Genetic Algorithms

GAs are based on the ideas of natural selection and genetics.

They are "intelligent exploitation" of "random searches"

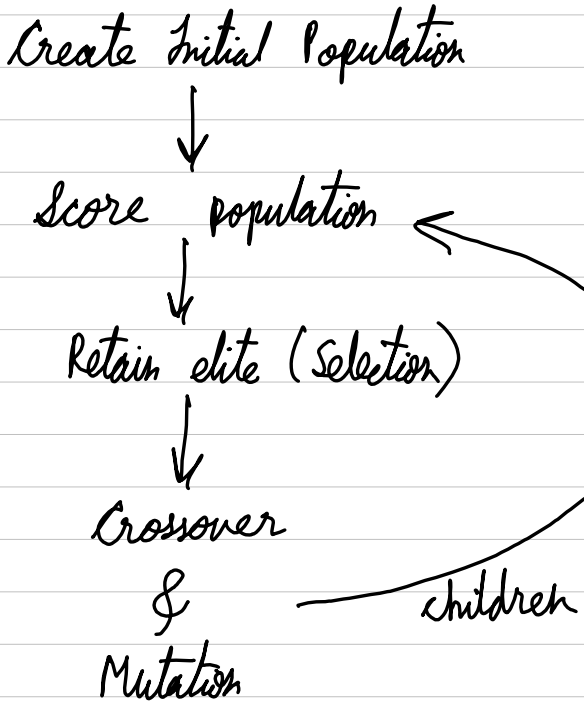
Natural selection → species that adapt to changes in their environment can survive and reproduce. Their genes are transmitted to the next generation

"Survival of the fittest"

GAs are useful when the solution space is very large.

They are versatile algorithms that can be applied in various domains like optimization, machine learning (feature selection, hyperparameter tuning)

They can handle complex multimodal and non-linear optimization problems.



In GA, we have a pool of possible solutions ie genes. They may not be perfect These go recombination and mutation to produce new children. The children who are fit live, rest die. This is determined by a fitness function. Then the entire process is repeated over and over again for several generations.

The genes evolve over generations and become better

Each new generation has on average better genes and better "partial solutions" than previous generation

Key Ideas

Initial Population: Randomly selected states

Fitness function: function to decide if a gene is fit for existance or not

Crossover – Reproduction of genes

Mutation – change in the gene structure (small tweak in the chromosome)

Selection – Choosing best genes & killing unfit ones

GAs have 5 steps

1. Initialization
2. Decide fitness function
3. Selection
4. Crossover
5. Mutation

} Reproduction

① Solve 0/1 knapsack for 1 generation

$(w, b) = \{ (7, 42), (3, 12), (4, 40), (5, 25) \}$ $\quad M = 10$

→ Let 0 denote Not taken
1 denote Taken

So 0 0 0 1 → chromozome means we take last item

step 1 → Initialize random population

A	B	C	D	Weight	Profit
0	0	1	0	4	40
0	1	0	1	8	37
1	1	0	0	10	54
⋮	⋮	⋮	⋮	⋮	⋮

step 2 → fitness creiterion

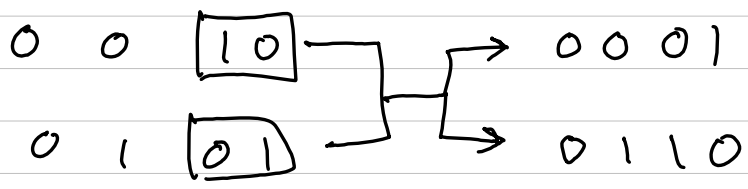
Profit ≥ 25
Weight ≤ 10

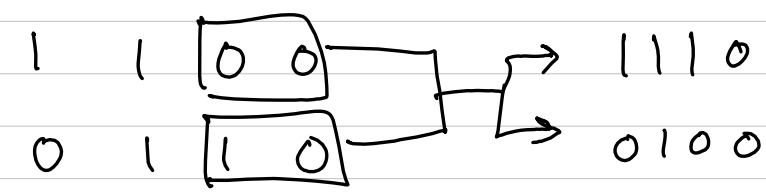
step 3 → select genes with $P \geq 25$

0 0 1 0 ✓
0 1 0 1 ✓
1 1 0 0 ✓
⋮ ⋮ ⋮ ✗

step 4 → Reproduce with crossover

Consider crossover point 2, 2





step 5 – Mutate → flip one bit at random

0 0 0 1 → 0 1 0 1 or 0 0 1 1 or 0 0 0 0

0 1 1 0 → 1 1 1 0 or 0 1 0 0 . . .

Generation over, start with next generation.

Advantages → ① Do not need derivative information

② Robust

③ Handle Noise

④ Provides a list of good solutions and not just one solution

Disadvantages → ① Time complexity depends on fitness value

② No gurantee of optimal solution

Applications →
① Function optimization
② Feature selection
③ Hyperparameter tuning
④ Game playing
⑤ Robotics

GAs are useful when search space is large and fitness function is not computationally expensive.