In [1]:

```
pip install pygad
```

```
Collecting pygad
  Downloading pygad-3.0.1-py3-none-any.whl (67 kB)
     -------------------------------------- 68.0/68.0 kB 367.6 kB/s eta 0:00:00
Requirement already satisfied: numpy in c:\users\teju\anaconda3\lib\site-packages (from pygad) (1.23.5)
Requirement already satisfied: matplotlib in c:\users\teju\anaconda3\lib\site-packages (from pygad) (3.7.0)
Requirement already satisfied: cloudpickle in c:\users\teju\anaconda3\lib\site-packages (from pygad) (2.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->py
gad) (3.0.9)
Requirement already satisfied: packaging>=20.0 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->pyg
ad) (23.0)
Requirement already satisfied: cycler>=0.10 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->pygad)
(0.11.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->py
gad) (1.0.5)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib
->pygad) (2.8.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->p
ygad) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->p
ygad) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in c:\users\teju\anaconda3\lib\site-packages (from matplotlib->pyga
d) (9.4.0)
Requirement already satisfied: six>=1.5 in c:\users\teju\anaconda3\lib\site-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_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]:

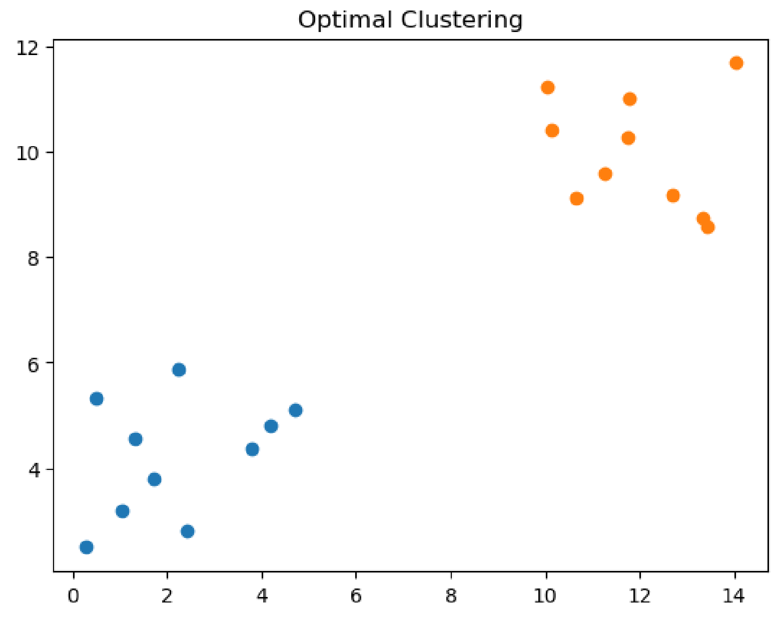
```
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

Out[4]:

```
array([[ 3.78719799,  4.38559715],
       [ 0.28742033,  2.50319599],
       [ 1.32244927,  4.5550102 ],
       [ 1.05192855,  3.19619002],
       [ 4.69579926,  5.10783567],
       [ 2.23859611,  5.86590847],
       [ 1.7198629 ,  3.81017365],
       [ 0.49018575,  5.34081295],
       [ 2.42135412,  2.81692614],
       [ 4.17997189,  4.82078988],
       [13.42611   ,  8.5849862 ],
       [13.33922078,  8.73497419],
       [11.75037604, 10.26424066],
       [11.79036335, 11.01524724],
       [10.13614501, 10.4188097 ],
       [12.69578563,  9.18725972],
       [10.03988086, 11.23560521],
       [14.03578426, 11.6806913 ],
       [10.65870633,  9.11748025],
       [11.26430934,  9.58208317]])
```

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 [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))
```

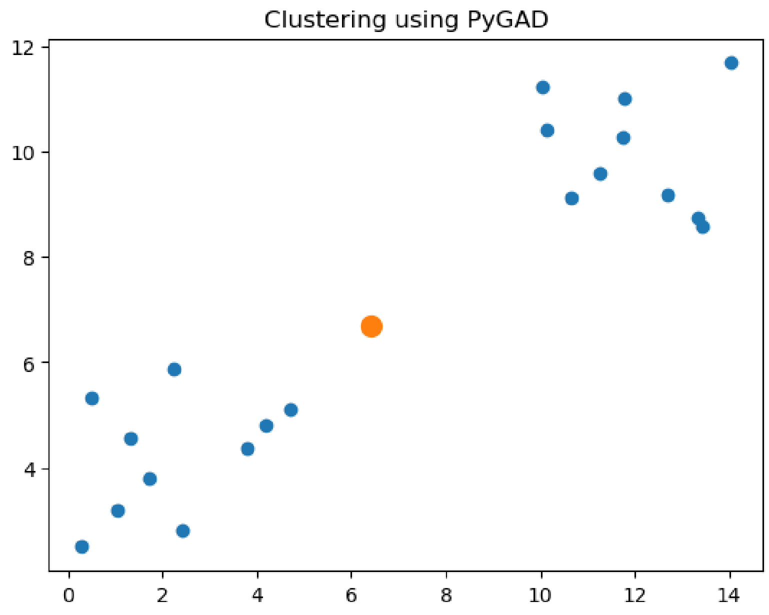
```
Best solution is [ 6.42586494  6.70890745 -2.65706702 -6.35361014]
Fitness of the best solution is 0.008712082050436575
Best solution found after 52 generations
```

In [12]:

```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist=cluster_data(best_solution, best_solution_idx
```

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



```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[13], line 2
      1 for cluster_idx in range(num_clusters):
----> 2  cluster_x = data[clusters[cluster_idx], 0]
      3  cluster_y = data[clusters[cluster_idx], 1]
      4  matplotlib.pyplot.scatter(cluster_x, cluster_y)

IndexError: list index out of range
```

In [ ]: