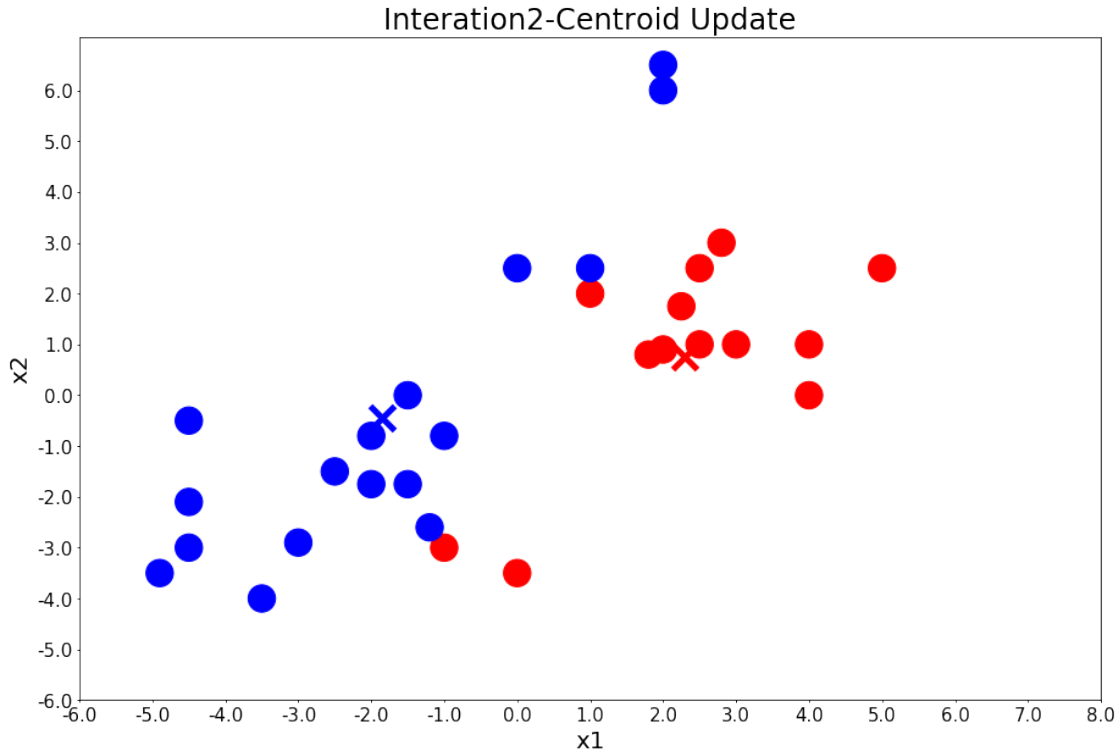

```
Iteration1-Press enter to update the centers
```



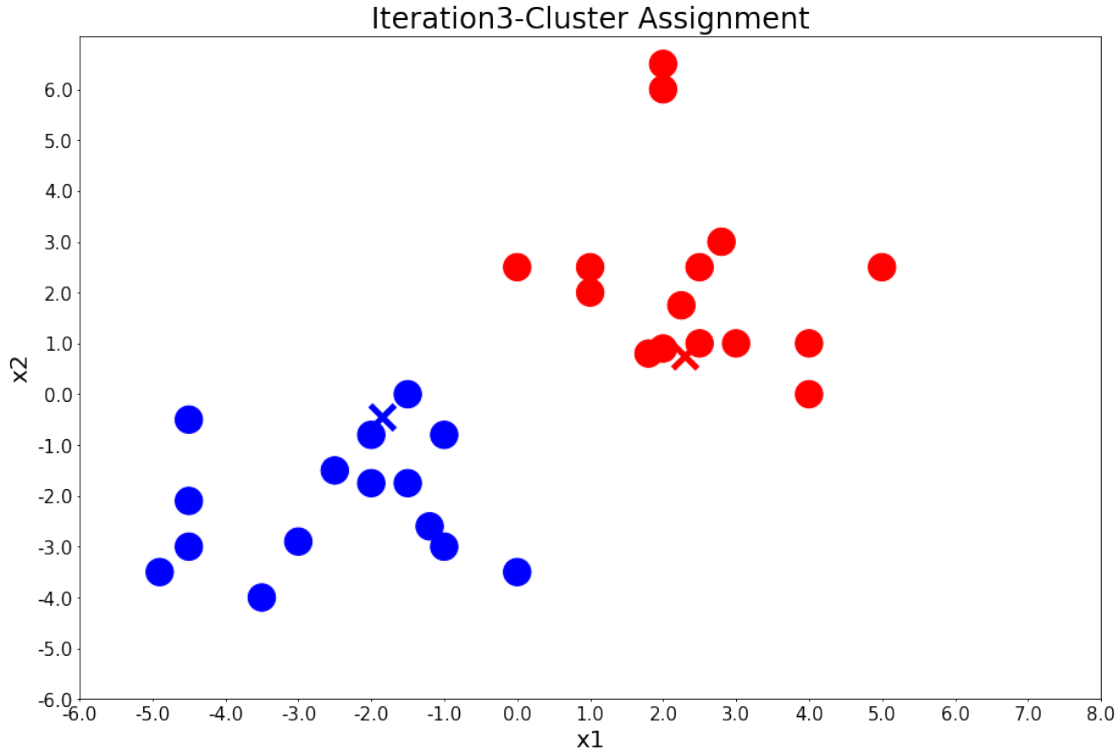
```
Iteration2-Press enter to update the member of each cluster
```



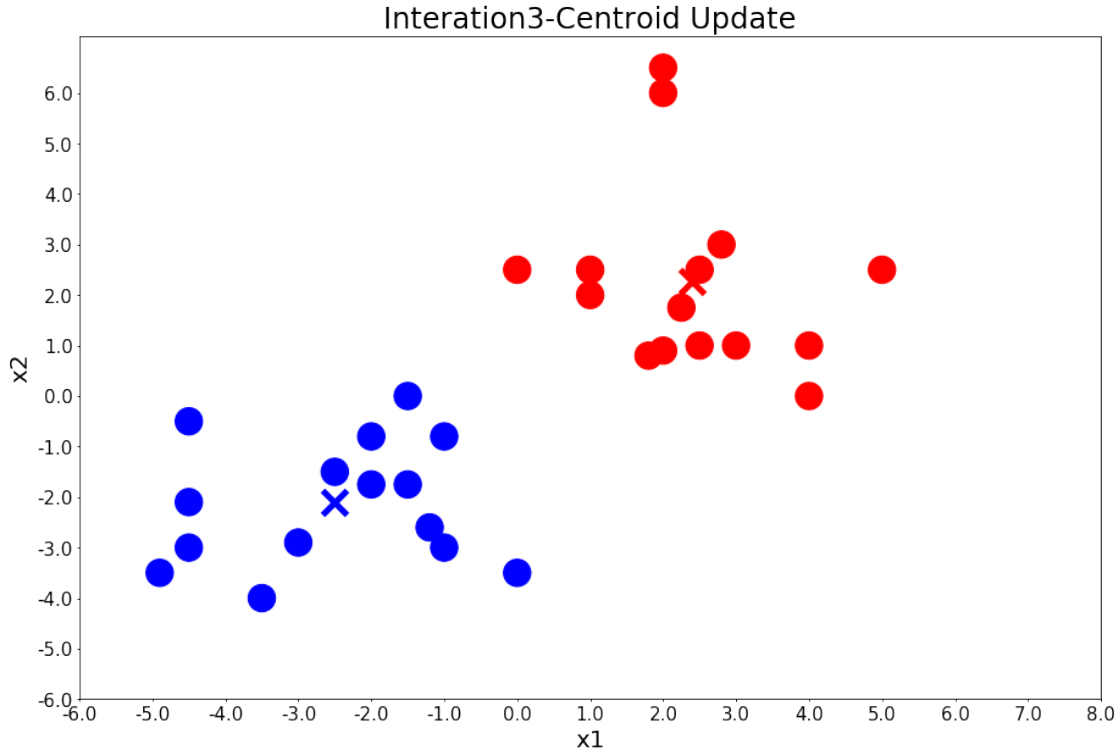

```
Iteration2-Press enter to update the centers
```



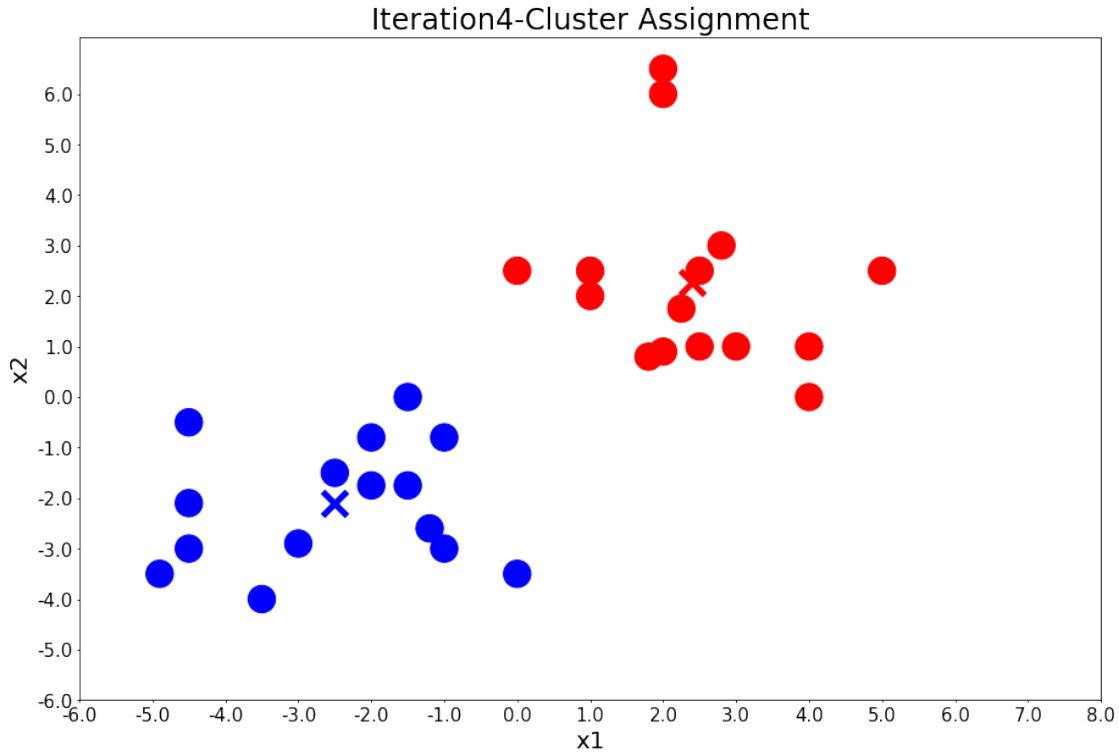

```
Iteration3-Press enter to update the member of each cluster
```



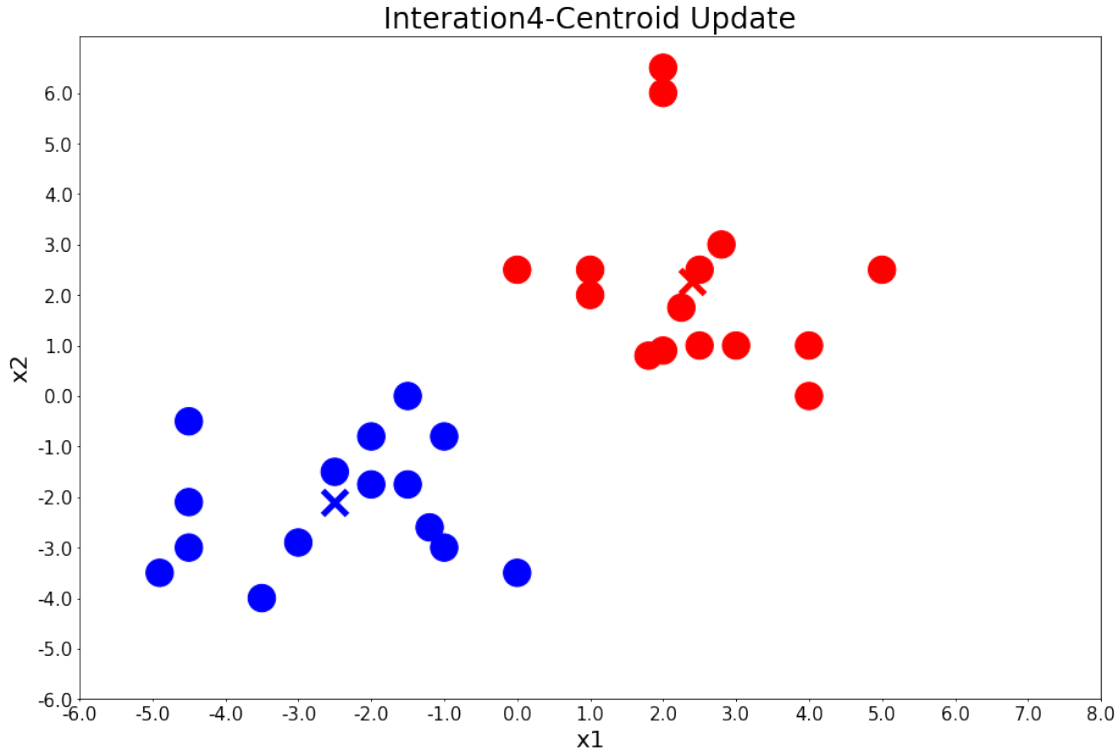
```
Iteration3-Press enter to update the centers
```




```
Iteration4-Press enter to update the member of each cluster
```




```
Iteration4-Press enter to update the centers
```

