PURDUE UNIVERSITY
FORT WAYNE

Clustering Using Genetic Algorithm

Prepared for
Meherdad Hajirbabi,
Purdue University, Fort Wayne

<u>Prepared by</u>
Rohan Patel

December 11 2022

Introduction

A genetic algorithm is a type of evolutionary algorithm that draws its inspiration from the idea of natural selection. Genetic algorithms frequently employ biologically inspired operators including mutation, crossover, and selection to produce high-quality solutions to optimization and search issues. Solving sudoku puzzles, hyperparameter optimization, and decision tree performance improvement are a few examples of GA applications. Whereas, Machine learning is the type of algorithm that learns the patterns from historical data and predict future outcomes. This project helps in understanding if the genetic algorithm can be used to learn patterns from the data. This sounds impossible at first but is actually possible. In Machine learning, there are 2 types of approaches which are Supervised learning and Unsupervised Learning.The training data in supervised machine learning is already labeled, so each occurrence of the data has a matching output (s). The data in unsupervised machine learning is labelless. Clustering is an unsupervised learning problem in which each data instance must have the best label assigned to it.

Clustering

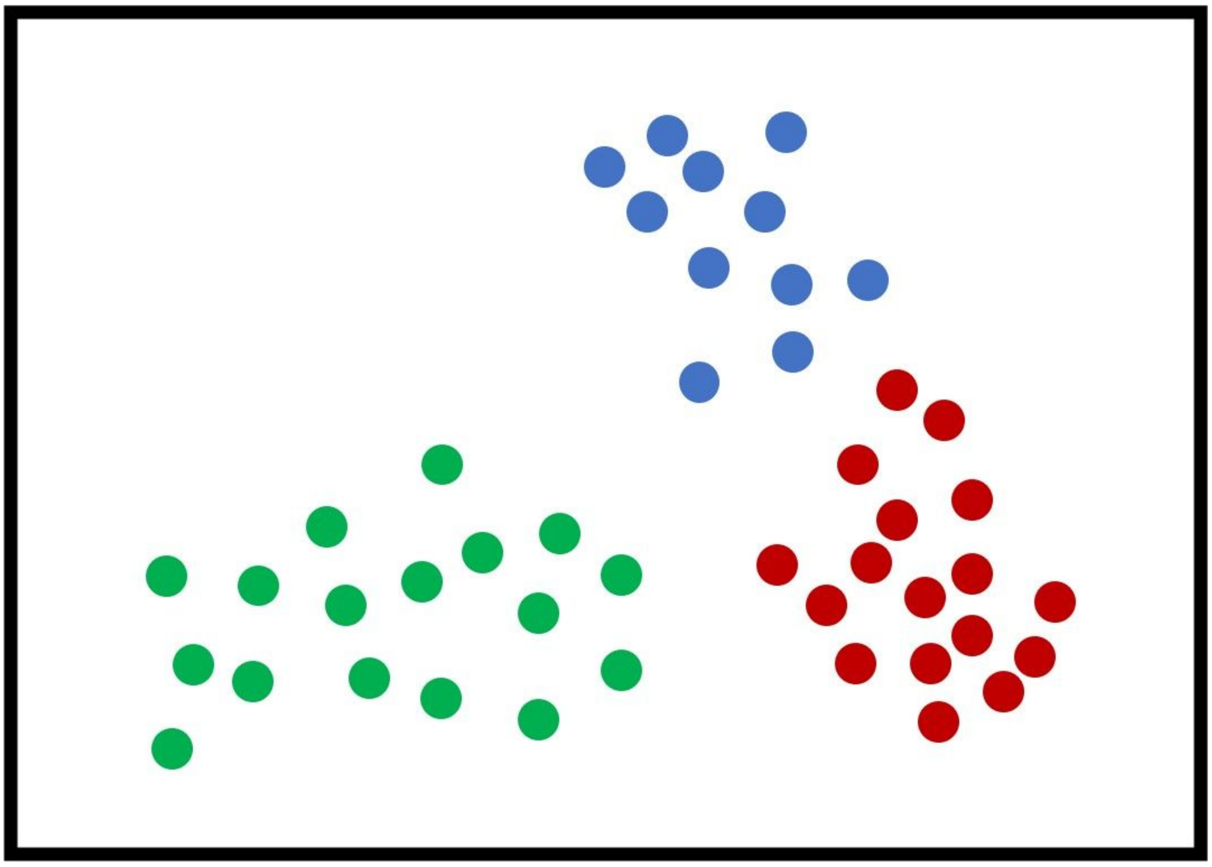
The process of clustering involves grouping the population or data points into a number of groups so that the data points within each group are more similar to one another than the data points within other groups. Simply said, the goal is to sort into clusters any groups of people that share similar characteristics. There are many Clustering algorithms to choose from but some of the most commonly used are K-Means, DB-Scan, Gaussian Mixture Model Etc.

K-Means Clustering Algorithm

K-Means The most well-known clustering technique, clustering, involves grouping instances in an effort to reduce variation within each cluster. It is the most straightforward unsupervised learning approach and is centroid-based. The technique aims to reduce data point variation inside a cluster.

Algorithm Working

K-means operates by choosing an initial center for each of the K clusters. Keep in mind that the K-means algorithm is highly sensitive to those initial centers and that altering the first centers may result in different outcomes. The distances between each sample and the three centers are then calculated, and each sample is assigned to the cluster of the closest center. The aim is to reduce the overall distance between the centers of all samples. The K-means algorithm adjusts the centers of its K clusters based on the existing clusters by averaging all samples inside each cluster. The cluster centers are moved from their existing positions to new ones when the new centers are computed in an effort to form new, improved clusters. The K-means algorithm keeps generating the centers of new clusters and allocating the samples to the cluster from which the distance is the shortest until it discovers that the new centers are precisely the same as the existing centers.



The Elbow Method:

The Elbow method chooses the optimum value of clusters by fitting the model with the range of values of k i.e no of clusters. A graph is plotted between the sum of squared error and the no of centroids present. The optimal number of clusters is then chosen when the model starts to converge, and the error doesn't decrease rapidly.

Clustering Using Genetic Algorithm

The genetic algorithm is an optimization technique that uses a population of more than one solution to find a solution to a given issue. The genetic algorithm not only finds a solution but also the globally optimal solution by applying random modifications to the solution in different directions.

The genetic algorithm follows these steps to find the best solution:

1. Initialize a population of solutions.
2. Calculate the fitness value of the solutions in the population.
3. Select the best solutions (with the highest fitness values) as parents.
4. Mate the selected parents using crossover and mutation.
5. Create a new population.
6. Repeat steps 2 to 5 for a number of generations, or until a condition is met.

The Fitness Function

The fitness function is a method of evaluating the answer. Because the genetic algorithm employs a fitness maximization function, the aim is to discover the solution with the highest fitness value. The genetic algorithm, which is based on the K-means method, computes the sum of distances between each sample and its cluster center using the following equation, where the Euclidean distance is used:

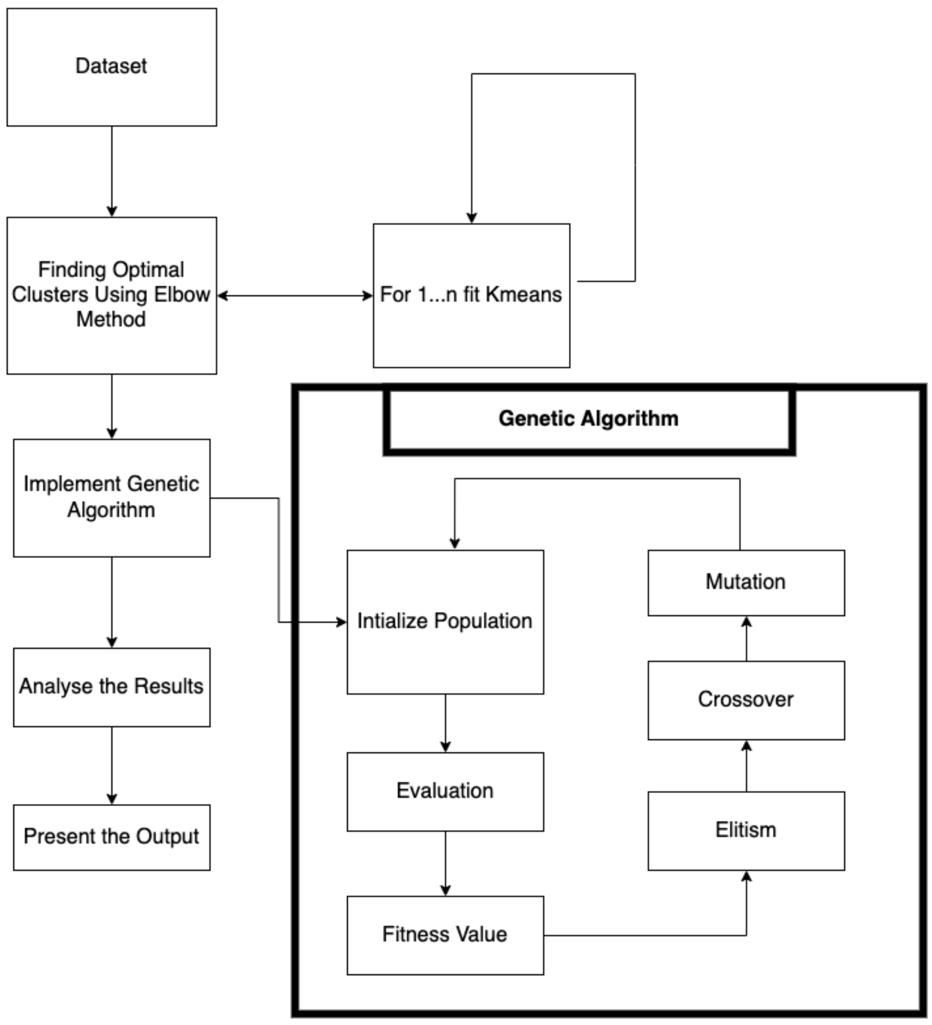
$$fitness = \frac{1}{\sum_{k=1}^{N_c} \sum_{j=1}^{N_k} \sqrt{\sum_{i=1}^{F} (C_{ki} - P_{jj})^2}}$$

Where :

- Nc is the number of clusters.
- Nk is the number of samples within the cluster k.
- F is the number of features representing the number of samples, which is 2 in our example.
- Ck is the center of cluster k

