

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

```
Requirement already satisfied: pygad in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (3.0.1)
Requirement already satisfied: cloudpickle in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from pygad) (2.2.1)
Requirement already satisfied: matplotlib in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (1.0.7)
Requirement already satisfied: cyclers>=0.10 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (4.39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (9.5.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\pappu\appdata\local\programs\python\python310\lib\site-packages (from python-dateutil>=2.7->matplotlib->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_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

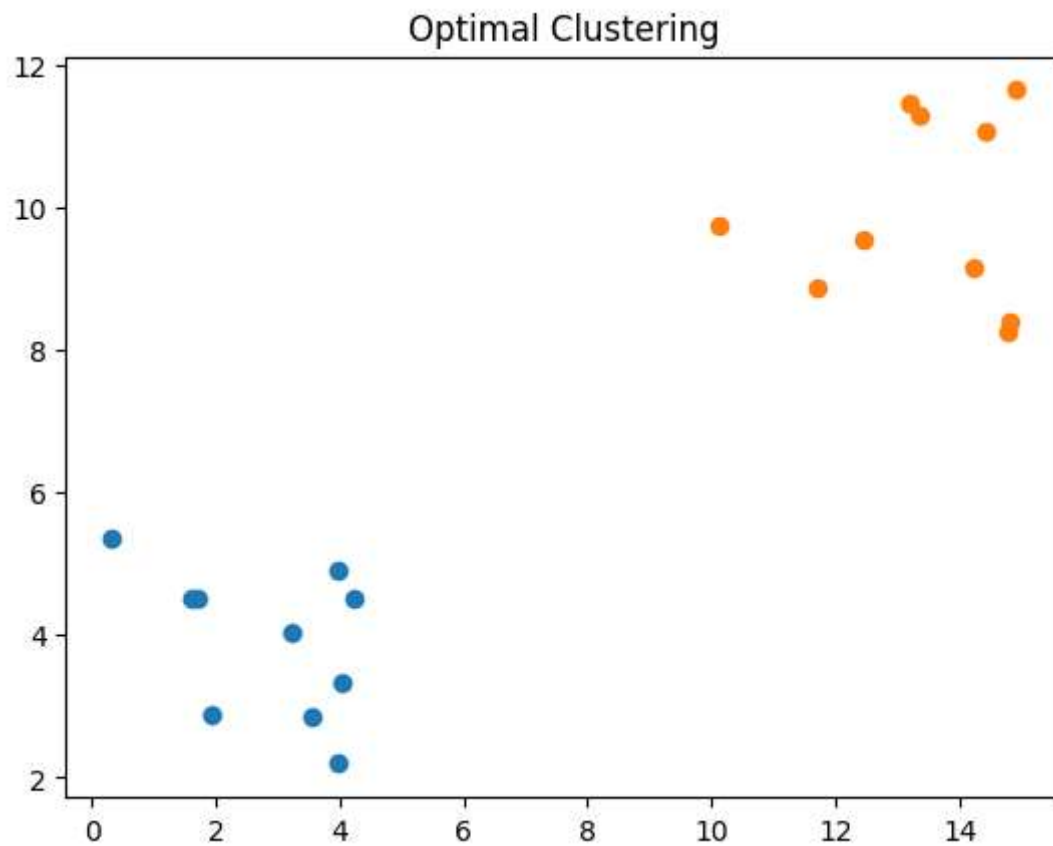
```

```

Out[4]: array([[ 3.98614653,  4.91778227],
 [ 0.30993678,  5.35113462],
 [ 3.9683663 ,  2.19461332],
 [ 3.54132733,  2.84525763],
 [ 4.02762821,  3.32501318],
 [ 1.93047602,  2.8734503 ],
 [ 3.23063635,  4.03217665],
 [ 1.62084844,  4.50760633],
 [ 4.24102438,  4.50703576],
 [ 1.69379197,  4.50904761],
 [14.80159799,  8.40237069],
 [14.21824261,  9.16270516],
 [12.45459967,  9.53808609],
 [13.3515838 , 11.30032575],
 [11.70396364,  8.87159108],
 [10.12518694,  9.75794394],
 [14.90619678, 11.64305423],
 [14.79189381,  8.24059524],
 [14.43686707, 11.07648656],
 [13.18213214, 11.44855441]])

```

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

```

In [11]: 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 [12]: 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 [13]: 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 [14]: 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 [13.55694755 10.05435194  3.17296235  3.92062963]
Fitness of the best solution is 0.03015795311899686
Best solution found after 50 generations
```

```
In [27]: cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_di
```

```
In [28]: 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()
```

