

In [1]:

```
pip install pygad
```

Requirement already satisfied: pygad in c:\users\shaik\anaconda3\lib\site-packages (3.0.1)
Requirement already satisfied: cloudpickle in c:\users\shaik\anaconda3\lib\site-packages (from pygad) (2.0.0)
Requirement already satisfied: matplotlib in c:\users\shaik\anaconda3\lib\site-packages (from pygad) (3.5.1)
Requirement already satisfied: numpy in c:\users\shaik\anaconda3\lib\site-packages (from pygad) (1.21.5)
Requirement already satisfied: cyclor>=0.10 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (1.3.2)
Requirement already satisfied: packaging>=20.0 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (21.3)
Requirement already satisfied: pillow>=6.2.0 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (9.0.1)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (3.0.4)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\shaik\anaconda3\lib\site-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\shaik\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib->pygad) (1.16.0)
Note: you may need to restart the kernel to use updated packages.

In [2]:

```
import numpy
import matplotlib.pyplot
import pygad
```

In [3]:

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

In [4]:

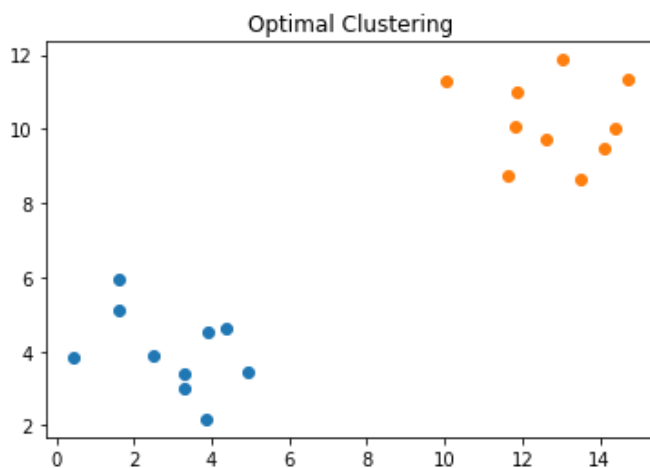
```
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

Out[4]:

```
array([[ 3.87984272,  2.17553359],
       [ 3.27817094,  3.41254597],
       [ 1.62414061,  5.11518023],
       [ 4.92701481,  3.43153179],
       [ 0.45190221,  3.84136845],
       [ 2.49207353,  3.89208332],
       [ 3.90250838,  4.54046904],
       [ 1.60076338,  5.94720009],
       [ 4.36554503,  4.62599069],
       [ 3.31362929,  2.9960262 ],
       [10.04037128, 11.2867089 ],
       [13.5055343 ,  8.64357774],
       [14.70227583, 11.34188797],
       [14.40445767, 10.03581097],
       [14.0956958 ,  9.48642218],
       [11.63103546,  8.73304732],
       [11.86707363, 11.01002445],
       [12.6122838 ,  9.7066366 ],
       [13.04846307, 11.87308429],
       [11.83003133, 10.06032469]])
```

In [5]:

```
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 numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
```

In [8]:

```
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_idx+1)])
        cluster_center_dists = euclidean_distance(data, cluster_centers[clust_idx])
        all_clusters_dists.append(numpy.array(cluster_center_dists))
    cluster_centers = numpy.array(cluster_centers)
    all_clusters_dists = numpy.array(all_clusters_dists)
    cluster_indices = numpy.argmin(all_clusters_dists, axis=0)
    for clust_idx in range(num_clusters):
        clusters.append(numpy.where(cluster_indices == clust_idx)[0])
        if len(clusters[clust_idx]) == 0:
            clusters_sum_dist.append(0)
        else:
            clusters_sum_dist.append(numpy.sum(all_clusters_dists[clust_idx, clusters[clust_idx]]))

    clusters_sum_dist = numpy.array(clusters_sum_dist)
    return cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist
```

In [9]:

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

In [10]:

```
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 [11]:

```
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))
```

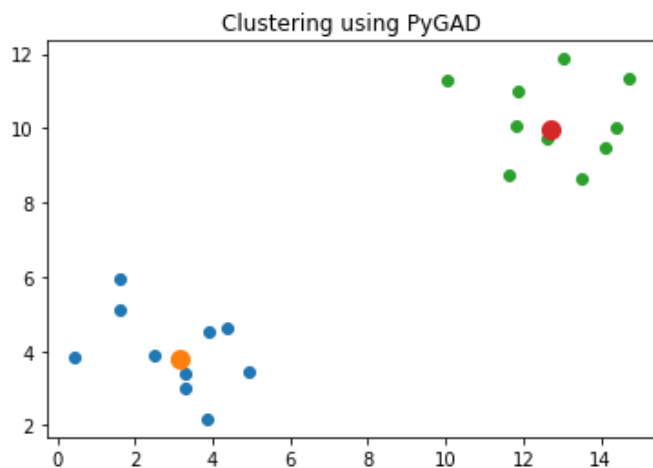
```
Best solution is [ 3.15786032  3.7784307  12.69267832  9.956237 ]
Fitness of the best solution is 0.031690055681857975
Best solution found after 86 generations
```

In [13]:

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

In [15]:

```
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], linev  
matplotlib.pyplot.title("Clustering using PyGAD")  
matplotlib.pyplot.show()
```



In []: