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

```
Requirement already satisfied: pygad in c:\users\lenovo\appdata\local\program
s\python\python311\lib\site-packages (3.0.1)
Requirement already satisfied: cloudpickle in c:\users\lenovo\appdata\local\p
rograms\python\python311\lib\site-packages (from pygad) (2.2.1)
Requirement already satisfied: matplotlib in c:\users\lenovo\appdata\local\pr
ograms\python\python311\lib\site-packages (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\lenovo\appdata\local\program
s\python\python311\lib\site-packages (from pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\lenovo\appdata\lo
cal\programs\python\python311\lib\site-packages (from matplotlib->pygad) (1.
0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\lenovo\appdata\local
\programs\python\python311\lib\site-packages (from matplotlib->pygad) (0.11.
0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\lenovo\appdata\l
ocal\programs\python\python311\lib\site-packages (from matplotlib->pygad) (4.
39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\lenovo\appdata\l
ocal\programs\python\python311\lib\site-packages (from matplotlib->pygad) (1.
4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\lenovo\appdata\loc
al\programs\python\python311\lib\site-packages (from matplotlib->pygad) (23.
1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\lenovo\appdata\local
\programs\python\python311\lib\site-packages (from matplotlib->pygad) (9.5.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\lenovo\appdata\lo
cal\programs\python\python311\lib\site-packages (from matplotlib->pygad) (3.
0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\lenovo\appdat
a\local\programs\python\python311\lib\site-packages (from matplotlib->pygad)
(2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\lenovo\appdata\local\prog
rams\python\python311\lib\site-packages (from python-dateutil>=2.7->matplotli
b->pygad) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

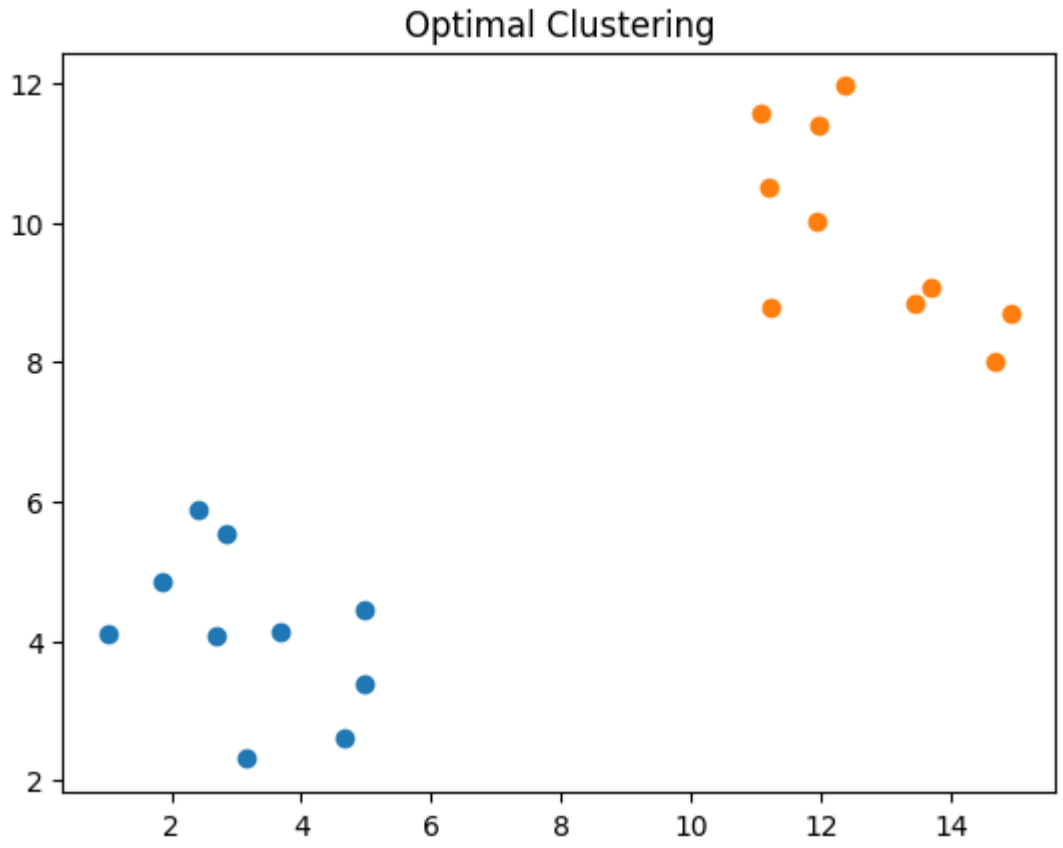
In [2]:
```
import numpy
import matplotlib.pyplot
import pygad
```

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
cluster1_x2 = numpy.random.random(size=(cluster1_num_samples))
cluster1_x2 = cluster1_x2 * (cluster1_x2_end - cluster1_x2_start) + cluster1_x2
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
cluster2_x2 = numpy.random.random(size=(cluster2_num_samples))
cluster2_x2 = cluster2_x2 * (cluster2_x2_end - cluster2_x2_start) + cluster2_x2
```

