9. Multi Layer Perceptron

Course: Introduction to AI
Instructor: Saumya Jetley
Teaching Assistant(s): Raghav Awasty & Subhrajit Roy

October 4, 2022

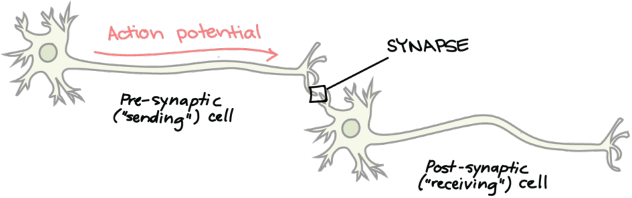
Mathematical model of a Neuron

Genesis of neural networks

- **Mathematical model of a Neuron**
  - *Understanding the mechanics of a single neuron*
- Building of a perceptron
- Single to Multi-layer perceptrons

Mathematical model of a Neuron

Neuron - A physical model[1]



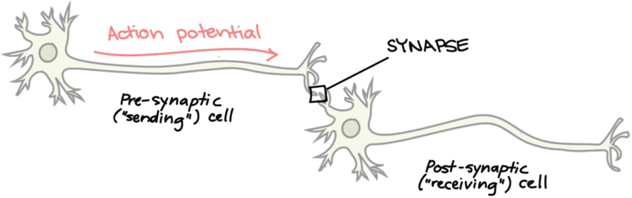
The simplest functional unit in the human brain that,

- Fires at a certain pattern of values at its input
- Passes this information to its neighbours

[1] Image courtesy of Khan Academy

Mathematical model of a Neuron

Neuron - A physical model[1]



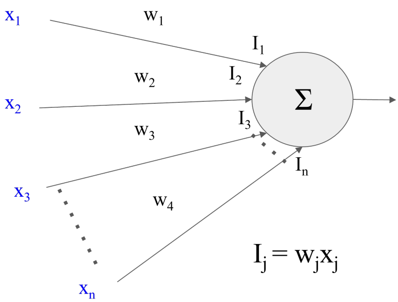
Hubel and Weisel (1962)

- Early neurons in the visual cortex fire at simpler patterns
- Later neurons in the visual cortex fire at more complex patterns

[1] Image courtesy of Khan Academy

Mathematical model of a Neuron

Neuron - A mathematical model (*McCulloch-Pitts Neuron*)

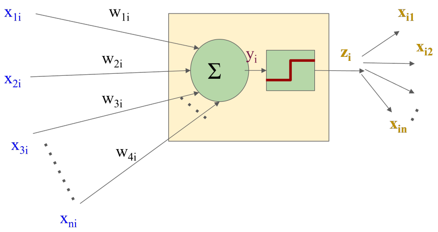


A **computational primitive** that:

- Accepts multiple inputs each of which is modelled as an analog signal (≥ 0)
- Performs a weighted combination of the inputs
- Applies a non-linear activation function
- Passes the output signal to a downstream neuron

Mathematical model of a Neuron

Neuron - A mathematical model



Parameters of this model?

- $\mathbf{x}_i^T = [x_{1i}, x_{2i}, x_{3i}, \ldots x_{ni}]$
- $\mathbf{w}_i^T = [w_{1i}, w_{2i}, w_{3i}, \ldots W_{ni}]$
- $y_i = \mathbf{w}_i^T \mathbf{x}_i$
- $z_i = max(y_i, 0)$

Building of a perceptron

Genesis of neural networks

- Mathematical model of a Neuron
- **Building of a perceptron**
  - *Building a network of neurons - one input layer and one output layer*
- Single to Multi-layer perceptrons

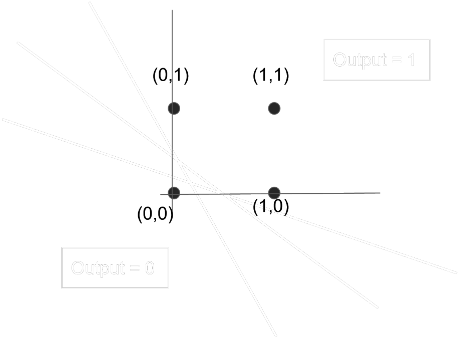
Building of a perceptron

- **Aim**: To use the network of neurons to perform mathematical operations
- **Context**:
  - Early computers (and modern too) are based on binary logic
  - Are neural networks able to implement logical operations?
- **Todo**:
  - Implement elementary logic gates – AND, OR and NOT
  - Any gate can be implemented using the 3 gates above

Building of a perceptron

OR Gate

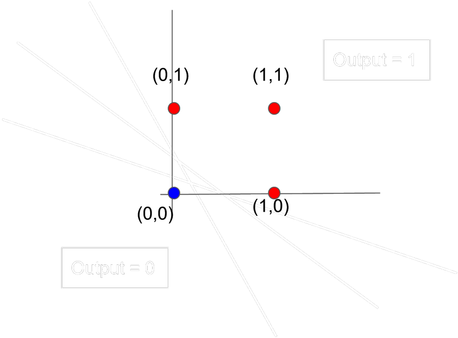
input1	input2	input3
0	0	0
0	1	1
1	0	1
1	1	1



Building of a perceptron

OR Gate

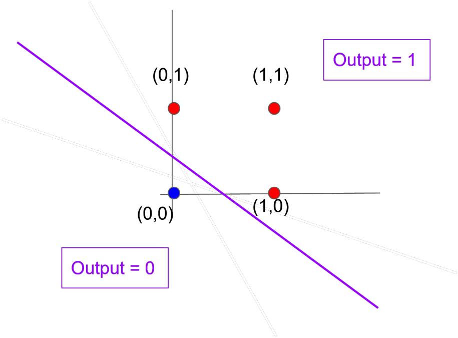
input1	input2	input3
0	0	0
0	1	1
1	0	1
1	1	1



Building of a perceptron

OR Gate

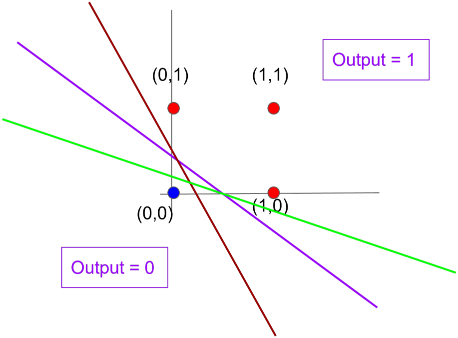
input1	input2	input3
0	0	0
0	1	1
1	0	1
1	1	1



Building of a perceptron

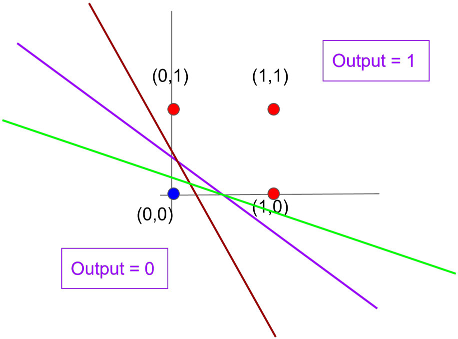
OR Gate

input1	input2	input3
0	0	0
0	1	1
1	0	1
1	1	1



Building of a perceptron

OR Gate



Decision boundary defined by,

$ax + by = c$

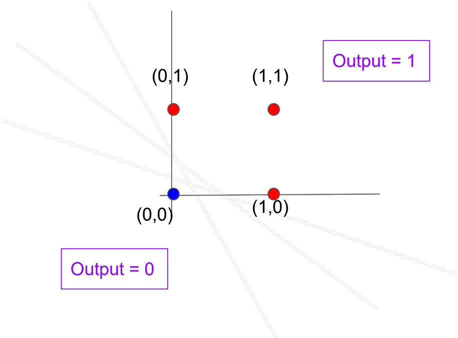
a = scaling of input1
b = scaling of input2
c = threshold of the activation function

Or

$y = (-\frac{a}{b})x + \frac{c}{b}$

Building of a perceptron

OR Gate

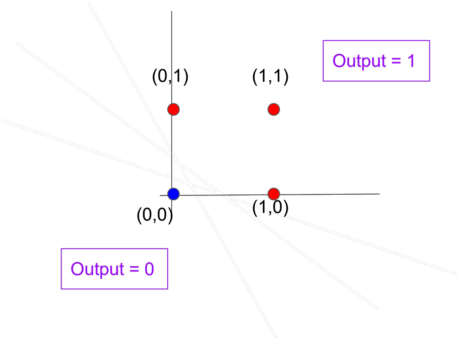


Sample decision boundary as per,

$y = (-\frac{a}{b})x + \frac{c}{b}$

Building of a perceptron

OR Gate



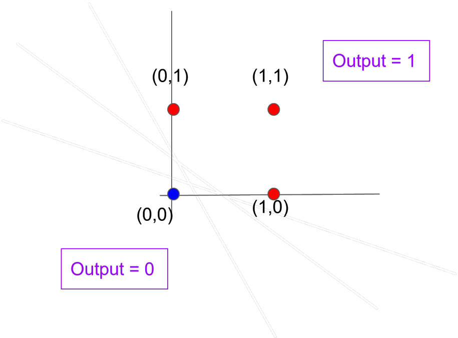
Sample decision boundary as per,

$y = (-\frac{a}{b})x + \frac{c}{b}$

$-\frac{a}{b} = -1; \frac{c}{b} = 0.5$

Building of a perceptron

OR Gate



Sample decision boundary as per,

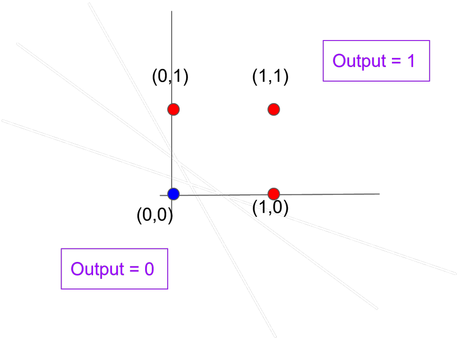
$y = (-\frac{a}{b})x + \frac{c}{b}$

$-\frac{a}{b} = -1; \frac{c}{b} = 0.5$

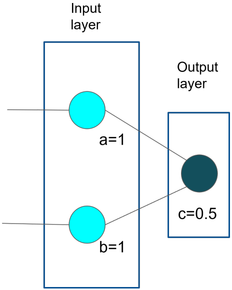
Solution space:
$(a, b, c) = (b, b, 0.5b)$

Building of a perceptron

OR Gate



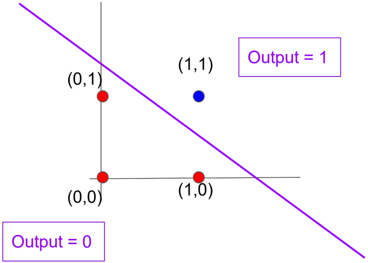
Solution space:
$(a, b, c) = (b, b, 0.5b)$



Building of a perceptron

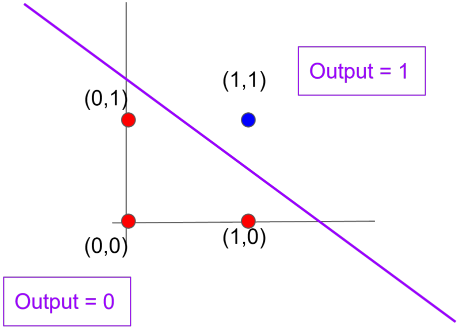
AND Gate

input1	input2	input3
0	0	0
0	1	0
1	0	0
1	1	1



Building of a perceptron

AND Gate



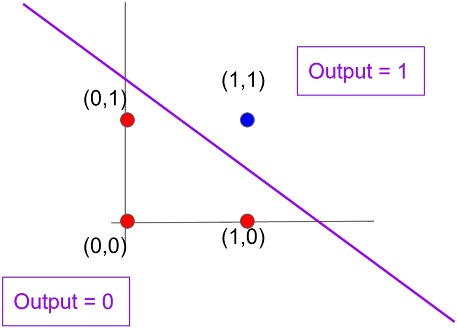
Sample decision boundary as per,

$y = (-\frac{a}{b})x + \frac{c}{b}$

$-\frac{a}{b} = -1; \frac{c}{b} = 1.5$

Building of a perceptron

AND Gate



Sample decision boundary as per,

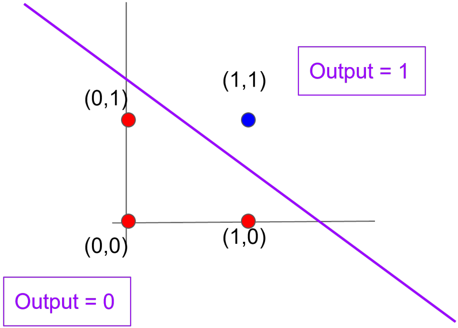
$y = (-\frac{a}{b})x + \frac{c}{b}$

$-\frac{a}{b} = -1; \frac{c}{b} = 1.5$

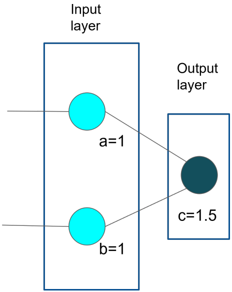
Solution space:
$(a, b, c) = (b, b, 1.5b)$

Building of a perceptron

AND Gate



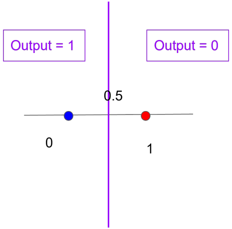
Solution space:
$(a, b, c) = (b, b, 1.5b)$



Building of a perceptron

NOT Gate

input1	input2
0	1
1	0



Sample decision boundary as per,

$y = (-\frac{a}{b})x + \frac{c}{b}$

$-\frac{a}{b} = -1; \frac{c}{b} = 0.5$

Solution space:

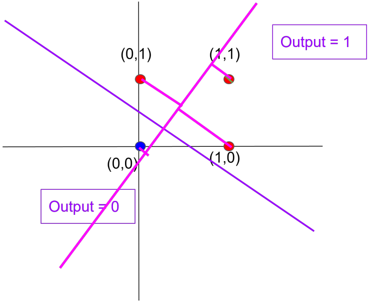
$(a, b, c) = (b, b, 0.5b)$

Building of a perceptron

Task of learning the model is equivalent to:

(a) Finding the weights on the incoming lines - **Axis of projection**

(b) Finding the appropriate threshold for a linear separation - **Threshold along the axis**

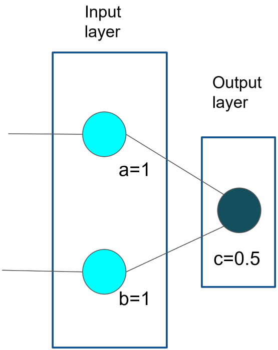


Building of a perceptron

Task of learning the model is equivalent to:

(a) Finding the **weights on the incoming lines**

(b) Finding the **appropriate threshold** for a linear separation



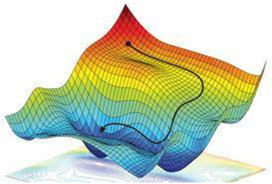
Building of a perceptron

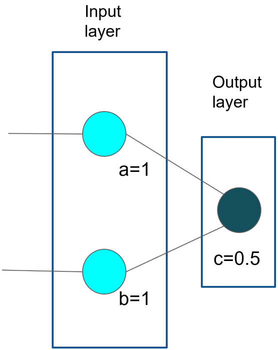
Task of learning the model is equivalent to:

(a) Finding the **weights on the incoming lines**

(b) Finding the **appropriate threshold** for a linear separation

1. Initialise parameters with random values
2. Calculate the error **e** or loss **l**
3. Update parameter $p = p - \frac{de}{dp}$



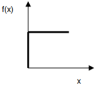


Building of a perceptron

Loss function description: **Map to Categorical distribution** + Measure loss

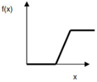
1. Step function - Non-continuous and Non-differentiable

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



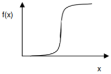
2. Piecewise linear - Continuous and Non-differentiable

$$f(x) = \begin{cases} 1 & \text{if } x \geq 0.5 \\ x + 0.5 & \text{if } -0.5 \leq x \leq 0.5 \\ 0 & \text{if } x \leq 0.5 \end{cases}$$



3. Sigmoid - Continuous and differentiable

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



Building of a perceptron

Loss function description: Map to Categorical distribution + **Measure loss**

Cross Entropy Loss

$$-\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Sigmoid

Sum Squared Error

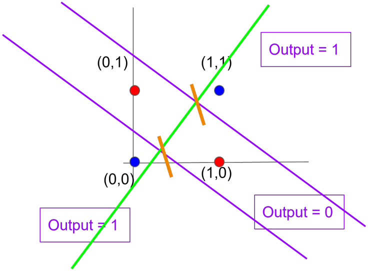
$$\sum_{i=1}^{n}(y_i - f(x_i))^2$$

Softmax

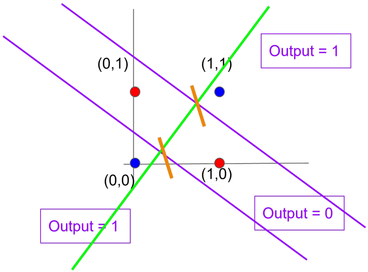
Building of a perceptron

XOR Gate

input1	input2	input3
0	0	0
0	1	1
1	0	1
1	1	0

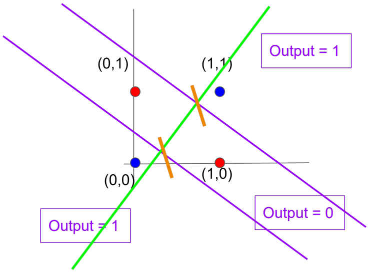


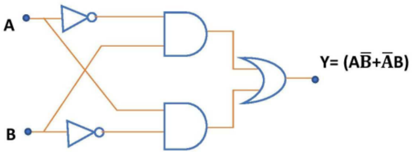
Building of a perceptron



1. Complexify the activation function

Building of a perceptron





1. Complexify the activation function
2. Complexify the mapping/projection

Single to Multi-layer perceptrons

Genesis of neural networks

- Mathematical model of a Neuron
- Building of a perceptron
- **Single to Multi-layer perceptrons**
  - *Limitations of single layer network and introduction of hidden layers*
  - *Complexify the mapping/projection*
  - *More general than adapting number of threshold to unknown settings*

Single to Multi-layer perceptrons

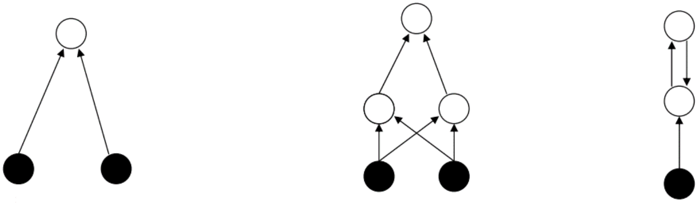


Figure: *Left*: Single layer perceptron; *Middle*: Multi layer perceptron; *Right*: Recurrent/ feedback perceptron

Overview

1. **Genesis of neural networks**
   - Mathematical model of a Neuron
   - Building of a perceptron
   - Single to Multi-layer perceptrons