

A background image of several glowing blue jellyfish swimming in a dark, deep-sea-like environment. The jellyfish are translucent with bright blue internal structures and outer edges. They are scattered across the frame, with some in the foreground and others in the background, creating a sense of depth. The overall lighting is very dark, with the primary light source being the bioluminescence of the jellyfish themselves.

Deep Learning:

Theories and Practices

Sirakorn Lamyai

Access to the slide materials

<http://bit.ly/dl-tnp>

About me

Sirakorn Lamyai

- [Outsourced] TA, PTT GC, 2019-2020
- Former RA Intern, VISTEC, 2019
- Former RA Intern, VISTEC, 2018

Scope of interests: machine learning, deep learning, adversarial attacks

Senior project topic: Cluster-based Method for Adversarial Retraining on Deep Learning models

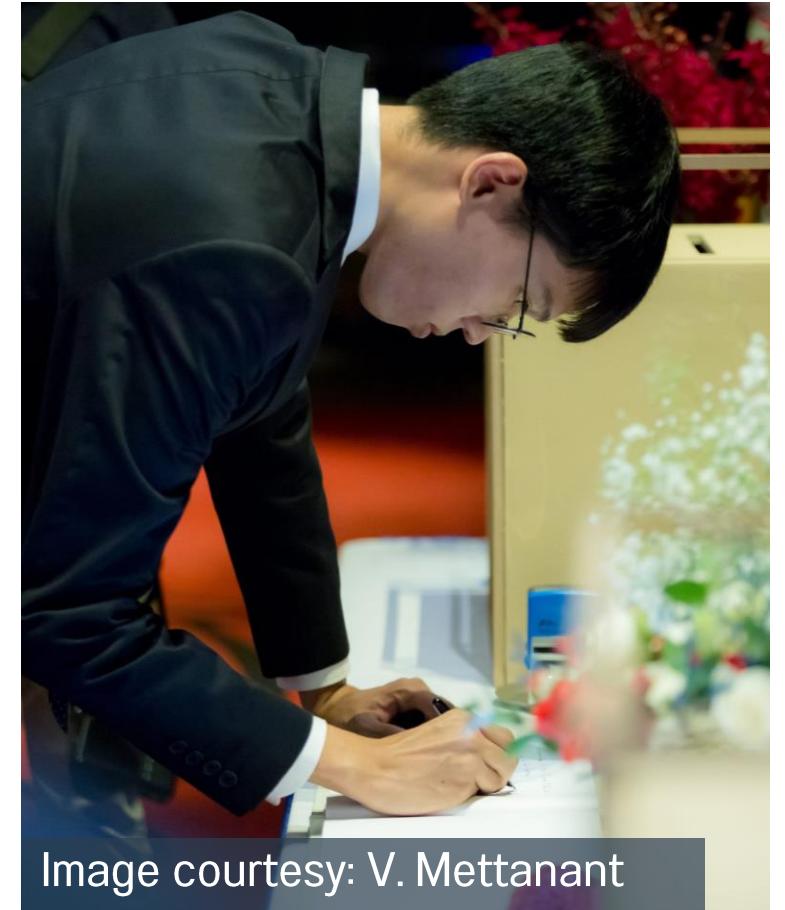


Image courtesy: V. Mettanant

My latest motivation to this course

===สาระ เรื่องหลักสูตร=== >> เราแต่ซื้อหมวดให้ใหม่ จะได้เข้าใจง่าย ถ้าอยากรู้ละเอียดๆไปหาเอานะ 5555 ใน [] คือ คห เราว่าวิชานี้มัน useless(เสียเวลาเรียน ไม่เกิดประโยชน์ใน อนาคต) useful(เรียนแล้วดีนะ) so so(เฉยๆ)

1.)วิชาเกี่ยวกับ การใช้ชีวิต, สร้าง Product, ศิลปะการใช้ชีวิต 30 หน่วยกิต [ค่อนข้าง useless]

2.)ฟิสิก เคมี ชีวะ แมท ดรออิง แลปฟิ/เคมี อิเล็กทรอนิกส์ ไฟฟ้า 40หน่วยกิต [ฟิ เคมี แมท ยากเกินไปจน uselessมากกกก , ที่เหลือ so so]

3.)คอมจริงๆ (แต่เลือกเรียนได้ไม่อิสระเท่าไร) 71 หน่วยกิต [ค่อนข้าง usefull]

4.)วิชาต่างๆ ไว้เก็บเกรด 6 หน่วยกิต [useless มาก คือ มก แค่อยากให้มีละบั้งคับทุกภาคเรียน เพื่อ?]

เรียนได้สูงสุด 22 หน่วยกิต/เทอม ปกติ ม จะแนะนำเรา 20-21 หน่วยกิต

**ชั่วโมงกิจกรรม ต้องครบ ไม่ครบไม่ได้เข้าพิธีรับปริญญาบัตร [บั้งคับเข้ากิจกรรม มันแปลกๆอะ เราให้ useless]

- Link to original forum topic removed in order to mitigate the risk from witch-hunting.
- Original comments proposes a constructive criticism to the university's environment.
- Please make no attempt to trace the real identity of the topic poster.
- Please refrain from non-constructive comments to the thread, especially on mood and tone. This make no changes to what our friends are perceiving from the Department.

First Semester 2016

01200101	Innovative Thinking	A	1
01204111	Computers & Programming	B+	3
01355111	Foundation English I	P	3
01355112	Foundation English II	P	3
01355113	Foundation English III	P	3
01417167	Engineering Mathematics I	C	3
01420111	General Physics I	B+	3
01420113	Laboratory in Physics I		
01999021	Thai Language for Commun		

sem. G.P.A. = 3.18

Second Semester 2016

01208111	Engineering Drawing	C+	3
01240011	Design in Everyday Life	B	3
01403114	Lab.in Fundamentals of General Chemistry	B+	1
01403117	Fundamentals of General Chemistry	C+	3
01417168	Engineering Mathematics II	D+	3
01420112	General Physics II	B	3
01420114	Laboratory in Physics II		
01999033	Arts of Living		

sem. G.P.A. = 2.78

First Semester 2017

01204211	Discrete Mathematics	A	3
01204212	Abstract Data Types & Problem Solving	A	3
01204215	Object-Oriented Programming Laboratory	A	1
01204222	Digital Systems Design	B	3
01204223	Practicum in Computer Engineering	A	1
01204312	Probability Theory & Stat. for Comp. Eng.	B	3
01205204	General Electronics I	C	3
01219211	Software Development Training Camp	A	1
01417267	Engineering Mathematics III	D+	3

sem. G.P.A. = 3.07

cum. G.P.A. = 2.99

Deep Learning

Deep?



Why Deep Learning?



Video

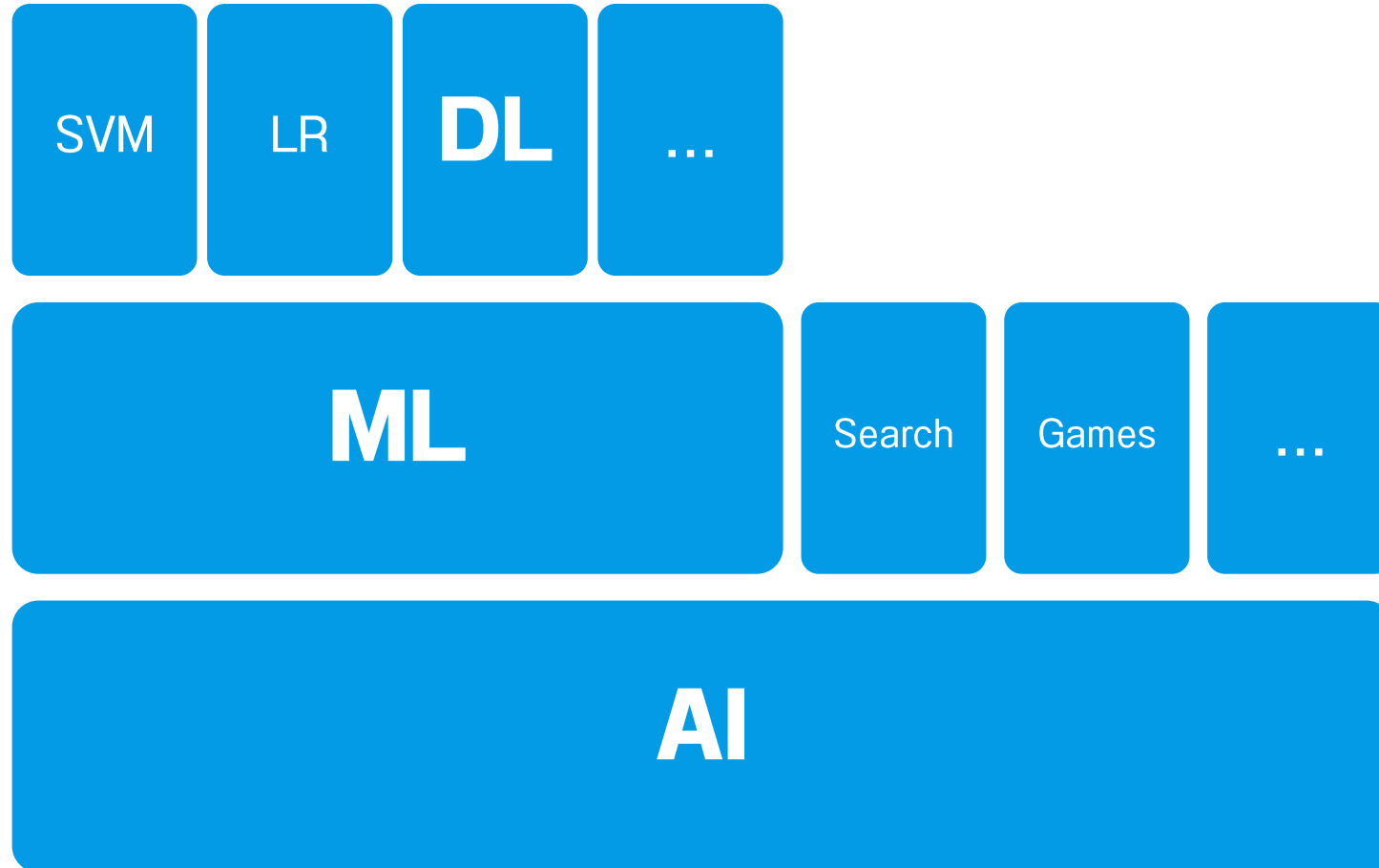
Fake videos of people – and how to spot them

Delivered by Supasorn S. at TED 2018



Basics of all the Basics

Artificial Intelligence

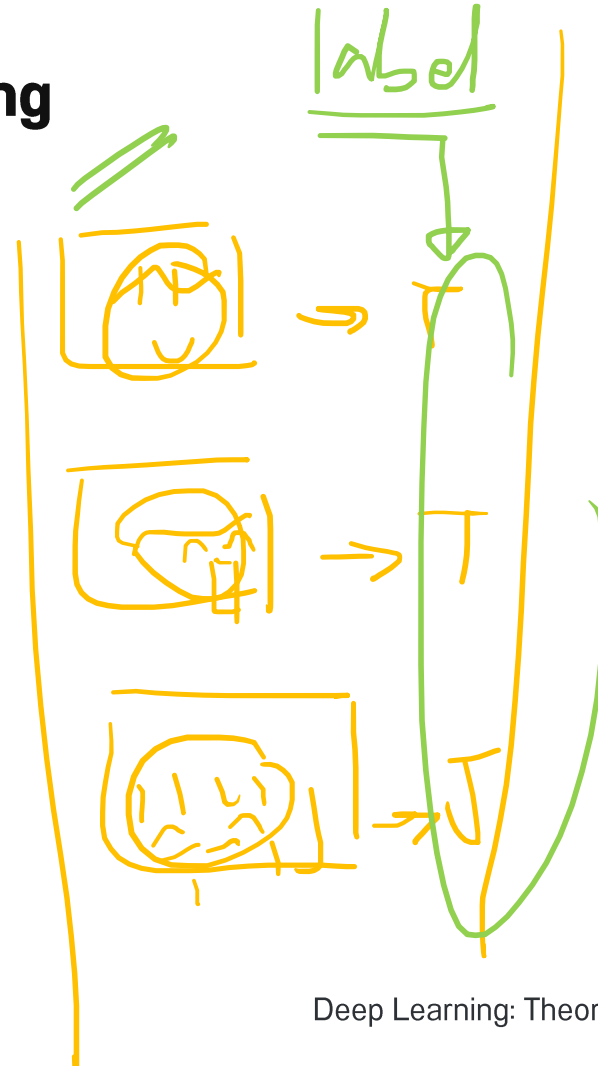
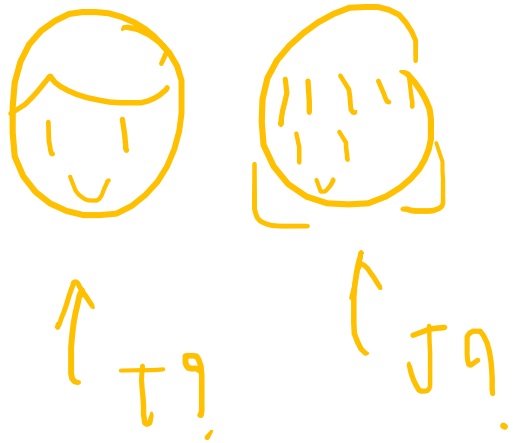


- Artificial Intelligence spans a very wide scope, even out of Machine Learning
- Deep Learning is just a small corner in Machine Learning

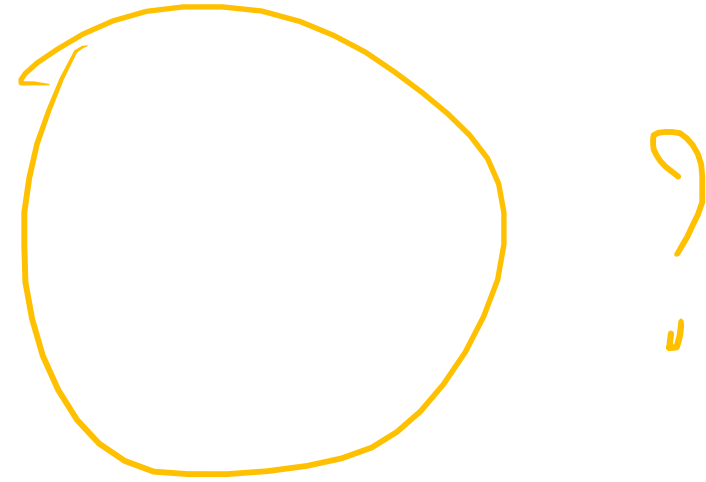
Machine Learning Problems

Supervised Learning

"With label"

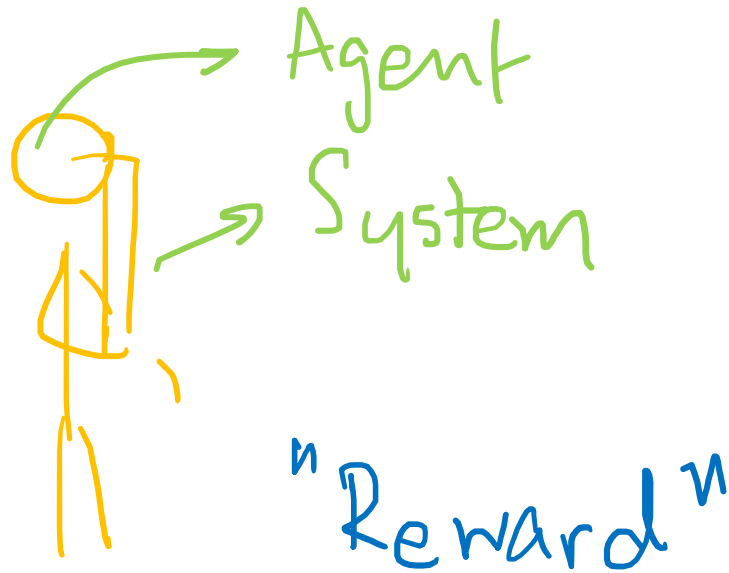


Unsupervised Learning

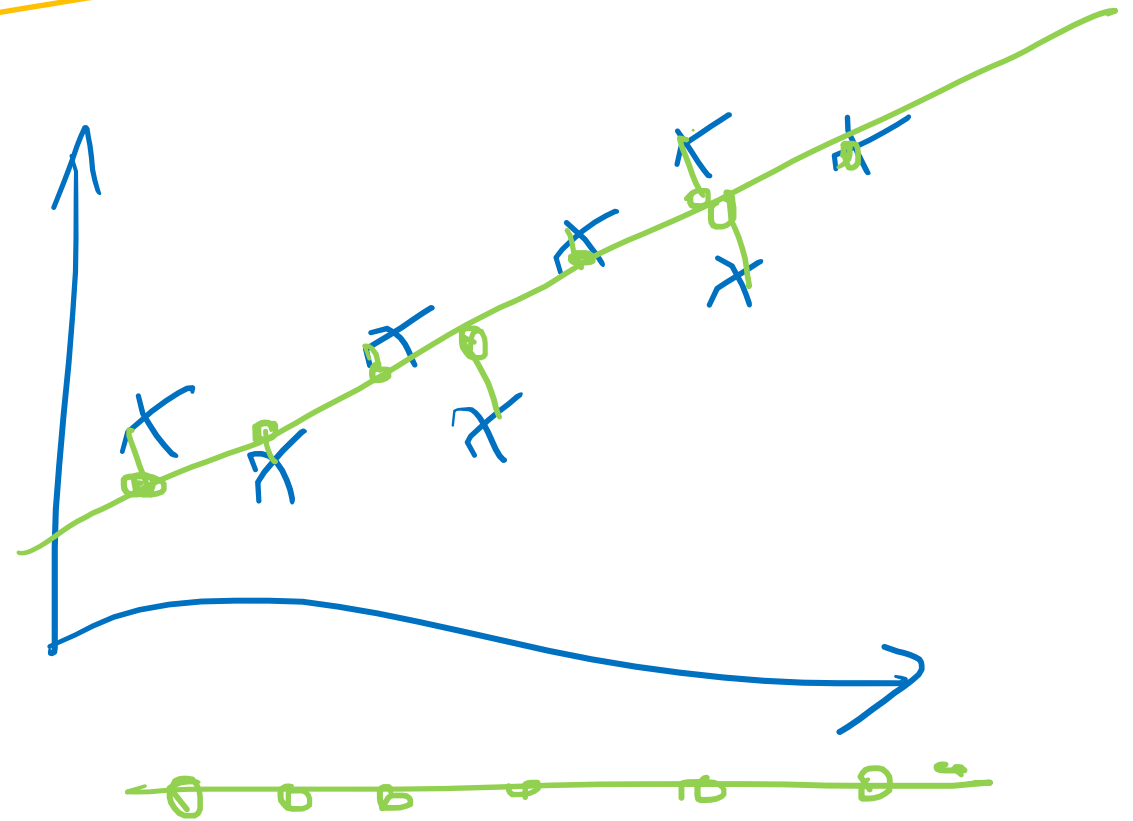


Machine Learning Problems

Reinforcement Learning



Dimensionality Reduction



Supervised Learning

Classification



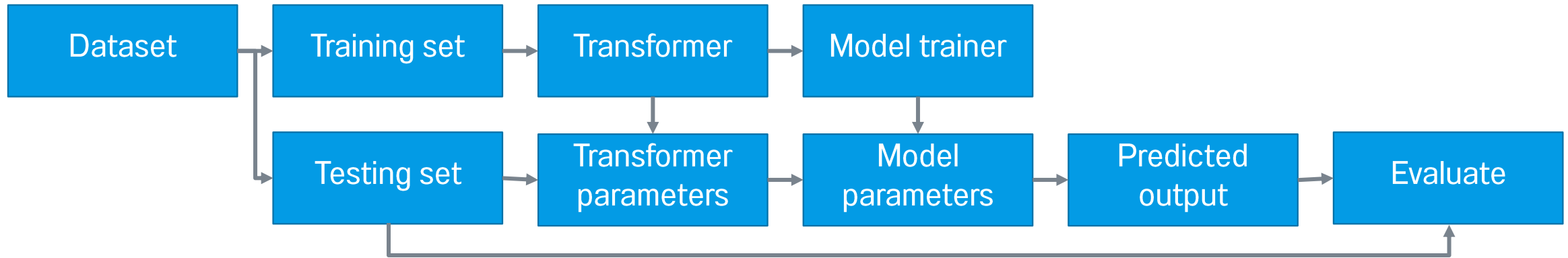
Discrete Values

Regression

Continuous

①	→	2 M
②	→	4 M
③	→	~ -6 M

Machine Learning Pipelines



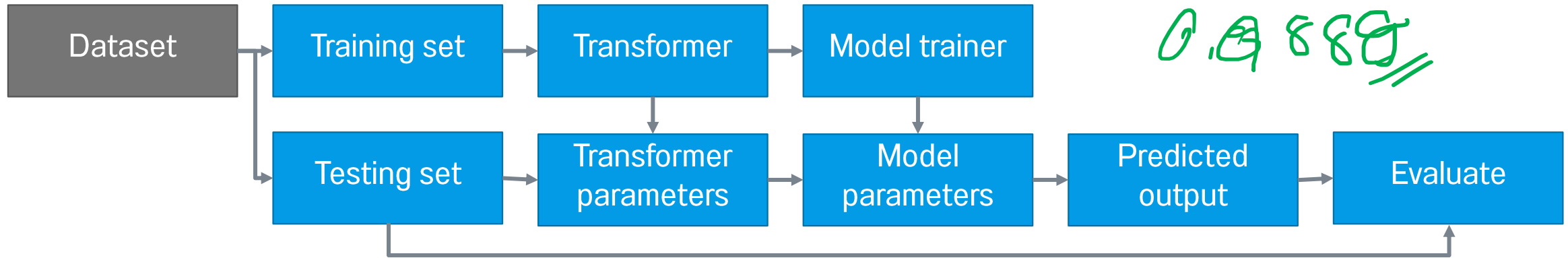
Problem

We wanted to create a model which given an input on one's English grade and score, returns an output whether x_{yr} * English skills is good or not.

* gender-neutral pronoun

his/her

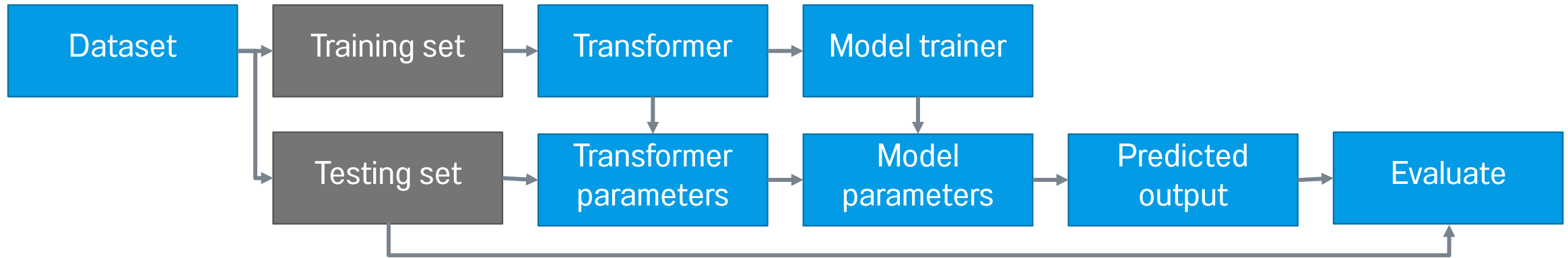
Dataset



- The data where we're interested in
- Contains both the expected “input” and the “output”
- Student A: IELTS 9, O-NET English 86, Good level of English
- Student B: IELTS 6, O-NET English 89, Insufficient level of English
- Student B: IELTS 8, O-NET English 92, Good level of English

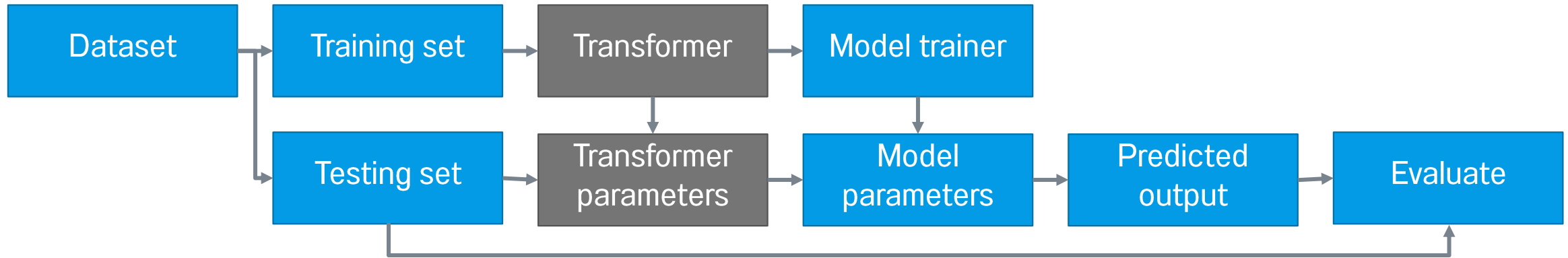
Training
Test

Train-Test Splitting



- We need to split our data into two chunks:
 - Training set, used to teach our model on how to “answer” based on this set.
 - Testing set, used as the “answer key” to the model’s predicted output
 - Testing with training set is cheating!

Transformer



- Who is better between...

- Student A: IELTS 9, O-NET English 86, Total: 95
- Student B: IELTS 6, O-NET English 89, Total: 95

- Transformer copes with these kinds of troubles

↓ 1.86

↑
≈ 1.65

IELTS

9/121

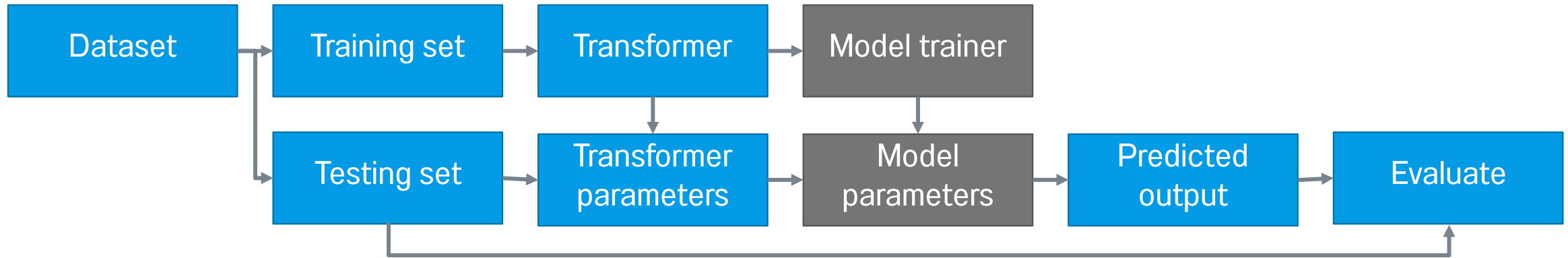
6/9 = $\frac{2}{3}$

O-NET

86/100 = 0.86

89/100 = 0.89

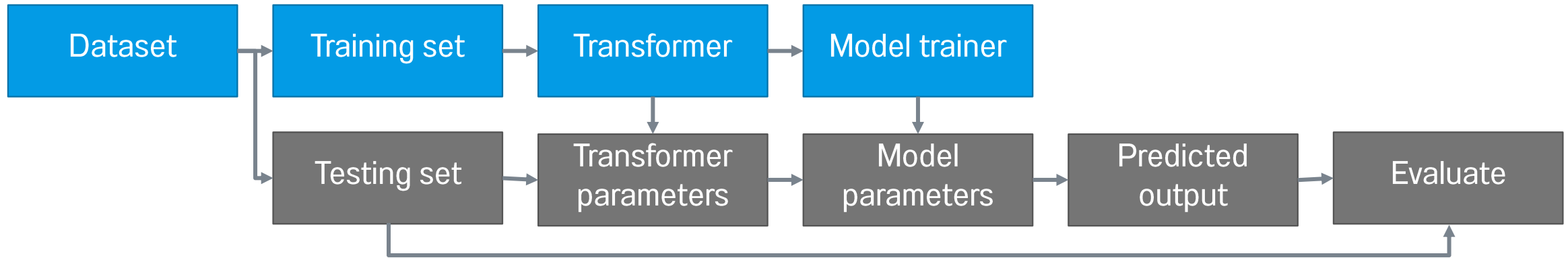
Model training



- Create a model with the desired algorithm
- “Teach” the model with the training set data

If IELTS > 0.75
→ Return "Good"
Else
→ Return "Insufficiently"

Evaluation



- Feed the testing set into the same manner to the training set
- Get the model's output, compare it to the true answer.

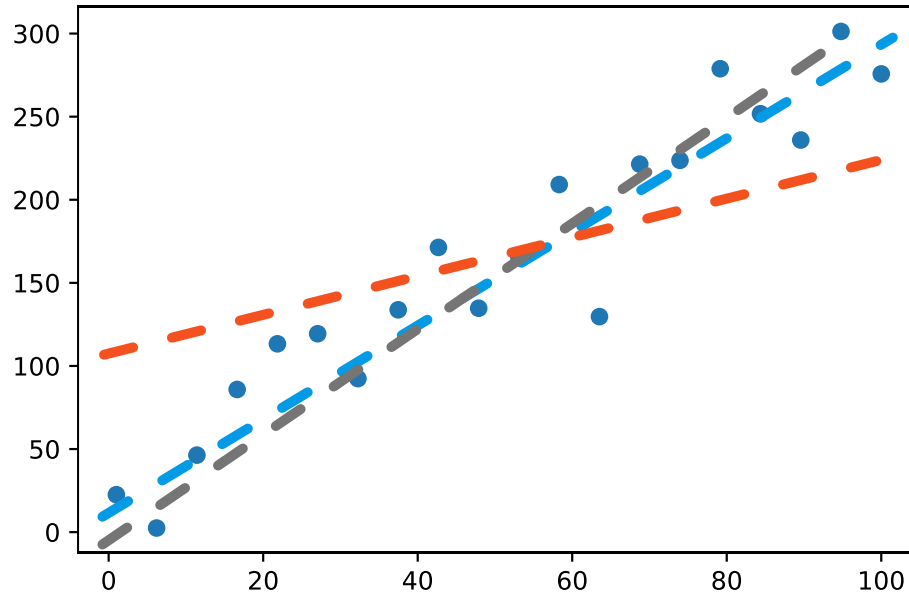
Linear Regression

Linear Regression: A Short Example

Number of bedrooms in house	Number of WCs in house	House price
2	1	2100000
4	3	4300000
2	1.5	2150000
3	2	?

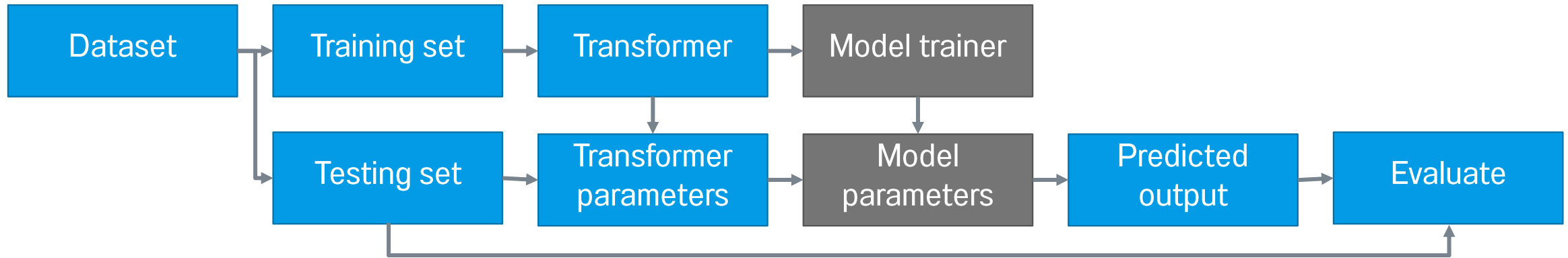
$$\hat{y} = (1M \times 950000) + (0.1M \times 950000) = 3,200,000$$

Linear Regression



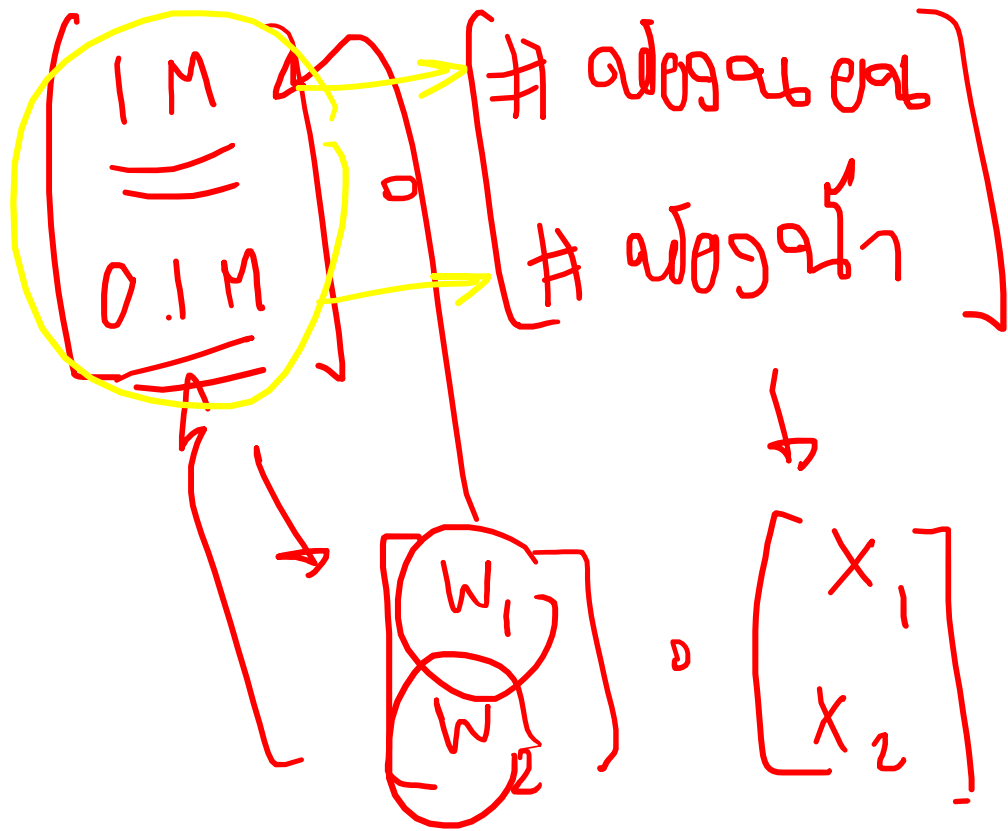
- Given this data, can we determine the **best line** to model the relationship between X and Y?
- Which one is better—the blue or the red?
- Which one is better—the blue or the green?
 - Why this can't be easily answered like the former question?

Linear Regression Model



Mathematics, here we go!

Linear Regression Model



- Suppose we have our data points x_1, x_2, \dots, x_n
- $X = [x_1, x_2, \dots, x_n]$
 - Where does x_0 comes from?
- We wanted to find the best value for $W = \hat{G}w_1, w_2, \dots, w_n \hat{H}$
- The scalar product between W and X will be our model's returned values

Good model: Ordinary Least Square

	True	Pred	Error
①	2M	1M	$(2-1)^2$
②	3M	2M	$(3-2)^2$
③	4M	6 2 M	$(4-\cancel{2})^2$

Error

$$= 1^2 + 1^2 + 2^2 = 6M^2$$

$$\text{Error} = \sum (\text{True} - \text{Pred})^2$$

- We need to “mathematically” describe what a good model is
- We can’t really tell how good models are, but at last we can compare models performing on the same task

A little math...

$$W = [w_0 \quad w_1 \quad w_2 \quad \dots \quad w_n]^T$$
$$X = [1 \quad x_1 \quad x_2 \quad \dots \quad x_n]^T$$

→ Features

$$\hat{y} = \frac{W^T X}{[w_0 \quad w_1 \quad w_2] \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}}$$

$$= w_0 + w_1 x_1 + w_2 x_2 + \dots$$

$$L = \sum (y_i - \hat{y}_i)^2$$

↑ Minimise: $\frac{d}{dx} = 0$

$$\sum (y_i - W^T x)$$

Calculating Weights

$$L = \sum (y_i - w^T x_i)$$

$$= \sum (y - Xw)^T (y - Xw)$$

$$= (y^T y - 2y^T Xw + w^T X^T Xw)$$

$$w = (X^T X)^{-1} X^T y$$

$$\left. \begin{array}{l} \frac{\partial}{\partial w} L \\ = X^T X w - y^T X \end{array} \right\}$$

House Prediction Model

# of bedrooms	# of WCs	Price (M)
3	2	35
2	2	30
4	2	47
2	1	28
4	3	50
3	2	33
2	1	27

$$W = (X^T X)^{-1} X^T Y$$

→ Train

$$40 + 12 = 52$$

→ Test

$X =$

$Y =$

$$\text{House price} = \frac{9.88}{4} \times \cancel{\# \text{bedr}} + \frac{3.08}{3} \times \cancel{\# \text{WC}}$$

Observation: Analytical method

- We solve our problem using derivatives, then plug in those numbers to the derived formula, and voila—we obtain our equation!
- This is called **analytical method**—using mathematical tools to solve numerical problems.
- The other method, of which we'll certainly see, is **computational method**.

Perceptron



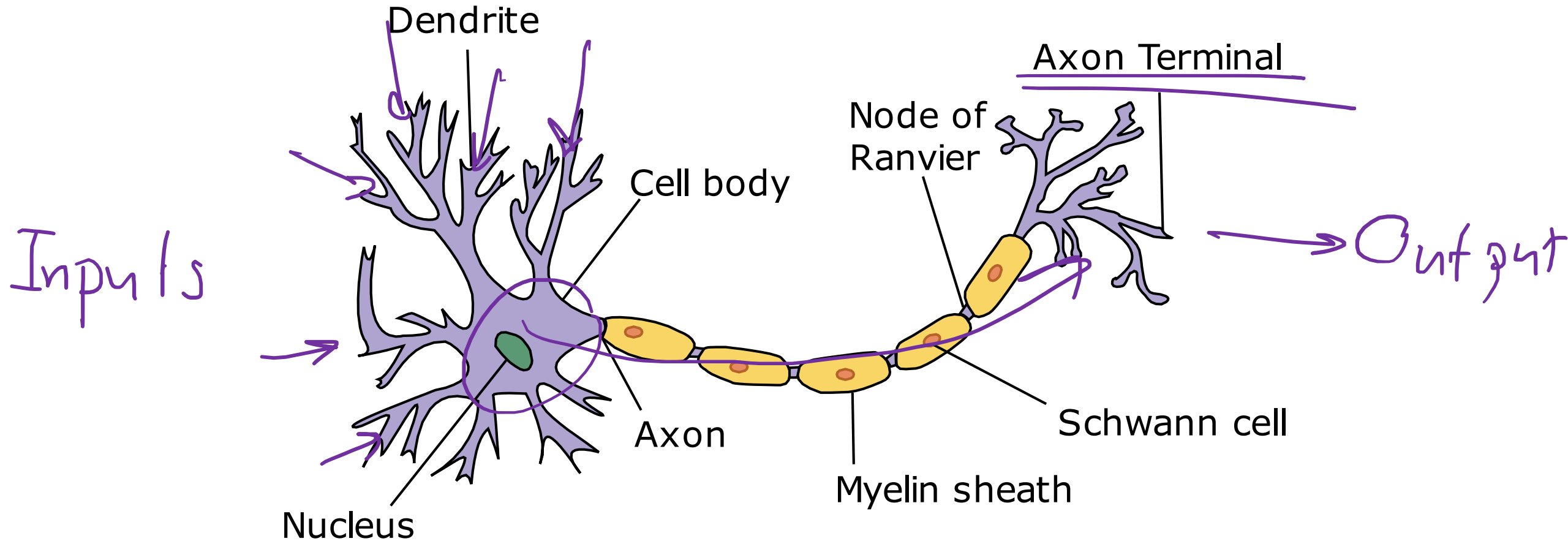

Thailand
Philharmonic Orchestra


PRINCE MAHIDOL HALL



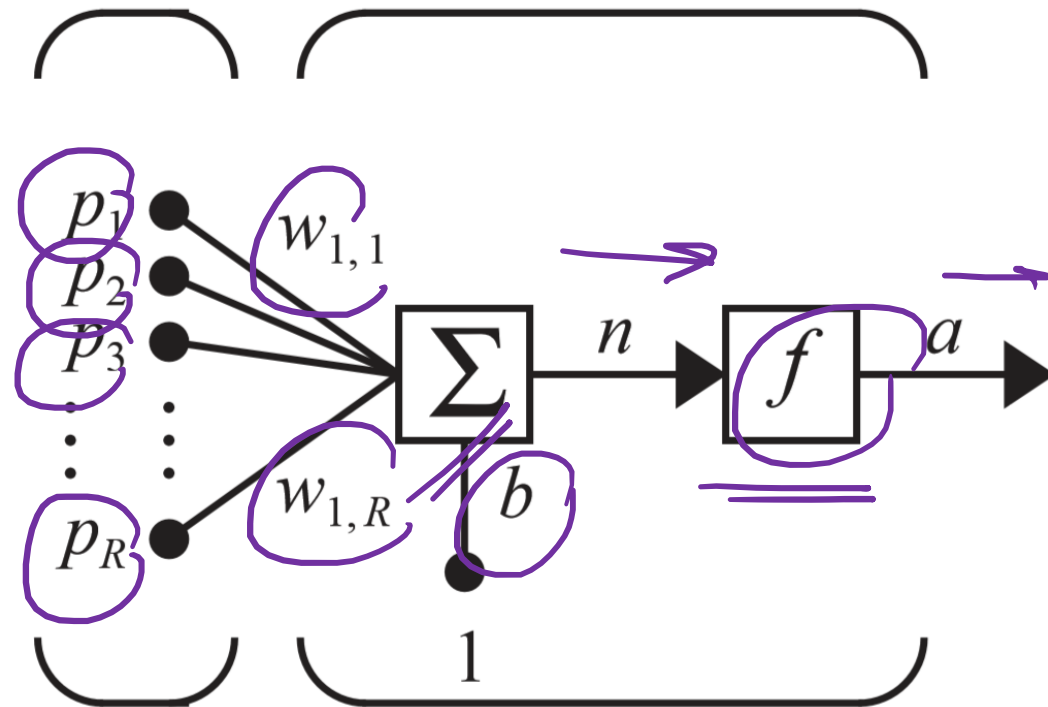


Why are we smart?



Wikimedia Commons / User:Dhp1080 / CC BY-SA
(<http://creativecommons.org/licenses/by-sa/3.0/>)

Introducing Perceptron



$$\underline{a = f(\mathbf{W}\mathbf{p} + b)}$$

Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

- The implementation of the Perceptron consists of the following features

- Inputs: p_1, p_2, \dots, p_R
- Inputs' weights: w_1, w_2, \dots, w_R
- Bias b
- Linear combination Σ
- Activation function f

$$\text{hardlim}(x) = \begin{cases} 1; & x > 0 \\ 0; & \text{otherwise} \end{cases}$$

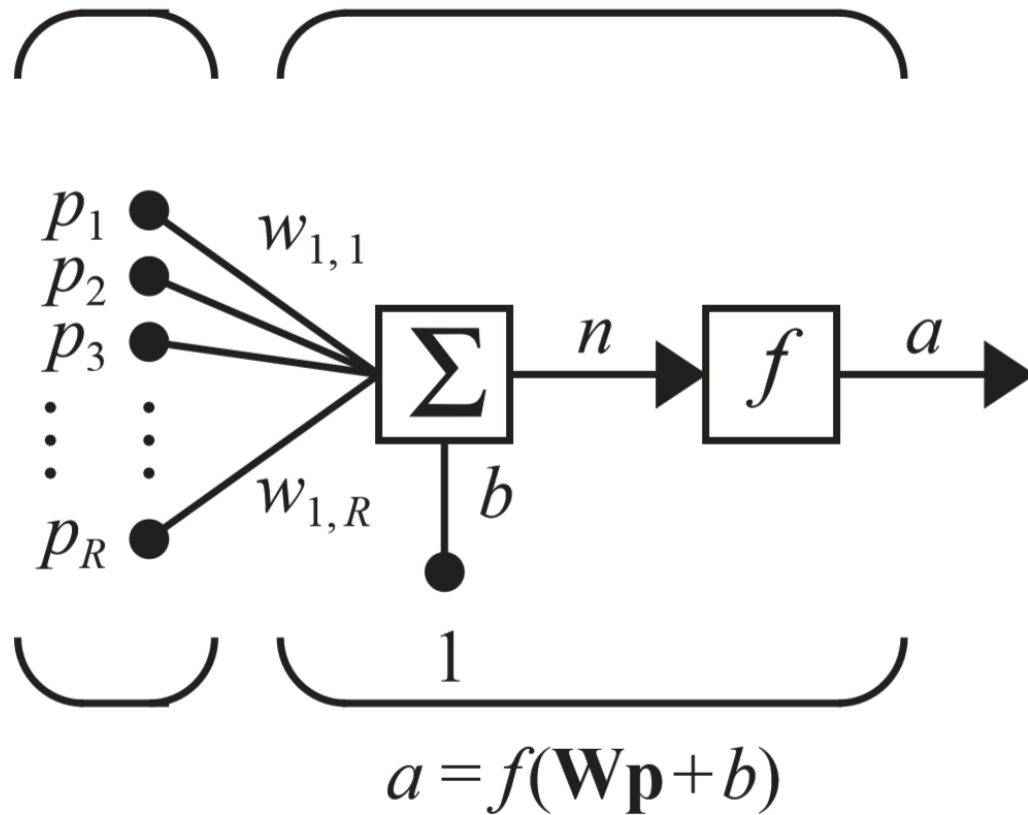
Activation functions

Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$		satlins
Log-Sigmoid	$a = \frac{1}{1 + e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$		poslin
Competitive	$a = 1 \quad \text{neuron with max } n$ $a = 0 \quad \text{all other neurons}$		compet

Table 2.1 Transfer Functions

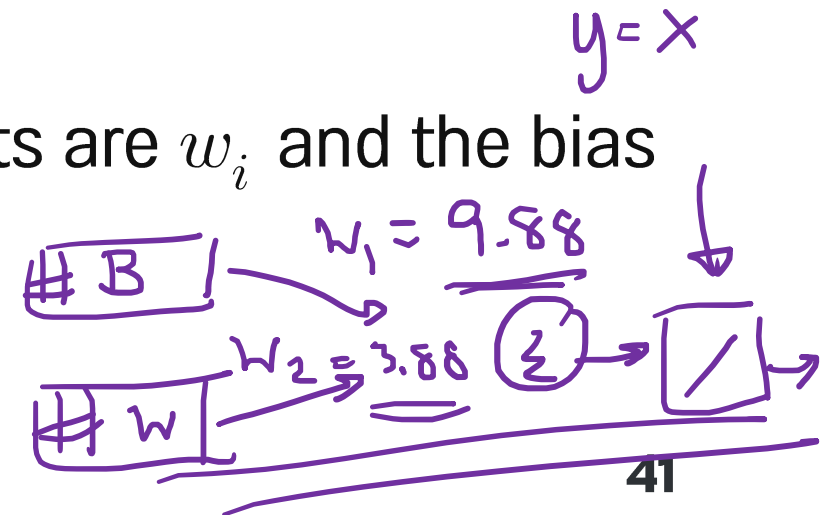
- Activation functions are used to introduce “nonlinearity” to the perceptron
- Without the activation function, the perceptron will act only as a linear tool.
 - What will this cause? We’ll mention it later.
- Popular functions: Log-Sigmoid (Sigmoid), PosLin (ReLU), tansig (tanh)

Perceptron as a linear regressor

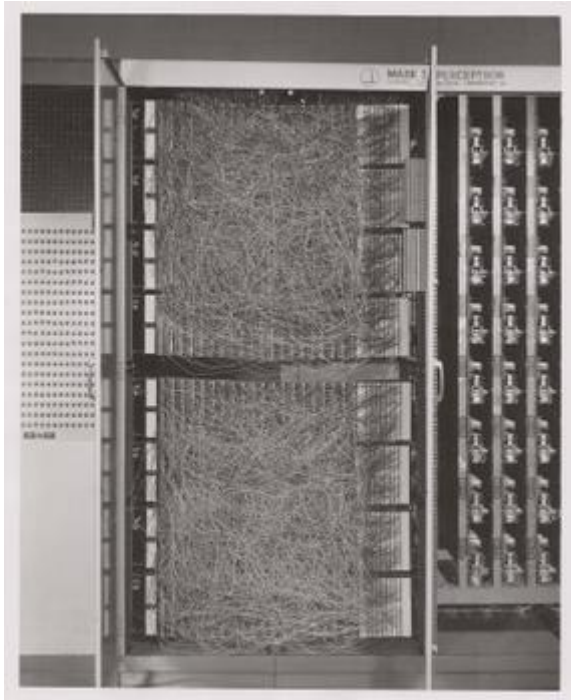


Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

- The perceptron model is quite generalised: we can create a linear regressor
- We simply use a linear function $f(x) = x$ as the activation function.
- The weights are w_i and the bias is w_0 .



The Mark I Perceptron

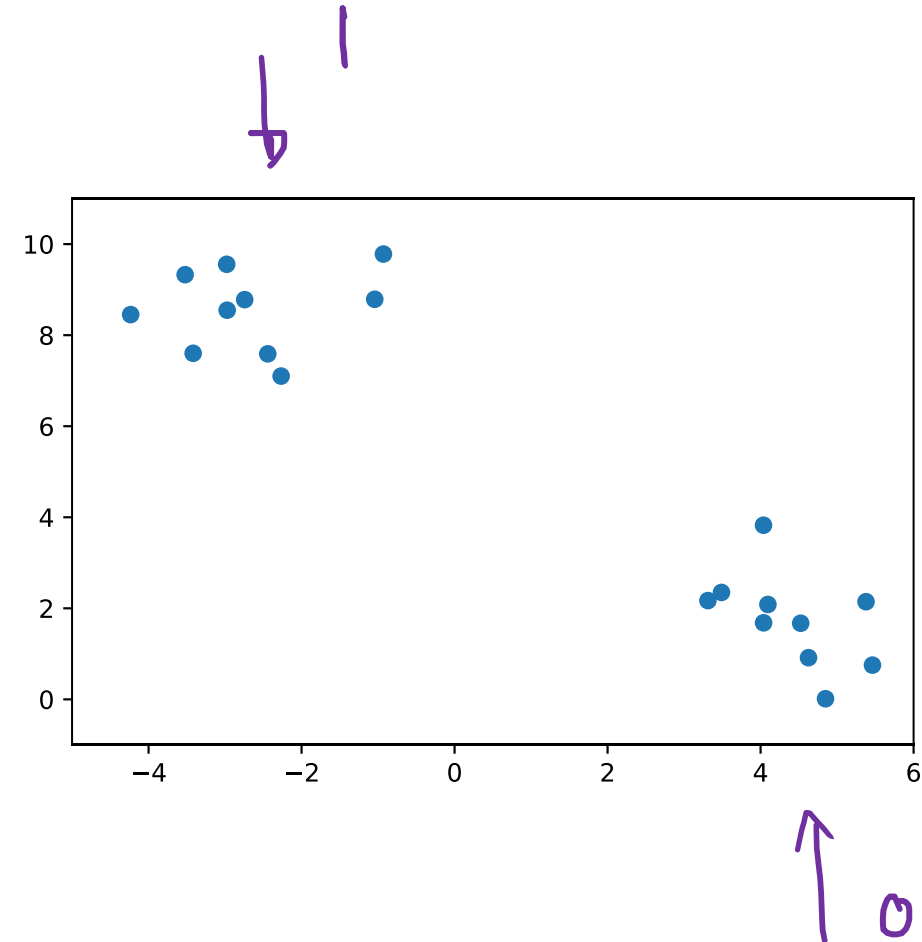


By Source (WP:NFCC#4), Fair use,
<https://en.wikipedia.org/w/index.php?curid=47541432>

- Perceptron, despite being “codes” right now, was intended to be a hardware
- Mark I is the first implementation of Perceptron physically
- Managing 20x20 pixels image
- Manually wired, adaptive weight using motors

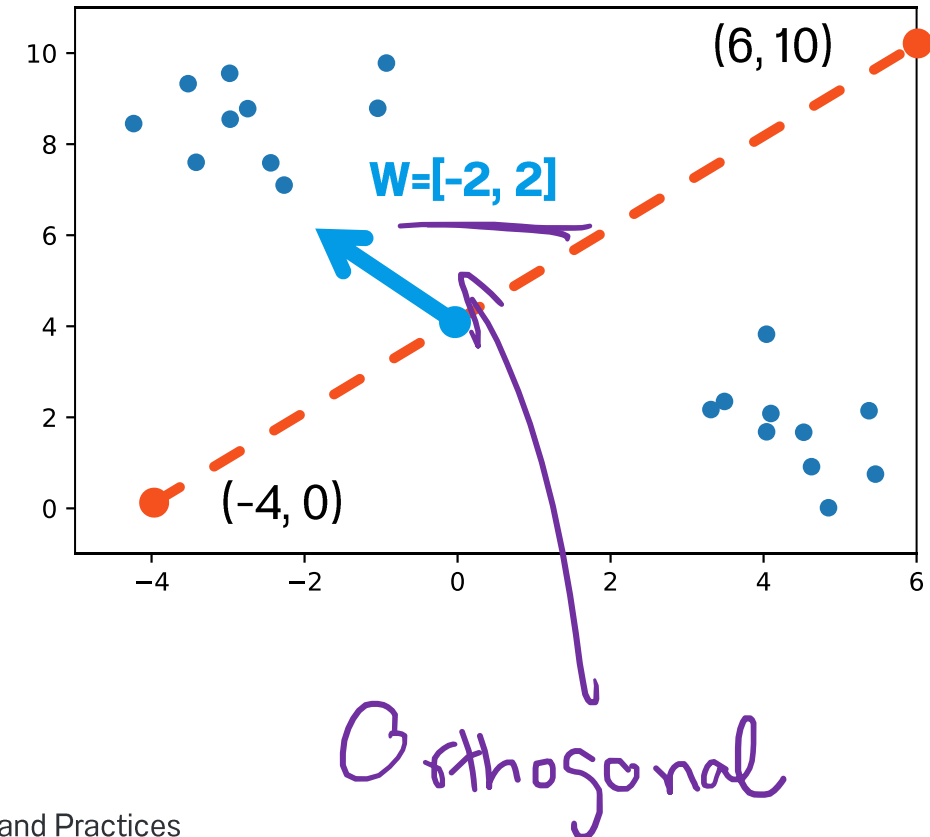
A simple Perceptron classifier

- Given the following data, how can we determine the Perceptron's weight that can classify the two groups of data?
- Perceptron equation:
$$\hat{y} = \text{hardlim}(WX + b)$$
- If $W^T X \geq b$, then the output will be 1.



A simple Perceptron classifier

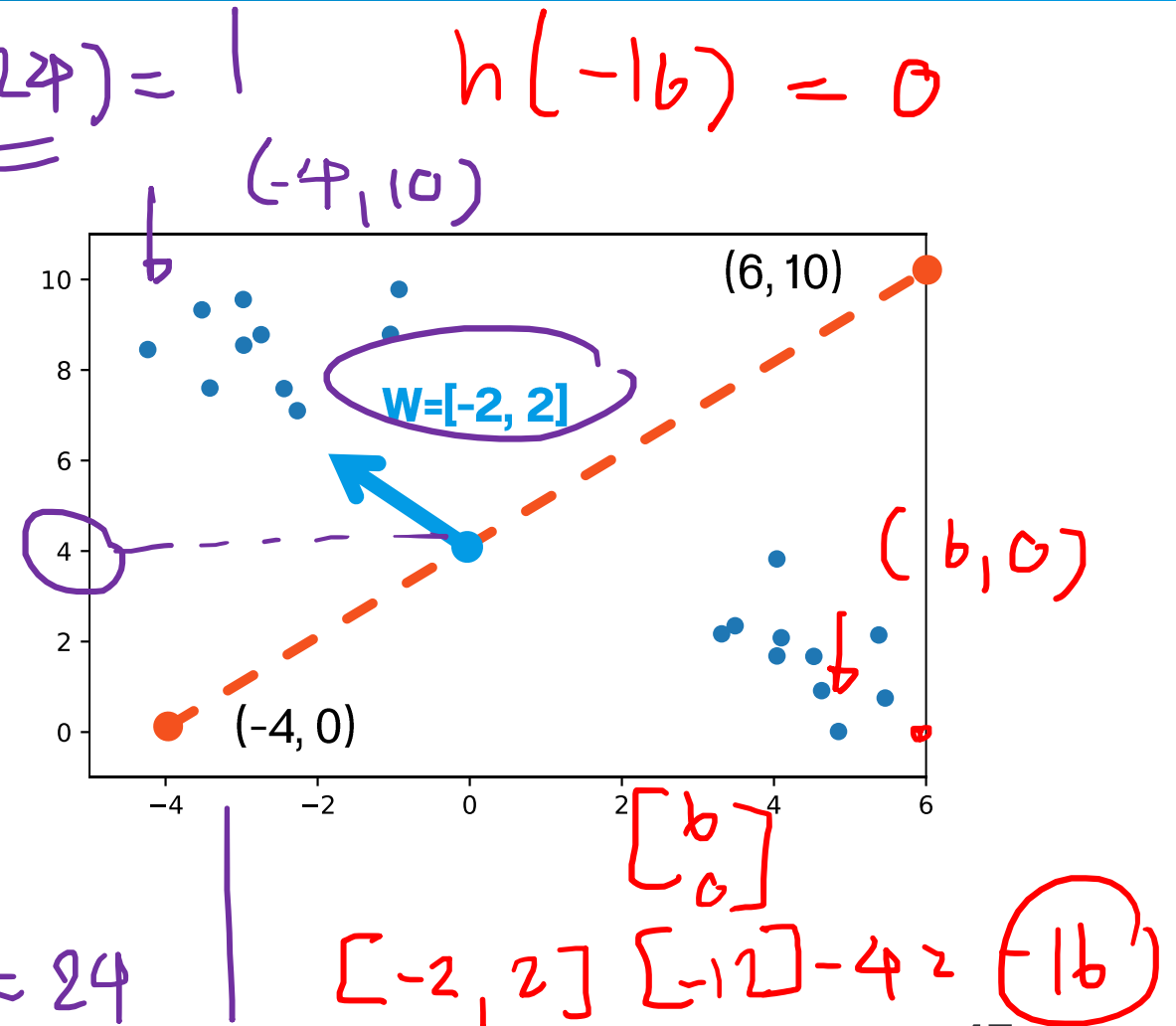
- We can simply draw a line that separates the data into two classes
- From mathematical knowledge, we know that this line is in the form of equation $W^T X + b = 0$
 - The weight vector W is orthogonal to the line
 - The y-intercept of the boundary is equal to $-b$
- Observe that there can be many lines



A simple Perceptron classifier

$$\hat{y} = \text{hardlim} \left(\begin{bmatrix} -2 \\ 2 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - 4 \right)$$

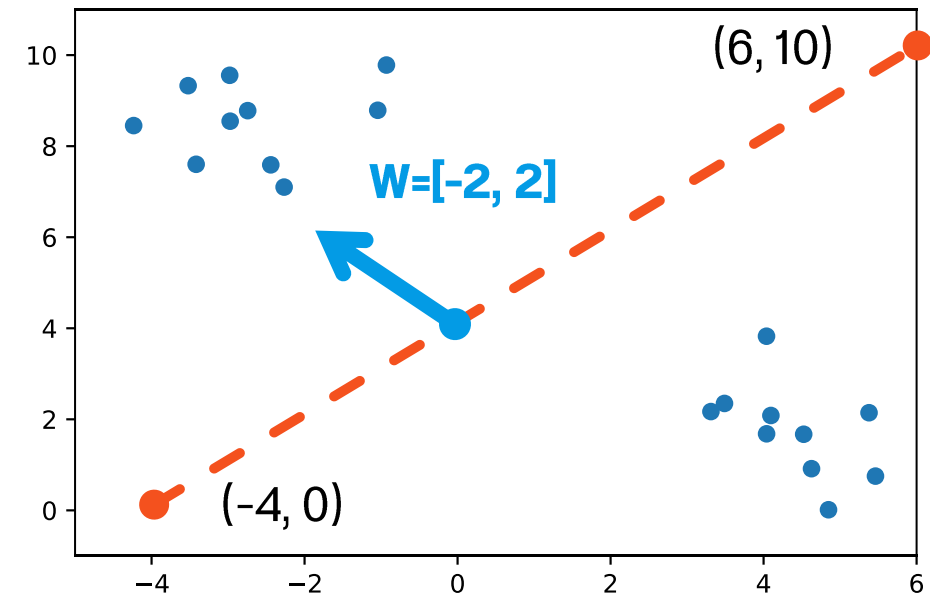
- Try plugging in random data points and observe its behaviour
- Thinking corner: what if the vector W pointed into another direction?



$$(-4, 10); W^T X = \begin{bmatrix} -2 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 8 \end{bmatrix} - 4 = 24$$

Can we do things better?

- Manually drawing weights is not a good idea.
 - *“It was said that you would destroy the manual works, not join them.”* –Obi-wan Kenobi (didn’t said this)
 - It will be soon desperate when you’re working with 3D data, and infeasible when you’re working with 4D data.



Learning 101

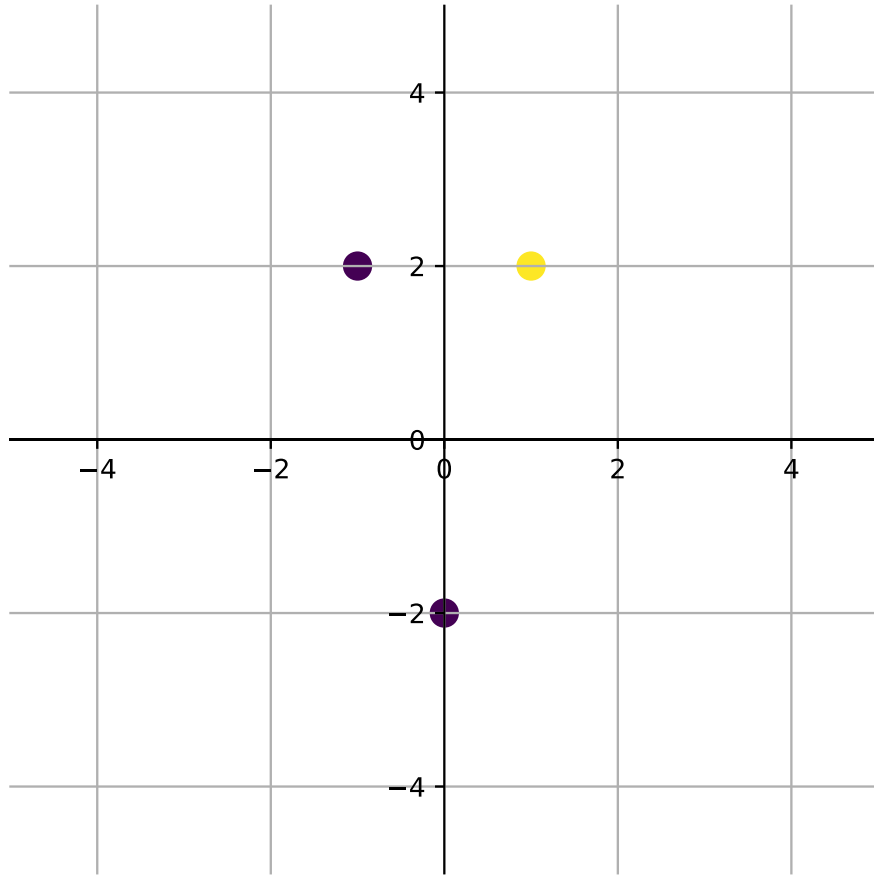
Learning theory

- A field dedicated to methods for algorithms to learn.
- Studies about the complexity and feasibility on learning problems.



This theory, but not learning in this Theory Research Group.

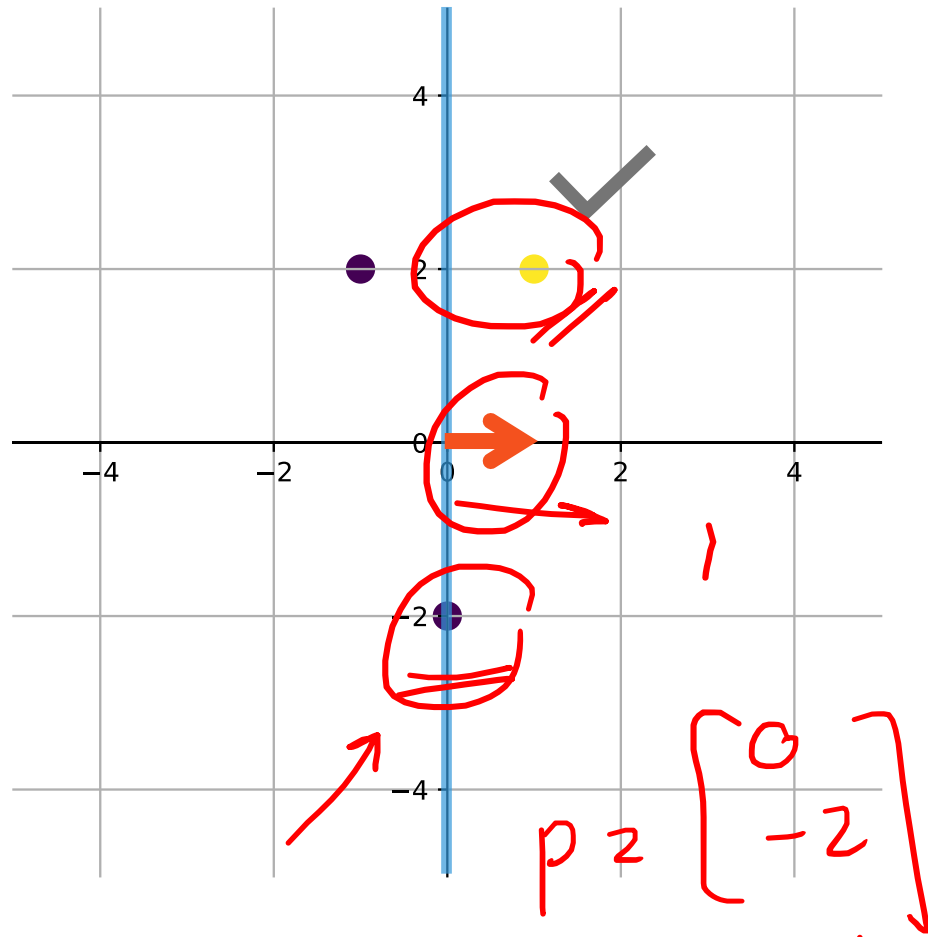
Unified Learning Rule



- Randomly initialise weights
- For x_i in X , do
- Calculate the output
$$\hat{y} = \text{hardlim}(WX + b)$$
- Calculate the error $e = t - o$
- $W' = W + ep$
 - $e = 1, W' = W + p$
 - $e = -1, W' = W - p$
 - $e = 0, W' = W$
- $b' = b + e$

Unified Learning Rule

$$w_2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} e = -1$$



- Randomly initialise weights
- For x_i in X , do
- Calculate the output

$$\hat{y} = \text{hardlim}(WX + b)$$

- Calculate the error $e = t - o$
- $W' = W + ep$

- $e = 1, W' = W + p$

→ • $e = -1$, $W' = W - p$

- $e = 0$, $W' = W$

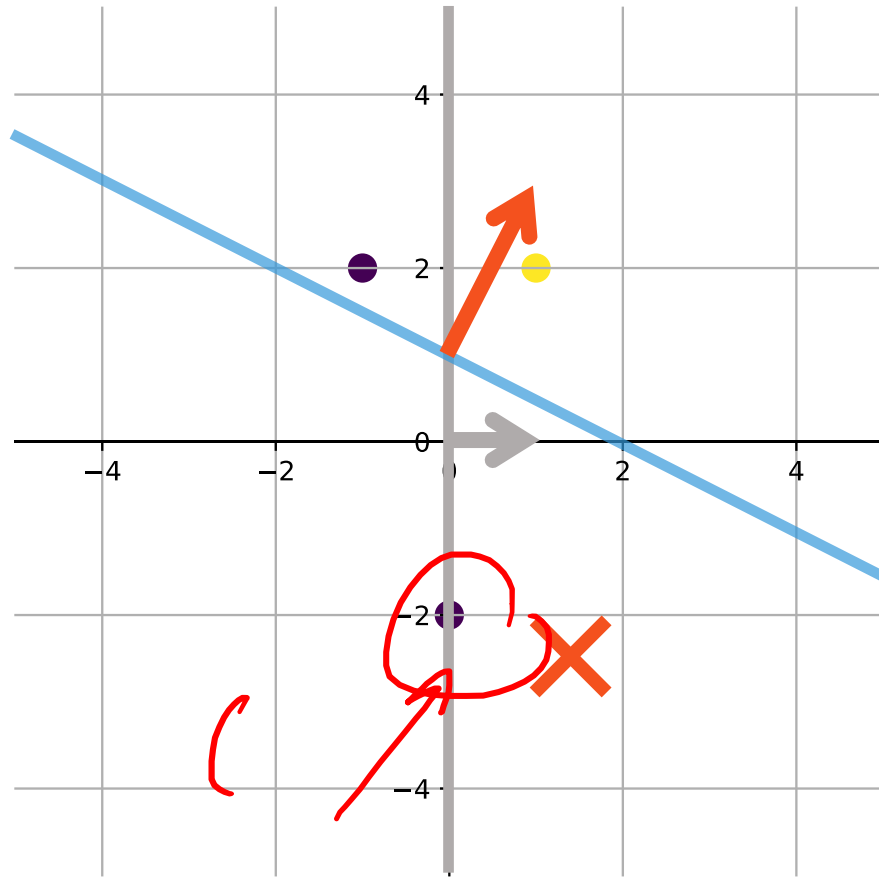
- $b' = b + e$

• $e = 0, W' = W$
 • $b' = b + e$
 $w' + ep = z$

$$= \begin{bmatrix} 1 \\ 0 \end{bmatrix} + (-1) \begin{bmatrix} 0 \\ -2 \end{bmatrix}$$

Practices **50**

Unified Learning Rule



- Randomly initialise weights
- For x_i in X , do
- Calculate the output

$$\hat{y} = \text{hardlim}(WX + b)$$

- Calculate the error $e = t - o$

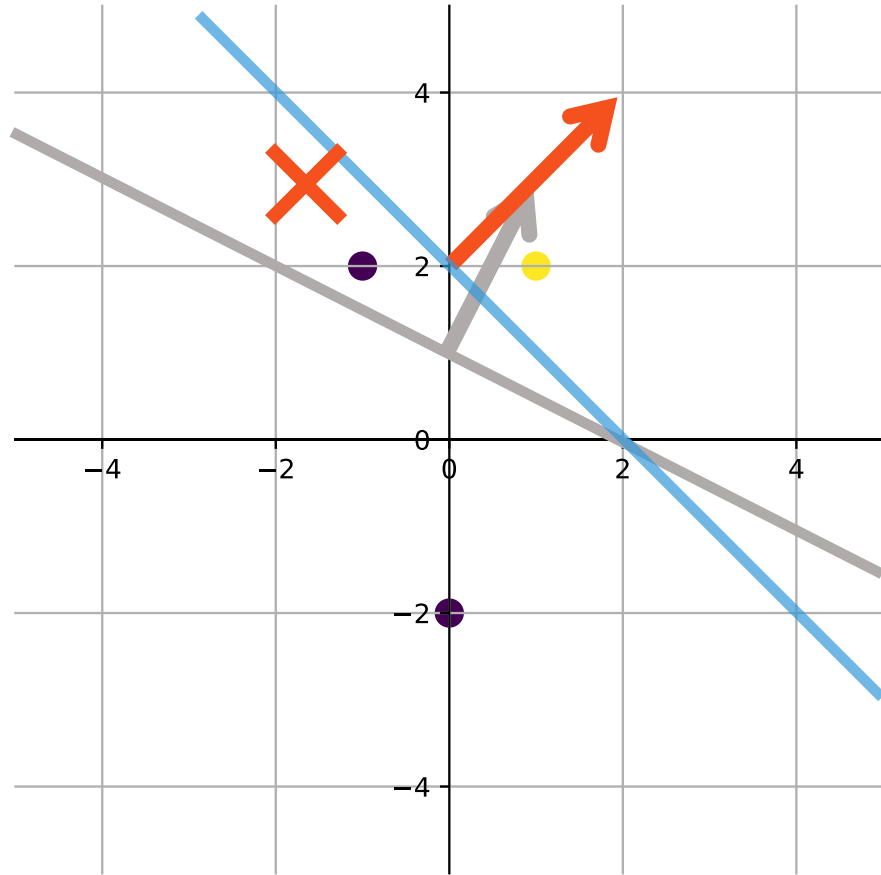
- $W' = W + ep$

- $e = 1, W' = W + p$
- $e = -1, W' = W - p$
- $e = 0, W' = W$

- $b' = b + e$

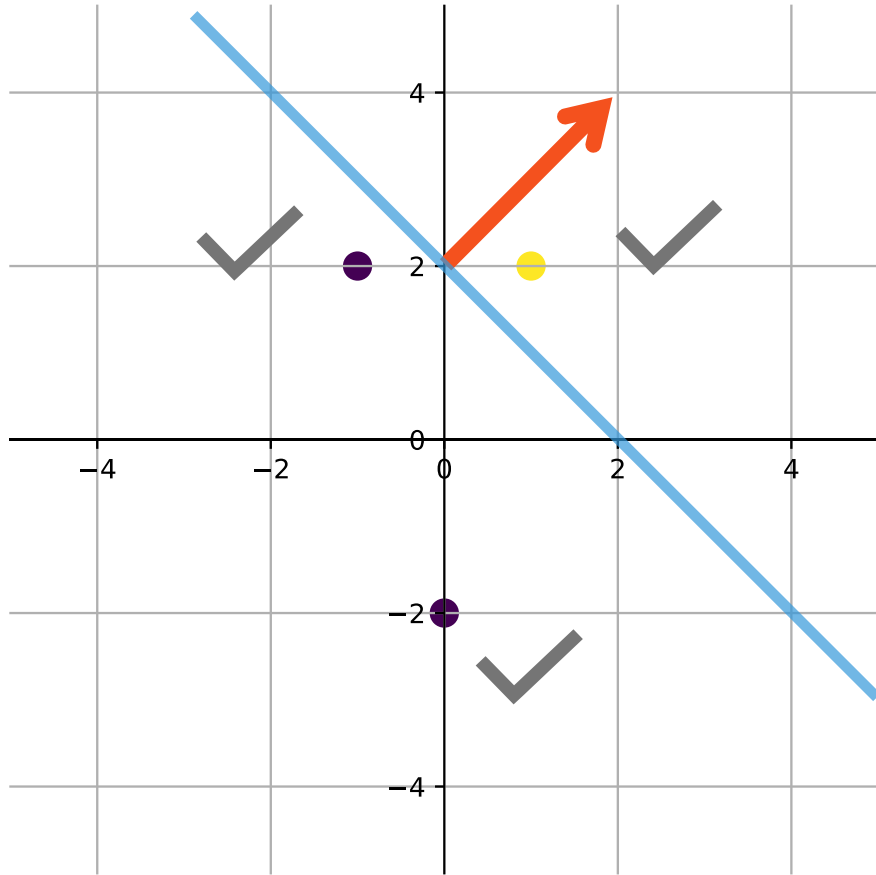
$e = -1$

Unified Learning Rule



- Randomly initialise weights
- For x_i in X , do
- Calculate the output
 $\hat{y} = \text{hardlim}(WX + b)$
- Calculate the error $e = t - o$
- $W' = W + ep$
 - $e = 1, W' = W + p$
 - $e = -1, W' = W - p$
 - $e = 0, W' = W$
- $b' = b + e$

Unified Learning Rule



- Randomly initialise weights
- For x_i in X , do
- Calculate the output
 $\hat{y} = \text{hardlim}(WX + b)$
- Calculate the error $e = t - o$
- $W' = W + ep$
 - $e = 1, W' = W + p$
 - $e = -1, W' = W - p$
 - $e = 0, W' = W$
- $b' = b + e$

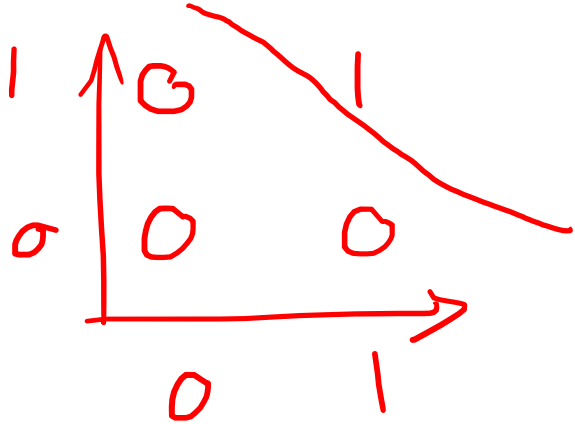
Perceptron

...the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

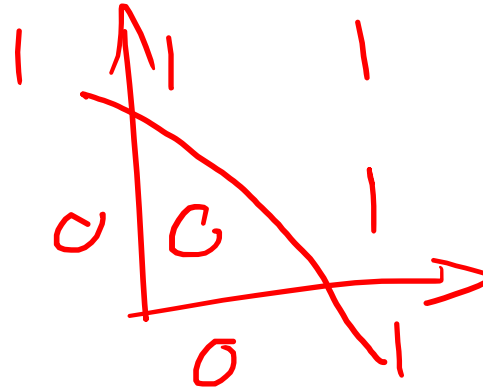
A Sociological Study of the Official History of the
Perceptrons Controversy, Mikel Olazaran

AND and OR gates with Perceptron

AND gate

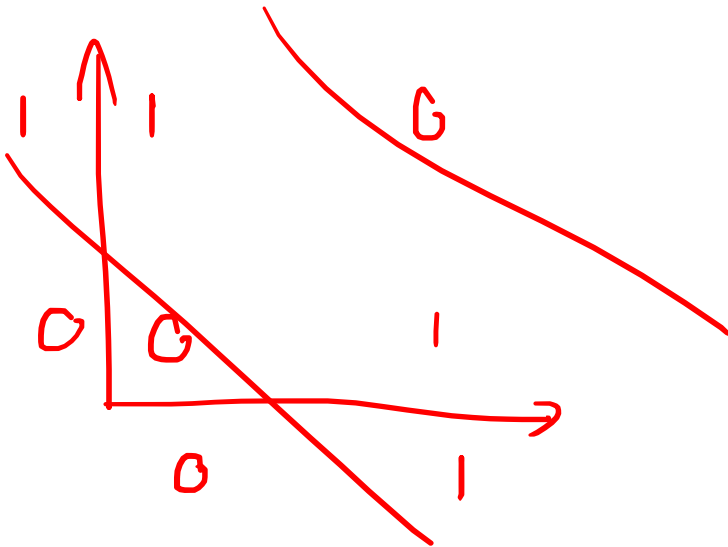


OR gate



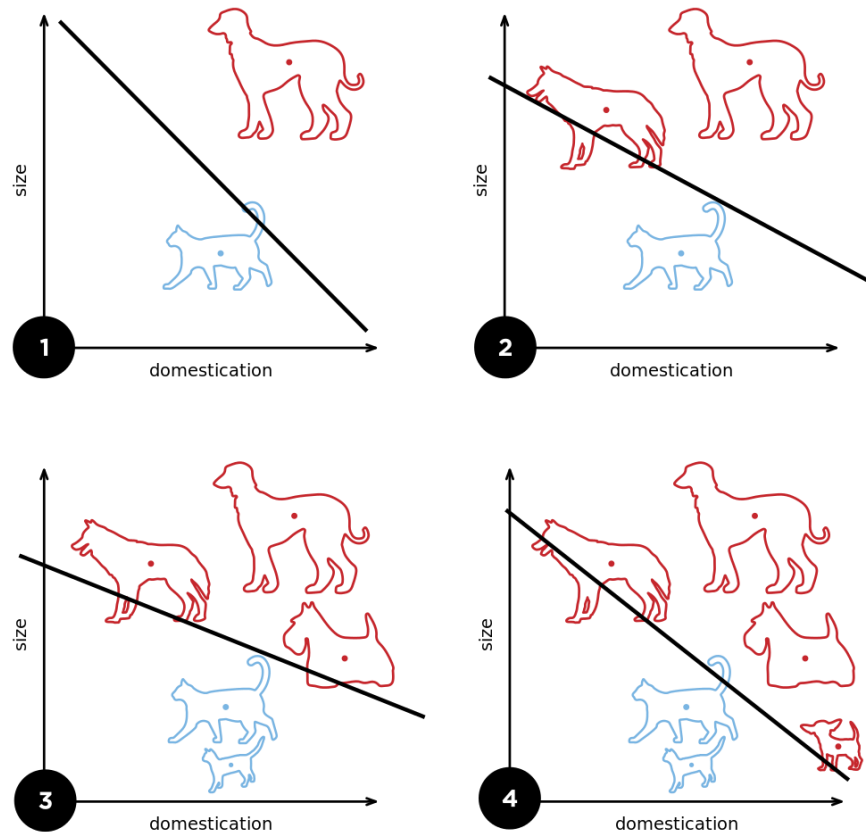
XOR gate with Perceptron

XOR Gate



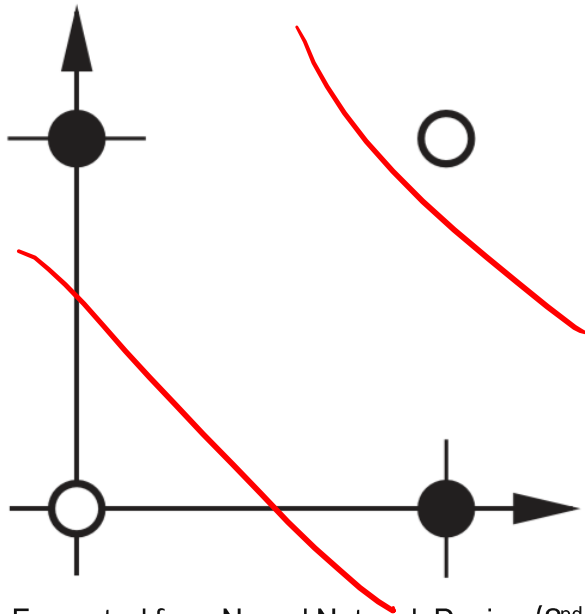
- Thinking corner: Why can't a perceptron be used to create a XOR gate?

Linearly Separable Problems

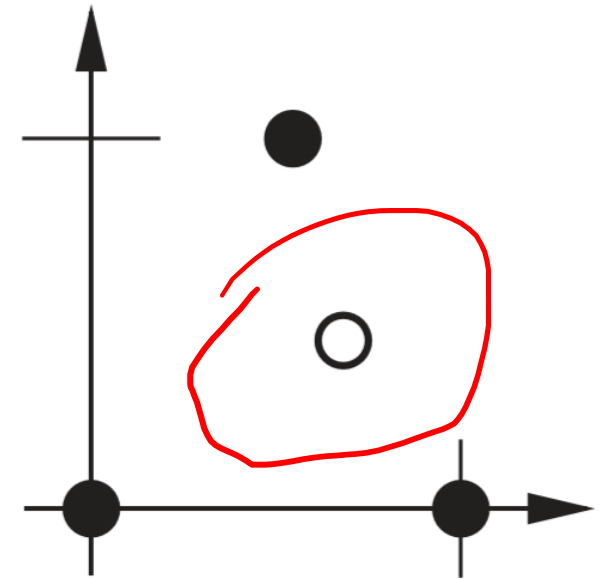
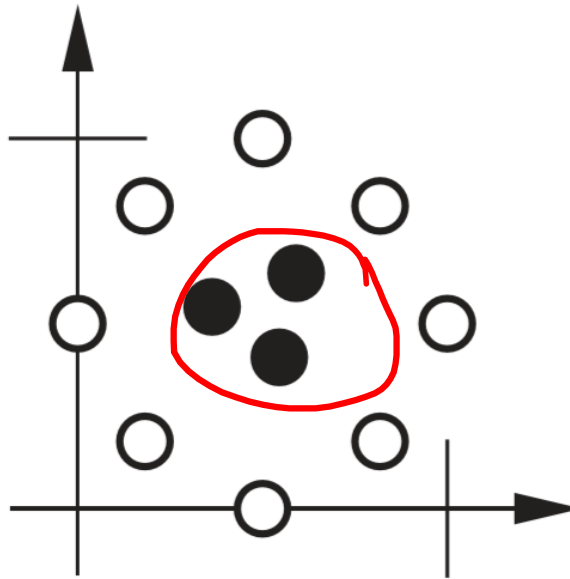


- Perceptrons can solve linearly separable problems
- It is incapable of solving nonlinearly separable problems
- You need two lines to separate the XOR gate answers—this is incapable with Perceptron.

Linearly Inseparable Problems



Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.



How can we make sure it converge?

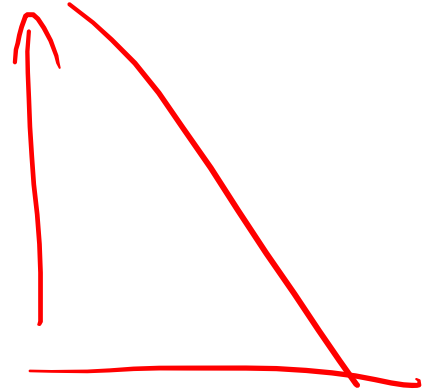
T rounds; M updates.

$$\|W_{M+1}\|_2 = \sqrt{\sum (\|W_{t-1}\|_2^2 - (\|W_t\|_2^2))} =$$

$$\leq \sqrt{M\gamma^2}$$

$\rho > 0 \quad W \in \mathbb{R}^d \text{ s.t. } \rho < \frac{y_t \langle w, x_t \rangle}{\|W\|_2}$

Linearly separable



Optimisation Problem

Taylor series

$$\begin{aligned} F(x) = & F(x^*) + \frac{d}{dx}F(x)\Big|_{x=x^*}(x-x^*) \\ & + \frac{1}{2}\frac{d^2}{dx^2}F(x)\Big|_{x=x^*}(x-x^*)^2 + \dots \\ & + \frac{1}{n!}\frac{d^n}{dx^n}F(x)\Big|_{x=x^*}(x-x^*)^n + \dots \end{aligned}$$

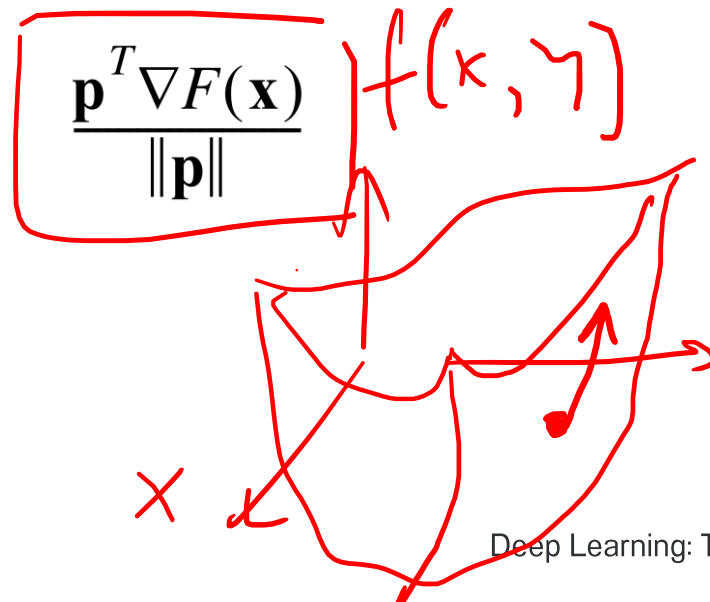
- We can approximate a function according to its derivatives and higher order derivatives at a single point.
- Taylor series will be used to approximate a performance index of our model.

$$F(x) \approx \sum_{i=1}^{\infty} \frac{1}{i!} \frac{d^i}{dx^i} F(x) \Big|_{x=x^*}$$

Gradient

$$\nabla F(\mathbf{x}) = \left[\frac{\partial}{\partial x_1} F(\mathbf{x}) \quad \frac{\partial}{\partial x_2} F(\mathbf{x}) \quad \dots \quad \frac{\partial}{\partial x_n} F(\mathbf{x}) \right]^T$$

- A gradient of function $F(\mathbf{x})$ is the partial derivation vector w.r.t. to all of x_i
- We can denote such the value in an arbitrary direction of \mathbf{p} by projecting the gradient onto \mathbf{p} .



x, y

x

y

Hessian

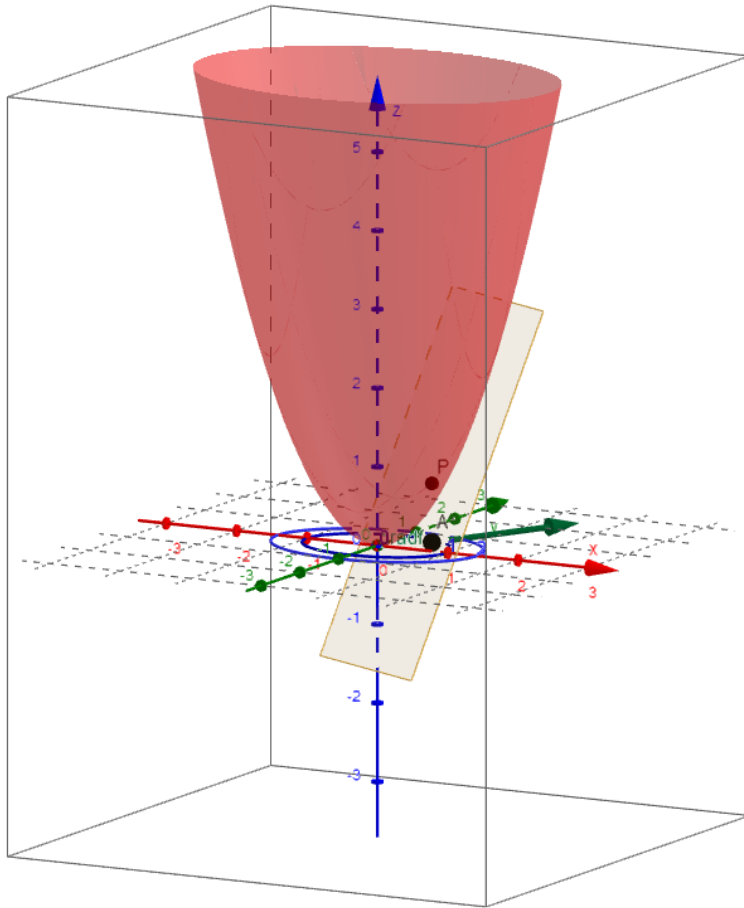
$$f(\mathbf{x}) = \underline{x_1} + \underline{x_2} + \underline{x_3} \dots \underline{x_n}$$

$$\nabla^2 F(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} F(\mathbf{x}) & \frac{\partial^2}{\partial x_1 \partial x_2} F(\mathbf{x}) & \dots & \frac{\partial^2}{\partial x_1 \partial x_n} F(\mathbf{x}) \\ \frac{\partial^2}{\partial x_2 \partial x_1} F(\mathbf{x}) & \frac{\partial^2}{\partial x_2^2} F(\mathbf{x}) & \dots & \frac{\partial^2}{\partial x_2 \partial x_n} F(\mathbf{x}) \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1} F(\mathbf{x}) & \frac{\partial^2}{\partial x_n \partial x_2} F(\mathbf{x}) & \dots & \frac{\partial^2}{\partial x_n^2} F(\mathbf{x}) \end{bmatrix}$$

$$\frac{\mathbf{p}^T \nabla^2 F(\mathbf{x}) \mathbf{p}}{\|\mathbf{p}\|^2}$$

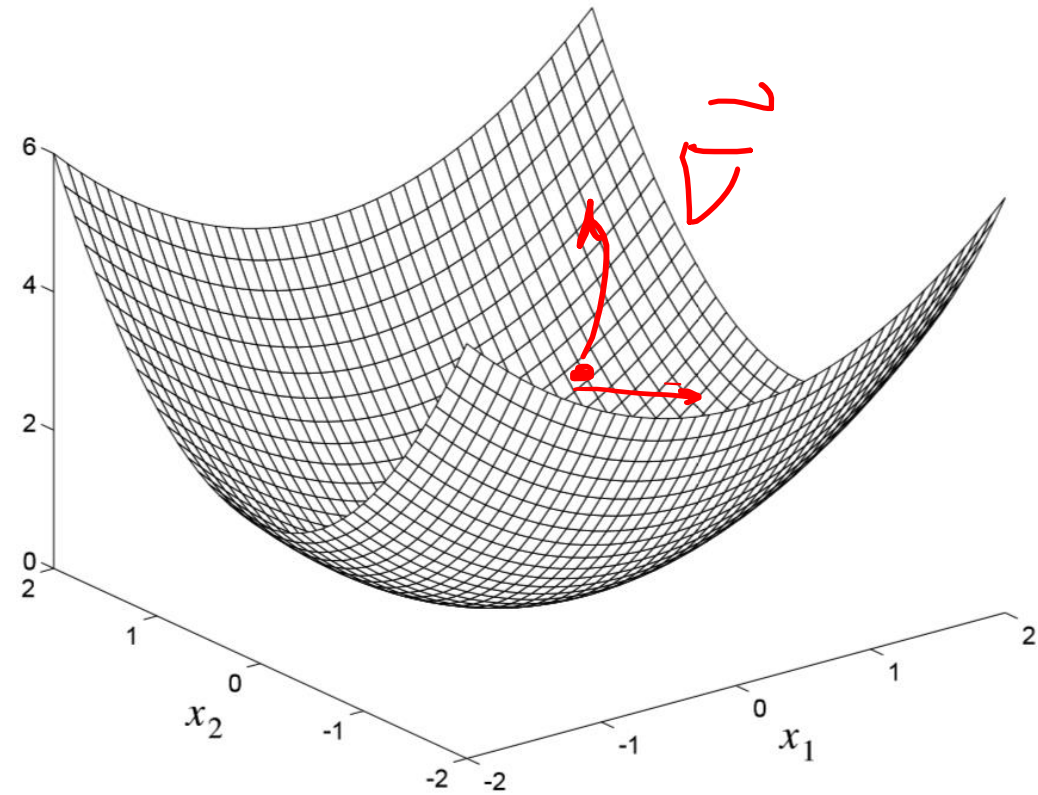
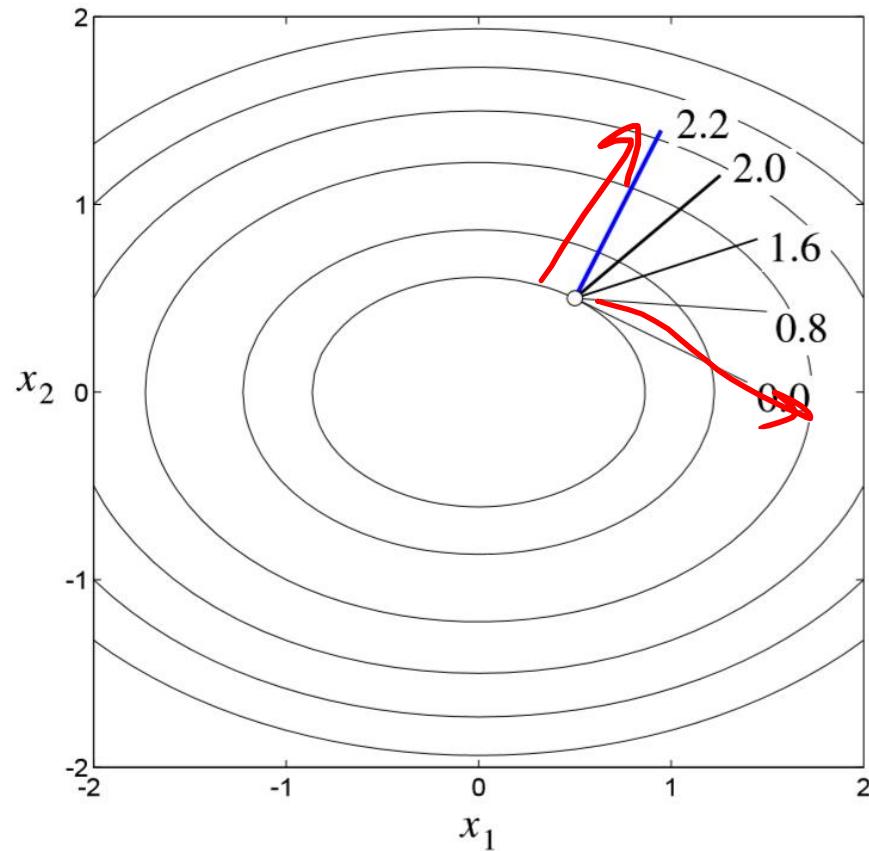
- A hessian of function $F(\mathbf{x})$ is the second order partial derivation matrix w.r.t. to all of x_{ij}
- In the same manner with gradient, we can calculate the direction of hessian on an arbitrary direction of \mathbf{p} .

Demo: Gradient of $z = x^2 + 2y^2$



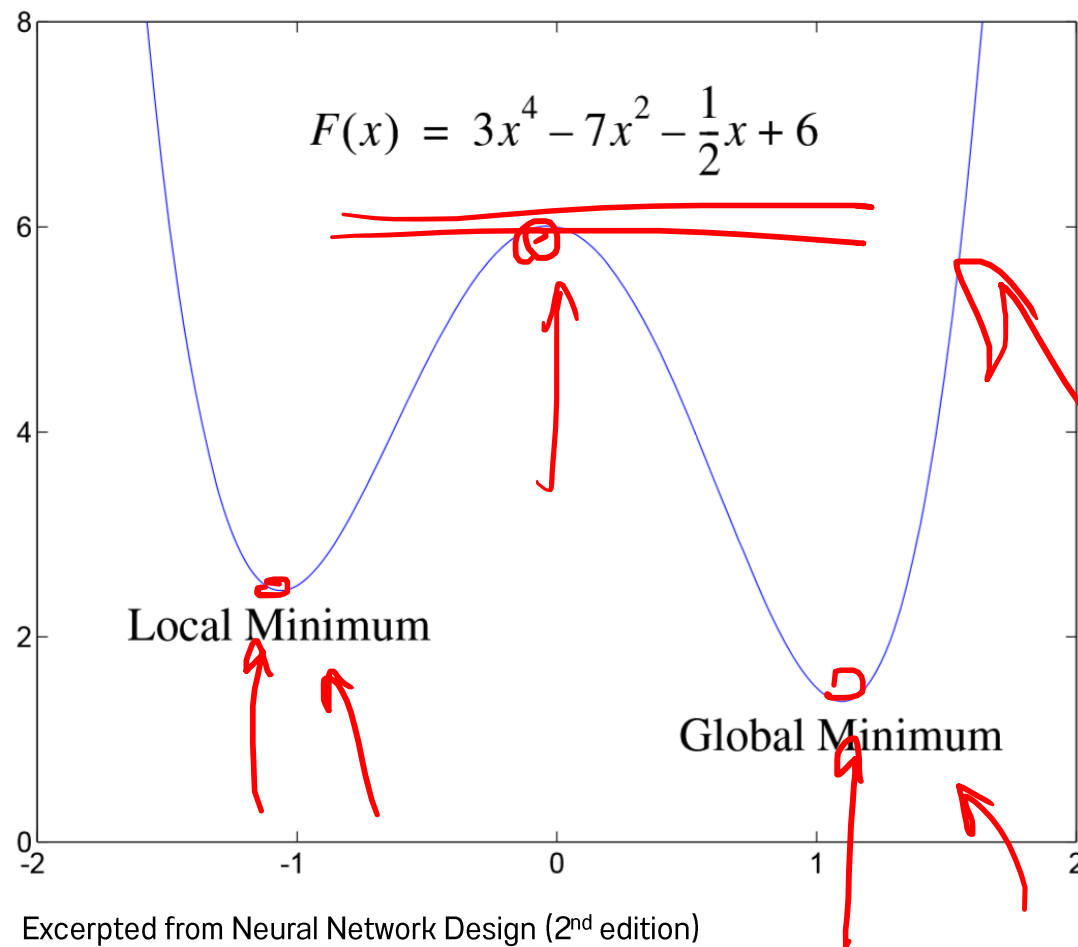
- Try calculating the gradient by hands
- Try visualising the gradient with <https://www.geogebra.org/m/sWsGNs86>
- Try calculating the gradient on the direction of $p = [-2, 1]$

Quadratic Function and Directional Derivatives



Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

Performance Optimisation

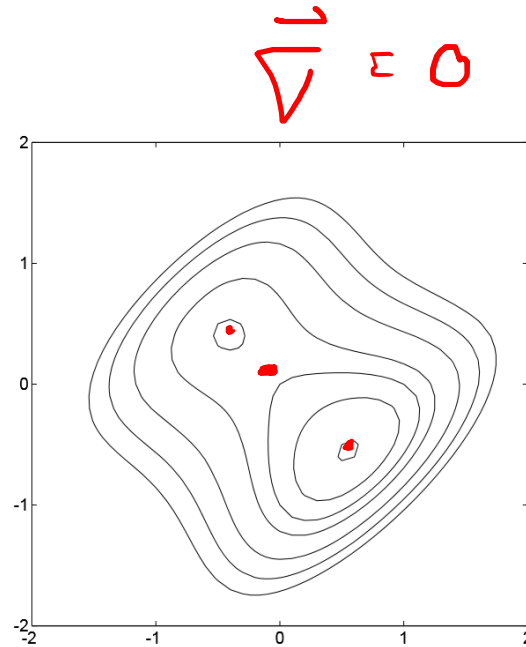


$$\frac{d}{dx} = 12x^3 - 14x - \frac{1}{2} = 0$$

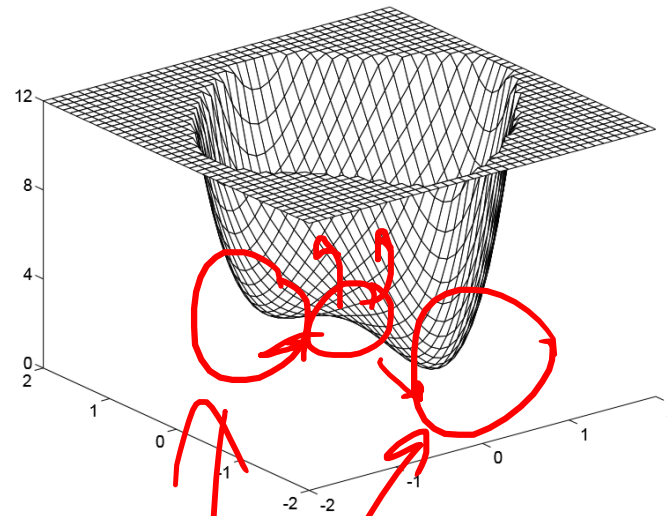
Solve for x

Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

Performance Optimisation



Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.



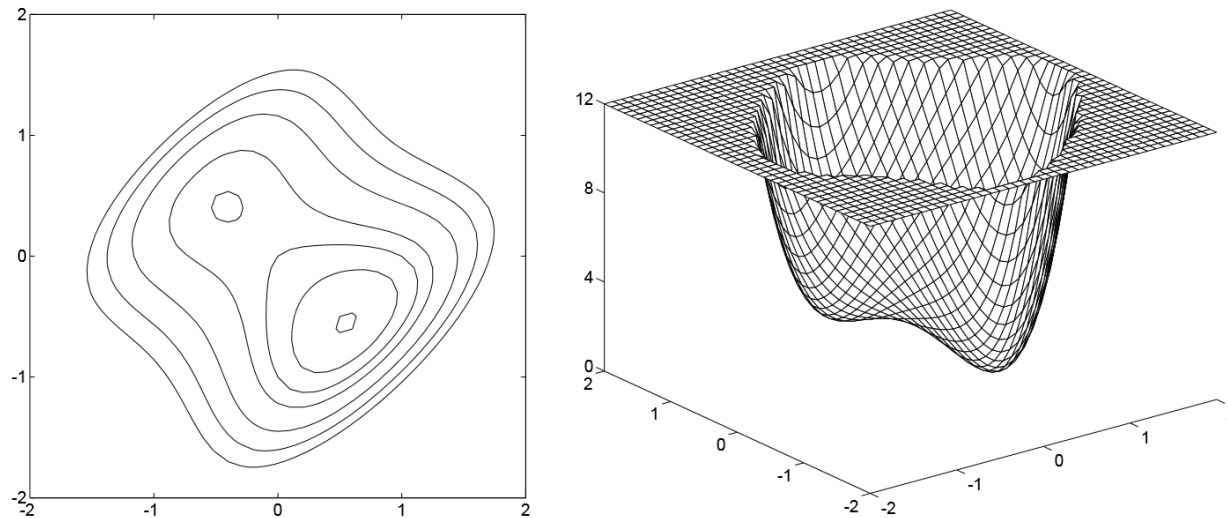
$$F(\mathbf{x}^* + \Delta \mathbf{x})$$

$$\rightarrow \nabla F(x) |_{x=x^*} = 0$$

$$\rightarrow \Delta x^T \nabla^2 F(x) |_{x=x^*} \Delta x > 0$$

minima

How can we really optimise problems?

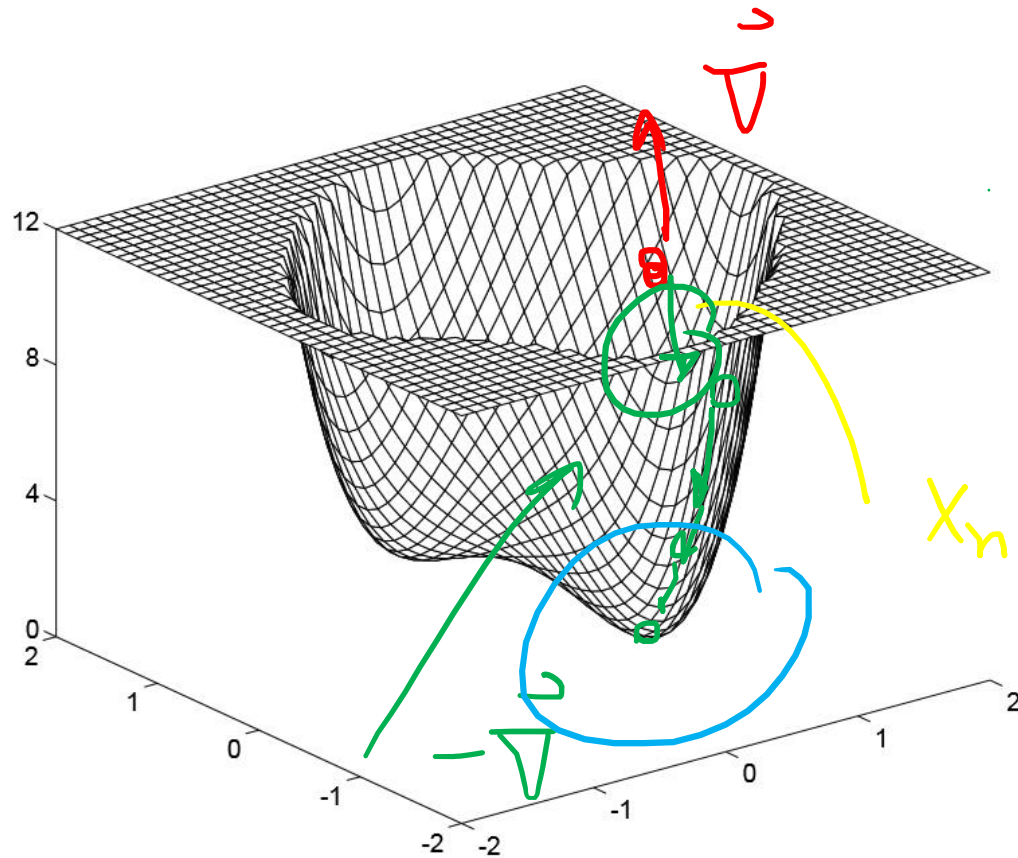


Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

- Steepest Descent Algorithm

- Motivation: $f(x_{i+1}) < f(x)$
- How can we “move” on this data efficiently?

Steepest Descent Algorithm



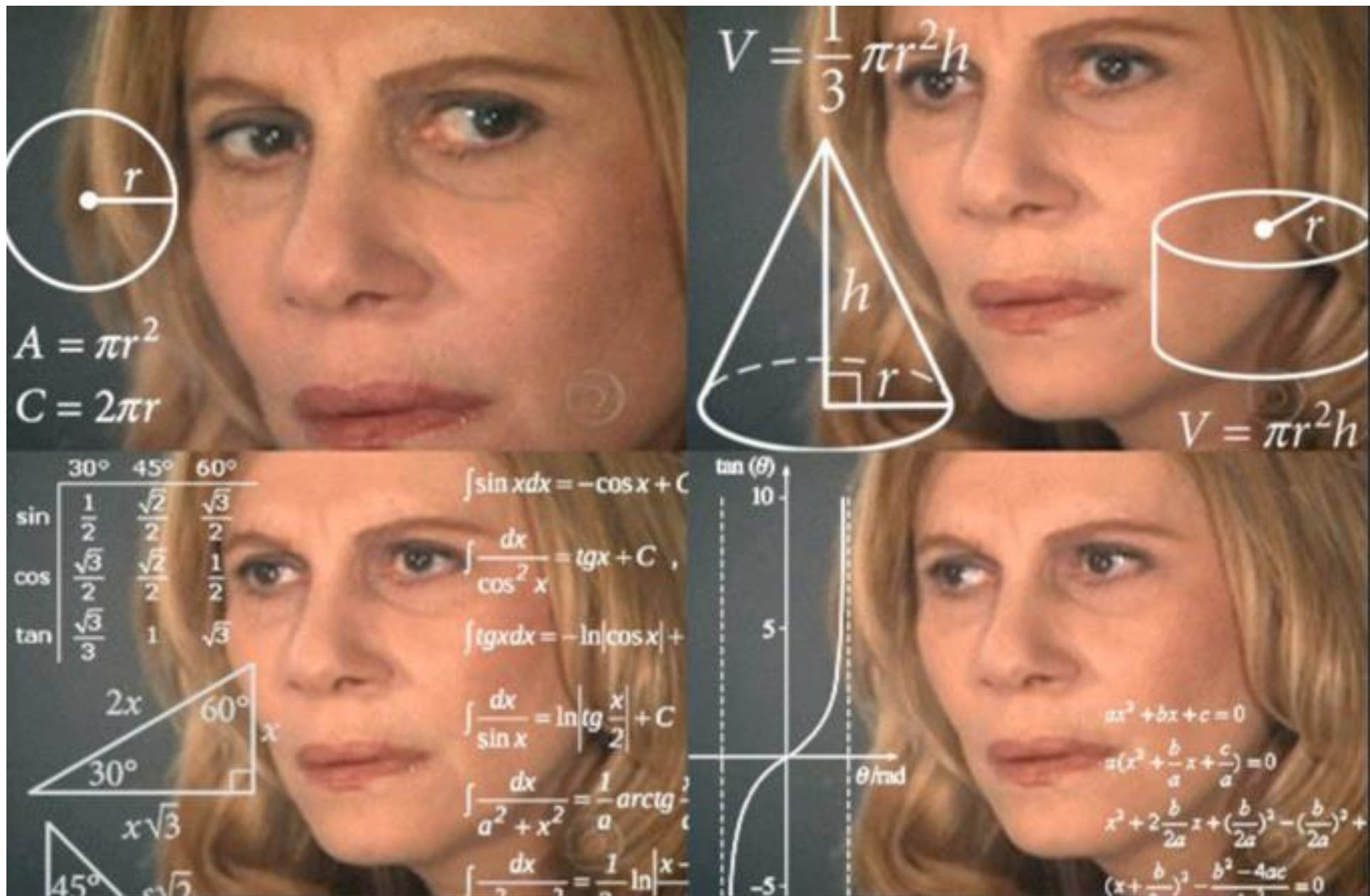
Gradient Descent

$$x_{n+1} = x - \underbrace{d}_{\text{Learning Rate}} \underbrace{\vec{\nabla} f(x)}_{*}$$

Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

Choosing the learning rate

Choosing the learning rate





Still here

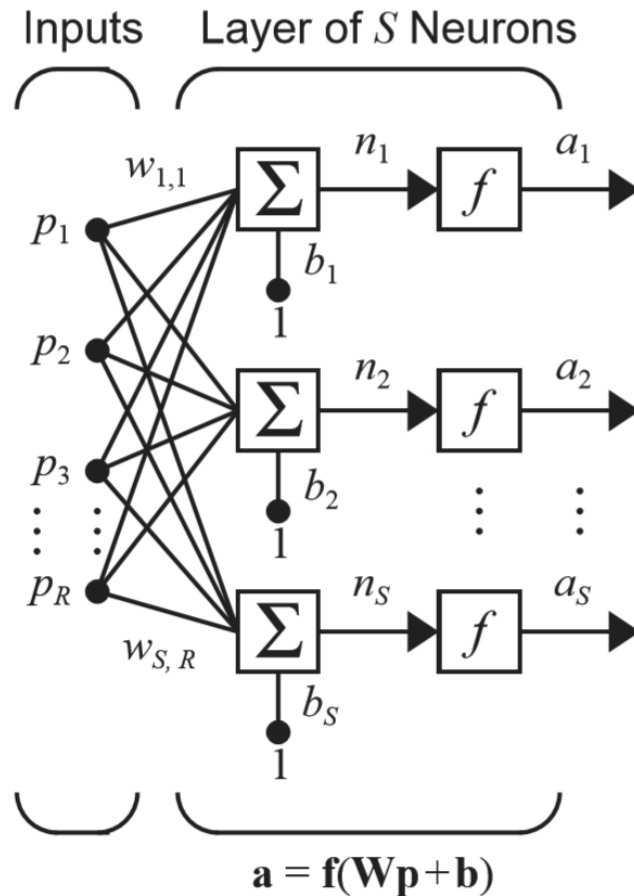


INTERMISSION

Previously on the Deep Learn Theory...

Multilayer Perceptron

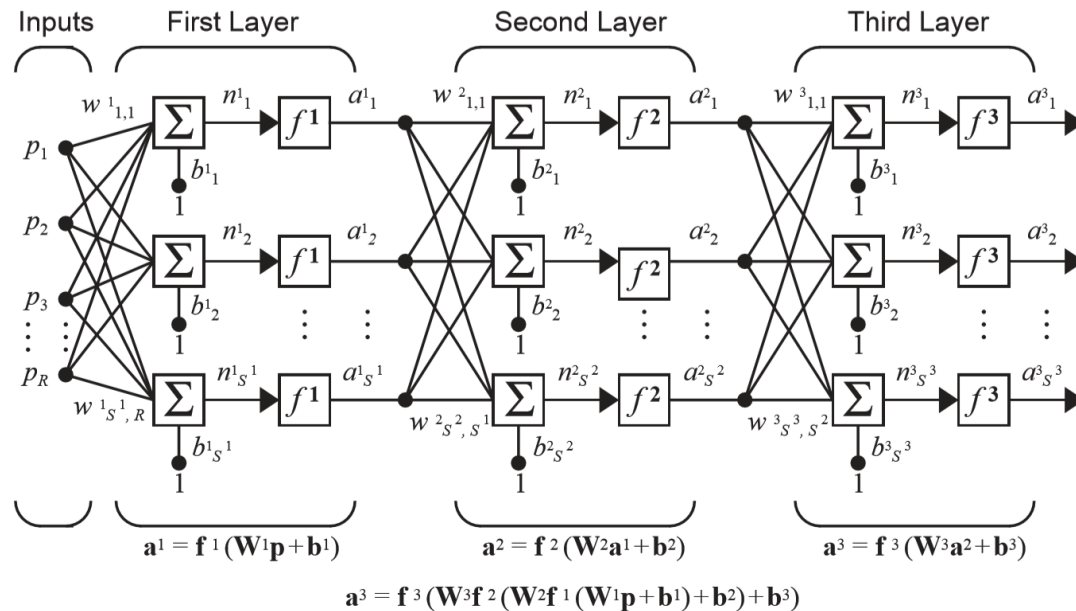
Vertically stacked Perceptron



- We can stack perceptrons together and form a *layer* of them
- These layer accepts the same input.

Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

Multilayer Perceptron

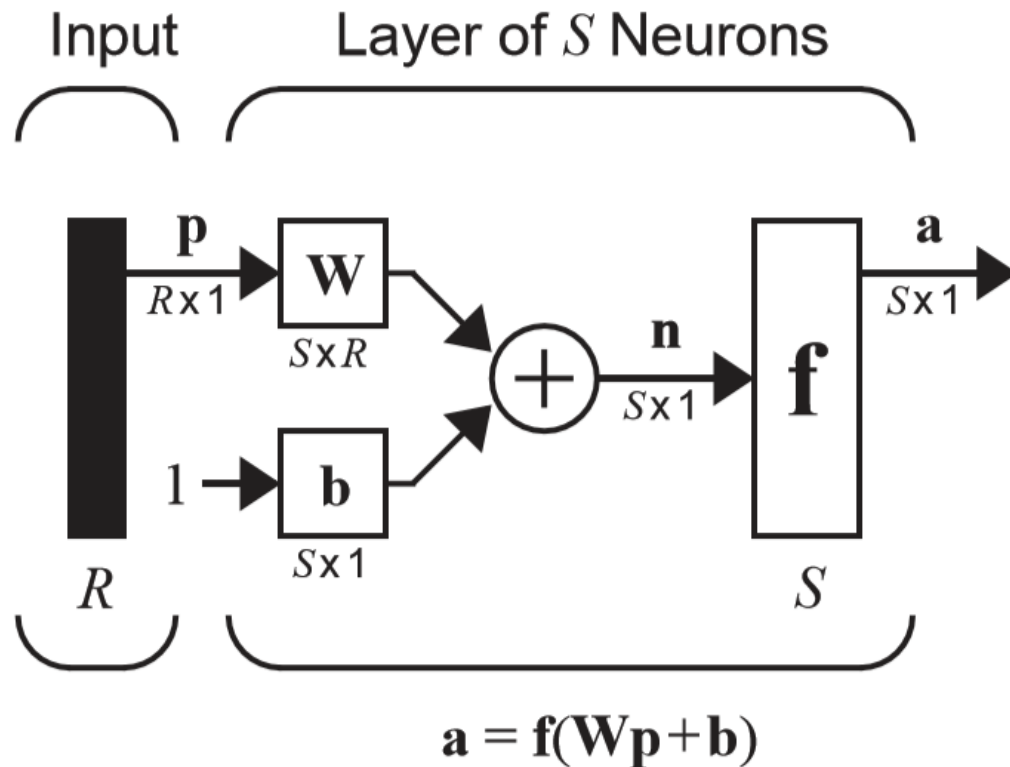


Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

- We can concatenate layers together to form a multilayer perceptron
- Stacking the perceptron in either way enhances the system's capability of being a more complex model

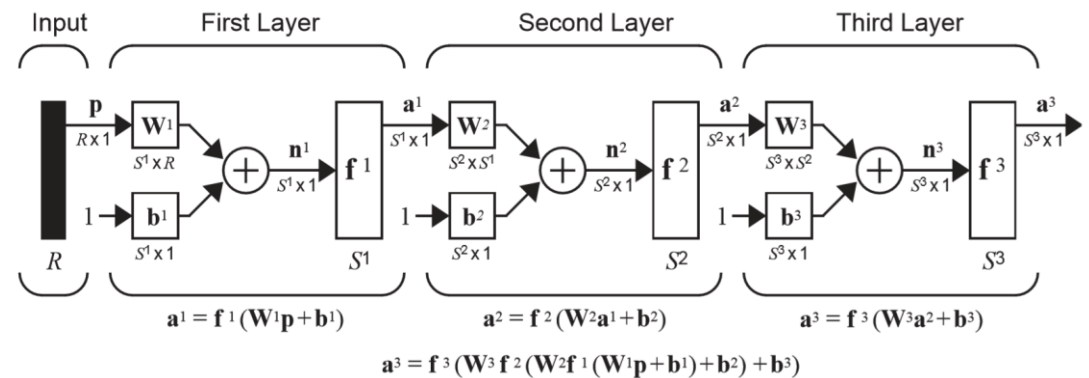
Abbreviated Notation

Vertically stacked Perceptron



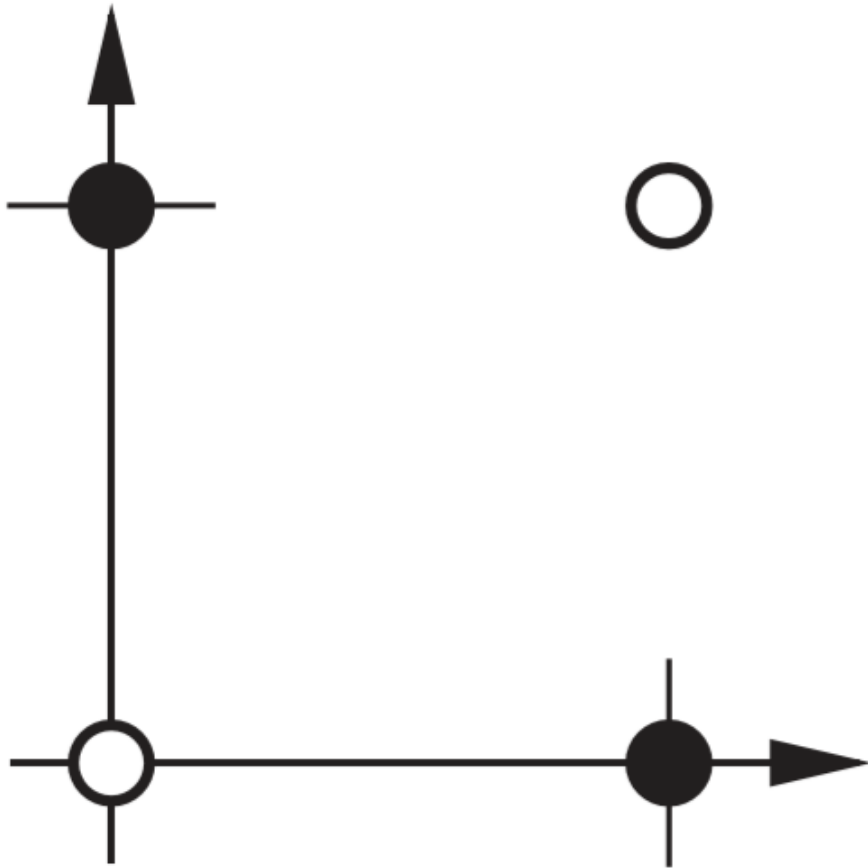
Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

Multilayer Perceptron

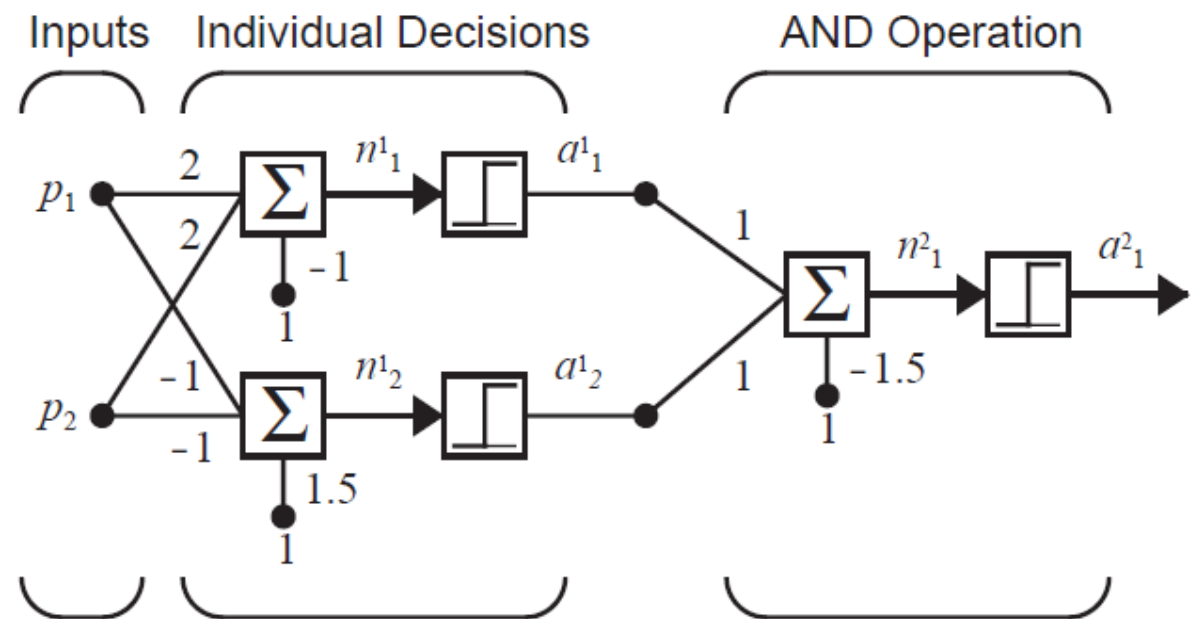
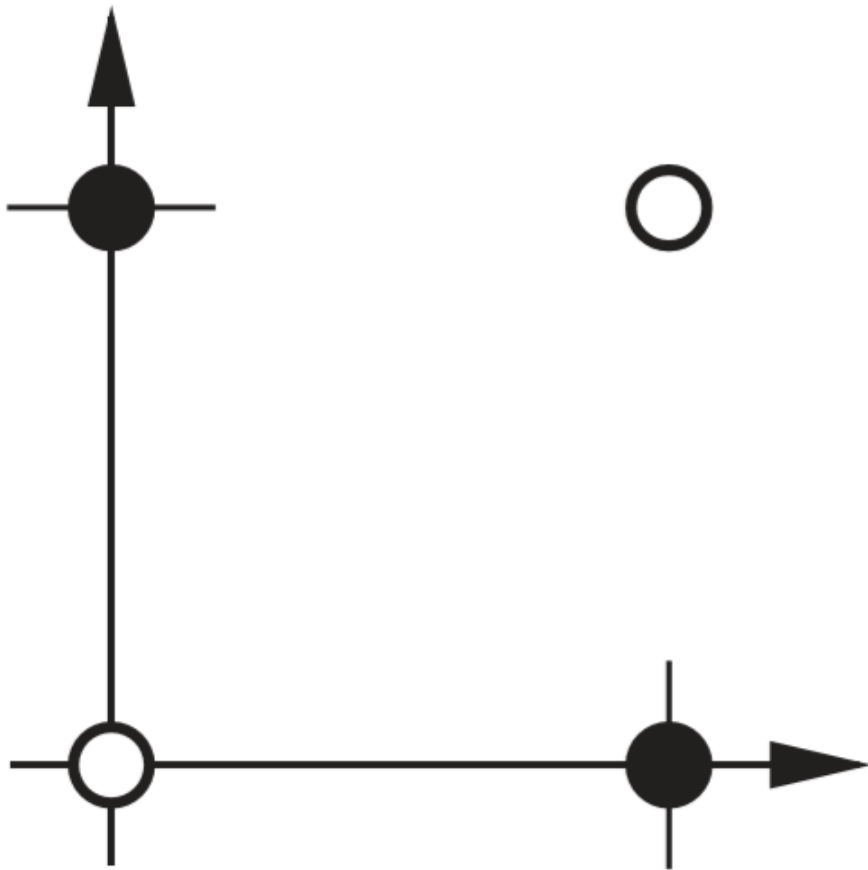


Excerpted from Neural Network Design (2nd edition)
by Hagen et. al.

XOR with MLP



XOR with MLP



Universal approximation theorem

- A feed-forward network can approximate continuous functions on compact subsets of real numbers
- The activation function must be nonlinear
 - Why?

What does that mean?

$$f\left(\boxed{\text{0}}\right) = 0$$

$$f\left(\boxed{\text{2}}\right) = 2$$

$$f\left(\boxed{\text{6}}\right) = 6$$

- We can use them to approximate weird functions like handwriting recognition and more.
- That's where the magic of Deep Learning lies!

Learning Problem's Loss Function

Regression

Classification

Practical Neural Networks

TensorFlow Playground

playground.tensorflow.org

Notebook 1: Introducing Neural Network

1 - Neural Networks - Jupyter

localhost:8888/notebooks/1 - Neural Networks.ipynb

jupyter 1 - Neural Networks (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3

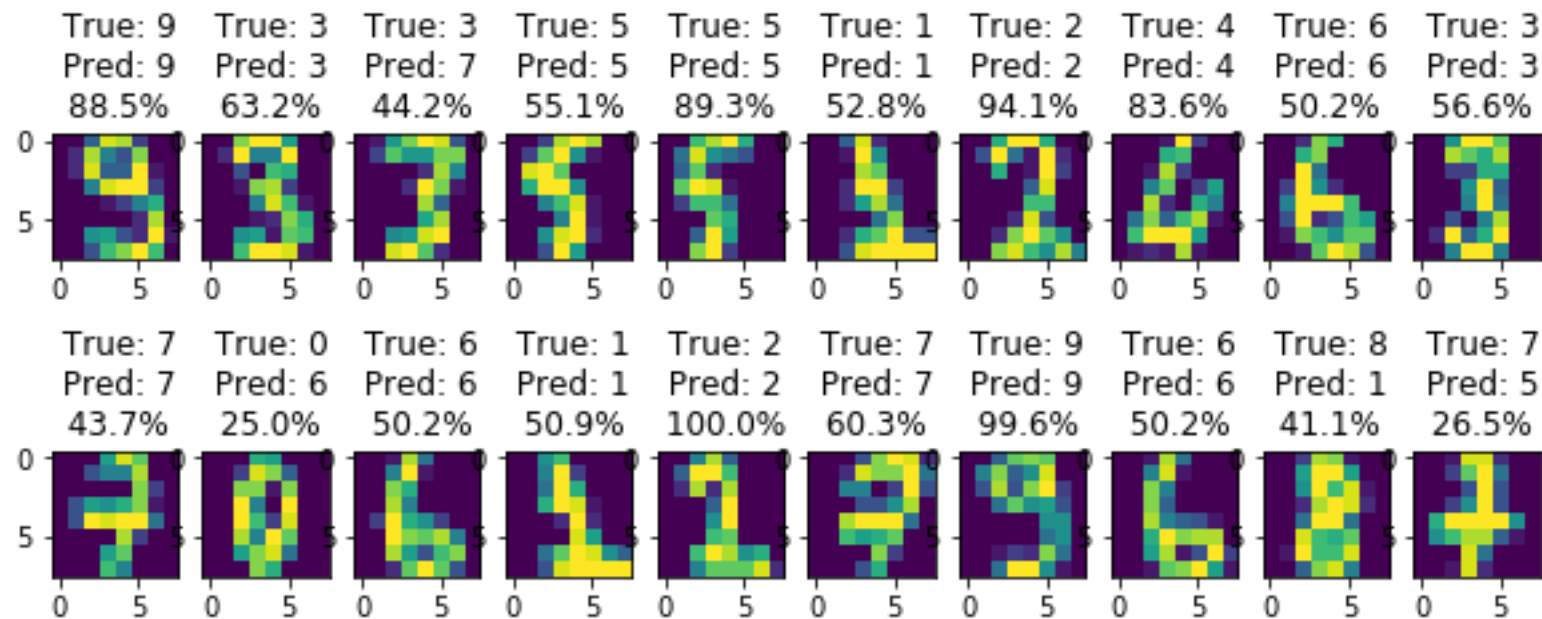
8	1033078	2	1	1	1	2	1	1
9	1033078	4	2	1	1	2	1	2

Perceptron: The smallest component

Perceptron is a smallest component of a neural network, we can simply write the perceptron model to the following components:

Observe that these are the main parts of the perceptron:

Notebook 2: Handwriting Recognition



Backpropagation

Gradient Descent Algorithm

Stochastic Gradient Descent

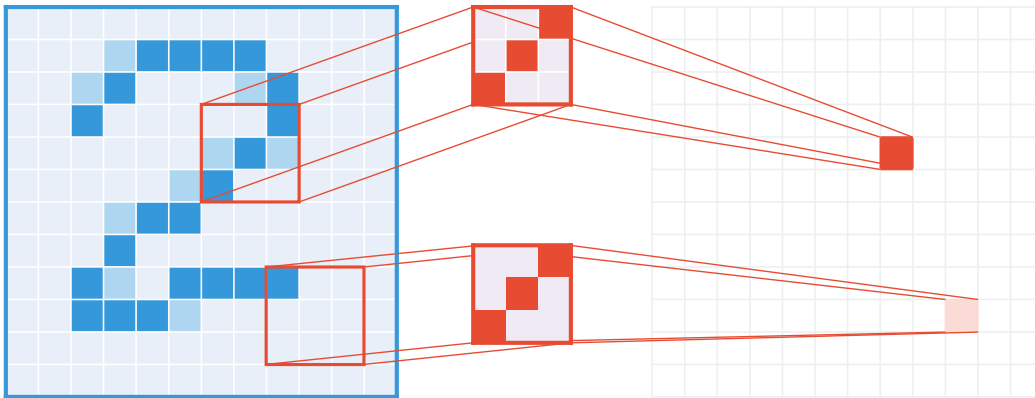
Backpropagation

Notebook 3: Manual Backpropagation

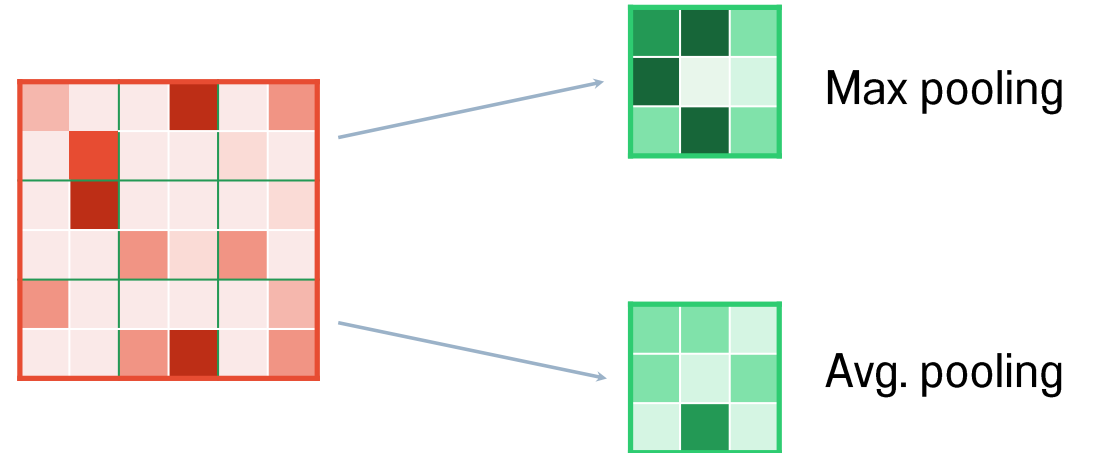
More on Deep Learning

CNNs

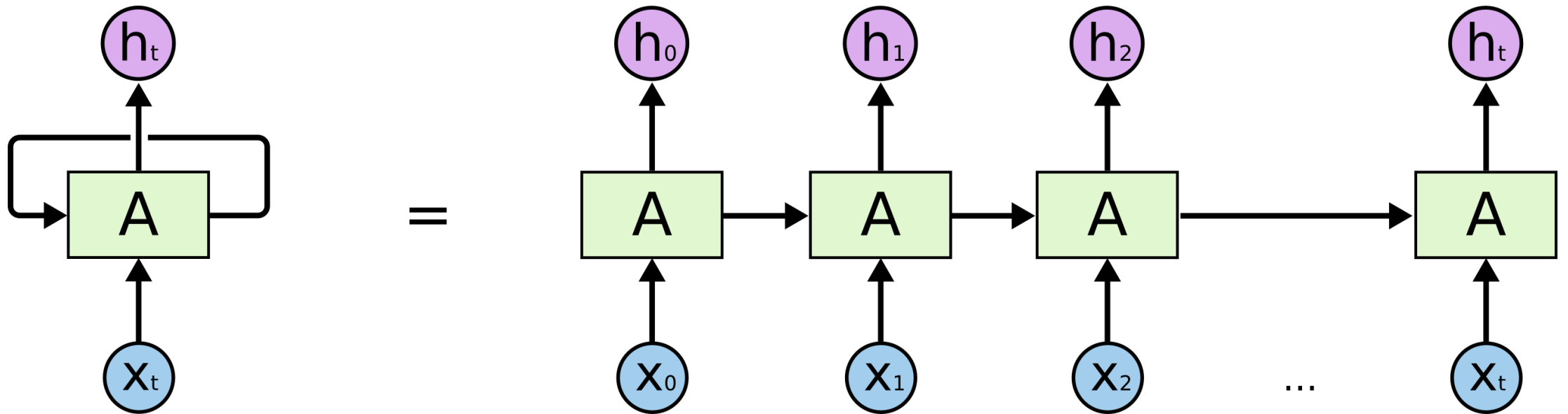
Convolution Layer



Pooling Layer

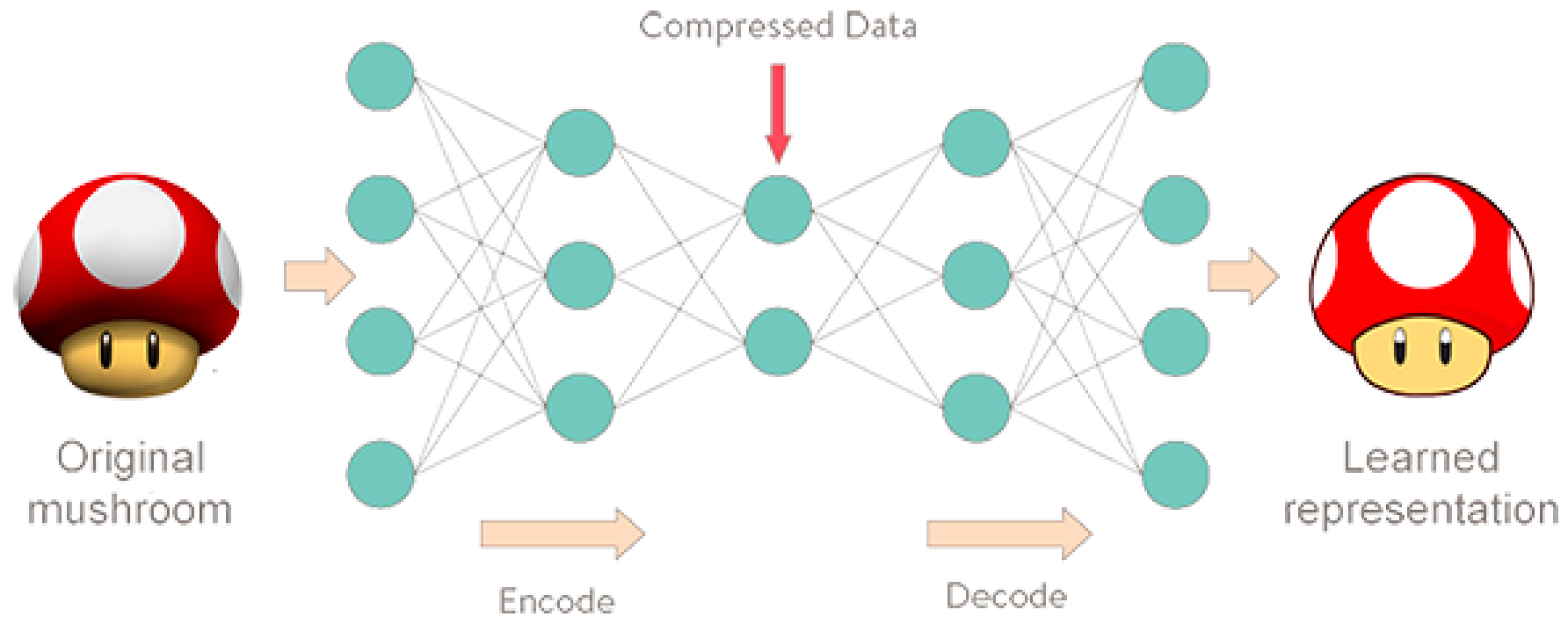


RNNs



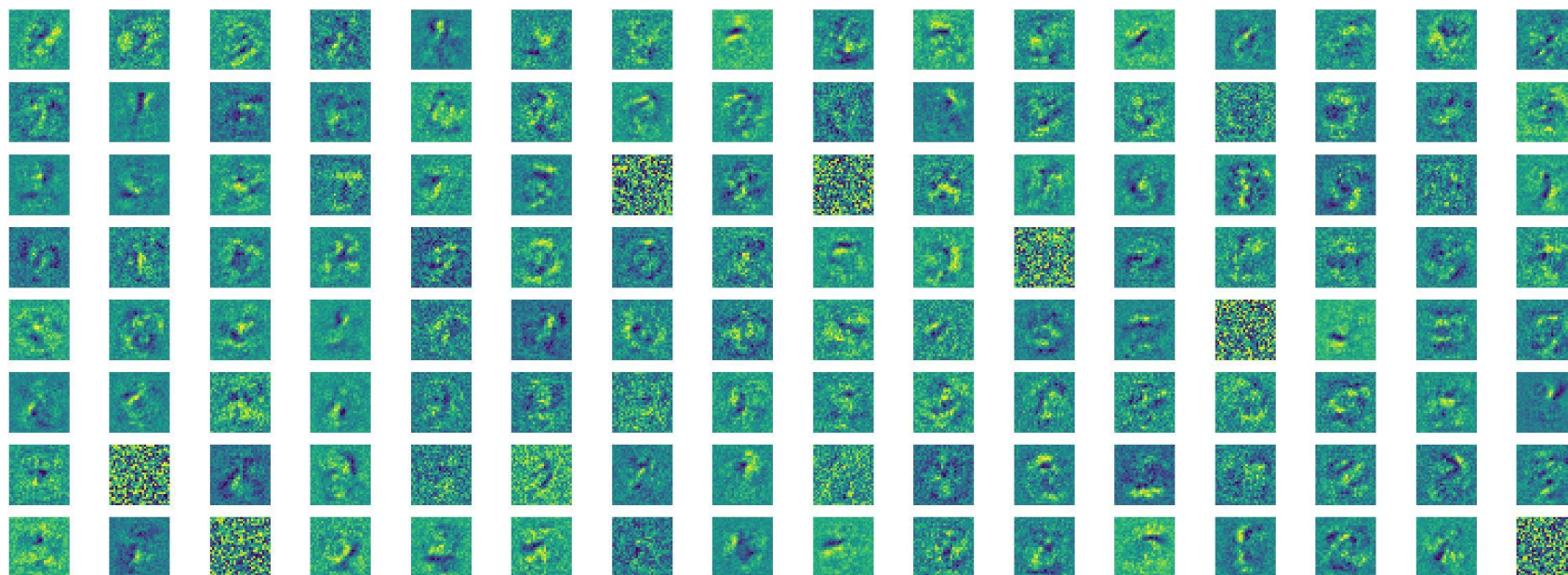
Source: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Autoencoders



Source: <https://www.pyimagesearch.com/2020/02/17/autoencoders-with-keras-tensorflow-and-deep-learning/>

Adversarial Learning





From here

to here.

Q&As

A background image of several glowing blue jellyfish swimming in a dark, deep-sea-like environment. The jellyfish are translucent with bright blue internal structures and edges, creating a ethereal and mysterious atmosphere. They are scattered across the frame, with some in sharp focus and others blurred in the background.

Deep Learning:

Theories and Practices

Sirakorn Lamyai