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

```
Collecting pygad
  Downloading pygad-3.0.1-py3-none-any.whl (67 kB)
                                              0.0/68.0 kB ? eta -:--:--
     -----------------                      30.7/68.0 kB 660.6 kB/s eta 0:00:01
     ----------------------                 41.0/68.0 kB 393.8 kB/s eta 0:00:01
     ---------------------------------------- 68.0/68.0 kB 461.7 kB/s eta 0:00:00
Collecting cloudpickle (from pygad)
  Downloading cloudpickle-2.2.1-py3-none-any.whl (25 kB)
Requirement already satisfied: matplotlib in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-package
s (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-packages (fr
om pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-p
ackages (from matplotlib->pygad) (1.0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-packa
ges (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-
packages (from matplotlib->pygad) (4.39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-
packages (from matplotlib->pygad) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-pa
ckages (from matplotlib->pygad) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-pack
ages (from matplotlib->pygad) (9.5.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-p
ackages (from matplotlib->pygad) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\lenovo\appdata\local\programs\python\python311\lib\si
te-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\lenovo\appdata\local\programs\python\python311\lib\site-packages
(from python-dateutil>=2.7->matplotlib->pygad) (1.16.0)
Installing collected packages: cloudpickle, pygad
Successfully installed cloudpickle-2.2.1 pygad-3.0.1
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
import numpy
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 = 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 [5]:
```
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[5]:
```
array([[ 4.29668939,  3.79076917],
       [ 3.29148156,  5.42289331],
       [ 2.98845481,  5.40185007],
       [ 3.0212868 ,  4.01762675],
       [ 4.92269542,  2.26412578],
       [ 4.48641625,  3.02811601],
       [ 1.7211604 ,  5.15205475],
       [ 3.18080787,  5.83973112],
       [ 0.10205455,  2.07204374],
       [ 0.59066958,  5.85432153],
       [12.99755397,  8.05705357],
       [11.06929417,  9.33603165],
       [14.97502941,  9.70259134],
       [13.62837996, 11.39342622],
       [14.63580869, 10.51613534],
       [14.41805664, 10.11410223],
       [10.63175081, 11.14269492],
       [13.40600983, 10.14791016],
       [10.66054277, 11.5309783 ],
       [14.49431358, 11.76686545]])
```

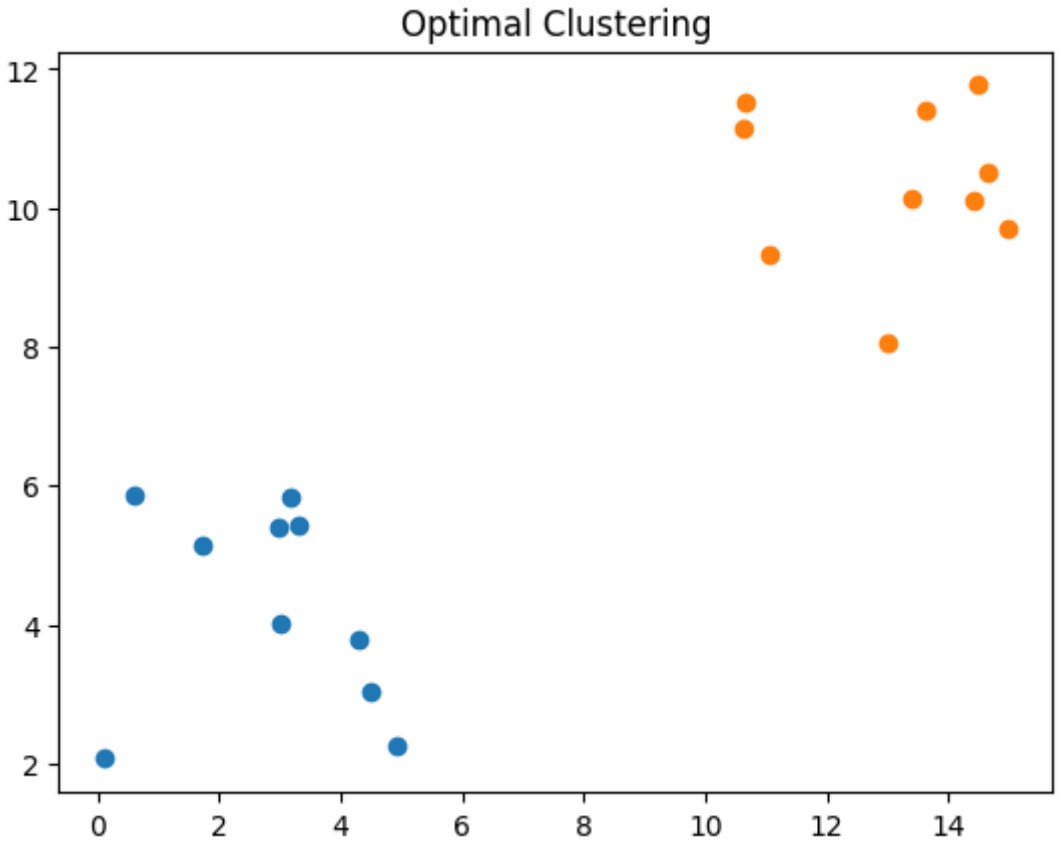
In [6]:
```
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

Out[6]:
```
array([[ 4.29668939,  3.79076917],
       [ 3.29148156,  5.42289331],
       [ 2.98845481,  5.40185007],
       [ 3.0212868 ,  4.01762675],
       [ 4.92269542,  2.26412578],