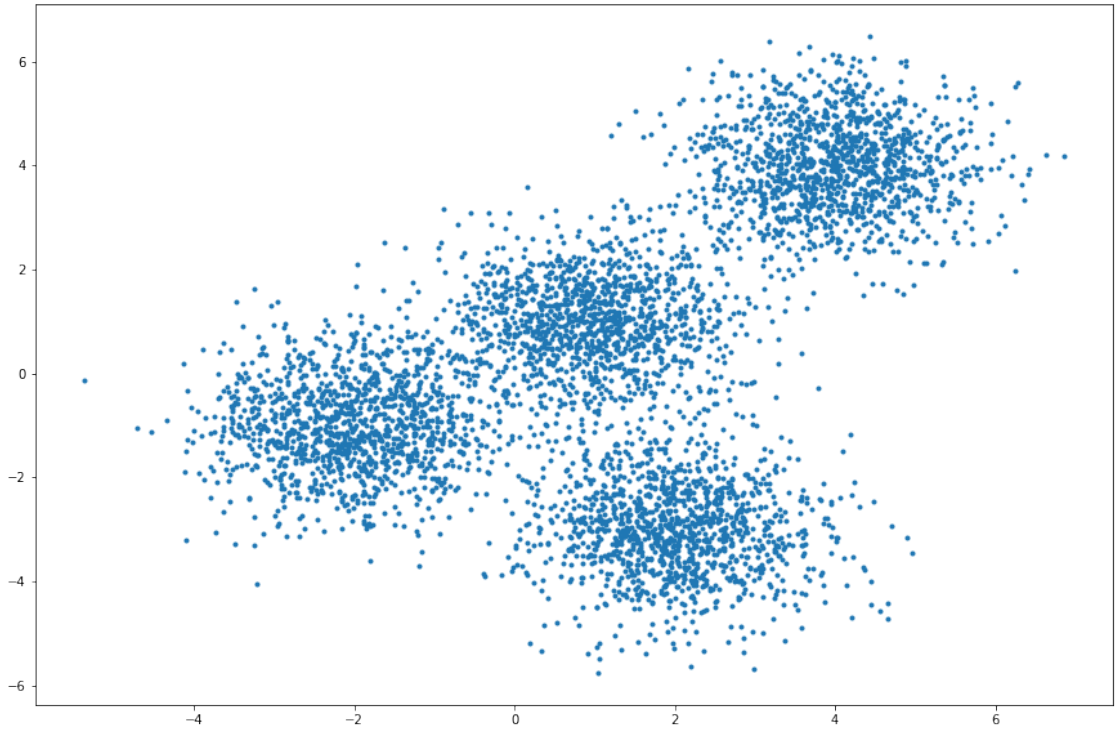


[10]:
```
np.random.seed(0)
```

[13]:
```
x,y=make_blobs(n_samples=5000,centers=[[4,4],[-2,-1],[2,-3],[1,1]],cluster_std=0.
↪9)
```

[14]:
```
plt.figure(figsize=(15,10))
plt.scatter(x[:,0],x[:,1],marker='.')
```

[14]:
```
<matplotlib.collections.PathCollection at 0x1a1f1ea190>
```



[16]:
```
k_means=KMeans(init='k-means++',n_clusters=4,n_init=12)
```

[17]:
```
k_means.fit(x)
```

[17]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=4, n_init=12, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

[18]:
```
k_means_labels=k_means.labels_
k_means_labels
```

[18]:
```
array([0, 3, 3, …, 1, 0, 0], dtype=int32)
```

[ ]:
```

```