Foundation for Advancement of Science & Technology

AI

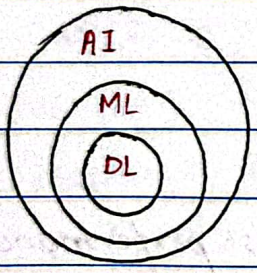
23-01-24

Artificial Intelligence : Computers mimicking human behaviour

Data
- continous (regression)
- discrete (classification)

tesla cars, ChatGPT all are examples of weak AI

AI
- weak AI (narrow AI) => task specific (all AI tech we use)
- strong AI machine is as smart as humans
- super AI machine surpasses human behavior



supervised learning : labeled training data

unsupervised learning: given unlabeled data

- Agent : software or system designed to act autonomously & perform specific tasks.
- sensors → perceives environment actuators → act through using them
- Rational agent → does the right thing
- percept sequence → the prior knowledge, history
- PEAS → sensors
  - performance measure
  - environment
  - Actuators
- Fully observable : complete knowledge of environment
- partially observable
- Unobservable → dish washer
- Deterministic → next state of environment is determined by current state & action executed by agent otherwise stochastic

- stochastic → certain degree of randomness or probability
- strategic → refers to decision making process
- single-agent vs multi-agent

solving crossword puzzle

chess playing

co-operative

competitive

partially competitive

- episodic vs sequential

↓ single action doesn't affect any other action

↓ current decision affects future decisions

- static vs dynamic → env can change

semi-dynamic → env itself doesn't change but agent's performance score does

- Discrete vs continous
- known vs unknown → focused on agent's state of knowledge

PIKP

intelligent behavior → perception, reasoning, learning, communicating

↓ action in complex environments

Agent function → maps the percept into an action

Foundation for Advancement of Science & Technology

♥ calto ♥
29-01-24

- Simple Reflex Agent → simple "if then" rules
  Model-based → percept history (previously-observed history) → takes into account the actions at a particular instance of time irrespective of the overall goal
  Goal-based → goal-oriented → focuses on overall goal
  Utility-based → efficiency, desirability of a particular state or outcome
  learning agent → improve their performance over time

- Goal
  Problem < states, actions
  Solution

  Abstraction ↓
  Removing details from a representation

- Toy problem: illustrating some problem solving method using a less complex problem than real world scenarios.

- $n!2^n$ → # of states in state space graph.
- 8-Queen Problem: Place 8 Queens on chessboard such that no Queen attack any other
  ↳ complete state ↓ all queens placed on the board
  ↳ Incremental ↓ adding one queen at a time
- Search strategy defined by picking order of node expansion

predicate Calculus

Foundation for Advancement of Science & Technology

31-01-2024

Uniformed Search Strategies : No prior knowledge about the no of nodes etc. Only "goal node" is known.

→ only info available in problem definition

BFS DFS UCS (uniform-cost) DLS (depth limited) IDS (iterative deepening)

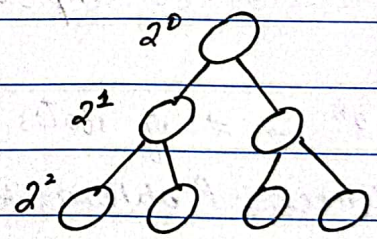
- Breadth-first search →

Queue → FIFO Expand shallowest unexpanded node

- Frontier → successors of visited nodes yet to be explored
- Explored → visited nodes

Branching factor → # of successors of a node

the branching factor ← b^d → depth



- Shallowest node → the node closest to root node.
- Time complexity & space complexity both → $O(b^d)$
- unequally & weighted & unweighted graphs → BFS optimal

- Depth-first search →

stack → LIFO Expand deepest unexpanded node in the current fringe.

Not complete, not optimal

Foundation for Advancement of Science & Technology

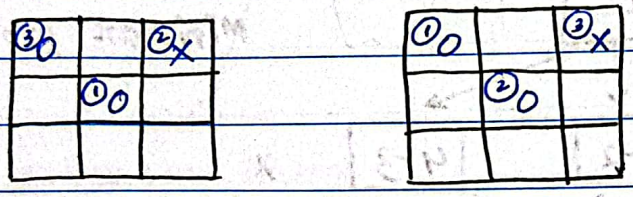
01-02-2024

→ Problem solving & state space:-

- initial state
- current state
- Goal state

State space → All possible states

Duplicate states problem → Because sometimes diff paths can lead to the same state



Tictactoe

different order & diff branches of tree.

- So, state space's size ≠ no of nodes
  (less than 9! in Tictactoe) (9! in case of tictactoe)
- we cannot do exhaustive search in case of this many nodes in tree
- travelling salesman problem complexity → (n-1)!

Graph : nodes & arcs

- Labeled graph , directed , undirected , Rooted graph
  ↓ one parent node from which all others originate
- State space → four-tuple
  [N, A, S, GD]
  - N → set of nodes
  - A → set of arcs
  - S → start state (S) (subset of N) non-empty
  - GD → Goal description, has goal states (subset of N) non-empty → also has a property of path developed in the search

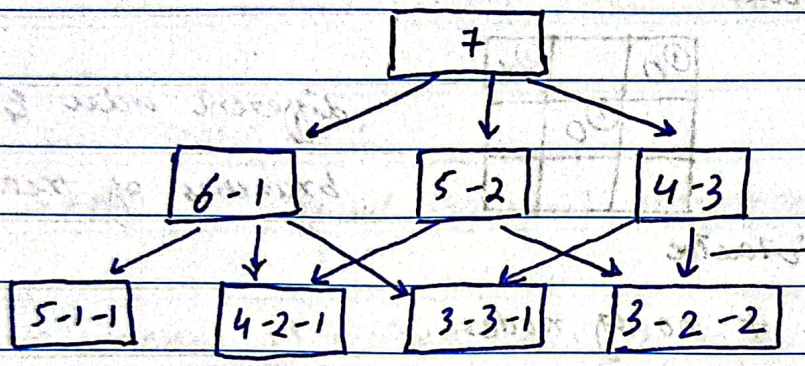
- Strategies for state space search
  - Data-Driven Forward chaining
  - Goal-Driven Backward-chaining

Foundation for Advancement of Science & Technology

$S \to A \to B \to D$ 7

↳ {[S → A → C → G, 4], [S → A → C → D → G, 6], [S → A → B → D, 7], [S → G, 12]}

UCS → path cost 0 → stuck in recursive loop.



Minimax procedure

X ——— cant do this.

, even split not allowed

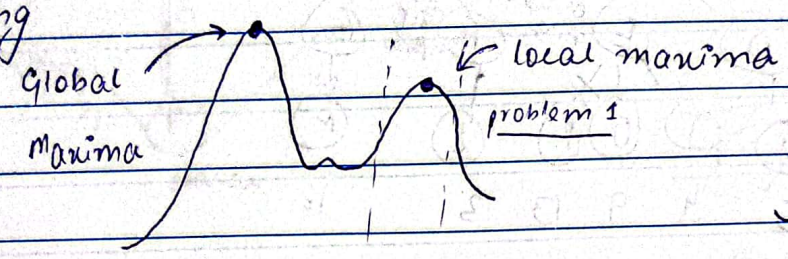
6 → 3-3 X 4 → 2X2 X

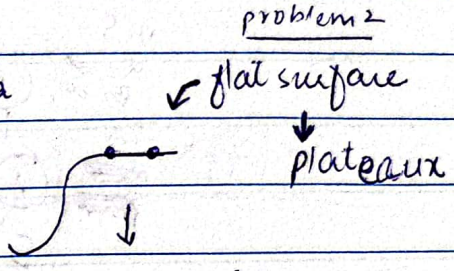
04-03-24

Local Search Algorithms :

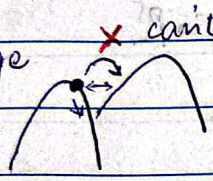
↳ path to goal irrelevant, goal state itself is the solution
↳ can't backtrack ↳ memory efficient

▶ Hill Climbing





current & next node are same (we got stuck)
↓
take a bad move & move to next node.
⇒ can be a plato or a shoulder



No previous info retained, like climbing everest in thick fog and amnesia.

