FEED FORWARD NEURAL NETWORK

Asst.Prof.Dr.Supakit Nootyaskool
IT-KMITL

Study map

- 1.Basic programming
  - R-programming
- 2. Perceptron
  - Activity function
- 3. Feed Forward NN
  - Logistic function
- 4. Multi-layer Perceptron
  - XOR gate
- 5. Deep Feed Forward
- 6. Apply DFF
- 7. Recurrent Neural Network
- 8. Apply RNN
- 9. Long / Short Term Memory
- 10. Apply LSTM
- 11. Evolutionary Computation

Learning outcome

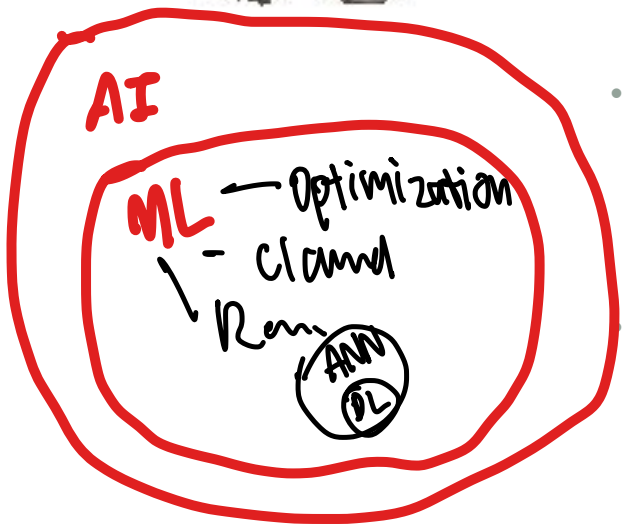
- To known ANN concepts
- To understand activation functions
- To have experience using **neuralnet()** implementing AND/OR gate

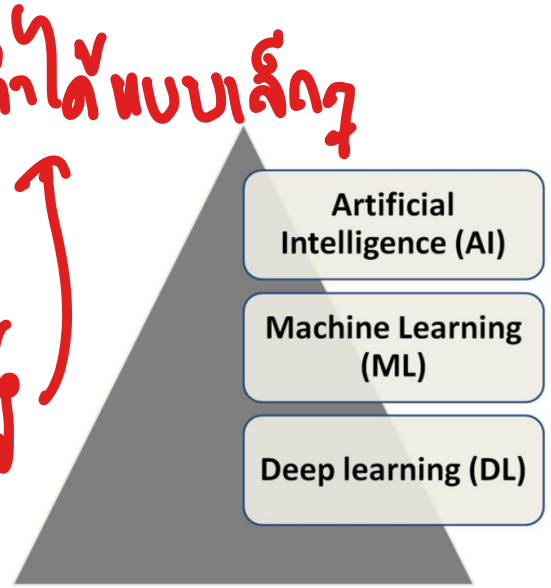
Topics

- ANN concepts and overview
- Neurons, perceptron, and multilayered neural networks
- Bias, weights, activation functions, and hidden layers
- Forward and backpropagation methods
- neuralnet()

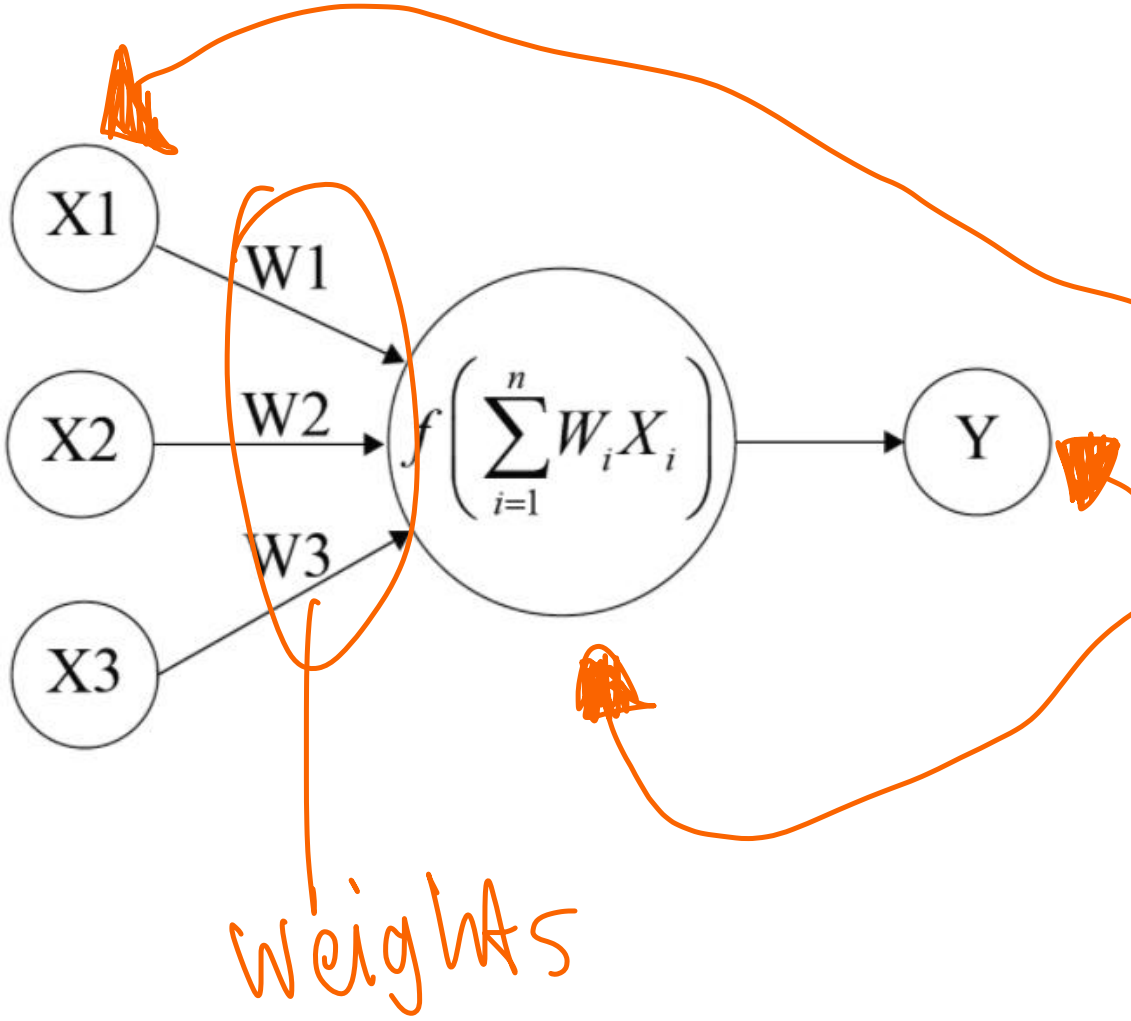
Overview

- The brain is importance organ of the human body.
  - Weighting only 1.5Kilos
  - 86 billion neurons
- Artificial Neural Network (ANN) is the thought the mimicking the function as the human brain
- Machine learning (ML) is a field of study giving the computer power to simulate the human intelligent
- Deep Learning (DL) is complex set of neural networks with more layers of processing, trying to work in image recognition, image classification, etc.





Overview

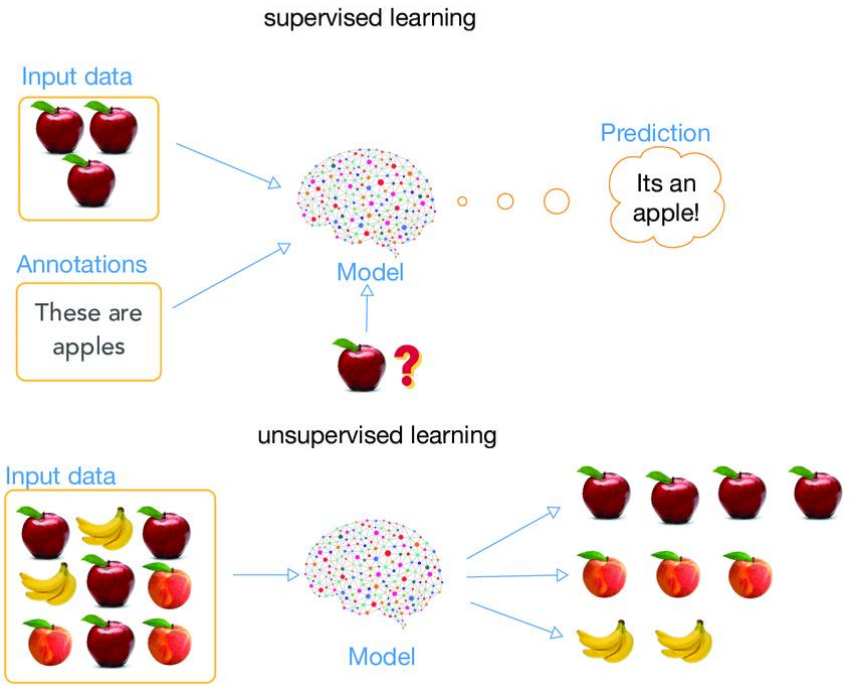


- The output of a neuron is a function of weighted sum of the input plus the bias.
- ANN has 6 parts
  - Input layer
  - Hidden layer
  - Output layer
  - Weights
  - Bias(B) ค่าที่ช่วยเลื่อนกราฟ
  - Activation function

Overview

- Training in Neural Networks
  - Supervised learning is the user that supply
    - Inputs
    - The desired output
  - Unsupervised learning is the user that supply
    - Inputs อย่างเดียว
    - The system identifies the pattern and differences in the input without any external assistance.

classification ออกมา

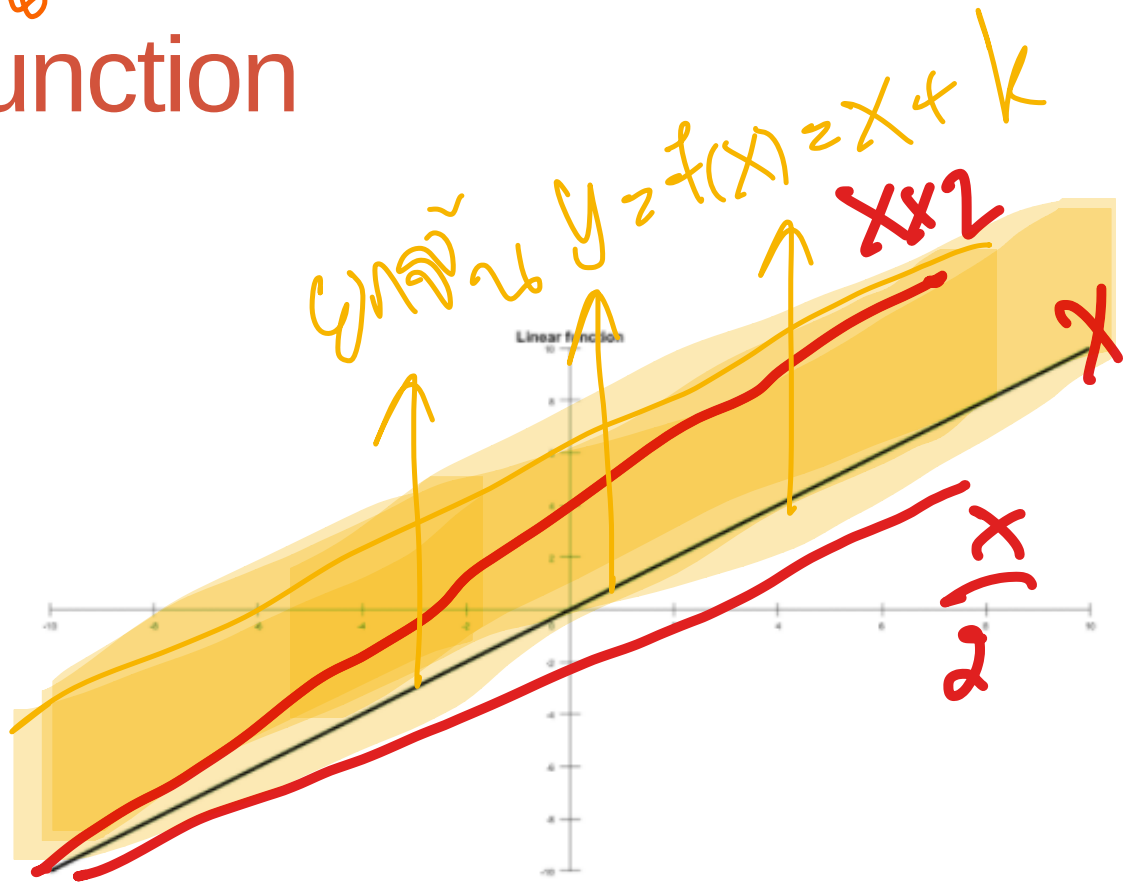


Activation function

- Activation function is a mathematical function that converts the long-range value of input to the output in a small range.
- Many activation function in NN
  - Linear function
  - Unit step activation function (binary function)
  - Sigmoid function
  - Hyperbolic tangent
  - Rectified Linear Unit

Activation function: Linear function

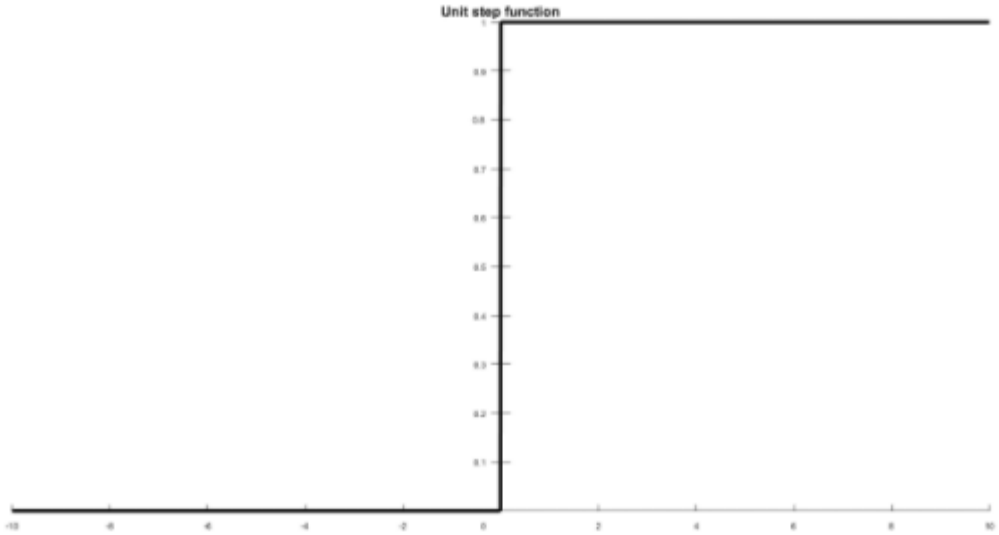
- Output = Input
- The output can be gain or lose the value from the input.
  - Output = Input / 2
  - Output = Input * 2



$$y = f(x) = x$$

Activation function: Unit step function

- The output is divided small levels such as
  - 1 / 0
  - True / False
  - High / Mid / Low



$$f(x) = o \; when \; x < o,$$

$$1 \; when \; x >= o$$

Activation function: Sigmoid function

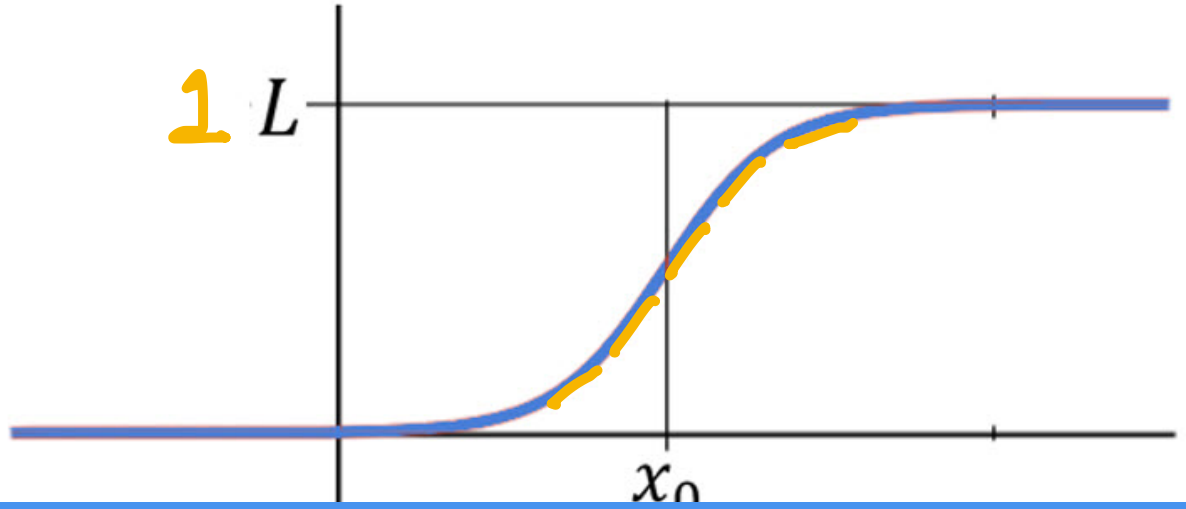
- Characteristic S-shaped curve,

  S-curve

- The class of sigmoid function
  - Logistic function
  - Hyperbolic tangent
  - Softmax function

$$f(x) = \frac{L}{1 + e^{-k(x - x_0)}}$$

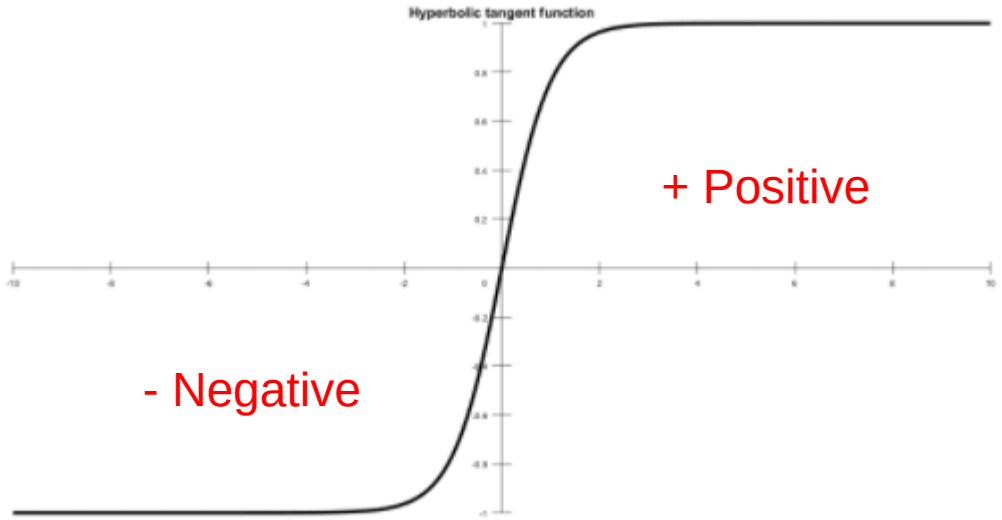
x_0 = x value of midpoint
L = maximum value
k = growth rate



พัฒนามาจาก Sigmoid

Activation function: hyperbolic tangent

- S-curve is in the positive and negative part.