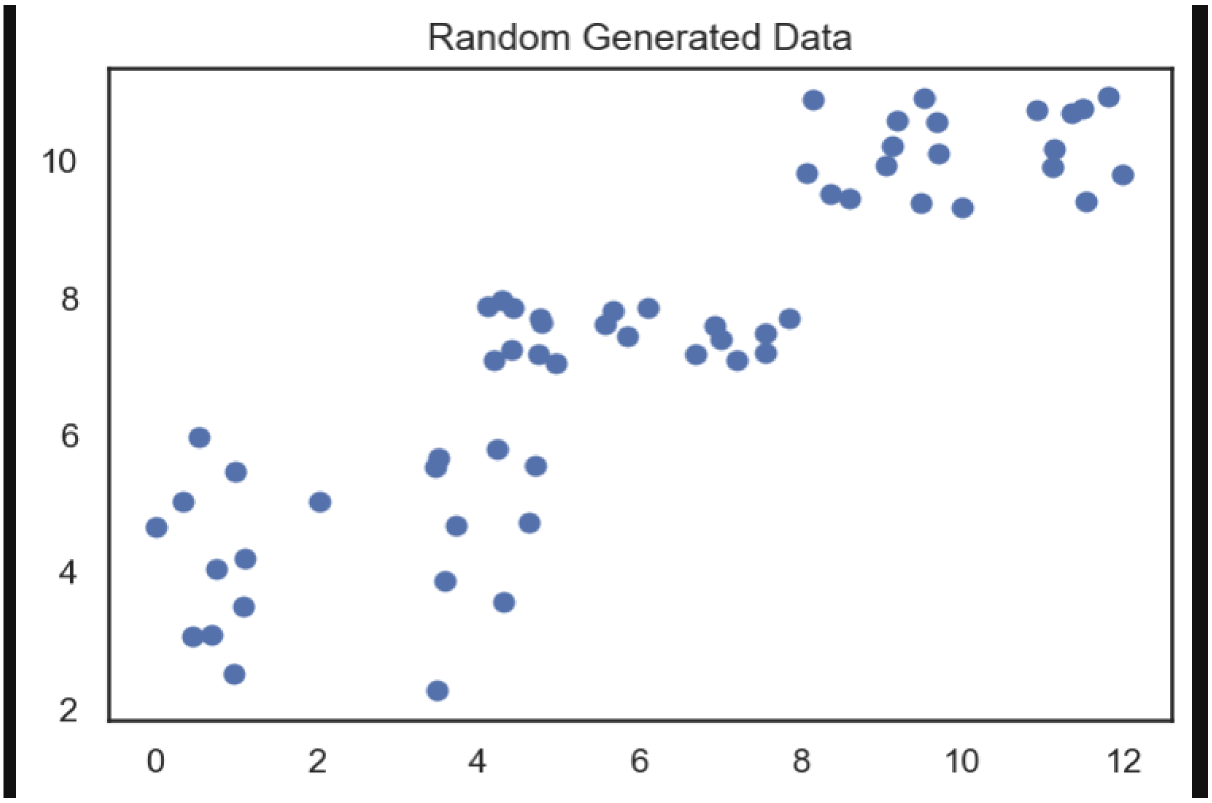
Note that the fitness function computes fitness using the inverse of the sum of distances. The reason for this is that directly employing the sum of distances turns the fitness function into a minimization function, which violates the evolutionary algorithm.

