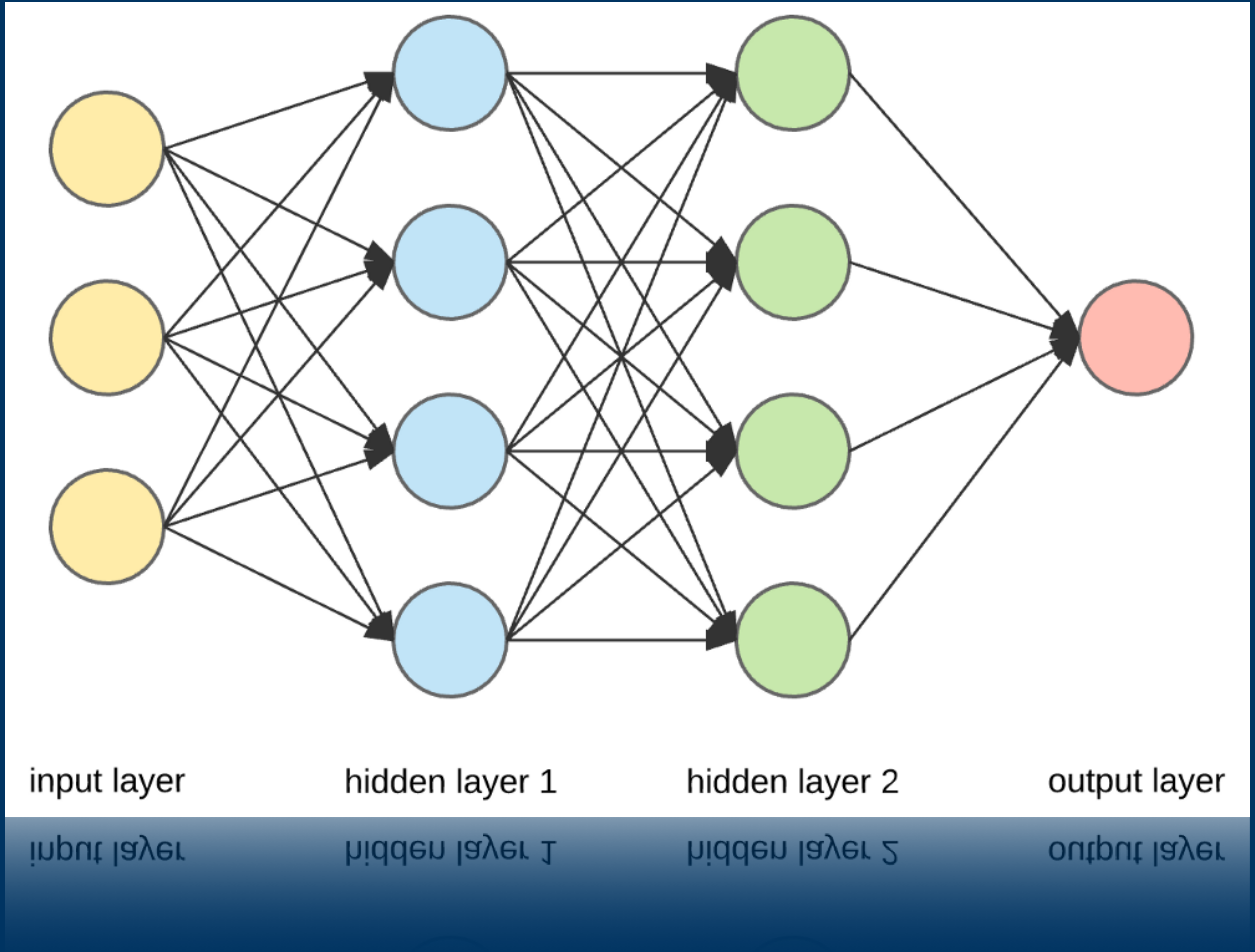


Neural Networks I

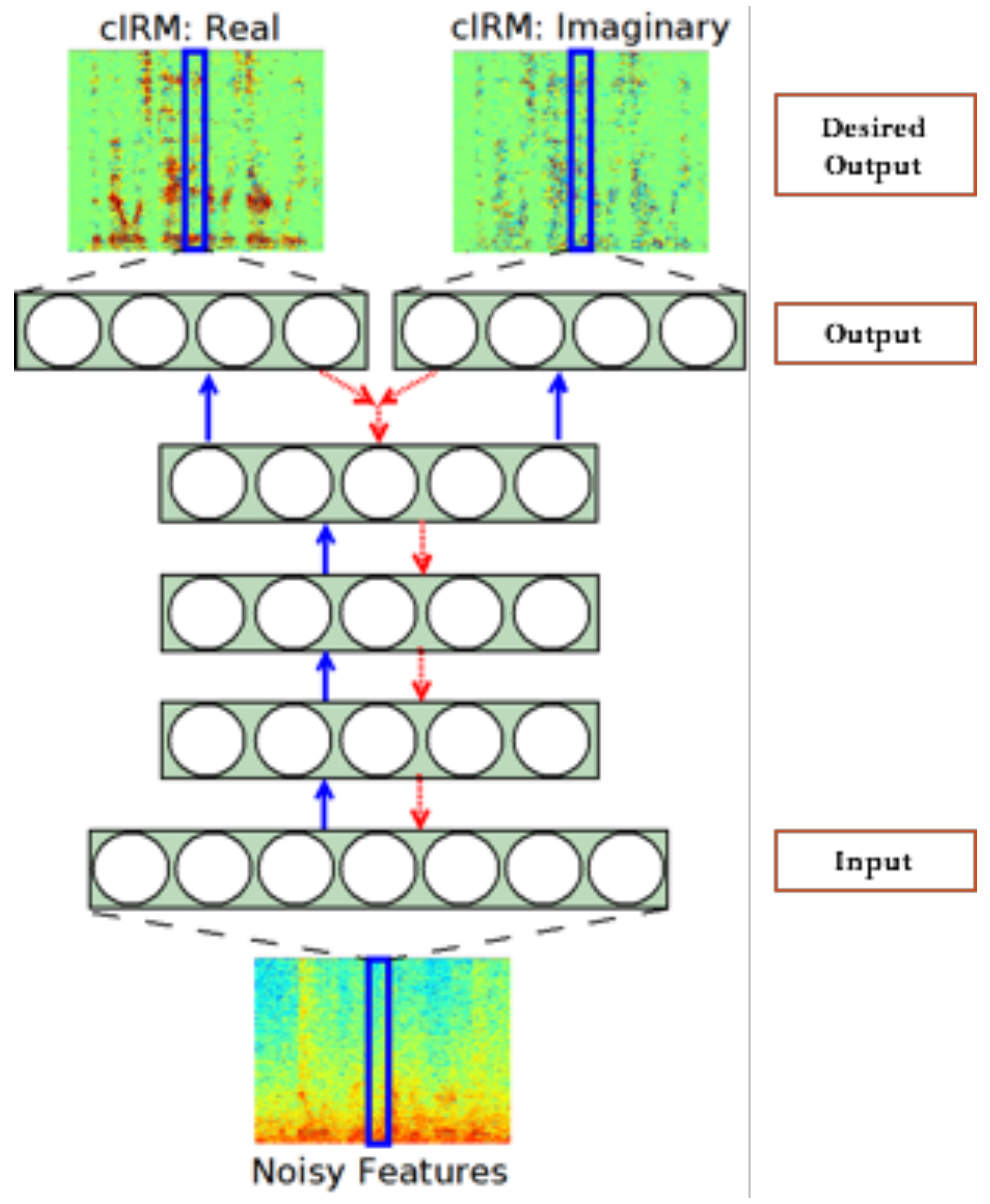
CSCI-P556 Applied Machine Learning
Lecture 13

D.S. Williamson

Agenda and Learning Outcomes

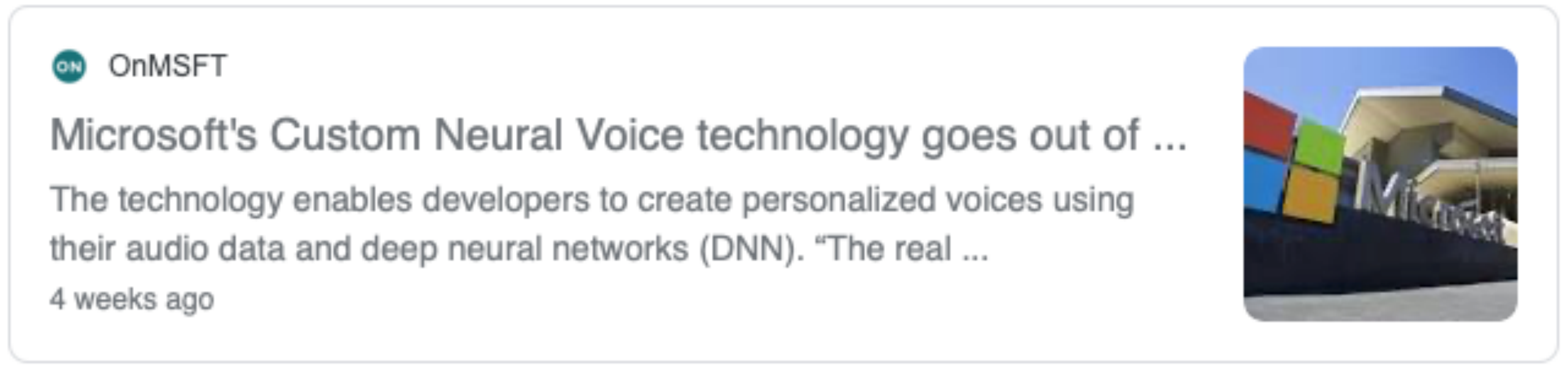
Today's Topic(s)

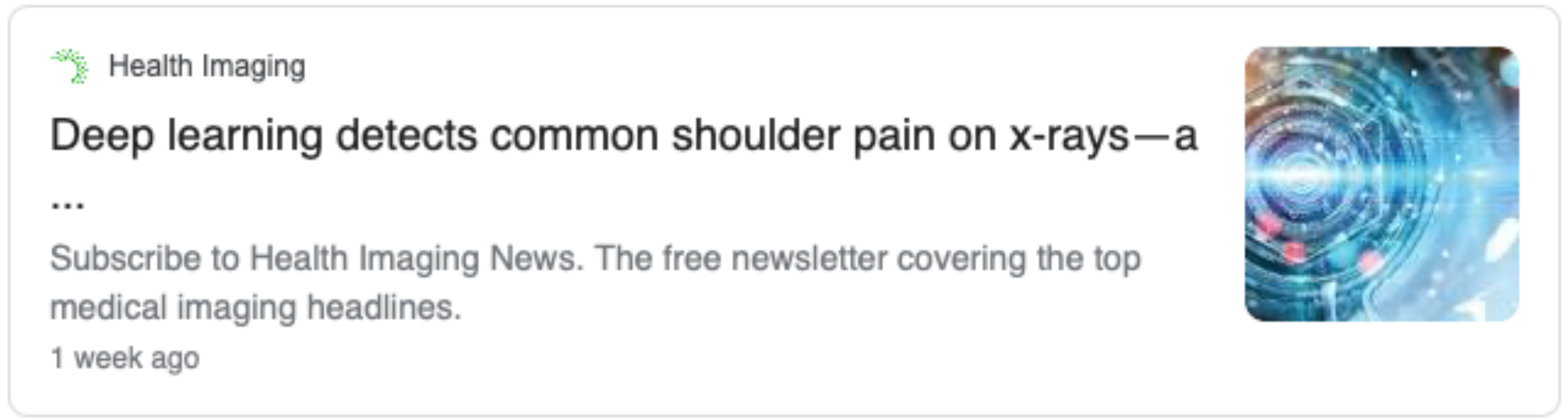
- **Topic(s): Neural Networks**
    - Understand the basics of a real neuron
    - Understand how real neurons inspired artificial neurons
    - Be able to describe the components of different neuron models
    - Understand the components of a neural network
    - Understand the concept of decision boundary and linearly separable
    - Begin to understand the perceptron learning rule
- **Announcements**
    - Homework #2 has been posted
    - Write post to Piazza if looking for group or group members

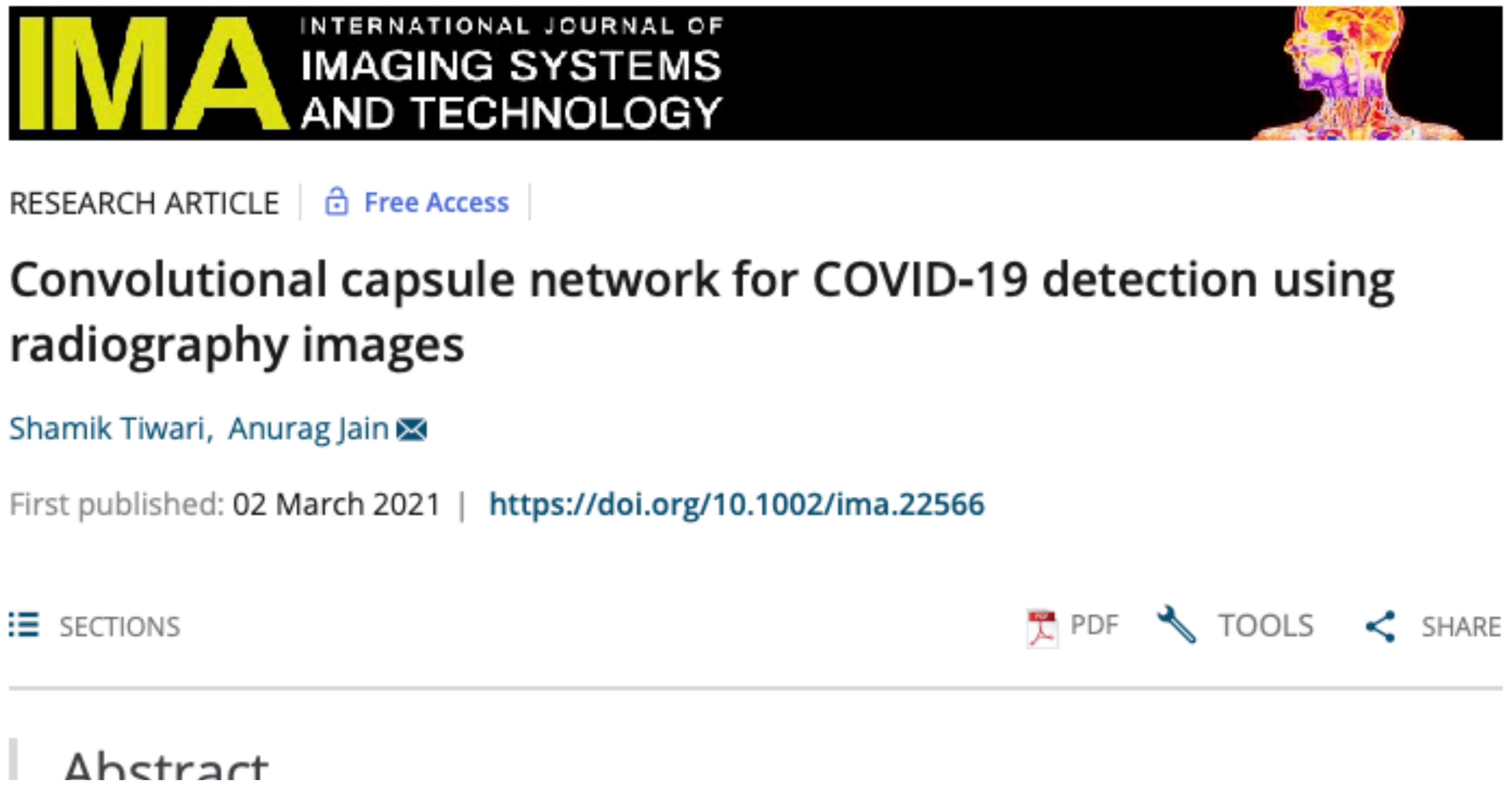


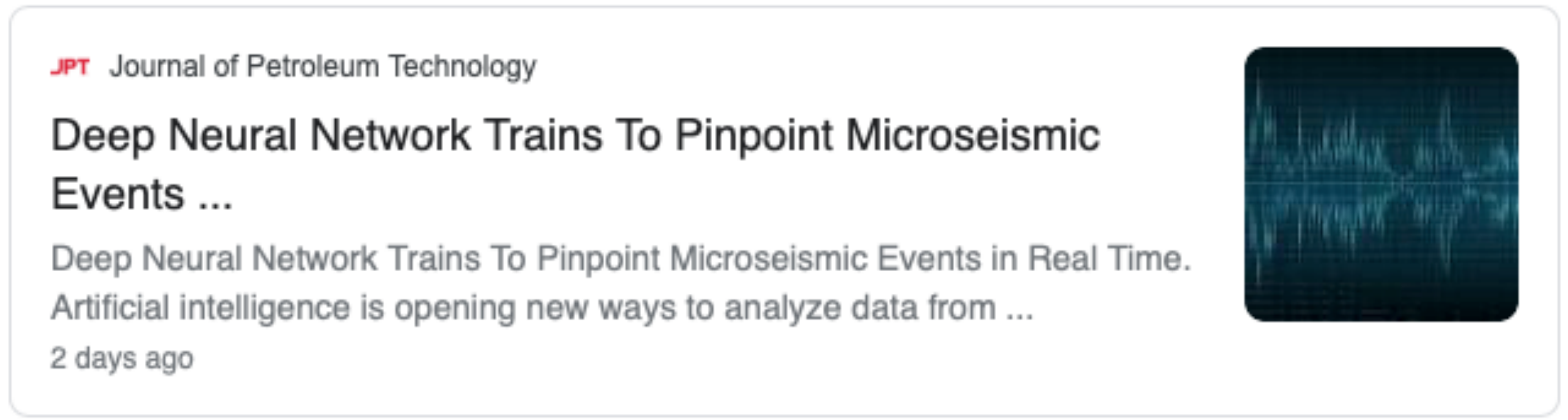
Neural Networks in the Wild

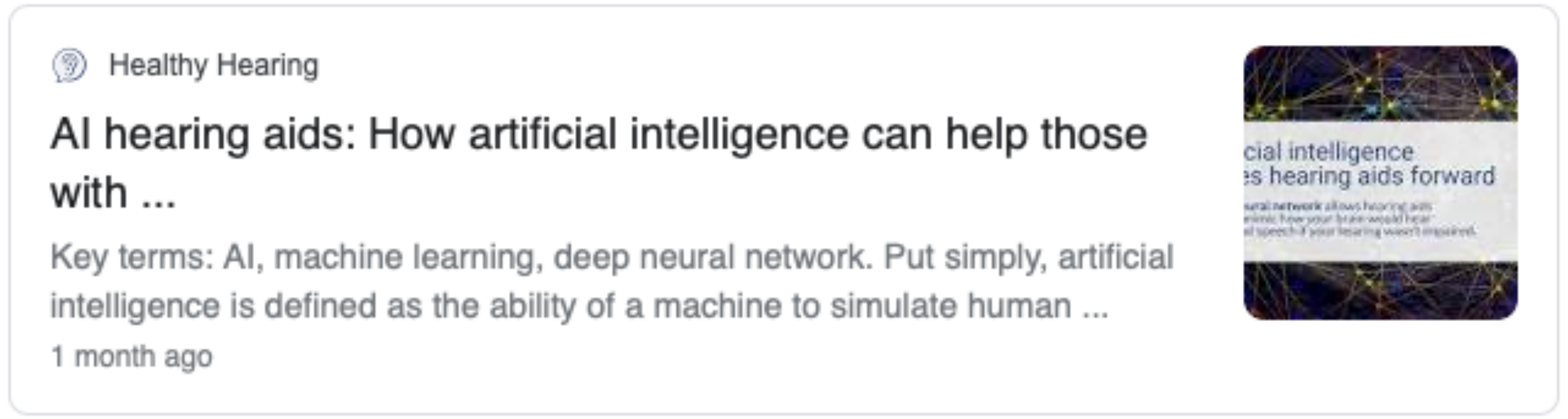
Applications where Neural Networks have been used

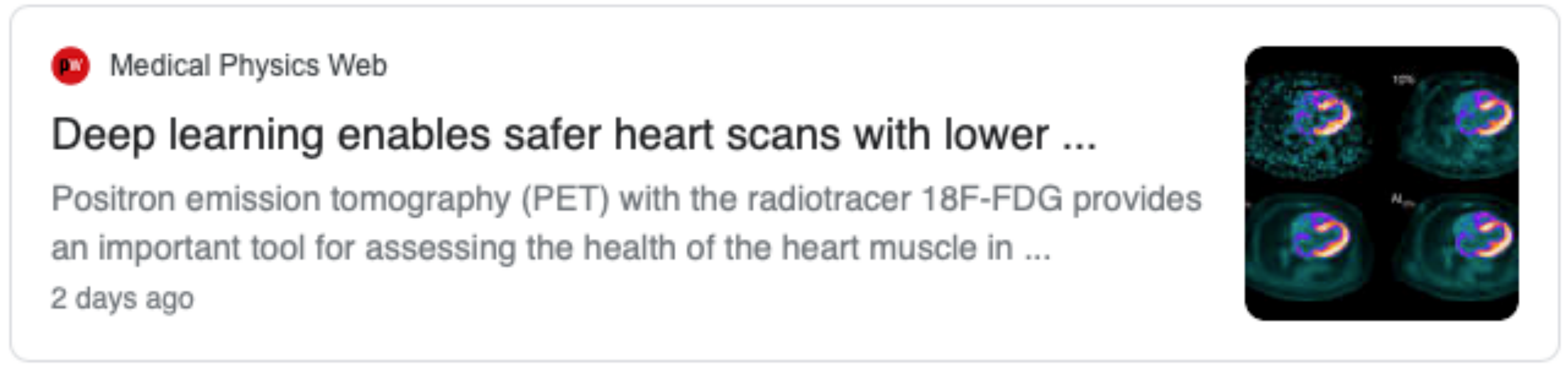

OnMSFT
Microsoft's Custom Neural Voice technology goes out of ...
The technology enables developers to create personalized voices using their audio data and deep neural networks (DNN). "The real ...
4 weeks ago

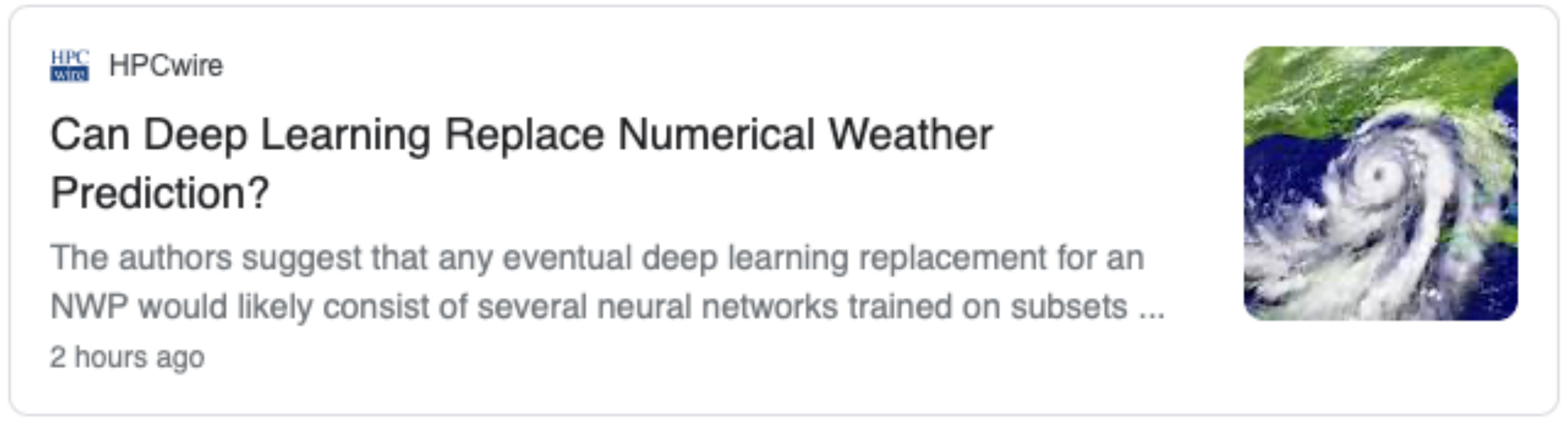

Health Imaging
Deep learning detects common shoulder pain on x-rays—a ...
Subscribe to Health Imaging News. The free newsletter covering the top medical imaging headlines.
1 week ago


IMA INTERNATIONAL JOURNAL OF IMAGING SYSTEMS AND TECHNOLOGY
RESEARCH ARTICLE | Free Access |
Convolutional capsule network for COVID-19 detection using radiography images
Shamik Tiwari, Anurag Jain
First published: 02 March 2021 | https://doi.org/10.1002/ima.22566
SECTIONS PDF TOOLS SHARE
Abstract


JPT Journal of Petroleum Technology
Deep Neural Network Trains To Pinpoint Microseismic Events ...
Deep Neural Network Trains To Pinpoint Microseismic Events in Real Time. Artificial intelligence is opening new ways to analyze data from ...
2 days ago


Healthy Hearing
AI hearing aids: How artificial intelligence can help those with ...
Key terms: AI, machine learning, deep neural network. Put simply, artificial intelligence is defined as the ability of a machine to simulate human ...
1 month ago


Medical Physics Web
Deep learning enables safer heart scans with lower ...
Positron emission tomography (PET) with the radiotracer 18F-FDG provides an important tool for assessing the health of the heart muscle in ...
2 days ago


HPCwire
Can Deep Learning Replace Numerical Weather Prediction?
The authors suggest that any eventual deep learning replacement for an NWP would likely consist of several neural networks trained on subsets ...
2 hours ago

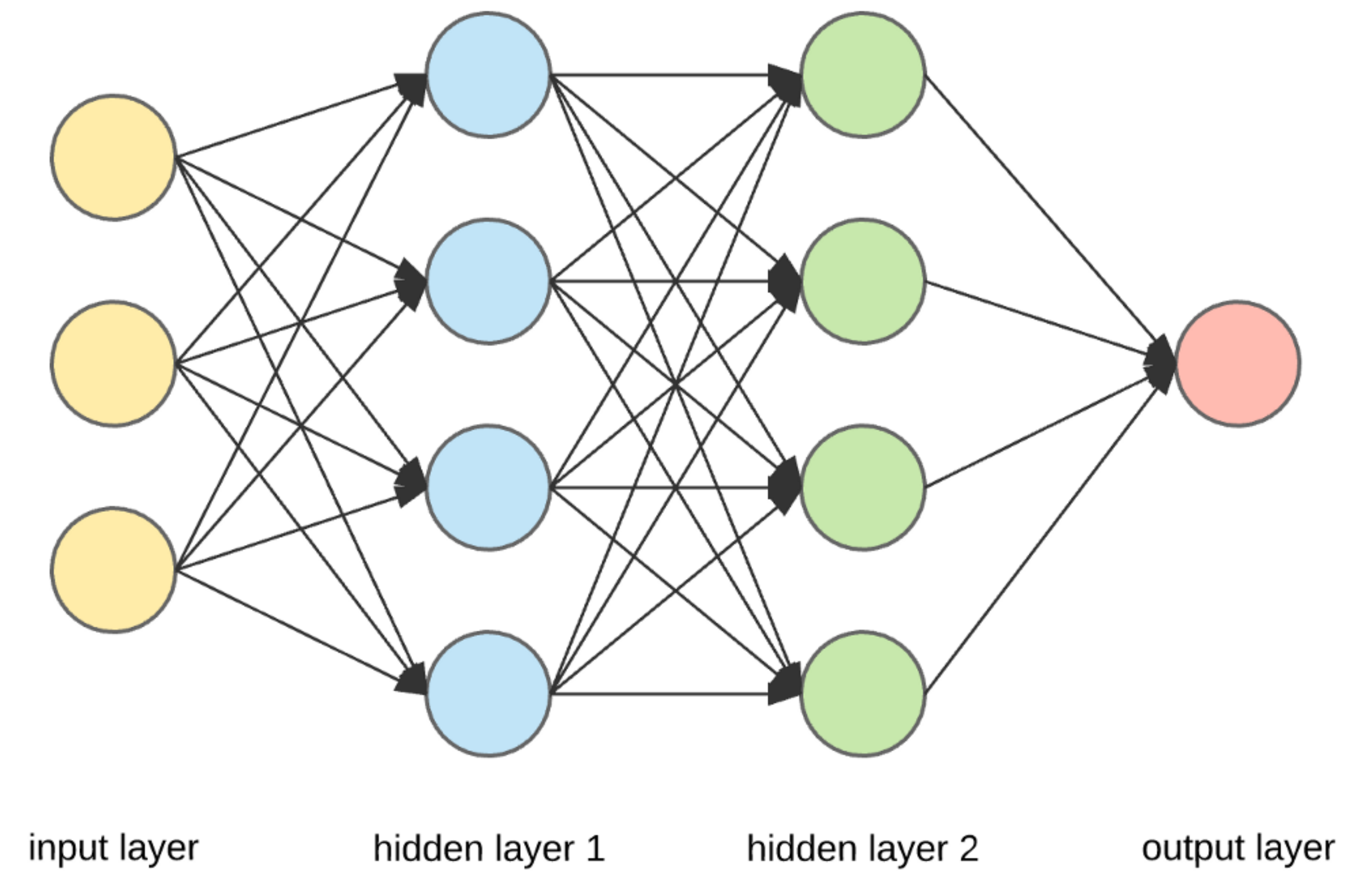
Neural Networks

A network of artificial neurons

- A network of artificial neurons are used to map input features to an output, which can be used for estimation (regression) or classification

- This artificial neural network (ANN) is used to learn complex non-linear mappings between the training pair

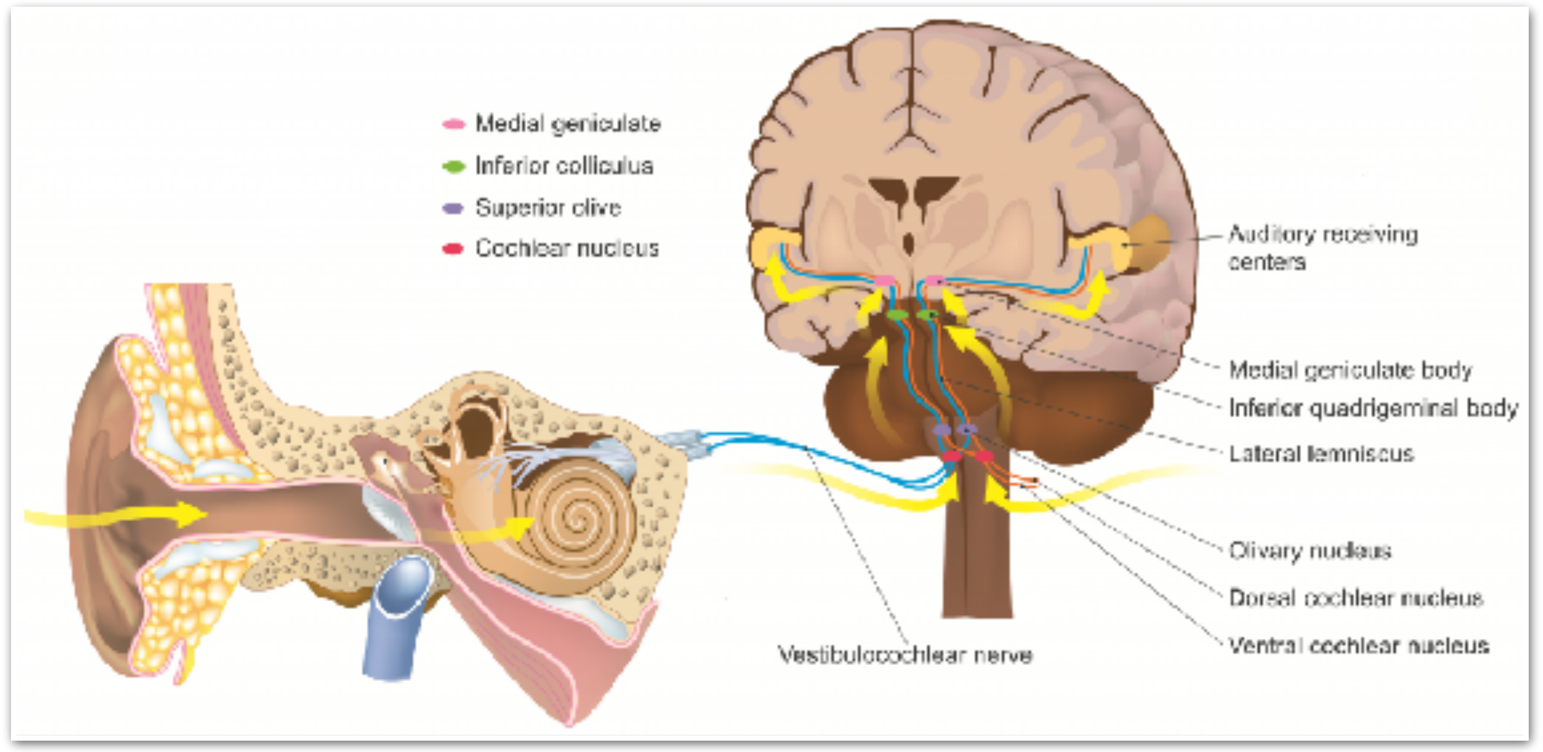
- Their main goal is to model some of the decision making that we do rather easily

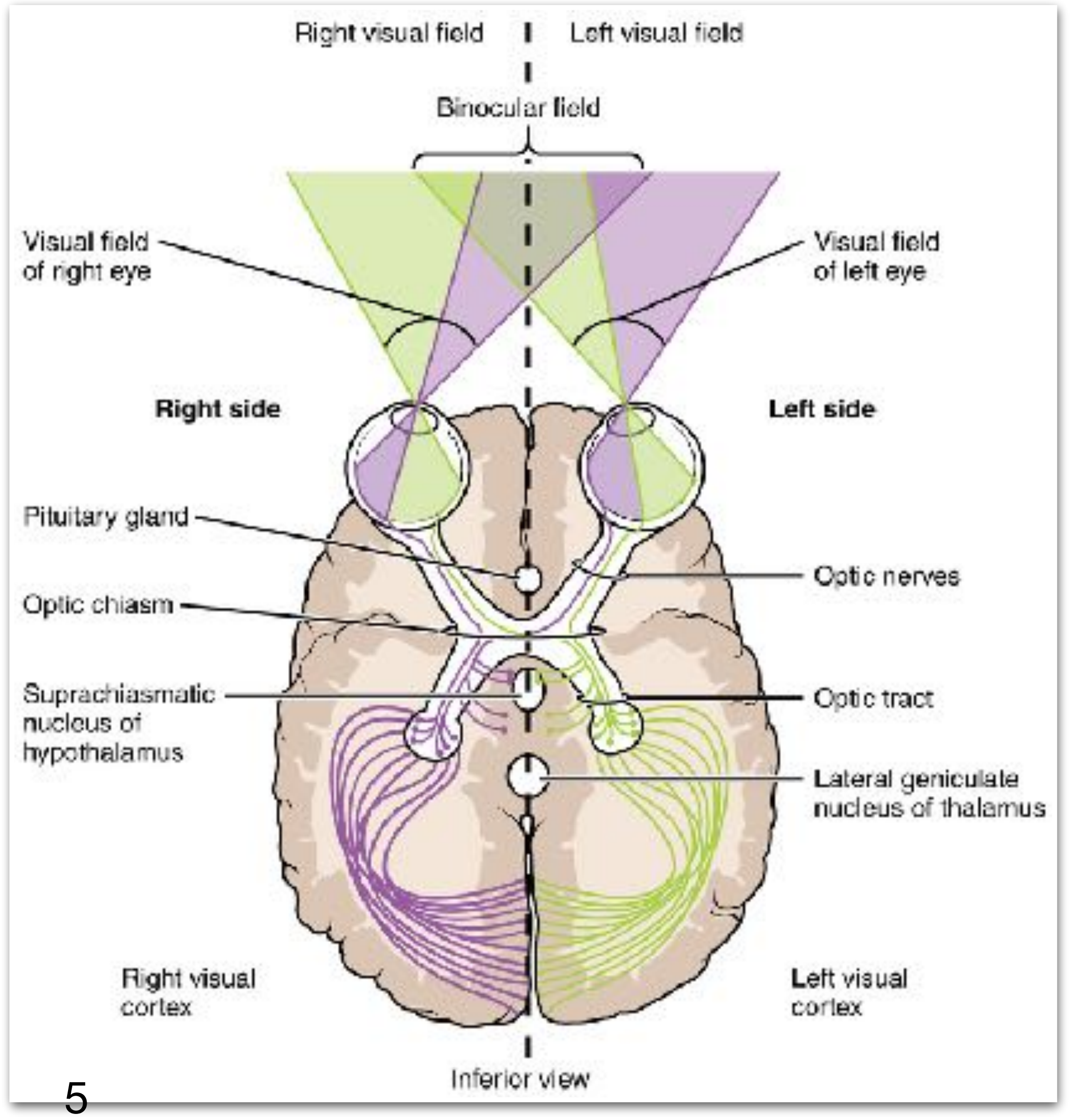


The Human Brain

Neural Networks are (loosely) modeled after our brain

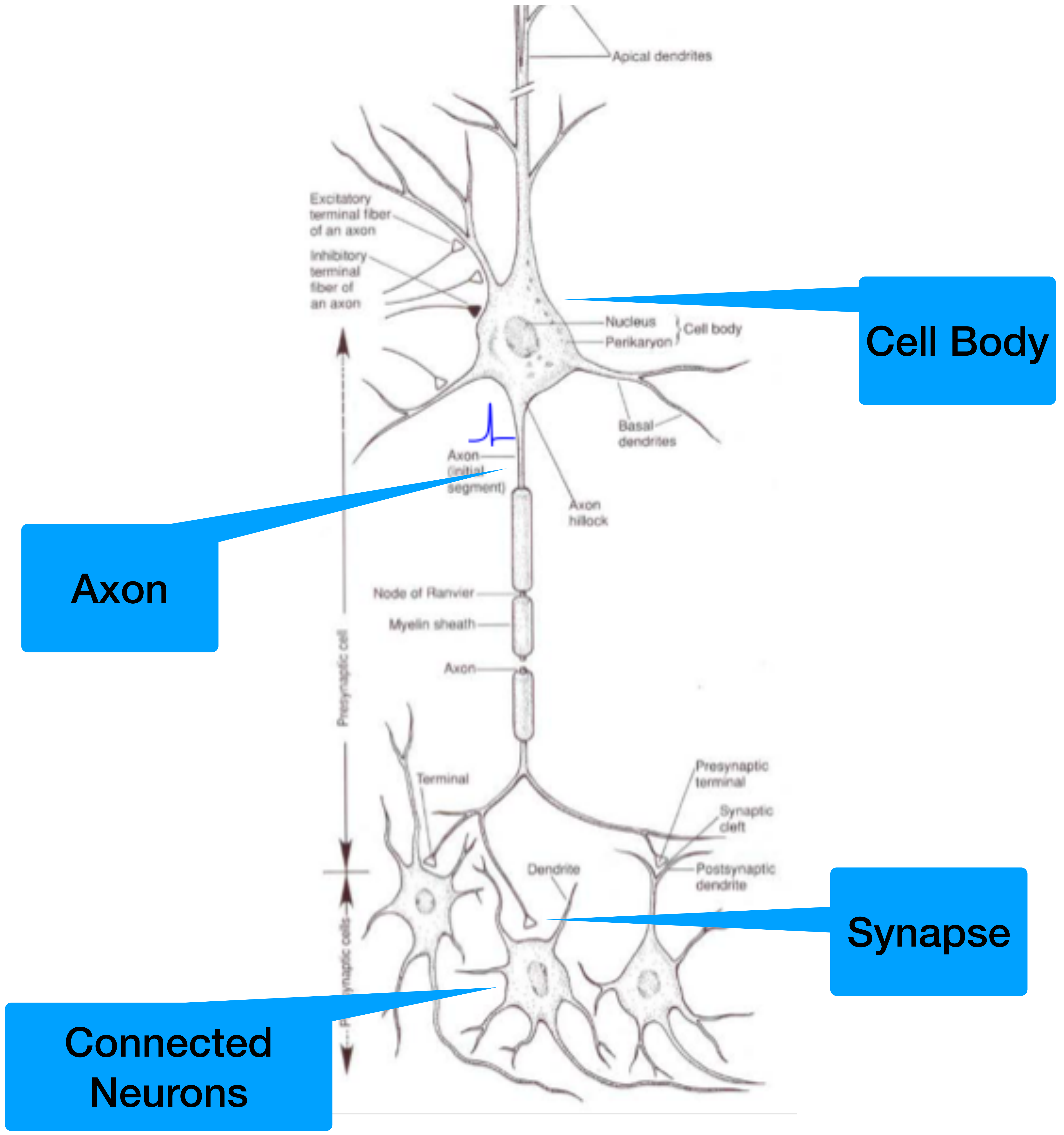
- Decision making occurs in our brain
- We make decisions (e.g. classification/ regression) from sensory input data (i.e. see, hear, smell, touch, taste)
  - Object detection
  - Voice recognition
  - Distant calculation
  - Etc.





Real Neurons (or nerve cell)

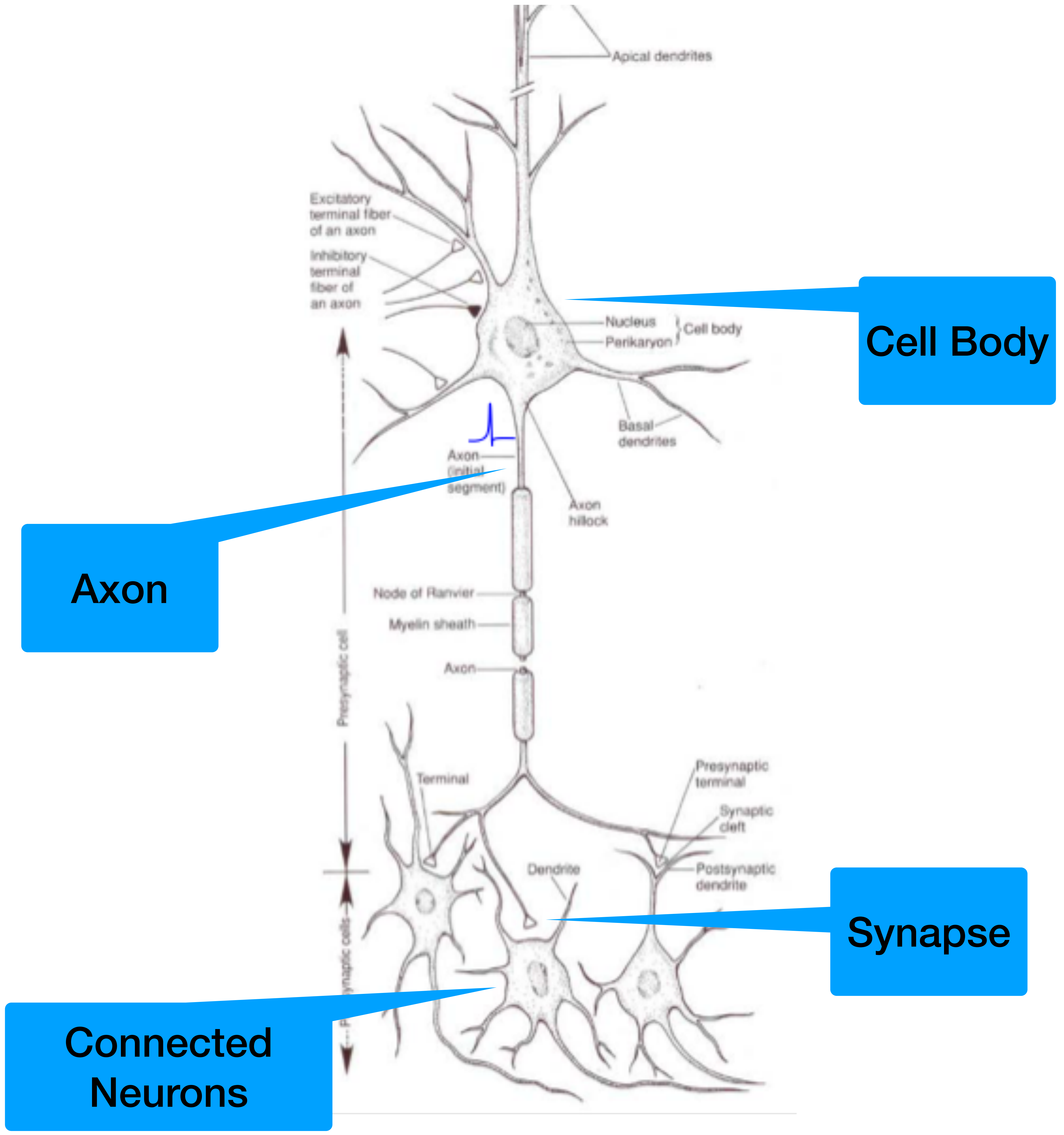
- Our brain consists of roughly 10^{11} neurons, each connected, on average, to 10^{4} other neurons. Our brain is an interconnected network
- A neuron has a tree like structure**. Main components of a neuron are:**
  - *Cell body* - generates a pulse (called an impulse)
  - *Axon* – impulse travels through this channel
  - *Synapse (or leaf)* – impulse terminates here. Connects to other neurons
- A neuron is considered *non-existent* unless it generates an impulse. Thus it is considered digital (binary), all or none



Real Neuron (or nerve cell) (cont.)

Types of Neurons

- There are two types of neuron contacts (synapses):
  - Excitatory (positive): receiving neuron increases activity
  - Inhibitory (negative): receiving neuron decreases activity
- Neuron can only be excitatory or inhibitory
- The synaptic contact is plastic, meaning it can be modified. This is essential for learning
- Terminology
  - **Neuron**: unit or node
  - **Synapse:** connection or architecture
  - **Synaptic weight** – connection strength (either positive or negative)



Artificial Neuron

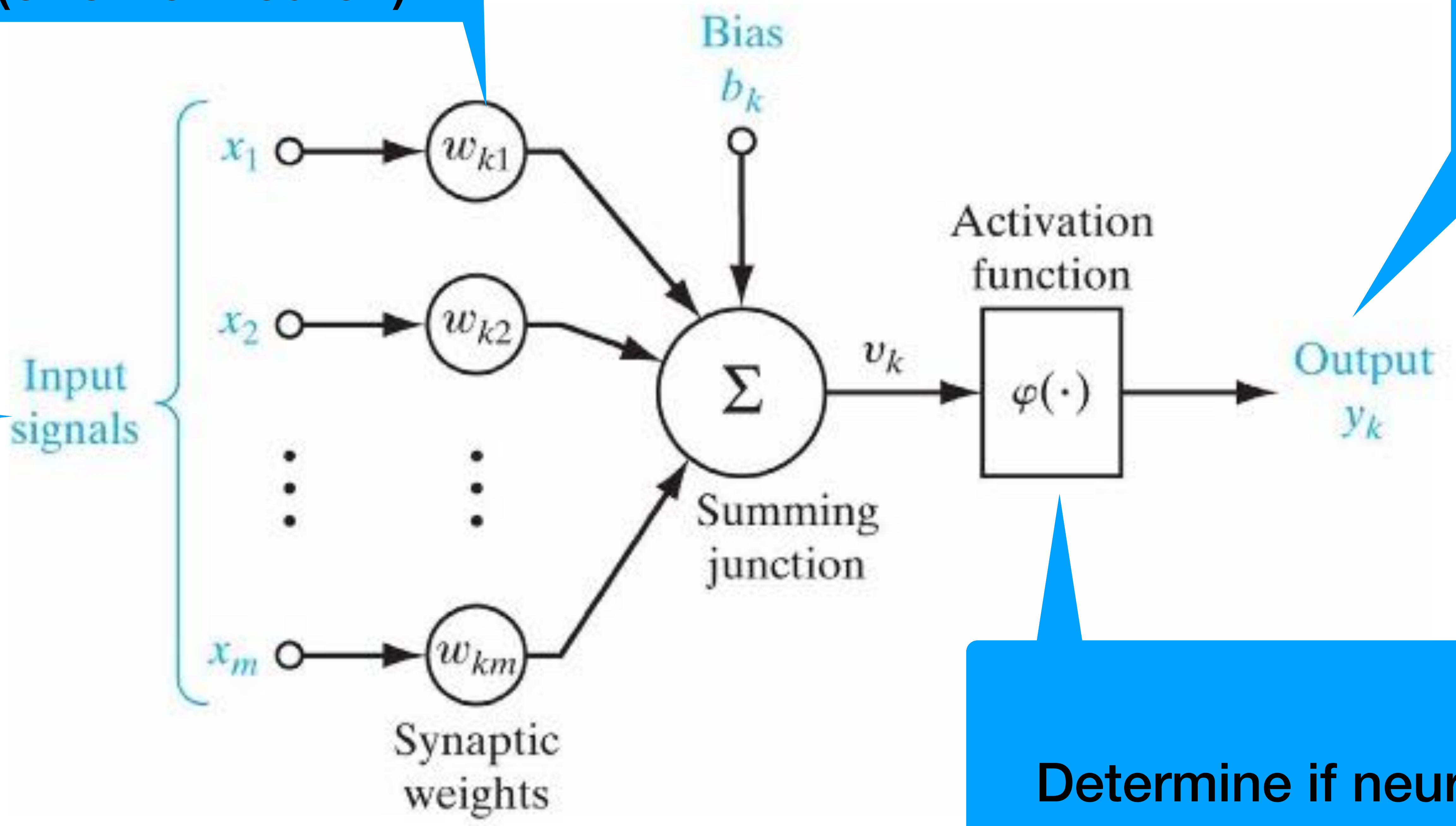
Components of an Artificial Neuron

Weights establish strength of connection to input (or other neuron)

Inputs can come from sensory data (e.g. image, audio, etc.) or from other neurons

Single scalar value per neuron

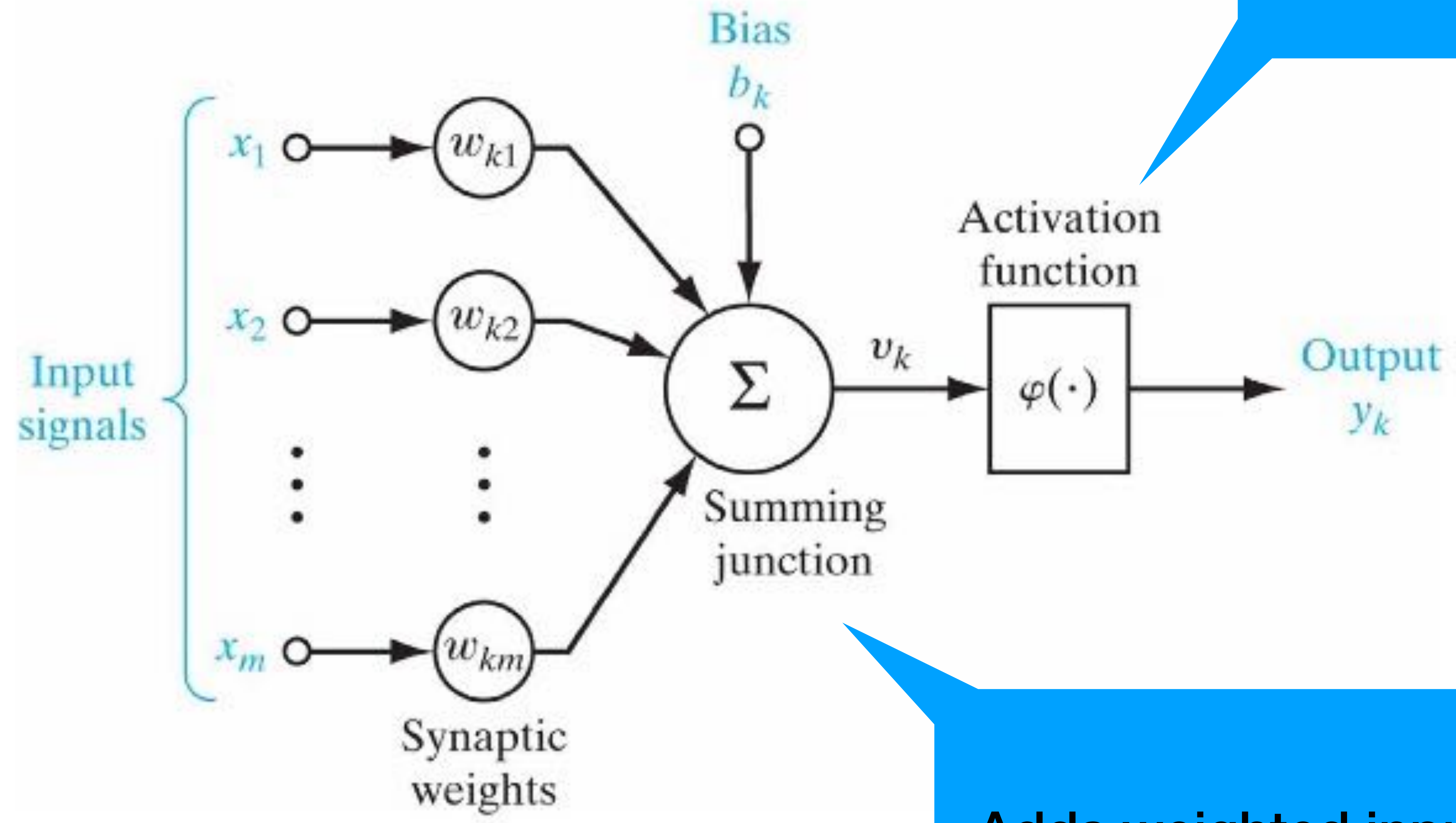
Determine if neuron "fires" or is active



Artificial Neuron

Components of an Artificial Neuron

Often nonlinear



Adds weighted inputs and bias

$$y_k = \phi(v_k)$$

Output
ϕ: activation function

$$u_k = \sum_{j=1}^{m} w_{kj} x_j$$

Adder, weighted sum, linear combiner

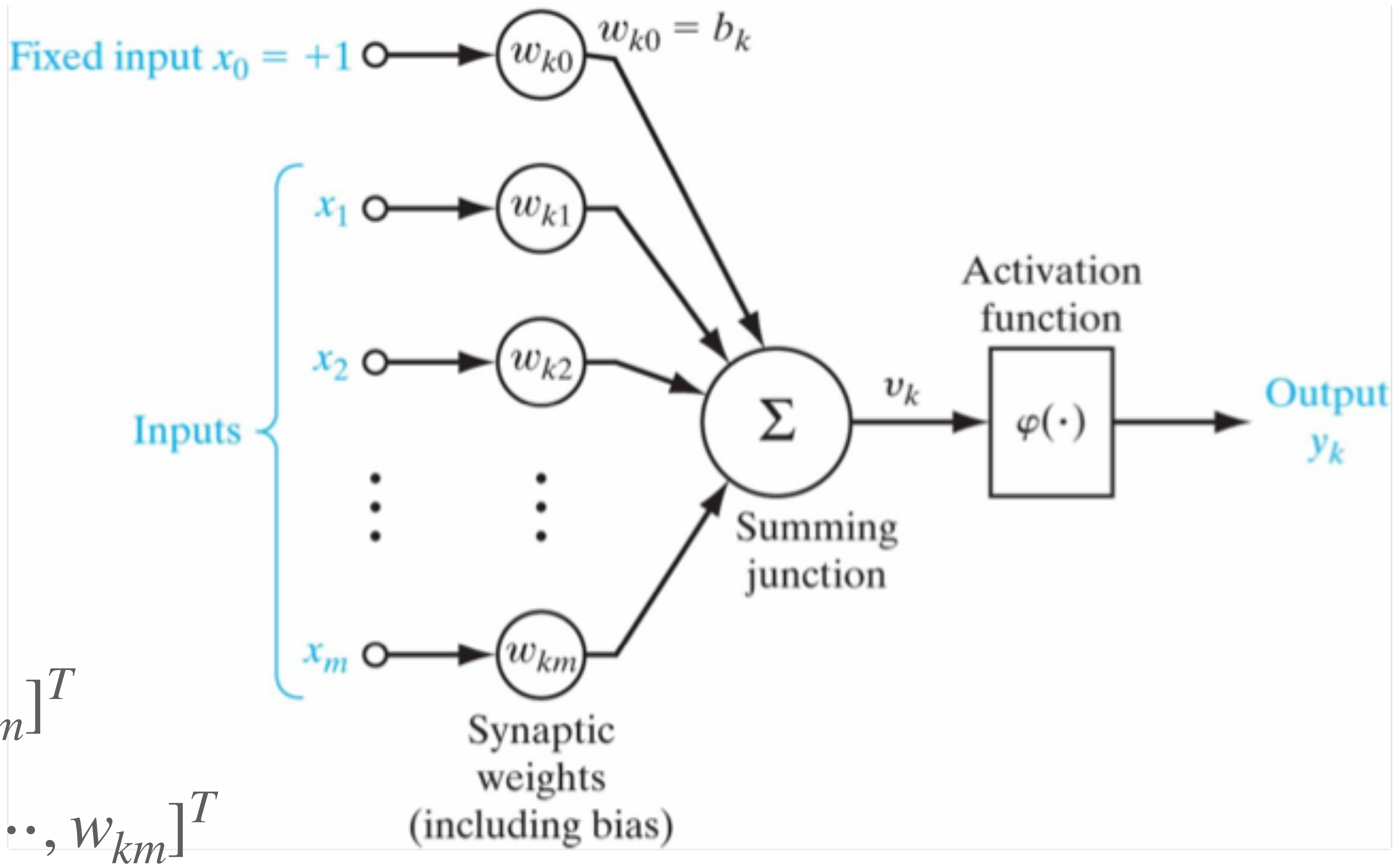
$$v_k = u_k + b_k$$

Activation potential
b_k: bias

Another way of including bias

Vector Notation

- As with linear regression, the bias can be seen as another input
  - Set $x_0 = +1$ and $w_{k0} = b_k$



- So we have

$$v_k = \sum_{j=0}^{m} w_{kj} x_j = \mathbf{w}_k^T \mathbf{x}$$

$$\mathbf{x} = [x_0, x_1, \cdots, x_m]^T$$

$$\mathbf{w}_k = [w_{k0}, w_{k1}, \cdots, w_{km}]^T$$

McCulloch-Pitts Neuron Model

- McCulloch-Pitts networks (1943) are the first class of abstract computing machines: finite-state automata. Finite-state automata can compute any logic (Boolean) function.
- Uses specific activation function.
- Uses bipolar inputs and generates bipolar outputs

$$x_i \in \{-1, 1\}$$

Bipolar input

$$y = \phi\left(\sum_{i=1}^{m} w_i x_i + b\right)$$

Output

$$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$$

Activation function

- A form of signum (sign) function

McCulloch-Pitts Model (cont.)

Used to represent logic gates (OR, AND, NOT)

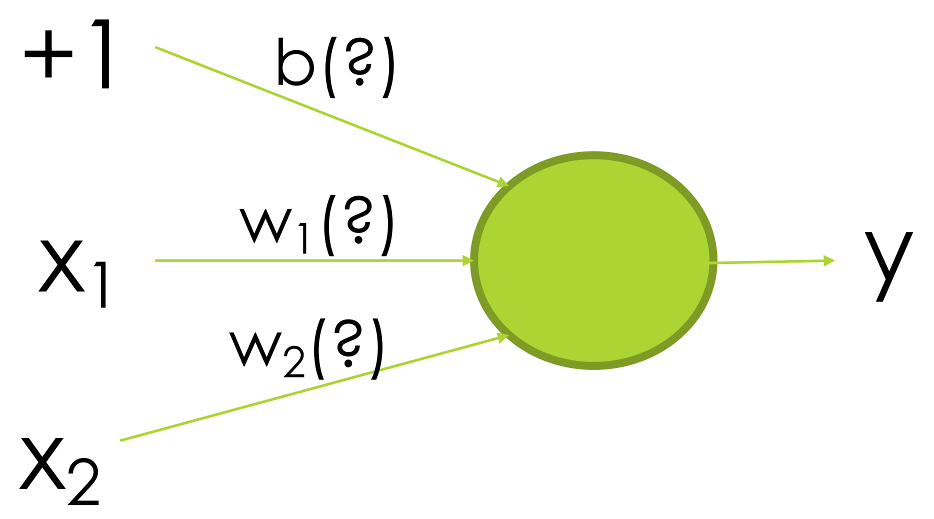
$$x_i \in \{-1, 1\}$$

$$y = \phi(\sum_{i=1}^{m} w_i x_i + b)$$

$$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$$

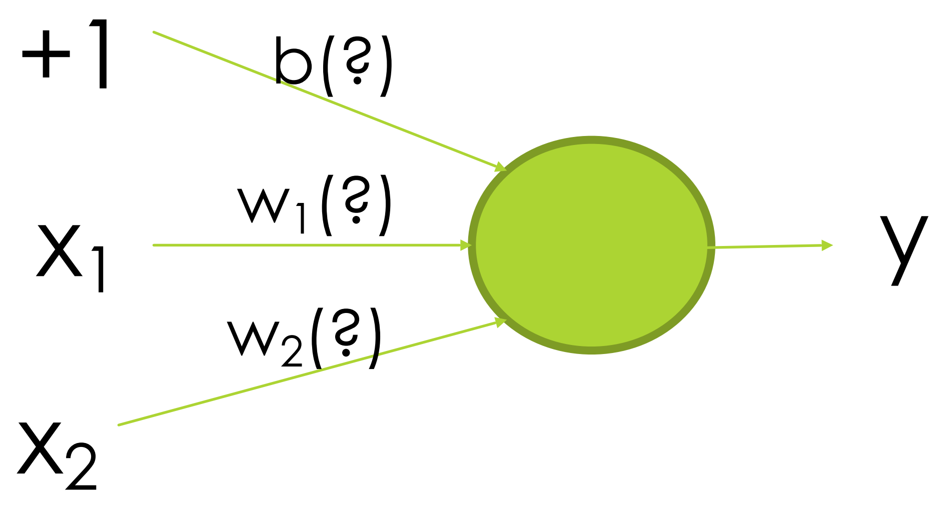
- Given the McCulloch-Pitts Model, define the weights that produce the correct output (y) for each input pair (x1, x2)

OR Function



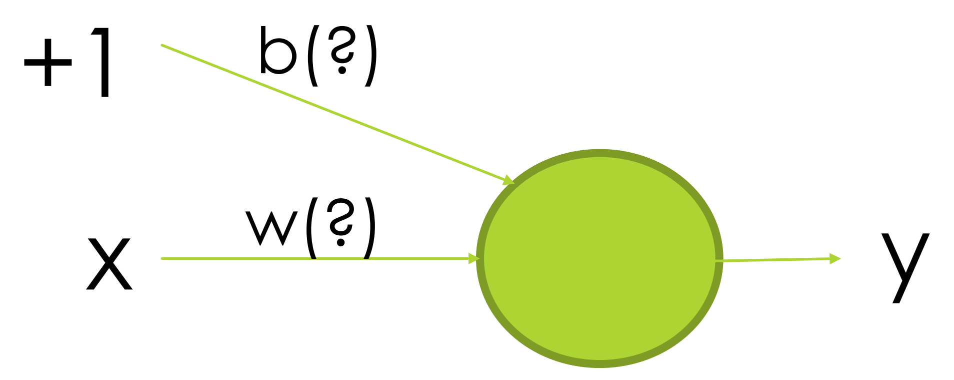
x1	x2	y
-1	-1	-1
-1	1	1
1	-1	1
1	1	1

AND Function



x1	x2	y
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

NOT Function



x	y
-1	1
1	-1

McCulloch-Pitts Model (cont.)

Used to represent logic gates (OR, AND, NOT)

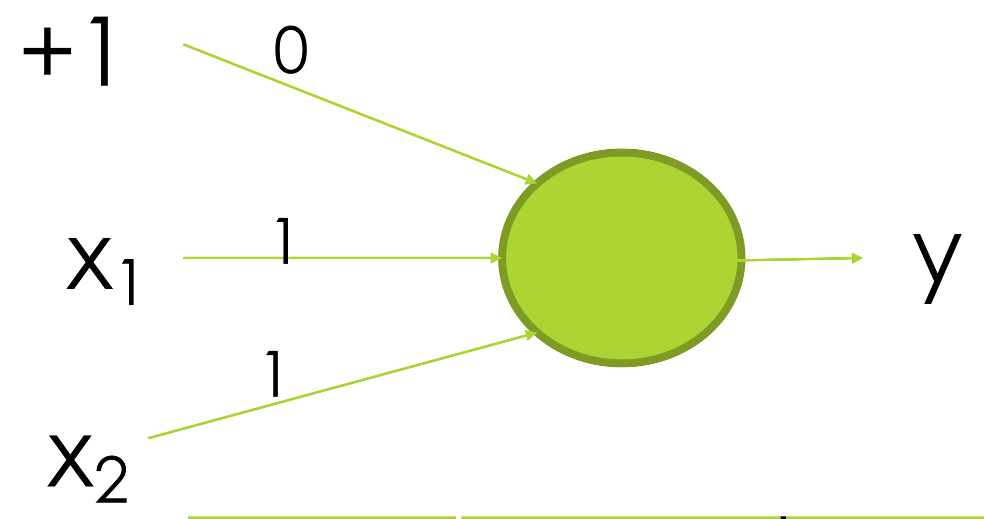
$$x_i \in \{-1, 1\}$$

$$y = \phi\left(\sum_{i=1}^{m} w_i x_i + b\right)$$

$$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$$

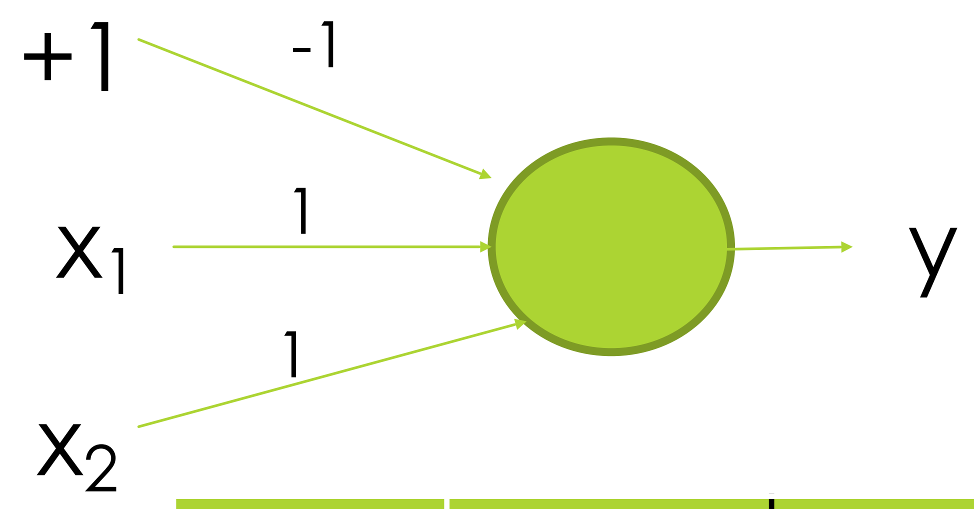
- Given the McCulloch-Pitts Model, define the weights that produce the correct output (y) for each input pair (x1, x2)

OR Function



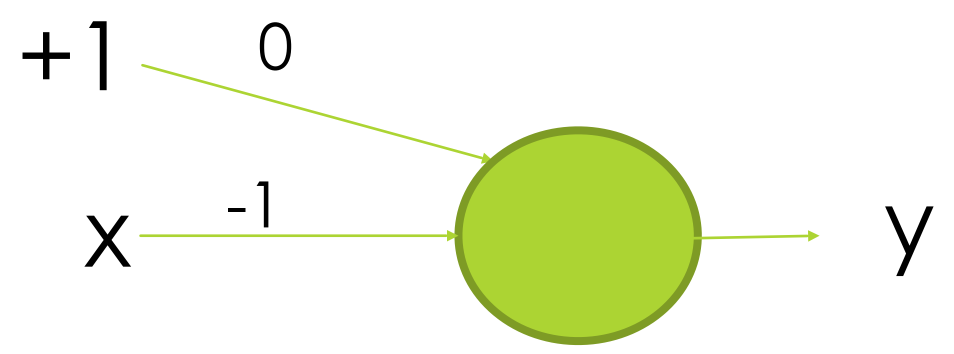
x1	x2	y
-1	-1	-1
-1	1	1
1	-1	1
1	1	1

AND Function



x1	x2	y
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

NOT Function



x	y
-1	1
1	-1

Perceptron Neuron Model

Modified version of McCulloch-Pitts neuron

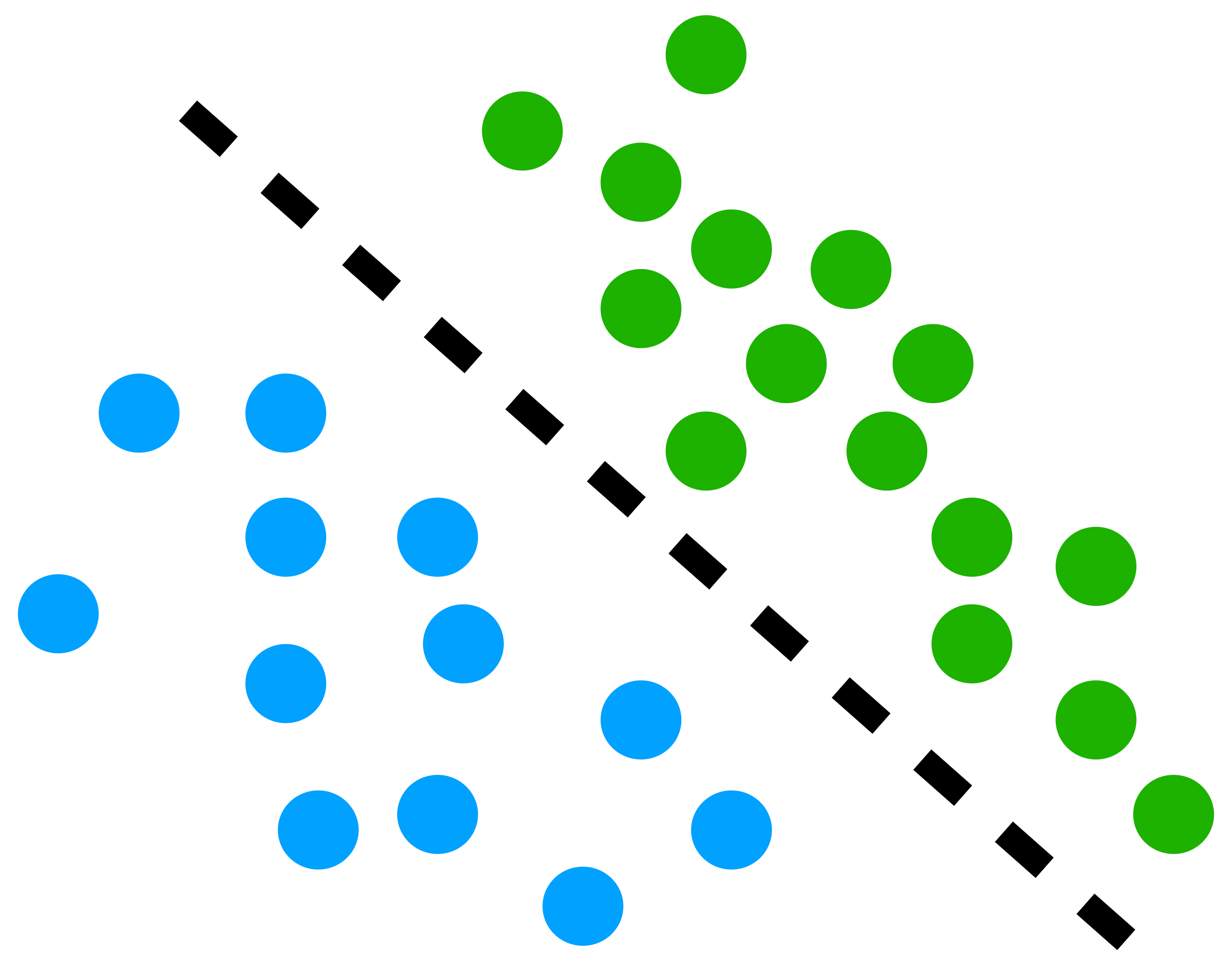
- McCulloch-Pitts neuron are limited by their inputs. **Perceptrons** address this by using ***real-valued inputs***.
- Perceptrons are also used for learning, whereas McCulloch-Pitts neurons are not
- Invented by Rosenblatt in 1957
- Useful for binary classification

$x_i \in \mathbb{R}$	Real-valued inputs
$v = \sum_{i=1}^{m} w_i x_i + b$	Activation potential
$y = \phi(v)$	Output
$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$	Activation function

Perceptrons for Classification

Decision Boundary

- **Goal**: Find values for weight vector (and bias), so that output of the perceptron matches the desired output, for each input.
    - For bipolar outputs, decide between -1 or 1
    - The **weights (and bias) result in a decision boundary** (e.g. discriminative approach), where features above the boundary are classified as one class, while features below the boundary are classified as the other.



- The ***decision boundary of a perceptron*** for a given **w**:

$$g(\boldsymbol{x}) = \sum_{i=1}^{m} w_i x_i + b = 0$$

Why is the decision boundary set to 0?

- *g(**x**)* is also called the discriminant function for the perceptron, and it is a linear function of **x**. *Hence, it is a linear discriminant function*

Decision Boundary (cont.)

Perceptrons

$$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$$

- The decision boundary of a perceptron is the line (or hyperplane) where the activation potential equals 0. This is based on the activation function of a perceptron
- For a 2-D input space with a bias: $$g(x_1, x_2) = w_1 x_1 + w_2 x_2 + b = 0$$

=> After re-writing the equation we can see that it follows the equation of a line

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

- The discriminant function is a line with slope $-w_1/w_2$ and intercept $-b/w_2$
- The distance of the function to the origin is $|b|/||\mathbf{w}||$, where

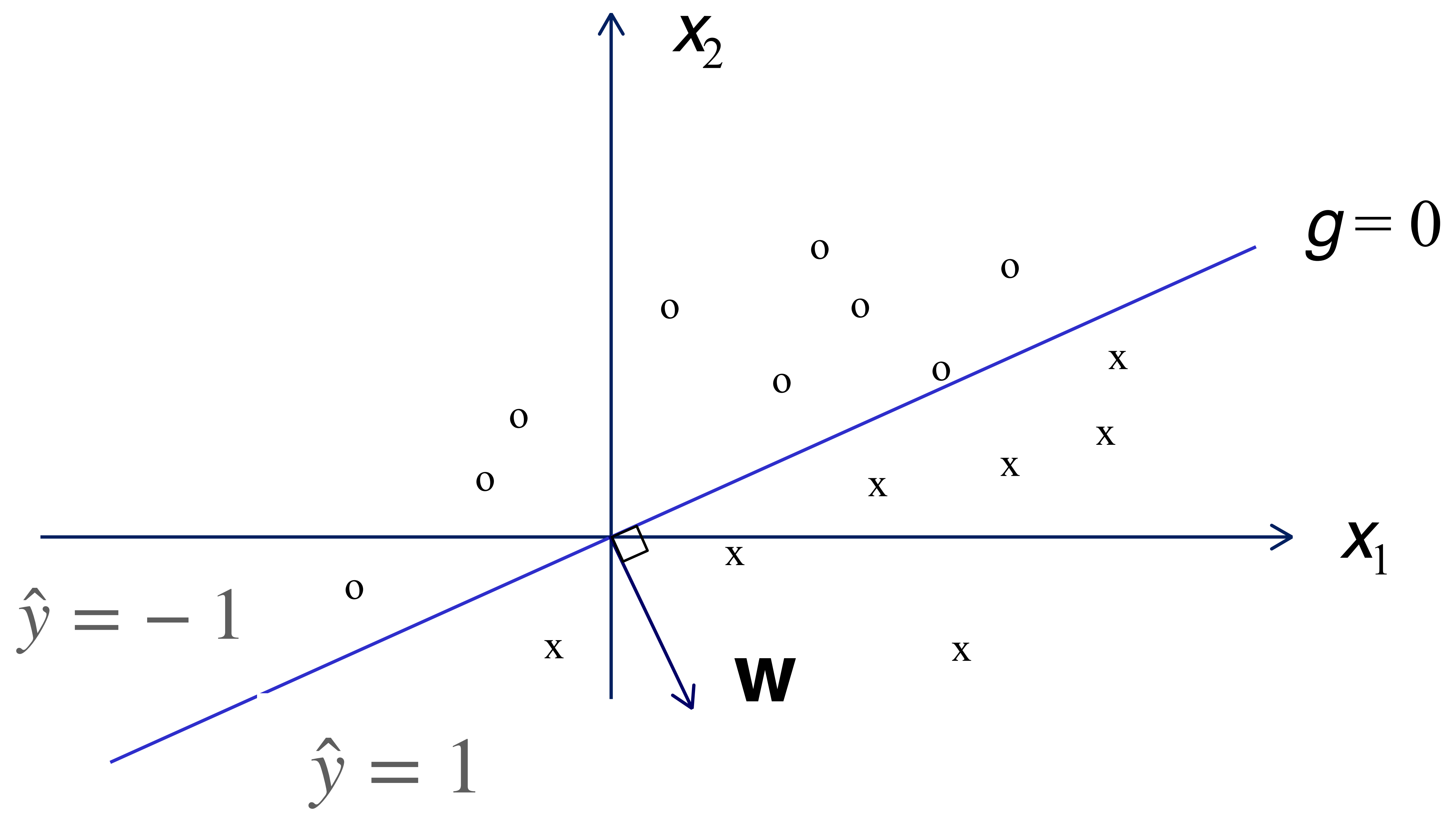
$$||\boldsymbol{w}|| = \sqrt{w_1^2 + w_2^2}$$

Decision Boundary (cont.)

Perceptrons

$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{b}{w_2}$$

- For an m-dimensional input space, the decision boundary is an (m-1)-dimensional hyperplane perpendicular to **w**.
    - By definition, weight vector is orthogonal to the decision boundary. **Why**? Recall: inner product
- The hyperplane separates the input space into two halves
    - One half having $\hat{y}$ = 1
    - The other half having $\hat{y}$ = -1
- When b=0, the hyperplane goes through the origin

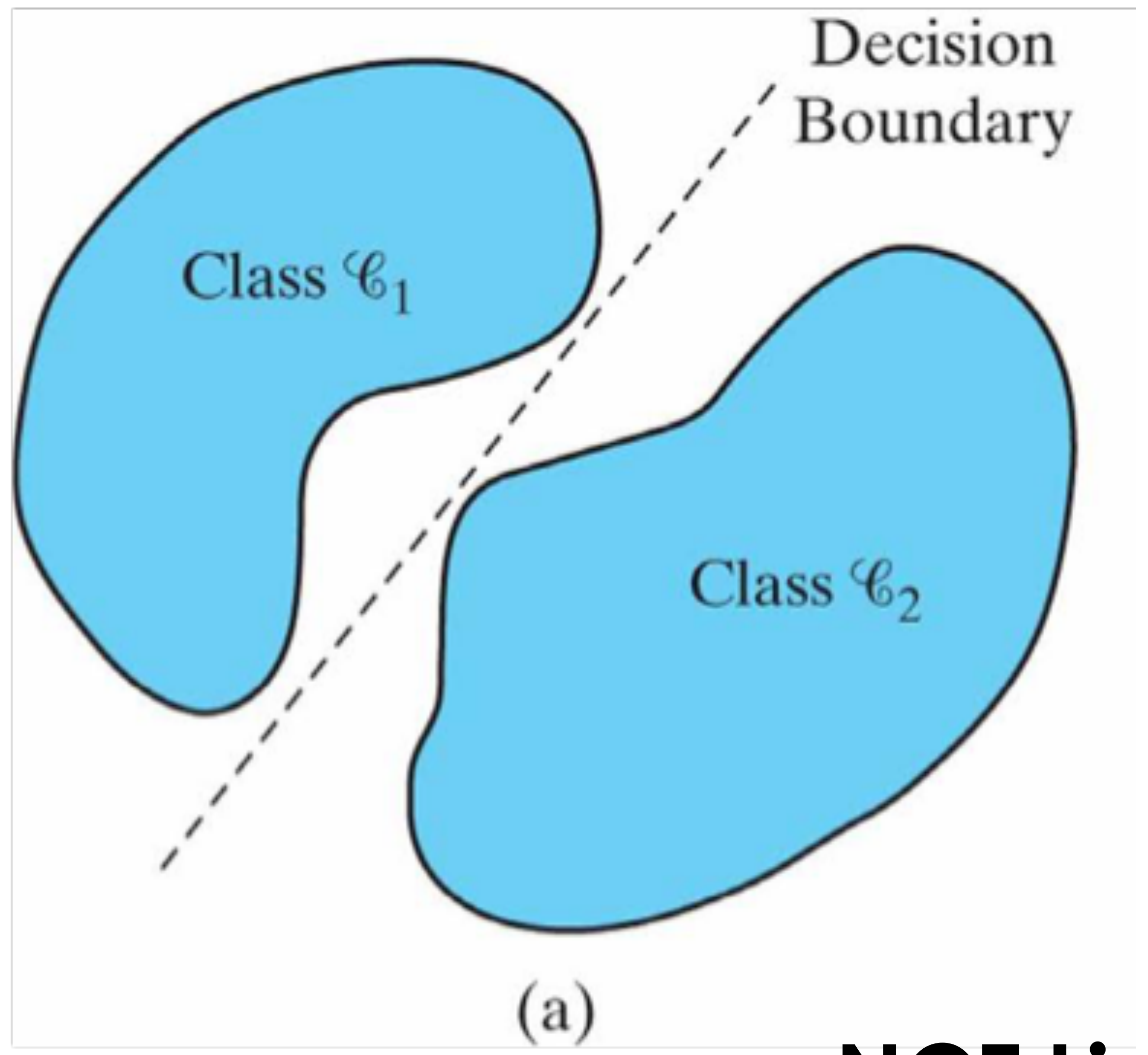


Linear Separability

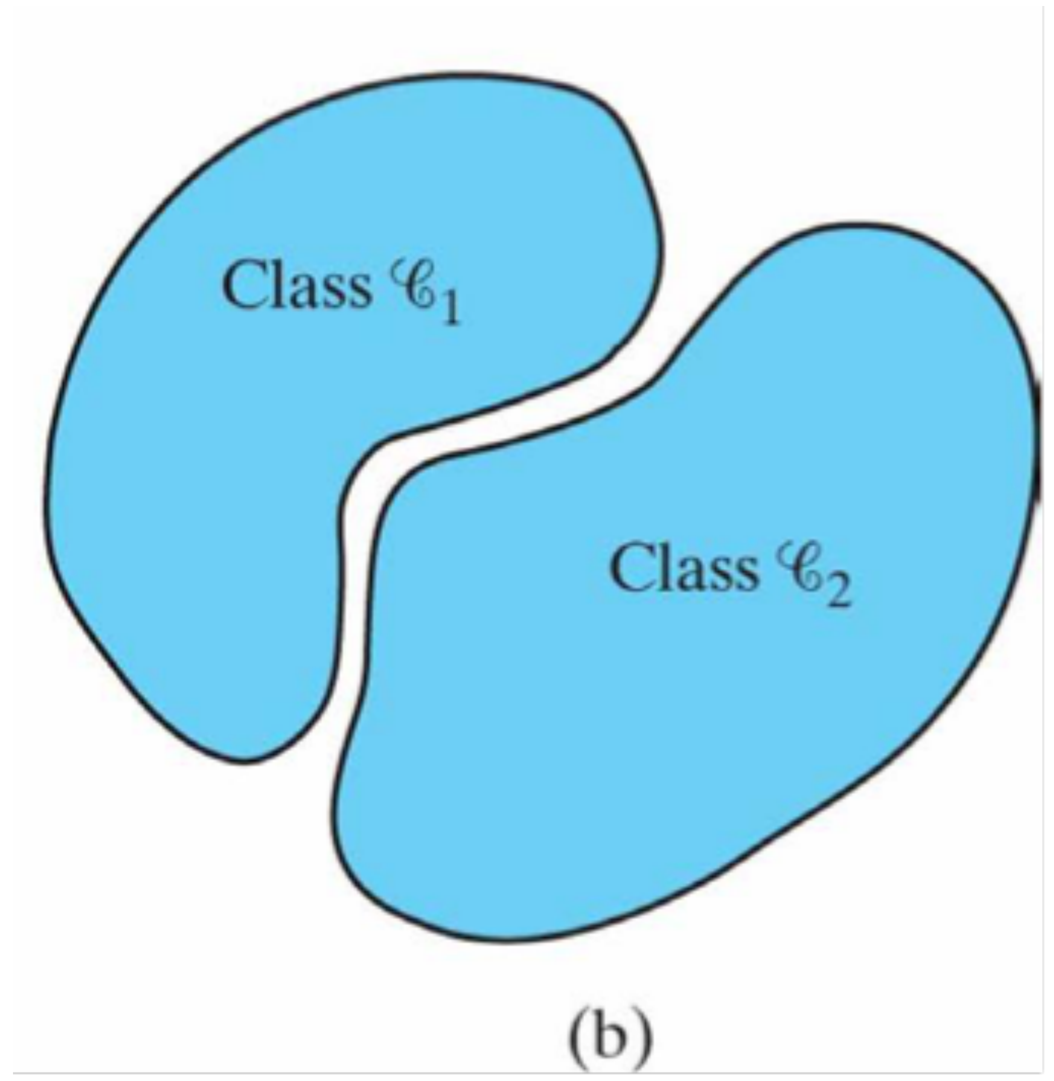
Perceptrons

- For the set of input patterns **x**, if there exists one **w** that separates y=1 (e.g. class A) patterns from y = -1 (class B) patterns, then the classification is called **<u>linearly separable</u>**
  - In other words, there exists a linear discriminant function that produces no classification error
  - Examples: AND, OR, XOR
- This is a very important concept!

<u>Linearly Separable</u>



<u>NOT Linearly Separable</u>



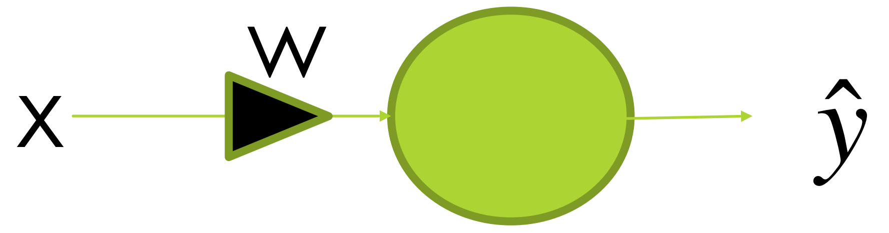
Perceptron Learning Rule

Finding weights that separate classes

$$v = \sum_{i=1}^{m} w_i x_i + b$$
$$y = \phi(v)$$
$$\phi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ -1, & \text{if } v < 0 \end{cases}$$

- For linearly separable problems, our goal is to "learn" a set of weights that results in zero classification error
- ***Perceptron Learning rule***:
  - **Strengthen the weight** if the neuron fails to fire when it should have fired
  - **Weaken the weight** if the neuron fires when it should not have fired
- Example: Single perceptron with single input

Ex. #1

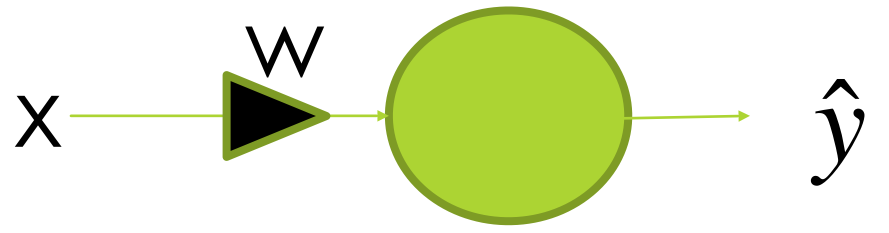


y = 1, but $\hat{y}$= -1

If want $\hat{y}$ to be 1, but the output is -1.
than this neuron should be strengthened.

In other words, the weight is too small. It needs to increase

Ex. #2



y = -1, but $\hat{y}$ = 1

This neuron should be weaken.
The weight should decrease.

Perceptron Learning Rule

Single Perceptron, Multiple inputs

- How do we change the weights if they are too small or too large?

$$\begin{aligned}\mathbf{w}^{n+1} &= \mathbf{w}^{n} + \nabla(\mathbf{w}) \\ &= \mathbf{w}^{n} + \eta(y - \hat{y})\mathbf{x}\end{aligned}$$

$n :$ iteration number

$\eta :$ step size or learning rate

Actual output

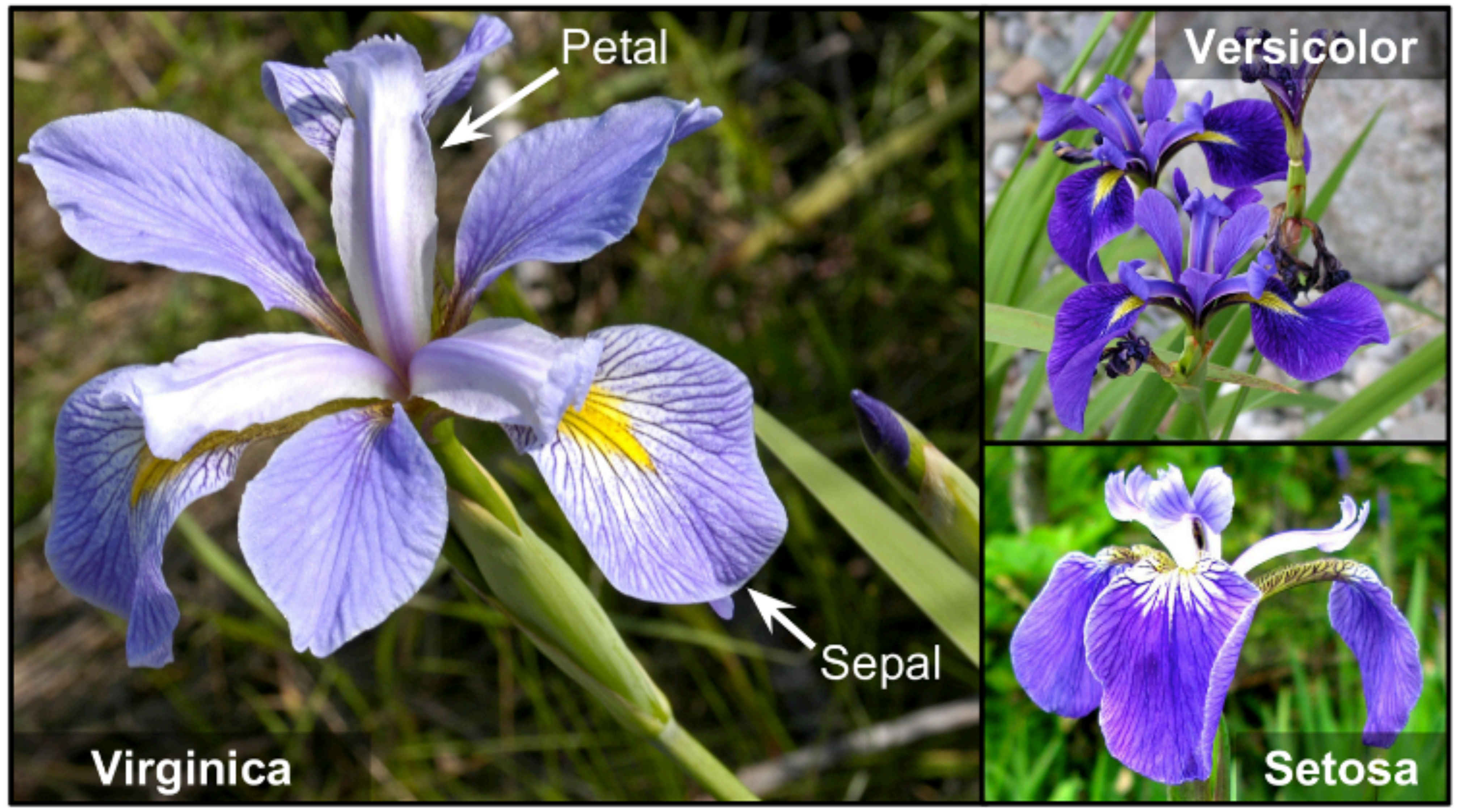
Desired output

- Note:
  - The outputs are bipolar {-1,1}
  - Thus, [y-$\hat{y}$] is either 0 (correct), -2 (too strong), or 2 (too weak)

Perceptron for Flower Classification

A Python Example

- Let's classify the type of Iris flower, based on the flowers sepal and petal length/width
  - Let's simplify even more by classifying if the flower is a Iris-Virginica (y = 1) or not, based on the petal length/width



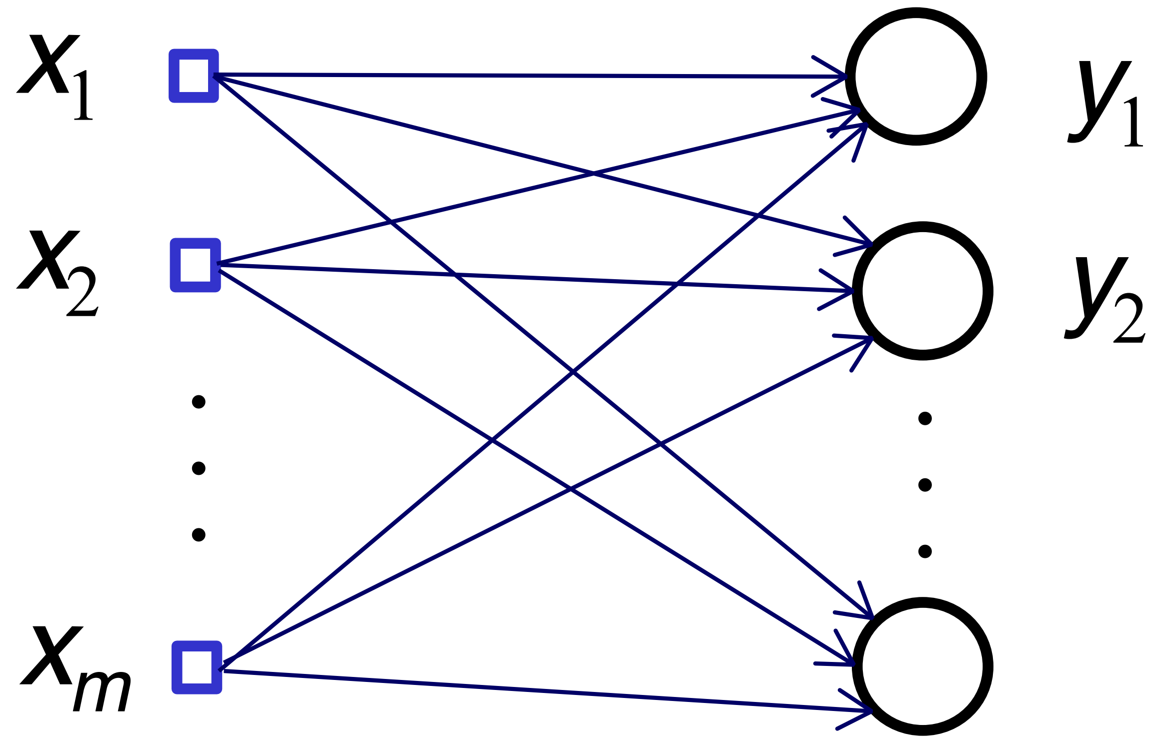
- See Jupyter Notebook

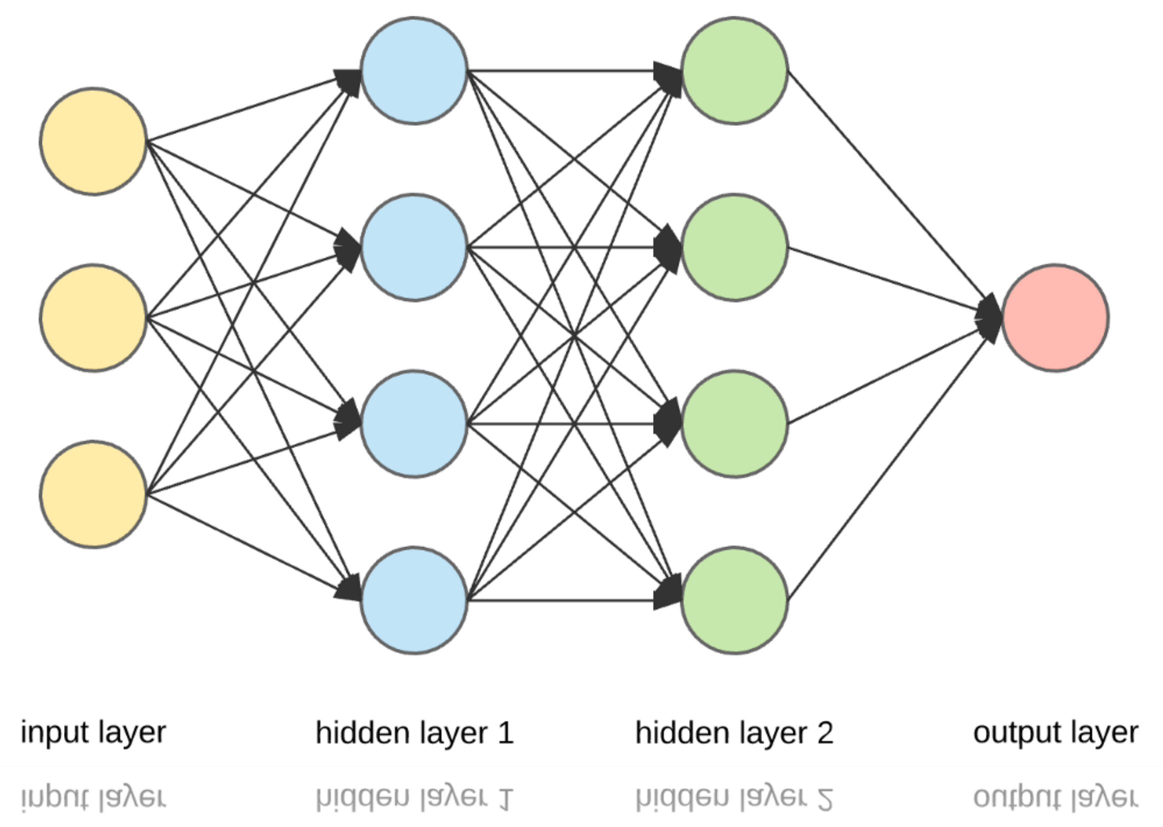
Perceptron Convergence Theorem

- **Theorem**: If a classification problem is linearly separable, a perceptron will reach a solution in a finite number of iterations
  - In other words, a perceptron can be trained to solve all linearly separable problems
  - A detailed proof is shown in any ML book, but we will not go over this in class
- The solution weight vector is **not** unique. There are infinite possible solutions and decision boundaries.
- If the problem is not linearly separable then modifications need to be made

Perceptrons for Multi-class classification

- Multiple perceptrons can be used when performing multi-class classification (one for each class)
  - Perceptron output is 1 when input is from the corresponding class
  - Its output is 0 otherwise
- When these perceptrons have the same inputs (but with different weights), this stacking of perceptrons is called a ***layer***
- When the outputs of one layer become the input to another, we call this a ***multi-layer perceptron (MLP)*** or ***neural network***. ***Deep neural networks*** have three or more layers





Next Class:

Neural Networks II: More on Multi-layer Perceptrons (MLPs) and back propagation