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

```
Collecting pygad
  Downloading pygad-3.0.1-py3-none-any.whl (67 kB)
     -------------------------------------- 68.0/68.0 kB 246.6 kB/s eta 0:00:
00
Requirement already satisfied: numpy in c:\users\y.saranya\anaconda3\lib\site
-packages (from pygad) (1.23.5)
Requirement already satisfied: cloudpickle in c:\users\y.saranya\anaconda3\li
b\site-packages (from pygad) (2.0.0)
Requirement already satisfied: matplotlib in c:\users\y.saranya\anaconda3\lib
\site-packages (from pygad) (3.7.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\y.saranya\ana
conda3\lib\site-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\y.saranya\anaconda3
\lib\site-packages (from matplotlib->pygad) (9.4.0)
Requirement already satisfied: cycler>=0.10 in c:\users\y.saranya\anaconda3\l
ib\site-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\y.saranya\anacond
a3\lib\site-packages (from matplotlib->pygad) (3.0.9)
Requirement already satisfied: packaging>=20.0 in c:\users\y.saranya\anaconda
3\lib\site-packages (from matplotlib->pygad) (22.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\y.saranya\anacon
da3\lib\site-packages (from matplotlib->pygad) (1.4.4)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\y.saranya\anacon
da3\lib\site-packages (from matplotlib->pygad) (4.25.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\y.saranya\anacond
a3\lib\site-packages (from matplotlib->pygad) (1.0.5)
Requirement already satisfied: six>=1.5 in c:\users\y.saranya\anaconda3\lib\s
ite-packages (from python-dateutil>=2.7->matplotlib->pygad) (1.16.0)
Installing collected packages: pygad
Successfully installed pygad-3.0.1
Note: you may need to restart the kernel to use updated packages.
```

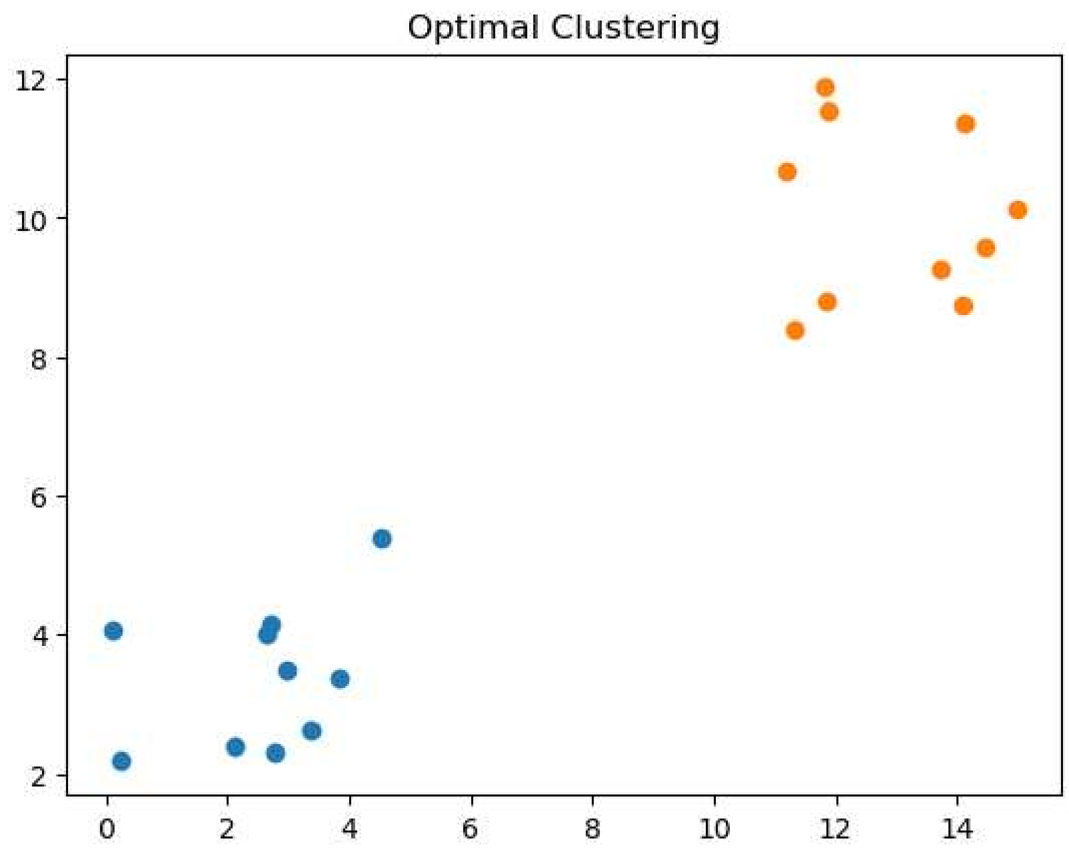
In [2]:
```
import numpy
import matplotlib.pyplot
import pygad
```

In [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_x
cluster1_x2 = numpy.random.random(size=(cluster1_num_samples))
cluster1_x2 = cluster1_x2 * (cluster1_x2_end - cluster1_x2_start) + cluster1_x


