MP Neuron, Perceptron, and Sigmoid Neuron

Praveen K. Chandaliya

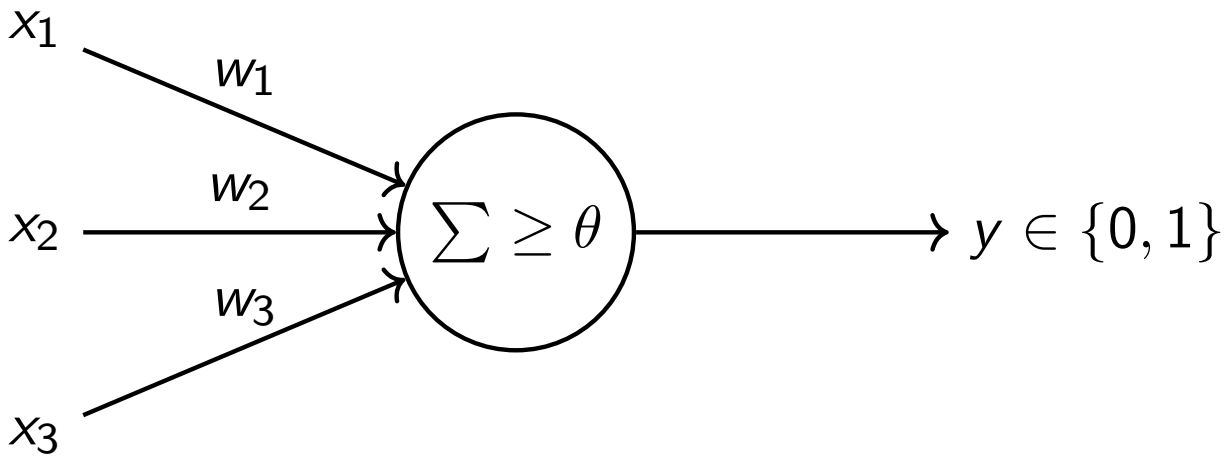
DoAI, SVNIT Surat

January 12, 2026

Motivation

- Artificial neurons are the building blocks of neural networks
- Neuron models evolved from **logic** to **learning** to **gradient-based optimization**
- We compare:
  - McCulloch–Pitts (MP) Neuron
  - Perceptron
  - Sigmoid Neuron

MP Neuron Model



McCulloch–Pitts (MP) Neuron

Key Idea: Logical neuron with hard threshold

Model:

$$y = \begin{cases} 1, & \sum w_i x_i \geq \theta \\ 0, & \text{otherwise} \end{cases}$$

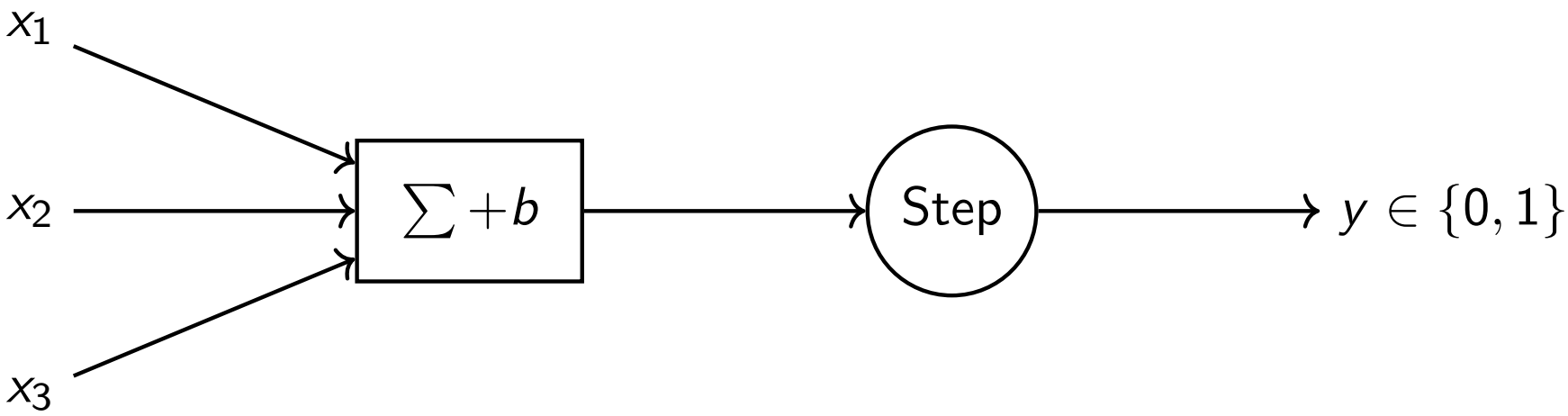
Characteristics:

- Binary inputs and outputs
- Fixed weights (no learning)
- Implements logic gates (AND, OR, NOT)

Limitations of MP Neuron

- No learning mechanism
- Not adaptive
- Cannot handle real-valued data
- Only suitable for hard-coded logic

Perceptron Model



Perceptron

Key Idea: MP neuron + learning

Model:

$$z = \sum w_i x_i + b$$

$$y = \text{step}(z)$$

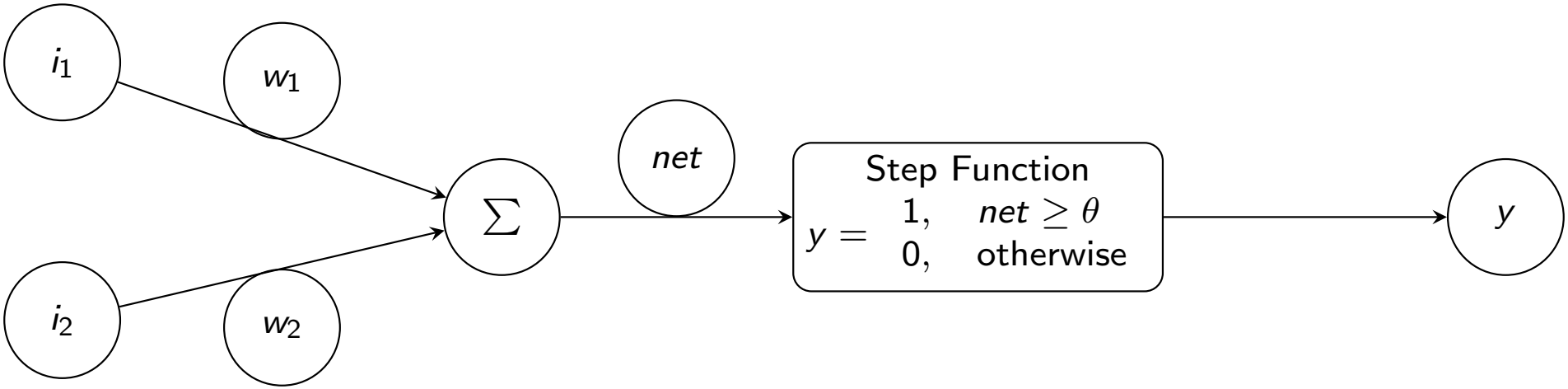
Learning Rule:

$$w_i \leftarrow w_i + \eta(y_{\text{true}} - y_{\text{pred}})x_i$$

Perceptron Example

Parameter	**Specification**
Input Pattern i_1	[0, 0, 1, 1]
Input Pattern i_2	[0, 1, 0, 1]
Target Output t	[0, 0, 0, 1]
Learning Rate (η)	0.4
Initial Weights	$w_1 = 0.6,\ w_2 = -0.2$
Activation Function	Step Function $$y = \begin{cases} 1 & \text{if } net \geq \theta \\ 0 & \text{otherwise} \end{cases}$$
Threshold (θ)	1.5
Number of Input Nodes	2
Number of Output Nodes	1

Single Layer Perceptron Model



Properties of Perceptron

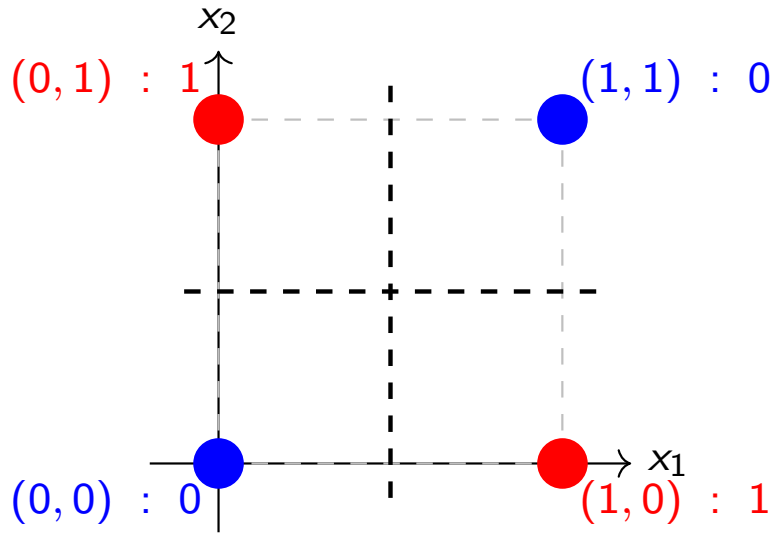
- Supervised learning
- Binary output
- Works only for linearly separable data
- Cannot solve XOR problem

XOR Problem

x_1	x_2	XOR Output
0	0	0
0	1	1
1	0	1
1	1	0

Key Question: Can a single linear decision boundary separate these points?

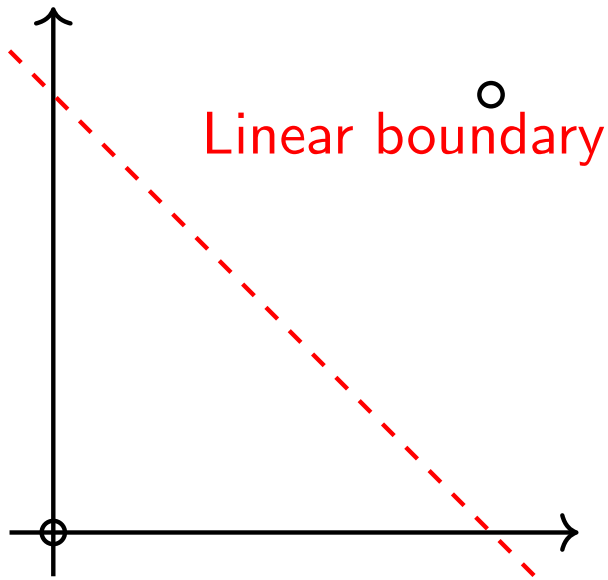
XOR Failure: Geometric View



No single straight line can separate the classes

Conclusion: Single-layer perceptron fails for XOR because XOR is not linearly separable.

Why Linear Separation Fails



Result: At least one point is always misclassified.

Why Single-Layer Perceptron Cannot Solve XOR

Assume a linear classifier:

$$y = \text{step}(w_1 x_1 + w_2 x_2 + b)$$

From XOR truth table:

$$\begin{aligned} (0,1): &\quad w_2 + b > 0 \\ (1,0): &\quad w_1 + b > 0 \\ (0,0): &\quad b < 0 \\ (1,1): &\quad w_1 + w_2 + b < 0 \end{aligned}$$

Adding first two inequalities:

$$w_1 + w_2 + 2b > 0$$

Contradiction with:

$$w_1 + w_2 + b < 0$$

Conclusion: XOR Failure

- MP neuron and Perceptron create **linear decision boundaries**
- XOR requires **non-linear separation**
- Therefore:
  - MP Neuron fails
  - Perceptron fails
  - Multi-layer networks succeed

XOR is not linearly separable

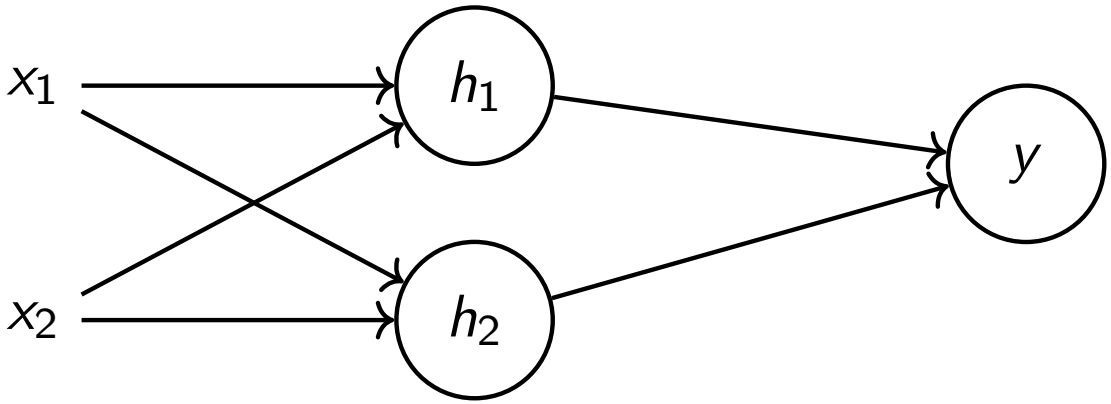
How to Solve XOR

- Use **two-layer network**
- Hidden neurons learn intermediate features
- Output neuron combines them non-linearly

Key Insight:

$$\text{Non-linearity} + \text{hidden layers} \Rightarrow \text{XOR solvable}$$

XOR Solution Using a Multi-Layer Network



Key Idea: Hidden neurons create non-linear feature space.

How Hidden Neurons Solve XOR

- Hidden neuron h_1 learns: $x_1 \vee x_2$
- Hidden neuron h_2 learns: $x_1 \wedge x_2$
- Output neuron computes:

$$\text{XOR} = (x_1 \vee x_2) - (x_1 \wedge x_2)$$

Insight:

$$\boxed{\text{XOR} = \text{combination of linearly separable problems}}$$

XOR vs AND / OR

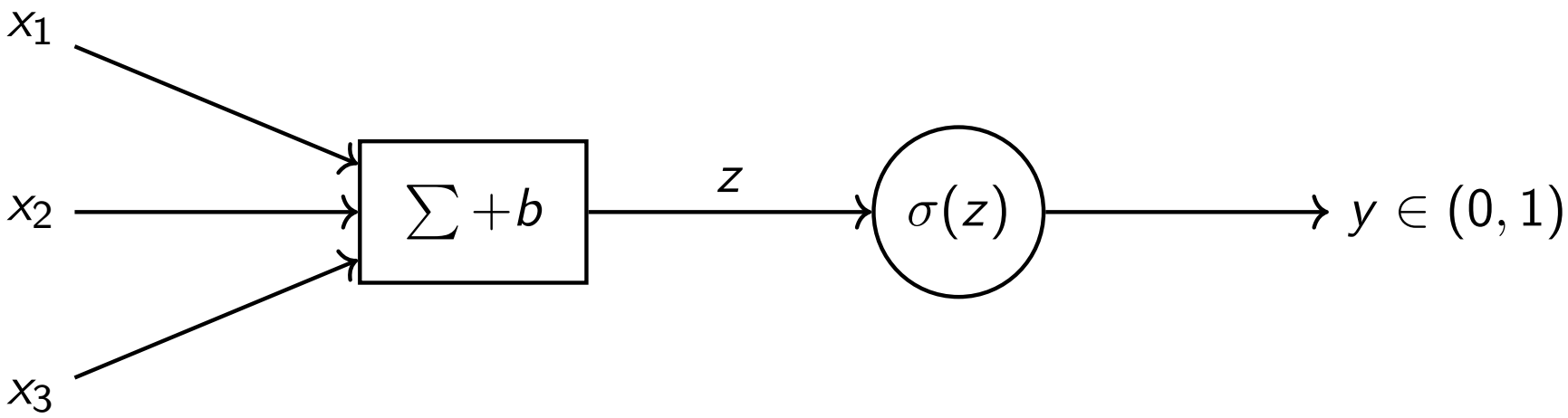
- AND and OR are linearly separable
- XOR is not linearly separable

x_1	x_2	AND	XOR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Conclusion:

Linear model fails only for XOR

Sigmoid Neuron Model



Sigmoid Neuron

Key Idea: Smooth, differentiable activation

Model:

$$z = \sum w_i x_i + b$$

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

Derivative:

$$\frac{d\sigma}{dz} = \sigma(z)(1 - \sigma(z))$$

Sigmoid Function Derivative

The sigmoid (logistic) function is defined as:

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

Rewrite it to clearly expose its structure:

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

The sigmoid function is a **nested function**.

Outer function:

$$f(u) = u^{-1}$$

Inner function:

$$u(x) = 1 + e^{-x}$$

Therefore:

$$\sigma(x) = f(u(x))$$

This decomposition is the key idea of the chain rule.

Step 2: Apply the Chain Rule

The chain rule states:

$$\frac{d}{dx} f(u(x)) = \frac{df}{du} \cdot \frac{du}{dx}$$

We will compute:

- $\frac{df}{du}$ — derivative of the outer function
- $\frac{du}{dx}$ — derivative of the inner function

Step 3: Differentiate the Outer Function

Given:

$$f(u) = u^{-1}$$

Using the power rule:

$$\frac{df}{du} = -u^{-2}$$

This handles the **outer layer** of the sigmoid function.

Step 4: Differentiate the Inner Function

Given:

$$u(x) = 1 + e^{-x}$$

Derivative of the constant:

$$\frac{d}{dx}(1) = 0$$

Derivative of the exponential term:

$$\frac{d}{dx}(e^{-x}) = -e^{-x}$$

Therefore:

$$\frac{du}{dx} = -e^{-x}$$

Step 5: Apply the Chain Rule

Using the chain rule:

$$\frac{d\sigma}{dx} = (-u^{-2}) \cdot (-e^{-x})$$

Substitute:

$$u = 1 + e^{-x}$$

$$\frac{d\sigma}{dx} = -(1 + e^{-x})^{-2} \cdot (-e^{-x})$$

The two negative signs cancel.

Step 6: Simplify

$$\frac{d\sigma}{dx} = (1 + e^{-x})^{-2} \cdot e^{-x}$$

Rewrite in fraction form:

$$\boxed{\frac{d\sigma}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2}}$$

Recall:

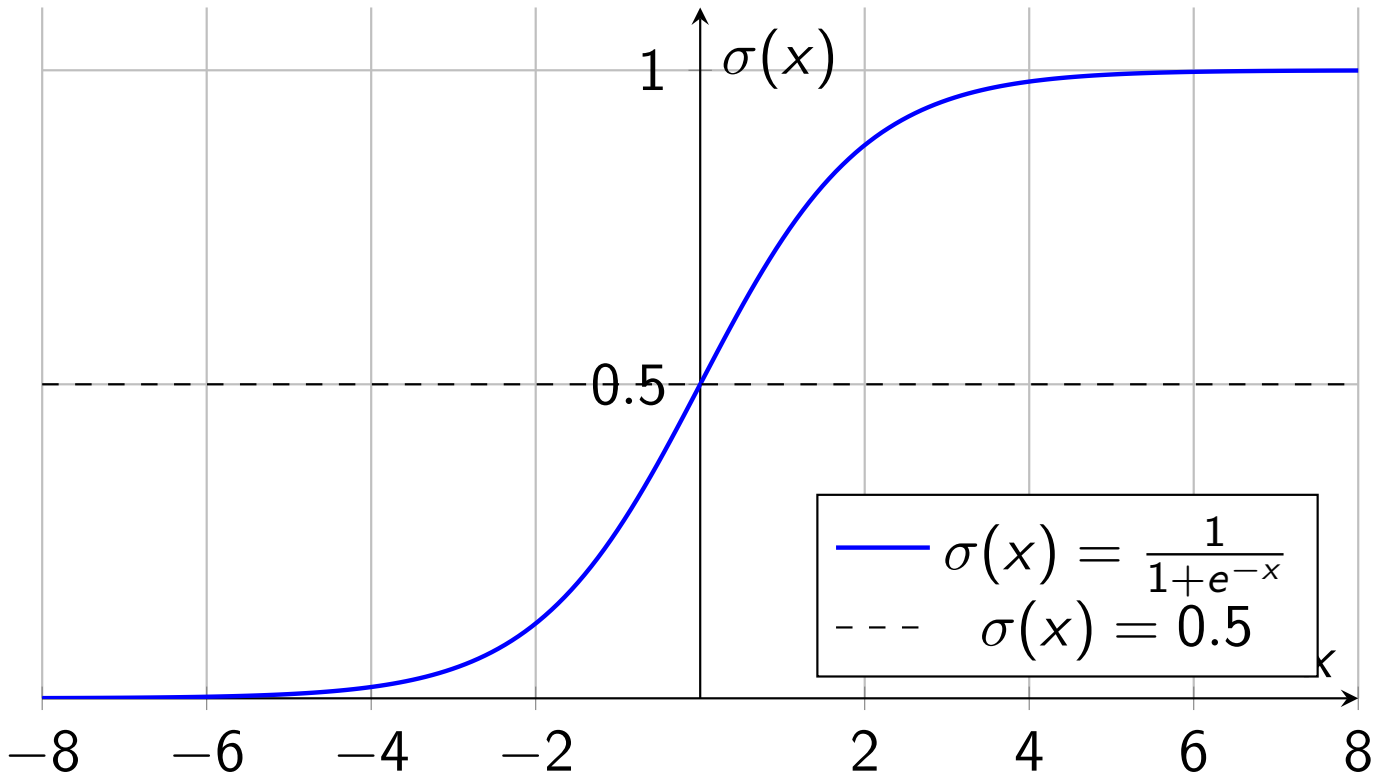
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

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

$$\sigma(x)\,[1 - \sigma(x)] = \frac{e^{-x}}{(1 + e^{-x})^2}$$

$$\boxed{\frac{d\sigma(x)}{dx} = \sigma(x)\,[1 - \sigma(x)]}$$

Sigmoid Function Plot



Why Sigmoid Neuron is Important

- Differentiable activation function
- Enables backpropagation
- Output interpreted as probability
- Foundation of multi-layer neural networks

Comparison Table

Feature	**MP**	**Perceptron**	**Sigmoid**
Inputs	Binary	Real	Real
Weights	Fixed	Learnable	Learnable
Bias	No	Yes	Yes
Activation	Threshold	Step	Sigmoid
Output	0 / 1	0 / 1	(0, 1)
Differentiable	No	No	Yes
Learning	No	Yes	Yes
Backpropagation	No	No	Yes

Evolution of Neuron Models

MP Neuron $\rightarrow$ Perceptron $\rightarrow$ Sigmoid Neuron

- Logic $\rightarrow$ Learning $\rightarrow$ Gradient-based optimization

Summary

- MP neuron is a threshold-based logical model
- Perceptron introduces supervised learning
- Sigmoid neuron enables smooth learning using gradients
- Sigmoid neuron forms the basis of modern neural networks

Thank You