

Module 1

(Basic tools of soft computing)

Content

- Basic characteristics of AI and soft computing
- Evolutionary computing
- Approximation of Multivariate functions
- Non-linear error surface and optimization

Lecture - 1

INTRODUCTION

What is intelligence?

Real intelligence is what determines the normal thought process of a human.

Artificial intelligence is a property of machines which gives it ability to mimic the human thought process. The intelligent machines are developed based on the intelligence of a subject, of a designer, of a person, of a human being.

How do we achieve it?

Before we model a system, we need to observe.

What is AI?

Artificial Intelligence is concerned with the design of intelligence in an artificial device.

Practical applications of AI –

AI components are embedded in numerous devices like -in copy machines for automatic correction of operation for copy quality improvement.

AI systems are in everyday use -

1. For identifying credit card fraud
2. For advising doctors
3. For recognizing speech and in helping complex planning tasks.

4. Intelligent tutoring systems that provide students with personalized attention.

Intelligent behavior –

This discussion brings us back to the question of what constitutes intelligent behavior.

Some of these tasks and applications are:

1. Perception involving image recognition and computer vision
2. Reasoning
3. Learning
4. Understanding language involving natural language processing, speech processing
5. Solving problems
6. Robotics

What is soft computing?

Soft computing is an approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision.

It is characterized by the use of inexact solutions to computationally hard tasks such as the solution of nonparametric complex problems for which an exact solution can't be derived in polynomial of time.

Why soft computing approach?

Mathematical model & analysis can be done for relatively simple systems. More complex systems arising in biology, medicine and management systems remain intractable to conventional mathematical and analytical methods. Soft computing deals with imprecision, uncertainty, partial truth and approximation to achieve tractability, robustness and low solution cost. It extends its application to various disciplines of Eng. and science. Typically human can:

1. Take decisions
2. Inference from previous situations experienced
3. Expertise in an area
4. Adapt to changing environment
5. Learn to do better

Social behavior of collective intelligence intelligent control strategies have emerged from the above mentioned characteristics of human/ animals. The first two characteristics have given rise to Fuzzy logic; 2nd, 3rd and 4th have led to Neural Networks; 4th, 5th and 6th have been used in evolutionary algorithms.

Lecture – 2

Characteristics of Neuro-fuzzy and soft computing

1. Human Expertise
2. Biologically inspired computing models
3. New Optimization Techniques
4. Numerical Computation
5. New Application domains
6. Model-free learning
7. Intensive computation
8. Fault tolerance
9. Goal driven characteristics
10. Real world applications

Types of Soft Computing

1. Fuzzy Logic
2. Neural networks
3. Evolutionary Algorithms

Lecture- 3

Fuzzy Logic

When we say fuzzy logic, that is the variables that we encounter in physical devices, fuzzy numbers are used to describe these variables and using this methodology when a controller is designed, it is a fuzzy logic controller.

- Let us take three statements: zero, almost zero, near zero.
- Zero is exactly zero with truth value assigned 1
- If it is almost 0, then I can think that between minus 1 to 1, the Values around 0 is 0, because this is almost 0.
- When I say near 0, maybe the bandwidth of the membership which represents actually the truth value.

You can see that it is more, bandwidth increases near 0.

-This is the concept of fuzzy number.

- Without talking about membership now, but a notion is that I allow some small bandwidth when I say almost 0.
- When I say near 0 my bandwidth still further increases.
- In the case minus 2 to 2, when I encounter any data between minus 2 to 2, still I will consider them to be near 0.
- As I go away from 0 towards minus 2, the confidence level how near they are to 0 reduces; like if it is very near to 0, I am very certain.

-As I progressively go away from 0, the level of confidence also goes down, but still there is a tolerance limit. So when zero I am precise, I become imprecise when almost and I further become more imprecise in the third case.

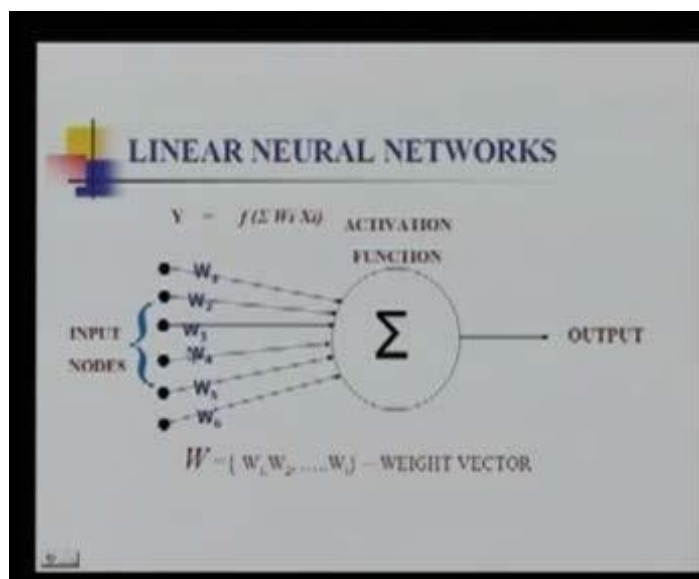
When we say fuzzy logic, that is the variables that we encounter in physical devices, fuzzy numbers are used to describe these variables and using this methodology when a controller is designed, it is a fuzzy logic controller.

Lecture-4

Neural Networks

- Neural networks are basically inspired by various way of observing the biological organism.
- Most of the time, it is motivated from human way of learning. It is a learning theory. This is an artificial network that learns from example and because it is distributed in nature, fault tolerant, parallel processing of data and distributed structure.
- The basic elements of artificial Neural Network are:
 1. Input nodes
 2. Weights
 3. Activation function and
 4. Output node.
- Inputs are associated with synaptic weights. They are all Summed and passed through an activation function giving output y . In a way, output is summation of the signal multiplied with synaptic weight over many input channels.

Fig. Basic elements of an artificial neuron



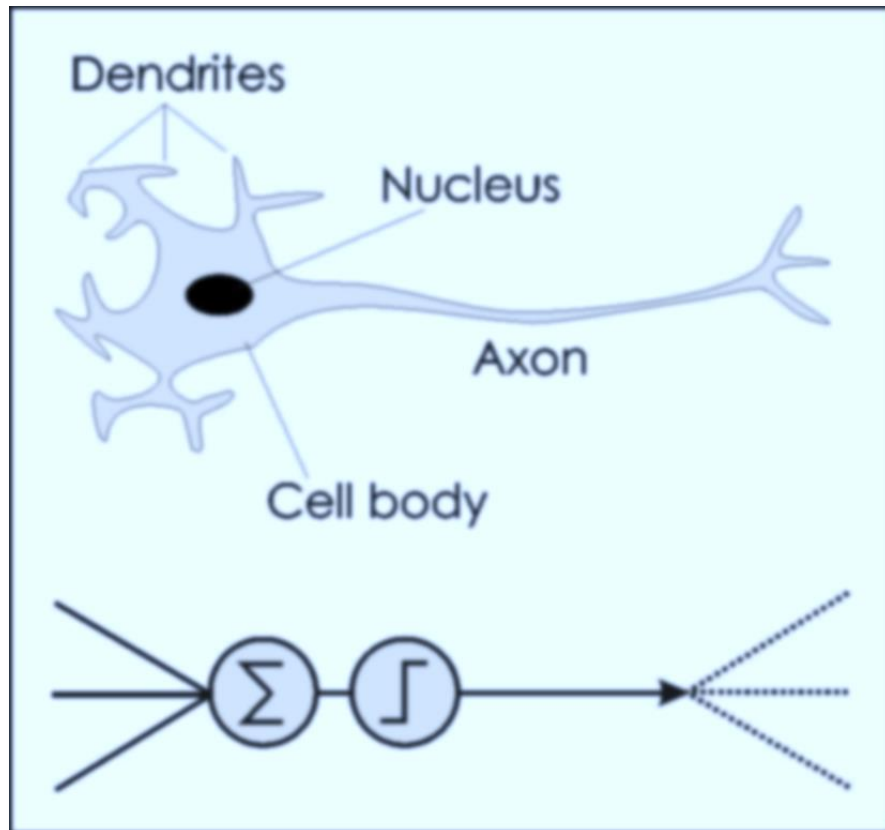


Fig. Analogy of Biological neuron and artificial neuron

- Above fig. Shows a biological neuron on top.
- Through axon this neuron actuates the signal and this signal is sent out through synapses to various neurons.
- Similarly shown a classical artificial neuron(bottom).
- This is a computational unit.
- There are many inputs reaching this.
- The input excites this neuron.
- Similarly, there are many inputs that excite this computational unit and the output again excites many other units like here.
- Like that taking certain concepts in actual neural network, we develop these artificial computing models having similar structure.
- There are various locations where various functions take place in the brain.

If we look at a computer and a brain, this is the central processing unit and a brain.

Let us compare the connection between our high speed **computers** that are available in the market today and **a brain.**

- Approximately there are 10^{14} synapses in the human brain, whereas typically you will have 10^8 transistors inside a CPU.
- The element size is almost comparable, both are 10^{-6} and energy use is almost like 30 Watts and comparable actually; that is energy dissipated in a brain is almost same as in a computer. But you see the processing speed.
- Processing speed is only 100 hertz; our brain is very slow, whereas computers nowadays, are some Giga hertz.
- When you compare this, you get an idea that although computer is very fast, it is very slow to do intelligent tasks like pattern recognition, language understanding, etc.
- These are certain activities which humans do much better, but with such a slow speed, 100 Hz contrast between these two, one of the very big difference between these two is the structure; one is brain, another is central processing unit is that the brain learns, we learn.
- Certain mapping that is found in biological brain that we have studied in neuroscience is not there in a central processing unit and we do not know whether self awareness takes place in the brain or somewhere else, but we know that in a computer there is

no self-awareness.

Neural networks are analogous to adaptive control concepts that we have in control theory and one of the most important aspects of intelligent control is to learn the control parameters, to learn the system model.

- Some of the learning methodologies we will be learning here is the error-back propagation algorithm, real-time learning algorithm for recurrent network, Kohonen's self organizing feature map & Hopfield network.



Lecture- 5

Features of Artificial Neural Network(ANN) models

1. Parallel Distributed information processing
2. High degree of connectivity between basic units
3. Connections are modifiable based on experience
4. Learning is a continuous unsupervised process
5. Learns based on local information
6. Performance degrades with less units

So, in this paper we will be discussing various tools.

- These tools are actually developed by mimicking the human behavior, but not the human way of working.
- An intelligent machine is one which learns, thinks and behaves in line with the thought process. (That we would like but we are very far from it. At least, at the moment, we are very far from this target of achieving real intelligence.)
- We perceive the environment in a very unique way, in a coherent manner.
- This is called unity of perception and intelligence has also something to do with this unity of perception, awareness and certain things are not very clear to us until now.
- So an intelligent machine is one which learns, thinks & behaves in line with thought process.

Lecture-6

Characteristics of soft computing

1. Human Expertise
2. Inspired computing models
3. New operation techniques
4. Model-free learning
5. Intensive Computation
6. Fault tolerance

Types of Soft Computing

1. Fuzzy Logic
2. Neural Networks
3. Evolutionary algorithms

Fuzzy logic

- It is a mathematical language to express something.
- It is a language; it has grammar, syntax for communication.
- It is another form of logic whose range lies between 0 to 1
- It means number of elements present will give the probable values to the computing procedure.

Logic is of two types

1. Fuzzy logic
2. Crisp logic

Crisp logic

Crisp logic depends upon the direct true or false value. It means exactly completely true or exactly completely false.

True $\rightarrow 1$

False $\rightarrow 0$

According to crisp logic:

Is Ram Honest ? $\{0,1\}$

\rightarrow True/Yes/1

\rightarrow False/No/0

According to fuzzy logic:

Is Ram Honest ?

\rightarrow True/Yes/1

\rightarrow Very Honest/0.85

\rightarrow Sometimes Honest/0.35

\rightarrow False/No/0

Fuzzy Set

Membership function(U_A)

U_A = Membership values for the set A

$U_{\bar{A}} = x \rightarrow [0,1]$

Crisp Set

$U_A : x \rightarrow [0,1]$

Fuzzy set is defined as

$\{(x, U_{\bar{A}}(x)) \mid x \in X\}$

Lecture-7

Q. Create a fuzzy set with its membership values to represent in the form of fuzzy equation.

$X = \{1, 2, 3, 4, 5, 6, \dots\}$ for $\forall x \in X$

$\bar{A} = \{\text{"two or so"}\}$

Ans-

$$U_{\bar{A}}(1) = 0.5$$

$$U_{\bar{A}}(2) = 1$$

$$U_{\bar{A}}(3) = 0.5$$

$$U_{\bar{A}}(4) = 0$$

$$U_{\bar{A}}(5) = 0$$

$$U_{\bar{A}}(6) = 0$$

So, the required fuzzy set for the above problem is

$$\bar{A} = \{(1, 0.5), (2, 1), (3, 0.5), (4, 0), (5, 0), (6, 0)\}$$

Fuzzy set equation is:

$$(U_{\bar{A}}(x_1)/x_1) + (U_{\bar{A}}(x_2)/x_2) + (U_{\bar{A}}(x_3)/x_3) + \dots$$

Where $x_1, x_2, x_3, \dots \in X$

$$\Rightarrow (0.5/1) + (1/2) + (0.5/3) + (0/4) + (0/5) + (0/6) + \dots$$

Q. $X = \{10, 20, 30, 40, 50, 60, 70, 80\}$ set of group

$\bar{A} = \{\text{"John is young"}\}$

Ans-

$$U_{\bar{A}}(10) = 1$$

$$U_{\bar{A}}(20) = 0.9$$

$$U_{\bar{A}}(30) = 0.8$$

$$U_{\bar{A}}(40) = 0.5$$

$$U_{\bar{A}}(50) = 0.2$$

$$U_{\bar{A}}(60) = 0$$

$$U_{\bar{A}}(70) = 0$$

$$U_{\bar{A}}(80) = 0$$

So, the required fuzzy set for the above problem is

$$\bar{A} = \{(10, 1), (20, 0.9), (30, 0.8), (40, 0.5), (50, 0.2), (60, 0), (70, 0), (80, 0)\}$$

Fuzzy set equation is:

$$(U_{\bar{A}}/x_1)/x_1 + (U_{\bar{A}}/x_2)/x_2 + (U_{\bar{A}}/x_3)/x_3 + \dots$$

Where $x_1, x_2, x_3, \dots \in X$

$$\Rightarrow (1/10) + (0.9/20) + (0.8/30) + (0.5/40) + (0.2/50) + (0/60) + (0/70) + (0/80)$$

Lecture-8

If the equation comes with same data but another

$\bar{A}_2 = \{\text{"John is young"}\}$

$(\bar{A}_2 \in \bar{A}_1)$ \bar{A}_2 is a subset of \bar{A}_1

$Q \rightarrow X = \{5, 15, 25, 35, 45, 55, 65, 75, 85\}$

Fuzzy set

1. Infant
2. Young
3. Adult
4. Senior

Age	Infant	Young	Adult	Senior
5	0	0	0	0
15	0	0.2	0	0
25	0	0.8	0.8	0
35	0	1	0.9	0
45	0	0.6	1	0
55	0	0.5	1	0.3
65	0	0.1	1	0.9
75	0	0	1	1
85	0	0	1	1

For young

$$U_{\bar{A}}(5) = 0$$

$$U_{\bar{A}}(15) = 0.2$$

$$U_{\bar{A}}(25) = 0.8$$

$$U_{\bar{A}}(35) = 1$$

$$U_{\bar{A}}(45) = 0.6$$

$$U_{\bar{A}}(55) = 0.5$$

$$U_{\bar{A}}(65) = 0.1$$

$$U_{\bar{A}}(75) = 0$$

$$U_{\bar{A}}(85) = 0$$

So, the required fuzzy set for very young is

$$\bar{A} = \{(5,0), (15,0.2), (25,0.8), (35,1), (45,0.6), (55,0.5), (65,0.1), (75,0), (85,0)\}$$

Fuzzy set equation is:

$$(U_{\bar{A}}/(x_1))/x_1 + (U_{\bar{A}}/(x_2))/x_2 + (U_{\bar{A}}/(x_3))/x_3 + \dots$$

Where $x_1, x_2, x_3, \dots \in X$

$$\Rightarrow (0/5) + (0.2/15) + (0.8/25) + (1/35) + (0.6/45) + (0.5/55) + (0.1/65) + (0/75) + (0/85)$$

For adult

$$U_{\bar{A}}(5) = 0$$

$$U_{\bar{A}}(15) = 0$$

$$U_{\bar{A}}(25) = 0.8$$

$$U_{\bar{A}}(35) = 0.9$$

$$U_{\bar{A}}(45) = 1$$

$$U_{\bar{A}}(55) = 1$$

$$U_{\bar{A}}(65) = 1$$

$$U_{\bar{A}}(75) = 1$$

$$U_{\bar{A}}(85) = 1$$

So, the required fuzzy set is

$$\bar{A} = \{(5,0), (15,0), (25,0.8), (35,0.9), (45,1), (55,1), (65,1), (75,1), (85,1)\}$$

Fuzzy set equation is:

$$(U_{\bar{A}}/(x1))/x1 + (U_{\bar{A}}/(x2))/x2 + (U_{\bar{A}}/(x3))/x3 + \dots$$

Where $x1, x2, x3, \dots \in X$

$$\Rightarrow (0/5) + (0/15) + (0.8/25) + (0.9/35) + (1/45) + (1/55) + (1/65) + (1/75) + (1/85)$$

For senior the fuzzy set is

$$\bar{A} = \{(5,0), (15,0), (25,0), (35,0), (45,0), (55,0.3), (65,0.9), (75,1), (85,1)\}$$

Fuzzy set equation is:

$$(U_{\bar{A}}/(x1))/x1 + (U_{\bar{A}}/(x2))/x2 + (U_{\bar{A}}/(x3))/x3 + \dots$$

Where $x1, x2, x3, \dots \in X$

$$\Rightarrow (0/5) + (0/15) + (0/25) + (0/35) + (0/45) + (0.3/55) + (0.9/65) + (1/75) + (1/85)$$

Module-2

Lecture-1

Implement the basic fuzzy set operations

$$\bar{A}=\{(1.0,1),(1.5,0.75),(2.0,0.3),(2.5,0.15),(3.0,0)\} \forall x \in X$$

$$\bar{B}=\{(1.0,1),(1.5,0.6),(2.0,0.2),(2.5,0.1),(3.0,0)\} \forall y \in Y$$

$$\bar{C} = \bar{A} \cup \bar{B} = \max(\mu_{\bar{A}}(x_1), \mu_{\bar{B}}(y_1)) / x_1 + y_1$$

$$\bar{C} = \bar{A} \cap \bar{B} = \min(\mu_{\bar{A}}(x_1), \mu_{\bar{B}}(y_1)) / x_1 + y_1$$

$$\bar{A}=\{(0.5/x_1)+(0.2/x_2)+(0.9/x_3)\}$$

$$X=x_1,x_2,x_3=3$$

$$\bar{B}=\{(1/y_1)+(0.5/y_2)+(1/y_3)\}$$

$$Y=y_1+y_2+y_3=3$$

$$\bar{A} = \{(x_1,0.5),(x_2,0.2),(x_3,0.9)\}$$

$$\bar{B} = \{(y_1,1),(y_2,0.5),(y_3,1)\}$$

The Cartesian of \bar{A} and \bar{B} are

	y1	y2	y3
R= x1	0.5	0.5	0.5
x2	0.2	0.2	0.2
x3	0.9	0.5	0.9

$$\Rightarrow \mu_R(x1,y1) = \text{Min}[\mu_{\bar{A}}(x1), \mu_{\bar{B}}(y1)] \\ = \min(0.5, 1)$$

Fuzzy equation

$$R = (0.5/x1y1) + (0.5/x1y2) + (0.5/x1y3) + (0.2/x2y1) + (0.2/x2y2) + (0.2/x2y3) + \\ (0.9/x3y1) + (0.5/x3y2) + (0.9/x3y3)$$

Lecture-2

Fuzzy Composition

A fuzzy relation \bar{R} is mapping function from the Cartesian space " $X * Y$ " to the interval between $[0, 1]$ where the membership function of the relation \bar{R} will be expressed as:

$$\mu_{\bar{R}}(x,y)$$

$$\mu_R : \bar{A} * \bar{B} \rightarrow [0,1]$$

$$R = \{(x,y), \mu_R(x,y) \mid \mu_R(x,y) \geq 0\}$$

Where $\forall : x \in \bar{A}, y \in \bar{B}$

Let \bar{R} is a fuzzy relation on the Cartesian space($X*Y$)

Let \bar{S} is a fuzzy relation in between ($Y * Z$)

$$T = X*Z \rightarrow \mu_T(x,z)$$

$$T = \bar{R} \circ \bar{S}$$

Different ways to solve the composition

- 1. Max-mini composition
- 2. max-product composition

Composition Rule

$$T(x,z) = \max(\min(R(x,y), s(y,z)))$$

$$y \in Y$$

Extra problem-

$$X=\{x_1, x_2\}, Y=\{y_1, y_2\}, Z=\{z_1, z_2, z_3\}$$

$$R = \begin{array}{cc} & \begin{array}{cc} y_1 & y_2 \end{array} \\ \begin{array}{c} x_1 \\ x_2 \end{array} & \begin{array}{cc} 0.6 & 0.3 \\ 0.2 & 0.9 \end{array} \end{array} \quad S = \begin{array}{cc} & \begin{array}{ccc} z_1 & z_2 & z_3 \end{array} \\ \begin{array}{c} y_1 \\ y_2 \end{array} & \begin{array}{ccc} 0.1 & 0.5 & 0.3 \\ 0.8 & 0.4 & 0.7 \end{array} \end{array}$$

Find T?

Ans-

$$\mu_T(x_1, z_1) = \max\{\min[\mu_R(x_1, y_1), \mu_S(y_1, z_1)], \min[\mu_R(x_1, y_2), \mu_S(y_2, z_1)]\}$$

$$= \max\{\min(0.6, 1), \min(0.3, 0.8)\}$$

$$= \max(0.6, 0.3)$$

$$= 0.6$$

$$\mu_T(x_2, z_2) = \max\{\min[\mu_R(x_2, y_1), \mu_S(y_1, z_2)], \min[\mu_R(x_2, y_2), \mu_S(y_2, z_2)]\}$$

$$= \max\{\min(0.2, 0.5), \min(0.9, 0.4)\}$$

$$= \max(0.2, 0.4)$$

$$= 0.4$$

$$\mu_T(x_1, z_2) = \max\{\min(0.6, 0.5), \min(0.3, 0.4)\}$$

$$= \max(0.5, 0.3)$$

$$= 0.5$$

$$\mu_T(x_1, z_3) = \max\{\min(0.6, 0.3), \min(0.3, 0.7)\}$$

$$= \max(0.3, 0.3)$$

=0.3

$$\mu_T(x_2, z_1) = \max\{\min(0.2, 1), \min(0.9, 0.8)\}$$

$$= \max(0.2, 0.8)$$

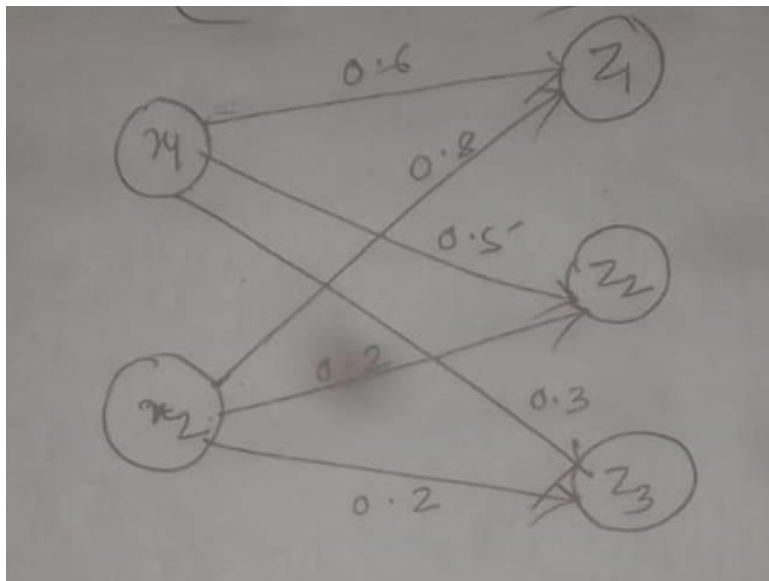
=0.8

$$\mu_T(x_2, z_3) = \max\{\min(0.2, 0.3), \min(0.9, 0.7)\}$$

$$= \max(0.2, 0.7)$$

=0.7

		z1	z2	z3
T=	x1	0.6	0.5	0.3
	x2	0.8	0.2	0.2



Lecture-2

Max-product composition

$$T(x,z)=\max\{\mu_R(x,y) \cdot \mu_S(y,z)\}$$

Same above of Q-1

$$\mu_T(x1,z1)=\max\{[\mu_R(x1,y1) \cdot \mu_S(y1,z1)], [\mu_R(x1,y2) \cdot \mu_S(y2,z1)]\}$$

$$=\max\{(0.6,0.1) \cdot (0.3,0.8)\}$$

$$=\max(0.6,0.24)$$

$$=0.6$$

$$\mu_T(x1,z1)=0.6$$

- Similarly

$$\mu_T(x1,z2)=0.3$$

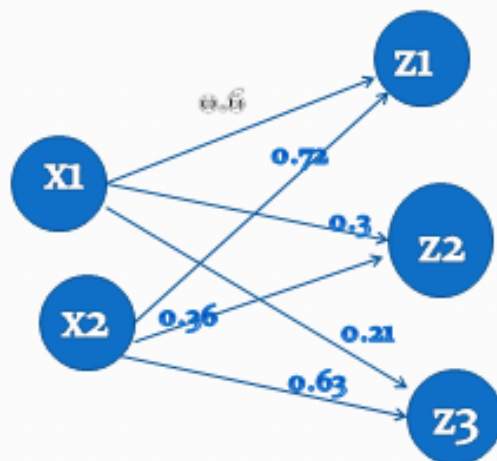
$$\mu_T(x1,z3)=0.21$$

$$\mu_T(x2,z1)=0.72$$

$$\mu_T(x2,z2)=0.36$$

$$\mu_T(x2,z3)=0.63$$

$$T = \begin{matrix} & \begin{matrix} z1 & z2 & z3 \end{matrix} \\ \begin{matrix} x1 \\ x2 \end{matrix} & \begin{bmatrix} 0.6 & 0.3 & 0.21 \\ 0.72 & 0.36 & 0.63 \end{bmatrix} \end{matrix}$$



Lecture-3

Types of membership function graph

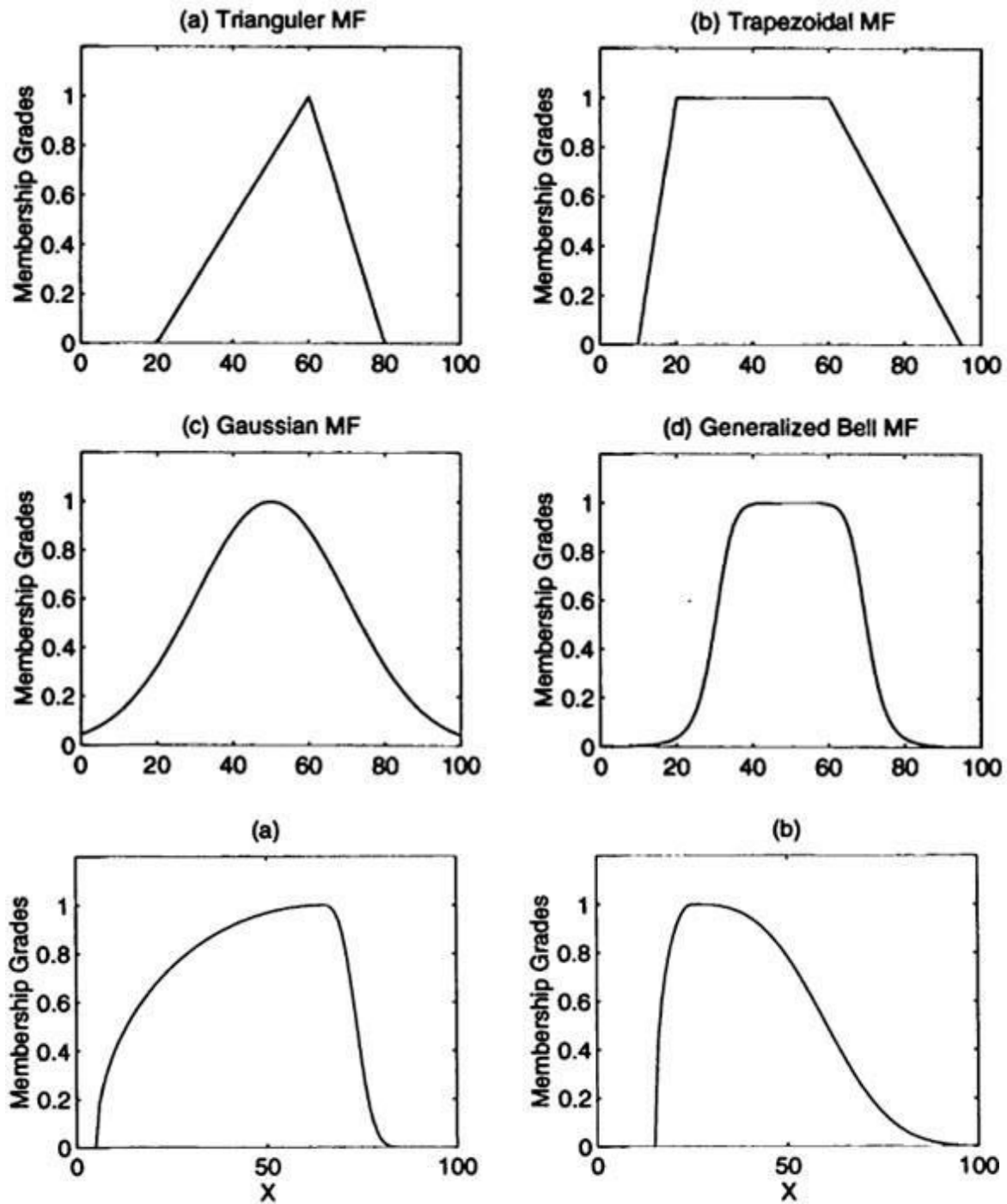


Fig. Membership functions a. Triangle b. Trapezoidal c. Gaussian d. Bell, e. Left f. Right

Fuzzy Formulation with parameters

- Triangular MFC is specified by 3 parameters {a,b,c}. So, it can be formulated as follows

$$\text{Triangular } (x:a,b,c) = \begin{cases} 0 & \text{if } x \leq a \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{if } b \leq x \leq c \\ 0 & \text{if } c \leq x \end{cases}$$

- Trapezoidal MFC is specified by 4 parameters {a,b,c,d} diagram

$$\text{Trapezoidal } (x:a,b,c,d) = \begin{cases} 0 & \text{if } x \leq a \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ 1 & \text{if } b \leq x \leq c \\ \frac{c-x}{c-b} & \text{if } c \leq x \leq d \\ 0 & \text{if } d \leq x \end{cases}$$

- Π shape MFC - Π shape MFC(x:c,σ)

$$= e^{-1/2((x-c)/\sigma)^2}$$

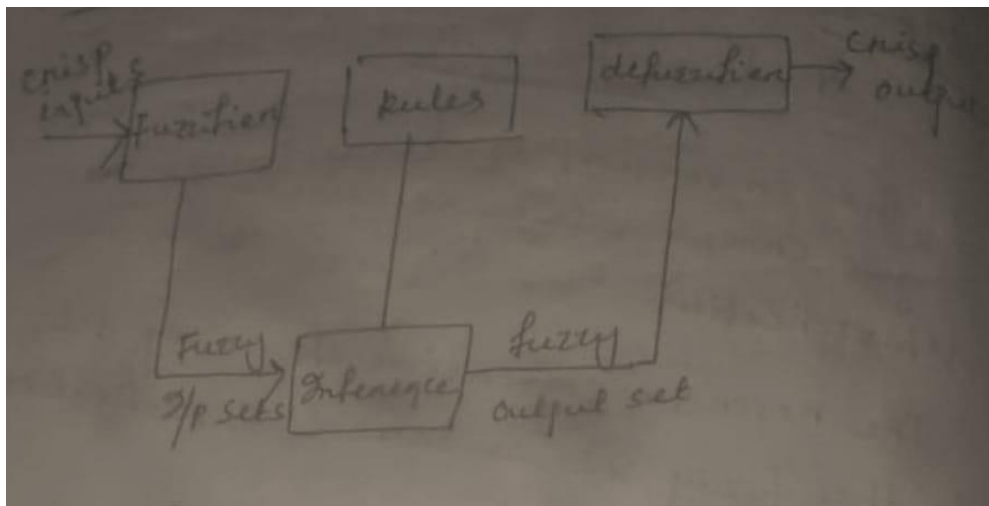
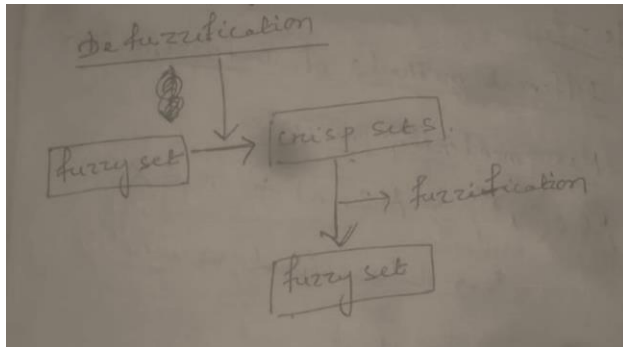
- Gaussian MFC- Gaussian(x:a,b,c)=(1)/(1+(|((x - c)/a)|)^{2b})

Lecture-4

Defuzzification-

The conversion of fuzzy set to a single crisp value is called as defuzzification.

The reverse conversion of crisp set to the fuzzy set is called fuzzification.



Different methods of defuzzification

1. Max-membership principle

$$\mu_C(x^*) \geq \mu_C(x) \quad \forall : x \in X$$

2. Centroid method

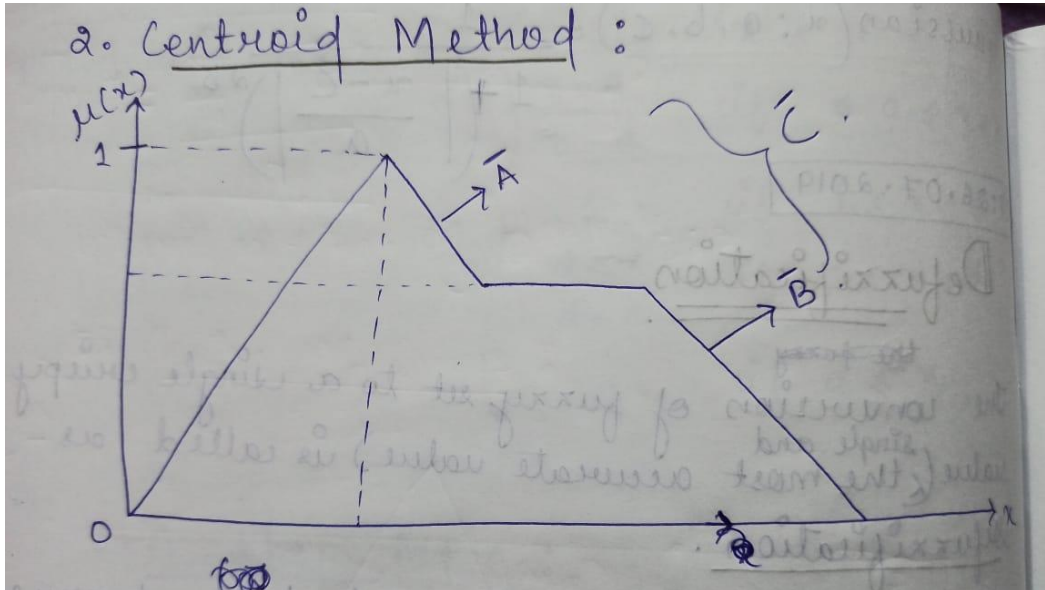
If $\bar{A} \rightarrow 1^{\text{st}}$ membership function

$\bar{B} \rightarrow 2^{\text{nd}}$ membership function

then $\bar{C} \rightarrow$ Defuzzification input of A and B

Then

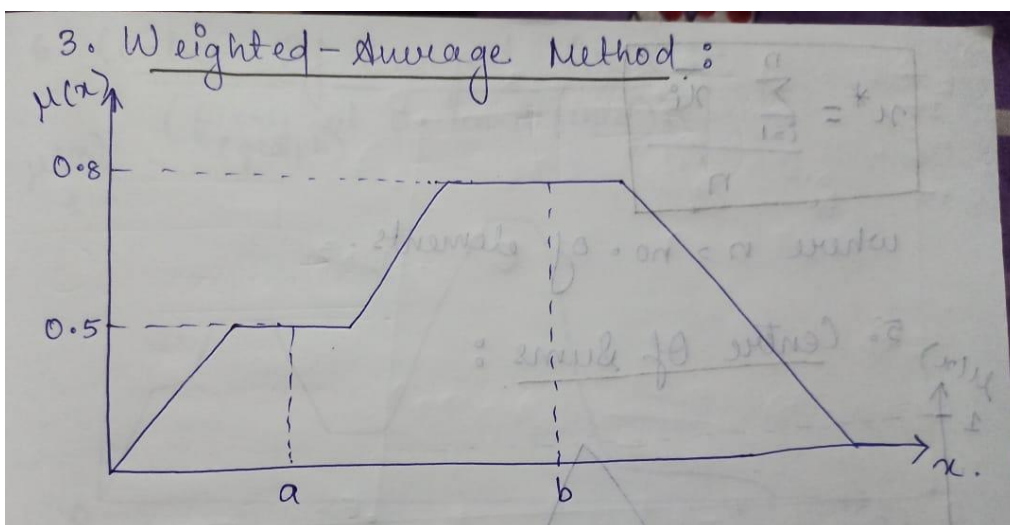
$$x^* = (\int \mu_{\bar{C}}(x) \cdot x \, dx) / (\int \mu_{\bar{C}}(x) \cdot dx)$$



3. Weighted Average method

$$x^* = (\sum \mu_C(x_i) \cdot x_i) / (\sum \mu_C(x_i))$$

This method is valid for symmetrical output.

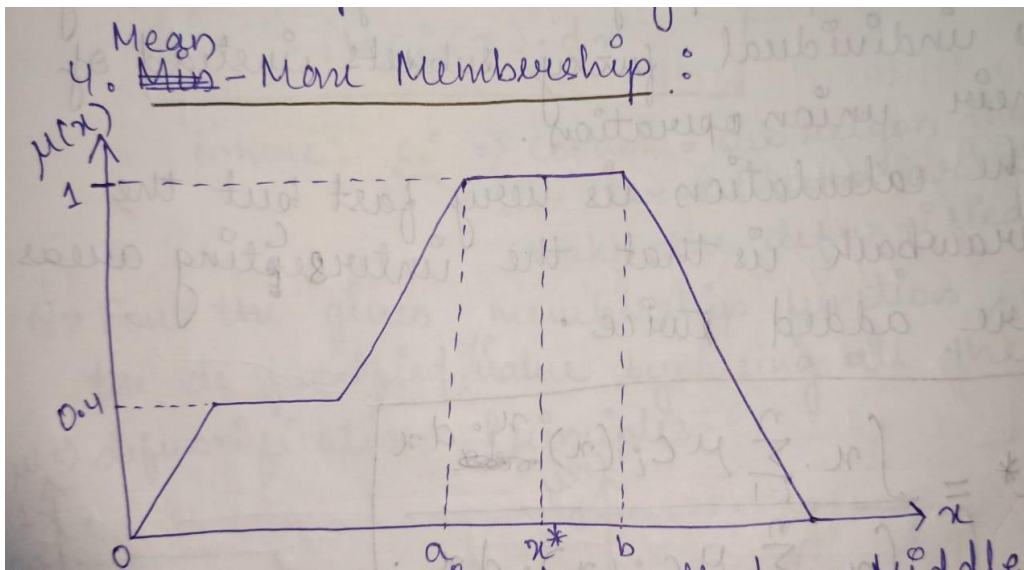


4. Mean-max membership

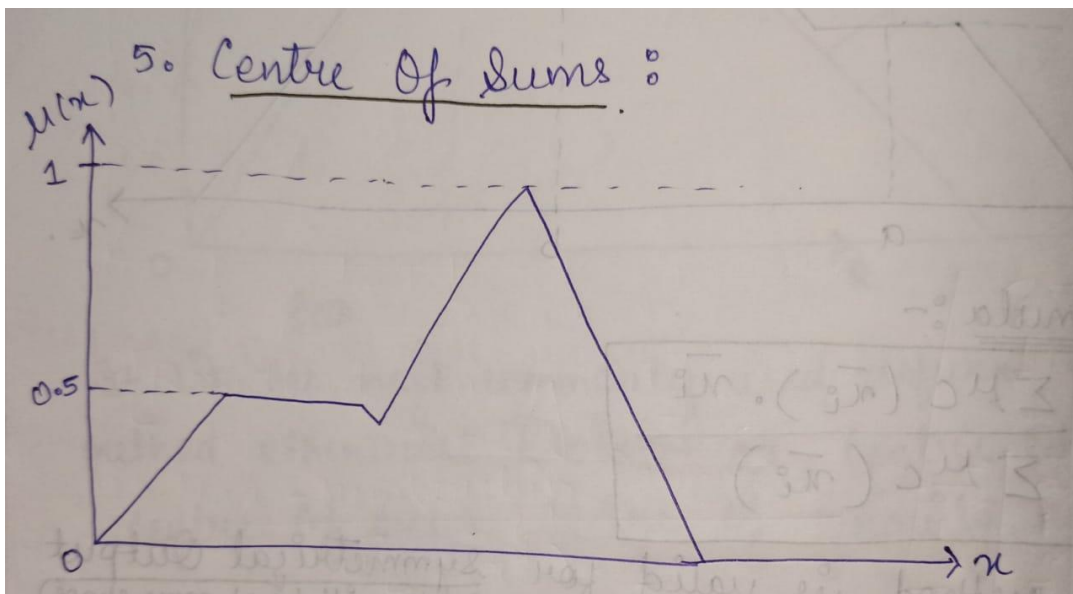
This method is also called as middle of maxima which is closely related to max membership function.

$$(x)^* = (\sum_{i=1}^n x_i) / n$$

where n = number of elements



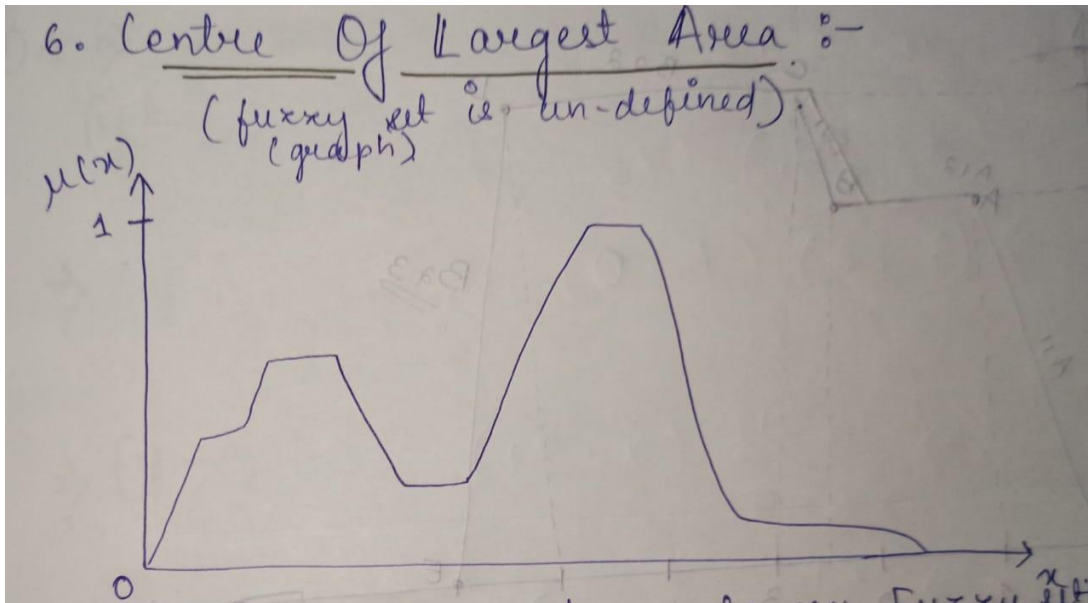
5. Centre of sums



This method is the algebraic sum of the individual subsets instead of their union operations.

$$(x)^* = \left(\int x \sum_{i=1}^n \mu_{Ci}(x) x \cdot dx \right) / \left(\int x \sum_{i=1}^n \mu_{Ci}(x) dx \right)$$

6. Centre of largest area



It is also called as convex fuzzy sets.

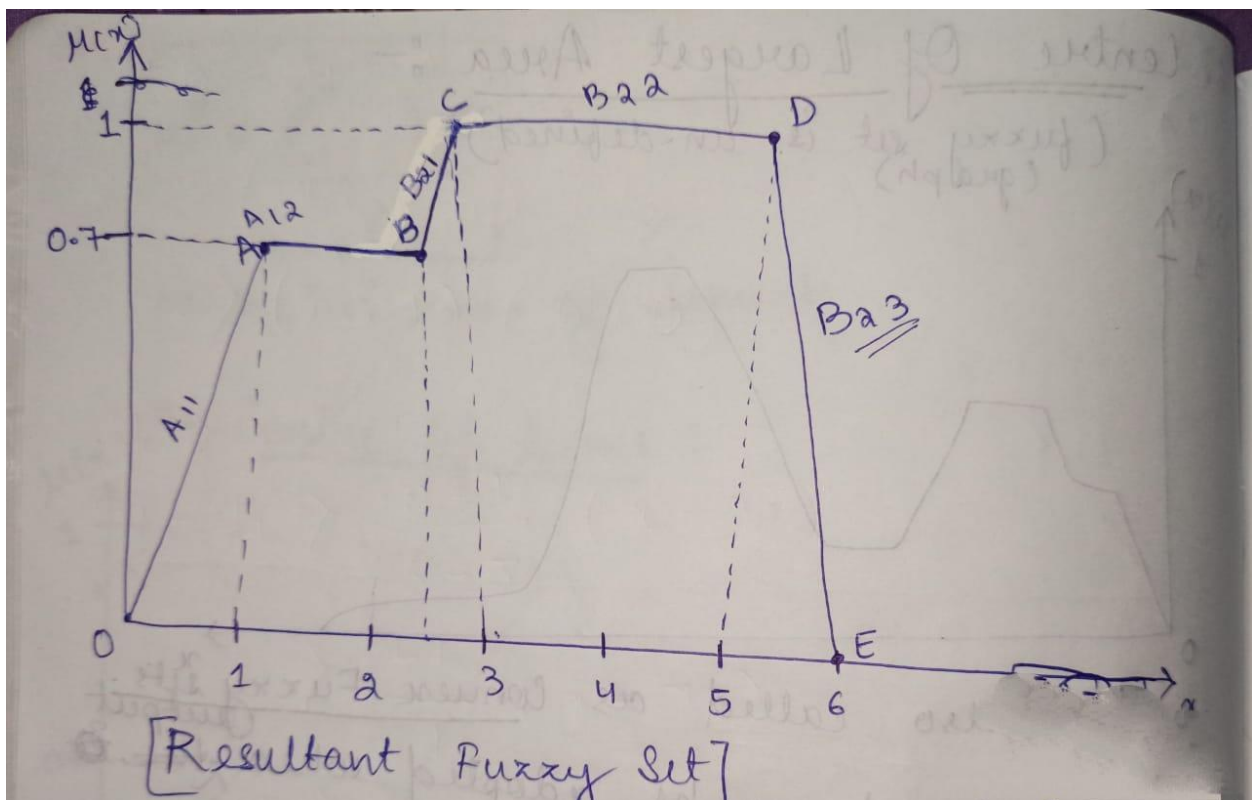
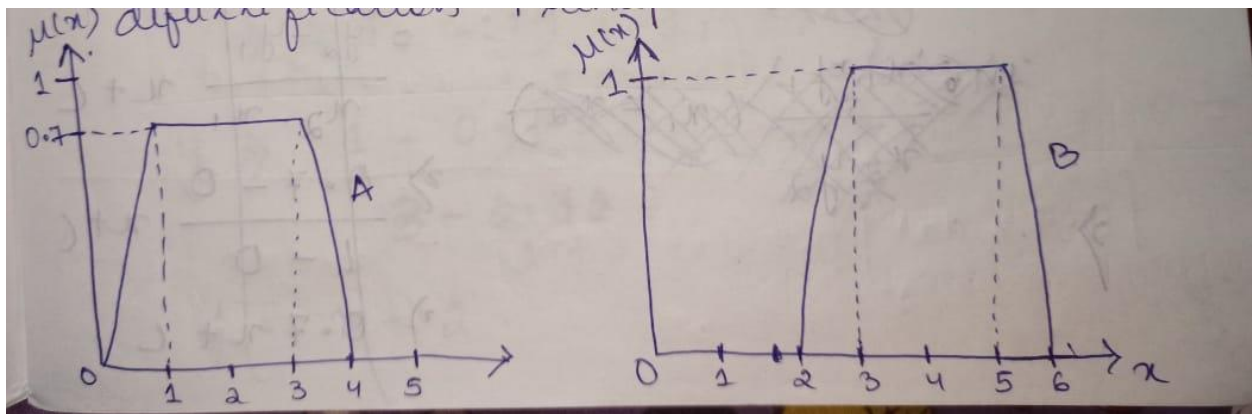
This method can be adopted when output consists of 2 convex-fuzzy subsets and which are not overlapping.

$$(x)^* = \left(\int \mu_{Ci}(x) x \cdot dx \right) / \left(\int \mu_{Ci}(x) dx \right)$$

Where C_i = convex sub-region that has the largest area to make the defuzzical value

Lecture-5

Q. For the given membership function, determine the de-fuzzified output value by using all the defuzzification principles.



i>max-membership principle

$$\mu_c(x^*) = 1$$

$$x^* = 1 \quad \text{yes}$$

ii>Centroid method

$$(x_1 \quad y_1) \quad (x_2 \quad y_2)$$

$$(0 \quad 0) \quad (1 \quad 0.7)$$

$$OA \Rightarrow y = mx + c$$

$$y - y_1 = \{(y_2 - y_1) / (x_2 - x_1)\} * (x - x_1)$$

$$\Rightarrow (y - y_1) = \{(0.7 - 0) / (1 - 0)\} * (x - x_1)$$

$$\Rightarrow (y - y_1) = (0.7/1)(x - x_1)$$

$$\Rightarrow y = 0.7(x)$$

$$AB \Rightarrow y = 0.7$$

$$BC \Rightarrow (x_1 \quad y_1) \quad (x_2 \quad y_2)$$

$$(2.7 \quad 0.7) \quad (3 \quad 1)$$

$$y \Rightarrow \{(1 - 0.7) / (3 - 2.7)\} * (x - x_1)$$

$$y = x - 2$$

$$CD = y = 1$$

$$DE = (x_1 \quad y_1) \quad (x_2 \quad y_2)$$

$$(5 \quad 1) \quad (6 \quad 0)$$

$$y - y_1 \Rightarrow \{(0 - 1) / (6 - 5)\} * (x - x_1)$$

$$\Rightarrow -1/1(x-x_1)$$

$$y \Rightarrow -x-5+1$$

$$\Rightarrow -x+6$$

$$y=x+6$$

$$DE=y=x+6$$

$$(x)^* = (\int \mu_C(x) x \cdot dx) / (\int \mu_C(x) dx)$$

$$\begin{aligned} x^* &= \frac{\int \mu_C(x) \cdot x \, dx}{\int \mu_C(x) \cdot dx} \\ &= \frac{\int_0^1 0.7x^2 \cdot dx + \int_1^{2.7} 0.7x \cdot dx + \int_{2.7}^3 (x^2 - 2x) \cdot dx + \int_3^5 x \cdot dx + \int_5^6 (6x - x^2) \cdot dx}{\int_0^1 0.7x \cdot dx + \int_1^{2.7} 0.7 \cdot dx + \int_{2.7}^3 (x-2) \cdot dx + \int_3^5 1 \cdot dx + \int_5^6 (6-x) \cdot dx} \end{aligned}$$

$$\begin{aligned}
& 0.7 \int_0^1 \left[\frac{x^3}{3} \right] \cdot dx + 0.7 \int_1^2 \left[\frac{x^2}{2} \right] \cdot dx + \int_2^3 \left[\frac{x^3}{3} \right] - 2 \left[\frac{x^2}{2} \right] dx \\
& \quad + \int_3^5 \left[\frac{x^2}{2} \right] \cdot dx + \int_5^6 \left[\frac{x^2}{2} \right] - 6 \left[\frac{x^3}{3} \right] \cdot dx \\
& \Rightarrow \frac{0.7 \int_0^1 \left[\frac{x^2}{2} \right] \cdot dx + 0.7 \int_1^2 [1] \cdot dx + \int_2^3 \left[\frac{x^2}{2} \right] - 2 \int_2^3 [1] \cdot dx + \int_3^5 \left[\frac{x^2}{2} \right] \cdot dx}{0.7 \times \frac{1}{3} + 0.7 \left(\frac{7 \cdot 29}{2} - \frac{1}{2} \right) + \frac{27 - 7 \cdot 29}{3} - 2 \frac{9 - 7 \cdot 2}{2}} \\
& \quad + 6 \cdot \frac{36 - 25}{2} + \frac{216 - 125}{3} \\
& \Rightarrow \frac{0.7 \times \left[\frac{1}{2} \right] + 0.7 \times 1 \cdot 7 + \left[\frac{1}{2} \right]_{2.7}^3 \cdot 2 + [1]_3^5 + 6 \left[\frac{1}{2} \right] \cdot dx}{\Rightarrow 3.22}
\end{aligned}$$

- Center of sum:-

- $(x)^* = \frac{(\int x \sum_{i=1}^n \mu C_i(x) x \cdot dx)}{(\int x \sum_{i=1}^n \mu C_i(x) dx)}$

- Area of trapezoidal-1:

A1=1st half of Δ1

$$\Delta 1 = (1/2) * 1 * 0.7 = 0.35$$

$$\Delta 2 = (1/2) * 1 * 0.7 = 0.35$$

$$\square \Rightarrow 2 * 0.7 = 1.4$$

$$\text{Total} \Rightarrow 0.35 + 0.35 + 1.4 \Rightarrow 2.1$$

$$A1 = 2.1$$

- Area of Trapezium-2:
- $\Delta 1 = (1/2) * 1 * 1 = 0.5$
- $\Delta 2 = 0.5$
- $\square = 2 * 1$
- Area = 3
- $A2 = 3$
- $\Rightarrow \{(2.1 * 2) + (3 * 4)\} / A1 + A2$
- $\Rightarrow \{(2.1 * 2) + (3 * 4)\} / 2.1 + 3$
- $\Rightarrow \{4.2 + 12\} / 5.1$
- $\Rightarrow 16.2 / 5.1 = 3.17$
- Centre of Largest Area
- Among this 2 trapezodial , the area of the 2nd is larger than the 1st so for this the de-fuzzified o/p will be implemented upon the 2nd area.

$$\int_2^3 \left(\frac{1}{2}x + 1 \right) x dx + \int_{-5}^6 2 \cdot x dx + \int_3^5 (0.5) x dx$$

$$\int_2^3 (0.5) dx + \int_3^5 (0.5) dx + \int_5^6 (2) dx$$

$$\Rightarrow 0.5 \left[\frac{x^2}{2} \right]_2^3 + 2 \left[\frac{x^2}{2} \right]_3^5 + 0.5 \left[\frac{x^2}{2} \right]_5^6$$

$$0.5 \left[x \right]_2^3 + 0.5 \left[x \right]_3^5 + 2 \left[x \right]_5^6$$

$$\Rightarrow 0.5 \left[\frac{9-4}{2} \right] + 2 \left[\frac{25-9}{2} \right] + 0.5 \left[\frac{36-25}{2} \right]$$

$$0.5 + 2 \times 2 \times 0.5$$

$$\Rightarrow \{(0.5 \times 2.5) + (2 \times 8) + (0.5 \times 5.5)\} / (0.5 + 4 + 0.5)$$

$$\Rightarrow 4$$

Weighted -Average Method

$$X^* = (0.7 \times 2) + (1 \times 4) / 0.7 + 1$$

$$= 1.4 + 4 / 1.7$$

$$= 3.17$$

Min-max Method

$$\Rightarrow (x)^* = (\sum_{i=1}^n x_i / n) = (3+5) / 2 = 4$$

Lecture-6

FUZZY PROPOSITIONS -

->Fuzzy proposition in a statement that drives a fuzzy truth value.

->Main difference between classical proposition and fuzzy propositions is in the range of their truth value

→ The proposition value for a classical proposition is either true or false but, in case of fuzzy composition the range is not confirmed but the range is 0 and 1.

->The classical proposition is otherwise known as crisp proposition

$$T(P) = \mu_A(x)$$

Where P=proposition logic

T=truth value

1. Conjunction

$$T(P) = \mu_A(x) \text{ \& } T(Q) = \mu_B(x)$$

Then $P \wedge Q$: x is A and B

$$T(P \wedge Q) = \min[T(P), T(Q)]$$

2. Disjunction

$P \vee Q$: x is A or B

$$T(P \vee Q) = \max[T(P), T(Q)]$$

3. Negation

$$T(P^c) = 1 - T(P)$$

4. Implication

$P \rightarrow C$: x is A then x is B

$$\Rightarrow T(P \rightarrow) = T(P^c \vee Q) = \max[T(P^c); T(Q)]$$

Fuzzy Inference: - It is the combination of set of rules which will be used to solve the problems related to the fuzzy logic by combining the number of fuzzy sets which creates a fuzzy relation "R"

1. IF-THEN statement
2. IF-THEN-ELSE statement

a. IF THEN statement (\Rightarrow)

If 'x' is A then y is B equivalent to

$R = (A * B) \cup (\bar{A} * Y)$ where $\bar{A} \rightarrow$ complement

$\Rightarrow \mu_R(x, y) = \max(\min[\mu_A(x), \mu_B(y)] \min([1 - \mu_A(x)], U)$

Lecture-7

$$Q \rightarrow X = \{a, b, c, d\}, Y = \{1, 2, 3, 4\}$$

$$A = \{(a, 0), (b, 0.8), (c, 0.6), (d, 1)\}$$

$$B = \{(1, 0.2), (2, 1), (3, 0.8), (4, 0)\}$$

Prove the 1st implication inference.

$$A \times B = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.2 & 0.8 & 0.8 & 0 \\ 0.2 & 0.6 & 0.6 & 0 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix} \end{matrix}$$

$$\mu_R(a, 1) = \min[\mu_{\bar{A}}(a), \mu_B(1)]$$

$$= \min(0, 0.2)$$

$$= 0$$

$$\mu_R(a, 2) = \min(0, 1) = 0 \quad \mu_R(d, 1) = 0.2$$

$$\mu_R(a, 3) = 0$$

$$\mu_R(d, 2) = 1$$

$$\mu_R(a, 4) = 0$$

$$\mu_R(d, 3) = 0.8$$

$$\mu_R(b, 1) = \min(0.8, 0.2) = 0.2 \quad \mu_R(d, 4) = 0$$

$$\mu_R(b, 2) = \min(0.8, 1) = 0.8$$

$$\mu_R(b, 3) = \min(0.8, 0.8) = 0.8$$

$$\mu_R(b, 4) = \min(0.8, 0) = 0$$

$$\mu_R(c, 1) = 0.2$$

$$\mu_R(c, 2) = 0.6$$

$$\mu_R(c, 3) = 0.6$$

$$\mu_R(c, 4) = 0$$

1)}
0)}.

def:

$$Y = \{(1,1), (2,1), (3,1), (4,1)\}$$

$$\bar{A} = \{(a,1), (b,0.2), (c,0.4), (d,0)\}$$

$$\bar{A} \times Y = \begin{matrix} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0.2 & 0.2 & 0.2 & 0.2 \\ 0.4 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{matrix}$$

$$\mu_R(1,a) = \min(1,1) = 1$$

$$\mu_R(1,b) = 1$$

$$\mu_R(1,c) = 1$$

$$\mu_R(1,d) = 1$$

$$\mu_R(2,a) = 0.2$$

$$\mu_R(2,b) = 0.2$$

$$\mu_R(2,c) = 0.2$$

$$\mu_R(2,d) = 0.2$$

$$\mu_R(3,a) = 0.4$$

$$\mu_R(3,b) = 0.4$$

$$\mu_R(3,c) = 0.4$$

$$\mu_R(3,d) = 0.4$$

$$\mu_R(4,a) = 0$$

$$\mu_R(4,b) = 0$$

$$\mu_R(4,c) = 0$$

$$\mu_R(4,d) = 0$$

$$\mu_R(x,y) = \max \begin{pmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.2 & 0.8 & 0.8 & 0 \\ 0.2 & 0.6 & 0.6 & 0 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix} \end{pmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0.2 & 0.2 & 0.2 & 0.2 \\ 0.4 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0.2 & 0.8 & 0.8 & 0.2 \\ 0.4 & 0.6 & 0.6 & 0 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0.2 & 0.8 & 0.8 & 0.2 \\ 0.4 & 0.6 & 0.6 & 0 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix}$$

Q. If x is A then y is B ELSE y is C .

$$R = (A \times B) \cup (\bar{A} \times C).$$

$$\mu_R(x, y) = \max \left\{ \min [\mu_A(x), \mu_B(y)], \min [1 - \mu_A(x), \mu_C(y)] \right\}$$

$$C = \{(1, 0), (2, 0.4), (3, 1), (4, 0.8)\}.$$

Ans: $\bar{A} = \{(a, 1), (b, 0.2), (c, 0.4), (d, 0)\}.$

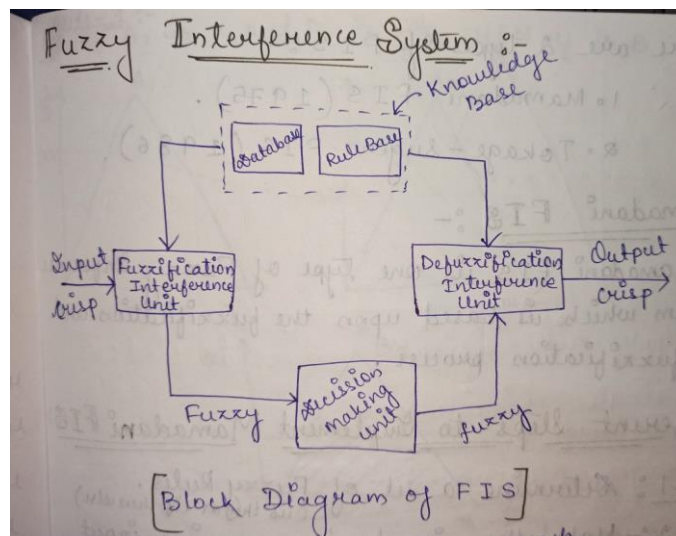
$$A \times B = \begin{matrix} a \\ b \\ c \\ d \end{matrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.2 & 0.8 & 0.8 & 0 \\ 0.2 & 0.6 & 0.6 & 0 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix}$$

$$\bar{A} \times C = \begin{matrix} a \\ b \\ c \\ d \end{matrix} \begin{bmatrix} 0 & 0.4 & 1 & 0.8 \\ 0 & 0.2 & 0.2 & 0.2 \\ 0 & 0.4 & 0.4 & 0.4 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R = \begin{matrix} a \\ b \\ c \\ d \end{matrix} \begin{bmatrix} 0 & 0.4 & 1 & 0.8 \\ 0.2 & 0.8 & 0.8 & 0.2 \\ 0.2 & 0.6 & 0.6 & 0.4 \\ 0.2 & 1 & 0.8 & 0 \end{bmatrix} \quad (\text{Ans})$$

Fuzzy Interface System

- ## 2.Takage-Sugena FIS



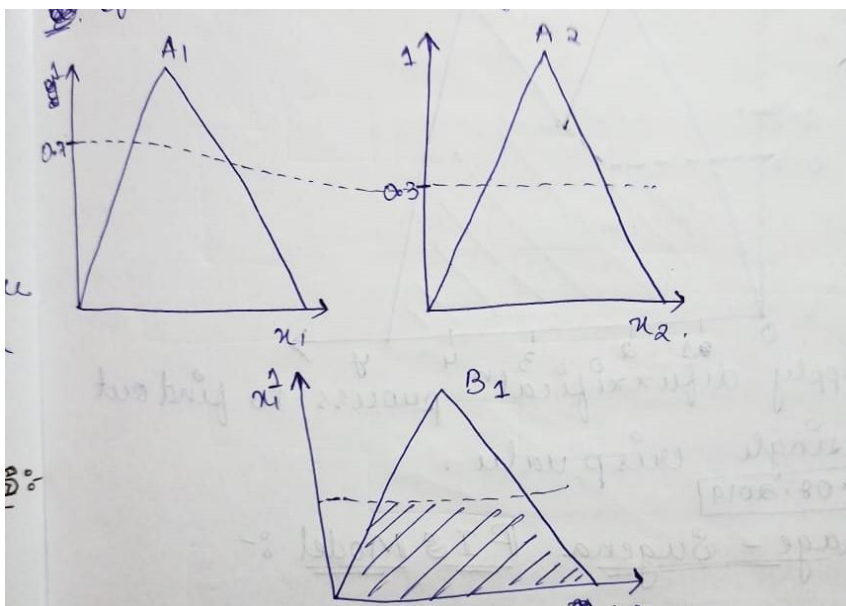
Mamadani FIS

- Mamadani FIS is one of the type of fuzzy Interface system which is based upon the fuzzification and defuzzification process.

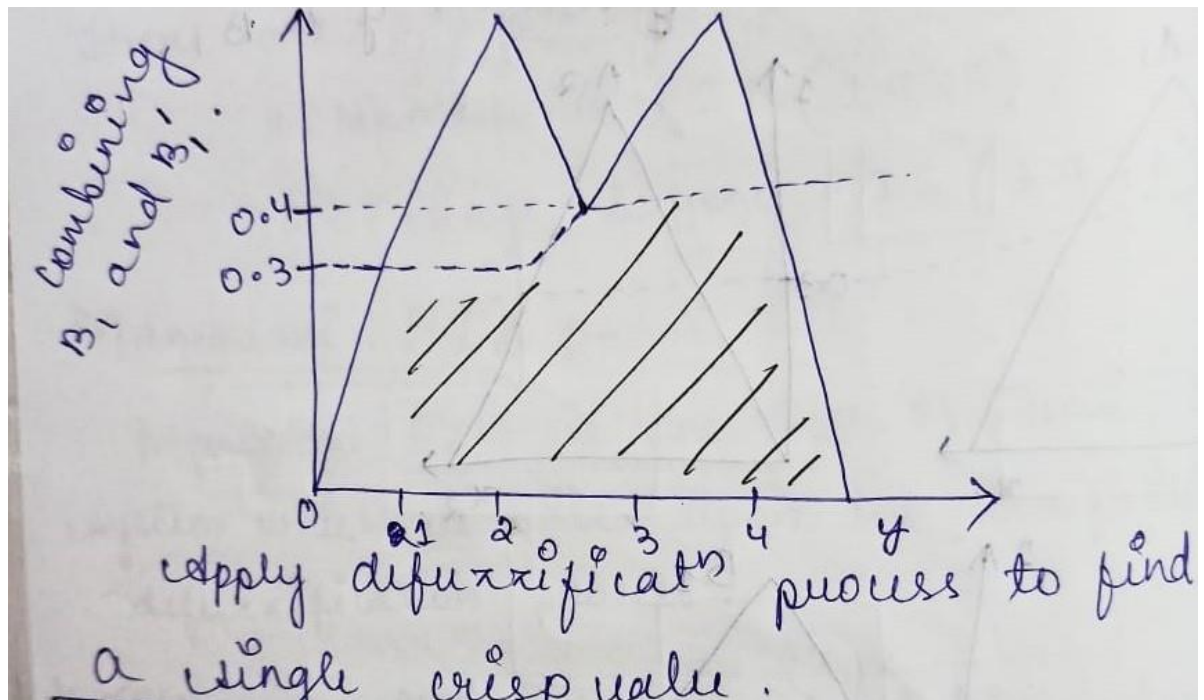
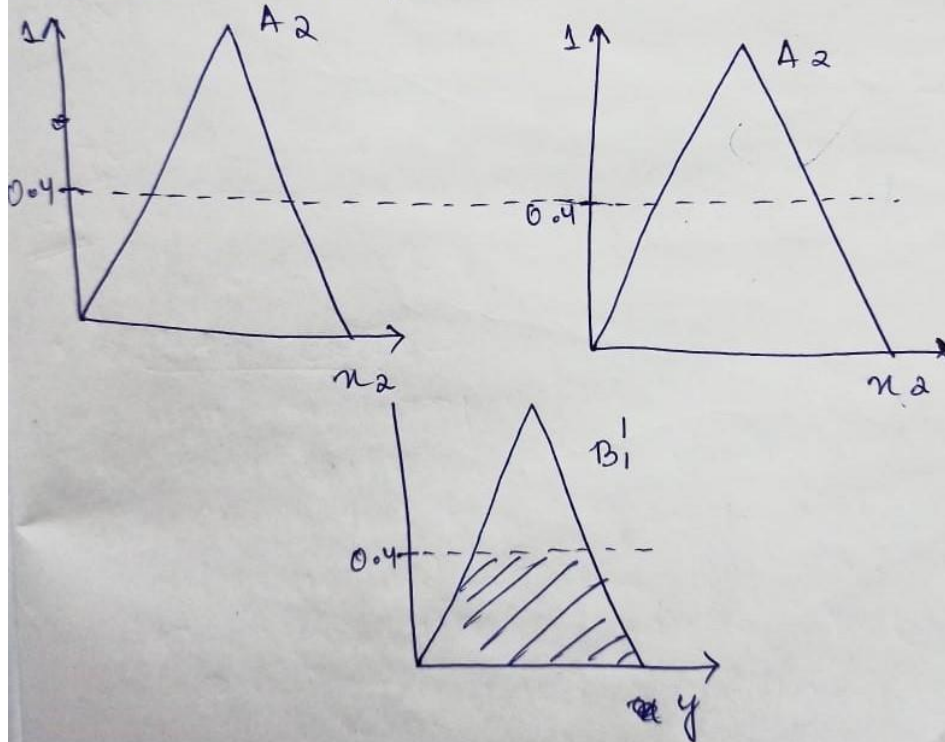
Steps to implement Mamadani FIS

- Step-1-Determine a set of Fuzy rules.
- Step-2-Make the inputs fuzzy using input membership functions.
- Step-3-Combine the fuzzzified inputs according to the fuzzy rules for establishing a rule strength.
- Step-4-Determine the consequent (resultant) of the rule by combining the rule strength and output membership function.
- Step-5-Combine all the consiquents to get all the output distribution.
- Step-6-Finally a defuzzified output distribution is obtained.

