Neural Networks

(P-ITEEA-0011)

Introduction to the course
Single layer perceptron

Akos Zarandy
Lecture 1
September 10, 2019

Outline

- Administration: requirements of the course
- Machine learning – Machine intelligence
- Artificial neuron
- Perceptron

Course requirements: Signature requirements

- Mandatory **attendance** 80% (lectures and practice sessions)
- **Short quiz** at every practice session.
  - You have to reach at least **60% of all points**
- **Lab report**: one can be skipped
- **Paper based test**: minimum 50%
- **Computer-based test**: minimum 50%

Course requirements: Lab Reports

- Lab reports are short summaries of the previous practice session
- You will have to work in teams of 3 (**talent program alone**)
- Submission: on the main page of the course until 4 am the day before the next practice session
- Contents:
  - Your names, your email addresses, the time and date of the practice session
  - A brief description of the new methods/techniques and their mathematical background (if applicable) we used
  - A general description of the dataset we used (with examples from the dataset) (if applicable)
  - If we used any new network architectures, a detailed description of that specific architecture.
- You may use Internet, however you must cite that source, else your report will not be accepted. The same goes for too similar lab reports.

Course requirements: **Midterm project**

- Not mandatory in general
  - Mandatory for the **talent program**
- Required to earn an offered grade
- You will need to apply for it after it is announced
- Once you choose a task, nobody else can, so there will be no possibility of changing your task, or cancelling your selection
- You will have to submit an acceptable solution, otherwise your final score will be reduced by 20%

Course requirements: Tests

- **Paper-based test**
  - 15. October
  - Theoretic questions and paper based calculations
  - In the time and location of the lecture
  - You need to score at least 50% to pass

- **Computer-based test**
  - Considered to be a part of the exam
  - The test will be held at the end of the semester, it will be 3-4 hours long
  - The test will be graded on the spot
  - You need to score at least 50% to pass

Course requirements: Exam and grade

- **Exam**
  - Oral exam
- **Offered grade**
  - Only a 4 or 5 can be received
  - Limits on the offered grades:
    - > 85% of the short quizzes, the closed-room test
  - Midterm project required, final grade depends on it
- **Early exam**
  - There will also be an exam in the first of the exam period (before the computer-based test) for those students who excelled most during the semester. This exam is invite-only by the lecturers, and if you are invited, you are excused from the computer-based test

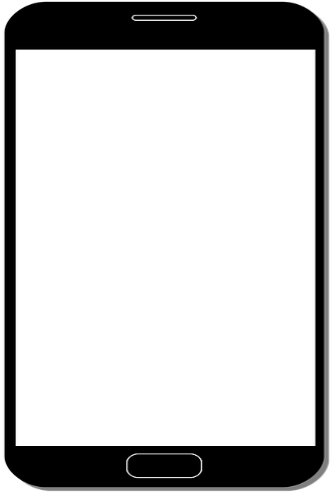
Detailed description of the requirements on the webpage of the course:
http://users.itk.ppke.hu/~konso1/neural_networks

Outline

- Administration: requirements of the course
- Machine learning – Machine intelligence
- Artificial neuron
- Perceptron

Machine learning, machine intelligence

- What is intelligence?
- The ability to acquire and apply knowledge and skills.
- The definition changes continuously



Machine learning, machine intelligence

- What is intelligence?
- The ability to acquire and apply knowledge and skills.

Intelligence is the ability to adapt to change

„Stephen Hawking”

Providing computers the ability to learn without being explicitly programmed:

Involves: programming, Computational statistics, mathematical optimization, image processing, natural language processing etc...

Conventional approach

- Trivial, or at least analitically solvable tasks
  - Well established mathematical solution exist or at least can be derived

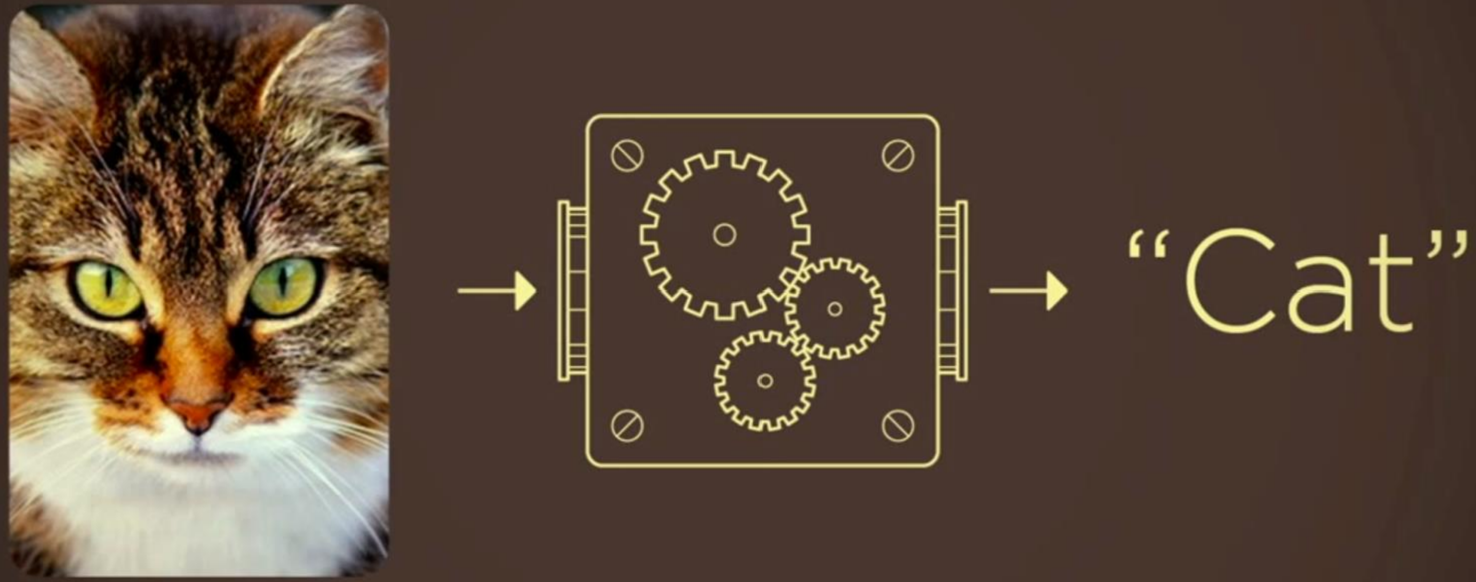
- Example:
  - Finding well defined data constellations in a database
  - Formal verification of the operation is easy

Machine learning approach

- Complex underspecified tasks
  - No exact mathematical solution exists, the function to be implemented is not known

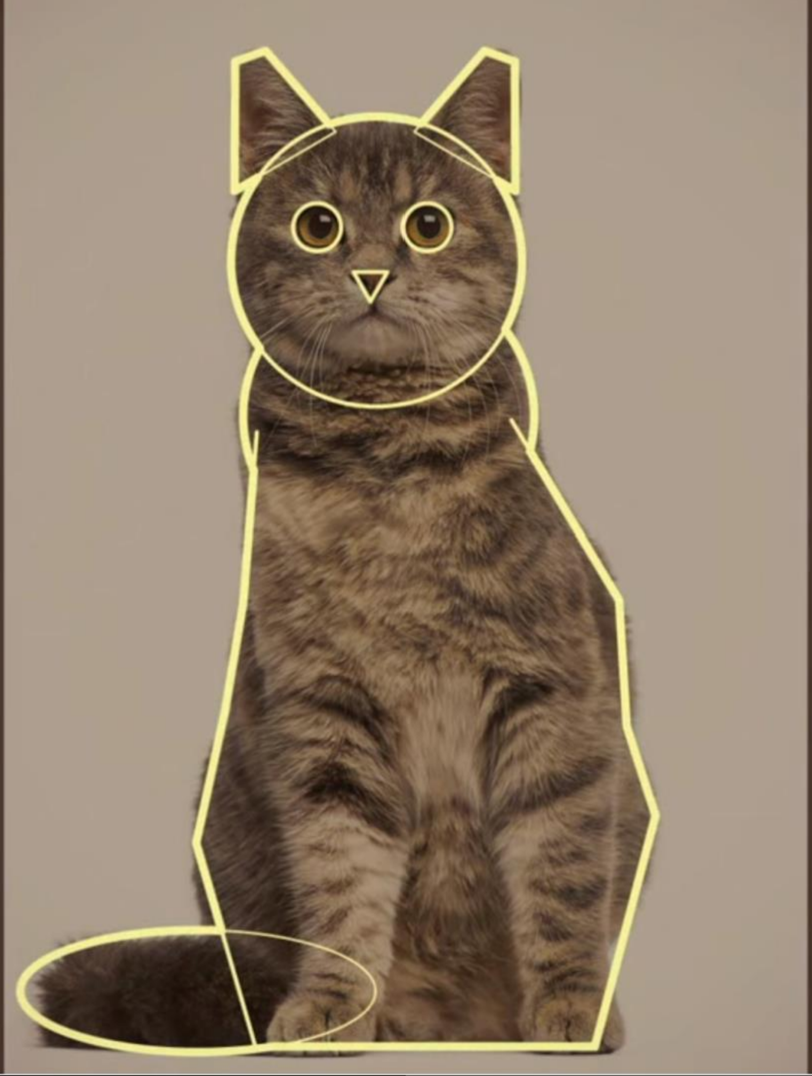
- Example:
  - Searching for “strange” data constellations in a database
  - Verification of the operation is difficult

In case of very complex problems, verification of the operation is very difficult. Typically done by exhaustive testing in case of machine learning.

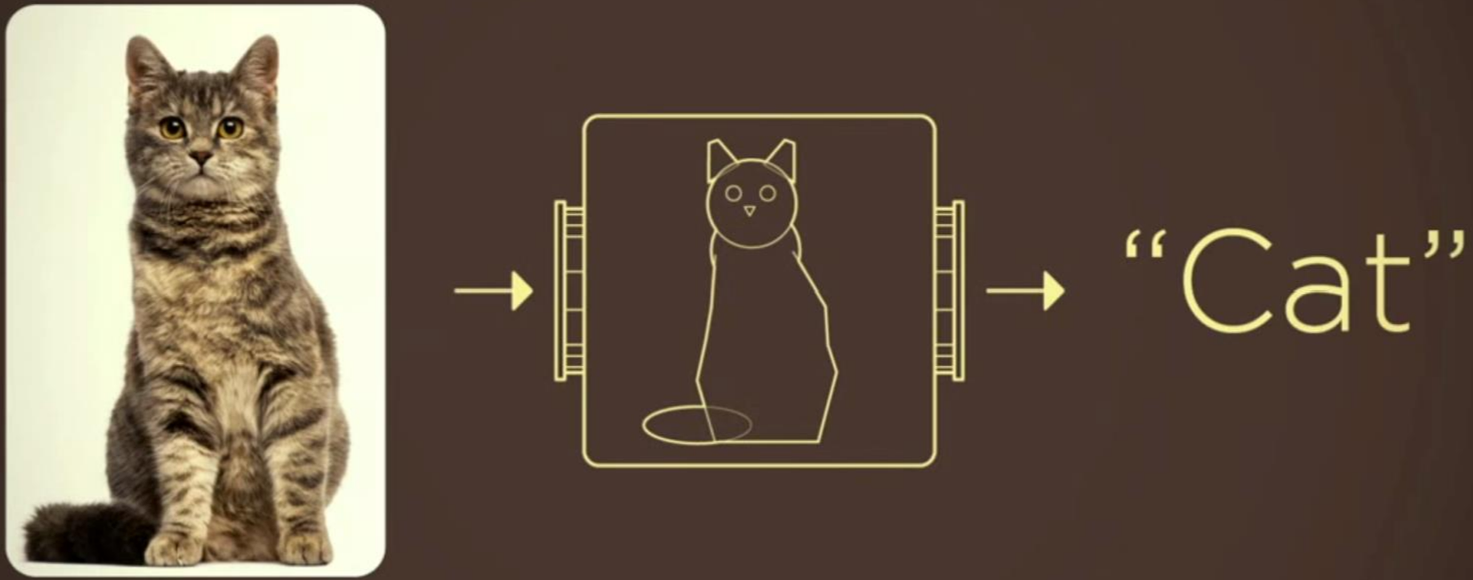


Credits: Fei Fei Li

Credits: Fei Fei Li

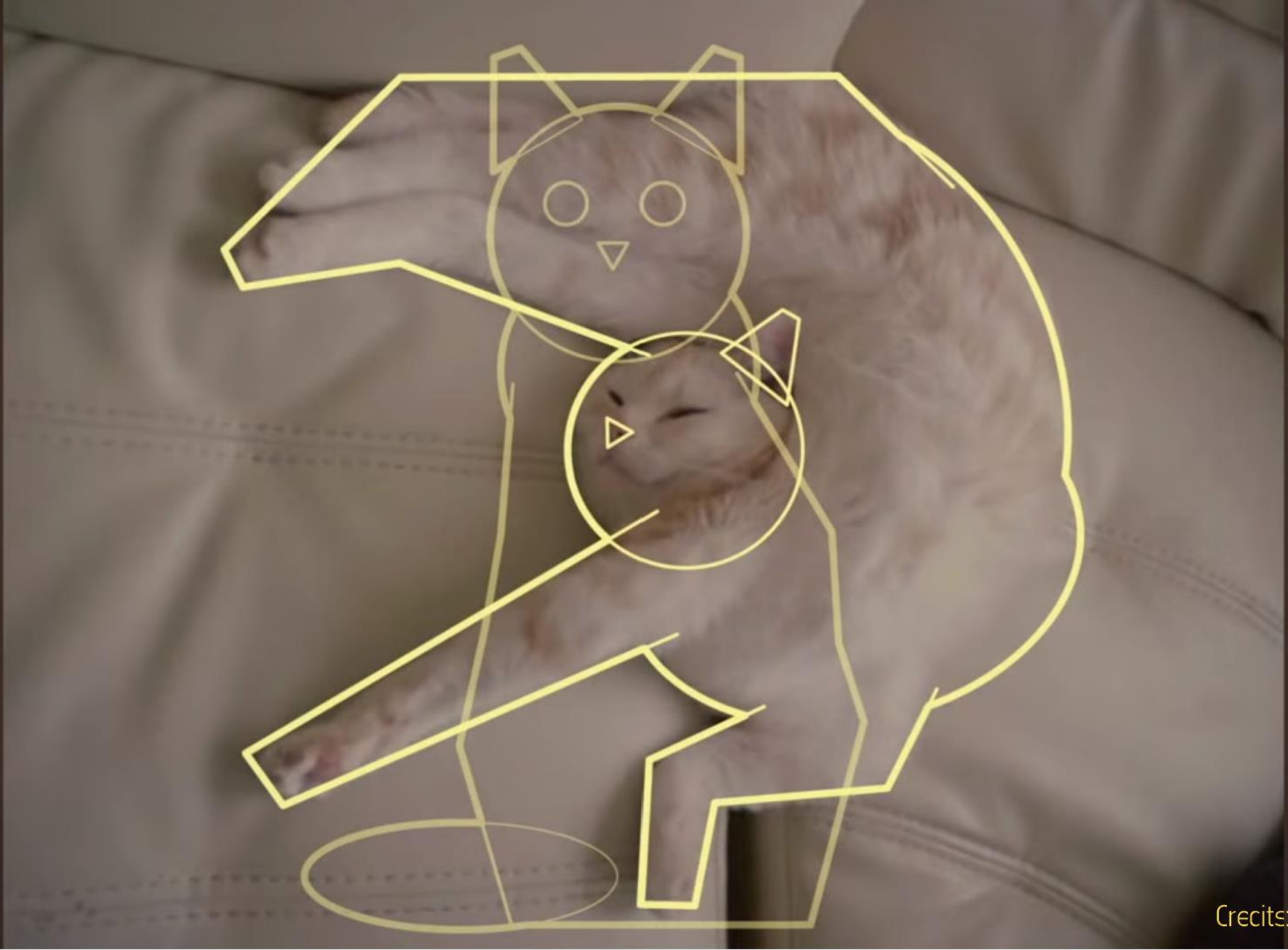


Crecits: Fei Fei Li



Credits: Fei Fei Li

Crecits: Fei Fei Li



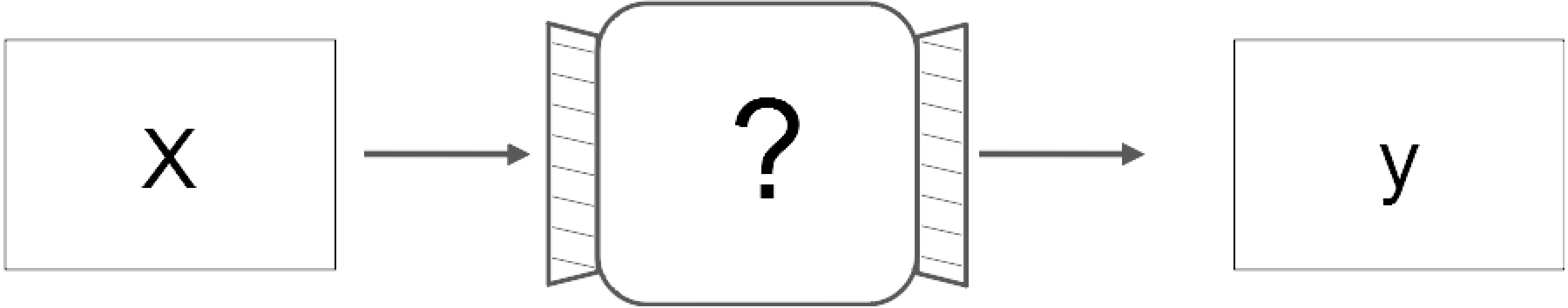
Crecits: Fei Fei Li

Crecits: Fei Fei Li

General truth: there are no general truths

Machine learning

We consider each task as an input-output problem



X: scalar, vector, array or a sequence of these (incl. text)

size(X) vs size(Y)
Data reduction
Data generation

Y: Decision or scalar, vector, array or a sequence of these (incl. text)

Conquests of machine learning

- 1952 Arthur Samuel (IBM): First machine learning program playing checkers

Arthur Samuel coined the term „machine learning”

Conquests of machine learning

- 1952 Arthur Samuel (IBM): First machine learning program playing checkers
- 1997 IBM Deep Blue Beats Kasparov

First match (1996 Nov):

Kasparov–Deep Blue (4–2)

Second Match (1997 May):

Deep Blue–Kasparov (3½–2½)

Conquests of machine learning

- 1952 Arthur Samuel (IBM): First machine learning program playing checkers
- 1997 IBM Deep Blue Beats Kasparov
- 2011 IBM Watson: Beating human champions in Jeopardy

It's a 4-letter term for a summit; the first 3 letters mean a type of simian : **Apex**

4-letter word for a vantage point or a belief : **View**

Music fans wax rhapsodic about this Hungarian's "Transcendental Etudes" : **Franz Liszt**

Conquests of machine learning

- 1952 Arthur Samuel (IBM): First machine learning program playing checkers
- 1997 IBM Deep Blue Beats Kasparov
- 2011 IBM Watson: Beating human champions in Jeopardy
- 2014 Deep face algorithm Facebook

Reached 97.35% accuracy
Human performance is around 97%

Conquests of machine learning

- 1952 Arthur Samuel (IBM): First machine learning program playing checkers

- 1997 IBM Deep Blue Beats Kasparov
- 2011 IBM Watson: Beating human champions in Jeopardy
- 2014 Deep face algorithm Facebook

- 2016 Alpha go: deep learning

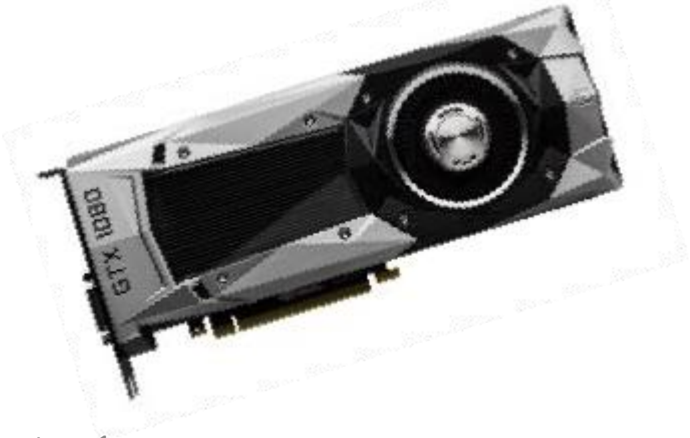
Fan Hui (5-0)
Lee Sedol (4-1)
99.8% win rate against other Go programs

Deep learning - why now?

1. Appearance of machine learning methods and frameworks, optimization know-how, new tools for rapid experimentation

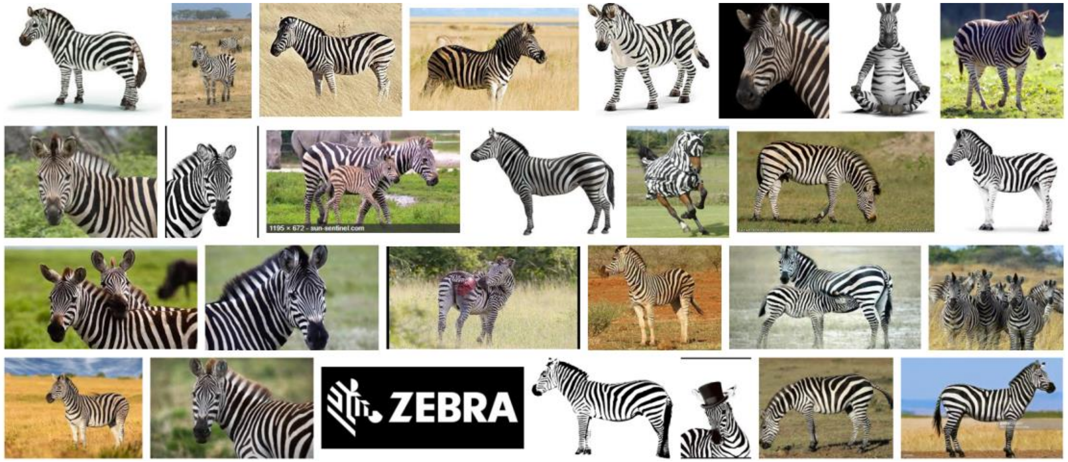
Deep learning - why now?

1. Appearance of machine learning methods and frameworks, optimization know-how, new tools for rapid experimentation
2. New architectures are available for computation
   - (1980: VIC-20 5kb RAM, MOS 6502 CPU 1.02Mhz)
   - (2018: NVIDIA GeForce GTX 1080, 8GB RAM, 1733 MHz, 2560 cores)



Deep learning - why now?

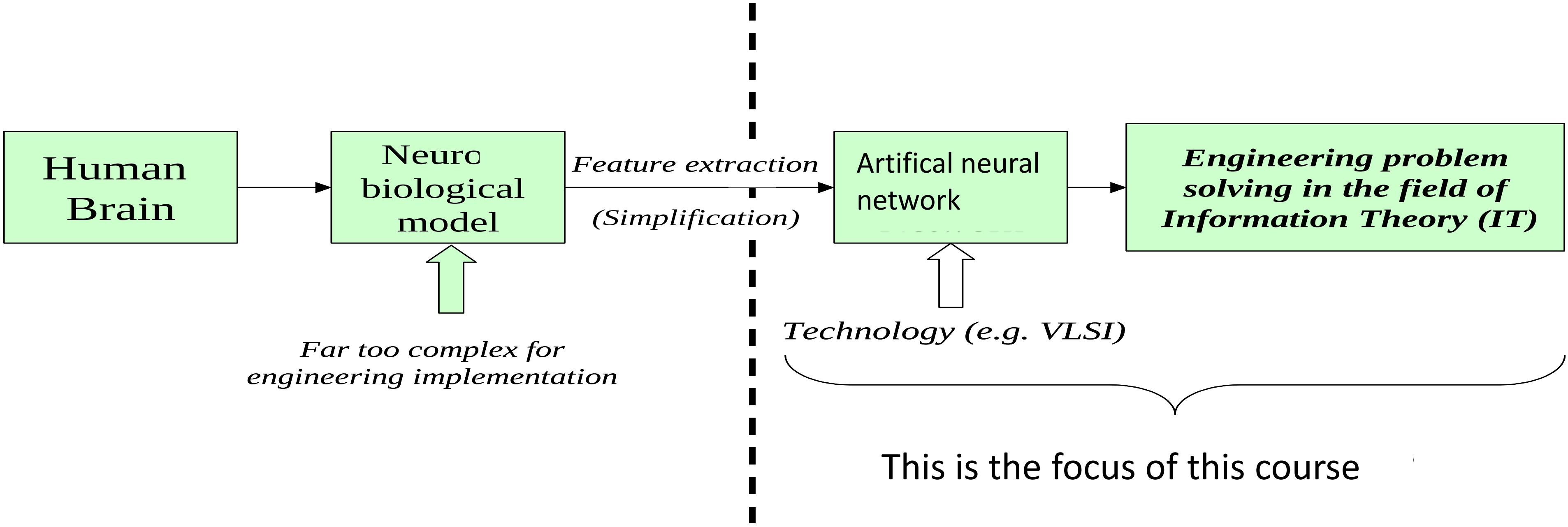
1. Appearance of machine learning methods and frameworks, optimization know-how, new tools for rapid experimentation
2. New architectures are available for computation
   - (1980: VIC-20 5kb RAM, MOS 6502 CPU 1.02Mhz)
   - (2018: NVIDIA GeForce GTX 1080, 8GB RAM, 1733 MHz, 2560 cores)
3. Vast amount of data is available
   - Billions of labeled images available quasi free



Outline

- Administration: requirements of the course
- Machine learning – Machine intelligence
- Artificial neuron
- Perceptron

Copying the brain?

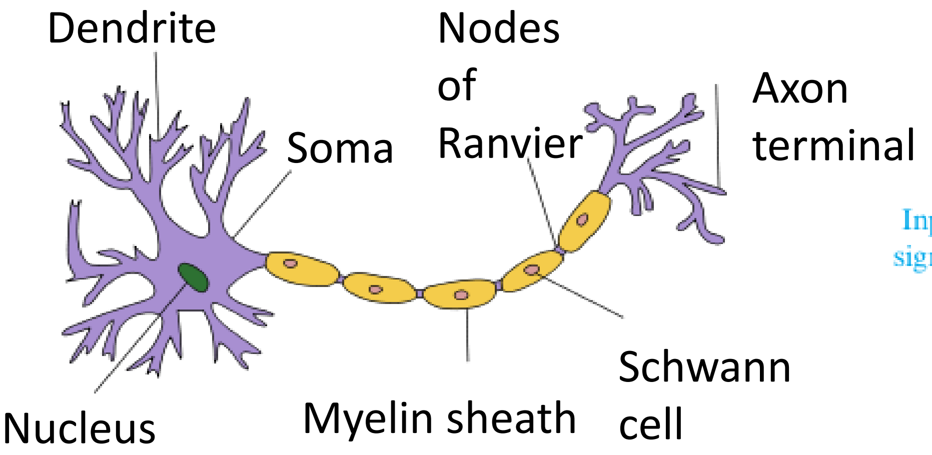


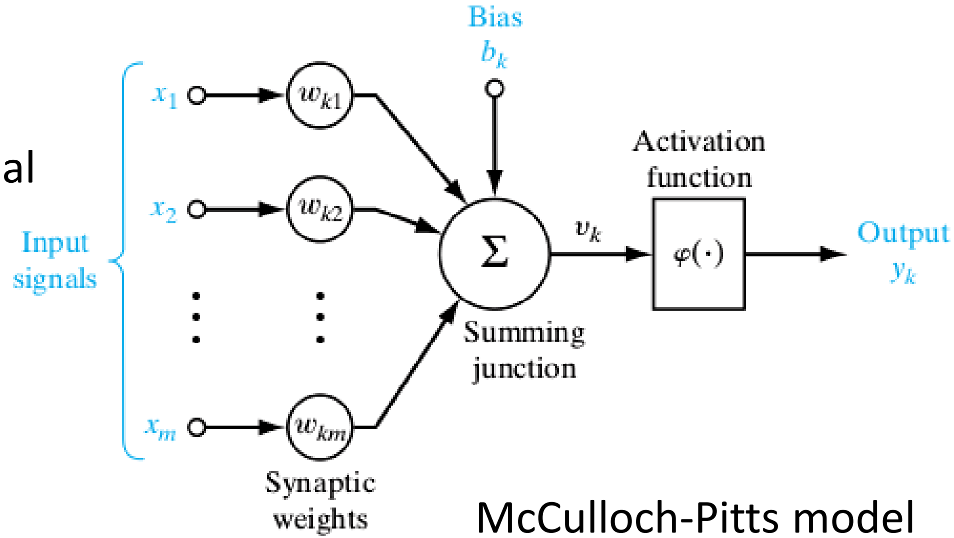
History of the artificial neural networks

- Artificial neuron model, 40's (McCulloch-Pitts, J. von Neumann);
- Synaptic connection strenghts increase for usage, 40's (Hebb)
- Perceptron learning rule, 50's (Rosenblatt);
- ADALINE, 60's (Widrow)
- Critical review ,70's (Minsky)
- Feedforward neural nets, 80's (Cybenko, Hornik, Stinchcombe..)
- Back propagation learning, 80's (Sejnowsky, Grossberg)
- Hopfield net, 80's (Hopfield, Grossberg);
- Self organizing feature map, 70's - 80's (Kohonen)
- CNN, 80's-90's (Roska, Chua)
- PCA networks, 90's (Oja)
- Applications in IT, 90's - 00's
- SVMs, statistical machines 2000-2010's
- Deep learning, Convolutional Neural Networks 2010-

The artificial neuron (McCulloch-Pitts)

- The artificial neuron is an information processing unit that is basic constructing element of an artificial neural network.
- Extracted from the biological model

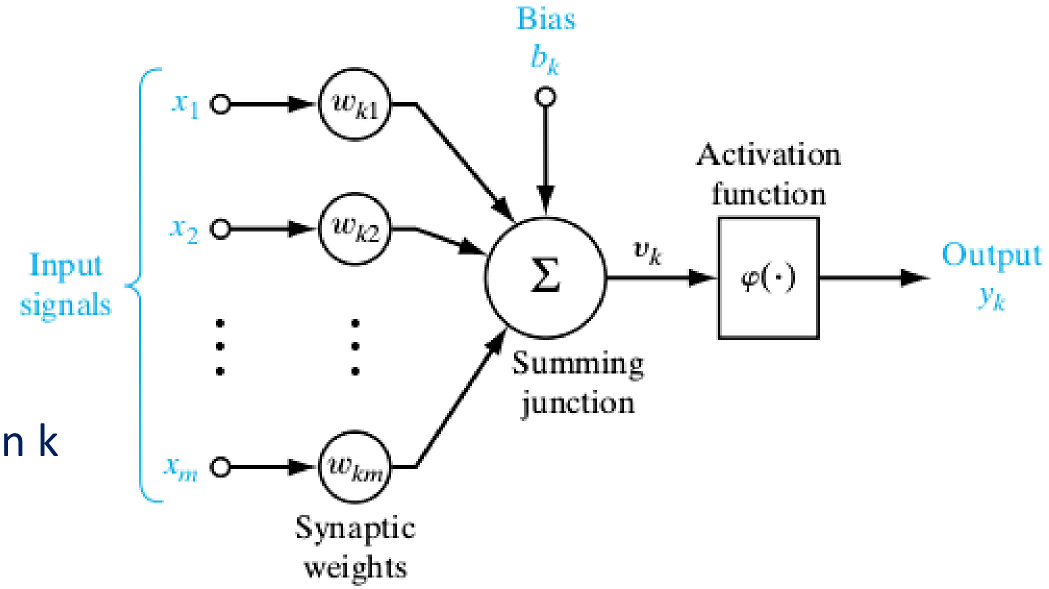




McCulloch-Pitts model

The artificial neuron

- Receives input through its synapsis (x_i)
- Synapsis are weighted (w_i)
  - if $w_i > 0$: amplified input from that source (excitatory input)
  - if $w_i < 0$: attenuated input from that source (inhibitory input)
- A *b* value biases the sum to enable asymmetric behavior
- A weighted sum is calculated
- Activation function shapes the output signal



x_i : input vector
w_{ki} : weight coefficient vector of neuron k
b_k : bias value of neuron k
o_k : output value of neuron k

The artificial neuron

- Output equation:

$$y_k = \varphi\left(\sum_{i=1}^{m} w_{ki} x_i + b_k\right)$$

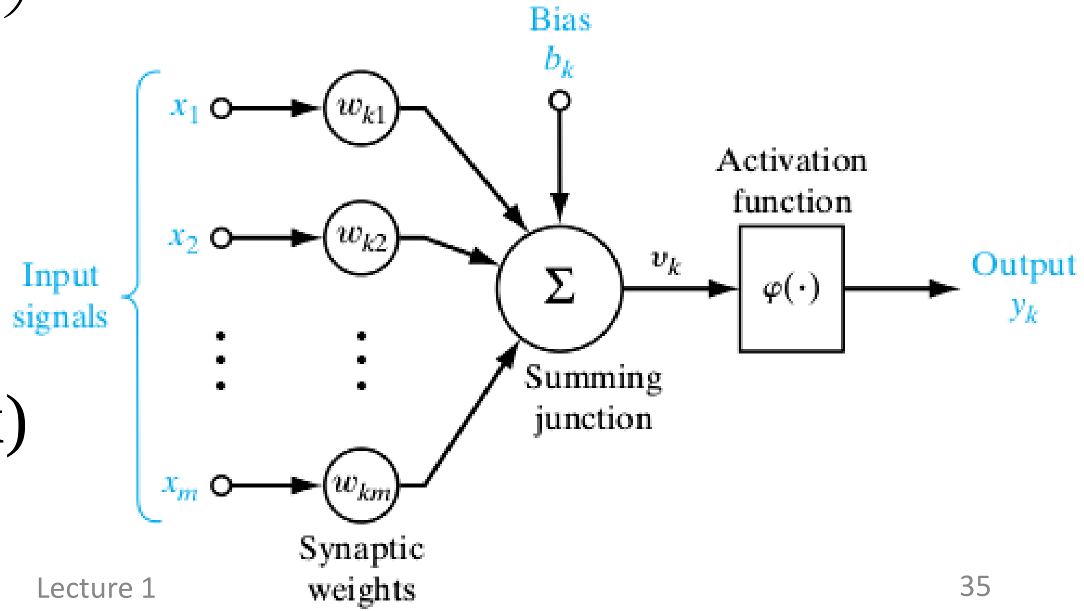
x_i : input vector (*i: 1….m*)
w_{ki} : weight coefficient vector of neuron k
b_k : bias value of neuron k
o_k : output value of neuron k

- Bias can be included as:

$w_0 = b$

$x_0 = 1$

$$y_k = \varphi\left(\sum_{i=0}^{m} w_{ki} x_i\right) = \varphi(\mathbf{w}^T \mathbf{x})$$



Activation functions (1)

- Activation function: $\varphi(.)$
  - Always a nonlinear function
  - Typically it clamps the output (introduces boundaries)
  - Monotonic increasing function
  - Differentiable
    - Important from theoretical point of view
  - Or at least continuous (except in simplified cases)
    - Sophisticated training algorithms require continuity

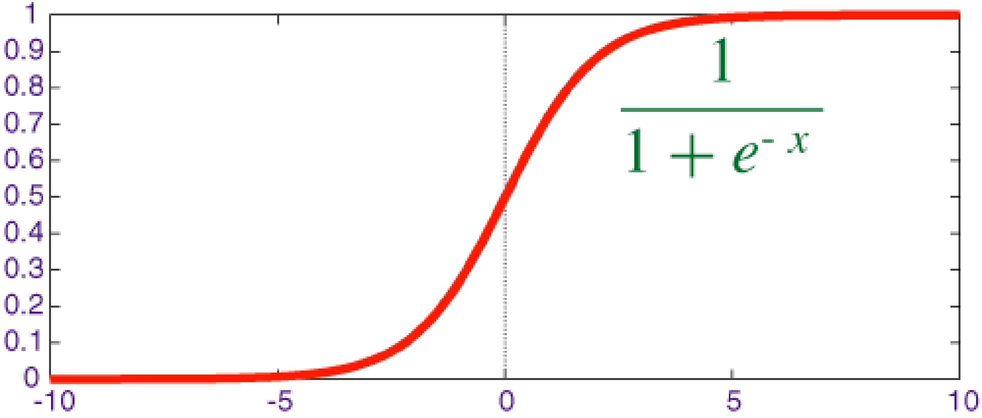
Activation functions (2)

- **Sigmoid** (or logistic) function is a widely activation function

$$y = \varphi(u) = \frac{1}{1+e^{-\lambda u}}$$

- where

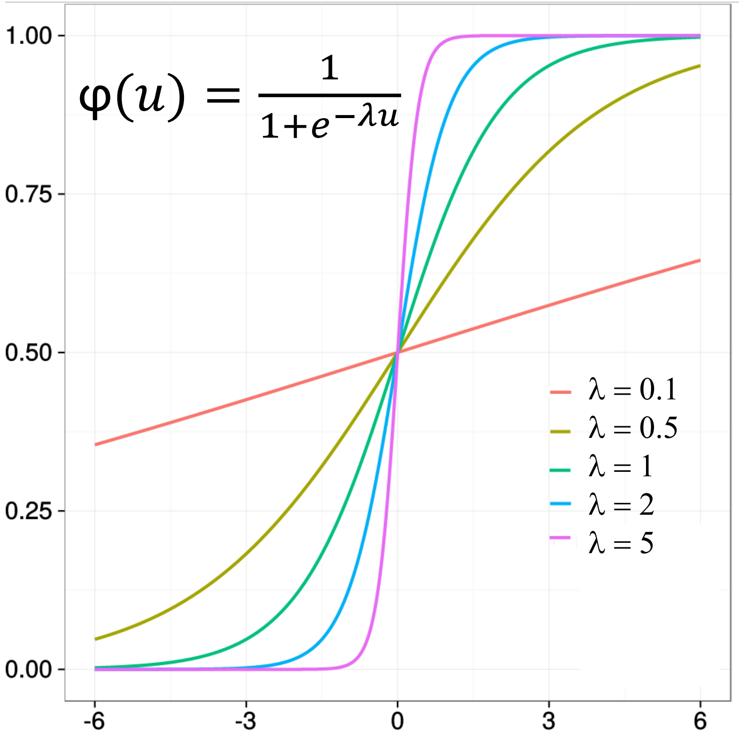
$$u = \sum_{i=0}^{m} w_i x_i = \mathbf{w}^T \mathbf{x}$$



Activation functions (3)

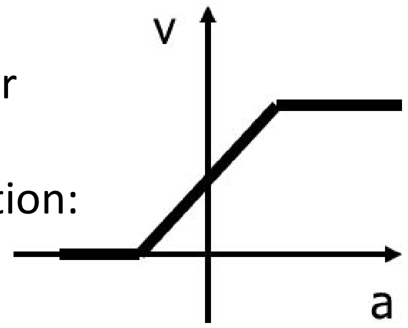
soft nonlinearity

(continuously differentiable)



hard nonlinearity

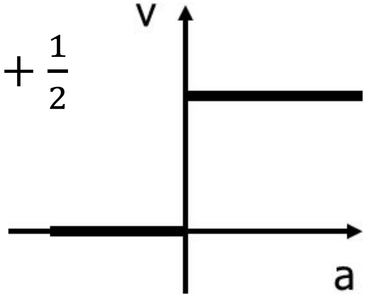
piece-wise linear implementation of sigmoid function:



$\lambda \longrightarrow \infty$

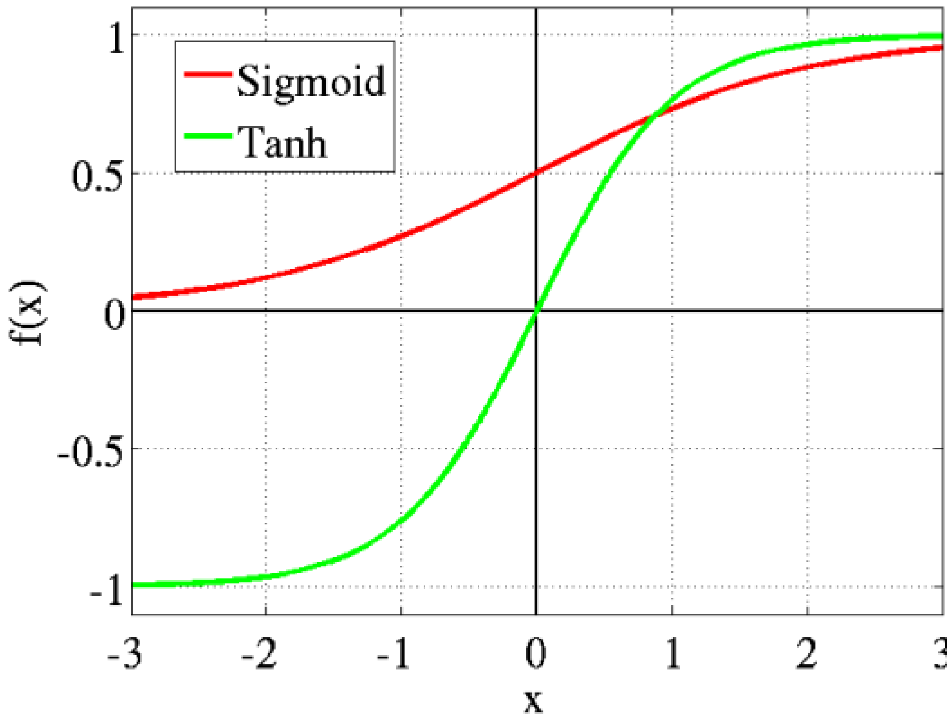
Step (threshold)function

$$\varphi(u) = \frac{sign(u)}{2} + \frac{1}{2}$$



Activation function (4)

- Bipolar activation function: tanh
- Continuously differentiable
- Monotonic
- Useful, when bipolar output is expected
- Hard approximations:
  - Piece-wise
  - Step-wise



Elementary set separation by a single neuron (1)

- Let us use φ(.) step nonlinear function for siplicity:

$$y = \varphi(u) = \frac{sign(u)}{2} + \frac{1}{2} = \begin{cases} 1, \text{ if } u \geq 0 \\ 0, \text{ else} \end{cases}$$

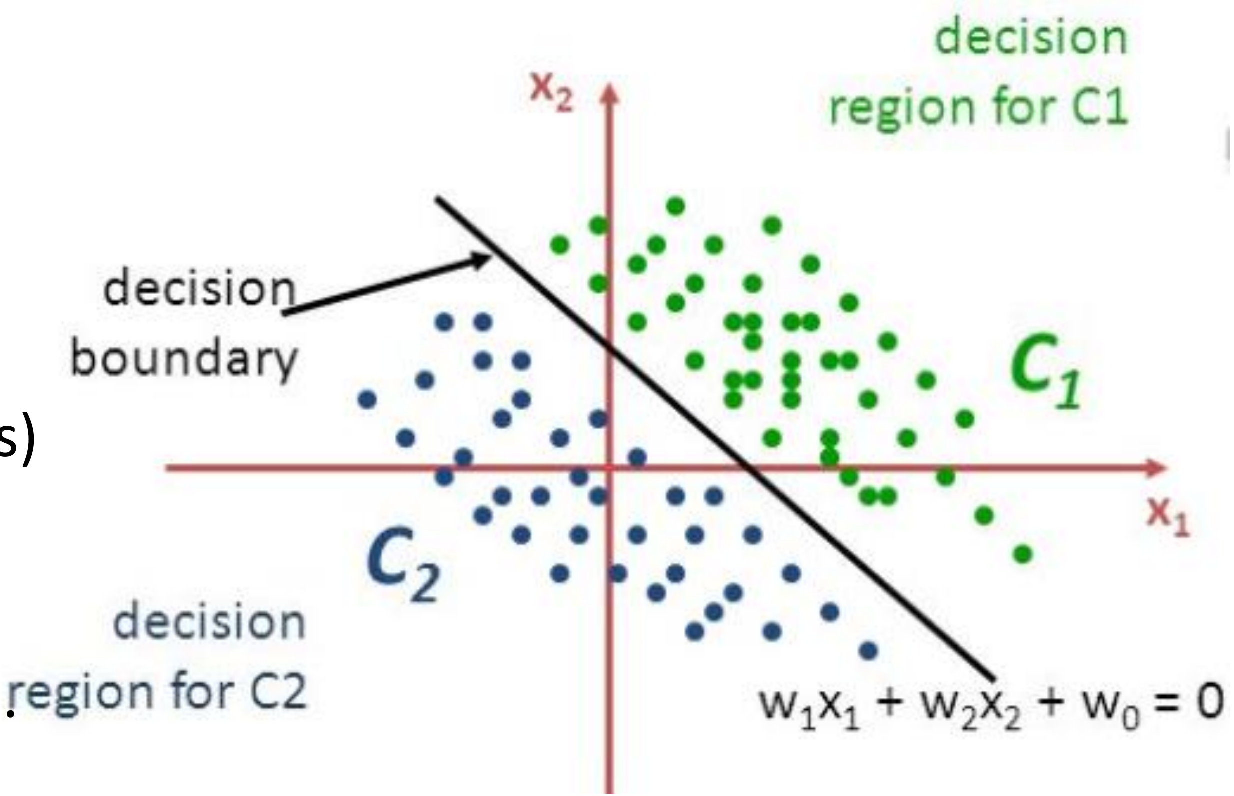
- The output of the neuron will be binary:

$$y = \varphi(u) = \frac{sign(\mathbf{w}^T \boldsymbol{x})}{2} + \frac{1}{2} = \begin{cases} 1, \text{ if } \mathbf{w}^T \boldsymbol{x} \geq 0 \\ 0, \text{ else} \end{cases}$$

DECISION!

Elementary set separation by a single neuron (2)

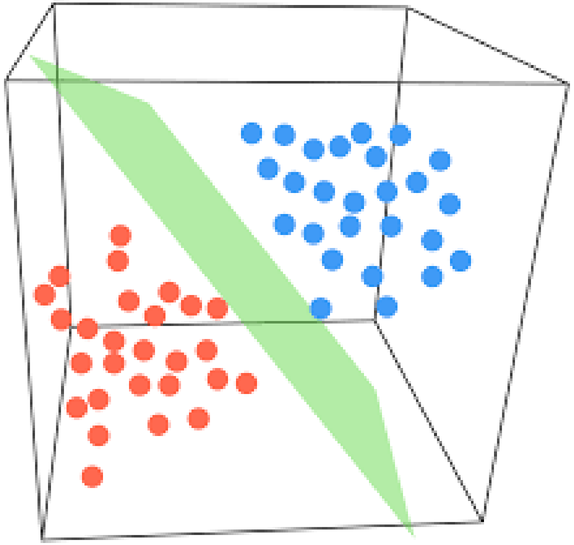
- in a 2-D input space, the hyper plane is a straight line.
- Above the line is classified: +1 (C1: yes)
- Below the line is classified : 0 (C2: no).



Elementary set separation by a single neuron (3)

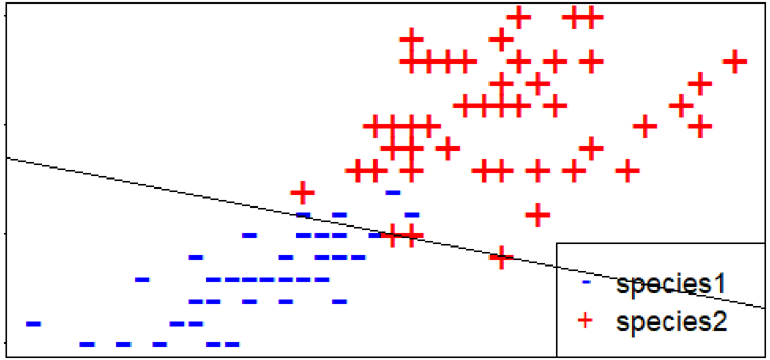
- Neuron with *m* inputs has an *m* dimensional input space
- Neuron makes a linear decision for a 2 class problem
- The decision boundary is a hyperplane defined:

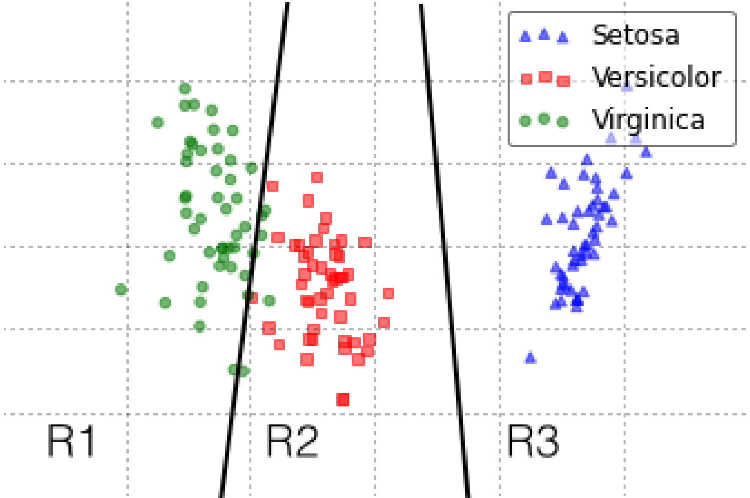
$$\mathbf{w}^T\mathbf{x} = 0$$



Why it is so important to use set separation by hyper plane? (1)

- Most logic functions has this complexity (OR, AND)
- There are plenty of mathematical and computational task which can be derived to a set separation problem by a linear hyper plane
- Application of multiple hyper plane provides complex decision boundary

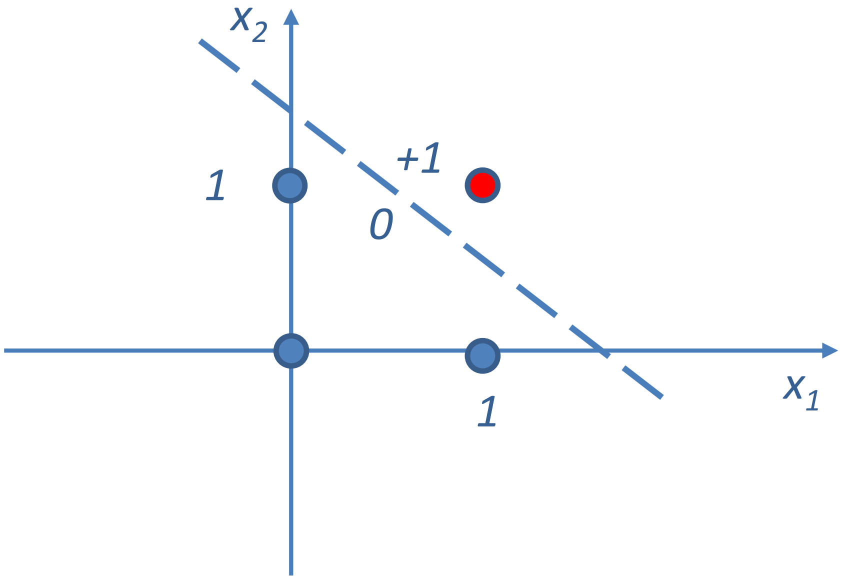




Implementation of a single logical function by a single neuron (1)

AND

x_1	x_2	$x_1 x_2$
0	0	0
0	1	0
1	0	0
1	1	1



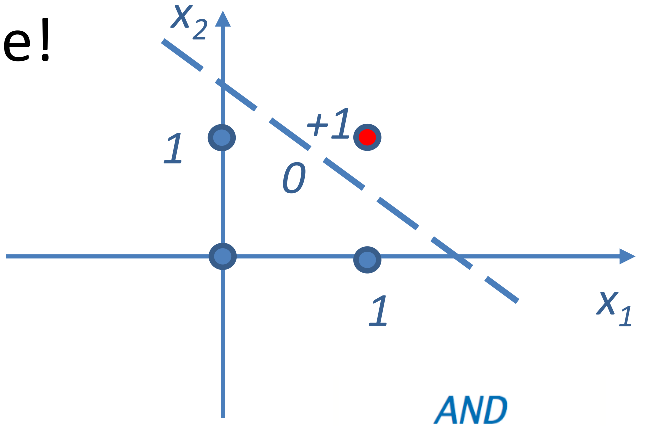
- The truth table of the logical AND function.
- 2-D AND input space and decision boundary

Implementation of a single logical function by a single neuron (2)

- We need to figure out the separation surface!
- Mathematically is the following equation:

$$-1.5 + x_1 + x_2 = 0$$

w_0=-1.5; w_1=1; w_2=1;



- The weight vector is:

$$y = \frac{sign(u)}{2} + \frac{1}{2} = \begin{cases} 1, & if \;\; u \geq 0 \\ 0, & else \end{cases}$$

w = (−1.5, 1, 1).

$$u = \sum_{i=0}^{m} w_i x_i = \mathbf{w}^T \mathbf{x}$$

$$x_0 = 1$$

AND

x_1	x_2	$x_1 x_2$
0	0	0
0	1	0
1	0	0
1	1	1

Implementation of a single logical function by a single neuron (3)

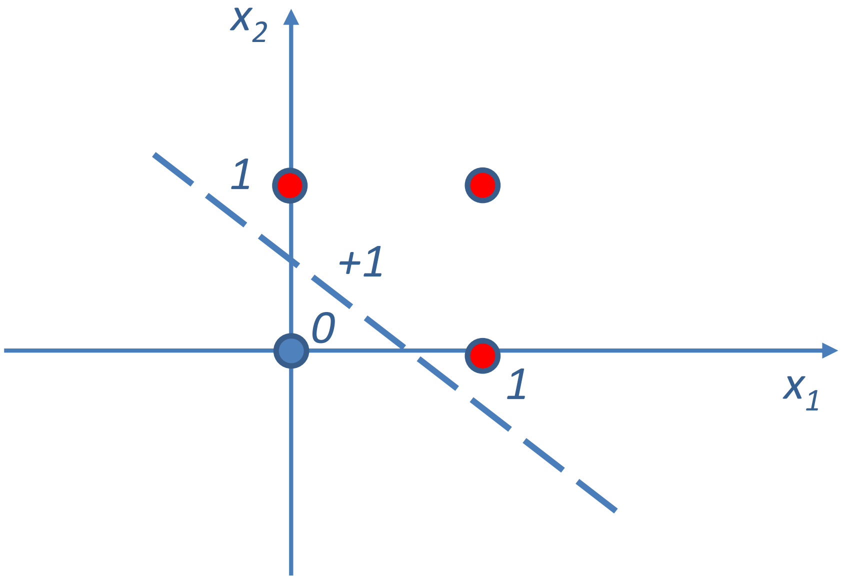
- Furthermore instead of 2D, we can actually come up with the *R* dimensional AND function.
- The weights corresponding to the inputs are all 1 and threshold should be *R* – 0.5. As a result the actual weights of the neuron are the following:

$$\mathbf{w}^{\mathrm{T}} = \left(-\left(R - 0.5\right), 1, \dots, 1\right)$$

Implementation of a single logical function by a single neuron (4)

OR

x_1	x_2	x_1 OR x_2
0	0	0
0	1	1
1	0	1
1	1	1



- The truth table of the logical OR function.

$\mathbf{w}$ =(−0.5, 1, 1).

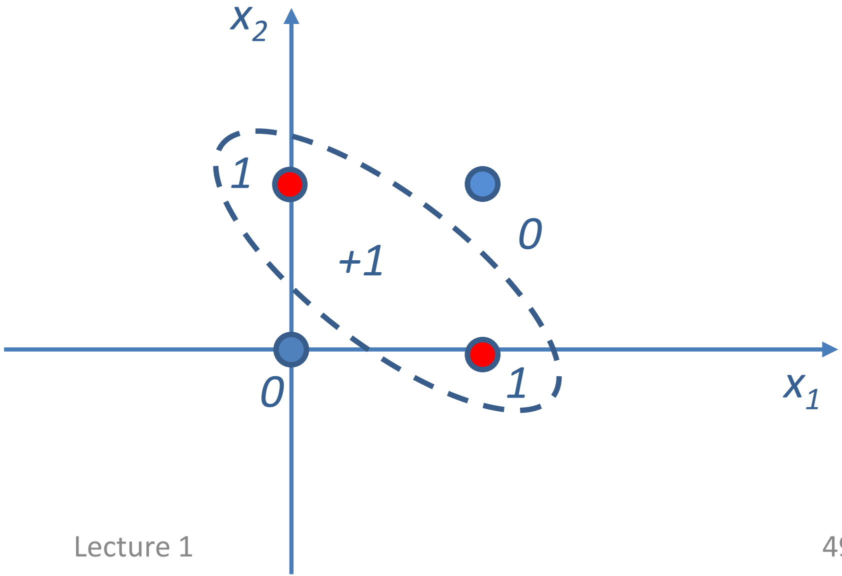
- 2-D OR input space and decision boundary

Implementation of a single logical function by a single neuron (5)

- However we cannot implement every logical function by a linear hyper plane.
- Exclusive OR (XOR) cannot be implemented by a single neuron (linearly not separable)

XOR

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

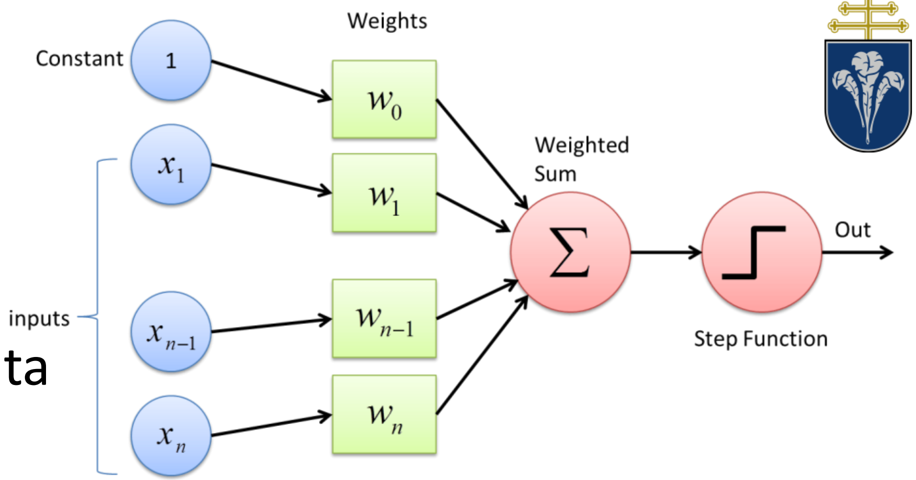


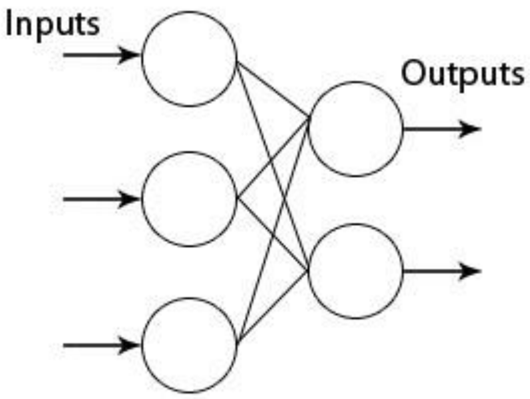
Outline

- Administration: requirements of the course
- Machine learning – Machine intelligence
- Artificial neuron
- Perceptron

Perceptron

- One or a set of neurons sharing the same input
- Typically used for decision making
- Multiple decisions from the same data
- Activation function
  - Originally step function
  - Sigmoid or Tanh or their piece-wise linear approximation is used nowadays
  - Sophisticated training algorithms require differentiable or at least continuous functions

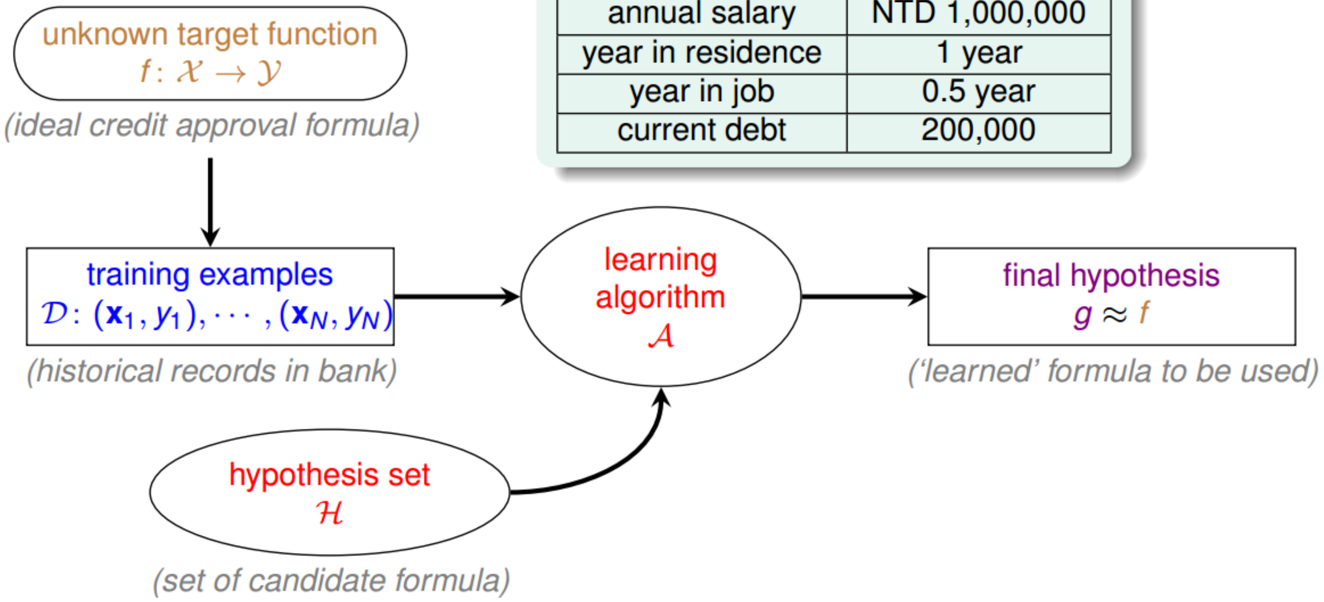




Credit Approval Problem Revisited

Applicant Information

age	23 years
gender	female
annual salary	NTD 1,000,000
year in residence	1 year
year in job	0.5 year
current debt	200,000



what hypothesis set can we use?

A Simple Hypothesis Set: the 'Perceptron'

age	23 years
annual salary	NTD 1,000,000
year in job	0.5 year
current debt	200,000

- For $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ '**features of customer**', compute a weighted 'score' and

$$\text{approve credit if} \quad \sum_{i=1}^{d} w_i x_i > \text{threshold}$$

$$\text{deny credit if} \quad \sum_{i=1}^{d} w_i x_i < \text{threshold}$$

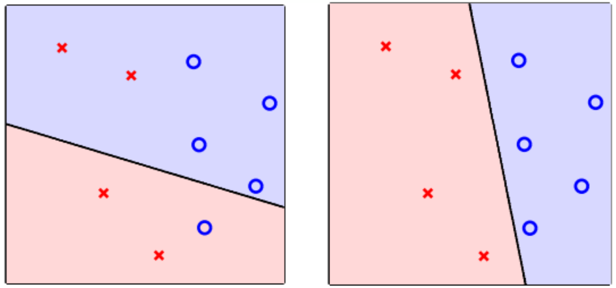
- $\mathcal{Y}$: $\{+1(\textbf{good}), -1(\textbf{bad})\}$, 0 ignored—linear formula $h \in \mathcal{H}$ are

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \text{threshold}\right)$$

called 'perceptron' hypothesis historically

Perceptrons in $\mathbb{R}^2$

$$h(\mathbf{x}) = \text{sign}\left(w_0 + w_1 x_1 + w_2 x_2\right)$$



- customer features $\mathbf{x}$: points on the plane (or points in $\mathbb{R}^d$)
- labels y: $\circ$ (+1), $\times$ (-1)
- hypothesis h: **lines** (or hyperplanes in $\mathbb{R}^d$)
  —positive on one side of a line, negative on the other side
- different line classifies customers differently

perceptrons $\Leftrightarrow$ **linear (binary) classifiers**