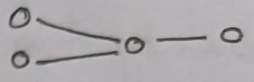
Module-1

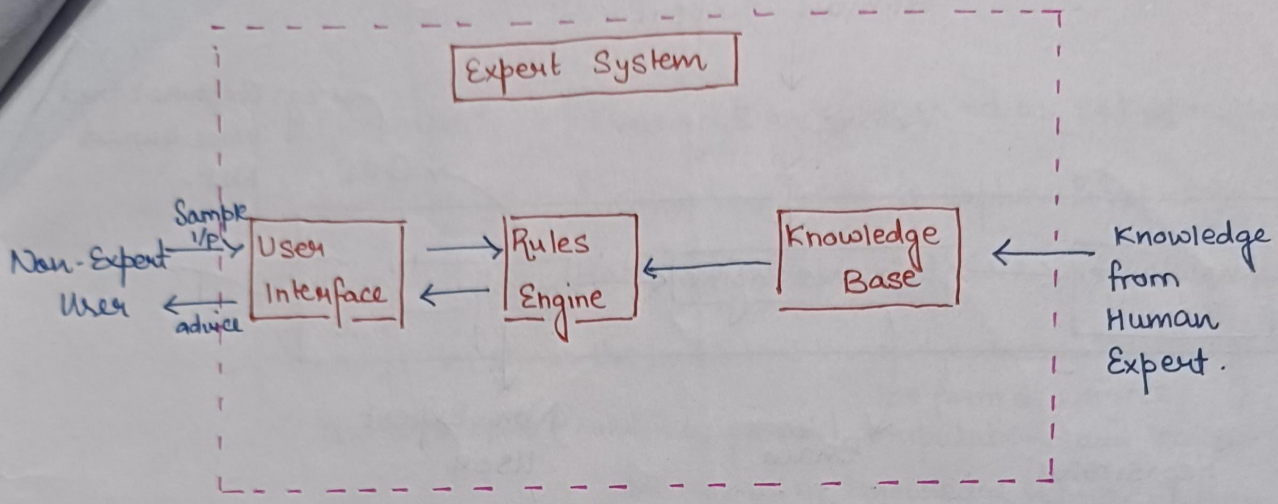
Introduction to Expert Systems,

- Knowledge Based agent,
  - In Artificial Intelligence, KB agent is an intelligent entity that requires an understanding of the world to make decisions and reason effectively.
  - These agents have the ability to maintain, reason over and update their internal knowledge based on observations and to take their actions accordingly.
  - Consists of two primary components,
    - Knowledge Base → set of sentences that encapsulates the facts about the world.
    - Inference Engine → applies logical rules to the KB to deduce new info & generate new facts.
  - Capabilities,
    - Ability to represent state, actions and more.
    - Capacity to integrate new perceptions.
    - Ability to update its internal world representation.
    - Capability to infer from its internal world representation.
    - Ability to deduce simple actions.
  - Performs 3 operations,
    - TELL → agent's perception of the environment.
    - ASK → queries KB about the actions it should perform.
    - PERFORM → executes the selected action.

* There are four ways of Knowledge representation,

- Logical Representation → P(x,y): x . . . y
- Semantic Network Representation → 
- Frame Representation → 
- Production Rules → IF X
  THEN Y

EXPERT SYSTEMS, {Prof. Edward Feigenbaum (1965)

