Introduction to Neural Networks

Introduction to Artificial Intelligence - 2023 Summer

Jul 27, 2023
Thu 4 PM

Kwangwoon University MI:RU
Artificial Intelligence Study

Agenda

In this course, you will learn

Part 1 – Quick Review of Partial Derivation and Chain Rule

Part 2 – Perceptron & Artificial Neuron Networks

Part 3 – Linear Regression & Backpropagation

Part 4 – Recurrent Neural Networks (RNN)

Quick review of Partial Derivative and Chain Rule

Partial derivative

$$f'_x,\, f_x,\, \partial_x f,\ D_x f,\, D_1 f,\, \frac{\partial}{\partial x} f,\ \text{or}\ \frac{\partial f}{\partial x}.$$

Notation for Partial derivative of variable x

Quick review of Partial Derivative and Chain Rule

Partial derivative

What we want to know

- Change rate of ONLY x

- Then, handle y as a constant

=> We can get the change rate of x ONLY!

$$f(x, y) = f_y(x) = x^2 + xy + y^2.$$

Handle y as a constant
=> Then, y will be eliminated

$$\frac{\partial f}{\partial x}(x, y) = 2x + y.$$

Quick review of Partial Derivative and Chain Rule

Chain Rule

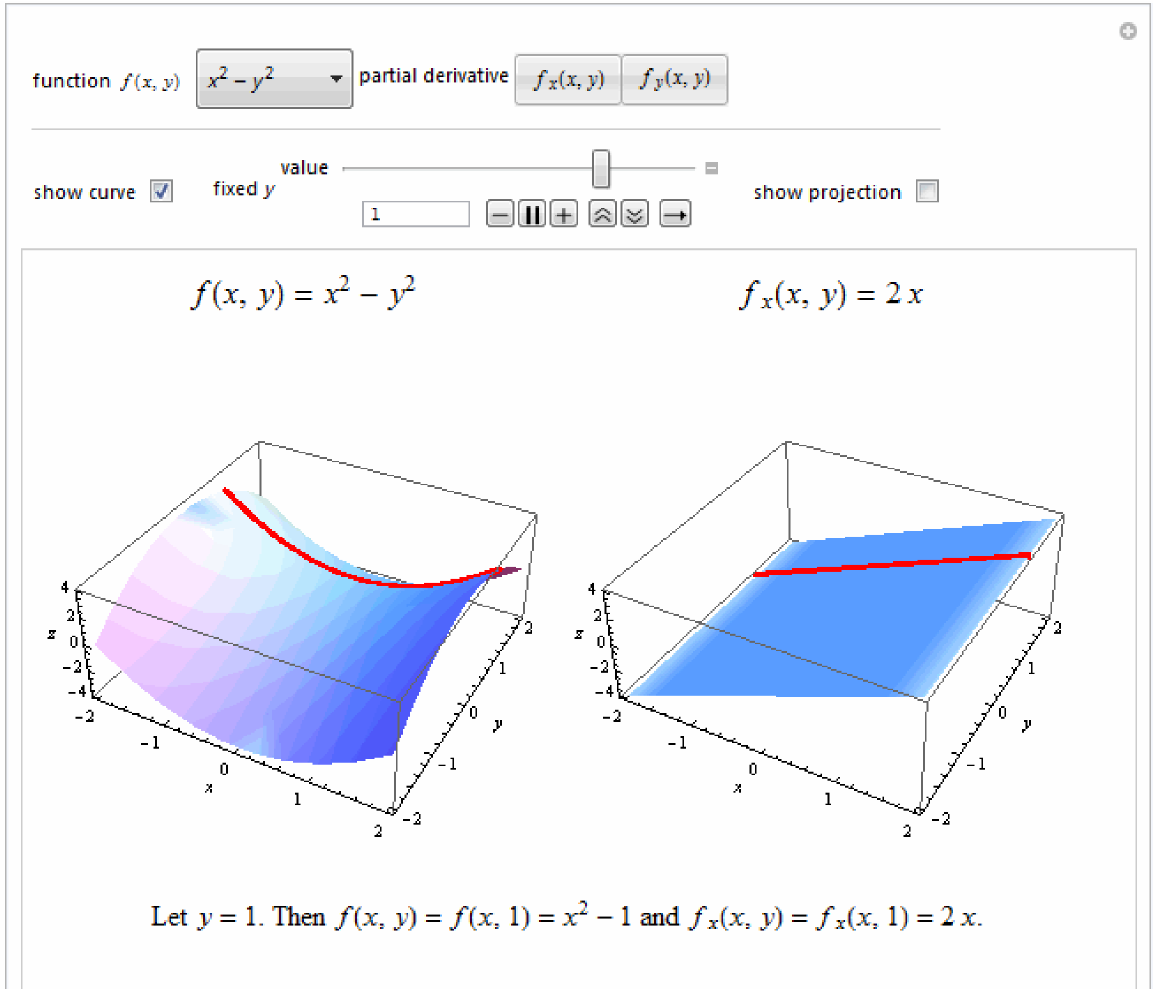
Chain rule

$$\frac{dz}{dx} = \frac{\frac{dz}{dt}}{\frac{dx}{dt}} \qquad \text{OR} \qquad \frac{dz}{dt} = \frac{dz}{dx} \cdot \frac{dx}{dt}$$

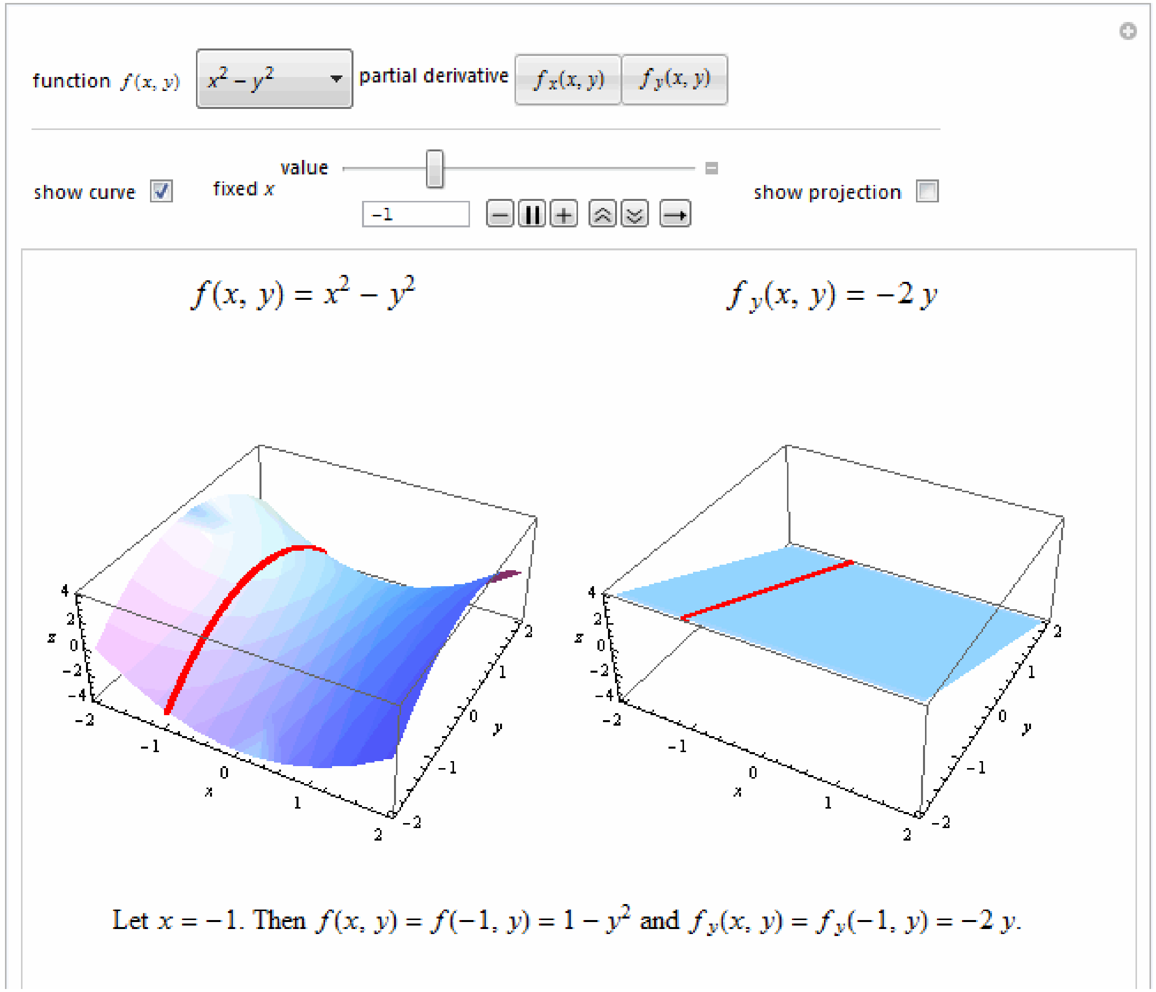
$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx} = 2 \cdot 4 = 8$$

Linear Regression & Backpropagation

Quick review of Partial derivation



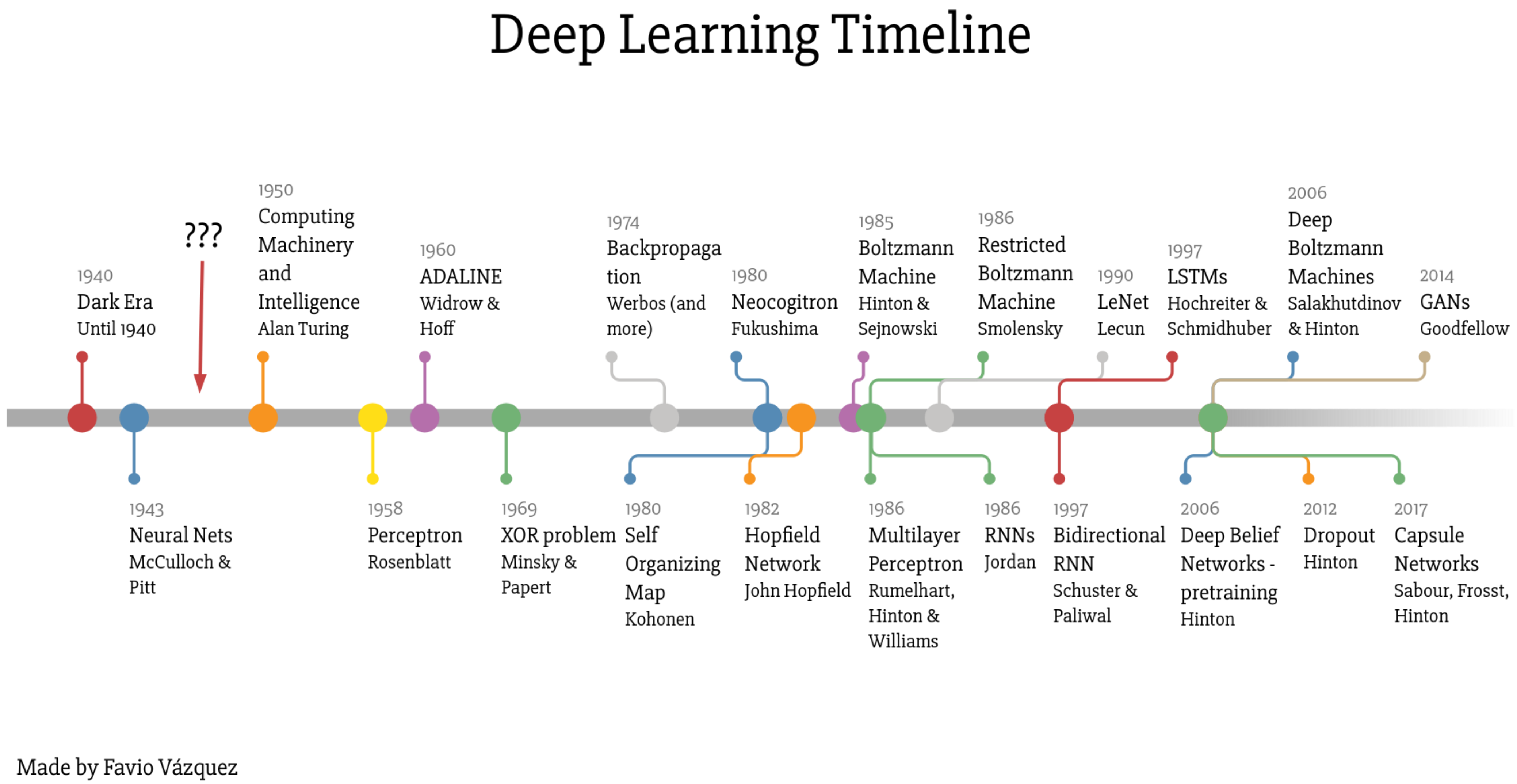
$f_x(x, y)$: y is fixed



$f_y(x, y)$: x is fixed

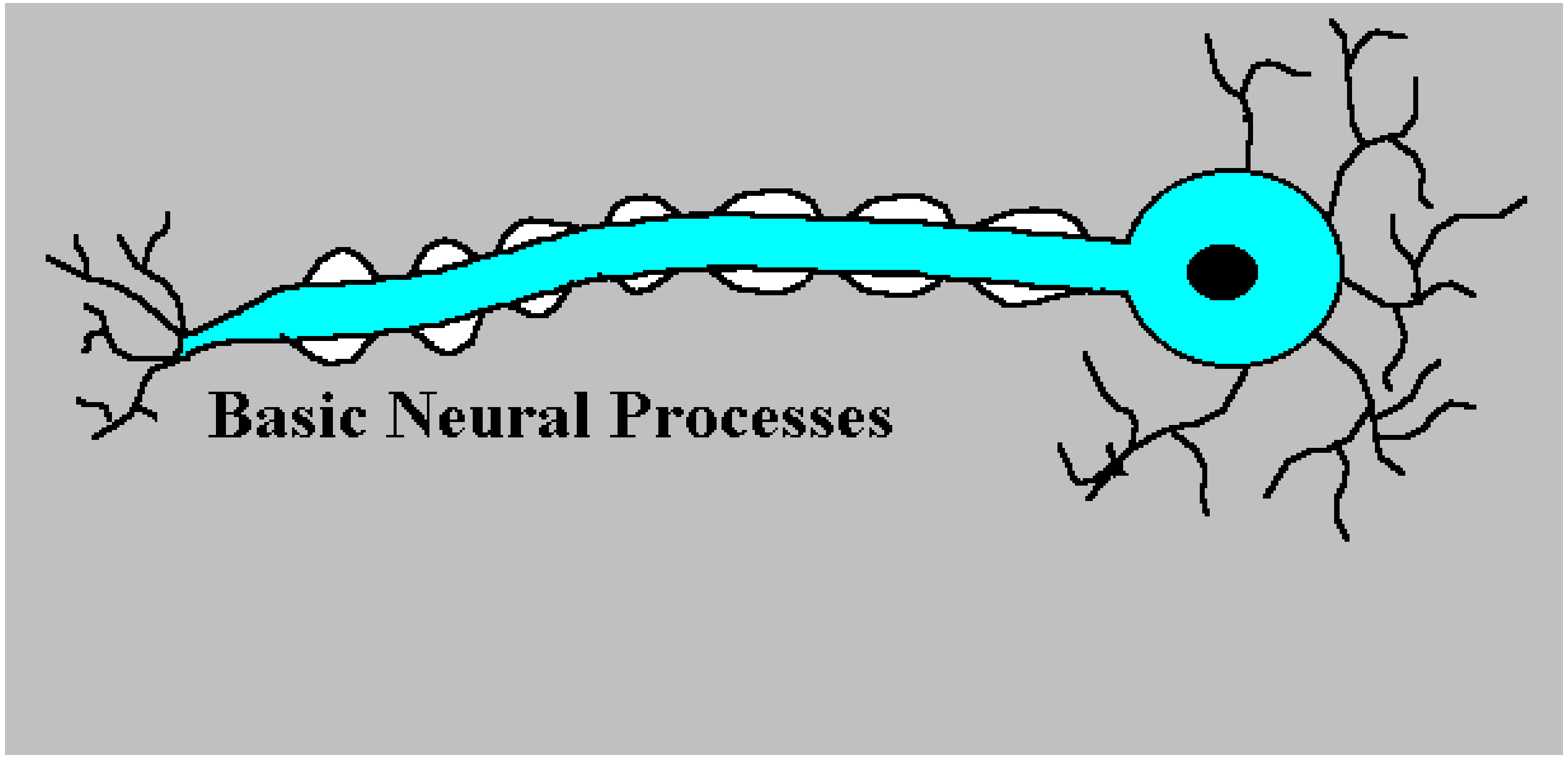
Perceptron & Artificial Neural Networks

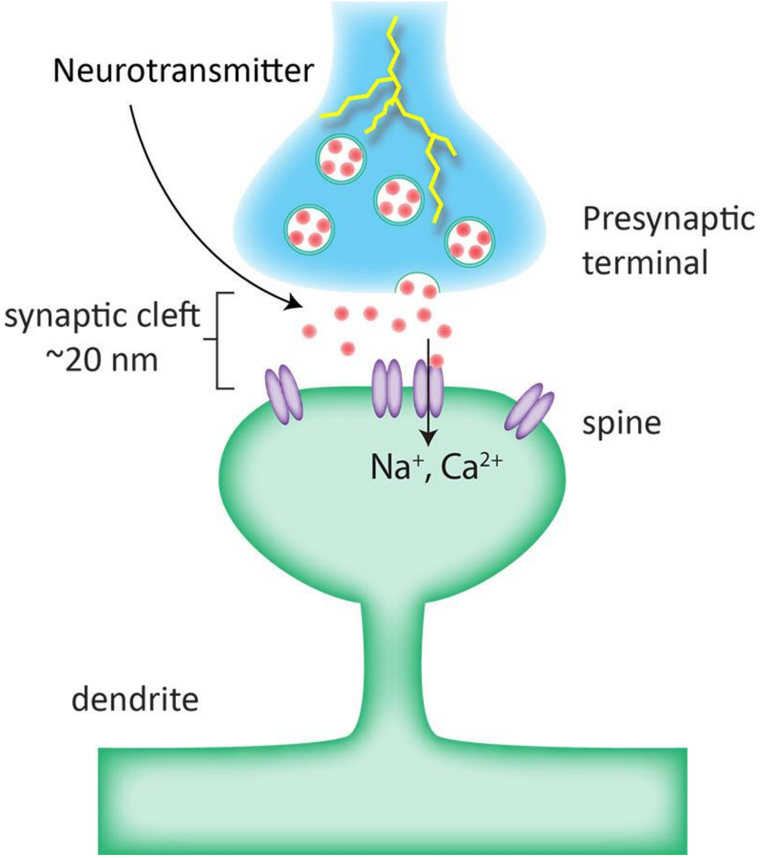
Neuron



Perceptron & Artificial Neural Networks

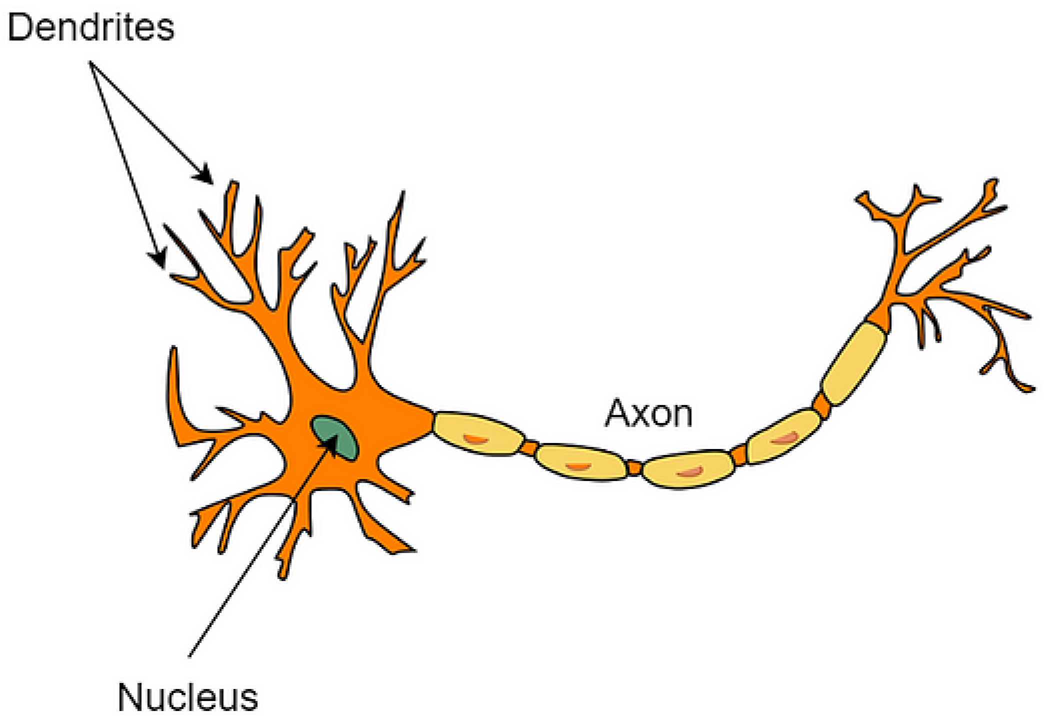
Neuron

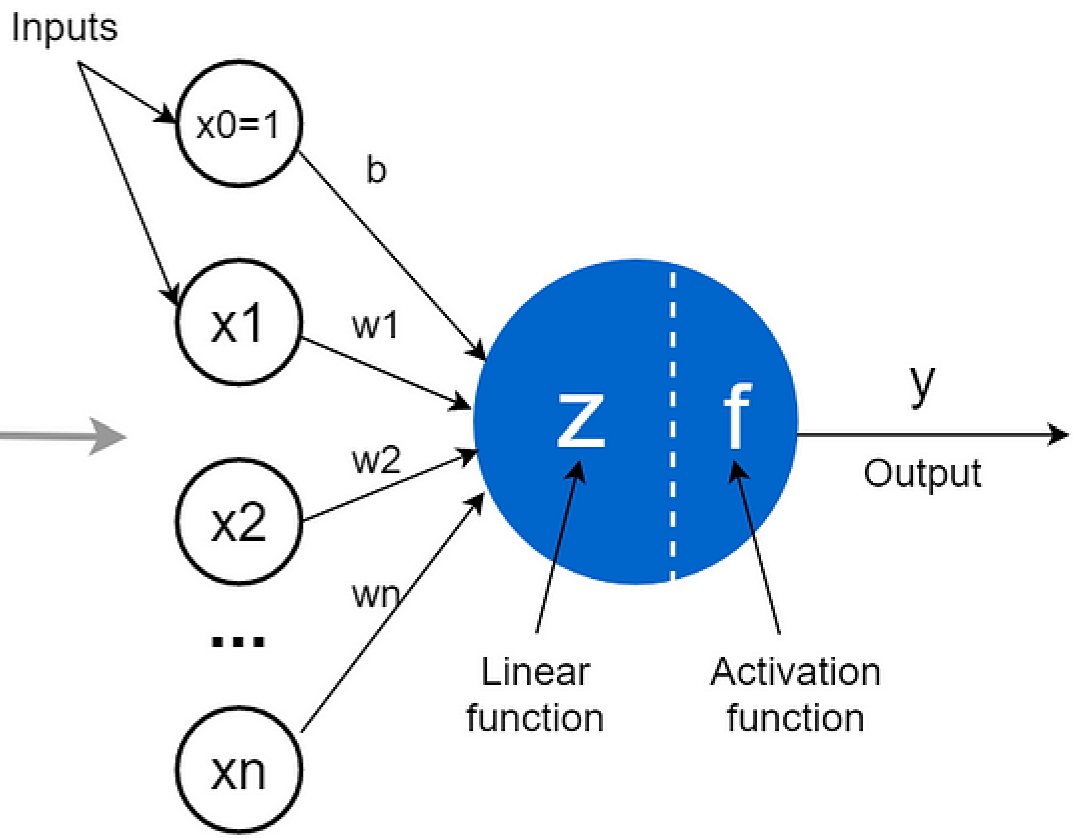




Perceptron & Artificial Neural Networks

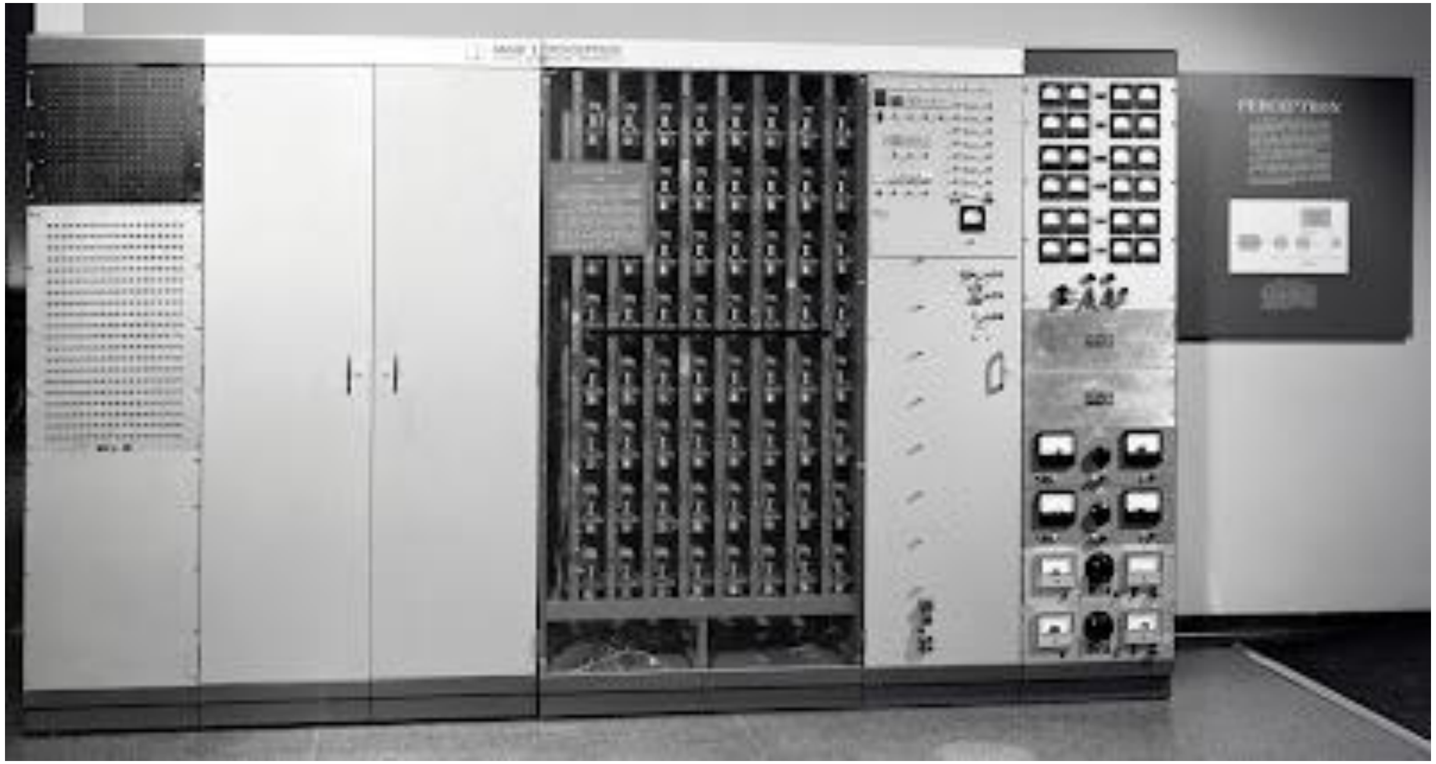
Perceptron





Perceptron & Artificial Neural Networks

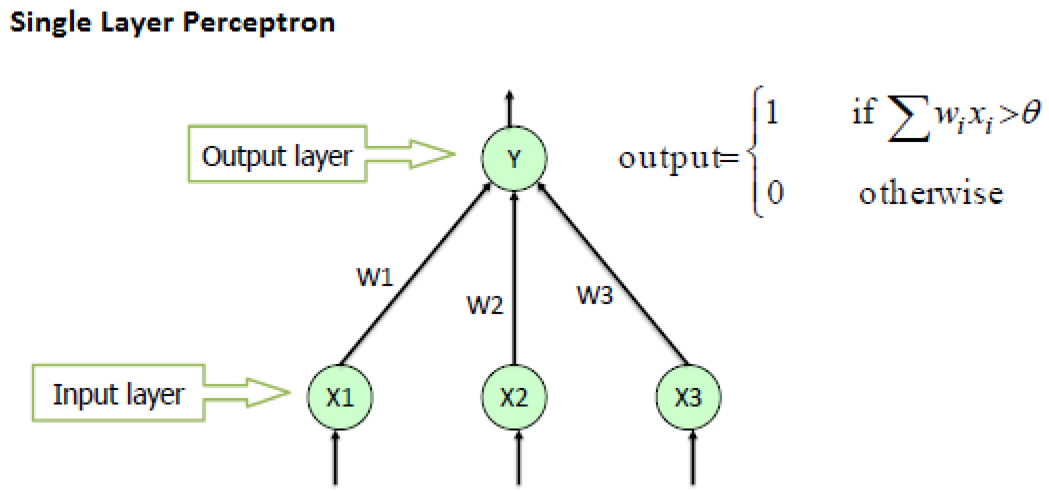
The first Perceptron machine

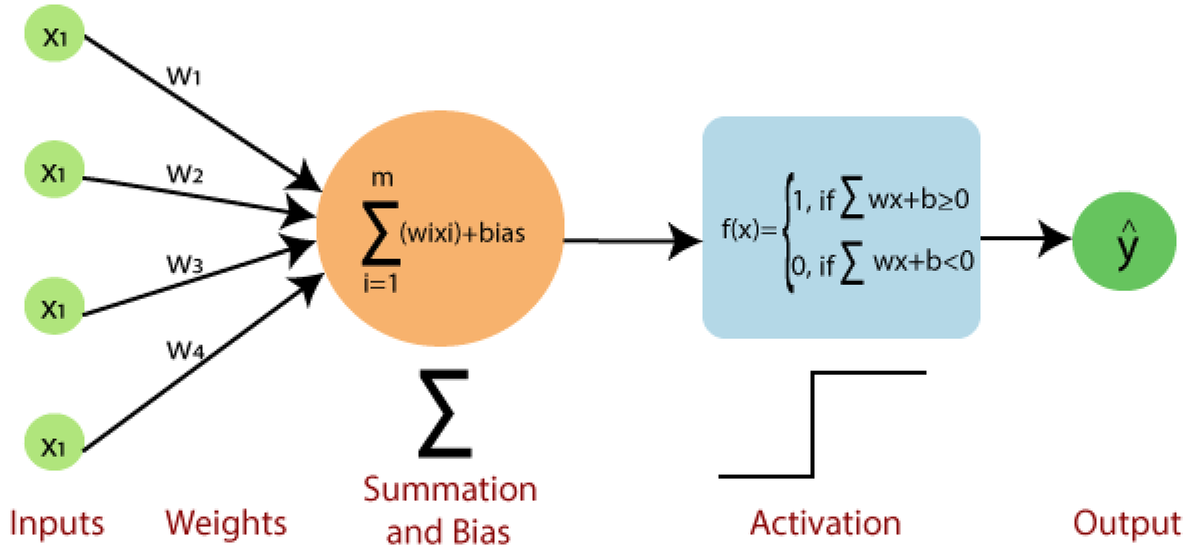


[One-Page Schoolhouse: Perceptron (ronkowitz.blogspot.com)](ronkowitz.blogspot.com)

Perceptron & Artificial Neural Networks

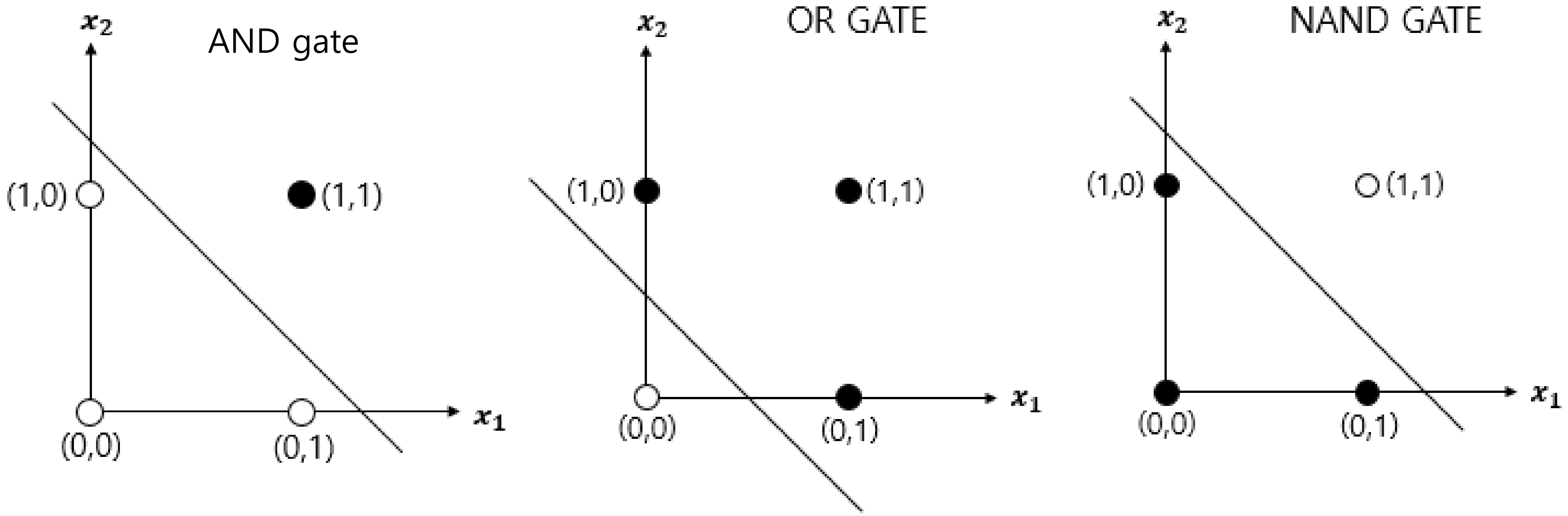
Single-Layer Perceptron





Perceptron & Artificial Neural Networks

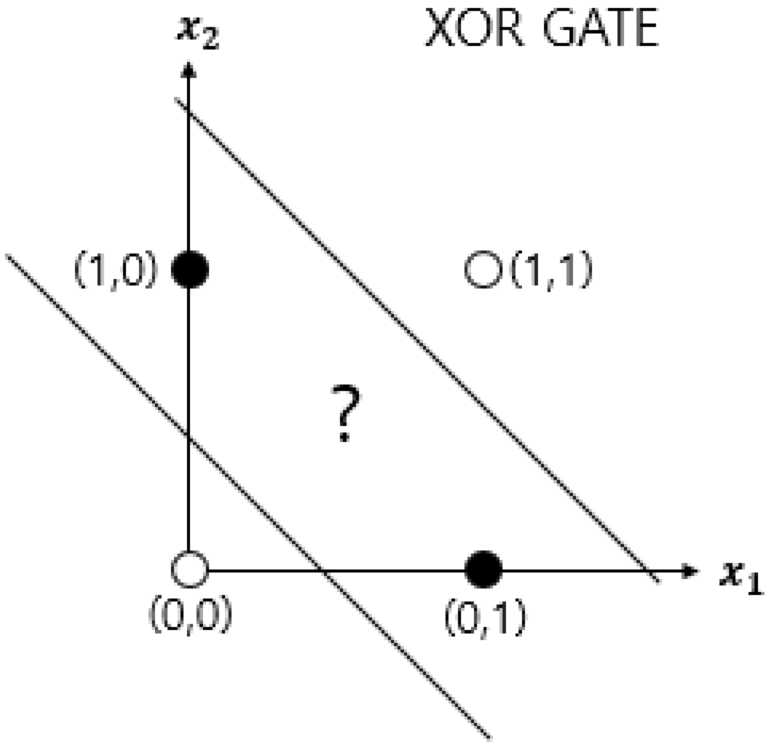
Classification using Single-Layer Perceptron
- A perspective of Logic gates



No problem occurred

Perceptron & Artificial Neural Networks

Classification using Single-Layer Perceptron
- A perspective of Logic gates

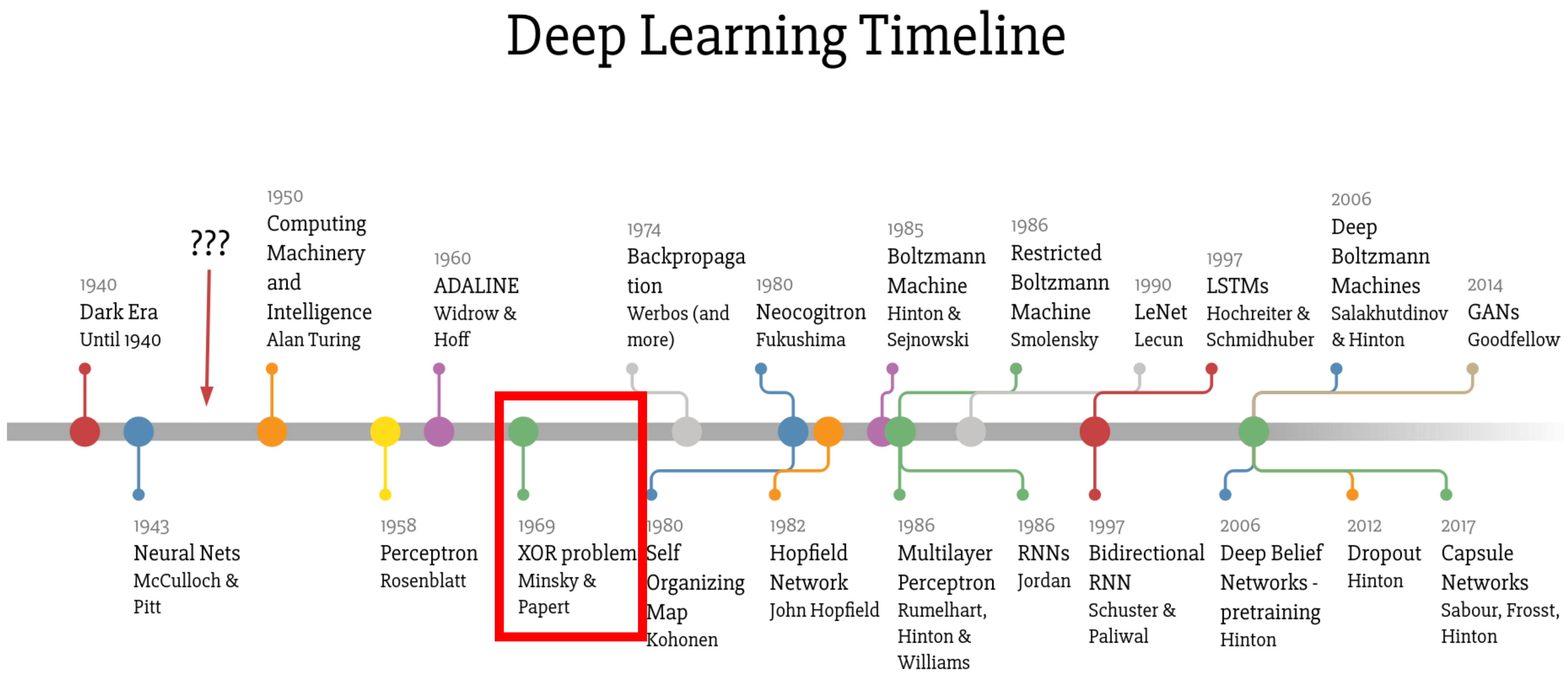


x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Problem occurred!!!

Perceptron & Artificial Neural Networks

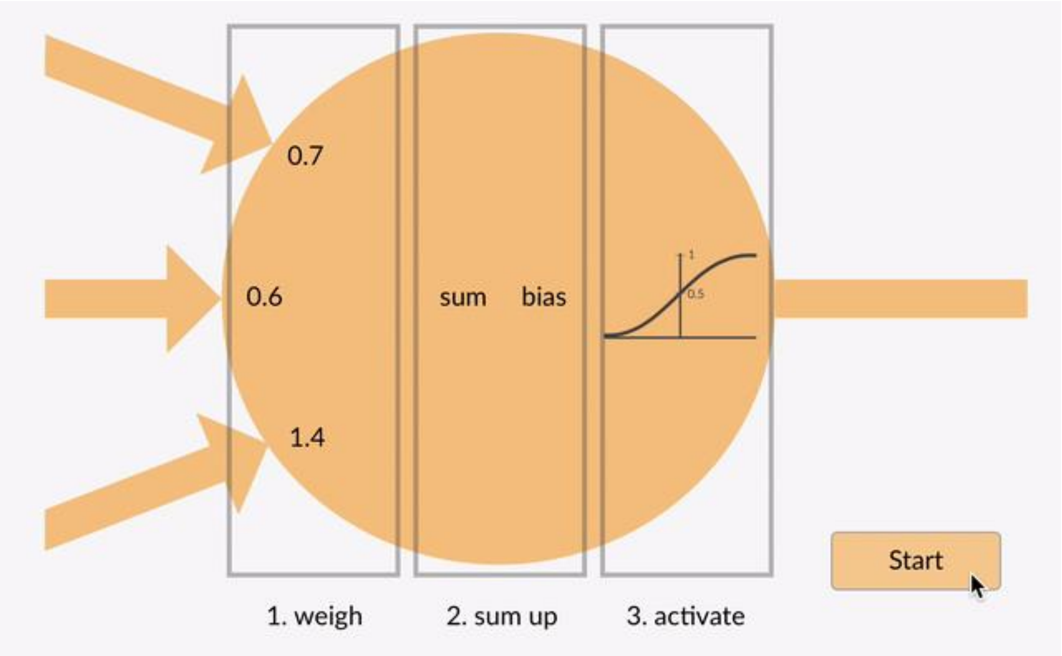
Neuron

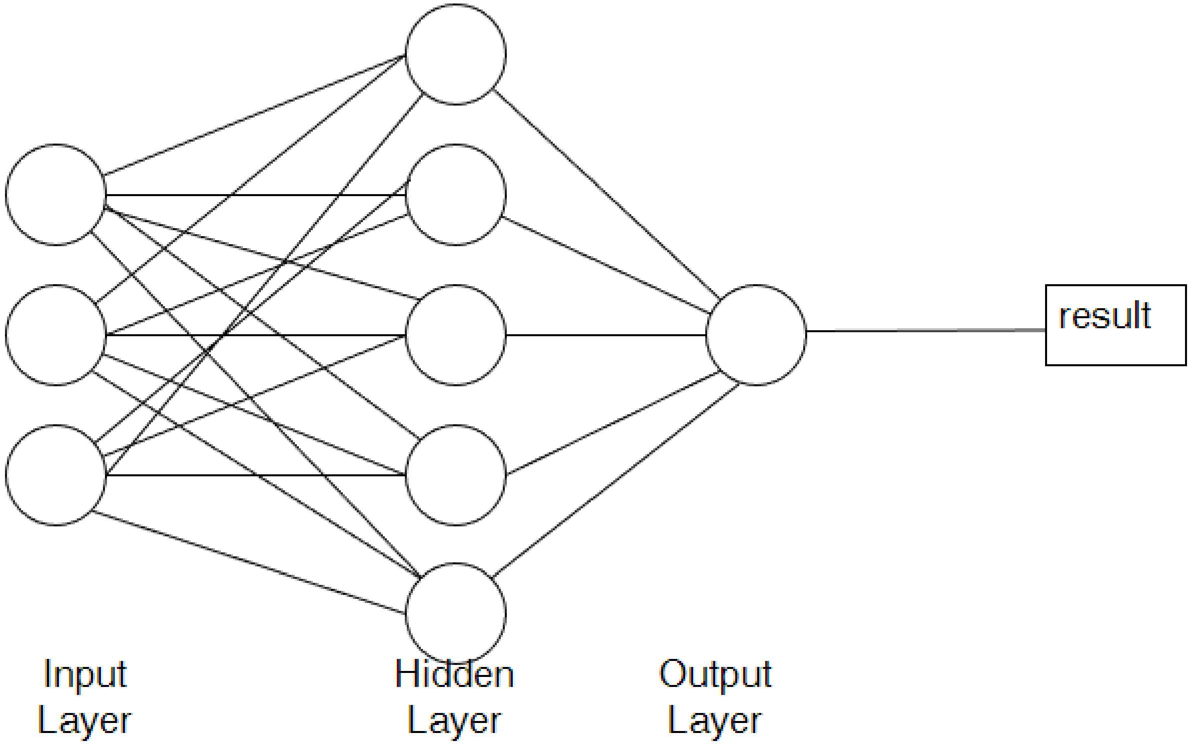


Made by Favio Vázquez

Perceptron & Artificial Neural Networks

Multi-Layer Perceptron (MLP)





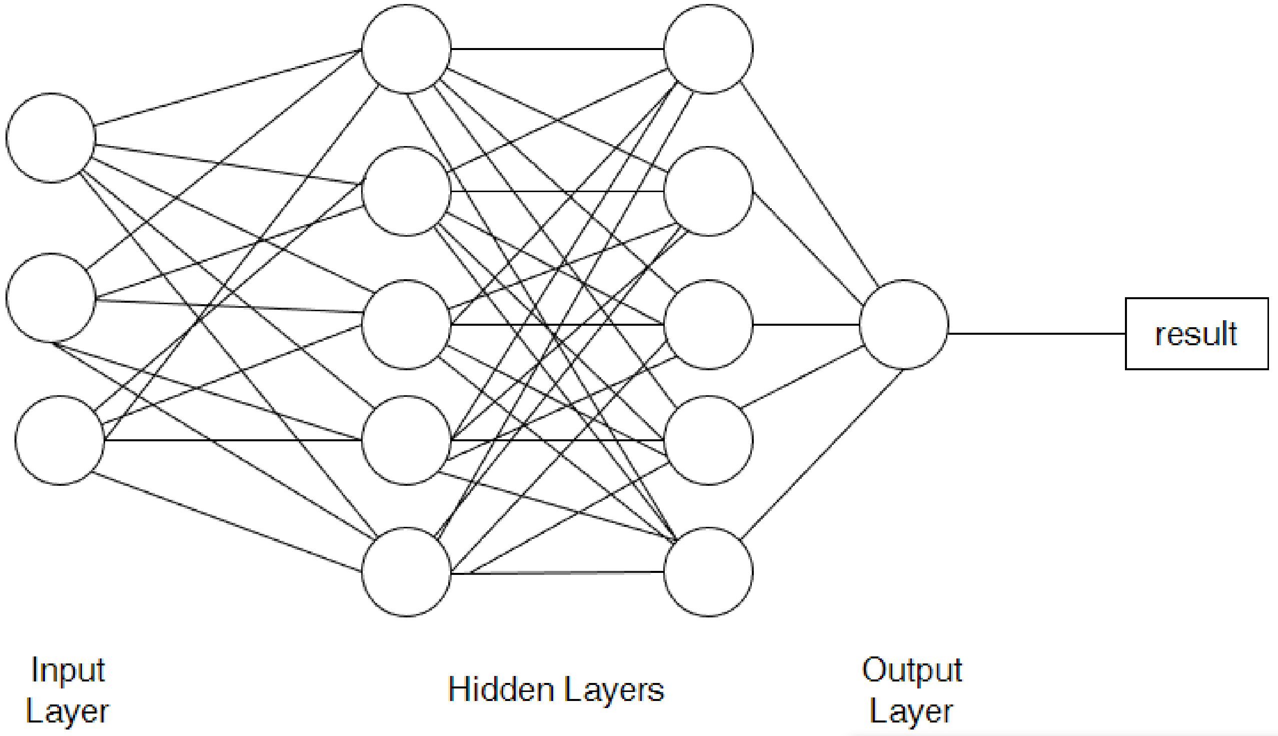
Multi-Layer Perceptron

==

Sequentially connected Single Layer Perceptron

Perceptron & Artificial Neural Networks

Deep Learning?



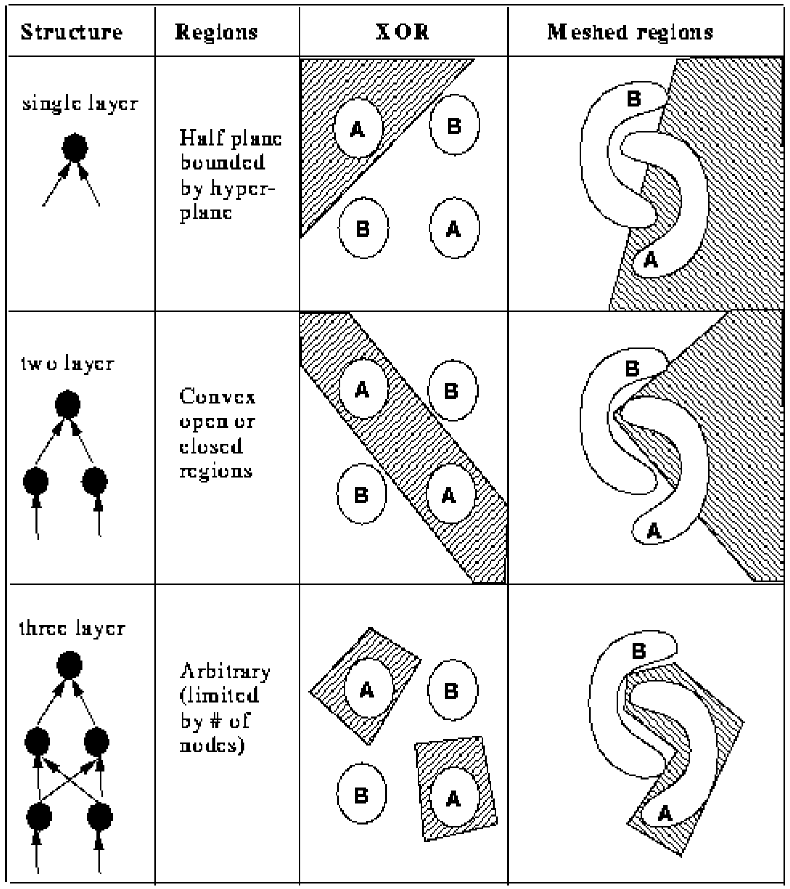
Deep Learning

==

Training a Deep Neural Network

Perceptron & Artificial Neural Networks

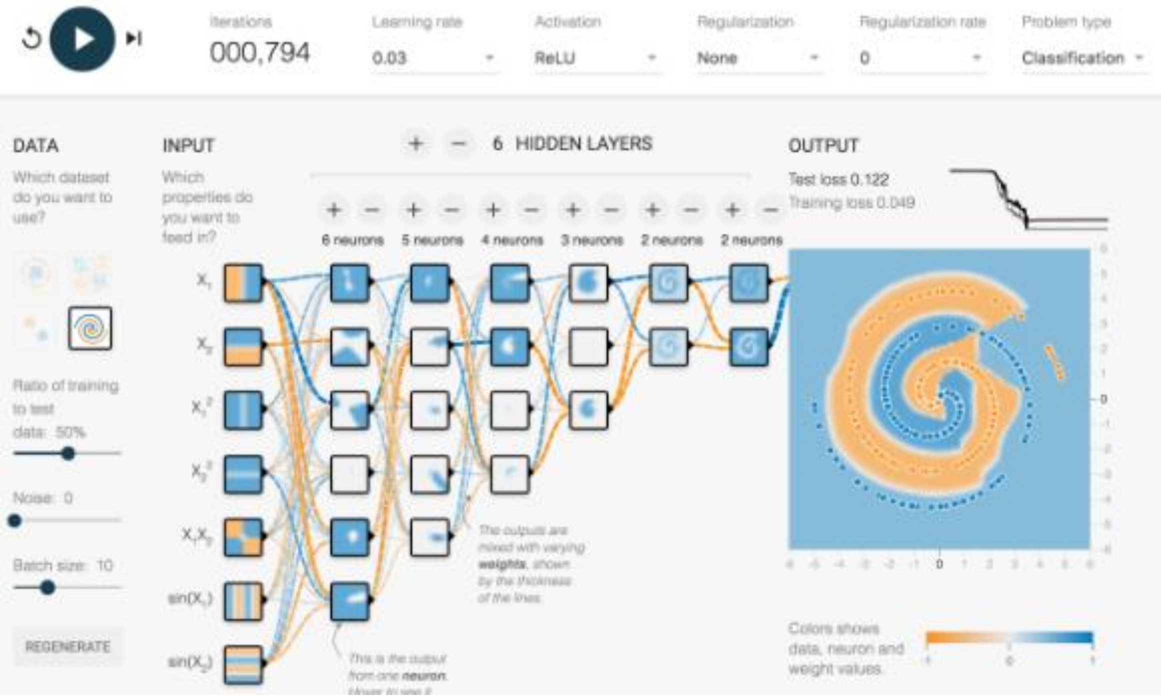
Multi-Layer Perceptron (MLP)



Massey University of New Zealand
Artificial Intelligence – Lecture 7 (The XOR problem)

Perceptron & Artificial Neural Networks

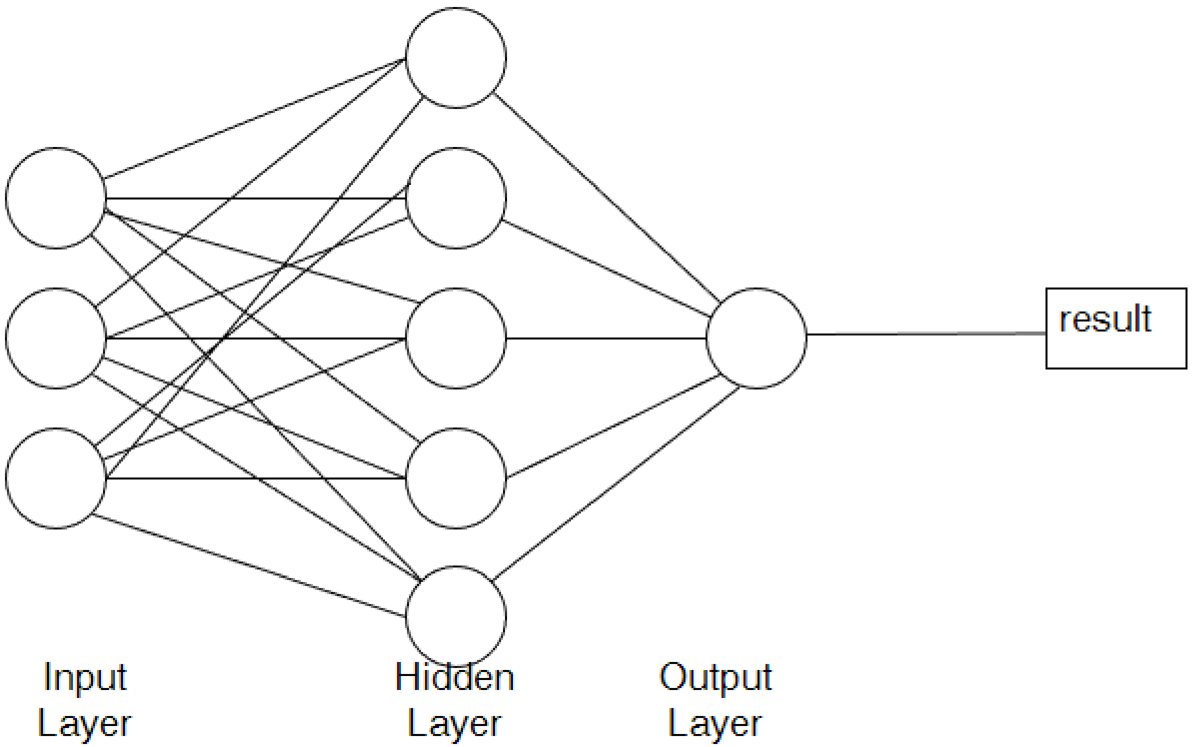
Deep Neural Network

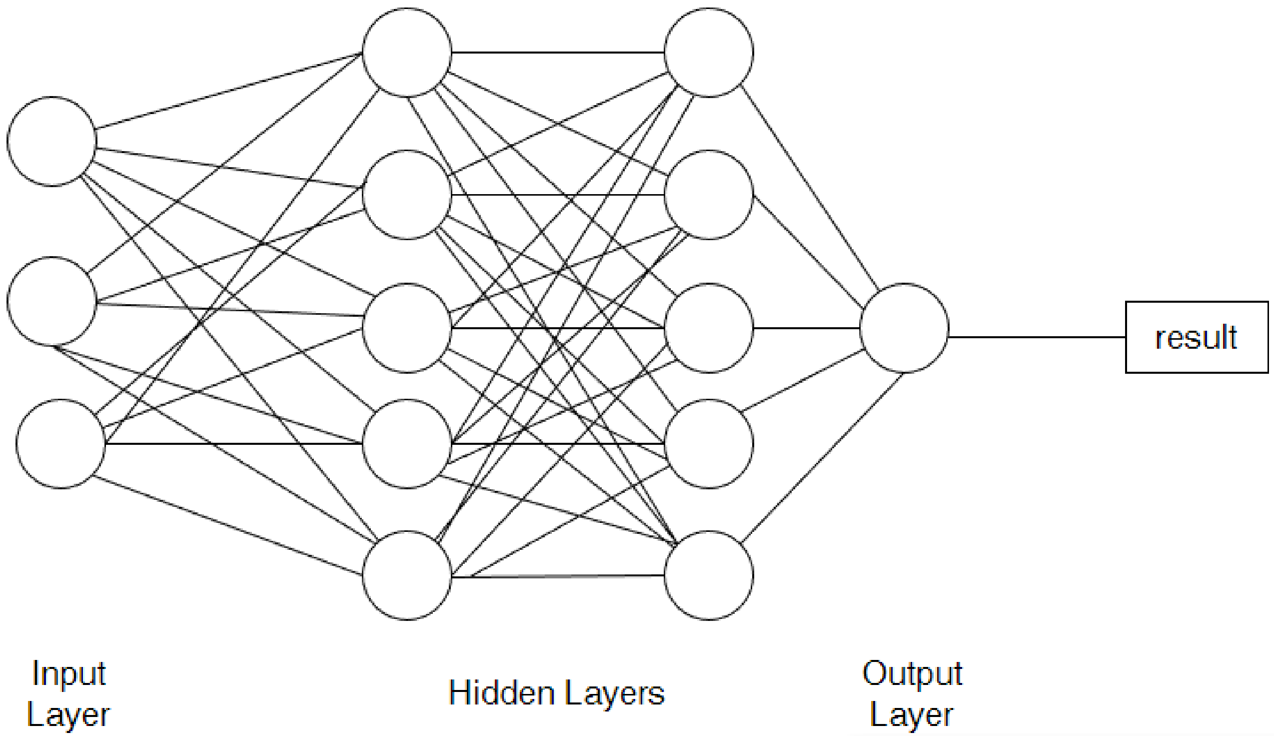


[A Neural Network Playground (tensorflow.org)](https://playground.tensorflow.org)

Perceptron & Artificial Neural Networks

Deep Neural Network (DNN)





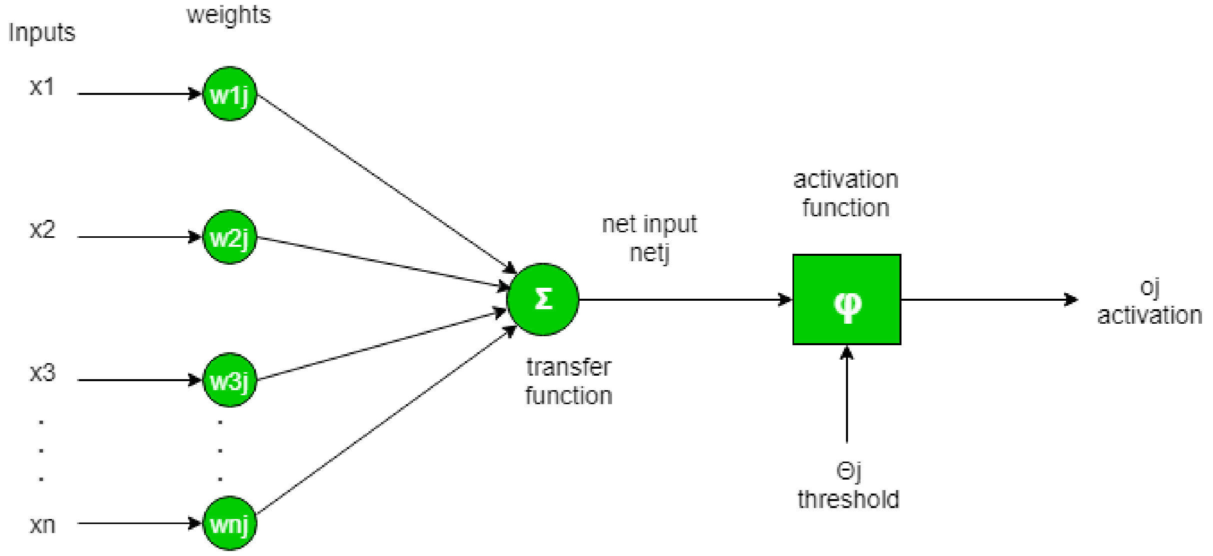
Sequentially connected Multi-Layer Perceptron

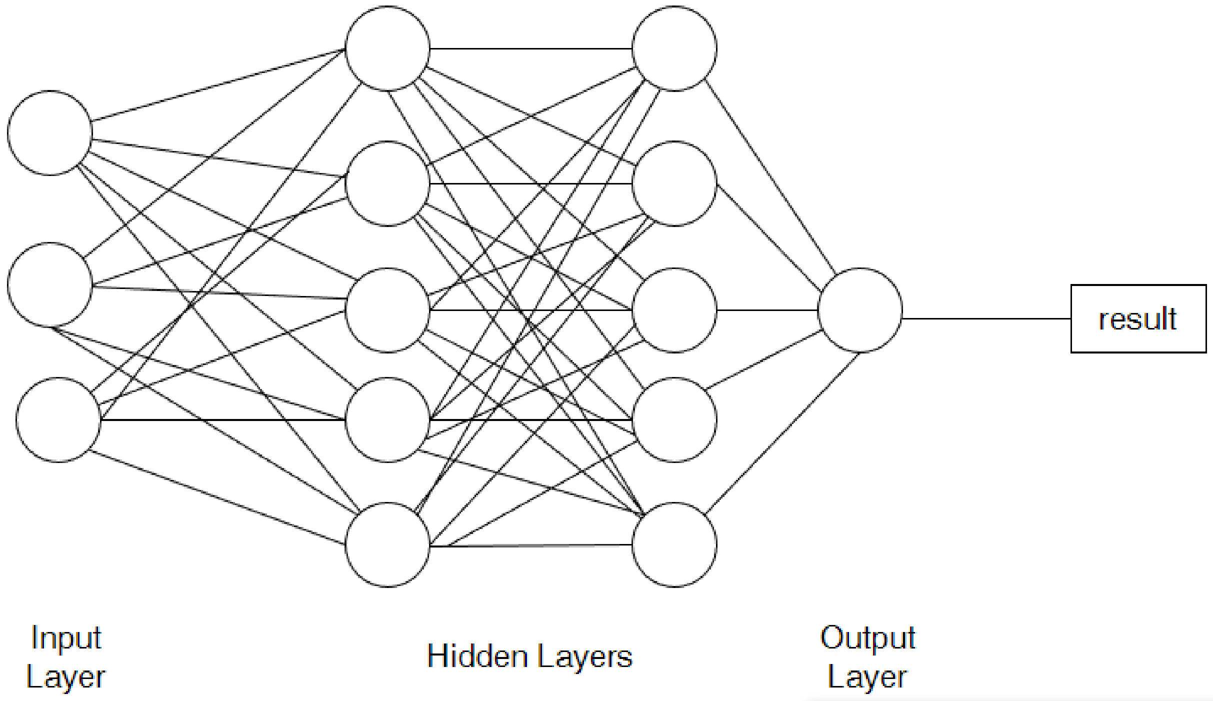
==

Deep Neural Network (DNN)

Perceptron & Artificial Neural Networks

Deep Neural Network (DNN)

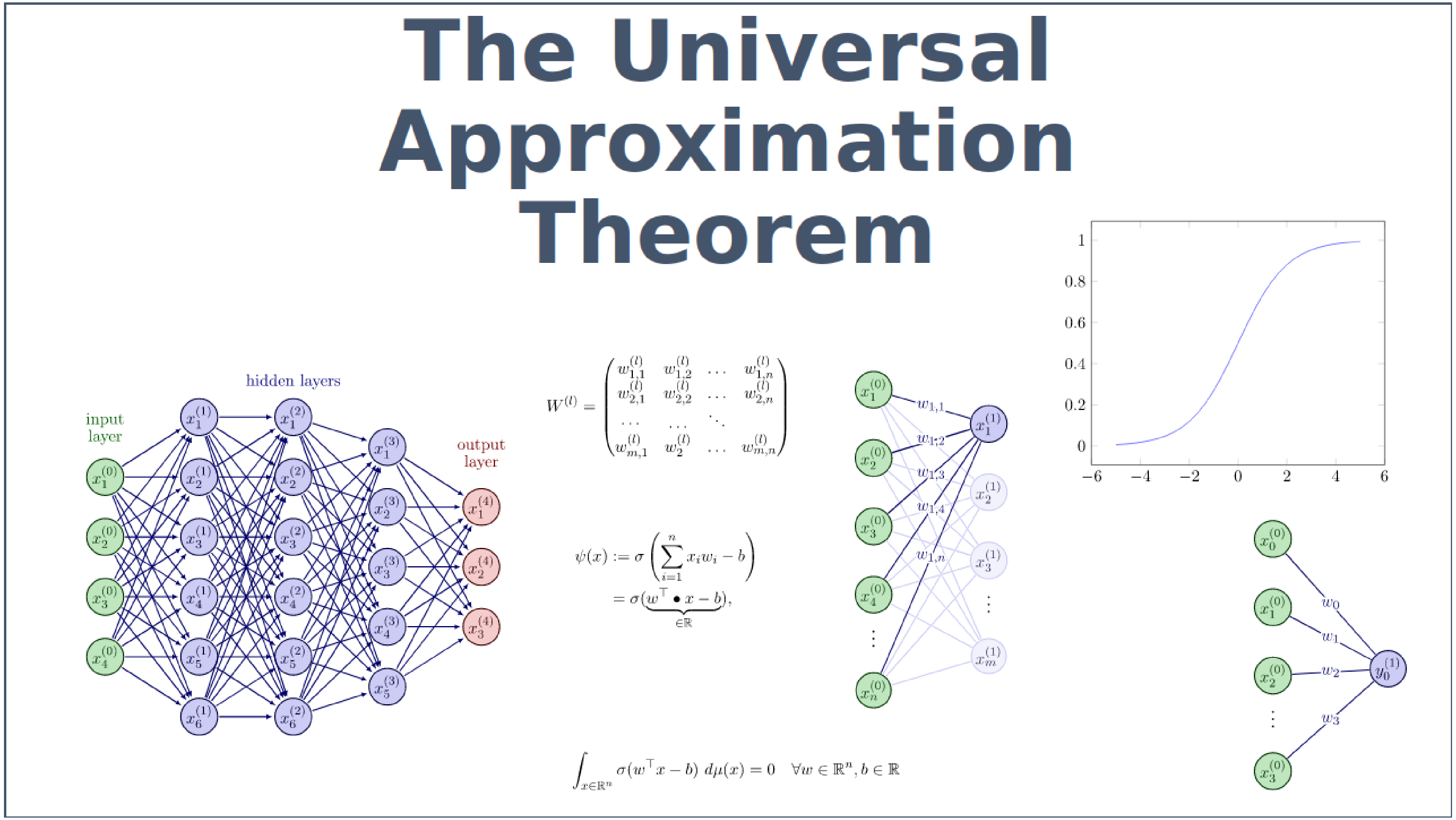




Structure of Neuron being used in Deep Neural Network

Activation function is included

Deep Neural Network (DNN)

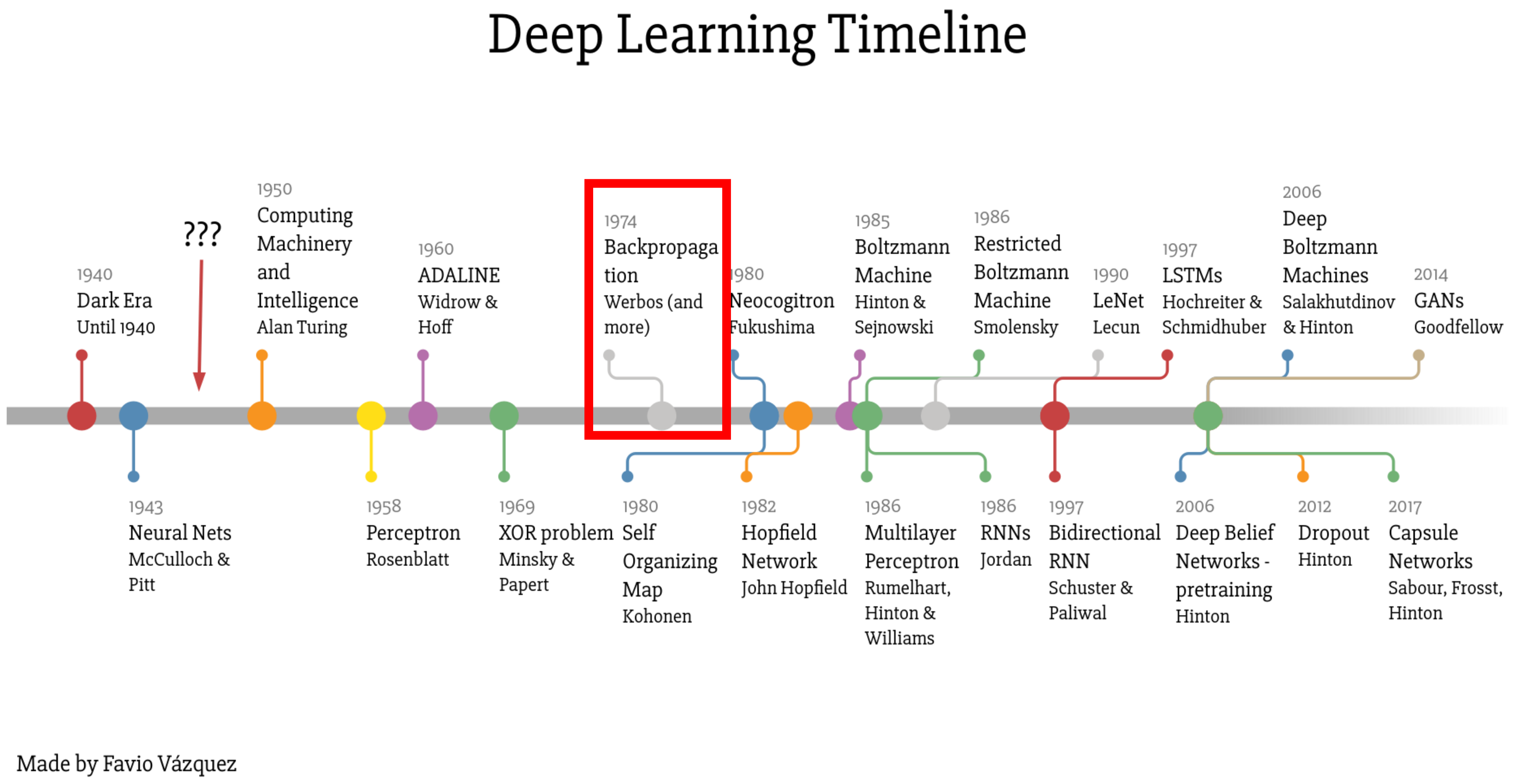


How does DNN works?

[Universal approximation theorem - Wikipedia]

Linear Regression & Backpropagation

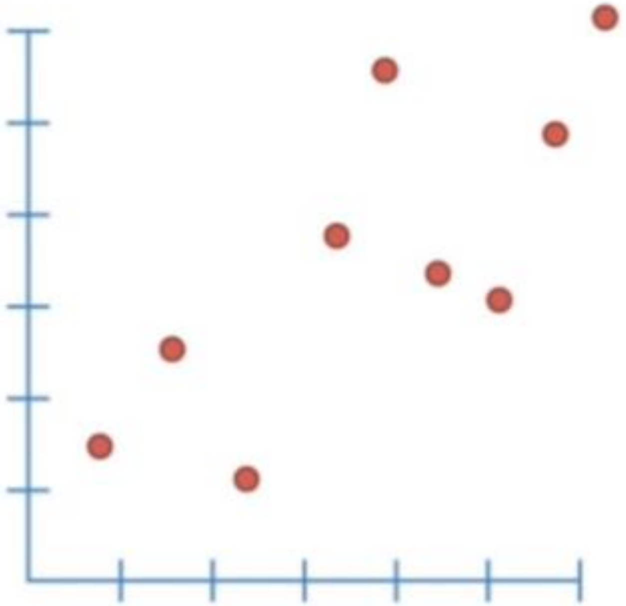
Backpropagation



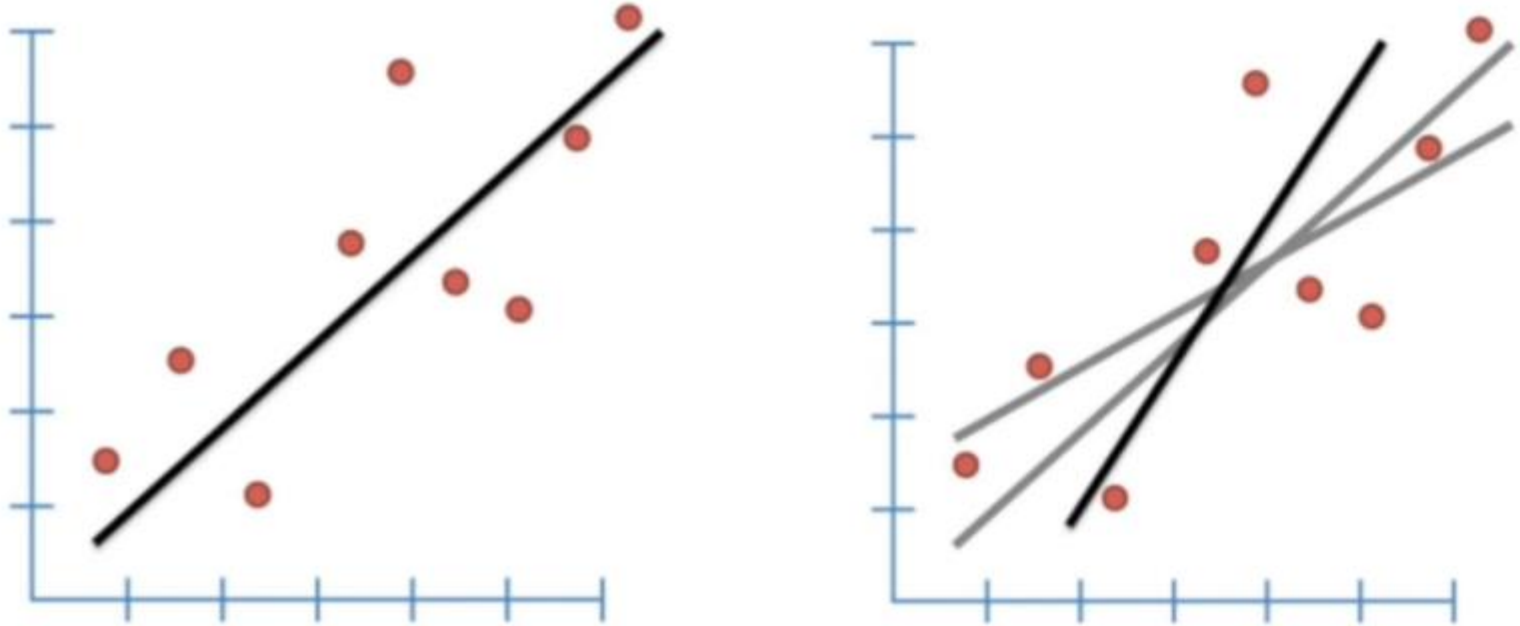
Linear Regression & Backpropagation

Linear Regression

Linear Regression
(also known as least squares)



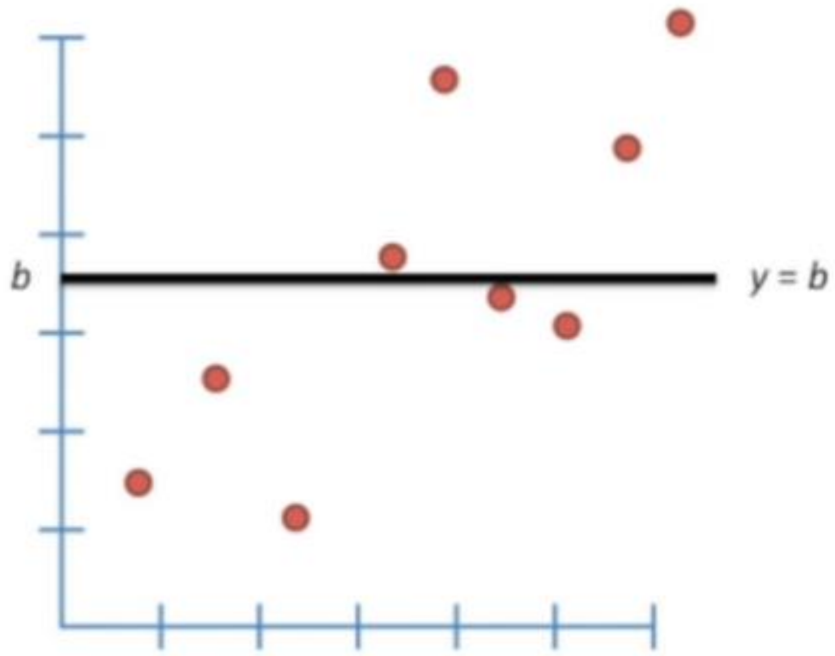
Linear Regression



Fitting a line so we can see what the trend is

Linear Regression & Backpropagation

Linear Regression



The bad fit

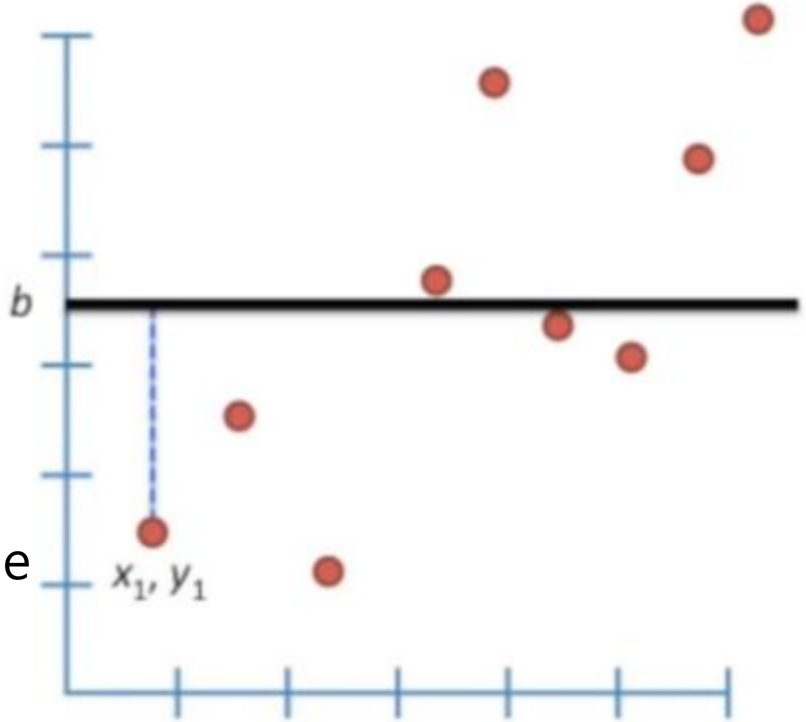
How do we estimate the fitting?

Linear Regression & Backpropagation

Linear Regression

We can measure how well this line fits the data
by seeing how close it is to the data points

The distance between the line
and 1st data point is b – y1

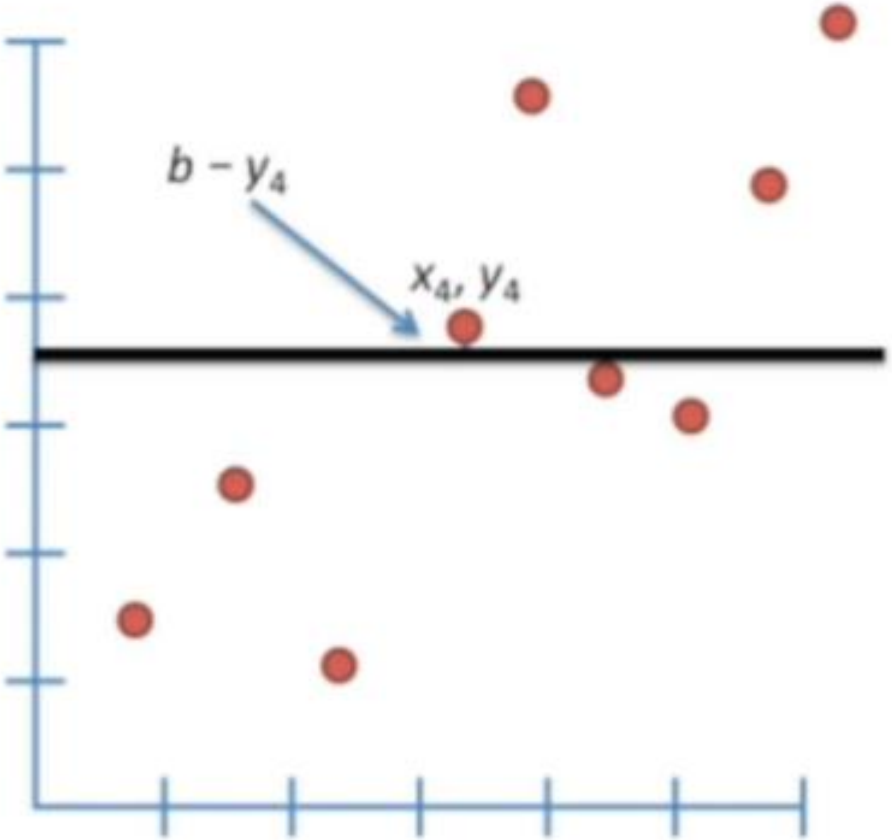


Linear Regression

Score = (b-y1) + (b-y2) + (b-y3) + (b-y4)

y4 > b
⇒ b-y4 will be negative.

⇒ Problem?

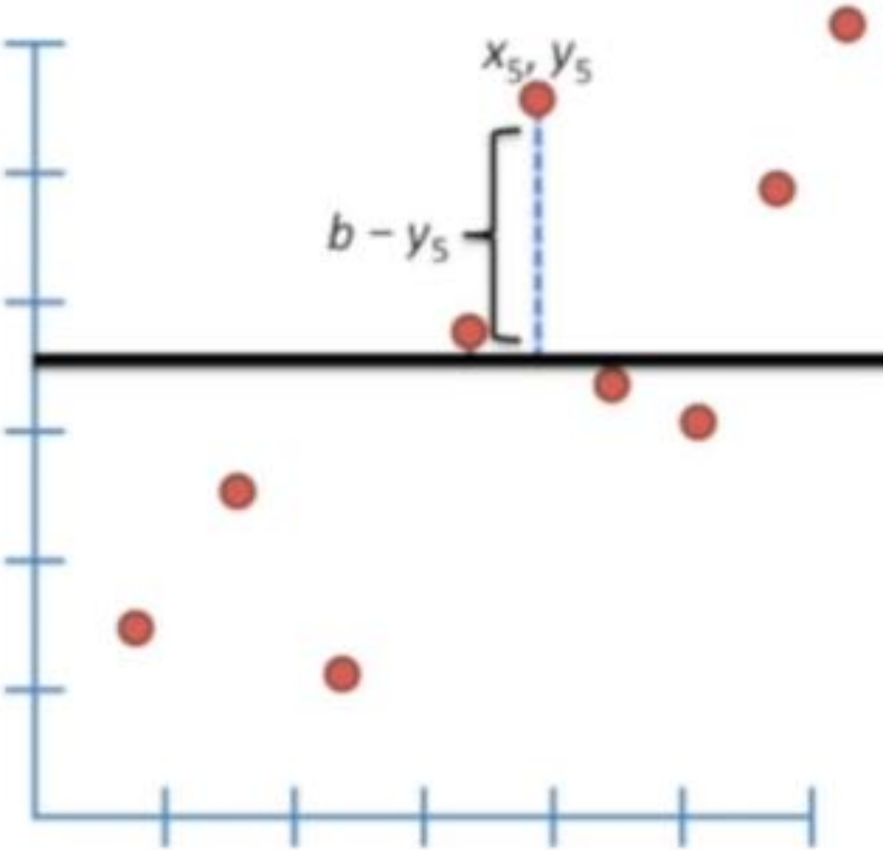


Linear Regression & Backpropagation

Linear Regression

Score = (b-y1) + (b-y2) + (b-y3) + (b-y4) + (b-y5)

Solution?



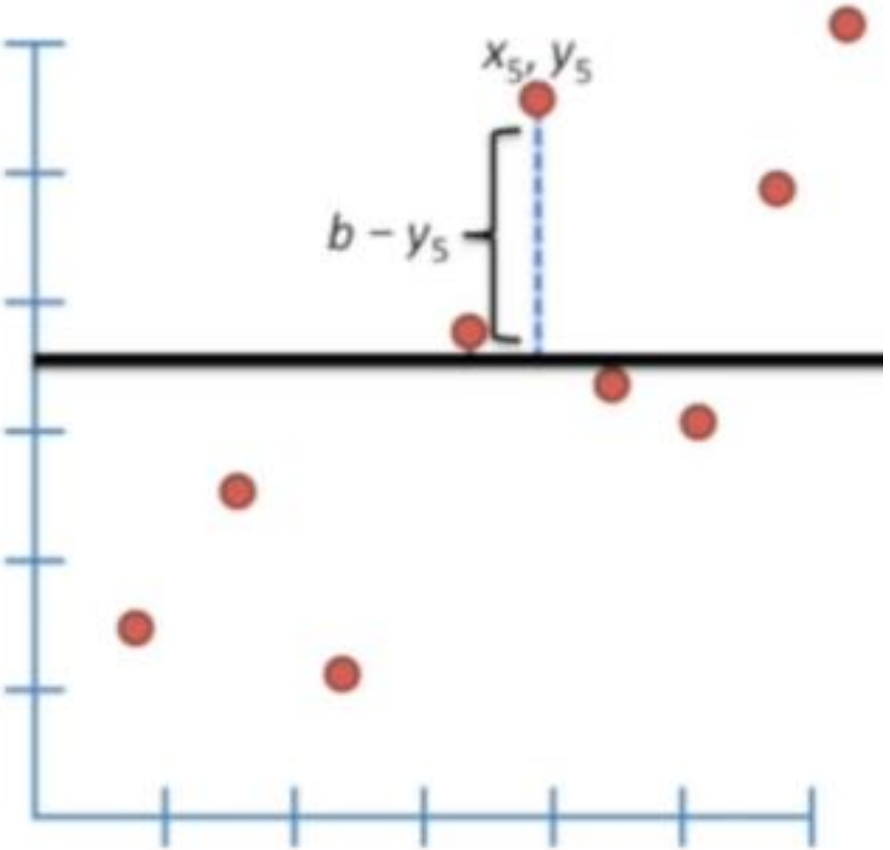
Linear Regression & Backpropagation

Linear Regression

Score = (b-y1) + (b-y2) + (b-y3) + (b-y4) + (b-y5)

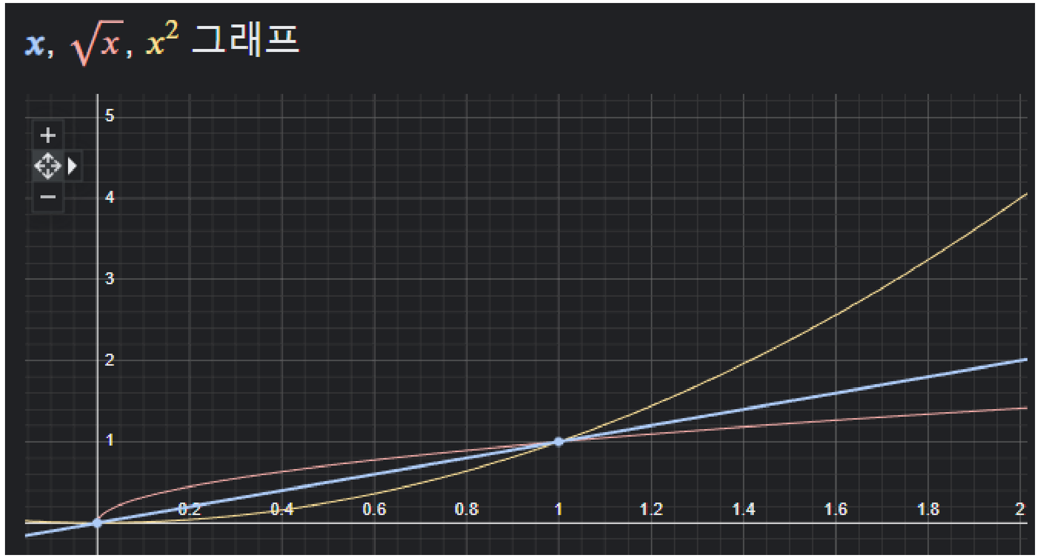
Solution?

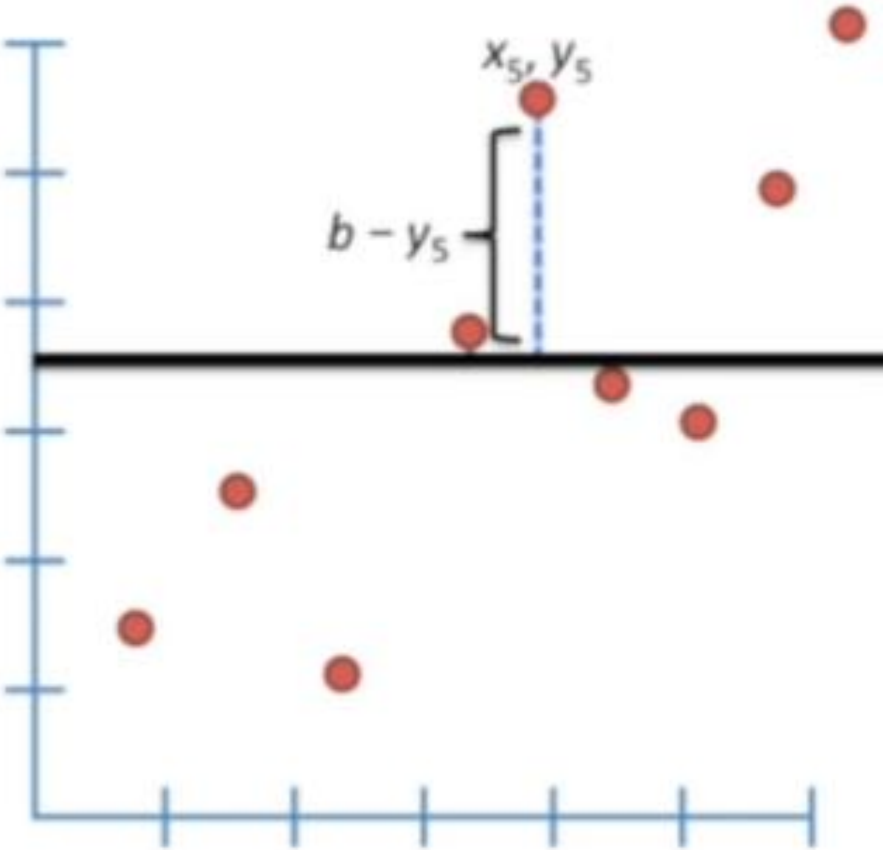
Score = |(b-y1)| + |(b-y2)| + |(b-y3)| + |(b-y4)| + |(b-y5)|



Linear Regression

Score = (b-y1) + (b-y2) + (b-y3) + (b-y4) + (b-y5)



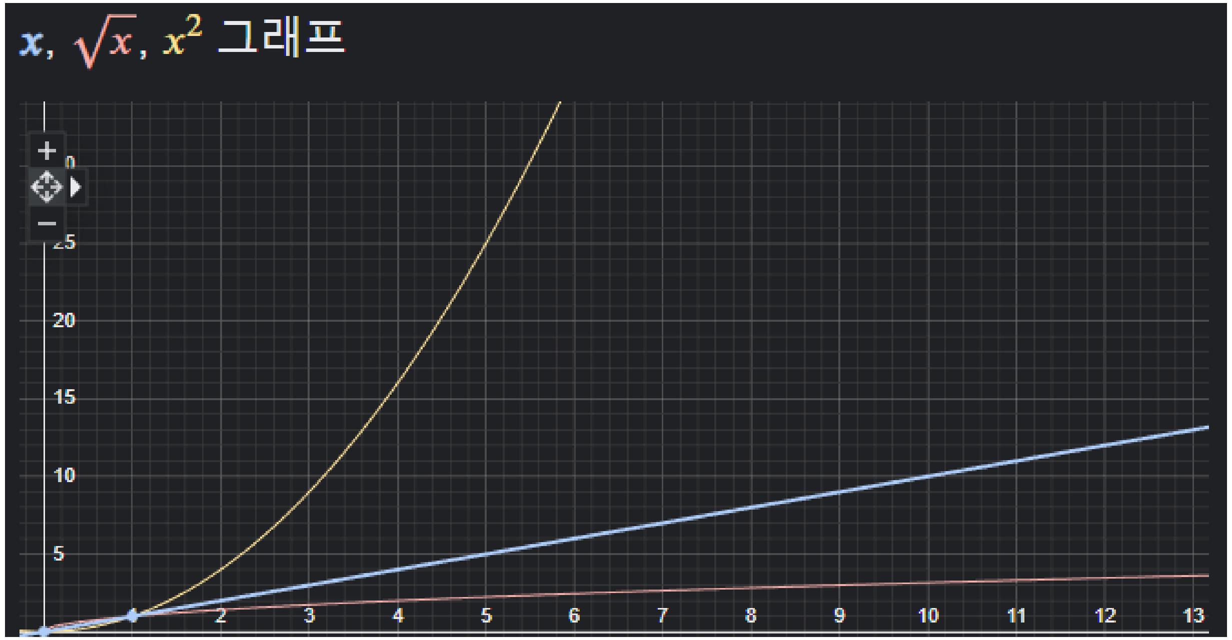


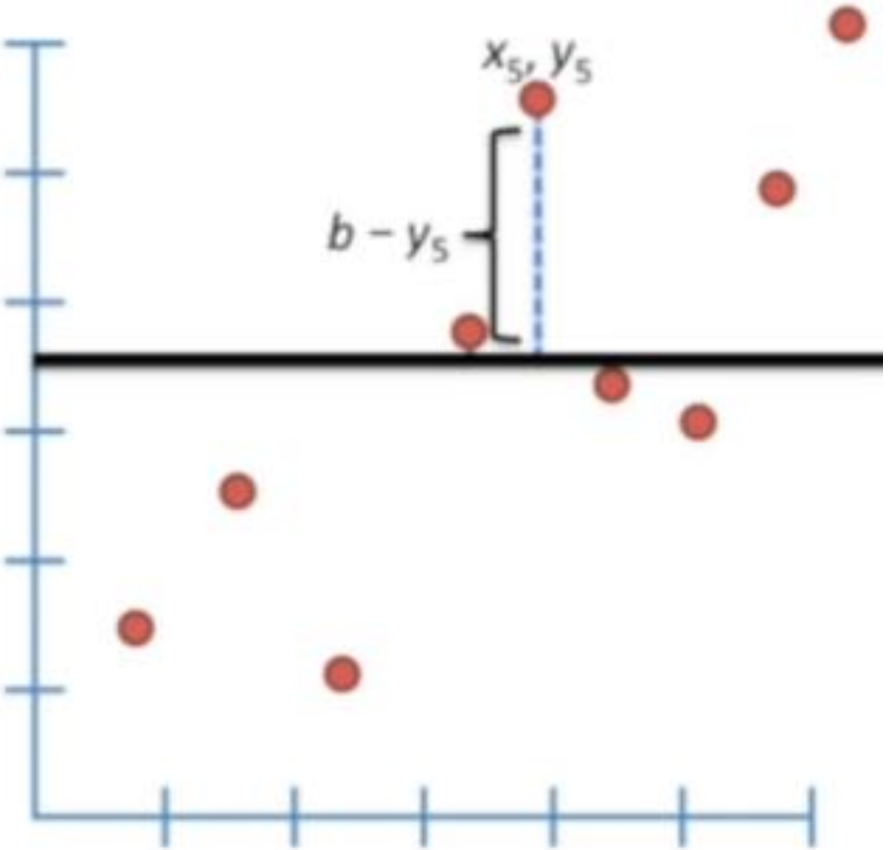
Linear Regression & Backpropagation

Linear Regression

Score = (b-y1) + (b-y2) + (b-y3) + (b-y4) + (b-y5)

Solve this mathematical problem efficiently!

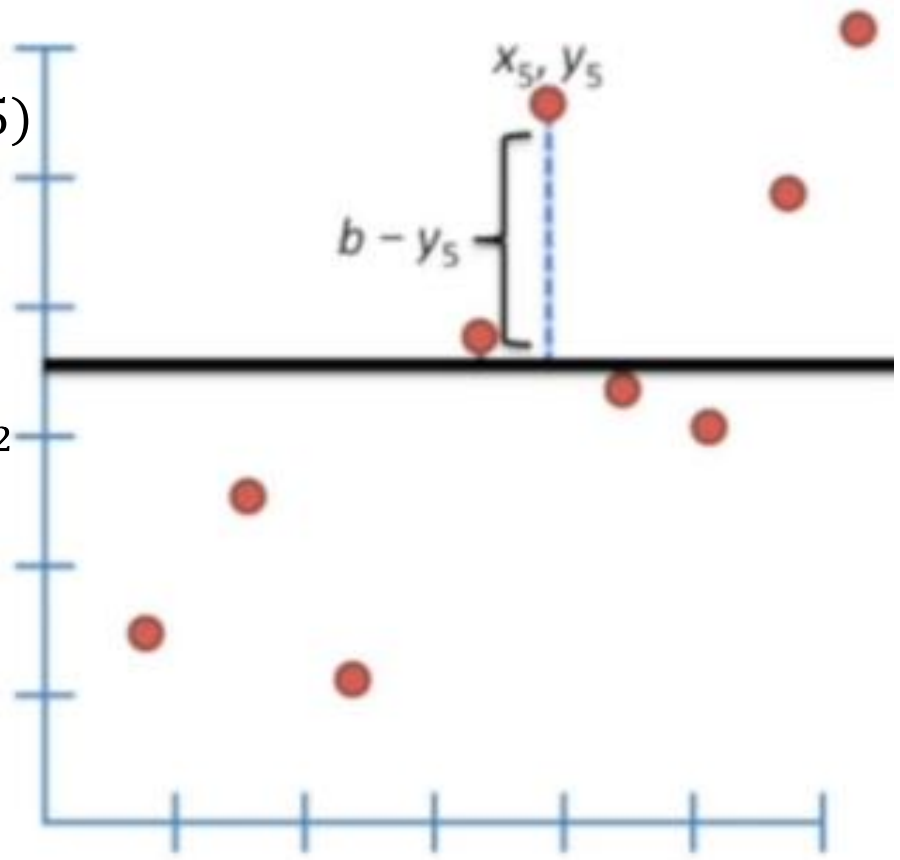




Linear Regression

$$Score = (b - y1) + (b - y2) + (b - y3) + (b - y4) + (b - y5)$$

Solve this mathematical problem efficiently!



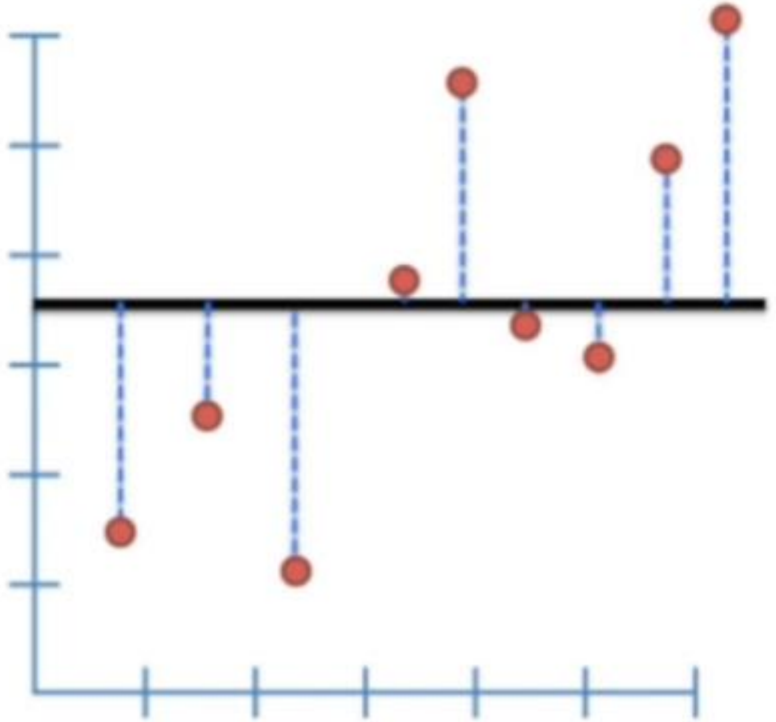
$$Score = (b - y1)^2 + (b - y2)^2 + (b - y3)^2 + (b - y4)^2 + (b - y5)^2$$

We can square each term!

Linear Regression & Backpropagation

Linear Regression
Let's rotate the line a little bit

$$Score = (b - y1)^2 + (b - y2)^2 + \cdots + (b - y4)^2 + (b - y9)^2$$
$$= 23.45$$

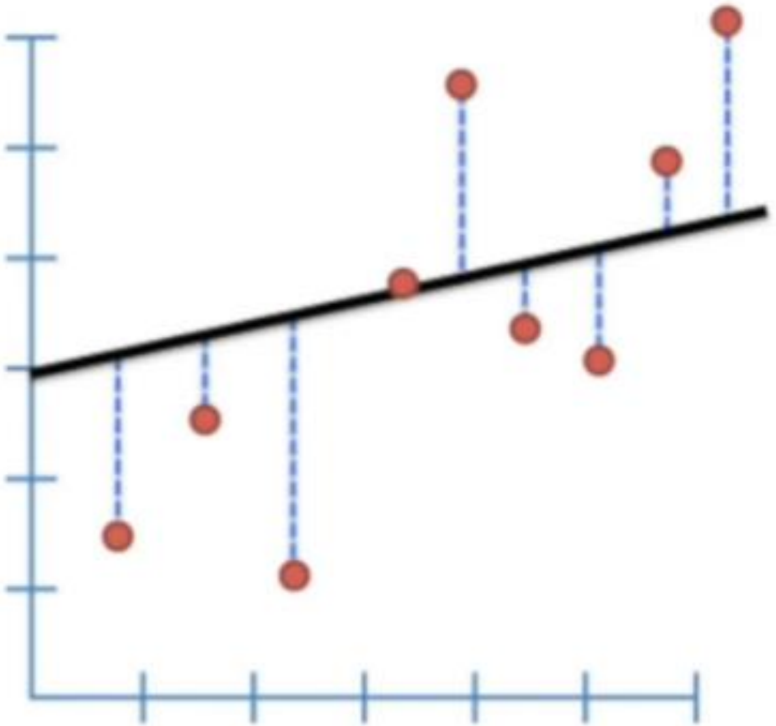


Linear Regression & Backpropagation

Linear Regression
Let’s rotate the line a little bit

$$Score = (b-y1)^2+(b-y2)^2+\cdots+(b-y4)^2+(b-y9)^2$$
$$= \cancel{23.45}$$

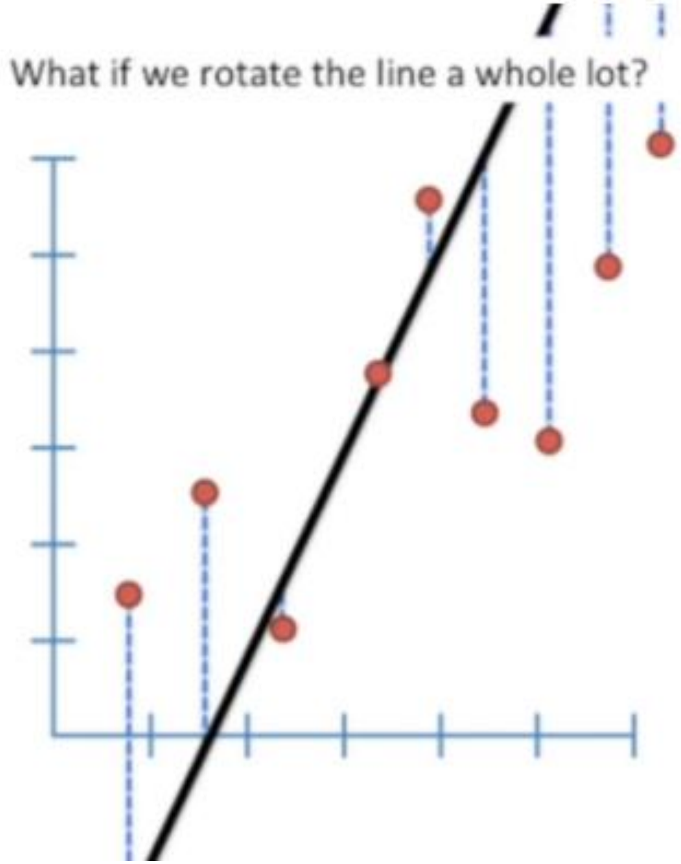
The better result



Linear Regression & Backpropagation

Linear Regression
What if we rotate too much?

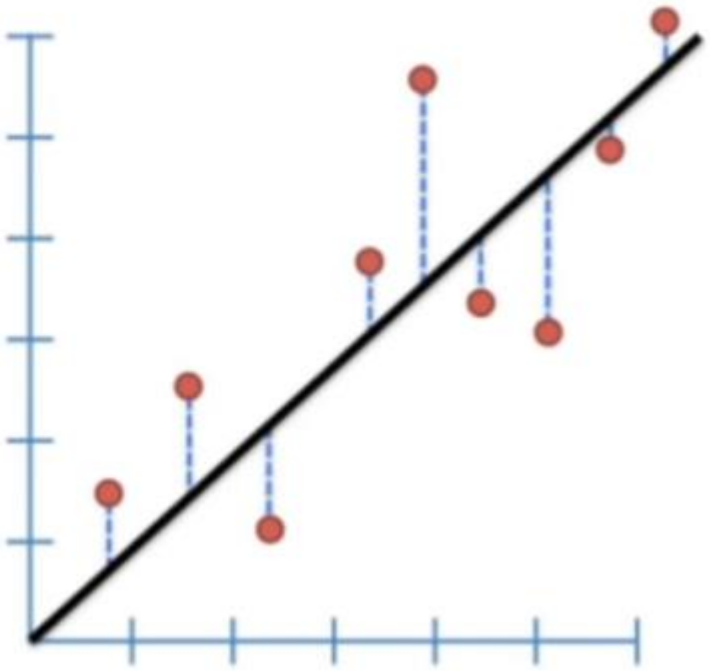
$$Score\ =\ (b-y1)^2+(b-y2)^2+\cdots+\ (b-y4)^2+(b-y9)^2$$
$$=34.56$$



Linear Regression & Backpropagation

Linear Regression
Equation expression

Generic line equation: $y = a * x + b$
Goal: Find optimal values for "a" and "b" so that we minimize the sum of squared residuals

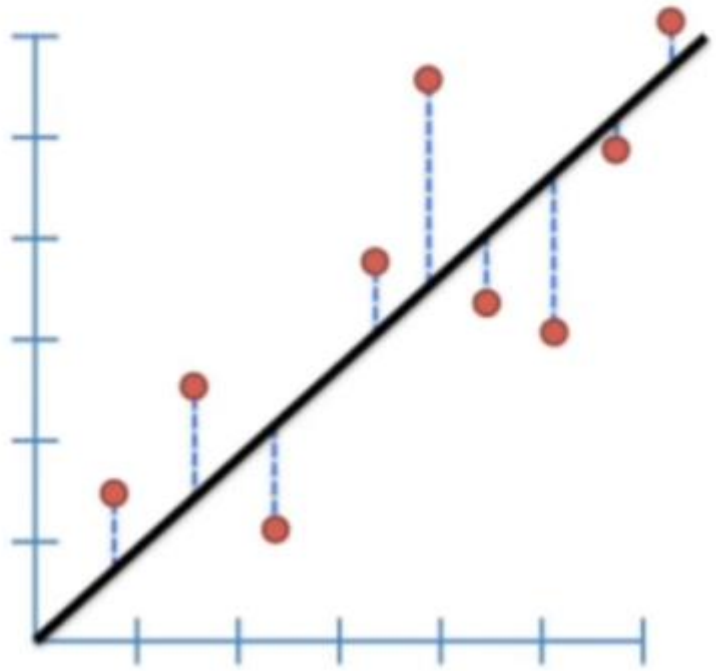


Linear Regression & Backpropagation

Linear Regression
Equation expression

Generic line equation: $y = a * x + b$
Goal: Find optimal values for “a” and “b” so that we minimize the sum of squared residuals

$$Sum\ of\ squared\ residuals$$
$$= ((a * x_1 + b) - y_1)^2 + ((a * x_2 + b) - y_2)^2 + \cdots$$
$$+ ((a * x_9 + b) - y_9)^2$$



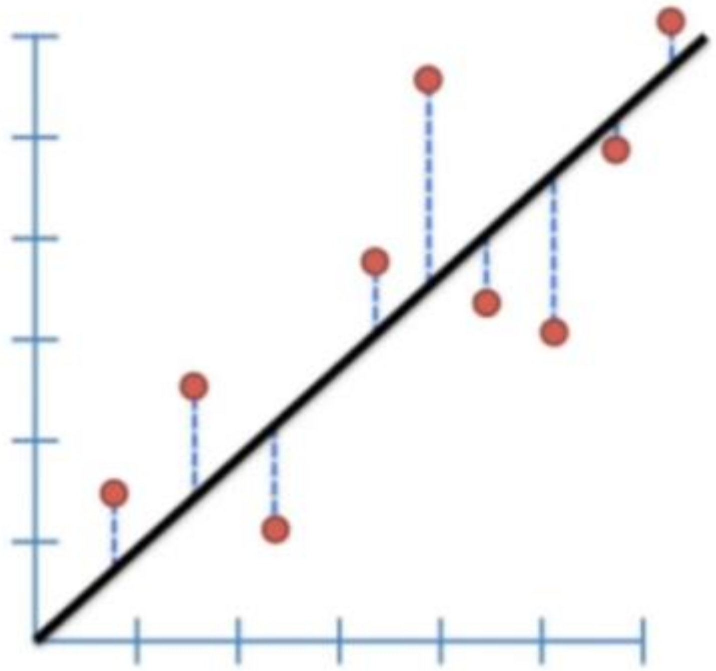
Linear Regression & Backpropagation

Linear Regression
Equation expression

Generic line equation: $y = a * x + b$
Goal: Find optimal values for "a" and "b" so that we minimize the sum of squared residuals

$$\textit{Sum of squared residuals} \\ = ((a * x_1 + b) - y_1)^2 + ((a * x_2 + b) - y_2)^2 + \cdots \\ + ((a * x_9 + b) - y_9)^2$$

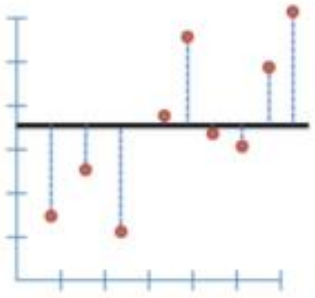
$(a * x_1 + b)$: The value of the line at x_1
y_1: observed value at x_1

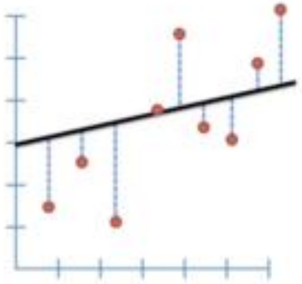


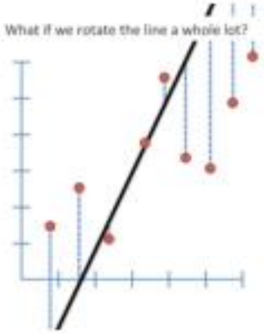
Since we want the line that give us the smallest sum of squares, this methods for finding the best value of "a" and "b" is called "least squares".

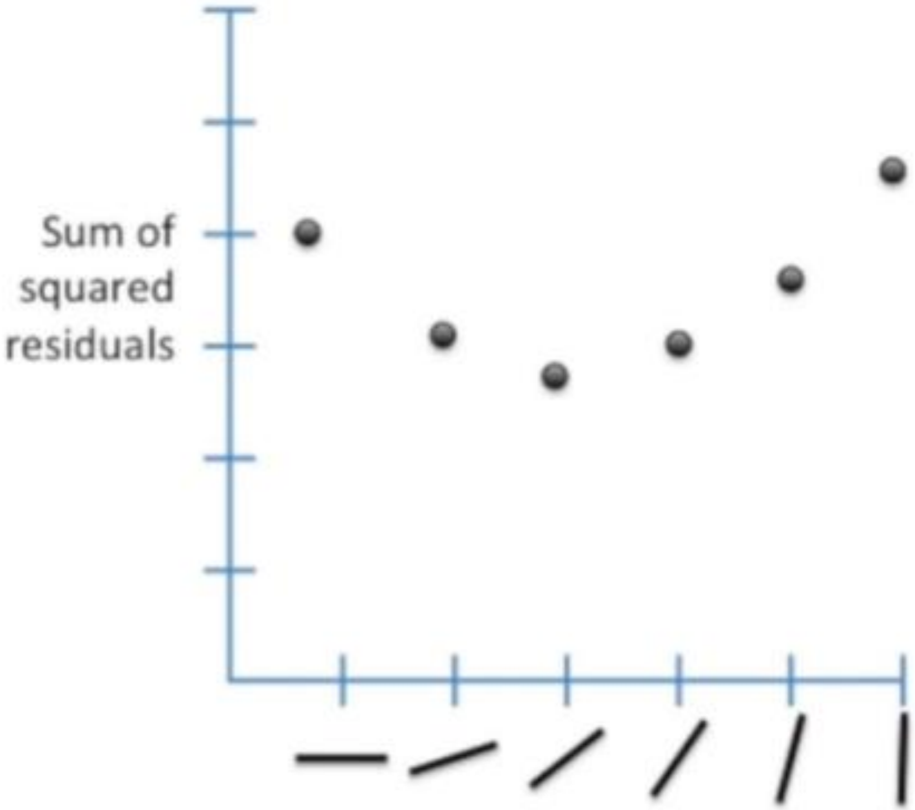
Linear Regression

Visualization of each rotation



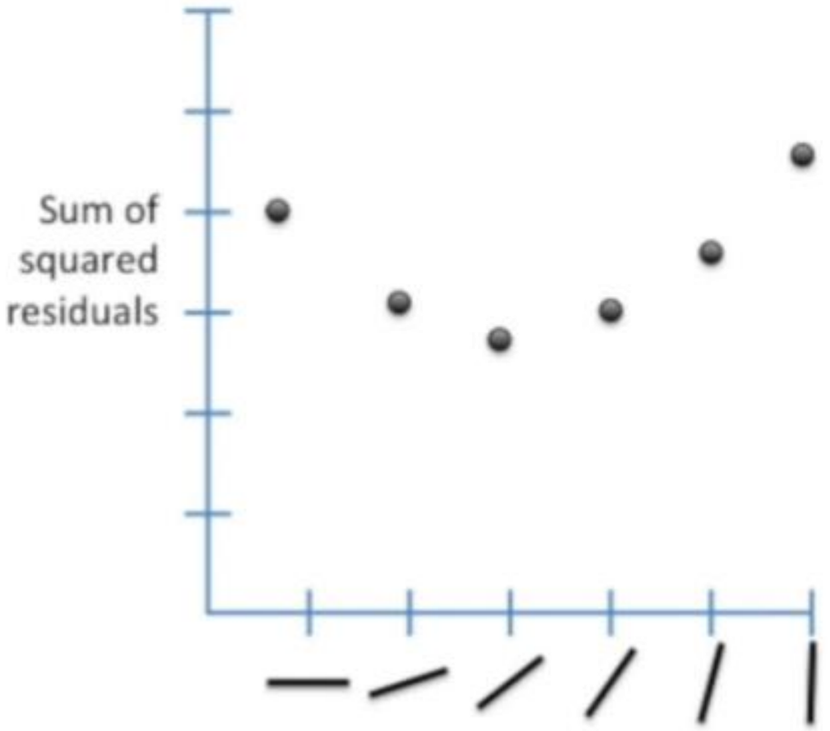






Linear Regression
Visualization of each rotation

How do we find the optimal rotation for the line?

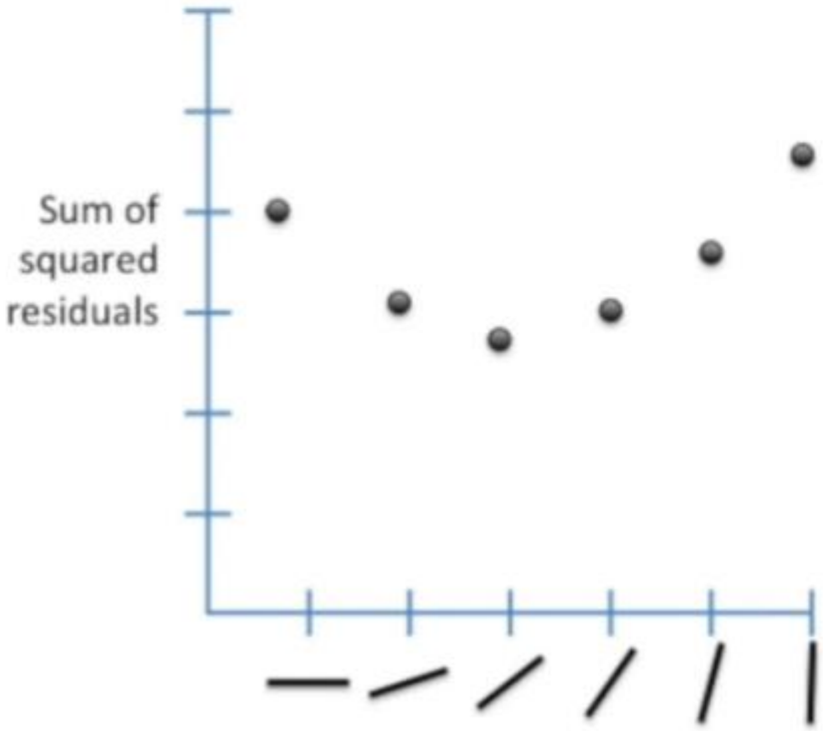


Linear Regression & Backpropagation

Linear Regression
Visualization of each rotation

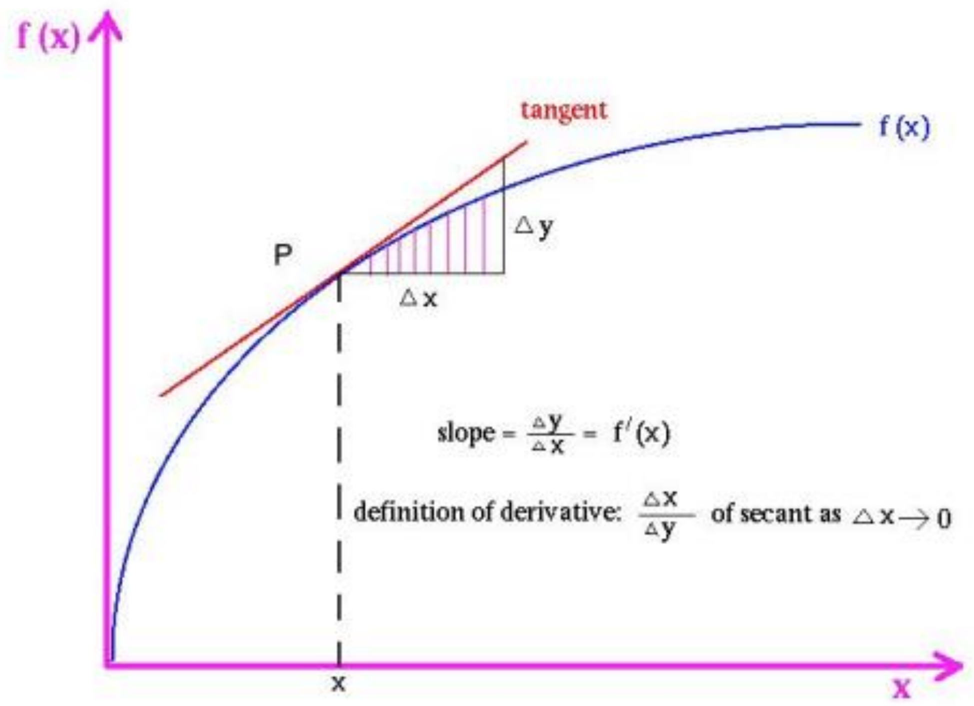
How do we find the optimal rotation for the line?

Derivatives!



Linear Regression & Backpropagation

Linear Regression
Derivative



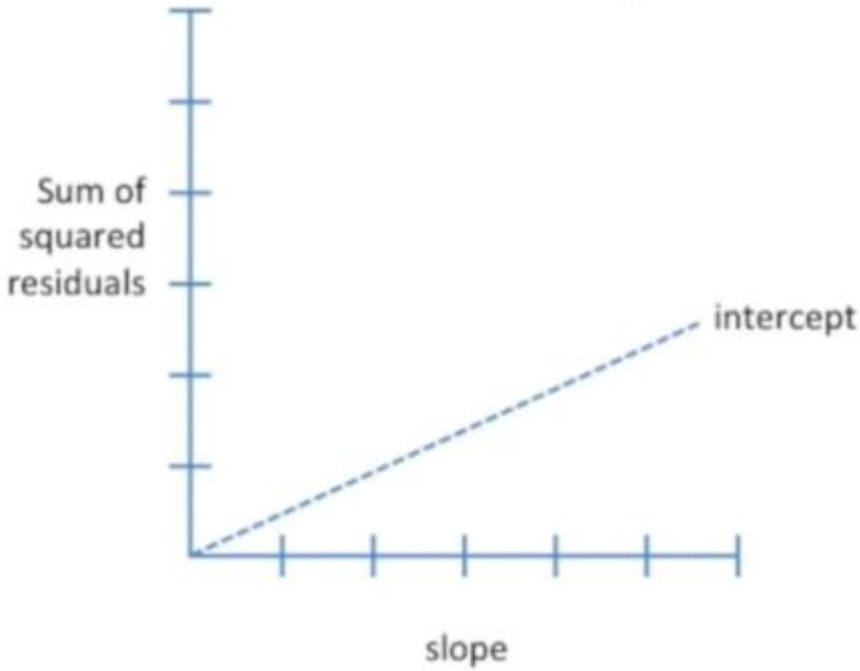
Linear Regression & Backpropagation

Linear Regression
Two independent variables

$y = a * x + b$

We have two independent variables that needs to be optimized

1) a: the slope
2) b: the intercept



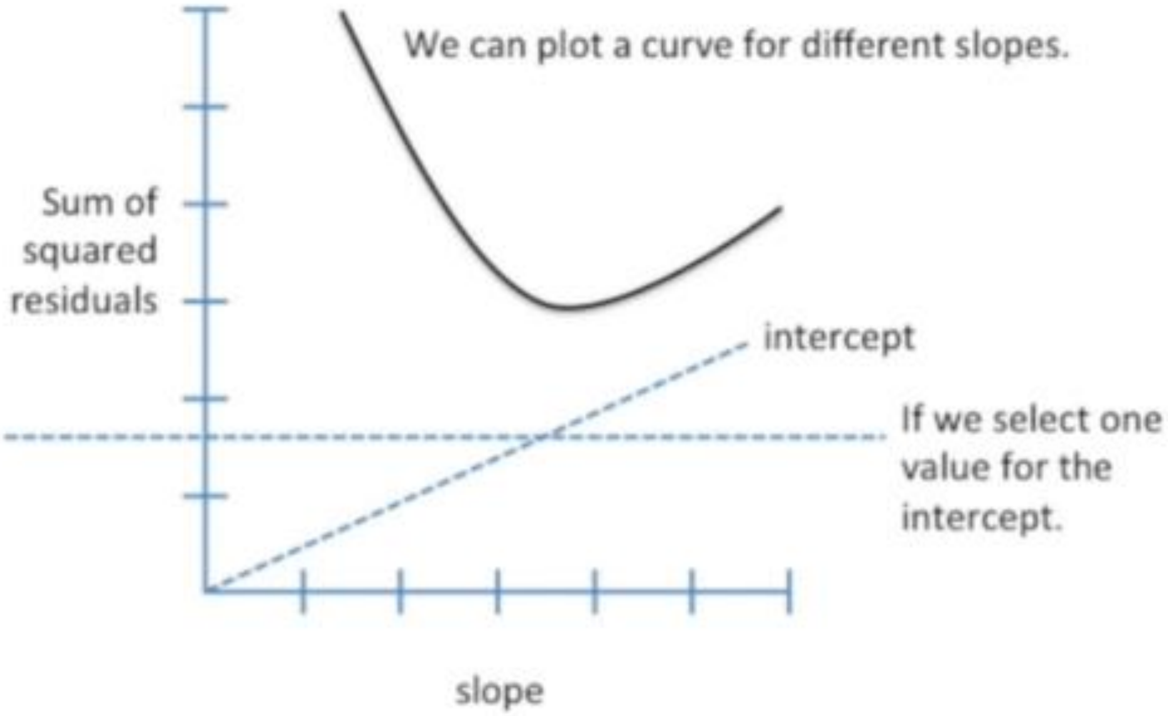
Linear Regression & Backpropagation

Linear Regression
If the intercept has fixed?
=> Partial derivative

$$y = a * x + b$$

We have two independent variables that needs to be optimized

1) a: the slope
2) b: the intercept



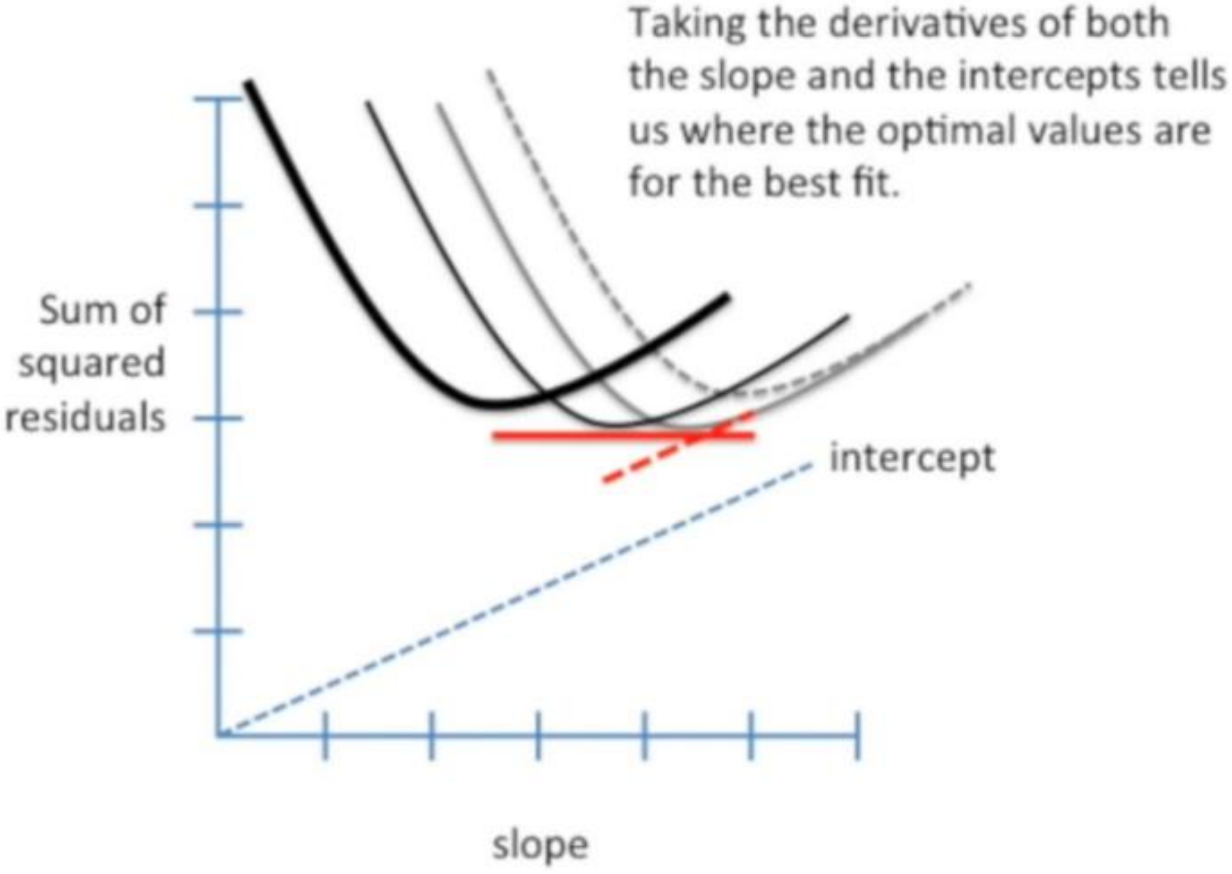
Linear Regression & Backpropagation

Linear Regression
Take the derivative for both independent variables
=> Total derivative

$y = a * x + b$

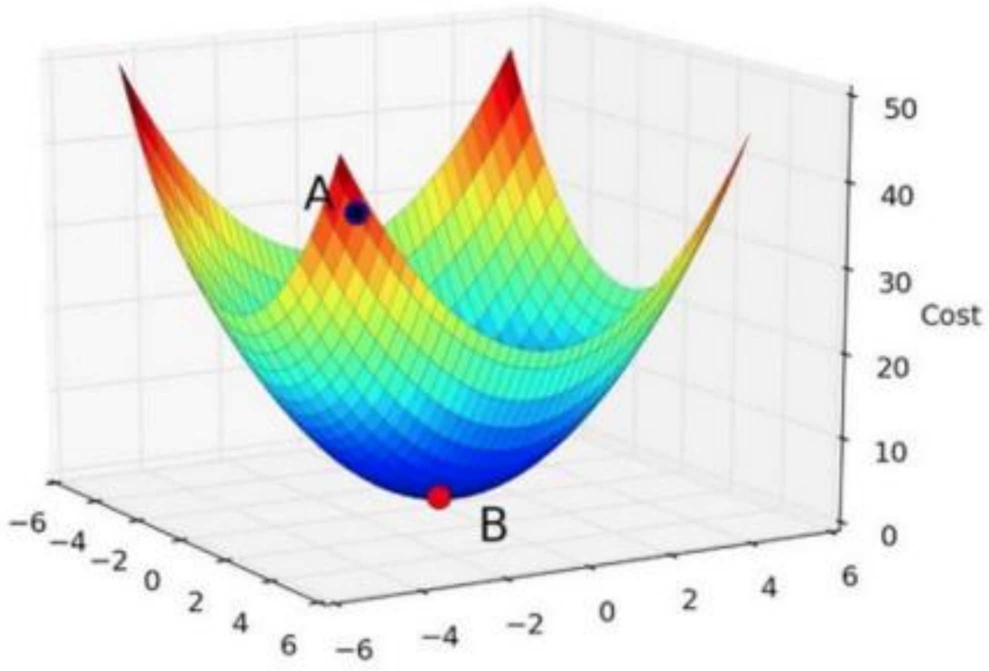
We have two independent variables that needs to be optimized

1) a: the slope
2) b: the intercept



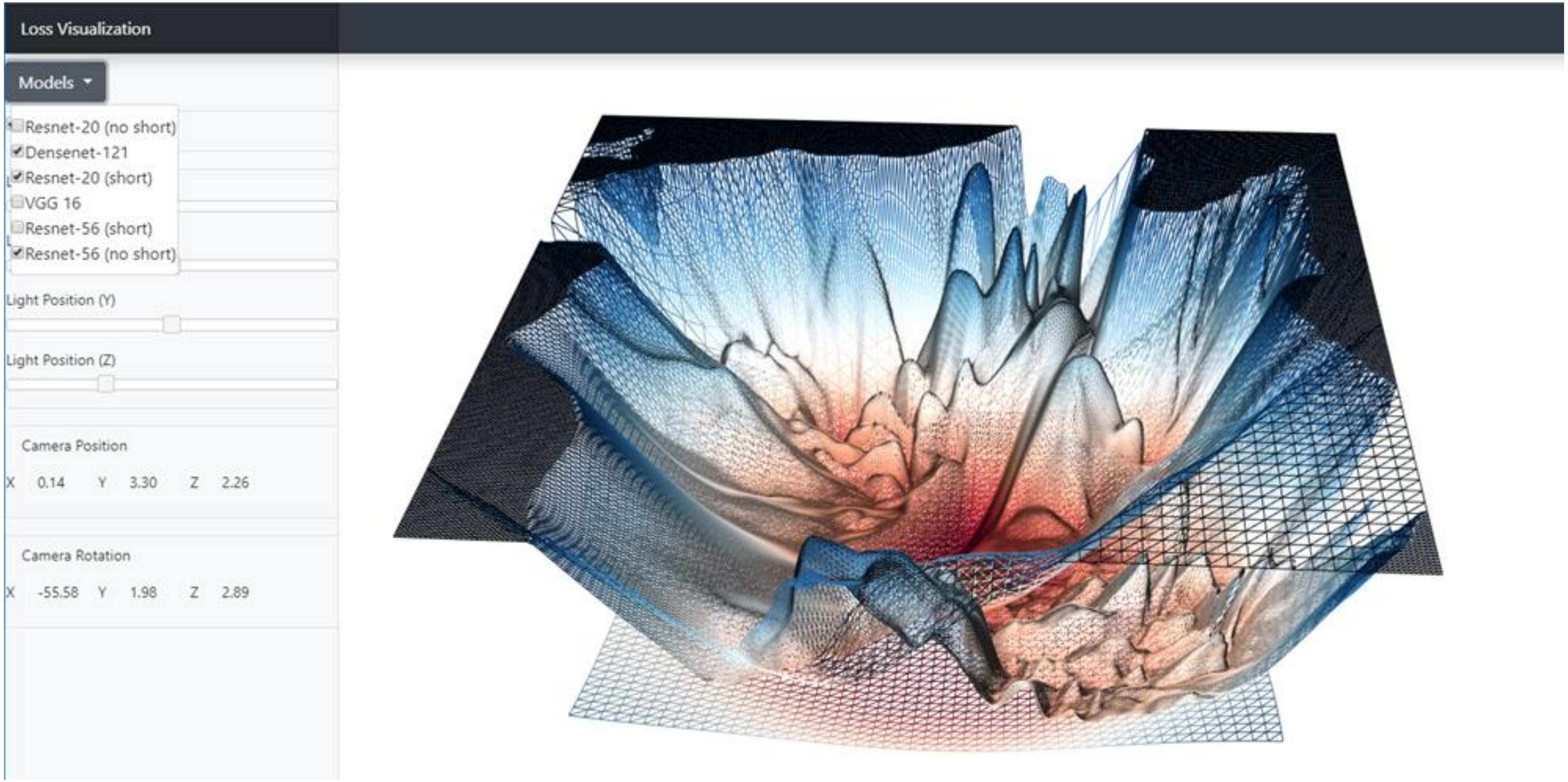
Linear Regression & Backpropagation

Linear Regression
Advanced topic: Gradient Descent



Back to ANN

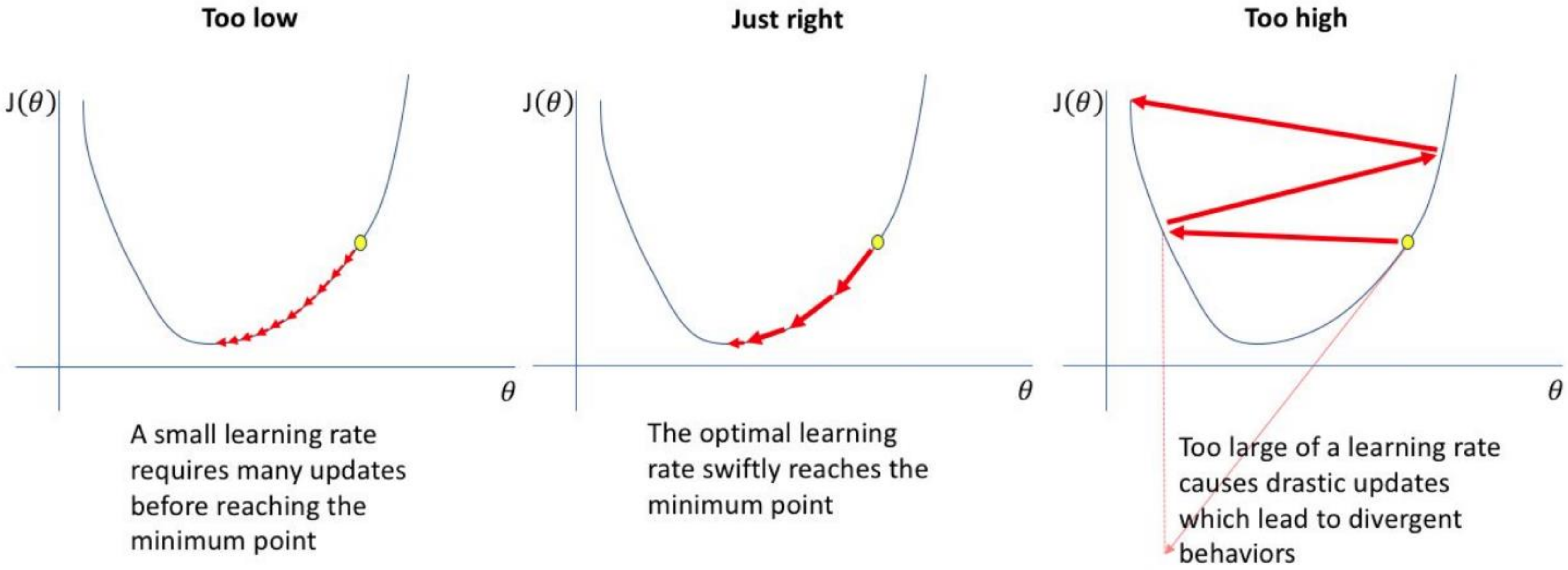
Deep Neural Network (DNN)



Visualized the loss map of
Deep Neural Network

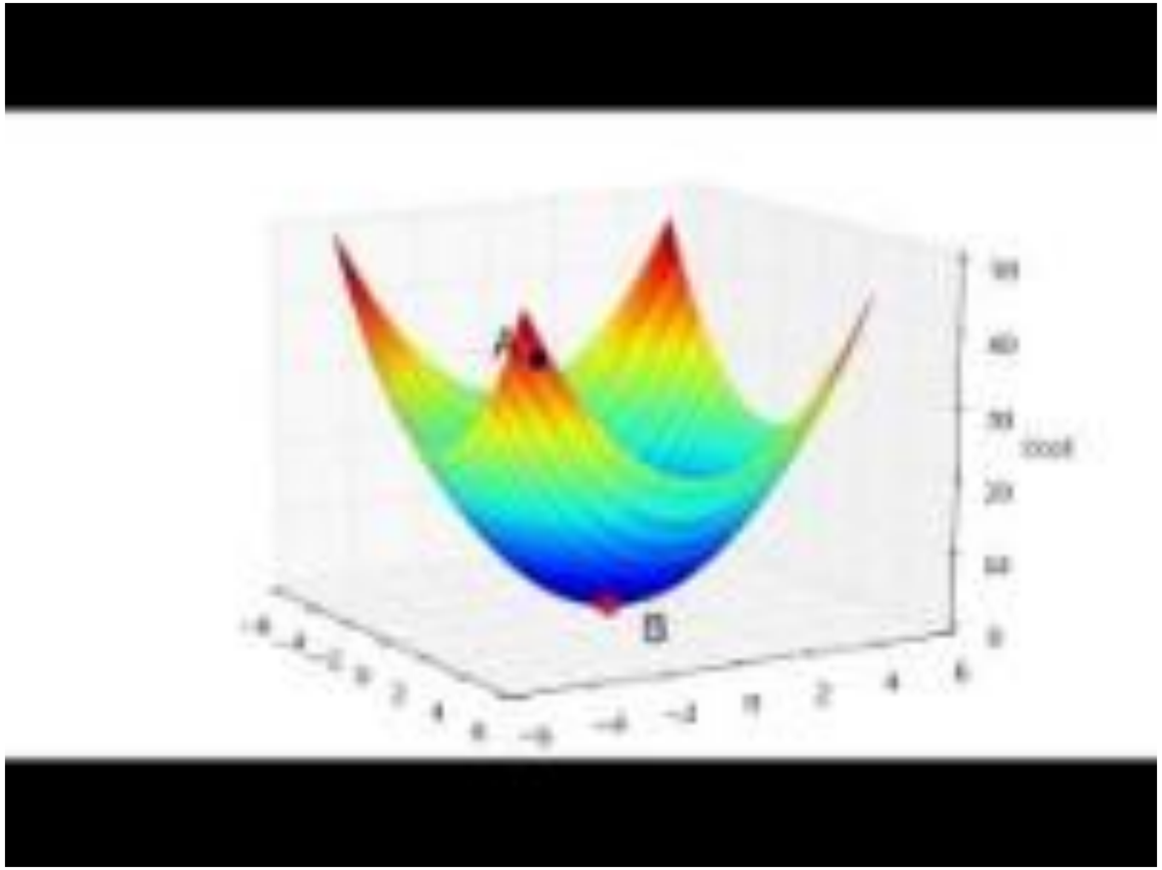
Linear Regression & Backpropagation

Linear Regression
Advanced topic: Gradient Descent



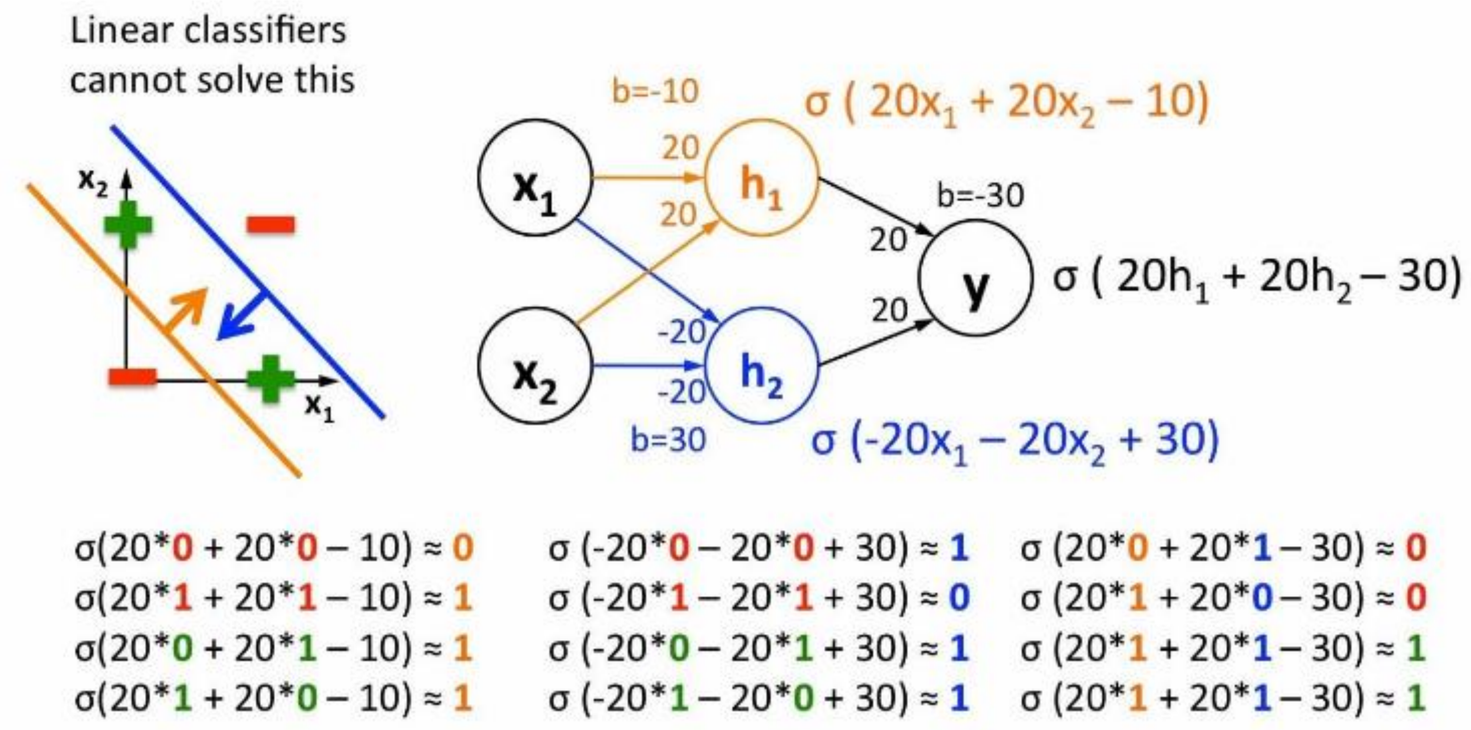
Linear Regression & Backpropagation

Linear Regression
Advanced topic: Gradient Descent



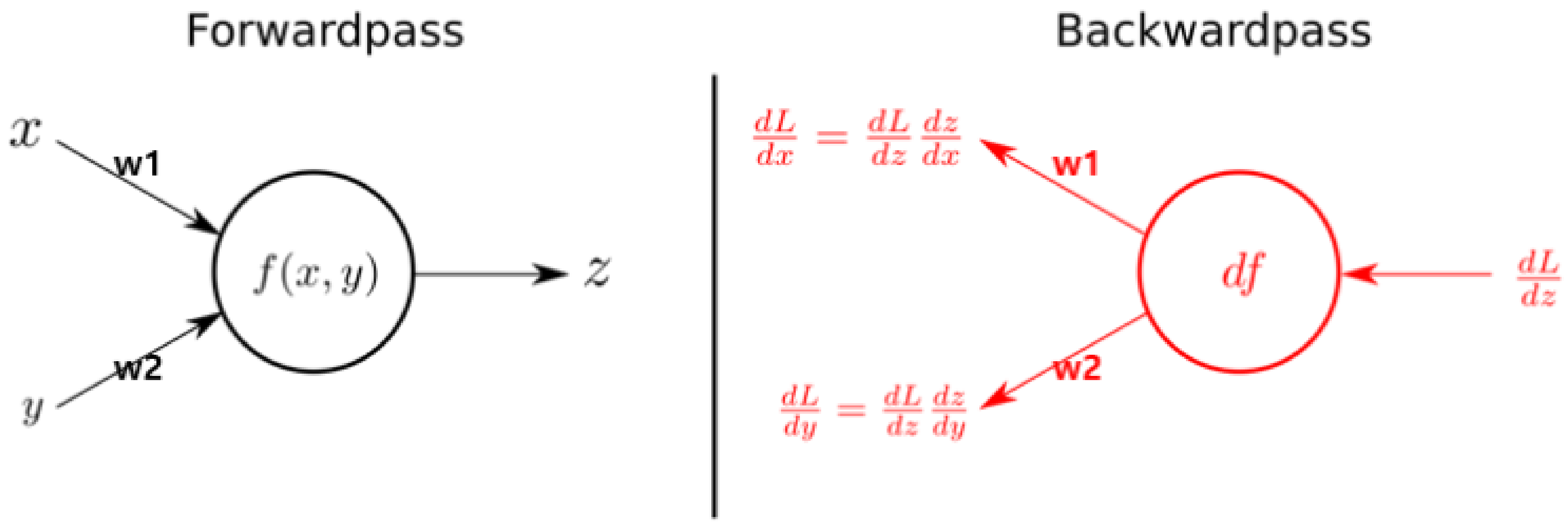
Linear Regression & Backpropagation

Linear Regression
Advanced topic: Backpropagation



Linear Regression & Backpropagation

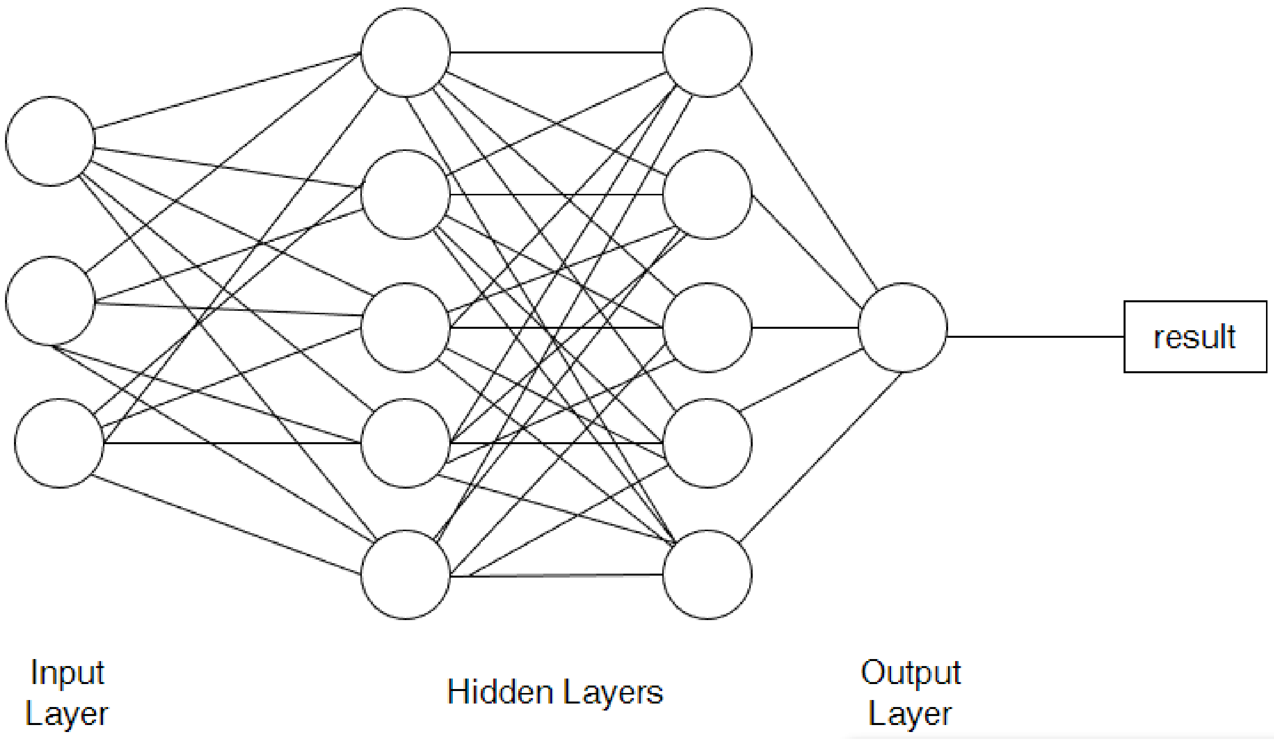
Linear Regression
Advanced topic: Backpropagation

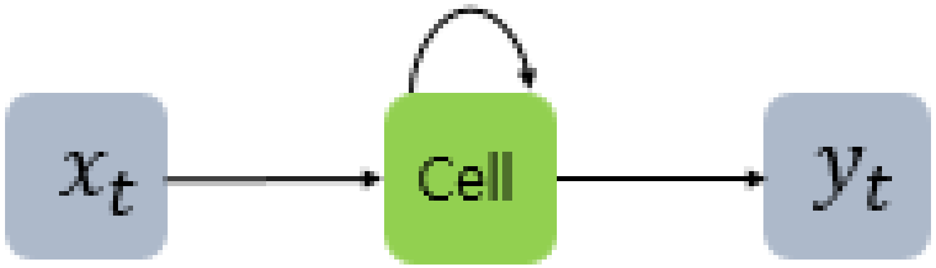


Calculate through the Chain rule!

Recurrent Neural Networks (RNN)

What is RNN?

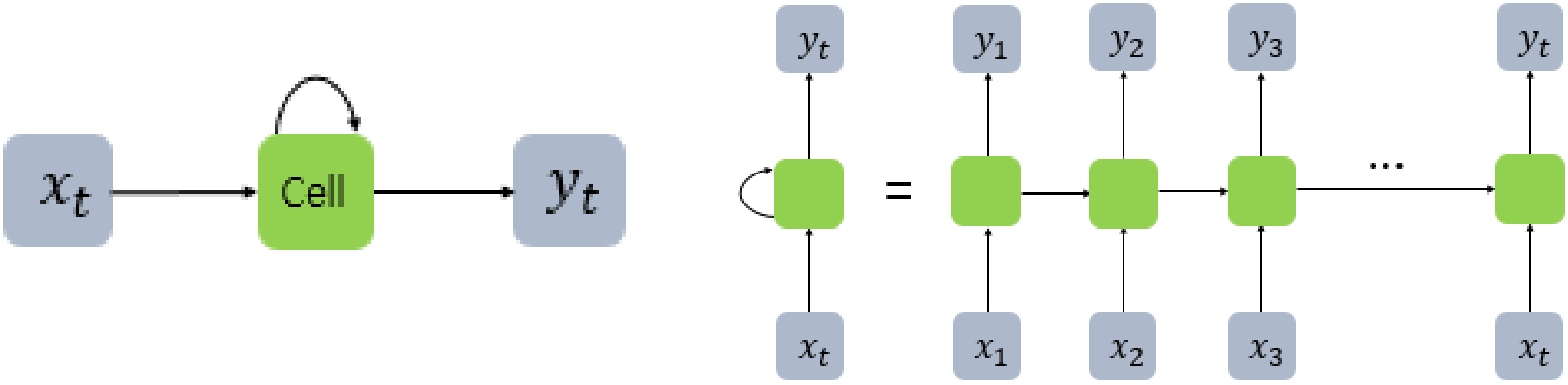




Normal DNN

Recurrent Neural Networks (RNN)

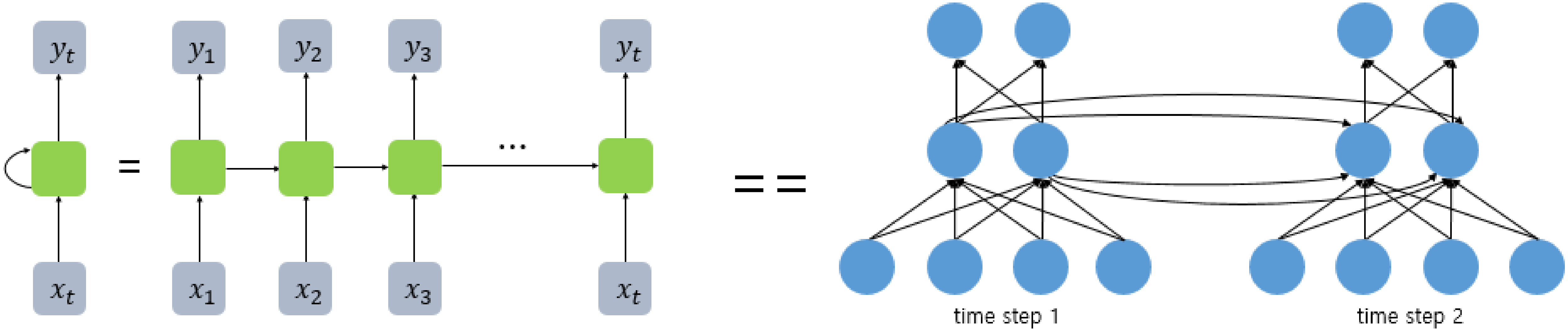
What is RNN?



Output is calculated sequentially!
(or Recursively)

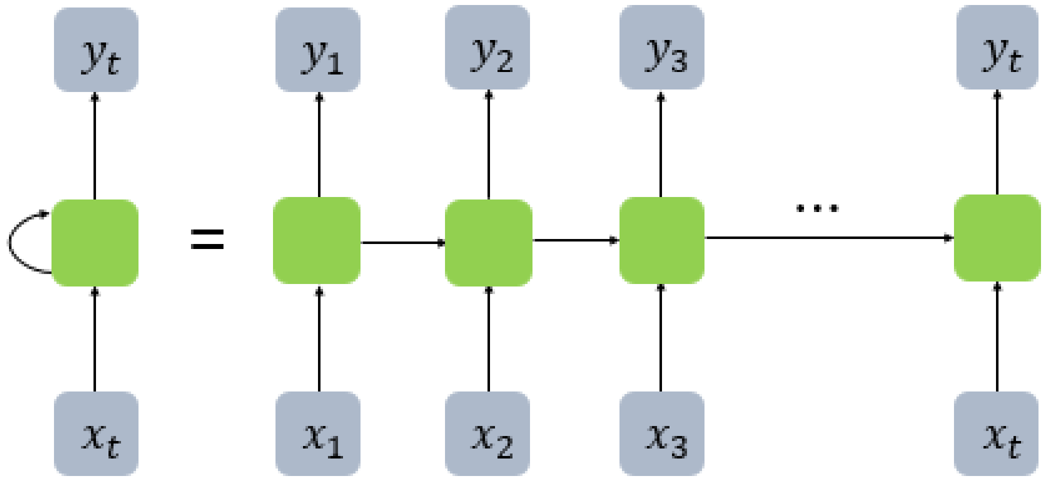
Recurrent Neural Networks (RNN)

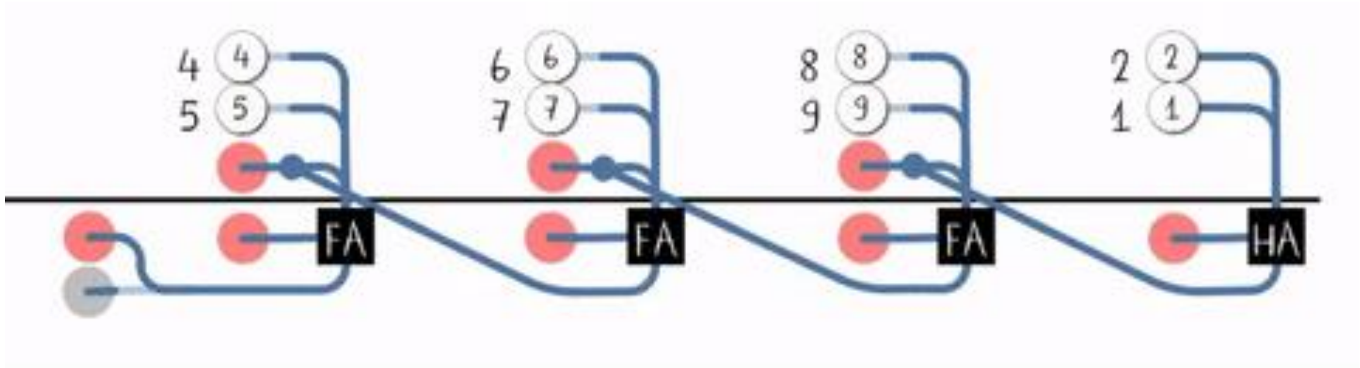
Visualization of RNN



Recurrent Neural Networks (RNN)

ETC

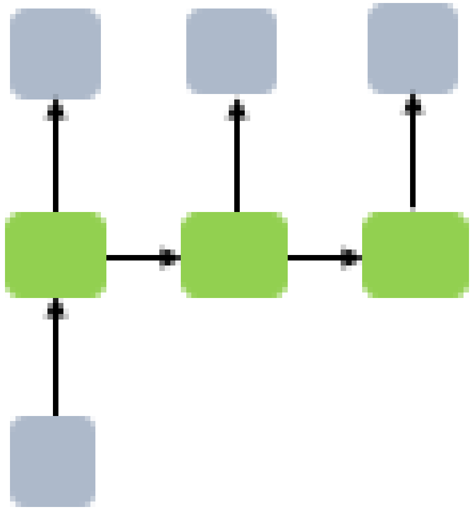




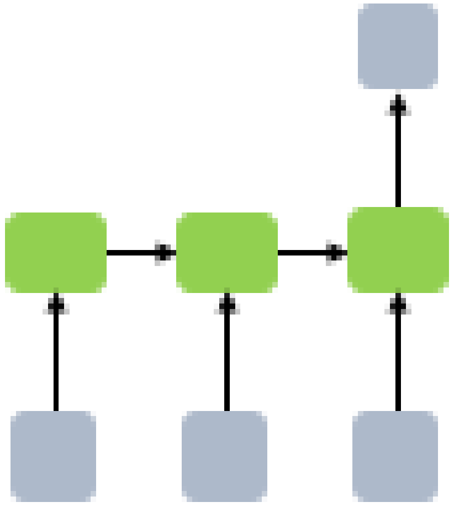
Ripple Carry Adder

Recurrent Neural Networks (RNN)

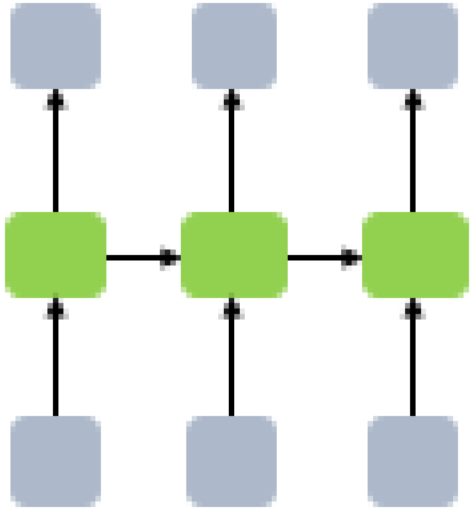
RNN: Examples



One-to-many



Many-to-many



Many-to-many

Ex)

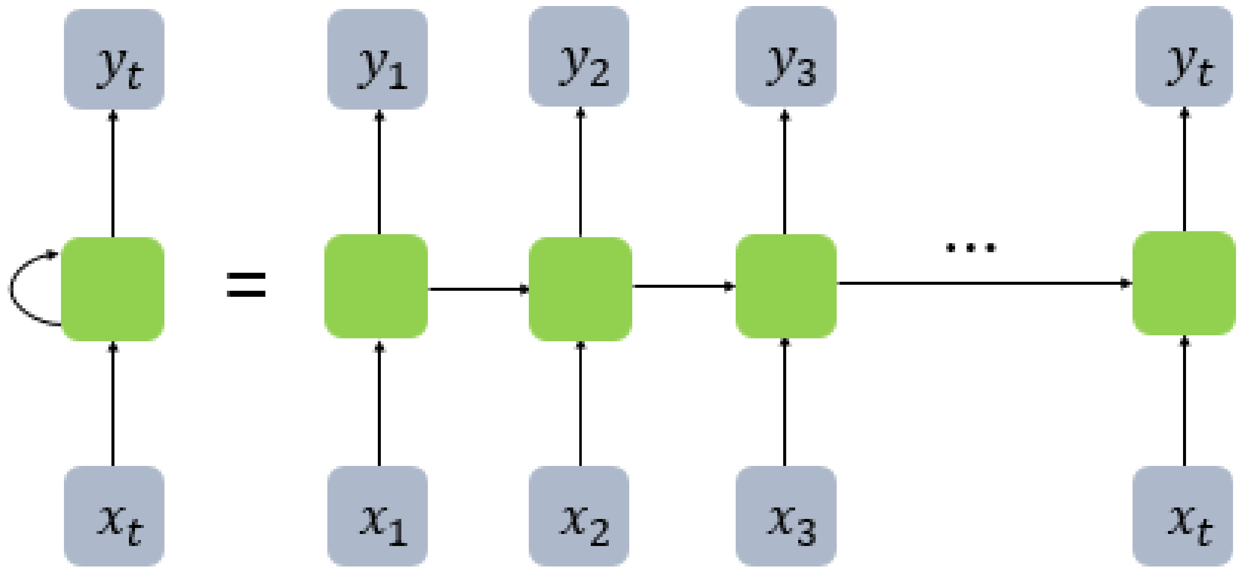
Input: Word
Output: Emotions

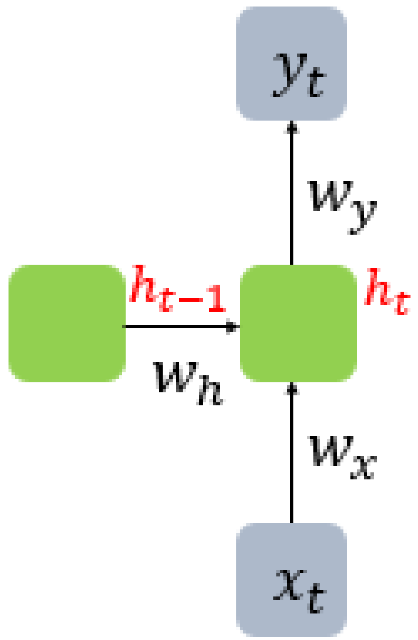
Input: Words
Output: Classes

Input: Stock price
Output: Stock price

Recurrent Neural Networks (RNN)

Equations for RNN



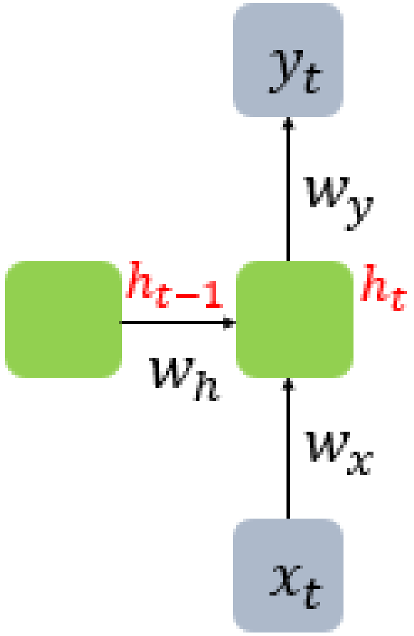


d: Dimension of the input (vector)
D_h: Size of the hidden state
x_t: $(d \times 1)$
W: $(D_h \times d)$
W_h: $(D_h \times D_h)$
h_{t-1}: $(D_h \times 1)$
b: $(D_h \times 1)$

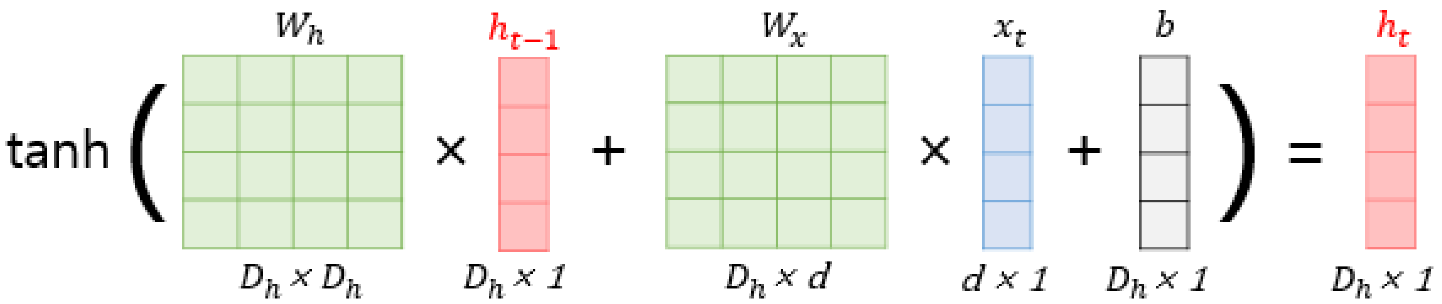
Hidden $h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$
Output $y_t = \mathrm{f}(W_y h_t + \mathrm{b})$
f is a nonlinear activation function.

Recurrent Neural Networks (RNN)

Equations for RNN



Batch size: 1
D_h and d: 4



Visualization of RNN calculation

Hidden $h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$
Output $y_t = \mathrm{f}(W_y h_t + \mathrm{b})$
f is a nonlinear activation function.

d: Dimension of the input (vector)
D_h: Size of the hidden state
x_t: $(d \times 1)$
W: $(D_h \times d)$
W_h: $(D_h \times D_h)$
h_{t-1}: $(D_h \times 1)$
b: $(D_h \times 1)$

Practice: Partial Derivatives for Linear Equations

Equations for RNN

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

n = number of data points

Y_i = observed values

$\hat{Y}_i$ = predicted values

$$y = mx + c$$

$$\frac{\partial MSE}{\partial m} = \frac{2}{n}\sum -x_i(y_i - (mx_i + c)$$

$$\frac{\partial MSE}{\partial c} = \frac{2}{n}\sum -(y_i - (mx_i + c)$$

Practice: Partial Derivatives for Linear Equations

Equations for RNN

$$MSE = \frac{1}{n}\sum (y_i - (mx_i + c))^2$$

$$y = mx + c$$

$$\frac{\partial MSE}{\partial m} = \frac{2}{n}\sum -x_i(y_i - (mx_i + c)$$

$$\frac{\partial MSE}{\partial c} = \frac{2}{n}\sum -(y_i - (mx_i + c)$$

Problem

(x1, y1) = (1, 2)
(x2, y2) = (2, 3)
(x3, y3) = (3, 4)

Initial condition
m:1, c:0
$y = x$
Learning rate = 0.01

Practice: Partial Derivatives for Linear Equations

Equations for RNN

$$MSE = \frac{1}{n}\sum (y_i - (mx_i + c))^2$$

$$y = mx + c$$

$$\frac{\partial MSE}{\partial m} = \frac{2}{n}\sum -x_i(y_i - (mx_i + c)$$

$$\frac{\partial MSE}{\partial c} = \frac{2}{n}\sum -(y_i - (mx_i + c)$$

Initial condition

m:1, c:0

$y = x$

Problem

(x1, y1) = (1, 2)

(x2, y2) = (2, 3)

(x3, y3) = (3, 4)

$$MSE = \frac{1}{3}[(2-1)^2 + (3-2)^2 + (4-3)^2] = 1$$

$$\frac{\partial MSE}{\partial m} = \frac{2}{3}[-1(2-1) - 2(3-2) - 3(4-3)] = -4$$

$$\frac{\partial MSE}{\partial c} = \frac{2}{3}[-(2-1) - (3-2) - (4-3)] = -2$$

References

https://youtube.com/playlist?list=PLblh5JKOoLUIzaEkCLIUxQFjPIlapw8nU

Section 01 (massey.ac.nz)