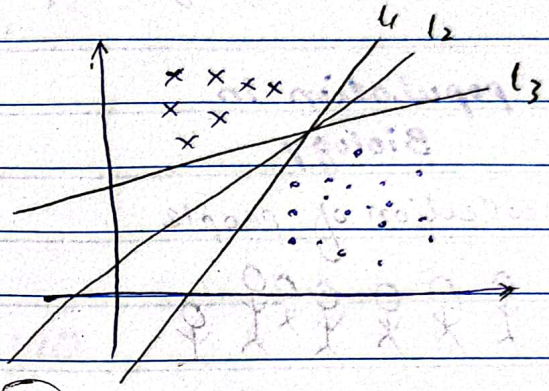
Variants

- Simple hill climbing examines one neighbour
- Steepest Ascend examines all neighbour
- Stochastic hill climbing chooses at random from among uphill moves
- Random-restart restart if you get stuck

Evolutionary Algorithms

→ Genetic Algorithm

→ biologically inspired model of intelligence
→ survival of the fittest
→ genetic operators → crossover, mutation



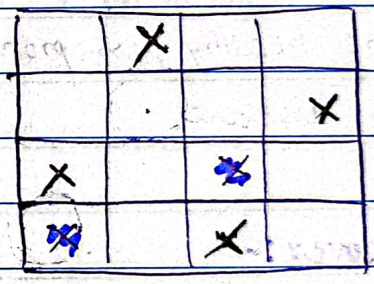
Attributes :-

- genetic representation of candidate solutions (chromosomes)
  ↓
  each one is unique
  phenotype → genotype conversion (encoding)
- Population size
- fitness func / evaluation func / score
- Genetic operators (crossover, mutation)
- selection algorithm
- generation gap (how many children produced and replaced into parent)
- amount of elitism used
- # of duplicates allowed

$y = mx + c$

2	4	1	2

index = col val = row



4-Queen problem

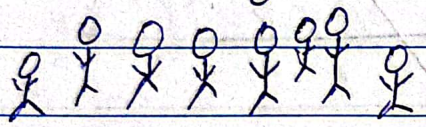
Chromosome :

Gene → basic unit, represents one characteristic of individual
Allele → value of each gene (values it can take i.e. eye color gene can be green, black, brown etc)
String of genes, represents a point in search space

population: collection of chromosomes

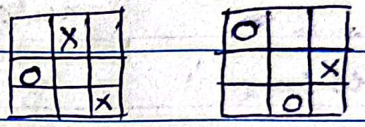
population in Biology

collection of people



population in Algorithms

collection of states



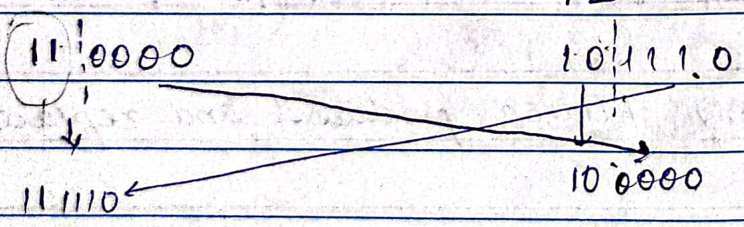
fittest chromosome in Biology

more healthy less prone to disease

fittest chromosome in algorithms

closest to final solution

Crossover:-



f.s ③ ③

sol	0 0 1 0 1 0	f.s
A	1 0 0 1 0 1	1
B	1 1 0 1 1 1	1
C	1 1 0 0 0 0	2
D	1 0 1 1 1 0	4

so, next generation doesn't gauranteedly gives optimal or better score than parents.

don't care terms = # or X

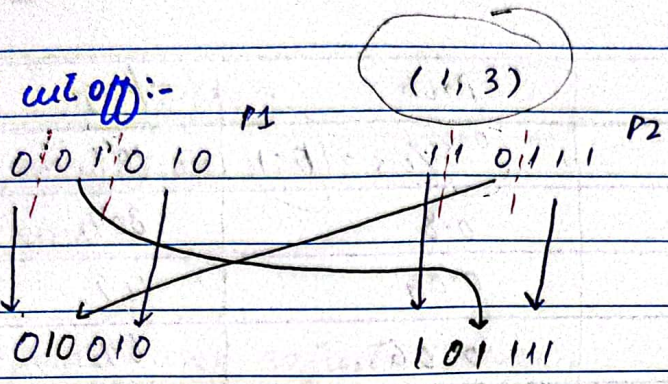
001010 sol

11#010 fitness score = 4

001010 sol

110010 f.s = 3

Two point cut off:-



multipoint crossover :-

011111 001110

Randomly select bits from both parents.

- Crossover mask :- | 0 | 1 | 0 | 0 | 1 | 0 |

  This means when there is 0, pick bit from P1 and when 1, pick bit from P2.

Mutation:- when the population is not diverse enough, we use mutation to handle it.

↳ Takes a single candidate & randomly changes some aspect (gene) of it.

- Mutation rate → less => convergence takes time
  more => original generation changed

- if generation gap is 1, completely replace all parents. If it is 0.5, half of generation is old, half is new.

Selection Algos

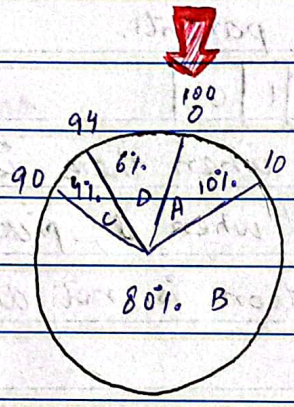
- Random → equal prob of all, not considering fos
- proportionate → assign prob based on fos
- linear Rank
- tournament

Foundation for Advancement of Science & Technology

→ Proportionate selection

chromosome	Fitness Score	Prob	%age
A	0.05	0.05/0.5 = 0.1	10%.
B	0.4	0.8	80%.
C	0.02	0.04	4%.
D	0.03	0.06	6%.
	0.5		

- Roulette wheel scenario:



To implement, define ranges for A, B, C, D ... & generate a random num & see where it lies.

if a chromosome say E has 0% prob then it will never go on the roullete wheel ⇒ disadvantage.

E | 0 | 0 | 0 | 0%.

→ linear Rank selection

rank	chrom	f.s	prob	%age	new fitness sc	new prob
1	B	0.4	0.8	80%.	5	33.3%.
2	A	0.05	0.1	10%.	4	26.7%.
3	D	0.03	0.06	6%.	3	20%.
4	C	0.02	0.04	4%.	2	13.3%.
5	E	0	0	0%.	1	6.7%.
					15	

$$\text{New fitness} = (P - r) + 1$$

P = population size

r = rank

to incorporate user adjusted slope:

$$N \cdot F = \frac{(P-r)(max - min)}{(P-1)} + min$$

let $max = 8$, $min = 3$

$$N \cdot F = \frac{(5-1)(8-3)}{(5-1)} + 3 = 8$$

m-method :-

r	chrom	f.s	prob	N.prob
1	B	0.4	0.8	0.8
2	A	0.05	0.1	(0.8)(1-0.8)
3	D	0.03	0.06	$(0.8)(1-0.8)^2$
4	C	0.02	0.04	$(0.8)(1-0.8)^3$
5	E	0	0	$(0.8)(1-0.8)^4$

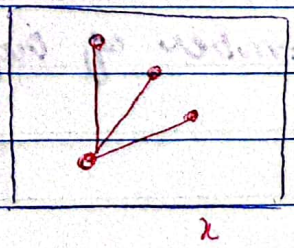
- **Survival of the most diverse**

It can be as good to be different as it is to be fit.

→ results in uniformity

↑ diversity = $\sum_j$ distance$_j^2$ ↑

↓ div = $\frac{1}{\sum_j d_j^2}$ ↑



Foundation for Advancement of Science & Technology

0.5 — 0 — 1

1 2 3 4 — 5 6 — 1/4

0.8 × 4 = ... 4 × 0.5 = 2

function = fitness + Diversity

Generation gap:

fraction of current population that gets replaced by the offspring. Between 0–1

Generation gap * P = offspring #

P = 6 gap = 0.5

6 × 0.5 = 3

c1
c2
c3
c4
c5
c6
} replaces least fittest 3 states by offsprings.

Amount of Elitism used: fraction that represents fraction of Best Induviduals of a population that will not get replaced. 0–1. only used if g.gap is not used.

Q:-

50

↓

5 + 45 — 30 new

elitism 15 old

generation gap = 30/50 = 0.6

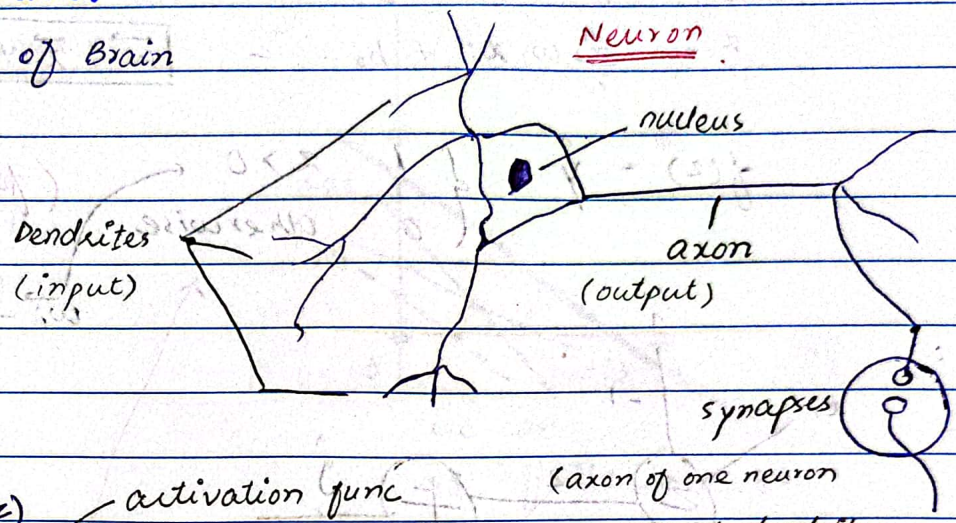
elitism = 5/50 = 0.1

20 ↓ old — 30 ↓ new — total

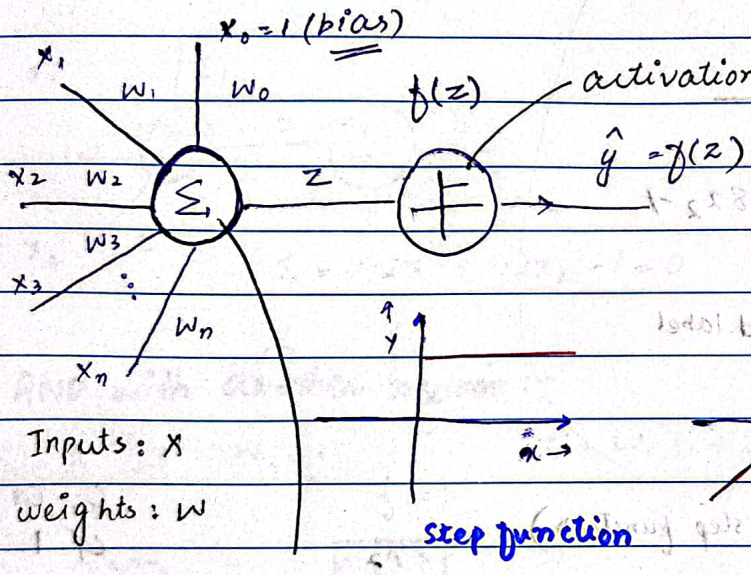
Number of Duplicates allowed:

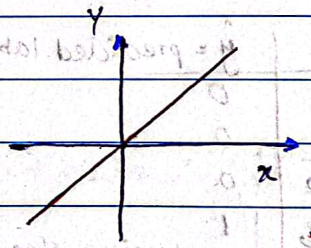
Artificial Neural Network:

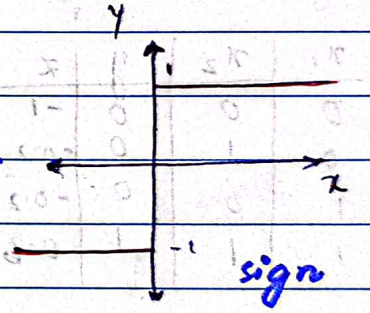
inspired by parallelism of Brain



Mathematical Neuron // perceptron

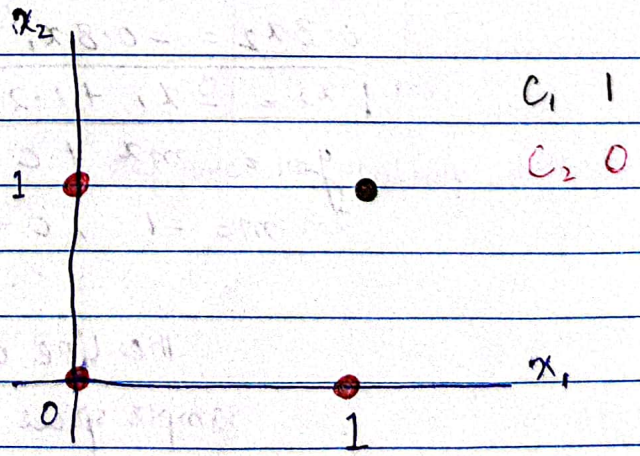






AND GATE:

f^1 x_1	f^2 x_2	y ← actual label / Ground truth.
0	0	0
0	1	0
1	0	0
1	1	1

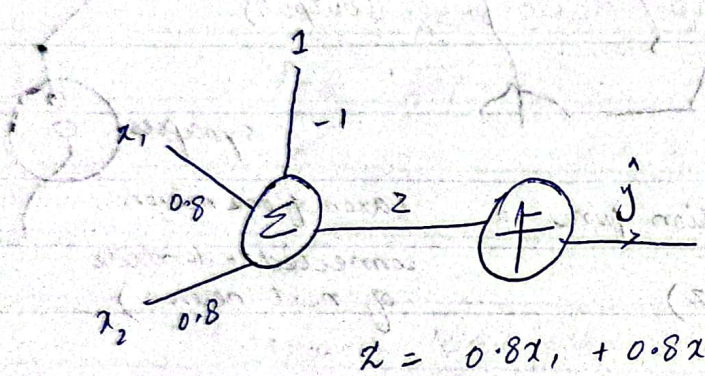


$$z = w_1 x_1 + w_2 x_2 + \dots w_n x_n + b_0$$

$$z = \sum_i w_i x_i + b_0 = \boxed{\vec{w} \cdot \vec{x} + b_0}$$

$$f(z) = \hat{y} = \begin{cases} 1 & z > 0 \\ 0 & \text{otherwise} \end{cases}$$ (for step function)

$\vec{w_i} - \vec{x_i} + b_0 > 0$



$z = 0.8x_1 + 0.8x_2 - 1$

x_1	x_2	y	z	$\hat{y}$ = predicted label
0	0	0	-1	0
0	1	0	-0.2	0
1	0	0	-0.2	0
1	1	1	0.6	1

(using step function)

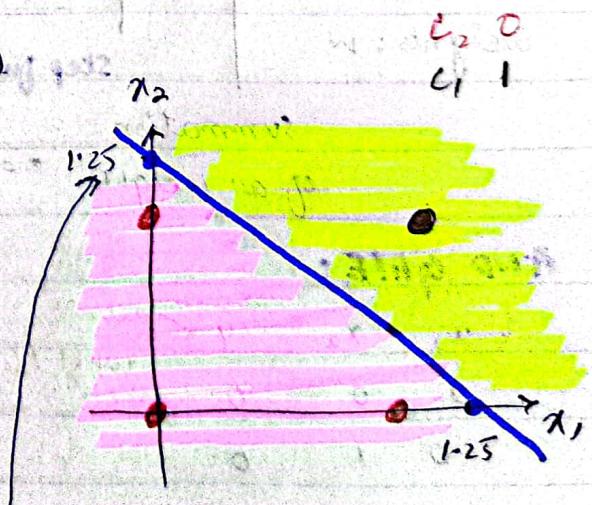
$0.8x_1 + 0.8x_2 - 1 = 0$

$0.8x_2 = -0.8x_1 + 1$

$\boxed{x_2 = -x_1 + 1.25}$

$y = mx + c$

$m = -1, \quad c = 1.25$



The line divides sample space into 2 classes (linear classifier)

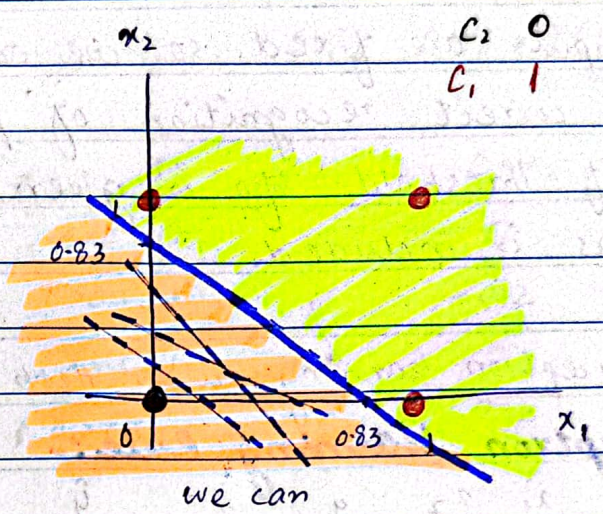
$z = 0.8x_1 + 0.8x_2 - 1 > 0$

$\Downarrow$

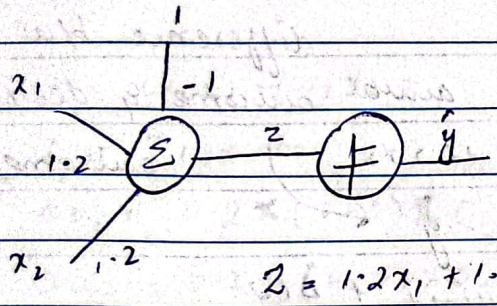
c_2

OR :

x_1	x_2	y	z	$\hat{y}$ = pred. val
0	0	0	-1	0
0	1	1	0.2	1
1	0	1	0.2	1
1	1	1	1.4	1



we can have infinite lines to separate these 2 classes based on m & c values & adjusting weights

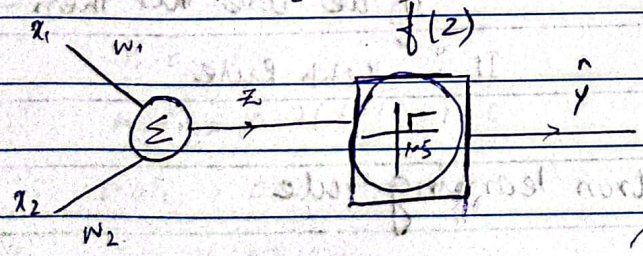


$z = 1.2x_1 + 1.2x_2 - 1 = 0 \Rightarrow x_2 = -x_1 + 0.83$

AND with another neuron :-

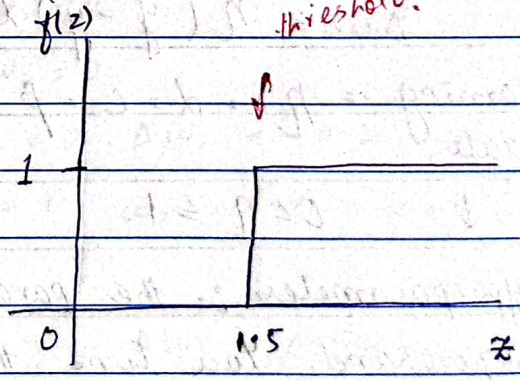
$w_1 = w_2 = 1$

$z = w_1 x_1 + w_2 x_2 = x_1 + x_2$



$\theta = 2$

x_1	x_2	y	z	$\hat{y}$	z_1
0	0	0	0	0	-1.5
0	1	0	1	0	-0.5
1	0	0	1	0	-0.5
1	1	1	2	1	0.5



step func horizontally shifted by 1.5

$z_1 = x_1 + x_2 - 1.5$ → bias is cancelling the shift effect.

- inputs are fixed so we can only adjust the weights for correct recognition of patterns.
- If Threshold info is given, then there is no bias. otherwise bias is included.

perceptron learning rule, Hebb rule

Perceptron learning Rule:-

x_1	x_2	y	z	$\hat{y}$
0	0	0	0	0
0	1	0	0.5	1
1	0	0	0.5	1
1	1	1	1	1

threshold = 0.4

$z = 0.5x_1 + 0.5x_2$

difference b/w actual outcome & desired outcome.

$error = y - \hat{y}$

$w_{new} = w_{old} + \Delta w$

$\Delta w = (y - \hat{y})x$

$\Delta w = \eta(y - \hat{y})x$ ← this is perceptron learning rule

If we use this then It is Hebb Rule

learning rate = $\eta = \alpha = c = \beta = a$

↓ $0 < \eta \leq 1$

hyperparameter :- the parameters whose value cannot be fixed beforehand. You tune the value by hit & trial & cannot predefine it.

Stopping condition:- No weight is updated.

Foundation for Advancement of Science & Technology

$\eta = 0.2$, threshold = 0.5 , $w_{new_i} = w_{old_i} + \Delta w_i$ $\Delta w_i = \eta (y - \hat{y}) x_i$

iteration # 01 :-

x_1	x_2	w_1	w_2	y/d	$\hat{y}$	Δw_1	Δw_2
0	0	1	1	0	0 ≯ 0.5 : 0	0	0
0	1	1	1	0	1 > 0.5 : 1	0	-0.2
1	0	1	0.8	1	1 > 0.5 = 1	0	0
1	1	1	0.8	1	0.8 > 0.5 = 1	0	0

① $z = w_1 x_1 + w_2 x_2$

$z = x_1 + x_2$

$\Delta w_1 = 0.2 \times (0-0) \times 0 = 0$

$\Delta w_2 = 0.2 \times (0-0) \times 0 = 0$

$w_1 = 1+0 = 1$ $w_2 = 1+0 = 1$

② $\Delta w_1 = 0.2(0-1)0 = 0$

$\Delta w_2 = 0.2(0-1)1 = -0.2$

$w_1 = 1$

$w_2 = 1 - 0.2 = 0.8$

③ $z = x_1 + 0.8 x_2$

$z = 1$

$\Delta w_1 = 0.2(1-1) \cdot 1 = 0$

$\Delta w_2 = 0.2(1-1) \cdot 0 = 0$

④ $\Delta w_1 = 0$

$\Delta w_2 = 0$

2 ⑦ $\Delta w_1 = 0$ $\Delta w_2 = -0.2$

$w_1 = 1$ $w_2 = 0.8 - 0.2 = 0.6$

Iteration # 02 :-

x_1	x_2	w_1	w_2	y/d	$\hat{y}$	Δw_1	Δw_2
0	0	1	0.8	0	0	0	0
0	1	1	0.8	0	0.8 > 0.5 : 1	0	-0.2
1	0	1	0.6	1	1	0	0
1	1	1	0.6	1	1	0	0

Foundation for Advancement of Science & Technology

iteration # 03 :-

x_1	x_2	w_1	w_2	y	$\hat{y}$	Δw_1	Δw_2
0	0	1	0.6	0	0	0	0
0	1	1	0.6	0	0.6 > 0.5 : 1	0	-0.2
1	0	1	0.4	1	1	0	0
1	1	1	0.4	1	1	0	0

3 (2)

$\Delta w_1 = 0.2(0-1)0 = 0$

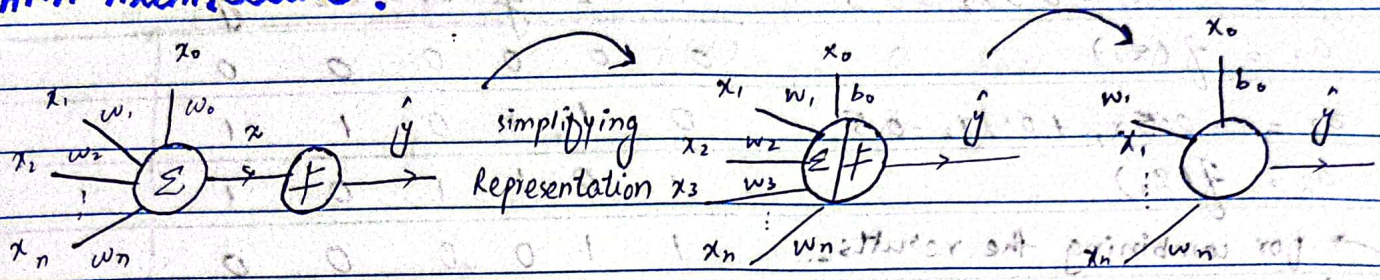
$\Delta w_2 = 0.2(0-1)1 = -0.2$

$w_1 = 1 \quad w_2 = 0.6 - 0.2 = 0.4$

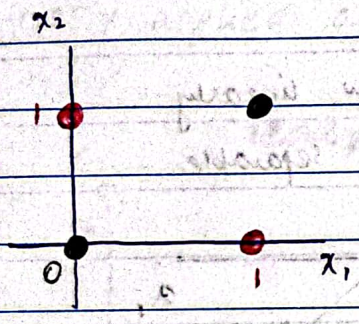
iteration # 04 :-

x_1	x_2	w_1	w_2	y	$\hat{y}$	Δw_1	Δw_2
0	0	1	0.4	0	0	0	0
0	1	1	0.4	0	0	0	0
1	0	1	0.4	1	1	0	0
1	1	1	0.4	1	1	0	0

ANN Architecture :



XOR = $x_1 \oplus x_2$



x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$x_1 \oplus x_2 = x_1\bar{x}_2 + \bar{x}_2 x_1$

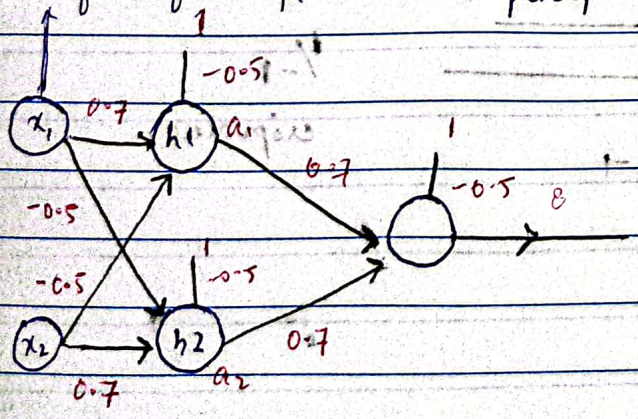
($x_1\bar{x}_2$ = a_1, $\bar{x}_2 x_1$ = a_2)

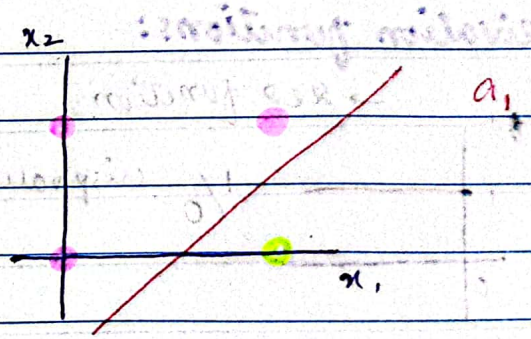
x_1	x_2	y	a_1	a_2
0	0	0	0	0
0	1	1	0	1
1	0	1	1	0
1	1	0	0	0

C1 = 1 C2 = 0

not linearly separable as 1 line cannot divide the classes so we need 2 perceptrons.

circle for uniformity (its not a perceptron)





now this is linearly separable. we split problem into 2 LTS problems.

level 0 input layer — level 1 hidden layer — level 2 output layer

→ Multilayer perceptron (MLPs)

Feed Forward NN:

The inputs only propagate forward

Recurrent NN:

inputs go backwards through some loop.

$z_1 = 0.7x_1 - 0.5x_2 - 0.5$

$a_1 = f(z_1)$

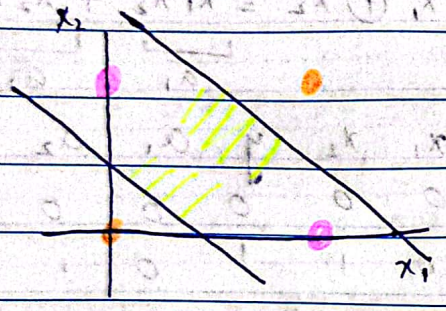
$z_2 = -0.5x_1 + 0.7x_2 - 0.5$

$a_2 = f(z_2)$

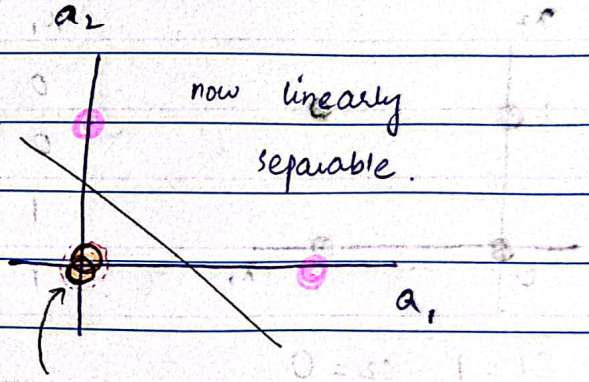
→ for combining the results:

$\hat{y} = 0.7a_1 + 0.7a_2 - 0.5$

x_1	x_2	y	a_1	a_2	$\hat{y}$
0	0	0	0	0	0
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	0	0



projecting it to a_1-a_2 space →

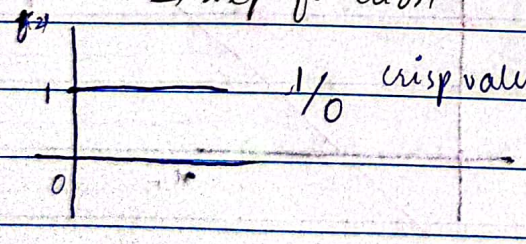


now linearly separable.

2 points here at single point.

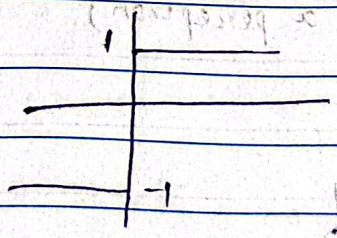
Activation functions:

→ step function



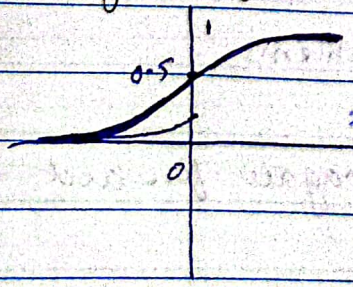
1/0 crisp value

→ sine func (Bipolar step)



1/-1 crisp value.

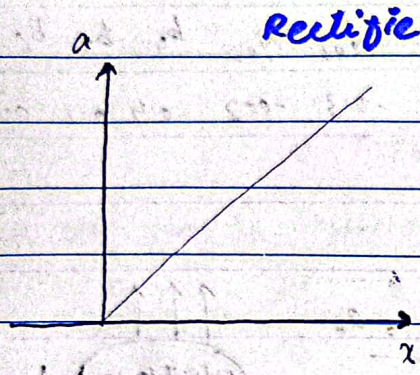
→ sigmoid function



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

continous range of probability.

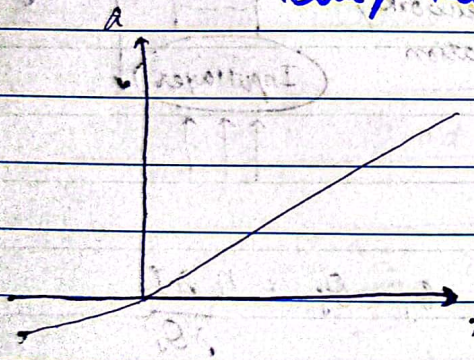
Rectified Linear (ReLU)



$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

→ only applied at the hidden layers. Not at output/input layers.

Leaky ReLU

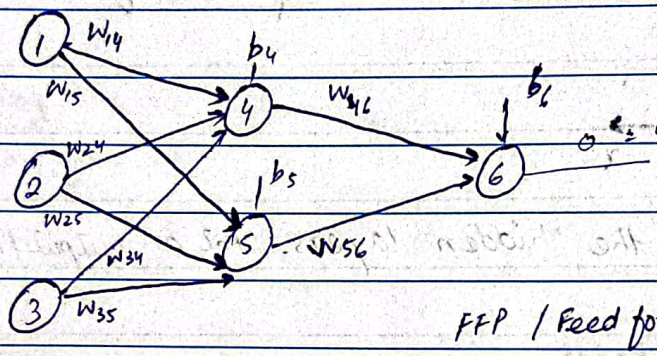


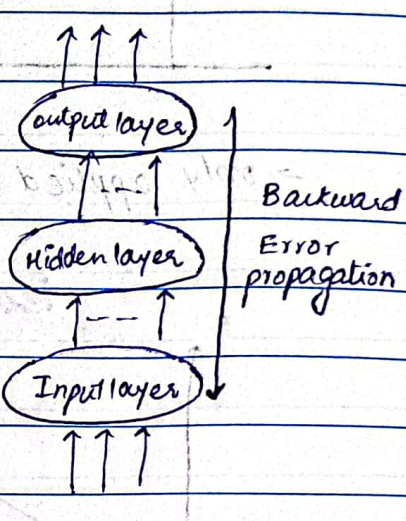
$$f(x) = \begin{cases} ax & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

Foundation for Advancement of Science & Technology

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	b_4	b_5	b_6	d
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1	0

i j k





FFP / Feed forward Network / FN activation

Delta Rule:-

$$w_{new} = w_{old} + \Delta w$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

$$E = \frac{1}{2} \sum_k (d_k - O_k)^2 = \frac{1}{2}(d-O)^2$$

$$\theta_j = \theta_j + \eta \frac{\partial J}{\partial \theta_j}$$

$$z_4 = w_{14}x_1 + w_{24}x_2 + w_{34}x_3 + b_4 = -0.7$$

$$a_4 = \sigma(z_4) = 0.331$$

$$z_5 = w_{15}x_1 + w_{25}x_2 + w_{35}x_3 + b_5 = 0.1$$

$$a_5 = \sigma(z_5) = 0.524$$

$$z_6 = w_{46}a_4 + w_{56}a_5 + b_6 = -0.104$$

$$O = a_6 = \sigma(z_6) = 0.473$$

$w_{jk} = w_{46}$

$j=4 \quad k=6$

$$\frac{\partial E}{\partial w_{46}} = \frac{\partial E}{\partial O} \times \frac{\partial O}{\partial z_6} \times \frac{\partial z_6}{\partial w_{46}}$$

$\partial \sigma(z)/\partial z$

$\sigma'(z) = \sigma(z)[1-\sigma(z)]$

$= O[1-O]$

θ = zero

$$\frac{\partial E}{\partial w_{46}} = -(d-O) \times O[1-O] \times a_4$$

$$= -(0-0.473) \times 0.473[1-0.473] \times 0.331$$

$$= 0.039$$

$c = 0.1$
learning rate

$W_{46\text{-new}} = W_{46\text{-old}} + \Delta W_{46}$

$= -0.3 - (0.1 \times 0.039)$

$= -0.3039$

W_{11}^1	W_{11}^2
①	① ①
②	②
③	

$W_{56\text{-new}} = W_{56\text{-old}} + \Delta W_{56}$

$\frac{\partial E}{\partial W_{56}} = \frac{\partial E}{\partial O} \times \frac{\partial O}{\partial Z_6} \times \frac{\partial Z_6}{\partial W_{56}}$

$= -(d-O) \times O[1-O] \times a_5$

$= 0.0617$

$W_{56_{new}} = -0.2 - (0.1 \times 0.0617) = -0.20617$

updated weights

$W_{46} = -0.3039$

$W_{56} = -0.20617$

$b_6 = 0.088$

$b_{6\text{-new}} = b_{6\text{-old}} + \Delta b_6 \qquad \Delta b_6 = -c \frac{\partial E}{\partial b_6}$

$\frac{\partial E}{\partial b_6} = \frac{\partial E}{\partial O} \times \frac{\partial O}{\partial Z_6} \times \frac{\partial Z_6}{\partial b_6}$

$= -(d-O) \times O[1-O] \times 1$

$= 0.117$

use old values for one whole iteration.

$b_{6\text{-new}} = 0.1 - (0.1 \times 0.117)$

$= 0.0883$

hidden layer:-

$W_{14\text{-new}} = W_{14\text{-old}} + \Delta W_{14}$

$\Delta W_{14} = -c \frac{\partial E}{\partial W_{14}}$

Backward Error propagation :

$\frac{\partial E}{\partial W_{14}} = \frac{\partial E}{\partial O} \times \frac{\partial O}{\partial Z_6} \times \frac{\partial Z_6}{\partial a_4} \times \frac{\partial a_4}{\partial Z_4} \times \frac{\partial Z_4}{\partial W_{14}}$

$= -(d-O) \times O[1-O] \times W_{46} \times a_4[1-a_4] \times X_1$

$= -(0-0.473) \times (0.473)[1-0.473] \times -0.3 \times 0.331[1-0.331] \times 1$

$\partial E / \partial w_{14} = -0.0078$

$w_{14\text{-old}} = 0.2 - (0.1 \times -0.0078) = 0.20078$

$b_{4\text{-new}} = b_{4\text{old}} + \Delta b_4$

$\frac{\partial E}{\partial b_4} = -(d-o) \times (o)(1-o) \times w_{46} \times a_4 \times (1-a_4) \times 1 = -0.0078$

$\Delta b_{4\text{-new}} = -0.4 - (0.1 \times -0.0078) = -0.39922$

$\frac{\partial E}{\partial w_{15}} = \frac{\partial E}{\partial o} \times \frac{\partial o}{\partial z_6} \times \frac{\partial z_6}{\partial a_5} \times \frac{\partial a_5}{\partial z_5} \times \frac{\partial z_5}{\partial w_{15}}$

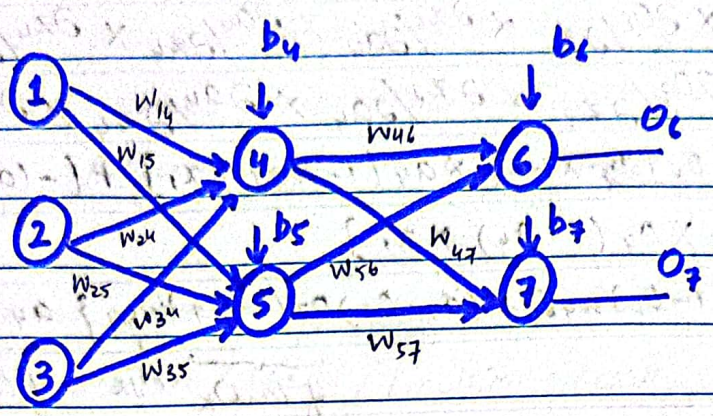
$= -(d-o) \times o(1-o) \times w_{56} \times a_5[1-a_5] \times x_1$

$= -(0-0.473) \times 0.473(1-0.473) \times (-0.2) \times 0.524 \times [1-0.524] \times 1$

$= -5.882 \times 10^{-5}$

$= -0.3 - 0.1(-5.88 \times 10^{-5})$

$w_{15,\text{new}} = -0.299$



$z_4 = w_{14}x_1 + w_{24}x_2 + w_{34}x_3 + b_3$

$a_4 = \sigma(z_4) = f(z_4)$

$z_5 = w_{15}x_1 + w_{25}x_2 + w_{35}x_3 + b_3$

$a_5 = \sigma(z_5)$

$z_6 = w_{46}a_4 + w_{56}a_5 + b_6$

$O_6 = \sigma(z_6)$

$z_7 = w_{47}a_4 + w_{57}a_5 + b_7$

$O_7 = \sigma(z_7)$

Output layer weights

Error in this case

$E = \frac{1}{2}\sum_k (d_k - O_k)^2$

$= \frac{1}{2}(d_6 - O_6)^2 + \frac{1}{2}(d_7 - O_7)$

$= E_6 + E_7$

- $w_{46\text{-new}} = w_{46\text{-old}} + \Delta w_{46}$

$\Delta w_{46} = -c\,\partial E/\partial w_{46}$

$\partial E_6/\partial w_{46} = \partial E/\partial O_6 \times \partial O_6/\partial z_6 \times \partial z_6/\partial w_{46}$

$\partial E/\partial w_{46} = \partial(E_6 + E_7)/\partial w_{46} = \partial E_6/\partial w_{46} + \partial E_7/\partial w_{46}$

$\partial E_7/\partial w_{46} = \partial E_7/\partial O_7 \times \partial O_7/\partial z_7 \times \partial z_7/\partial w_{46} = 0$

$\partial E/\partial w_{46} = \partial E_6/\partial w_{46} = -(d_6 - O_6) \times O_6[1 - O_6] \times a_4$

(For output layer)

Generalized Equations

$\Delta w_{jk} = -c\,\partial E/\partial w_{jk}$

$= +c[-(d_k - O_k) \times f'(act)_k \times x_j]$

(x_j: input to that node)

- $\partial E/\partial w_{47} = -(d_7 - O_7) \times O_7(1 - O_7) \times a_4$

- $\partial E/\partial w_{57} = -(d_7 - O_7) \times O_7(1 - O_7) \times a_5$

Hidden layer weights

$w_{14\text{-new}} = w_{14\text{-old}} + \Delta w_{14}$

$\Delta w_{14} = -c\,\partial E/\partial w_{14}$

$\partial E/\partial w_{14} = \partial E_6/\partial w_{14} + \partial E_7/\partial w_{14}$

$$\partial E_6/\partial w_{14} = \partial E_6/\partial o_6 \times \partial o_6/\partial z_6 \times \partial z_6/\partial a_4 \times \partial a_4/\partial z_4 \times \partial z_4/\partial w_{14}$$

$$\partial E_7/\partial w_{14} = \partial E_7/\partial o_7 \times \partial o_7/\partial z_7 \times \partial z_7/\partial a_4 \times \partial a_4/\partial z_4 \times \partial z_4/\partial w_{14}$$

$$\partial E/\partial w_{14} = [-(d_6-o_6)\,o_6(1-o_6) \times w_{46} \times a_4(1-a_4) \times x_1] + [-(d_7-o_7)\,o_7(1-o_7) \times w_{47} \times a_4(1-a_4) \times x_1]$$

$$= [-\{(d_6-o_6)o_6(1-o_6)w_{46} + (d_7-o_7)o_7(1-o_7)w_{47}\}\, a_4(1-a_4)x_1]$$

$\{\}$: $f'(\text{act})_k$ w_{jk} — $a_4(1-a_4)$: $f'(\text{act})_j$ x_i

Generalized Equation for weights of Hidden layer (single)

$$\Delta w_{ij} = -c\,\partial E/\partial w_{ij} = -c[-\Sigma_k\{(d_k-o_k)f'(\text{act})_k w_{jk}\} f'(\text{act})_j x_i]$$

- $\Delta b_6 = -c[-(d_6-o_6)o_6(1-o_6) \times 1]$
- $\Delta b_4 = -c[-\{(d_6-o_6)o_6(1-o_6)w_{46} + (d_7-o_7)o_7(1-o_7)w_{47}\} a_4(1-a_4)]$

Gradient Descent Algorithm

used during backward propagation to update weights

Batch Gradient Descent	Stochastic Gradient Descent (SGD)	Mini-batch Gradient Descent
(whole batch)	(only 1 updated)	(a specific size of batch)

ϵ (threshold) $> E$ → stopping condition

or for a specific # of times.

$2+2+[3\times2]+[2\times2] = 14$ times weight updated in prev example (2 biases)

Epoch: entire training set is passed through the model, forward & backward propagation are performed & parameters are updated.

Overfitting: gives accurate prediction for training data but not for new data

Underfitting: model cannot determine meaningful relationship b/w input & output data. (performs bad on both training & testing data)

F1-score: harmonic mean of precision & Recall

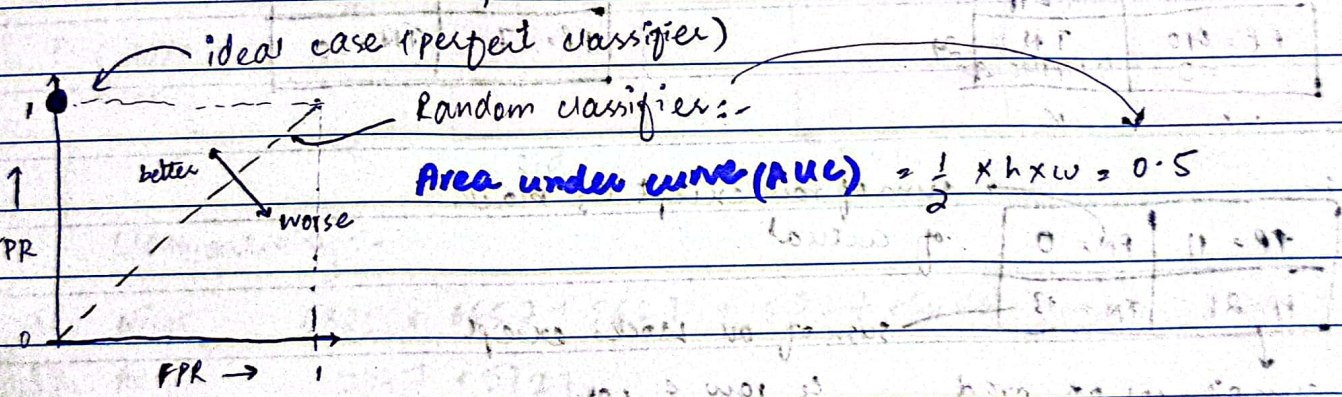
F1-score = 2 * Precision * Recall / Precision + Recall. 0 — 1 ↑ better

Recall: True positive Rate = TP / TP + FN

Specificity: True negative Rate = TN / TN + FN

False positive Rate: 1 - Sp = FP / FP + TN 0 — 1 ↓ better

ROC curve: Recieved operated characteristic curve



Accuracy: correctly classified

Missed classification Rate (error): 1 - Accuracy

In case of multi-classes: N X N

	Pre C_1	Pre C_2	Pre C_3
Actual C_1	1	1	1
Actual C_2	1	2	0
Actual C_3	0	1	1

True positives.

sample	y	ŷ
1	2	2
2	1	2
3	1	3
4	3	3
5	2	1
6	3	2
7	1	1
8	2	2

Accuracy = 4/8 = 0.5

Foundation for Advancement of Science & Technology

	pred		
actual	5	7	4
	0	1	17
	0	0	11

Acc = 5+11+1 / 5+7+4+1+17+11 = 17/45

False Negative = // First we need class wise values.

Recall = TP/TP+FN

class-wise calculation:-

C1 :- setosa vs all,

	pred +ve	pred -ve
actual +ve	TP = 5	FN = 7+4 = 11
actual -ve	FP = 0+0 = 0	TN = 1+17+11+0 = 29

C2 :- versicolor vs all

TP = 1		FN = 17
FP = 7+1		TN = 20

C3 :-

TP = 11	FN = 0
FP = 21	TN = 13

FN = 0 → sum of row except C_3 block of actual

TN = 13 → sum of all blocks except C_3 row & col.

FP = 21 → sum of col of pred except C_3 block

pre1, Recall1, pre2, ...

then average of all this for overall model's values:-

Mauro. average = (pre1 + pre2 + pre3) / 3 = pre.

(class imbalance case).

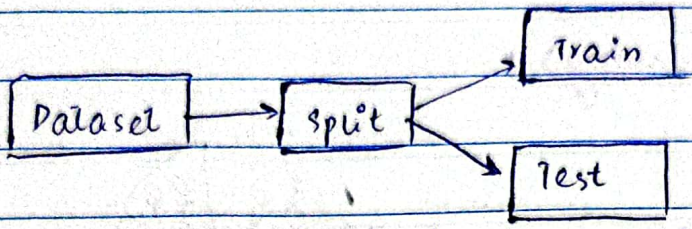
same weight to all classes. (for balanced data)

same weight to all samples = micro. average :-

$$pre = \frac{TP_1 + TP_2 + TP_3}{TP_1 + TP_2 + TP_3 + FP_1 + FP_2 + FP_3}$$

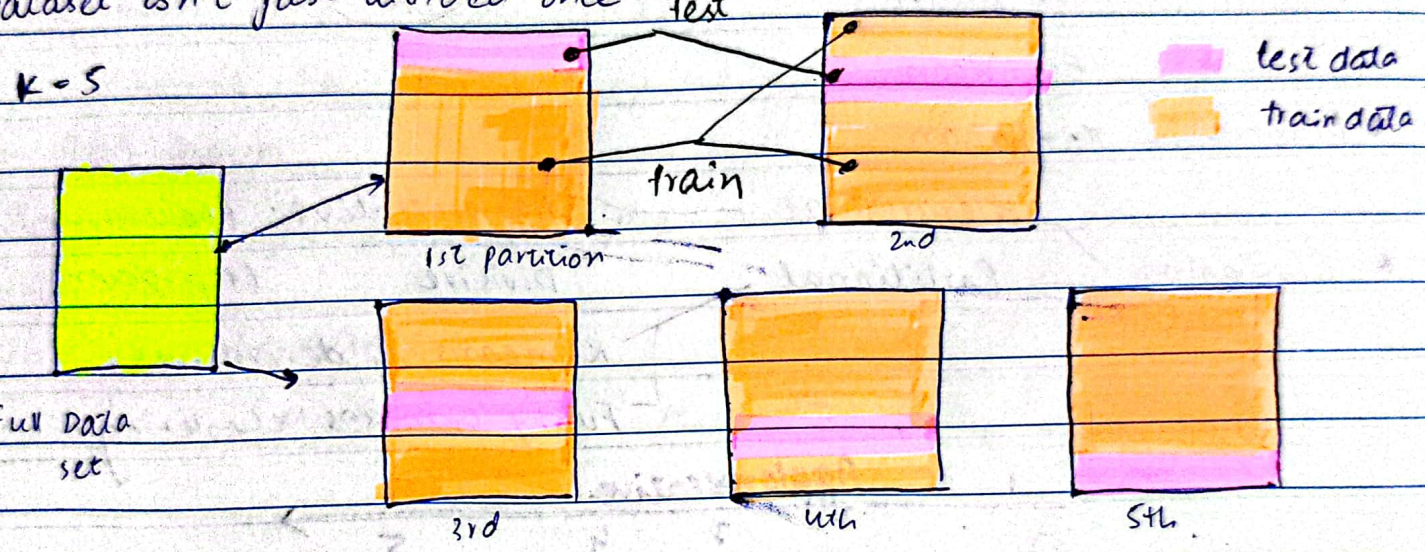
Foundation for Advancement of Science & Technology

Train test split :-



Cross-validation:- K-fold cross-validation.

Dataset isn't just divided once.



Class activity :-

TP_1 Micro = $\frac{5825 + 6657 + 5627 + 5027 + 5480 + 5095 + 5789 + 5872 + 5447 + 5727}{57346 + 9885 + 3731 + 3039 + 3562 + 3326 + 3129 + 3931 + 4044 + 3222}$

$= \frac{57346}{60000}$

$= 0.9557$

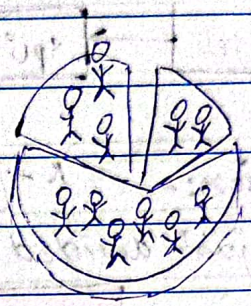
TP_2 Avg

TP_3 Precision

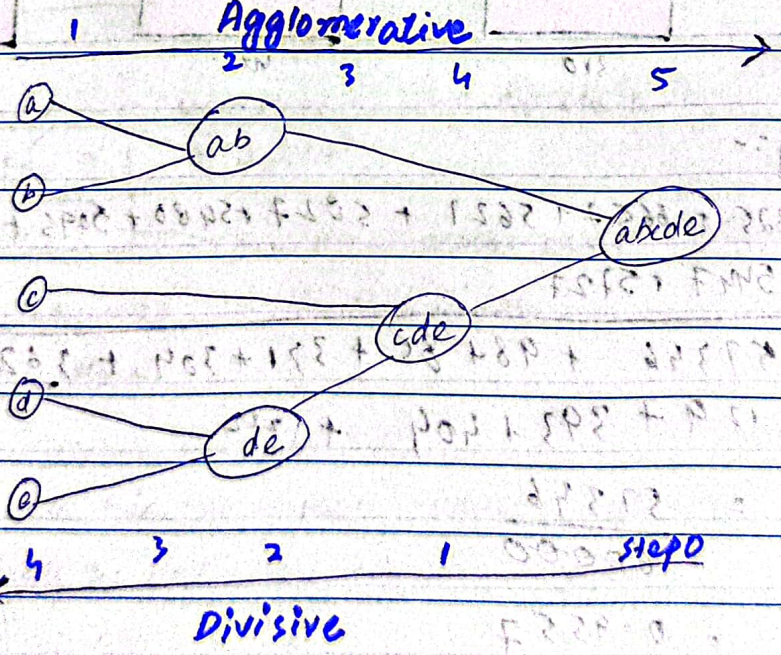
Foundation for Advancement of Science & Technology

Clustering (unsupervised learning):

1. Market Segmentation
2. Social Network Analysis
3. Astronomical Data Analysis
4. Organize computer clusters



- Distance — Euclidean, Manhattan
- Clustering — Hierarchical — Agglomerative (bottom-up), Divisive (top-down)
- Clustering — Partitional — K-means & derivatives, Fuzzy c-means clustering



K-means algorithm: cluster n objects based on attributes into k-partitions, where $k < n$.

↓ +ve integer.

→ minimize the sum of square of distances b/w data & corresponding cluster centroid to do grouping.

→ First we randomly pick 2 centroids (will be predefined) # of centroids. calculate distance of each data point from the centroids & group them to the lesser distance centroid point.

→ re-calculate the centroids by taking average of data points in that group.

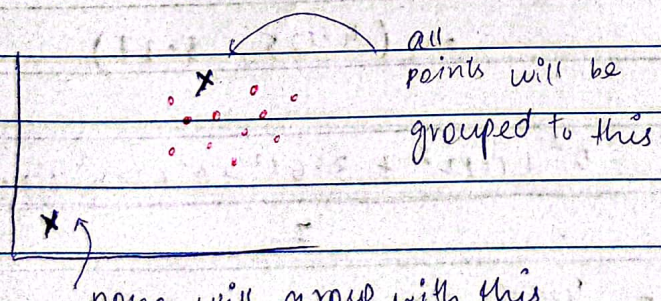
→ Repeat this process until the centroid point does not change anymore.

Optimization objective:-

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_k) = \frac{1}{m} \sum \| x^{(i)} - \mu_{c(i)} \|^2$$

When doing Random selection:

$$\frac{\sum_i \|dist\|^2}{m} = \sum \left(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \right)$$



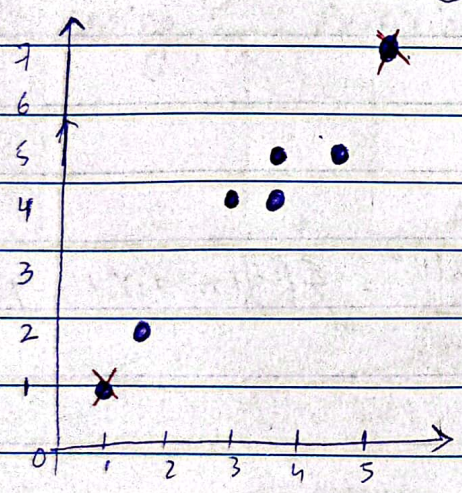
To overcome this issue, we randomly select centroids from the data points themselves.

Foundation for Advancement of Science & Technology

$\mu_1 = (1,1)$ $\quad \mu_1 = (1.83, 2.33)$

$\mu_2 = (5,7)$ $\quad \mu_2 = (4.12, 5.37)$

	x	y	d_1	d_2	d_1	d_2	d_1	d_2
1	1	1	0 ①	7.2	1.56 ①	5.36	0.55	5.02
2	1.5	2	1.118 ①	6.103	0.46 ①	4.26	0.55	3.92
3	3	4	3.605 ①	3.605	2.03	1.76 ②	3.05	1.42
4	5	7	7.21	0 ②	5.64	1.85 ②	6.6	2.19
5	3.5	5	4.72	2.5 ②	3.14	0.72 ②	4.16	0.41
6	4.5	5	5.3	2.06 ②	3.77	0.5302 ②	4.77	0.60
7	3.5	4.5	4.3	2.92 ②	2.73	1.068 ②	3.75	0.72



$$\mu_{1new} = \left(\frac{1+1.5+3}{3}, \frac{1+2+4}{3}\right)$$

$$= (1.83, 2.33)$$

$$\mu_{2new} = \left(\frac{5+3.5+4.5+3.5}{4}, \frac{7+5+5+4.5}{4}\right)$$

$$= (4.125, 5.37)$$

Cost / error of 1st iteration $= \frac{0^2 + 1.12^2 + 3.61^2 + 0^2 + 2.5^2 + 2.06^2 + 2.92^2}{7}$

$J = 4.75$

$$\mu_{1new} = \left(\frac{1+1.5}{2}, \frac{1+2}{2}\right) = (1.25, 1.5)$$

$$\mu_{2new} = \left(\frac{3+5+3.5+4.5+3.5}{5}, \frac{4+7+5+5+4.5}{5}\right)$$

2 $\quad (3.9, 5.1)$ $\quad J = 1.58$

Foundation for Advancement of Science & Technology

k-mean algorithm → sub-optimal results → because of random initialization of centroids.

k-means ++ : picks initial centroids intelligently.

$c_1 = (1,1)$ $c_2 = (5,7)$

	x	y	d_1	d_2
1	1	1	0	7.21
2	1.5	2	1.12	6.1
3	3.0	4	3.61	3.61
4	5	7	7.21	0
5	3.5	5	4.72	2.5
6	4.5	5	5.31	2.06
7	3.5	4.5	4.3	2.92

next centroid will be the one having max distance.