# Machine Learning
# CSC261

# Introduction to Machine Learning

- Learning is a **complex process** that involves **acquiring knowledge, skills, or attitudes** through various **experiences and interactions.**
- Here are some key concepts related to learning:
  - **Acquisition**: The process of obtaining new information or skills and incorporating them into one's existing knowledge or abilities.
  - **Understanding**: The ability to comprehend and make sense of the acquired knowledge, grasping the underlying principles or concepts.
  - **Retention**: The capacity to remember and store the acquired information or skills over time.

# Introduction to Machine Learning

- Here are some key concepts related to learning:
  - **Application**: The practical use or implementation of learned knowledge or skills in real-life situations to solve problems or achieve goals.
  - **Adaptation**: The ability to adjust or modify one's existing knowledge or behavior based on new experiences or information.
  - **Transfer**: The application of previously learned knowledge or skills to new, unfamiliar situations or contexts.
  - **Feedback**: The information received about the performance or outcome of learning, which helps in evaluating and improving one's understanding or abilities.

# Introduction to Machine Learning

- Here are some key concepts related to learning:
  - **Feedback**: The information received about the performance or outcome of learning, which helps in evaluating and improving one's understanding or abilities.
  - **Reinforcement**: The use of positive or negative stimuli to encourage desired behaviors or discourage undesirable ones, enhancing the learning process.
  - **Practice**: Repetition and application of learned knowledge or skills to reinforce understanding and develop proficiency.

# Introduction to Machine Learning

- Here are some key concepts related to learning:
  - **Motivation**: The internal or external factors that drive an individual's desire and commitment to learn, such as curiosity, interest, rewards, or personal goals.
  - **Collaboration**: Learning through interactions and cooperation with others, exchanging ideas, perspectives, and feedback to enhance understanding and problem-solving abilities.
  - **Reflection**: The process of consciously thinking about and analyzing one's learning experiences, identifying strengths, weaknesses, and areas for improvement.

# Introduction to Machine Learning

- Here are some key concepts related to learning:
  - **Meta-cognition:** The awareness and understanding of one's own thinking processes, enabling self-regulation and strategic learning approaches.
  - **Continuous learning**: The recognition that learning is an ongoing, lifelong process, and the commitment to acquiring new knowledge and skills throughout one's life.
- These concepts provide a foundational understanding of the multifaceted nature of learning and can help inform educational practices and strategies to optimize the learning experience.

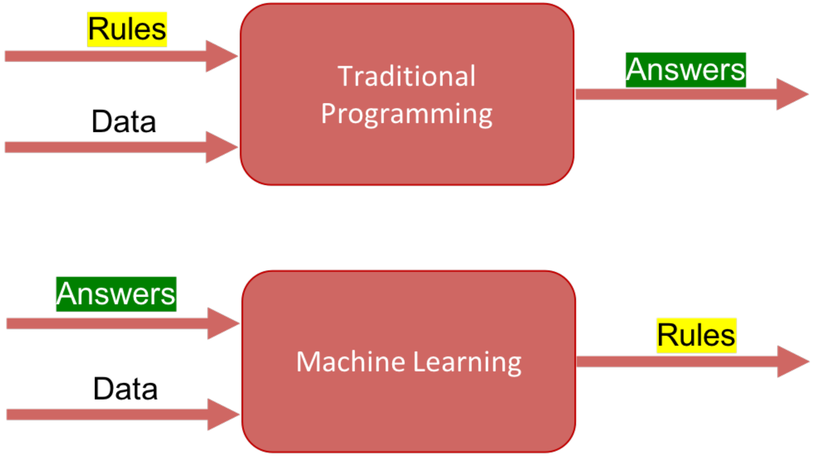
# Introduction to Machine Learning

- Why Learning?
  - To understand and improve the efficiency of human learning
  - To discover new knowledge
  - To fill the incomplete specifications of respective domain
  - Adaptation and resilience
  - Career advancement
  - Innovation and progress
  - Continuous improvement of overall domain and so on…..!

  ***Overall, learning is a vital process that enables personal development, societal progress, and the pursuit of knowledge and understanding.***

  ***It empowers individuals to adapt, grow, and thrive in an ever-changing world.***

- What is Machine Learning?




Machine learning is an application of AI that enables systems to learn and improve from experience without being explicitly programmed.

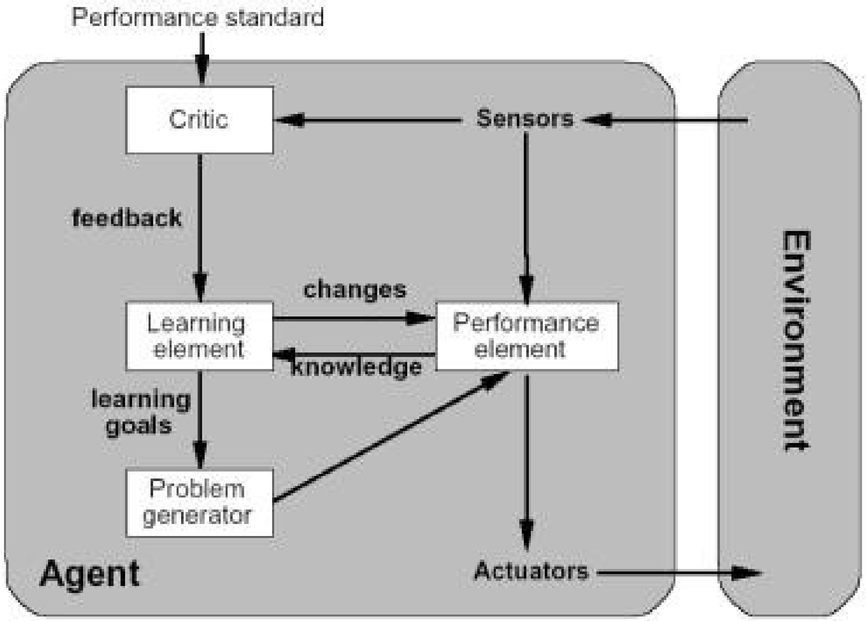
- What is Machine Learning?
  - Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. - IBM
  - *“ Field of study that gives computers the ability to learn without being explicitly programmed. ”*

- Why is Machine Learning?
  - An agent is learning if it improves its performance on future tasks after making observations about the world and past experience.
  - *Why should we want an agent to learn?*
  - First, the designers **cannot anticipate all possible situations** that the agent might find itself in. For example, a robot designed to navigate mazes must learn the layout of each new maze it encounters.
  - Second, the **designers cannot anticipate all changes over time**; a program designed to predict tomorrow's stock market prices must learn to adapt when conditions change from boom to bust.

- # Why is Machine Learning?
  - *Why should we want an agent to learn?*
  - Third, sometimes human programmers have no idea how to program a solution themselves. For example, most people are good at recognizing the faces of family members, but even the best programmers are unable to program a computer to accomplish that task, except by using learning algorithms.

# Introduction to Machine Learning

- ***Learning Framework***
  - *A Machine Learning Framework is an interface, library or tool which allows developers to more easily and quickly build machine learning models.*
  - *Learning framework is required because it provides a clear, concise way for defining machine learning models using a collection of pre-built, optimized components.*



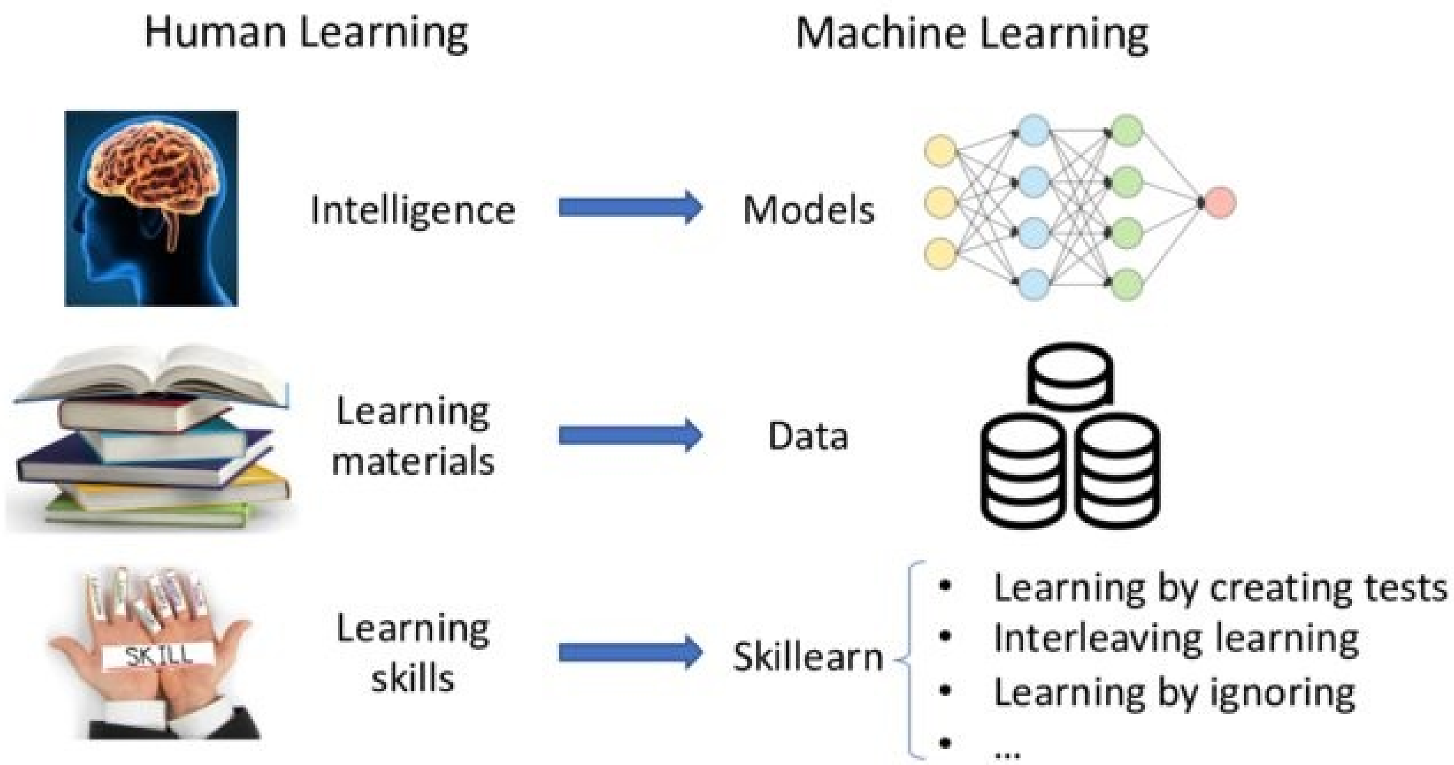
# Introduction to Machine Learning

- ***ML framework consists of the following components:***
  - ***Learning element:*** *This element is responsible for making improvements.*
  - ***Performance element:*** *It is responsible for selecting external actions according to the percepts it takes.*
  - ***Critic:*** *It provides feedback to the learning agent about how well the agent is doing, which could maximize the performance measure in the future.*
  - ***Problem Generator:*** *It suggests actions which could lead to new and informative experiences.*

# Introduction to Machine Learning

- ***ML framework consists of the following components:***
  - ***Example:***
    - *Performance element: turning, accelerating, breaking are the performance elements on road.*
    - *Learning element: learn rules for breaking, accelerating, learn geography of the city.*
    - *Critic: quick right turn across three lanes of traffic, observes reaction of other drivers.*
    - *Problem generator: Try south city road.*

# Introduction to Machine Learning

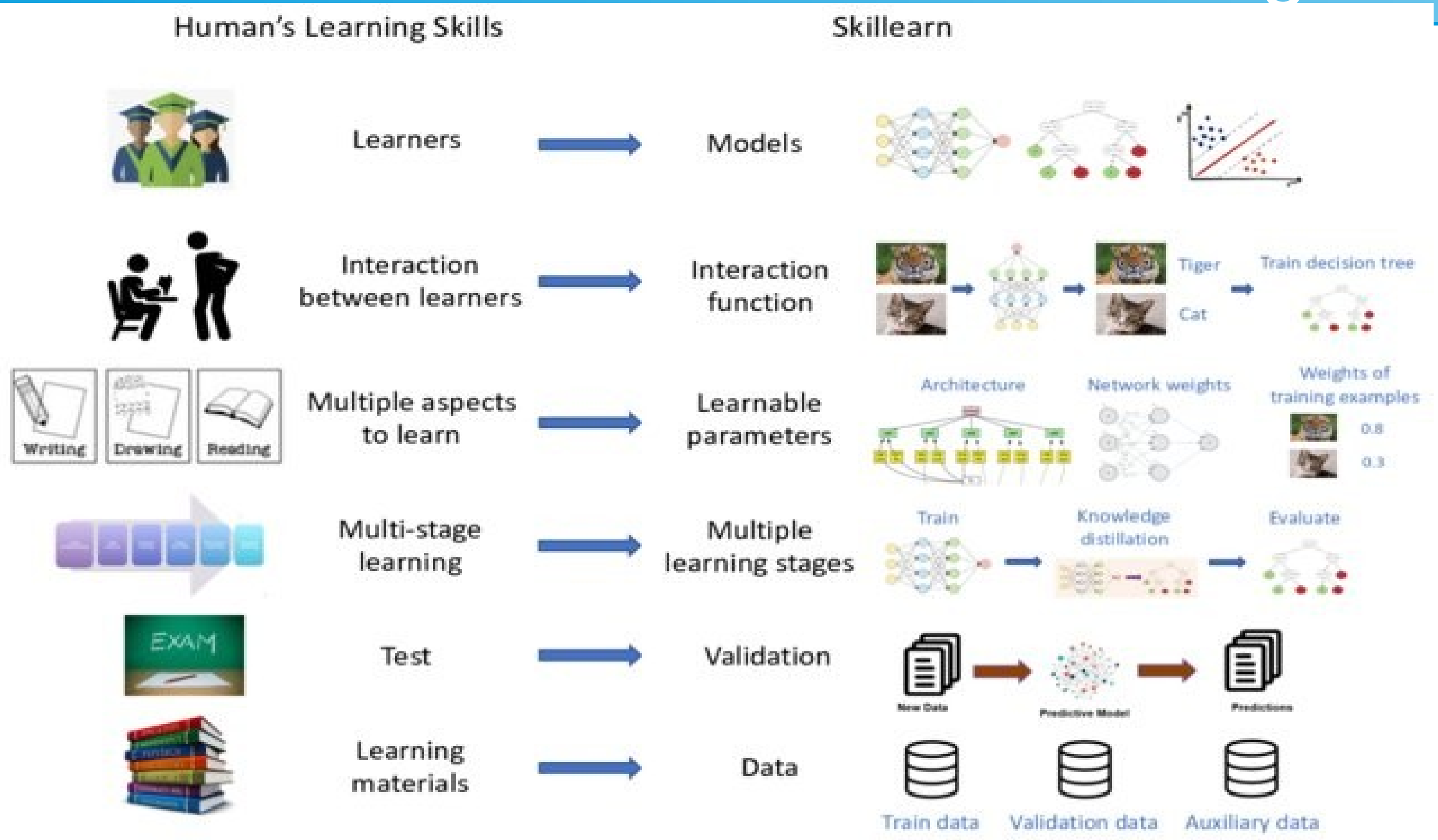



Figure[1] : Human learning (HL) versus machine learning (ML).



