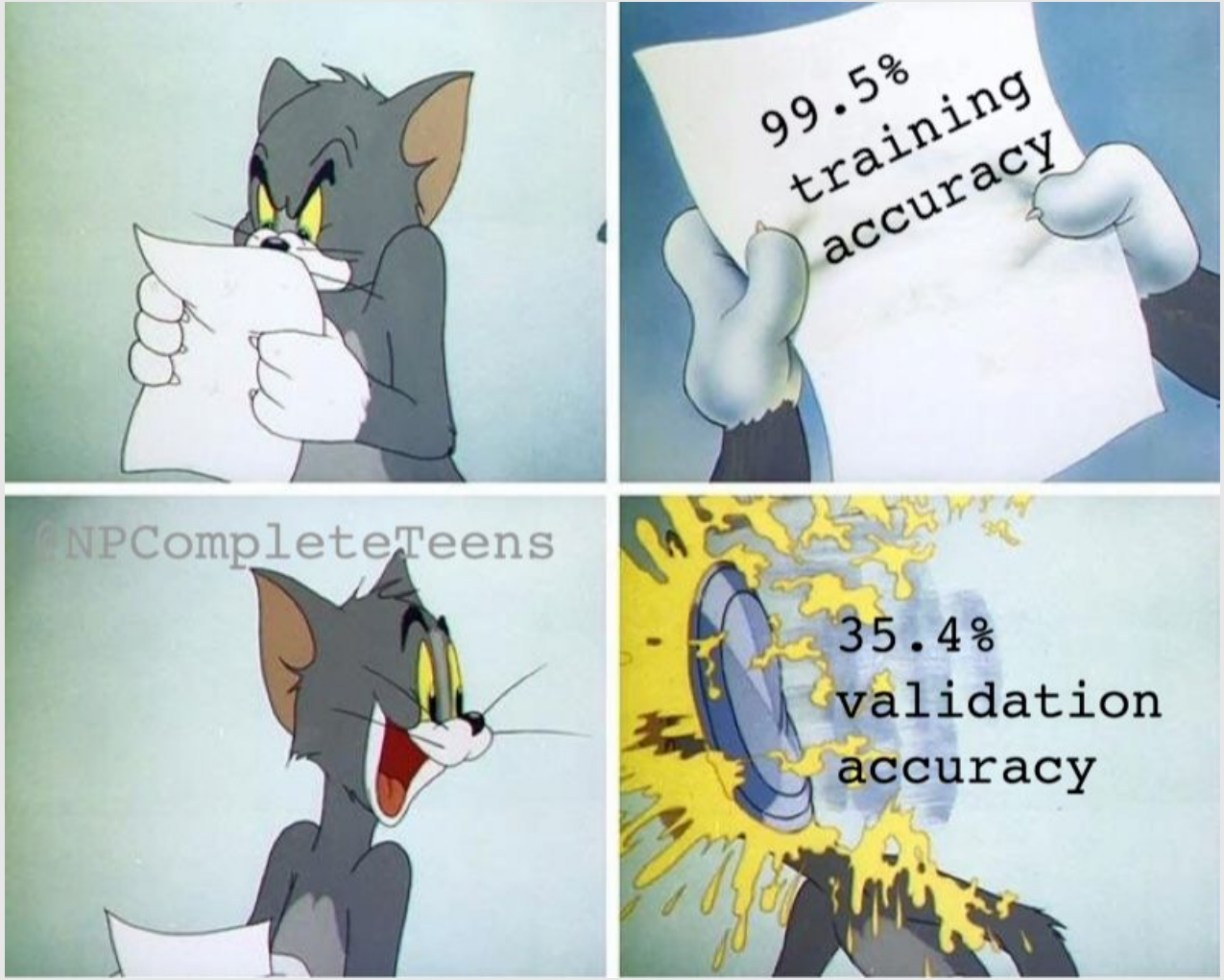
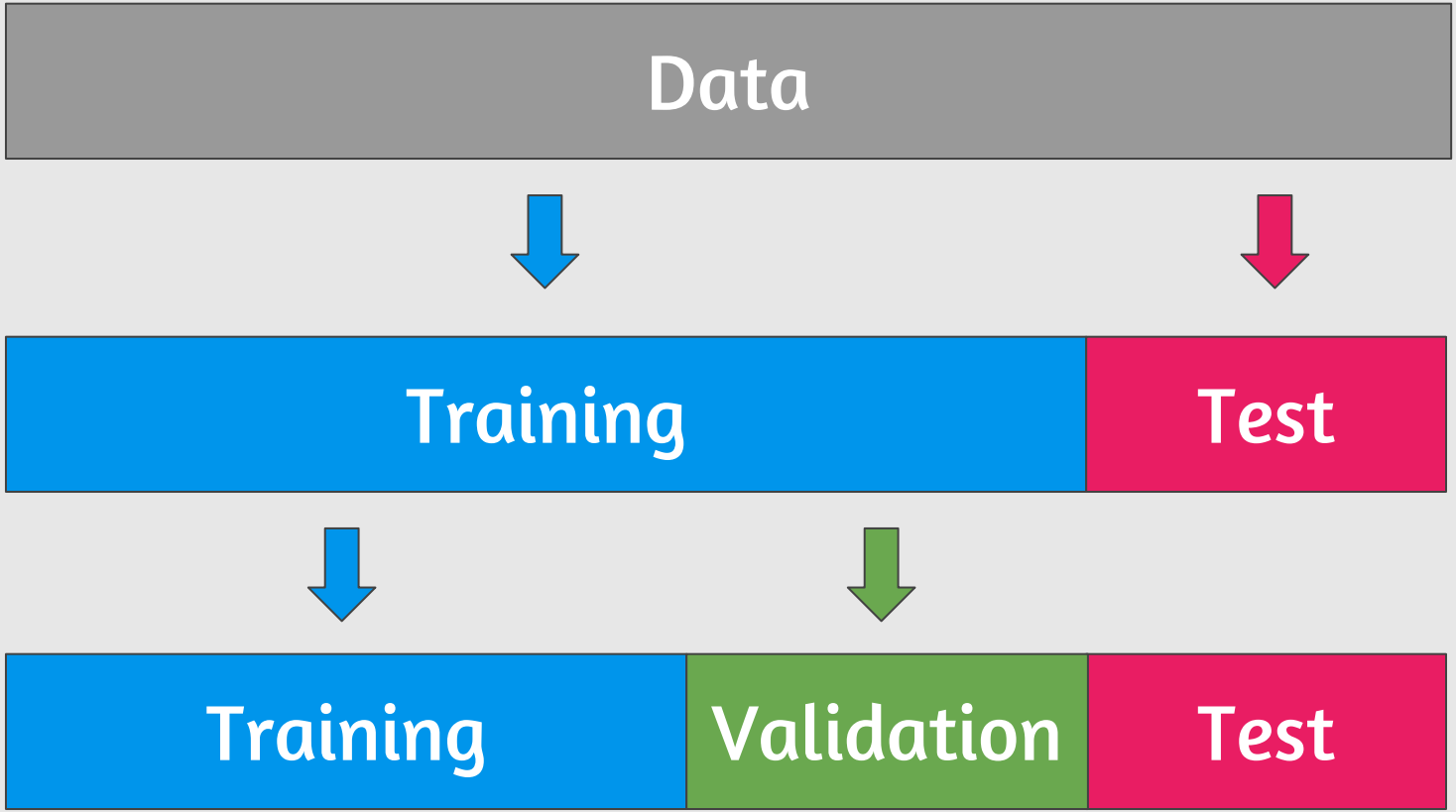
Recall from last time ...

Friends don’t let friends
use testing data
for training


99.5%
training
accuracy
@NPCompleteTeens
35.4%
validation
accuracy

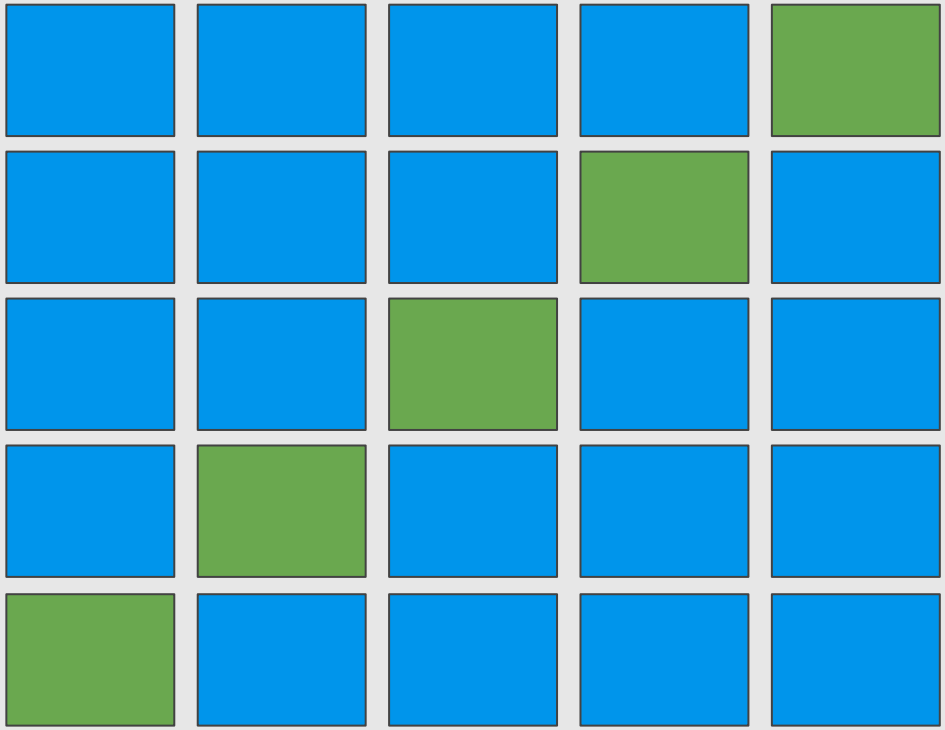

Data
Training
Test
Training
Validation
Test

k-fold Cross Validation

Training

Validation

$k = 5$



k×2-fold Cross Validation

Training

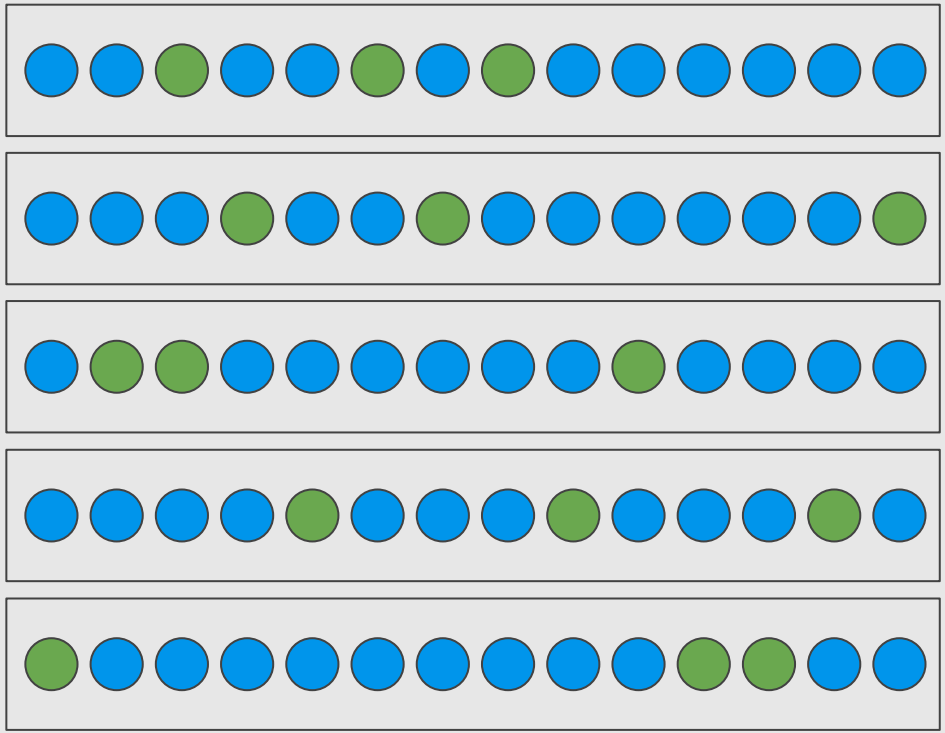
Validation

k = 5

...

k times = k×2 folds

Randomizing in Cross Validation



Evaluation Metrics

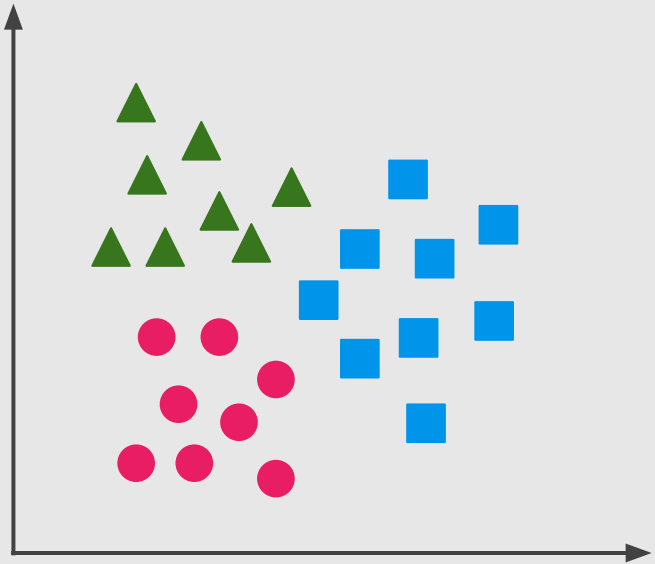
How well is my model doing?

Confusion Matrix Table

	Diagnosed Sick	Diagnosed Healthy
Sick	True Positive 	False Negative 
Healthy	False Positive 	True Negative 

Confusion Matrix Table (n classes)

Class 1: ▲
Class 2: ■
Class 3: ●



Predicted Class

True Class

	Guessed Class 1	Guessed Class 2	Guessed Class 3
Class 1	5	2	1
Class 2	3	6	0
Class 3	0	1	7

Accuracy

Patients \ Diagnosis	Diagnosed Sick	Diagnosed Healthy
Sick	1,000	200
Healthy	800	8,000

Accuracy:
Out of all the **patients**, how many did we classify correctly?

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{1{,}000 + 8{,}000}{10{,}000} = 90\%$$

Normalized Accuracy

Patients \ Diagnosis	Diagnosed Sick	Diagnosed Healthy
Sick	1,000	200
Healthy	800	8,000

Normalized Accuracy =

$$\frac{\frac{TP}{TP + FN} + \frac{TN}{TN + FP}}{2} =$$

$$\frac{\frac{1000}{1000 + 200} + \frac{8000}{8000 + 800}}{2} =$$

$$\frac{83.3 + 90.9}{2} = 87.1\%$$

Precision

Patients \ Diagnosis	Diagnosed Sick	Diagnosed Healthy
Sick	1,000	200
Healthy	800	8,000

Precision:
Out of all the patients we diagnosed with illness, how many were actually sick?

Precision =

$$\frac{1{,}000}{1{,}000 + 800} = 55.7\%$$

Recall

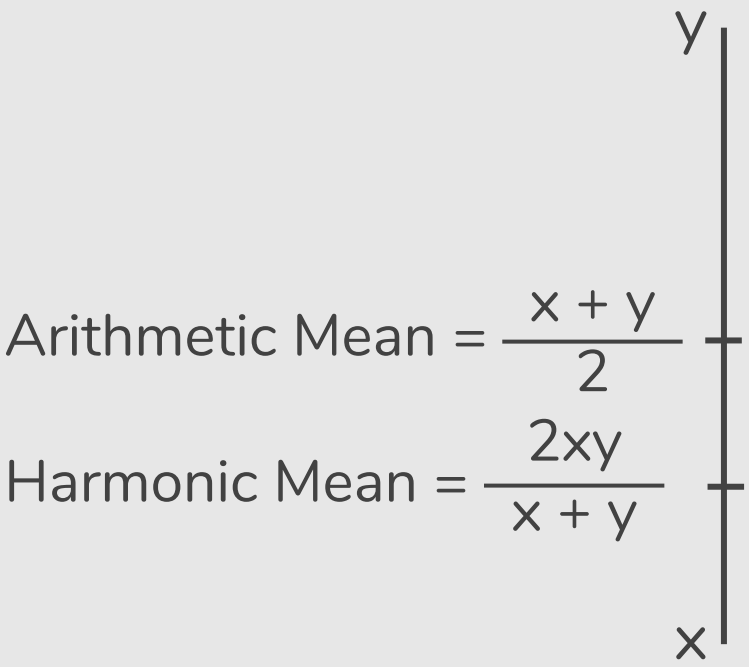
Patients \ Diagnosis	Diagnosed Sick	Diagnosed Healthy
Sick	1,000	200
Healthy	800	8,000

Recall:
Out of all the sick patients, how many did we correctly diagnose as sick?

Recall =

$$\frac{1{,}000}{1{,}000 + 200} = 83.3\%$$

Harmonic Mean



$$\text{Arithmetic Mean} = \frac{x + y}{2}$$

$$\text{Harmonic Mean} = \frac{2xy}{x + y}$$

Precision: 1
Recall: 0
Average = 0.5
Harmonic Mean = 0

Precision: 0.2
Recall: 0.8
Average = 0.5
Harmonic Mean = 0.32

F1 Score = Harmonic Mean (Precision, Recall)

F_β Score

$$F_1 = 2 \frac{\text{precison} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$F_\beta = (1 + \beta^2) \frac{\text{precison} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

F_β Score

F10 Score

Precision
(no false positive)

F0.5 Score

F1 Score

F2 Score

Recall
(no false negative)

References

- https://en.wikipedia.org/wiki/Precision_and_recall
- https://en.wikipedia.org/wiki/Binary_classification
- https://en.wikipedia.org/wiki/F1_score
- https://www.quora.com/What-is-an-intuitive-explanation-of-F-score
- “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms”, Neural Computation, p. 1895-1923, 1998 https://www.mitpressjournals.org/doi/10.1162/0899766698300017197
- **“A systematic analysis of performance measures for classification tasks”, 2009** https://www.sciencedirect.com/journal/information-processing-and-management
- Luis Serrano: https://www.youtube.com/watch?v=aDW44NPhNw0

RECOD
reasoning for complex data

Artificial Neural Networks

Machine Learning and Pattern Recognition

Prof. Sandra Avila
Institute of Computing (IC/Unicamp)

MC886/MO444, September 4, 2018

Many inventions were inspired by Nature ...

Birds inspired us to fly

https://i.pinimg.com/736x/18/4a/cc/184accf6b4d7980d2c2e80909ff295cd7--animal-tracks-friends.jpg

Many inventions were inspired by Nature ...

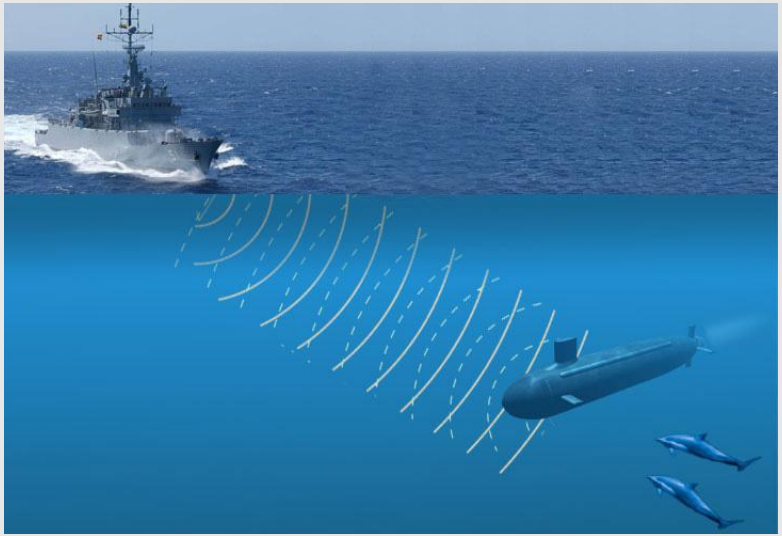
Burdock plants inspired velcro

http://www.youngworldclub.com/wp-content/uploads/2017/04/velcro.jpeg

https://c1.staticflickr.com/3/2887/12882961455_f2d6bee40c_b.jpg

Many inventions were inspired by Nature ...

Dolphins inspired sonar development



https://sites.google.com/site/echolocationkawproject/_/rsrc/1459209762464/sonars/image.jpeg

http://www.dolphinkind.com/images/dolphin_echolocation.jpg

Many inventions were inspired by Nature ...

Dogs inspired ...

https://farm2.staticflickr.com/1586/24734777053_d1bdb8bba4_o.jpg

It seems logical to look at the brain's architecture for inspiration on how to build an intelligent machine.

Today's Agenda

- Artificial Neural Networks
  - From Biological to Artificial Neurons
  - Biological Neurons
  - Logical Computations with Neurons
  - The Perceptron
  - Examples
  - Neural Network

From Biological to Artificial Neurons

Deep Neural Network (Pretraining)

SVM

Multi-layered Perceptron (Backpropagation)

ADALINE

XOR Problem

Perceptron

Golden Age

Dark Age ("AI Winter")

Electronic Brain

1943 1957 1960 1969 1986 1995 2006

1940 1950 1960 1970 1980 1990 2000 2010

S. McCulloch - W. Pitts

F. Rosenblatt

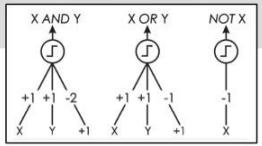
B. Widrow - M. Hoff

M. Minsky - S. Papert

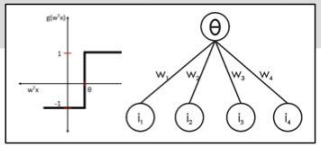
D. Rumelhart - G. Hinton - R. Wiliams

V. Vapnik - C. Cortes

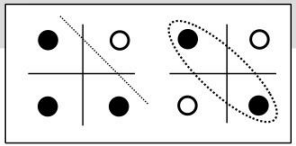
G. Hinton - S. Ruslan



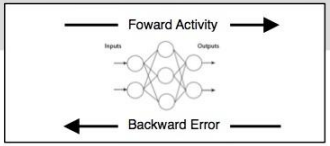
- Adjustable Weights
- Weights are not Learned



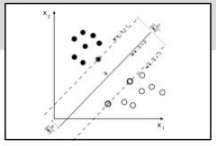
- Learnable Weights and Threshold



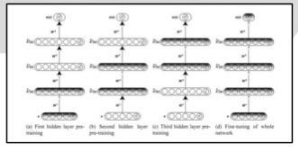
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



- Limitations of learning prior knowledge
- Kernel function: Human Intervention



- Hierarchical feature Learning

References

- “A logical calculus of the ideas immanent in nervous activity”, 1943: https://link.springer.com/content/pdf/10.1007%2FBF02478259.pdf
- “The Perceptron: A perceiving and recognizing automaton”, 1957 https://www.import.io/wp-content/uploads/2017/06/rosenblatt-1957.pdf
- “Perceptrons: An Introduction to Computational Geometry”, 1969
- “Learning representations by back-propagating errors”, 1986 https://www.nature.com/articles/323533a0

  https://web.stanford.edu/class/psych209a/ReadingsByDate/02_06/PDPVolIChapter8.pdf

ILSVRC 2012 — Image Classification task

Rank	Name	Error Rate (%)	Description
1	University of Toronto		Deep Learning
2	University of Tokyo	26.2	Hand-crafted features and learning models
3	University of Oxford	26.9	
4	Xerox/INRIA	27.0	

Object recognition over 1,000,000 images and 1,000 categories (2 GPU)

“ImageNet classification with deep convolutional neural networks”. Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton. In: NIPS, 2012.

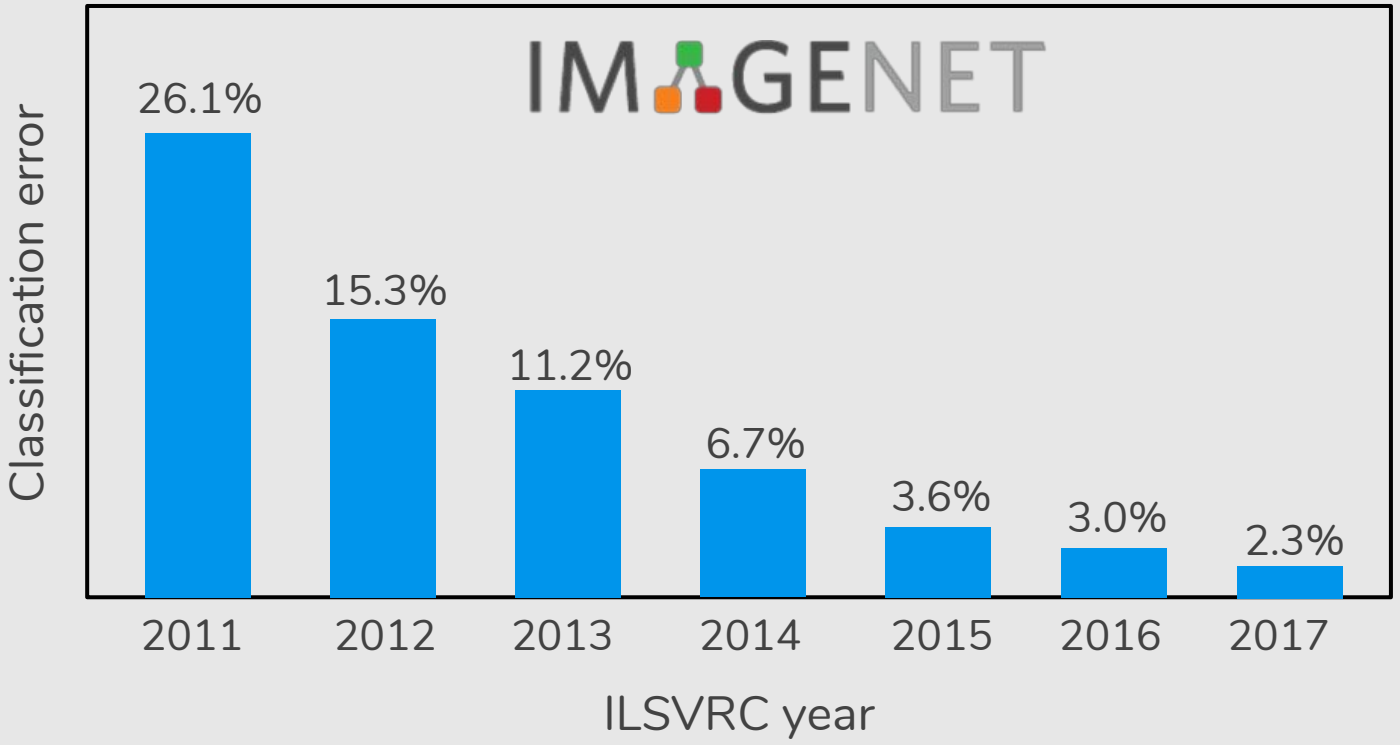
IMAGENET

ILSVRC 2012 — Image Classification task

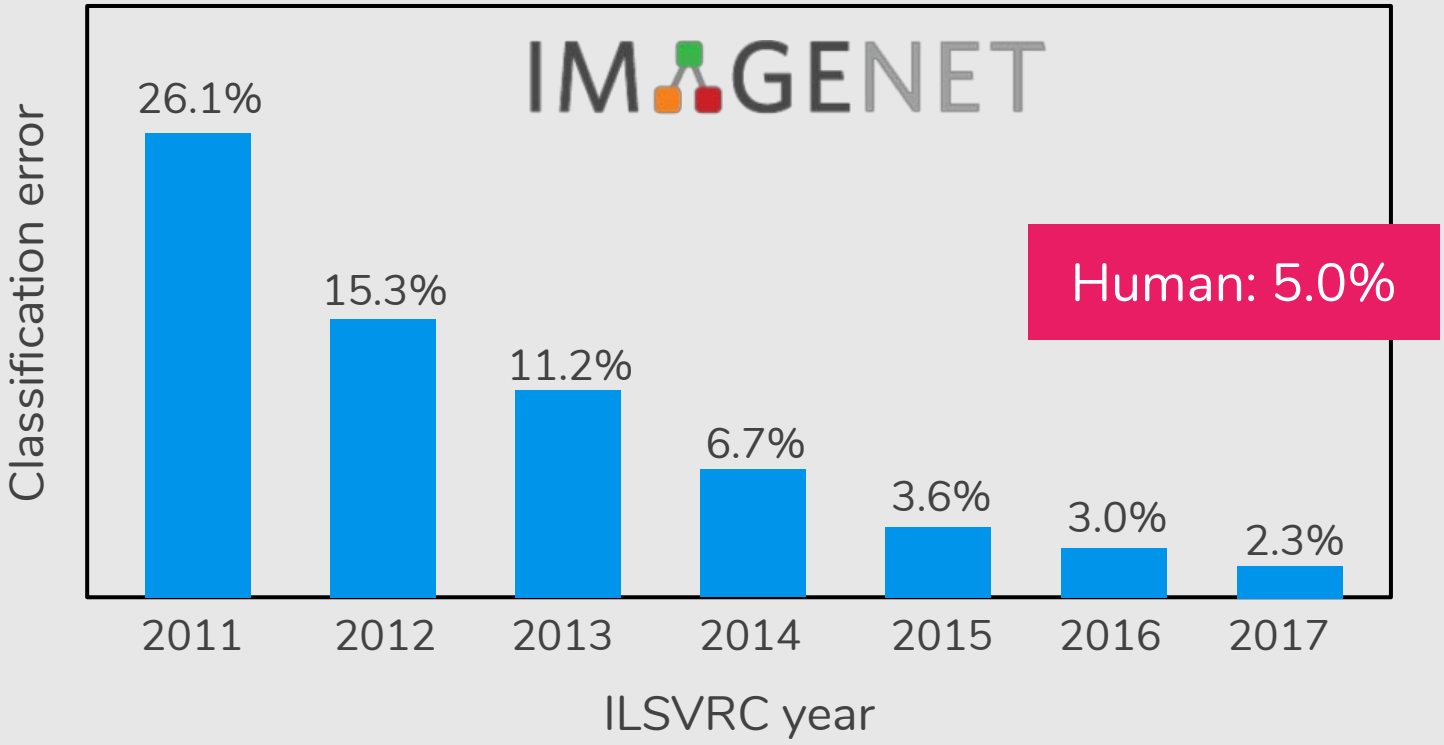
Rank	Name	Error Rate (%)	Description
1	University of Toronto	**15.3**	Deep Learning
2	University of Tokyo	26.2	Hand-crafted features and learning models
3	University of Oxford	26.9	
4	Xerox/INRIA	27.0	

Object recognition over 1,000,000 images and 1,000 categories (2 GPU)

"ImageNet classification with deep convolutional neural networks". Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton. In: NIPS, 2012.



“ImageNet classification with deep convolutional neural networks”. Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton. In: NIPS, 2012.



“ImageNet classification with deep convolutional neural networks”. Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton. In: NIPS, 2012.

Will this wave die out like the previous ones did?

From Biological to Artificial Neurons

---

1. There is now a **huge quantity of data** available to train neural networks.

IMAGENET

www.image-net.org

22K categories and 14M images

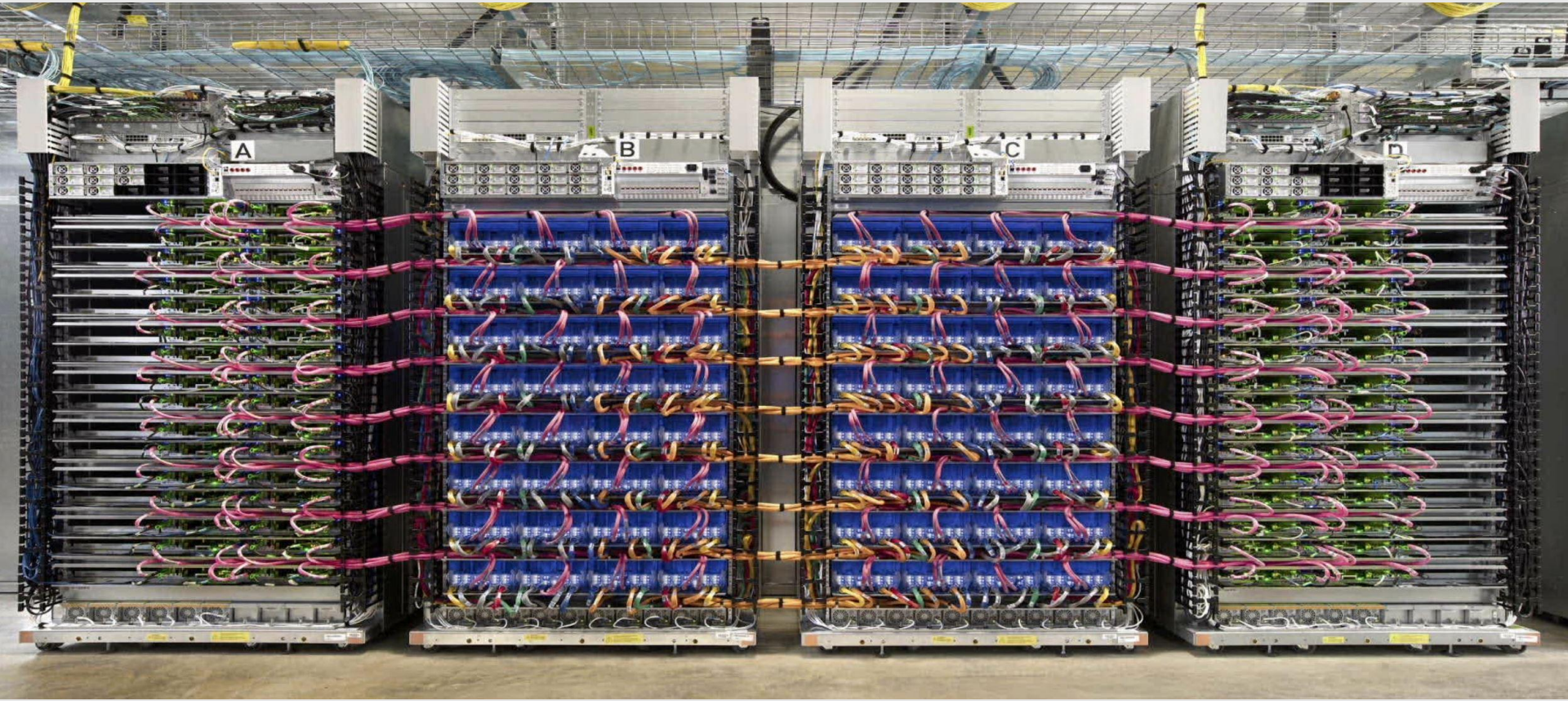
- Animals
  - Bird
  - Fish
  - Mammal
  - Invertebrate
- Plants
  - Tree
  - Flower
- Food
- Materials
- Structures
- Artifact
  - Tools
  - Appliances
  - Structures
- Person
- Scenes
  - Indoor
  - Geological Formations
- Sport Activities

Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009

From Biological to Artificial Neurons

---

1. There is now a **huge quantity of data** available to train neural networks.
2. **Computing power** now makes it possible to train large neural networks in a reasonable amount of time.



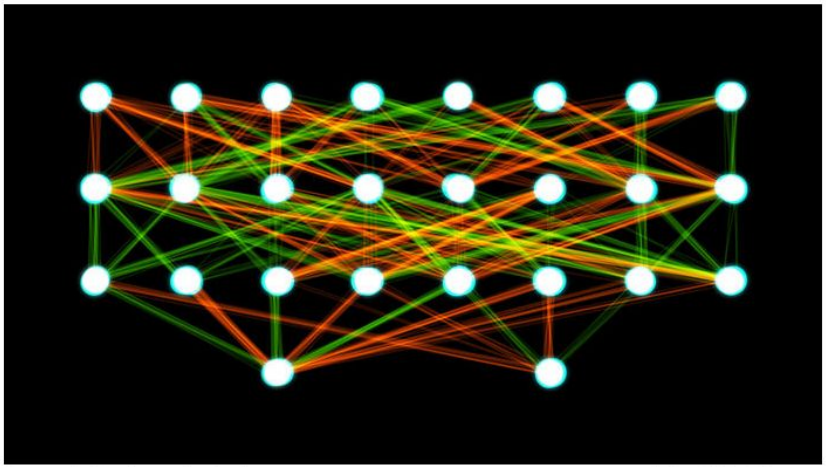
http://www.tomshardware.com/news/google-automl-aritifical-intelligence-ai,34533.html

From Biological to Artificial Neurons

---

1. There is now a **huge quantity of data** available to train neural networks.
2. **Computing power** now makes it possible to train large neural networks in a reasonable amount of time.
3. The **training algorithms** have been **improved**.

SHARE



A representation of a neural network. Akritasa/Wikimedia Commons

Brainlike computers are a black box. Scientists are finally peering inside

By Jackie Snow | Mar. 7, 2017 , 3:15 PM

Last month, Facebook announced software that could simply look at a photo and tell, for example, whether it was a picture of a cat or a dog. A related program identifies cancerous

From Biological to Artificial Neurons

---

1. There is now a **huge quantity of data** available to train neural networks.
2. **Computing power** now makes it possible to train large neural networks in a reasonable amount of time.
3. The **training algorithms** have been **improved**.
4. ANNs seem to have entered a virtuous circle of funding and progress.

Biological Neurons

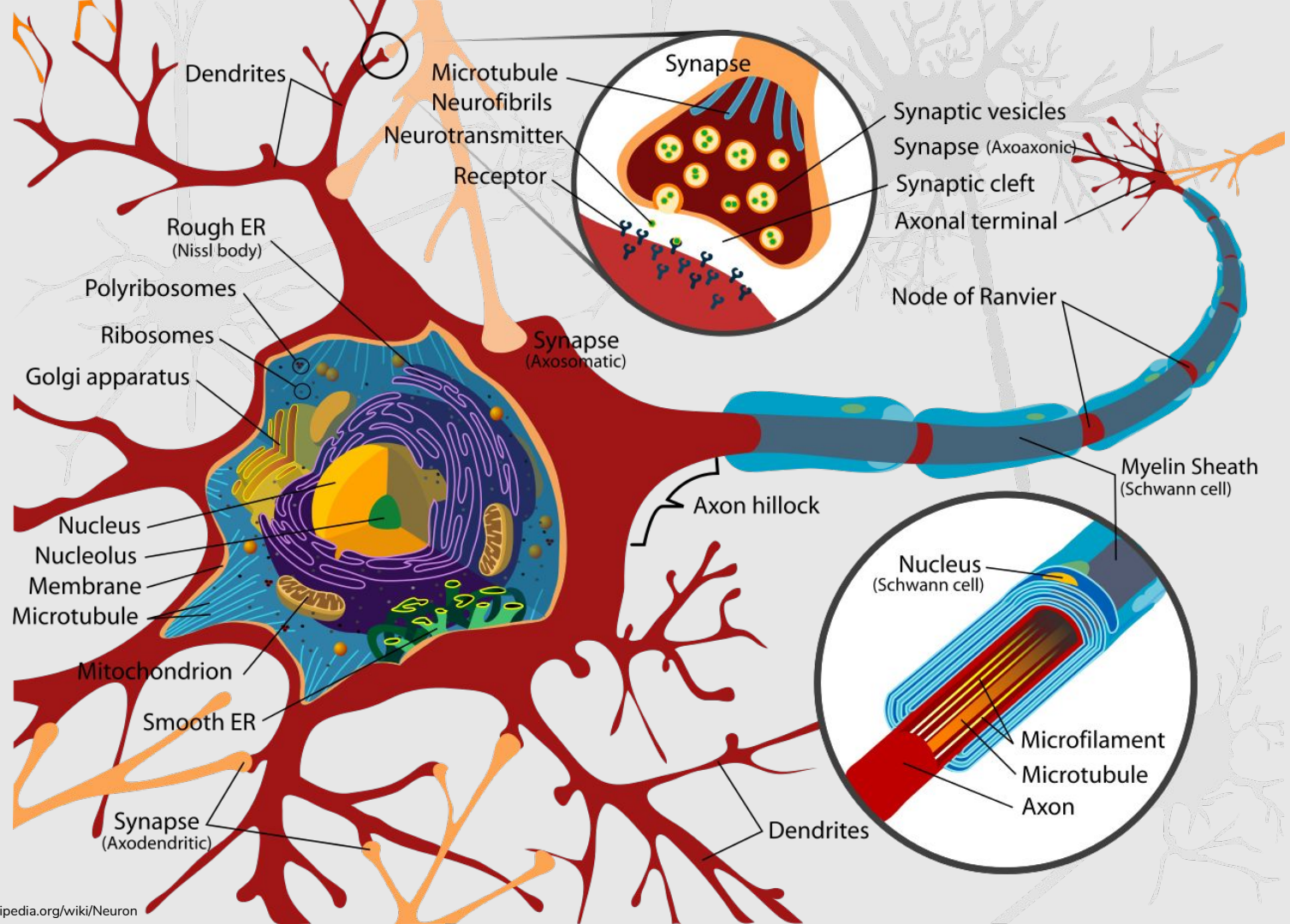
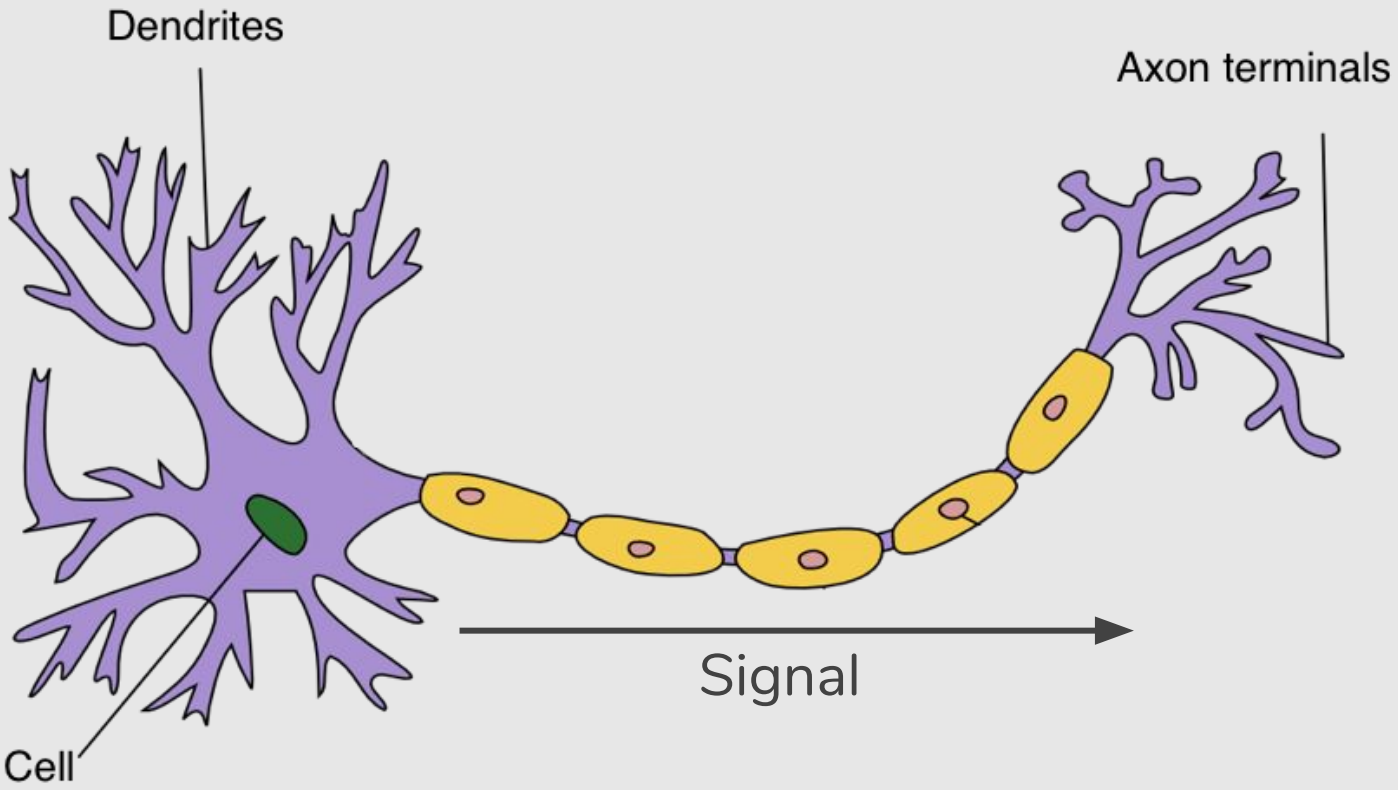
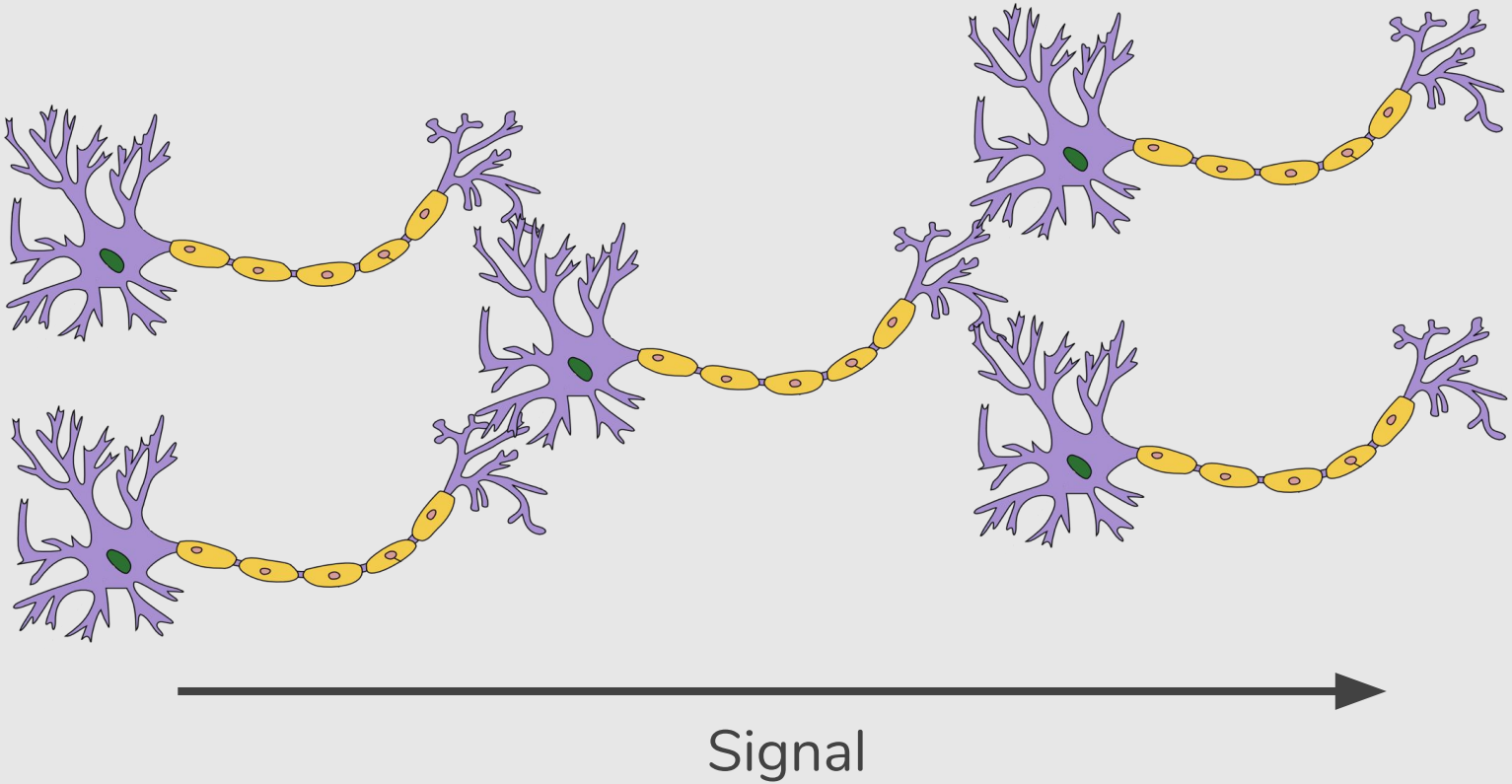


Image Credit: https://en.wikipedia.org/wiki/Neuron

https://www.youtube.com/watch?v=A9Xru1ReRwc



http://www.jeremyjordan.me/content/images/2017/06/Neuron_Hand-tuned.png



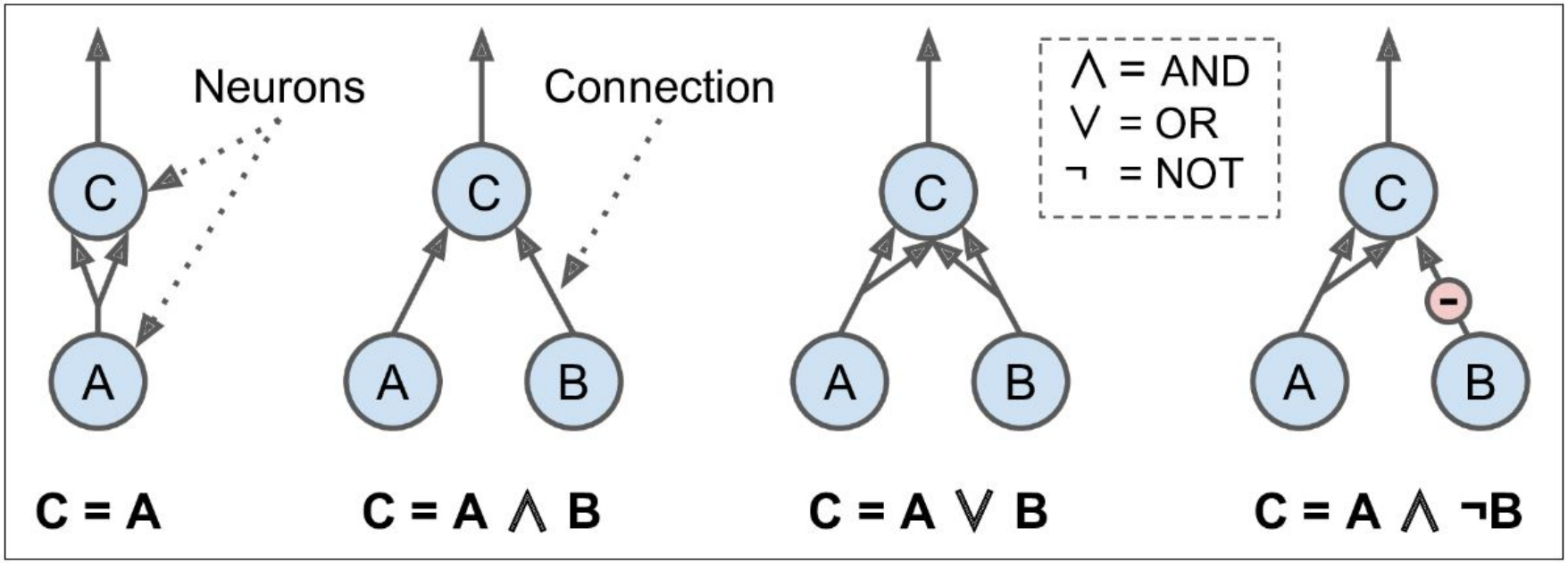
http://www.jeremyjordan.me/content/images/2017/06/Neuron_Hand-tuned.png

Logical Computations with Neurons

Logical Computations with Neurons

McCulloch and Pitts (1943) proposed a very simple model:

- It has one or more binary (on/off) inputs and one binary output.
- The artificial neuron **simply** activates its output when more than a certain number of its inputs are active.



Artificial Neural Networks performing simple logical computations

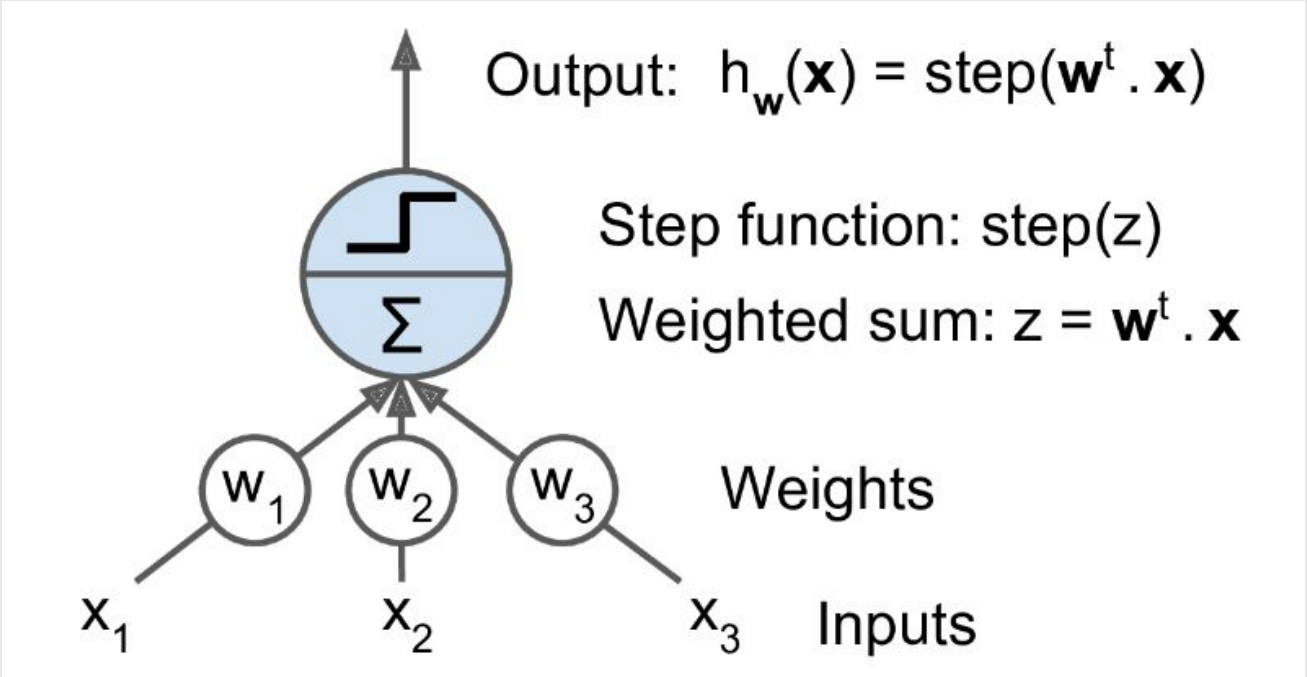
The Perceptron

The Perceptron

Invented in 1957 by Frank Rosenblatt.

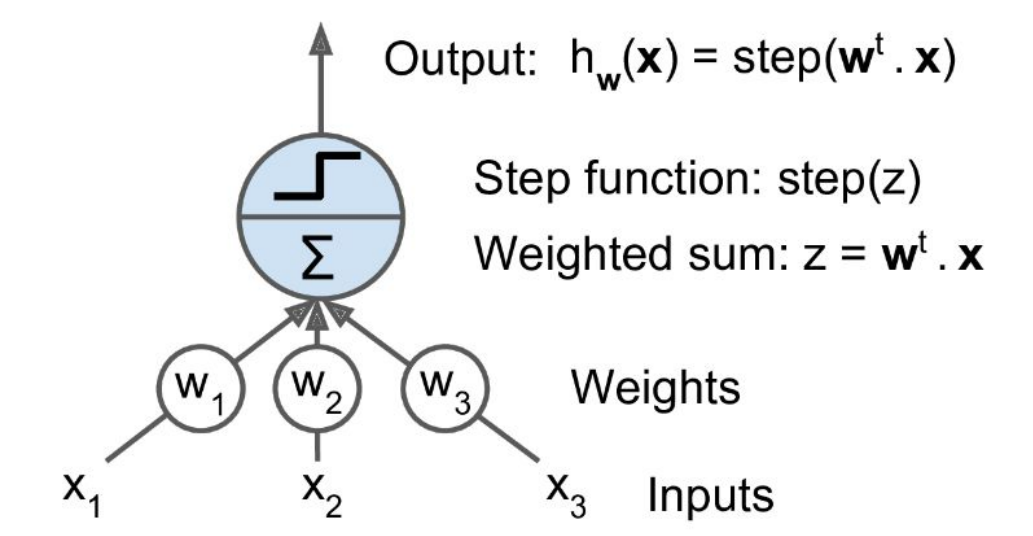
- It is based on a Linear Threshold Unit (LTU):
  - The inputs and output are now **numbers** (instead of binary on/off values) and each input connection is associated with a **weight**.
- The LTU computes a weighted sum of its inputs then it applies a step function to that sum and outputs the result.

The Perceptron



Linear Threshold Unit

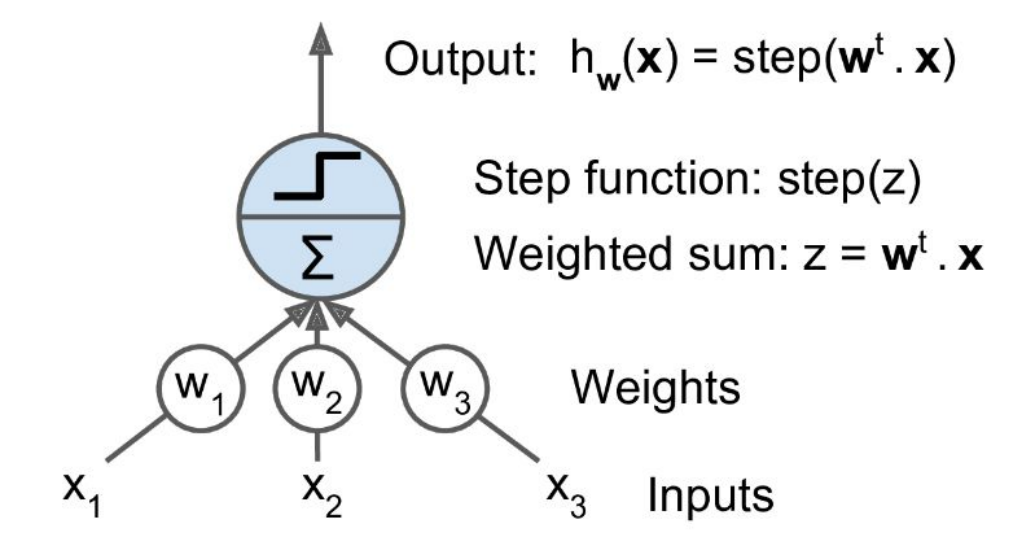
The Perceptron



Linear Threshold Unit

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

The Perceptron

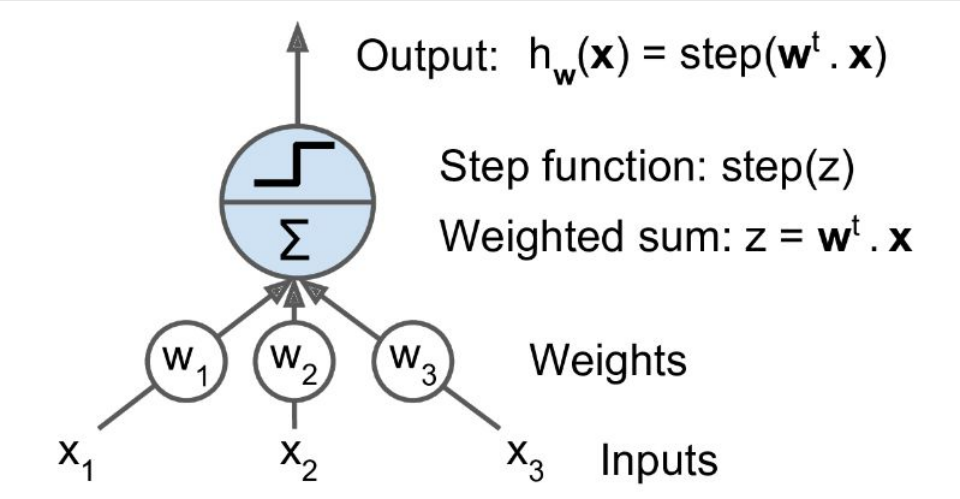


Linear Threshold Unit

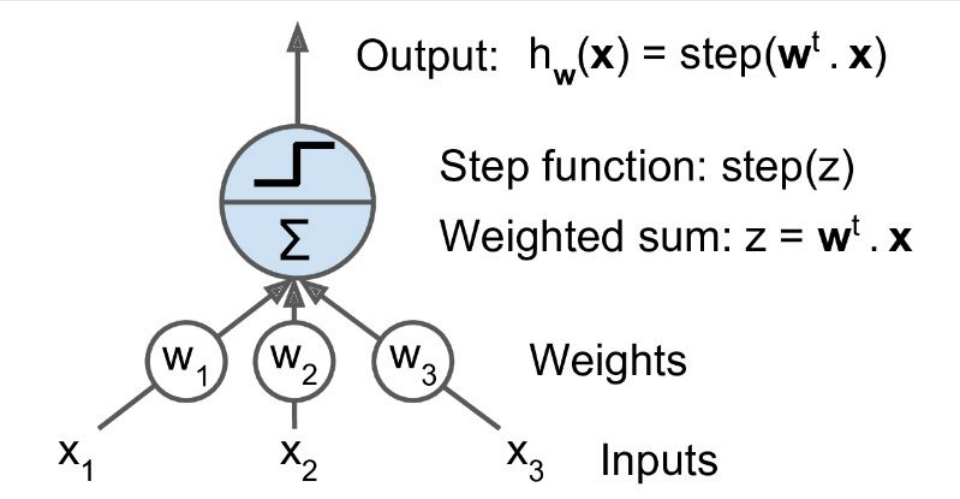
$$\text{heaviside}(z) = \begin{cases} 0 & \text{if } z < 0 \\ 1 & \text{if } z \geq 0 \end{cases}$$

$$\text{sign}(z) = \begin{cases} -1 & \text{if } z < 0 \\ 0 & \text{if } z = 0 \\ +1 & \text{if } z > 0 \end{cases}$$

Neuron Model: Logistic Unit



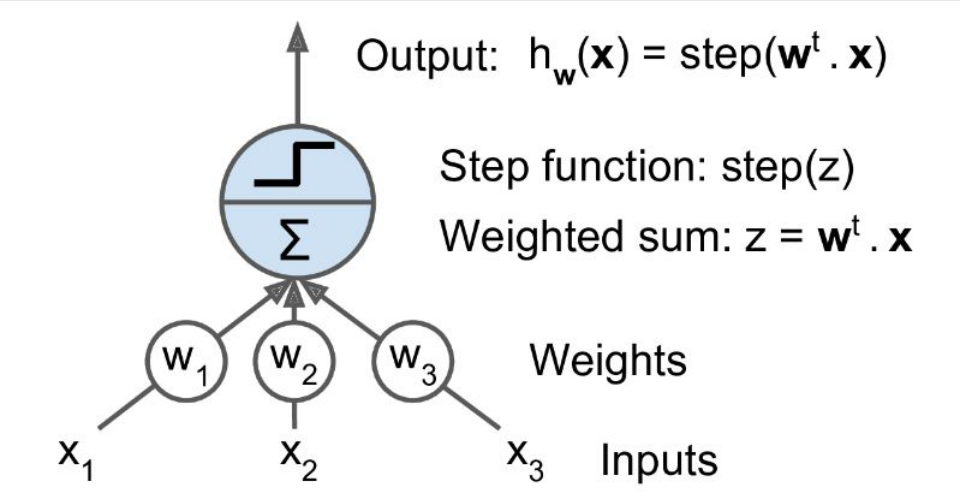
Neuron Model: Logistic Unit

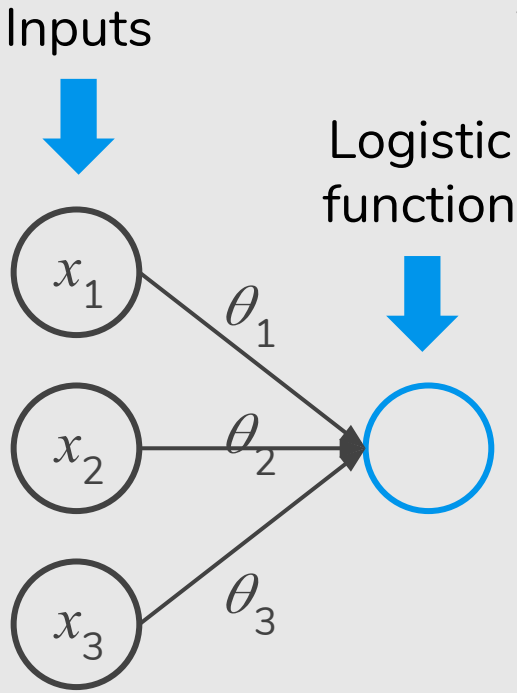


Inputs

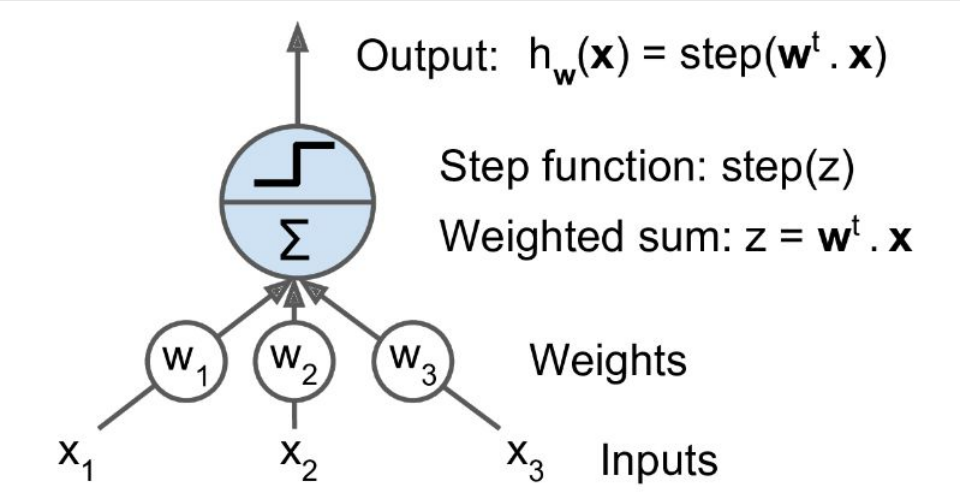
Neuron Model: Logistic Unit

$$\frac{1}{1+e^{-\theta^T x}}$$

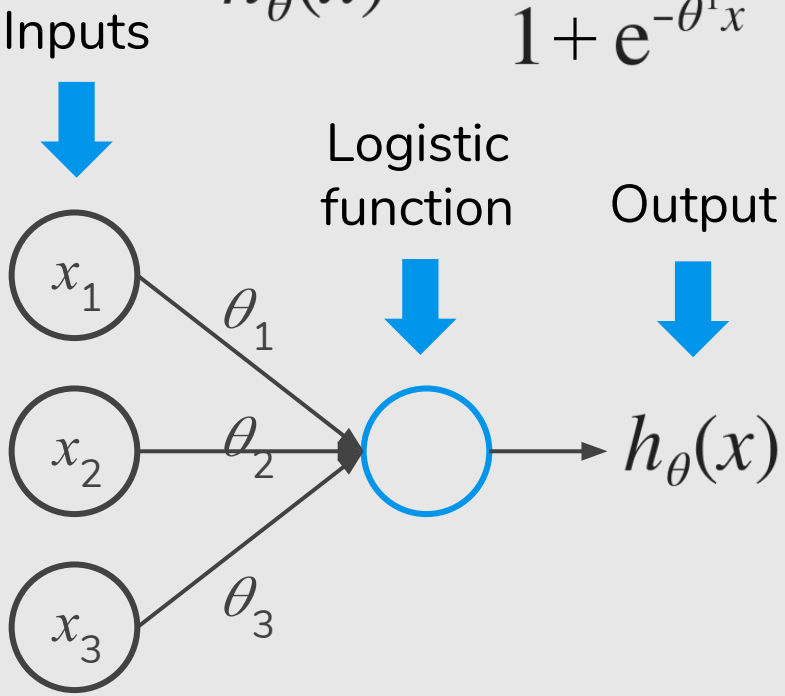




Neuron Model: Logistic Unit

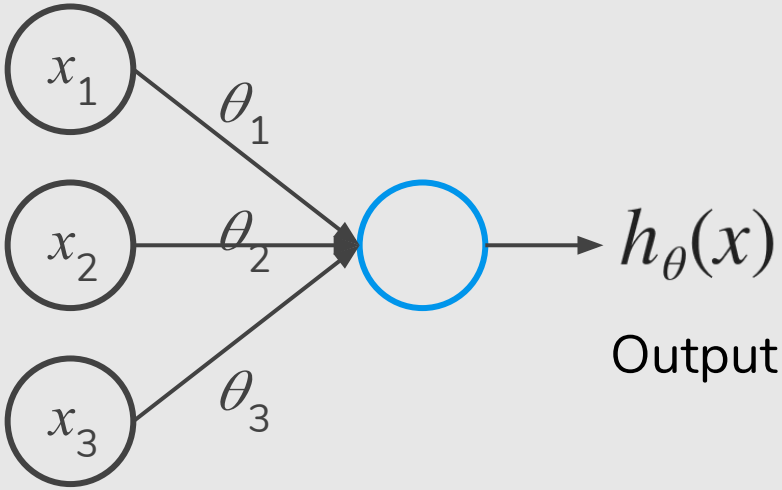


$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$



Neuron Model: Logistic Unit

weights



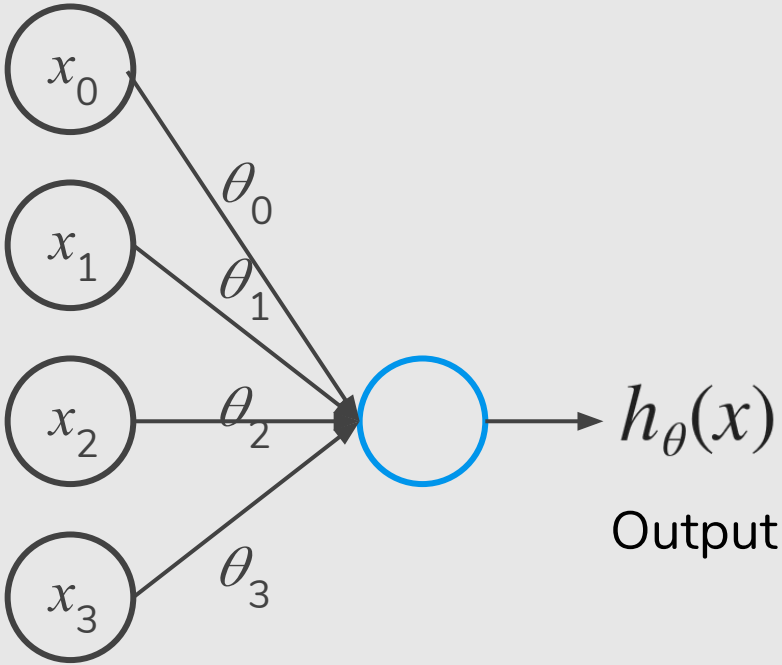
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

Inputs

Sigmoid (Logistic)
activation function

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Neuron Model: Logistic Unit



weights

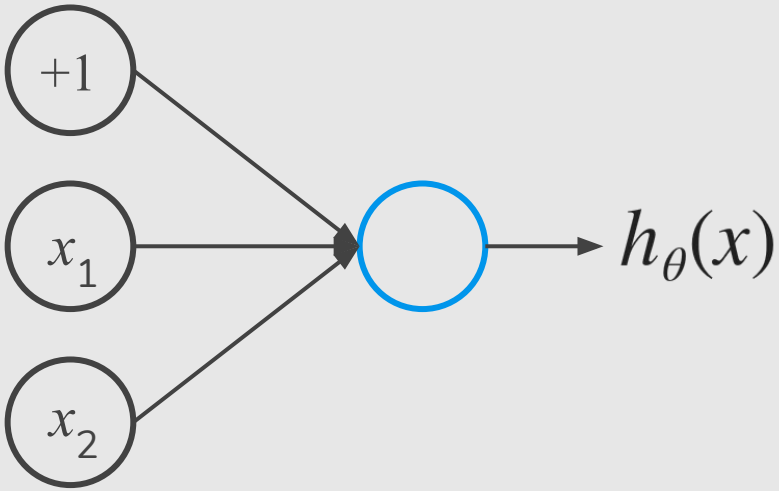
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Examples

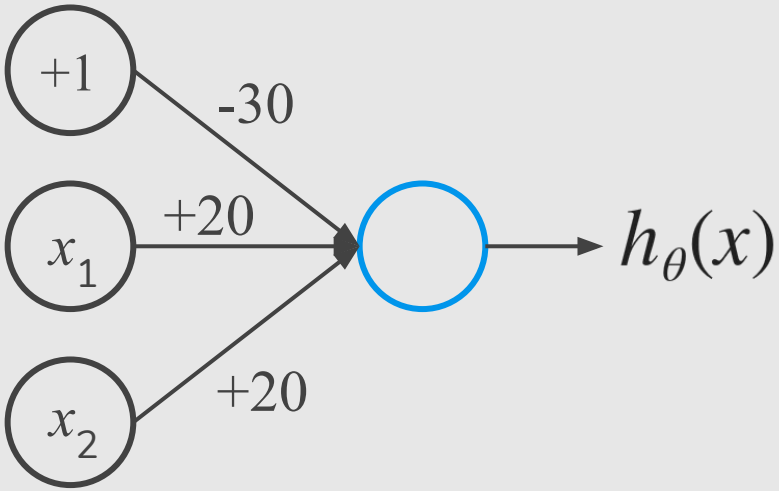
Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1 \text{ AND } x_2$



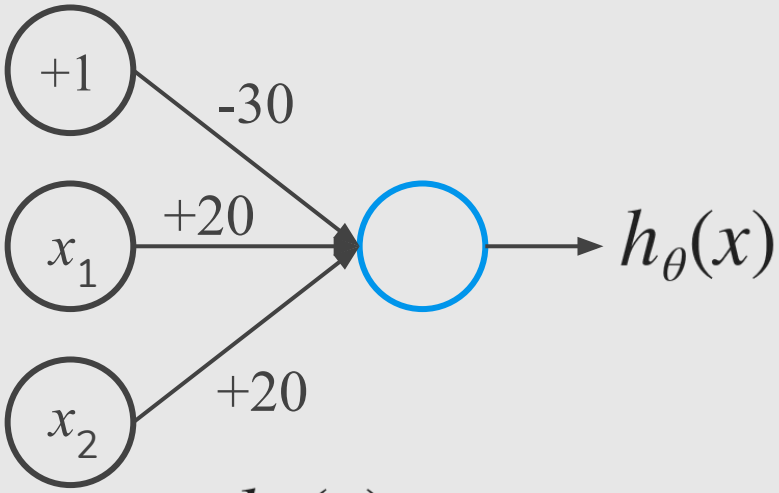
Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1 \text{ AND } x_2$



Simple Example: AND

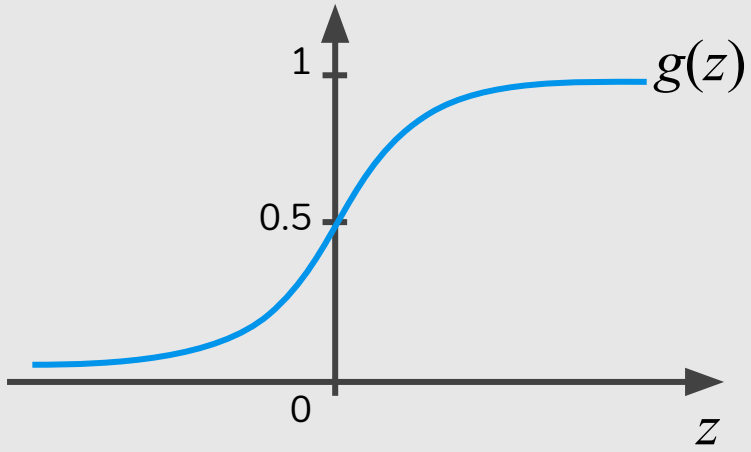
$x_1, x_2 \in \{0,1\}$ $\quad y = x_1 \text{ AND } x_2$

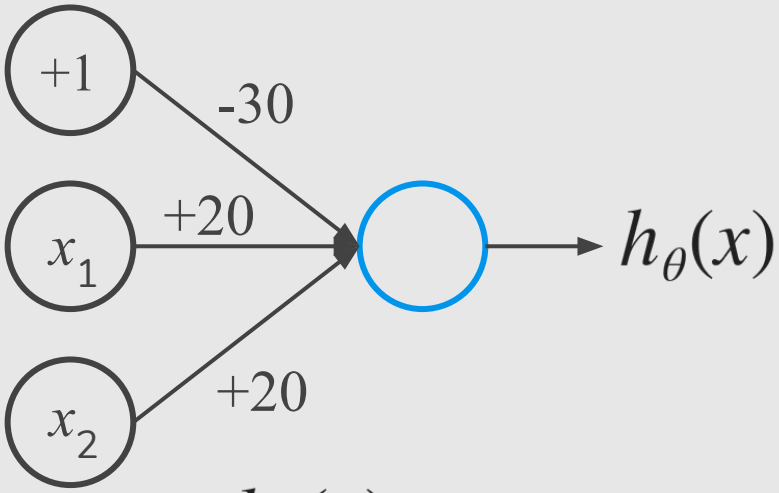


$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ AND x_2



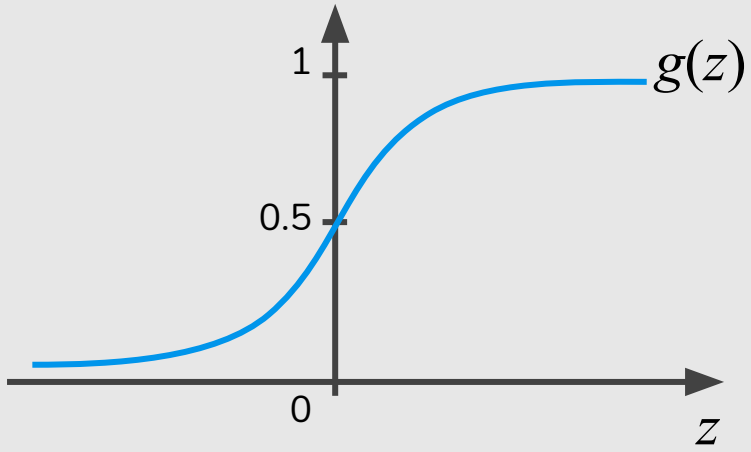


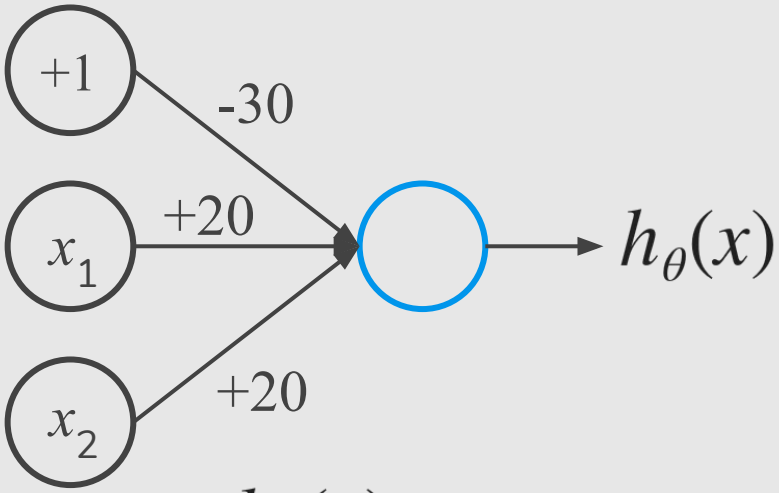
x_1	x_2	$h_\theta(x)$
0	0	
0	1	
1	0	
1	1	

$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ AND x_2



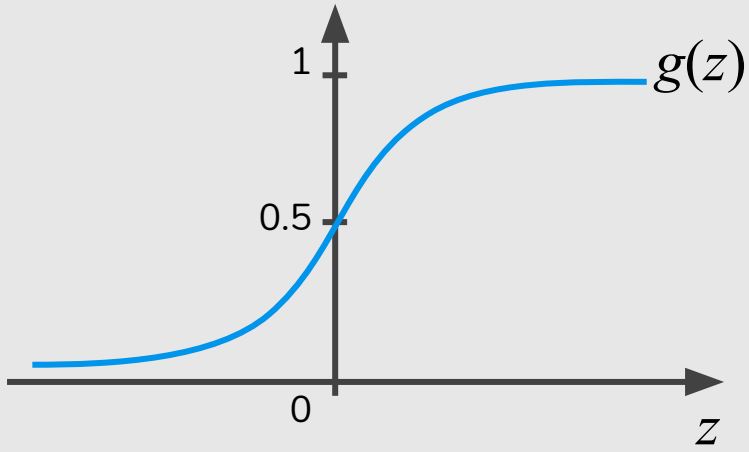


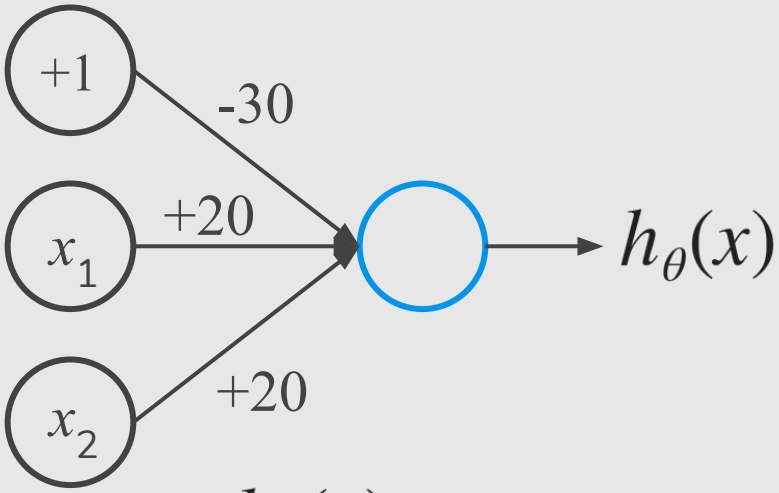
x_1	x_2	$h_\theta(x)$
0	0	$g(-30) \approx 0$
0	1	
1	0	
1	1	

$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ AND x_2



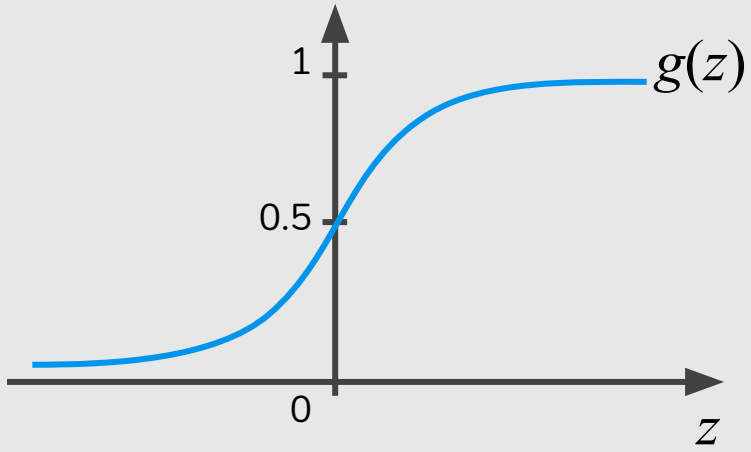


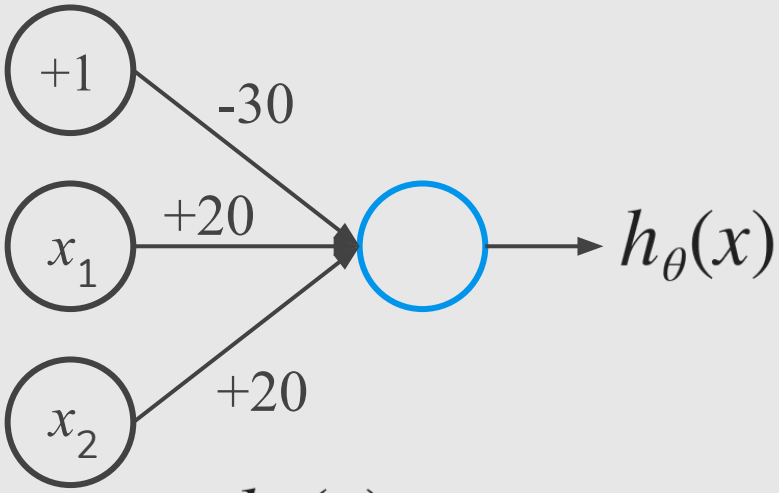
x_1	x_2	$h_\theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	
1	1	

$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ AND x_2



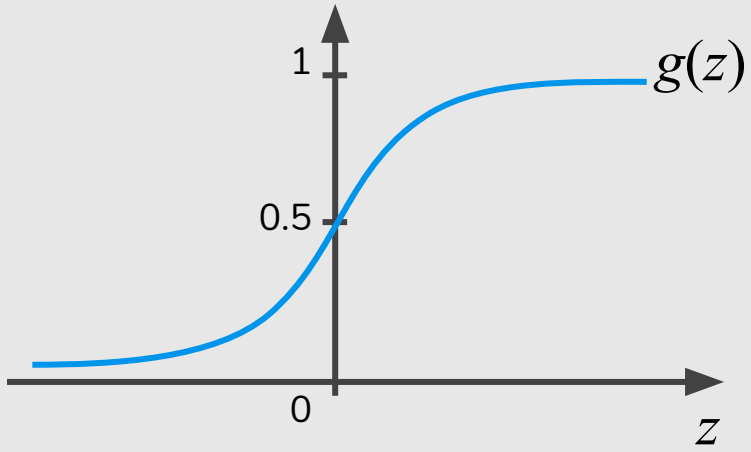


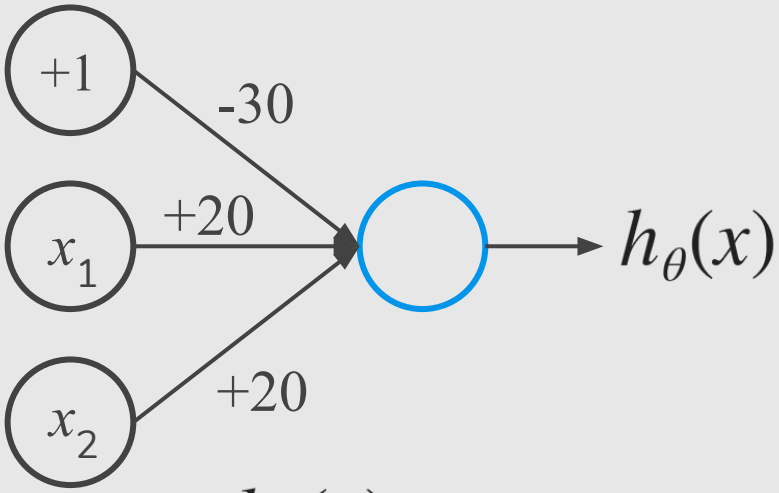
x_1	x_2	$h_\theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	

$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

Simple Example: AND

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ AND x_2



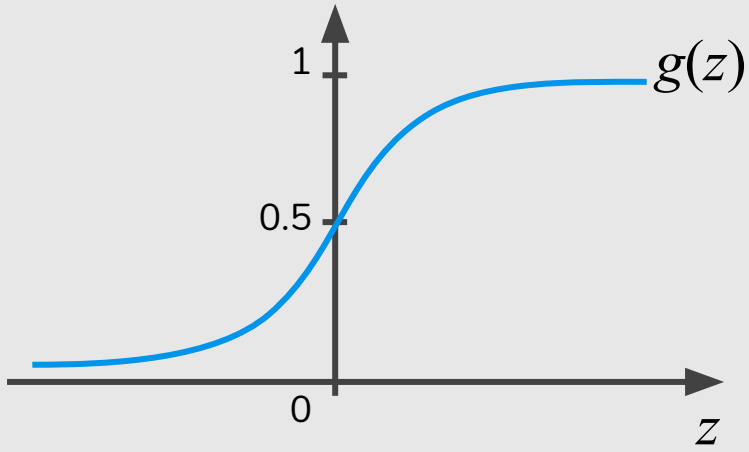


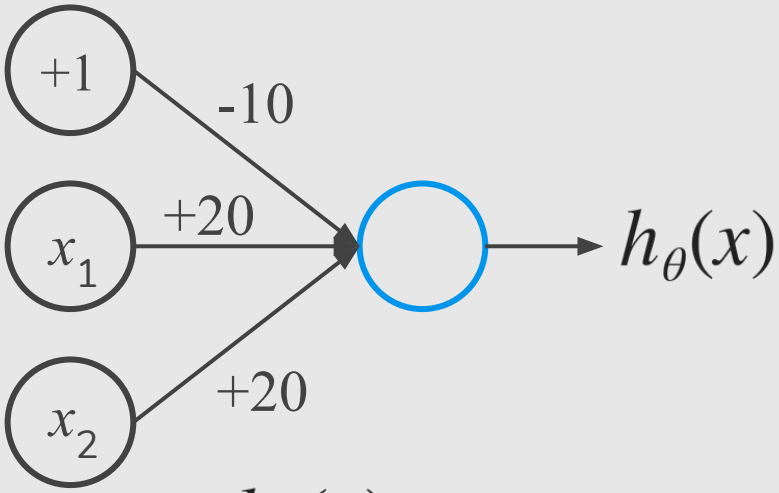
$$h_\theta(x) = g(-30 + 20x_1 + 20x_2)$$

x_1	x_2	$h_\theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

Simple Example: OR

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ OR x_2



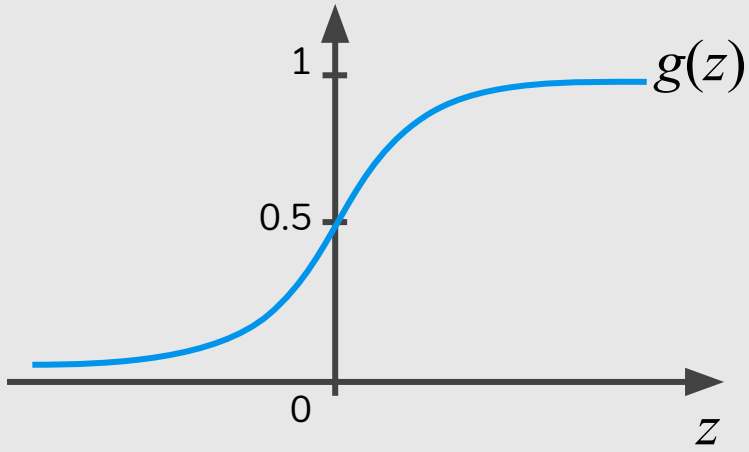


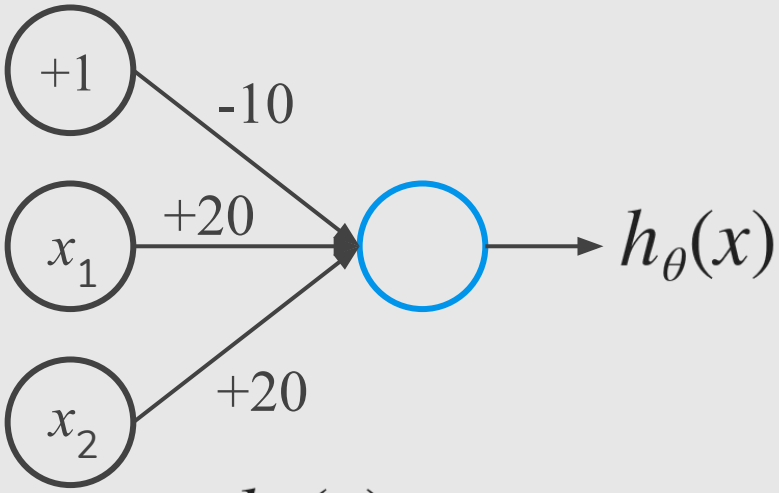
x_1	x_2	$h_\theta(x)$
0	0	
0	1	
1	0	
1	1	

$$h_\theta(x) = g(-10 + 20x_1 + 20x_2)$$

Simple Example: OR

$x_1, x_2 \in \{0,1\}$ $\quad y = x_1$ OR x_2





$$h_\theta(x) = g(-10 + 20x_1 + 20x_2)$$

x_1	x_2	$h_\theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	$g(10) \approx 1$
1	1	$g(30) \approx 1$

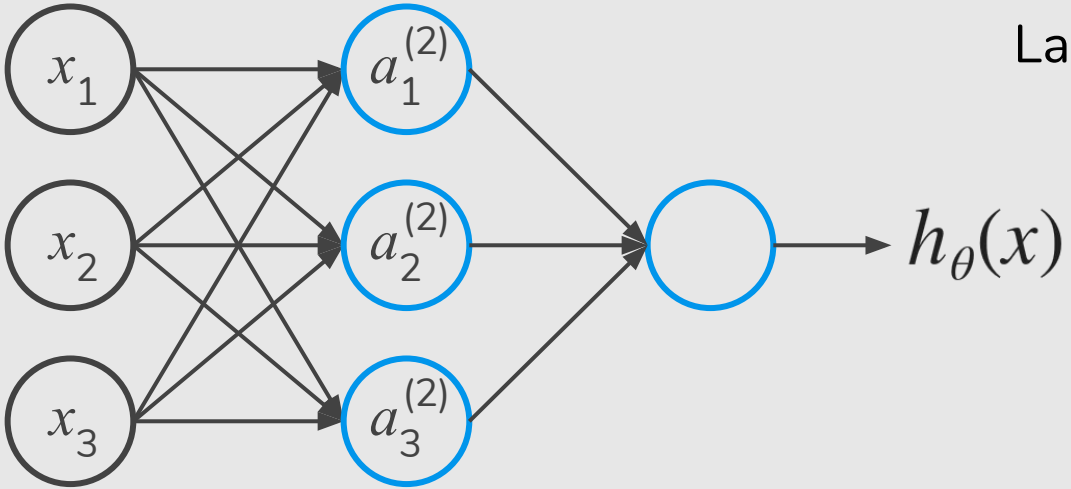
Neural Network

Neural Network

Layer 1 = Input layer

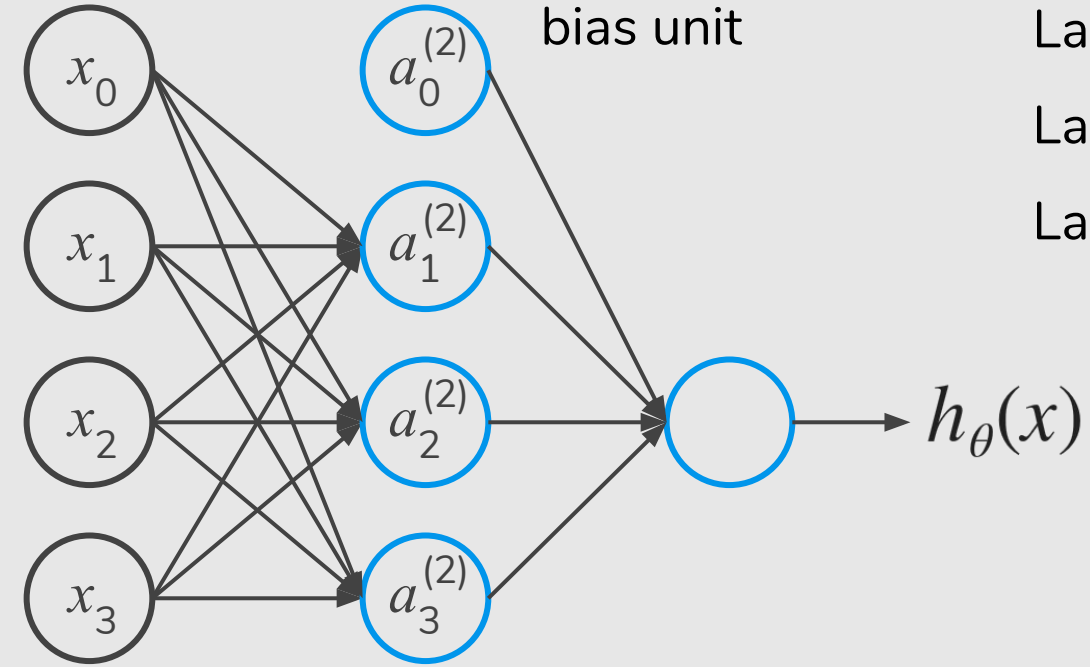
Layer 2 = Hidden layer

Layer 3 = Output layer



Layer 1 Layer 2 Layer 3

Neural Network

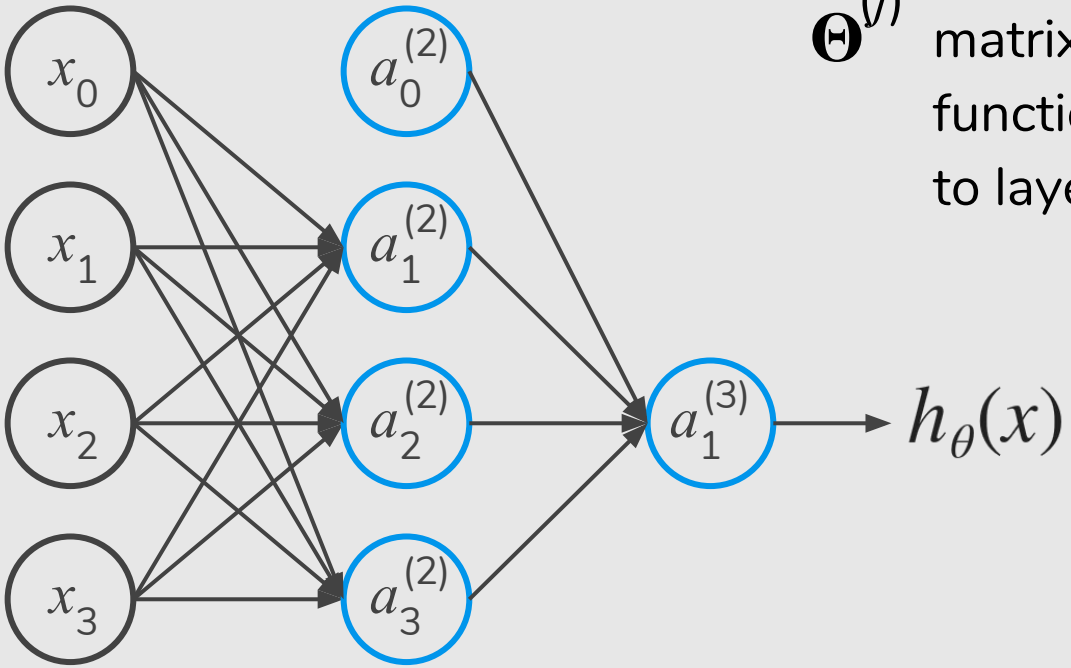


Layer 1 = Input layer

Layer 2 = Hidden layer

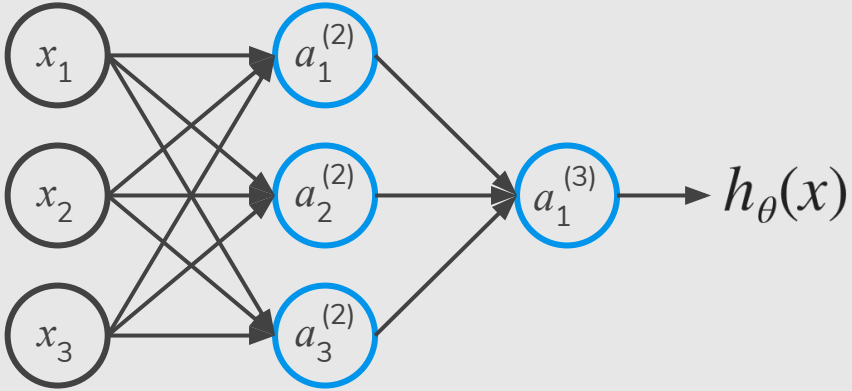
Layer 3 = Output layer

Neural Network



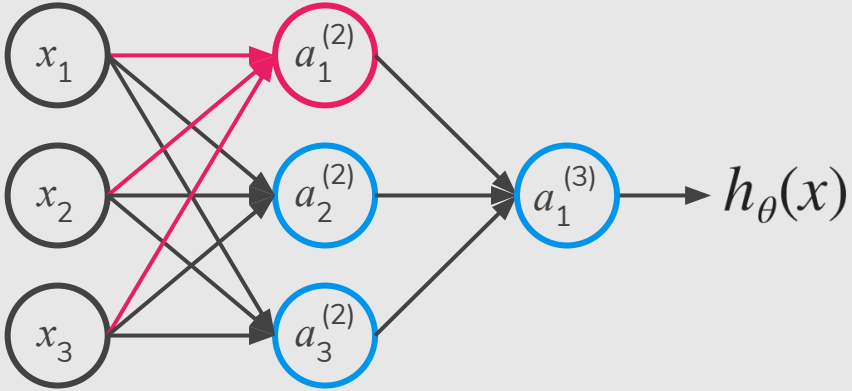
$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$



$a_i^{(j)}$ “activation” of unit i in layer j

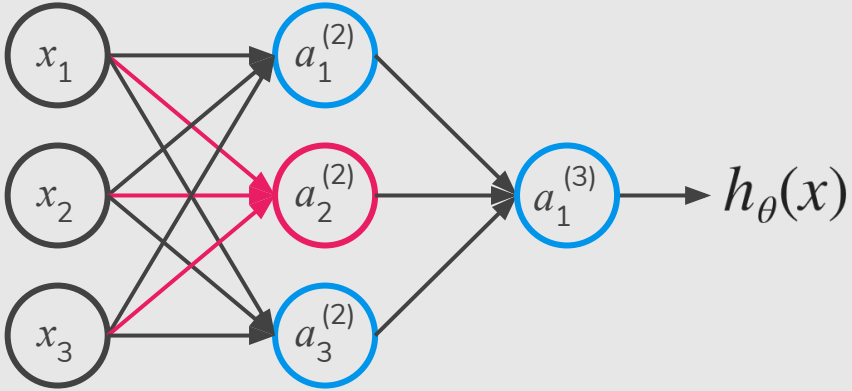
$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$



$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

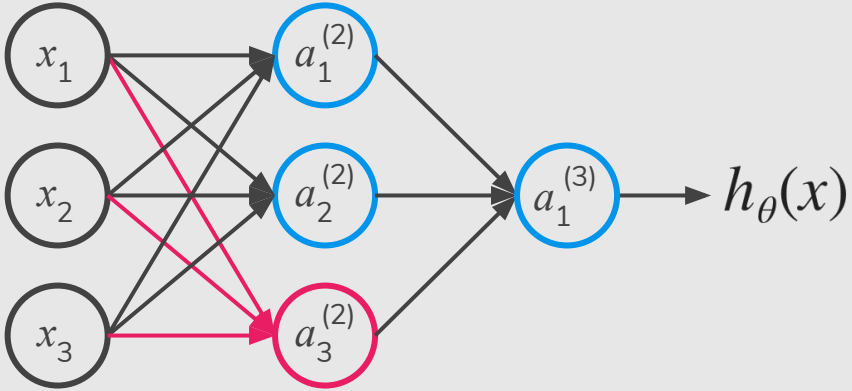


$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$



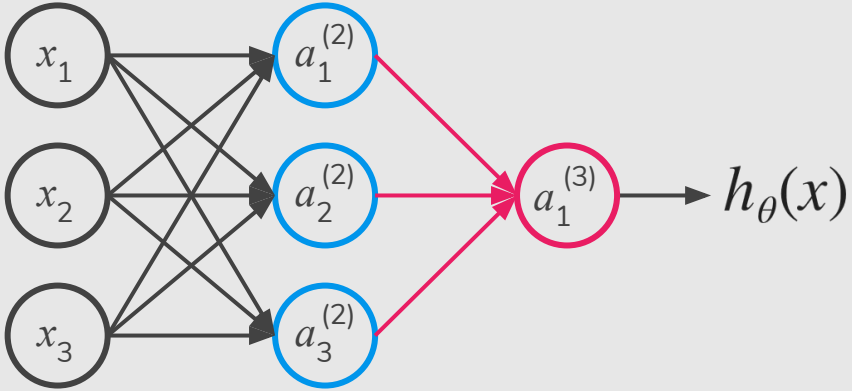
$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$



$a_i^{(j)}$ "activation" of unit i in layer j

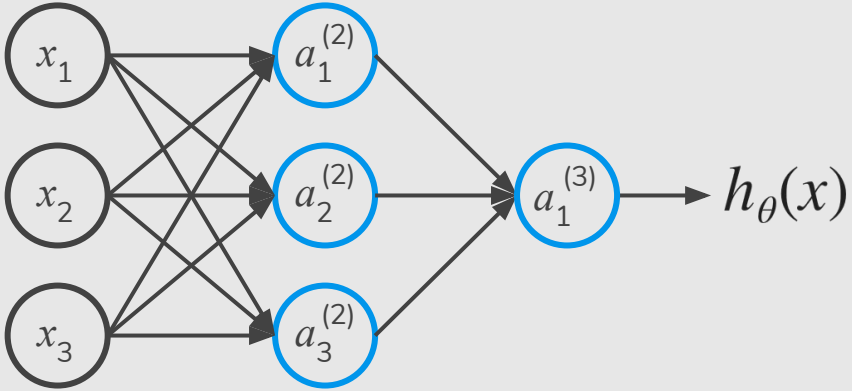
$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$



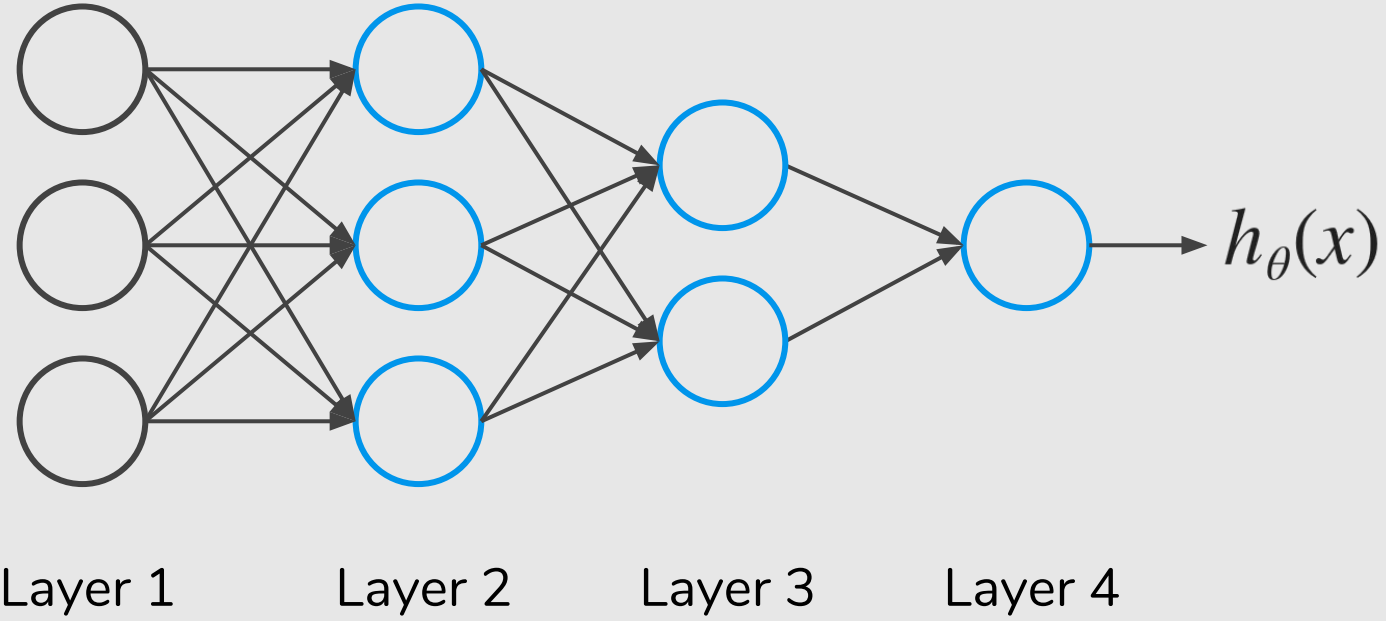
$a_i^{(j)}$ "activation" of unit i in layer j

$\Theta^{(j)}$ matrix of weights controlling function mapping from layer j to layer $j+1$

Feedforward Neural Network (forward propagating)

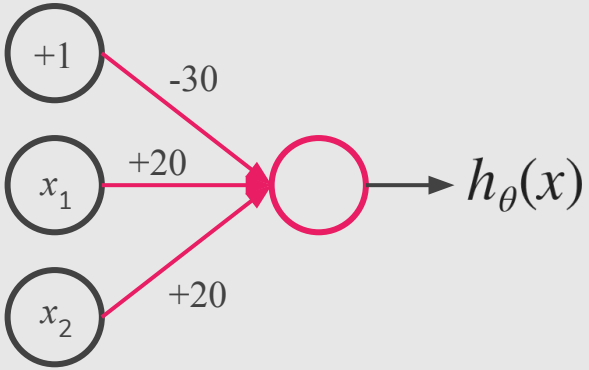
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

Other Network Architectures

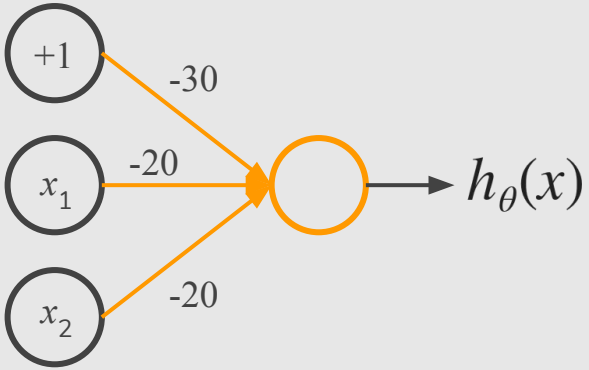


Example: XNOR $x_1, x_2 \in \{0,1\}$ $y = x_1 \text{ XNOR } x_2$

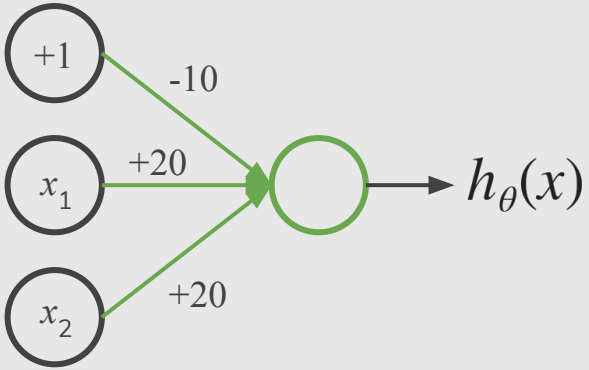
Example: XNOR $x_1, x_2 \in \{0,1\}$ $y = x_1$ XNOR x_2



x_1 AND x_2

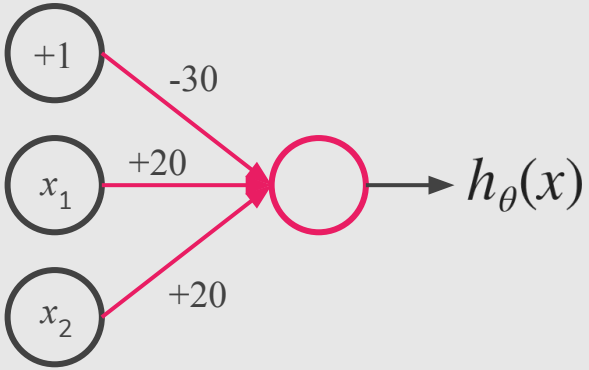


(NOT x_1) AND (NOT x_2)

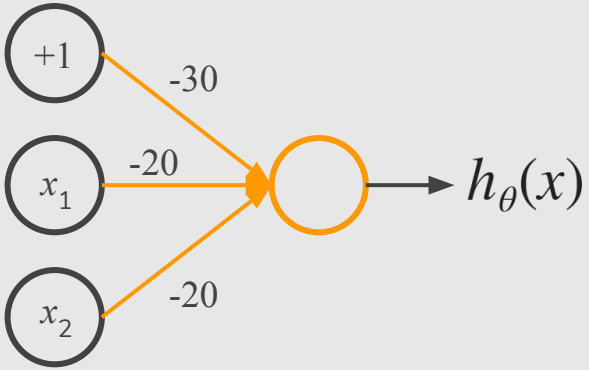


x_1 OR x_2

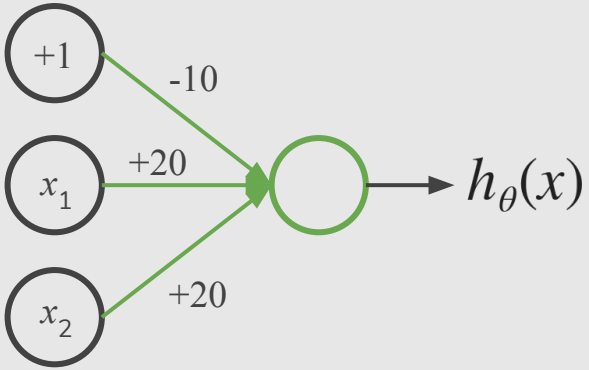
Example: XNOR $x_1, x_2 \in \{0,1\}$ $y = x_1$ XNOR x_2



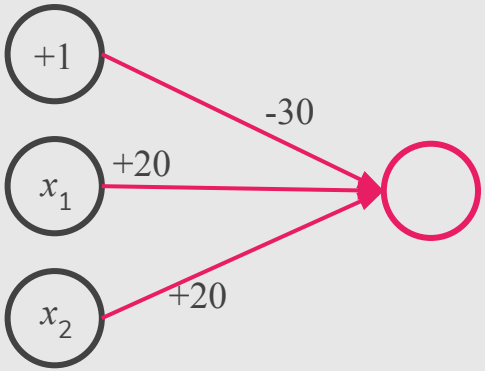
x_1 AND x_2



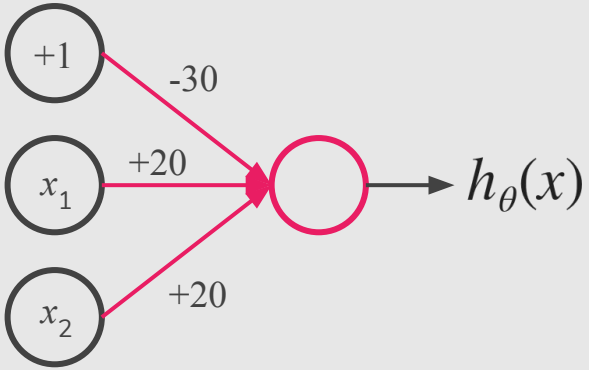
(NOT x_1) AND (NOT x_2)



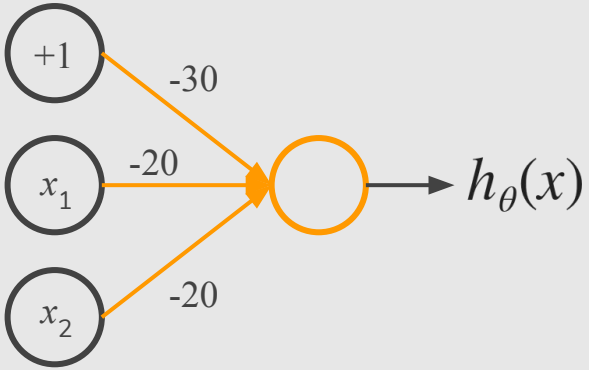
x_1 OR x_2



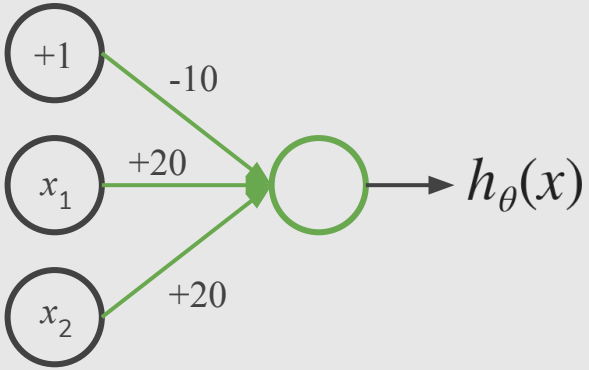
Example: XNOR $x_1, x_2 \in \{0,1\}$ $y = x_1$ XNOR x_2



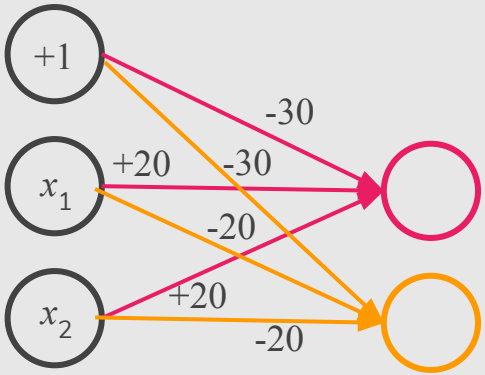
x_1 AND x_2



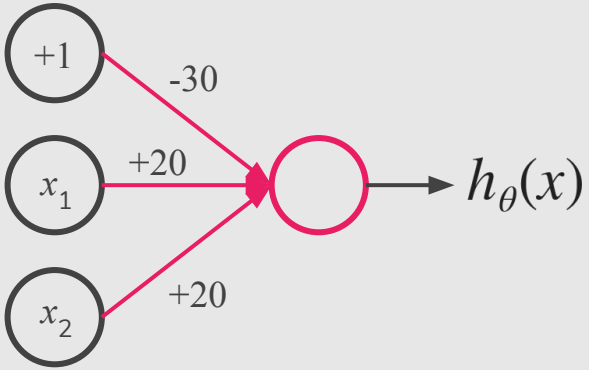
(NOT x_1) AND (NOT x_2)



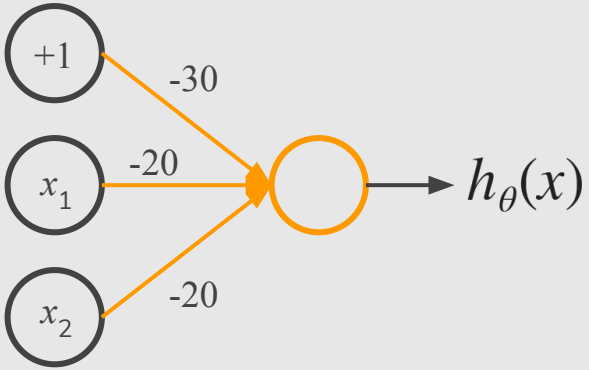
x_1 OR x_2



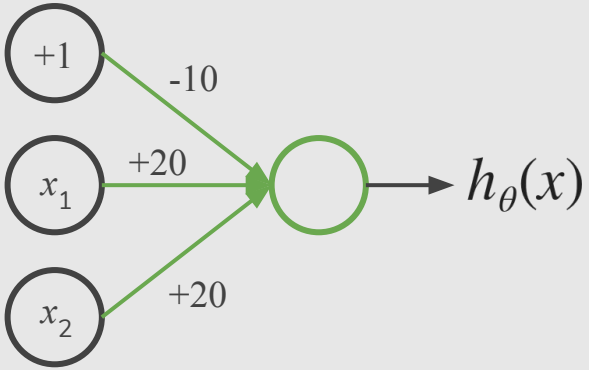
Example: XNOR $x_1, x_2 \in \{0,1\}$ $y = x_1$ XNOR x_2



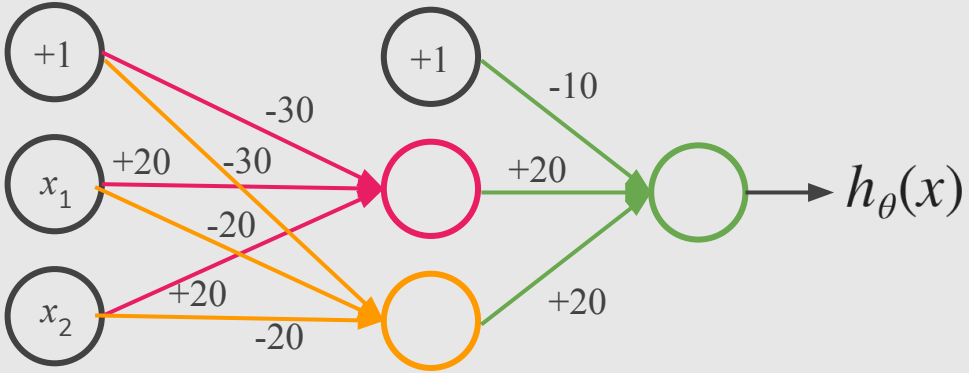
x_1 AND x_2



(NOT x_1) AND (NOT x_2)

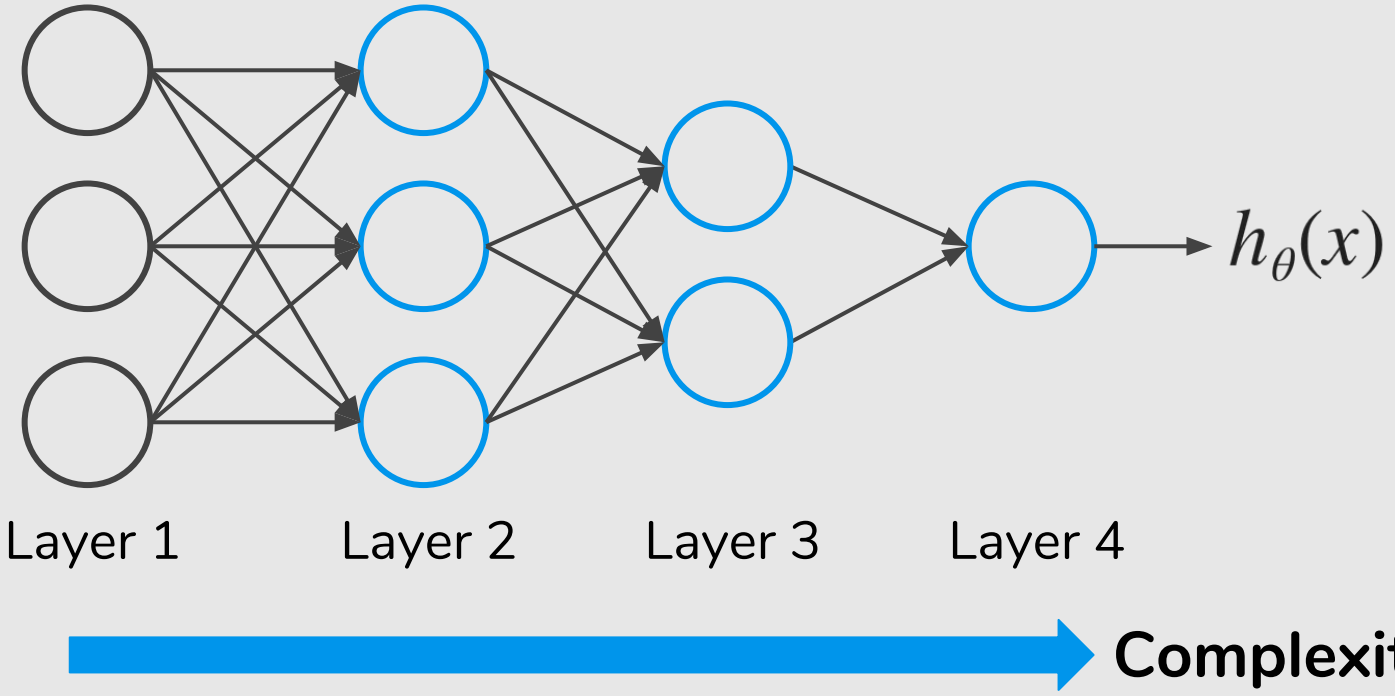


x_1 OR x_2



x_1	x_2	$h_\theta(x)$
0	0	1
0	1	0
1	0	0
1	1	1

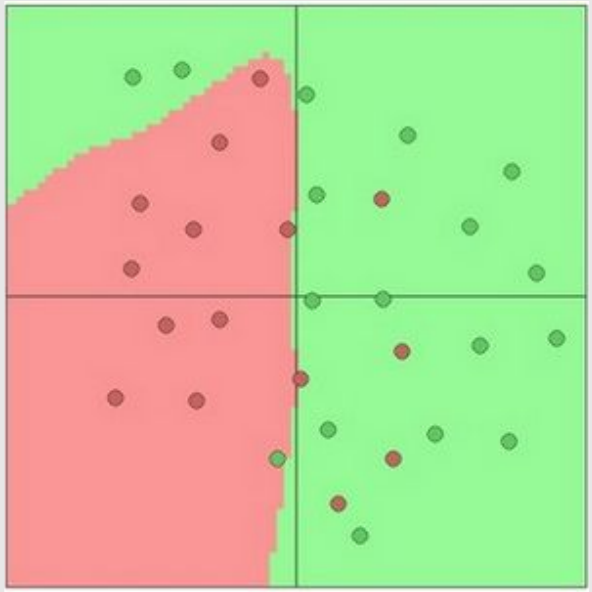
Neural Network Intuition



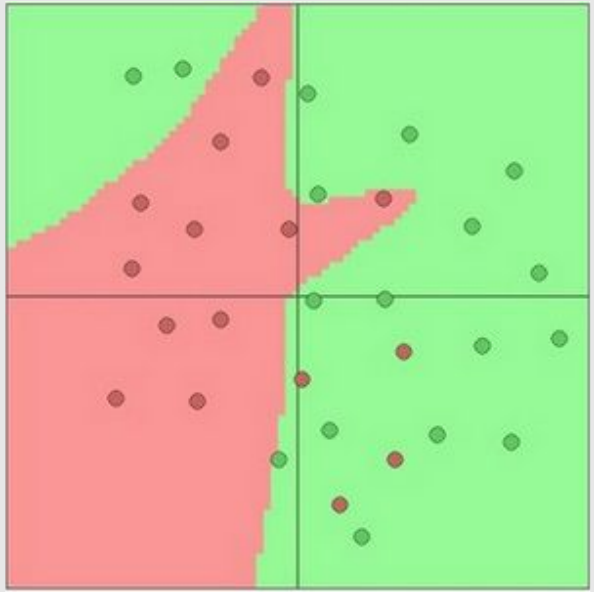
Neural Network Intuition

Toy 2d classification with 2-layer neural network

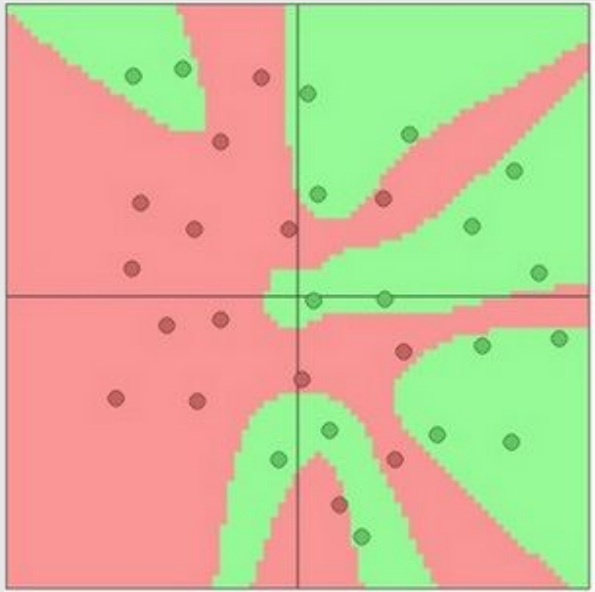
http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html



3 hidden neurons

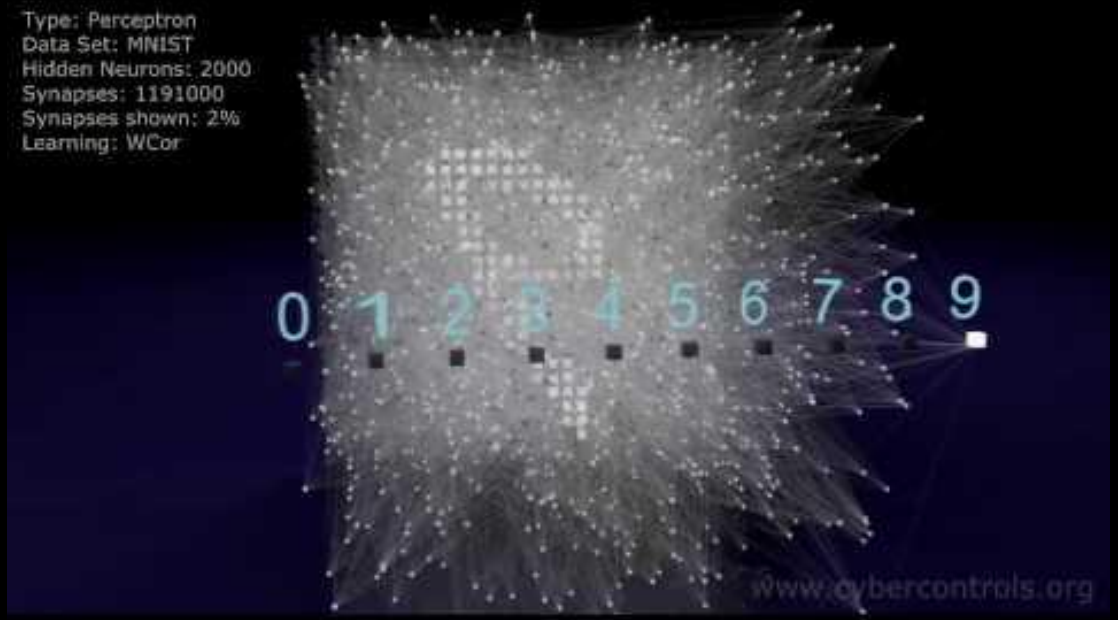


6 hidden neurons



20 hidden neurons

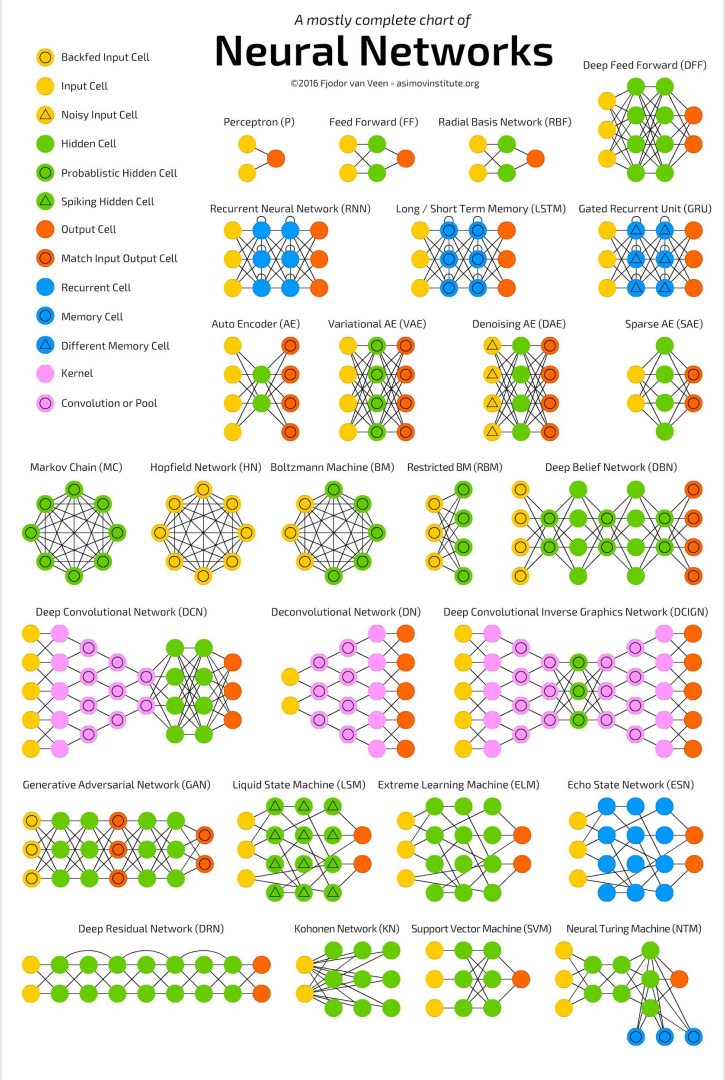
http://cs231n.github.io/assets/nn1/layer_sizes.jpeg



https://youtu.be/3JQ3hYko51Y

Neural Network Zoo

http://www.asimovinstitute.org/neural-network-zoo/



To be continued ...

References

Machine Learning Books

- Hands-On Machine Learning with Scikit-Learn and TensorFlow, Chap. 10
- Pattern Recognition and Machine Learning, Chap. 5
- Pattern Classification, Chap. 6
- Free online book: http://neuralnetworksanddeeplearning.com

Machine Learning Courses

- https://www.coursera.org/learn/machine-learning, Week 4 & 5
- https://www.coursera.org/learn/neural-networks