Rule-1- If A1 is x_1 and A2 is x_2 THEN y is B1,



Rule-2 :- If x_1 is A_1 and x_2 is A_2 THEN y is B



Takage-Sugena FIS Model

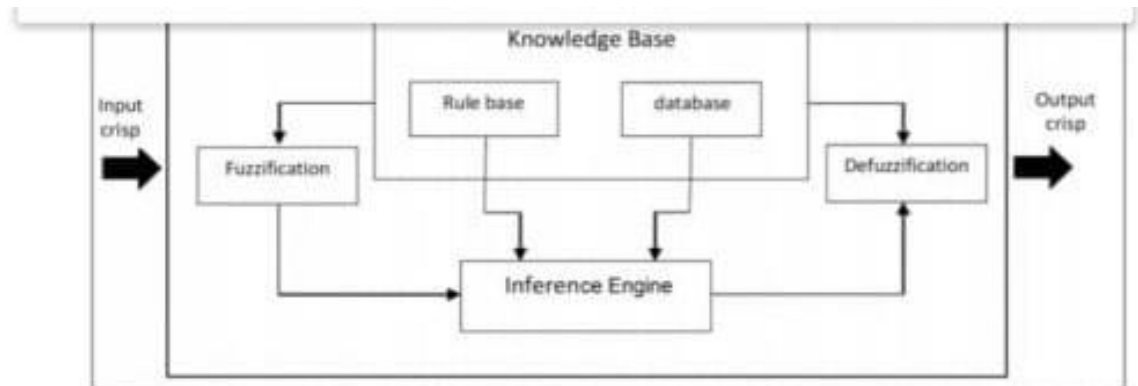


Figure 1. Architecture of a fuzzy expert system.

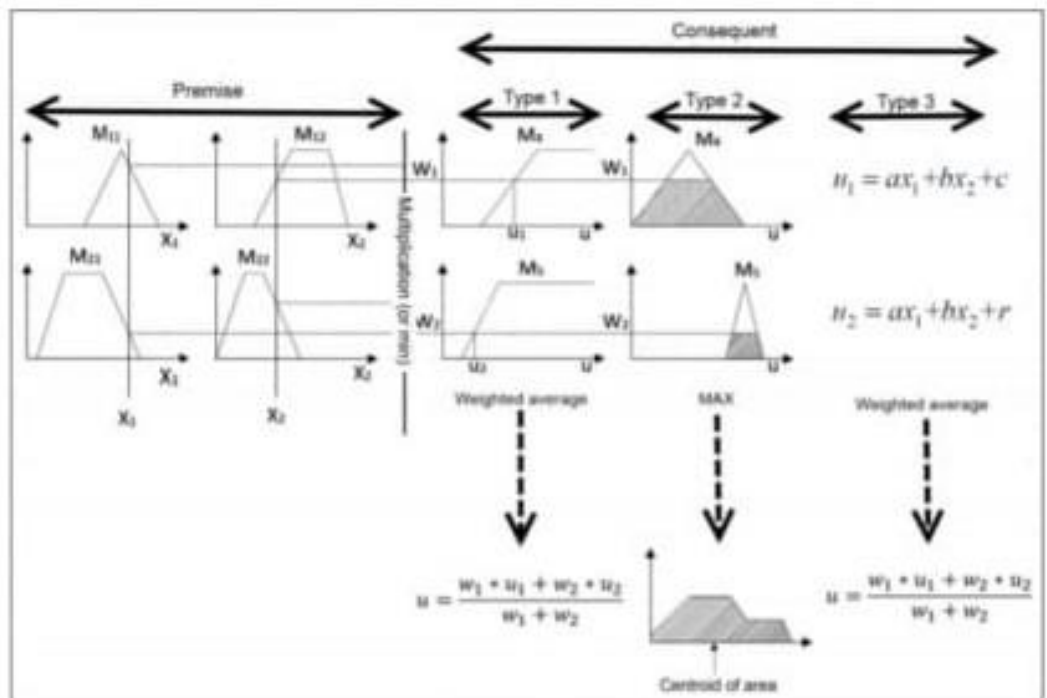


Figure 2. Different types of Fuzzy inference system [30].

MODULE-3

NEURAL NETWORK

- Neural networks are those information processing system which are constructed and implemented to model the human brain.
- The main objective of the neural network is to develop a computational device for modelling the brain to perform various computational task at a faster rate than the traditional system.
- An artificial neural network is a computational model based on the structure and functions of biological neural network consists of neurons.

NEURAL NETWORK ARCHITECTURE

- An artificial neural network (ANN) is defined as a data processing system consisting of a large no. of simple ,highly interconnected processing elements(i.e,known as artificial neurons)
- ANN can be represented by using a directed graph.
- The graph G consists of a V and E i.e;

V for set of vertices

E for set of Edges

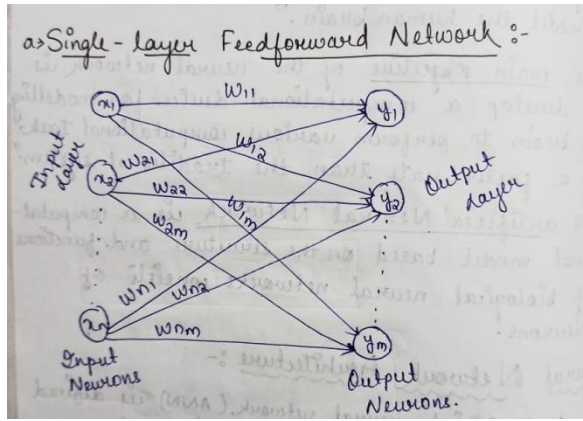
So, our neural network Architecture is divided into 3 types-

1. Single layer Feed Forward Network
2. Multi-layered feed Forward Network
3. Recurrent Network

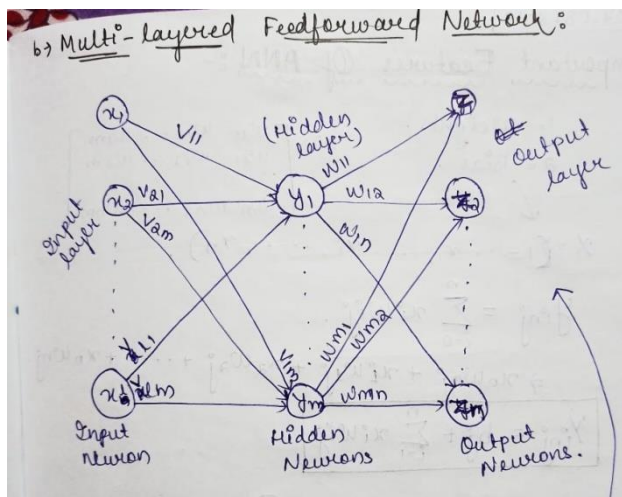
Single layer Feedforward Network

- As in the input layer the information is being grasped for processing means it only accepts all the information but in the output layer all the information will be processed simultaneously the action/the output will also be generated.
- So, as the processing and the output declaration is being handled in the output declaration is being handled in the output layer so to reduce the burden the multilayer feedforward network has been discovered.

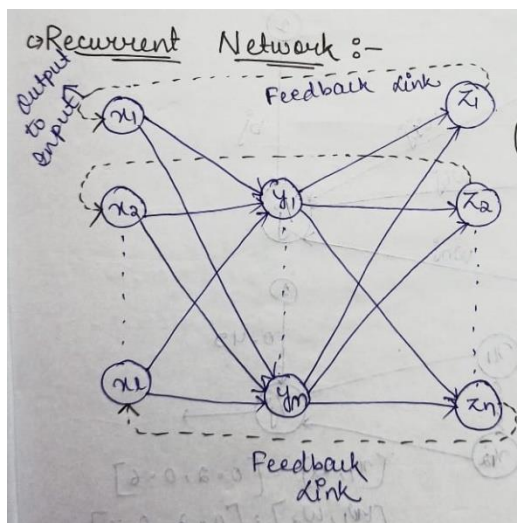
Single layer Feedforward Network



Multi-layered Feed Forward Network



Recurrent network



IMPORTANT FEATURES OF ANN

- 1. Weights
- 2. Bias

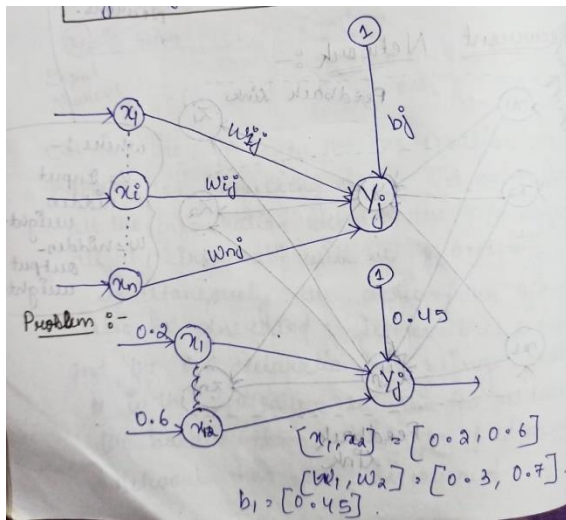
$$W = \begin{matrix} & \begin{matrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \vdots & \vdots & & \vdots \\ w_{n1} & w_{n2} & \dots & w_{nm} \end{matrix} \end{matrix}$$

$$X = (1, \dots, x_1, \dots, x_i, \dots, x_n)$$

$$Y_{in_j} = \sum_{i=0}^n x_i W_{ij}$$

$$\Rightarrow x_0 W_{0j} + x_1 W_{1j} + x_2 W_{2j} + \dots + x_n W_{nj}$$

$$Y_{in_j} = b_j + \sum_{i=1}^n x_i W_{ij}$$



$$[x_1, x_2] = [0.2, 0.6] \quad [w_1, w_2] = [0.3, 0.7] \quad b_1 = [0.45]$$

Calculate the net output Y.

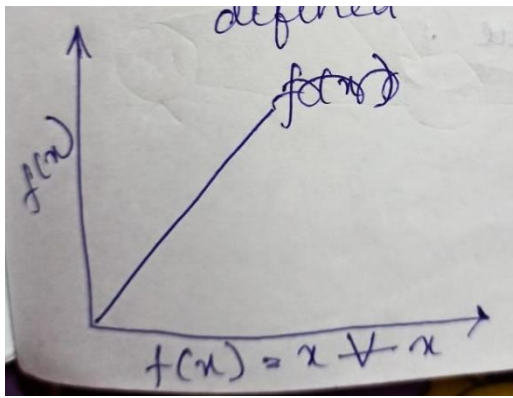
- Ans- $Y_{in} = 0.45 + (0.2 \times 0.3) + (0.6 \times 0.7)$
 $= 0.93$

ACTIVATION FUNCTION

- To make more efficient and to obtain exact output some force (push) or activation may be given.
- This activation helps in achieving the exact output.
- In similar way there are some activation function are there which will be applied over the net input to calculate the net output of the ANN.
- There are 5 types of activation functions such as –
 1. Identity function
 2. Binary Step Function
 3. Bipolar step function
 4. Sigmoidal Function
 5. Ramp function

1.IDENTITY FUNCTION

It is a linear function and can be defined as

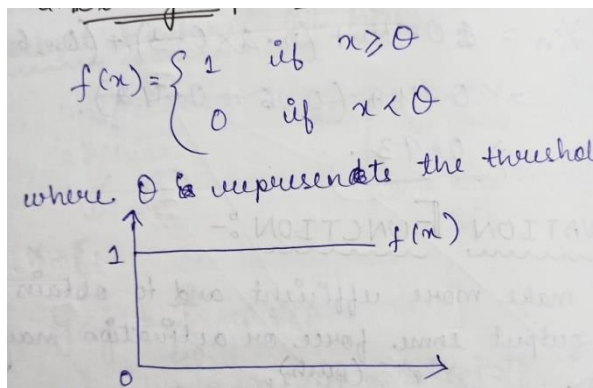


2.BINARY STEP FUNCTION

- $f(x) = \begin{cases} 1 & \text{if } x \geq \Theta \\ 0 & \text{if } x < \Theta \end{cases}$

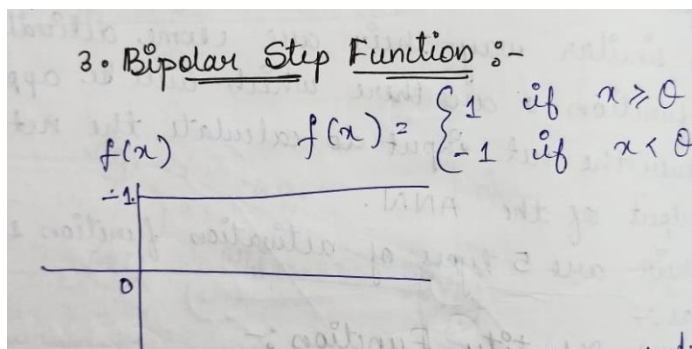
Where Θ represents the threshold value

This function is mostly used in single layered feed-forward network.



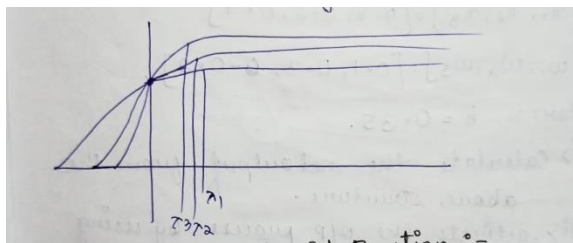
3. BIPOLAR STEP FUNCTION

- $f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ -1 & \text{if } x < \theta \end{cases}$
- It is also used for single layer feed forward network where θ represents the threshold value.



4. SIGMOIDAL FUNCTION

1. Binary Sigmoid function



$$f(x) = 1/(1+e^{-\lambda x})$$

Where λ = steepness parameter

Range of binary sigmoid function is from 0 to 1.

2. Bipolar sigmoid function

(ii) Bipolar Sigmoid Function :-

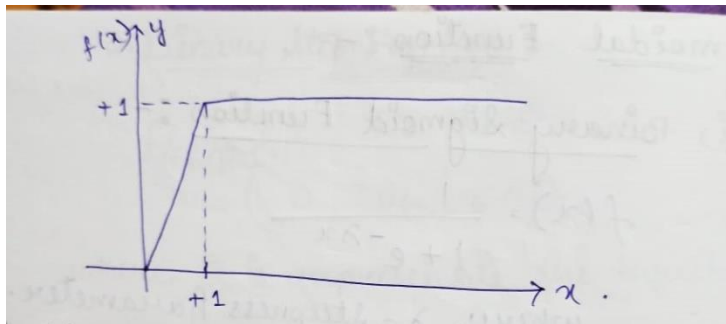
$$f(x) = \frac{2}{1 - e^{-\lambda x}} - 1$$

$$f(x) = \frac{1 - e^{-\lambda x}}{1 + e^{-\lambda x}}$$

$$f(x) = (1 - (e^{-\lambda x})) / (1 + (e^{-\lambda x}))$$

5. RAMP FUNCTION

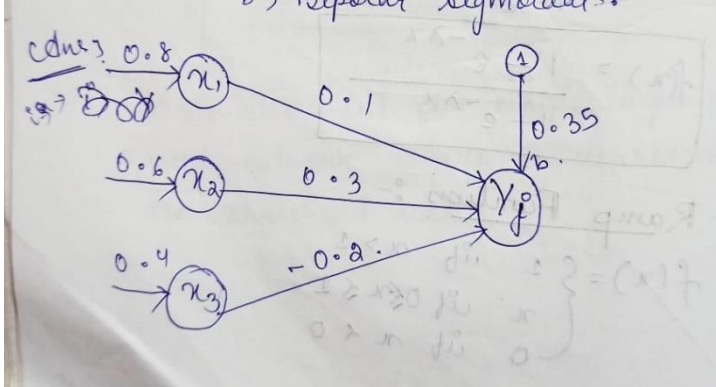
- $f(x) = \begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } 0 \leq x \leq 1 \\ 0 & \text{if } x < 0 \end{cases}$



PROBLEM 1

- $[x_1, x_2, x_3] = [0.8, 0.6, 0.4]$
- $[W_1, W_2, W_3] = [0.1, 0.3, -0.2]$
- Where $b = 0.35$
- 1. Calculate the net output from the above structure
- 2. Activate the output process by using activation method
- (A) Binary Sigmoid

- (B) Bipolar Sigmoidal



$$(1) Y_{in} = 0.35 + (0.8 * 0.1) + (0.6 * 0.3) + (0.4 * -0.2)$$

$$= 0.35 + 0.18 = 0.53$$

(2(A)) Binary Sigmoidal

$$f(x) = 1 / (1 + (e^{-\lambda x}))$$

$$\Rightarrow f(x) = 1 / (1 + (e^{-0.53})) = 0.629 = 0.63$$

(2(B)) Bipolar Sigmoidal

$$f(x) = (1 - (e^{-\lambda x})) / (1 + (e^{-\lambda x}))$$

$$= (1 - (e^{-0.53})) / (1 + (e^{-0.53}))$$

$$= 0.258$$

PROBLEM 2(vey important)

- $[x_1, x_2, x_3, x_4] = [0.5, 0.9, 0.2, 0.3]$
- $[W_1, W_2, W_3, W_4] = [0.2, 0.3, -0.6, -0.1]$
- $Y = Y_{in1} + Y_{in2}$
- $\{Y_{in1} = 0.5$ - calculate the net output
- $\{Y_{in2} = 0.9$ - Activation- Binary
- - Bipolar
- $[x_1, x_2, x_3, x_4] = [0.5, 0.9, 0.2, 0.3]$

- $[W_1, W_2, W_3, W_4] = [0.2, 0.3, -0.6, -0.1]$
- $b_1 = 0.5$
- $b_2 = 0.9$

For Y_1 : $Y_{in1} = 0.5 + ((0.5 * 0.2) + (0.9 * 0.3) + (0.2 * -0.6) + (0.3 * -0.1))$

$$= 0.5 + 0.1 + 0.27 - 0.12 - 0.03$$

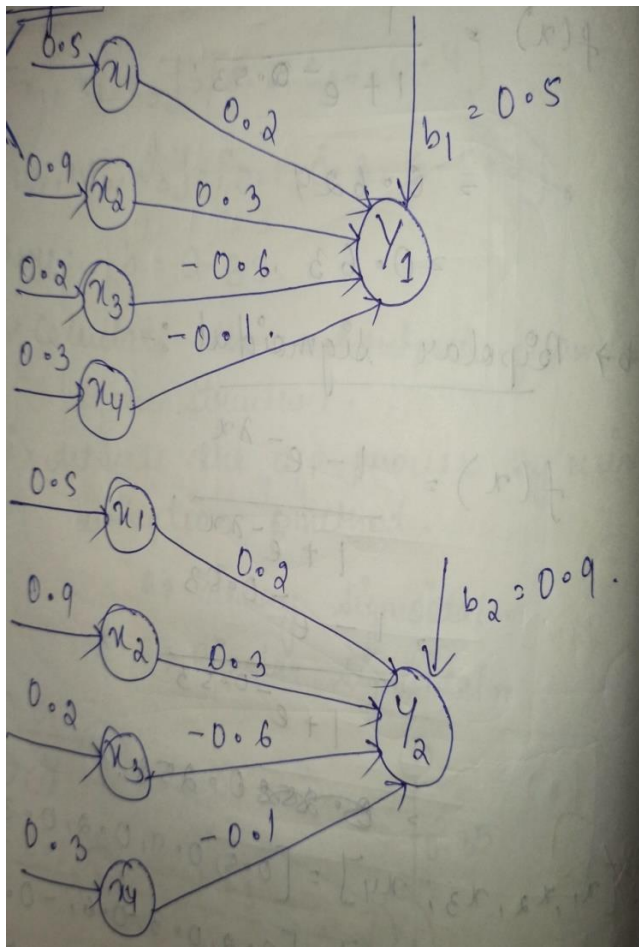
$$= 0.72$$

$Y_{in2} = 0.9 + ((0.5 * 0.2) + (0.9 * 0.3) + (0.2 * (-0.6)) + (0.3 * (-0.1)))$

$$= 0.9 + 0.1 + 0.27 - 0.12 - 0.03$$

$$= 1.12$$

$$Y = Y_{in1} + Y_{in2} = 0.72 + 1.12 = 1.84$$



- Binary Sigmoidal

- $f(x) = 1/(1+(e^{-\lambda x}))$

- $f(x) = 1/(1+(e^{-1.84}))$

$$= 0.8629$$

- Bipolar Sigmoidal

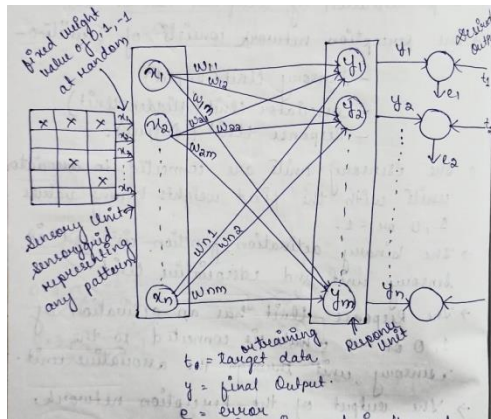
- $f(x) = (1-(e^{-\lambda x}))/ (1+(e^{-\lambda x}))$

- $f(x) = (1-(e^{-1.84}))/ (1+(e^{-1.84}))$

$$= 0.7258 = 0.726$$

PERCEPTION NETWORK (Implementation of neural network)

- Perceptual Network comes under single layer feedforward network which is one of the implementation of artificial neural network.
- The perception network consists of 3 units-
 - -Sensory unit(Input)
 - -Associator unit(Hidden unit)
 - -Response Unit(Output)
- The sensory units are connected to associator units with the fixed weights having values 1,0 or -1.
- The binary activation function is used in sensory unit and associative unit.
- The response unit has an activation of 1,0 or -1 which is connected to the sensory unit through the associative unit.
- The output of the perception network is given by-
 - $Y = f(y_{in})$
 - $f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \Theta \\ 0 & \text{if } -\Theta < y_{in} < \Theta \\ -1 & \text{if } y_{in} < -\Theta \end{cases}$ Where Θ is the threshold value

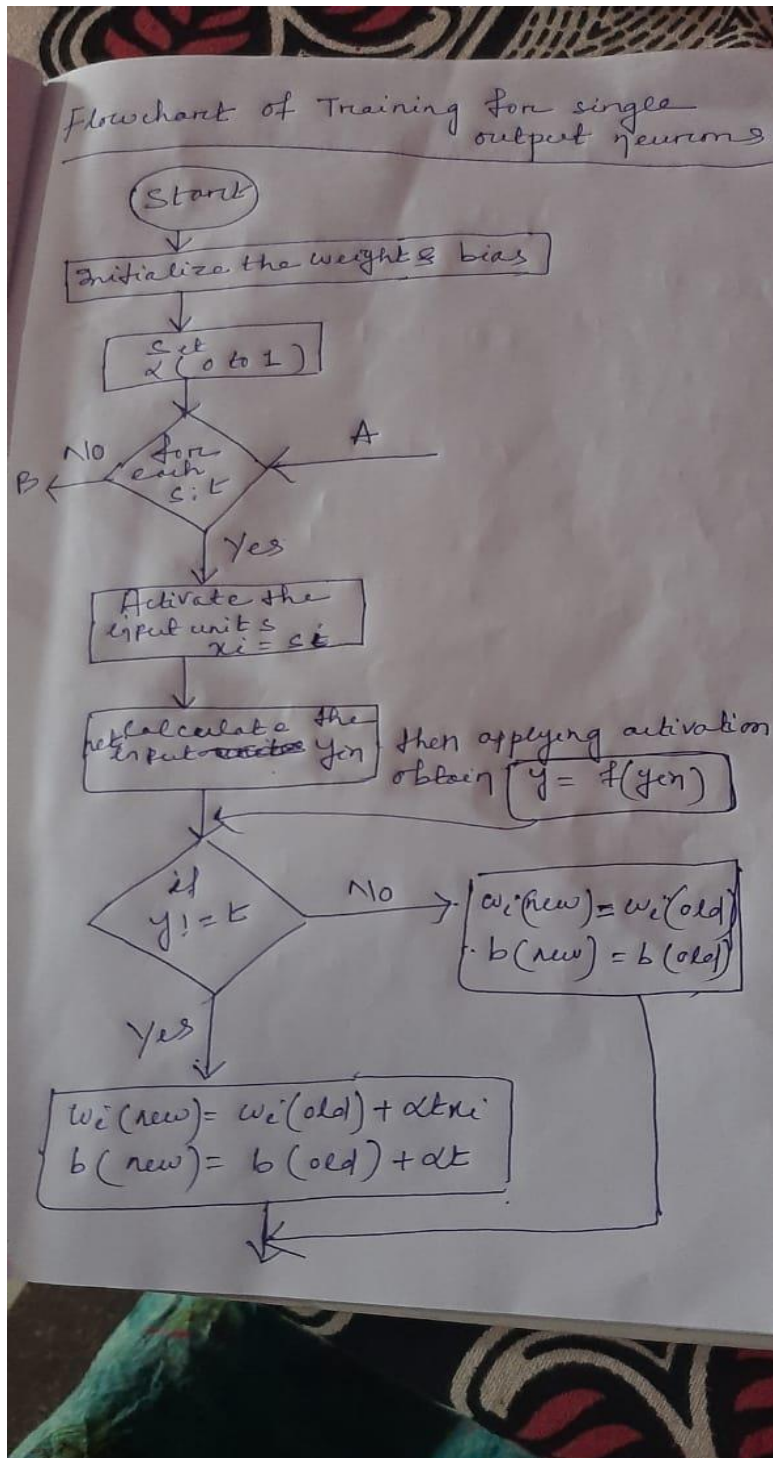


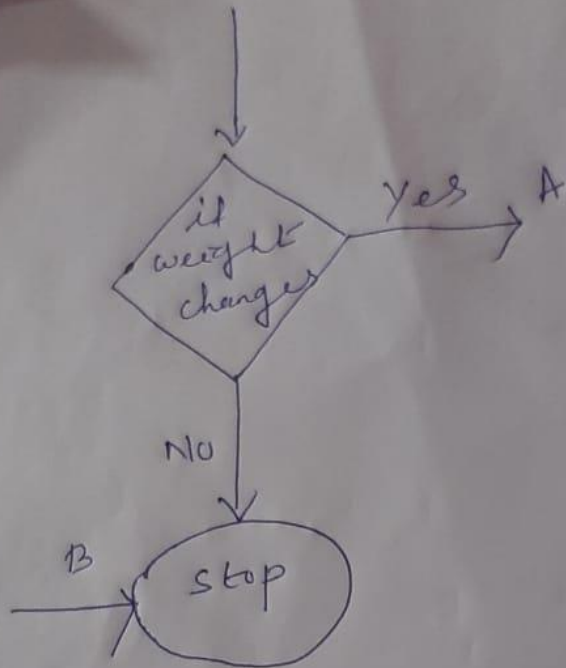
- The perceptron learning is used in the net updation between the associator unit and the response unit.
- For each training input, the net will calculate the response and it will determine whether or not an error has occurred.
- The error calculation is based on the comparison of the target values (training data) with the calculated output.
- The weights will be adjusted on the basis of the learning rule if an error has occurred for a particular training pattern.
- $W_i(\text{new}) = W_i(\text{old}) + \alpha t x_i$
- $b_i(\text{new}) = b_i(\text{old}) + \alpha t$

where α = learning rate

- Perceptron network can be of the single class means only 1 single output neurons and the perceptron network can be of multiple class means many outputs neurons will be compromise of a single y .

FLOW CHART OF TRAINING PROCESS FOR SINGLE OUTPUT NEURONS



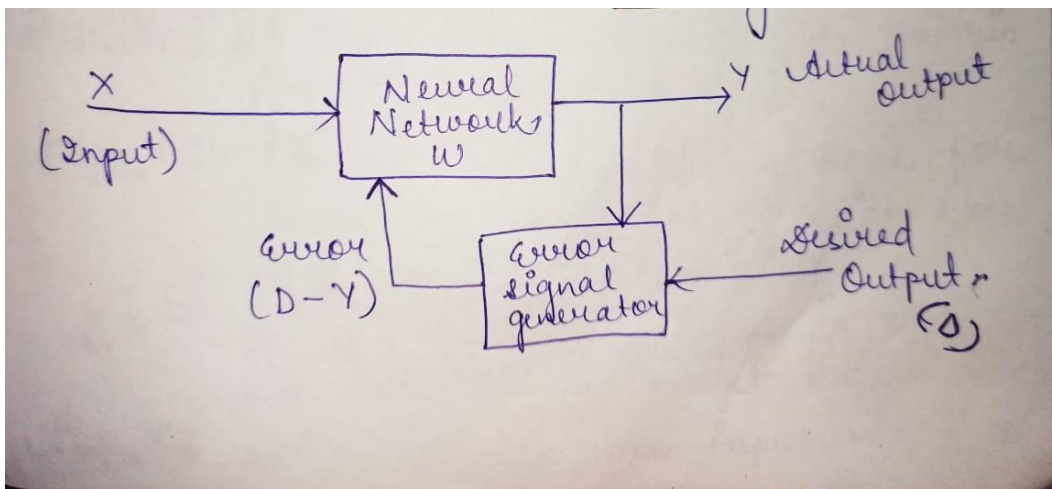


Learning process in the neural network

- The main property of a neural network is the capability to learn.
- Learning-Learning /training is a process by means of which a neural network adapts itself to stimulate by making proper parameter adjustments.
- There are 3 types of learning
 1. Supervised learning
 2. Unsupervised learning
 3. Perception /Reinforcement learning

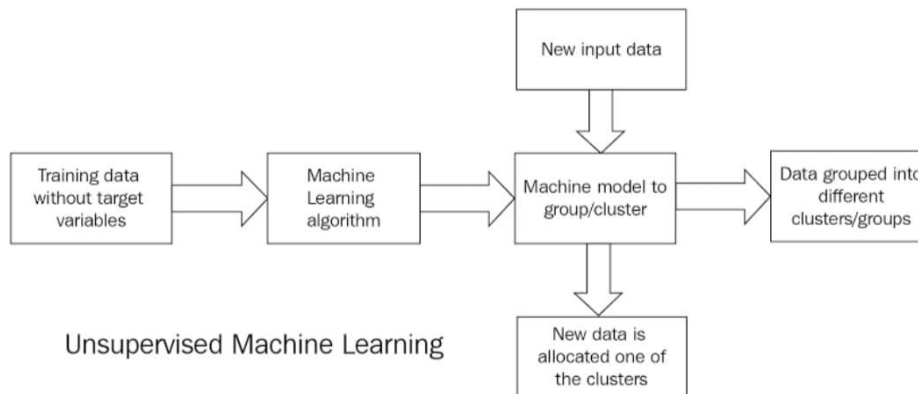
Supervised Learning

- In supervised learning each input vector requires a corresponding target vector which represents the divided output.
- The input vector along with the target vector is called the training pair.
- In this type of learning process a supervisor is required for error submission.
- In supervised learning it is assumed(predicted) that the correct target values are known for each input pattern.



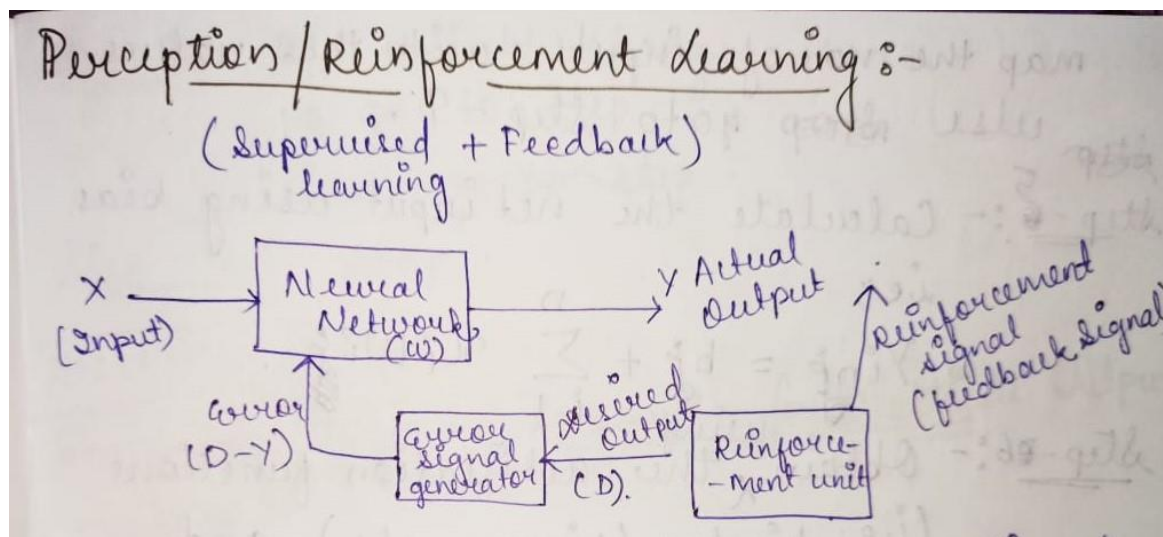
Unsupervised Learning

- In unsupervised learning, the input vector requires its corresponding vector which represents the desired output.
- The input vector along with the target value is called Training pair.
- Here, no supervisor required for the error minimization.



Perception/Reinforcement Learning

- In this type of learning, after getting the actual output, the output neuron will send a feedback to its respective input signal that the work which you have assigned it has been completed.



Perception training algorithm for single output class

- Step-1- x_i =each possible inputs

s =set of inputs

Start the perception process.

- Step-2-Initialize the set of inputs i.e;set of weight(w) with its respective bias(b).
- Step-3-Input the set the learning rate(α) within its limit from 0 to 1.
- Step-4-Mapping is done for each set of input(s) with each set of target value. If all the set of input is mapped with the set of target value then activate the input units i.e; map the number of inputs with the values else go to step-9.
- Step-5-Calculate the net input using bias i.e;
- $$Y_{in} = b + \sum_{i=1}^n x_i w_i$$
- Step-6-Obtain using the activation functions (i.e;bipolar /binary etc) and obtain the desired output.
- $y = f(y_{in})$
 $= \{ 1 \text{ if } y_{in} > \theta$
- $f(y_{in}) = \{ 0 \text{ if } -\theta < y_{in} < \theta$
 $= \{-1 \text{ if } y_{in} < -\theta$
- Step-7- If $y \neq t$, then

$$W_i(\text{new}) = W_i(\text{old}) + \alpha t x_i$$

$$b(\text{new}) = b(\text{old}) + \alpha t$$

else

$$W_i(\text{new}) = W_i(\text{old})$$

$$b_i(\text{new}) = b_i(\text{old})$$

- Step-8- If weight changes then go to step-4 else
- Step-9- STOP

Adaline Network

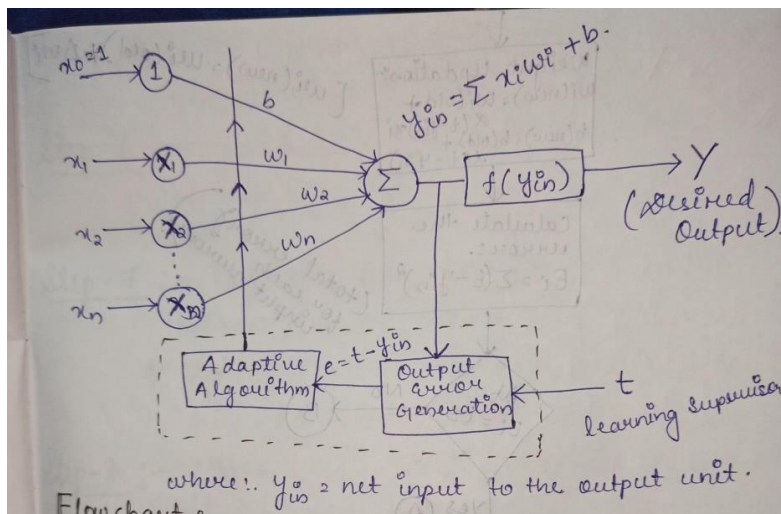
- A network with a single linear unit is called as Adaline.
- In Adaline network, the input output relationship is linear.
- Adaline uses bipolar activation function for its input signals to reach at its target output.
- The weights between the inputs and outputs are adjustable.
- Adaline is a network which have one output unit.
- The adaline network may be trained by using Δ -rule.
- The Δ -rule updates the weights between the connection so as to minimize the difference between the net input to the output unit and the target value.
- The Δ rule for adjusting the weights of i th pattern means $i=1$ to n

i.e; $\Delta W_i = \alpha(t - y_{in})x_i$

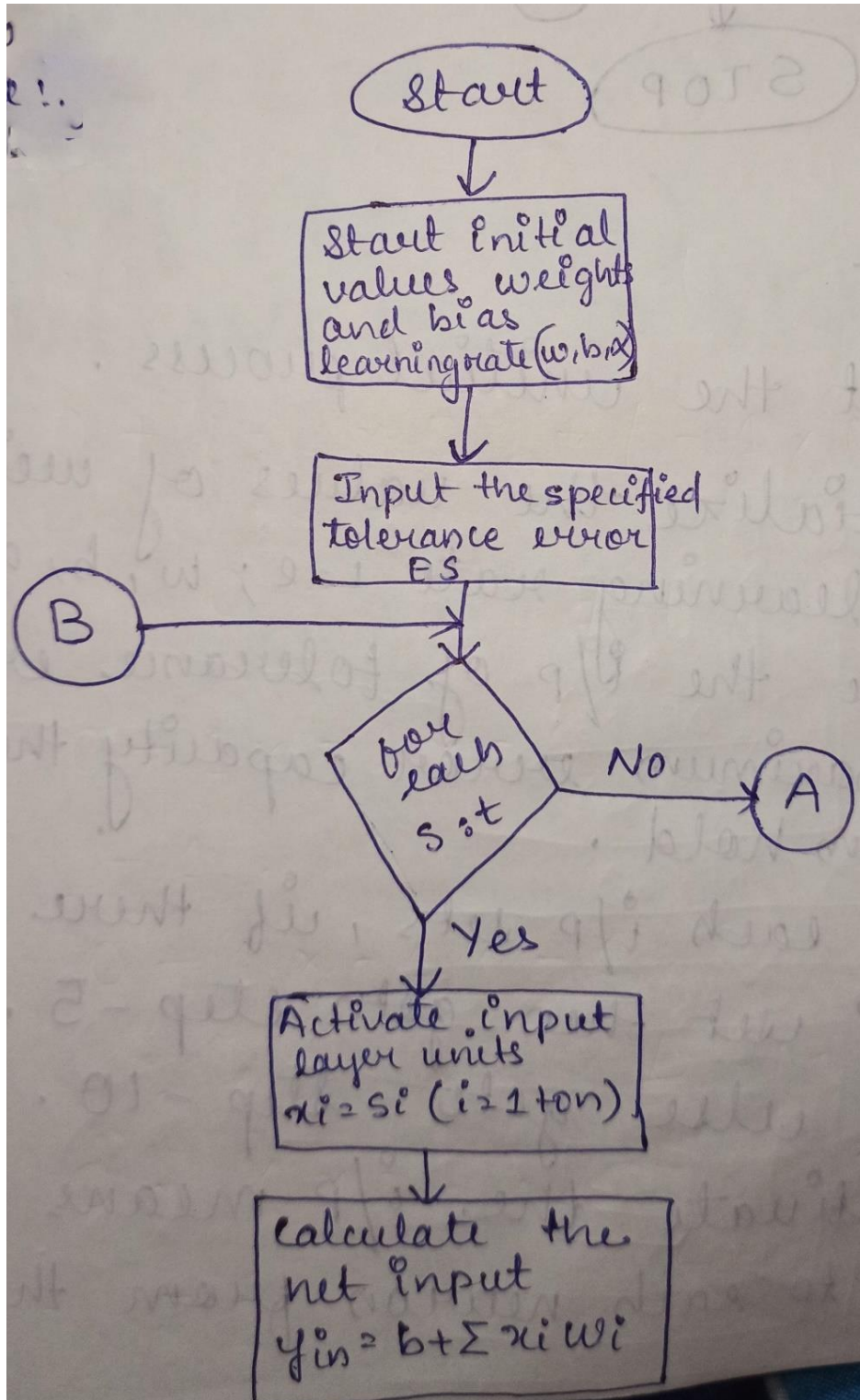
Where $\Delta W_i \rightarrow$ weight change , $\alpha \rightarrow$ learning rate , $x_i \rightarrow$ vector of activation of input unit , $y_{in} \rightarrow$ net input to the output unit

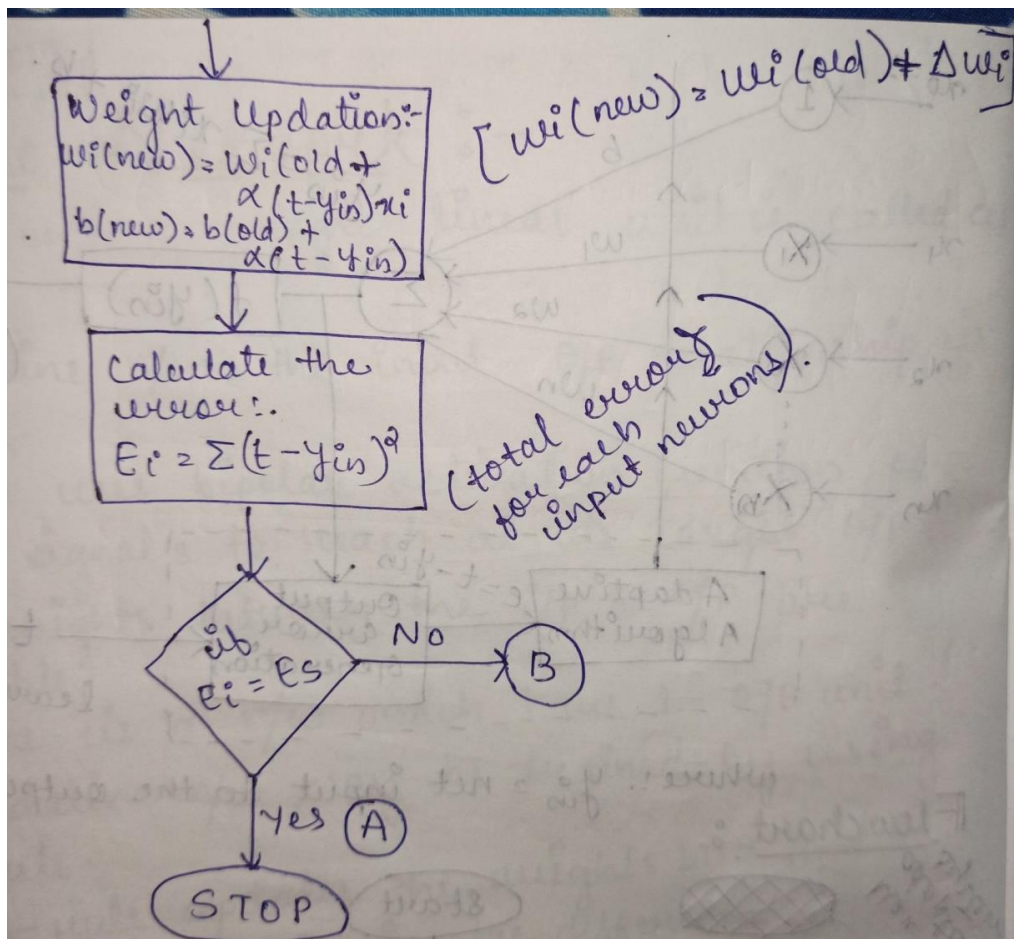
$$Y = \sum_{i=1}^n x_i w_i + b \quad \text{-----} \rightarrow \text{target output}$$

- The Δ - rule in case of several output unit for adjusting the weights from i th input units j th output unit for each pattern.
- $\Delta w_{ij} = \alpha(t_j - y_{in j})x_i$



Flow chart





Algorithm

- Step-1:-Start the initial process.
- Step-2:-Initialize the values of weight, bias and learning rate i.e;w,b,α respectively.
- Step-3:-Give the output of tolerance error (E)i.e; the maximum error capacity that the system hold.
- Step-4:-For each input sets , if there is matching output set then goto step-5.

Else goto step-10

- Step-5:-Activate the input means gives the input to each neuron from the input set , i.e; from x_i to s_i

where $i \rightarrow 1$ to n

- Step-6:-Calculate the net input i.e;

$$Y = \sum_{i=1}^n x_i w_i + b$$

- Step-7:- Update the weights and bias as:-

$$W_i(\text{new}) = W_i(\text{old}) + \Delta w_i$$

$$\Rightarrow W_i(\text{new}) = W_i(\text{old}) + \alpha(t - y_{in})x_i$$

$$\Rightarrow b_i(\text{new}) = b_i(\text{old}) + \alpha(t - y_{in})$$

- Step-8:-Check the total error at the output unit.

$$E_i = \sum (t - y_{in})^2$$

- Step-9:-If ($E_i == E_s$)

then goto step 10 else goto step 4

- Step-10:-STOP

Perception training algorithm for multiple output

- Step-1:- Start the perception process.
- Step-2:-Initialize the set of inputs i.e;set of weights (w) with its respective bias.
- Step-3:-Input the set of learning rate(α) within its limit from 0 to 1.
- Step-4:-Mapping is done for each set of input (s) with each set of target value.
- If all the set of target value then activate the input units i.e; map the number of inputs with the values else goto step-9.
- Step-5:-Calculate the output of

the network.

$$Y_{in_j} = \sum_{i=1}^n x_i w_i + b_j$$

where $i=1$ to n , $j=1$ to m

- Step-6:-The final output using activation functions are:-

$$Y_j = f(y_{in_j})$$

$$= \{ 1 \text{ if } y_{inj} > \Theta$$

- $f(y_{in}) = \{ 0 \text{ if } -\Theta \leq y_{inj} \leq \Theta$ where $i=1 \text{ to } n, j=1 \text{ to } m$

$$= \{-1 \text{ if } y_{inj} < -\Theta$$

- Step-7:-if $t_j \neq y_j$ then;

$$W_{ij}(\text{new}) = W_{ij}(\text{old}) + \alpha t_j x_i$$

$$b_j(\text{new}) = b_j(\text{old}) + \alpha t_j$$

Else

$$W_{ij}(\text{new}) = W_{ij}(\text{old})$$

$$b_j(\text{new}) = b_j(\text{old})$$

- Step-8:-If weight changes then go to step-4
- Step-9:-STOP

Back-propagation Network(imp)

In this ,the network signals are sent in the reverse direction.

It means whatever the signals sent by the input neurons it can get its feedback in the reverse direction to inform the completion of task.

Its respective elements are :-

x =Input training vector ($x_1, x_2, \dots, x_i, \dots, x_n$)

t = target output vector($t_1, \dots, t_k, \dots, t_m$)

α =Learning rate parameter

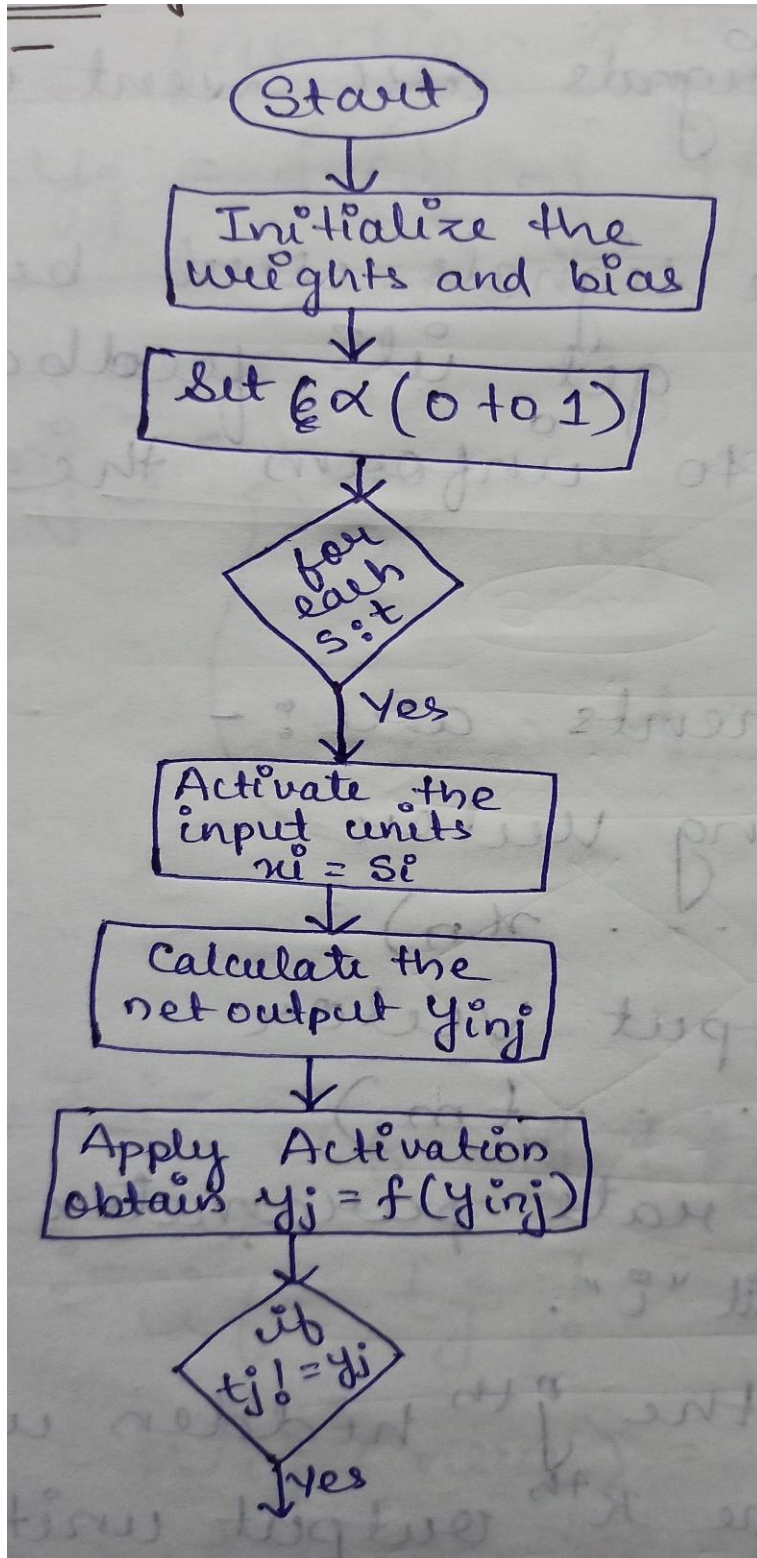
x_i = input unit "i" where $\{i=1 \text{ to } n\}$

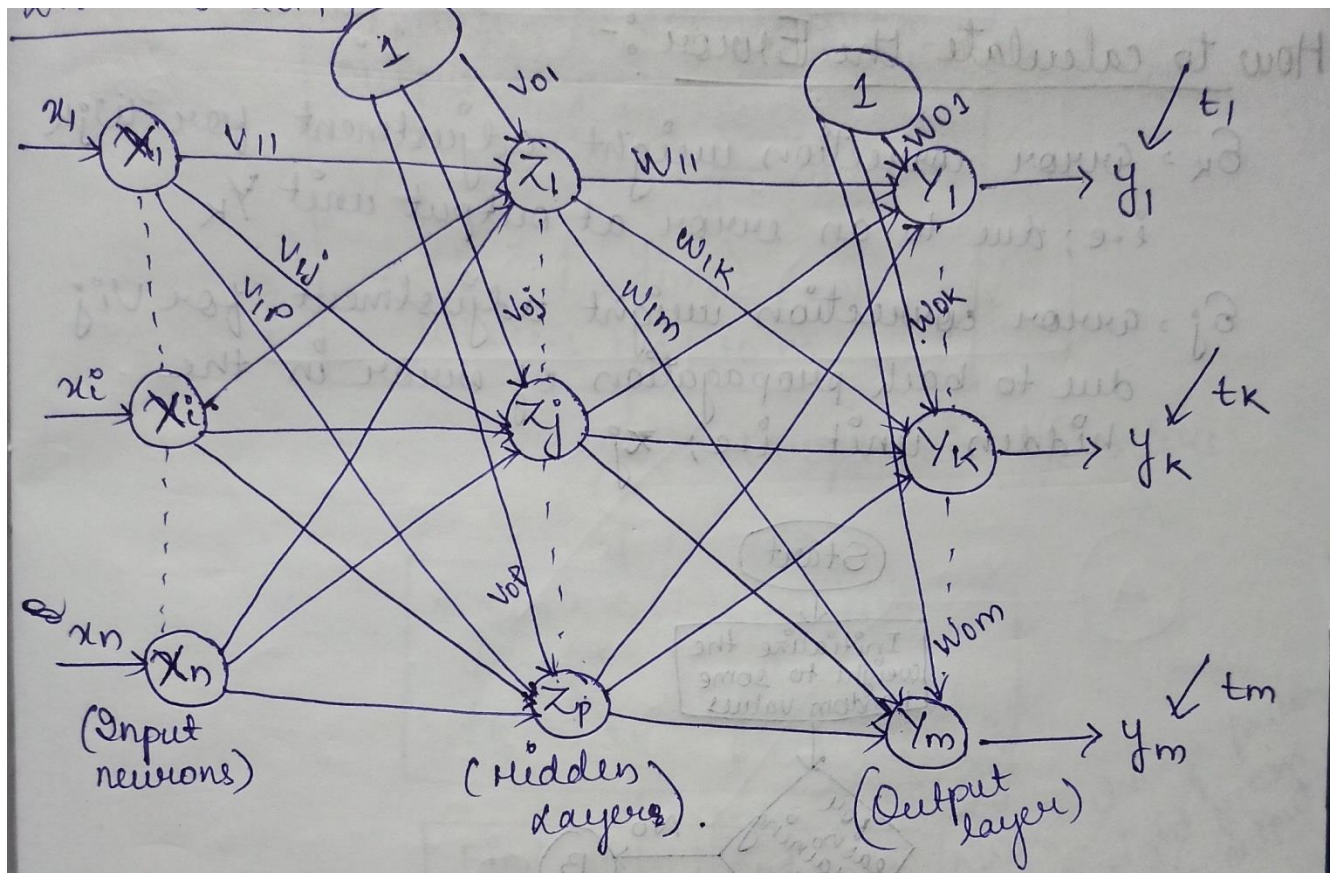
W_{oj} =bias m the j th hidden unit $\{j=1 \text{ to } p\}$

W_{ok} =bias m the k th output unit $\{k=1 \text{ to } m\}$

Z_j =hidden unit j

Flow chart of perception training algorithm for multiple output class





- $Z_{inj} = V_{0j} + \sum_{i=1}^n x_i v_{ij}$ → the net input of z_j
- The total output is $Z_j = f(Z_{inj})$
- Calculation at the hidden layer which the final output will be forwarded to the output layer
- **Output layer:-**
- Y_k = the output at the k th unit
- $Y_{jpk} = W_{0k} + \sum_{j=1}^p Z_j W_{jk}$
- The total output is :
- $Y_k = f(Y_{jpk})$

How to calculate error :-

- δ_k = Error correction

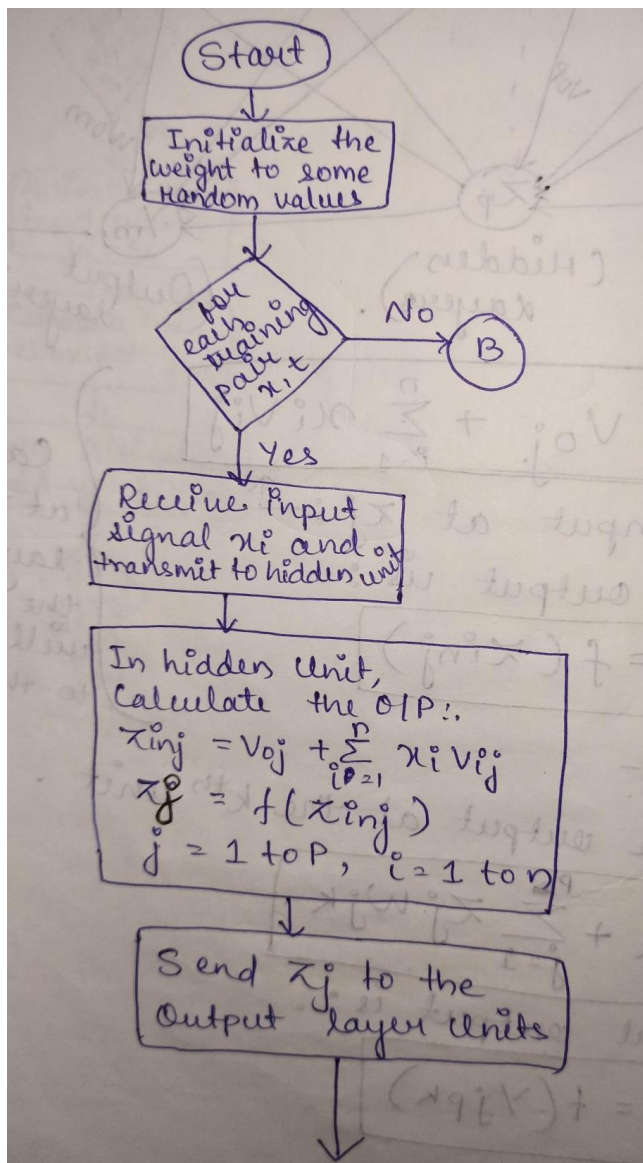
Weight adjustment for W_{jk} i.e; due to an error at

Output unit Y_k

- δ_j = Error correction

weight adjustment for V_{ij} due to back propagation of error in the hidden unit

i.e; Z_j



Calculate o/p signals from output layer:

$$y_{ink} = W_{ok} + \sum_{j=1}^p z_j w_{jk}$$

$$y_k = f(y_{ink})$$

where $k = 1$ to m

A

target pair t_k enters

Compute error correction factor

$$f_k = (t_k - y_k) f'(y_{ink})$$

Find weight and bias correction term:

$$\Delta w_{jk} = \alpha \delta_k z_j$$

$$\Delta W_{ok} = \alpha \delta_k$$

Δw_{jk} = change factor in weight from hidden to output
 ΔW_{ok} = change in bias in output unit.

Calculate error term δ_j (between hidden & i/p)

$$\delta_{in j} = \sum_{k=1}^m \delta_k w_{jk}$$
$$\delta_j = \delta_{in j} f'(z_{in j})$$

Compute change in weights and bias based on:

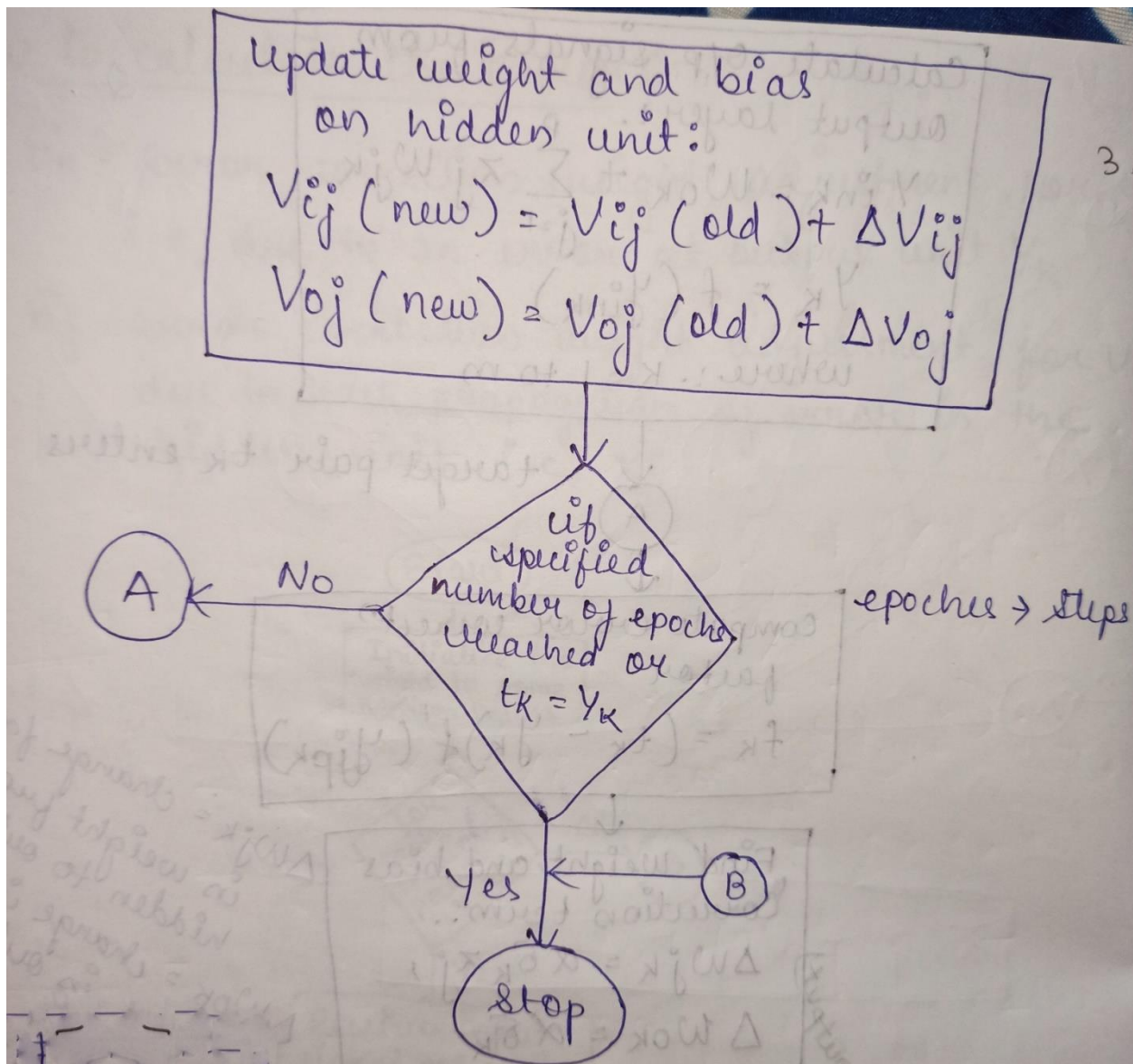
$$\Delta v_{ij} = \alpha \delta_j x_i$$

$$\Delta v_{oj} = \alpha \delta_j$$

Update weight and bias on o/p unit

$$w_{jk}(\text{new}) = w_{jk}(\text{old}) + \Delta w_{jk}$$

$$W_{ok}(\text{new}) = W_{ok}(\text{old}) + \Delta W_{ok}$$



Algorithm:-

- Step-1:-Start the perception process.
- Step-2:-Initialize the weight to some random values.
- Step-3:-For each training pair, the set of input values(x)will be mapped with the set of target values(t)

Map the number of inputs with the values else goto step-11(receive the input signal (xi) and transmit it to the hidden layer)

- Step-4:-If the input signals are received ,then the output is calculated in hidden unit area.

$$Z_{inj} = V_{oj} + \sum_{i=1}^n x_i V_{ij}$$

$$Z_j = f(Z_{inj})$$

Where j =1 to p, i=1 to n

- The final output from the hidden layer is then transmitted to the output layer.
- Step-5:-Output layer calculates the final output :

$$Y_{ink} = W_{ok} + \sum_{j=1}^p Z_j W_{jk}$$

$$Y_k = f(y_{ink}) \quad \text{where } k=1 \text{ to } m$$

- Step-6:-Target pair “A” enters (the target value). Now the output is being mapped with the target values.

$$(\text{Error correction factor}) f_k = (t_k - y_k) f'(y_{jpk})$$

- Step-7:-If the error is being calculated, then find the weight and bias:-
- $\Delta W_{jk} = \alpha \delta_k Z_j$;
- $\Delta W_{ok} = \alpha \delta_k$;
- Where ΔW_{jk} =change factor in weight from hidden to the output layer
- Δw_{ok} =change in bias in the output unit
- Step-8:-Then calculate the error correction term:-

(inbetween input and hidden layer)

$$\delta_{inj} = \sum_{k=1}^m \delta_k W_{jk}$$

$$\delta_j = \delta_{inj} f'(z_{inj})$$

- Step-9:-Calculate the error in weight and bias in hidden unit:-

$$\Delta V_{ij} = \alpha \delta_j x_i$$

$$\Delta V_{oj} = \alpha \delta_j$$

- Step-10:-After the calculation again:-update the weight and bias the output unit as:-

$$W_{jk}(\text{new}) = W_{jk}(\text{old}) + \Delta W_{jk}$$

$$W_{ok}(\text{new}) = W_{ok}(\text{old}) + \Delta W_{ok}$$

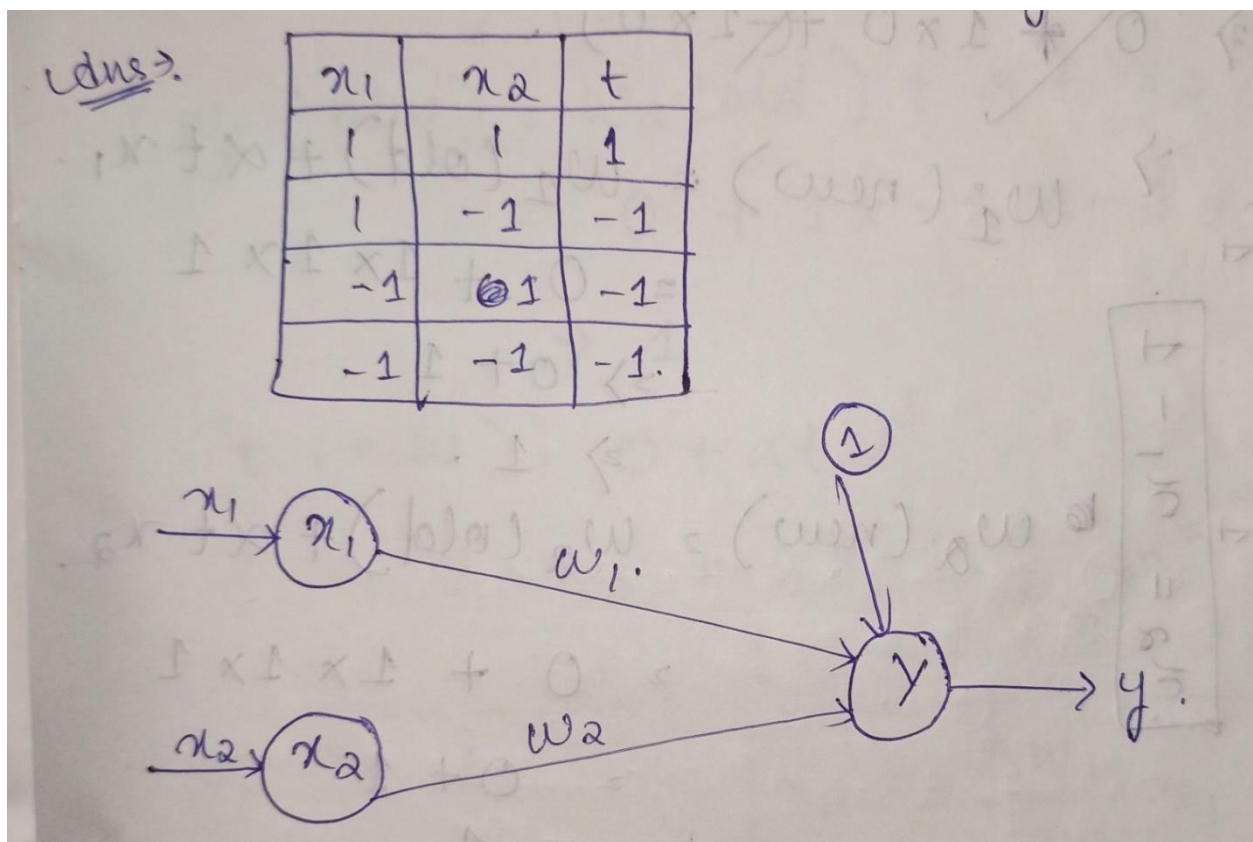
- Step-11:- Update the weight and bias on the hidden unit as :-

$$V_{ij}(\text{new}) = V_{ij}(\text{old}) + \Delta V_{ij}$$

$$V_{oj}(\text{new}) = V_{oj}(\text{old}) + \Delta V_{oj}$$

- Step-12:- If the target matches the output then STOP else goto step-6

Q) Implement the AND function using perception network for bipolar inputs and targets (imp)

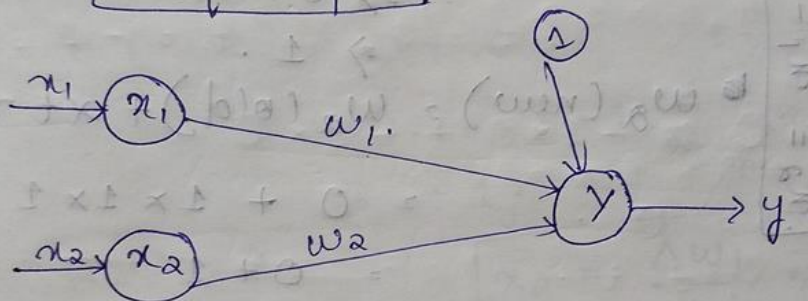


Question:-

1. Implement the AND Function using perceptron n/w for bipolar inputs and ~~bias~~ targets.

Ans:-

x_1	x_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1



Assume:-

$$w_1 = w_2 = b = 0 \quad \theta = 0$$

$$\alpha = 1$$

For 1st input pattern:-

$$x_1 = 1 \quad x_2 = 1 \quad t = 1 \text{ with}$$

weight and bias

$$\Rightarrow 0 + 1 \times 0 + 1 \times 0$$

$$\Rightarrow \text{~~1 + 1 + 1~~ } 0$$

$$x_1 = 1, x_2 = -1, t = -1$$

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } -\theta < y_{in} < \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

$$y = 0 \quad y \neq t$$

$$x_1 = 1, x_2 = -1$$

$$\Rightarrow 0 + 1 \times 0 + (-1) \times 0$$

$$\Rightarrow 0$$

$$w_i(\text{new}) = w_i(\text{old}) + \alpha t x_i$$

$$= 0 + 1 \times 1 \times 1$$

$$\Rightarrow 1$$

$$w_a(\text{new}) = w_a(\text{old}) + \alpha t x_a$$

$$= 0 + 1 \times 1 \times (-1)$$

$$= -1$$

$$b(\text{new}) = b(\text{old}) + \alpha t$$

$$= 0 + 1 \times 1 \times 1$$

$$= 1$$

$$\Delta w_1 = 1$$

$$\Delta w_2 = -1$$

$$\Delta b = 1$$

For: $x_1 = 1, x_2 = -1, t = 1$

$$w_1 = w_2 = b = 0$$

$$y = 0 + 1 \times 0 + (-1) \times 0$$

$$= 0$$

It is not matching with the target $y \neq t$

$$w_1(\text{new}) = w_1(\text{old}) + \alpha t x_1$$

$$= 0 + 1 \times 1 \times 1$$

$$\Rightarrow 1$$

$$w_2(\text{new}) = w_2(\text{old}) + \alpha t x_2$$

$$= 0 + 1 \times 1 \times (-1)$$

$$= -1$$

$$\Rightarrow 1$$

$$b(\text{new}) = 0 + \alpha t$$

$$= 0 + 1 \times 1 \times 1$$

$$= 1$$

$$\Delta w_1 = 1$$

$$\Delta w_2 = -1$$

$$\Delta b = 1$$

For: $x_1 = -1, x_2 = 1, t = -1$

$$w_1 = w_2 = b = 0$$

$$y = 0 + (-1) \times 0 + 1 \times 0$$

$$= 0$$

$$y \neq t$$

$$w_1(\text{new}) = 0 + \alpha t x_1$$

$$= 0 + 1 \times (-1) \times (-1)$$

$$= 1$$

$$w_2(\text{new}) = 0 + \alpha t x_2$$

$$= 0 + 1 \times 1 \times 1$$

$$= 1$$

$$b = 0 + 1 \times (-1)$$

$$= -1$$

$$\Delta w_1 = 1$$

$$\Delta w_2 = -1$$

$$\Delta b = -1$$

For: $x_1 = -1, x_2 = -1, t = 1$

$$y = 0 + (-1) \times 0 + (-1) \times 0$$

$$= 0$$

$$y \neq t$$

$$w_1 = 0 + 1 \times (-1) \times (-1)$$

$$= 1$$

$$w_2 = 0 + 1 \times (-1) \times (-1)$$

$$= 1$$

$$b = 0 + 1 \times (-1)$$

$$= -1$$

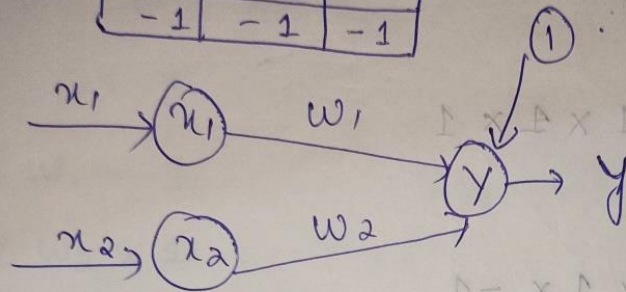
$$\Delta w_1 = 1$$

$$\Delta w_2 = -1$$

$$\Delta b = -1$$

Q) Implement the OR operation using network for bipolar input and targets(imp)

x_1	x_2	t
1	1	1
1	-1	1
-1	1	1
-1	-1	-1



Assume :- $w_1 = w_2 = b = 0$. $\theta = 0$.
 $\alpha = 1$.

For 1st input :-

$$x_1 = 1, \quad x_2 = 1, \quad t = 1.$$

$$y = 0 + 1 \times 0 + 1 \times 0 = 0.$$

$$y \neq t.$$

$$w_1 = 0 + 1 \times 1 \times 1 = 1.$$

$$= (1 \times 1) + (0 \times 1) + 0 = 1.$$

$$w_2 = 0 + 1 \times 1 \times 1 = 1.$$

$$= 1.$$

$$b = 0 + 1 \times 1 = 1.$$

$$\Delta w_1 = 1 \quad x_2 = \frac{1}{1} x_1 - \frac{1}{1}$$

$$\Delta w_2 = 1 \quad = x_1 - 1.$$

$$\Delta b = 1$$

$$x_2 = x_1 - 1$$

For 2nd Input:-

$$x_1 = 1 \quad x_2 = -1 \quad t = 1$$

$$y = 0 + 1 \times 0 + (-1 \times 0) = 0$$

$$y \neq t$$

$$w_1 = 0 + 1 \times 1 \times 1 = 1$$

$$w_2 = 0 + 1 \times 1 \times -1 = -1$$

$$b = 0 + 1 \times 1 = 1$$

$$\Delta w_1 = 1$$

$$\Delta w_2 = -1$$

$$\Delta b = 1$$

For 3rd Input:-

$$x_1 = -1 \quad x_2 = 1 \quad t = 1$$

$$y = 0 + (-1 \times 0) + (1 \times 0) = 0$$

$$y \neq t$$

$$w_1 = 0 + 1 \times -1 \times -1 = 1$$

$$w_2 = 0 + 1 \times 1 \times -1 = -1$$

$$b = 0 + 1 \times -1 = -1$$

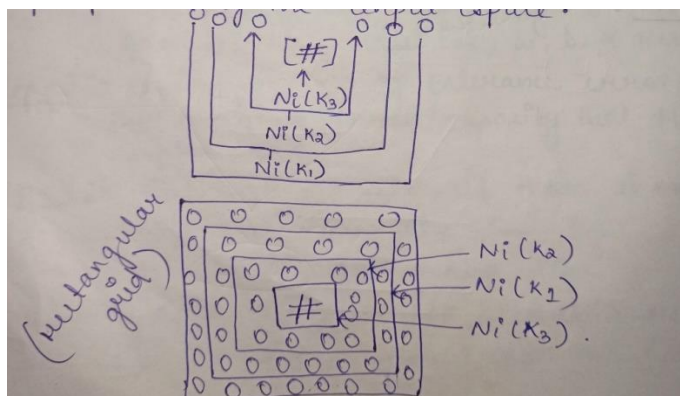
$$\Delta w_1 = -1$$

$$\Delta w_2 = -1$$

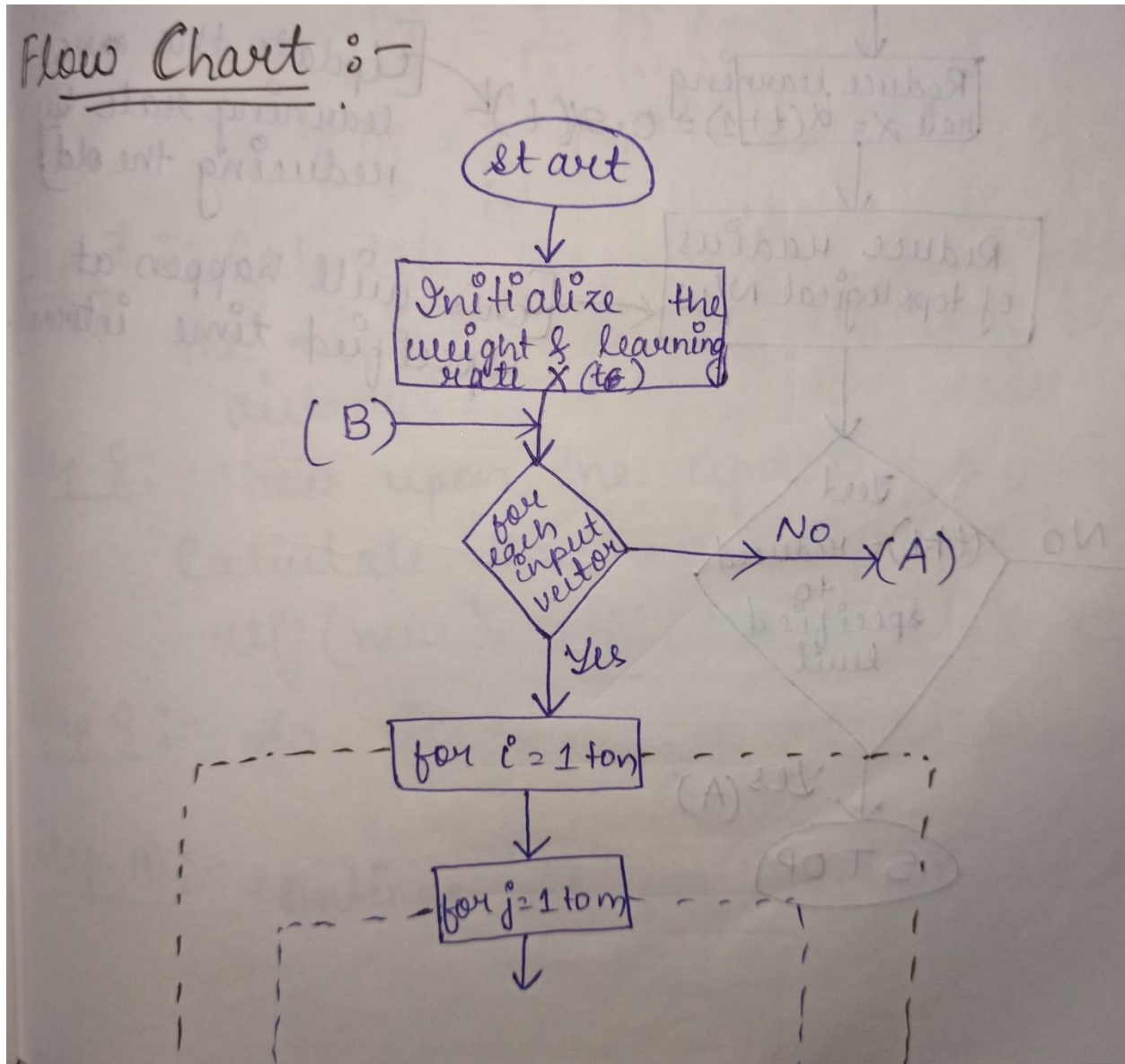
$$\Delta b = -1$$

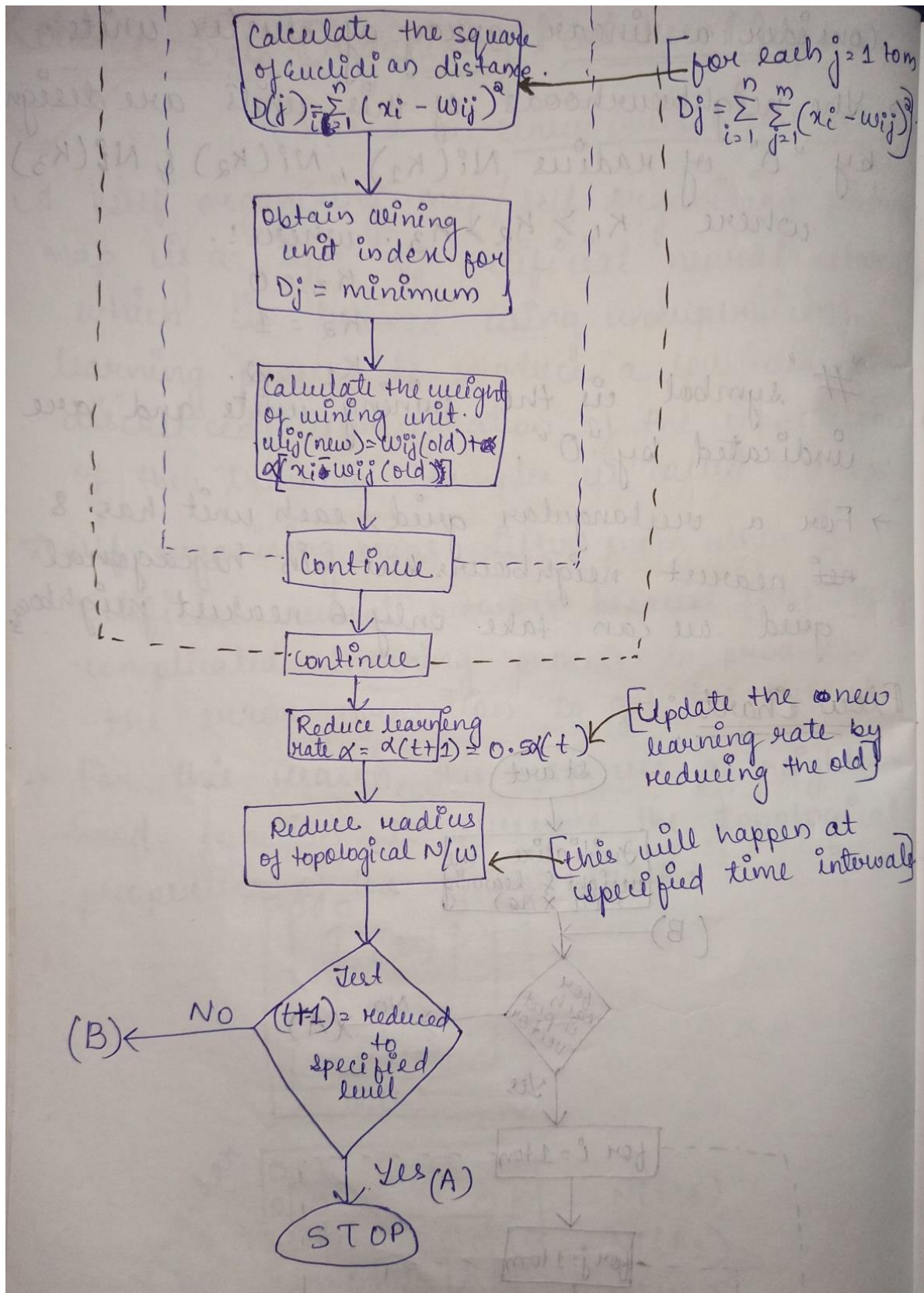
Kohonen-Self Organizing Features Maps(v.imp)

- A self organizing map/self organizing feature map is a type of artificial neural network which is trained using unsupervised learning process to produce a low-dimensional, discretized representation of the input space of the training samples is a map.
- Self organising maps differ from other artificial neural network because they apply complicated learning process to produce the error correction to get the output.
- For this reason, this map use a neighborhood function to preserve the topological properties of input space.



- Consider a linear array of cluster units.
- The neighborhoods of this units are designed by "0" of radius $Ni(k1), Ni(k2)$ & $Ni(k3)$ where $k1 > k2 > k3$ where : $k3=0, k2=1, k1=2$.
- "#" symbol is the winning units and are indicated by "0".
- For a rectangular grid, each unit has 8 nearest neighbors but in hexagonal grid all can take only 6 nearest neighbors.



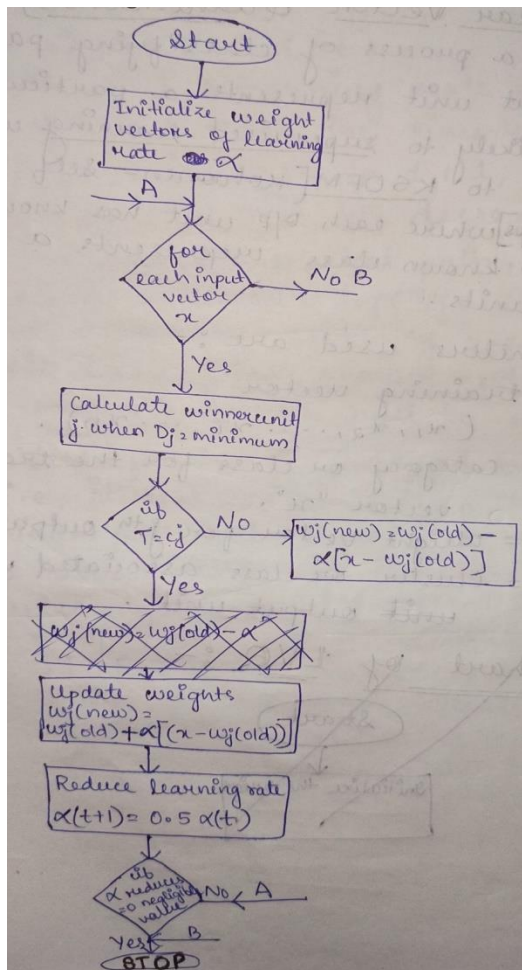


Algorithm

- Step-1:- Start the initial process.
- Step-2:-Initialize the learning rate and weight for each given input 'x' at particular time interval 't' i.e; x(t).
- Step-3:-If each input have corresponding update values then goto next(step-4) else goto Step-14
- Step-4:-for i<-1 to n
- Step-5:-for j<-1 to m
- Step-6:-Calculate the Euclidean distance for each j=1 to m.
- $D_j = \sum_{i=1}^n \sum_{j=1}^m (x_i - w_{ij})^2$
- $D_j = \sum_{i=1}^n (x_i - w_{ij})^2$
- Step-7:-Calculate the winning unit under 'j' i.e; the minimum distance among the number of distances.
- Step-8:-Then upon the updated winning unit, Calculate the new weight as:
- $W_{ij}(\text{new}) = W_{ij}(\text{old}) + \alpha [x_i - W_{ij}(\text{old})]$
- Step-9:-Do it for each j<-1 to m.
- Step-10:-Continue it for each i->1 to n.
- Step-11:-Reduce the learning rate by: $\alpha = \alpha(t+1)$
- $\Rightarrow \alpha = 0.5 \alpha(t)$
- Update the new learning rate by reducing the old, the old takes the time 't' & next the time will increase to t+1, alternatively the learning rate will be decreased.
- Step-12:-To find the shortest path, the system will make a network topology structure.
- For that reason by exploring one node then the path starts from another node , in that's why the radius will be reduced.
- Step-13:-Test for (t+1)=reduced to specified level i.e; it test within a time constraint the minimum path will explore or not. If it can do so then go to Step-4. Else goto step-3.
- Step-14:-STOP

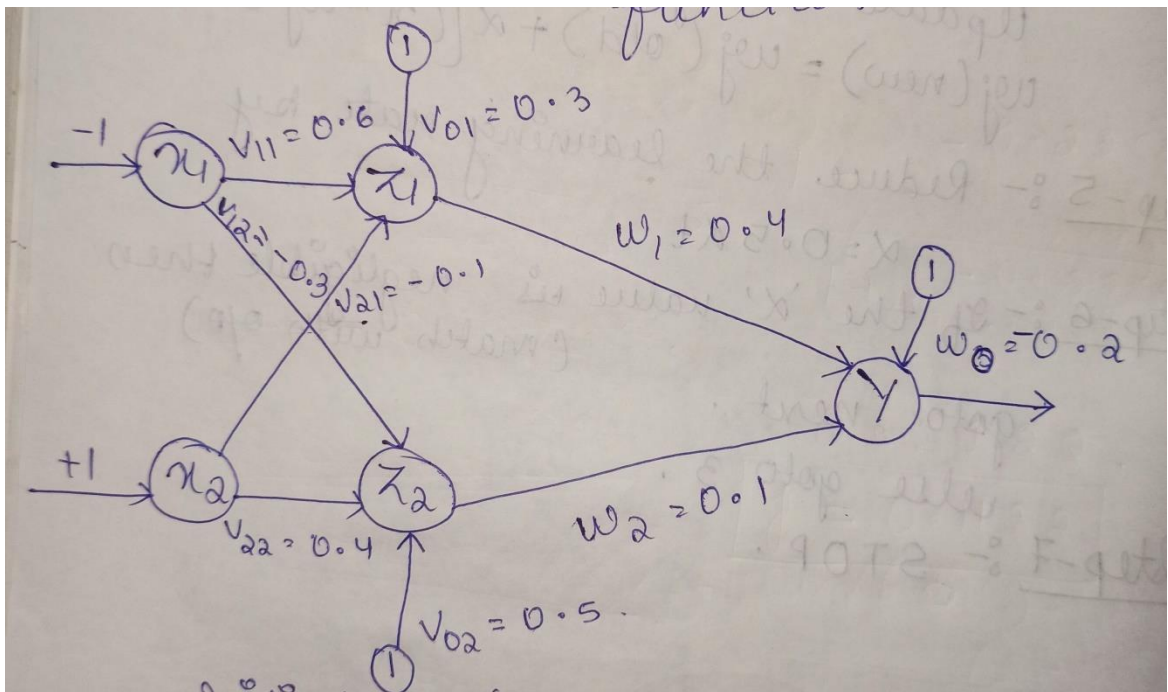
Linear Vector Quantization

- LVQ is a process of classifying patterns where each input unit represents a particular class.
- It is likely to supervised learning which is similar to KSOFM[Kohonen-self Organizing Feature maps] where each output unit has knowledge about a known class represents a set of input units.
- Parameters used are:-
- x =training vector($x_1, x_2, \dots, x_i, \dots, x_n$)
- T =category or class for the training vector " x "
- W_j =weight vector for j th output unit
- C_j =cluster or class associated with j th output unit



Algorithm

- Step-1:-Start the initial process
- Step-2:-Initialize the weights of the learning rate α
- Step-3:-Enter each input vector 'x' to calculate the winner unit "j"
- Step-4:-If $D_j = \text{minimum}$ and the target value = cluster
- Update eights:-
- $W_j(\text{new}) = W_j(\text{old}) + \alpha[(x - W_j(\text{old}))]$
- Step-5:-Reduce the learning rate by $\alpha = 0.5$ at
- Step-6:-If the ' α ' value is negligible then (match with output) goto next.
- Else goto 3
- Step-7:-STOP
- Q)Find the new weights using back propagation networks for the network shown in figure, the network is represented in input patter= $[-1,1]$
- Target output=+1
- Use learning rate $\alpha = 0.25$ & using bipolar sigmoidal activation function.