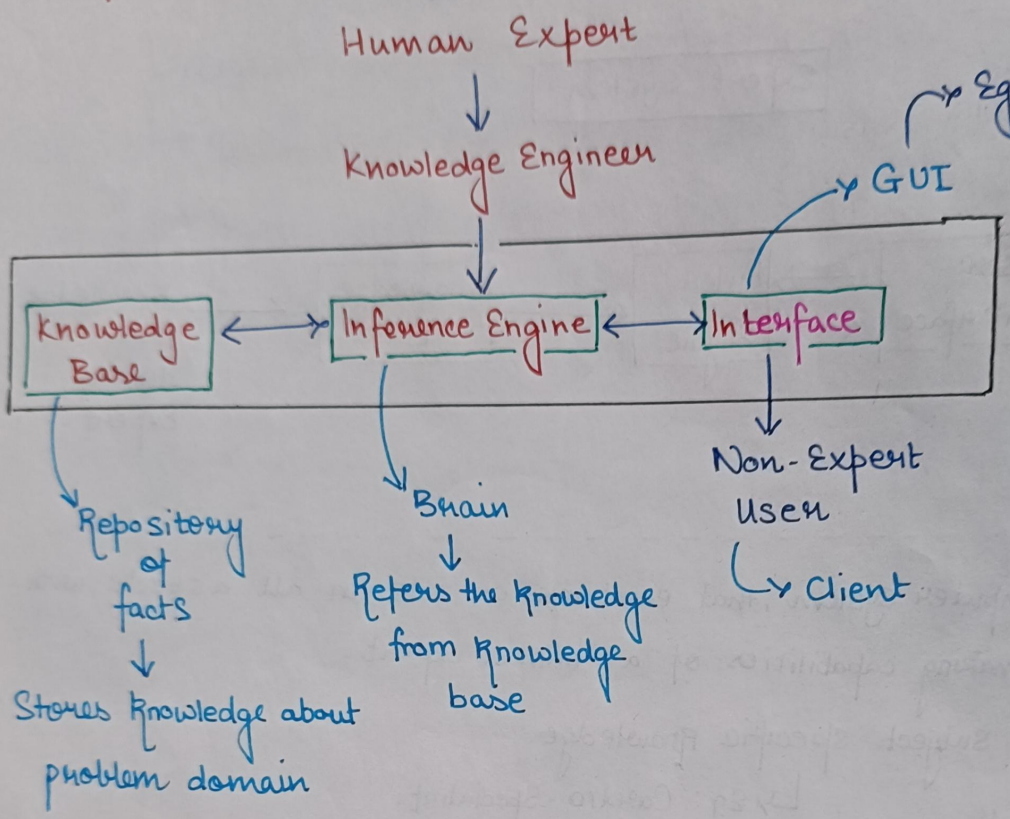


- It is a computer system, that emulates or acts in all aspects, with the decision making capabilities of a human expert.
- It contains subject specific knowledge.
  - Eg: Cardio-Specialist.
    Eye-Specialist.
- It can solve most complex issues in a specific domain.
- Eg: A recommendation system integrated with the knowledge base will provide better results.

* Characteristics,

- Can explain their reasoning or suggested decisions.
- Can display intelligent behaviours
- Can draw conclusions from complex relationships.
- Can provide portable knowledge.
- Can deal with uncertainity.

Components :



Steps	Stages
1. Identify Problem Domain	→ Identification
2. Design the System	→ Conceptualisation
3. Develop the prototype	→ Formalisation (Designing)
4. Test the prototype	→ Implementation
5. Update Knowledgebase	→ Testing {validation, verification & Maintainance}

egories of Expert Systems,

- 5 types,
  - Rule-based → {Knowledge is represented as set of rules}
  - Frame-based → {frames are used to capture & represent knowledge}
  - fuzzy → {differentiate b/w members of class from non-members}
  - neural → {stores the traditional knowledge as neural knowledge in the form of weights.
  - neuro-fuzzy → {combines parallel computation and learning abilities with knowledge representation & explanation abilities.

Propositional Logic,

- Analytical statement which is either TRUE or FALSE.
- Technique that represents the knowledge in logical and mathematical form.
- Two types,
  - Atomic. → Single Proposition symbol, eg: 2+2=4 → T
  - Compound. → Combining Atomic Propositions,
    eg: "It is raining (P) today. the street is wet (Q)".
    ∴ P → Q
- Note → ALL / SOME / NONE
  - Cannot be used.

First Order Logic,

- Also known as predicate Logic.
- Extension of propositional logic.
- It contains variables that may be true or false depending on the values of these variables.

↳ Example,

↳ All men are mortal, Socrates is a man.

∴ Socrates is mortal

↳ She lives in a city.

P(x,y) : x lives in y {x, y are predicate variable}

P(Anshika, Lucknow) is a proposition

Anshika lives in Lucknow.