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([[ 1.0252692 ,  4.09960515],
       [ 4.98299521,  3.39403624],
       [ 3.16014679,  2.30775115],
       [ 2.68611054,  4.06337258],
       [ 1.85887087,  4.83755174],
       [ 4.98178687,  4.44627321],
       [ 3.66816275,  4.13845642],
       [ 4.67481645,  2.61807804],
       [ 2.84389888,  5.54117974],
       [ 2.41487373,  5.87521008],
       [11.18617787, 10.50130368],
       [12.35797028, 11.95640301],
       [13.44938962,  8.84918388],
       [14.91790909,  8.70220088],
       [13.6920485 ,  9.06803071],
       [11.23385218,  8.77682536],
       [11.94525444, 10.00955349],
       [14.67554879,  8.0178914 ],
       [11.96934378, 11.40718162],
       [11.06547982, 11.56720907]])
```

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 [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:featur
        cluster_center_dists = euclidean_distance(data, cluster_centers[clust_
        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[clus
        clusters_sum_dist = numpy.array(clusters_sum_dist)
        return cluster_centers, all_clusters_dists, cluster_indices, clusters,
```

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_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
```

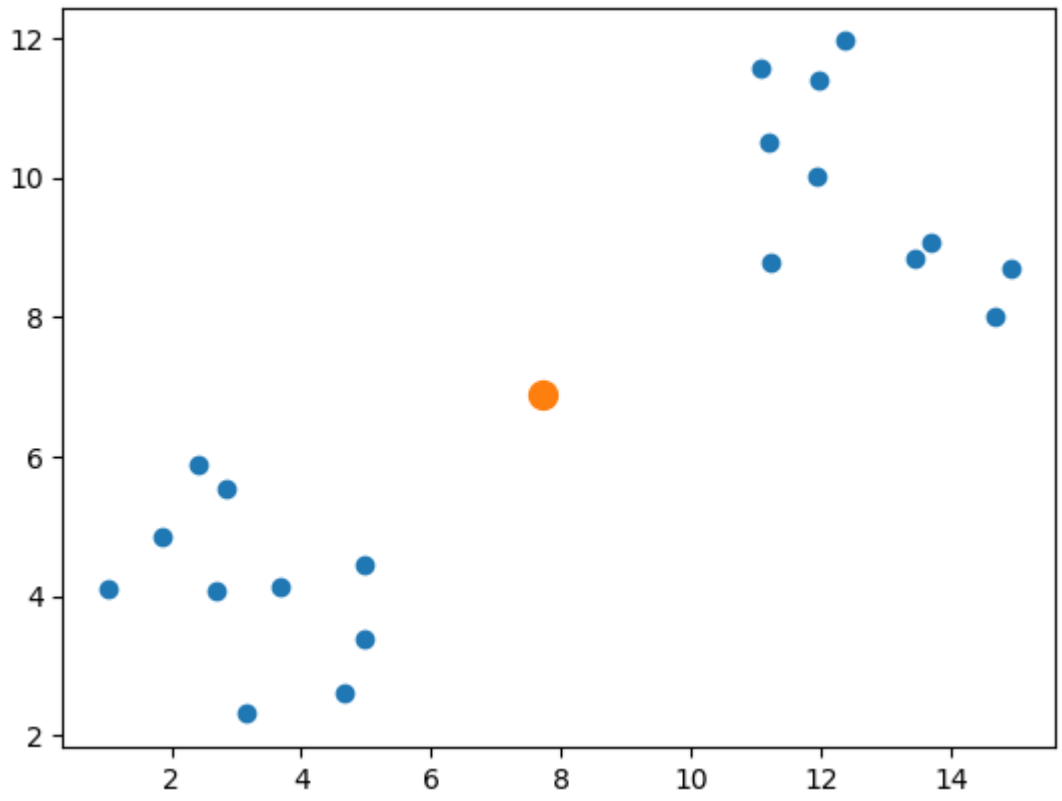
```
Best solution is [ 7.70413752  6.89763411 16.28451668 20.31042537]
Fitness of the best solution is 0.008744003836695475
Best solution found after 99 generations
```

In [11]:
```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_d
```

In [12]:
```
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()
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[12], line 5
      3     cluster_y = data[clusters[cluster_idx],1]
      4     matplotlib.pyplot.scatter(cluster_x, cluster_y)
----> 5     matplotlib.pyplot.scatter(cluster_centers[cluster_idx,0], cluster
_centers[cluster_idx,1], linewidths=5)
      6 matplotlib.pyplot.title("Clustering using PyGAD")
      7 matplotlib.pyplot.show()

IndexError: index 1 is out of bounds for axis 0 with size 1
```



In [ ]:

In [ ]:

In [ ]: