13/5/22

III Bayesian Learning

Bayesian learning Provides a probabilistic approach to inference. derive
It is based on the assumption that the quantities of interest are goverened by Probability distributions.
It is important machine learning because it provides a quantitative approach to weighing the evidence supporting alternative hypothesis

Features of Machine learning Methods.

1) Each observed training example can incrementally decrease or increase the estimated probability that a hypotheses is correct
2) Prior knowledge can be combined with the observed data to determine the final Probability of a hypotheses
3) Bayesian methods can accomadate hypotheses that main Probabilistic Predictions.
4) New instances can be classified by combining the predictions of multiple hypotheses, weighted by a probability.

Bayes Theorem :-

In machine learning we are often intrested in determing the best hypotheses from some space 'H', given in the observed training data 'D'.

Notations :-

P(h) → initial probability that hypothesis 'h' holds
(Prior Probability of h)

P(D) → The Prior probability that training data 'D'

will be observed

$P(D/h)$ → Probability of observing data 'D' given some word in which hypothesis 'h' holds.

$P(x/y)$ → Probability of x given y

In machine learning we are intrested in probability $P(h/D)$ that 'h' holds given observed training data 'D', is called "Posterior Probability" of 'h'

$$\therefore \text{Bayes Theorem: } P(h/D) = \frac{P(D/h)\,P(h)}{P(D)}$$

In many learning scenarios the learner intrested in finding the most Probable hypothesis. Any search maximally probable hypothesis is called a "Maximum a Posteriori (MAP) hypothesis".

$$\therefore h_{MAP} \equiv \underset{h \in H}{\text{argmax}}\ P(h/D)$$

$$= \underset{h \in D}{\text{argmax}}\ \frac{P(D/h)P(h)}{P(D)}$$

$$= \underset{h \in H}{\text{argmax}}\ P(D/h)P(h)$$

In some Cases we assume that every hypothesis in 'H' is equally Probable aprori

$$(P(h_i) = P(h_j))$$

∴ Maximum likelihood (ML) hypothesis, h_{ML}.

$$h_{ML} \equiv \underset{h \in H}{\text{argmax}}\ P(D/h)$$

Probability Formulaes

Product Rule: Probability $P(A \wedge B)$ of a conjunction of two events A & B is $P(A \wedge B) = P(A/B)P(B) = P(B/A)P(A)$

Sum Rule: Probability of a disjunction of two events A & B is

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

Bayes Theorem: The Posterior probability $P(h/D)$ of 'h' given 'D' is

$$P(h/D) = \frac{P(D/h) \cdot P(h)}{P(D)}$$

Theorem of Total probability: If events $A_1, A_2, \ldots A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^{n} P(B/A_i) P(A_i)$$

Bayes Theorem & Concept Learning

Bayes theorem provides a principle way to calculate for Posteriori probability of each hypothesis given the training data, we can use it as the basis for a learning algorithm that calculates the probability for each possible hypothesis. then outputs the most probable hypothesis.

i) Brute-Force MAP Learning Algorithm:-

step 1:- For each hypothesis h in H, calculate the posterior probability

$$P(h/D) = \frac{P(D/h) \cdot P(h)}{P(D)}$$

step 2:- Output the hypothesis h_{MAP} with highest posterior probability

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} \; P(h/D)$$

In order to specify a learning problem for the brute-force MAP learning algo we must specify what values are to be used for $P(h)$ & for $P(D/h)$. It is reasonable to assign the same prior probability to every hypothesis 'h' in H when there is no prior knowledge, Sum to one.

$\therefore P(h) = \frac{1}{|H|}$ for all h in H

Sum to One

$$P(D/h) = \begin{cases} 1 \text{ if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 \text{ if } d_i \neq h(x_i) \end{cases}$$

In other words the probability of data D given hypothesis 'h' is 1 if D is consistent with 'h', & 0 otherwise

Case1:- h is inconsistent

$$P(h/D) = \frac{0 \cdot P(h)}{P(D)} = 0, \text{ if h is inconsistent with D}$$

Case2:- h is consistent

$$P(h/D) = \frac{1 \cdot 1/|H|}{P(D)} = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|} \text{ if h is consistent with D.}$$

$VS_{H,D}$ — subsets of hypothesis from H that are consistent with D.

(Version space of H wrt D)

$$\therefore P(h/D) = \begin{cases} \frac{1}{|VS_{H,D}|}, \text{ if h is consistent with D} \\ 0, \text{ otherwise} \end{cases}$$

$|VS_{H,D}| \rightarrow$ no of hypotheses from H consistent with D.

$\therefore$ under our choice for $P(h)$ and $P(D/h)$, every consistent hypothesis has Posterior Probability $\left[\frac{1}{|VS_{H,D}|}\right]$, and every inconsistent

hypothesis has posterior probability 0.

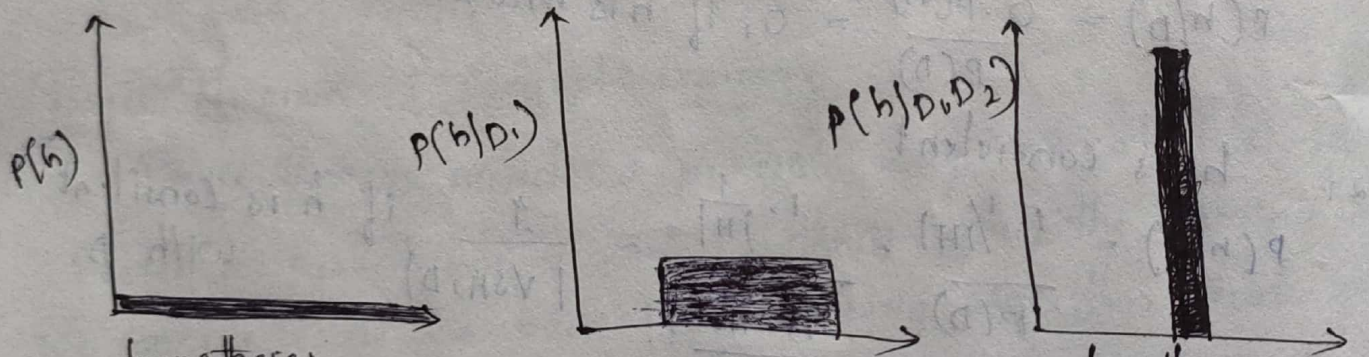
→ every consistent hypothesis is therefore a MAP hypothesis

(ii) MAP hypotheses & consistent learners :-

A learning algo is a "consistent learner" provided it outputs a hypothesis that contains zero errors over the training examples

Every consistent learner o/ps a MAP hypothesis if we assume a uniform real probability distribution over H $(P(h_i) = P(h_j)) \forall i, j$

→ Concept learning algo find-S searches the hypothesis space H from specific to general hypothesis, outputing a maximally specific consistent hypothesis.



~~equal probability to each hypothesis~~ (a)

~~increas~~ (b)

(c)

evolution of Posterior Probabilities $P(h|D)$ with increase in training data.

a. Uniform Priors assign equal probability to each hypothesis.

b. as training data increases first to D_1

b. Then to $D_1 \wedge D_2$

c. The posterior Prob of inconsistent hypothesis becomes zero while Posterior Prob increase for hypotheses remaining in the version space.

Eg:- Consider a medical diagnosis pblm in which there are 2 alternative hypotheses:

i. That the patient has a particular form of Cancer

i- That the patient does not

* Two Possible outcomes :- ⊕ Positive
  ⊖ Negative

* Prior knowledge that over the entire population of people, only 0.008 have this disease

* Correct posible result → 98%. } Test
  Correct negative result → 97%.

In other Case the test returns the opposite result

$\therefore P(\text{cancer}) = 0.008 \quad P(\neg \text{Cancer}) = 0.992$

$P(\oplus/\text{cancer}) = 0.98 \;,\; P(\ominus/\text{cancer}) = 0.02$

$P(\oplus/\neg\text{cancer}) = 0.03, \quad P(\ominus/\neg\text{cancer}) = 0.97$

Suppose we have obseirve a new patient for whom the lab test return a positive result. should we diagonsis the patient as having cancer or not?

The maximum a posterior (MAP) hypothesis can be found using the following equation.

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\; P(D/h)P(h)$$

$P(\oplus/\text{cancer})\,P(\text{cancer}) = (0.98)(0.008) = 0.0078$

$P(\oplus/\neg\text{cancer})\,P(\neg\text{cancer}) = (0.03)(0.992) = 0.0298$

Thus $h_{MAP} = \neg$ cancer

Minimum Description Length (MDL) Principle

The minimum Description length MDL principle is motivated by interpreting the definition of h-MAP.

$$h_{MAP} \equiv \underset{h \in H}{\operatorname{argmax}} P(D/h) P(h)$$

which can be equivalently expressed in terms of maximizing the $\log_2$.

$$h_{MAP} = \underset{h \in H}{\arg\max} \log_2 P(D/h) + \log_2 P(h)$$

or alternative, minimizing the negative of this quantity

$$h_{MAP} = \underset{h \in H}{\arg\min} - \log_2 P(D/h) - \log_2 P(h)$$

It can be interpreted as a statement that short hypotheses are preferred

We can rewrite the above eqn as

$$h_{MAP} = \underset{h}{\operatorname{argmin}} L_{C_H} + L_{C_{D/h}}(D/h)$$

C_H and $C_{D/h}$ are optimal encoding for H and for D given h.

∴ Minimum Description length principles

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}} L_{C_1}(h) + L_{C_2}(D/h)$$

C_1 & C_2 are optimal encodings of hypotheses

Bayes optimal classifier:-

What is the most probable ~~classification~~ hypothesis given the training data?

What is the most probable classification of the new instance given the training data.

Eg:- Posterior probabilities of 3 hypothesis h_1, h_2, h_3 are 0.4, 0.3, 0.3

Thus h_1 is the MAP hypothesis

Suppose a new instance x is encountered which is classified positive by h_1, but negative by h_2 & h_3

Taking all hypotheses into account the probability that x is positive is 0.4, and the probability that x is negative is 0.6 (0.3 + 0.3)

The most probable classification (-ve) with 0.6 in this case is different from classification generated by the MAP hypothesis (0.4)

Bayes optimal classification is

$$\underset{v_j \in V}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i) P(h_i|D)$$

$P(h_1|D) = 0.4$, $P(\ominus|h_1) = 0$, $P(\oplus|h_1) = 1$

$P(h_2|D) = 0.3$, $P(\ominus|h_2) = 1$, $P(\oplus|h_2) = 0$

$P(h_3|D) = 0.3$, $P(\ominus|h_3) = 1$, $P(\oplus|h_3) = 0$

Therefore $\sum_{h_i \in H} P(\oplus|h_i) P(h_i|D) = 0.4$

$\sum_{h_i \in H} P(\ominus|h_i) P(h_i|D) = 0.6$

and $\underset{v_j \in \{\oplus, \ominus\}}{\operatorname{argmax}} \sum P(v_j|h_i) P(h_i|D) = \ominus$

Instance Based Learning

Learning in instance based algo consists of simply storing the resulting training presented data. When a new query instance is encountered a set of similar related instances is retreived from memory & used to classify the new query instance.

K-Nearest Neighbour (KNN) Learning

This algo assumes all instances correspond to points in the n dimensional space $\mathbb{R}^n$

The nearest neighbours of an instance are defined in terms of standard euclidian distance.

Let an arbitary instance x be described by the feature vector

$$\langle a_1(x), a_2(x), \ldots a_n(x) \rangle$$

where $a_r(x) \rightarrow$ value of r^{th} attribute of instance x.

Then the distance b/w the two instances x_i and x_j is defined to be $d(x_i, x_j)$

where,

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

Algorithm :-

Training algorithm :-

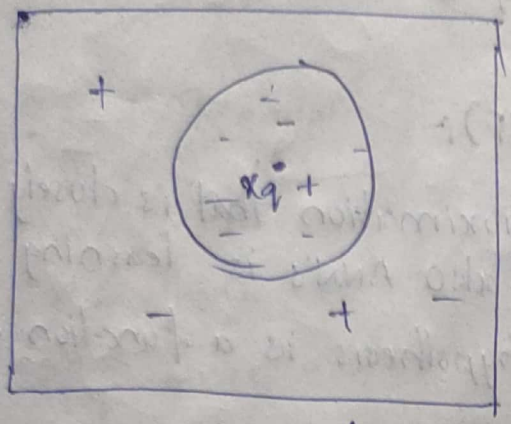
For each training example $\langle x, f(x) \rangle$ add the example to the list training examples

Classification Algorithm :-

- Given a query instance x_q to be classified,
  - let $x_1 \ldots x_k$ denote the k instances from training examples that are nearest to x_q.
  - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a,b) = 1$ if $a = b$ and where $\delta(a,b) = 0$, otherwise



A set of +ve and -ve training examples is shown on the graph along with new query instance x_q to be classified.

~~One~~ 1- nearest neighbour algo classifies x_q as +ve, whereas 5-nearest neighbour classifies x_q as -ve.

Locally Weighted Regulation Regression :-

The nearest neighbour approach is approximating the target function $f(x)$ at the single query point $x_q = x$.

where as locally weighted regression constructs an explicit approximation to $f(x)$ over a local region surrounding x_q.

Locally weighted regression uses near by or distance weighted training examples to form this local approximation to the target function $f(x)$.

let us consider the case of locally weighted Regression in which the target func^n f is approximated near x_q using the linear function of the form

$$\hat{f}(x) = w_0 + w_1 a_1(x) + \cdots + w_n a_n(x)$$

where, $a_i(x) \rightarrow$ The value of ith attribute of the instance x

we derived methods to choose weights that minimize the squared error

Summed over, the set D of training examples.

$$E \equiv \frac{1}{2} \sum_{x \in D} (f(x) - \hat{f}(x))^2$$

which let us to gradient descent training rule

$$\Delta w_j = \eta \sum_{x \in D} (f(x) - \hat{f}(x)) \, a_j(x)$$

** Radial Basis Functions (RBF):-

One approach to function approximation that is closely related to distance weighted regression & also ANN's is learning with RBF's.

→ In this approach the learned hypothesis is a function of the form

$$\hat{f}(x) = w_0 + \sum_{u=1}^{k} w_u k_u(d(x_u, x)) \quad - (1)$$

x_u → instance from X

Kernal function, $k_u(d(x_u, x))$ is decreases, as the distance $d(x_u, x)$ increases.

Even though $\hat{f}(x)$ is a global approximation to $f(x)$, the contribution from each of the $k_u(d(x_u, x))$ term is localized to a region nearby the point x_u.

It is common to choose each function $k_u(d(x_u, x))$ to be Gaussian function, centered at the point x_u with some variance, σ_u^2

$$\therefore \quad k_u(d(x_u, x)) = e^{\frac{1}{2\sigma_u^2} d^2(x_u, x)}$$

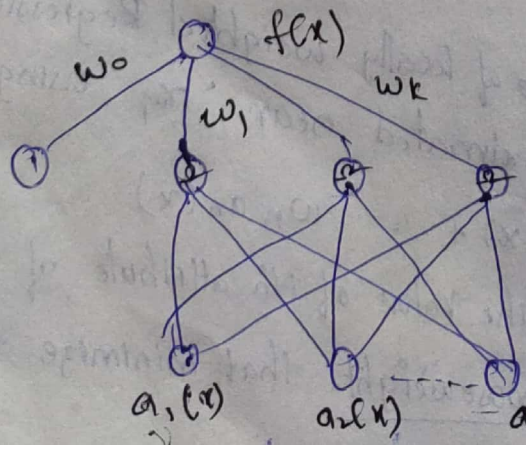


fig: A Radial Basis Functions (RBF) Network.

Each hidden node produces an activation determined by a Gaussian func centered at some instance x_u. Therefore its activation will be closed to zero unless the input x is near to x_u.

→ The o/p unit produces a linear combination of the hidden unit activations.

Case Based Reasoning (CBR):-

In CBR instances are typically represented using rich symbolic descriptions & methods used to retrieve similar instances or correspondingly more alloborate.

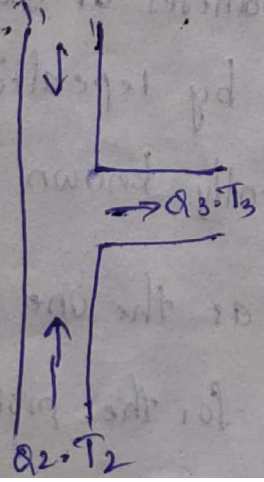
→ CBR has been applied to pblms such as conceptual design of mechanical devices based on a stored library of previous designs, reasoning about new legal cases based on previous rulings & solving planning and scheduling problems by reusing and combining portions of prev solns to similar problems

Eg:- CADET System

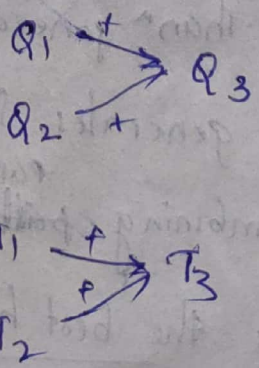
The CADET system employees a case based reasoning to assist in the conceptual design of simple mechanical devices such as water faucets. It uses a library containing approximate designs to ~~read~~ meet the specifications of new design problems. To previous designs & design fragments to suggest conceptual

A stored Case:

structure



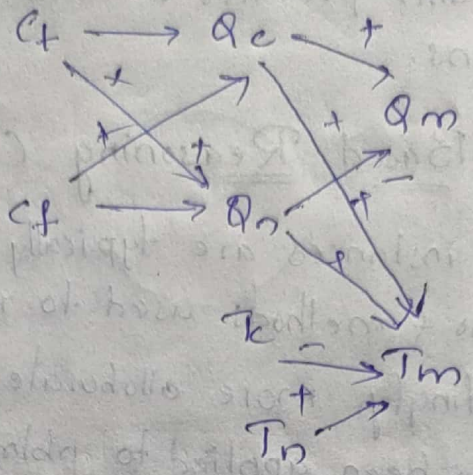
Function:-



A Boolean Specification:- water-faucet

Structure :-

Function:



Genetic Algorithms :-

Genetic algorithms provide an approach to learning that is based loosely on simulated evaluation.

The search for an appropriate hypothesis begins with a population, or collection of initial hypotheses.

members of the current population give rise to the next generation population by means of operations such as random 'mutations' and 'crossover' which are patterned after processes in biological evaluations

→ At each step, the hypotheses in current population or evaluated related to a given measure of 'fitness' with the most fit hypotheses selected probabilistically as 'seeds' for producing the next generation.

Rather than search from general to specific hypotheses or from simple to complex, GA's generates successor hypotheses by repeatedly mutating and recombining parts of the best currently known hypothesis

In GAs the best hypothesis is depend as the one that optimizes a predefined numerical measure for the problem at hand

called hypothesis 'fitness'

Algorithm:-

GA (fitness, fitness_threshold, p, r, m)

fitness : A function that assigns an evaluation score, given a hypothesis.

fitness_threshold : A threshold specifying an termination criterian.

p : The number of hypotheses to be included in the population

r : The fraction of the population to be replaced by crossover at each step

m : mutation rate

- Initialise population : P ← Generate p hypotheses at random.
- Evaluate : for each h in P, compute Fitness(h)
- while [$\max_h$ Fitness(h)] < fitness_threshold do.

Create a new generations, Ps:

1. Select: Probabalistically select (1-r)p members of p to add Ps.

The probabilrty Pr(hi) of selecting hypothesis hi from P is given by

$$Pr(h_i) = \frac{fitness(h_i)}{\sum_{j=1}^{p} fitness(h_j)}$$

2. Crossover: Probabilistically select $\frac{r \cdot p}{2}$ Pairs of hypotheses from p, according to Pr(hi) given above. for each Pair(h_1, h_2), Produce two offspring by applying The crossover operator

Add all offspring to Ps.

3. Mulate :- choose m percent of the members of P_s with Confirm Probability for each invert one randomly selected bit in its representation.

4. Update : $P \leftarrow P_s$

5. Evaluate : For each h in P, compute Fitness (h).

• Return the hypothesis from p that has the highest fitness

$P_s \leftarrow$ Sucessor Population.

Representing hypothesis :-

Hypotheses in GAs are often represented by bit strings, so that they can be easily manipulated by genetic operators such as mutation, and crossover.

→ Sets of ~~in~~ 'If-then' rules can easily be represented in this way by choosing an encoding of rules that allocates specific substrings for each rule precondition & postcondition.

Eg :- Consider the attribute 'outlook' which can take on any of three ~~attributes~~ values : sunny, overcast, Rain.

outlook = {sunny, overcast, Rain}

One way to represent a constraint on outlook is to use a bit string of length three where each bit position corresponds each ~~bit position through~~ one of its three possible values

Outlook = Overcast ⇒ 010

Outlook = Overcast ∨ Rain (more general) ⇒ 011

Outlook = Sunny ∨ Overcast ∨ Rain (most general) ⇒ 111

Consider a second attribute 'wind'

wind = {strong, weak}

For example, a rule pre condition such as

(outlook = Orercast ∨ Rain) ∧ (wind = strong)

outlook	wind
011	10

Rule post condition (such as PlayTennis = Yes) can be represented in a similar fashion.

For example, the rule

IF wind = strong THEN play Tennis = Yes will be represented by the string.

Outlook	wind	PlayTennis
111	10	10

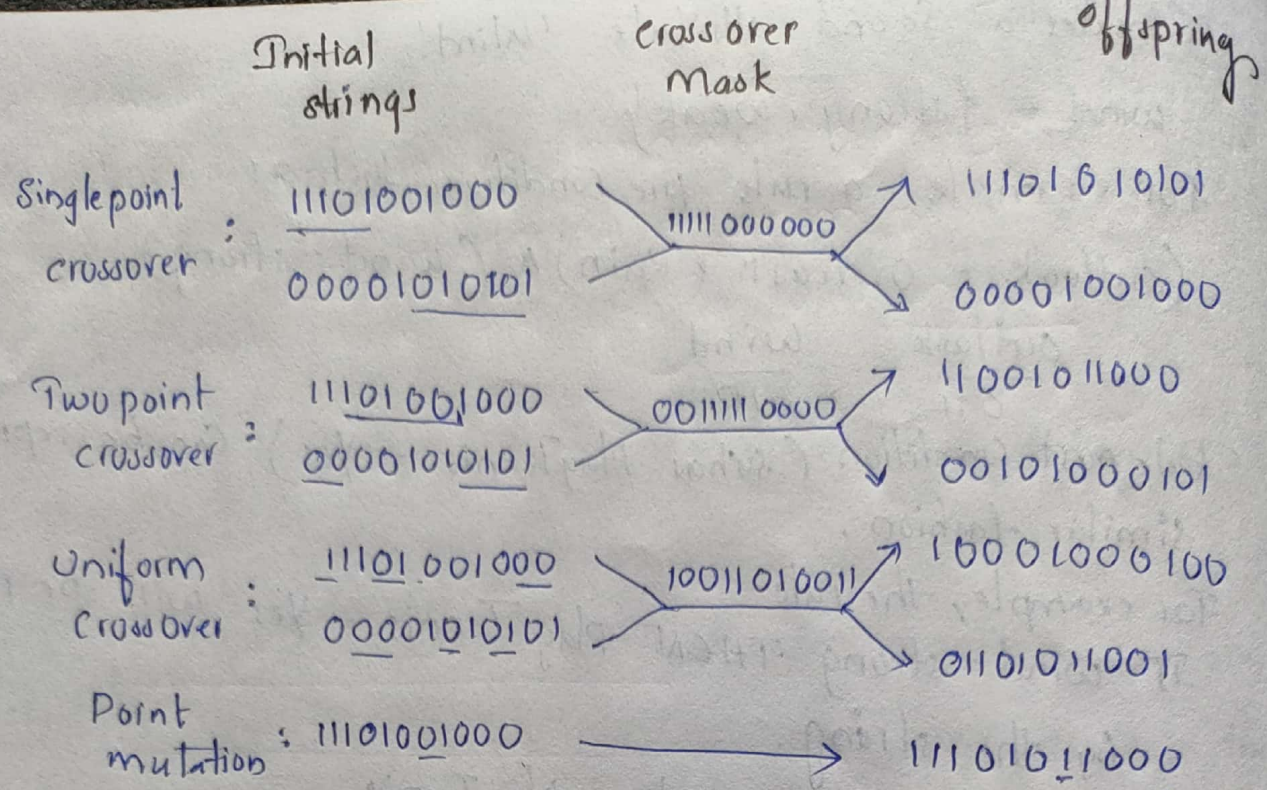
In some GAs hypotheses are represented by symbolic descriptions rather than bitstrings:

Genetic Operators

The generation of sucessors in a GA is determined by a set of Operators that recombine and mutate selected members of the current population. The two most common operators are 'Crossover' and 'mutation'. The 'Crossover' operator produces two new offspring from two parent strings by copying selected bits from each parent. The bit at position 'i' in each offspring is copied from the bit at position 'i' in one of the two parents.

The choice of which parent contributes the bit for position 'i' is determined by an additional Spring called the 'crossover mask'.

Ex:-



In particular, the 'mutation' operator produces small random changes to the bitstring by choosing a single bit at random, then changing its value.

Mutation is often performed after crossover has been applied, as in our 'prototypical algorithms'

Genetic Programming.

~~Genter~~ Genetic programming is a form of evolutionary computation in which individuals in the evolving population or computer programs rather than bit strings.

→ Programs manipulate by GP are typically represented by trees ~~Crosse~~ Coressponding to the parse tree of the program.

→ Each func call is represented by a node in a tree and arguments to the function are given by it's descending nodes.

Ex:- Function : $\sin(x) + \sqrt{x^2+y}$ is represented by the tree structure.

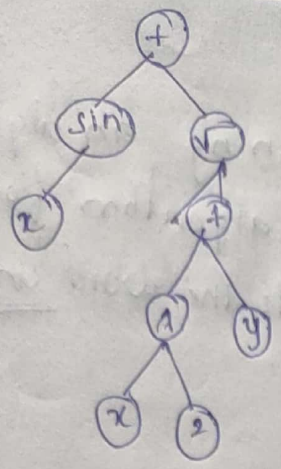


fig :- Program tree representation in GP.

In the algo on each iteration, in the algo it produces a new generation of individuals using selection, crossover, and mutation.

→ Crossover operations are performed by replacing a randomly choosen subtree of one pairent program by a subtree from other parent program.

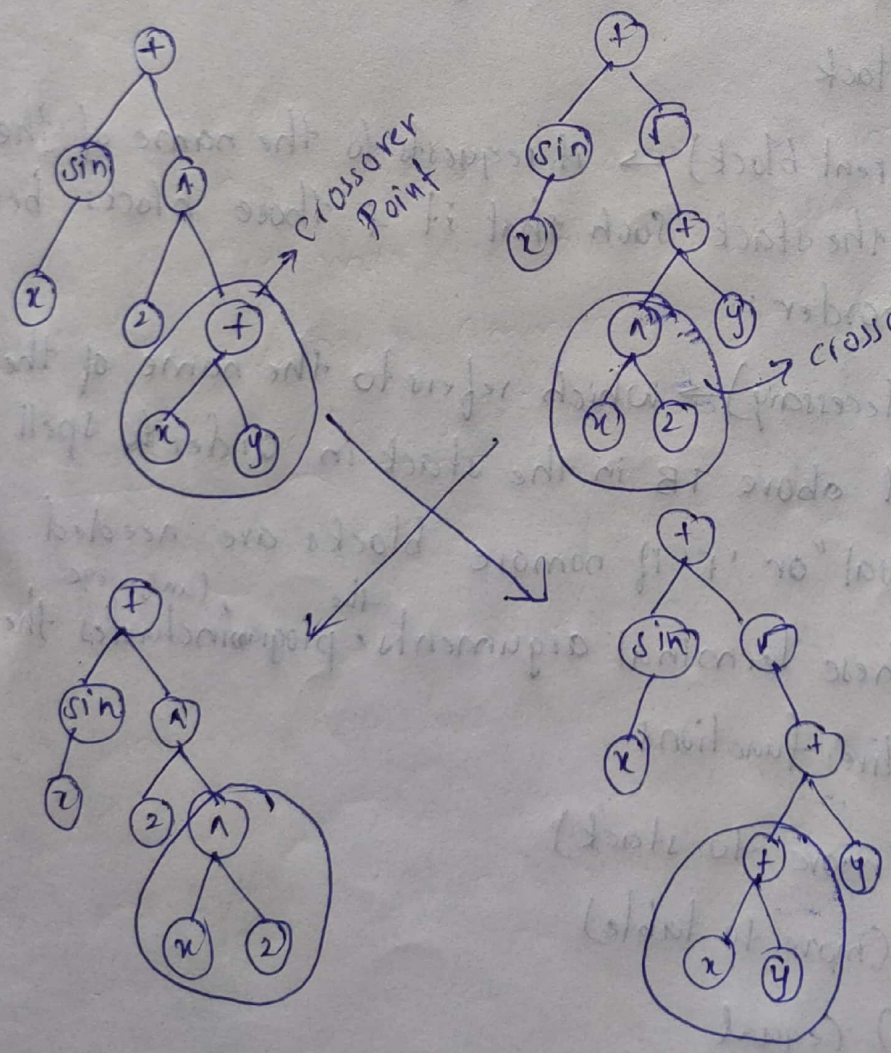


fig:- Crossover Operation applied to two programs.

Illustrative example :-

One illustrative example involves learning an algo for stacking the blocks. The task is to develop a general algorithm for stacking the blocks into a single stack that spells the word "universal"

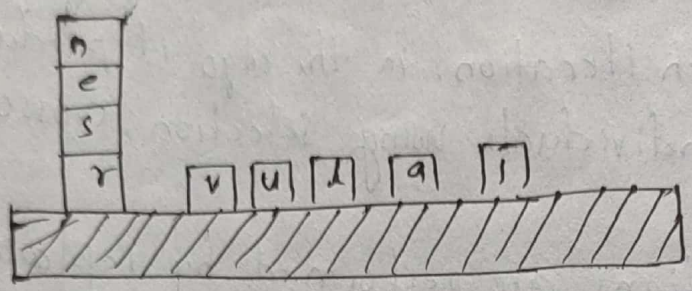


fig :- A block stacking program.

The following are the three terminal arguments

(i) CS (current stack) → which refers to the name of the top block on the stack

(ii) TB (Top current block) → It requests to the name of the topmost block on the stack such that it & those blocks beneath it are in the correct order.

(iii) NN (Next Necessary) → which refers to the name of the next block needed above TB in the stack in order to spell the word "universal" or 'F' if nomore blocks are needed.

In addition to these terminal arguments, the program language includes the following primitive functions.

i) (MS x) (move to stack)

ii) (MT x) (move to table)

iii) (EQ x y) (equal

iv) (NOT x)

v) (DU x y) (do until)

The population was initialize to a set a of 300 random programs after 10 generations the system discovered the following program, which solved all training example.

(EQ(DU(MT CS)(NOT CS))(DU(MS NN)(NOT NN)))