[1] https://www.researchgate.net/publication/346857783_Skillearn_Machine_Learning_Inspired_by_Humans%27_Learning_Skills/figures?lo=1

# Introduction to Machine Learning




Figure[2] : The elements of Skillearn and their counterparts in human learning.

[2] https://www.researchgate.net/publication/346857783_Skillearn_Machine_Learning_Inspired_by_Humans%27_Learning_Skills/figures?lo=1

# Introduction to Machine Learning

- Machine Learning(ML) is a subset of Artificial Intelligence(AI), defined as:
    - a computer's ability to learn from data by using algorithms to imitate intelligent human behavior of decision making and predictions.
- In other words, ML is the science of making computers learn and act like humans by feeding data and information without being explicitly programmed.
- Machine learning algorithms are trained with training data. When new data comes in, they can make predictions and decisions accurately based on past data.

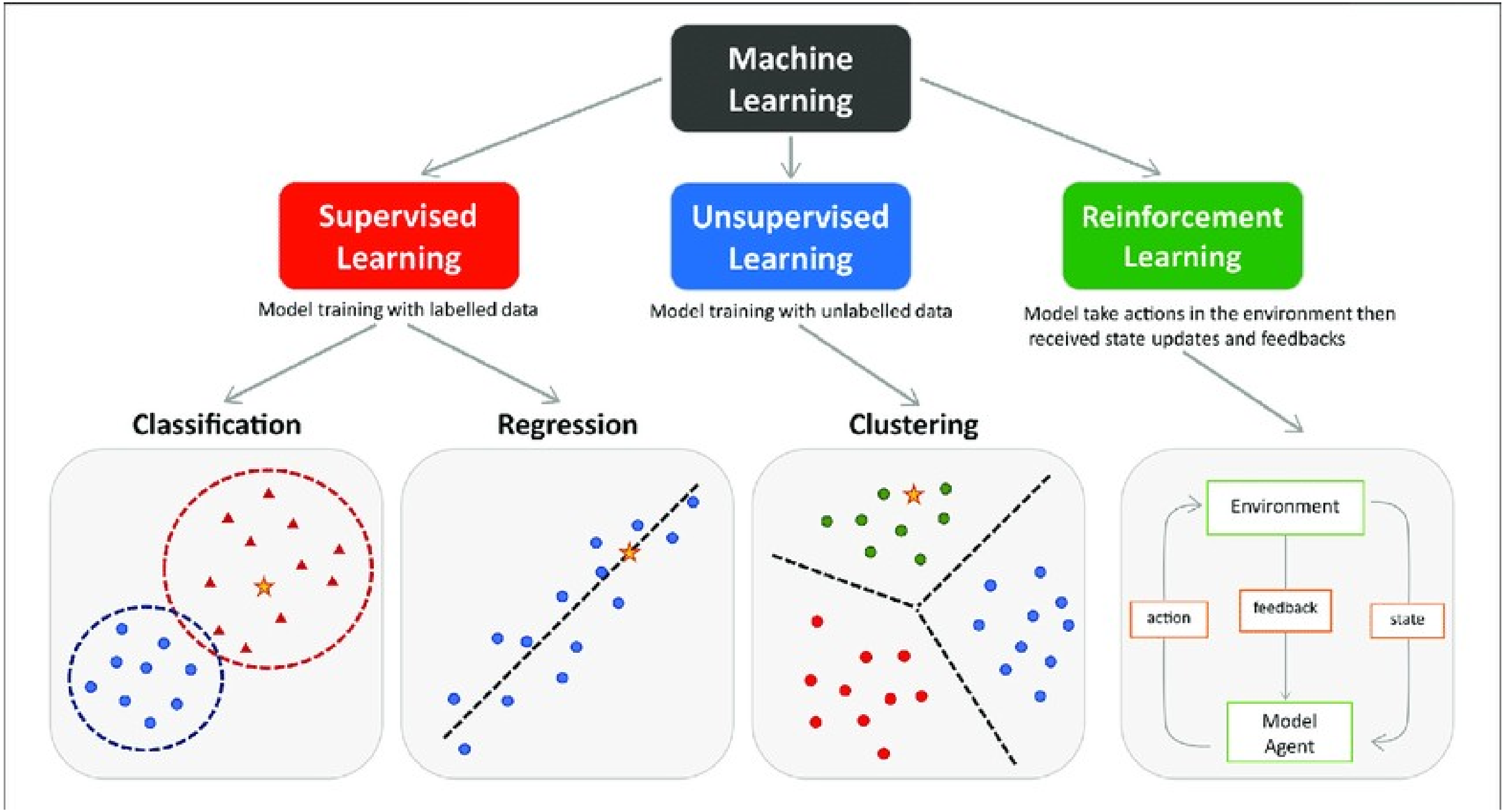
# Introduction to Machine Learning

- For example, whenever you ask Siri / Google Assistant to do something, a powerful speech recognition converts the audio into its corresponding textual form.
- This is sent to the Apple servers / Google server for further processing where language processing algorithms are run to understand the user's intent.
- Then finally, Siri / Google assistant tells you the answer.

# Introduction to Machine Learning

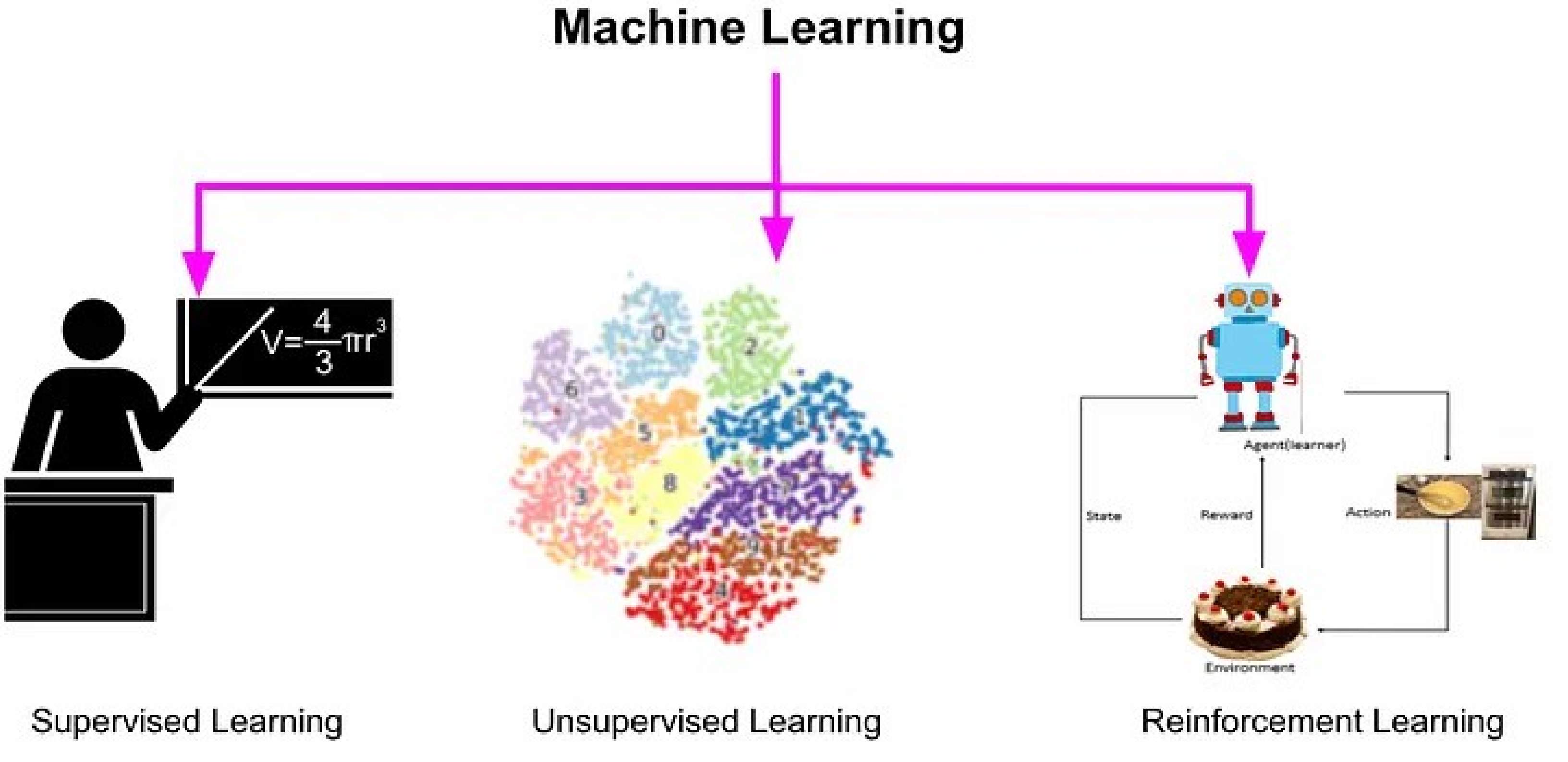
- There are three main groups of algorithms in ML:
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning

# Introduction to Machine Learning

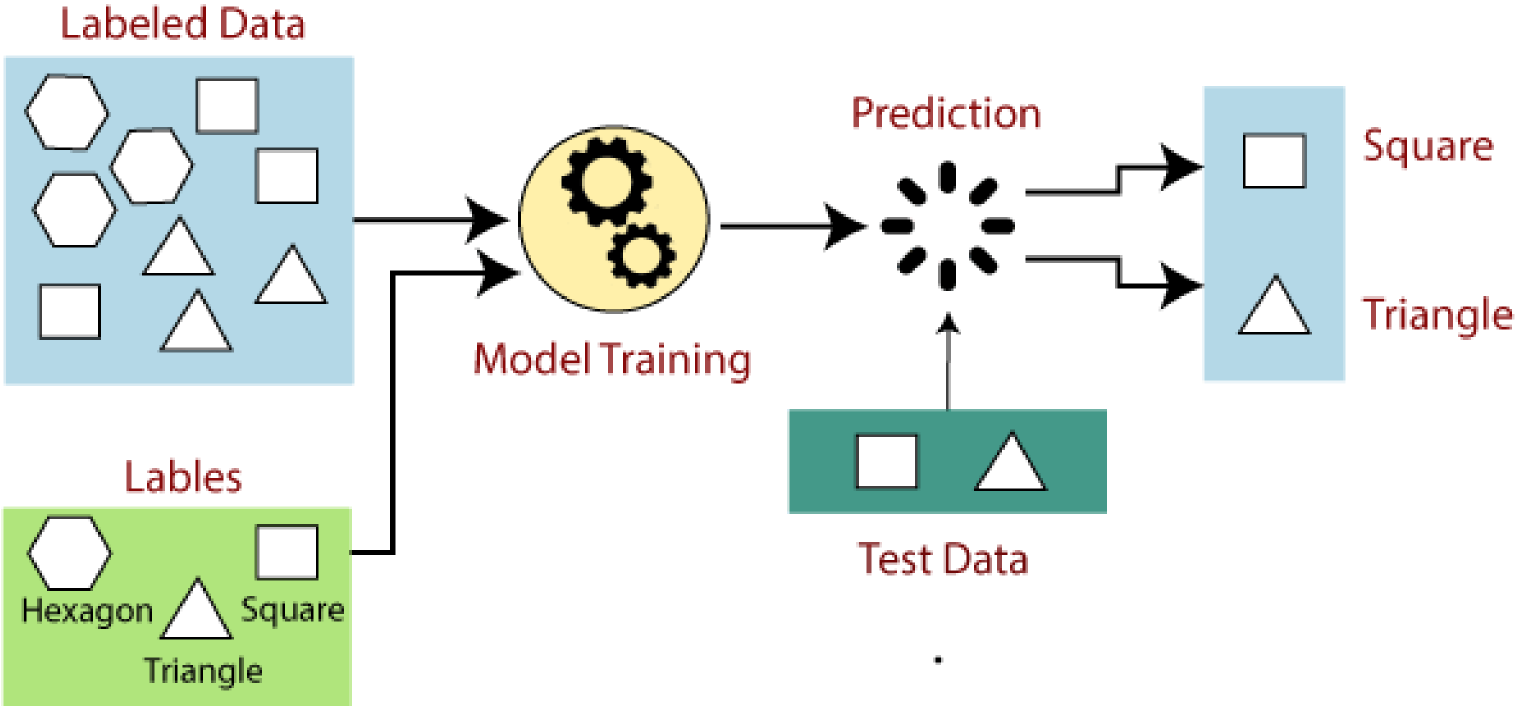


# Introduction to Machine Learning

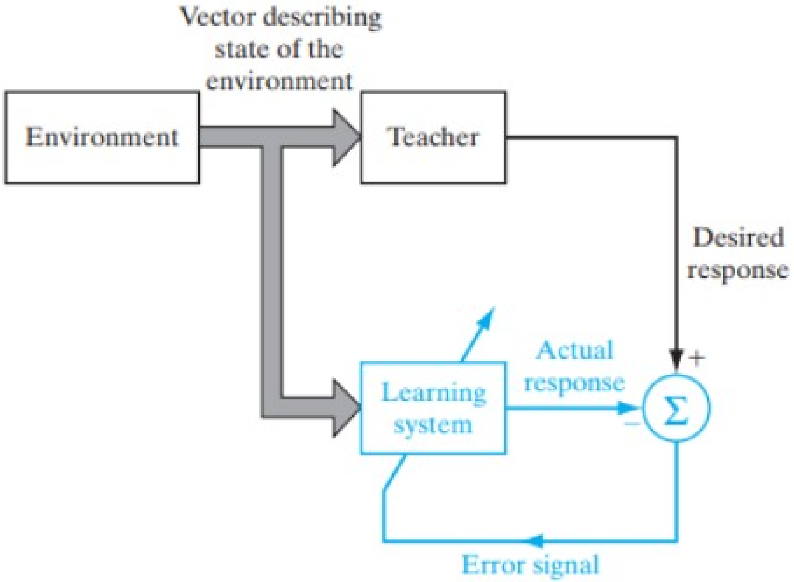
- There are three main groups of algorithms in ML:



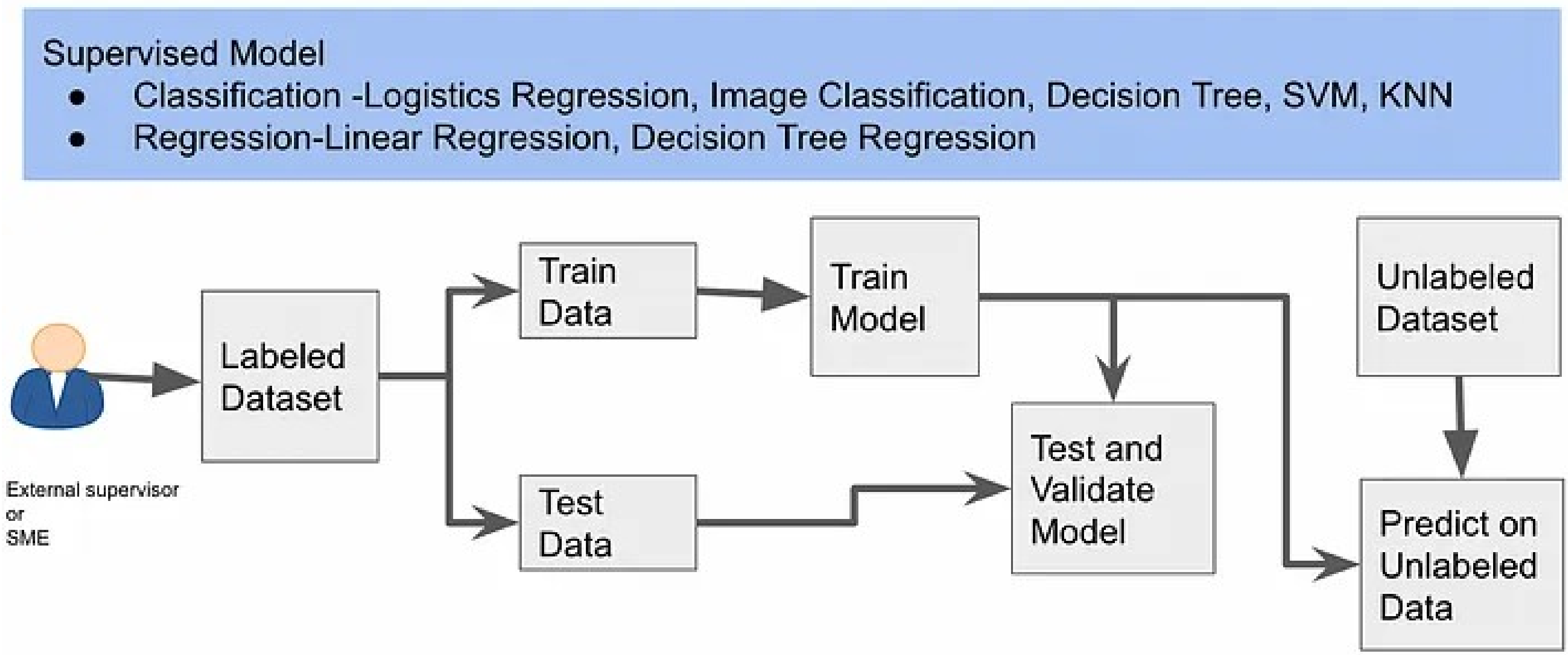
- **Supervised Learning:**



- **Supervised Learning:**
  - Learning with a teacher is also referred to as supervised learning.



- **Supervised Learning:**
  - In Supervised Learning, the machine learns under supervision.
  - It contains a model that is able to predict with the help of a labeled dataset.
  - A labeled dataset is one where you already know the target answer.
  - Input and output data is provided to a Supervised machine model, so supervised learning is learning by example.

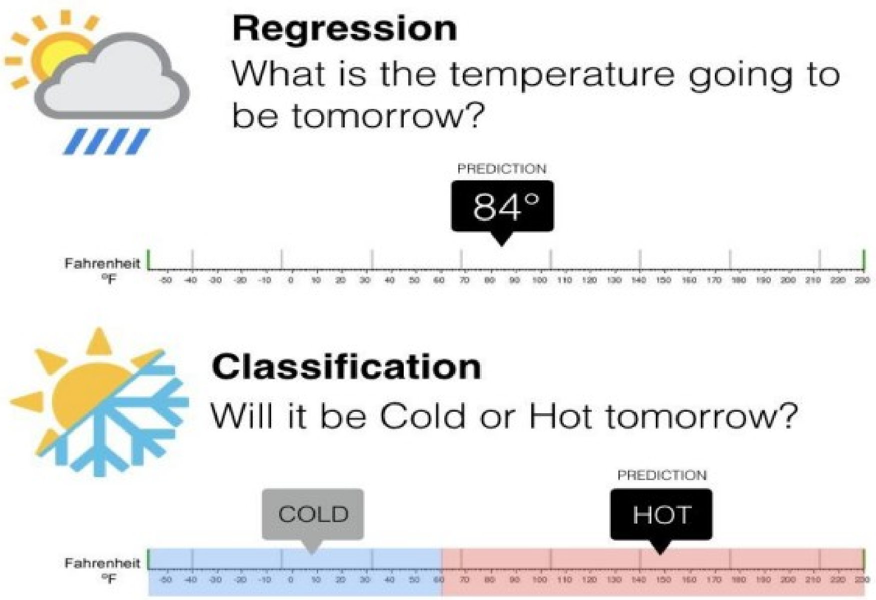

Supervised Model
● Classification -Logistics Regression, Image Classification, Decision Tree, SVM, KNN
● Regression-Linear Regression, Decision Tree Regression
External supervisor
or
SME
Labeled Dataset
Train Data
Test Data
Train Model
Test and Validate Model
Unlabeled Dataset
Predict on Unlabeled Data

# Introduction to Machine Learning

- **Supervised Learning:**
  - You are learning python using manuals and online tutorials by following the code examples.
  - Supervised learning is where you learn python by
    - understanding its features by practicing the examples that act as labeled data and
    - then using the knowledge acquired to write python programs for unseen use cases.

- **Supervised Learning:**
  - Supervised learning uses a labeled dataset, typically labeled by an external supervisor, subject matter expert(SME), or an algorithm/program.
  - The dataset is split into training and test dataset for training and then validating the model.
  - The supervised learned model is then used to generate predictions on previously unseen unlabeled data that belongs to the category of data the model was trained on.

- **Examples of Supervised Learning:**

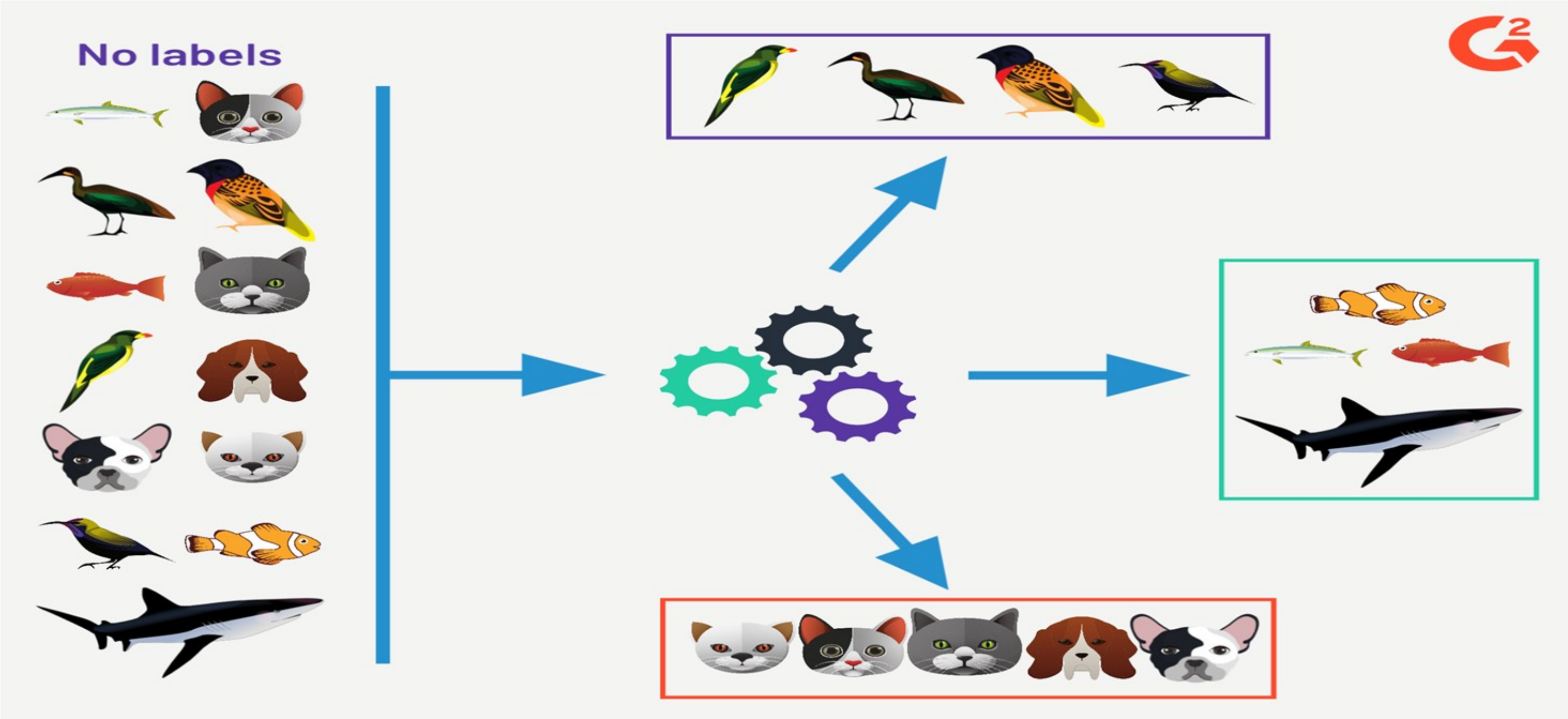


- **Examples of Supervised Learning:**
  - Examples of Supervised Learning are
    - Classification and
    - Regression
  - Classification is used in applications like Image Classification and K- Nearest Neighbors for identifying customer churn. Regression algorithms are used to predict sales, home prices, etc.

# Introduction to Machine Learning

- **Supervised Learning Algorithms:**
  - Linear regression
  - Logistic regression
  - Naive Bayes
  - Linear discriminant analysis
  - Decision trees
  - K-nearest neighbor algorithm
  - Neural networks (Multilayer perceptron)
  - Random forest algorithm
  - Support Vector Machine etc.

- **Unsupervised Learning:**



- **Unsupervised Learning:**
  - In Unsupervised Learning, the machine uses unlabeled data and learns on itself without any supervision.
  - The machine tries to find a pattern in the unlabeled data and gives a response.
  - Unsupervised learning identifies hidden patterns and relationships in an unlabeled dataset by grouping data into clusters or by association.

- **Unsupervised Learning:**

Unsupervised Model

- Clustering-K-Means Clustering, Hierarchical Clustering, Agglomerative Clustering
- Association - Recommendation system
- Dimensionality Reduction - Principal Component Analysis(PCA)
- Anomaly detection

# Introduction to Machine Learning

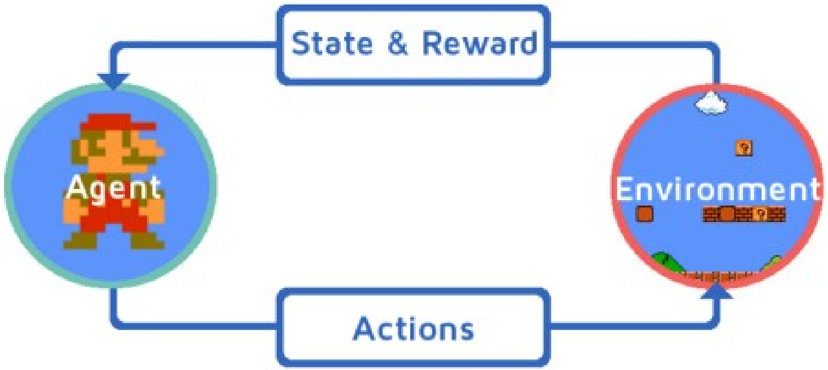
- **Unsupervised Learning:**
    - A child playing with toys can arrange them by identifying patterns based on colors, shapes, sizes, or just based on their interests.
    - The kid discovers new ways to cluster the toys without needing external supervision is similar to unsupervised learning.
- Unsupervised learning is learning by reasoning to identify hidden patterns in an unlabeled dataset.
- They have no supervision like the Supervised algorithms and are hence Unsupervised.

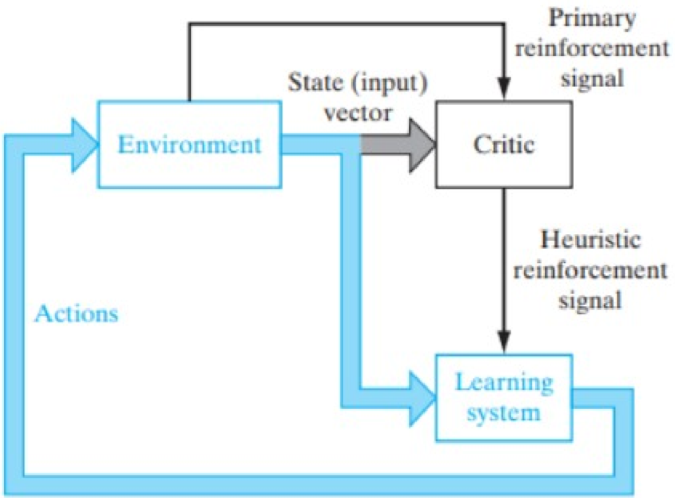
- **Examples of Unsupervised Learning:**
  - **Clustering** identifies hidden data patterns in an uncategorized/unlabeled dataset based on the similarities or differences in the dataset as in market segmentation.
  - **Association rules** allow you to establish associations amongst data objects inside large datasets by identifying relationships between variables in a given dataset, as in market basket analysis and recommendation engines.
  - **Anomaly detection** is an unsupervised algorithm to identify anomalous data in a dataset. It is used for fault diagnosis, network security intrusion, and fraud detection.

- **Unsupervised Learning Algorithms:**
  - K-means clustering
  - Hierarchical clustering
  - Gaussian Mixture Models
  - Apriori algorithms
  - FP Growth
  - Principal Component Analysis
  - Singular Value Decomposition
  - Autoencoders
  - Local Outlier Factor
  - Expected-Maximization

- **Reinforcement Learning:**
  - In reinforcement learning the agent learns **from a series of reinforcements—rewards or punishments.**
  - For example, the lack of a tip at the end of the journey gives the taxi agent an indication that it did something wrong.
  - *The two points for a win at the end of a chess game tells the agent it did something right.*
  - It is up to the agent to decide which of the actions prior to the reinforcement were most responsible for it.

- **Reinforcement Learning:**
  - In reinforcement learning, the learning of an input–output mapping is performed through continued interaction with the environment



# Introduction to Machine Learning

- **Reinforcement Learning:**
  - It is a Machine Learning algorithm that allows software agents and machines to automatically determine the ideal behaviour within a specific context to maximize its performance.
  - It does not have a labelled dataset or results associated with data so the only way to perform a given task is to learn from experience.
  - For every correct action or decision of an algorithm, it is rewarded with positive reinforcement
  - whereas, for every incorrect action, it is rewarded with negative reinforcement.
  - In this way, it learns which actions are needed to perform and which are not. Reinforcement learning can, therefore, help in industrial automation as well as the gaming sector primarily.

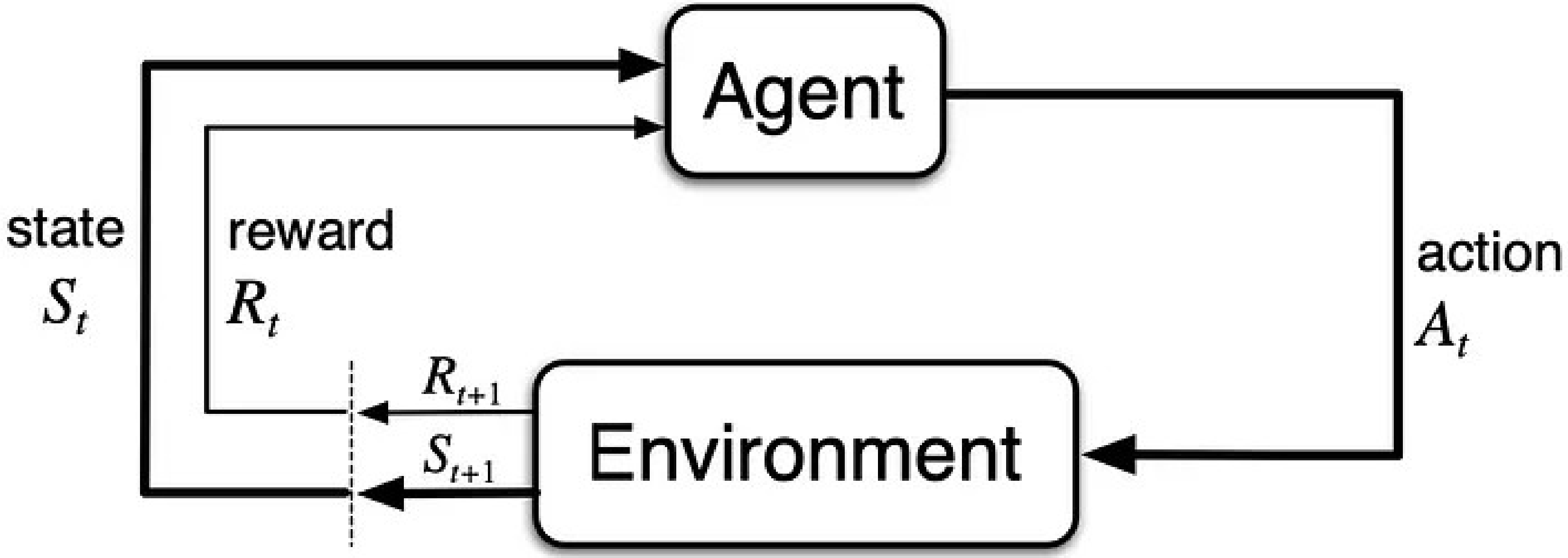
# Introduction to Machine Learning

- **Reinforcement Learning:**
  - A chef explores different ingredients by exploring and experimenting with different recipes in the hope of creating that perfect recipe that wows everyone.
  - This is similar to Reinforcement Learning, where the chef tries a variety of actions like trying different ingredients in different proportions to progressively favors those that appear to taste the best.

# Introduction to Machine Learning

- **Reinforcement Learning:**
  - In Reinforcement learning an agent interacts with the environment by sensing its state and learns to take actions in order to maximize long-term reward.
  - As the agent takes actions it needs to maintain a balance between exploration and exploitation by performing a variety of actions using trial and error to favor the actions that yield the maximum reward in the future.

- **Reinforcement Learning:**
  -



- **Reinforcement Learning:**
  - The goal of RL is for the agent to learn to make sequential decisions in an uncertain environment by mapping the different environment states to actions to maximize long-term rewards.

# Introduction to Machine Learning

- **Difference between Supervised, Unsupervised, and Reinforcement Learning**
  - **Dataset**
    - Supervised learning requires a labeled dataset for training. Unsupervised learning identifies hidden data patterns from an unlabeled dataset, while Reinforcement learning does not require data as it learns by interacting with the environment.
  - **Training**
    - Training of a Supervised algorithm is offline, whereas training of Unsupervised and Reinforcement learning is online and happens in real-time.
  - **Learnings**
    - Supervised learning is instructional-based and needs supervision. Unsupervised learning learns by reasoning. Reinforcement learning learns by experience.

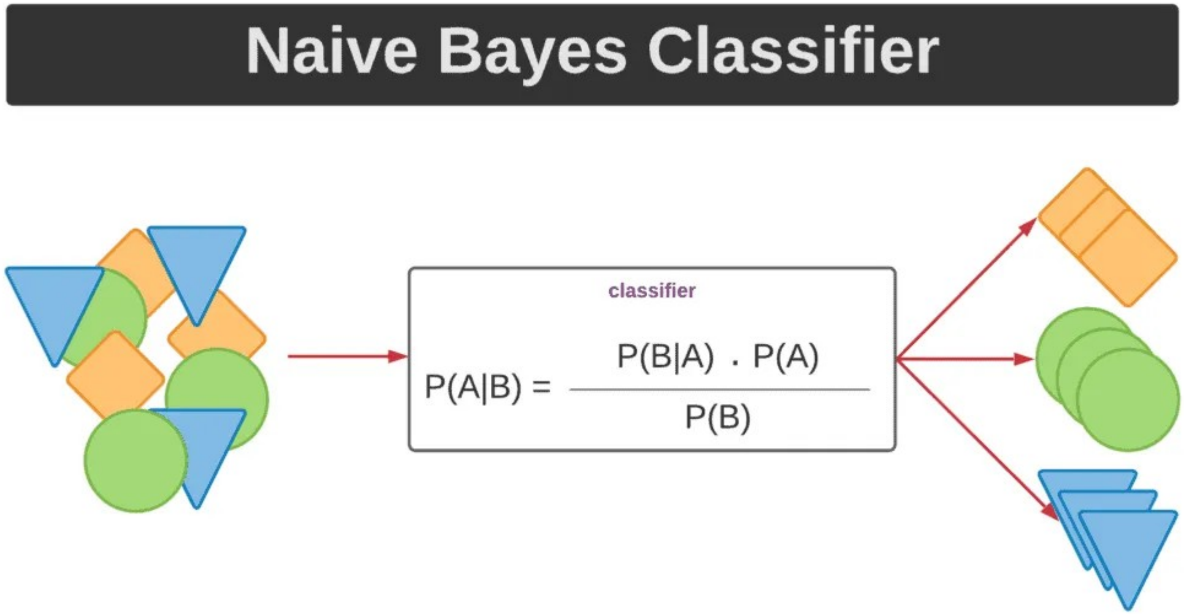
- **Difference between Supervised, Unsupervised, and Reinforcement Learning**
  - **Objective**
    - Supervised algorithms learn only one type of task based on the labeled dataset. The goal is to predict outcomes for new data belonging to the same domain as the same model need not be applied to a different domain. In contrast, the unsupervised algorithm's objective is to gain insights from the unlabeled data that can predict if the new data is part of the cluster or an anomaly. Reinforcement learning is goal-oriented, and the agent aims to learn sequences of actions by exploration and exploitation in an uncertain environment to maximize future rewards. Reinforcement learning can handle an entirely new scenario it has never encountered.

# Introduction to Machine Learning

- **Difference between Supervised, Unsupervised, and Reinforcement Learning**
  - **Algorithms Types and Applications**
    - **S**upervised learning consists of Classification and Regression. Classification algorithms are applied to detect fraud, spam detection, and classify images, and Regression algorithms help predict sales, house prices, etc.
    - Unsupervised learning consists of Clustering, Association, Anomaly detection, and dimensionality reduction. Unsupervised learning applications are customer segmentation, market basket analysis, fraud detection, network security analysis, etc.
    - Reinforcement learning algorithms are either Value-based, Policy-based, or Model-based. Deep Q-Network(DQN), state-Action-Reward-State-Action(SARSA), Asynchronous Advantage Actor-Critic Algorithm(A3C), and Deep Deterministic Policy Gradient(DDPG) are a few Reinforcement algorithms used in Robotics, developing business strategies.

- **Statistical-based Learning: Naive Bayes Model**
  - **Naive** Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
  - It is mainly used in text classification that includes a high-dimensional training dataset.

- **Statistical-based Learning: Naive Bayes Model**



- **Statistical-based Learning: Naive Bayes Model**
  - **Naive** Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
  - It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
  - Some popular examples of **Naive** Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

- **Statistical-based Learning: Naive Bayes Model**
  - The **Naive** Bayes algorithm is comprised of two words **Naive** and Bayes, Which can be described as:
    - **Naive**: It is called **Naive** because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
    - Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem

- **Statistical-based Learning: Naive Bayes Model**
  - Bayes' Theorem:
    - Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
  - The formula for Bayes' theorem is given as:

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

- **Statistical-based Learning: Naive Bayes Model**
  - Bayes' Theorem:
    - Where,
      - P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
      - P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
      - P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
      - P(B) is Marginal Probability: Probability of Evidence.

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

# Introduction to Machine Learning

- **Statistical-based Learning: Naive Bayes Model**
  - Working of **Naive** Bayes' Classifier can be understood with the help of the below example:
    - **Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:**
      - Convert the given dataset into frequency tables.
      - Generate Likelihood table by finding the probabilities of given features.
      - Now, use Bayes theorem to calculate the posterior probability.

- **Statistical-based Learning: Naive Bayes Model**
  - Problem: If the weather is sunny, then the Player should play or not?

- **Statistical-based Learning: Naive Bayes Model**
  - Problem: If the weather is sunny, then the Player should play or not?
    - Solution: To solve this, first consider the below dataset:

# Introduction to Machine Learning

| | Outlook | Play |
|---|---|---|
| 0 | Rainy | Yes |
| 1 | Sunny | Yes |
| 2 | Overcast | Yes |
| 3 | Overcast | Yes |
| 4 | Sunny | No |
| 5 | Rainy | Yes |
| 6 | Sunny | Yes |
| 7 | Overcast | Yes |
| 8 | Rainy | No |
| 9 | Sunny | No |
| 10 | Sunny | Yes |
| 11 | Rainy | No |
| 12 | Overcast | Yes |
| 13 | Overcast | Yes |

**Frequency table for the Weather Conditions:**

| Weather | Yes | No |
|---|---|---|
| Overcast | 5 | 0 |
| Rainy | 2 | 2 |
| Sunny | 3 | 2 |
| Total | 10 | 5 |

**Likelihood table weather condition:**

| Weather | No | Yes | |
|---|---|---|---|
| Overcast | 0 | 5 | 5/14= 0.35 |
| Rainy | 2 | 2 | 4/14=0.29 |
| Sunny | 2 | 3 | 5/14=0.35 |
| All | 4/14=0.29 | 10/14=0.71 | |

- **Applying Bayes theorem:**
- P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)
  - P(Sunny|Yes)= 3/10= 0.3
  - P(Sunny)= 0.35
  - P(Yes)=0.71
- So P(Yes|Sunny) = 0.3*0.71/0.35= 0.60

- **Applying Bayes theorem:**
- P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)
  - P(Sunny|NO)= 2/4=0.5
  - P(No)= 0.29
  - P(Sunny)= 0.35
- So P(No|Sunny)= 0.5*0.29/0.35 = 0.41
- As we can see from the above calculation that:
  - P(Yes|Sunny)>P(No|Sunny)
- Hence, on a Sunny day, Player can play the game.

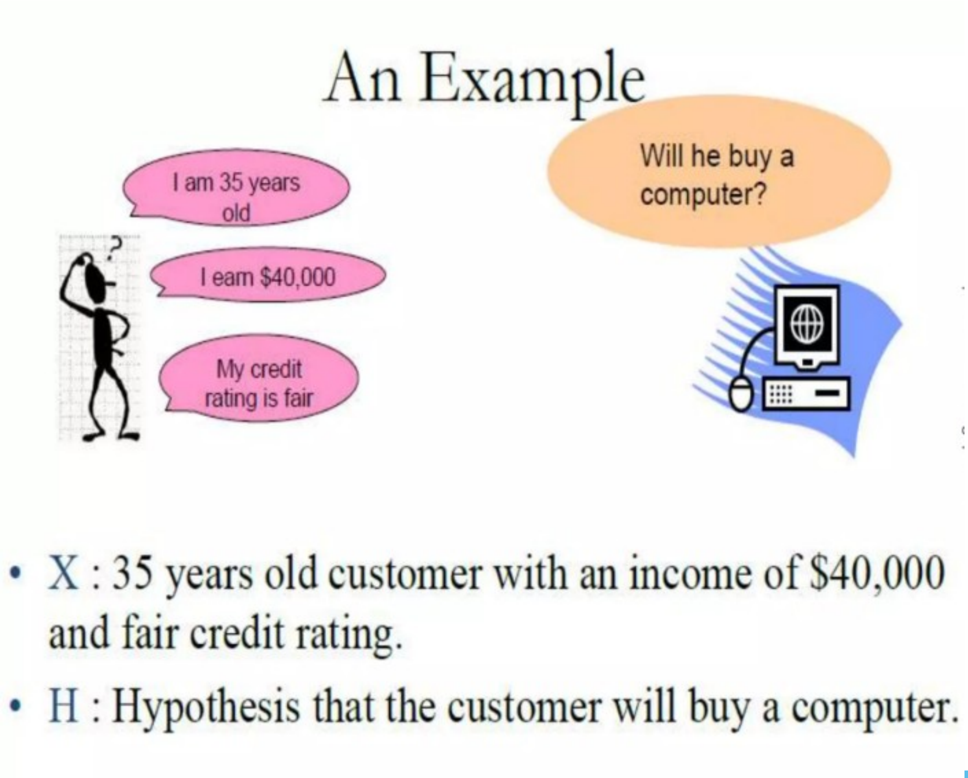
- **Statistical-based Learning: Naive Bayes Model**
  - Advantages of **Naive** Bayes Classifier:
    - **Naive** Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
    - It can be used for Binary as well as Multi-class Classifications.
    - It performs well in Multi-class predictions as compared to the other Algorithms.
    - It is the most popular choice for text classification problems.
  - Disadvantages of **Naive** Bayes Classifier:
    - Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

- **Statistical-based Learning: Naive Bayes Model**
  - Applications of **Naive** Bayes Classifier:
    - It is used for Credit Scoring.
    - It is used in medical data classification.
    - It can be used in real-time predictions because **Naive** Bayes Classifier is an eager learner.
    - It is used in Text classification such as Spam filtering and Sentiment analysis.

- **Example 1: Statistical-based Learning: Naive Bayes Model**
  - **Data**

| ID | age | income | student | credit-rating | Class: buys-computer |
|---|---|---|---|---|---|
| 1 | ≤30 | high | no | fair | no |
| 2 | ≤30 | high | no | excellent | no |
| 3 | 31…40 | high | no | fair | yes |
| 4 | >40 | medium | no | fair | yes |
| 5 | >40 | low | yes | fair | yes |
| 6 | >40 | low | yes | excellent | no |
| 7 | 31…40 | low | yes | excellent | yes |
| 8 | ≤30 | medium | no | fair | no |
| 9 | ≤30 | low | yes | fair | yes |
| 10 | >40 | medium | yes | fair | yes |
| 11 | ≤30 | medium | yes | excellent | yes |
| 12 | 31…40 | medium | no | excellent | yes |
| 13 | 31…40 | high | yes | fair | yes |
| 14 | >40 | medium | no | excellent | no |

- **Example: Statistical-based Learning: Naive Bayes Model**



- **Example: Statistical-based Learning: Naive Bayes Model**
  - $P(C_i)$: P(buys_computer = “yes”) = 9/14 = 0.643
  - P(buys_computer = “no”) = 5/14= 0.357
  - Compute $P(X|C_i)$ for each class
    - P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.222
    - P(age = “<= 30” | buys_computer = “no”) = 3/5 = 0.6
    - P(income = “medium” | buys_computer = “yes”) = 4/9 = 0.444
    - P(income = “medium” | buys_computer = “no”) = 2/5 = 0.4
    - P(student = “yes” | buys_computer = “yes) = 6/9 = 0.667
    - P(student = “yes” | buys_computer = “no”) = 1/5 = 0.2
    - P(credit_rating = “fair” | buys_computer = “yes”) = 6/9 = 0.667
    - P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.4
  - **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

- **Example: Statistical-based Learning: Naive Bayes Model**

  - **$P(X|C_i)$ :** P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
  - P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019
  -
  - **$P(X|C_i)*P(C_i)$ :** P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
  - P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007
  - **Therefore, X belongs to class ("buys_computer = yes")**

- **Example 2: Statistical-based Learning: Naive Bayes Model**

The following table presents a dataset of 10 objects, with attributes Color, Type, Origin, and the "class", whether the customer who bought was satisfied or not:

| Sr. No. | Color | Type | Origin | Satisfied? |
|---|---|---|---|---|
| 1 | Red | Casual | Domestic | Yes |
| 2 | Red | Casual | Domestic | No |
| 3 | Red | Casual | Domestic | Yes |
| 4 | Yellow | Casual | Domestic | No |
| 5 | Yellow | Casual | Imported | Yes |
| 5 | Yellow | Casual | Imported | Yes |
| 6 | Yellow | Formal | Imported | No |
| 7 | Yellow | Formal | Imported | Yes |
| 8 | Yellow | Formal | Domestic | No |
| 9 | Red | Formal | Imported | No |
| 10 | Red | Casual | Imported | Yes |

We want to classify a new object with the following properties: Color = Red, Origin = Domestic and Type = Formal. Use Naïve Bayes algorithm and tell the class of this test instance.

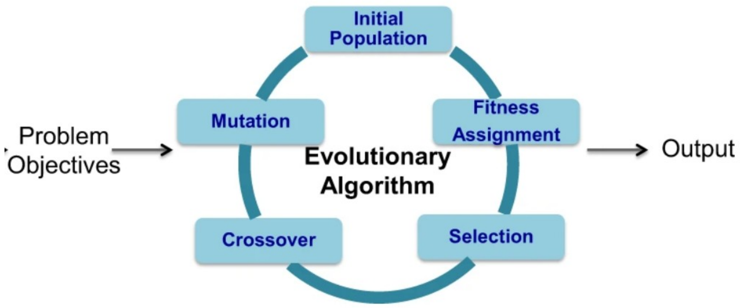
- Genetic Algorithm

✓ A genetic algorithm is used to **solve complicated problems with a greater number of variables & possible outcomes/solutions.**

✓ The combinations of different solutions are passed through the Darwinian based algorithm to find the best solutions.

✓ The poorer solutions are then replaced with the offspring of good solutions.

**https://www.upgrad.com/blog/genetic-algorithm-in-ai**



- Genetic Algorithm
  - The genetic algorithm works on the evolutionary generational cycle to generate high-quality solutions.
  - *These algorithms use different operations that either enhance or replace the population to give an improved fit solution.*

✓ In GAs, **we have a pool or a population of possible solutions** to the given problem.

✓ These solutions then undergo recombination and mutation (like in natural genetics), producing new children, and the process is repeated over various generations.

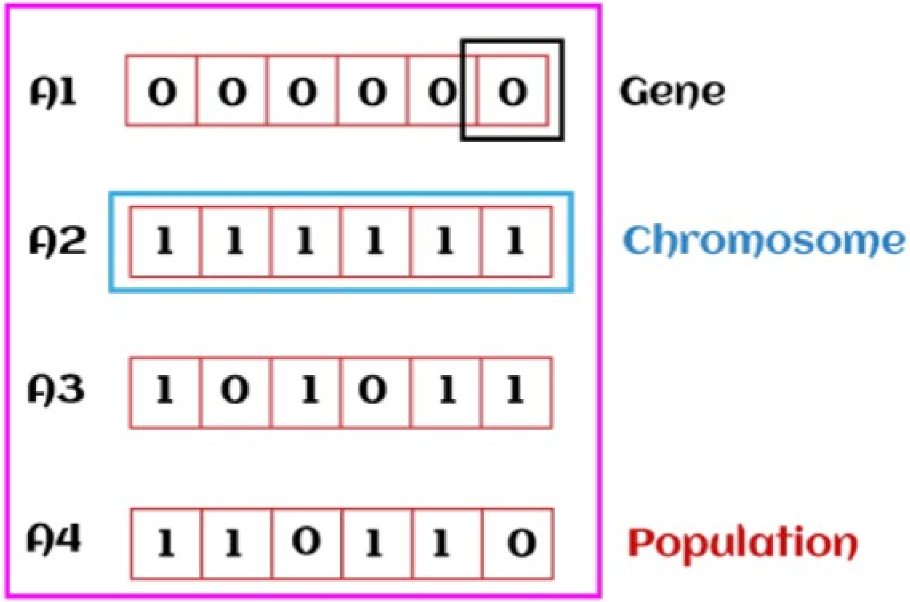
✓ Each individual (or candidate solution) is assigned a fitness value and the fitter individuals are given a higher chance to reproduce and yield more "fitter" individuals.

- Genetic Algorithm

> ✓ In GAs, **we have a pool or a population of possible solutions** to the given problem.
>
> ✓ These solutions then undergo recombination and mutation (like in natural genetics), producing new children, and the process is repeated over various generations.
>
> ✓ Each individual (or candidate solution) is assigned a fitness value and the fitter individuals are given a higher chance to reproduce and yield more "fitter" individuals.

- This is in line with the Darwinian Theory of "Survival of the Fittest".
- In this way we keep "evolving" better individuals or solutions over generations, till we reach a stopping criterion.

- Genetic Algorithm: **Terminologies**
- **Population:** Population is the subset of all possible or probable solutions, which can solve the given problem.
- **Chromosomes:** A chromosome is one of the solutions in the population for the given problem, and the collection of gene generate a chromosome.
- **Gene:** A chromosome is divided into a different gene, or it is an element of the chromosome.



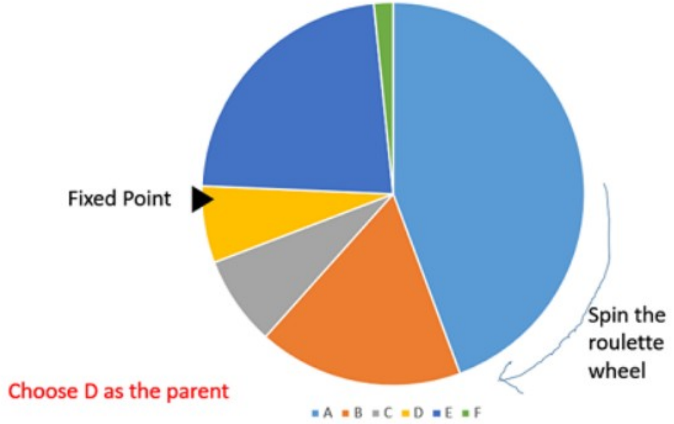
- **Genetic Algorithm:** Terminologies
- **Fitness Function:** A fitness function simply defined is a function which takes the solution as input and produces the suitability of the solution as the output.
- **Genetic Operators:** They alter the genetic composition of the offspring. These include ***selection***, ***crossover***, and ***mutation***.

- **Genetic Algorithm: Selection Operator**
- This operator in **genetic algorithms in AI** is responsible for *selecting the individuals with better fitness scores for reproduction.*
- The idea is to give preference to the individuals with good fitness scores and allow them to pass their genes to successive generations.
- There are three types of Selection methods available, which are:
- - Roulette wheel selection
- - Stochastic universal sampling
- - Tournament selection
- - Rank-based selection

- **Genetic Algorithm: Selection Operator**
- Roulette Wheel Selection
- In a roulette wheel selection, the circular wheel is divided into slots of region.
- A fixed point is chosen on the wheel circumference as shown and the wheel is rotated.
- *The region of the wheel which comes in front of the fixed point is chosen as the parent.*
- For the second parent, the same process is repeated. The probability for selection of an individual is proportional to its fitness.

- **Genetic Algorithm: Selection Operator**
- <u>Roulette Wheel Selection</u>

- Here the sum of fitness is = 18.5
- Here the percentage of area in the wheel for the chromosome A = 8.2/18.5=0.44=44%.
- Similarly for D=0.648=6.5%



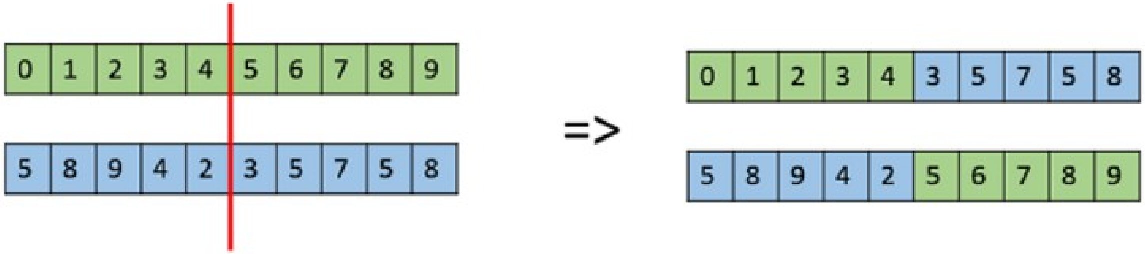

| Chromosome | Fitness Value |
|---|---|
| A | 8.2 |
| B | 3.2 |
| C | 1.4 |
| D | 1.2 |
| E | 4.2 |
| F | 0.3 |

If $f_i$ is the fitness of individual $I$ in the population, its probability of being selected is

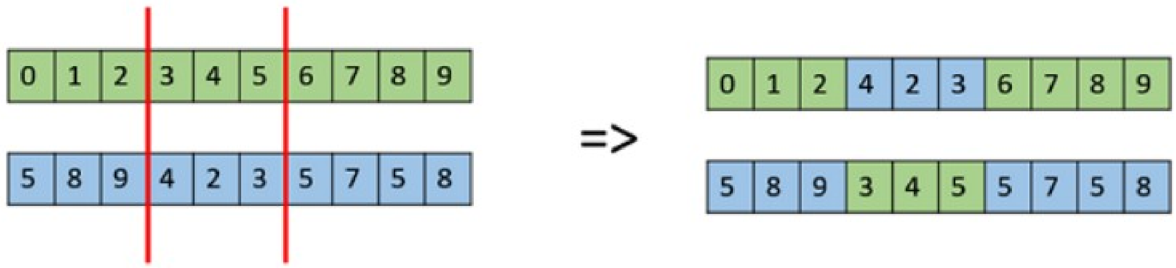
$$p_i = \frac{f_i}{\Sigma_{j=1}^{N} f_j},$$

- **Genetic Algorithm: Crossover Operator**
  - The crossover operator *is analogous to biological crossover.*
  - In this more than one parent is selected and one or more off-springs are produced using the genetic material of the parents.

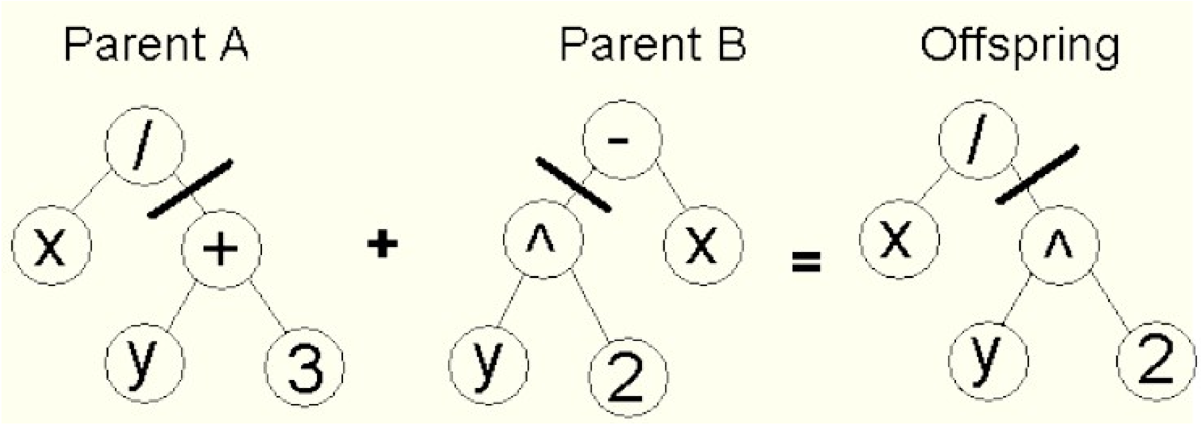
  - One Point Crossover
  - In this one-point crossover, a random crossover point is selected and the tails of its two parents are swapped to get new off-springs.



- **Genetic Algorithm: Crossover Operator**
  - ii. Multipoint Crossover
  - Multi point crossover is a generalization of the one-point crossover wherein alternating segments are swapped to get new off-springs.



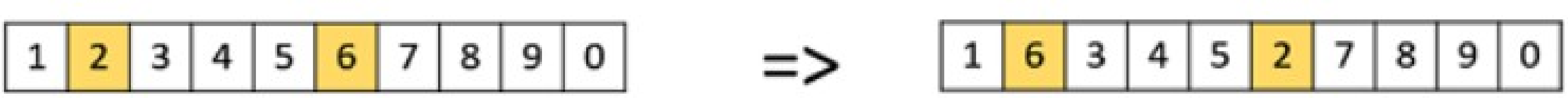
- **Genetic Algorithm: Crossover Operator**
  - Tree crossover
  - One crossover point is selected in both parents, parents are divided in that point and the parts below crossover points are exchanged to produce new offspring.



- **Genetic Algorithm: Mutation Operator**
  - Mutation may be defined as a *small random tweak in the chromosome, to get a new solution.*
  - It is *used to maintain and introduce diversity in the genetic population.*
  - The mutation operator inserts random genes in the offspring (new child) to maintain the diversity in the population.
  - It can be done by flipping some bits in the chromosomes.
    - Mutation operators can be
    - Bit flip mutation
    - Random resetting
    - Swap mutation and so on.

- **Genetic Algorithm: Mutation Operator**
  - Bit Flip Mutation
    - In this bit flip mutation, we select one or more random bits and flip them. This is used for binary encoded GAs.


  - Swap Mutation
    - In swap mutation, we select two positions on the chromosome at random, and interchange the values. This is common in permutation based encodings.



# Parameters of GA

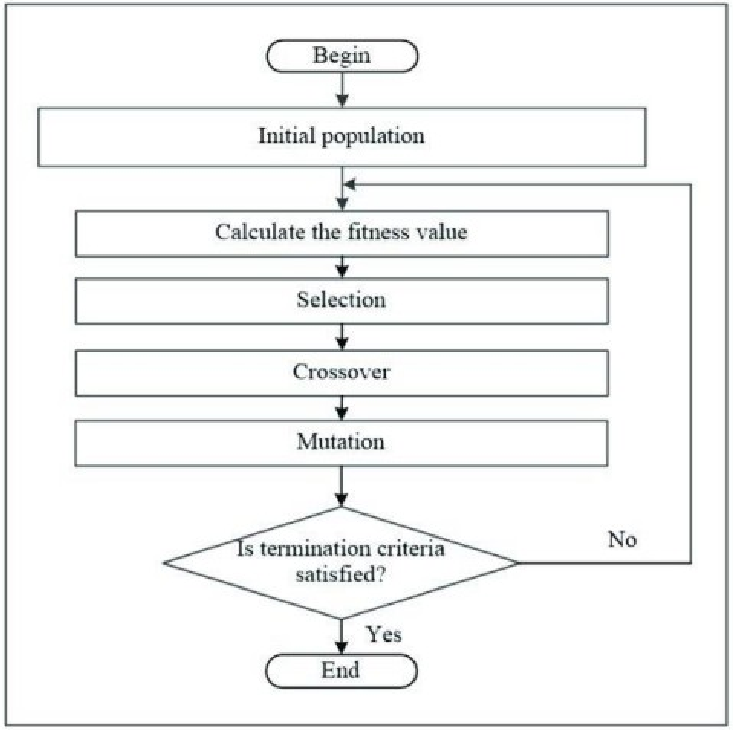
- **Crossover probability:**
- How often crossover will be performed.
- *If there is no crossover, offspring are exact copies of parents.*
- If there is crossover, offspring are made from parts of both parent's chromosome.
- If crossover probability is 100%, then all offspring are made by crossover.
- If it is 0%, whole new generation is made from exact copies of chromosomes from old population.

- **Mutation probability:**
- How often parts of chromosome will be mutated.
- *If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change.*
- If mutation is performed, one or more parts of a chromosome are changed.
- If mutation probability is 100%, whole chromosome is changed, if it is 0%, nothing is changed.

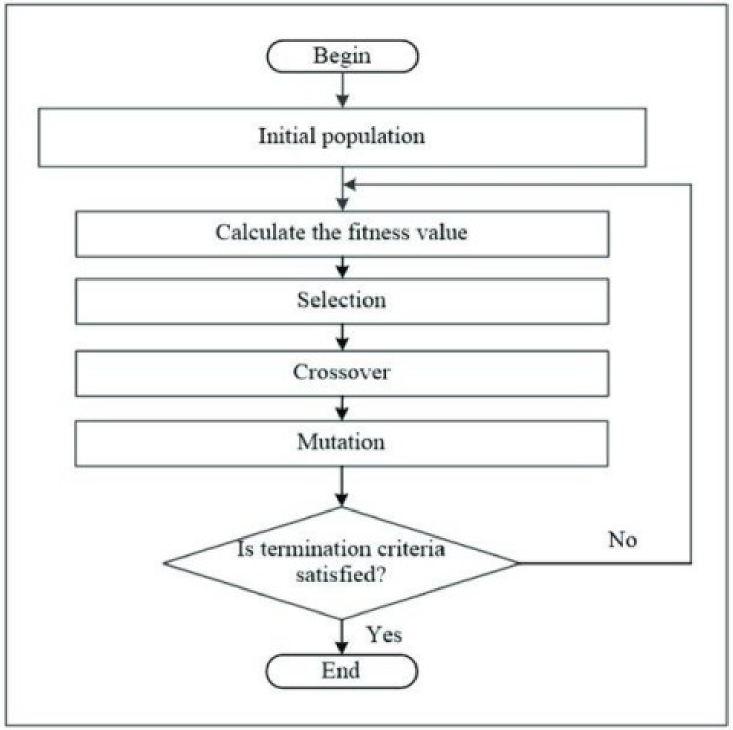
# GA Algorithm

1. Generate random population of n individuals.
2. Evaluate the fitness of each individual.
3. Create a new population.
   a. Select two parents from a population according to their fitness.
   b. Crossover operation performed between the two parents
   c. Mutation operation is performed this will change the trip from the output of crossover step, if not the children will be the same
   d. Add new children in a new population.
4. Evaluate the fitness of each individual.
5. If the stopping criteria is satisfied, the algorithm stops and shows the best trip, if not it will start over from step 3 and continues the iteration.

# GA Algorithm

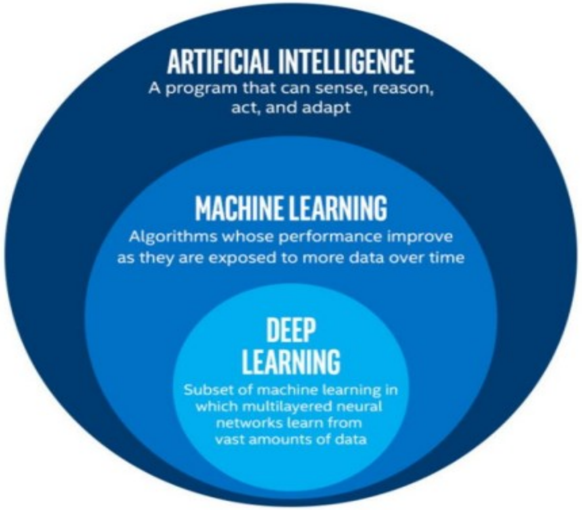


# GA Algorithm

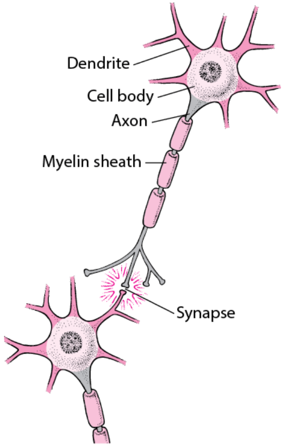


# Deep Learning

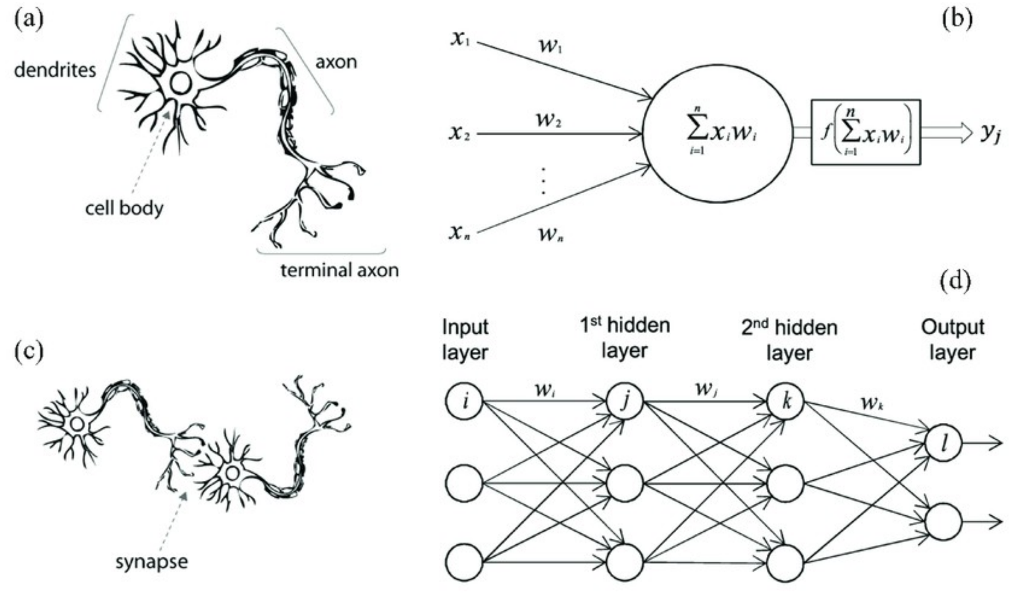
- Deep learning is an aspect of artificial intelligence (AI) that is to simulate the activity of the human brain specifically, pattern recognition by passing input through various layers of the neural network.



- Neural Networks
  - *A neuron is a cell in brain whose principle function is the collection, Processing, and dissemination of electrical signals.*
  - Brains Information processing capacity comes from networks of such neurons.
  - Due to this reason some earliest AI work aimed to create such artificial networks. (Other Names are Connectionism; Parallel distributed processing and neural computing).



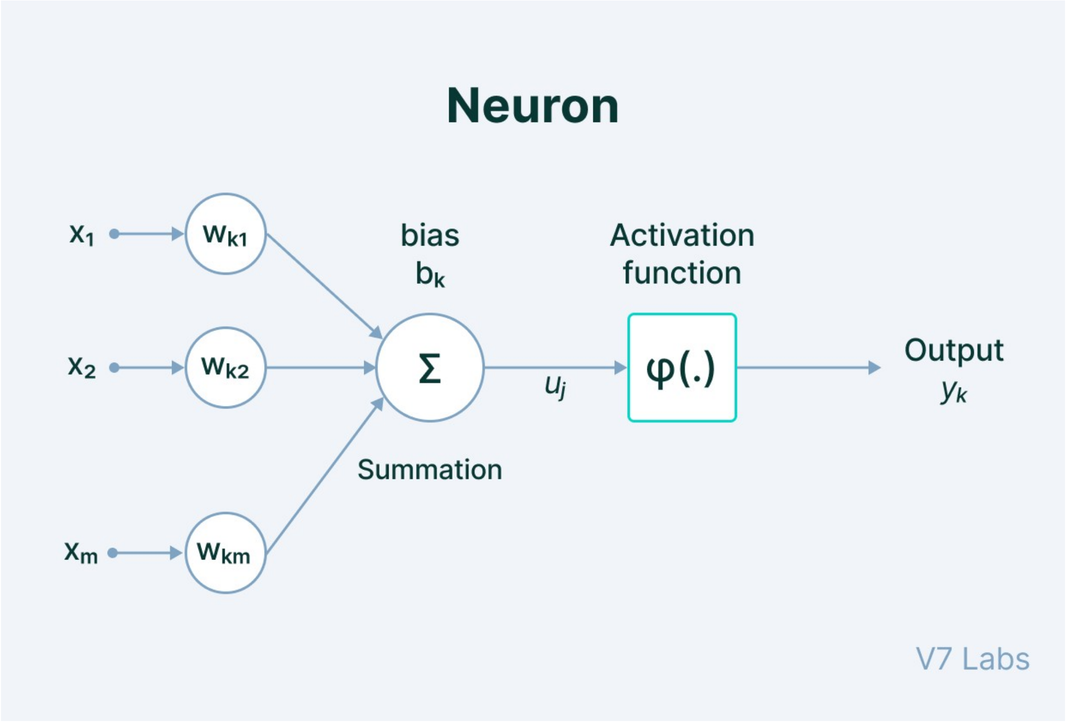
- Neural Networks
  - An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information.
  - *It is composed of a large number of highly interconnected processing elements (neurons) working in unison 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.
  -



- Neural Networks
  - Why use neural network?
    - Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
    - Self-Organisation: An ANN can create its own organization or representation of the information it receives during learning time.
    - Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
    - Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage

Units of Neural Network

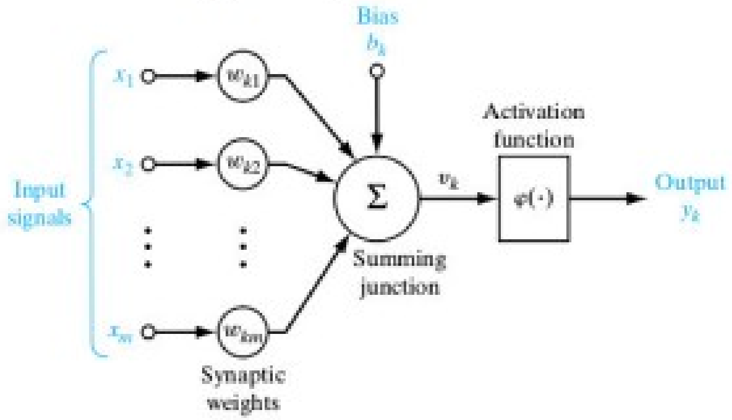
- Nodes (units): Nodes represent a cell of neural network.
- Links: Links are directed arrows that show propagation of information from one node to another node.
- Activation: Activations are inputs to or outputs from a unit
- Weight: Each link has weight associated with it which determines strength and sign of the connection.
- Activation function: A function which is used to derive output activation from the input activations to a given node is called activation function.
- Bias Weight: Bias weight is used to set the threshold for a unit. Unit is activated when the weighted sum of real inputs exceeds the bias weight.



- Neural Networks
  - **Mathematical Model**
  - A simple mathematical model of neuron is devised by McCulloch and Pit is given in the figure given below:

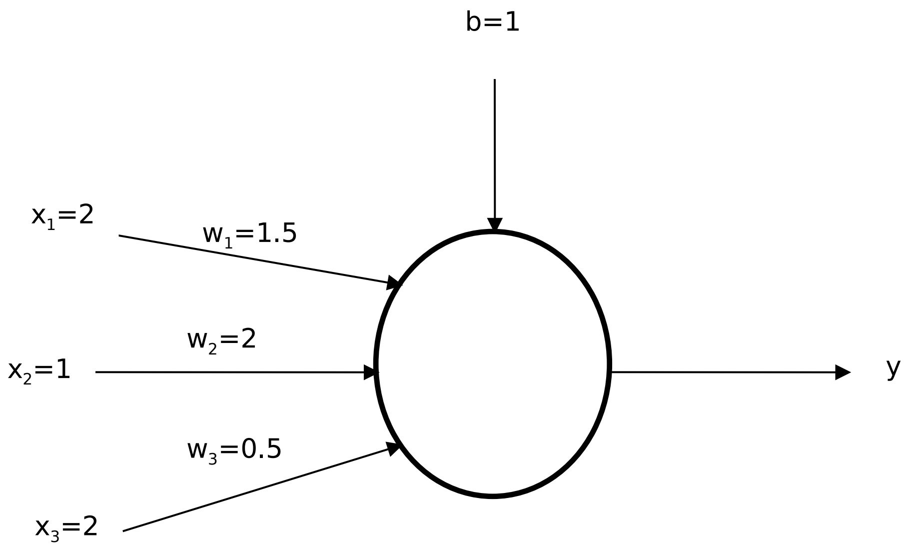
$$u_k = \sum_{j=1}^{n} x_j * w_{kj} \quad v_k = u_k + b_k$$

$$y_k = \varphi(u_k + b_k) = \varphi(v_k)$$

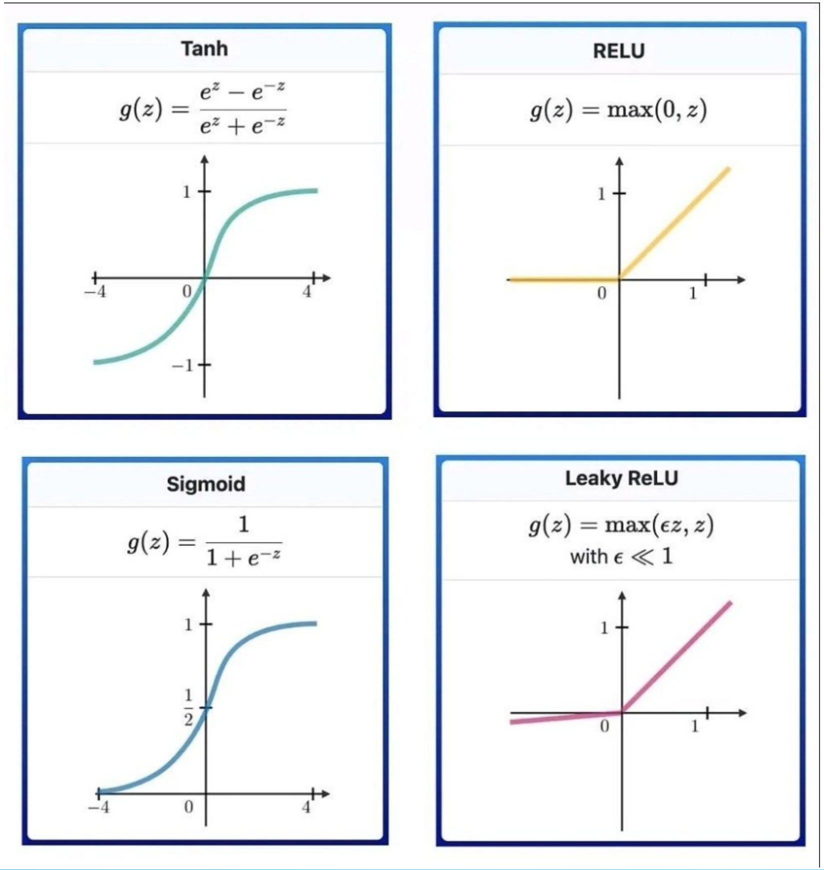



  - *where, $x_1$, $x_2$,….,$x_n$ are input signals, $w_{k1}$,$w_{k2}$,….,$w_{kn}$ are weights, $u_k$ is weighted sum of inputs, φ is activation function, and $y_k$ is output signal*

- Neural Networks
  - **Example:** *Consider following neuron and compute its output by assume activation function F(x)=1 if x>5 and F(x)=0, otherwise*
  - $u = x_1 * w_1 + x_2 * w_2 + x_3 * w_3$
  - $= 2*1.5+1*2+2*0.5=6$
  - $v=u+b=6+1=7$
  - Now,
  - $y=f(v)=1$

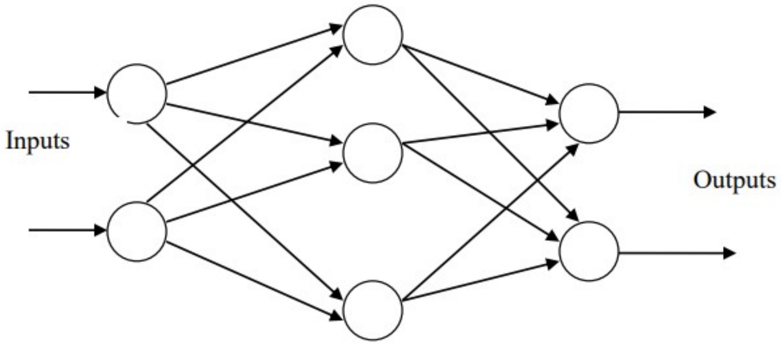


- **<u>Other activation function</u>**
  - Sigmoid Function
  - Tanh function
  - Relu function
  - Softmax function

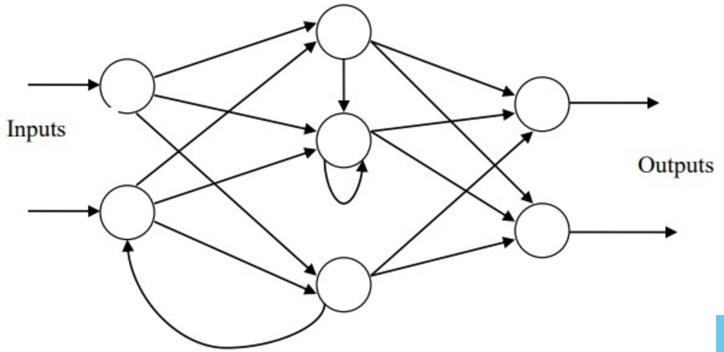


## Structure of Neural Networks

- **Feed-forward networks**
- Feed-forward ANNs allow signals to travel one way only; from input to output.
- There is no feedback (loops) i.e. the output of any layer does not affect that same layer.

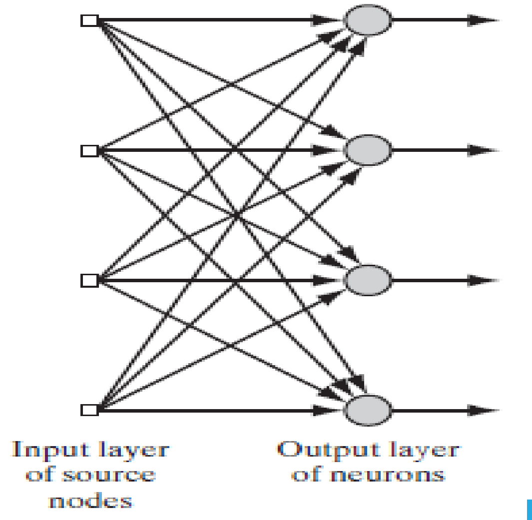


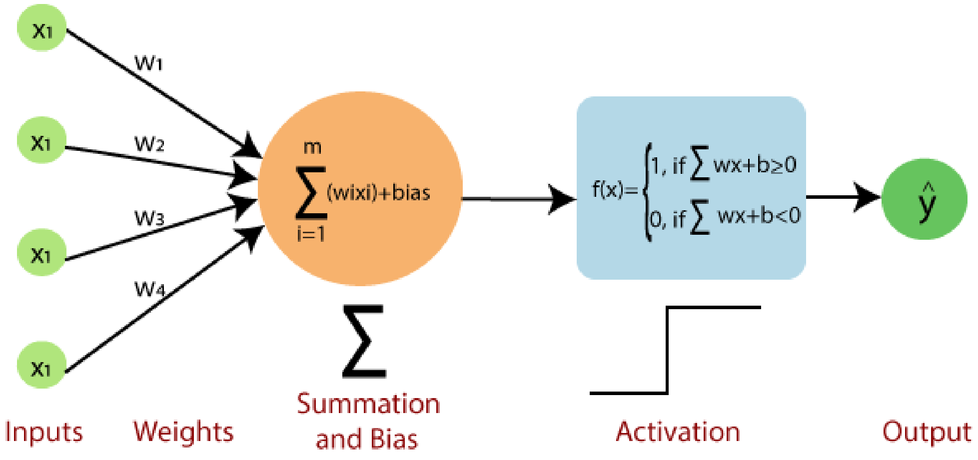

- **Feedback networks (Recurrent networks)**
- Feedback networks can have signals traveling in both directions by introducing loops in the network.
- Feedback networks are very powerful and can get extremely complicated.
- Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point.



Structure of Neural Networks

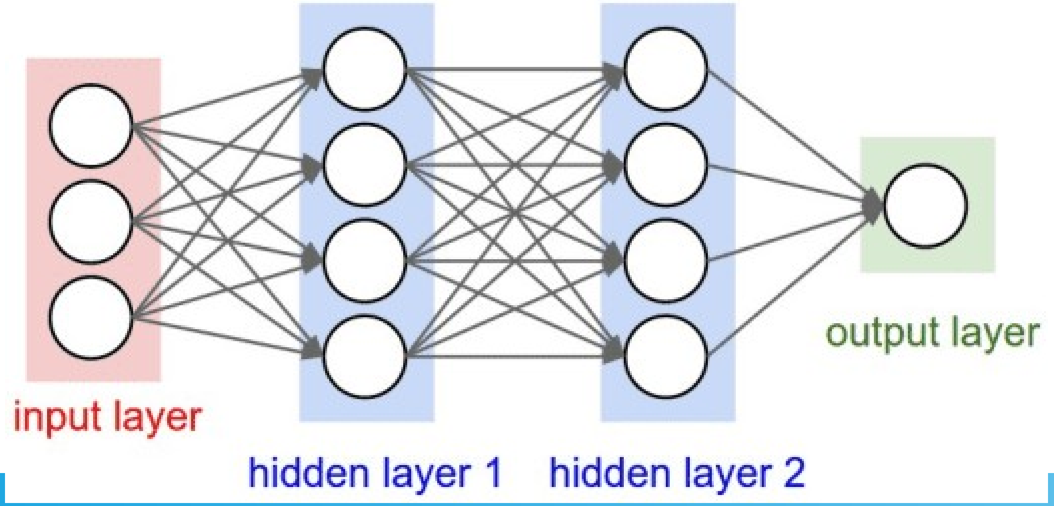
- Single Layer Networks
- It is the simplest form of a network architecture.
- In this architecture we have an *input layer* of source nodes that are connected directly with an *output layer* of neurons



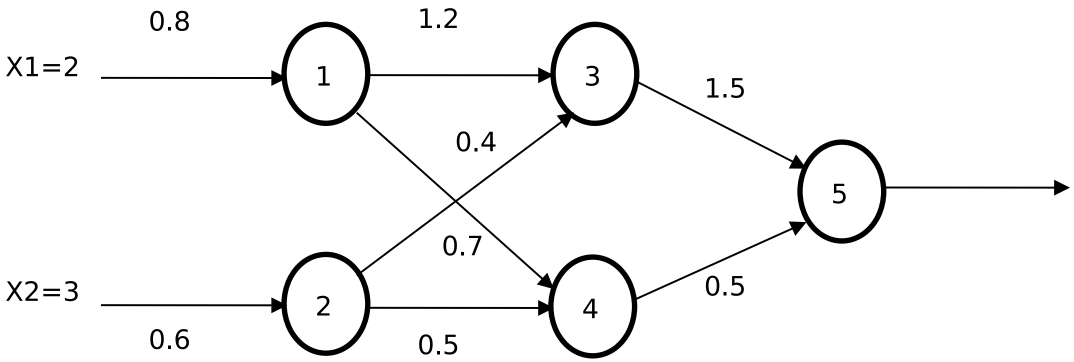
Structure of Neural Networks

- Multi Layer Neural Network
- In this type of network architecture, one or more hidden layers are present between input and output layers.
- These layers are not directly visible and information only flows in the direction of input to output layer.



Neural Network Architectures

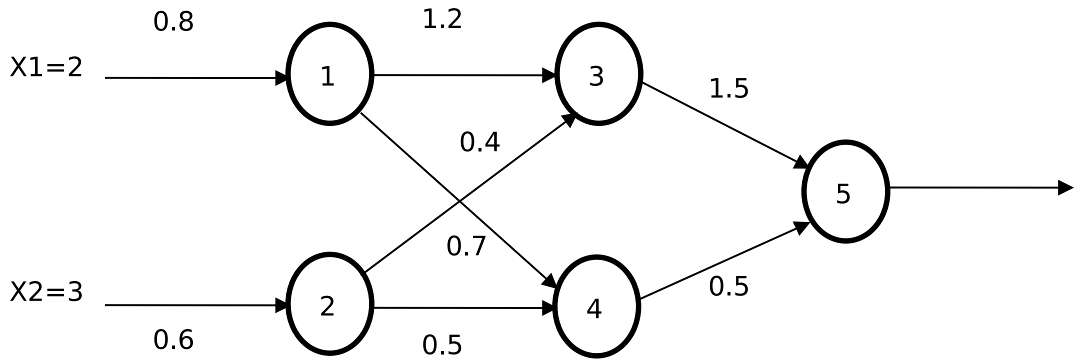
- **Example:** Consider following Neural Network and compute its output using activation function F(x)=2x-1. Weights of synaptic links are provided above each link.




- ***For Node 1***
  - ***u1=2*0.8+3*1=4.6 => y1=f(u1)=2*4.6-1=8.2***
- ***For Node 2***
  - ***u2=2*0.4+3*0.6=2.6 => y2=f(u2)=4.2***

Neural Network Architectures

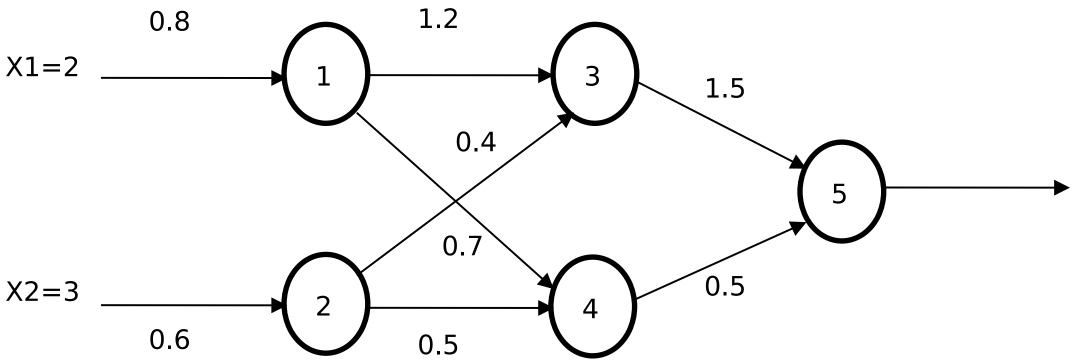
- **Example:** Consider following Neural Network and compute its output using activation function F(x)=2x-1. Weights of synaptic links are provided above each link.




- ***For Node 3***
  - ***u3=8.2*1.2+4.2*0.4=11.51 =>y3=f(u3)=22.04***
- ***For Node 4***
  - ***u4=8.2*0.7+4.2*0.5=7.84 =>y4=f(u4)=14.68***

Neural Network Architectures

- **Example:** Consider following Neural Network and compute its output using activation function F(x)=2x-1. Weights of synaptic links are provided above each link.

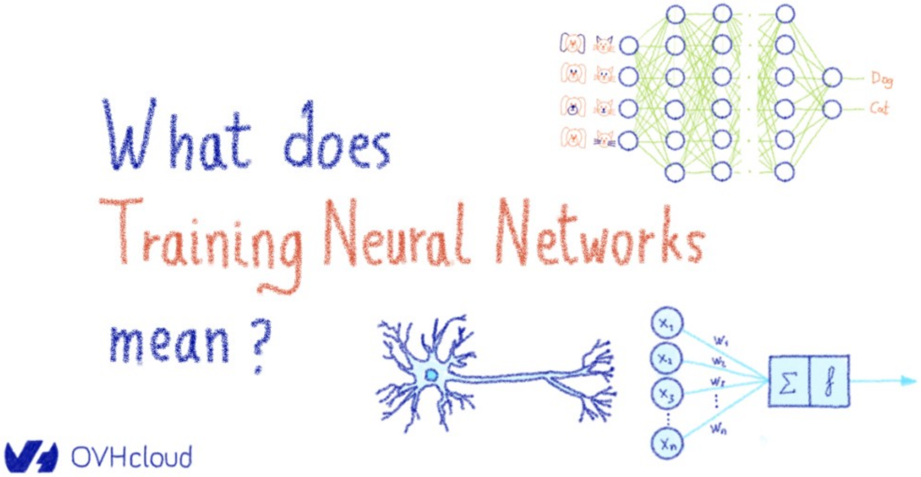



- ***For Node 5***
  - ***u5=22.04*1.5+14.68*0.5=40.4 => y5=f(u5)=79.8***
- **Thus,**
- **Final output of the neural network (y)=79.8**

# Introduction to Machine Learning

## Learning in Neural Networks:

- One of the powerful features of neural networks is learning.
- *Learning in neural networks is carried out by adjusting the connection weights among neurons.*
- It is similar to a biological nervous system in which learning is carried out by changing synapses connection strengths, among cells.
- There is no algorithm that determines how the weights should be assigned in order to solve specific problems.
- Hence, the weights are determined by a learning process



Structure of Neural Networks

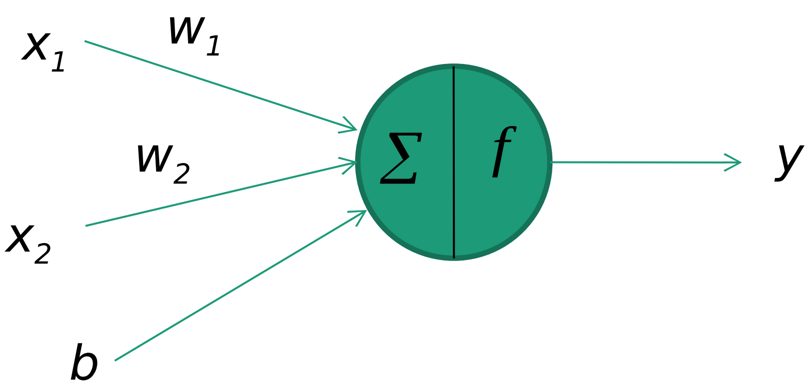
- **Hebbian Learning**
- Hebbian Learning Rule was proposed by Donald Hebb in 1949.
- The rule is based on the assumption that if two neighbor neurons activated and deactivated at the same time. Then the weight connecting these neurons should increase.

**Hebbian Learning Algorithm**

1. Initialize all weights and bias to zero
2. For each training vector s and target t perform steps *3 to 6*
3. Set $x_i = s_i \quad for\ i = 1\ to\ n$
4. Set *y=t*
5. Adapt weights as: $w_i = w_i + \alpha\, x_i\, y \quad for\ i = 1\ to\ n$
6. Adapt bias as: $b = b + \alpha\, y$
7. Test for stopping criteria

Structure of Neural Networks

- **Hebbian Learning**
  - ***Example***
  - Train the following NN by using given training set



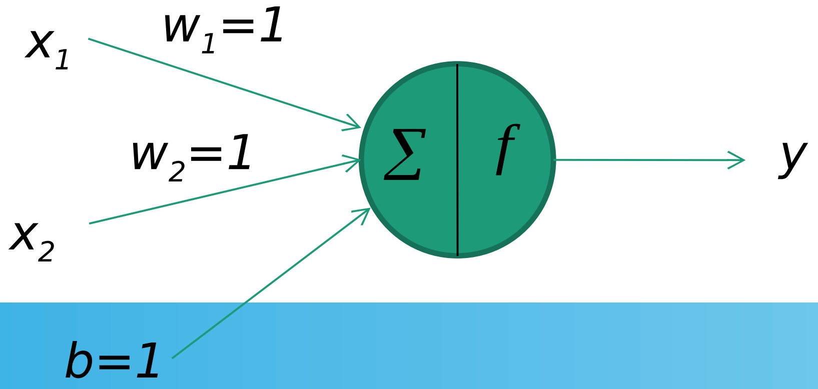

| $x_1$ | $x_2$ | $t$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

Structure of Neural Networks

- **Hebbian Learning**

| *Input* | $w_1(old)$ | $w_2(old)$ | $b(old)$ | $w_1(new)$ | $w_2(new)$ | $b(new)$ |
|---|---|---|---|---|---|---|
| (1,1,1) | 0 | 0 | 0 | 0+1*1*1=1 | 0+1*1*1=1 | 0+1*1=1 |
| (1,0,0) | 1 | 1 | 1 | 1+1*1*0=1 | 1+1*0*0=1 | 1+1*0=1 |
| (0,1,0) | 1 | 1 | 1 | 1+1*0*0=1 | 1+1*1*0=1 | 1+1*0=1 |
| (0,0,0) | 1 | 1 | 1 | 1+1*0*0=1 | 1+1*0*0=1 | 1+1*0=1 |

Thus, Final Neuron is



## Perceptron Learning

- The term "Perceptron" was coined by Frank RosenBlatt in 1962 and is used to describe the connection of simple neurons into networks.
- The perceptron is the simplest form of a neural network used for the classifying linearly separable patterns. Patterns that can be separated by a hyperplane are called linearly separable patterns.
- Basically, perceptron consists of a single neuron with adjustable synaptic weights and bias

Perceptron Learning

**Perceptron Learning Algorithm**

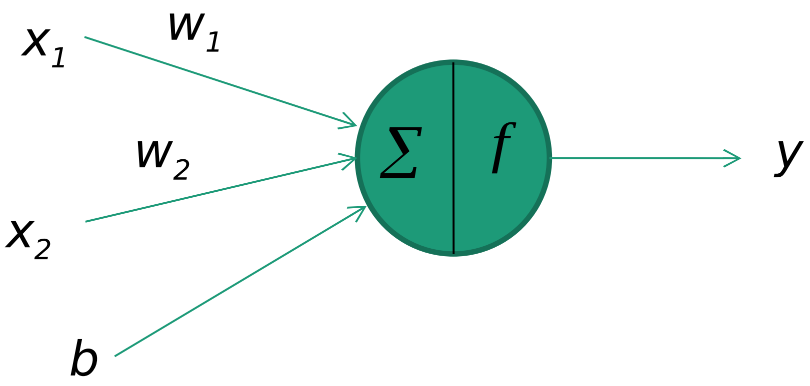
1. Initialize all weights and bias to zero
2. For each training vector s and target t perform steps *3 to 6*
3. Set $x_i = s_i \quad for\ i = 1\ to\ n$
4. Compute output using Hard limiter activation function as below

$$y_{in} = b + \sum_{i=1}^{n} w_i x_i \qquad y = f(y_{in})$$

5. Adapt weights as: $w_i = w_i + \alpha(t - y)x_i \quad for\ i = 1\ to\ n$
6. Adapt bias as: $b = b + \alpha(t - y)$
7. Test for Stopping Criteria

Perceptron Learning

- ***Perceptron Training: Example***
  - Train the following perceptron by using given training set




| $x_1$ | $x_2$ | $t$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | -1 | -1 |
| -1 | 1 | -1 |
| -1 | -1 | -1 |

Perceptron Learning

- ***Perceptron Training: Example***
  - First Epoch

| *Input* | $w_1$ *(old)* | $w_2$ *(old)* | *b (old)* | *v* | *y* | $w_1(new)$ | $w_2(new)$ | *b(new)* |
|---|---|---|---|---|---|---|---|---|
| (1,1,1) | 0 | 0 | 0 | 0 | 0 | 0+1*1*1=1 | 0+1*1*1=1 | 0+1*1=1 |
| (1,-1,-1) | 1 | 1 | 1 | 1 | 1 | 1+1*-2*1=-1 | 1+1*-2*-1=3 | 1+1*-2=-1 |
| (-1,1,-1) | -1 | 3 | -1 | 3 | 1 | -1+1*-2*-1=1 | 3+1*-2*1=1 | -1+1*-2=-3 |
| (-1,-1,-1) | 1 | 1 | -3 | -5 | -1 | 1+1*0*-1=1 | 1+1*0*-1=1 | -3+1*0=-3 |

$$\varphi(x) = \begin{cases} 1 & if \quad x > 0 \\ -1 & if \quad x < 0 \\ 0 & if \quad x = 0 \end{cases}$$

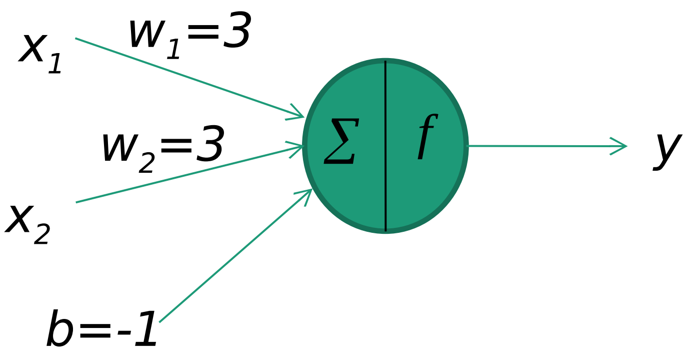
*Assimed hard Limiter Activation function*

Perceptron Learning

- ***Perceptron Training:* *Example***
  - Second Epoch

| Input | $w_1$ (old) | $w_2$ (old) | $b$ (old) | $v$ | $y$ | $w_1(new)$ | $w_2(new)$ | $b(new)$ |
|---|---|---|---|---|---|---|---|---|
| (1,1,1) | 1 | 1 | -3 | -1 | -1 | 1+1*2*1=3 | 1+1*2*1=3 | -3+1*2=-1 |
| (1,-1,-1) | 3 | 3 | -1 | -1 | -1 | 3+1*0*1=3 | 3+1*0*-1=3 | -1+1*0=-1 |
| (-1,1,-1) | 3 | 3 | -1 | -1 | -1 | 2+1*0*-1=3 | 3+1*0*1=3 | -1+1*0=-1 |
| (-1,-1,-1) | 3 | 3 | -1 | -7 | -1 | 3+1*0*-1=3 | 3+1*0*-1=3 | -1+1*0=-1 |

Thus, Final Neuron is



## Backpropagation

- A popular method for the training of multilayer perceptron is the back-propagation algorithm. The training proceeds in two phases:
- **Forward Phase**:
  - In this phase, the synaptic weights of the network are fixed and the input signal is propagated through the network, layer by layer, until it reaches the output.
  - Thus, in this phase, changes are confined to the activation potentials and outputs of the neurons in the network.
- **Backward Phase**:
  - In this phase, an error signal is produced by comparing the output of the network with a desired response.
  - The resulting error signal is propagated in backward direction through the network, again layer by layer. In this second phase, successive adjustments are made to the synaptic weights of the network

## Backpropagation

**Algorithm**

1. Initialize all weights and biases in *network*
2. While terminating condition is not satisfied
3. for each training tuple ***X*** in *D*
4. for each input layer unit *j*: $y_j = x_j$; // output of an input unit is its actual input value
5. for each hidden or output layer unit *j*
   1. *compute the net input of unit j with respect to the previous layer,*

      $$v_j = \sum_i w_{ji} y_i$$

   2. *compute the output of each unit j,* $y_j = \dfrac{1}{1+e^{-v_j}}$

## Backpropagation

**Algorithm**

6. for each unit *j* in the output layer

$$\delta_j = e_j \varphi_j'(v_j) = \varphi_j'(v_j)(d_j - y_j)$$

7. for each unit *j* in the hidden layers, from the last to the first hidden layer

$$\delta_j = \varphi_j'(v_j) \sum_{\mathrm{k}} \delta_k w_{kj}$$

8. for each weight $w_{ji}$ in *network*

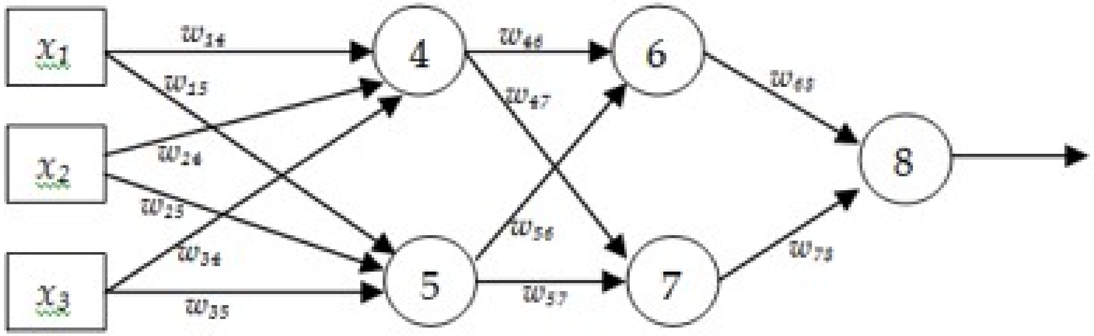
$$\Delta w_{ji} = \alpha \delta_j y_i$$

$$w_{ji} = w_{ji} + \Delta w_{ji}$$ *//Weight Update*

## Backpropagation

- **Example:** Consider a MLP given below. Let the learning rate be 1. The initial weights of the network are given in the table below. Assume that first training tuple is (1, 0, 1) and its target output is 1. Calculate weight updates by using back-propagation algorithm.
  - Assume

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




| $W_{14}$ | $W_{15}$ | $W_{24}$ | $W_{25}$ | $W_{34}$ | $W_{35}$ | $W_{46}$ | $W_{47}$ | $W_{56}$ | $W_{57}$ | $W_{68}$ | $W_{78}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.6 | 0.4 | 0.2 | -0.3 | 0.7 | -0.6 | 0.4 | 0.7 | 0.1 | 0.8 | 0.2 | 0.5 |

Backpropagation

- **Solution**
- *Forward Pass*
- $v_4=1*0.6+0*0.2+1*0.7=1.3$ $y_4=1/1+e^{-1.3}=0.786$
- $v_5=1*0.4+0*(-0.3)+1*(-0.6)=-0.2$ $y_5=1/1+e^{0.2}=0.45$
- $v_6=0.786*0.4+0.45*0.1=0.36$ $y_6=1/1+e^{-0.36}=0.59$
- $v_7=0.786*0.7+0.45*0.8=0.91$ $y_7=1/1+e^{-0.91}=0.71$
- $v_8=0.59*0.2+0.71*0.5=0.47$ $y_8=1/1+e^{-0.47}=0.61$

# Introduction to Machine Learning

Types of Algorithms in Deep Learning

- Convolutional Neural Networks (CNNs)
- Recurrent Neural Networks (RNNs)
- Long Short Term Memory Networks (LSTMs)
- Generative Adversarial Networks (GANs)
- Multilayer Perceptron (MLPs)
- Deep Belief Networks (DBNs)
- Autoencoders

Thank you!!!