Evolutionary Algorithms

Evolutionary algorithms are based on concepts of biological evolution. A 'population' of possible solutions to the problem is first created with each solution being scored using a 'fitness function' that indicates how good they are. The population evolves over time and (hopefully) identifies better solutions. Of the various types of evolutionary algorithm, the genetic algorithm is the most well-known.

Genetic Algorithm: As machine learning is used for making better and faster decisions, one can use the artificial intelligence method called 'Genetic Algorithm' for optimizing the results further by letting algorithm evolve as it goes through iterations. Basic premise is that, one starts with set of possible solutions (randomly generated and mostly unfit for the problem) and rank them on fitness score for the given problem. Then, the two top solutions (often referred as parent) are picked that have high fitness score and then create another solution (referred as child) which takes best of features from both parents. This is done using cross over and mutation operations. The crossover operation involves swapping random parts of selected pairs (parents) to produce new and different offspring that become part of the new generation of solutions. Mutation involves substitution of some random part of a solution with some other random part of a solution. Finally, the child solution is added to the stack of solution and again, the solutions are ranked for fitness. This process is done again and again via many iterations till a solution is reached which is far superior compared to the starting ones. Multiple runs (dozens to hundreds) are usually necessary to produce a very good result. There are various application of this technique such as evolving website design, designing games, automated trading systems and so on.

8 Queens Problem with Genetic Algorithm: There are many applications where Genetic Algorithm can be used especially where one does not know the exact solution and how to find it. One such problem is '8 Queens' puzzle. The objective here is to place 8 queens on a Chess board of dimension 8x8 such that they don't threaten each other i.e. no 2 queens share the same row, column and diagonal. There are only 92 solutions which meet the criteria out of 16,777,216 possible combinations.

i. **Initial Population**: First objective is to generate random 15 combinations which can be used as seed: say as an example, below is example combination where the column position indicates the column in which the Queen is present and the number indicates the row.

7	0	3	6	2	5	1	6

If one was to visualize this particular combination on chess board, it would look like below.

	X						
						X	
				X			
		X					
					X		
			X				X
X							

ii. **Fitness Score**: As it can be observed above there are few queens in positions where they can attack each other; so a function can be developed that returns the fitness score of the combination. The fitness score adds up any possibilities that are there for the queens to attack. So, ideal fitness score is 0 and any higher number is not good. Once all initial 15 combinations are scored, they would be ranked in the ascending order of the fitness scores.

	Combination	Fitness Score
0	[6, 3, 7, 0, 5, 3, 1, 4]	4
1	[3, 6, 0, 2, 7, 7, 1, 1]	6
2	[1, 5, 7, 2, 6, 1, 3, 5]	6
3	[6, 3, 6, 5, 2, 4, 6, 7]	7
4	[2, 1, 3, 3, 7, 0, 3, 3]	7
5	[1, 0, 1, 0, 7, 4, 2, 6]	7
6	[1, 7, 1, 1, 6, 4, 4, 0]	8
7	[0, 7, 3, 6, 0, 6, 2, 6]	8
8	[0, 3, 5, 1, 4, 3, 2, 2]	8
9	[0, 7, 2, 3, 0, 1, 2, 5]	9
10	[5, 1, 1, 3, 3, 4, 5, 3]	9
11	[6, 5, 0, 5, 3, 3, 0, 5]	10
12	[0, 2, 2, 3, 5, 1, 0, 1]	10
13	[4, 2, 7, 3, 4, 4, 7, 3]	10
14	[2, 3, 6, 3, 0, 5, 2, 5]	11

iii. **Crossover and Mutation**: Next, the 2 combinations with least fitness scores are picked and Crossover and Mutation techniques are applied. Let's start with the combinations and name them as Parent 1 and 2.

Parent 1 [**6, 3, 7, 0**, 5, 3, 1, 4] with score 4
Parent 2 [3, 6, 0, 2, **7, 7, 1, 1**] with score 6

While creating a cross over solution, given that there are 8 values, cross over point would be taken as 4

Cross over child 1 [**6, 3, 7, 0, 7, 7, 1, 1**] by combining first half of Parent 1 and second half of Parent 2
Cross over child 2 [**3, 6, 0, 2, 5, 3, 1, 4**] by combining second half of Parent 1 and first half of Parent 2
Then, Mutation is applied by randomly changing one of the columns with a random value between 0-7
Crossover child 1 after Mutation [6, 3, 7, 0, 4, 7, 1, 1] with score of 5 -Not better than one of parents
Crossover child 2 after Mutation [3, 7, 0, 2, 5, 3, 1, 4] with score of 4 -Better than one of parents

Now, these next generation solutions are added back to the original set of combinations and sort that dataset again on Fitness Score ascending. Then, again the process begins from Step 3 again and reiterate in loops. Combinationsare sorted on Fitness score (ascending) before each iteration to ensure thatbest ones would be used to create next generation solution.

Mutation Techniques: There are several mutation techniques for the Genetic Algorithm. Some of them are mentioned below.

- **Scramble Mutation:** Scramble mutation is popular with permutation representations. In this, from a candidate solution, a subset of values is chosen and they are scrambled or shuffled randomly.

Before							
0	1	**2**	**3**	**4**	**5**	**6**	7

After							
0	1	**3**	**6**	**4**	**2**	**5**	7

- **Inversion Mutation:** In inversion mutation, a subset of values is selected and they are merely inverted.

Before							
0	1	**2**	**3**	**4**	**5**	**6**	7

After							
0	1	**6**	**5**	**4**	**3**	**2**	7

- **Swap Mutation**: In swap mutation, two positions are selected at random, and the values are interchanged.

Before							
0	1	**2**	3	4	5	**6**	7

After							
0	1	**6**	3	4	5	**2**	7