**Chapter 5**

**Research about optimization algorithms for feature selection**

Heuristic algorithms involve two major components: -

1. Exploration

2. Exploitation

Exploration: -

* It is also called diversification.
* Here we generate diverse solutions which enables us to explore the search space and thus, it helps us to avoid getting stuck in local optima.
* Thus, it helps us to explore new solutions in different regions of search space and find global optima which may be far way from the current best solution discovered so far.

Exploitation

* It is also called intensification.
* Here we focus on a small region of the search space and exploit the information about a current solution which is better than the other solutions observed in the region.
* Thus, it helps us get the best solution present in the local search space.

Thus, heuristic algorithms have lesser likelihood than traditional approaches of getting stuck in local optima and more often return results that are global optima.

However, the success of heuristic algorithms is achieved by balancing both exploration and exploitation.

If an algorithm performs more of exploitation than exploration, we may quickly find the optimal results, but the probability of finding global optima reduces.

If an algorithm performs more of exploration than exploitation, it may become extremely slow to get optimal results, we may hope to get global optima at some point in time.

The algorithms also involve a process known as “Evolution”. In this process, the population in each generation is updated based on best solution observed, thus, enabling us to achieve more fitter individuals (subset of population) as we move from one generation to the other.

Thus, the goal to minimize the error of outcome by iterative trial and error and by using an objective function to make decisions and determine which is the best solution.

Heuristic search aims to give good solution each time it is used, however, it does not guarantee of finding the correct solution.

Thus, if we run the same algorithm on the same dataset multiple times, we may more often get different set of results, each of them having best fitness among other solutions when the search was carried out.

Heuristic algorithms are inspired from nature because the elements in nature continuously evolve based on the conditions and environment around them, enabling to tackle complex problems and also fulfilling the criteria of “Survival of the fittest” as the process continues. Moreover, since nature has evolved over many generations, the strength of population of elements gets better and closer to the needs of current situations. In nature, constraints and conditions are many and more often non-linear, thus, the approaches to handle them become simpler but lengthy.

In Data Science and Machine Learning, we often find problems which have large search space, and constraints are non-linear and complex. As the result, taking inspiration from the processes used by nature and adapting them in our solutions helps us to leverage the power of naturally occurring processes and solve problems having large search spaces, multiple constraints and non-linear relationships.

There are more than 40 nature inspired algorithms. Some of them are: -

1. Particle Swarm Optimization

2. Artificial Bee Colony Optimization

3. Artificial Immune System

4. Ant Colony Optimization

5. Cat Swarm Optimization

6. Crow Search Optimization

7. Elephant Intelligence Behavior

8. Grasshopper Optimization

9. Water Wave Optimization

10. Brain Storm Optimization

11. Whale Optimization

12. Grey Wolves Optimization

13. Insects – Firefly Optimization

14. Salp Swarm Optimization

15. Flower Pollination Algorithm

16. Bat Algorithm

Following algorithms were studied in depth in order to understand how they work and to solve the problem of feature selection: -

1. Artificial Bee Colony Optimization

2. Flower Pollination Algorithm

**5.1 Artificial Bee Colony optimization: -**

* It has three phases: -
  1. Employed bee phase
  2. Onlooker bee phase
  3. Scout bee phase
* Parameters of the algorithm: -
  1. Trial: Vector that keeps track of total number of failures irrespective of employed bee phase and onlooker bee phase.
  2. Limit: It sets the threshold up to which a given solution can fail beyond which the solution may participate in Scout bee phase.
  3. maxGenerations: Number of generations (or iterations) for which the process is carried out.
  4. Np: Number of food sources, which indicates number of feature subsets that can participate in a population.
  5. P: Random partner selected
  6. r: Random number
  7. f: Objective function
  8. fit: Fitness function
* Employed bee phase: -
  1. Each employed bee is tagged to a food source.
  2. Food source means a possible solution (In our case a subset of features).
  3. From that food source its fitness value was initially computed.
  4. Now, the employed bee generates a partner solution in the neighborhood of that food source and compute its fitness value.
  5. If the fitness value of the new food source is better than the current food source, then the new food source is added to the population and the original food source is removed from the population.
  6. Thus, in Employed bee phase, Greedy selection is performed to determine the best solution.
  7. And every bee tagged with a food source performs this process.
  8. If the new food source is inferior to the current food source, then we increment the trial counter of current food source by 1.
  9. If the new food source is superior than the current food source, then the trial counter of the new food source is 0.
* Onlooker bee phase: -
  1. Prior to this process, we compute probability of each food source using the equation: -  
     prob(i)=0.9\*(fit(i)/max(fit))+0.1  
     where i is a given food source  
      prob(i) is the probability of ith food source (or solution in the population)  
      fit(i): Fitness of the ith food source  
      max(fit): Maximum fitness among all the food sources
  2. A food source with higher fitness value will have higher probability.
  3. For each food source, a new random number: r is generated between 0 and 1.
  4. If the probability of a given food source is greater than r, then the food source is added to onlooker bee phase.
  5. If the probability of a given food source is smaller than r, then the food source is not added to onlooker bee phase.
  6. It may be possible that a given food source may not participate in the onlooker bee phase in a given generation. But it may participate in the onlooker bee phase in the next generation. This will depend on the random number generated for that food source in each generation.
  7. For the food sources that participate in onlooker bee phase, they perform steps similar to employed bee phase.
* Scout bee phase: -
  1. Solutions which have trial greater than limit are the candidates to be discarded. Thus, if the trial value is greater than limit the solution can potentially enter the scout phase.
  2. The trial counter of the abandoned solution is reset to 0.
  3. Out of all solutions in the population which have trial counter greater than the limit, only one among them can participate in the Scout bee phase.
  4. However, it is possible that we end up eliminating the best solution from the population due to the limit.
  5. Thus, prior performing Scout bee phase, we need to memorize the best solution in the population.
  6. Scenarios in Scout bee phase: -  
     + Case 1: In a given population, all the solutions have trial lesser than the limit. Thus, none of the solutions will participate in scout bee phase.
     + Case 2: In a given population, multiple solutions have trial greater than the limit. Thus, the solution with highest trial value is selected for scout bee phase.
     + Case 3: In a given solution, multiple solutions have trial greater than the limit, and have same value. Thus, randomly one of them are selected for scout bee phase.

* Fitness is related to objective function using below relation: - fit=1/1+f, if f >= 0

fit=1+|f|, if f< 0

Thus: -

If objective value>=0, then fitness function value = 1/1+f

If objective value<0, then fitness function value = 1+|f|

Thus, as objective function value increases, fitness function value decreases.

Flowchart of Artificial Bee Colony Optimization

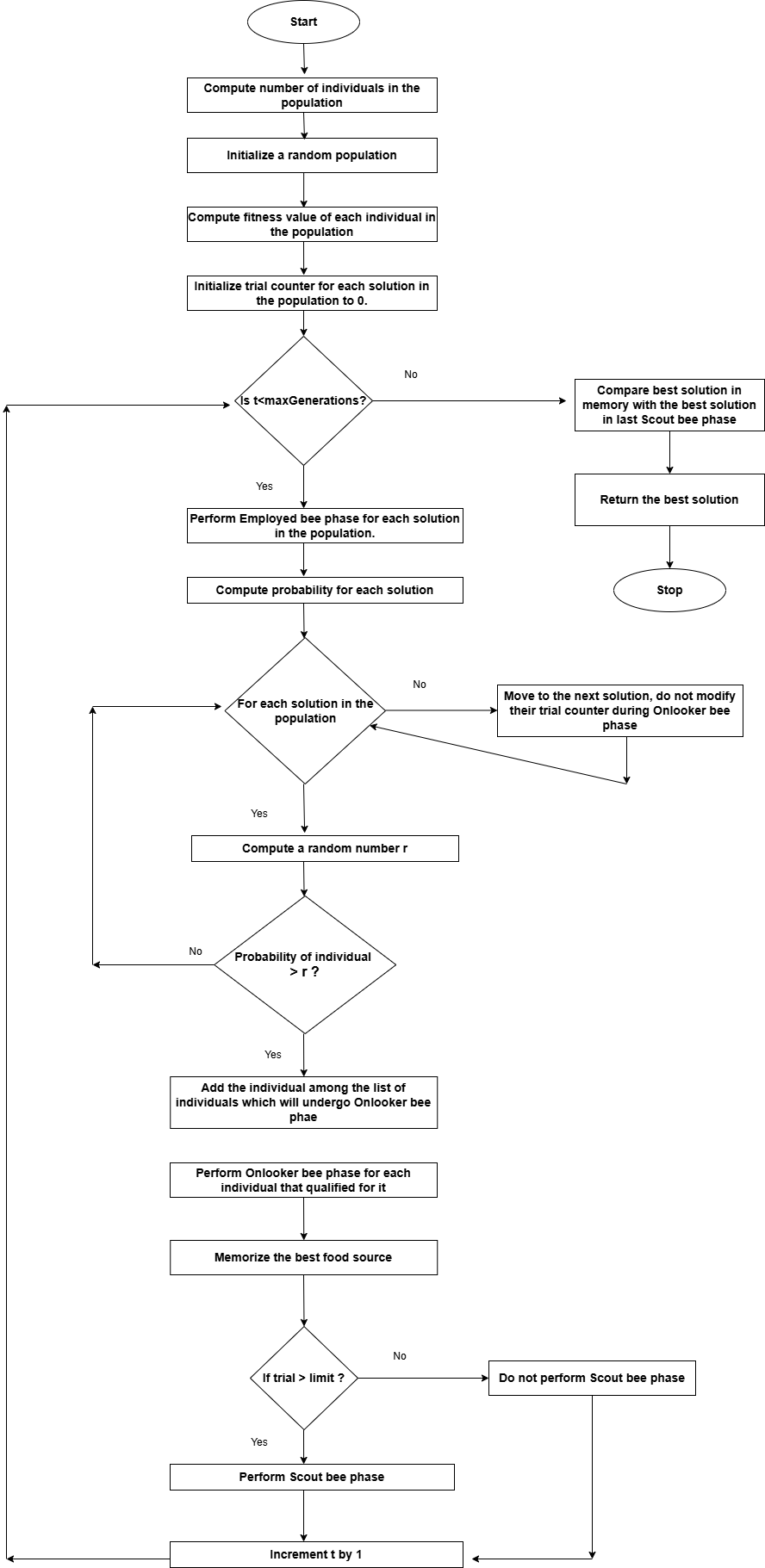


Figure 5.1.1 Flow chart of Artificial Bee Colony Optimization

**5.2 Flower Pollination Algorithm: -**

* It has two main components: -
  1. Local pollination
  2. Global pollination
* Parameters of the algorithm: -
  1. maxGenerations: Number of generations (or iterations) for which the process is carried out.
  2. p: Switch probability which helps to decide whether we perform global pollination or local pollination.
  3. λ: It helps to control the step size in Levy flight.
  4. γ: Pulse emission rate which helps to determine the intensity of global pollination.
  5. L(): Levy distribution computed using Levy flight.
  6. g\*: Current best solution
  7. ε: Normal distribution
* Local pollination: -
  1. We create an 1D array: ε whose dimension is equal to number of features.
  2. The array is made using normal distribution of points in the range of 0 and 1.
  3. Two unique solutions from the population are randomly selected: xj and xk.
  4. A new solution is formed by performing local pollination using the below equation: -  
     new\_solution=current\_solution+ε \* (xj-xk)
* Global pollination: -
  1. We create an 1D array: L whose dimension is equal to number of features.
  2. The array is made using Levy distribution.
  3. A new solution is formed be performing global pollination using the below equation: -  
     new\_solution=current\_solution+ γ\*L(λ)\*(g\*-current\_solution)
* Switch probability - p: -
  1. If value of switch probability p is high, then we perform a greater number of global pollinations and lesser local pollinations.
  2. If the value of switch probability p is small, then we perform a greater number of local pollinations and lesser global pollinations.
  3. Thus, switch probability is a hyper parameter which needs to be set based on the requirements.

Flowchart of Flower Pollination Algorithm

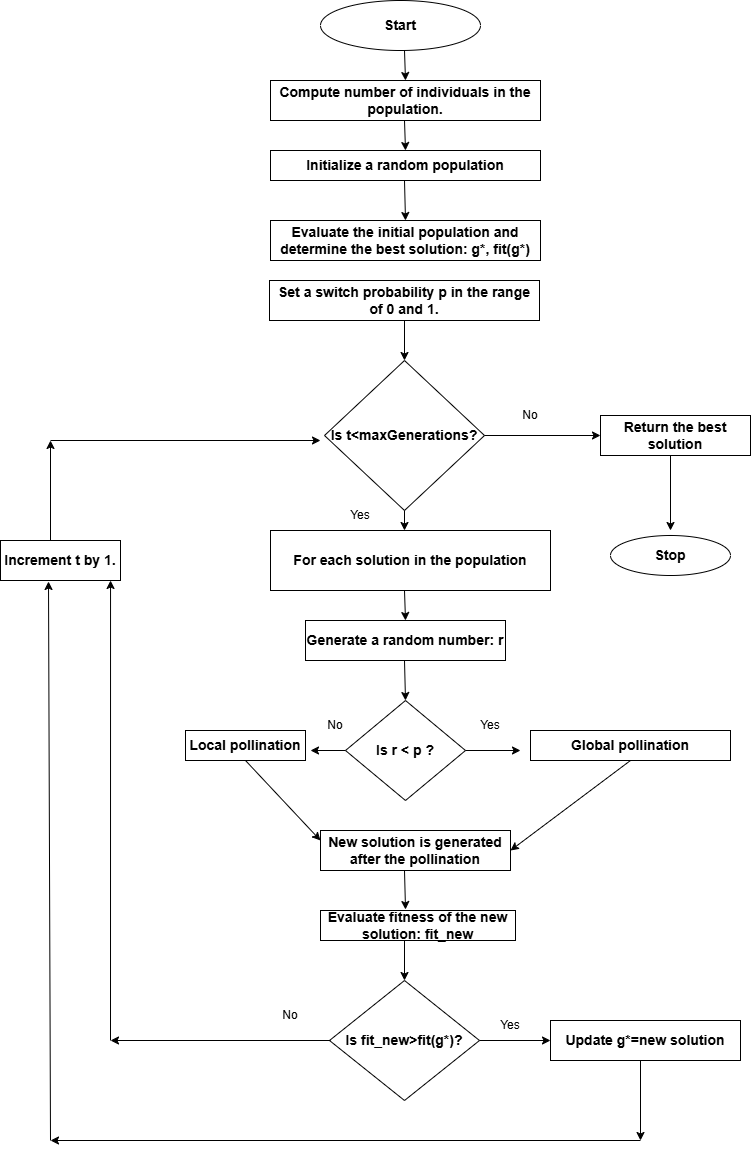


Figure 5.2.1 Flow chart of Flower Pollination Algorithm