Project Architecture

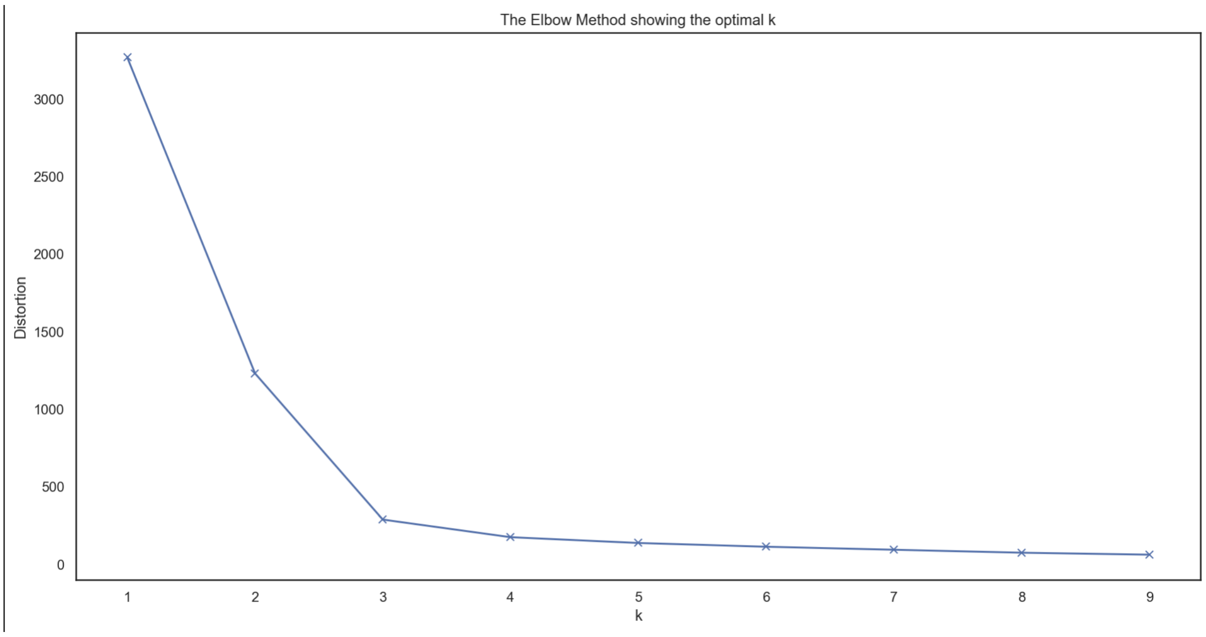


Dataset

The Dataset is randomly genrated using the numpy library to test the algorithm.



