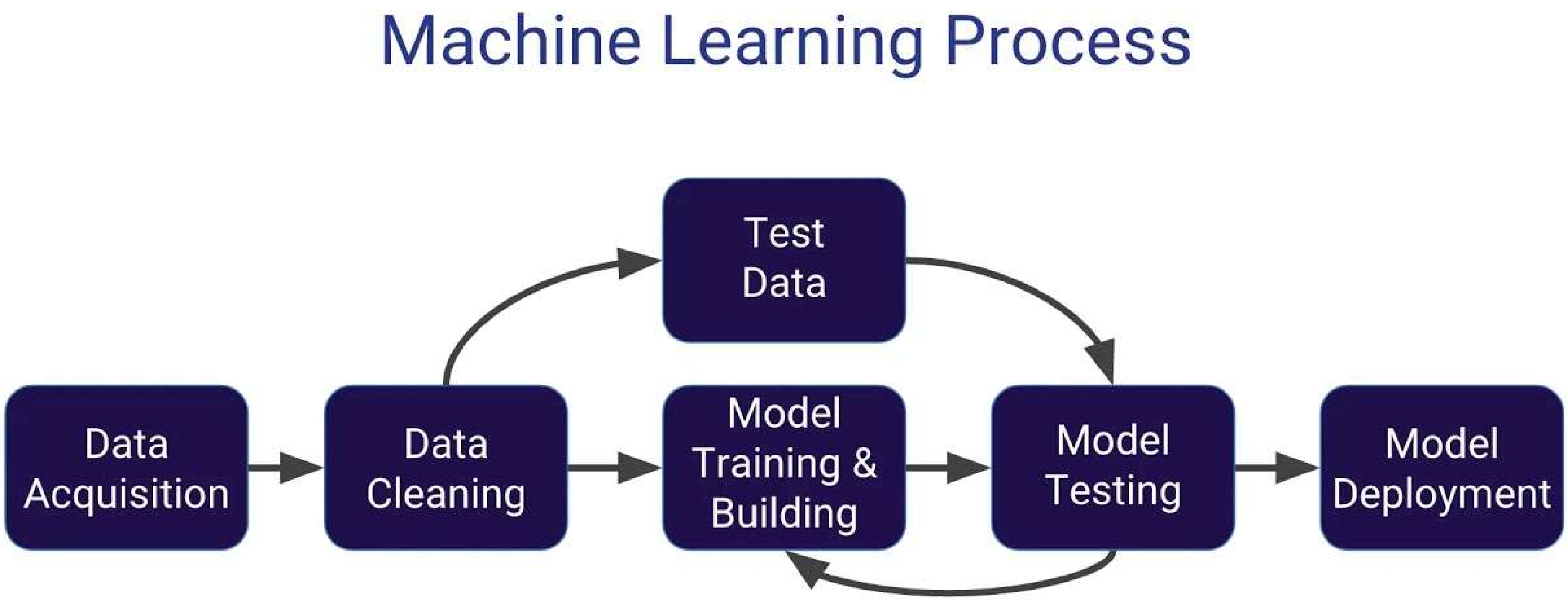
1.2

- Basic Components and Key Terminology of Machine Learning,
- Data types
- Training and testing, activation functions,
- Overfitting and under fitting
- Bias- variance Tradeoff.

Basic Components and Key Terminology of Machine Learning

[redacted]

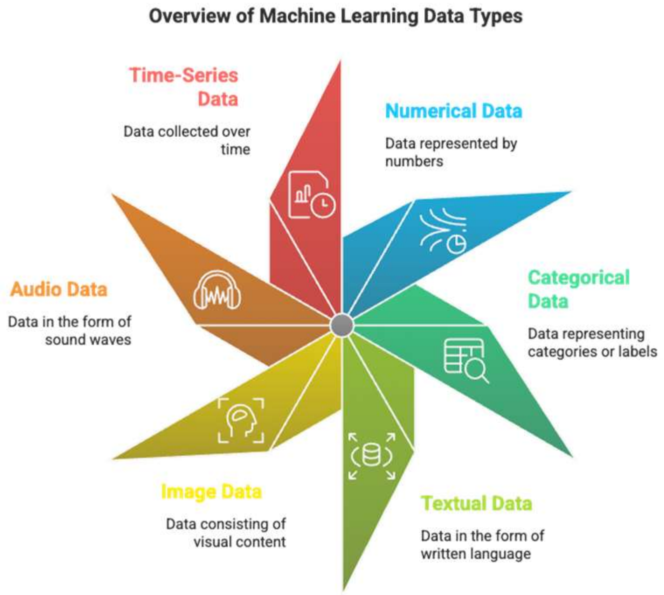


[redacted]

Key Terminology in Machine Learning

Term	**Definition**
Supervised Learning	Learning with labeled data (e.g., classification, regression).
Unsupervised Learning	Learning patterns from unlabeled data (e.g., clustering, dimensionality reduction).
Reinforcement Learning	Learning via rewards and penalties from interactions with an environment.
Overfitting	Model performs well on training data but poorly on unseen data.
Underfitting	Model is too simple to capture the patterns in data.
Generalization	Model's ability to perform well on new, unseen data.
Cross-Validation	A method to evaluate model performance by splitting data into multiple subsets.
Hyperparameters	Settings that define the model structure or how it's trained (e.g., learning rate, number of layers).
Epoch	One complete pass through the entire training dataset.
Bias	Error due to overly simplistic assumptions in the model.
Variance	Error due to model sensitivity to small fluctuations in the training set.
Confusion Matrix	Table used to evaluate the performance of a classification algorithm.
Accuracy, Precision, Recall, F1 Score	Metrics used to evaluate model performance.

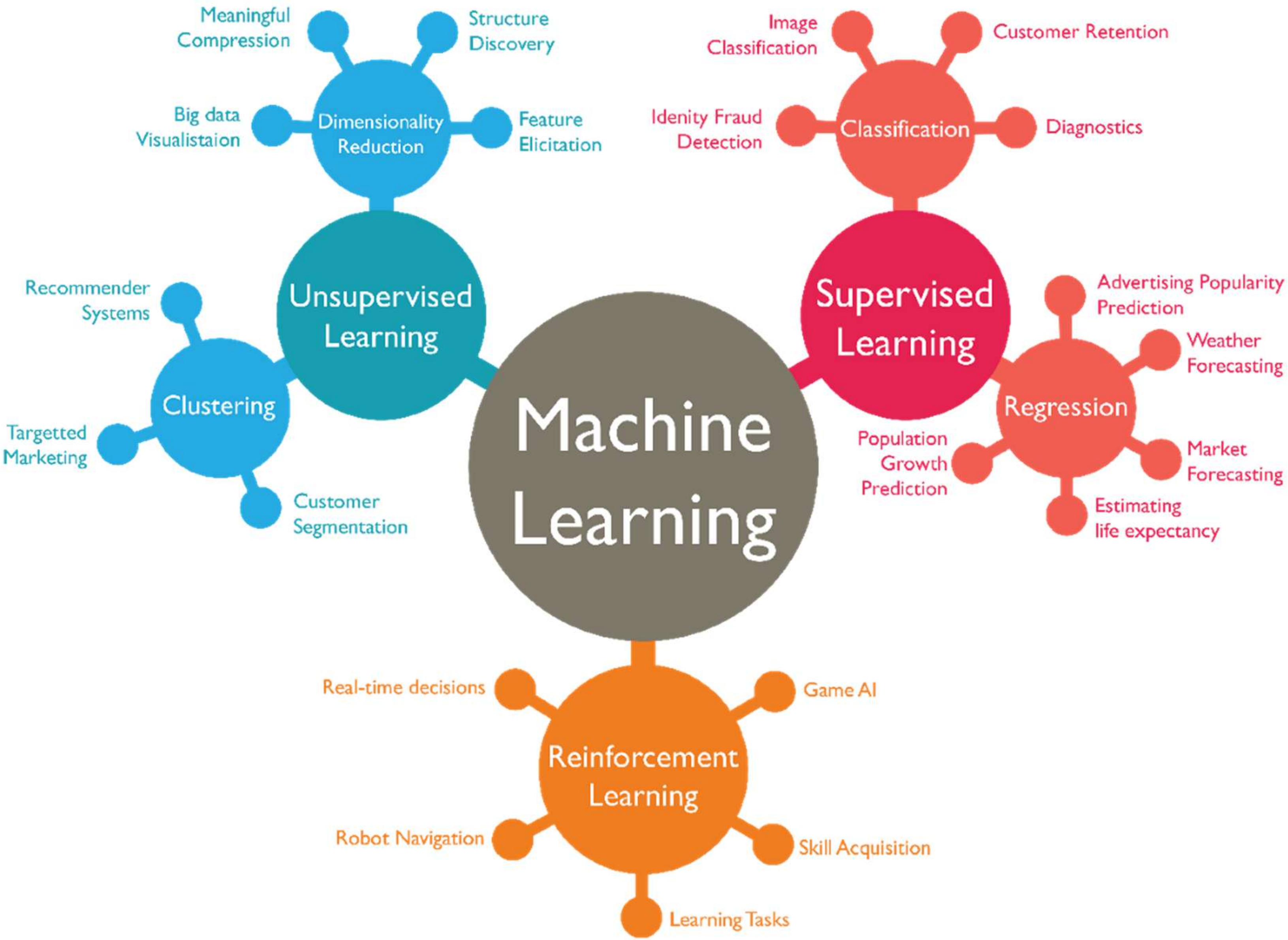
Data Types in ML

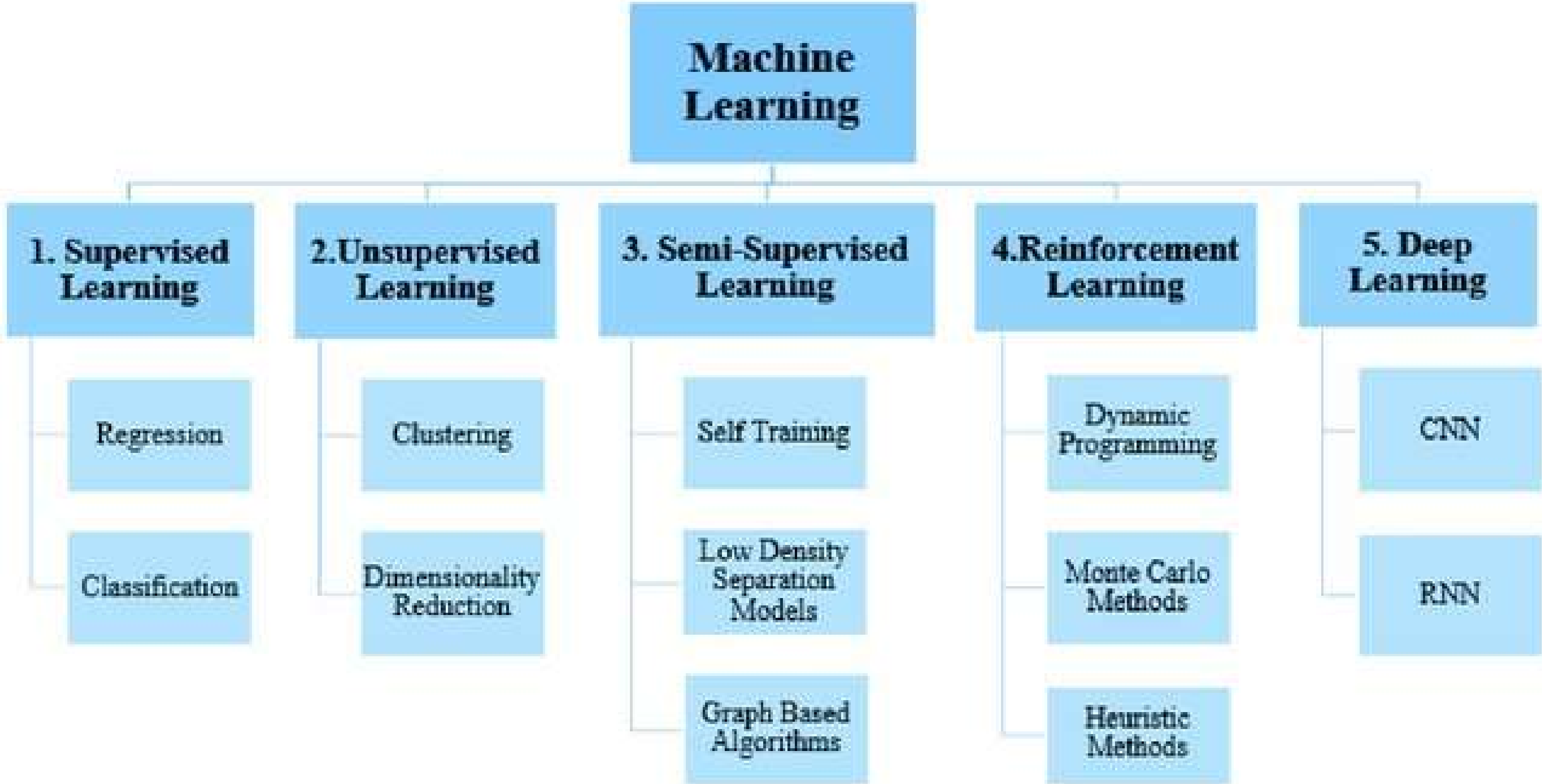


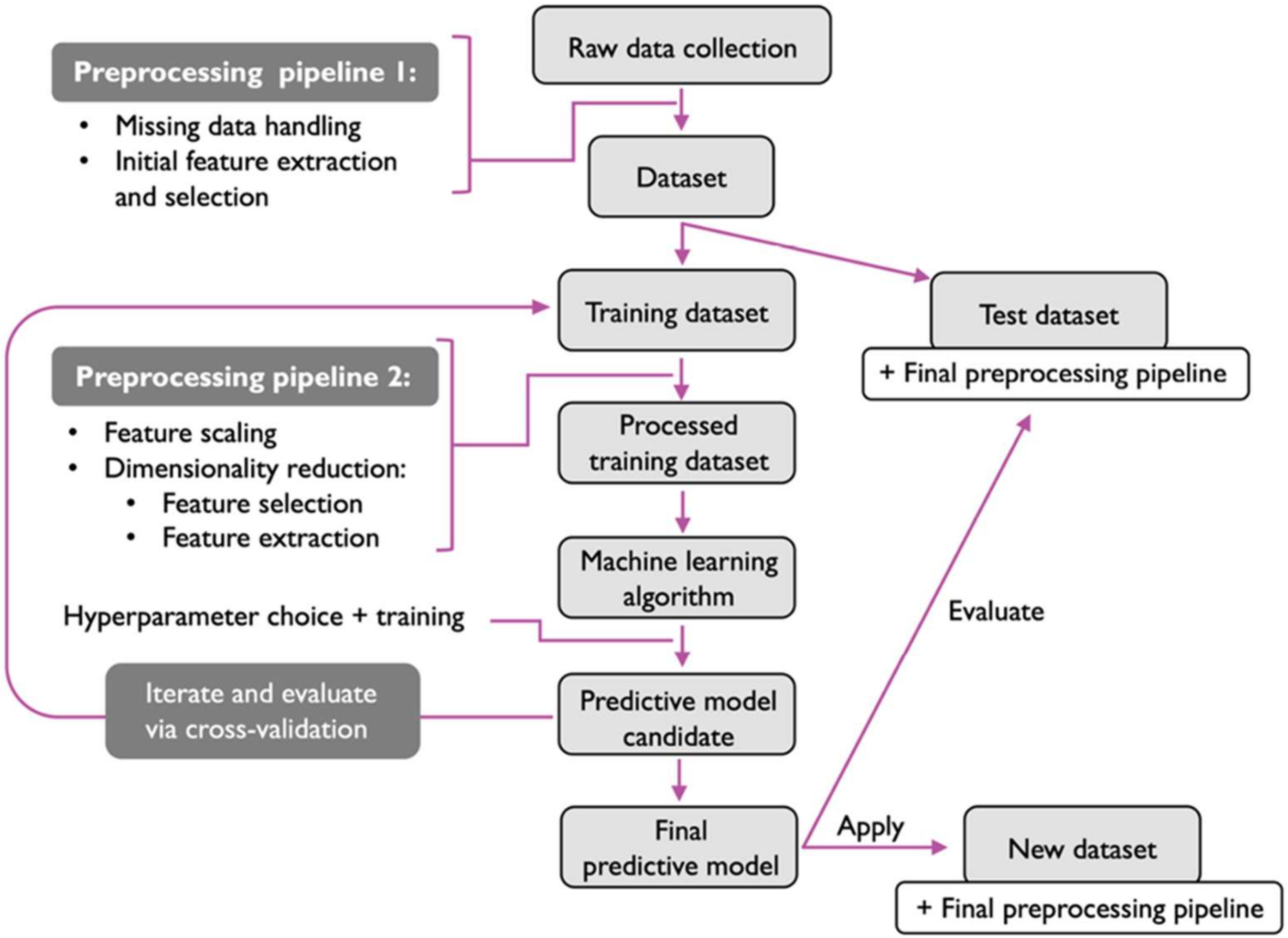
Types of Data in Machine Learning

1. **Structured Data**
    - Data that is organized into rows and columns and is stored in databases or spreadsheets. (e.g., spreadsheets, databases).
    - It follows a fixed schema, making it easy to query and analyze.
    - Examples: Customer databases, sales transactions, student grades, Excel sheets Examples: Sales records, sensor data, and user profiles.
2. **Unstructured Data**
    - Data that does not follow a predefined data model or structure.
    - It is harder to analyze and process directly but contains valuable insights when processed with the right tools.
    - Examples: Emails, social media posts, audio files, video recordings, images, and text documents.
3. **Semi-Structured Data**
    - Data that does not reside in a relational database but still has some structure through tags or markers.
    - It combines elements of both structured and unstructured data.
    - Examples: JSON files, XML documents, HTML pages, log files.
4. **Quantitative (Numerical) Data**
    - Expressed in numbers; measurable.
    - Types:
        - **Discrete**: Countable (e.g., number of clicks).
        - **Continuous**: Measurable (e.g., temperature, price).
5. **Qualitative (Categorical) Data**
    - Descriptive; represents categories.
    - Types:
        - **Nominal**: No order (e.g., gender, colors).
        - **Ordinal**: Ordered (e.g., education level, rating scale).
6. **Time Series Data**
    - Data collected at different time points.
    - Example: Stock prices, weather logs.
7. **Text Data**
    - Unstructured data in natural language.
    - Example: Tweets, reviews, news articles.
8. **Image & Video Data**
    - Pixels forming visual content.
    - Used in computer vision tasks like classification, detection, and segmentation.

Type of Machine learning

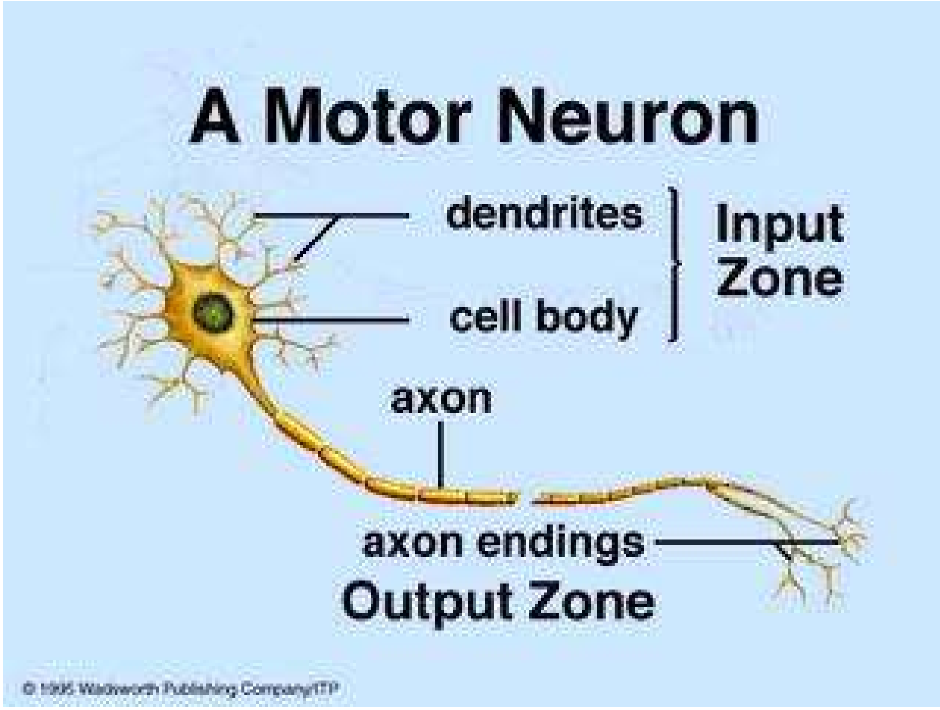



Machine Learning
1. Supervised Learning
2.Unsupervised Learning
3. Semi-Supervised Learning
4.Reinforcement Learning
5. Deep Learning
Regression
Classification
Clustering
Dimensionality Reduction
Self Training
Low Density Separation Models
Graph Based Algorithms
Dynamic Programming
Monte Carlo Methods
Heuristic Methods
CNN
RNN


Raw data collection
Preprocessing pipeline 1:
Missing data handling
Initial feature extraction and selection
Dataset
Training dataset
Test dataset
+ Final preprocessing pipeline
Preprocessing pipeline 2:
Feature scaling
Dimensionality reduction:
Feature selection
Feature extraction
Processed training dataset
Machine learning algorithm
Hyperparameter choice + training
Evaluate
Iterate and evaluate via cross-validation
Predictive model candidate
Final predictive model
Apply
New dataset
+ Final preprocessing pipeline

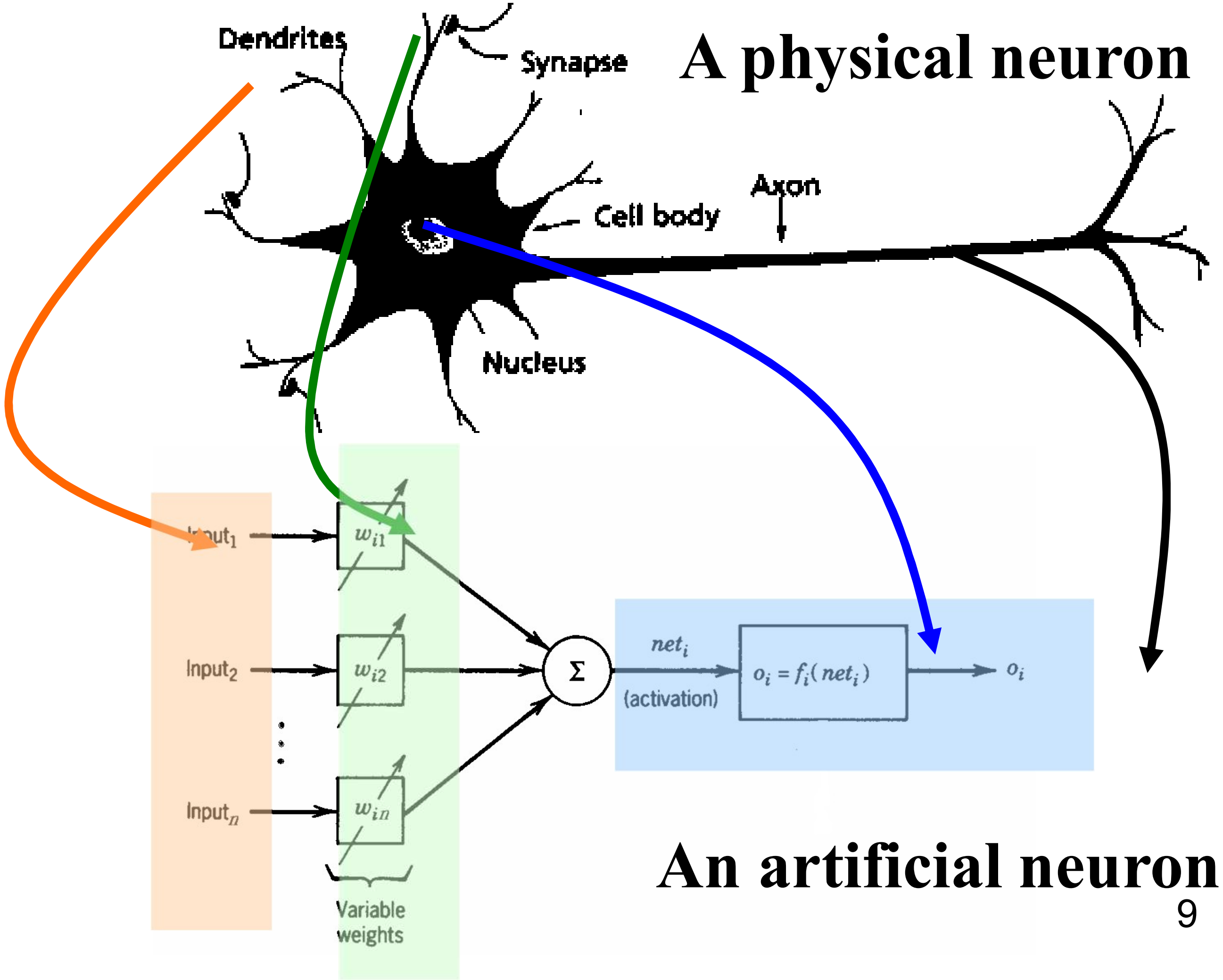
Biological Neural Networks

- A biological neuron has three types of main components; <u>dendrites</u>, <u>soma</u> (or cell body) and <u>axon</u>.
- Dendrites receives signals from other neurons.
- The soma, sums the incoming signals. When sufficient input is received, the cell fires; that is it transmit a signal over its axon to other cells.

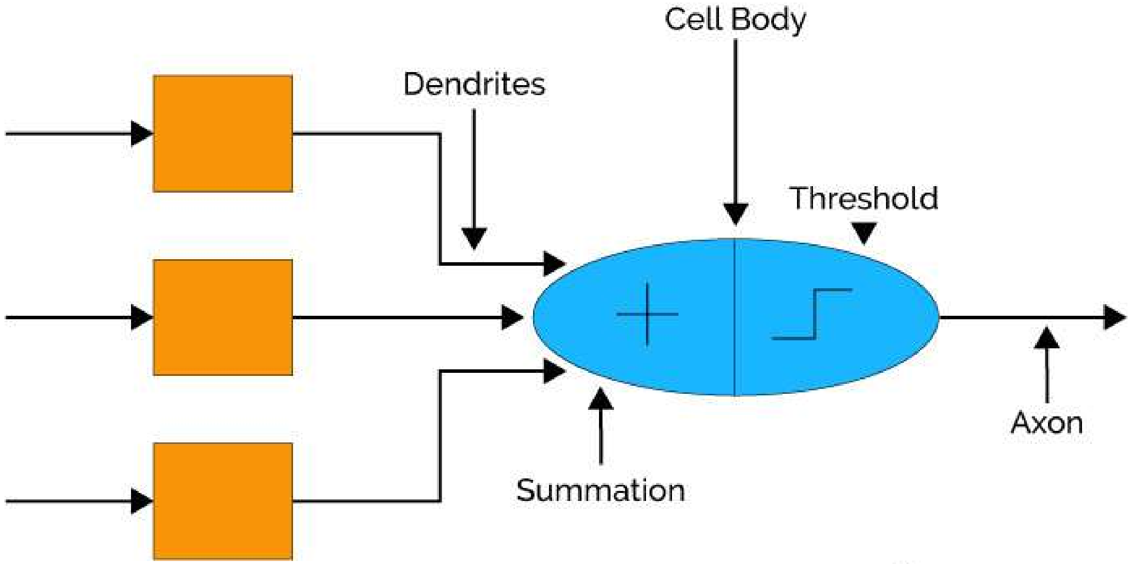


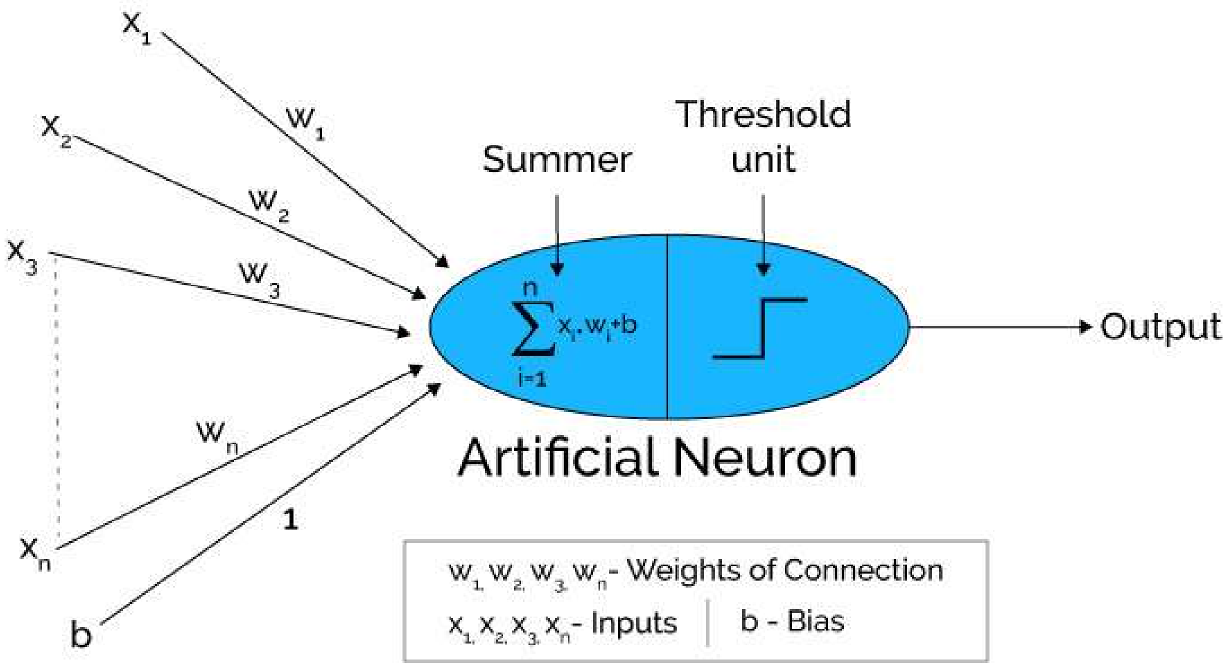
Artificial Neurons

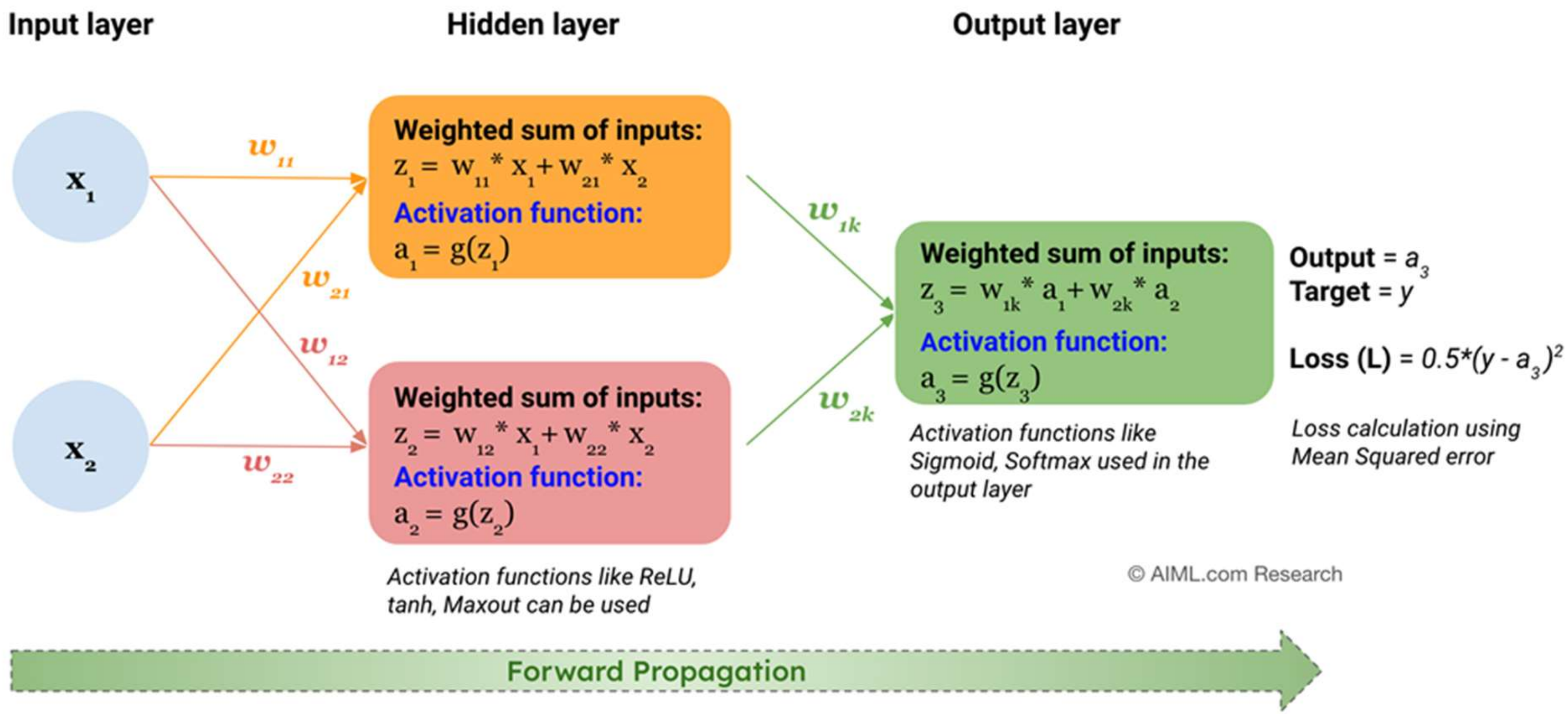
- From experience: examples / training data
- Strength of connection between the neurons is stored as a weight-value for the specific connection.
- Learning the solution to a problem = changing the connection weights



Brain Vs Computer







Title: Role of Activation Function in a Neural Network

Activation functions serve two main purposes:

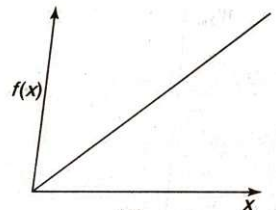
1. **Introduce Non-linearity:** Without non-linearity, a neural network would behave like a linear model, no matter how deep it is. Activation functions allow the network to learn and represent complex, non-linear mappings between inputs and outputs.
2. **Control Neuron Activation:** Activation functions control the firing behavior of neurons. Depending on the activation function's output, a neuron might become activated (output a non-zero value) or remain inactive (output zero).

3. Activation Functions

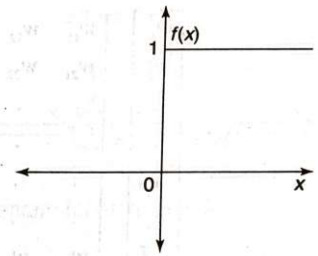
1. *Identity function:* It is a linear function and can be defined as

$$f(x) = x \quad \text{for all } x$$

The output here remains the same as input. The input layer uses the identity activation function.



(A)



(B)

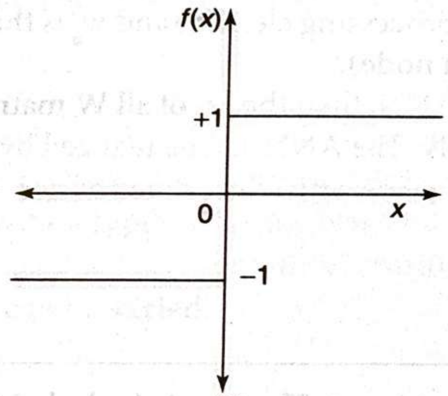
2. *Binary step function:* This function can be defined as

$$f(x) = \begin{cases} 1 & \text{if} \quad x \geq \theta \\ 0 & \text{if} \quad x < \theta \end{cases}$$

where θ represents the threshold value. This function is most widely used in single-layer nets to convert the net input to an output that is a binary (1 or 0).

3. *Bipolar step function:* This function can be defined as

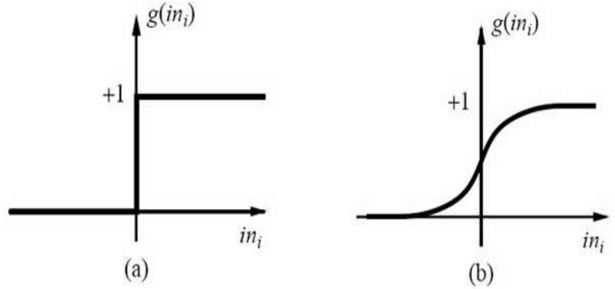
$$f(x) = \begin{cases} 1 & \text{if} \quad x \geq \theta \\ -1 & \text{if} \quad x < \theta \end{cases}$$



(C)

3. Activation Functions

Sigmoidal Functions



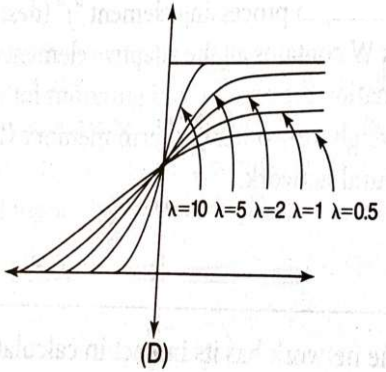
step function **sigmoid function**

- *Bipolar sigmoid function:* This function is defined as

$$f(x)=\frac{2}{1+e^{-\lambda x}}-1=\frac{1-e^{-\lambda x}}{1+e^{-\lambda x}}$$

where λ is the steepness parameter and the sigmoid function range is between −1 and +1. The derivative of tl function can be

$$f'(x)=\frac{\lambda}{2}[1+f(x)][1-f(x)]$$

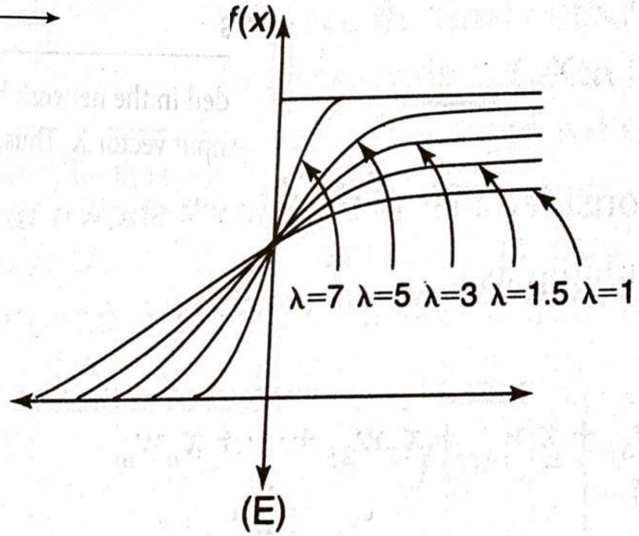


The bipolar sigmoidal function is closely related to hyperbolic tangent function, which is written as

$$h(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}=\frac{1-e^{-2x}}{1+e^{-2x}}$$

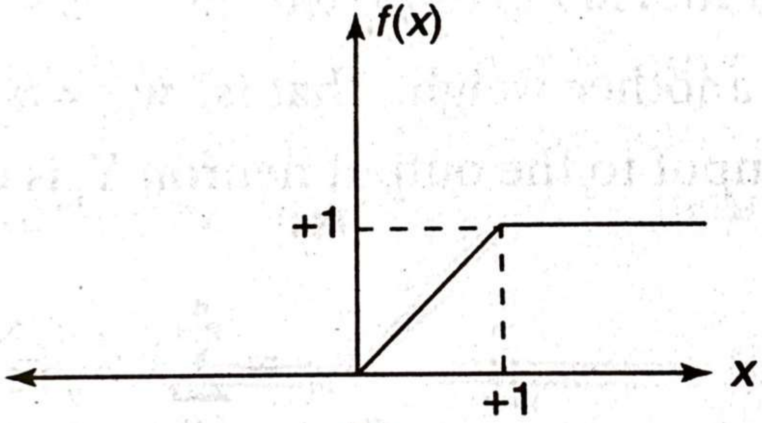
The derivative of the hyperbolic tangent function is

$$h'(x)=[1+h(x)][1-h(x)]$$



If the network uses a binary data, it is better to convert it to bipolar form and use the bipolar sigmoidal activation function or hyperbolic tangent function.

3. Activation Functions



(F)

5. *Ramp function*: The ramp function is defined as

$$f(x)=\begin{cases}1 & \text{if} \quad x>1\\ x & \text{if} \quad 0\leq x\leq 1\\ 0 & \text{if} \quad x<0\end{cases}$$

...f all the activation functions are shown in Figure 2-15(A)–(F).

3. Activation Functions

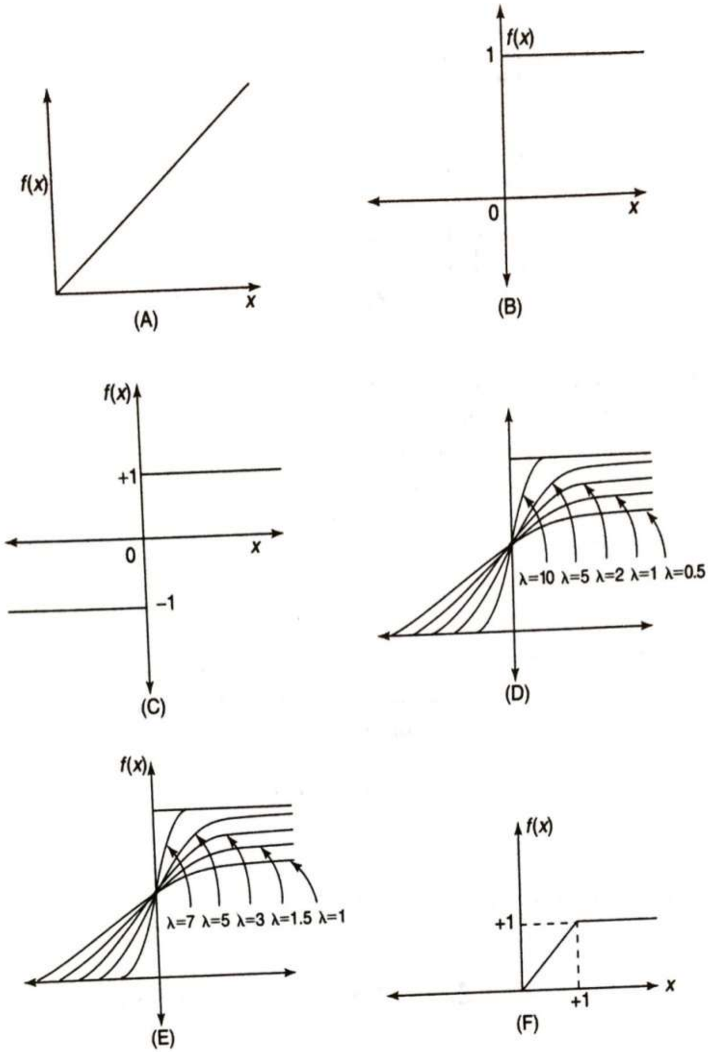


Figure 2-15 Depiction of activation functions: (A) identity function; (B) binary step function; (C) bipolar step function; (D) binary sigmoidal function; (E) bipolar sigmoidal function; (F) ramp function.

- Identity

  $f(x) = x$

- Binary step

  $f(x) = 1$ if $x >= \theta$

  $f(x) = 0$ otherwise

- Binary sigmoid

  $f(x) = 1 / (1 + e^{-\sigma x})$

- Bipolar sigmoid

  $f(x) = -1 + 2 / (1 + e^{-\sigma x})$

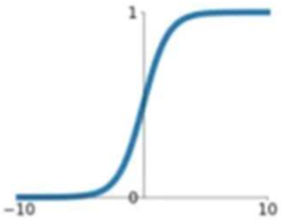
- Hyperbolic tangent

  $f(x) = (e^{x} – e^{-x}) / (e^{x} + e^{-x})$

Activation Functions

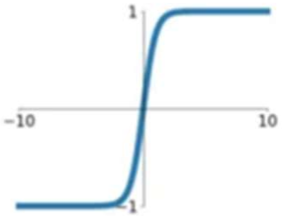
Sigmoid

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



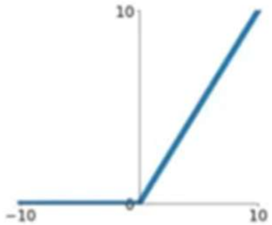
tanh

$\tanh(x)$



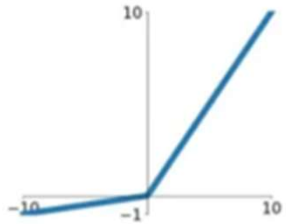
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

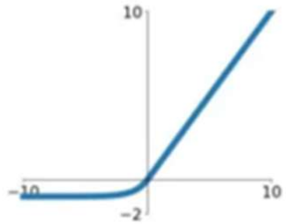


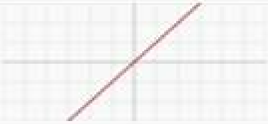
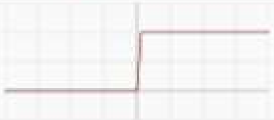
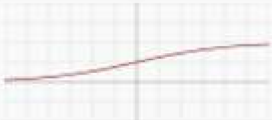
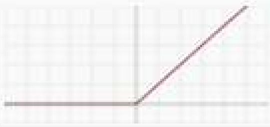
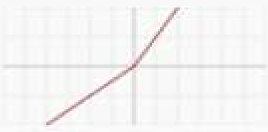
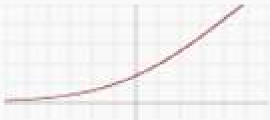
Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Name	Plot	Equation	Derivative
Identity		$f(x)=x$	$f'(x)=1$
Binary step		$f(x)=\begin{cases}0 & \text{for} \quad x<0\\ 1 & \text{for} \quad x\geq 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for} \quad x\neq 0\\ ? & \text{for} \quad x=0\end{cases}$
Logistic (a.k.a Soft step)		$f(x)=\frac{1}{1+e^{-x}}$	$f'(x)=f(x)(1-f(x))$
TanH		$f(x)=\tanh(x)=\frac{2}{1+e^{-2x}}-1$	$f'(x)=1-f(x)^2$
ArcTan		$f(x)=\tan^{-1}(x)$	$f'(x)=\frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x)=\begin{cases}0 & \text{for} \quad x<0\\ x & \text{for} \quad x\geq 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for} \quad x<0\\ 1 & \text{for} \quad x\geq 0\end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x)=\begin{cases}\alpha x & \text{for} \quad x<0\\ x & \text{for} \quad x\geq 0\end{cases}$	$f'(x)=\begin{cases}\alpha & \text{for} \quad x<0\\ 1 & \text{for} \quad x\geq 0\end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x)=\begin{cases}\alpha(e^x-1) & \text{for} \quad x<0\\ x & \text{for} \quad x\geq 0\end{cases}$	$f'(x)=\begin{cases}f(x)+\alpha & \text{for} \quad x<0\\ 1 & \text{for} \quad x\geq 0\end{cases}$
SoftPlus		$f(x)=\log_e(1+e^x)$	$f'(x)=\frac{1}{1+e^{-x}}$

Used in layer	Activation Function	Details	Pros	Cons
Hidden / Output	**Sigmoid**	$\sigma(x) = 1/(1+e^{-x})$ Output range: [0,1]	- Smooth activation that outputs values between 0 and 1, making it suitable for binary classification - Historically popular	- Vanishing gradient problem due to saturated neurons - Output not zero-centered as sigmoid outputs are always positive - exp() operation is computationally intensive
Hidden	**Tanh (Hyperbolic tangent)**	$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ Output range: [-1,1]	- Zero centered outputs that help networks train faster	- Suffers with vanishing gradient problem when saturated
Hidden	**ReLU** **(Rectified Linear Unit)**	$f(x) = max(0,x)$ Output range: [0, ∞)	- Does not saturate: avoids vanishing gradient issues for positive inputs - Computationally efficient since only certain number of neurons are activated at the same time - Much faster convergence compared to sigmoid/tanh	- Output not zero-centered - Prone to a "dying ReLU" problem, where neurons can get stuck during training and never activate again, leading to a dead neuron that doesn't update its weights. © AIML.com Research

Used in layer	Activation Function	Details	Pros	Cons
Hidden / Output	**Sigmoid**	$\sigma(x) = 1/(1+e^{-x})$ Output range: [0,1]	- Smooth activation that outputs values between 0 and 1, making it suitable for binary classification - Historically popular	- Vanishing gradient problem due to saturated neurons - Output not zero-centered as sigmoid outputs are always positive - exp() operation is computationally intensive
Hidden	**Tanh (Hyperbolic tangent)**	$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ Output range: [-1,1]	- Zero centered outputs that help networks train faster	- Suffers with vanishing gradient problem when saturated
Hidden	**ReLU** **(Rectified Linear Unit)**	$f(x) = max(0,x)$ Output range: $[0, \infty)$	- Does not saturate: avoids vanishing gradient issues for positive inputs - Computationally efficient since only certain number of neurons are activated at the same time - Much faster convergence compared to sigmoid/tanh	- Output not zero-centered - Prone to a "dying ReLU" problem, where neurons can get stuck during training and never activate again, leading to a dead neuron that doesn't update its weights. © AIML.com Research

Which activation function to choose from?

Activation function for hidden layers:

- In practice, the choice of activation functions is as follows in order of priority:
- ReLU is the top choice as it is simpler, faster, much lower run time, better convergence performance and do not suffer from vanishing gradient issues
- Leaky ReLU, PReLU, Maxout and ELU
- tanh
- Sigmoid (not preferred anymore)

Activation function for output layers:

- Sigmoid for binary classification, multi-label classification
- Softmax for multi-class classification
- Identity / linear function for regression problems

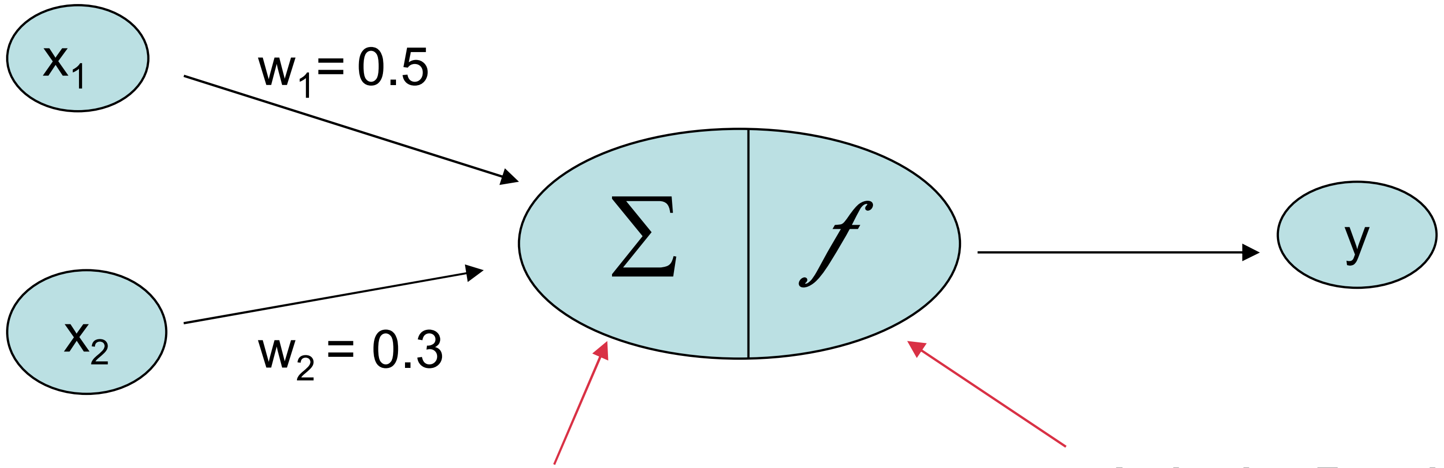
Exercise

- 2 input AND

1	1	1
1	0	0
0	1	0
0	0	0

- 2 input OR

1	1	1
1	0	1
0	1	1
0	0	0



$y_{in} = x_1w_1 + x_2w_2$

<u>Activation Function:</u>
Binary Step Function
θ = 0.5,

f(y-in) = 1 if y-in >= θ
dan f(y-in) = 0