In [1]:
```
pip install pygad
```

```
Requirement already satisfied: pygad in c:\users\dell\appdata\local\programs\python\python311\lib\s
ite-packages (3.0.1)
Requirement already satisfied: cloudpickle in c:\users\dell\appdata\local\programs\python\python311
\lib\site-packages (from pygad) (2.2.1)
Requirement already satisfied: matplotlib in c:\users\dell\appdata\local\programs\python\python311
\lib\site-packages (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\dell\appdata\local\programs\python\python311\lib\s
ite-packages (from pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\dell\appdata\local\programs\python\pyth
on311\lib\site-packages (from matplotlib->pygad) (1.0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\dell\appdata\local\programs\python\python31
1\lib\site-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\dell\appdata\local\programs\python\pyt
hon311\lib\site-packages (from matplotlib->pygad) (4.39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dell\appdata\local\programs\python\pyt
hon311\lib\site-packages (from matplotlib->pygad) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\appdata\local\programs\python\pytho
n311\lib\site-packages (from matplotlib->pygad) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\dell\appdata\local\programs\python\python3
11\lib\site-packages (from matplotlib->pygad) (9.5.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\dell\appdata\local\programs\python\pyth
on311\lib\site-packages (from matplotlib->pygad) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\dell\appdata\local\programs\python
\python311\lib\site-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\dell\appdata\local\programs\python\python311\li
b\site-packages (from python-dateutil>=2.7->matplotlib->pygad) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

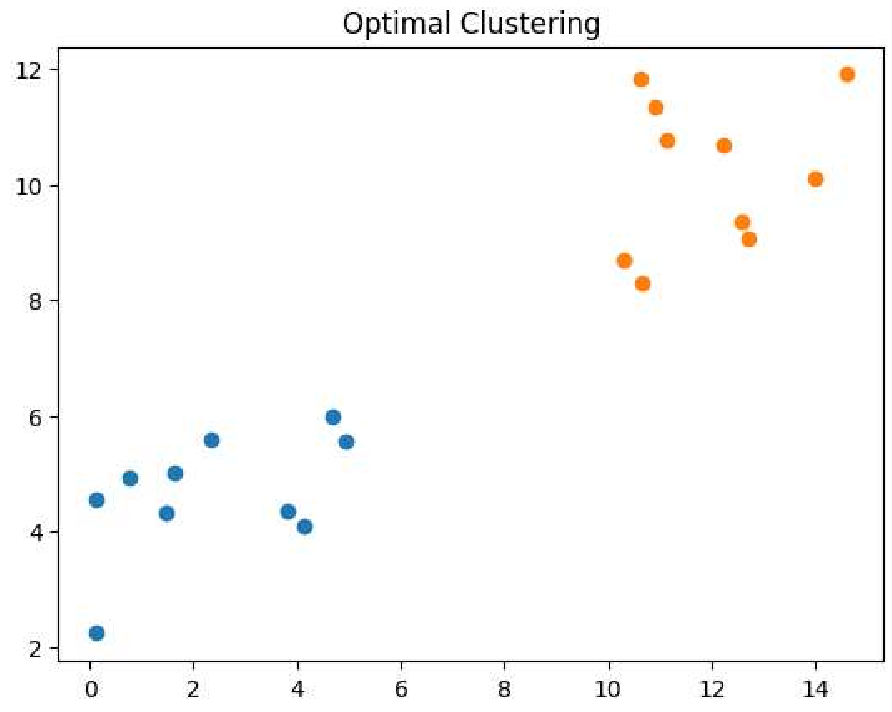
In [2]:
```
import numpy
import matplotlib.pyplot
import pygad
```

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]:
```python
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([[ 1.64247331,  5.01959231],
       [ 0.7731698 ,  4.92023091],
       [ 0.12051101,  2.23485528],
       [ 4.92856383,  5.56494236],
       [ 0.13766874,  4.55967157],
       [ 4.68141981,  5.97849355],
       [ 3.81380022,  4.35242205],
       [ 2.33242186,  5.59125285],
       [ 1.48119508,  4.32639122],
       [ 4.13611635,  4.09910399],
       [12.23479378, 10.67573147],
       [10.31057944,  8.70924322],
       [10.63448083, 11.84501831],
       [12.57872281,  9.3459005 ],
       [13.9994821 , 10.10663392],
       [10.63864179,  8.28208118],
       [11.12940417, 10.78043937],
       [14.59562943, 11.90890087],
       [10.91160103, 11.35569887],
       [12.7142163 ,  9.0792624 ]])
```

In [5]:
```python
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
```



In [6]:
```python
def euclidean_distance(X, Y):
 return numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
```

In [7]:
```
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(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 [8]:
```
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
```

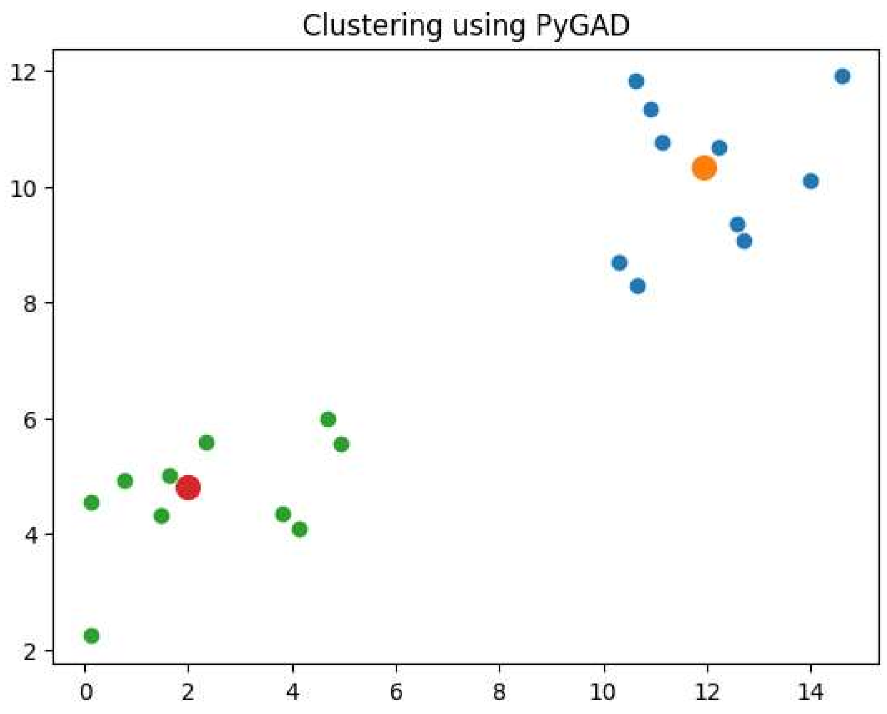
In [9]:
```
num_clusters = 2
num_genes = num_clusters * data.shape[1]
ga_instance = pygad.GA(num_generations=100,
                       sol_per_pop=10,
                       num_parents_mating=5,
                       init_range_low=-6,
                       init_range_high=20,
                       keep_parents=2,
                       num_genes=num_genes,
                       fitness_func=fitness_func,
                       suppress_warnings=True)
ga_instance.run()
```

In [10]:
```
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 [11.92415383 10.33225481  1.98804977  4.81172018]
Fitness of the best solution is 0.02800366103485835
Best solution found after 95 generations
```

In [12]:
```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist = cluster_data(bes
```

In [13]:
```
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], linew
matplotlib.pyplot.title("Clustering using PyGAD")
matplotlib.pyplot.show()
```



In [ ]: