In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot
import pygad
```

In [4]:

```
cluster1_num_samples = 10
cluster1_x1_start = 0
cluster1_x1_end = 5
cluster1_x2_start = 2
cluster1_x2_end = 6
cluster1_x1 = np.random.random(size=(cluster1_num_samples))
cluster1_x1 = cluster1_x1 * (cluster1_x1_end - cluster1_x1_start) + cluster1_x1_start
cluster1_x2 = np.random.random(size=(cluster1_num_samples))
cluster1_x2 = cluster1_x2 * (cluster1_x2_end - cluster1_x2_start) + cluster1_x2_start
cluster2_num_samples = 10
cluster2_x1_start = 10
cluster2_x1_end = 15
cluster2_x2_start = 8
cluster2_x2_end = 12
cluster2_x1 = np.random.random(size=(cluster2_num_samples))
cluster2_x1 = cluster2_x1 * (cluster2_x1_end - cluster2_x1_start) + cluster2_x1_start
cluster2_x2 = np.random.random(size=(cluster2_num_samples))
cluster2_x2 = cluster2_x2 * (cluster2_x2_end - cluster2_x2_start) + cluster2_x2_start
```

In [5]:

```
c1 = np.array([cluster1_x1, cluster1_x2]).T
c2 = np.array([cluster2_x1, cluster2_x2]).T
data = np.concatenate((c1, c2), axis=0)
data
```

Out[5]:

```
array([[ 1.27898768,  4.68548343],
       [ 3.61945701,  3.26190686],
       [ 3.07839194,  5.47084461],
       [ 2.38205502,  2.78383971],
       [ 0.69906865,  4.65059247],
       [ 1.23731864,  2.91766968],
       [ 0.37730323,  4.72014926],
       [ 4.56216652,  5.05882869],
       [ 3.6955702 ,  2.44550353],
       [ 1.63370019,  5.16619157],
       [10.49505539, 11.38343319],
       [10.42020043,  8.87963689],
       [10.71666831,  8.68699844],
       [14.85192003, 11.06524778],
       [12.70516146,  8.53392134],
       [10.88464926,  9.1646559 ],
       [13.73402563, 11.53691185],
       [10.82438817, 11.78955707],
       [11.24654249,  9.07758307],
       [13.86478611,  9.25591543]])
```

In [6]:

```
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()

```

In [7]:

```
def euclidean_distance(X, Y):
 return np.sqrt(np.sum(np.power(X - Y, 2), axis=1))

```

In [25]:

```
def cluster_data(solution, solution_idx):
    global num_cluster, data
    feature_vector_length = data.shape[1]
    cluster_centers = []
    all_clusters_dists = []
    clusters = []
    clusters_sum_dist = []

    for clust_idx in range(num_clusters):
        cluster_centers.append(solution[feature_vector_length*clust_idx:feature_vector_length*(clust
        cluster_center_dists = euclidean_distance(data, cluster_centers[clust_idx])
        all_clusters_dists.append(np.array(cluster_center_dists))

    cluster_centers = np.array(cluster_centers)
    all_clusters_dists = np.array(all_clusters_dists)

    cluster_indices = np.argmin(all_clusters_dists, axis=0)
    for clust_idx in range(num_clusters):
            clusters.append(np.where(cluster_indices == clust_idx)[0])
            if len(clusters[clust_idx]) == 0:
                clusters_sum_dist.append(0)
            else:
                clusters_sum_dist.append(np.sum(all_clusters_dists[clust_idx, clusters[clust_idx]]))
    clusters_sum_dist = np.array(clusters_sum_dist)

    return cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist
```

In [26]:

```
def fitness_func(ga_instance,solution, solution_idx):
    _, _, _, _, clusters_sum_dist = cluster_data(solution, solution_idx)
    fitness = 1.0 / (np.sum(clusters_sum_dist) + 0.00000001)
    return fitness
```

In [27]:

```
num_clusters = 2
num_genes = num_clusters * data.shape[1]
ga_instance = pygad.GA(num_generations=100,
                       sol_per_pop=10,
                       num_parents_mating=5,
                       init_range_low=-6,
                       init_range_high=20,
                       keep_parents=2,
                       num_genes=num_genes,
                       fitness_func=fitness_func,

                       suppress_warnings=True)
ga_instance.run()
```

In [28]:

```
best_solution, best_solution_fitness, best_solution_idx = ga_instance.best_solution()
print("Best solution is {bs}".format(bs=best_solution))
print("Fitness of the best solution is {bsf}".format(bsf=best_solution_fitness))
print("Best solution found after {gen} generations".format(gen=ga_instance.best_solution_generation))
```

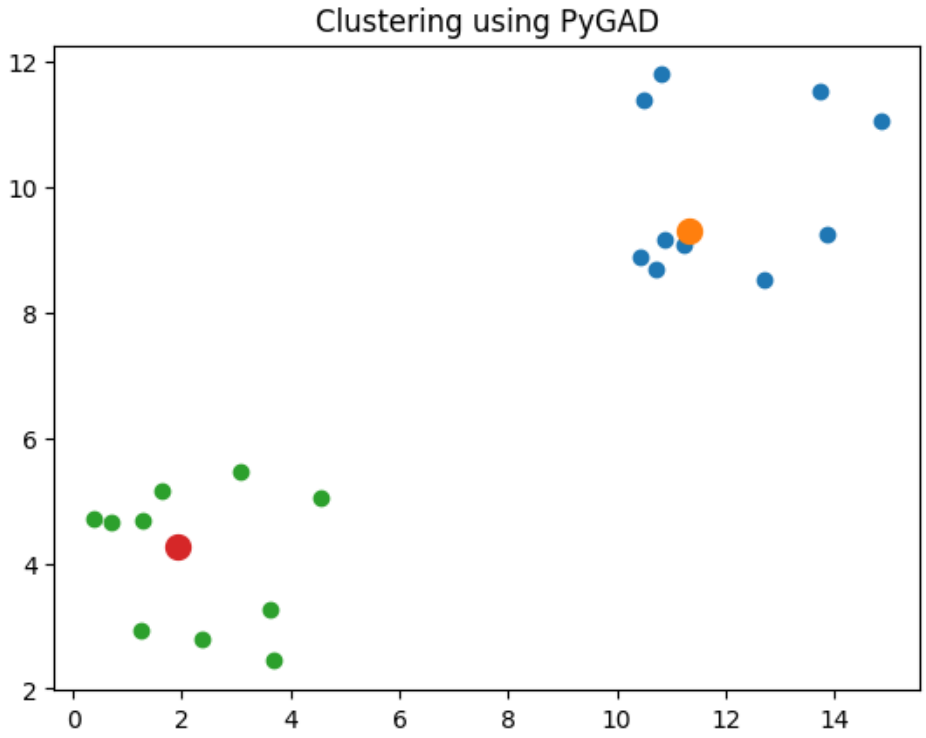
```
Best solution is [11.33809485  9.31276481  1.92397849  4.27798275]
Fitness of the best solution is 0.0283305456175161
Best solution found after 67 generations
```

In [31]:

```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist= cluster_data(best_
```

In [32]:

```
for cluster_idx in range(num_clusters):
 cluster_x = data[clusters[cluster_idx], 0]
 cluster_y = data[clusters[cluster_idx], 1]
 matplotlib.pyplot.scatter(cluster_x, cluster_y)
 matplotlib.pyplot.scatter(cluster_centers[cluster_idx, 0], cluster_centers[cluster_idx, 1], linewid
matplotlib.pyplot.title("Clustering using PyGAD")
matplotlib.pyplot.show()
```



In [ ]:

```

```