* Strategy of I.E for acquiring Knowledge,

↳ Forward Chaining

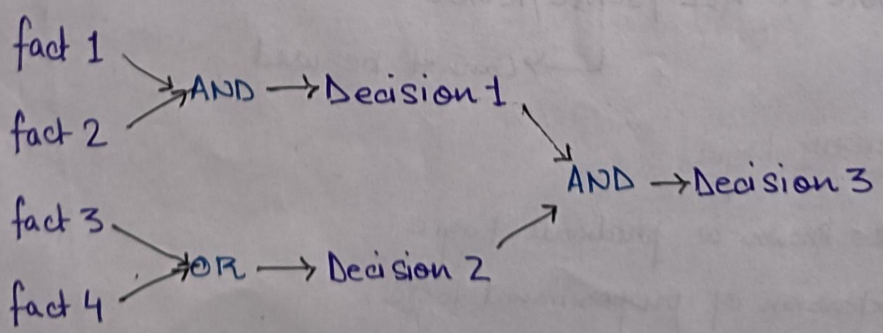
↳ Backward Chaining

* Forward Chaining, → used for planning, monitoring, controlling and interpreting apps.

↳ It is a strategy of an expert system to answer the question {WHAT CAN HAPPEN NEXT ?}

↳ Here, the inference engine follows the chain of conditions and derivations and finally deduces the outcomes.

↳ It considers all the facts & rules, and sorts them before concluding to a solution.

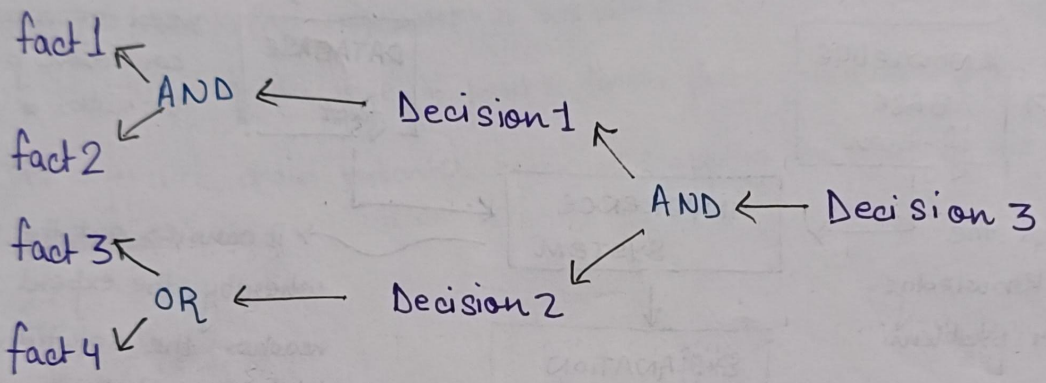


↳ Eg: It is raining. If it is raining, then the street is wet. The street is wet.

↳ Dendral uses FC to create chemical structures.

Backward Chaining,

- An I.E using backward chaining would search the inference rules until it finds one which has a consequent (THEN) that matches a desired goal.
- It starts with the goal and works backward to determine what facts or rules are needed to reach the goal.
- It checks whether the goal can be reached by the existing KB.

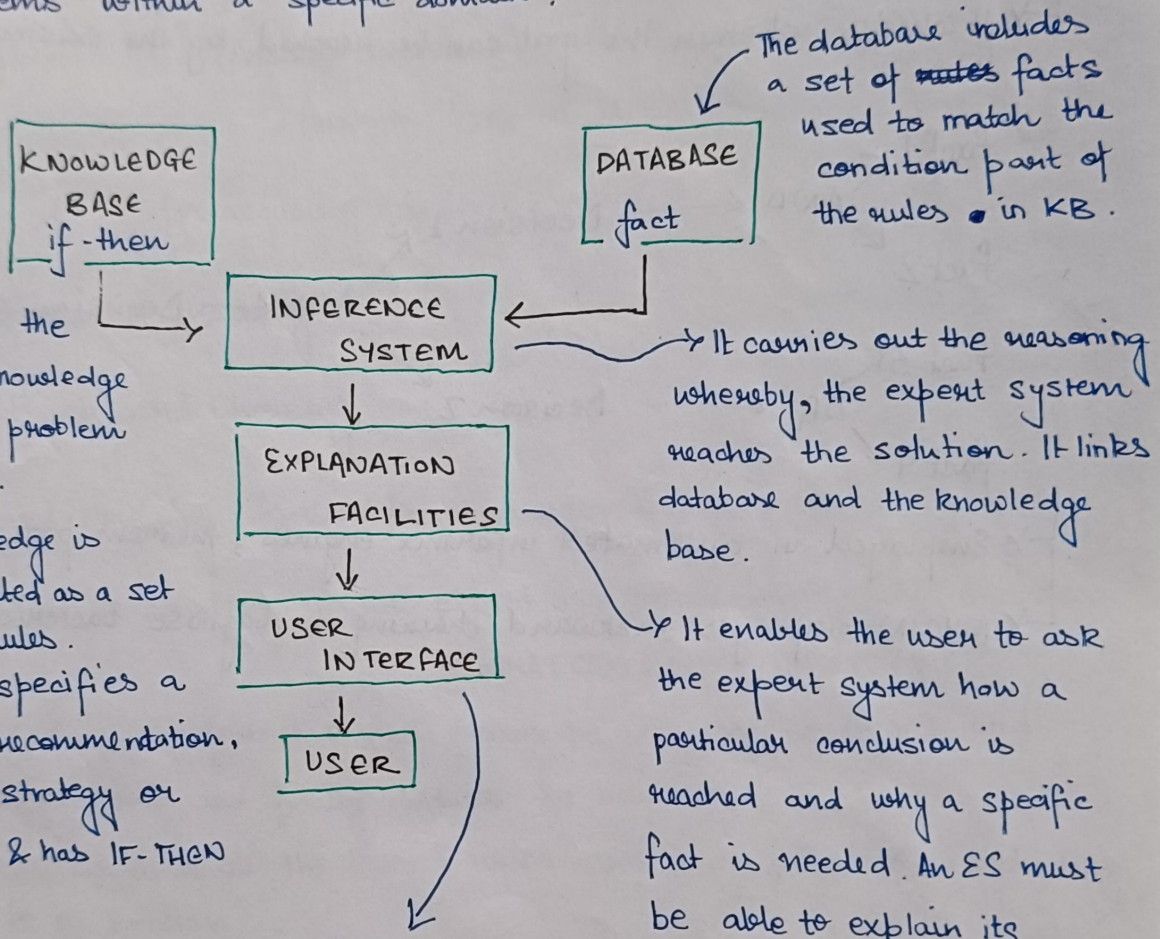


- Employed in automated inference engines, theorem proofs, etc.
- MYCIN employes backward chaining to diagnose bacterial infections.

→ Module 2

* Rule based expert Systems,

↳ A rule-based expert system is a type of artificial intelligence system that utilizes a set of pre-defined rules to make decisions or solve problems within a specific domain.



- Relation → IF fuel tank is empty, THEN the car is dead.
- Recommendation → IF the season is autumn AND the sky is cloudy THEN advice is take an umbrella
- Directive → IF the car is dead AND the fuel tank is empty THEN refuel the car.
- Strategy → IF X, THEN S1 is complete. IF S1 is complete AND Y THEN Z S2 is
- Heuristic → IF the spill is liquid, AND the spill PH < 6, AND the spill smell is vinegar Complete. THEN the spill material is acetic acid.

Rule Conflict,

- In a rule based expert system, the domain knowledge is represented by a set of IF-THEN Rules and data is represented by a set of facts and the current situation.
- The inference engine ~~compares~~ compares each rule stored in the knowledge base with facts contained in the database.
- If the IF (conditions) part of the rule matches a fact, the rule is fired and its THEN (action) part is executed.
- The matching of the rule IF parts to the facts produces inference chains.
- The inference chain indicates how an ES applies the rules to reach a conclusion.

Rule 1,

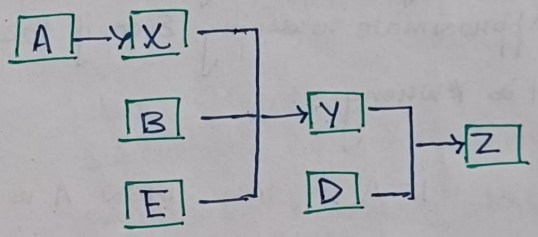
IF Y is true
AND D is true
THEN Z is true

Rule 2,

IF X is true
AND B is true
AND E is true
THEN Y is true

Rule 3,

IF A is true
THEN X is true



Example,

* **Rule 1**
  - IF the traffic light is green
  - THEN the action is go.
* **Rule 2**
  - IF the traffic light is red
  - THEN the action is stop.
* **Rule 3**
  - IF the traffic light is red
  - THEN the action is go.

* The inference engine must determine which rule to fire from such a set.
* A method for choosing a rule to fire when more than one rule can be fired in a given cycle is called conflict resolution.

Methods,

- Fire the rule with the highest priority.
- Fire the most specific rule.
- Fire the rule that uses the data most recently entered in the database.
- META knowledge.

- ↳ UK Know
- ↳ Weak Implication
- ↳ Impressive Know.
- ↳ Contra Opinion.

→ Uncertainity Management,

↳ Information can be incomplete, inconsistent, uncertain or all three at their same time. In other words, information is often unsuitable for solving a problem.

↳ Uncertainity is defined as the lack of the exact knowledge that would prevent us from reaching the exact or reliable conclusion.

↳ Existing classical logic system permits only exact reasoning, not the approximate reasoning. Even if the system is giving 99.99% accuracy, it is wrong.

IF A is true THEN A is not false
IF A is false THEN A is not true

Rules,

cf(P1 and P2) = min(cf(P1), cf(P2))

cf(P1 or P2) = max(cf(P1), cf(P2))

if P1 & P2 then : cf = C
cf(P2) = cf(P1) * C

→ Ways to compute certainity factor,

↳ Bayesian Reasoning,

↳ IF H is true
THEN E is true {with probability p}

↳ H represents hypothesis.

↳ Use of confidence factor,

↳ cf/mb {measure of belief}

↳ Used to increase the confidence of result when more and more facts are provided.

FRAME BASED EXPERT SYSTEMS, Marvin Minsky [1970s]

A frame based expert systems employ a knowledge representation technique based on frames, which are heirarchically organised data structures.

Inheritance mechanism play a vital role in these systems, facilitating the propagation of attributes and behaviours from parent frames to child frames.

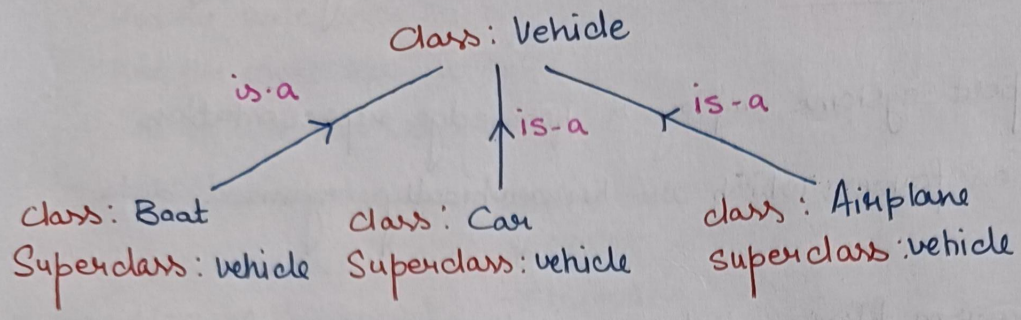
Each frames has its own name and a set of attributes associated with it.

The concept of frame is defined by collection of slots.
used to store values. ← Each slot describes a particular attribute or operation of the frame.

* Eg:

class: Passenger Car
Engine type: Inline 4-cylinder, V6:
Horsepower:
Drivetrain type: Rear wheel Drive
front wheel Drive
four wheel Drive
Transmission type:
5 speed Manual
4 speed automatic
fuel consumption (mpg):
Seating capacity:

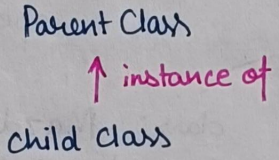
class: Mazda
Superclass: Passenger Car
Engine Type: Inline 4-cylinder, V6,
Horse Power:
Drive train type: Rear wheel drive
front wheel drive
four wheel drive
Transmission type:
5-Speed manual
4-speed automatic
fuel consumption:
Seating capacity:
Country of Manufacturing: Japan.



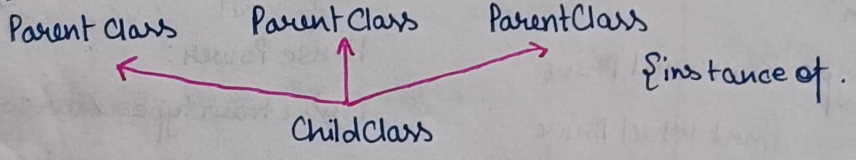
* Association,
  - describes semantic relationship b/w diff classes which are unrelated otherwise

* Aggregation,
  - Superclass represents a whole.

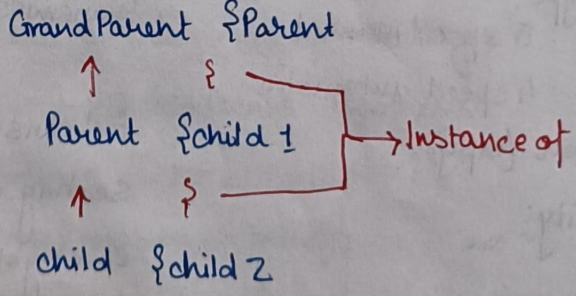
* Inheritance,
  - Single,



  - Multiple,



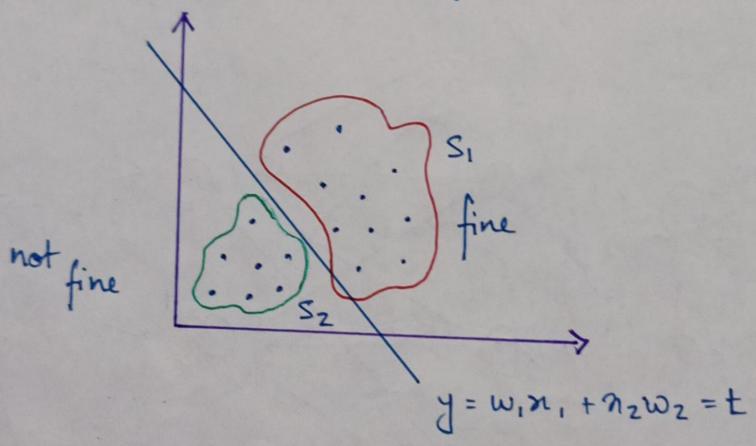
  - Multilevel,



Module-3,

Perceptron,

- It might include a learning rate, α.
- When the training data is linearly separable, the perceptron algorithm finds a successful weight.
- The perceptron is a binary classification algorithm that learns a linear decision boundary to separate two classes of data.



S_1, S_2 → linearly separable patterns.

$S_2 :: x_1w_1 + x_2w_2 \geq t$, $y = 1$ → fine

$S_1 :: x_1w_1 + x_2w_2 < t$, $y = 0$ → Not fine.

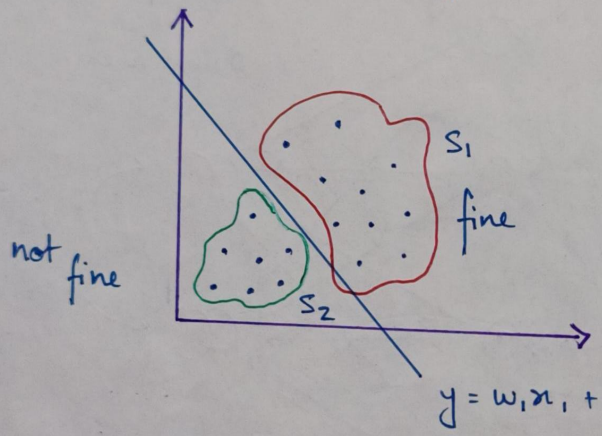
Steps,

1. Create a perceptron, {n+1 nodes}
   $x_0, x_1, x_2 \ldots x_n$
2. Initialise random weights,
   $w = (w_0, w_1, w_2 \ldots w_n)$
3. Compute total input,
   $I = \sum_{i=0}^{n} x_i w_i$
4. Compute o/p → y using activation function.
5. Compare computed output with target output.
6. Update the weights
7. Go to step 3.