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_x
cluster2_x2 = numpy.random.random(size=(cluster2_num_samples))
cluster2_x2 = cluster2_x2 * (cluster2_x2_end - cluster2_x2_start) + cluster2_x
```

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([[ 4.52769009,  5.37980885],
       [ 2.12815788,  2.38432149],
       [ 3.36464956,  2.61703471],
       [ 0.23687437,  2.17555125],
       [ 0.10890786,  4.0561301 ],
       [ 2.71714394,  4.16305379],
       [ 2.79297603,  2.30111465],
       [ 3.81964883,  3.37940159],
       [ 2.6586036 ,  4.00826257],
       [ 2.96481303,  3.48444858],
       [14.46441675,  9.5724436 ],
       [14.12111216, 11.35694939],
       [14.97678256, 10.13921577],
       [14.09408369,  8.7542959 ],
       [11.83234217,  8.80990643],
       [11.20148315, 10.66489539],
       [11.31234548,  8.4016673 ],
       [13.73657544,  9.26398545],
       [11.8815429 , 11.52777077],
       [11.80582405, 11.8730264 ]])
```

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

In [9]:
```
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_ve
    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, c
 clusters_sum_dist = numpy.array(clusters_sum_dist)
 return cluster_centers, all_clusters_dists, cluster_indices, clusters, cluste
```

In [10]:
```
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
```

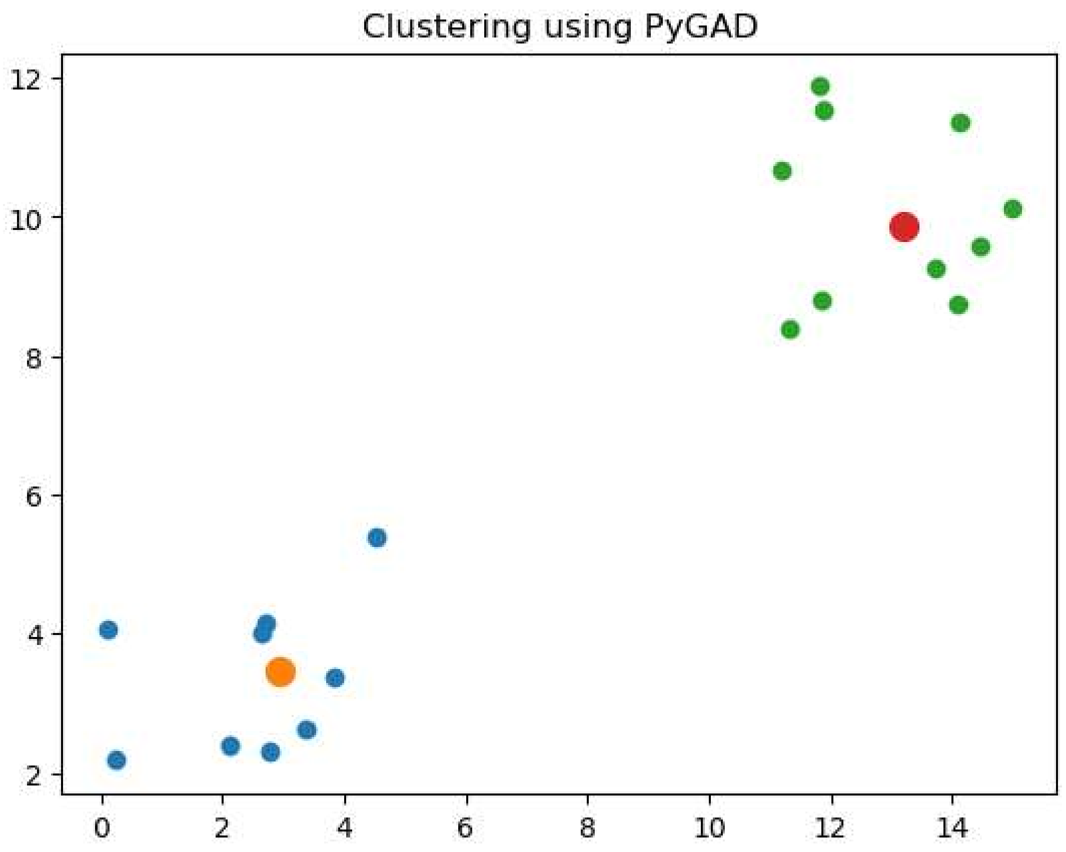
In [11]:
```
num_clusters = 2
num_genes = num_clusters * data.shape[1]
ga_instance = pygad.GA(num_generations=100,
                       sol_per_pop=10,
                       num_parents_mating=5,
                       init_range_low=-6,
                       init_range_high=20,
                       keep_parents=2,
                       num_genes=num_genes,
                       fitness_func=fitness_func,
                       suppress_warnings=True)
ga_instance.run()
```

In [12]:
```
best_solution, best_solution_fitness, best_solution_idx = ga_instance.best_sol
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.bes
```

```
Best solution is [ 2.9345784   3.46671698 13.19818494  9.8766525 ]
Fitness of the best solution is 0.031229773959915046
Best solution found after 99 generations
```

In [17]:
```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_d
```

In [18]:
```
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
matplotlib.pyplot.title("Clustering using PyGAD")
matplotlib.pyplot.show()
```



In [ ]: