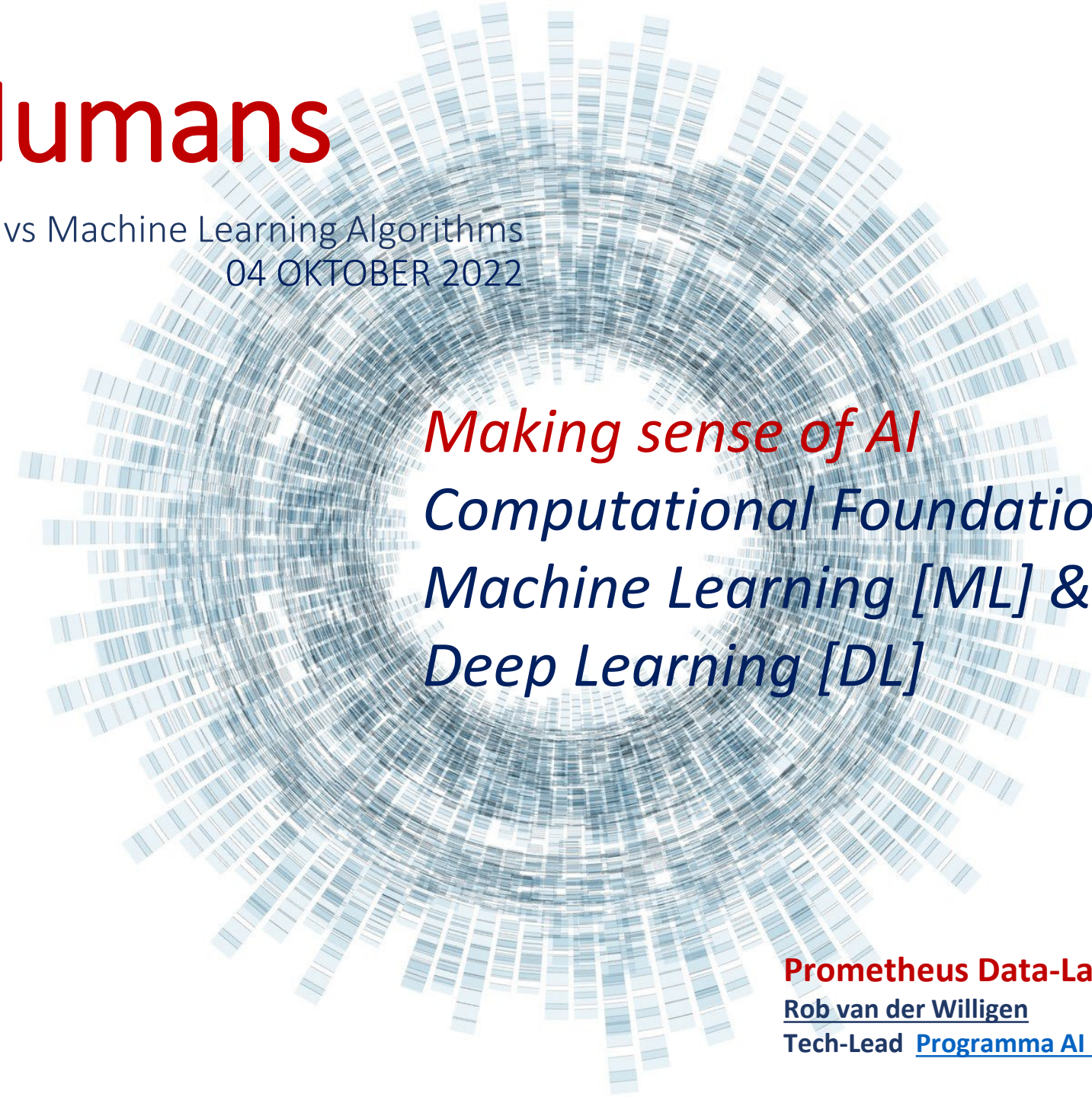
Digital Humans

Lesson 3 + 4: AI models vs Machine Learning Algorithms
04 OKTOBER 2022



Making sense of AI

Computational Foundations of Machine Learning [ML] & Deep Learning [DL]

HOGESCHOOL ROTTERDAM

Prometheus Data-Lab (Wijnhaven 103)
Rob van der Willigen
Tech-Lead Programma AI & Ethics

CONTEXT

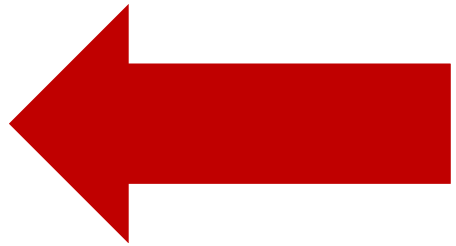
HOGESCHOOL ROTTERDAM

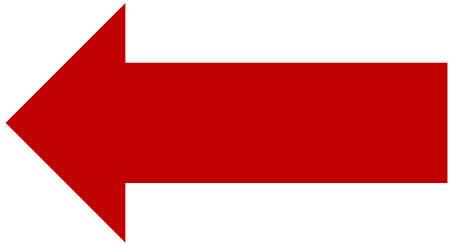
- Prerequisites: basics in linear algebra, probability, and analysis of algorithms.
- Workload: Do it your-self [DIY] assignments

Elements of AI ?

Elements of [any] AI

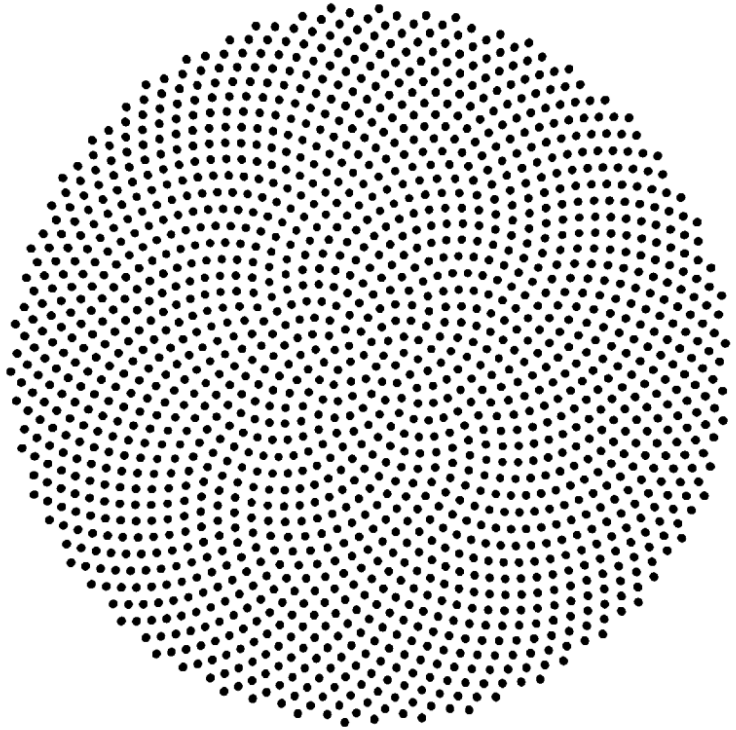
- Needs a [computational?] **Learning Algorithm**
- Needs [unstructured / structured?] **data**
- Needs [input?] **sensors**
- Needs [output?] **effectors**
- Needs [cybernetic?] **modeling**

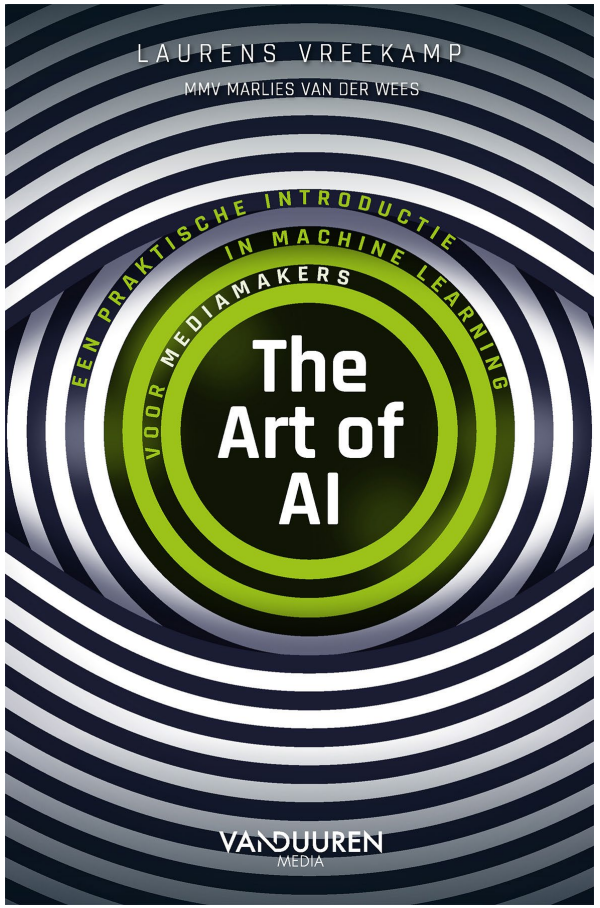




{01}
Fundamentals: [Cybernetic] Modeling

HOGESCHOOL ROTTERDAM





Modeling [Modellen / Modeleren]

- is een hulpmiddel
- is een vereenvoudiging van de werkelijkheid
- helpt om de werkelijkheid te beschrijven en te begrijpen
- is eenduidig

- is niet *per se* normatief en/of voorschrijvend
- is te vertalen naar en te begrijpen in de beoogde probleem context

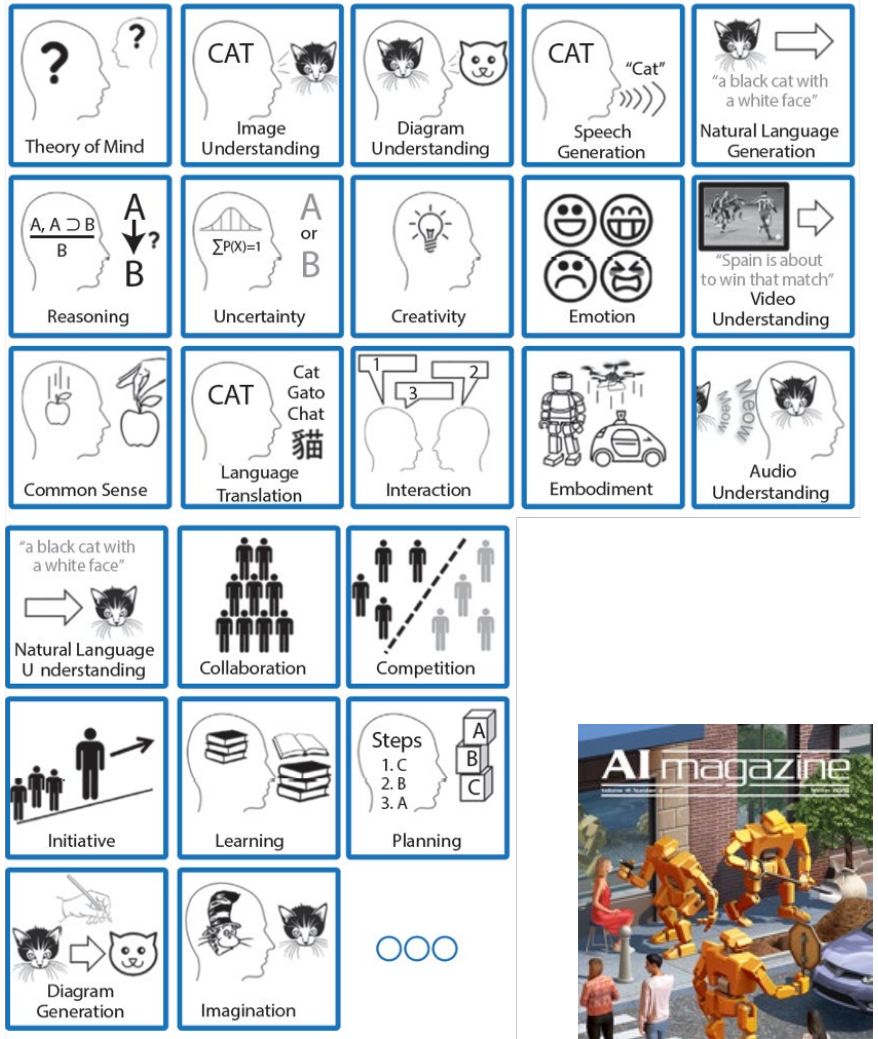
- (ML-)model: een computer programma;
  Het gebruiksklare resultaat van het ML-leerproces (trainen)

"….Een model is een schematische, vaak versimplelde weergave van de werkelijkheid." ……..

HOGESCHOOL ROTTERDAM

{01}
Fundamentals

AI aims to **mimic & automatise** tasks which otherwise require human **perception, cognition** and/or **motor skills.**



AI is Modeled on How Humans Proces Information

Scale (sec)	Time Units	System	World (theory)
10^7	Months		SOCIAL BAND
10^6	Weeks		
10^5	Days		
10^4	Hours	Task	RATIONAL BAND
10^3	10 min	Task	
10^2	Minutes	Task	
10^1	10 sec	Unit task	COGNITIVE BAND
10^0	1 sec	Operations	
10^{-1}	100 ms	Deliberate act	
10^{-2}	10 ms	Neural circuit	BIOLOGICAL BAND
10^{-3}	1 ms	Neuron	
10^{-4}	100µs	Organelle	

Sensation: [Sensibilisatie]

'…immediate and basic experiences generated as stimuli fall on our sensory systems'
➔ *Verwerken van ruwe data (prikkels of Fysieke stimuli) volgens een vast patroon*

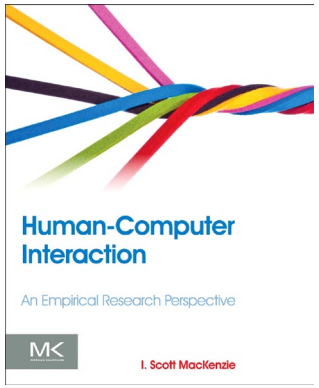
Perception: [Perceptie]

'…interpretation of those sensations, giving them meaning and organization'
➔ *Gestuurd door "ingebouwde" informatie (niet lerend),*

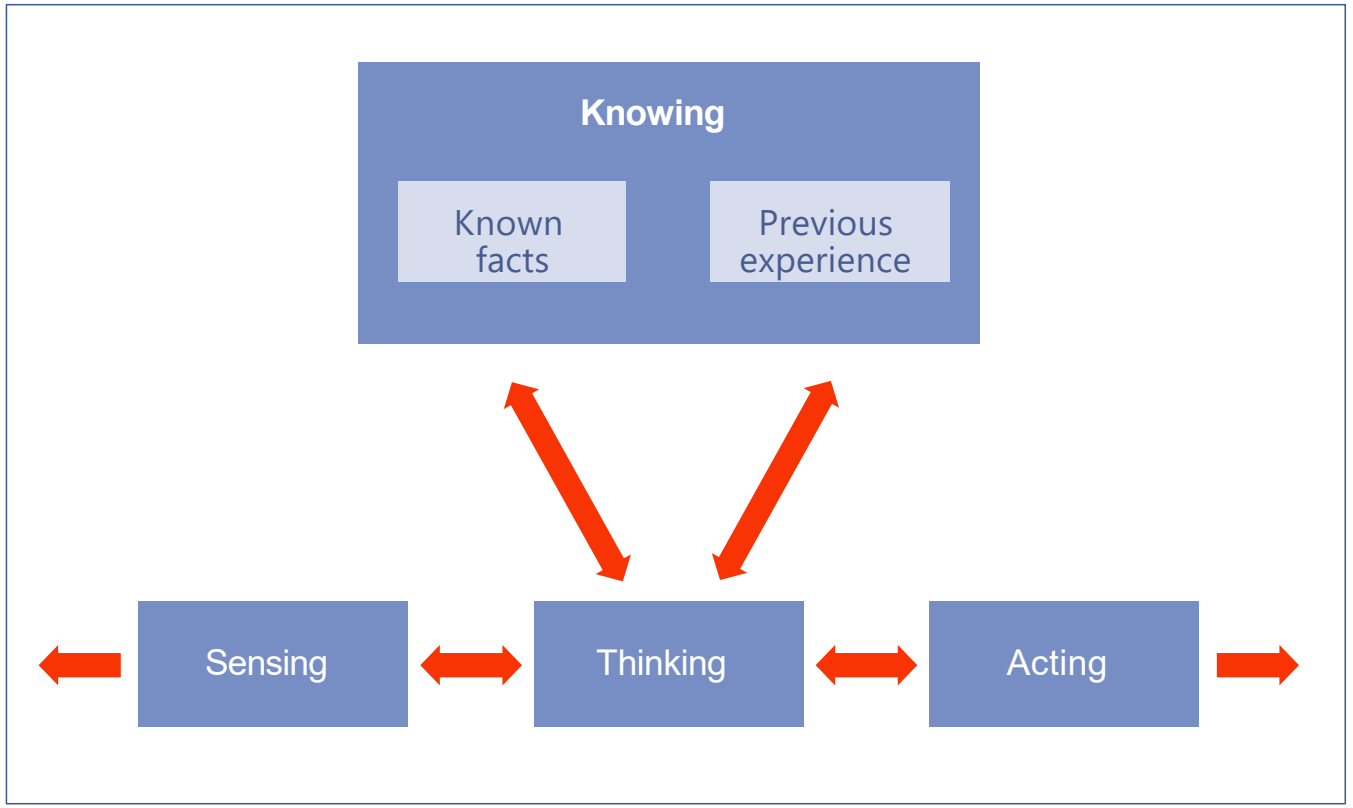
Cognition: [Cognitie]

'…acquisition, storage, retrieval, and use of information'
➔ *Gestuurd door "verworven" informatie (zelf-lerend)*

M.W. Matlin & H.J. Foley, 1992



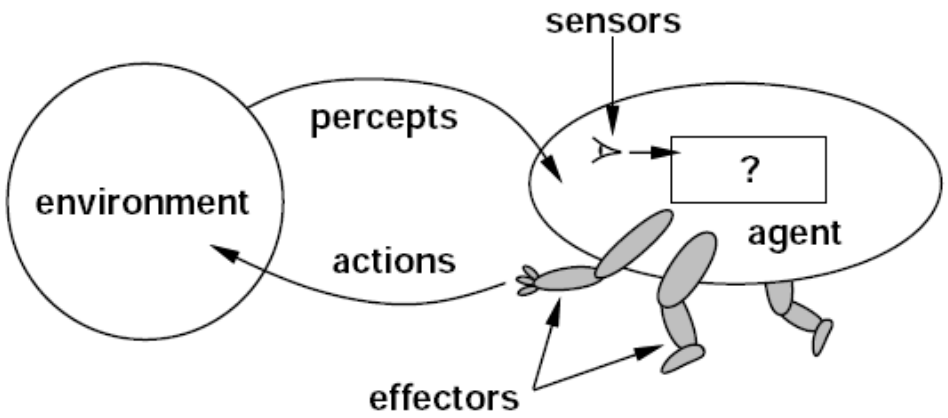
AI is modeled on how humans proces information



The model of mind

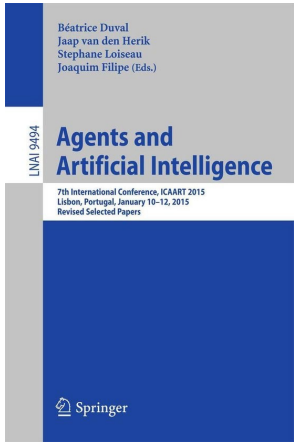
HOGESCHOOL ROTTERDAM

AI is modeled through Agents



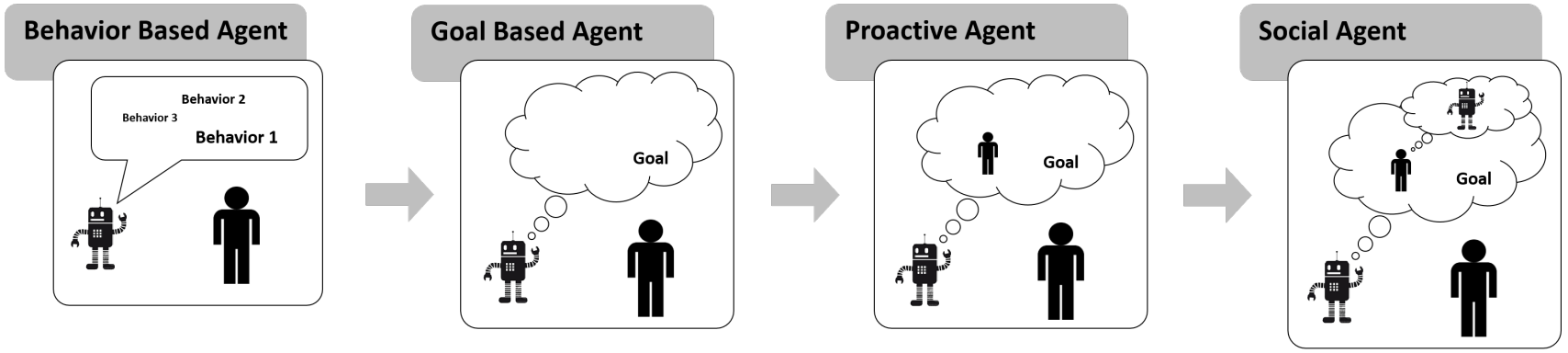
An agent

is anything that can perceive its environment through sensors and acts upon that environment through effectors.



HOGESCHOOL ROTTERDAM

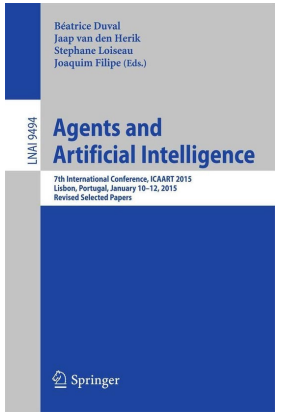
AI is modeled through Agents



Human agent has sensory organs such as eyes, ears, nose, tongue and skin parallel to the sensors, and other organs such as hands, legs, mouth, for effectors.

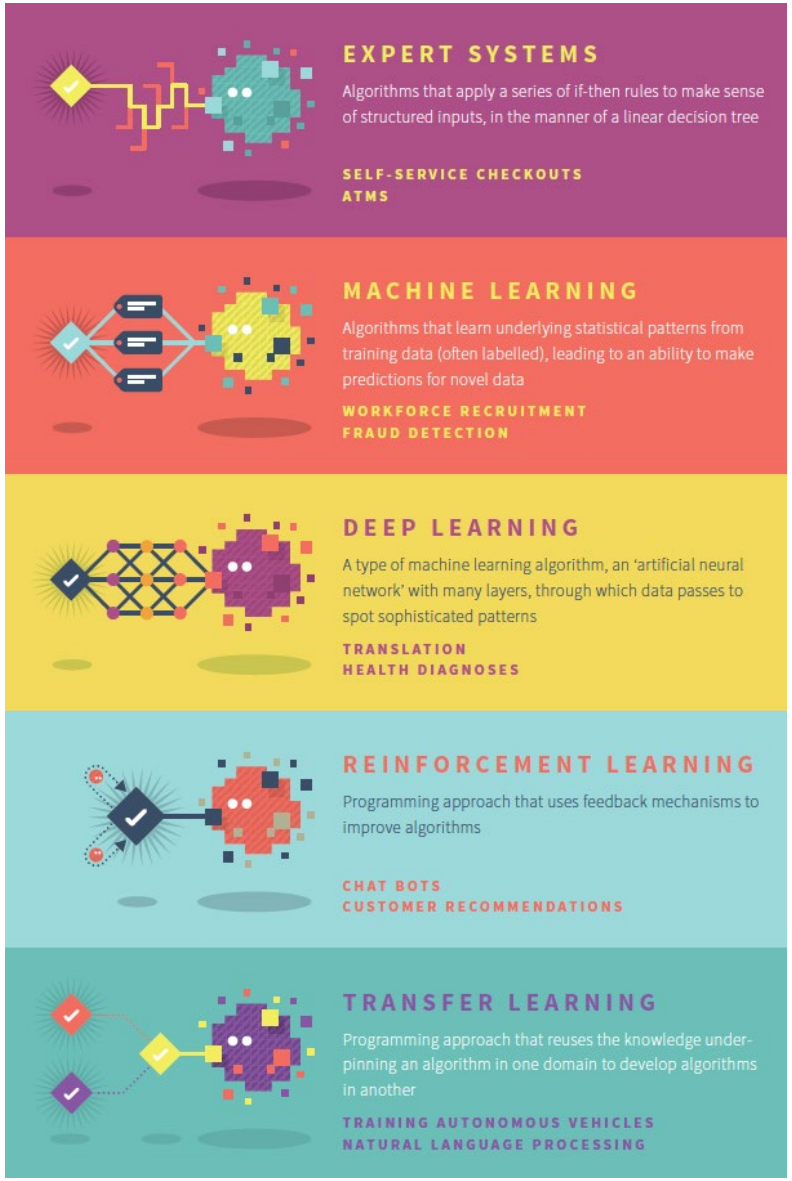
Robotic agent replaces cameras and infrared range finders for the sensors, and various motors and actuators for effectors.

Software agent has encoded bit strings as its programs and actions.



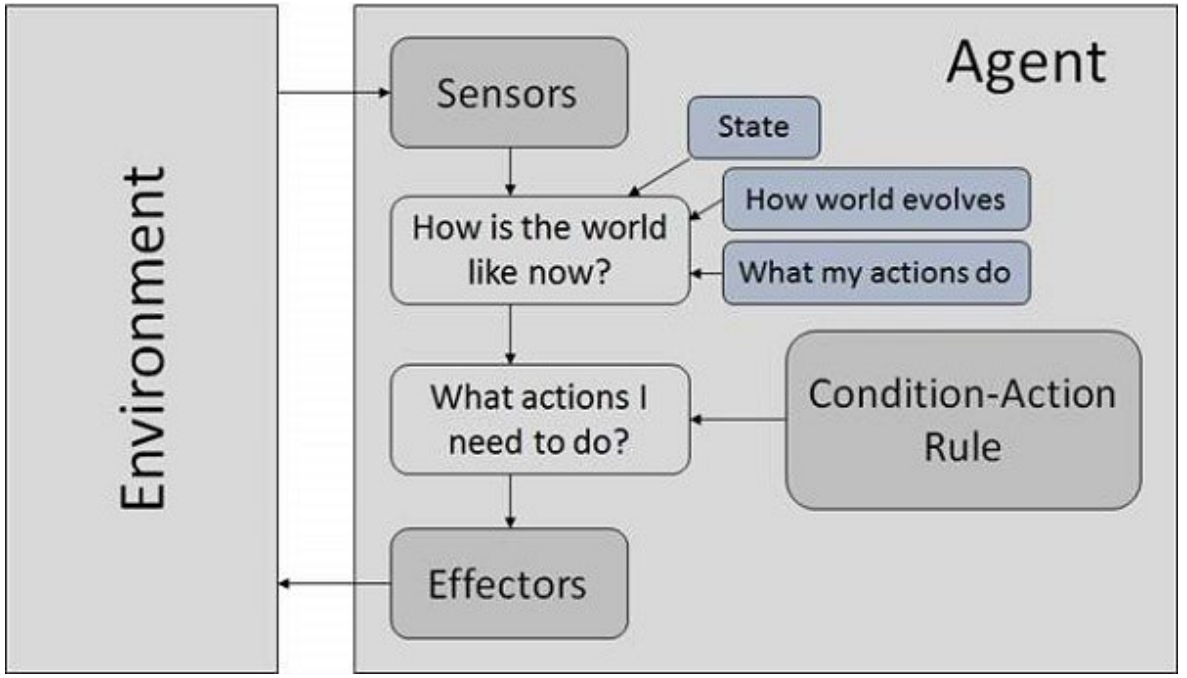
HOGESCHOOL ROTTERDAM

AI is often defined as software agent



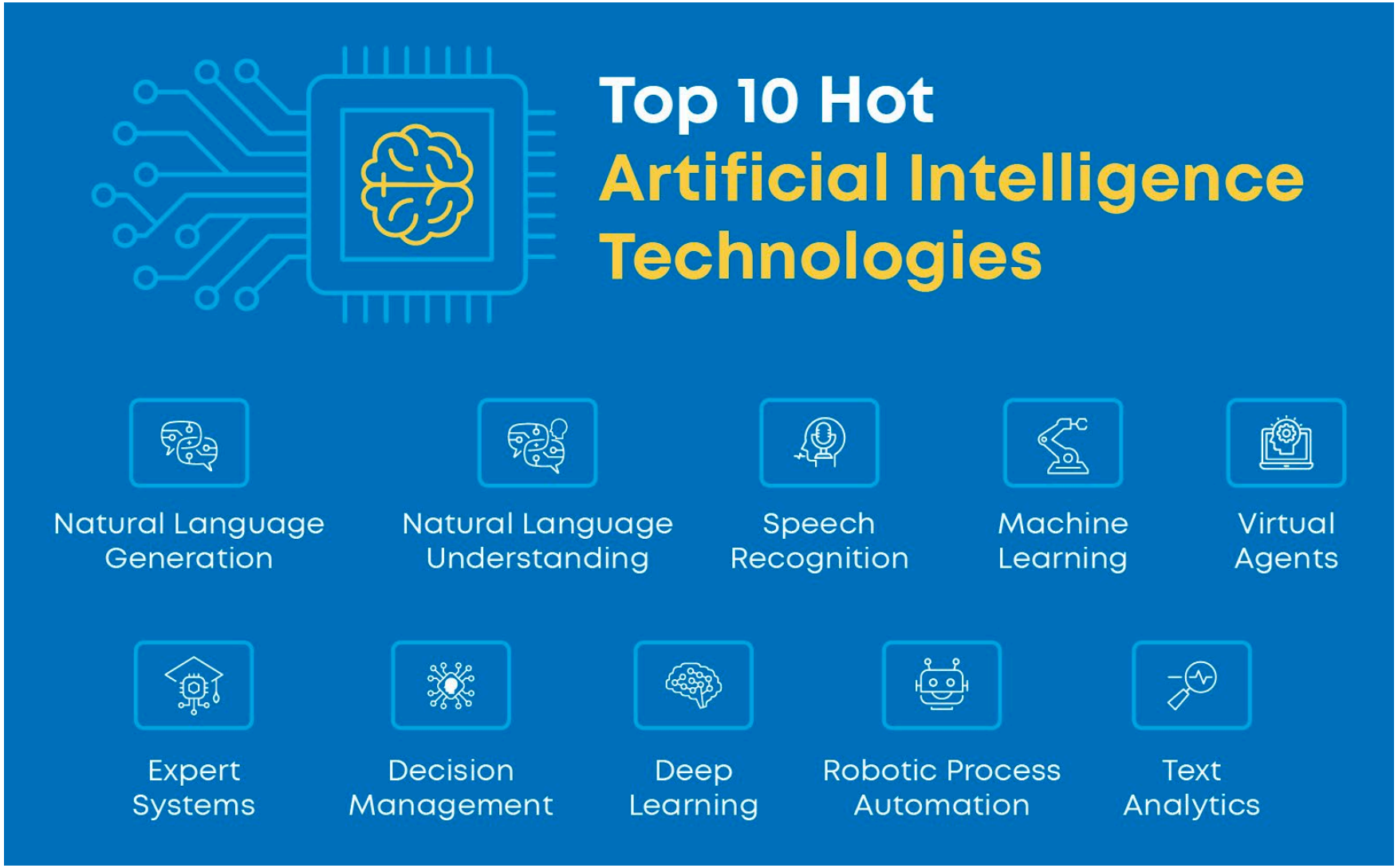
HOGESCHOOL ROTTERDAM

Goal based Robotic Agents need AI (human cognition)

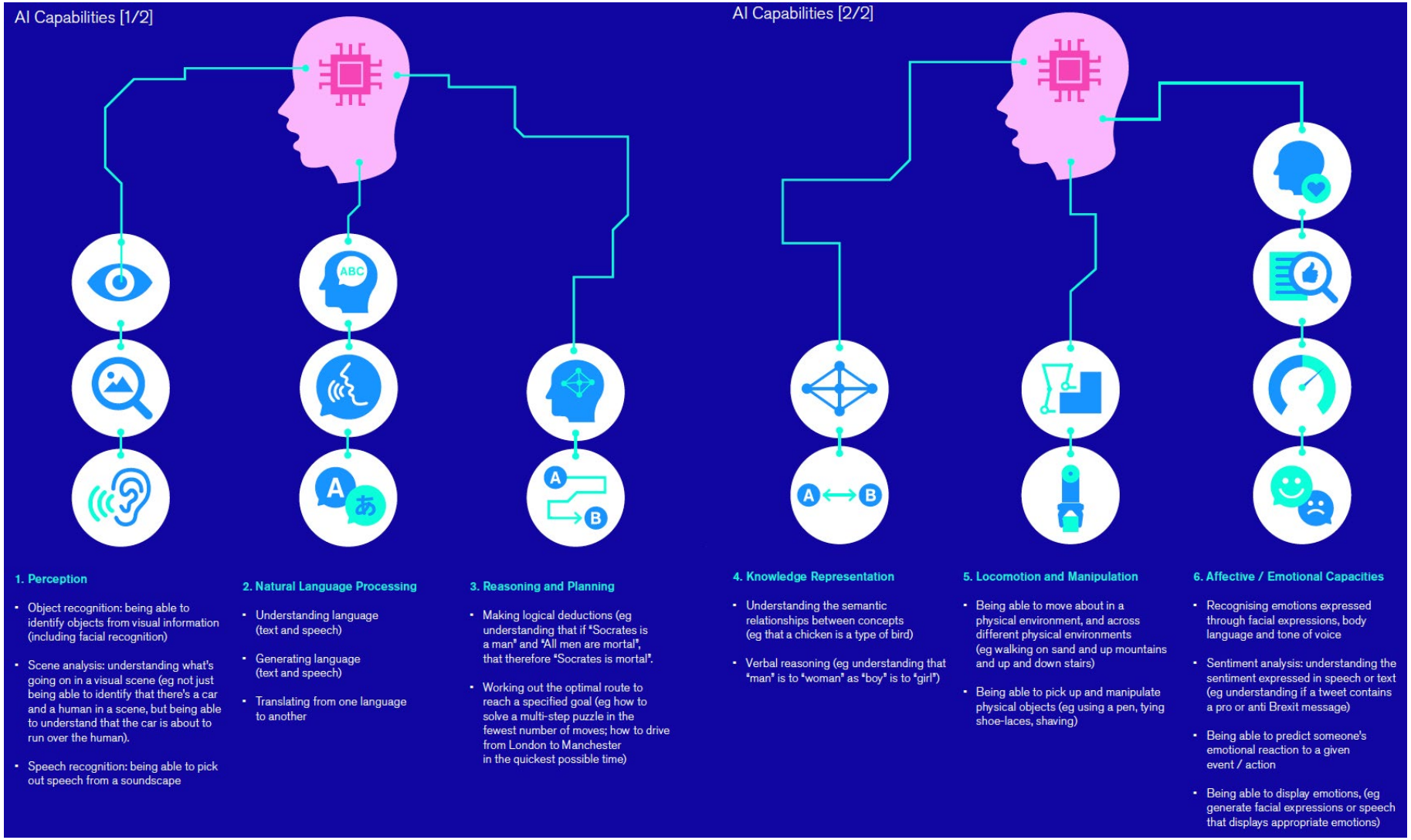


Stanford Encyclopedia of Philosophy

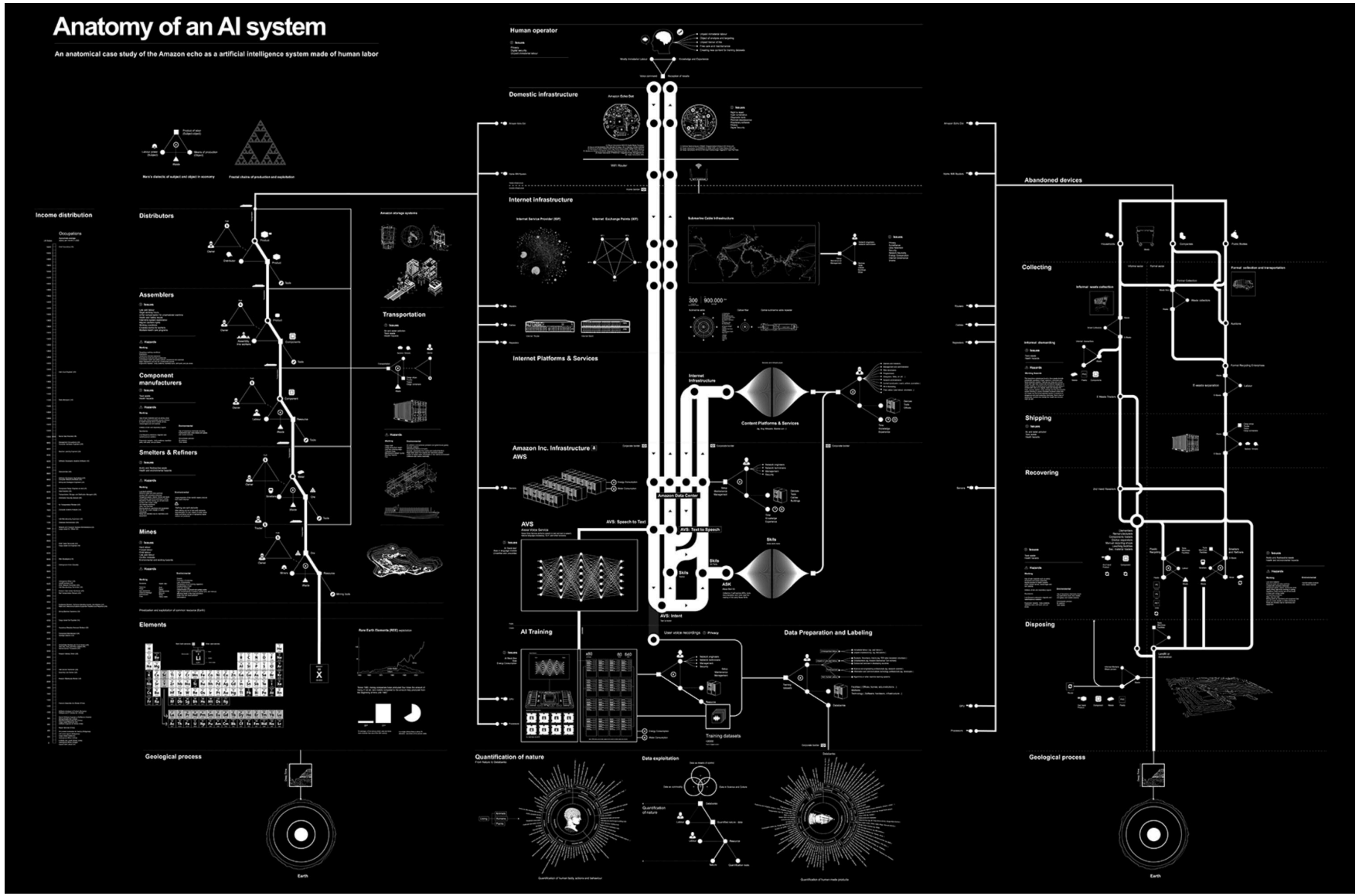
{01}
Fundamentals



{01}
Fundamentals

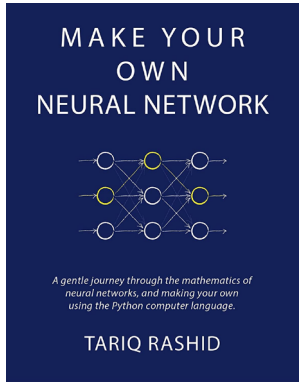


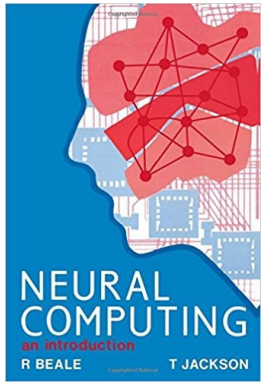
{01}
Fundamentals

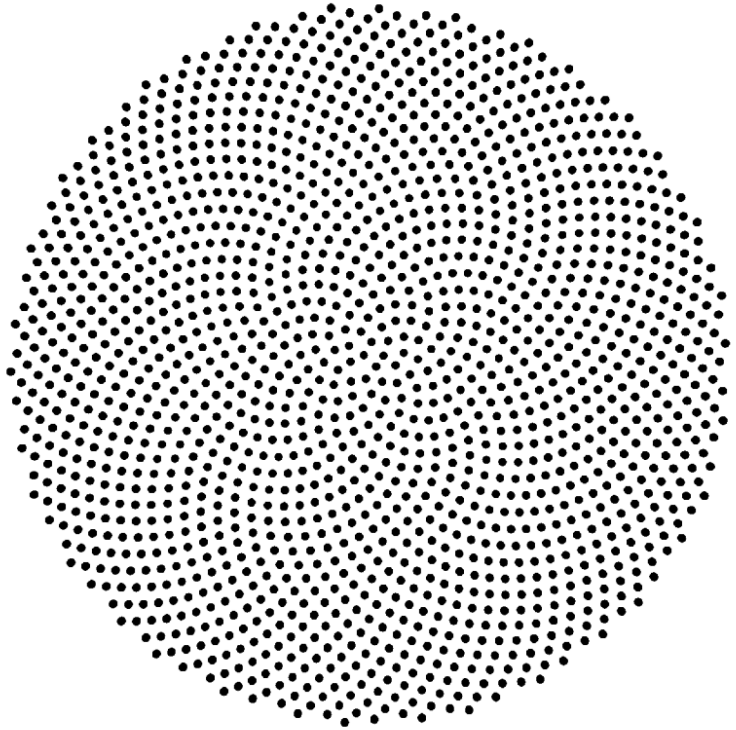


http://www.anatomyof.ai/img/ai-anatomy-map.pdf

{02}
Fundamentals: Learning Algorithms





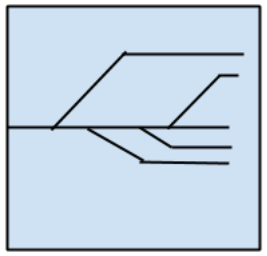


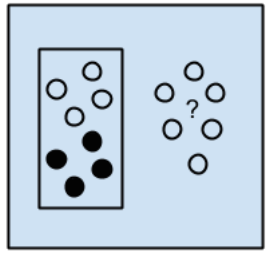
HOGESCHOOL ROTTERDAM

A Tour of Machine Learning Algorithms

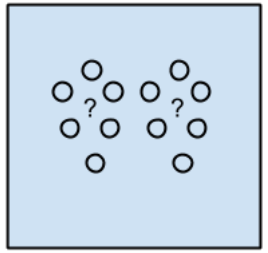
by **Jason Brownlee** on August 12, 2019 in **Machine Learning Algorithms**

Algorithms Grouped by Learning Style

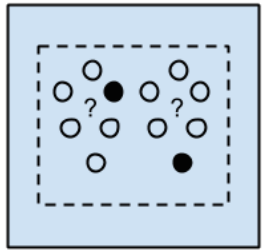

Regularization Algorithms



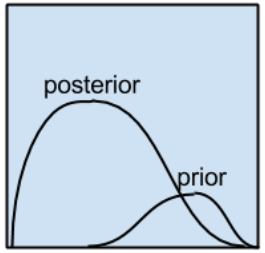
Supervised Learning Algorithms



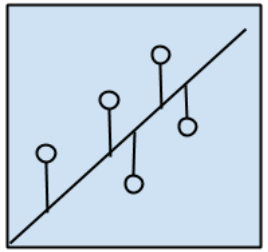
Unsupervised Learning Algorithms

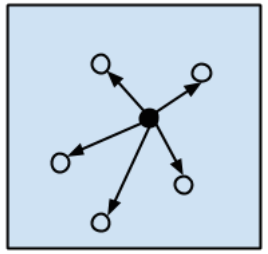


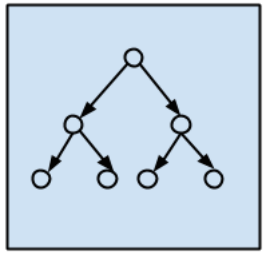
Semi-supervised Learning Algorithms

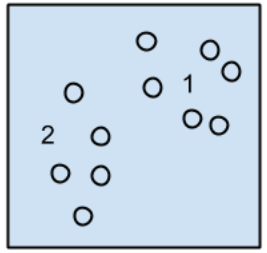


Bayesian Algorithms

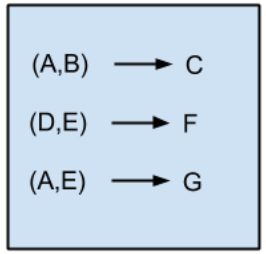

Regression Algorithms


Instance-based Algorithms

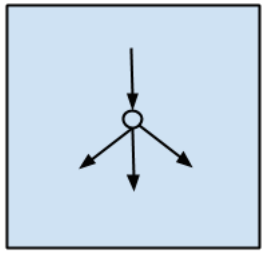

Decision Tree Algorithms

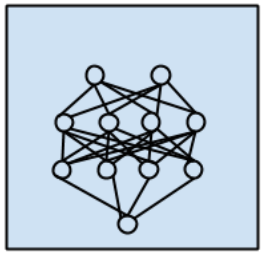


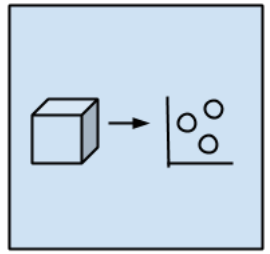
Clustering Algorithms

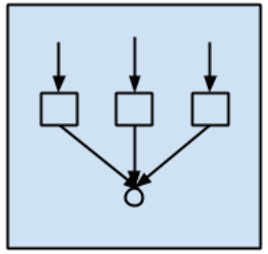


Association Rule Learning Algorithms


Artificial Neural Network Algorithms

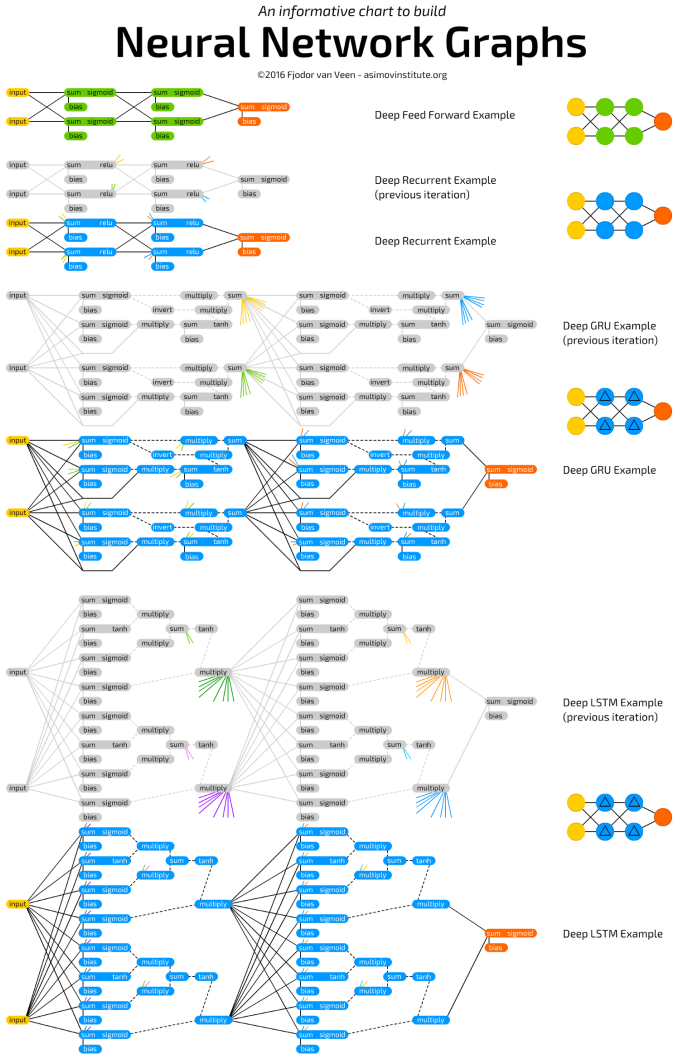

Deep Learning Algorithms

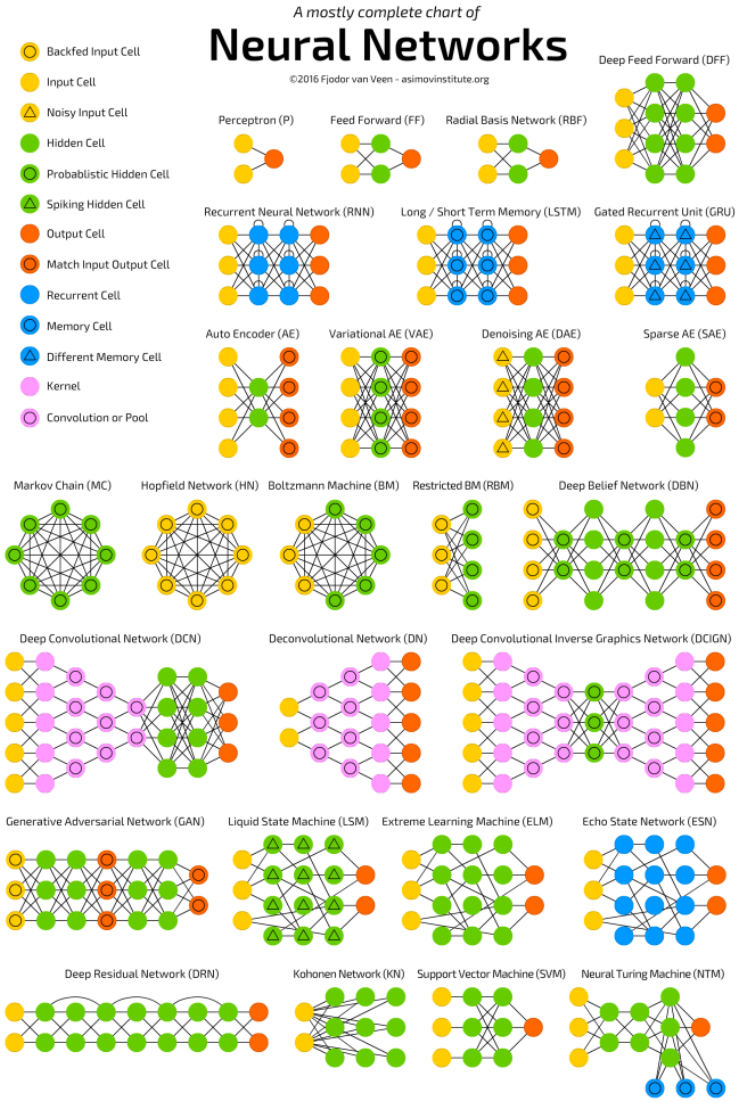

Dimensional Reduction Algorithms


Ensemble Algorithms

https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

{01}
Fundamentals





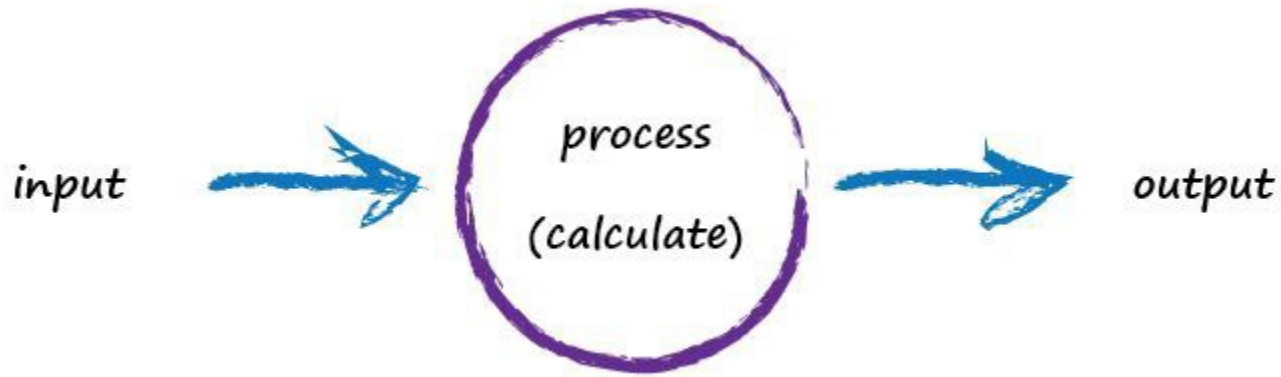
https://www.asimovinstitute.org/neural-network-zoo/

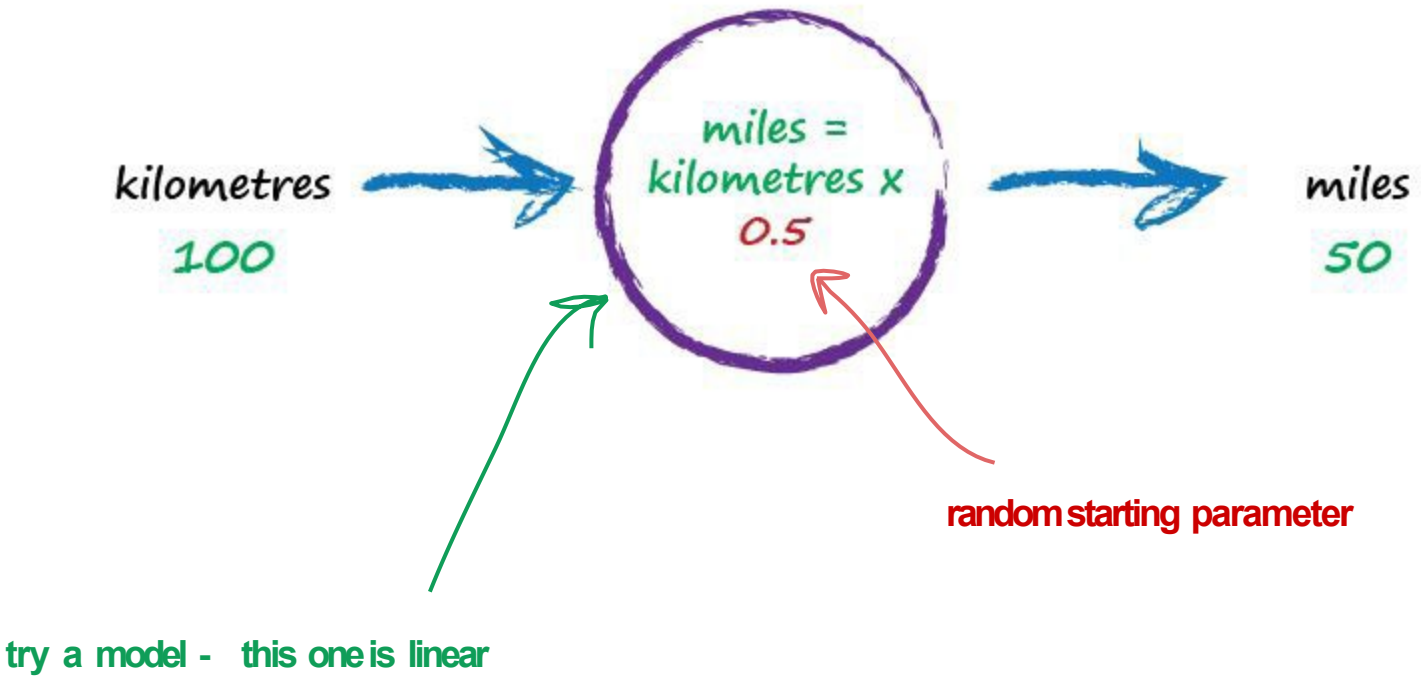
How do Machines / Humans Learn?

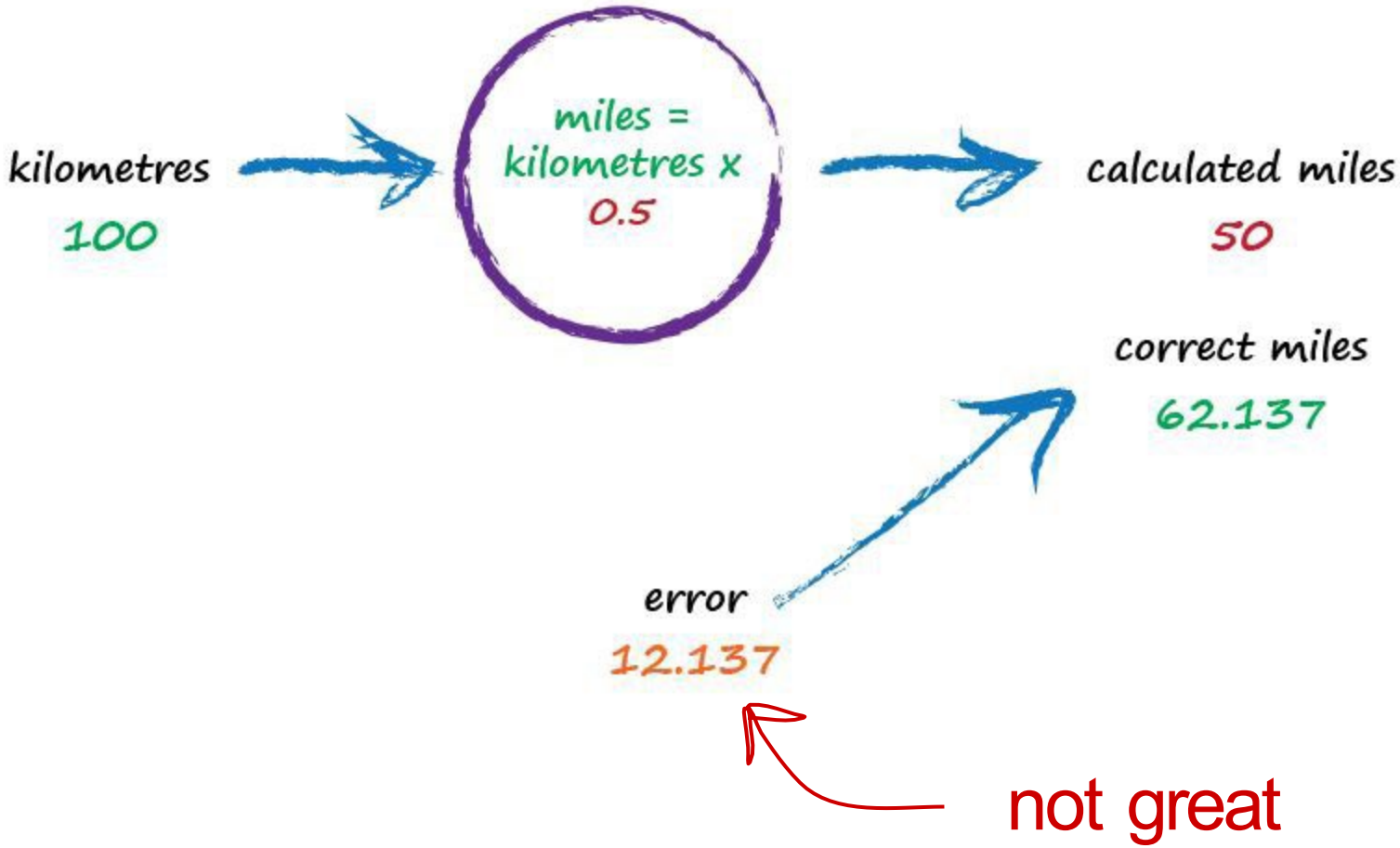
LEARNING NEEDS?

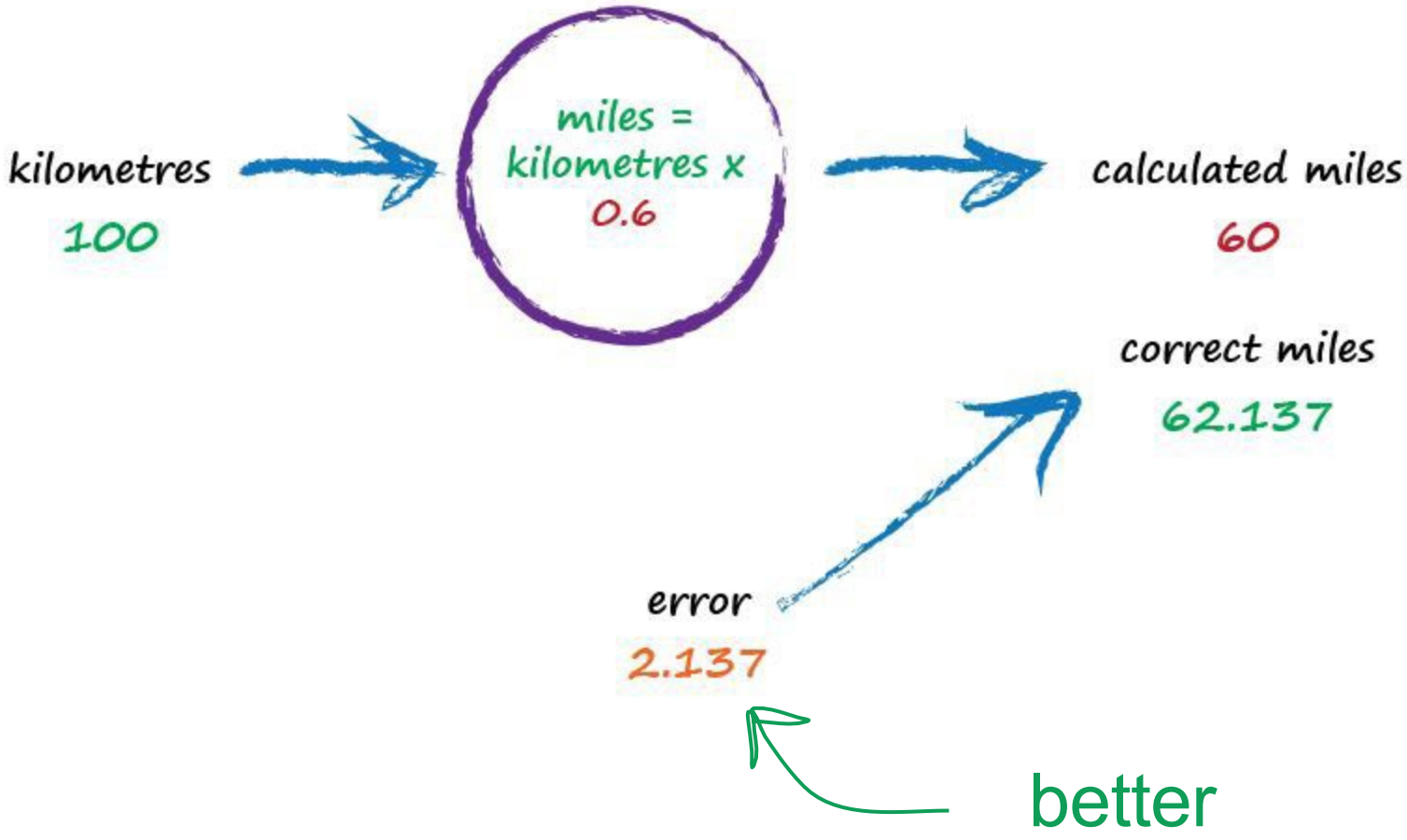
- Needs a [computational?] **Learning Algorithm**
- Needs [unstructured / structured?] **data**
- Needs [input?] **sensors**
- Needs [output?] **effectors**
- Needs [cybernetic?] **modeling**

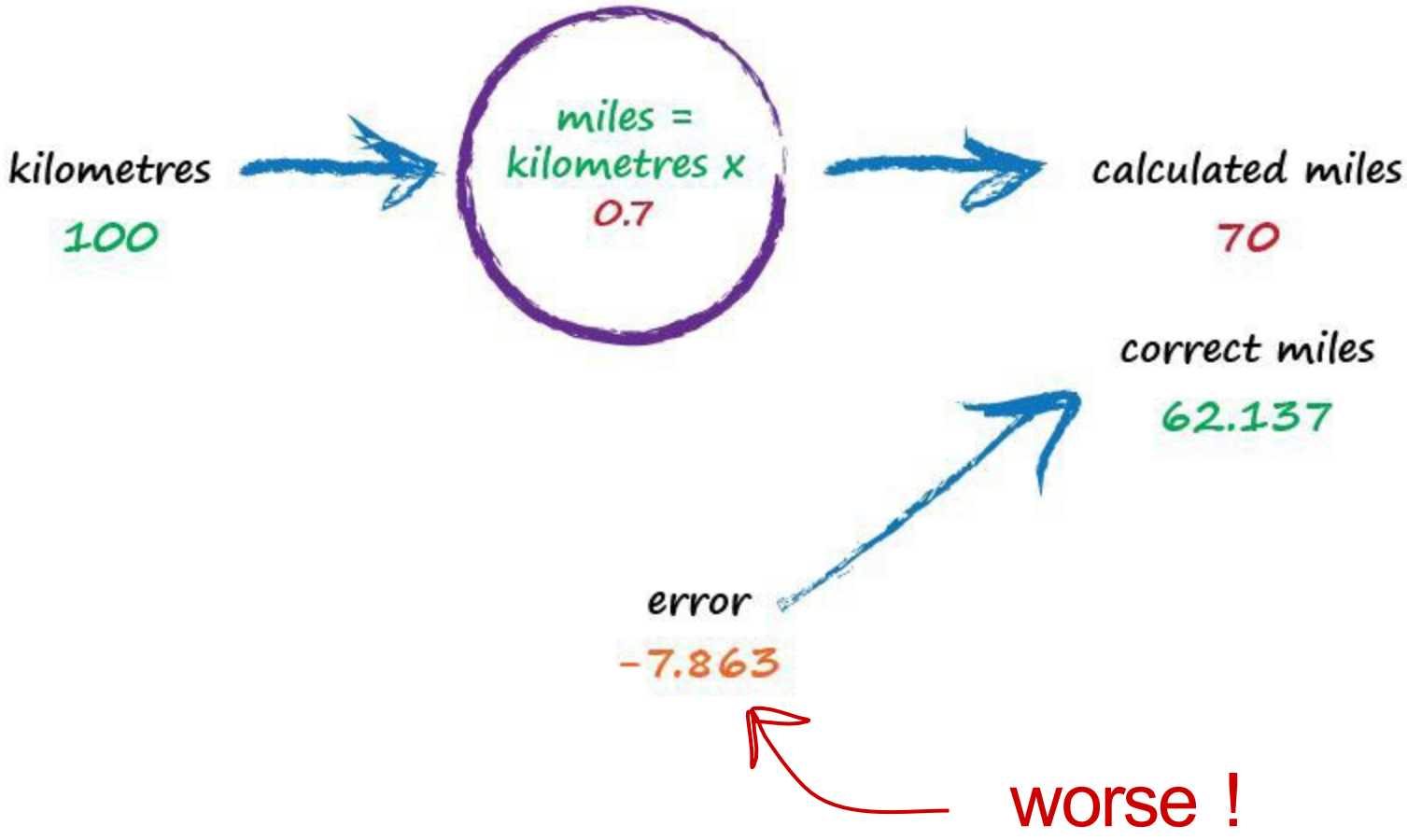

question
think
answer

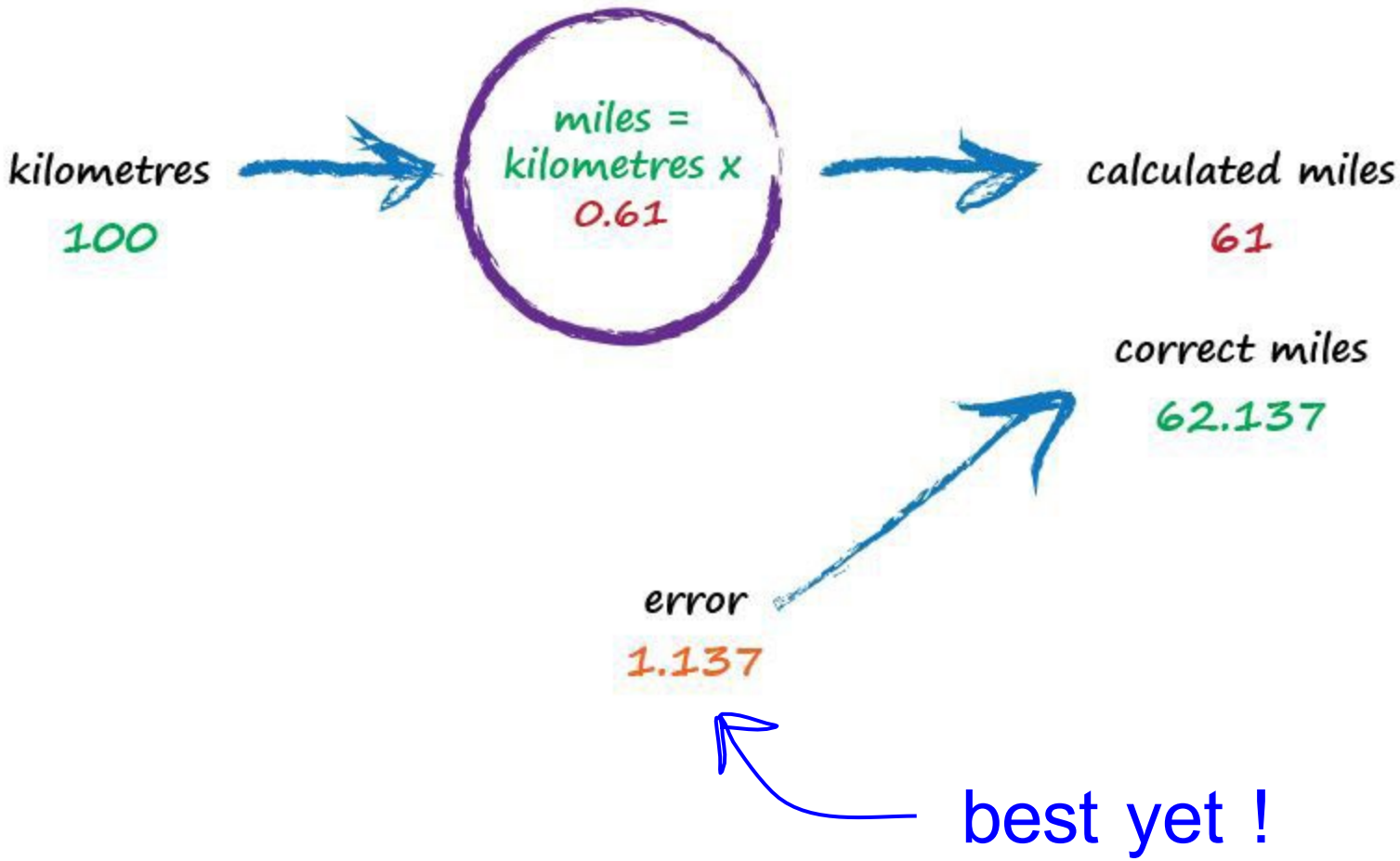

input
process
(calculate)
output


kilometres
100
miles =
kilometres x
0.5
miles
50
random starting parameter
try a model - this one is linear


kilometres
100
miles =
kilometres x
0.5
calculated miles
50
correct miles
62.137
error
12.137
not great


kilometres
100
miles =
kilometres x
0.6
calculated miles
60
correct miles
62.137
error
2.137
better


kilometres
100
miles =
kilometres x
0.7
calculated miles
70
correct miles
62.137
error
-7.863
worse !


kilometres
100
miles =
kilometres x
0.61
calculated miles
61
correct miles
62.137
error
1.137
best yet !

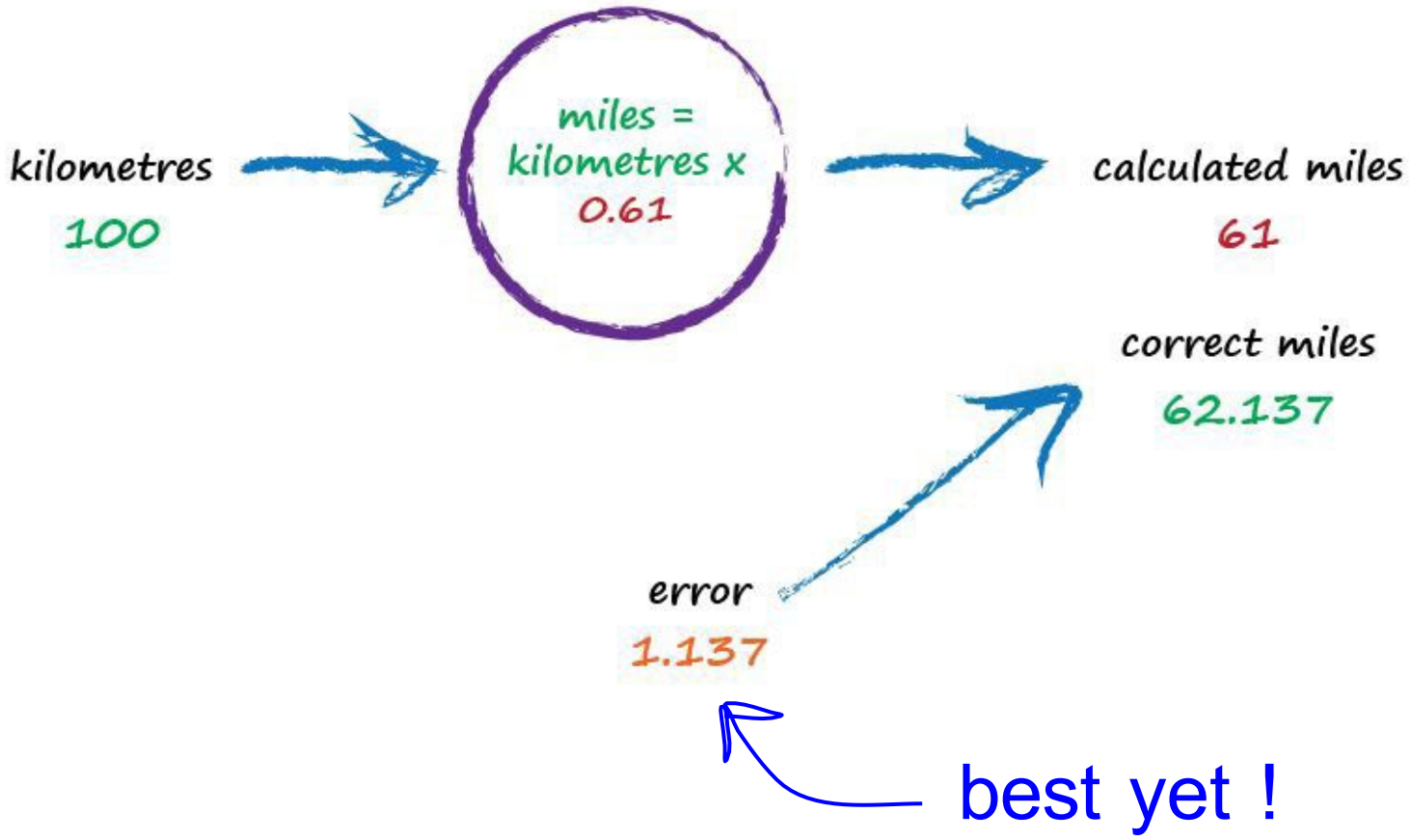
1. Don’t know how something works exactly? Try a **model** with **adjustable** parameters.

2. Use the **error** to refine the parameters.

Basic Concept of Machine Learning (ML)

is iterative in nature;

requires a teacher & memory



LEARNING in AI is defined as a computational algorithm

HOGESCHOOL ROTTERDAM

A tool for solving a well-defined (computational) problem by manipulating symbols

The word stems from the Persian mathematician al-Khwārizmī, who lived in the 9th century. His book describing the 'Indian numbers', which today we call Arabic numerals, introduced our modern decimal notation and its rules for arithmetic into Europe. The use of this system was called 'algorism' in the late middle ages and is transformed into today's 'algorithm'.

Crossley J.N., & Henry A.S. (1990)Thus spake al-Khwārizmī:
a translation of the text of Cambridge University Library Ms. Ii.vi.5 *Hist. Math.* 17(2) 103–31

Computational Thinking

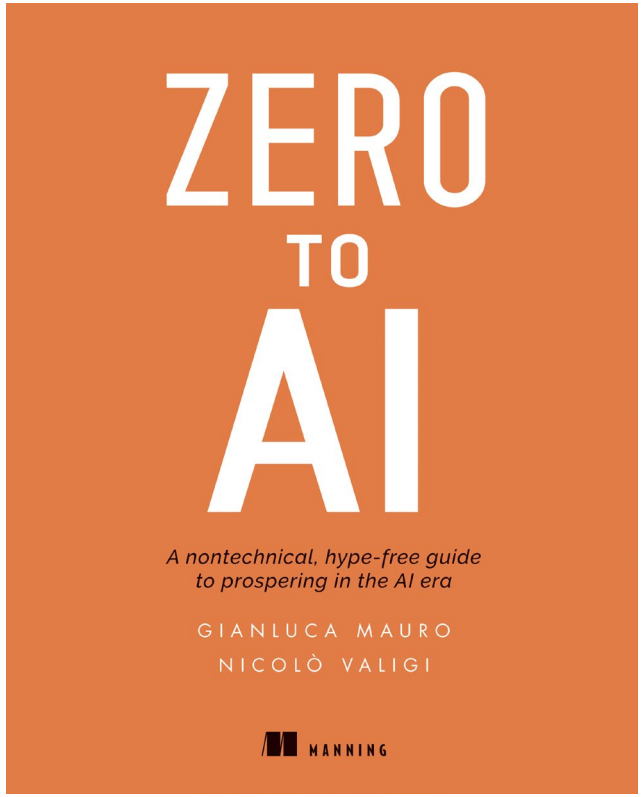
It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

HOGESCHOOL ROTTERDAM

> " Computational thinking
> is a way of solving problems,
> designing systems, and
> understanding human behavior
> that draws on concepts
> fundamental to computer science."

COMMUNICATIONS OF THE ACM March 2006/Vol. 49, No. 3

HOGESCHOOL ROTTERDAM



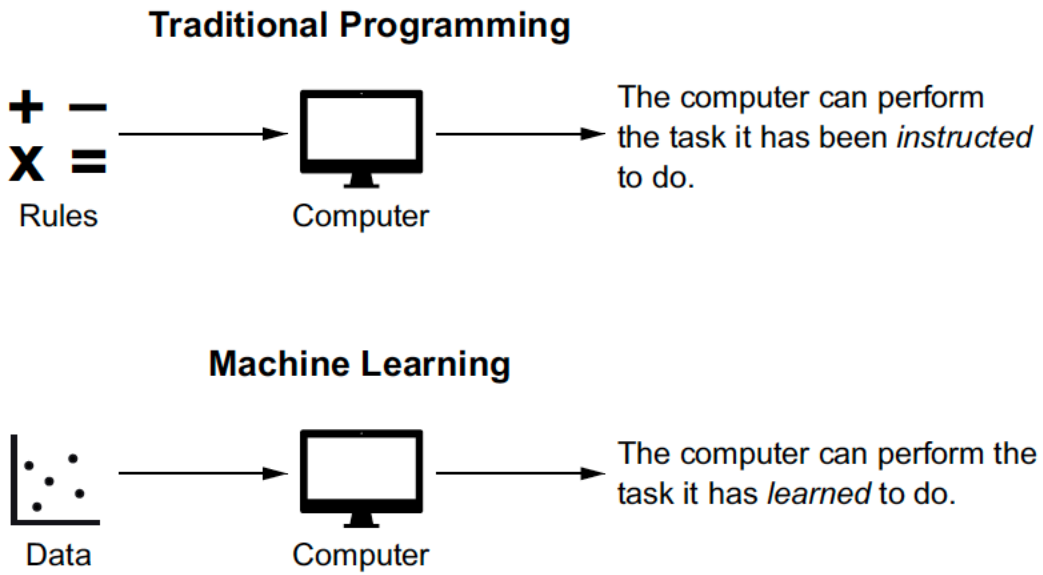
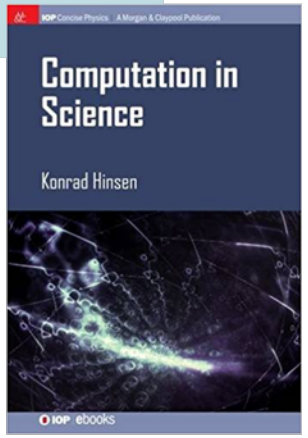


Figure 1.1 The difference between the traditional programming approach and machine learning: the first relies on precise rules and instructions, the latter on data and learning.

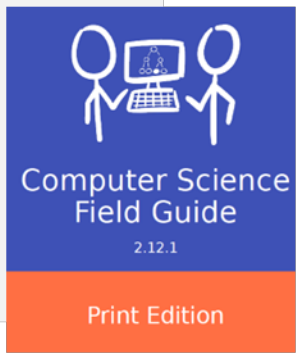
HOGESCHOOL ROTTERDAM

Wat is computation?

$$\begin{array}{rrr} 1 & 7 & 3 \\ 0 & 5 & 1 \\ & 0 & \\ \hline & & 4 \end{array} \quad \rightarrow \quad \begin{array}{rrr} 1 & 7 & 3 \\ 0 & 5 & 1 \\ 1 & & \\ \hline & 2 & 4 \end{array} \quad \rightarrow \quad \begin{array}{rrr} 1 & 7 & 3 \\ 0 & 5 & 1 \\ & & \\ \hline 2 & 2 & 4 \end{array}$$



Computations are arithmetic operations or numerical calculations that we all do in everyday life: adding up, multiplying, dividing etc.



HOGESCHOOL ROTTERDAM

Wat is computation?

When thinking about computation,
it is important to recall that the
Universe of **symbols** & **meanings** are separate.

This is known as the distinction between
syntax and **semantics**.

HOGESCHOOL ROTTERDAM

NUMBERS vs SYMBOLS

Computations do not work on numbers alone,
but on a specific representation for numbers: **Symbols**.

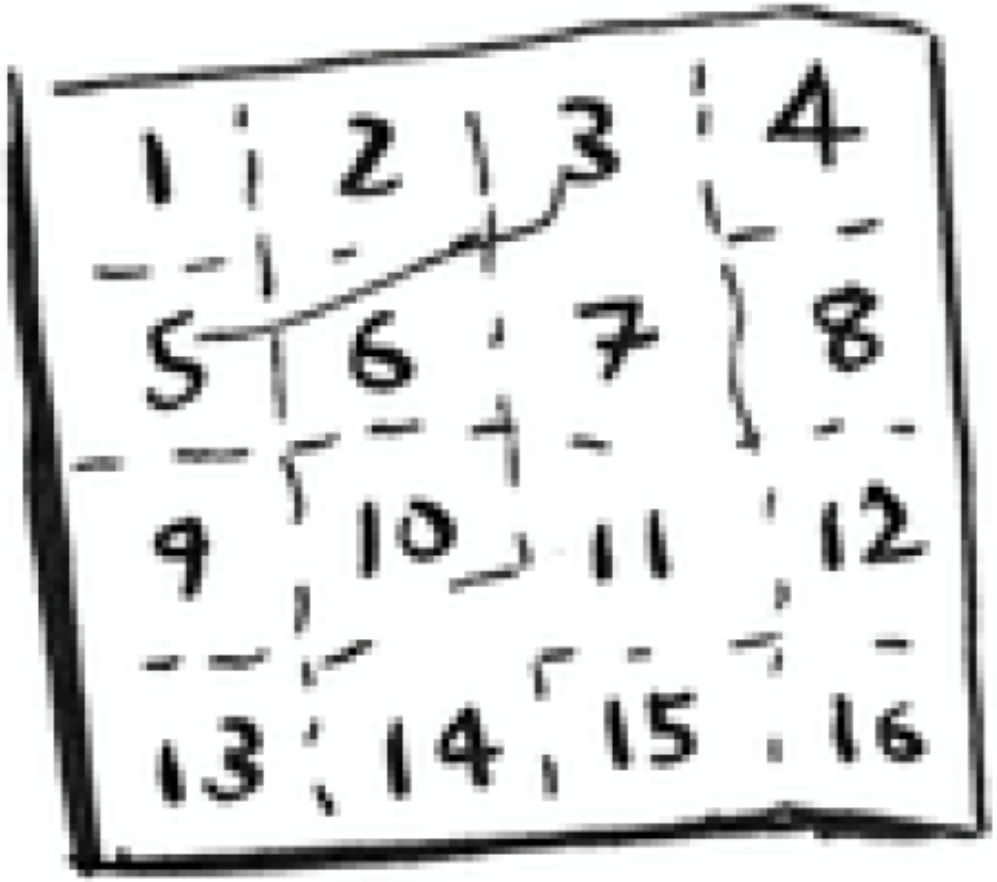
Numbers are an abstract concept,
which cannot be manipulated using mechanical rules.

HOGESCHOOL ROTTERDAM

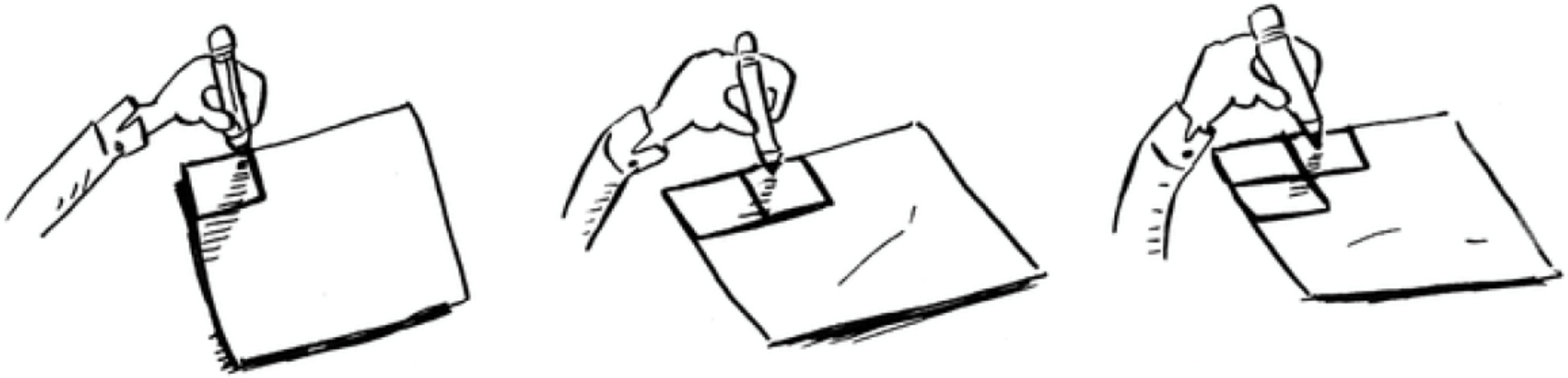
An algorithm is a

step by step process or recipe that **describes how to solve a problem** and/or **complete a task**, which will **always give** the **correct result**

What's a good algorithm to draw this grid?

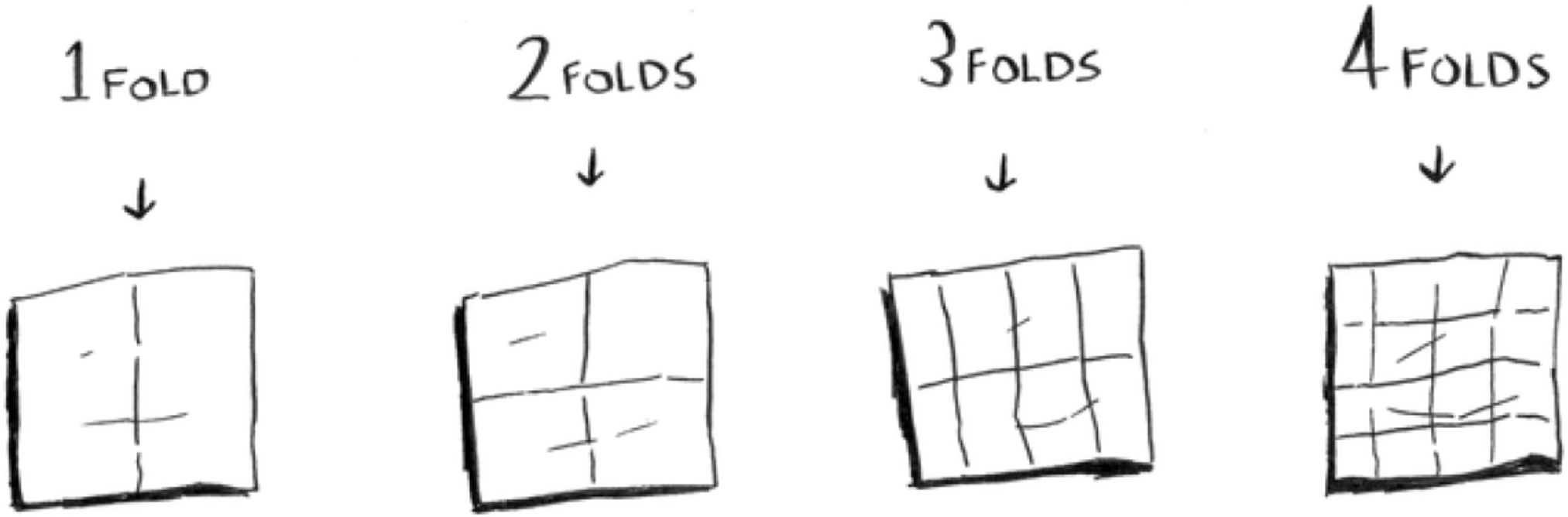


Algorithm 1:
Drawing a grid one box at a time



One way to do it is to draw 16 boxes, one at a time.
How many operations will it take, drawing one box at a time?

Algorithm 2:
Fold the paper again, and again, and again.



You can "draw" twice as many boxes with every fold,
so you can draw 16 boxes in 4 steps.

Algorithm 3:
The Fox, the Goose, and the Corn

HOGESCHOOL ROTTERDAM

PROBLEM: HOW TO CROSS THE RIVER?

A farmer with a fox, a goose, and a sack of corn needs to cross a river. The farmer has a rowboat, but there is room for only the farmer and one of his three items. Unfortunately, both the fox and the goose are hungry. The fox cannot be left alone with the goose, or the fox will eat the goose. Likewise, the goose cannot be left alone with the sack of corn, or the goose will eat the corn. How does the farmer get everything across the river?

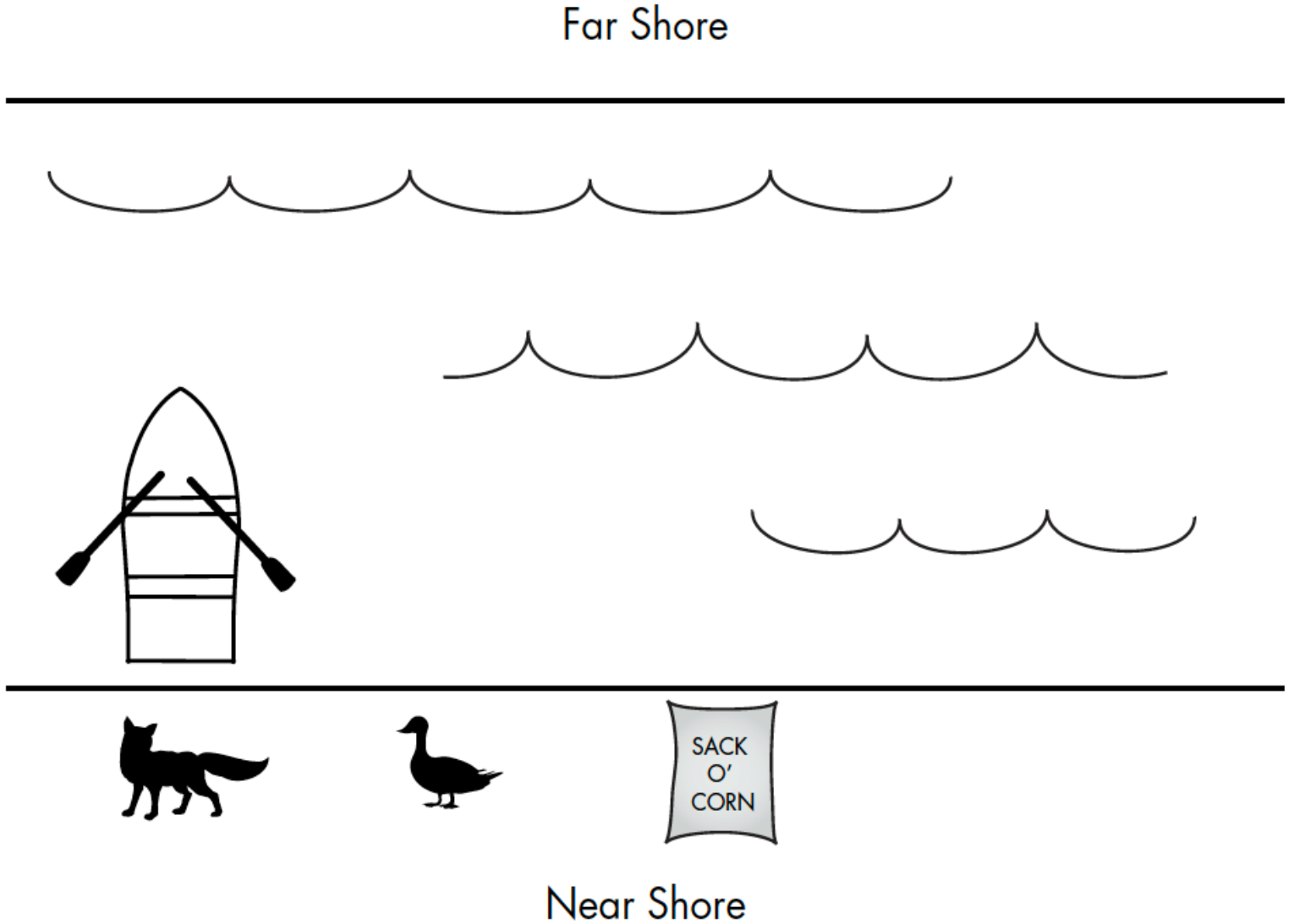
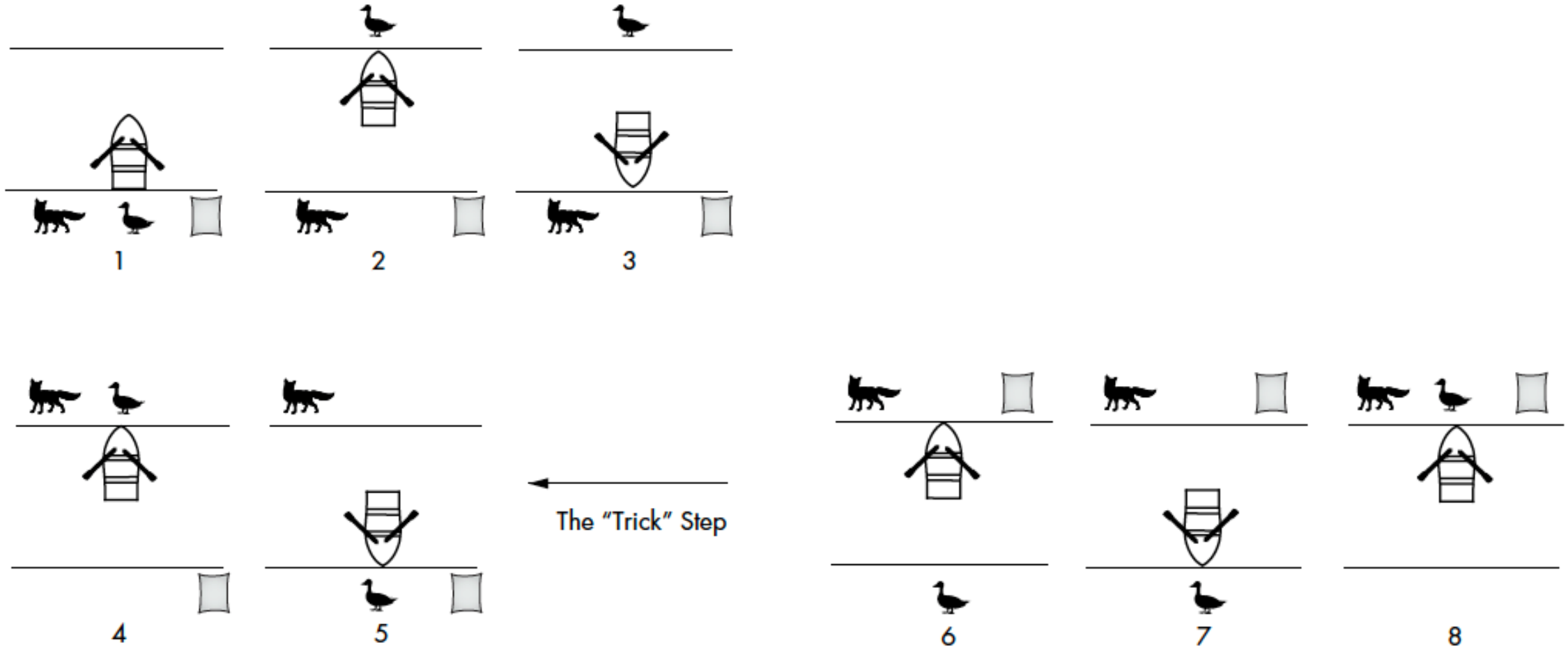


Figure 1-1: The fox, the goose, and the sack of corn. The boat can carry one item at a time. The fox cannot be left on the same shore as the goose, and the goose cannot be left on the same shore as the sack of corn.

Algorithm 3:
Step-by-step solution to the fox, goose, and corn problem

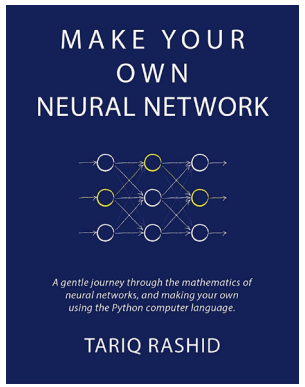


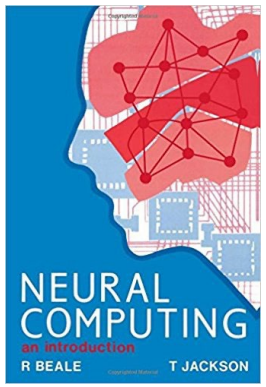
HOGESCHOOL ROTTERDAM

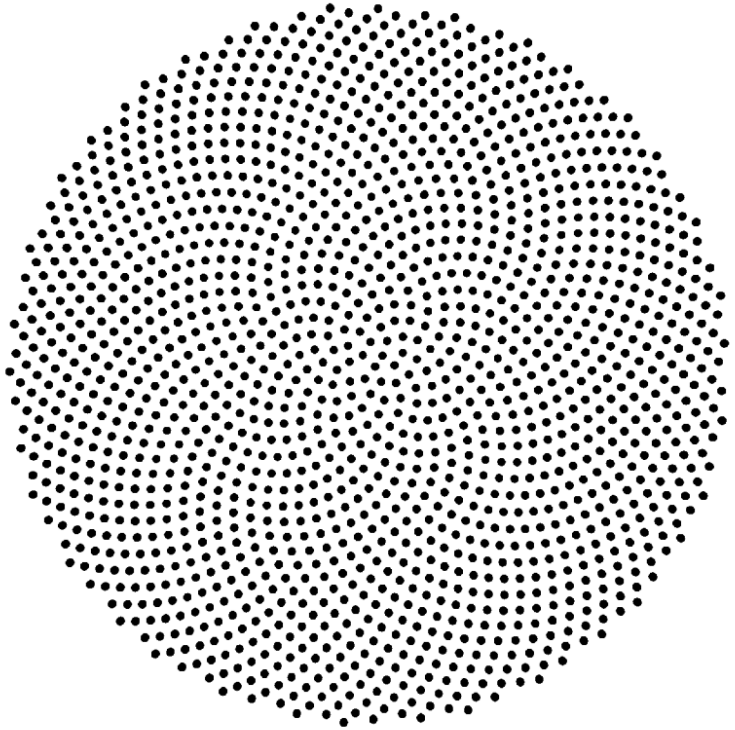
{03}

Fundamentals: **PERCEPTRON Algorithm**

HOGESCHOOL ROTTERDAM







HOGESCHOOL ROTTERDAM

Algorithm: Perceptron Learning Algorithm

$P \leftarrow$ *inputs with label* 1;

$N \leftarrow$ *inputs with label* 0;

Initialize **w** randomly;

while !*convergence* **do**

  Pick random $\mathbf{x} \in P \cup N$;

  if $\mathbf{x} \in P$ *and* $\sum_{i=0}^{n} w_i * x_i < 0$ **then**

    $\mathbf{w} = \mathbf{w} + \mathbf{x}$;

  end

  if $\mathbf{x} \in N$ *and* $\sum_{i=0}^{n} w_i * x_i \geq 0$ **then**

    $\mathbf{w} = \mathbf{w} - \mathbf{x}$;

  end

end

//the algorithm converges when all the inputs are classified correctly

Key-words / Concepts / Labels

Artificial neuron === Model of single biological neuron / Perceptron

Feedforward / Feed-backward / Hebbian learning

Bias / Threshold

Weights /adjustable parameters / Memory

Unit / Node / Activation Functions (Sigmoid / Step)

Input / Output / Layers / Input vs Output Space

Target / Teacher / Error / learningrate

Iterations / Epochs / Baches

Logical functions AND / OR / XOR (Boolean Algebra)

Maths + Python Key-words / Concepts

HOGESCHOOL ROTTERDAM

PYTHON MODULES

```
import Numpy as np
import Matlibplot.pyplot as plt
```

```
class
def return
for (loops)
if else
zip
print
list
append
+=
```

Matrix Calculus

```
np.zeros
np.random.normal
np.dot
```

Plotting

```
plt.subplots()
plt.subplot(221)
plt.plot(epoch, error)
plt.xlabel('Epoch')
plt.ylabel('Error')
```

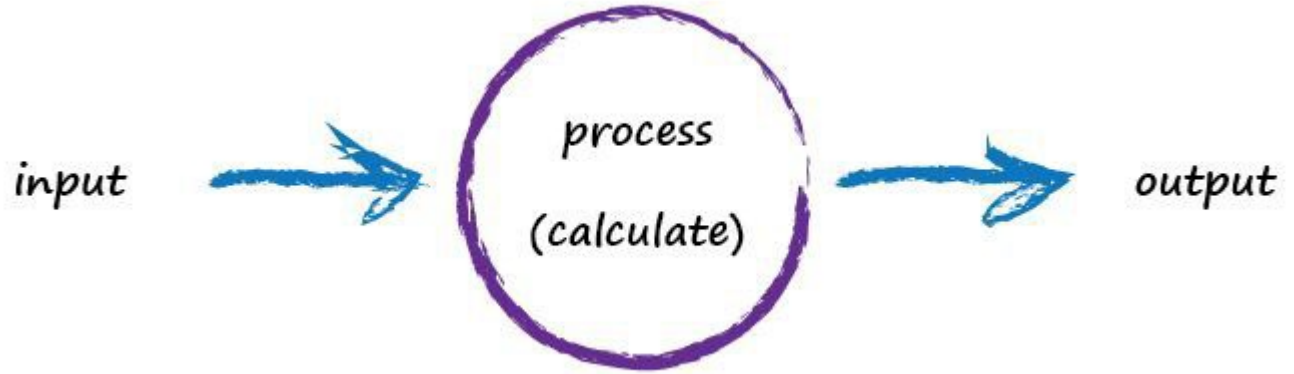
The PERCEPTRON

The perceptron was very promising, but it was soon discovered that is has serious limitations as it only works for linearly-separable classes.

In 1969, Marvin Minsky and Seymour Papert demonstrated that it could not learn even a simple logical function such as XOR.

This led to a significant decline in the interest in perceptron's and Machine learning as a whole.

The Concept of Input / Output model



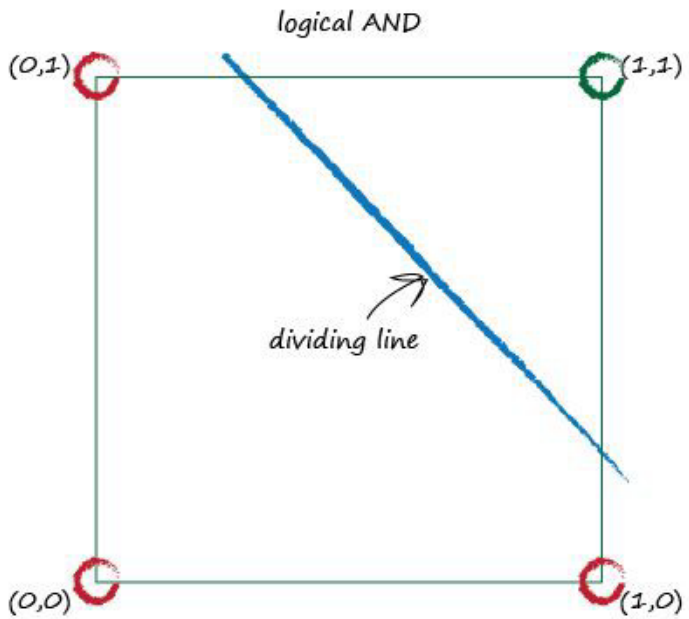
Boolean Logic

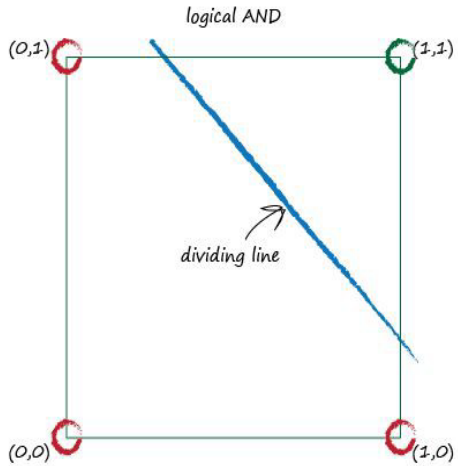
IF I have eaten my vegetables **AND** I am still hungry **THEN** I can have ice cream.

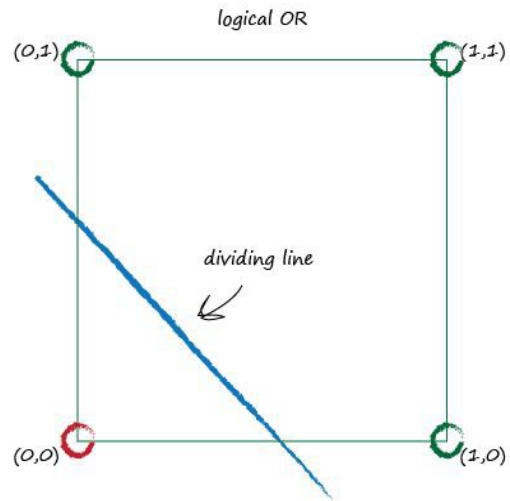
IF it's the weekend **OR** I am on annual leave **THEN** I'll go to the park.

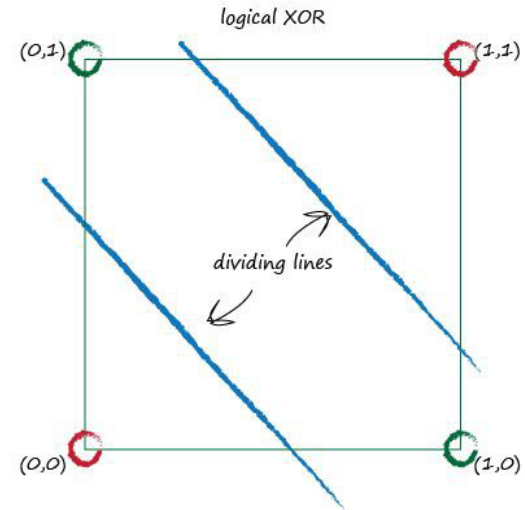
Input A	Input B	AND	OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Input x1	Input x2	AND y
0	0	0
0	1	0
1	0	0
1	1	1

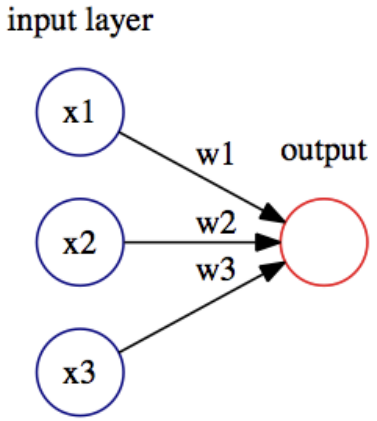


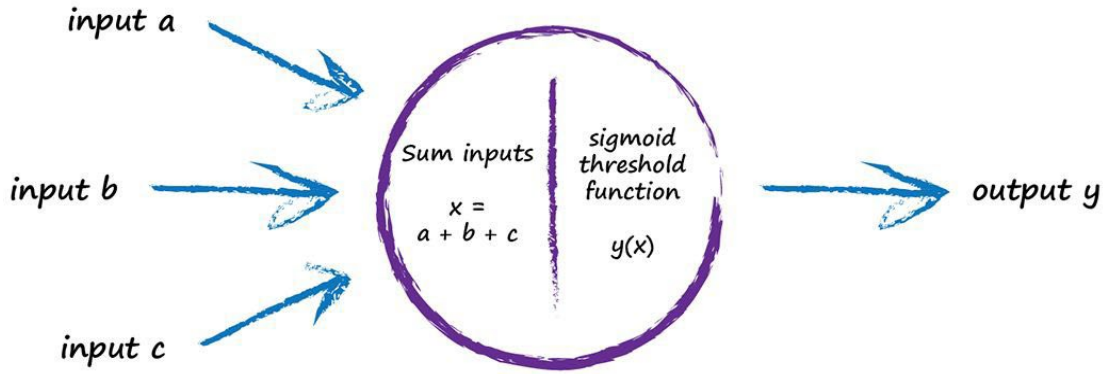

logical AND
(0,1)
(1,1)
dividing line
(0,0)
(1,0)

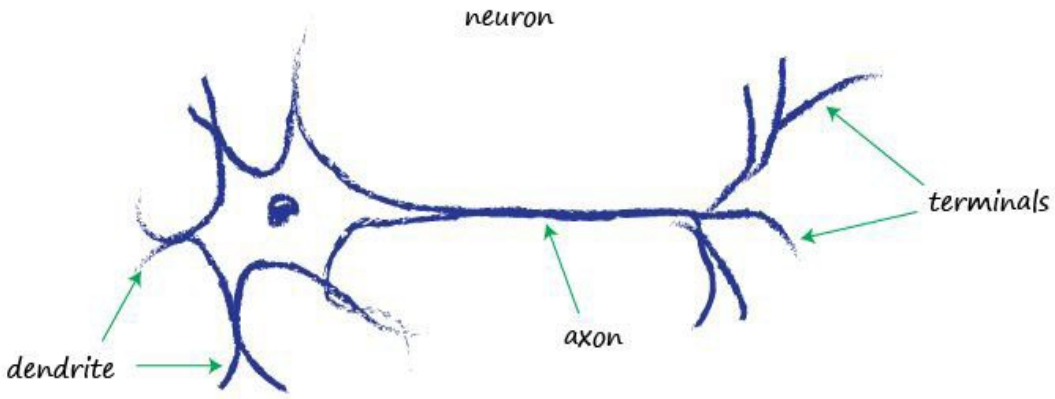

logical OR
(0,1)
(1,1)
dividing line
(0,0)
(1,0)


logical XOR
(0,1)
(1,1)
dividing lines
(0,0)
(1,0)

Artificial Neuron Model: input vs output



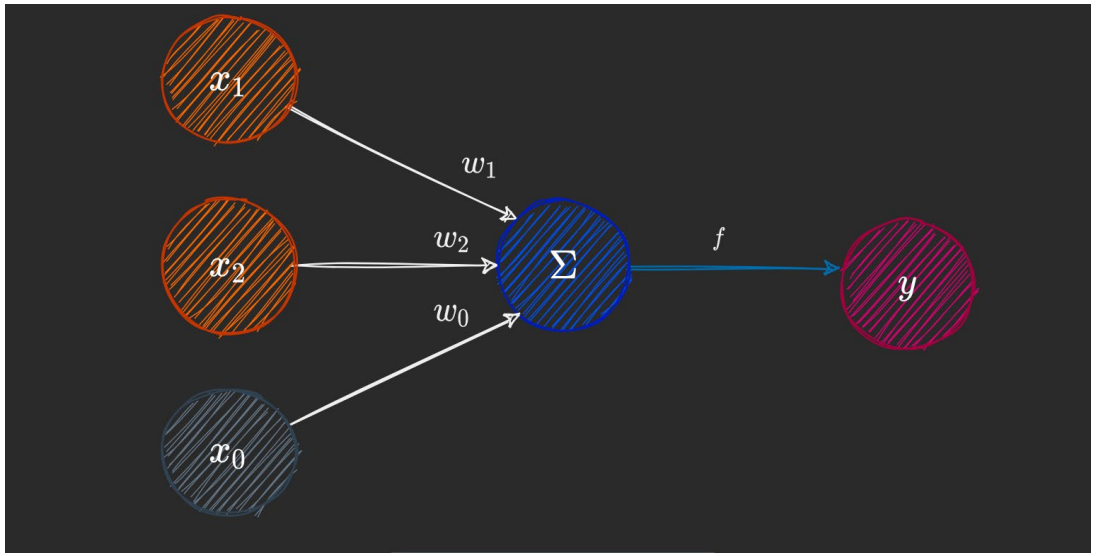




Artificial Neuron Model: PERCEPTRON structure

Perceptron Components :

- Input nodes x0 … Xn
- Output node y
- An activation function
- Weights and biases
- Summation
- Activation Function f



Artificial Neuron Model: PERCEPTRON maths

FEED FORWARD CACULATION:

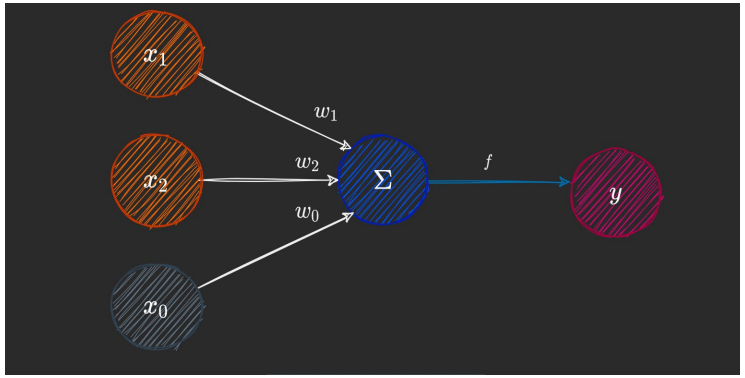
The output calculation is straightforward.

Compute the DOT PRODUCT of the input [x1 x2] and weight vector [w1 w2]

The bias b equals w0

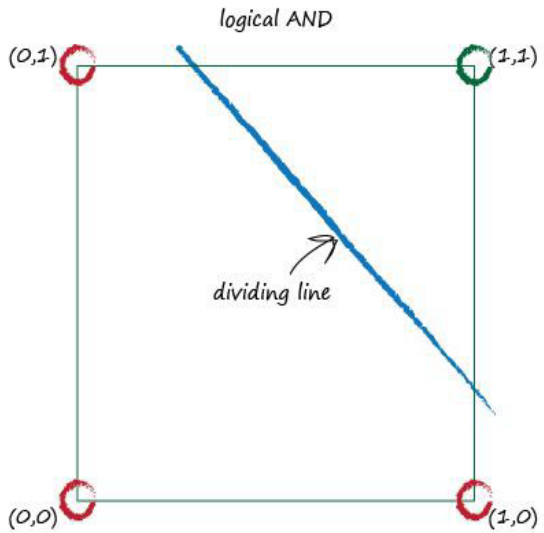
Apply the activation function.

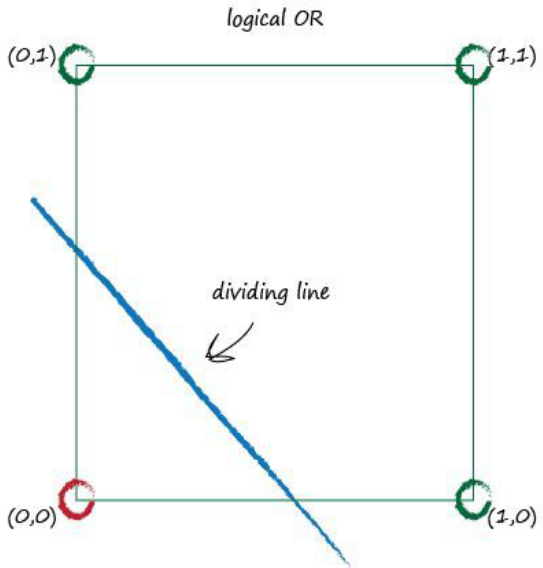
$$y = f(w \cdot X + b)$$



Feedforward computation:
(DOT PRODUCT is a form of mathematical summation)

$$y = f(x_1 \cdot w_1 + x_2 \cdot w_2 + b)$$

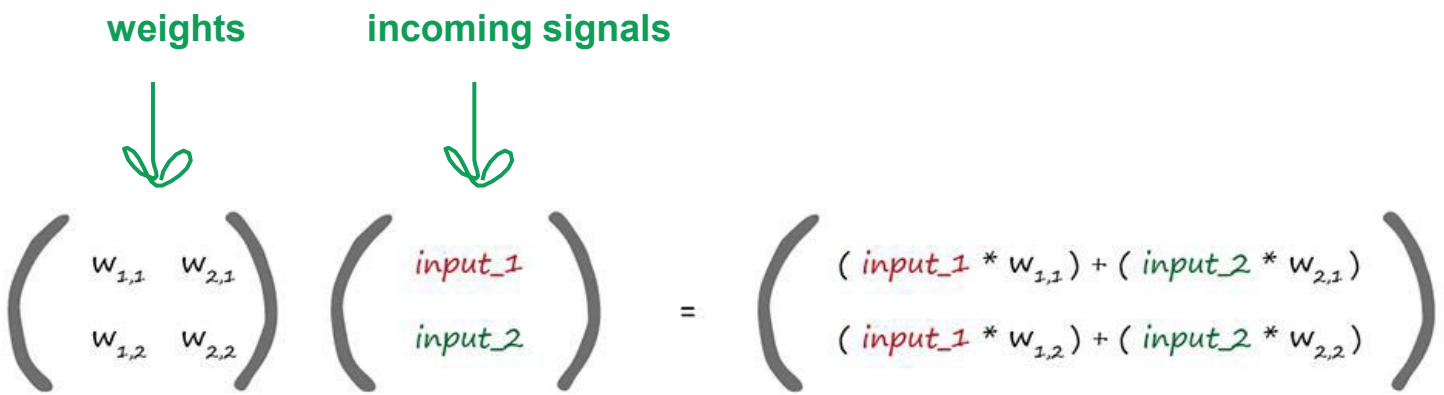




$$Class_1 : w \cdot X + b \geq 0$$

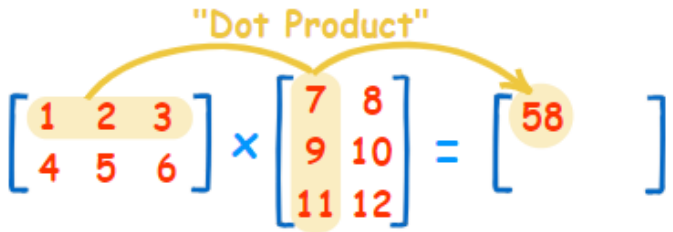
$$Class_2 : w \cdot X + b < 0$$

Artificial Neuron Model:
PERCEPTRON maths



$$W \cdot I = X$$

dot product

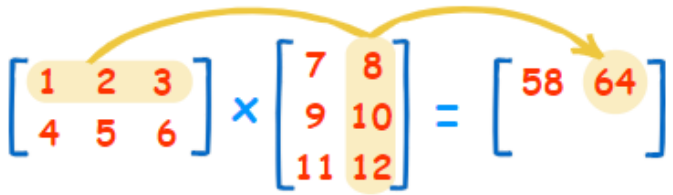


The "Dot Product" is where we **multiply matching members**, then sum up:

$$(1, 2, 3) \bullet (7, 9, 11) = 1\times7 + 2\times9 + 3\times11$$
$$= 58$$

We match the 1st members (1 and 7), multiply them, likewise for the 2nd members (2 and 9) and the 3rd members (3 and 11), and finally sum them up.

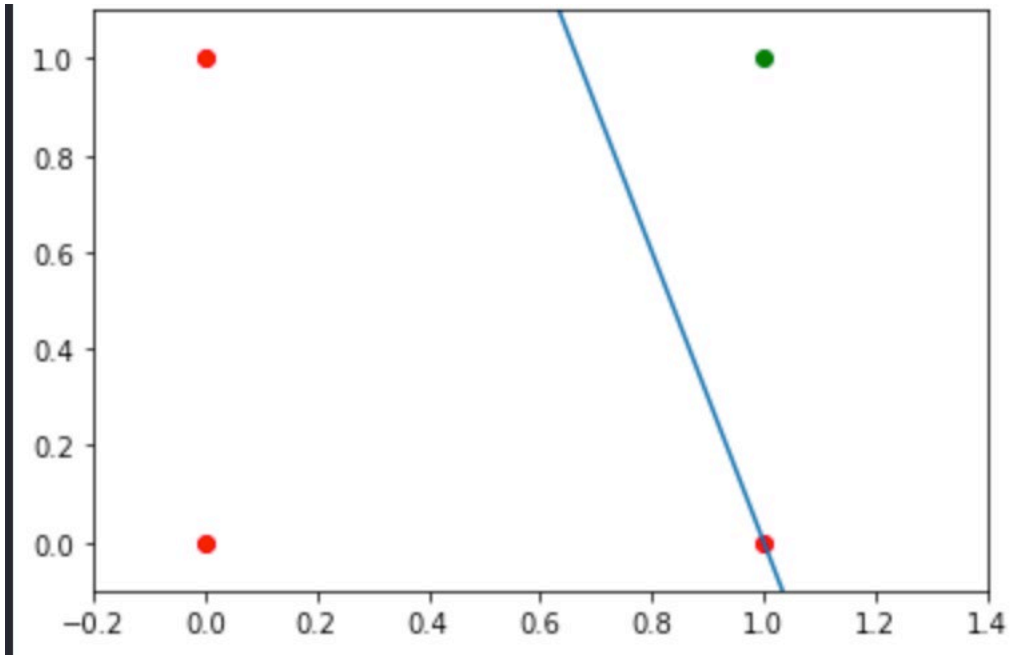
Want to see another example? Here it is for the 1st row and **2nd column**:



$$(1, 2, 3) \bullet (8, 10, 12) = 1\times8 + 2\times10 + 3\times12$$
$$= 64$$

{01}
Fundamentals

```
p = Perceptron(weights=[0.3, 0.3, 0.3],
               learning_rate=0.2)

for in_data, label in labelled_samples(30):
    p.adjust(label,
             in_data)

test_data, test_labels = list(zip(*labelled_samples(30)))

evaluation = p.evaluate(test_data, test_labels)
print(evaluation)


fig, ax = plt.subplots()
xmin, xmax = -0.2, 1.4
X = np.arange(xmin, xmax, 0.1)
ax.scatter(0, 0, color="r")
ax.scatter(0, 1, color="r")
ax.scatter(1, 0, color="r")
ax.scatter(1, 1, color="g")
ax.set_xlim([xmin, xmax])
ax.set_ylim([-0.1, 1.1])
m = -p.weights[0] / p.weights[1]
c = -p.weights[2] / p.weights[1]
print(m, c)
ax.plot(X, m * X + c )
plt.plot()
```



PYTHON PERCEPTRON MODEL code

```
import Numpy as np
import Matlibplot.pyplot as plt
```

Learn how to specify: input/output relationship

Initialize weights

Forward computation

```python
import numpy as np
import matplotlib.pyplot as plt

class Perceptron(object):

    def __init__(self, no_of_inputs, no_of_training_epochs=20, learning_rate=0.01):
        self.no_of_training_epochs = no_of_training_epochs
        self.learning_rate = learning_rate
        self.weights = np.random.normal(0, 10, no_of_inputs + 1)
        self.bias = self.weights[0]
        self.epoch_list=[]
        self.error_history=[]


    def step_function(self, inputs):
        netto_summation = np.dot(inputs, self.weights[1:]) + self.bias
        if netto_summation > 0:
          activation = 1
        else:
          activation = 0
        return activation

    def train(self, training_inputs, teacher_labels):
        epoch =0;
        i =0;
        for _ in range(self.no_of_training_epochs):
          epoch+=1
          for inputs, teacher in zip(training_inputs, teacher_labels):
              activated_perceptron_output = self.step_function(inputs)
              error = (teacher - activated_perceptron_output)
              self.weights[1:] += self.learning_rate * (error) * inputs
              self.bias += self.learning_rate  * error
              i+=1
              self.epoch_list.append(epoch)
              self.error_history.append(np.sum(np.abs(error)))
```

PYTHON PERCEPTRON MODEL code

import Numpy as np
import Matlibplot.pyplot as plt

Where are input & ouput specified?

Where are the weights initialized?

How is the Forward computation specified?

To Do:
Change the code as to print
Interation number
Bach number
Error
And weights

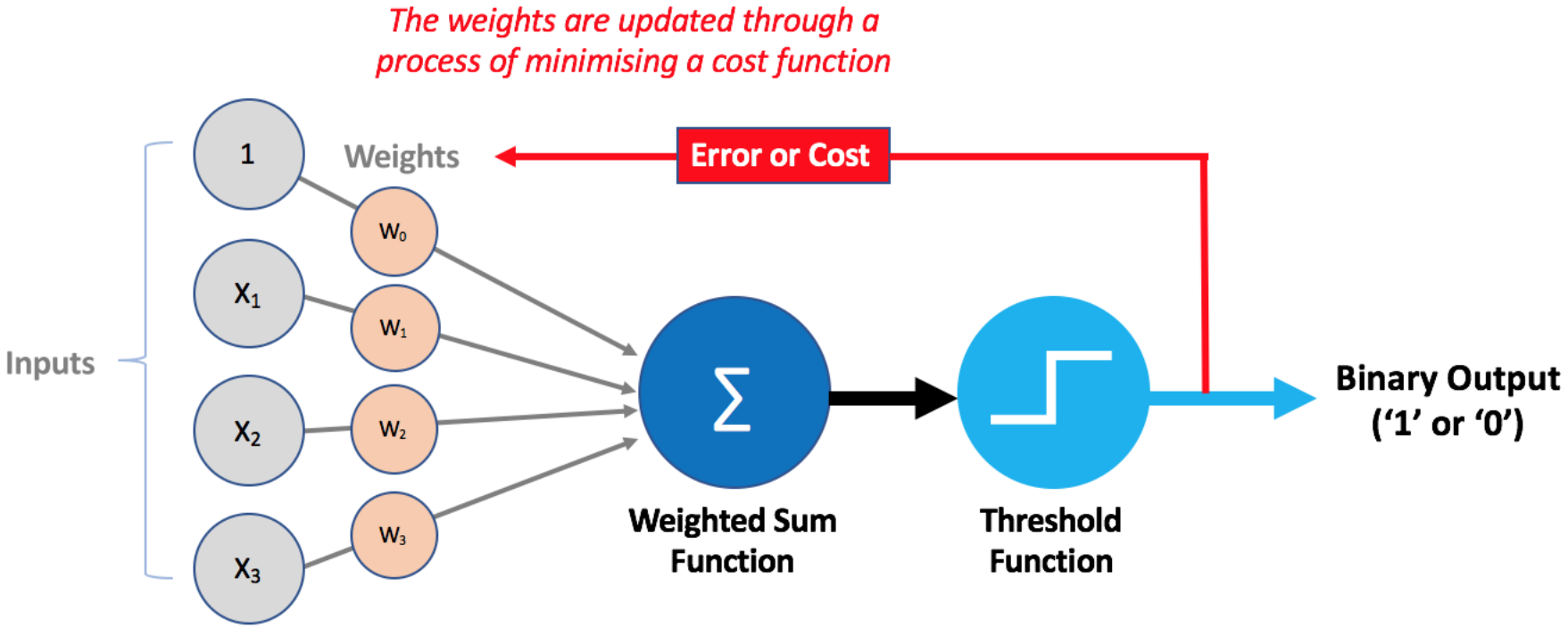
PYTHON
INPUT / OUTPUT relationship code

```
import Numpy as np
import Matlibplot.pyplot as plt

import numpy as np

inputs = []
inputs.append(np.array([1, 1]))
inputs.append(np.array([1, 0]))
inputs.append(np.array([0, 1]))
inputs.append(np.array([0, 0]))

teacher = np.array([1, 0, 0, 0])
```

USING ZIP

```
print([inputs for teacher in zip(inputs, teacher)])

print([i for i in zip(inputs, teacher)])
```

NEXT WEEK:


The weights are updated through a process of minimising a cost function
Error or Cost
Weights
1
X_1
X_2
X_3
Inputs
W_0
W_1
W_2
W_3
Σ
Weighted Sum Function
Threshold Function
Binary Output ('1' or '0')

Artificial Neuron Model:
PERCEPTRON maths

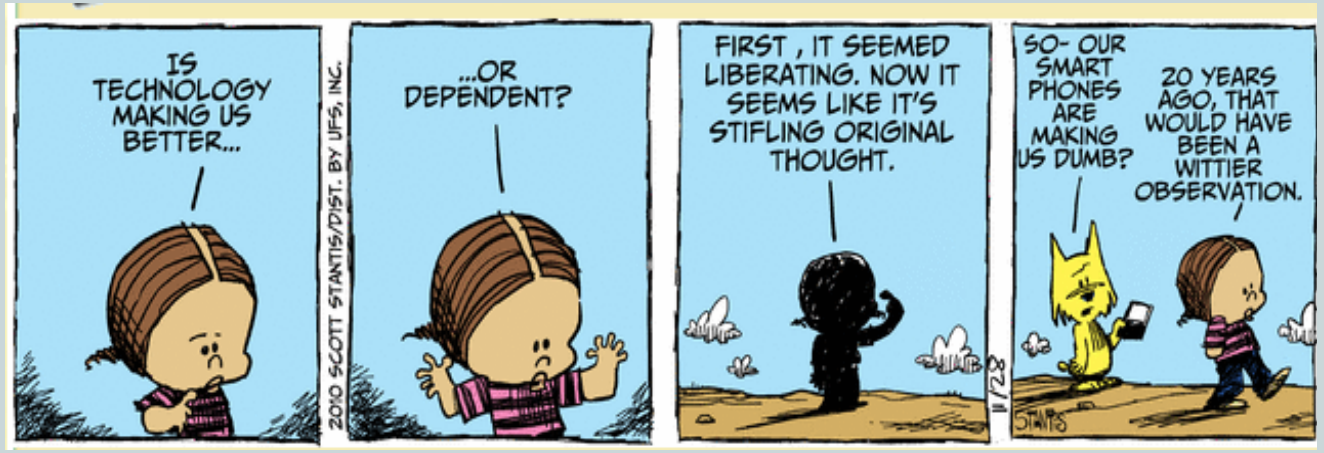
$$\Delta w = node \cdot (actual - computed)$$

Error calculation

Artificial Neuron Model:
PERCEPTRON maths

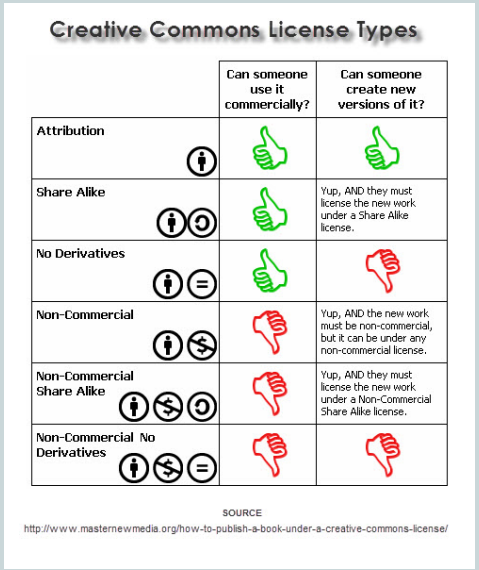
$$w_{updated} = w_{old} + lr \cdot \Delta w$$

FEED BACKWARD calculation



This lesson was developed by:

Robert Frans van der Willigen
CMD, Hogeschool Rotterdam
OKT 2020



http://empoweringthenatives.edublogs.org/2012/03/15/creative-commons-licenses/

http://creativecommons.org/licenses/


the functional art
alberto cairo
TECHNOLOGY TRENDS TO WATCH
Christian Leborg
Visual Grammar
the analytics revolution
Bill Franks
MADE to STICK
generative art
matt pearson
EMPATHIC THINGS
INTERNET OF THINGS
MAKERS
THE NEW INDUSTRIAL REVOLUTION
CHRIS ANDERSON
MARITIEME MAAKINDUSTRIE TERUG NAAR RDM
TV AND VIDEO
An analysis of evolving consumer habits
THE DISRUPTION MACHINE
Learning to Program with MATLAB
BUILDING GUI TOOLS
Port Management Program
R Data Analysis
10 TRENDS FOR 2014
A giant cage
Design for Information
HACKATHO
OPEN HAVE
1 VOLLE DAG
7 SEPTEMBER
WERELDHAVENDAGEN 2013
the myths of innovation
scott berkun
O'REILLY
resonate
Cool Infograph
Effective Communication with Data Visualization and Design
The VISU
PHIL SIMON
INVENT TO LEARN
Making, Tinkering, and Engineering in the Classroom
MORE SECRETS with Big Data Analytics
Processing
A Programming Handbook for Visual Designers and Artists
OUR WORK HERE IS DONE
VISIONS OF A ROBOT ECONOMY
THE WALL STREET JOURNAL.
Guide to Information Graphics
Dona M. Wong
Customer Journey Mapping
Wearable Technology Design
The Hidden Pitfall in your Design
Making Things See
Learn by Discovery
TEACHING that STICKS
Chip Heath & Dan Heath
Protecting Data and Privacy in the Cloud
DISCOVER METEOR
SOGETI
Making Things Talk
The Internet of Things
Business Adapts to a New Style of Computer
EVOLUTION OF AGILE