Module-3,

Perceptron,

- It might include a learning rate, α.
- When the training data is linearly separable, the perceptron algorithm finds a successful weight.
- The perceptron is a binary classification algorithm that learns a linear decision boundary to separate two classes of data.



S_1, S_2 → linearly separable patterns.

$S_2 :: x_1w_1 + x_2w_2 \geq t$, $y = 1$ → fine

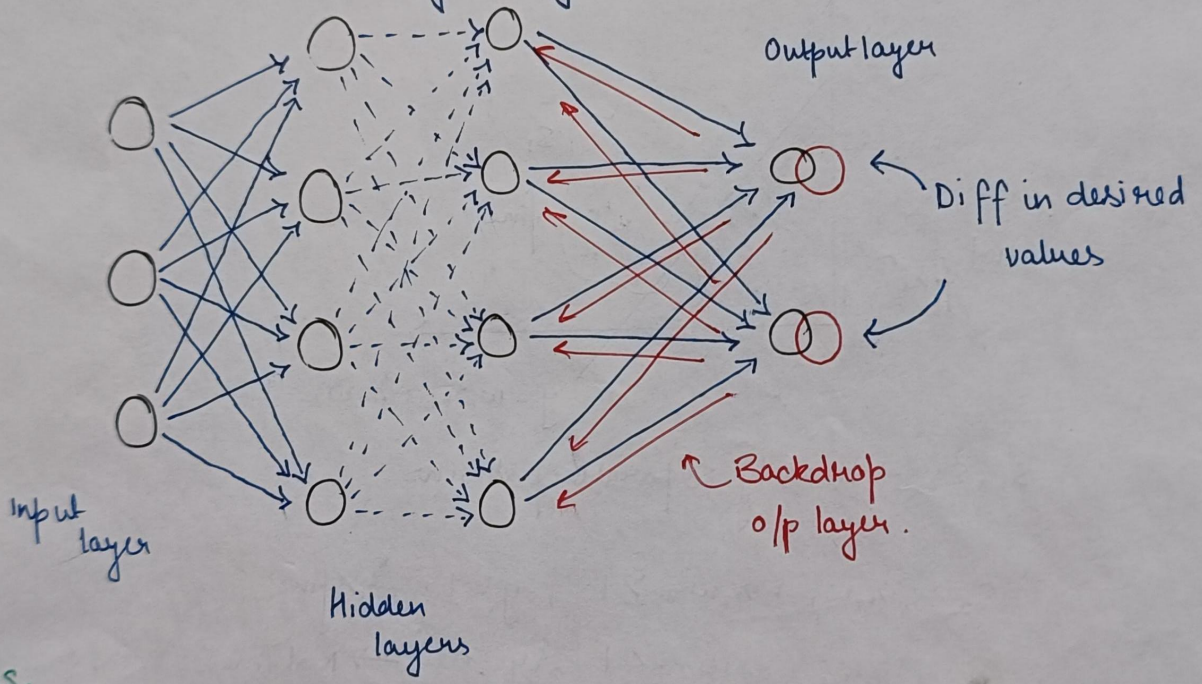
$S_1 :: x_1w_1 + x_2w_2 < t$, $y = 0$ → Not fine.

Steps,

1. Create a perceptron, {n+1 nodes}
   $x_0, x_1, x_2 \ldots x_n$
2. Initialise random weights,
   $w = (w_0, w_1, w_2 \ldots w_n)$
3. Compute total input,
   $$I = \sum_{i=0}^{n} x_i w_i$$
4. Compute o/p → y using activation function.
5. Compare computed output with target output.
6. Update the weights
7. Go to step 3.

→ Back Propagation,

- → Standard method of training artificial neural networks.
- → It is the method of fine tuning the weights of a neural net based on the error rate.
- → There are 3 phases,
  - → Phase 1 → Feed Forward
  - → Phase 2 → Back Propagation of error
  - → Phase 3 → Updating of weights.



Steps,

1. Input X arrive through the path.
2. Input is modelled using real weights {randomly selected}
3. Calculate the output for every neuron.
4. Calculate the error in the outputs.

   Error = Actual Output - Desired Output.
5. Back Propagate back to the hidden layer and adjust the weights.
6. Continue until desired output is achieved.