Unit 4
Learning

(10 Hours)

4.1 Introduction

4.2 Concept of Learning

4.3 Types of Learning: Supervised, Unsupervised and Reinforcement Learning

4.4 Learning by Genetic Algorithms

4.5 Learning with Neural Networks

4.5.1 Introduction, Biological Neural Networks Vs. Artificial Neural Networks (ANN),

4.5.2 Mathematical Model of ANN,

4.5.3 Activation Functions: Linear, Step Sigmoid,

4.5.4 Types of ANN: Feed-forward, Recurrent, Single Layered, Multi-Layered,

4.5.5 Application of Artificial Neural Networks, Learning by Training ANN, Perceptron Learning, Back-propagation Learning

Sanjeev Thapa, Er. DevOps, SRE, CKA, RHCSA, RHCE, RHCSA-Openstack, MTCNA, MTCTCE, UBSRS, HEv6, Research Evangelist

4.1 Introduction

- Learning is a key aspect of intelligence in both humans and machines.
- In Artificial Intelligence (AI), learning refers to the ability of a system to improve its behavior or performance by gaining experience from data or interactions with the environment. Instead of being programmed with fixed rules, a learning system adapts its internal structure based on past observations.
- The importance of learning in AI arises from the complexity and dynamic nature of real-world problems. Many situations cannot be solved using static rules because environments change and new patterns emerge over time.
- Learning allows AI systems to recognize patterns, make predictions, and take decisions automatically.
- Today, learning techniques form the foundation of applications such as speech recognition, image processing, medical diagnosis, recommendation systems, robotics, and autonomous systems.

A baby is naturally curious and does not know what electricity is. When the baby **first puts a finger into an electric** socket, it receives a painful shock. This pain becomes a negative experience. The next time the baby sees the socket, it avoids touching it. This change in behavior happens because of past experience. Therefore, this situation clearly shows **learning**, where behavior improves through experience. It is a simple and real-life example of **reinforcement learning**, in which an action followed by punishment teaches what should not be done.

Is this learning?

Yes.
The baby has **learned from experience**.

Why is it learning?

At first:

- The baby **does not know** what electricity is.
- There is **no prior knowledge** or warning.
- The action (putting a finger in the socket) is done **out of curiosity**.

What happens:

- The baby receives a **shock** (pain).
- This pain acts as **negative feedback** (penalty).

Next time:

- The baby **avoids** putting the finger into the socket.
- The behavior **changes due to past experience**.

👉 A **change in behavior due to experience** is the **definition of learning**.

Learning using AI terms

Real Life (Baby)	**AI / Learning Concept**
Baby	Agent
Electric socket	Environment
Putting finger in socket	Action
Electric shock (pain)	Negative reward (penalty)
Avoiding socket next time	Learned behavior / Policy

Which type of learning is this?

✅ **Reinforcement Learning**

Because:

- There is **no teacher** telling the baby what is right or wrong
- Learning happens through **trial and error**
- Behavior is guided by **reward and punishment**

No.	Domain	Situation (Experience)	Feedback (Reward / Punishment)	Learned Behavior
1	Child Safety	Baby touches electric socket	Electric shock (pain)	Avoids touching the socket
2	Home Safety	Child touches hot stove	Burn / pain	Avoids touching hot objects
3	Animal Behavior	Dog goes near fire	Heat and pain	Stays away from fire
4	Animal Training	Dog sits when given command	Food reward	Repeats sitting behavior
5	Education	Student forgets homework	Teacher scolds	Submits homework on time
6	Road Safety	Person crosses road carelessly	Near accident	Uses zebra crossing
7	Workplace	Employee misses deadline	Warning from manager	Manages time better
8	Sports	Player ignores warm-up	Muscle injury	Always warms up before playing
9	Daily Life	Person slips on wet floor	Fall and pain	Walks carefully on wet floor
10	Nature / Animals	Monkey touches electric fence	Electric shock	Avoids the fence

4.2 Concept of Learning

In AI, learning can be understood as a process through which a system acquires knowledge and improves its performance over time. A system is said to learn if its performance on a given task improves with experience. This experience is usually provided in the form of data, examples, or feedback from the environment.

The learning process generally involves:

- Collecting data or experiences
- Extracting useful information or patterns
- Updating internal parameters or knowledge structures
- Using the updated knowledge to perform better in future situations

1	Age	Gender	Education Level	Job Title	Years of Experience	Salary
2	32	Male	Bachelor's	Software Engineer	5	90000
3	28	Female	Master's	Data Analyst	3	65000
4	45	Male	PhD	Senior Manager	15	150000
5	36	Female	Bachelor's	Sales Associate	7	60000
374	29	Female	Bachelor's	Junior Project Manager	2	40000
375	34	Male	Bachelor's	Senior Operations Coordinator	7	90000
376	44	Female	PhD	Senior Business Analyst	15	150000

Learning is evaluated using a performance measure such as accuracy, error rate, speed, or reward.

For example, if a program learns to classify emails more accurately as spam or non-spam after seeing more examples, it is considered to have learned. Thus, learning bridges the gap between raw data and intelligent behavior.

https://github.com/sanjeevlcc/notes_2081/blob/main/Simulation%20and%20modeling/CCT%20AI%20ML%20trainings%202025/Day_2/Models/Salary_Prediction_Model_using_Multiple_Linear_Regression%20(1).ipynb

```
]:  #example
    predict_salary(age=30,gender='Male',education_level='Bachelors',job_title='Data Scientist',years_of_experience=15)

Predicted Salary for Input: NPR 85905.37
```

Concept of Learning using KNN (Simple Example)

Given Data (6 People)

Each person is described using **two features**:

- **Age**
- **Cigarettes per day**

Person	Age	Cigarettes/Day	Class
P1	22	10	Smoker

New Person (Unknown Class)

Age	Cigarettes/Day
26	11

KNN Algorithm (k = 3, simple idea)

1. Measure distance between the new person and all 6 people

P2	25	12	Smoker
P3	28	15	Smoker
P4	23	0	Non-Smoker
P5	27	1	Non-Smoker
P6	30	0	Non-Smoker

2. Select **3 nearest neighbors**
3. Use **majority voting** to decide the class

Nearest Neighbors (Closest 3)

- P2 (25, 12) → Smoker
- P1 (22, 10) → Smoker
- P3 (28, 15) → Smoker

Output (Learning Result)

Smoker = 3
Non-Smoker = 0

✅ **Prediction: Smoker**

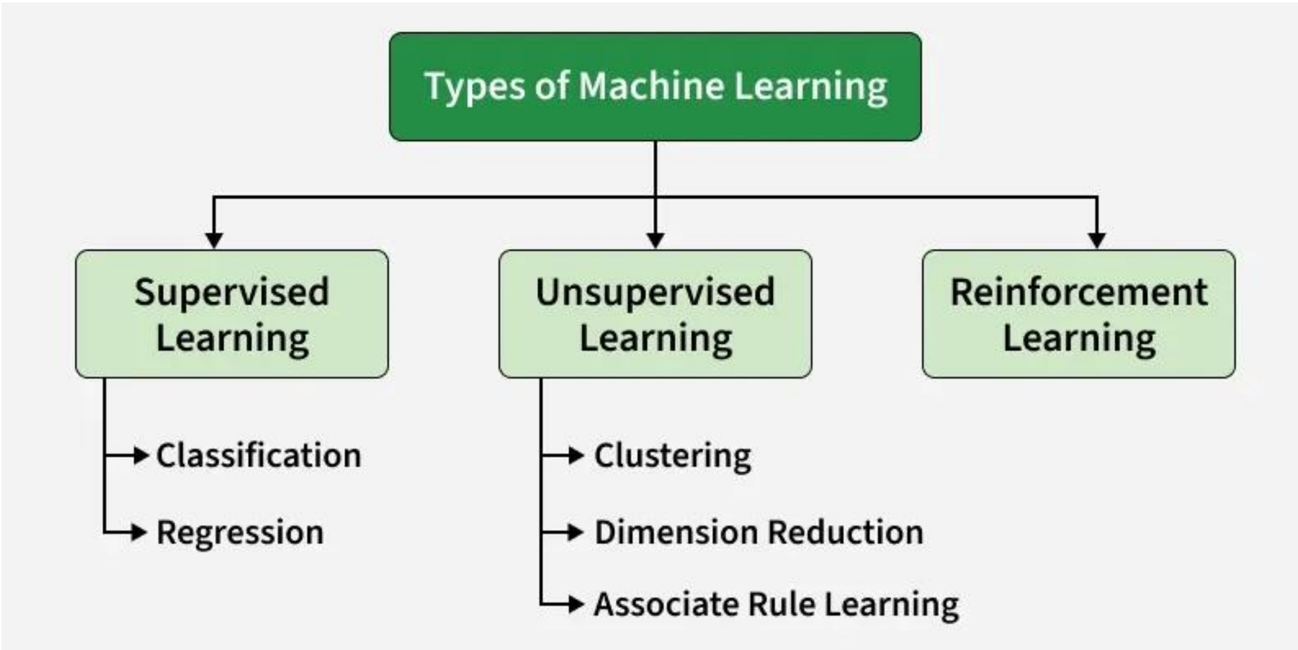
Why this is Learning?

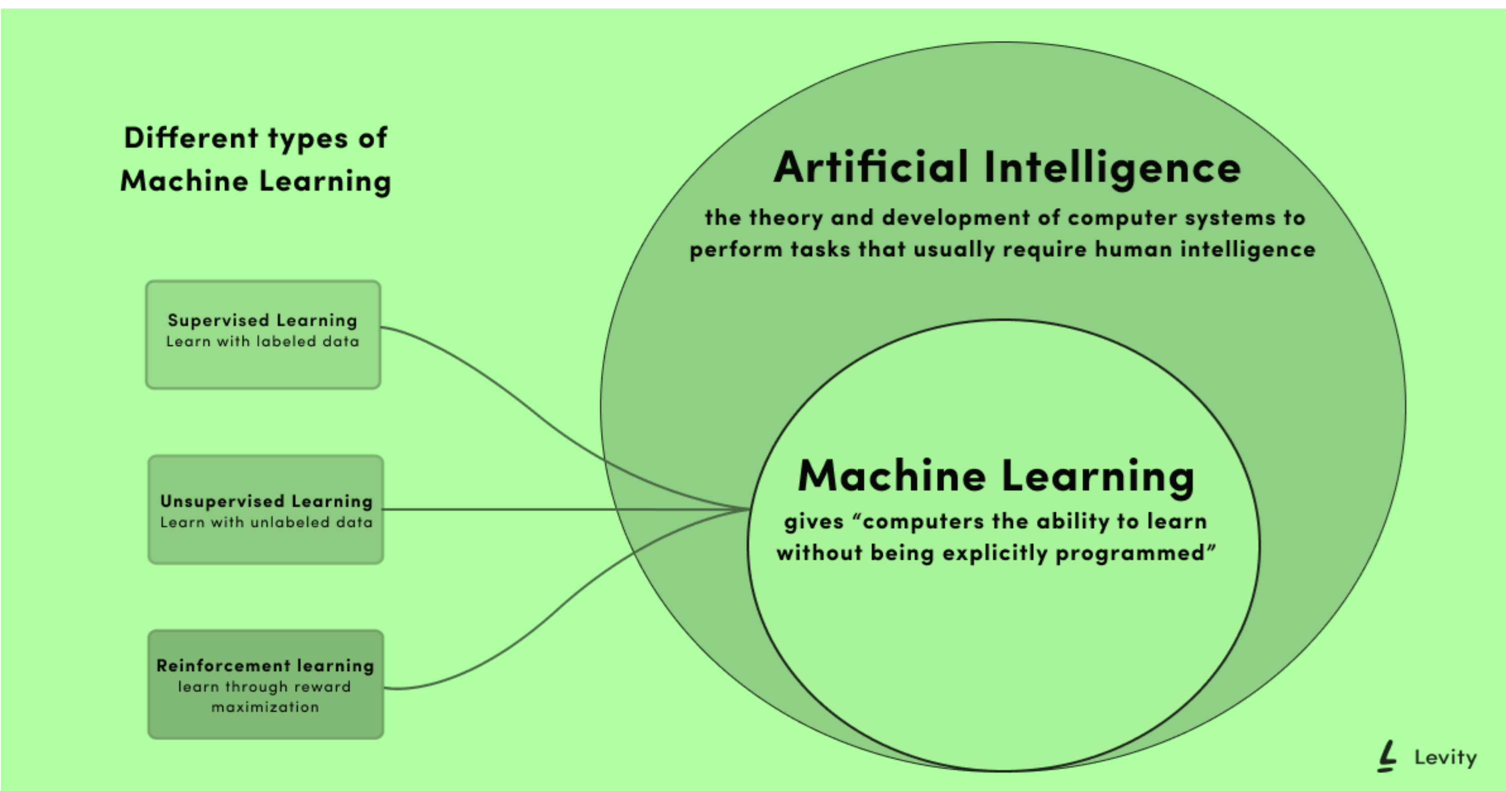
- The system **uses past data (experience)**
- It **compares new data with known examples**
- It **classifies without explicit rules**

👉 This is **supervised learning** using the **K-Nearest Neighbors (KNN) algorithm**.

4.3 Types of Learning:

In Artificial Intelligence and Machine Learning, learning methods are classified based on how a system receives information and feedback. The three main types of learning are **Supervised Learning**, **Unsupervised Learning**, and **Reinforcement Learning**.





4.3.1 Supervised Learning

Supervised learning is a type of learning in which the system is trained using **labeled data**. Each input is provided with a **correct output**, and the system learns by comparing its predicted output with the actual output.

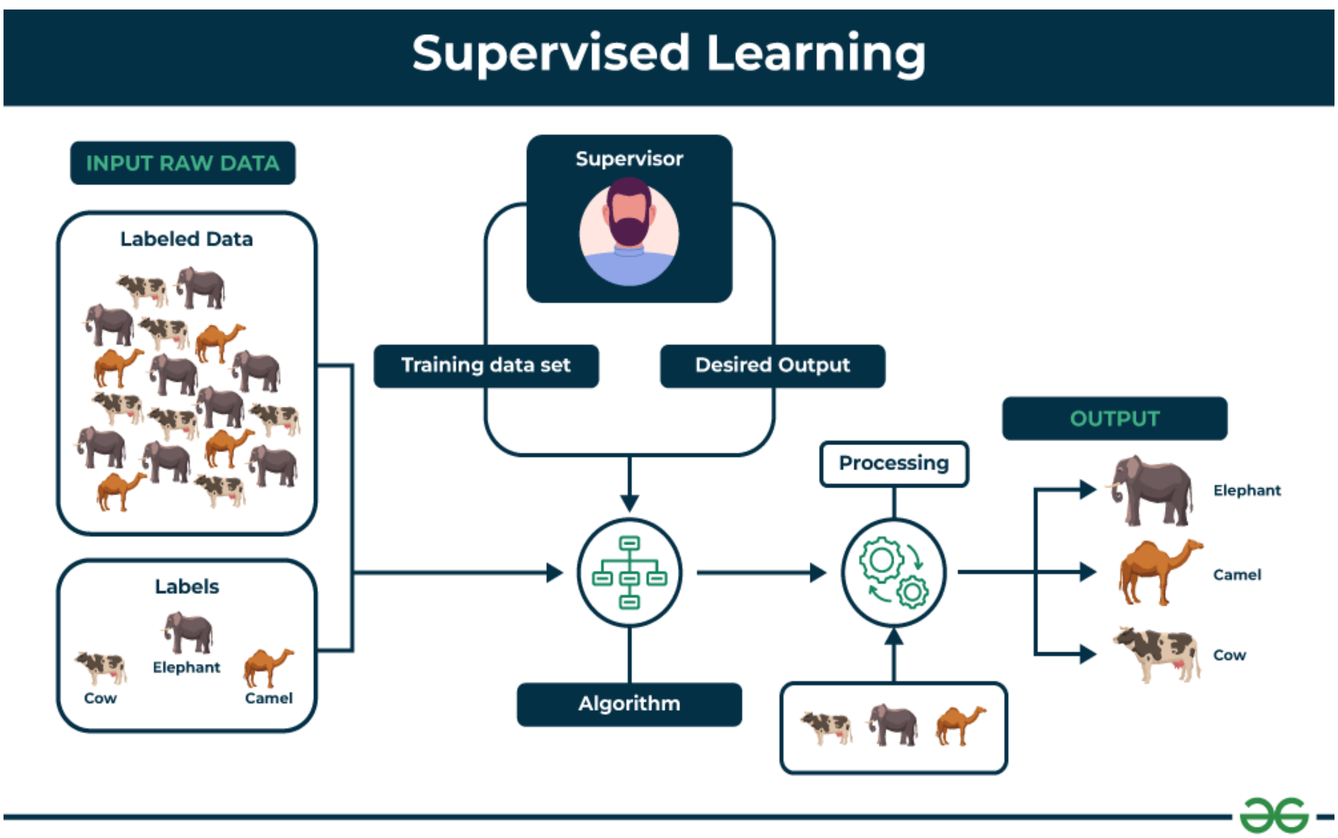
The objective of supervised learning is to learn a mapping between input and output so that the system can correctly predict outputs for new, unseen data.

Key characteristics

- Uses labeled training data
- Learning is guided by a "teacher" (known output)
- Error is calculated and minimized

Common tasks

- Classification
- Regression



Examples

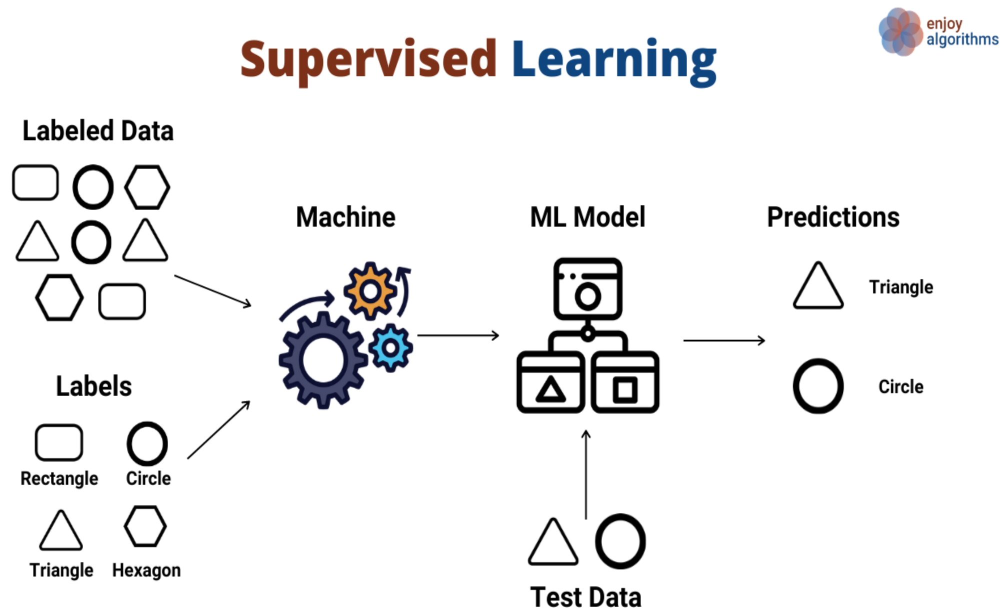
1. **Email Spam Detection**
   Emails are labeled as *spam* or *not spam*. The system learns patterns and classifies new emails.

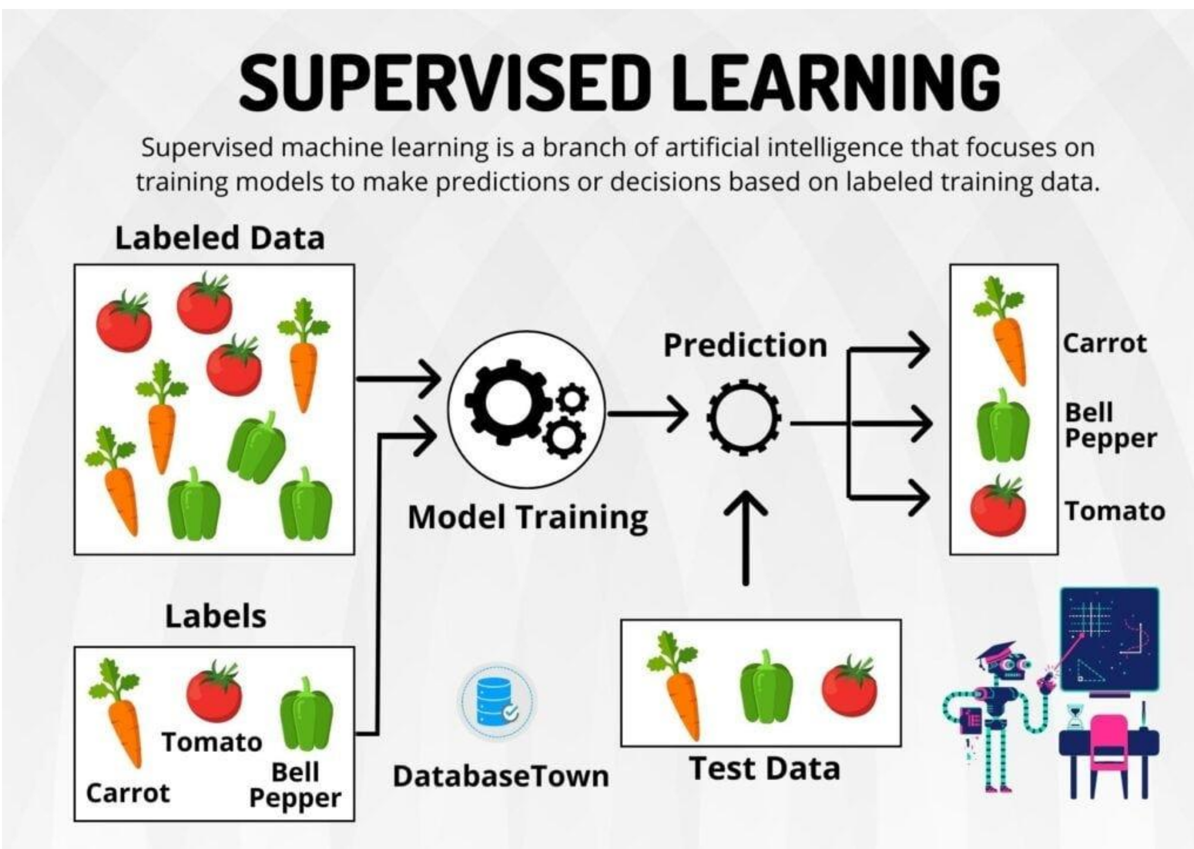
2. **Student Result Prediction**
   Inputs: study hours, attendance, test scores
   Output: pass or fail

3. **Salary Prediction**
   Inputs: experience, education, skills
   Output: salary amount

Popular algorithms

- Linear Regression
- Decision Tree
- K-Nearest Neighbors (KNN)
- Support Vector Machine (SVM)
- Neural Networks





4.3.2 Unsupervised Learning

Unsupervised learning is a type of learning in which the system is trained using **unlabeled data**. The system does not know the correct output in advance and must discover hidden patterns or structures on its own.

The main goal of unsupervised learning is **pattern discovery** rather than prediction.

Key characteristics

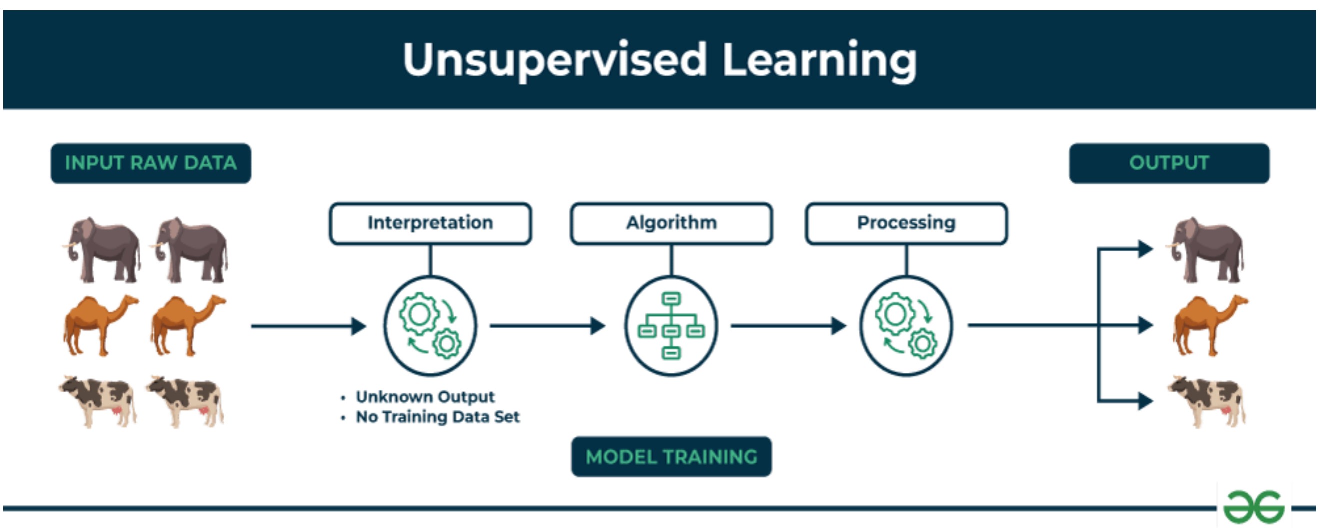
- No labeled output data
- No teacher or guidance
- Learns structure from data

Common tasks

- Clustering
- Association
- Dimensionality reduction

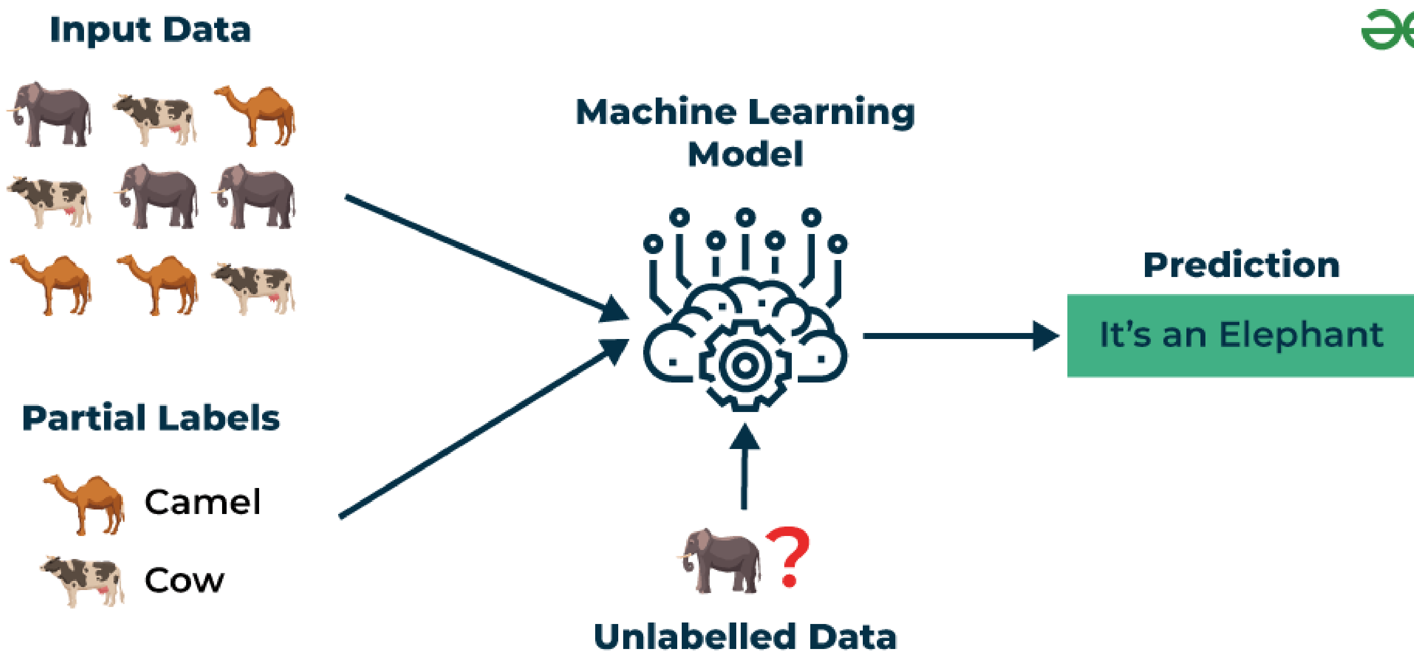
Examples

1. **Customer Segmentation**
   Customers are grouped based on buying behavior without predefined labels.
2. **Grouping Students**
   Students are grouped based on marks or interests without naming groups in advance.
3. **Market Basket Analysis**
   Finding products that are often bought together.



Popular algorithms

- K-Means Clustering
- Hierarchical Clustering
- Apriori Algorithm
- Principal Component Analysis (PCA)



4.3.3 Reinforcement Learning

Reinforcement learning is a type of learning where an **agent learns by interacting with an environment**. The agent performs actions and receives feedback in the form of **rewards or punishments**.

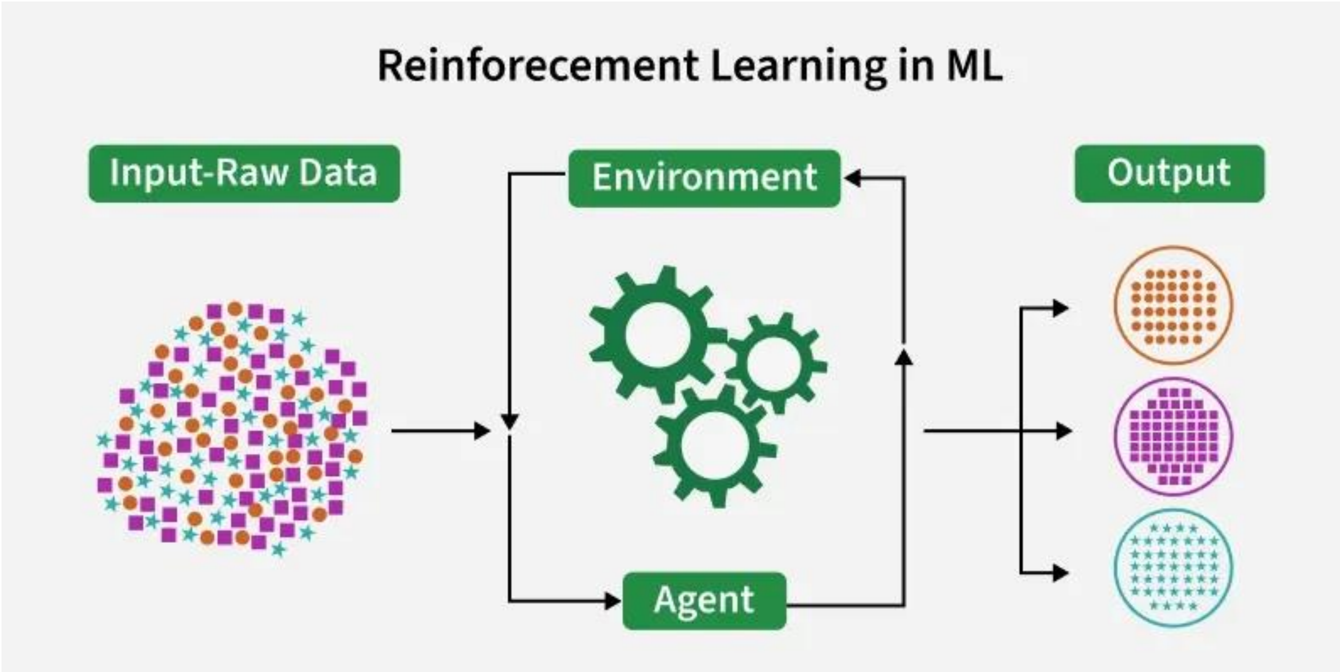
The goal is to learn a strategy (policy) that **maximizes total reward over time**.

Key characteristics

- Learning through trial and error
- No labeled data
- Feedback is delayed
- Behavior is guided by reward or punishment

Main components

- Agent
- Environment
- Action
- Reward



Examples

1. **Baby Touching Electric Socket**
   Pain acts as punishment, and the baby avoids the socket next time.

2. **Dog Training**
   Dog sits → gets food (reward) → repeats action.

3. **Game Playing (Chess / Video Games)**
   Winning gives reward; losing gives penalty.

4. **Robot Navigation**
   Robot learns the best path by avoiding obstacles and reaching the goal.

Popular algorithms

- Q-Learning
- SARSA
- Deep Q-Network (DQN)

Comparison

Learning Type	Data Used	Feedback	Example
Supervised	Labeled data	Correct output	Spam detection
Unsupervised	Unlabeled data	No feedback	Customer clustering
Reinforcement	Interaction data	Reward / punishment	Game playing

KNN Numerical (Smoking Prediction)

Question

You are given data of **10 people** with two features:

- **Age (years)**
- **Cigarettes per day**

Class label: **Smoker (S)** or **Non-Smoker (NS)**

Use **KNN** to predict whether the **new person** is **Smoker or Non-Smoker** for **k = 1, 2, 3** using **Euclidean distance**:

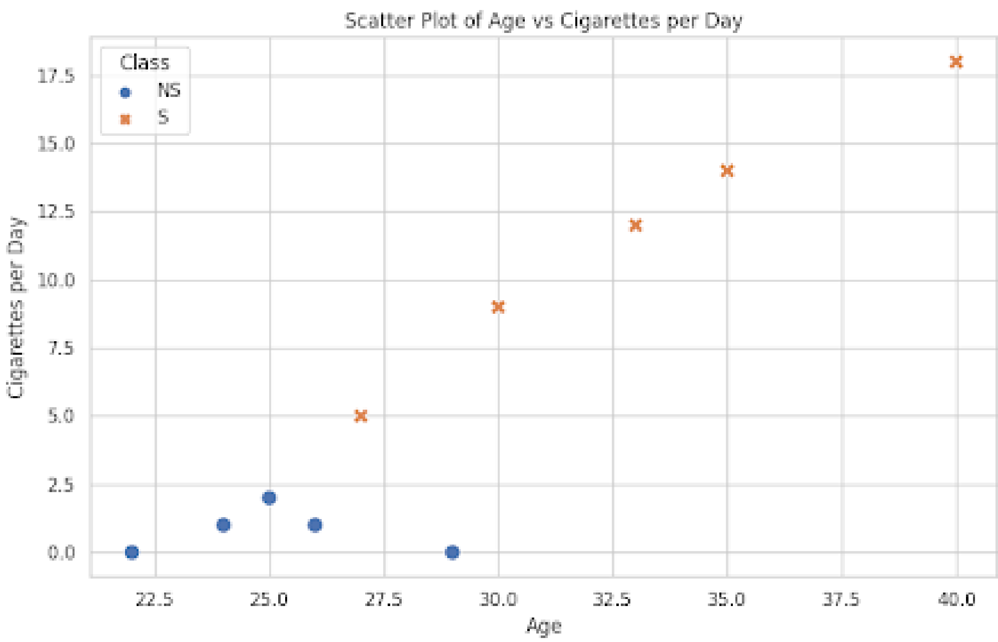
$$d = \sqrt{(Age_2 - Age_1)^2 + (Cig_2 - Cig_1)^2}$$

Given Training Data (10 People)

Person	Age	Cig/Day	Class
P1	22	0	NS
P2	24	1	NS

P3	25	2	NS
P4	26	1	NS
P5	29	0	NS
P6	27	5	S
P7	30	9	S
P8	33	12	S
P9	35	14	S
P10	40	18	S

New Person (Query Point) : X = (Age = 26, Cig/Day = 4)



Step 1: Compute Distances from X

Person	Point (Age, Cig)	Class	Distance to X
P6	(27, 5)	S	$\sqrt{1^2 + 1^2} = 1.414$
P3	(25, 2)	NS	$\sqrt{1^2 + 2^2} = 2.236$
P4	(26, 1)	NS	$\sqrt{0^2 + 3^2} = 3.000$
P2	(24, 1)	NS	$\sqrt{2^2 + 3^2} = 3.606$
P5	(29, 0)	NS	$\sqrt{3^2 + 4^2} = 5.000$
P1	(22, 0)	NS	$\sqrt{4^2 + 4^2} = 5.657$
P7	(30, 9)	S	$\sqrt{4^2 + 5^2} = 6.403$
P8	(33, 12)	S	$\sqrt{7^2 + 8^2} = 10.630$
P9	(35, 14)	S	$\sqrt{9^2 + 10^2} = 13.454$
P10	(40, 18)	S	$\sqrt{14^2 + 14^2} = 19.799$

Step 2: Sort Nearest Neighbors (Smallest distance)

1. **P6 (S)** — 1.414
2. **P3 (NS)** — 2.236

3. **P4 (NS)** — 3.000

Predictions

✅ **For k = 1**

Nearest neighbor = **P6 (Smoker)**
➡️ **Prediction: Smoker**

✅ **For k = 2**

Neighbors: **P6 (S)** and **P3 (NS)** → **1 Smoker, 1 Non-Smoker (Tie)**
Tie rule (common): choose the class of the closest neighbor
Closest is P6 (S)
➡️ **Prediction (k=2): Smoker**

(Note: Another tie rule is "distance-weighted voting"; but nearest-neighbor tie break is simplest.)

✅ **For k = 3**

Neighbors: **P6 (S), P3 (NS), P4 (NS)**
Votes → Smoker = 1, Non-Smoker = 2
➡️ **Prediction (k=3): Non-Smoker**

Final Answer

- **k = 1 → Smoker**
- **k = 2 → Smoker** (tie → pick nearest)
- **k = 3 → Non-Smoker**

Question: KNN Classification (Study vs Non-Study)

A class survey collected data from **20 people**. Each person is described using two attributes:

	Study Hours per Day	**Attendance (%)**
Class label:	**Study (S)**	**Non-Study (NS)**

Use the **K-Nearest Neighbors (KNN)** algorithm to classify a **new person: Janaki** for **k = 1, 2, 3, and 4** using **Euclidean distance**.

Training Data

Person	**Study Hours**	**Attendance**	**Class**
Kritika	6	85	S
Suraj	5	80	S
Bindya	7	90	S
Surakshya	6	88	S
Aswriya	8	92	S
Suraksha	5	82	S
Durga	6	86	S
Ramesh	2	60	NS
Sita	1	55	NS
Gopal	3	65	NS
Nabin	2	62	NS

Prakash	4	70	NS
Anil	3	68	NS
Mina	1	50	NS
Sunita	2	58	NS
Pooja	7	89	S
Kiran	6	84	S
Bikash	3	66	NS
Deepak	4	72	NS
Laxmi	7	91	S

New Person (To Be Classified)

Name	**Study Hours**	**Attendance**
Janaki	**5**	**83**

Tasks

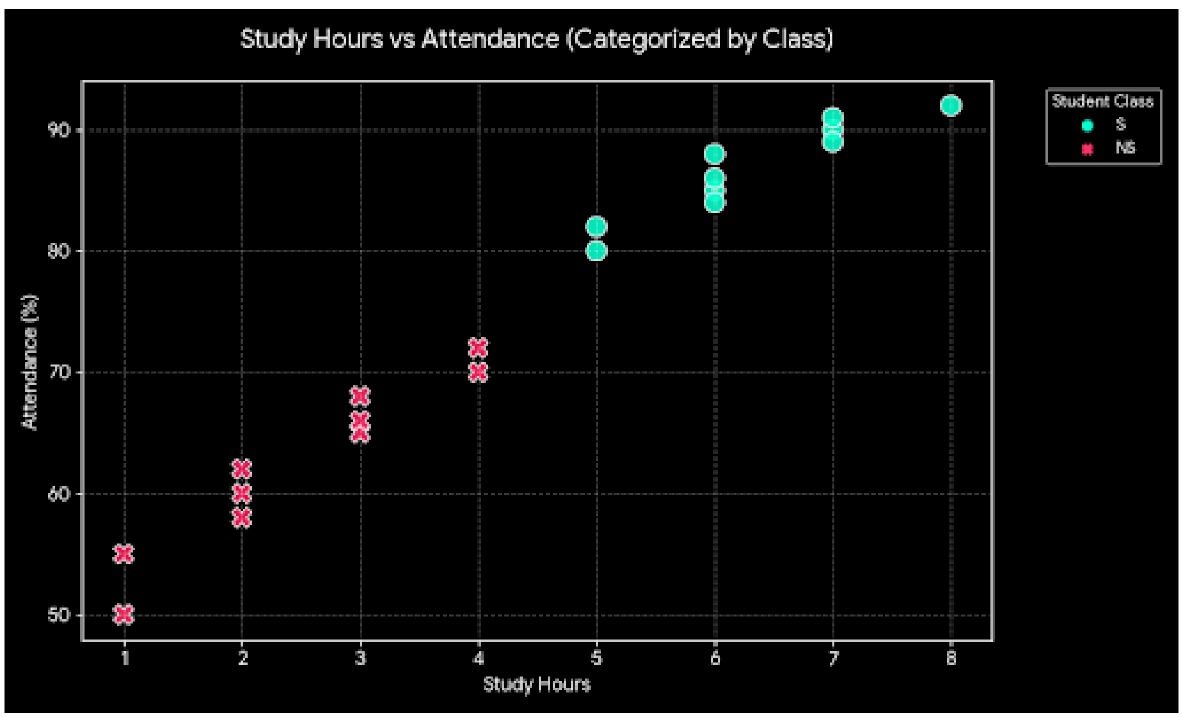
1. Calculate the distance of Janaki from all people.
2. Identify the nearest neighbors.
   - Predict Janaki's class for: **k = 1** **k = 2** **k = 3** **k = 4**

Answers

Nearest Neighbors (Closest First)

Neighbor	**Distance**	**Class**
Suraksha (5,82)	Very close	S
Kiran (6,84)	Close	S
Kritika (6,85)	Close	S
Suraj (5,80)	Slightly farther	S

(All nearest neighbors belong to **Study (S)** group.)



Predicted Class

k = 1 → Study **k = 2 → Study** **k = 3 → Study** **k = 4 → Study**

Using the KNN algorithm, Janaki is classified as a ***Study*** *person for all values of k (1, 2, 3, and 4), because the nearest neighbors mostly belong to the Study category.*

Question: KNN Classification (Exercise vs Non-Exercise)

A health survey collected data from **10 people**.
Each person is described using two attributes:

- **Daily Walking Time (minutes)**
- **Body Mass Index (BMI)**

Class label:

- **Exercise (E)**
- **Non-Exercise (NE)**

Use the **K-Nearest Neighbors (KNN)** algorithm with **Euclidean distance** to classify a **new person**.

Training Dataset (10 People)

Person	Walking Time (min)	BMI	Class
Asha	40	22	E
Rohan	35	23	E
Meena	45	21	E
Suman	30	24	E

Anita	50	20	E
Bikram	10	29	NE
Hari	15	28	NE
Nisha	12	30	NE
Prabin	20	27	NE
Sita	18	26	NE

New Person (To Be Classified)

Name	**Walking Time (min)**	**BMI**
Kamal	**28**	**25**

Tasks

1. Calculate the Euclidean distance between Kamal and each of the 10 people.
2. Identify the nearest neighbors.
3. Predict Kamal's class using KNN for: **k = 1** **k = 3** **k = 5**

OR

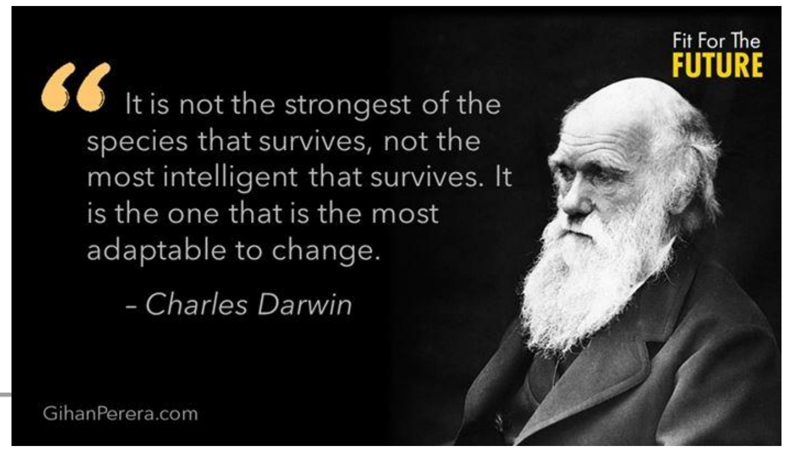
A health survey represents 10 people as coordinate points where **X = walking time (minutes)** and **Y = BMI**. Each person is labeled as **Exercise (E)** or **Non-Exercise (NE)**. The given points are: Asha (40,22), Rohan (35,23), Meena (45,21), Suman (30,24), Anita (50,20) labeled E; and Bikram (10,29), Hari (15,28), Nisha (12,30), Prabin (20,27), Sita (18,26) labeled NE. A new person, **Kamal**, has coordinates **(28,25)**. Using the **K-Nearest Neighbors (KNN)** algorithm with Euclidean distance, classify Kamal as Exercise or Non-Exercise for **k = 1, 3, and 5**.

4.4 Learning by Genetic Algorithms

Genetic Algorithms (GA) are **search and learning techniques** inspired by the process of **natural evolution**. They are used to find **optimal or near-optimal solutions** to complex problems where traditional methods are slow or ineffective.

In genetic algorithms, learning occurs by evolving better solutions over generations, rather than learning from labeled data or direct feedback.

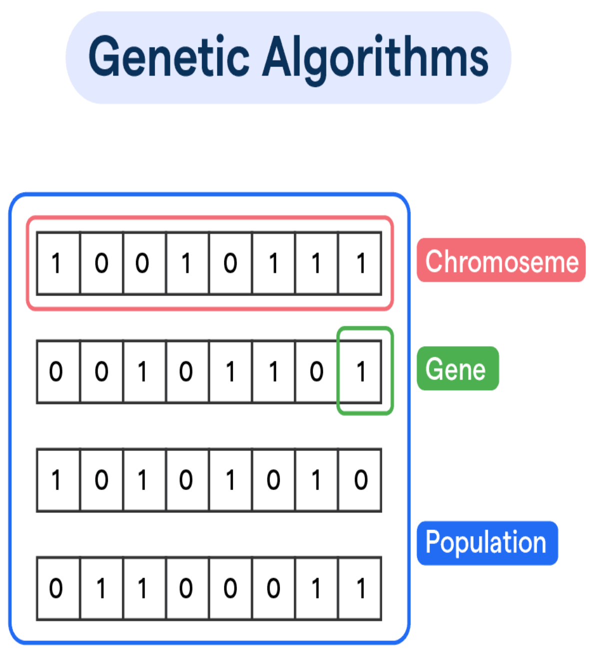
Basic Idea of Genetic Algorithms



Genetic Algorithms imitate the principle of **“survival of the fittest”**.
A population of possible solutions is created, and the best solutions are selected, combined, and modified to produce improved solutions.

Each solution is treated as an **individual**, and its quality is measured using a **fitness function**.

Key Components of Genetic Algorithms



1. Chromosome

A chromosome represents a **possible solution** to a problem. It is usually encoded as a **binary string**, numbers, or symbols.

Example:
101101 → a possible solution

---

2. Population

A population is a **set of chromosomes** (solutions) at a given time.

Example:
{101101, 110010, 011011, 100111}

3. Fitness Function

The fitness function evaluates **how good a solution is**.

- Higher fitness → better solution
- Lower fitness → weaker solution

Example:
Fitness = total profit, accuracy, or performance score

---

4. Selection

Selection chooses the **best chromosomes** for reproduction.

Common methods:

- Roulette wheel selection
- Tournament selection

Better solutions have a **higher chance of being selected**.

---

5. Crossover

Crossover combines two parent chromosomes to produce new offspring.

Example:
Parent 1: 101|011
Parent 2: 110|100

After crossover:
Child 1: 101100
Child 2: 110011

---

6. Mutation

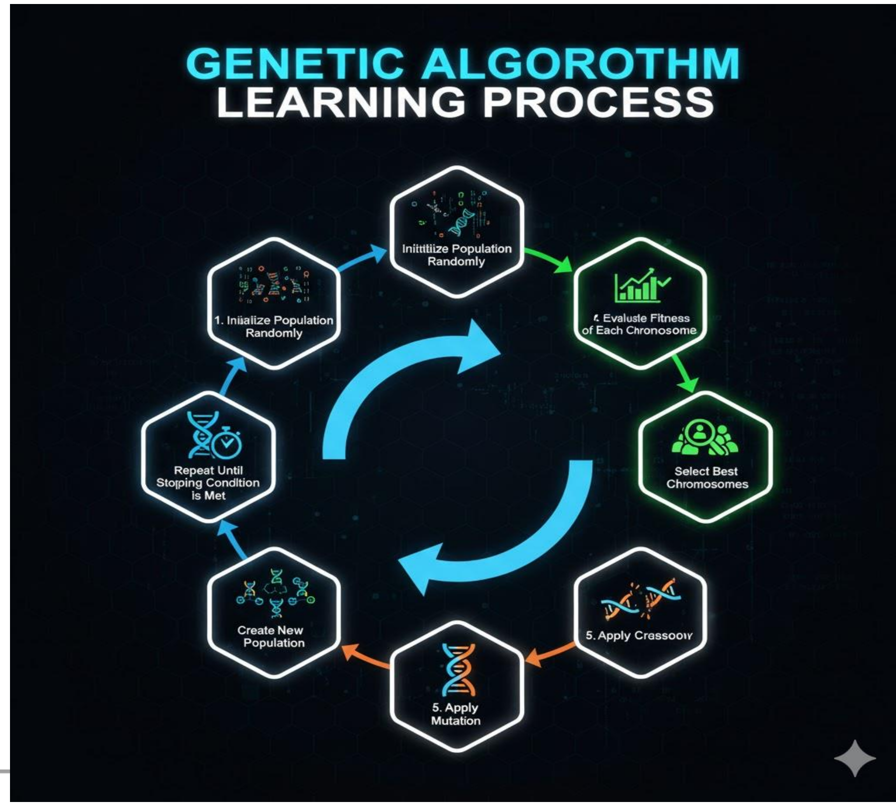
Mutation randomly changes some bits to maintain diversity.

Example:
Before mutation: 101100
After mutation: 101110

Mutation helps avoid **local optimum** solutions.

Genetic Algorithm Learning Process

1. Initialize population randomly
2. Evaluate fitness of each chromosome
3. Select best chromosomes
4. Apply crossover
5. Apply mutation
6. Create new population
7. Repeat until stopping condition is met



Why GA is Considered Learning

- GA **improves solutions over time**
- Uses **experience from previous generations**
- Adapts solutions based on fitness
- Does not require explicit programming rules

👉 This improvement through experience represents **learning**.

Suitable Examples

Example 1: Exam Timetable Scheduling

A college wants to create an exam timetable with:

- No subject clashes
- Minimum gaps between exams

GA generates multiple timetables, evaluates them using a fitness function, and evolves better schedules over generations.

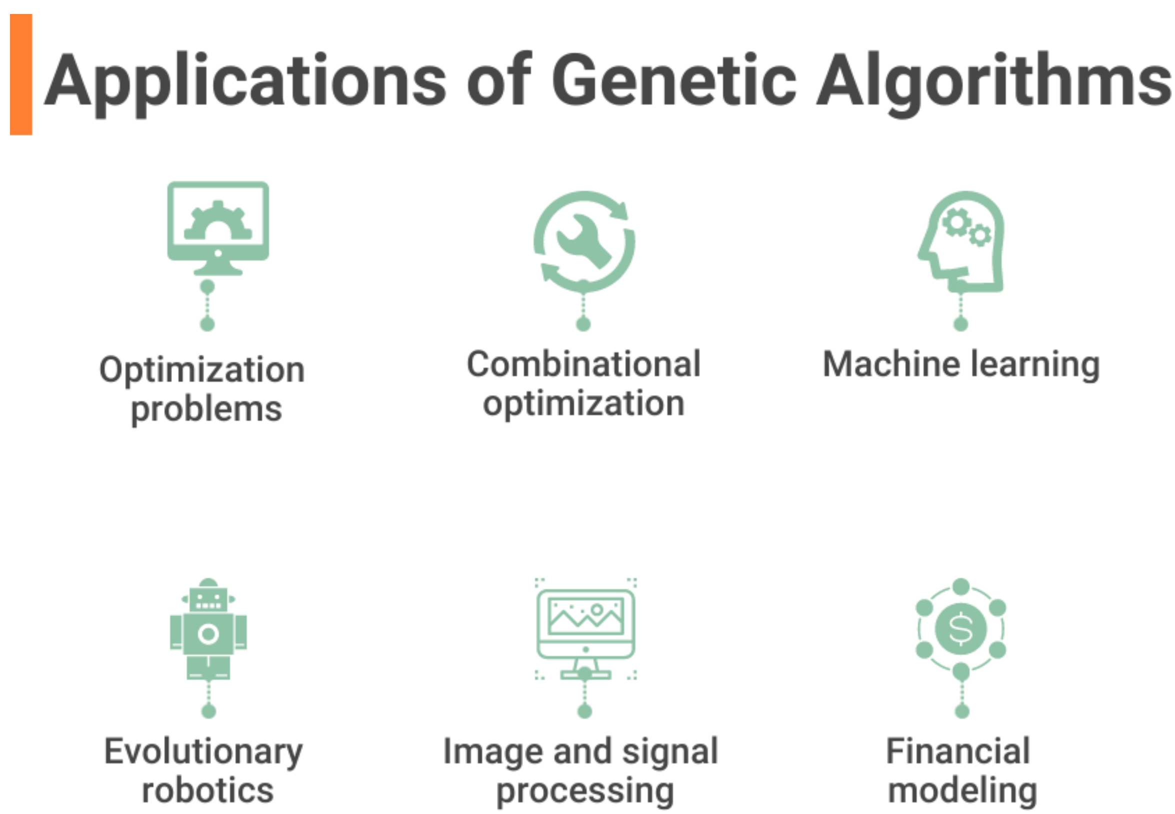
Example 2: Feature Selection in Machine Learning

From many features, GA selects the **best subset** that gives maximum accuracy.
Poor feature combinations are eliminated, and better ones are evolved.

Example 3: Path Finding in Robotics

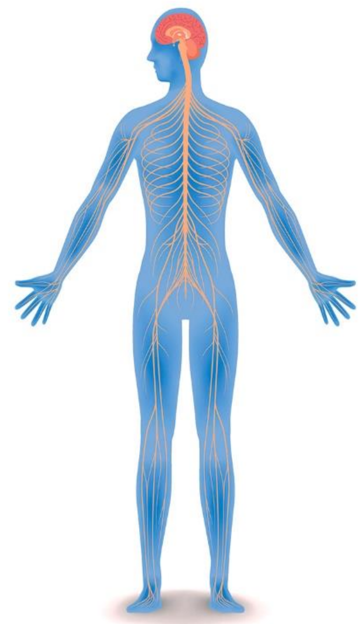
A robot must find the shortest path avoiding obstacles.
GA tries different paths and gradually learns the **best route**.

Advantages of Genetic Algorithms	**Limitations of Genetic Algorithms**
• Works well for complex and large search spaces • Does not require gradient or mathematical model • Avoids getting stuck in local optimum • Can handle multi-objective problems	• Computationally expensive • No guarantee of exact optimal solution • Requires careful parameter tuning • Slower than traditional methods for simple problems

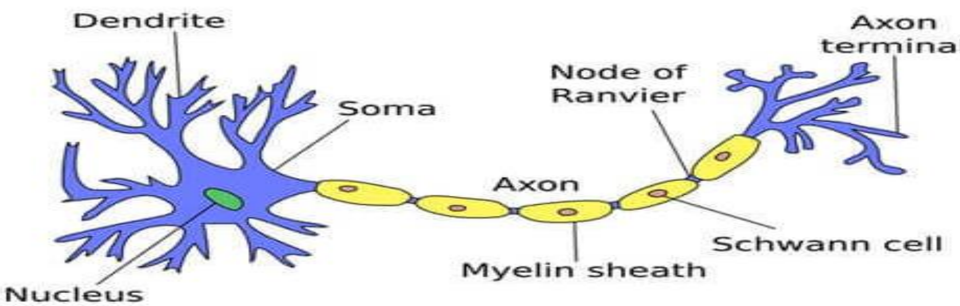


4.5 Learning with Neural Networks

Learning with Neural Networks refers to the process by which a system learns patterns, relationships, or functions from data by adjusting internal parameters called **weights**. Neural Networks are inspired by the working of the **human brain**, where learning occurs through experience and repeated exposure.



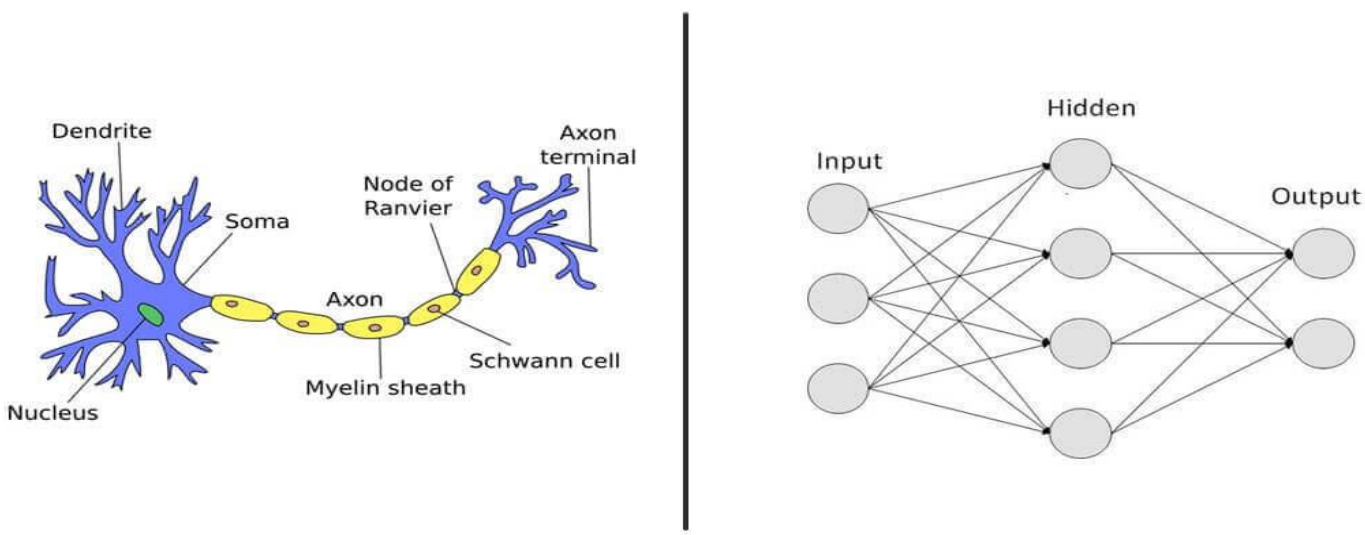
A neural network consists of interconnected processing units called **neurons**. These neurons work together to process input data and produce an output. Learning in neural networks happens when the network modifies its weights to reduce errors between the predicted output and the actual output.



Neural Networks are widely used because they can:

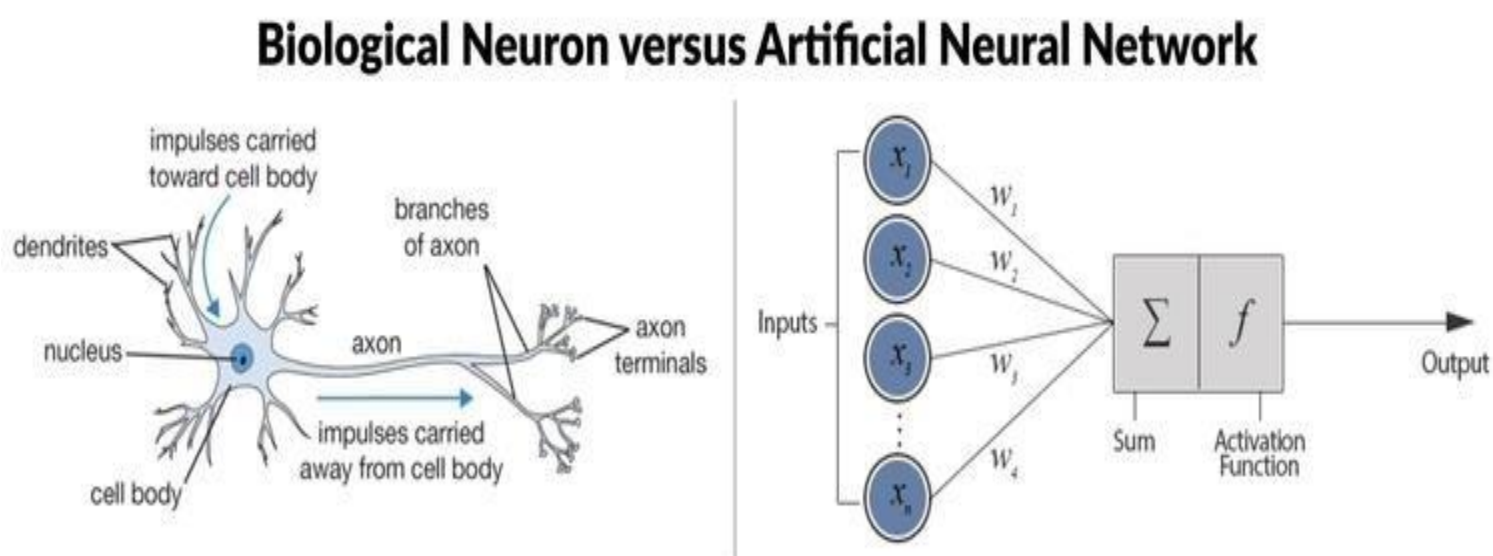
- Learn complex and non-linear relationships
- Generalize knowledge to unseen data
- Improve performance with more training data

Applications include image recognition, speech recognition, handwriting recognition, medical diagnosis, stock prediction, and natural language processing.



4.5.1 Biological Neural Networks Vs. Artificial Neural Networks (ANN)

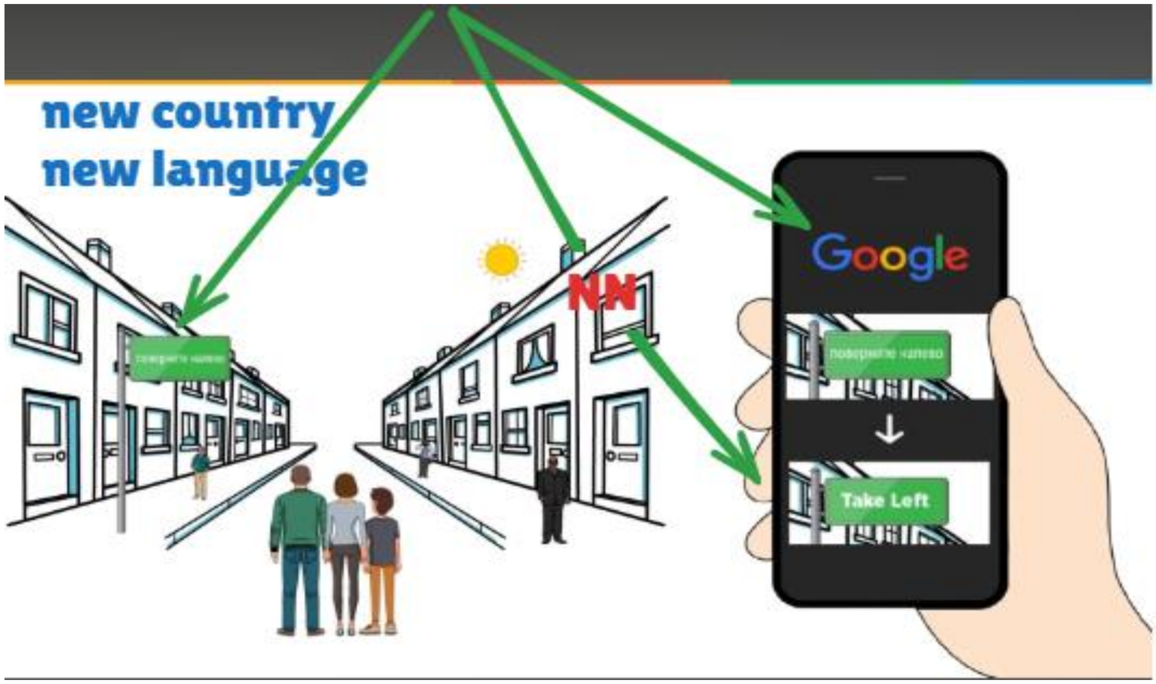
Biological Neural Networks	**Artificial Neural Networks (ANN)**
Biological Neural Networks exist in the human brain and nervous system. Key characteristics: • Neurons are biological cells • Communication occurs through electrical and chemical signals • Learning happens by strengthening or weakening synapses • Extremely complex and highly parallel • Capable of learning, reasoning, and creativity Example: Humans learning to recognize faces or languages through experience.	Artificial Neural Networks are computer-based models inspired by biological neurons. Key characteristics: • Neurons are mathematical functions • Communication happens through numerical weights • Learning occurs by adjusting weights using algorithms • Faster and scalable on machines • Designed for specific tasks Example: A neural network trained to recognize handwritten digits or predict house prices.



Comparison Table: Biological NN vs ANN

Aspect	**Biological Neural Network**	**Artificial Neural Network**
Nature	Biological	Mathematical / Computational
Processing Unit	Neuron	Artificial neuron
Signal Type	Electrical & chemical	Numerical values
Learning Method	Synaptic strength changes	Weight adjustment
Speed	Slow but highly parallel	Fast but less parallel
Flexibility	Very high	Task-specific
Energy Efficiency	Very high	Lower

Scenarios



- If you are in a **new country,** Road signs in a **foreign language,** A mobile phone using **Google Translate**
- Arrows pointing from the real world → phone → translated output, "NN" written near the scene

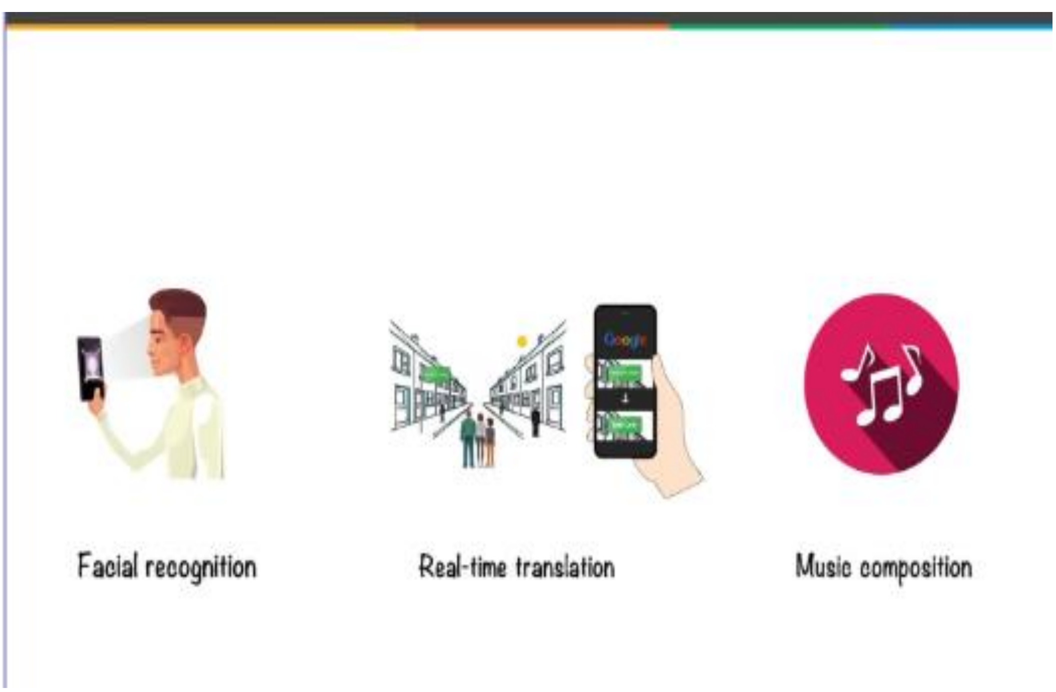
This image represents **real-time translation using a Neural Network**.

How it works conceptually:

1. Camera captures the image of a sign (input data)
2. Neural Network analyzes pixels and text patterns
3. Language model translates text into the user's language
4. Translated instruction appears on the phone (output)

👉 **Key idea:**
A neural network can **understand visual and language patterns** and convert them into meaningful output.

Applications of Neural Networks



- Facial recognition
- Real-time translation
- Music composition

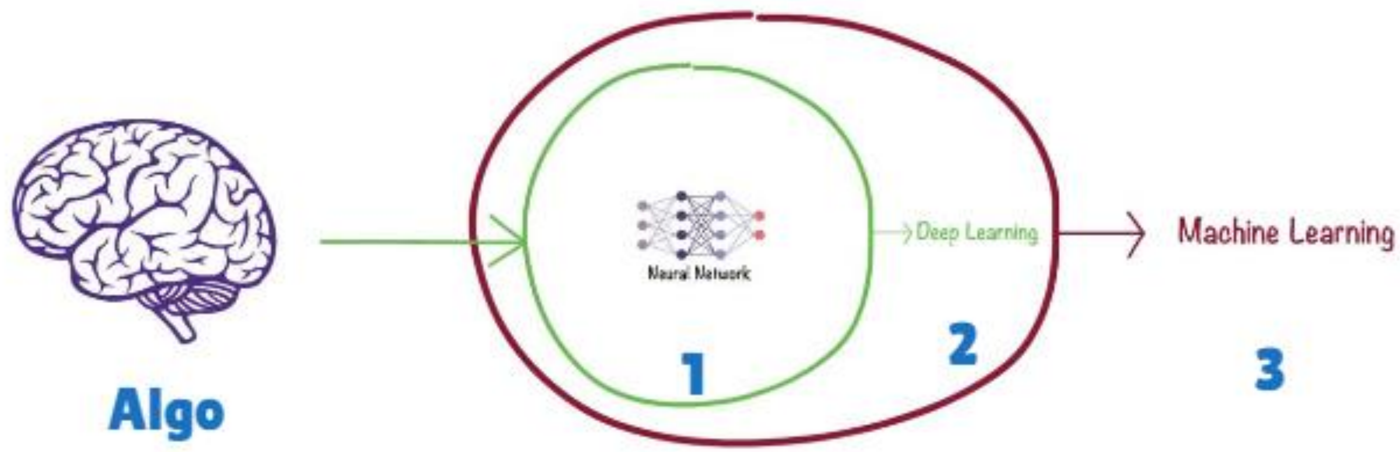
Interpretation

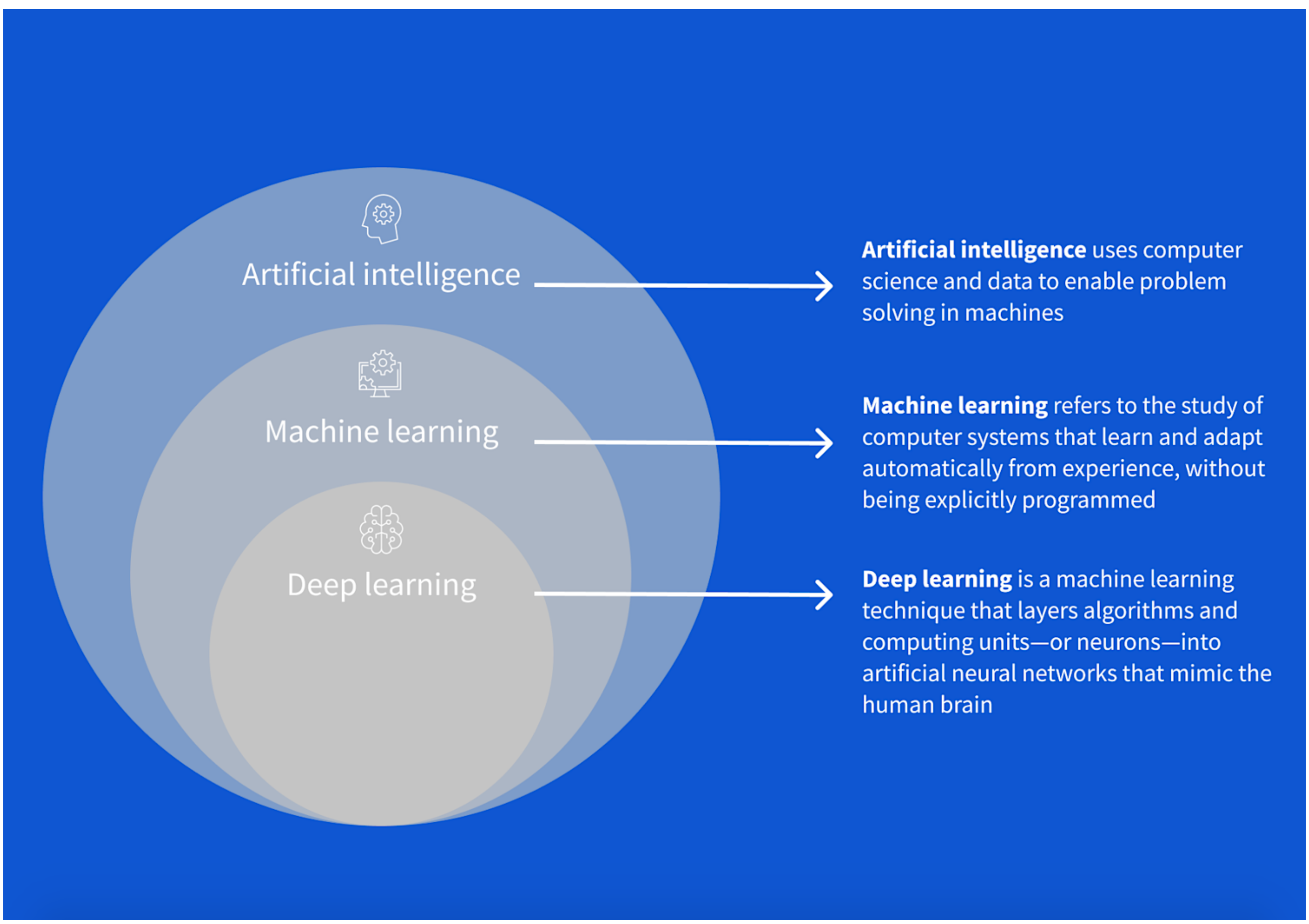
This image highlights **where neural networks are used**.

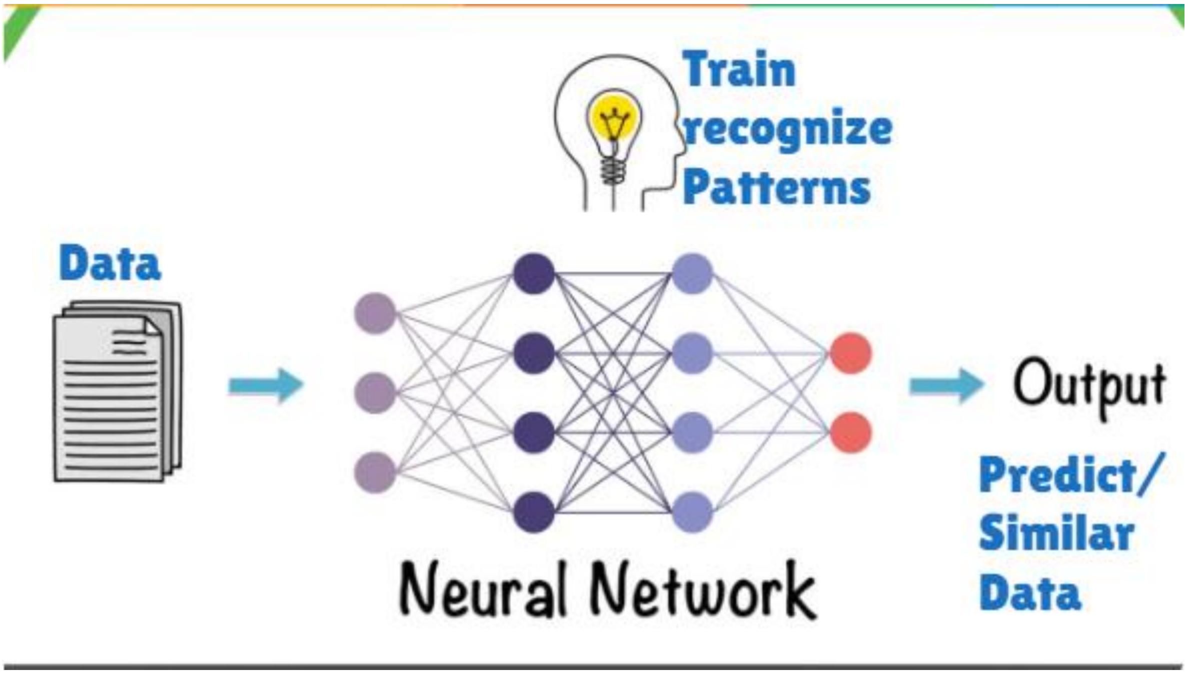
- Neural Networks are **general-purpose pattern learners**

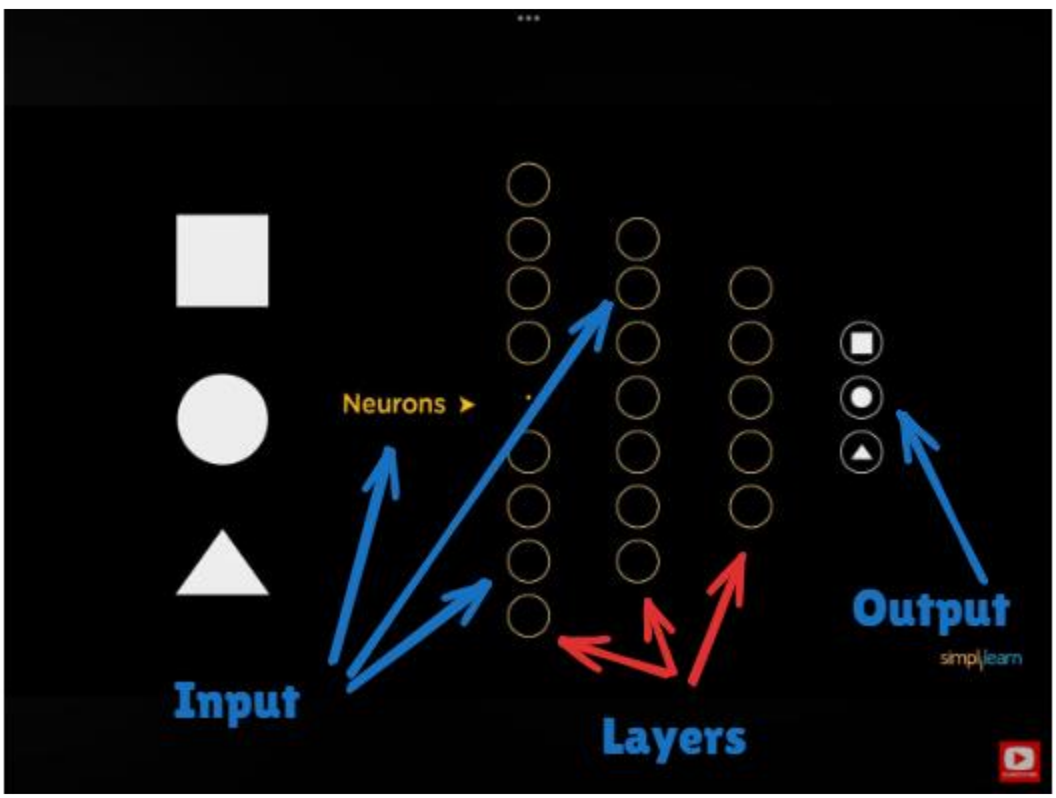
- Same core idea works for:
  - Face identification
  - Language translation
  - Music generation

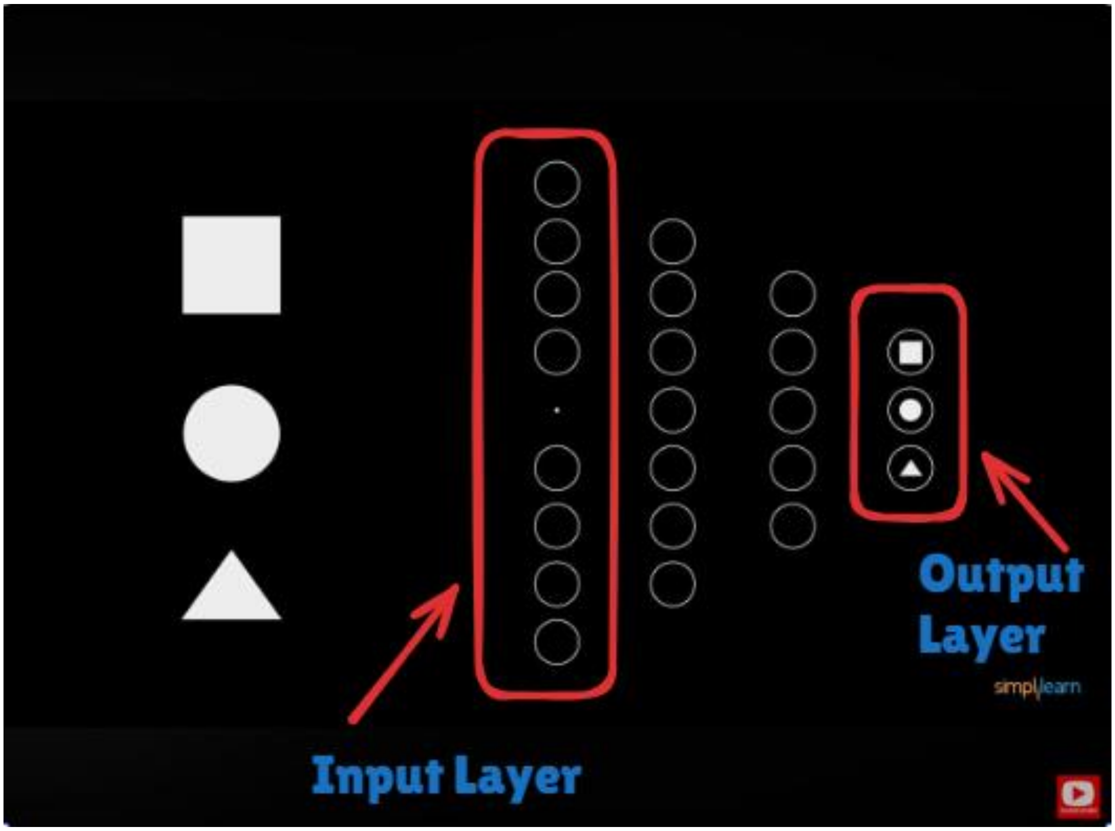
👉 **Key idea:**
Neural networks do not solve one problem only; they can learn **many types of patterns** from data.

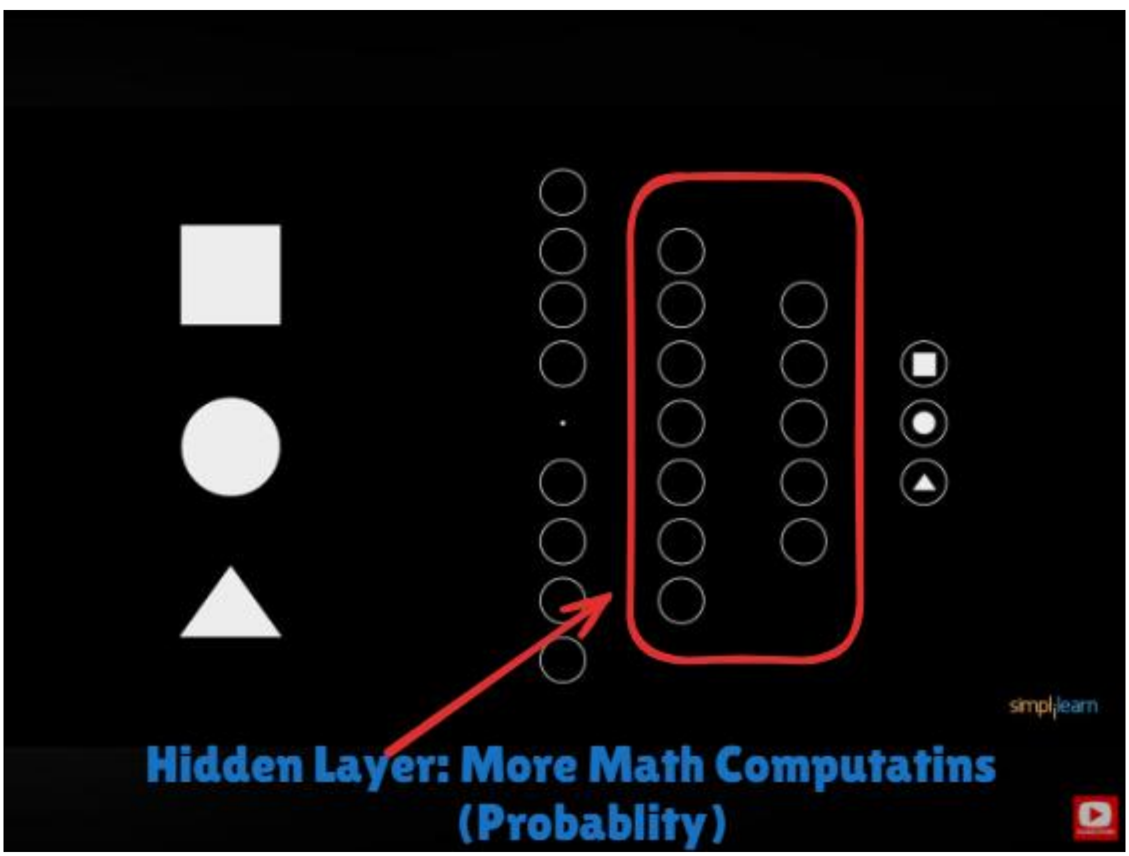





Train
recognize
Patterns
Data
Output
Predict/
Similar
Data
Neural Network


Neurons >
Input
Layers
Output
simplilearn

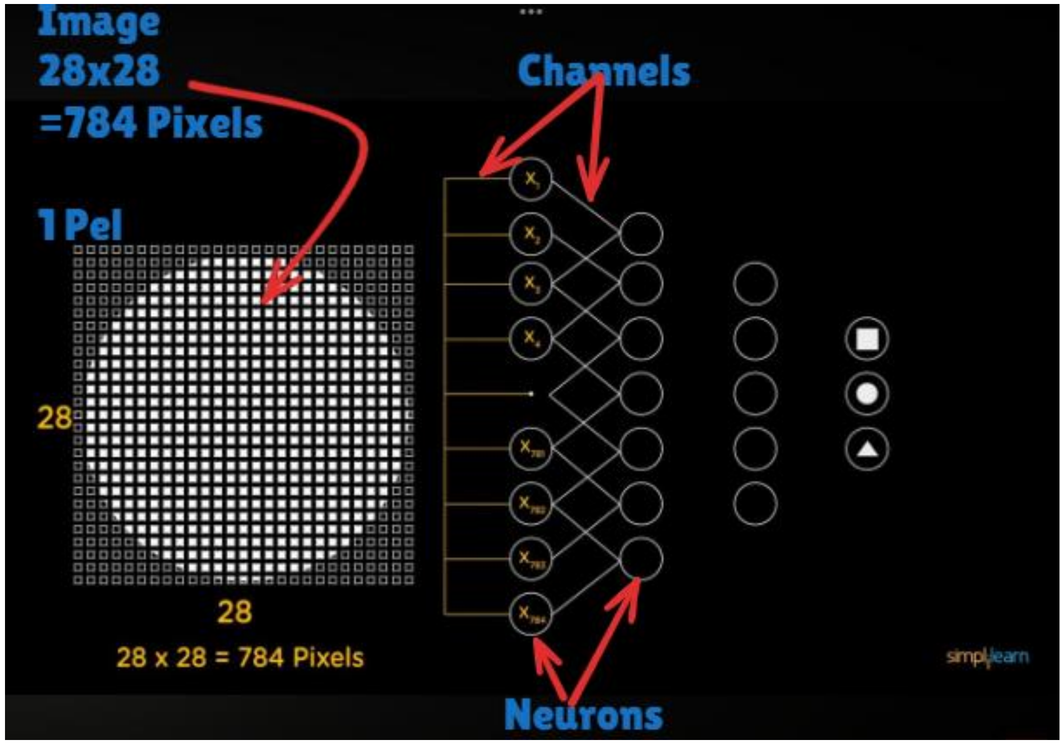

Output
Layer
simplilearn
Input Layer


simplilearn
Hidden Layer: More Math Computatins
(Probablity)

1. Given 28 × 28 Image → 784 Inputs

What you see:

- A **28 × 28 grid**
- Total pixels = **28 × 28 = 784**
- Each small square = **1 pixel**



Meaning:

- This is a **digit image** (like MNIST dataset)
- Each pixel becomes **one input feature**
- Pixel value is usually between **0 and 1** (after normalization)

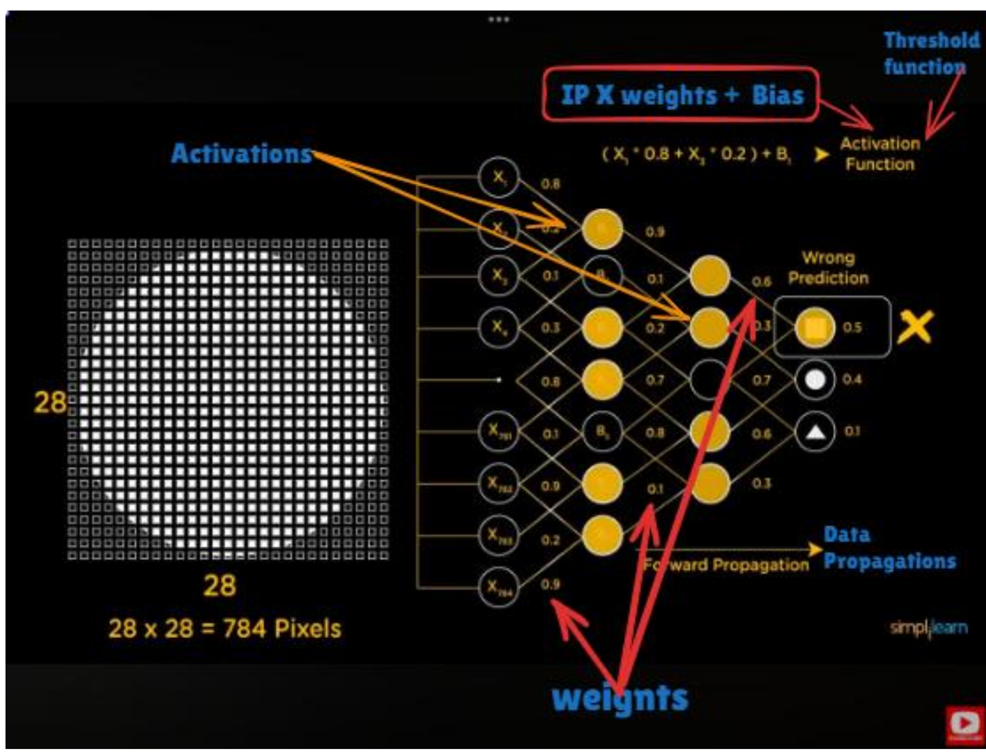
Mathematically: $X = [x_1, x_2, x_3, \dots, x_{784}]$

👉 These 784 values are fed into the **input layer** of the neural network.

2. Input Layer → Hidden Layers (Fully Connected ANN)

What you see:

- Nodes labeled $X_1, X_2, \dots, X_{784}$
- Multiple hidden layers with neurons
- Lines with numbers (weights)



Meaning:

- This is a **Fully Connected Neural Network (FCNN / MLP)**
- Every neuron in one layer connects to all neurons in the next layer
- Each connection has a **weight**

Mathematical operation inside **each neuron**:

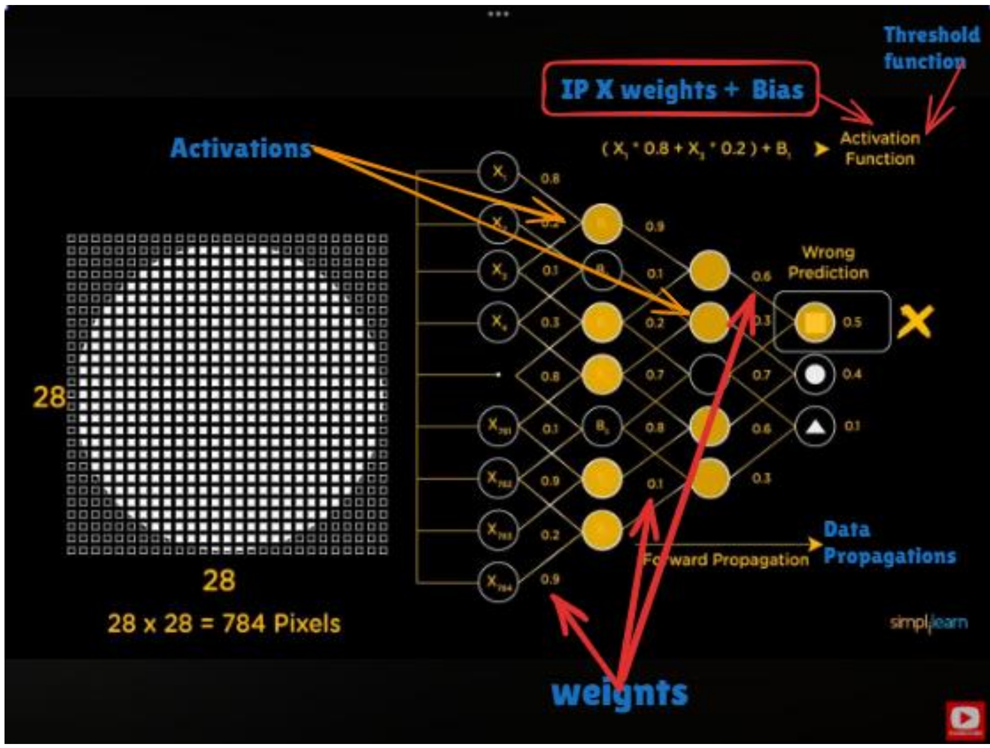
$$z_j = \sum_{i=1}^{n} w_{ij}\, x_i + b_j$$

$$a_j = f(z_j)$$

Where: x_i= input value w_{ij}= weight b_j= bias $f(\cdot)$= activation function (ReLU / Sigmoid)

👉 This step is called **Forward Propagation**.

3. Activation Function (Yellow Highlight in Images)



What you see:

- Text like **Activation Function**
- Bright (yellow/golden) neurons

Meaning:

- Activation decides **whether a neuron fires**
- Introduces **non-linearity**
- Without activation, the network becomes only linear

Common functions:

- ReLU: $f(z) = \max(0, z)$
- Sigmoid: $f(z) = \frac{1}{1+e^{-z}}$

👉 This allows the network to learn **complex patterns like curves, edges, faces**.

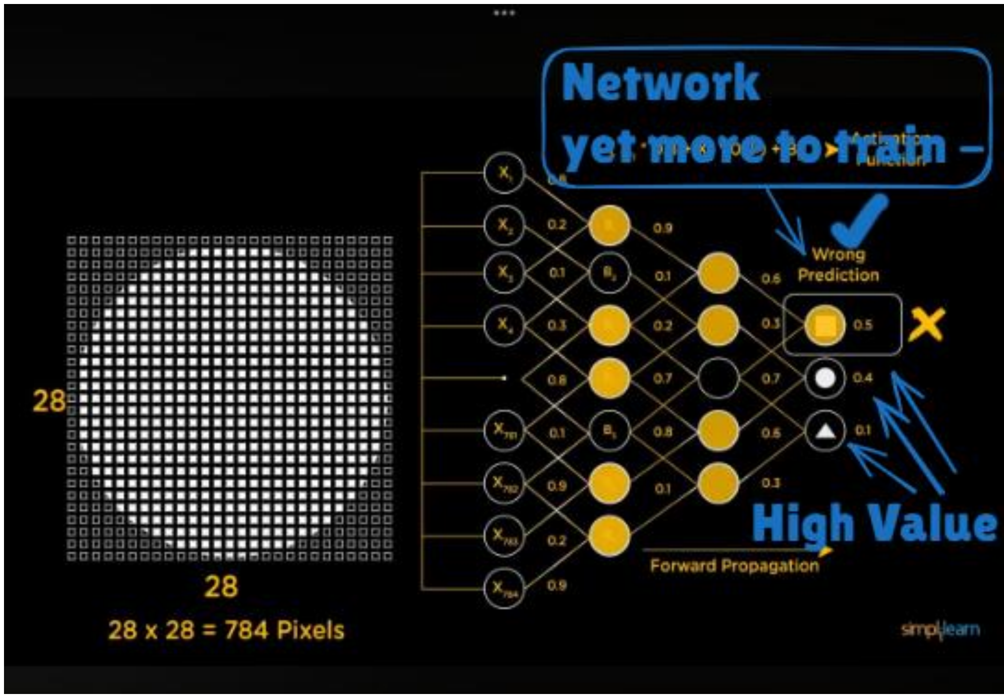
---

4. Output Layer (Shapes / Symbols on Right)

What you see:

- Shapes (square, circle, triangle)

- Values like **0.6, 0.4, 0.1**
- "Actual Output" and "Error" table



Meaning:

- Each output neuron represents **one class**
    - e.g., digit 0–9
- Output values are **probabilities**
- **Highest value = predicted class**

Example:

$$\text{Output} = [0.1, 0.2, 0.6, 0.1]$$

→ Predicted class = **index with 0.6**

👉 This is **classification**.

5. Error Calculation (Loss Function)

What you see:

- "Actual Output"
- "Error" (+ or − values)

Meaning:

- Compare predicted output with true label
- Error tells **how wrong the network is**

Simple error: $$\textbf{Error} = \textbf{Target} - \textbf{Prediction}$$

Or loss (example): $$L = (y - \hat{y})^2$$

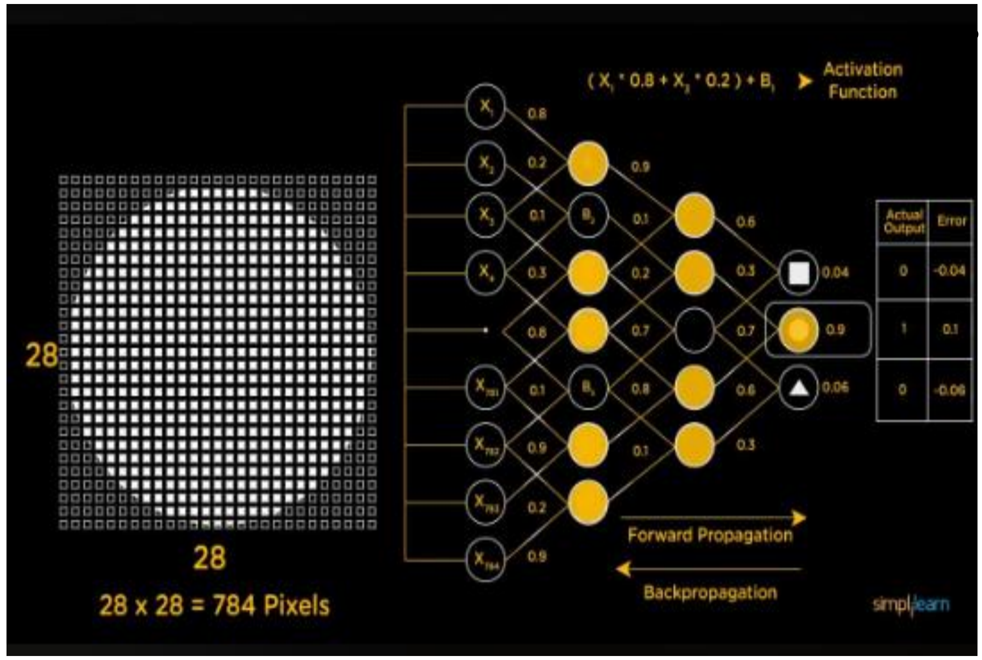
👉 Learning starts **only after error is known**.

---

6. Backpropagation (Right → Left Arrows)

What you see:

- Arrow labeled **Backpropagation**
- Weight values changing in different images



Meaning:

- Error is sent **backwards**
- Weights are adjusted to reduce error
- Uses **gradient descent**

Weight update rule:

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

Where: η= learning rate

👉 This is how the network **learns from mistakes**.

---

7. Repeated Frames = Learning Over Time

What you see:

- Same network shown multiple times
- Different output values each time

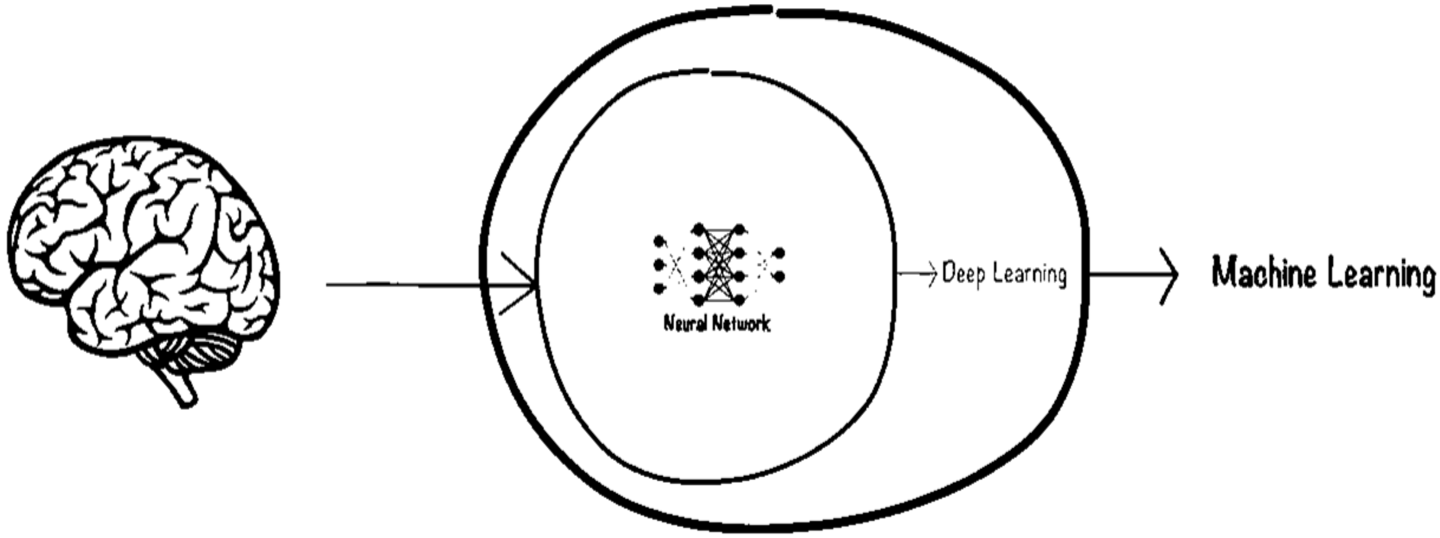
Meaning:

- Each frame = **one training iteration (epoch)**
- Weights gradually improve
- Error becomes smaller

👉 Learning is **iterative**, not instant.

8. Neural Network ⊂ Deep Learning ⊂ Machine Learning Diagram

- Brain → Neural Network → Deep Learning → Machine Learning



- **Neural Network**: basic model
- **Deep Learning**: neural networks with many layers
- **Machine Learning**: broader field (includes KNN, SVM, etc.)

Relationship:	Neural Networks ⊂ Deep Learning ⊂ Machine Learning

1 Facial Recognition (Top Image)

What it shows

- A person's face scanned by a phone
- Facial landmarks (eyes, nose, jawline)
- Output like age/identity

How it works

1. **Input**: Face image (pixels)
2. **Feature learning**: Neural network learns facial patterns (edges → eyes → full face)
3. **Comparison**: Extracted features are matched with stored features

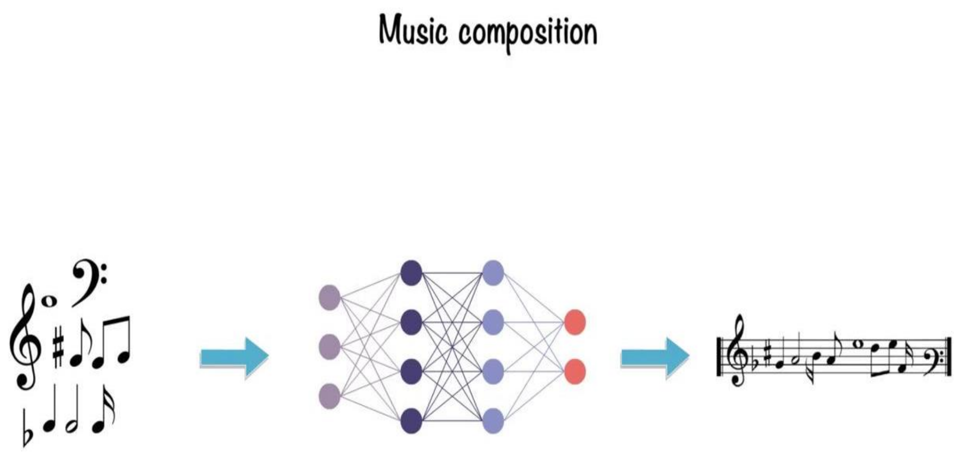
4. **Output**: Identity / age / verification result

Key idea

Neural networks learn **unique facial patterns** and recognize people even under different lighting or angles.

3 Music Composition using Neural Networks

What it shows



- Musical notes as input
- Neural network in the middle
- New musical notes as output

How it works

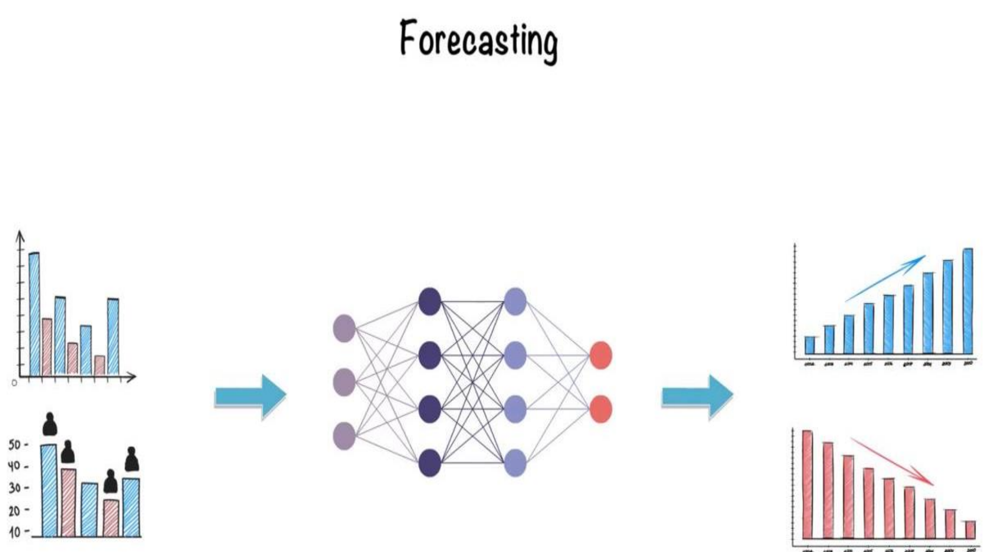
1. **Input**: Existing music sequences (notes, rhythm)
2. **Learning**: Network learns musical patterns and transitions
3. **Generation**: Predicts the next note or composes new music

Key idea

Neural networks can **create new data**, not just classify existing data.

4 Forecasting (Prediction of Future Trends)

What it shows



- Past data (charts, bars)
- Neural network processing
- Future trends (increasing/decreasing graphs)

How it works

1. **Input**: Historical numerical data (sales, population, demand)

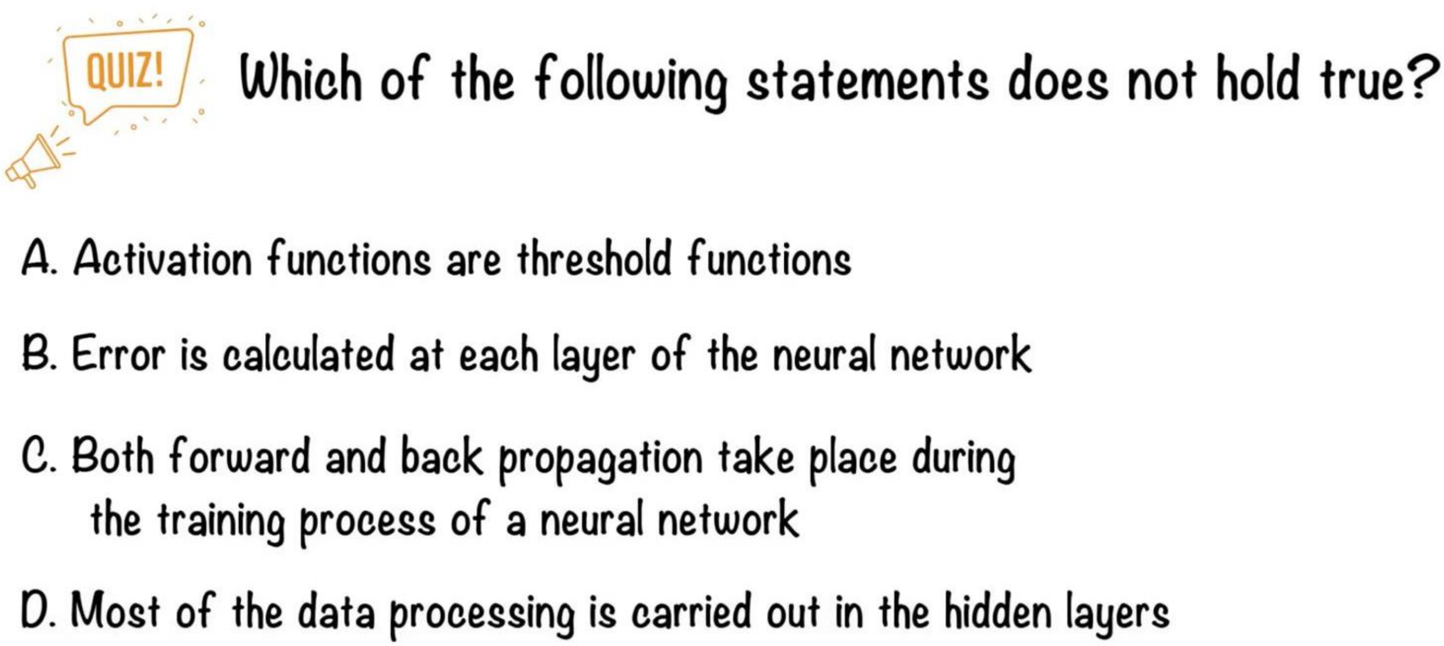
2. **Learning**: Network learns trends and seasonality
3. **Output**: Future predictions

Applications

- Sales forecasting
- Stock trends
- Weather prediction

Key idea

Neural networks learn **patterns over time** to predict the future.



Question
"Which of the following statements does not hold true?"

Options include:

- Activation functions are threshold functions
- Error is calculated at each layer
- Forward & backpropagation happen during training
- Most processing happens in hidden layers

Conceptual interpretation

- Activation functions are **not only threshold** (ReLU, Sigmoid, Tanh exist)
- **Error is mainly calculated at the output layer**, then propagated backward
- Forward + Backpropagation **do occur during training**
- Hidden layers perform **most of the feature extraction**

Purpose of image: To test conceptual understanding of **ANN training mechanics**.

4.5.2 Mathematical Model of ANN

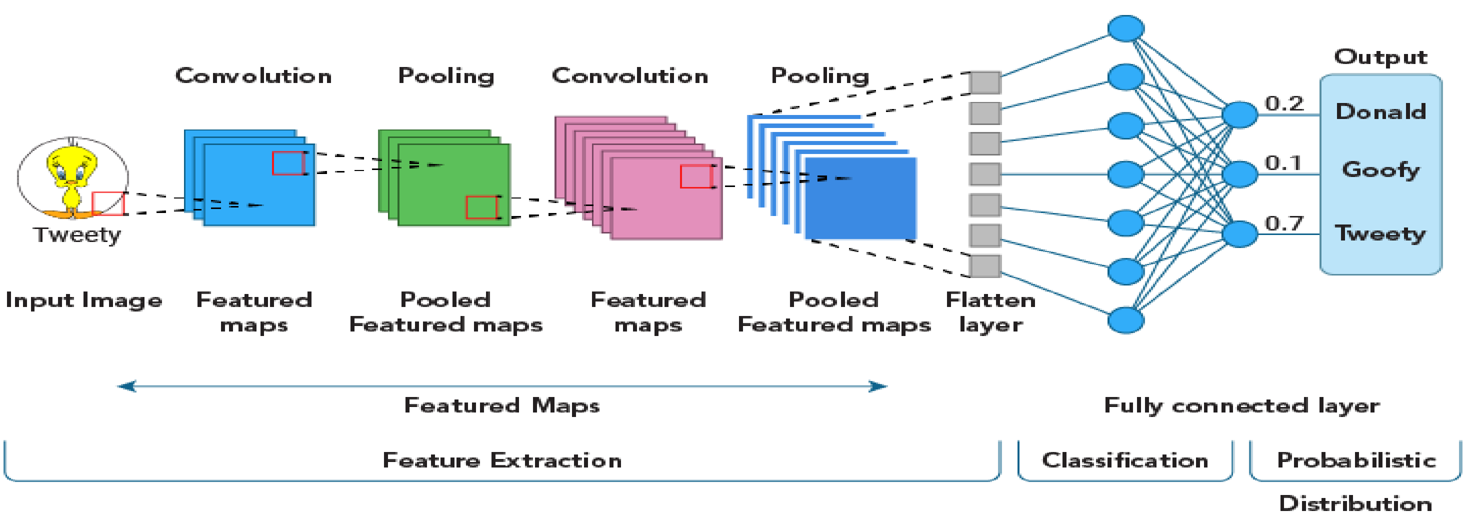
https://www.mdpi.com/2673-9984/6/1/4

https://www.jetir.org/papers/JETIR2103402.pdf

A mathematical model of an Artificial Neural Network (ANN) expresses how input variables are mathematically transformed through layers of neurons to produce an output prediction.

In the MDPI study, ANN was used to predict **nickel removal efficiency** based on input parameters (initial concentration, adsorbent dosage, pH).

An ANN **captures complex non-linear relationships** between inputs and outputs by learning weights and biases through training data.



The **mathematical model of ANN** explains how an artificial neuron receives inputs, processes them using weights and bias, applies an activation function, and produces an output. This model is inspired by biological neurons but is completely mathematical.

1. Components of the Mathematical Model

An artificial neuron consists of:

1. **Inputs**

$$x_1, x_2, x_3, \ldots, x_n$$

2. **Weights**

$$w_1, w_2, w_3, \ldots, w_n$$

3. **Bias**

$$b$$

4. **Summation Function**
5. **Activation Function**

---

2. Mathematical Representation

Step 1: Weighted Sum

$$z = \sum_{i=1}^{n} w_i x_i + b$$

This equation combines all inputs according to their importance (weights).

Step 2: Activation Function

$$y = f(z)$$

The activation function decides whether the neuron should fire.

---

3. Common Activation Functions

(a) Linear Function

$$f(z) = z$$

(b) Step (Threshold) Function

$$f(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

(c) Sigmoid Function

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

(d) ReLU Function

$$\boldsymbol{f(z) = \max(0, z)}$$

---

4. Multi-Layer ANN Mathematical Model

For a network with one hidden layer:

Hidden Layer

$$h_j = f\left(\sum_i w_{ij}\, x_i + b_j\right)$$

Output Layer

$$y_k = g\left(\sum_j v_{jk}\, h_j + b_k\right)$$

Where:

- f = hidden layer activation
- g = output layer activation

---

5. Learning in ANN (Error Minimization)

Error Function (Mean Squared Error)

$$E = \frac{1}{2}(t - y)^2$$

Where:

- t = target output
- y = predicted output

Weight Update Rule

$$w_{new} = w_{old} - \eta \frac{\partial E}{\partial w}$$

Where:

- η= learning rate

This process is called **backpropagation**.

---

6. Numerical Example

Given:

- $x_1 = 1,$ $x_2 = 2$
- $w_1 = 0.4,$ $w_2 = 0.6$
- $b = 0.2$
- Activation: **Sigmoid**

Step 1:

$$z = (1 \times 0.4) + (2 \times 0.6) + 0.2 = 1.8$$

Step 2:

$$y = \frac{1}{1 + e^{-1.8}} \approx 0.858$$

---

7. Importance of Mathematical Model of ANN

- Explains how ANN processes information
- Forms the basis of training algorithms
- Enables prediction and classification
- Helps analyze network behavior mathematically

4.5.5 Application of Artificial Neural Networks, Learning by Training ANN, Perceptron Learning, Back-propagation Learning

A. Applications of Artificial Neural Networks (ANN)

Artificial Neural Networks are widely used because they can **learn patterns**, **handle non-linear data**, and **adapt from experience**.

Major Applications

1. **Pattern Recognition**
   - Face recognition
   - Handwritten digit recognition (MNIST)
   - Speech recognition
2. **Prediction and Forecasting**
   - Stock market prediction
   - Weather forecasting
   - Sales and demand forecasting
3. **Classification**
   - Spam vs non-spam email
   - Disease diagnosis (positive / negative)
   - Credit approval systems
4. **Image and Signal Processing**
   - Image enhancement
   - Object detection
   - Noise removal
5. **Natural Language Processing**
   - Language translation
   - Chatbots
   - Text summarization
6. **Control Systems**

- Robotics
- Autonomous vehicles
- Industrial process control

📌 **Key Point:**
ANNs are used where **rule-based programming is difficult**.

B. Learning by Training ANN

What is Training?

Training is the process by which an ANN **learns from data** by adjusting its **weights and biases** to reduce error.

Training Process Steps

1. Provide **input data**
2. Perform **forward propagation**
3. Generate **output**
4. Compare with **target output**
5. Calculate **error**
6. Adjust weights using a learning algorithm
7. Repeat until error is minimized

Mathematically:

$$\text{Error} = \text{Target} - \text{Output}$$

📌 **Learning happens iteratively** over many epochs.

C. Perceptron Learning

What is a Perceptron?

A perceptron is the **simplest neural network model** with:

- Single neuron
- Binary output (0 or 1)

Perceptron Model Equation

$$y = f(\sum w_i x_i + b)$$

Where:

- f= step (threshold) function

Learning Rule (Perceptron Learning Rule)

$$w_{new} = w_{old} + \eta(t - y)x$$

Where:

- η= learning rate
- t= target output
- y= predicted output

Characteristics

- Works only for **linearly separable problems**
- Cannot solve XOR problem
- Fast and simple

📌 **Example:** AND, OR logic gates

D. Back-Propagation Learning

What is Back-Propagation?

Back-propagation is a **supervised learning algorithm** used to train **multi-layer neural networks**.

It minimizes error by **propagating the error backward** from output layer to hidden layers.

Back-Propagation Steps

1. Initialize weights randomly
2. Perform **forward propagation**
3. Calculate **error at output layer**
4. Propagate error backward
5. Update weights using gradient descent
6. Repeat until convergence

Error Function (Example: Mean Squared Error)

$$E = \frac{1}{2}(t - y)^2$$

Weight Update Rule

$$w_{new} = w_{old} - \eta \frac{\partial E}{\partial w}$$

Advantages

- Can learn **complex non-linear patterns**
- Used in deep learning
- High accuracy

Limitations

- Slow training
- Requires large datasets
- May get stuck in local minima

E. Comparison: Perceptron vs Back-Propagation

Feature	Perceptron	Back-Propagation
Network type	Single layer	Multi-layer
Output	Binary	Continuous
Problem type	Linear	Non-linear
Learning	Simple	Gradient-based
XOR problem	❌ No	✅ Yes