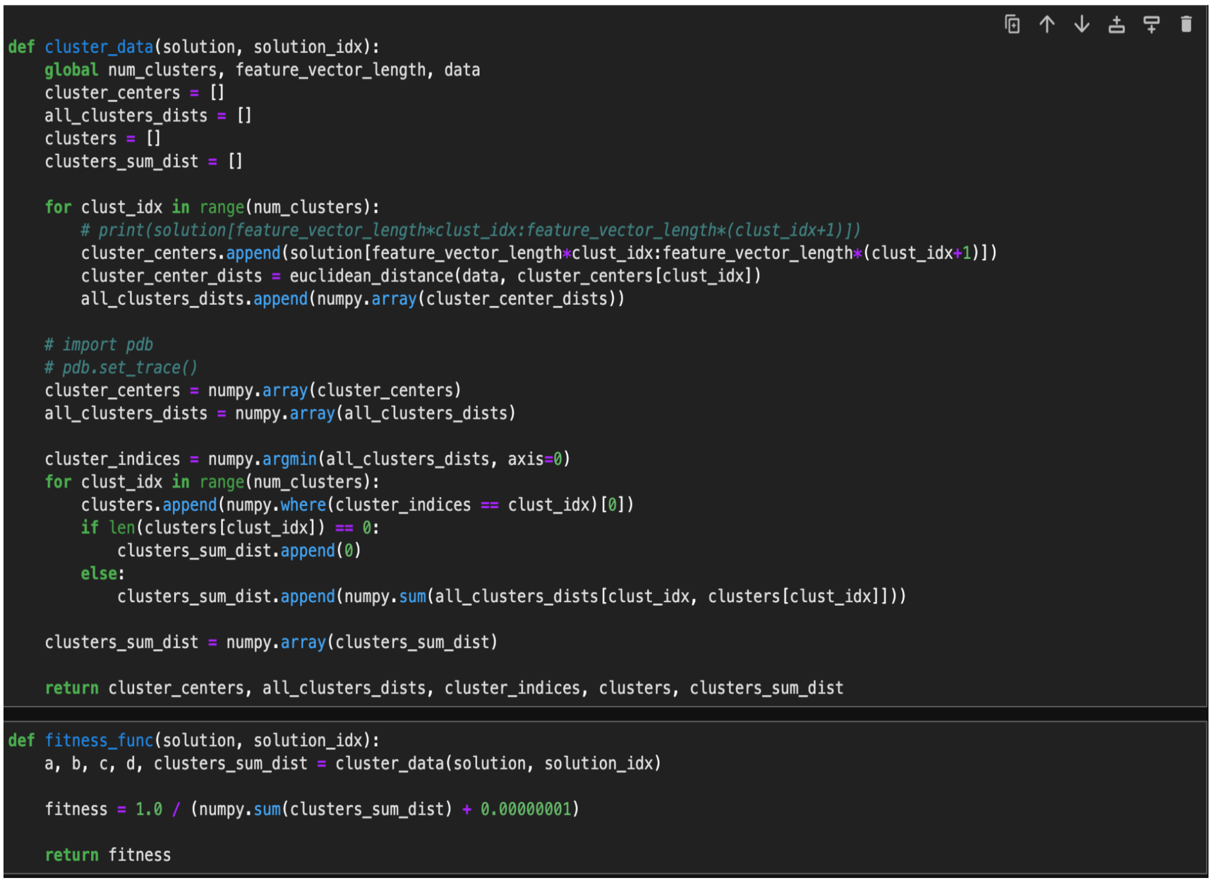
Finding the optimal number of Clusters



Using the Elbow method the optimal number of clusters is tending towards 3

Algorithm Code

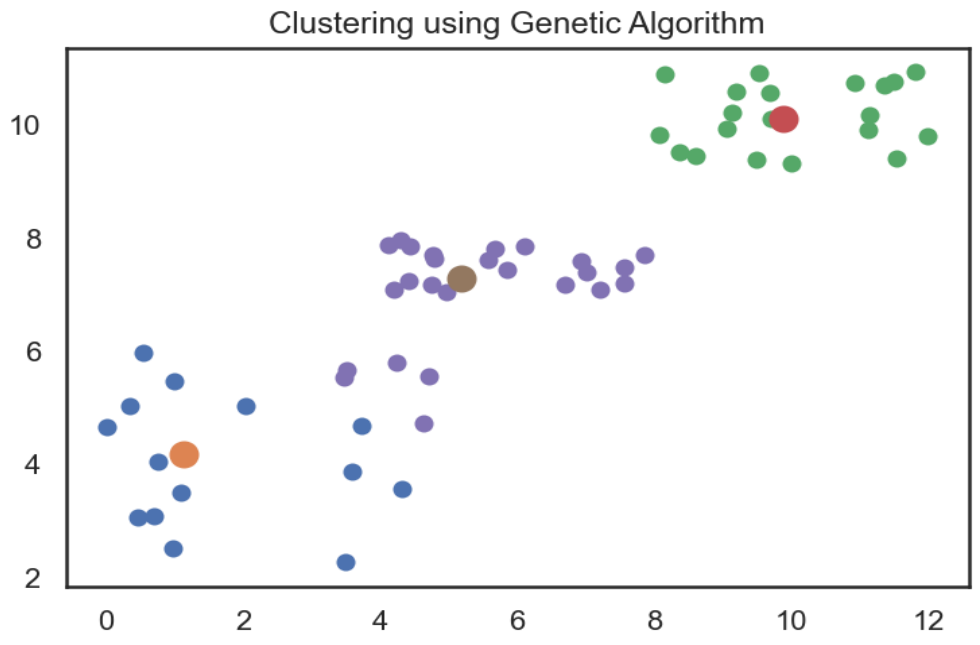
The Algorithm uses pygad library for doing all the operations of genetic algorithm but to calculate the fitness score and assigning the data points to the cluster.



```python
def cluster_data(solution, solution_idx):
    global num_clusters, feature_vector_length, data
    cluster_centers = []
    all_clusters_dists = []
    clusters = []
    clusters_sum_dist = []

    for clust_idx in range(num_clusters):
        # print(solution[feature_vector_length*clust_idx:feature_vector_length*(clust_idx+1)])
        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))

    # import pdb
    # pdb.set_trace()
    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
```

```python
def fitness_func(solution, solution_idx):
    a, b, c, d, clusters_sum_dist = cluster_data(solution, solution_idx)

    fitness = 1.0 / (numpy.sum(clusters_sum_dist) + 0.00000001)

    return fitness
```

Algorithm Results



The Algorithm is able to cluster the data points correctly. However, there are few miss assignment of cluster can be seen in the in the bottom left corner but with some tuning the performance could be better.

Conclusion

The Algorithm demonstrate how a evolutionary algorithm could be used as cluster algorithm. The key reason to use the genetic algorithm instead of the well-liked k-means method is because it is less sensitive to the initial cluster centers and can find an optimal solution without stopping at the first one that is found.

Future prospects

There are many future implications of the algorithm because of it's nature of constantly evolving. Comparing the Genetic Algorithm to various clustering algorithm such as DB-Scan, Gaussian etc