Overview of Fuzzy Logic

The notion central to fuzzy systems is that truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value on the range [0.0, 1.0].

0.0 represents absolute Falseness and 1.0 representing absolute Truth.

For example, let us take the statement:

"Jane is old."

If Jane's age was 75, we might assign the statement the truth value of 0.80.

Overview of Fuzzy Logic

The statement could be translated into set terminology as follows:

"Jane is a member of the set of old people."

This statement would be rendered symbolically with fuzzy sets as:

mOLD(Jane) = 0.80

Where m is the membership function, operating in this case on the fuzzy set of old people, which returns a value between 0.0 and 1.0.

Overview of Fuzzy Logic

Example:

Probabilistic approach:

"*There is an 80% chance that Jane is old*“

Fuzzy terminology:

“Jane's degree of membership within the set of old people is 0.80“

The semantic difference is significant:

Overview of Fuzzy Logic

Distinction between fuzzy systems and probability:

Both operate over the same numeric range, and at first glance both have similar values:

0.0 representing False (or non- membership),

and 1.0 representing True (or membership).

However, the probabilistic approach yields the natural-language statement while the fuzzy terminology corresponds to the membership within a set.

Overview of Fuzzy Logic

The next step in establishing a complete system of fuzzy logic is to define the operations of:

EMPTY,

EQUAL,

COMPLEMENT (NOT),

CONTAINMENT,

UNION (OR),

and INTERSECTION (AND).

The formal definitions for these operations are as follows:

Overview of Fuzzy Logic

Definition 1:

Let X be some set of objects, with elements noted as x. Thus, X = {x}.

Definition 2:

A fuzzy set A in X is characterized by a membership function mA(x) which maps each point in X onto the real interval [0.0, 1.0].

As mA(x) approaches 1.0, the "grade of membership" of x in A increases.

Overview of Fuzzy Logic

Definition 3:

A is EMPTY iff for all x, mA(x) = 0.0.

Definition 4:

A = B iff for all x: mA(x) = mB(x) [or, mA = mB].

Definition 5:

mA' = 1 - mA.

Definition 6:

A is CONTAINED in B iff mA <= mB.

Overview of Fuzzy Logic

Definition 7:

C = A UNION B, where: mC(x) = MAX(mA(x), mB(x)).

Definition 8:

C = A INTERSECTION B where: mC(x) = MIN(mA(x), mB(x)).

It is important to note the last two operations, UNION (OR) and INTERSECTION (AND), which represent the clearest point of departure from a probabilistic theory for sets to fuzzy sets.

Operationally, the differences are as follows:

Overview of Fuzzy Logic

For independent events, the probabilistic operation for AND is multiplication, which is counterintuitive for fuzzy systems.

For example: let us presume that x = Bob, S is the fuzzy set of smart people, and T is the fuzzy set of tall people.

Then, if mS(x) = 0.90 and uT(x) = 0.90,

the probabilistic result would be:

mS(x) * mT(x) = 0.81

whereas the fuzzy result would be:

MIN(uS(x), uT(x)) = 0.90

Overview of Fuzzy Logic

The probabilistic calculation yields a result that is lower than either of the two initial values, which when viewed as "the chance of knowing" makes good sense.

However, in fuzzy terms the two membership functions would read something like

"Bob is very smart" and "Bob is very tall.“

If we presume for the sake of argument that "very" is a stronger term than "quite," and that we would correlate "quite" with the value 0.81, then the semantic difference becomes obvious.

Overview of Fuzzy Logic

The probabilistic calculation would yield the statement:

If Bob is very smart, and Bob is very tall, then Bob is a quite tall, smart person.

The fuzzy calculation, however, would yield:

If Bob is very smart, and Bob is very tall, then Bob is a very tall, smart person.

Overview of Fuzzy Logic

The probabilistic calculation would yield the statement:

If Bob is very smart, and Bob is very tall, then Bob is a quite tall, smart person.

The fuzzy calculation, however, would yield:

If Bob is very smart, and Bob is very tall, then Bob is a very tall, smart person.

Fuzzy Reasoning - continued

The process of fuzzy reasoning is incorporated into what is called a Fuzzy Inferencing System.

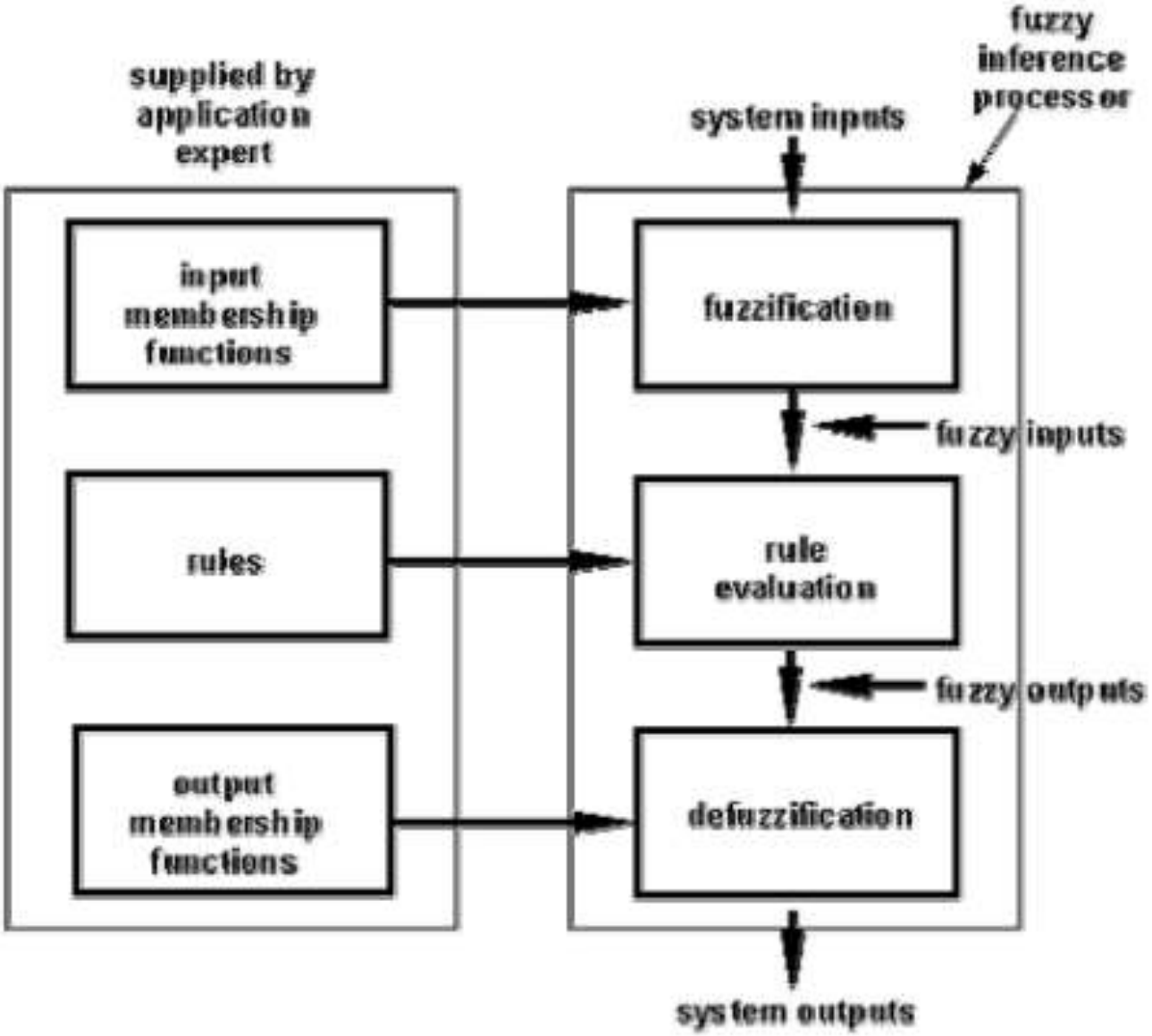
It is comprised of three steps that process the system inputs to the appropriate system outputs.

These steps are:

1) Fuzzification,
2) Rule Evaluation,
3) Defuzzification.

Fuzzy Reasoning - continued

Fuzzy Inferencing System is as follows:



Fuzzy Reasoning - continued

Fuzzification is the first step in the fuzzy inferencing process.

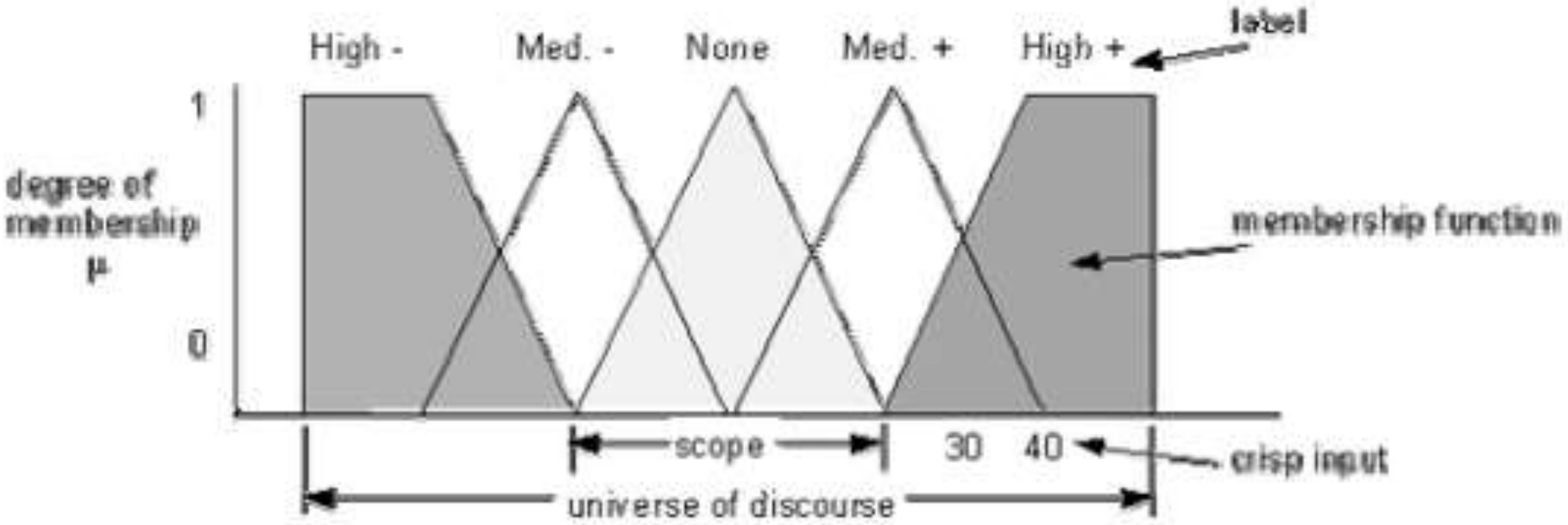
This involves a domain transformation where crisp inputs are transformed into fuzzy inputs.

Crisp inputs are exact inputs measured by sensors and passed into the control system for processing, such as temperature, pressure, rpm's, etc.

Each crisp input that is to be processed by the FIU has its own group of membership functions or sets to which they are transformed.

Fuzzy Reasoning - continued

Membership function structure:



degree of membership: degree to which a crisp value is compatible to a membership function, value from 0 to 1, also known as truth value or fuzzy input.

Fuzzy Reasoning - continued

membership function, MF: defines a fuzzy set by mapping crisp values from its domain to the sets associated degree of membership.

crisp inputs: distinct or exact inputs to a certain system variable, usually measured parameters external from the control system, e.g. 6 Volts.

label: descriptive name used to identify a membership function.

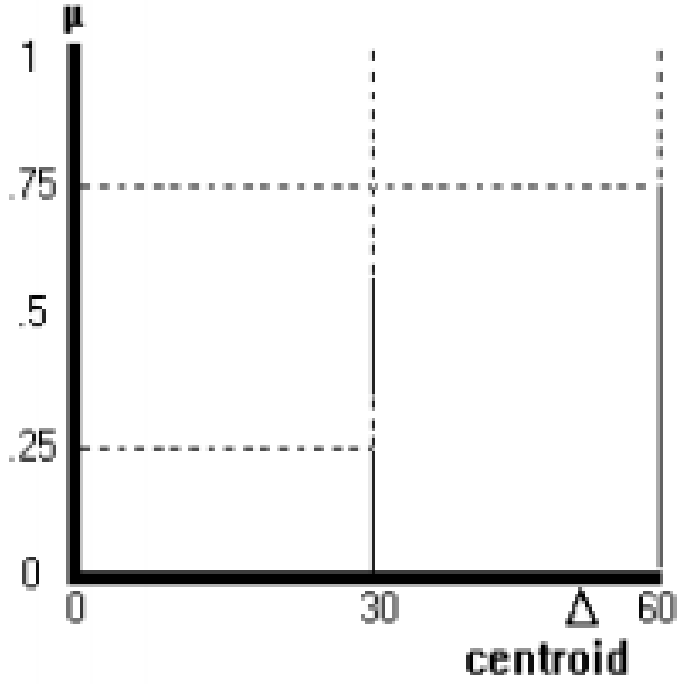
scope: or domain, the width of the membership function, the range of concepts, usually numbers, over which a membership function is mapped.

Fuzzy Reasoning - continued

Defuzzification involves the process of transposing the fuzzy outputs to crisp outputs.

A method of averaging is utilized here, and is known as the Center of Gravity method or COG, it is a method of calculating centroids of sets.

$$\text{Crisp Output (Y)} = \frac{\sum_i \left(\text{fuzzy output}_i\right) \times \left(\begin{matrix}\text{Singleton position} \\ \text{on x axis}_i\end{matrix}\right)}{\sum_i \left(\text{fuzzy output}_i\right)}$$



Fuzzy Reasoning - continued

The output membership functions to which the fuzzy outputs are transposed are restricted to being singletons.

This is so to limit the degree of calculation intensity in the microcontroller.

The fuzzy outputs are transposed to their membership functions similarly as in fuzzification.

With COG the singleton values of outputs are calculated using a weighted average.

The crisp output is the result and is passed out of the fuzzy inferencing system for processing elsewhere.

Fuzzy Reasoning - continued

Applications:

- A navigation system for automatic cars.
- A predicative fuzzy-logic controller for automatic operation of trains.
- Laboratory water level controllers.
- Controllers for robot arc-welders .
- Feature-definition controllers for robot vision.
- Graphics controllers for automated police sketchers.

Learning: Concept of learning

Machine Learning is the study of how to build computer systems that adapt and improve with experience.

It is a subfield of Artificial Intelligence and intersects with cognitive science, information theory, and probability theory, among others.

Classical AI deals mainly with deductive reasoning, learning represents inductive reasoning.

Classical AI often suffers from the knowledge acquisition problem in real life applications where obtaining and updating the knowledge base is costly and prone to errors.

Learning: Concept of learning

Machine learning is particularly attractive in several real life problem because of the following reasons:

- Some tasks cannot be defined well except by example
- Working environment of machines may not be known at design time .
- Explicit knowledge encoding may be difficult and not available .
- Environments change over time.
- Biological systems learn.

Learning: Concept of learning

Recently, learning is widely used in a number of application areas including:

- Data mining and knowledge discovery
- Speech/image/video (pattern) recognition
- Adaptive control
- Autonomous vehicles/robots
- Decision support systems
- Bioinformatics
- WWW

Learning: Concept of learning

Formally, a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Thus a learning system is characterized by:

- task T
- experience E, and
- performance measure P

Learning: Concept of learning

Examples:

Learning to play chess:

T: Play chess

P: Percentage of games won in world tournament

E: Opportunity to play against self or other players

Learning to drive a van:

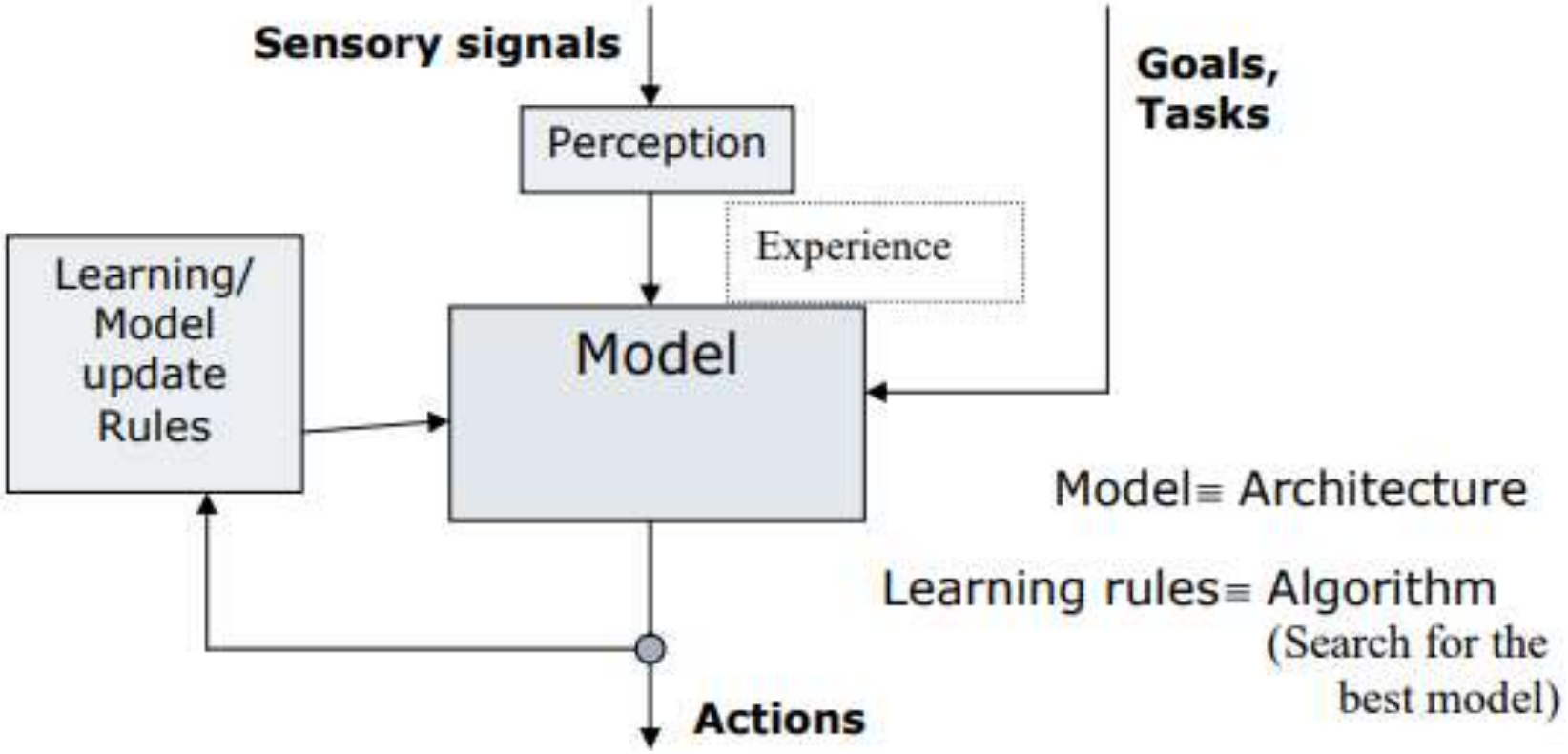
T: Drive on a public highway using vision sensors

P: Average distance traveled before an error (according to human observer)

E: Sequence of images and steering actions recorded during human driving.

Learning: Concept of learning

The block diagram of a generic learning system:



Learning: Concept of learning

As can be seen from the above diagram the system consists of the following components:

Goal: Defined with respect to the task to be performed by the system

Model: A mathematical function which maps perception to actions

Learning rules: Which update the model parameters with new experience such that the performance measures with respect to the goals is optimized.

Experience: A set of perception (and possibly the corresponding actions).

Learning Models

Propositional and FOL rules

Decision trees

Linear separators

Neural networks

Graphical models

Temporal models like hidden Markov models

Learning from experience

Learning algorithms use experiences in the form of perceptions or perception action pairs to improve their performance.

The nature of experiences available varies with applications. Some common situations are described below:

Supervised learning:

In supervised learning a teacher or oracle is available which provides the desired action corresponding to a perception. A set of perception action pair provides what is called a training set. Examples include an automated vehicle where a set of vision inputs and the corresponding steering actions are available to the learner.

Learning from experience

Unsupervised learning:

- In unsupervised learning no teacher is available.
- The learner only discovers persistent patterns in the data consisting of a collection of perceptions.
- This is also called exploratory learning.
- Finding out malicious network attacks from a sequence of anomalous data packets is an example of unsupervised learning.

Learning from experience

Active learning:

- Here not only a teacher is available, the learner has the freedom to ask the teacher for suitable perception-action example pairs which will help the learner to improve its performance.

- Consider a news recommender system which tries to learn an users preferences and categorize news articles as interesting or uninteresting to the user. The system may present a particular article (of which it is not sure) to the user and ask whether it is interesting or not.

Learning from experience

Reinforcement learning:

- In reinforcement learning a teacher is available, but the teacher instead of directly providing the desired action corresponding to a perception, return reward and punishment to the learner for its action corresponding to a perception.

- Examples include a robot in a unknown terrain where its get a punishment when its hits an obstacle and reward when it moves smoothly.

Learning Decision Tree

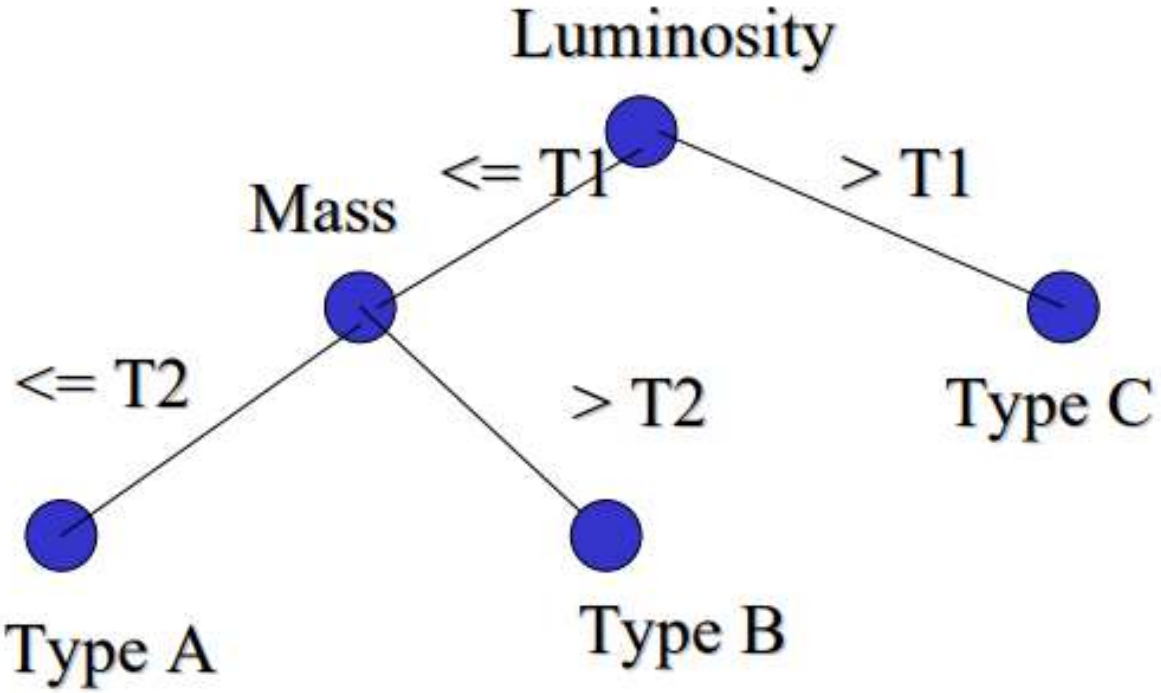
Decision trees are a class of learning models that are more robust to noise as well as more powerful as compared to concept learning.

A decision-tree learning algorithm approximates a target concept using a tree representation, where each internal node corresponds to an attribute, and every terminal node corresponds to a class.

Consider the problem of classifying a star based on some astronomical measurements.

Learning Decision Tree

It can naturally be represented by the following set of decisions on each measurement arranged in a tree like fashion.



Learning Decision Tree

There are two types of nodes:

1. Internal node: Splits into different branches according to the different values the corresponding attribute can take.
   Example: luminosity <= T1 or luminosity > T1.

2. Terminal Node.- Decides the class assigned to the example.

Learning Decision Tree

Decision trees adopt a DNF (Disjunctive Normal Form) representation.

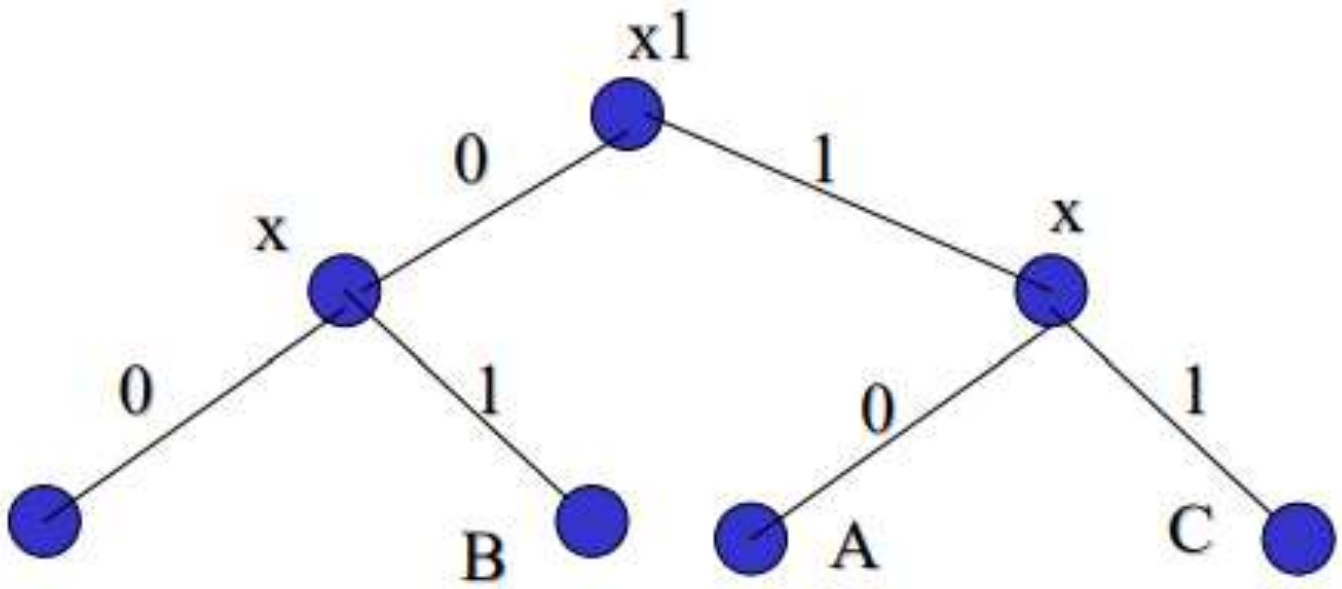
For a fixed class, every branch from the root of the tree to a terminal node with that class is a conjunction of attribute values.

Different branches ending in that class form a disjunction.

Learning Decision Tree

In the following example, the rules for class A are:

(~X1 & ~x2) OR (X1 & ~x3)



Learning Decision Tree

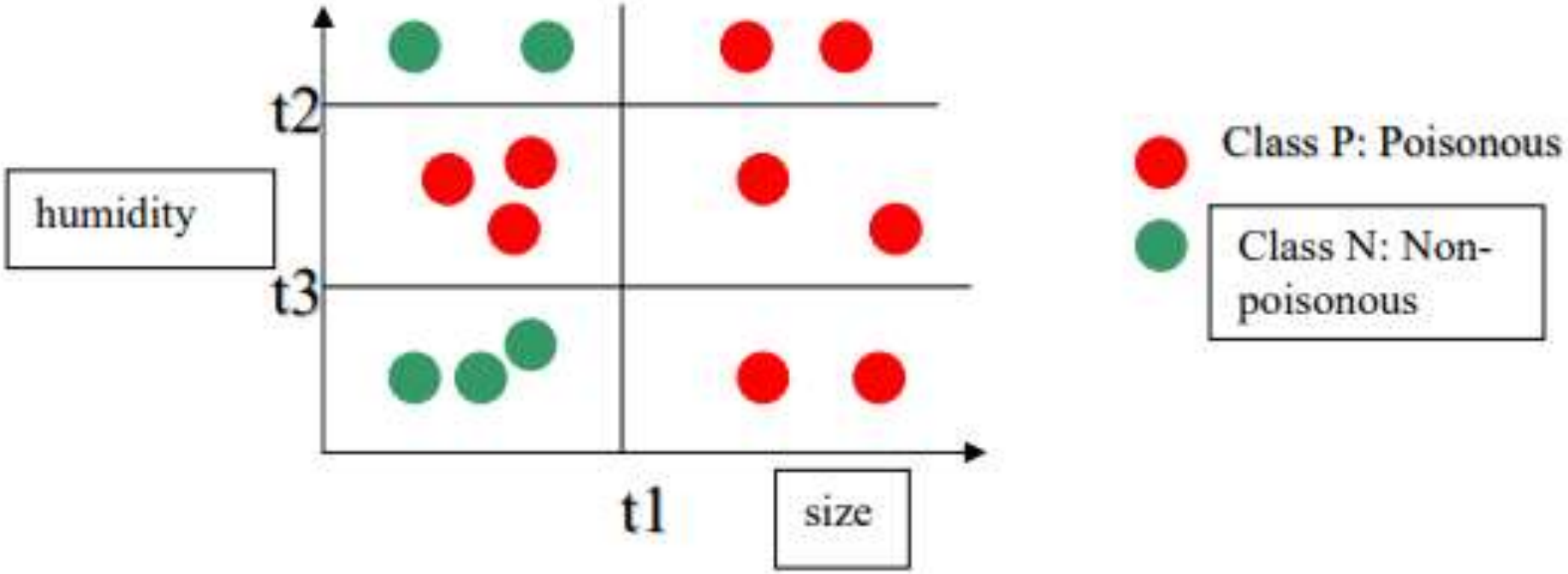
There are different ways to construct trees from data. We will concentrate on the top down, greedy search approach:

Basic idea:

1. Choose the best attribute a* to place at the root of the tree.
2. Separate training set D into subsets {D1, D2, .., Dk} where each subset Di contains examples having the same value for a*.
3. Recursively apply the algorithm on each new subset until examples have the same class or there are few of them.

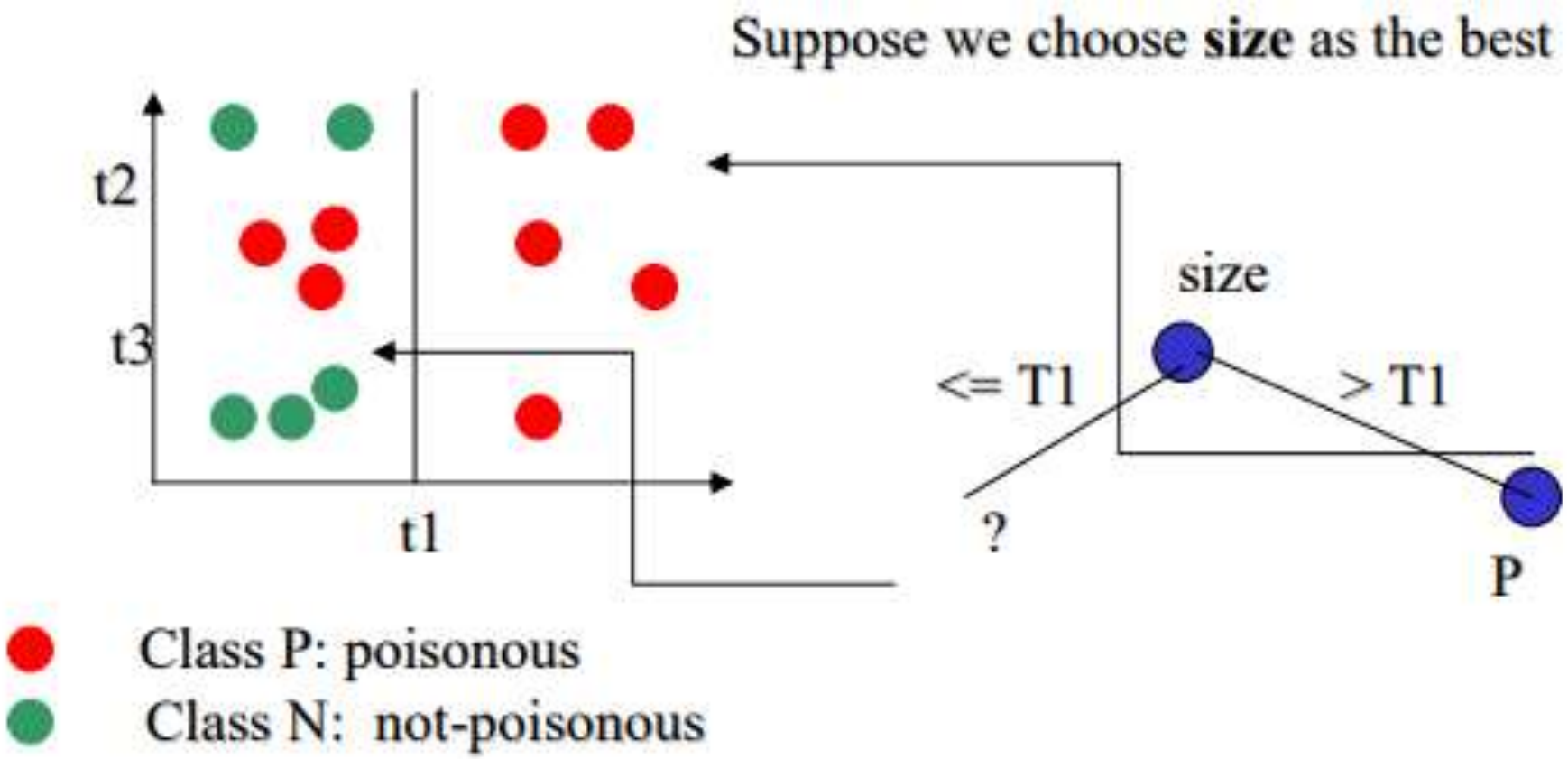
Learning Decision Tree

Illustration:

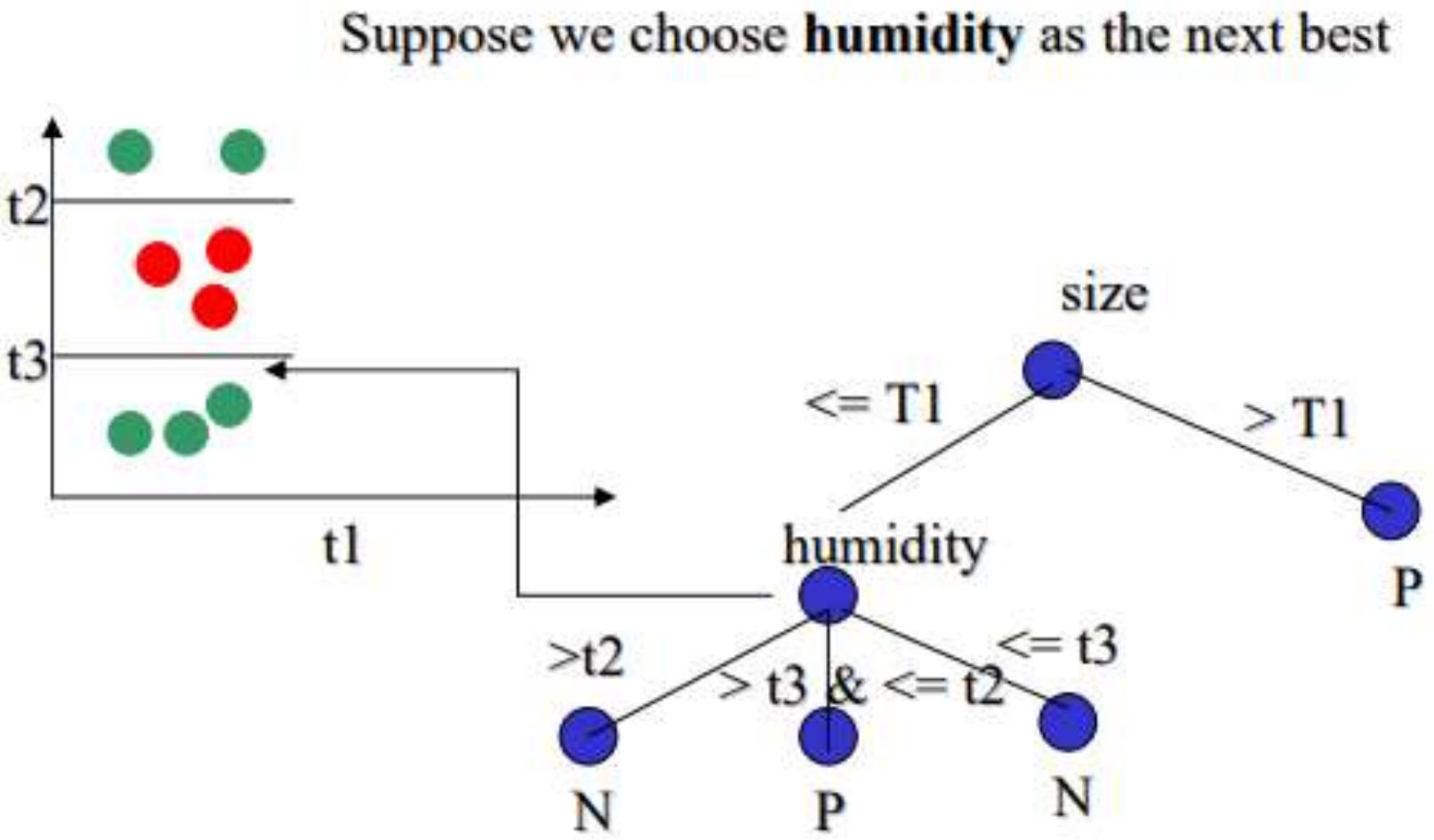


Attributes: size and humidity.
Size has two values: >t1 or <= t1
Humidity has three values: >t2, (>t3 and <=t2), <= t3

Learning Decision Tree



Learning Decision Tree



Learning Decision Tree

Steps:

- Create a root for the tree.
- If all examples are of the same class or the number of examples is below a threshold return that class.
- If no attributes available return majority class.
- Let a* be the best attribute.
- For each possible value v of a*:
  - Add a branch below a* labeled “a = v”
  - Let Sv be the subsets of example where attribute a*=v
  - Recursively apply the algorithm to Sv

Machine Learning Paradigms

Learning Paradigms basically state a particular pattern on which something or someone learns.

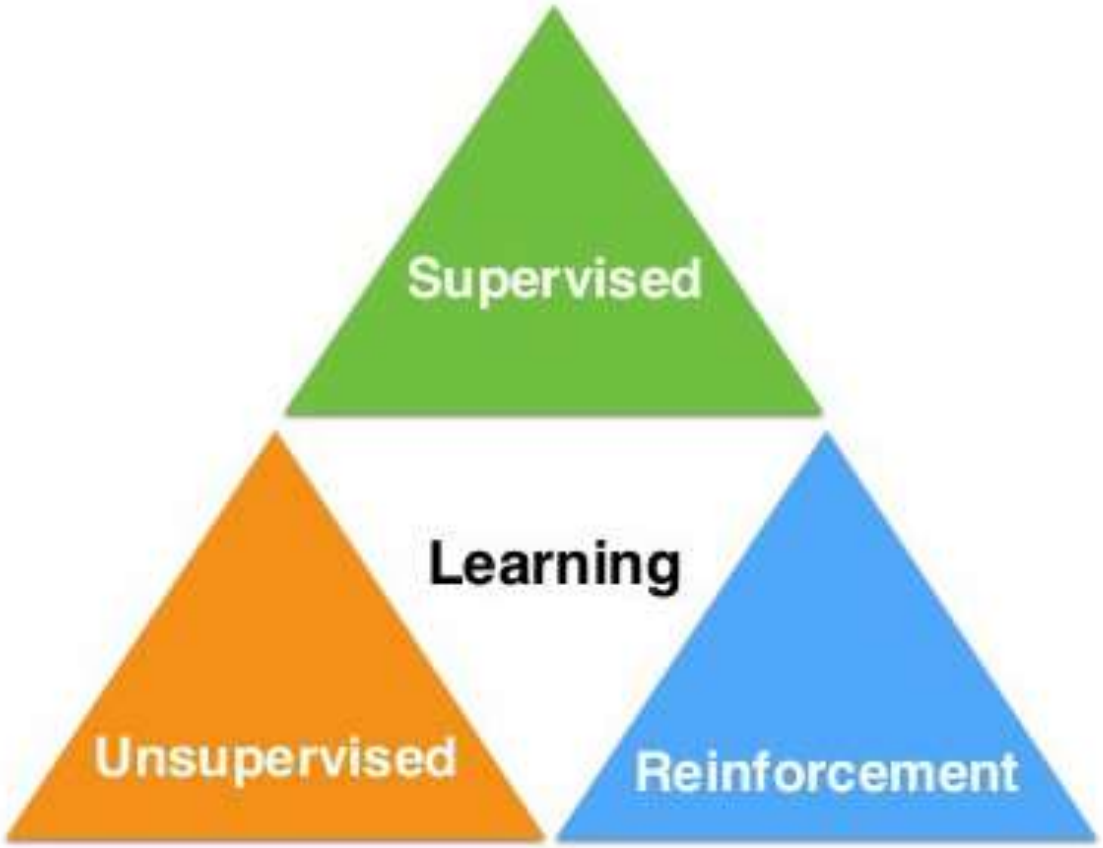
They describe how a machine learns when some data is given to it, its pattern of approach for some particular data.

There are three basic types of learning paradigms widely associated with machine learning, namely:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Machine Learning Paradigms

- Labeled data
- Direct feedback
- Predict outcome/future



- No labels
- No feedback
- "Find hidden structure"

- Decision process
- Reward system
- Learn series of actions

Supervised Learning

Broadly speaking, we utilize machine learning to determine an unknown function (f) by analyzing a specific set of data (D).

In supervised learning, the example data set contains both input and output values.

This means it is in the form:

$D = \{ (x1, y1), (x2, y2), \ldots, (xN, yN) \}$

Since we have the yn values (the output), our learning algorithm is tasked with finding the best approximate function

Supervised Learning

Some practical examples of the same are :

Classification: Machine is trained to classify something into some class.

- classifying whether a patient has disease or not.
- classifying whether an email is spam or not.

Regression: Machine is trained to predict some value like price, weight or height.

- predicting house/property price.
- predicting stock market price.

Unsupervised Learning

When outputs do not exist within the example data, unsupervised learning techniques are used.

Without outputs, the data set is in this form:

$$D = \{ x1, x2, \ldots, xN \}$$

Since we do not have yn values, it's harder to find a suitable approximation.

The unsupervised learning algorithm can pre-process data so that subsequent analysis by a supervised learning algorithm will have a better chance of success.

Unsupervised Learning

Some real life examples of the same are:

Clustering: A clustering problem is where you want to discover the inherent groupings in the data.

such as grouping customers by purchasing behavior.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data.

such as people that buy X also tend to buy Y.

Reinforcement Learning

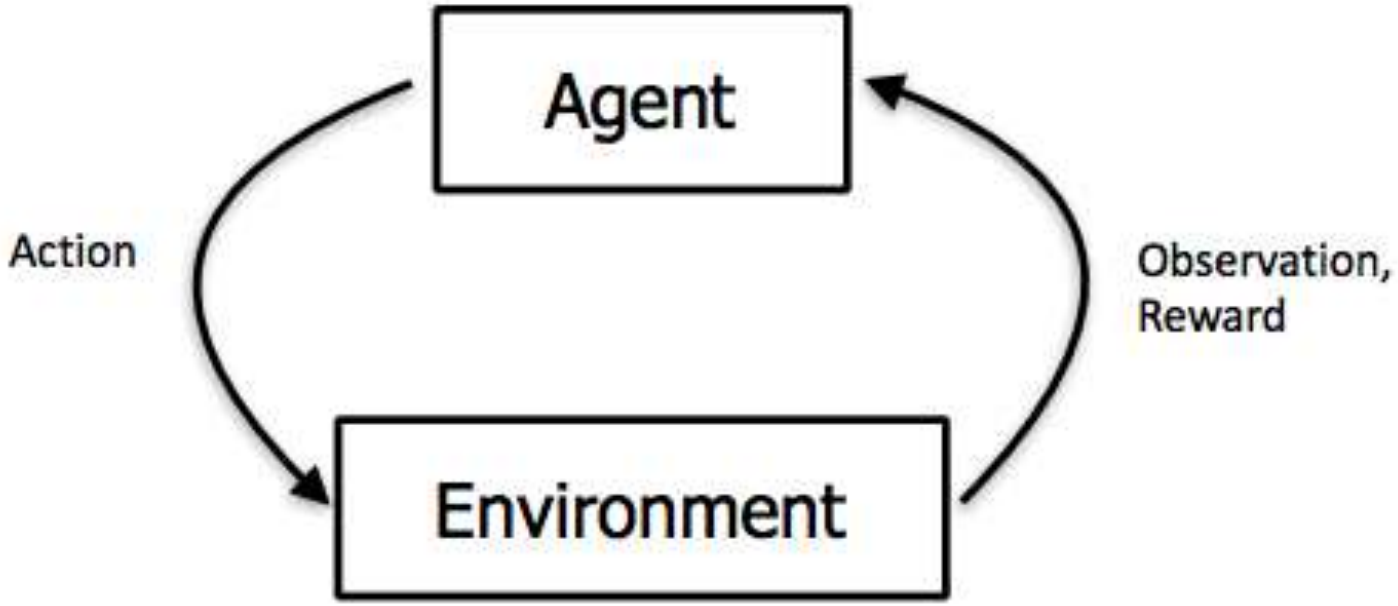
Reinforcement learning approaches are used when we don't even have an example set to begin with.

Instead, our learning algorithm must construct function f based on some set of rules. Then an iterative process is used to improve (or "reinforce") f.

This learning paradigm is like a dog trainer, which teaches the dog how to respond to specific signs, like a whistle, clap, or anything else. Whenever the dog responds correctly, the trainer gives a reward to the dog, which can be a "Bone or a biscuit".

Reinforcement Learning

It allows machines and software agents to automatically determine the ideal behavior within a specific context, in order to maximize its performance.



Reinforcement Learning

The best suited problems for reinforcement learning are:

Game playing — determining the best move to make in a game often depends on a number of different factors, hence the number of possible states that can exist in a particular game is usually very large.

Control problems — such as elevator scheduling. Again, it is not obvious what strategies would provide the best, most timely elevator service. For control problems such as this, RL agents can be left to learn in a simulated environment and eventually they will come up with good controlling policies.

Learning by Induction

Inductive learning is an inherently conjectural process because any knowledge created by generalization from specific facts cannot be proven true.

It can only be proven false. Hence, inductive inference is falsity preserving, not truth preserving.

To generalize beyond the specific training examples, we need constraints or biases on what function f is best. That is, learning can be viewed as searching the Hypothesis Space H of possible f functions.

Learning by Induction

A bias allows us to choose one f over another one.

A completely unbiased inductive algorithm could only memorize the training examples and could not say anything more about other unseen examples.

Two types of biases are commonly used:

1. Restricted Hypothesis Space Bias Allow only certain types of f functions, not arbitrary ones in machine learning:
2. Preference Bias Define a metric for comparing f's so as to determine whether one is better than another .

Learning by Induction

Inductive Learning Framework:

Raw input data from sensors are preprocessed to obtain a feature vector, x, that adequately describes all of the relevant features for classifying examples.

Each x is a list of (attribute, value) pairs.

For example:

x = (Person = Sue, Eye-Color = Brown, Age = Young, Gender = Female)

The number of attributes (also called features) is fixed (positive, finite). Each attribute has a fixed, finite number of possible values.

Learning using Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information.

It is composed of a large number of highly interconnected simple processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example.

An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

Learning using Neural Networks

Advantages of Neural Networks:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.

2. Self-Organization: An ANN can create its own organization or representation of the information it receives during learning time.

3. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices.

4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance.

Learning using Neural Networks

Artificial neural networks are represented by a set of nodes, often arranged in layers, and a set of weighted directed links connecting them.

The nodes are equivalent to neurons, while the links denote synapses. The nodes are the information processing units and the links acts as communicating media.

There are a wide variety of networks depending on the nature of information processing carried out at individual nodes.

Learning using Neural Networks

These networks are configured based on the topology of the links, and the algorithm for adaptation of link weights.

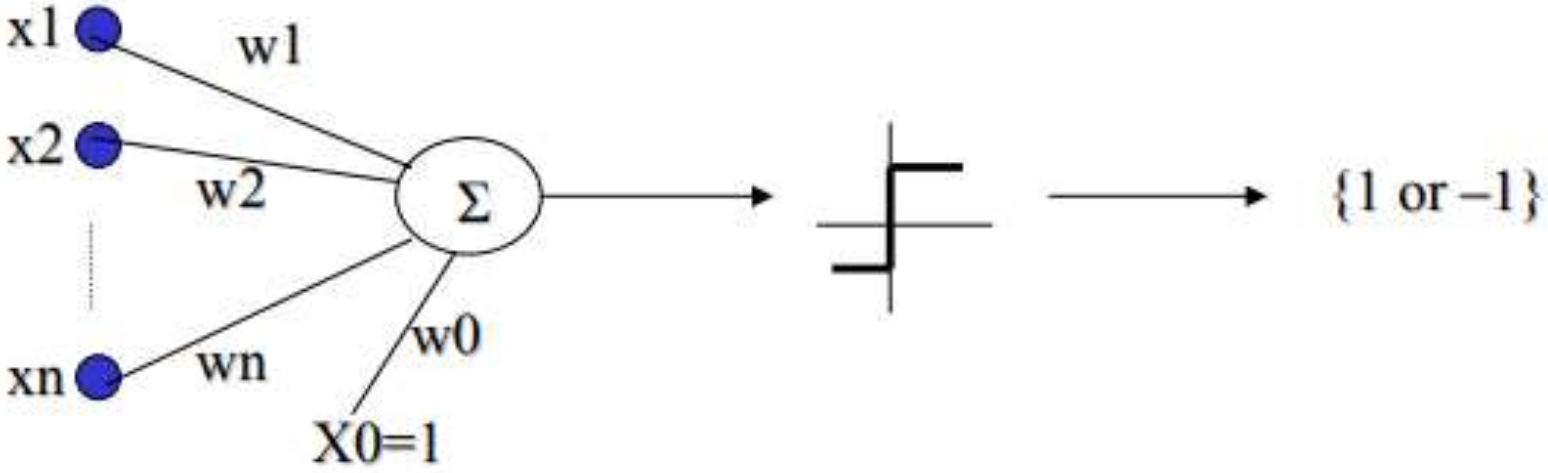
Some of the popular among them include:

- Perceptron
- Multi-layered Perceptron
- Recurrent Neural Networks
- Self-Organizing Maps

Neural Networks - Perceptron

Definition: It's a step function based on a linear combination of real-valued inputs. If the combination is above a threshold it outputs a 1, otherwise I

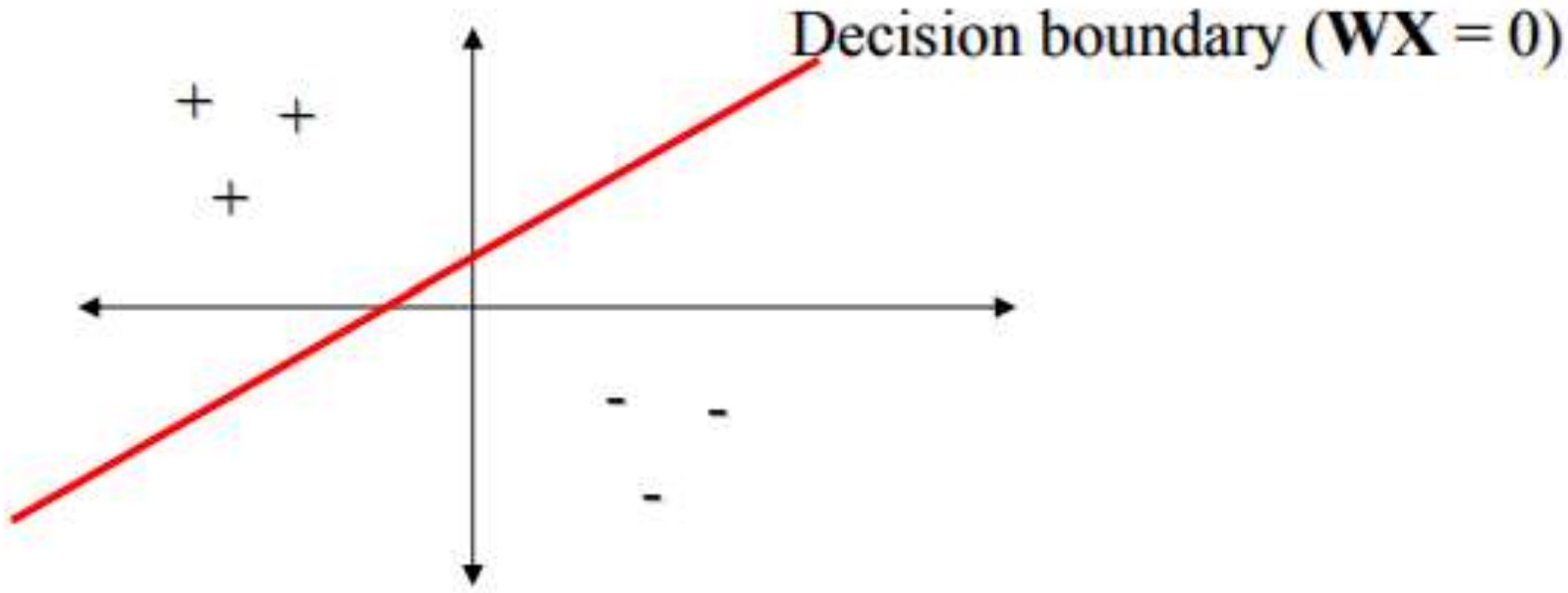
t outputs a –1.



$$O(x1,x2,\ldots,xn) = \begin{cases} 1 \text{ if } w0 + w1x1 + w2x2 + \ldots + wnxn > 0 \\ -1 \text{ otherwise} \end{cases}$$

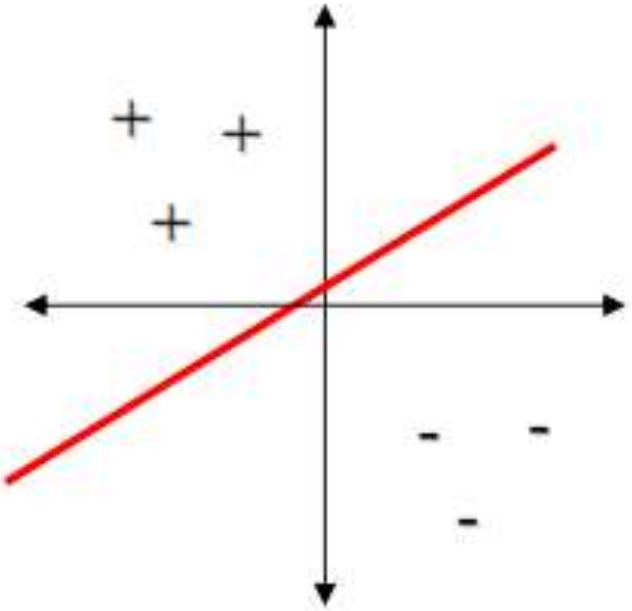
Neural Networks - Perceptron

A perceptron draws a hyperplane as the decision boundary over the (n-dimensional) input space.

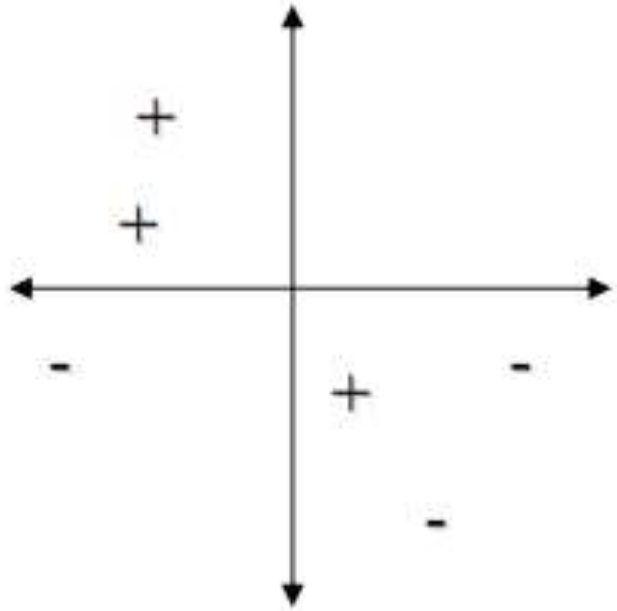


Neural Networks - Perceptron

A perceptron can learn only examples that are called “linearly separable”. These are examples that can be perfectly separated by a hyperplane.



Linearly separable



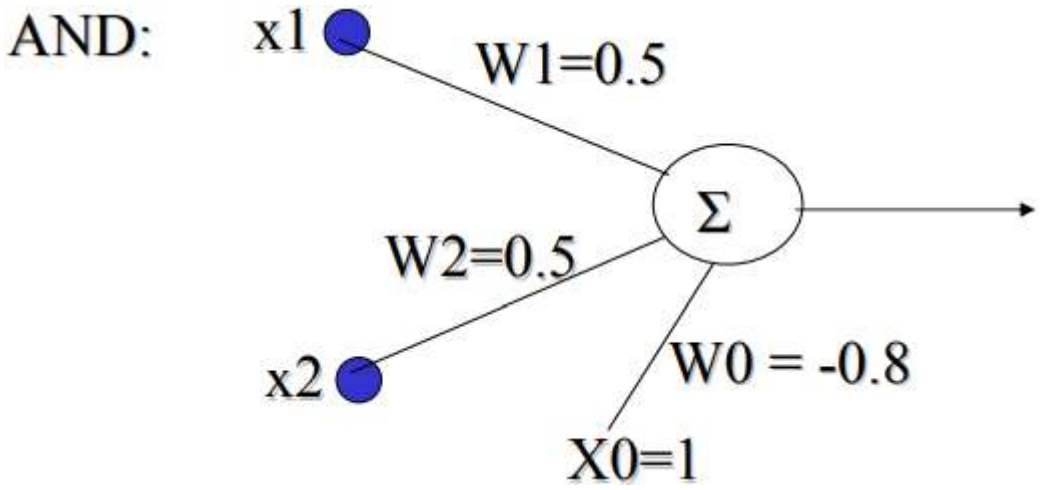
Non-linearly separable

Neural Networks - Perceptron

Perceptrons can learn many boolean functions: AND, OR, NAND, NOR, but not XOR.

However, every boolean function can be represented with a perceptron network that has two levels of depth or more.

The weights of a perceptron implementing the AND function is shown below:



Neural Networks - Perceptron

Perceptron Learning:

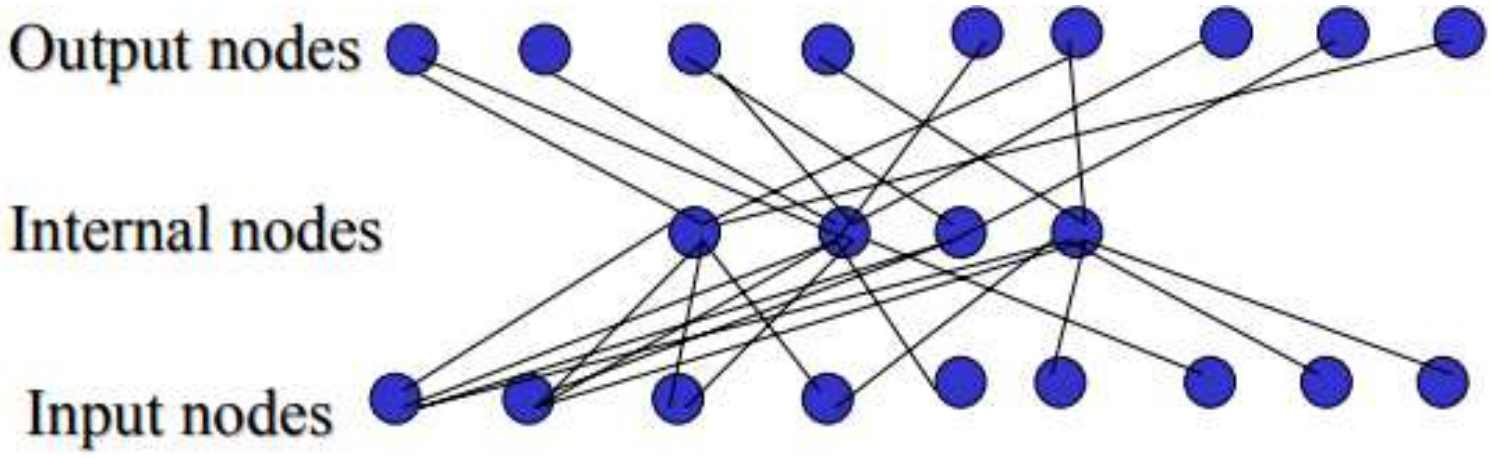
Learning a perceptron means finding the right values for W. The hypothesis space of a perceptron is the space of all weight vectors. The perceptron learning algorithm can be stated as below:

1. Assign random values to the weight vector
2. Apply the weight update rule to every training example
3. Are all training examples correctly classified?
    a. Yes. Quit
    b. No. Go back to Step 2.

Neural Networks - Multi Layer Perceptrons

In contrast to perceptrons, multilayer networks can learn not only multiple decision boundaries, but the boundaries may be nonlinear.

The typical architecture of a multi-layer perceptron (MLP) is shown below:



Neural Networks -
Recurrent Neural Networks

Recurrent Neural Network(RNN) are a type of Neural Network where the output from previous step are fed as input to the current step.

In traditional neural networks, all the inputs and outputs are independent of each other.

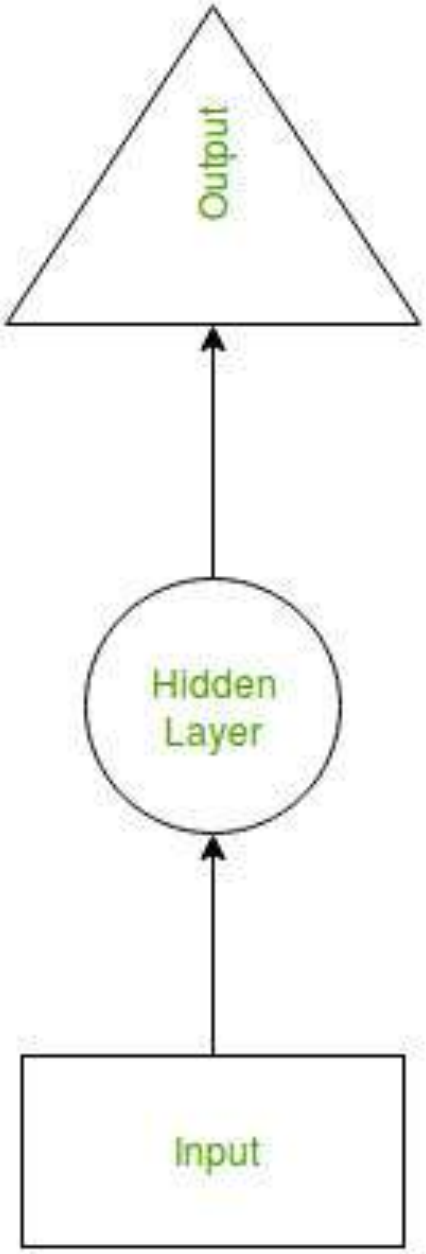
But in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words.

Neural Networks -
Recurrent Neural Networks

The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

RNN have a “memory” which remembers all information about what has been calculated.

It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output.



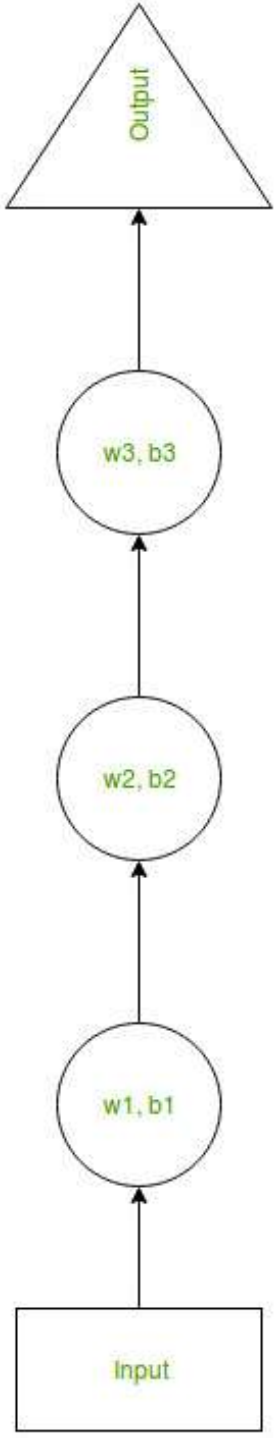
Neural Networks -
Recurrent Neural Networks

Suppose there is a deeper network with one input layer, three hidden layers and one output layer.

Then like other neural networks, each hidden layer will have its own set of weights and biases,

let's say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for second hidden layer and (w3, b3) for third hidden layer.

This means that each of these layers are independent of each other, i.e. they do not memorize the previous outputs.

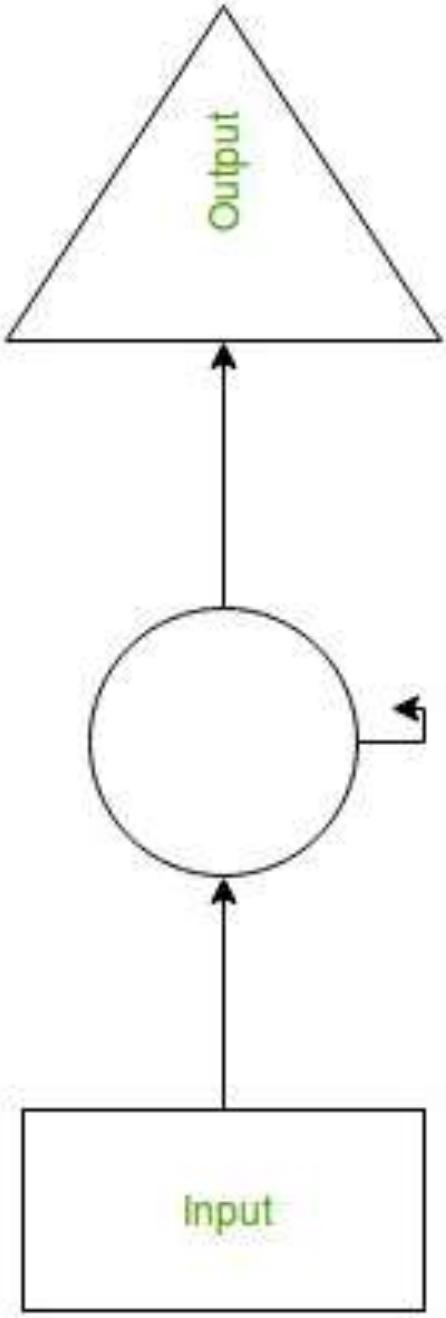


Neural Networks -
Recurrent Neural Networks

RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers.

Thus reducing the complexity of increasing parameters and memorizing each previous outputs by giving each output as input to the next hidden layer.

Hence these three layers can be joined together such that the weights and bias of all the hidden layers is the same, into a single recurrent layer.



Neural Networks -
Recurrent Neural Networks

Advantages:

A RNN remembers each and every information through time.

It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short Term Memory.

Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighborhood.

Neural Networks -

Self-Organizing Maps

SOMs or Kohonen networks have a grid topology, wit unequal grid weights.

The topology of the grid provides a low dimensional visualization of the data distribution.

These are thus used in applications which typically involve organization and human browsing of a large volume of data.

Learning is performed using a winner take all strategy in an unsupervised mode.

Learning using Neural Networks

Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

- sales forecasting
- industrial process control
- customer research
- data validation
- risk management
- target marketing

Learning using Neural Networks

Because of their adaptive and non-linear nature they have also been used in a number of control system application domains like:

- process control in chemical plants
- unmanned vehicles
- robotics
- consumer electronics

Neural networks are also used a number of other applications which are too hard to model using classical techniques. These include, computer vision, path planning and user modeling.