In [43]:
```
pip install pygad
```

```
Requirement already satisfied: pygad in c:\users\chinta pavani\appdata\lo
cal\programs\python\python311\lib\site-packages (3.0.1)Note: you may need
to restart the kernel to use updated packages.

Requirement already satisfied: cloudpickle in c:\users\chinta pavani\appd
ata\local\programs\python\python311\lib\site-packages (from pygad) (2.2.
1)
Requirement already satisfied: matplotlib in c:\users\chinta pavani\appda
ta\local\programs\python\python311\lib\site-packages (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\chinta pavani\appdata\lo
cal\programs\python\python311\lib\site-packages (from pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\chinta pavan
i\appdata\local\programs\python\python311\lib\site-packages (from matplot
lib->pygad) (1.0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\chinta pavani\app
data\local\programs\python\python311\lib\site-packages (from matplotlib->
pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\chinta pavan
i\appdata\local\programs\python\python311\lib\site-packages (from matplot
```

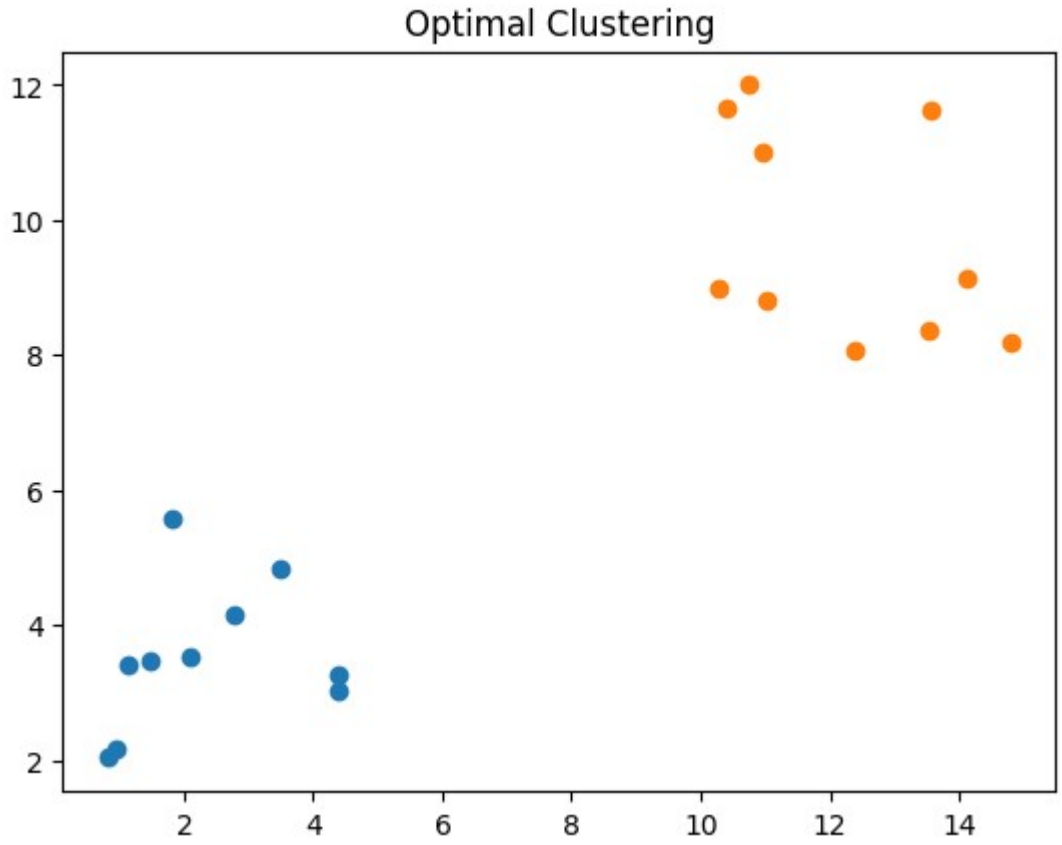
In [44]:
```
import numpy
import matplotlib.pyplot
import pygad
```

In [45]:
```
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) + cluste
cluster1_x2 = numpy.random.random(size=(cluster1_num_samples))
cluster1_x2 = cluster1_x2 * (cluster1_x2_end - cluster1_x2_start) + cluste
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) + cluste
cluster2_x2 = numpy.random.random(size=(cluster2_num_samples))
cluster2_x2 = cluster2_x2 * (cluster2_x2_end - cluster2_x2_start) + cluste
```

In [46]:
```
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[46]:
```
array([[ 0.94967446,  2.17533983],
       [ 1.81238873,  5.56360421],
       [ 1.1337907 ,  3.42300445],
       [ 4.37889977,  3.0230836 ],
       [ 2.08705237,  3.53723311],
       [ 1.46593602,  3.47108224],
       [ 0.82223992,  2.03405237],
       [ 2.77976982,  4.14312448],
       [ 3.4883491 ,  4.83573169],
       [ 4.37714651,  3.25838175],
       [11.015522  ,  8.81060409],
       [10.74679591, 11.996733  ],
       [14.79859085,  8.18366598],
       [12.38811227,  8.06960769],
       [10.28878516,  8.98518618],
       [13.52909535,  8.37347725],
       [10.95020021, 11.01559026],
       [13.56113256, 11.61672966],
       [14.13027625,  9.14620113],
       [10.38802154, 11.66358198]])
```

In [47]:
```
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
```



In [48]:
```
def euclidean_distance(X, Y):
    return numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
```

In [49]:
```
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:fe
        cluster_center_dists = euclidean_distance(data, cluster_centers[cl
        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_id
    clusters_sum_dist = numpy.array(clusters_sum_dist)
    return cluster_centers, all_clusters_dists, cluster_indices, clusters,
```

In [50]:
```
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 [51]:
```
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 [52]:
```
best_solution, best_solution_fitness, best_solution_idx = ga_instance.best
print("Best solution is {bs}".format(bs=best_solution))
print("Fitness of the best solution is {bsf}".format(bsf=best_solution_fit
print("Best solution found after {gen} generations".format(gen=ga_instance
```

```
Best solution is [ 2.14950979  3.56668145 12.06979512  9.5896742 ]
Fitness of the best solution is 0.027408195183288634
Best solution found after 97 generations
```

In [55]:
```
luster_centers, all_clusters_dists, cluster_indices, clusters, clusters_su
```

In [66]:

```
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_cen
matplotlib.pyplot.title("Clustering using PyGAD")
matplotlib.pyplot.show()
```

```
---------------------------------------------------------------------------
--
NameError                                 Traceback (most recent call las
t)
Cell In[66], 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], cl
uster_centers[cluster_idx, 1], linewidths=5)
      6 matplotlib.pyplot.title("Clustering using PyGAD")
      7 matplotlib.pyplot.show()

NameError: name 'cluster_centers' is not defined
```