       [ 4.48641625,  3.02811601],
       [ 1.7211604 ,  5.15205475],
       [ 3.18080787,  5.83973112],
       [ 0.10205455,  2.07204374],
       [ 0.59066958,  5.85432153],
       [12.99755397,  8.05705357],
       [11.06929417,  9.33603165],
       [14.97502941,  9.70259134],
       [13.62837996, 11.39342622],
       [14.63580869, 10.51613534],
       [14.41805664, 10.11410223],
       [10.63175081, 11.14269492],
       [13.40600983, 10.14791016],
       [10.66054277, 11.5309783 ],
       [14.49431358, 11.76686545]])
```

In [8]:
```
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

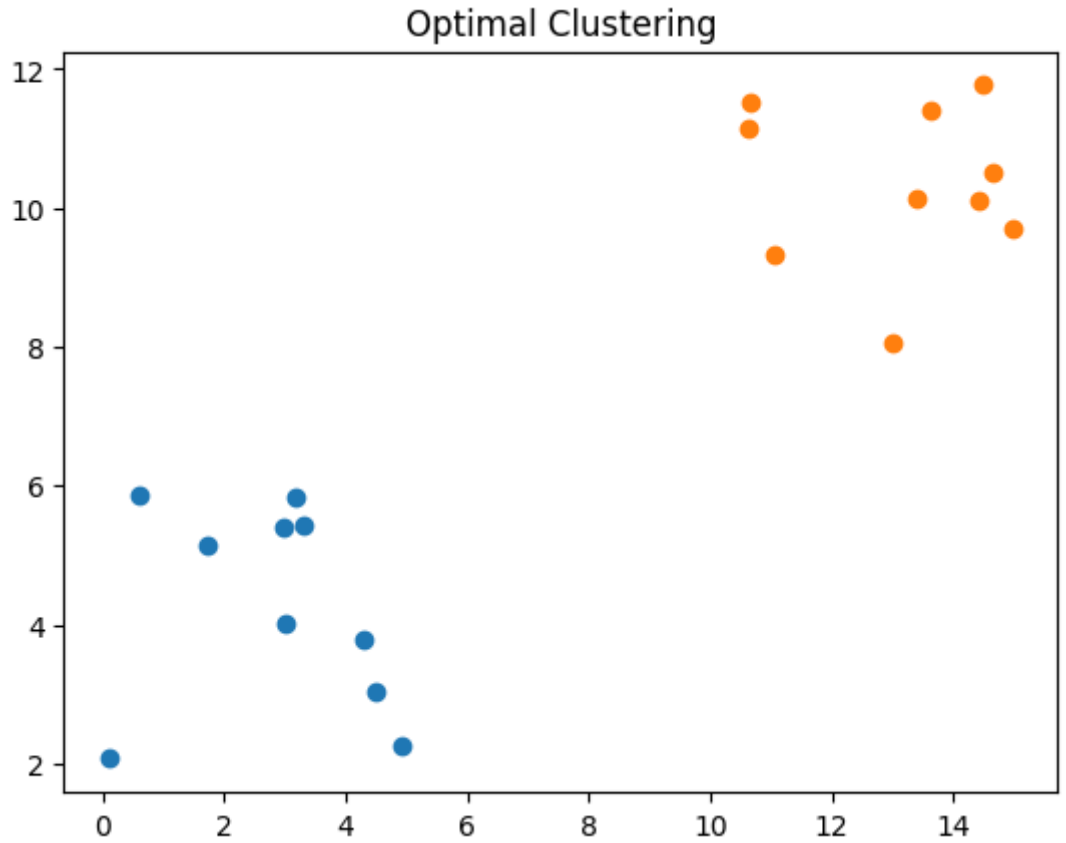
Out[8]:
```
array([[ 4.29668939,  3.79076917],
       [ 3.29148156,  5.42289331],
       [ 2.98845481,  5.40185007],
       [ 3.0212868 ,  4.01762675],
       [ 4.92269542,  2.26412578],
       [ 4.48641625,  3.02811601],
       [ 1.7211604 ,  5.15205475],
       [ 3.18080787,  5.83973112],
       [ 0.10205455,  2.07204374],
       [ 0.59066958,  5.85432153],
       [12.99755397,  8.05705357],
       [11.06929417,  9.33603165],
       [14.97502941,  9.70259134],
       [13.62837996, 11.39342622],
       [14.63580869, 10.51613534],
       [14.41805664, 10.11410223],
       [10.63175081, 11.14269492],
       [13.40600983, 10.14791016],
       [10.66054277, 11.5309783 ],
       [14.49431358, 11.76686545]])
```

In [9]:
```
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
```



In [10]:

```
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
def euclidean_distance(X, Y):
 return numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
```



In [42]:
```
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 [44]:
```
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
```

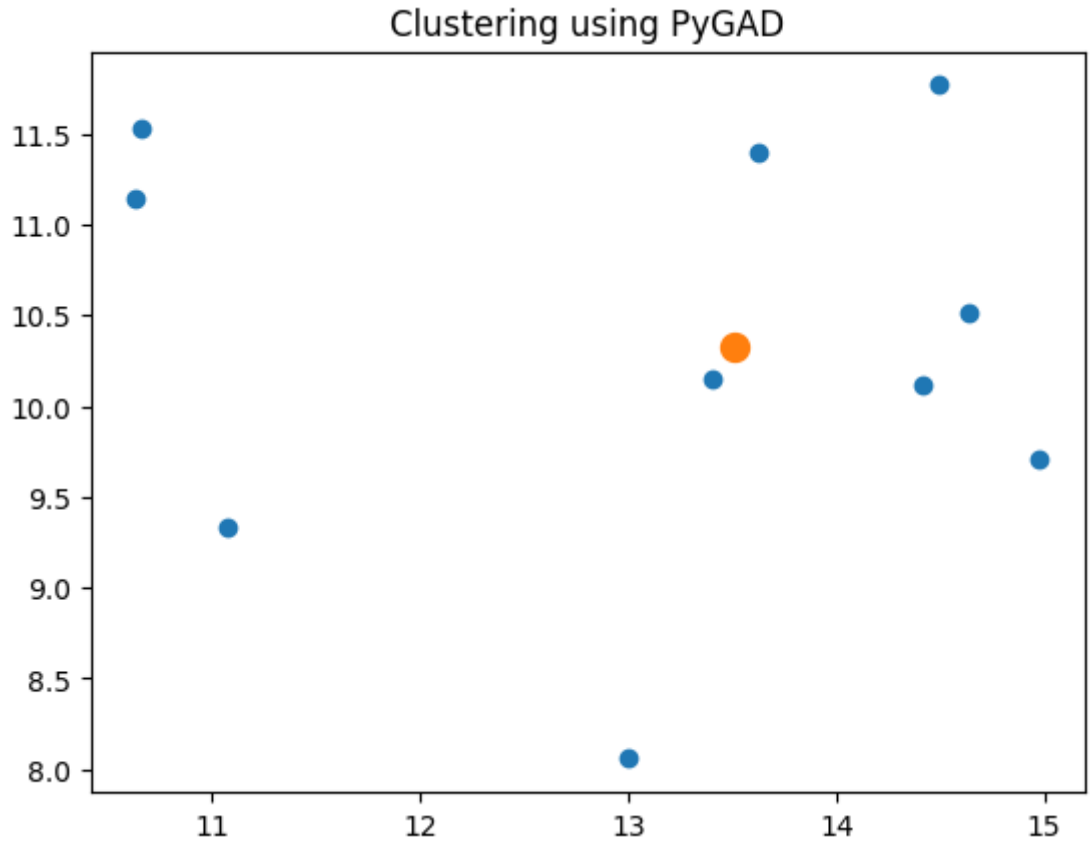
In [45]:
```
num_clusters = 2
num_genes = num_clusters * data.shape[1]
ga_instance = pygad.GA(num_generations=100,
 sol_per_pop=10,
num_parents_mating=5,
init_range_low=-6,
init_range_high=20,
keep_parents=2,
num_genes=num_genes,
fitness_func=fitness_func,
 suppress_warnings=True)
ga_instance.run()
```

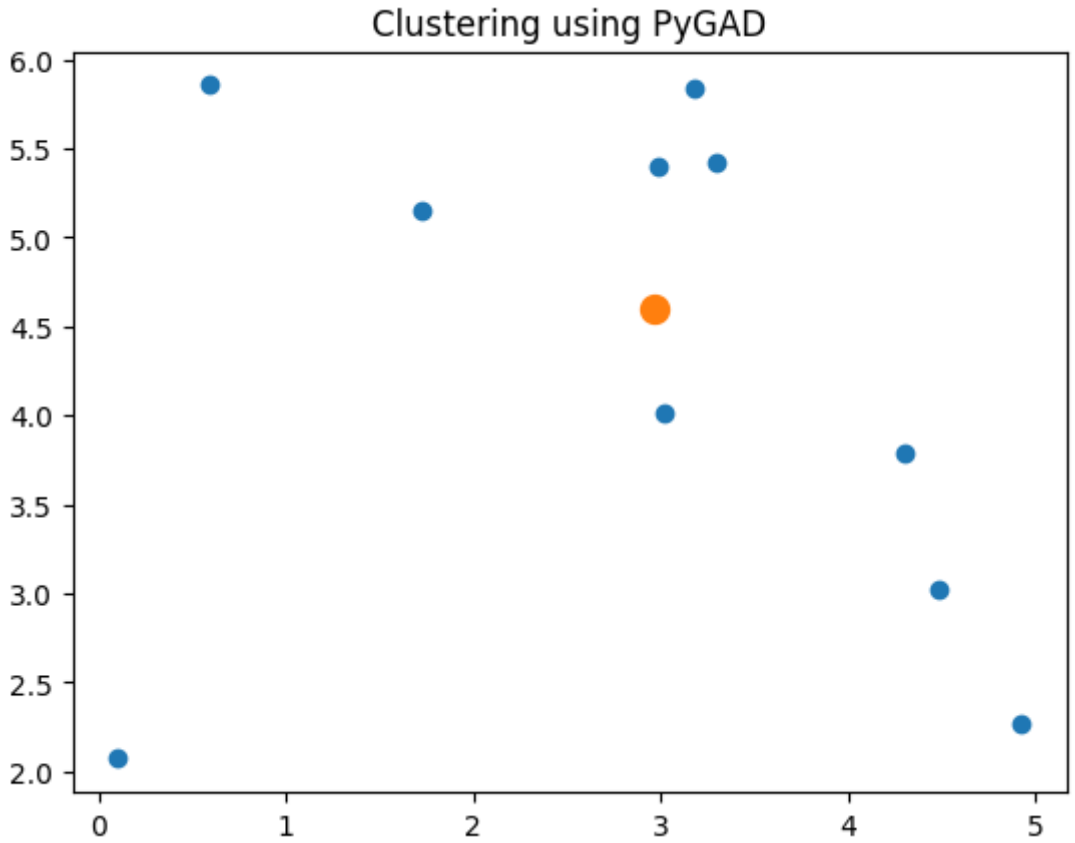
In [46]:
```
best_solution, best_solution_fitness, best_solution_idx = ga_instance.best_solution()
print("Best solution is {bs}".format(bs=best_solution))
print("Fitness of the best solution is {bsf}".format(bsf=best_solution_fitness))
print("Best solution found after {gen} generations".format(gen=ga_instance.best_solution_generation))
```

```
Best solution is [13.51433822 10.32743324  2.96330856  4.60173822]
Fitness of the best solution is 0.027833711082310194
Best solution found after 78 generations
```

In [47]:
```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist = cluster_data(best_solution, best_s
```

In [48]:
```
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], linewidths=5)
 matplotlib.pyplot.title("Clustering using PyGAD")
 matplotlib.pyplot.show()
```





In [ ]:

In [ ]: