Reasoning in Artificial intelligence

In previous topics, we have learned various ways of knowledge representation in artificial intelligence. Now we will learn the various ways to reason on this knowledge using different logical schemes.

Reasoning:

The reasoning is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. Or we can say, "**Reasoning is a way to infer facts from existing data**." It is a general process of thinking rationally, to find valid conclusions.

In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

Types of Reasoning

In artificial intelligence, reasoning can be divided into the following categories:

Backward Skip 10sPlay VideoForward Skip 10s

- Deductive reasoning
- Inductive reasoning
- Abductive reasoning
- Common Sense Reasoning
- Monotonic Reasoning
- Non-monotonic Reasoning

Note: Inductive and deductive reasoning are the forms of propositional logic.

1. Deductive reasoning:

Deductive reasoning is deducing new information from logically related known information. It is the form of valid reasoning, which means the argument's conclusion must be true when the premises are true.

Deductive reasoning is a type of propositional logic in AI, and it requires various rules and facts. It is sometimes referred to as top-down reasoning, and contradictory to inductive reasoning.

✓

In deductive reasoning, the truth of the premises guarantees the truth of the conclusion.

Deductive reasoning mostly starts from the general premises to the specific conclusion, which can be explained as below example.

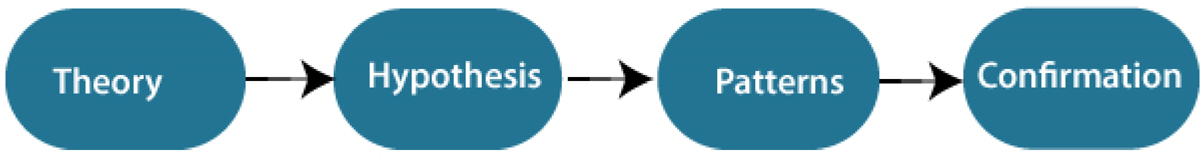
Example:

Premise-1: All the human eats veggies

Premise-2: Suresh is human.

Conclusion: Suresh eats veggies.

The general process of deductive reasoning is given below:



2. Inductive Reasoning:

Inductive reasoning is a form of reasoning to arrive at a conclusion using limited sets of facts by the process of generalization. It starts with the series of specific facts or data and reaches to a general statement or conclusion.

Inductive reasoning is a type of propositional logic, which is also known as cause-effect reasoning or bottom-up reasoning.

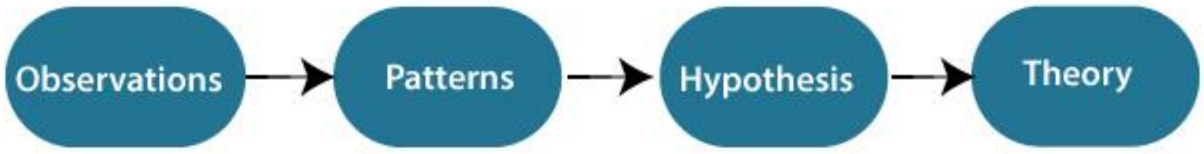
In inductive reasoning, we use historical data or various premises to generate a generic rule, for which premises support the conclusion.

In inductive reasoning, premises provide probable supports to the conclusion, so the truth of premises does not guarantee the truth of the conclusion.

Example:

Premise: All of the pigeons we have seen in the zoo are white.

Conclusion: Therefore, we can expect all the pigeons to be white.



3. Abductive reasoning:

Abductive reasoning is a form of logical reasoning which starts with single or multiple observations then seeks to find the most likely explanation or conclusion for the observation.

Abductive reasoning is an extension of deductive reasoning, but in abductive reasoning, the premises do not guarantee the conclusion.

Example:

Implication: Cricket ground is wet if it is raining

Axiom: Cricket ground is wet. ✓

Conclusion It is raining.

4. Common Sense Reasoning

Common sense reasoning is an informal form of reasoning, which can be gained through experiences.

Common Sense reasoning simulates the human ability to make presumptions about events which occurs on every day.

It relies on good judgment rather than exact logic and operates on **heuristic knowledge** and **heuristic rules**.

Example:

1. **One person can be at one place at a time.**
2. **If I put my hand in a fire, then it will burn.**

The above two statements are the examples of common sense reasoning which a human mind can easily understand and assume.

5. Monotonic Reasoning:

In monotonic reasoning, once the conclusion is taken, then it will remain the same even if we add some other information to existing information in our knowledge base. In monotonic reasoning, adding knowledge does not decrease the set of prepositions that can be derived.

To solve monotonic problems, we can derive the valid conclusion from the available facts only, and it will not be affected by new facts.

Monotonic reasoning is not useful for the real-time systems, as in real time, facts get changed, so we cannot use monotonic reasoning.

Monotonic reasoning is used in conventional reasoning systems, and a logic-based system is monotonic.

Any theorem proving is an example of monotonic reasoning.

Example:

- **Earth revolves around the Sun.** ✓

It is a true fact, and it cannot be changed even if we add another sentence in knowledge base like, "The moon revolves around the earth" Or "Earth is not round," etc.

Advantages of Monotonic Reasoning:

- In monotonic reasoning, each old proof will always remain valid.
- If we deduce some facts from available facts, then it will remain valid for always.

Disadvantages of Monotonic Reasoning:

- We cannot represent the real world scenarios using Monotonic reasoning.
- Hypothesis knowledge cannot be expressed with monotonic reasoning, which means facts should be true.
- Since we can only derive conclusions from the old proofs, so new knowledge from the real world cannot be added.

6. Non-monotonic Reasoning

In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base.

Logic will be said as non-monotonic if some conclusions can be invalidated by adding more knowledge into our knowledge base.

Non-monotonic reasoning deals with incomplete and uncertain models.

"Human perceptions for various things in daily life, "is a general example of non-monotonic reasoning.

Example: Let suppose the knowledge base contains the following knowledge:

- **Birds can fly** ✓
- **Penguins cannot fly** ✓
- **Pitty is a bird** ✓

So from the above sentences, we can conclude that **Pitty can fly**.

However, if we add one another sentence into knowledge base "**Pitty is a penguin**", which concludes "**Pitty cannot fly**", so it invalidates the above conclusion.

Advantages of Non-monotonic reasoning:

- For real-world systems such as Robot navigation, we can use non-monotonic reasoning. ✓
- In Non-monotonic reasoning, we can choose probabilistic facts or can make assumptions. ✓

Disadvantages of Non-monotonic Reasoning:

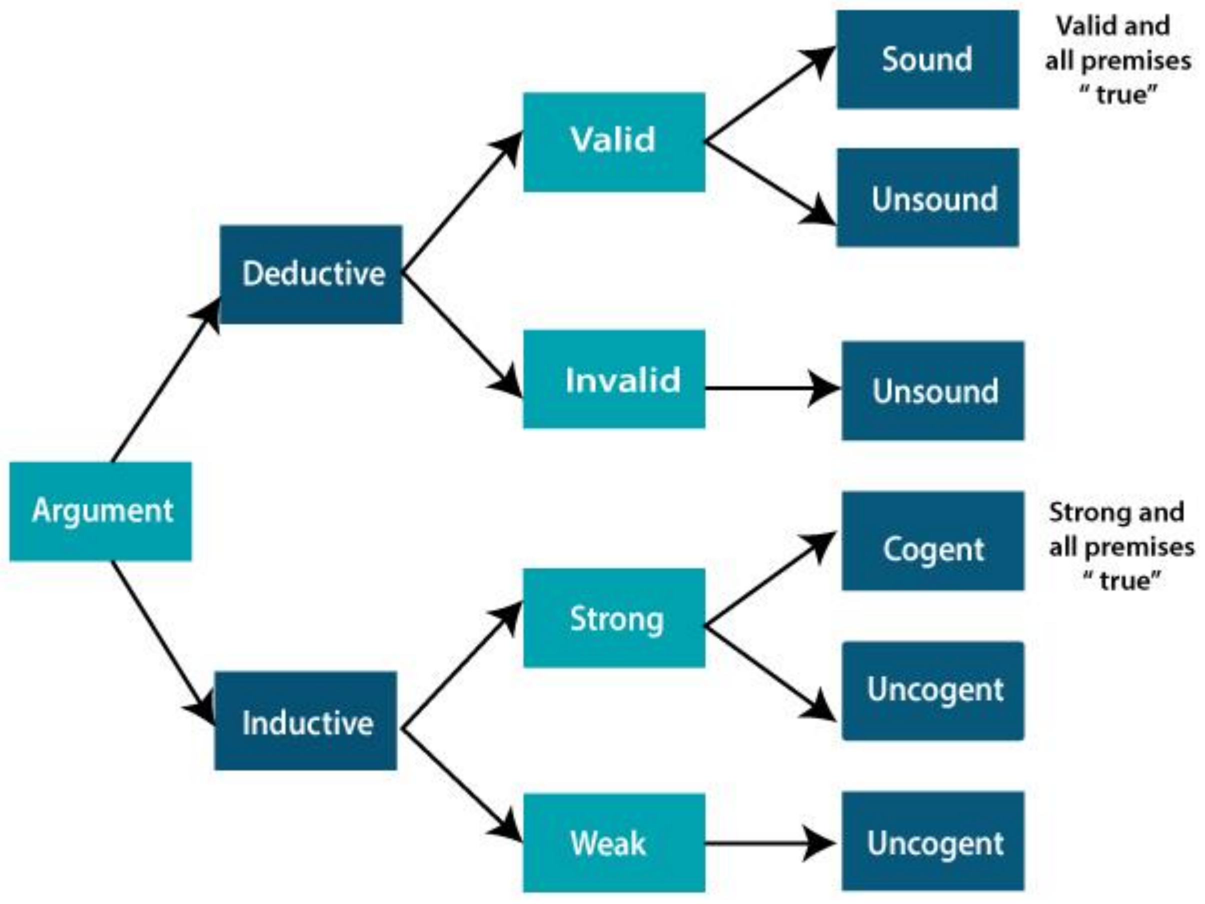
- In non-monotonic reasoning, the old facts may be invalidated by adding new sentences. ✓
- It cannot be used for theorem **proving**. ✓

Difference between Inductive and Deductive reasoning

Reasoning in artificial intelligence has two important forms, Inductive reasoning, and Deductive reasoning. Both reasoning forms have premises and conclusions, but both reasoning are contradictory to each other. Following is a list for comparison between inductive and deductive reasoning:

- Deductive reasoning uses available facts, information, or knowledge to deduce a valid conclusion, whereas inductive reasoning involves making a generalization from specific facts, and observations.
- Deductive reasoning uses a top-down approach, whereas inductive reasoning uses a bottom-up approach.
- Deductive reasoning moves from generalized statement to a valid conclusion, whereas Inductive reasoning moves from specific observation to a generalization.
- In deductive reasoning, the conclusions are certain, whereas, in Inductive reasoning, the conclusions are probabilistic.
- Deductive arguments can be valid or invalid, which means if premises are true, the conclusion must be true, whereas inductive argument can be strong or weak, which means conclusion may be false even if premises are true.

The differences between inductive and deductive can be explained using the below diagram on the basis of arguments:



Comparison Chart:

Basis for comparison	Deductive Reasoning	Inductive Reasoning
Definition	Deductive reasoning is the form of valid reasoning, to deduce new information or conclusion from known related facts and information.	Inductive reasoning arrives at a conclusion by the process of generalization using specific facts or data.
Approach	Deductive reasoning follows a top-down approach.	Inductive reasoning follows a bottom-up approach.
Starts from	Deductive reasoning starts from Premises.	Inductive reasoning starts from the Conclusion.
Validity	In deductive reasoning conclusion must be true if the premises are true.	In inductive reasoning, the truth of premises does not guarantee the truth of conclusions.
Usage	Use of deductive reasoning is difficult, as we need facts which must be true.	Use of inductive reasoning is fast and easy, as we need evidence instead of true facts. We often use it in our daily life.
Process	Theory→ hypothesis→ patterns→confirmation.	Observations-→patterns→hypothesis→Theory.
Argument	In deductive reasoning, arguments may be valid or invalid.	In inductive reasoning, arguments may be weak or strong.
Structure	Deductive reasoning reaches from general facts to specific facts.	Inductive reasoning reaches from specific facts to general facts.

Probabilistic reasoning in Artificial intelligence

Uncertainty:

Till now, we have learned knowledge representation using first-order logic and propositional logic with certainty, which means we were sure about the predicates. With this knowledge representation, we might write A→B, which means if A is true then B is true, but consider a situation where we are not sure about whether A is true or not then we cannot express this statement, this situation is called uncertainty.

So to represent uncertain knowledge, where we are not sure about the predicates, we need uncertain reasoning or probabilistic reasoning.

Causes of uncertainty:

Following are some leading causes of uncertainty to occur in the real world.

1. Information occurred from unreliable sources.
2. Experimental Errors
3. Equipment fault
4. Temperature variation
5. Climate change.

Probabilistic reasoning:

Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge. In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.

We use probability in probabilistic reasoning because it provides a way to handle the uncertainty that is the result of someone's laziness and ignorance.

In the real world, there are lots of scenarios, where the certainty of something is not confirmed, such as "It will rain today," "behavior of someone for some situations," "A match between two teams or two players." These are probable sentences for which we can assume that it will happen but not sure about it, so here we use probabilistic reasoning.

Need of probabilistic reasoning in AI:

- When there are unpredictable outcomes.
- When specifications or possibilities of predicates becomes too large to handle.
- When an unknown error occurs during an experiment.

In probabilistic reasoning, there are two ways to solve problems with uncertain knowledge:

- **Bayes' rule**
- **Bayesian Statistics**

Note: We will learn the above two rules in later chapters.

As probabilistic reasoning uses probability and related terms, so before understanding probabilistic reasoning, let's understand some common terms:

Probability: Probability can be defined as a chance that an uncertain event will occur. It is the numerical measure of the likelihood that an event will occur. The value of probability always remains between 0 and 1 that represent ideal uncertainties.

1. $0 \leq P(A) \leq 1$, where P(A) is the probability of an event A.
1. P(A) = 0, indicates total uncertainty in an event A.
1. P(A) =1, indicates total certainty in an event A.

We can find the probability of an uncertain event by using the below formula.

$$\text{Probability of occurrence} = \frac{\text{Number of desired outcomes}}{\text{Total number of outcomes}}$$

- P(¬A) = probability of a not happening event.
- P(¬A) + P(A) = 1.

Event: Each possible outcome of a variable is called an event.

Sample space: The collection of all possible events is called sample space.

Random variables: Random variables are used to represent the events and objects in the real world.

Prior probability: The prior probability of an event is probability computed before observing new information.

Posterior Probability: The probability that is calculated after all evidence or information has taken into account. It is a combination of prior probability and new information.

Conditional probability:

Conditional probability is a probability of occurring an event when another event has already happened.

Let's suppose, we want to calculate the event A when event B has already occurred, "the probability of A under the conditions of B", it can be written as:

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

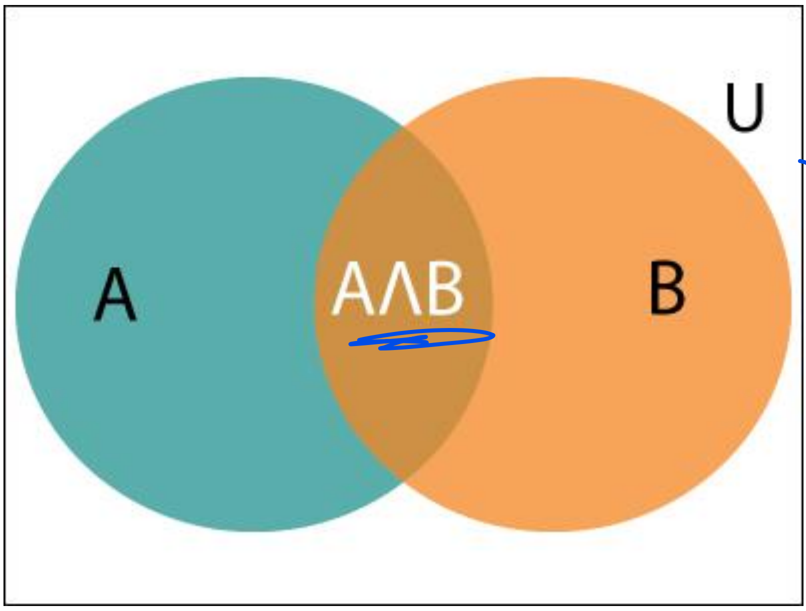
Where P($A \wedge B$)= Joint probability of a and B

P(B)= Marginal probability of B.

If the probability of A is given and we need to find the probability of B, then it will be given as:

$$P(B|A) = \frac{P(A \land B)}{P(A)}$$

It can be explained by using the below Venn diagram, where B is occurred event, so sample space will be reduced to set B, and now we can only calculate event A when event B is already occurred by dividing the probability of **P(A∧*B*) by P(B)**.



$=70$

$\frac{P(A)}{P(A \land B)} = 70$

$P(A \land L)$

$= ?$

Example:

In a class, there are 70% of the students who like English and 40% of the students who likes English and mathematics, and then what is the percent of students those who like English also like mathematics?

Solution:

Let, A is an event that a student likes Mathematics

B is an event that a student likes English.

Bayes' theorem in Artificial intelligence

Bayes' theorem:

Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.

In probability theory, it relates the conditional probability and marginal probabilities of two random events.

Bayes' theorem was named after the British mathematician **Thomas Bayes**. The **Bayesian inference** is an application of Bayes' theorem, which is fundamental to Bayesian statistics.

It is a way to calculate the value of P(B|A) with the knowledge of P(A|B).

Bayes' theorem allows updating the probability prediction of an event by observing new information of the real world.

Example: If cancer corresponds to one's age then by using Bayes' theorem, we can determine the probability of cancer more accurately with the help of age.

Bayes' theorem can be derived using product rule and conditional probability of event A with known event B:

As from product rule we can write:

1. $P(A \wedge B) = P(A|B) P(B)$ or

Similarly, the probability of event B with known event A:

1. $P(A \wedge B) = P(B|A) P(A)$

Equating right hand side of both the equations, we will get:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \quad \text{....(a)}$$

The above equation (a) is called as **Bayes' rule** or **Bayes' theorem**. This equation is basic of most modern AI systems for **probabilistic inference**.

It shows the simple relationship between joint and conditional probabilities. Here,

P(A|B) is known as **posterior**, which we need to calculate, and it will be read as Probability of hypothesis A when we have occurred an evidence B.

P(B|A) is called the likelihood, in which we consider that hypothesis is true, then we calculate the probability of evidence.

P(A) is called the **prior probability**, probability of hypothesis before considering the evidence

P(B) is called **marginal probability**, pure probability of an evidence.

In the equation (a), in general, we can write P (B) = P(A)*P(B|Ai), hence the Bayes' rule can be written as:

$$P(A_i|B) = \frac{P(A_i)*P(B|A_i)}{\sum_{i=1}^{k} P(A_i)*P(B|A_i)}$$

Where A_1, A_2, A_3,........, A_n is a set of mutually exclusive and exhaustive events.

Applying Bayes' rule:

Bayes' rule allows us to compute the single term P(B|A) in terms of P(A|B), P(*B*), and P(A). This is very useful in cases where we have a good probability of these three terms and want to determine the fourth one. Suppose we want to perceive the effect of some unknown cause, and want to compute that cause, then the Bayes' rule becomes:

A - B

$$\mathbf{P(cause|effect)} = \frac{P(effect|cause)\, P(cause)}{P(effect)}$$

Example-1:

Question: what is the probability that a patient has diseases meningitis with a stiff neck?

Given Data:

A doctor is aware that disease meningitis causes a patient to have a stiff neck, and it occurs 80% of the time. He is also aware of some more facts, which are given as follows:

- The Known probability that a patient has meningitis disease is 1/30,000.
- The Known probability that a patient has a stiff neck is 2%.

Let a be the proposition that patient has stiff neck and b be the proposition that patient has meningitis. , so we can calculate the following as:

P(a|b) = 0.8

P(b) = 1/30000

P(a)= .02

$$\mathbf{P(b|a)} = \frac{P(a|b)P(b)}{P(a)} = \frac{0.8*(\frac{1}{30000})}{0.02} = 0.001333333.$$

Hence, we can assume that 1 patient out of 750 patients has meningitis disease with a stiff neck.

Example-2:

Question: From a standard deck of playing cards, a single card is drawn. The probability that the card is king is 4/52, then calculate posterior probability P(King|Face), which means the drawn face card is a king card.

Solution:

$$\mathbf{P(king|face)} = \frac{P(Face|king)*P(King)}{P(Face)} \quad ....(i)$$

P(king): probability that the card is King= 4/52= 1/13

P(face): probability that a card is a face card= 3/13

P(Face|King): probability of face card when we assume it is a king = 1

Putting all values in equation (i) we will get:

$$P(king|face) = \frac{1*(\frac{1}{13})}{(\frac{3}{13})} = \mathbf{1/3,}\ \textbf{it is a probability that a face card is a king card.}$$

Application of Bayes' theorem in Artificial intelligence:

Following are some applications of Bayes' theorem:

- It is used to calculate the next step of the robot when the already executed step is given.
- Bayes' theorem is helpful in weather forecasting.
- It can solve the Monty Hall problem.

Bayesian Belief Network in artificial intelligence

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

Backward Skip 10sPlay VideoForward Skip 10s

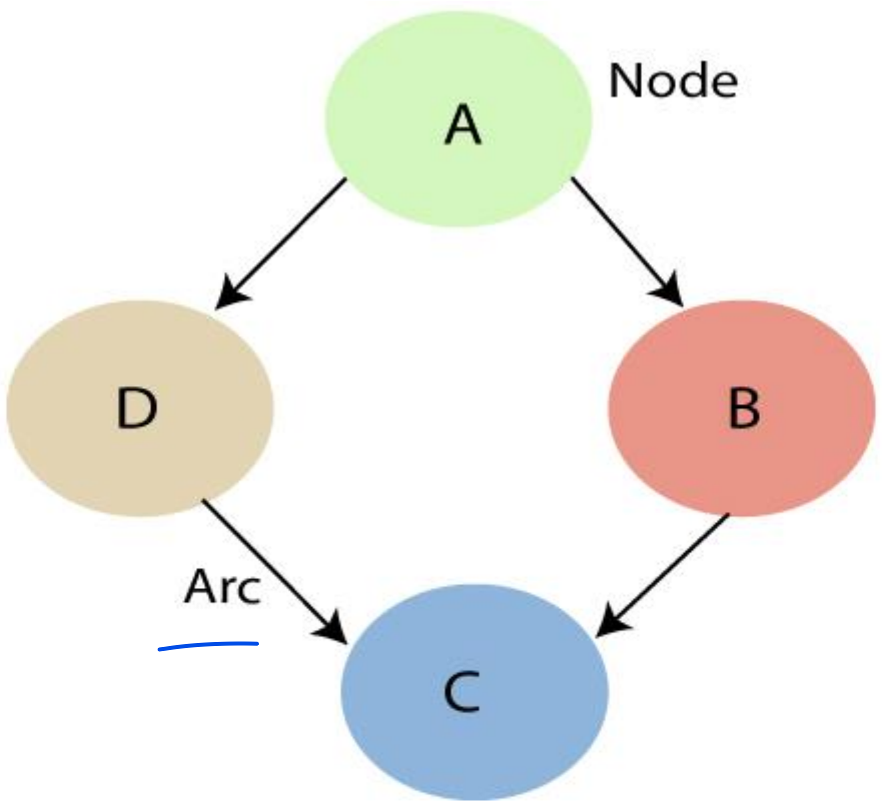
Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

- **Directed Acyclic Graph** ✓
- **Table of conditional probabilities.** ✓

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:



- Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
- **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph. These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other
    - **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
    - **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
    - **Node C is independent of node A.**

Note: The Bayesian network graph does not contain any cyclic graph. Hence, it is known as a **directed acyclic graph or DAG**.

The Bayesian network has mainly two components:

- **Causal Component** ✓
- **Actual numbers** ✓

Each node in the Bayesian network has condition probability distribution **$P(X_i | Parent(X_i))$**, which determines the effect of the parent on that node.

Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution:

Joint probability distribution:

If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.

$P[x_1, x_2, x_3, \ldots, x_n]$, it can be written as the following way in terms of the joint probability distribution.

$= P[x_1 | x_2, x_3, \ldots, x_n] P[x_2, x_3, \ldots, x_n]$

$= P[x_1 | x_2, x_3, \ldots, x_n] P[x_2 | x_3, \ldots, x_n] \ldots P[x_{n-1} | x_n] P[x_n]$.

In general for each variable Xi, we can write the equation as:

$$P(X_i | X_{i-1}, \ldots\ldots\ldots, X_1) = P(X_i | Parents(X_i))$$

Explanation of Bayesian network:

Let's understand the Bayesian network through an example by creating a directed acyclic graph:

Example: Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbors David and Sophia, who have taken a responsibility to inform Harry at work when they hear the alarm. David always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Sophia likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

Problem:

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

Solution:

- The Bayesian network for the above problem is given below. The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.

- The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
- The conditional distributions for each node are given as conditional probabilities table or CPT.
- Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.
- In CPT, a boolean variable with k boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values

List of all events occurring in this network:

- **Burglary (B)** ✓
- **Earthquake(E)** ✓
- **Alarm(A)** ✓
- **David Calls(D)** ✓
- **Sophia calls(S)** ✓

We can write the events of problem statement in the form of probability: **P[D, S, A, B, E]**, can rewrite the above probability statement using joint probability distribution:

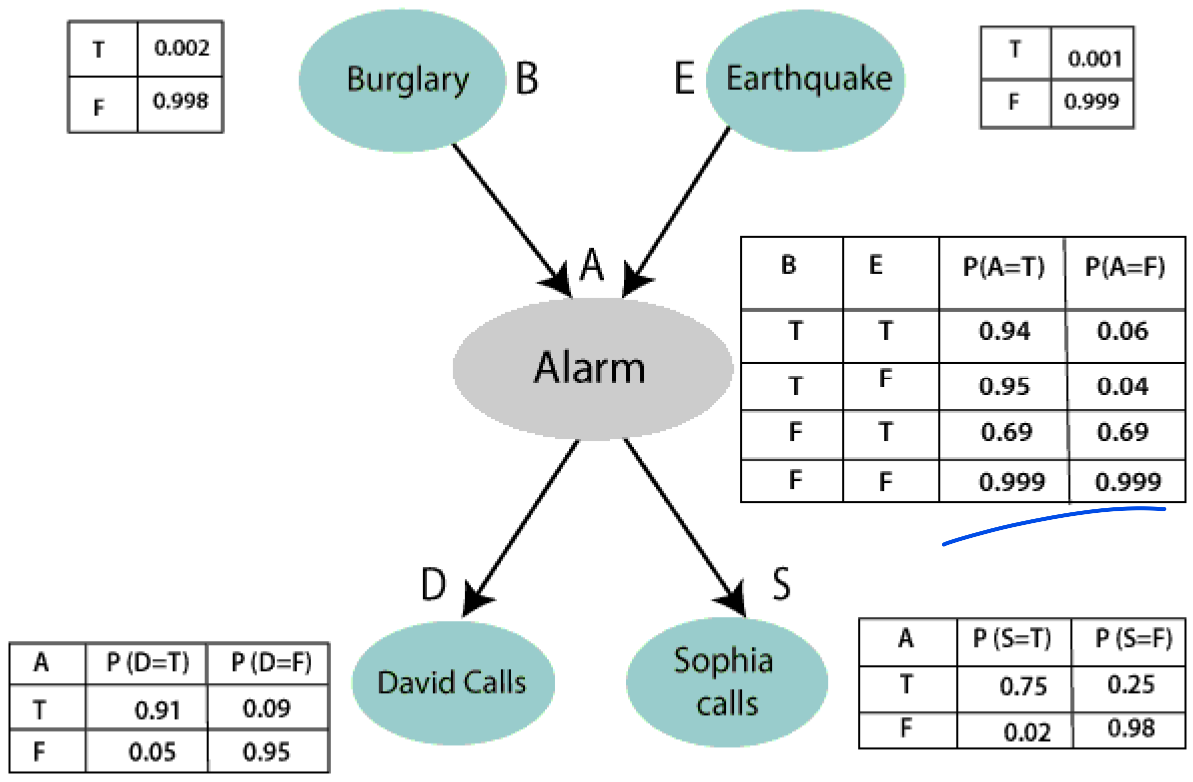
P[D, S, A, B, E]= P[D | S, A, B, E]. P[S, A, B, E]

=P[D | S, A, B, E]. P[S | A, B, E]. P[A, B, E]

= P [D| A]. P [S| A, B, E]. P[A, B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B |E]. P[E]



Let's take the observed probability for the Burglary and earthquake component:

P(B= True) = 0.002, which is the probability of burglary.

P(B= False)= 0.998, which is the probability of no burglary.

P(E= True)= 0.001, which is the probability of a minor earthquake

P(E= False)= 0.999, Which is the probability that an earthquake not occurred.

We can provide the conditional probabilities as per the below tables:

Conditional probability table for Alarm A:

The Conditional probability of Alarm A depends on Burglar and earthquake:

B	E	P(A= True)	P(A= False)
True	True	0.94	0.06
True	False	0.95	0.04
False	True	0.31	0.69

False	False	0.001	0.999

Conditional probability table for David Calls:

The Conditional probability of David that he will call depends on the probability of Alarm.

A	P(D= True)	P(D= False)
True	0.91	0.09
False	0.05	0.95

Conditional probability table for Sophia Calls:

The Conditional probability of Sophia that she calls is depending on its Parent Node "Alarm."

A	P(S= True)	P(S= False)
True	0.75	0.25
False	0.02	0.98

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

P(S, D, A, ¬B, ¬E) = P (S|A) *P (D|A)*P (A|¬B ^ ¬E) *P (¬B) *P (¬E).

= 0.75* 0.91* 0.001* 0.998*0.999

= 0.00068045. ✓

Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

The semantics of Bayesian Network:

There are two ways to understand the semantics of the Bayesian network, which is given below:

1. To understand the network as the representation of the Joint probability distribution.

It is helpful to understand how to construct the network.

2. To understand the network as an encoding of a collection of conditional independence statements.

It is helpful in designing inference procedure.

ML | Dempster Shafer Theory

-

What Dempster-Shafer Theory was given by Arthur P. Dempster in 1967 and his student Glenn Shafer in 1976. This theory was released because of the following reason:-

- Bayesian theory is only concerned about single evidence.
- Bayesian probability cannot describe ignorance.

DST is an evidence theory, it combines all possible outcomes of the problem. Hence it is used to solve problems where there may be a chance that a piece of different evidence will lead to some different result.

The **uncertainty in this model** is given by:-

1. Consider all possible outcomes.
2. **Belief will lead to belief in some possibility by bringing out some evidence. (What is this supposed to mean?)**
3. Plausibility will make evidence compatible with possible outcomes.

Example: Let us consider a room where four people are present, A, B, C, and D. Suddenly the lights go out and when the lights come back, B has been stabbed in the back by a knife, leading to his death. No one came into the room and no one left the room. We know that B has not committed suicide. Now we have to find out who the murderer is.

To solve these there are the **following possibilities**:

- Either {A} or {C} or {D} has killed him.
- Either {A, C} or {C, D} or {A, D} have killed him.
- Or the three of them have killed him i.e; {A, C, D}
- None of them have killed him {o} (let's say).

There will be possible evidence by which we can find the murderer by the measure of plausibility.

Using the above example we can say:
Set of possible conclusion (P): {p1, p2….pn}
where P is a set of possible conclusions and cannot be exhaustive, i.e. at least one (p) I

must be true.
(p)I must be mutually exclusive.
Power Set will contain 2^n elements where n is the number of elements in the possible set.

For eg:-
If P = { a, b, c}, then Power set is given as
{o, {a}, {b}, {c}, {a, d}, {d ,c}, {a, c}, {a, c ,d }}= 2^3 elements.

Mass function m(K): It is an interpretation of m({K or B}) i.e; it means there is evidence for {K or B} which cannot be divided among more specific beliefs for K and B.

Belief in K: The belief in element K of Power Set is the sum of masses of the element which are subsets of K. This can be explained through an example
Lets say K = {a, d, c}
Bel(K) = m(a) + m(d) + m(c) + m(a, d) + m(a, c) + m(d, c) + m(a, d, c)

Plausibility in K: It is the sum of masses of the set that intersects with K.
i.e; Pl(K) = m(a) + m(d) + m(c) + m(a, d) + m(d, c) + m(a, c) + m(a, d, c)

Characteristics of Dempster Shafer Theory:

- **It will ignorance part such that the probability of all events aggregate to 1. (What is this supposed to mean?)**
- Ignorance is reduced in this theory by adding more and more evidence.
- Combination rule is used to combine various types of possibilities.

Advantages:
- As we add more information, the uncertainty interval reduces.
- DST has a much lower level of ignorance.
- Diagnose hierarchies can be represented using this.
- Person dealing with such problems is free to think about evidence.

Disadvantages:
- In this, computation effort is high, as we have to deal with 2^n sets

Planning in Artificial Intelligence (AI) is a critical process that involves making decisions to achieve a specific goal[12]. It's about choosing a sequence of tasks that are most likely to accomplish a specific task[1].

Here are the key components of planning in AI:

1. **Domain Description**: This describes the tasks and domains of a particular problem[1].
2. **Task Specification**: This involves deciding the tasks to be performed by the AI system[1].
3. **Goal Description**: This involves defining the goal that the AI system aims to achieve[1].
4. **Plan**: A plan is a sequence of actions, where each action has its preconditions that must be satisfied before it can act, and some effects that can be positive or negative[1].

There are two basic types of planning:

- **Forward State Space Planning (FSSP)**: Given an initial state S in any domain, necessary actions are performed to obtain a new state S' (which also contains some new terms), called a progression[1]. This continues until the target position is reached[1].
- **Backward State Space Planning (BSSP)**: This involves moving from the target state g to the sub-goal g, tracing the previous action to achieve that goal[1]. This process is called regression[1].

For an efficient planning system, it's beneficial to combine the features of FSSP and BSSP[1]. Planning techniques such as hierarchical, optimal, partial order, and informed search are also used in AI planning[3].

Remember, planning in AI is fundamentally about creating a strategic and organized approach to employing AI systems to accomplish particular aims and objectives[2]. It includes defining the scope and purpose of AI projects, choosing suitable algorithms, data sources, and performance measures, developing and testing models, and monitoring and adapting AI systems over time[2].

What is the Role of Planning in Artificial Intelligence?

Artificial intelligence is an important technology in the future. Whether it is intelligent robots, self-driving cars, or smart cities, they will all use different aspects of artificial intelligence!!! But Planning is very important to make any such AI project.

Even Planning is an important part of Artificial Intelligence which deals with the tasks and domains of a particular problem. Planning is considered the logical side of acting.

Everything we humans do is with a definite goal in mind, and all our actions are oriented towards achieving our goal. Similarly, Planning is also done for Artificial Intelligence.

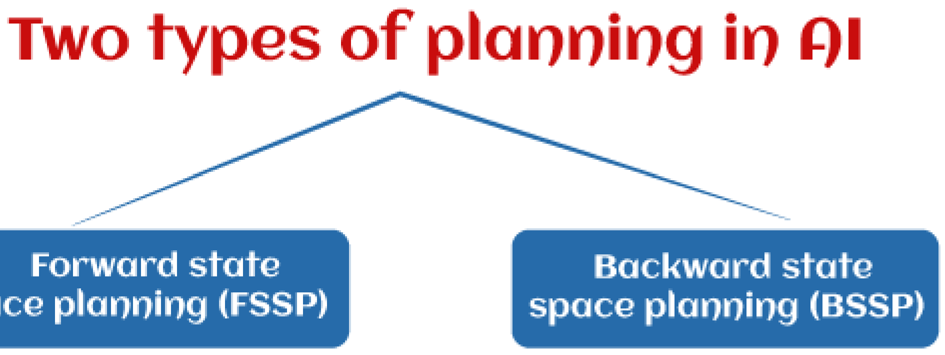
For example, Planning is required to reach a particular destination. It is necessary to find the best route in Planning, but the tasks to be done at a particular time and why they are done are also very important.

That is why Planning is considered the logical side of acting. In other words, Planning is about deciding the tasks to be performed by the artificial intelligence system and the system's functioning under domain-independent conditions.

What is a Plan?

We require domain description, task specification, and goal description for any planning system. A plan is considered a sequence of actions, and each action has its preconditions that must be satisfied before it can act and some effects that can be positive or negative.

So, we have **Forward State Space Planning (FSSP)** and **Backward State Space Planning (BSSP)** at the basic level.


Two types of planning in AI
Forward state space planning (FSSP)
Backward state space planning (BSSP)

1. Forward State Space Planning (FSSP)

FSSP behaves in the same way as forwarding state-space search. It says that given an initial state S in any domain, we perform some necessary actions and obtain a new state S' (which also contains some new terms), called a progression. It continues until we reach the target position. Action should be taken in this matter.

- **Disadvantage**: Large branching factor
- **Advantage**: The algorithm is Sound

2. Backward State Space Planning (BSSP)

BSSP behaves similarly to backward state-space search. In this, we move from the target state g to the sub-goal g, tracing the previous action to achieve that goal. This process is called regression (going back to the previous goal or sub-goal). These sub-goals should also be checked for consistency. The action should be relevant in this case.

- **Disadvantages**: not sound algorithm (sometimes inconsistency can be found)
- **Advantage**: Small branching factor (much smaller than FSSP)

So for an efficient planning system, we need to combine the features of FSSP and BSSP, which gives rise to target stack planning which will be discussed in the next article.

What is planning in AI?

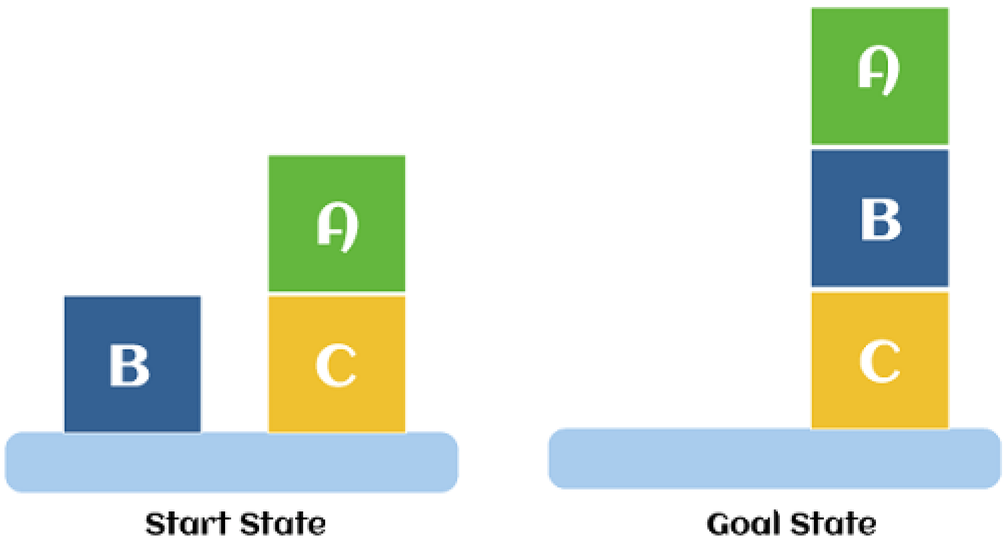
Planning in artificial intelligence is about decision-making actions performed by robots or computer programs to achieve a specific goal.

Execution of the plan is about choosing a sequence of tasks with a high probability of accomplishing a specific task.

Block-world planning problem

- The block-world problem is known as the Sussmann anomaly.
- The non-interlaced planners of the early 1970s were unable to solve this problem. Therefore it is considered odd.
- When two sub-goals, G1 and G2, are given, a non-interleaved planner either produces a plan for G1 that is combined with a plan for **G2** or vice versa.
- In the block-world problem, three blocks labeled 'A', 'B', and 'C' are allowed to rest on a flat surface. The given condition is that only one block can be moved at a time to achieve the target.

The start position and target position are shown in the following diagram.



Word planning problem

Components of the planning system

The plan includes the following important steps:

- Choose the best rule to apply the next rule based on the best available guess.
- Apply the chosen rule to calculate the new problem condition.
- Find out when a solution has been found.
- Detect dead ends so they can be discarded and direct system effort in more useful directions.
- Find out when a near-perfect solution is found.

Target stack plan

- It is one of the most important planning algorithms used by STRIPS.
- Stacks are used in algorithms to capture the action and complete the target. A knowledge base is used to hold the current situation and actions.
- A target stack is similar to a node in a search tree, where branches are created with a choice of action.

The important steps of the algorithm are mentioned below:

1. Start by pushing the original target onto the stack. Repeat this until the pile is empty. If the stack top is a mixed target, push its unsatisfied sub-targets onto the stack.

2. If the stack top is a single unsatisfied target, replace it with action and push the action precondition to the stack to satisfy the condition.

iii. If the stack top is an action, pop it off the stack, execute it and replace the knowledge base with the action's effect.

If the stack top is a satisfactory target, pop it off the stack.

Non-linear Planning

This Planning is used to set a goal stack and is included in the search space of all possible sub-goal orderings. It handles the goal interactions by the interleaving method.

Advantages of non-Linear Planning

Non-linear Planning may be an optimal solution concerning planning length (depending on the search strategy used).

Disadvantages of Nonlinear Planning

It takes a larger search space since all possible goal orderings are considered.

Complex algorithm to understand.

Algorithm

1. Choose a goal 'g' from the goal set
2. If 'g' does not match the state, then
    - Choose an operator 'o' whose add-list matches goal g
    - Push 'o' on the OpStack
    - Add the preconditions of 'o' to the goal set
3. While all preconditions of the operator on top of OpenStack are met in a state
    - Pop operator o from top of opstack
    - state = apply(o, state)
    - plan = [plan; o]

Artificial Intelligence (AI) is a vast field that encompasses many techniques and methods. Some of the basic techniques in AI are:

- Regression: a predictive modeling technique that estimates the relationship between variables.
- Classification: a technique that assigns labels to data based on predefined categories.
- Transfer Learning: a technique that leverages the knowledge gained from one domain to another domain.
- Clustering: a technique that groups data based on similarity or proximity.
- Ensemble Methods: a technique that combines multiple models to improve the accuracy or robustness of predictions.
- Neural Networks and Deep Learning: a technique that uses artificial neural networks to learn complex patterns and features from data.
- Dimensionality Reduction: a technique that reduces the number of features or variables in data to improve efficiency or performance.
- Word Embeddings: a technique that represents words as vectors of numbers based on their semantic and syntactic relationships.
- NLP (Natural Language Processing): a technique that enables machines to understand and generate natural language.
- Automation and Robotics: a technique that uses machines to perform tasks that are repetitive, dangerous, or require precision.
- Machine Vision: a technique that enables machines to perceive and interpret visual information.
- Bayesian Networks: a technique that uses probability theory to model the causal relationships between variables.
- Constraint Satisfaction Problems: a technique that solves problems by finding values that satisfy a set of constraints.
- Logic: a technique that uses formal rules and symbols to represent and reason about knowledge.

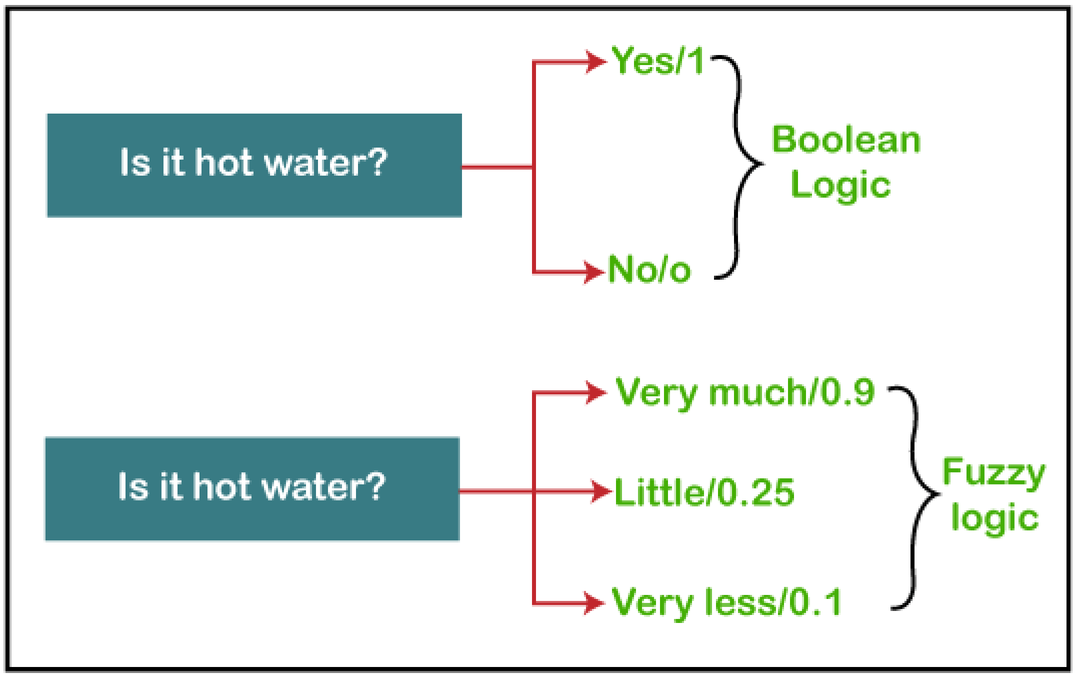
If you are a beginner, you can start with learning the basics of AI like what is AI, history of AI, types of AI, and more. You can also learn about the applications of AI in various industries like healthcare, finance, and transportation. There are many resources available online to learn AI, including courses, books, and tutorials. Some popular platforms for learning AI are Coursera, Udemy, and edX. You can also find many free resources on YouTube and other websites. Good luck! □

Fuzzy Logic Tutorial

What is Fuzzy Logic?

The **'Fuzzy'** word means the things that are not clear or are vague. Sometimes, we cannot decide in real life that the given problem or statement is either true or false. At that time, this concept provides many values between the true and false and gives the flexibility to find the best solution to that problem.

Example of Fuzzy Logic as comparing to Boolean Logic



Fuzzy logic contains the multiple logical values and these values are the truth values of a variable or problem between 0 and 1. This concept was introduced by **Lofti Zadeh** in **1965** based on the **Fuzzy Set Theory**. This concept provides the possibilities which are not given by computers, but similar to the range of possibilities generated by humans.

In the Boolean system, only two possibilities (0 and 1) exist, where 1 denotes the absolute truth value and 0 denotes the absolute false value. But in the fuzzy system, there are multiple possibilities present between the 0 and 1, which are partially false and partially true.

The Fuzzy logic can be implemented in systems such as micro-controllers, workstation-based or large network-based systems for achieving the definite output. It can also be implemented in both hardware or software.

PauseNext
Unmute

Current TimeÂ 0:09

/

DurationÂ 18:10
Loaded: 4.40%
Â
Fullscreen

Characteristics of Fuzzy Logic

Following are the characteristics of fuzzy logic:

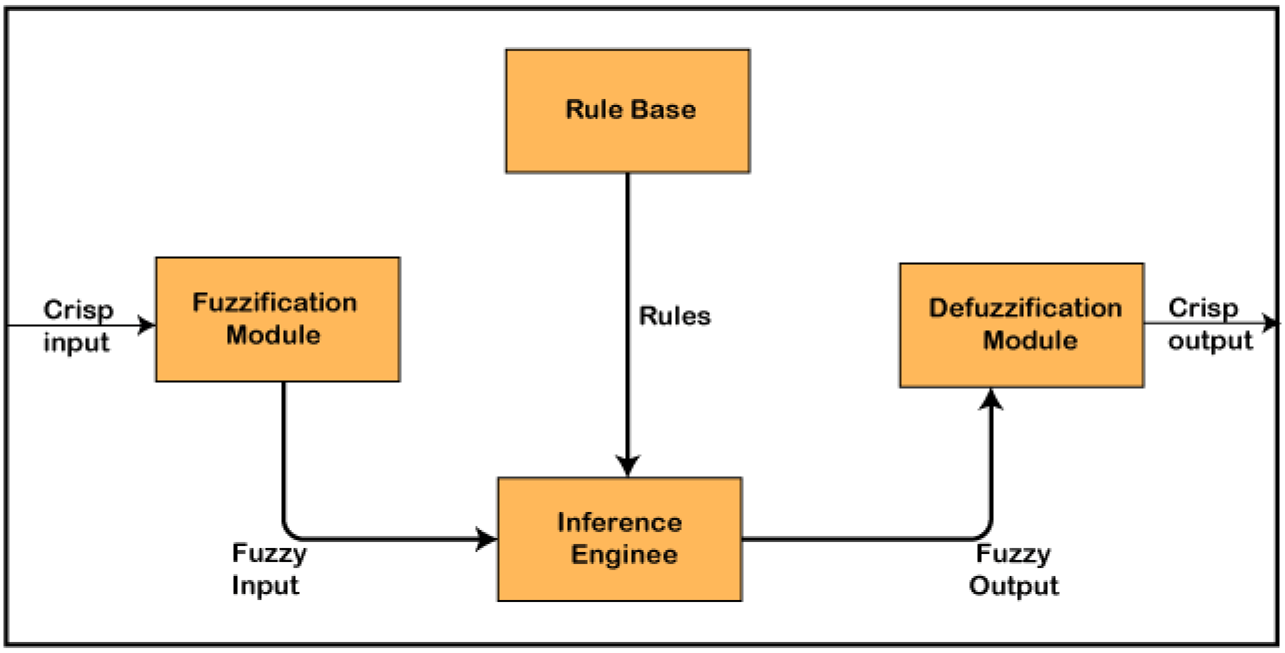
1. This concept is flexible and we can easily understand and implement it.
2. It is used for helping the minimization of the logics created by the human.
3. It is the best method for finding the solution of those problems which are suitable for approximate or uncertain reasoning.
4. It always offers two values, which denote the two possible solutions for a problem and statement.
5. It allows users to build or create the functions which are non-linear of arbitrary complexity.
6. In fuzzy logic, everything is a matter of degree.
7. In the Fuzzy logic, any system which is logical can be easily fuzzified.
8. It is based on natural language processing.
9. It is also used by the quantitative analysts for improving their algorithm's execution.
10. It also allows users to integrate with the programming.

Architecture of a Fuzzy Logic System

In the architecture of the **Fuzzy Logic** system, each component plays an important role. The architecture consists of the different four components which are given below.

1. Rule Base
2. Fuzzification
3. Inference Engine
4. Defuzzification

Following diagram shows the architecture or process of a Fuzzy Logic system:



1. Rule Base

Rule Base is a component used for storing the set of rules and the If-Then conditions given by the experts are used for controlling the decision-making systems. There are so many updates that come in the Fuzzy theory recently, which offers effective methods for designing and tuning of fuzzy controllers. These updates or developments decreases the number of fuzzy set of rules.

2. Fuzzification

Fuzzification is a module or component for transforming the system inputs, i.e., it converts the crisp number into fuzzy steps. The crisp numbers are those inputs which are measured by the sensors and then fuzzification passed them into the control systems for further processing. This component divides the input signals into following five states in any Fuzzy Logic system:

- Large Positive (LP)
- Medium Positive (MP)
- Small (S)
- Medium Negative (MN)
- Large negative (LN)

3. Inference Engine

This component is a main component in any Fuzzy Logic system (FLS), because all the information is processed in the Inference Engine. It allows users to find the matching degree between the current fuzzy input and the rules. After the matching degree, this system determines which rule is to be added according to the given input field. When all rules are fired, then they are combined for developing the control actions.

4. Defuzzification

Defuzzification is a module or component, which takes the fuzzy set inputs generated by the **Inference Engine**, and then transforms them into a crisp value. It is the last step in the process of a fuzzy logic system. The crisp value is a type of value which is acceptable by the user. Various techniques are present to do this, but the user has to select the best one for reducing the errors.

Membership Function

The membership function is a function which represents the graph of fuzzy sets, and allows users to quantify the linguistic term. It is a graph which is used for mapping each element of x to the value between 0 and 1.

This function is also known as indicator or characteristics function.

This function of Membership was introduced in the first papers of fuzzy set by **Zadeh**. For the Fuzzy set B, the membership function for X is defined as: $\mu B:X \rightarrow [0,1]$. In this function X, each element of set B is mapped to the value between 0 and 1. This is called a degree of membership or membership value.

Classical and Fuzzy Set Theory

To learn about classical and Fuzzy set theory, firstly you have to know about what is set.

Set

A set is a term, which is a collection of unordered or ordered elements. Following are the various examples of a set:

1. A set of all-natural numbers
2. A set of students in a class.
3. A set of all cities in a state.
4. A set of upper-case letters of the alphabet.

Types of Set:

There are following various categories of set:

1. Finite

2. Empty
3. Infinite
4. Proper
5. Universal
6. Subset
7. Singleton
8. Equivalent Set
9. Disjoint Set

Classical Set

It is a type of set which collects the distinct objects in a group. The sets with the crisp boundaries are classical sets. In any set, each single entity is called an element or member of that set.

Mathematical Representation of Sets

Any set can be easily denoted in the following two different ways:

1. Roaster Form: This is also called as a tabular form. In this form, the set is represented in the following way:

```
Set_name = { element1, element2, element3, ......, element N}
```

The elements in the set are enclosed within the brackets and separated by the commas.

Following are the two examples which describes the set in Roaster or Tabular form:

Example 1:

```
Set of Natural Numbers: N={1, 2, 3, 4, 5, 6, 7, ......,n).
```

Example 2:

```
Set of Prime Numbers less than 50: X={2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47}.
```

2. Set Builder Form: Set Builder form defines a set with the common properties of an element in a set. In this form, the set is represented in the following way:

```
A = {x:p(x)}
```

The following example describes the set in the builder form:

The set {2, 4, 6, 8, 10, 12, 14, 16, 18} is written as:
B = {x:2 ≤ x < 20 and (x%2) = 0}

Operations on Classical Set

Following are the various operations which are performed on the classical sets:

1. Union Operation
2. Intersection Operation
3. Difference Operation
4. Complement Operation

1. Union:

This operation is denoted by (A U B). A U B is the set of those elements which exist in two different sets A and B. This operation combines all the elements from both the sets and make a new set. It is also called a Logical OR operation.

It can be described as:

A ∪ B = { x | x ∈ A OR x ∈ B }.

Example:

Set A = {10, 11, 12, 13}, Set B = {11, 12, 13, 14, 15}, then A ∪ B = {10, 11, 12, 13, 14, 15}

2. Intersection

This operation is denoted by (A ∩ B). A ∩ B is the set of those elements which are common in both set A and B. It is also called a Logical OR operation.

It can be described as:
A ∩ B = { x | x ∈ A AND x ∈ B }.

Example:

Set A = {10, 11, 12, 13}, Set B = {11, 12, 14} then A ∩ B = {11, 12}

3. Difference Operation

This operation is denoted by (A - B). A-B is the set of only those elements which exist only in set A but not in set B.

It can be described as:

$A - B = \{ x \mid x \in A \text{ AND } x \notin B \}$.

4. Complement Operation: This operation is denoted by (A\`). It is applied on a single set. A\` is the set of elements which do not exist in set A.

It can be described as:

$A' = \{x \mid x \notin A\}$.

Properties of Classical Set

There are following various properties which play an essential role for finding the solution of a fuzzy logic problem.

1. Commutative Property:

This property provides the following two states which are obtained by two finite sets A and B:

$A \cup B = B \cup A$
$A \cap B = B \cap A$

2. Associative Property:

This property also provides the following two states but these are obtained by three different finite sets A, B, and C:

$A \cup (B \cup C) = (A \cup B) \cup C$
$A \cap (B \cap C) = (A \cap B) \cap C$

3. Idempotency Property:

This property also provides the following two states but for a single finite set A:

$A \cup A = A$
$A \cap A = A$

4. Absorption Property

This property also provides the following two states for any two finite sets A and B:

$A \cup (A \cap B) = A$
$A \cap (A \cup B) = A$

5. Distributive Property:

This property also provides the following two states for any three finite sets A, B, and C:

$A\cup (B \cap C) = (A \cup B)\cap (A \cup C)$
$A\cap (B \cup C) = (A\cap B) \cup (A\cap C)$

6. Identity Property:

This property provides the following four states for any finite set A and Universal set X:

$A \cup \varphi = A$
$A \cap X = A$
$A \cap \varphi = \varphi$
$A \cup X = X$

7. Transitive property

This property provides the following state for the finite sets A, B, and C:

If $A \subseteq B \subseteq C$, then $A \subseteq C$

8. Ivolution property

This property provides following state for any finite set A:

$$\boxed{\overline{\overline{A}} = A}$$

9. De Morgan's Law

This law gives the following rules for providing the contradiction and tautologies:

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

Fuzzy Set

The set theory of classical is the subset of Fuzzy set theory. Fuzzy logic is based on this theory, which is a generalisation of the classical theory of set (i.e., crisp set) introduced by Zadeh in 1965.

A fuzzy set is a collection of values which exist between 0 and 1. Fuzzy sets are denoted or represented by the tilde (~) character. The sets of Fuzzy theory were introduced in 1965 by Lofti A. Zadeh and Dieter Klaua. In the fuzzy set, the partial membership also exists. This theory released as an extension of classical set theory.

This theory is denoted mathematically asA fuzzy set (Ã) is a pair of U and M, where U is the Universe of discourse and M is the membership function which takes on values in the interval [0, 1]. The universe of discourse (U) is also denoted by Ω or X.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$$

Operations on Fuzzy Set

Given Ã and B are the two fuzzy sets, and X be the universe of discourse with the following respective member functions:

$\mu_{\tilde{A}}(x)$ and $\mu_{\tilde{B}}(x)$

The operations of Fuzzy set are as follows:

1. Union Operation: The union operation of a fuzzy set is defined by:

```
μA∪B(x) = max (μA(x), μB(x))
```

Example:

Let's suppose A is a set which contains following elements:

A = {(X_1, 0.6), (X_2, 0.2), (X_3, 1), (X_4, 0.4)}

And, B is a set which contains following elements:

B = {(X_1, 0.1), (X_2, 0.8), (X_3, 0), (X_4, 0.9)}

then,

AUB = {(X_1, 0.6), (X_2, 0.8), (X_3, 1), (X_4, 0.9)}

Because, according to this operation

For X_1

$\mu_{A\cup B}(X_1) = \max (\mu_A(X_1), \mu_B(X_1))$
$\mu_{A\cup B}(X_1) = \max (0.6, 0.1)$
$\mu_{A\cup B}(X_1) = 0.6$

For X_2

$\mu_{A\cup B}(X_2) = \max (\mu_A(X_2), \mu_B(X_2))$
$\mu_{A\cup B}(X_2) = \max (0.2, 0.8)$
$\mu_{A\cup B}(X_2) = 0.8$

For X_3

$\mu_{A\cup B}(X_3) = \max (\mu_A(X_3), \mu_B(X_3))$
$\mu_{A\cup B}(X_3) = \max (1, 0)$
$\mu_{A\cup B}(X_3) = 1$

For X_4

$\mu_{A\cup B}(X_4) = \max (\mu_A(X_4), \mu_B(X_4))$
$\mu_{A\cup B}(X_4) = \max (0.4, 0.9)$
$\mu_{A\cup B}(X_4) = 0.9$

2. Intersection Operation: The intersection operation of fuzzy set is defined by:

$\mu_{A\cap B}(x) = \min (\mu_A(x), \mu_B(x))$

Example:

Let's suppose A is a set which contains following elements:

$A = \{(X_1, 0.3), (X_2, 0.7), (X_3, 0.5), (X_4, 0.1)\}$

And, B is a set which contains following elements:

$B = \{(X_1, 0.8), (X_2, 0.2), (X_3, 0.4), (X_4, 0.9)\}$

then,

$A \cap B = \{(X_1, 0.3), (X_2, 0.2), (X_3, 0.4), (X_4, 0.1)\}$

Because, according to this operation

For X_1

$\mu_{A\cap B}(X_1) = \min (\mu_A(X_1), \mu_B(X_1))$
$\mu_{A\cap B}(X_1) = \min (0.3, 0.8)$
$\mu_{A\cap B}(X_1) = 0.3$

For X_2

$\mu_{A\cap B}(X_2) = \min (\mu_A(X_2), \mu_B(X_2))$
$\mu_{A\cap B}(X_2) = \min (0.7, 0.2)$
$\mu_{A\cap B}(X_2) = 0.2$

For X_3

$\mu_{A\cap B}(X_3) = \min (\mu_A(X_3), \mu_B(X_3))$
$\mu_{A\cap B}(X_3) = \min (0.5, 0.4)$
$\mu_{A\cap B}(X_3) = 0.4$

For X_4

$\mu_{A\cap B}(X_4) = \min (\mu_A(X_4), \mu_B(X_4))$
$\mu_{A\cap B}(X_4) = \min (0.1, 0.9)$
$\mu_{A\cap B}(X_4) = 0.1$

3. Complement Operation: The complement operation of fuzzy set is defined by:

$\mu_{\bar{A}}(x) = 1-\mu_A(x)$,

Example:

Let's suppose A is a set which contains following elements:

$A = \{(X_1, 0.3), (X_2, 0.8), (X_3, 0.5), (X_4, 0.1)\}$

then,

$\bar{A} = \{(X_1, 0.7), (X_2, 0.2), (X_3, 0.5), (X_4, 0.9)\}$

Because, according to this operation

For X_1

$\mu_{\bar{A}}(X_1) = 1 - \mu_A(X_1)$
$\mu_{\bar{A}}(X_1) = 1 - 0.3$
$\mu_{\bar{A}}(X_1) = 0.7$

For X_2

$\mu_{\bar{A}}(X_2) = 1 - \mu_A(X_2)$
$\mu_{\bar{A}}(X_2) = 1 - 0.8$
$\mu_{\bar{A}}(X_2) = 0.2$

For X_3

$\mu_{\bar{A}}(X_3) = 1 - \mu_A(X_3)$
$\mu_{\bar{A}}(X_3) = 1 - 0.5$
$\mu_{\bar{A}}(X_3) = 0.5$

For X_4

$\mu_{\bar{A}}(X_4) = 1 - \mu_A(X_4)$
$\mu_{\bar{A}}(X_4) = 1 - 0.1$
$\mu_{\bar{A}}(X_4) = 0.9$

Classical Set Theory	Fuzzy Set Theory
1. This theory is a class of those sets having sharp boundaries.	1. This theory is a class of those sets having ur sharp boundaries.

2. This set theory is defined by exact boundaries only 0 and 1.	2. This set theory is defined by ambiguou boundaries.
3. In this theory, there is no uncertainty about the boundary's location of a set.	3. In this theory, there always exists uncertaint about the boundary's location of a set.
4. This theory is widely used in the design of digital systems.	4. It is mainly used for fuzzy controllers.

Applications of Fuzzy Logic

Following are the different application areas where the Fuzzy Logic concept is widely used:

1. It is used in **Businesses** for decision-making support system.
2. It is used in **Automative systems** for controlling the traffic and speed, and for improving the efficiency of automatic transmissions. **Automative systems** also use the shift scheduling method for automatic transmissions.
3. This concept is also used in the **Defence** in various areas. Defence mainly uses the Fuzzy logic systems for underwater target recognition and the automatic target recognition of thermal infrared images.
4. It is also widely used in the **Pattern Recognition and Classification** in the form of Fuzzy logic-based recognition and handwriting recognition. It is also used in the searching of fuzzy images.
5. Fuzzy logic systems also used in **Securities**.
6. It is also used in **microwave oven** for setting the lunes power and cooking strategy.
7. This technique is also used in the area of **modern control systems** such as expert systems.
8. **Finance** is also another application where this concept is used for predicting the stock market, and for managing the funds.
9. It is also used for controlling the brakes.
10. It is also used in the **industries of chemicals** for controlling the ph, and chemical distillation process.
11. It is also used in the **industries of manufacturing** for the optimization of milk and cheese production.
12. It is also used in the vacuum cleaners, and the timings of washing machines.

13. It is also used in heaters, air conditioners, and humidifiers.

Advantages of Fuzzy Logic

Fuzzy Logic has various advantages or benefits. Some of them are as follows:

1. The methodology of this concept works similarly as the human reasoning.
2. Any user can easily understand the structure of Fuzzy Logic.
3. It does not need a large memory, because the algorithms can be easily described with fewer data.
4. It is widely used in all fields of life and easily provides effective solutions to the problems which have high complexity.
5. This concept is based on the set theory of mathematics, so that's why it is simple.
6. It allows users for controlling the control machines and consumer products.
7. The development time of fuzzy logic is short as compared to conventional methods.
8. Due to its flexibility, any user can easily add and delete rules in the FLS system.

Disadvantages of Fuzzy Logic

Fuzzy Logic has various disadvantages or limitations. Some of them are as follows:

1. The run time of fuzzy logic systems is slow and takes a long time to produce outputs.
2. Users can understand it easily if they are simple.
3. The possibilities produced by the fuzzy logic system are not always accurate.
4. Many researchers give various ways for solving a given statement using this technique which leads to ambiguity.
5. Fuzzy logics are not suitable for those problems that require high accuracy.
6. The systems of a Fuzzy logic need a lot of testing for verification and validation.

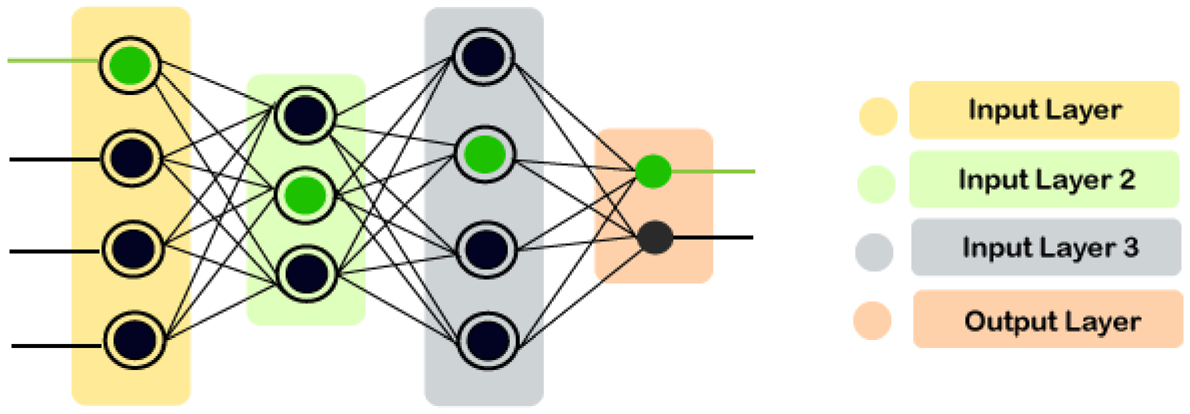
Artificial Neural Networks

At earlier times, the conventional computers incorporated algorithmic approach that is the computer used to follow a set of instructions to solve a problem unless those specific steps need that the computer need to follow are known the computer cannot solve a problem. So, obviously, a person is needed in order to solve the problems or someone who can provide instructions to the computer so as to how to solve that particular problem. It actually restricted the problem-solving capacity of conventional computers to problems that we already understand and know how to solve.

But what about those problems whose answers are not clear, so that is where our traditional approach face failure and so Neural Networks came into existence. Neural Networks processes information in a similar way the human brain does, and these networks actually learn from examples, you cannot program them to perform a specific task. They will learn only from past experiences as well as examples, which is why you don't need to provide all the information regarding any specific task. So, that was the main reason why neural networks came into existence.

Artificial Neural Network is biologically inspired by the neural network, which constitutes after the human brain.

Neural networks are modeled in accordance with the human brain so as to imitate their functionality. The human brain can be defined as a neural network that is made up of several neurons, so is the Artificial Neural Network is made of numerous perceptron.



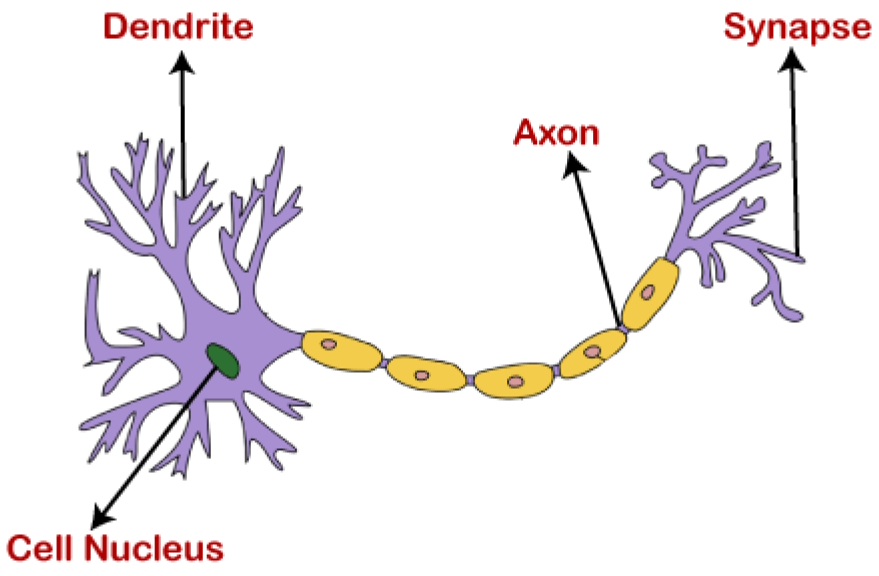
A neural network comprises of three main layers, which are as follows;

- **Input layer:** The input layer accepts all the inputs that are provided by the programmer.
- **Hidden layer:** In between the input and output layer, there is a set of hidden layers on which computations are performed that further results in the output.

- **Output layer:** After the input layer undergoes a series of transformations while passing through the hidden layer, it results in output that is delivered by the output layer.

Motivation behind Neural Network

Basically, the neural network is based on the neurons, which are nothing but the brain cells. A biological neuron receives input from other sources, combines them in some way, followed by performing a nonlinear operation on the result, and the output is the final result.



The **dendrites** will act as a receiver that receives signals from other neurons, which are then passed on to the **cell body**. The cell body will perform some operations that can be a summation, multiplication, etc. After the operations are performed on the set of input, then they are transferred to the next neuron via **axion**, which is the transmitter of the signal for the neuron.

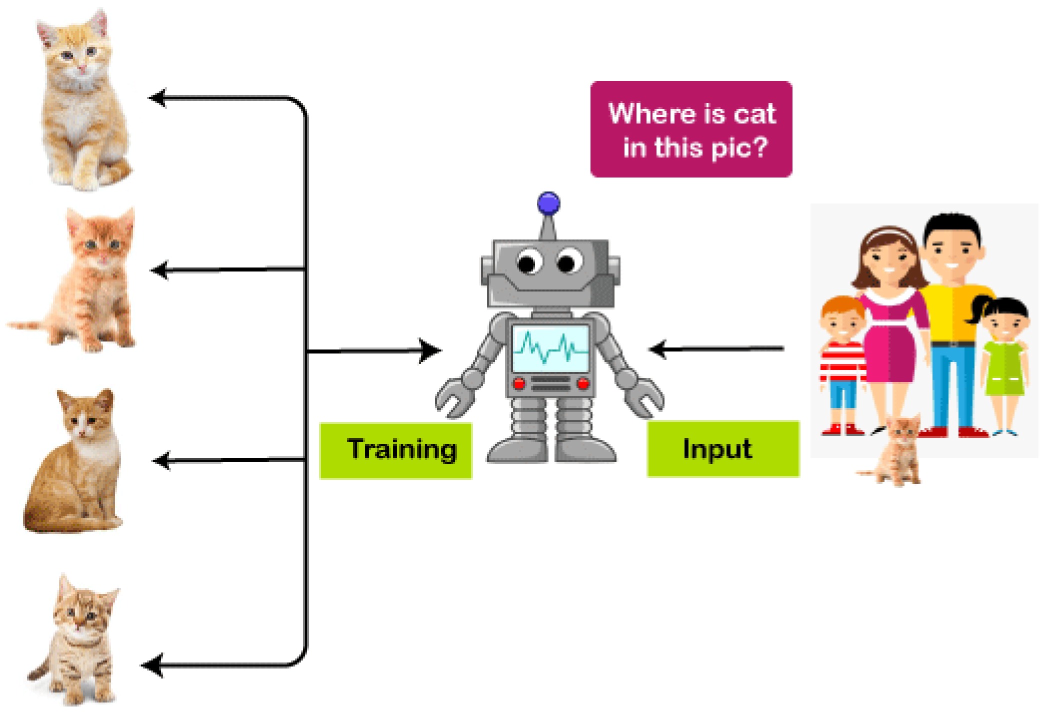
What are Artificial Neural Networks?

Artificial Neural Networks are the computing system that is designed to simulate the way the human brain analyzes and processes the information. Artificial Neural Networks have self-learning capabilities that enable it to produce a better result as more data become available. So, if the network is trained on more data, it will be more accurate because these neural networks learn from the examples. The neural network can be configured for specific applications like data classification, pattern recognition, etc.

With the help of the neural network, we can actually see that a lot of technology has been evolved from translating webpages to other languages to having a virtual assistant to order groceries online. All of these things are possible because of neural networks. So, an artificial neural network is nothing but a network of various artificial neurons.

Importance of Neural Network:

- **Without Neural Network:** Let's have a look at the example given below. Here we have a machine, such that we have trained it with four types of cats, as you can see in the image below. And once we are done with the training, we will provide a random image to that particular machine that has a cat. Since this cat is not similar to the cats through which we have trained our system, so without the neural network, our machine would not identify the cat in the picture. Basically, the machine will get confused in figuring out where the cat is.



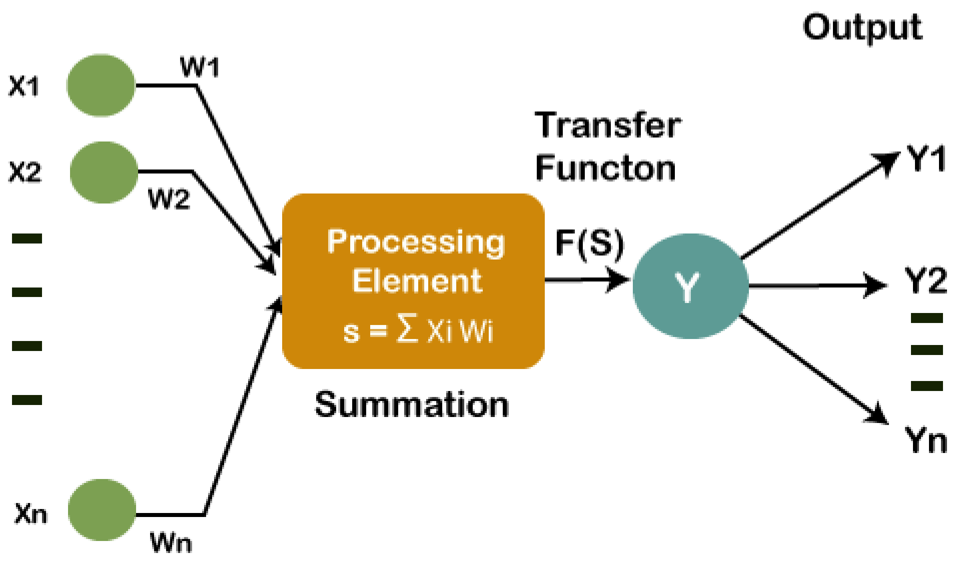
- **With Neural Network:** However, when we talk about the case with a neural network, even if we have not trained our machine with that particular cat. But still, it can identify certain features of a cat that we have trained on, and it can match those features with the cat that is there in that particular image and can also identify the cat. So, with the help of this example, you can clearly see the importance of the concept of a neural network.

Working of Artificial Neural Networks

Instead of directly getting into the working of Artificial Neural Networks, lets breakdown and try to understand Neural Network's basic unit, which is called a **Perceptron**.

So, a perceptron can be defined as a neural network with a single layer that classifies the linear data. It further constitutes four major components, which are as follows;

1. Inputs
2. Weights and Bias
3. Summation Functions
4. Activation or transformation function



The main logic behind the concept of Perceptron is as follows:

The inputs (x) are fed into the input layer, which undergoes multiplication with the allotted weights (w) followed by experiencing addition in order to form weighted sums. Then these inputs weighted sums with their corresponding weights are executed on the pertinent activation function.

Weights and Bias

As and when the input variable is fed into the network, a random value is given as a weight of that particular input, such that each individual weight represents the importance of that input in order to make correct predictions of the result.

However, bias helps in the adjustment of the curve of activation function so as to accomplish a precise output.

Summation Function

After the weights are assigned to the input, it then computes the product of each input and weights. Then the weighted sum is calculated by the summation function in which all of the products are added.

Activation Function

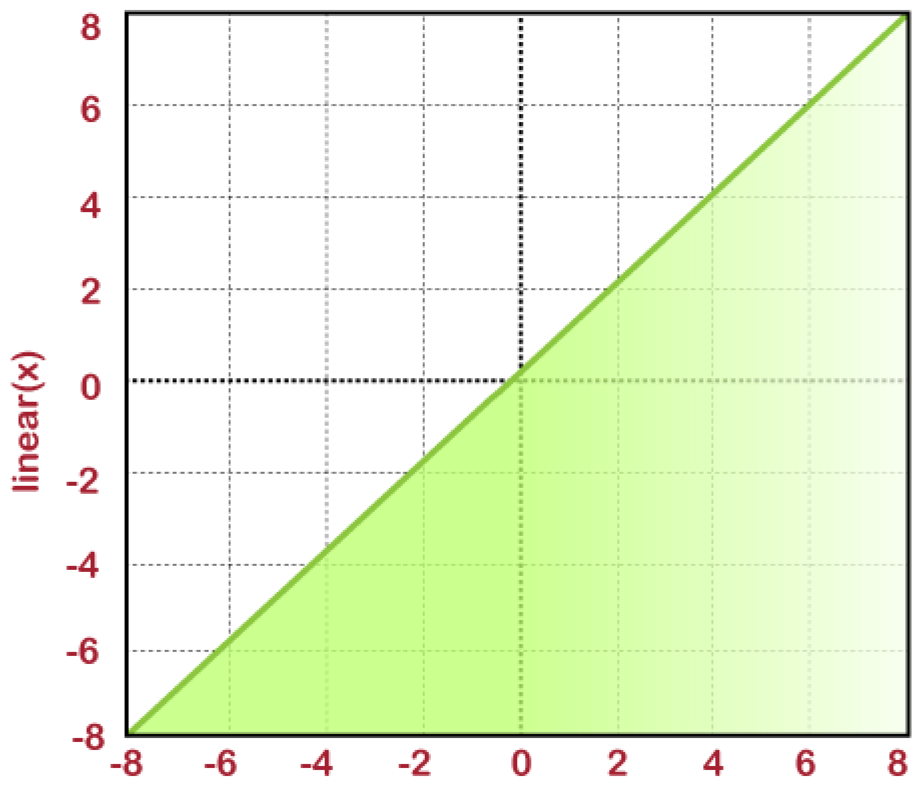
The main objective of the activation function is to perform a mapping of a weighted sum upon the output. The transformation function comprises of activation functions such as tanh, ReLU, sigmoid, etc.

The activation function is categorized into two main parts:

1. Linear Activation Function
2. Non-Linear Activation Function

Linear Activation Function

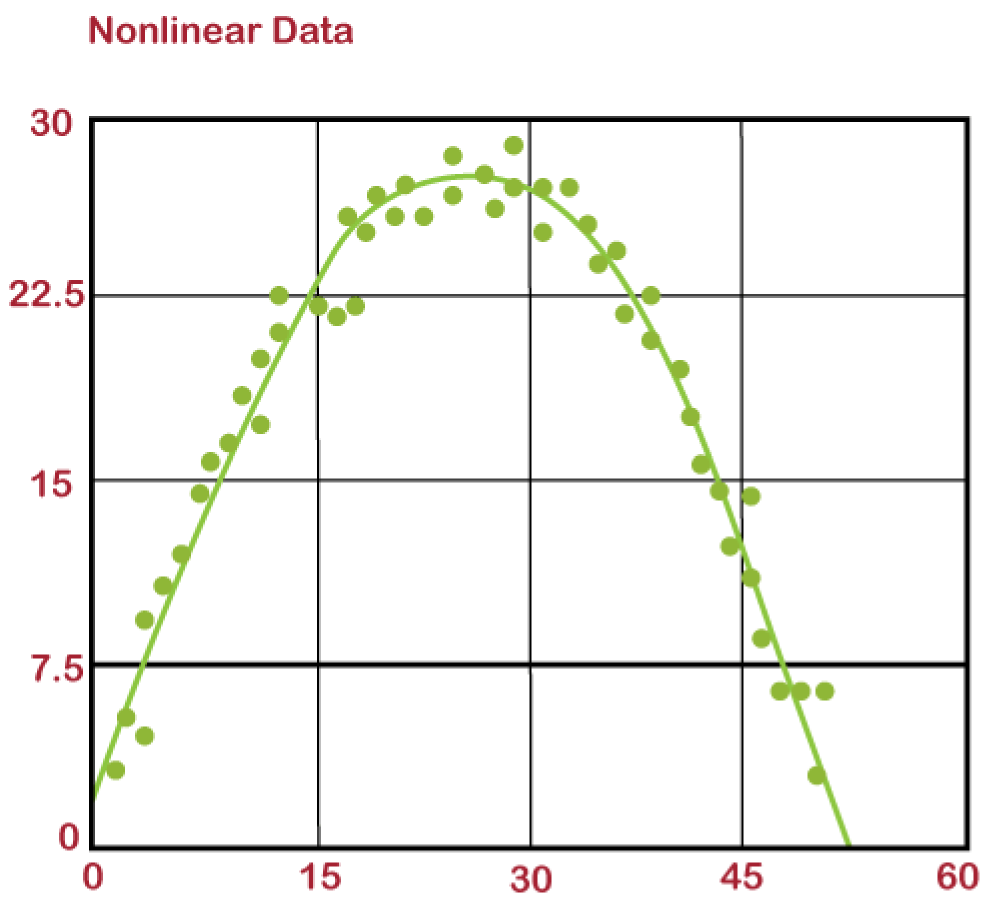
In the linear activation function, the output of functions is not restricted in between any range. Its range is specified from -infinity to infinity. For each individual neuron, the inputs get multiplied with the weight of each respective neuron, which in turn leads to the creation of output signal proportional to the input. If all the input layers are linear in nature, then the final activation of the last layer will actually be the linear function of the initial layer's input.



Non- linear function

These are one of the most widely used activation function. It helps the model in generalizing and adapting any sort of data in order to perform correct differentiation among the output. It solves the following problems faced by linear activation functions:

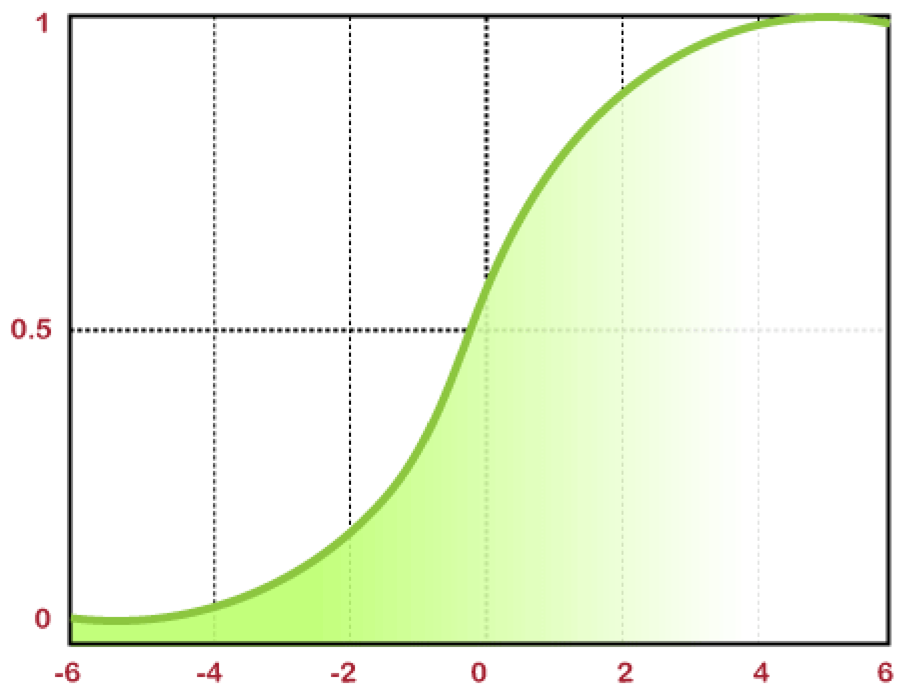
- Since the non-linear function comes up with derivative functions, so the problems related to backpropagation has been successfully solved.
- For the creation of deep neural networks, it permits the stacking up of several layers of the neurons.



The non-linear activation function is further divided into the following parts:

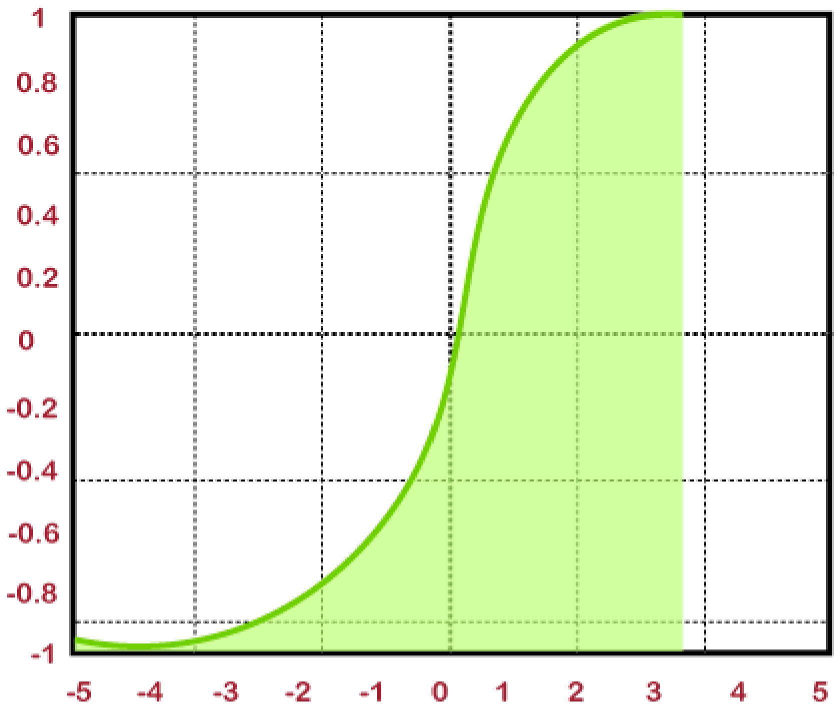
1. **Sigmoid or Logistic Activation Function**
   It provides a smooth gradient by preventing sudden jumps in the output values. It has an output value range between 0 and 1 that helps in the normalization of each neuron's output. For X, if it has a value above 2 or below -2, then the values of y will be much steeper. In simple language, it means that even a small change in the X can bring a lot of change in Y.
   It's value ranges between 0 and 1 due to which it is highly preferred by binary

classification whose result is either 0 or 1.



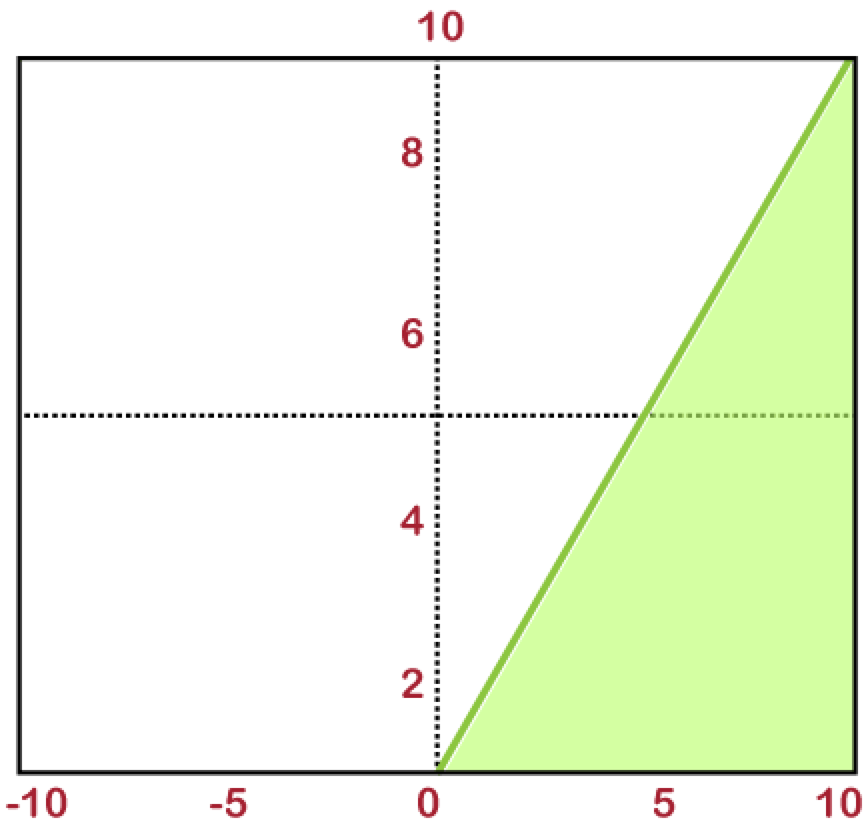
2. **Tanh or Hyperbolic Tangent Activation Function**
The tanh activation function works much better than that of the sigmoid function, or simply we can say it is an advanced version of the sigmoid activation function. Since it has a value range between -1 to 1, so it is utilized by the hidden layers in the neural network, and because of this reason, it has made the process

of learning much easier.



3. **ReLU(Rectified Linear Unit) Activation Function**
ReLU is one of the most widely used activation function by the hidden layer in the neural network. Its value ranges from 0 to infinity. It clearly helps in solving out the problem of backpropagation. It tends out to be more expensive than the sigmoid, as well as the tanh activation function. It allows only a few neurons to get activated at a particular instance that leads to effectual as well as easier

computations.



4. **Softmax Function**
   It is one of a kind of sigmoid function whereby solving the problems of classifications. It is mainly used to handle multiple classes for which it squeezes the output of each class between 0 and 1, followed by dividing it by the sum of outputs. This kind of function is specially used by the classifier in the output layer.

Gradient Descent Algorithm

Gradient descent is an optimization algorithm that is utilized to minimize the cost function used in various machine learning algorithms so as to update the parameters of the learning model. In linear regression, these parameters are coefficients, whereas, in the neural network, they are weights.

Procedure:

It all starts with the coefficient's initial value or function's coefficient that may be either 0.0 or any small arbitrary value.

```
coefficient = 0.0
```

For estimating the cost of the coefficients, they are plugged into the function that helps in evaluating.

```
cost = f(coefficient)
or, cost = evaluate(f(coefficient))
```

Next, the derivate will be calculated, which is one of the concepts of calculus that relates to the function's slope at any given instance. In order to know the direction in which the values of the coefficient will move, we need to calculate the slope so as to accomplish a low cost in the next iteration.

```
delta = derivative(cost)
```

Now that we have found the downhill direction, it will further help in updating the values of coefficients. Next, we will need to specify alpha, which is a learning rate parameter, as it handles the amount of amendments made by coefficients on each update.

```
coefficient = coefficient - (alpha * delta)
```

Until the cost of the coefficient reaches **0.0** or somewhat close enough to it, the whole process will reiterate again and again.

It can be concluded that gradient descent is a very simple as well as straightforward concept. It just requires you to know about the gradient of the cost function or simply the function that you are willing to optimize.

Batch Gradient Descent

For every repetition of gradient descent, the main aim of batch gradient descent is to processes all of the training examples. In case we have a large number of training examples, then batch gradient descent tends out to be one of the most expensive and less preferable too.

Algorithm for Batch Gradient Descent

Let **m** be the number of training examples and **n** be the number of features.

Now assume that **h_Θ** represents the hypothesis for linear regression and **$\sum$** computes the sum of all training examples from **i=1 to m**. Then the cost of function will be computed by:

$J_{train}(\Theta) = (1/2m) \sum (h_\Theta(x^{(i)}) - (y^{(i)})^2$

```
Repeat {

$\Theta j = \Theta j$ - (learning rate/m) * $\sum (h_\Theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$

For every j = 0…n

}
```

Here **$x^{(i)}$** indicates the **j**th feature of the **i**th training example. In case if **m** is very large, then derivative will fail to converge at a **global minimum**.

Stochastic Gradient Descent

At a single repetition, the stochastic gradient descent processes only one training example, which means it necessitates for all the parameters to update after the one single training example is processed per single iteration. It tends to be much faster than that of the batch gradient descent, but when we have a huge number of training examples, then also it processes a single example due to which system may undergo a large no of repetitions. To evenly train the parameters provided by each type of data, properly shuffle the dataset.

Algorithm for Stochastic Gradient Descent

```
Suppose that $(x^{(i)}, y^{(i)})$ be the training example

Cost $(\Theta, (x^{(i)}, y^{(i)})) = (1/2) \sum (h\Theta(x^{(i)}) - (y^{(i)})^2$

$J_{train} (\Theta) = (1/m) \sum$ Cost $(\Theta, (x^{(i)}, y^{(i)}))$

Repeat {

For i=1 to m{

$\Theta j = \Theta j$ - (learning rate) * $\sum (h_\Theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$

For every j=0…n

}

}
```

Convergence trends in different variants of Gradient Descent

The Batch Gradient Descent algorithm follows a straight-line path towards the minimum. The algorithm converges towards the **global minimum**, in case the cost function is **convex**, else towards the **local minimum**, if the cost function is not convex. Here the learning rate is typically constant.

However, in the case of Stochastic Gradient Descent, the algorithm fluctuates all over the global minimum rather than converging. The learning rate is changed slowly so that it can converge. Since it processes only one example in one iteration, it tends out to be noisy.

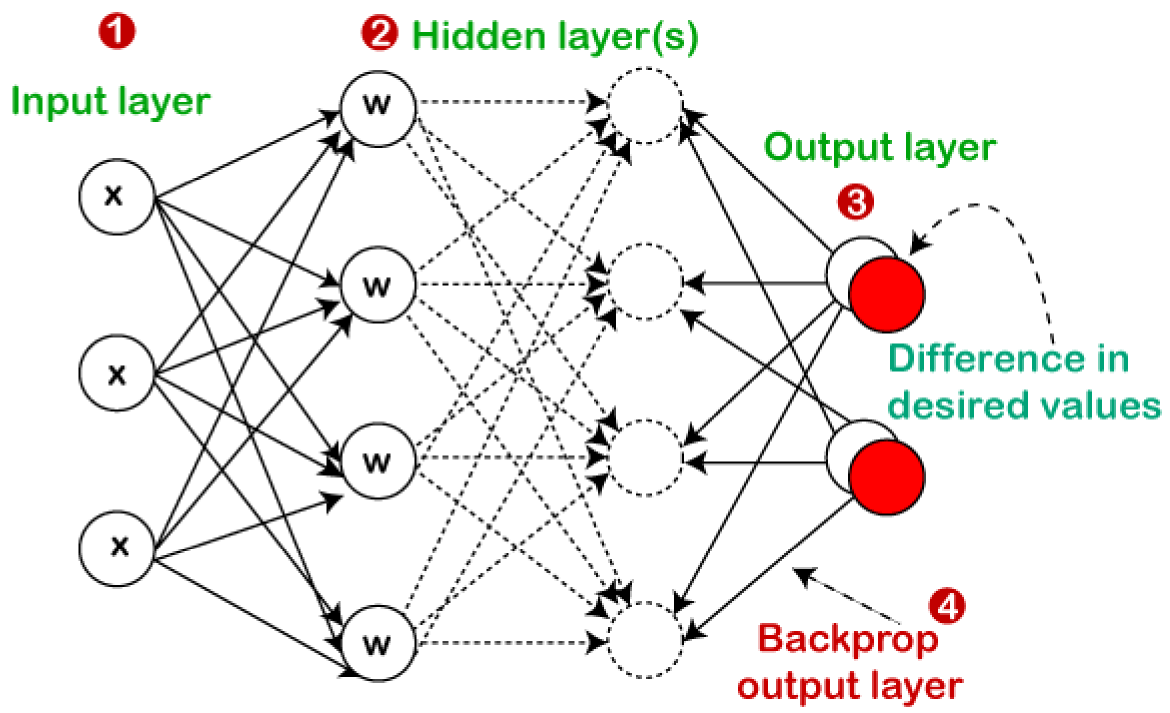
Backpropagation

The backpropagation consists of an input layer of neurons, an output layer, and at least one hidden layer. The neurons perform a weighted sum upon the input layer, which is then used by the activation function as an input, especially by the sigmoid activation function. It also makes use of supervised learning to teach the network. It constantly updates the weights of the network until the desired output is met by the network. It includes the following factors that are responsible for the training and performance of the network:

- Random (initial) values of weights.
- A number of training cycles.
- A number of hidden neurons.
- The training set.
- Teaching parameter values such as learning rate and momentum.

Working of Backpropagation

Consider the diagram given below.



1. The preconnected paths transfer the inputs **X**.
2. Then the weights **W** are randomly selected, which are used to model the input.
3. After then, the output is calculated for every individual neuron that passes from the input layer to the hidden layer and then to the output layer.
4. Lastly, the errors are evaluated in the outputs. **Error_B= Actual Output - Desired Output**
5. The errors are sent back to the hidden layer from the output layer for adjusting the weights to lessen the error.
6. Until the desired result is achieved, keep iterating all of the processes.

Need of Backpropagation

- Since it is fast as well as simple, it is very easy to implement.
- Apart from no of inputs, it does not encompass of any other parameter to perform tuning.
- As it does not necessitate any kind of prior knowledge, so it tends out to be more flexible.
- It is a standard method that results well.