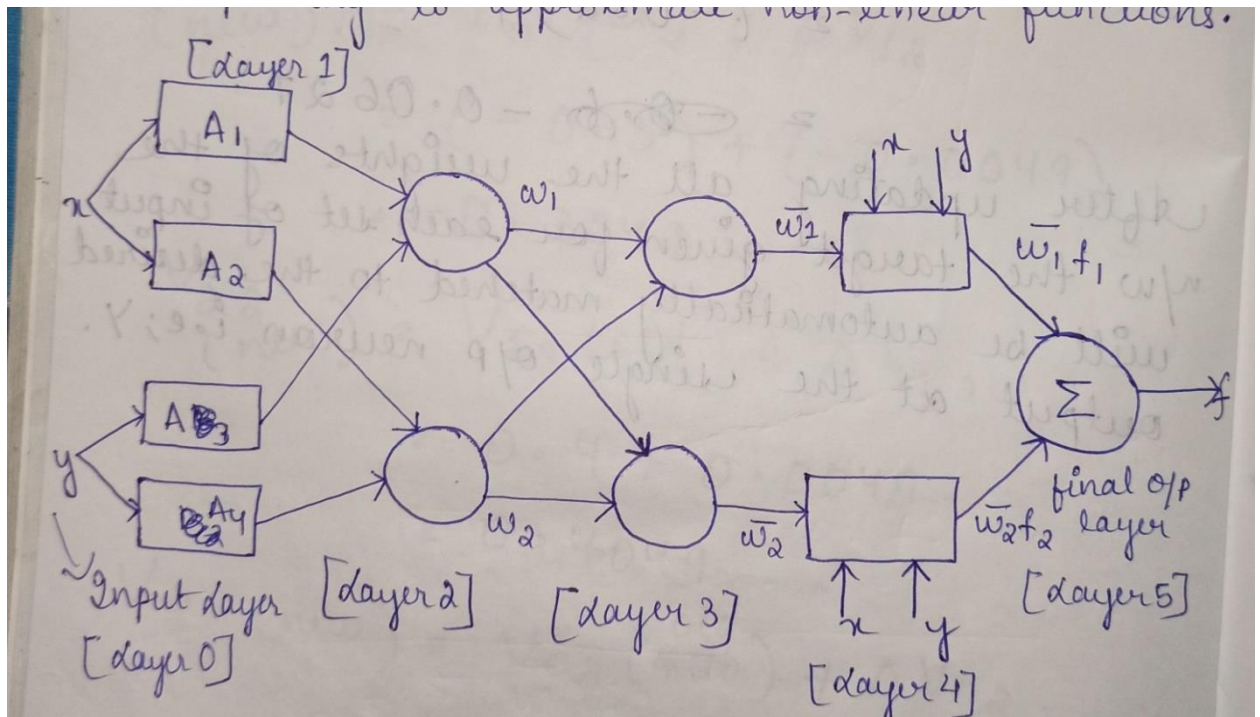
- Initial weight:-
- $[V11, V21, V01] = [0.6, -0.1, 0.3]$
- $[V12, V22, V02] = [-0.3, 0.4, 0.5]$
- $[W1, W2, W3] = [0.4, 0.1, -0.2]$
- $A = 0.25$
- **Input Unit:-**
- $[x1, x2] = [-1, +1]$
- Target $t = +1$
- Calculate the net input z :-
- $Zin1 = x1V11 + x2V21 + V01 = 0.6 - 0.1 + 0.3 = -0.4$
- $Zin2 = x1V12 + x2V22 + V02 = -0.3 + 0.4 + 0.5 = 1.2$
- **Output:-**
- $Z1 = f(Zin1) = 1 - e^{-zin1} / (1 + e^{-zin1}) = 1 - e^{0.4} / (1 + e^{0.4})$
- $= 0.1974$
- $Z2 = 0.537$
- Calculate the net input to the output layer:
- $Fin = w0 + z1w1 + z2w2 = -0.2 + (-0.1974 * 0.4) + (0.537 * 0.1)$
- $= -0.22526$
- $y = f(yin) = 1 - e^{-yin1} / (1 + e^{-yin1})$
- $= 1 - e^{-0.22526} / (1 + e^{-0.22526}) = 0.1122$
- Compute the error portion (δk)
- $\delta k = (tk - yk) f'(yink) = (tk - yk) f'(yin)$
- $f'(yin) = 0.25 [1 + f(yin)] * [1 - f(yin)] = 0.5 [1 + 0.1122] * [1 - 0.1122]$
- $= 0.4937$

- $f'(y_{in})=0.4937$
- $\delta_{1k} = (1+0.1122)*0.4937=0.5491$
- Find the changes in weight between output to hidden:-
- $\Delta w_1 = \alpha \delta_{1z1}=0.25*0.5491*0.1974=-0.02701$
- $\Delta w_2 = \alpha \delta_{1z2}=0.25*0.5491*0.537=0.0737$
- $\Delta w_0 = \alpha \delta_k=0.25*0.5491=0.1373$
- Compute the error portion δ_j between hidden to input where $j=1$ to 2 .
- $\delta_{in1} = \delta_{1W11}=0.5491*0.4=0.21964$
- $\delta_{in2} = \delta_{1W21}=0.5491*0.1=0.05491$
- **Error:-**
- $\delta_1 = \delta_{in1}f'(z_{in1})=0.21964*0.5*(1+0.1974)(1-0.1974)=0.1056$
- $\delta_2 = \delta_{in2}f'(z_{in2})=0.05491*0.5*(1+0.537)(1-0.537)=0.0195$
- **Now find the changes in weights in input and hidden.**
- $V_{11} = \alpha \delta_1 x_1=0.25*0.1056*(-1)=-0.0264$
- $V_{21} = \alpha \delta_1 x_2=0.25*0.1056*1=0.0264$
- $V_{01} = \alpha \delta_1 =0.25*0.1056=0.0264$
- $V_{12} = \alpha \delta_2 x_1=0.25*0.0195*(-1)=-0.0049$
- $V_{22} = \alpha \delta_2 x_2=0.25*0.0195*(+1)=0.0049$
- $V_{02} = \alpha \delta_2=0.25*0.0195=0.0049$
- Compute the final weight of the network:-
- $V_{11}(\text{new})=V_{11}(\text{old})+\Delta V_{11}=0.6+(-0.0264)=0.5736$
- $V_{21}(\text{new})=V_{21}(\text{old})+\Delta V_{21}=(-0.1)+0.0264=-0.0736$
- $V_{01}(\text{new})=V_{01}(\text{old})+\Delta V_{01}=0.3+0.0264=0.3264$
- $V_{12}(\text{new})=V_{12}(\text{old})+\Delta V_{12}=(-0.3)+(-0.0049)=-0.3049$

- $V22(\text{new}) = V22(\text{old}) + \Delta V22 = 0.4 + 0.0049 = 0.4049$
- $V02(\text{new}) = V02(\text{old}) + \Delta V02 = 0.5 + 0.0049 = 0.5049$
- $W1(\text{new}) = W1(\text{old}) + \Delta W1 = 0.4 + (-0.0271) = 0.3729$
- $W2(\text{new}) = W2(\text{old}) + \Delta W2 = 0.1 + 0.0737 = 0.1737$
- $W0(\text{new}) = W0(\text{old}) + \Delta W0 = (-0.2) + 0.1373 = -0.0627$
- After updating all the weights of the network the target given for each set of input will be automatically matched to the desired output at the single output neuron i.e; y.

Adaptive Neuro-Fuzzy Interface System (ANFIS)

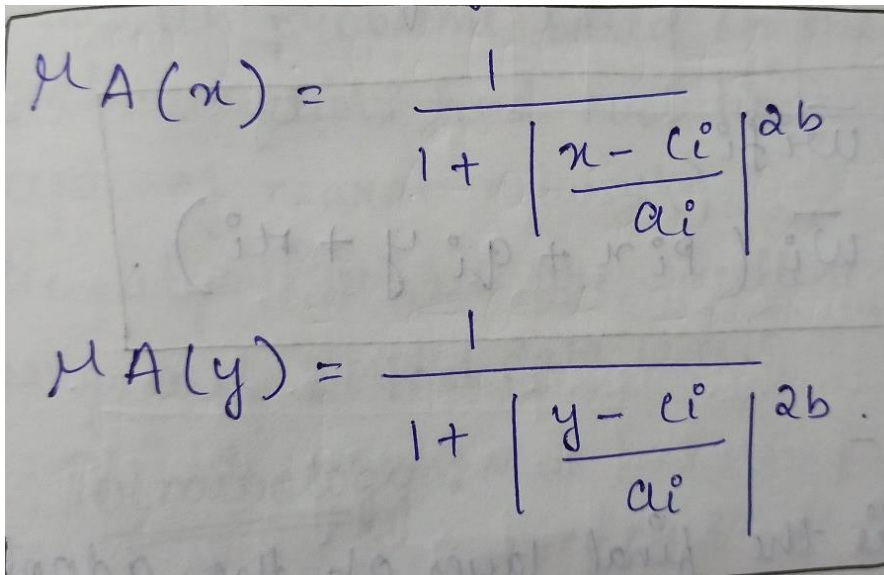
- It is a kind of artificial neural network based on Takagi-Sugeno Fuzzy Interface System.
- It integrates both neural networks and fuzzy logic principles.
- This inference system corresponds a set of fuzzy if-then rules that have learning capability to approximate non-linear functions.



It consists of 5 layers.

- **Layer-1-(Membership Function Layer)**

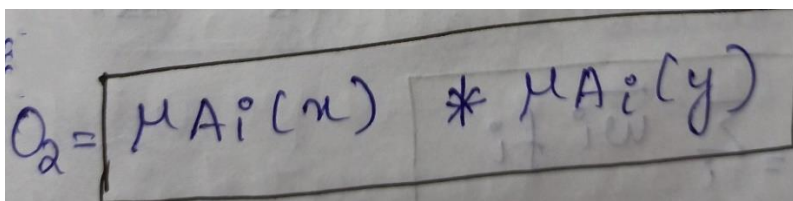
- It is known as membership function layer.
- In this layer every node 'i' will be calculated upon the membership function value " μ "
- $i = \mu_{A_i}(x)$ {for x input}
- $i = \mu_{A_i}(y)$ {for y input}
- For $i=A1$ to $A2$, $i=B1$ & $B2$
- Here at this input layer (layer 1), the total set of input will be calculated upon 4 fuzzy sets. $A1, A2$ & $A3-A4$ by the formula



The image shows two handwritten formulas for membership functions. The first formula is $\mu_{A_i}(x) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b}}$. The second formula is $\mu_{A_i}(y) = \frac{1}{1 + \left| \frac{y - c_i}{a_i} \right|^{2b}}$. Both formulas are written in blue ink on a white background.

- **Layer-2-**

- It is known as rule layer.
- In this layer 2 make the product of 2 incoming signals.
- $O_2 \rightarrow$ output of the layer 2



The image shows a handwritten formula for the output of Layer 2, which is $O_2 = \mu_{A_i}(x) * \mu_{A_i}(y)$. The formula is enclosed in a rectangular box and written in blue ink.

- **Layer-3-**

- It is also known as the non-linear layer.
- It is based upon finding out the final weight i.e; \hat{W}_1 and \hat{W}_2
- $O_3 = w_i / (w_1 + w_2)$
- where $i=1,2$
- $i=3,4$

- **Layer-4-**

- It is known as the prior of the output layer.
- Every node 'i' in this layer is an adoption node with a node function.

$$O_4 = \bar{w}_i f_i$$

$$= \bar{w}_i (p_i x + q_i y + r_i)$$

- **Layer-5-**

- This is the final layer of the adoptive system which computes the overall output as the submission of all incoming signals interms of the final output i.e; 'f'.
- (the final output will be interms of the fuzzy)

$$O_5 = \sum_i \bar{w}_i f_i$$

$$= \frac{\sum_i w_i f_i}{\sum_i w_i}$$

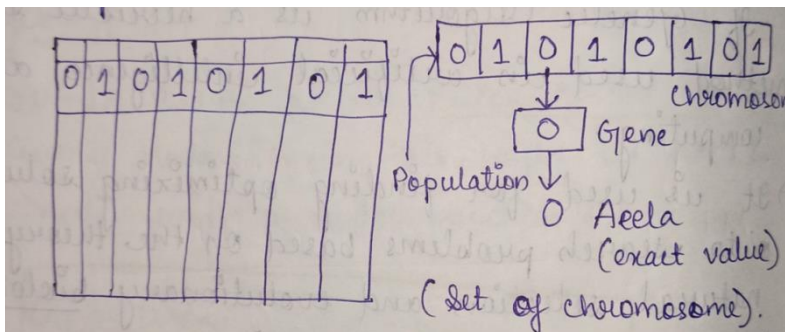
MODULE-4

Genetic Algorithm

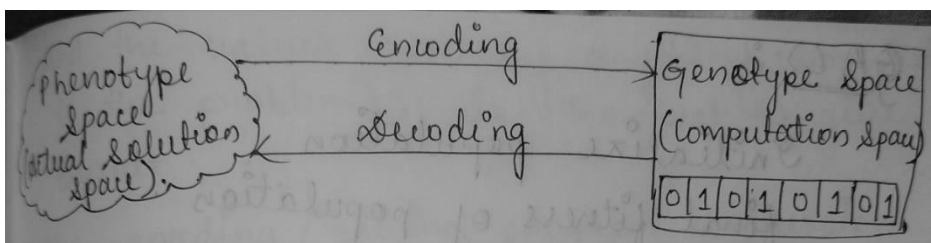
- Definition:
- If Genetic algorithm is a heuristic search method used in artificial intelligence and computing.
- It is used for finding optimizing solution to search problems based on the theory of natural selection and evolutionary biology (concept of neural network).
- Genetic algorithm are excellent for searching the large and complex data sets.

Basic Terminology

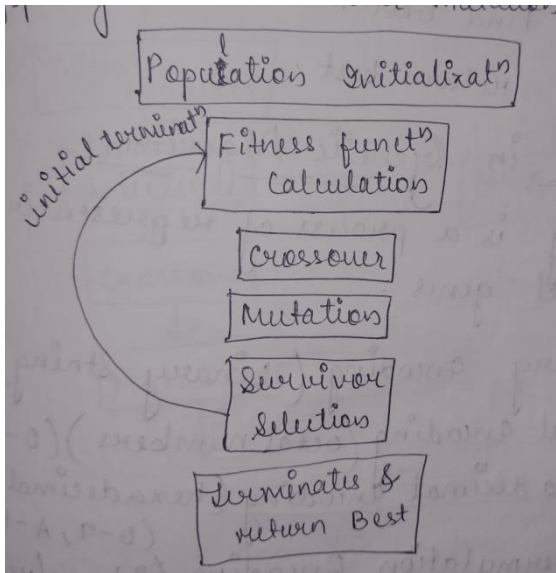
- Population: It is a subset of all the possible encoded solution to the given problem.
- Chromosomes: It is one such solution to the given problem.
- Gene: It is one element position of a chromosome.
- Allele: (exact value of particular gene) It is the value of a gene takes for a particular chromosome.



- Genotype: It is the population in the computation space in the computation space, the solution are represented in a way which can be easily understood and manipulated using a computing system.
- Phenotype: It is the population in the actual real world solution space in which solution are represented in a way they are represented in a real world situations.



- Fitness Function:- A fitness function $f(x)$ simply defined as a function which takes the solution as input and produce the suitability of the solution as the output.
- Genetic operators:- These alter the genetic composition of the offspring. These crossover mutation, selection etc.



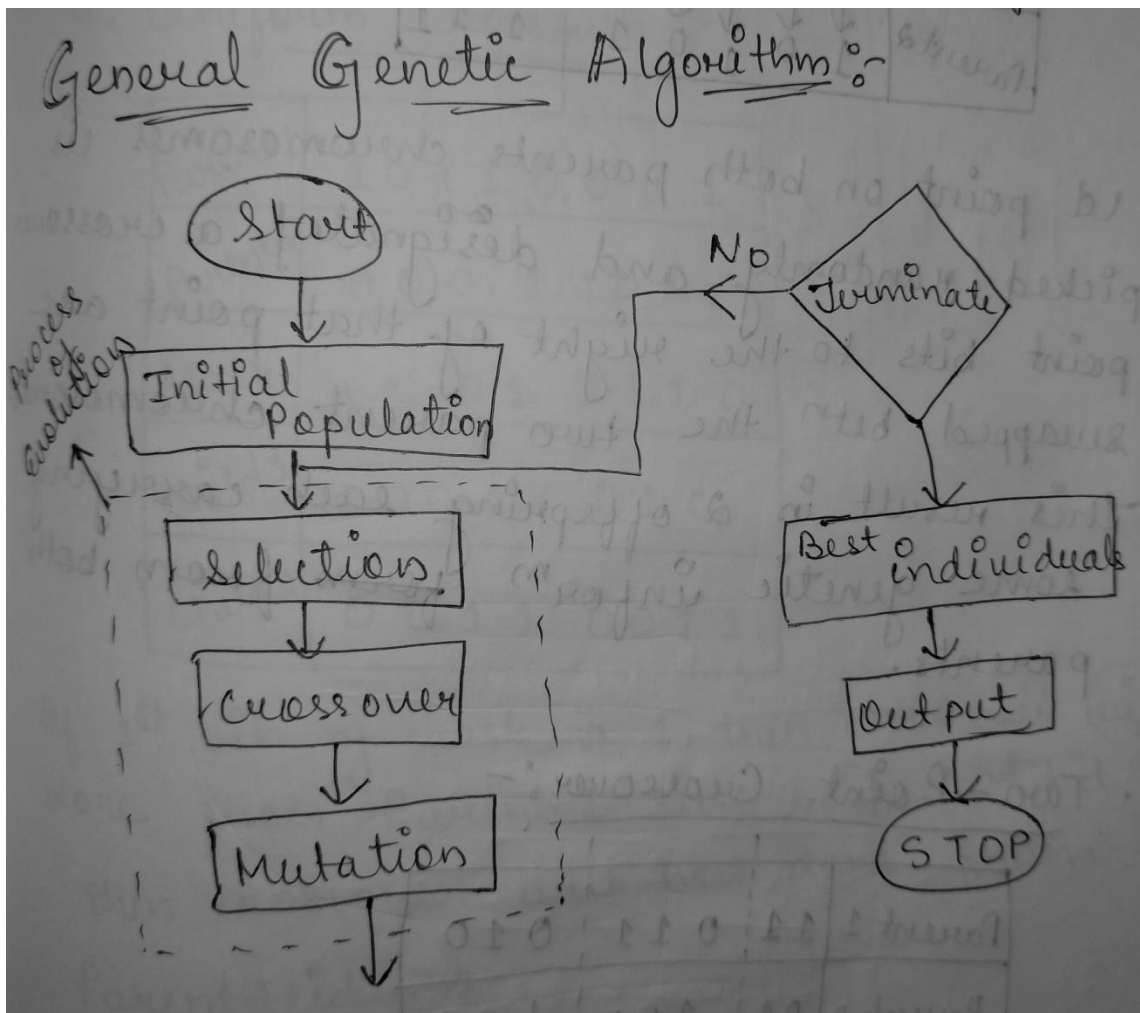
GA()

- Initialize population.
- Final fitness of population
- While (termination criteria is reached)
- Do parent selection
- Crossover with probability-PC
- Mutation with probability-PM
- Decode and fitness calculation
- Survivor selection
- Find best
- Return best

Operators in Genetic Algorithm

- Encoding is a process of representing individual genes.
1. Binary encoding(binary string)
 2. Octal encoding(octal numbers)(0-7)
 3. Hexa-decimal encoding(hexadecimal nos)(0-9.A-F)
 4. Permutation Encoding(integer/real number)
 5. Value encoding(every chromosome is a string of values and the values can be anything connected to the problem) it is the best to give result

The encoding(solving program expression for genetic programming. Every chromosome is a tree of some objects such as functions and commands of programming language)



Crossover: (Recombination)

- It is the process of taking 2 parent solutions producing from them a child.

1. Single Point Crossover:-

A point on both parents chromosomes is picked randomly and designated a crossover point bits to the right of that point are swapped between the two parent chromosomes.

This result in 2 offspring each carrying some genetic information from both parents.

Parent 1	1	0	1	1	0	0	1	0
Parent 2	1	0	1	0	1	1	1	1

2. Two-point crossover:-

- Two crossover points are picked randomly from the parent chromosomes .
- The bits in between the two points are swapped between the parent organisms.
- It is equivalent to performing two single-point crossovers with different cross points.

Parent 1	1	1	0	1	1	0	1	0
Parent 2	0	1	1	0	1	1	0	0

3. Multipoint crossover

- More than one points crossover will initiated between 2 points.

4. Uniform Crossover:-

If 1st bit of mask is 1, then crossover will be done from parent 1 side, if 2nd bit is 1 then crossover will be done from parent2 side.

Parent1	1 0 1 1 0 0 1 1
Parent2	0 0 0 1 1 0 1 0
Mask	1 1 0 1 0 1 1 0
child1	1 0 0 1 1 0 1 0
child2	0 0 1 1 0 0 1 1

5. Three point crossover:-

- It allow at time three parent crossover, but the process will be at time in between 2 parents.

Parent1	1 1 0 1 0 0 0 1
Parent2	0 1 1 0 1 0 0 1
Parent3	0 1 1 0 1 1 0 0
child	0 1 1 0 1 0 0 1

Mutation Operation:-

- After crossover, the strings are subjected to mutation.
- Mutation of a bit involves flipping it changing 0 to 1 and vice versa with a small mutation probability PM.

- **Mutation Rate:-**

- It is the probability of mutation which is used to calculate number of bits to be mutated.

1. 1's complement operator

a=0100 0001 ->4 1

-a=1011 1110 ->11 14

1. Logical bit-wise operator

1. Bit wise AND(&) operator

2. Bit wise OR(^) operator

- Shift left Operator:-

- a-> 1010 0110 ->10 6

- a<< 2->1001 1000->9,8

- Shift right operator:-

- a= 1010 0110 ->10,6

- a>>2=0010 1001 ->2,9

- Parent 1a=10101010 ->10,13

- Parent 2b=11000011->12,3

- AND : child a&b =10000010 ->8 2

OR: child a^b = 01101001 ->6 9

Difference between Genetic Algorithm and traditional algorithm:-

- Genetic algorithm is based on the principle of genetics and natural selection to solve optimization problems.
- While traditional algorithm is step by step procedure to follow in order to solve a given problem.

- GA uses population of solution rather than a single solution for searching. It improves the chance of reaching the global optimization.
- GA uses fitness function for evaluation rather than derivations.
- GA's use probabilistic transaction operates while conventional methods is used for continuous optimization.

Problem solving using Genetic algorithm

- Q-Consider the problem of minimizing the function $f(x) = x^2$ where x is permitted to vary between 0 and 31.

Gen No	Initial Population	Calculate the value of x	Fitness $f(x)$	Probability $\frac{f(x)}{\sum f(x)}$	% of Probability	Expected count	Actual count
1.	01100	12	144	0.1247	12.47%	0.4987	1.
2.	11001	25	625	0.5411	54.11%	2.164	2.
3.	00101	5	25	0.0216	2.16%	0.0866	0.
4.	10011	19	361	0.3125	31.25%	1.2502	1.
			1155				

$$Probability_i = \frac{f(x)_i}{\sum_{i=1}^n f(x)_i}$$

$$1 = \frac{144}{1155} = 0.1247$$

- $2 = (625/1155) = 0.5411$
- $3 = (25/1155) = 0.0216$
- $4 = 0.3255$
- Expected count = $(f(x)_i) / (\text{Avg}(f(x)_i))$
- $1 = \{(144) / (1155/4)\} = 144/288.75 = 0.4987$
- $2 = 625/288.75 = 2.164$
- $3 = 25/288.75 = 0.0866$
- $4 = 361/288.75 = 1.2502$
- Step-6- Now the actual count is to be obtained to select the individuals who would participate in the cross-over cycle using Roulette wheel selection procedure.

- Step-7-The Roulette wheel selection procedure will be forward with a entire Roulette wheel covers 100%.

Roulette Wheel:-

- According to Roulette wheel :-

Selection	Actual count
54.11%	2
31.26%	1
12.47%	1
2.16%	0
- After the selection process:

Cross Over Table :-

String No.	Mating Pool	Crossover Point	Offspring after Crossover	x value	Fitness value $f(x) = x^2$
1.	011100	4	01101	13	169
2.	11001	4	11000	24	576
3.	00101	2	00010	2	4
4.	10011	2	10100	20	400

$T = 1149$
 $Avg = 287.25$

Final output after mutation

Final Output After Mutation :-

String No.	Offspring after Crossover	Mutation chromosomes by Flipping	Offspring after Mutation	x value	Fitness value $f(x) = x^2$
1.	01101	100110	100110	189	35721
2.	11000	001111	001100	26	676
3.	00010	11101	10010	26	676
4.	10100	01011	01100	12	144

$189 + 676 + 676 + 144 = 1217$
 $Avg = 304.25$