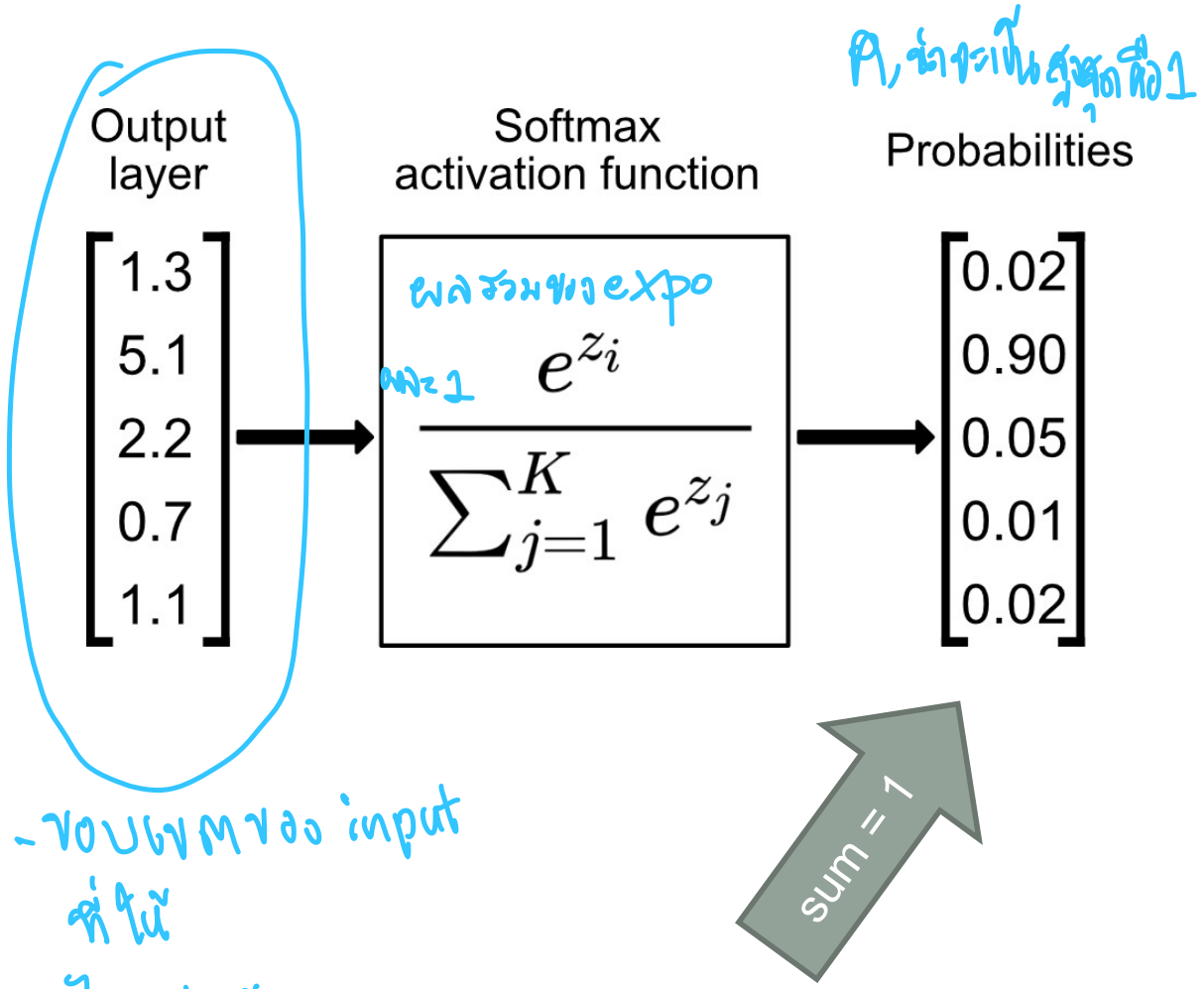


ถ้าสัญญาณติดลบ
ก็พลิกค่าจากสีๆ
Scale/ยกให้ค่าสัน สบ/ยกขึ้นมา

$$f(x) = \tanh(x)$$

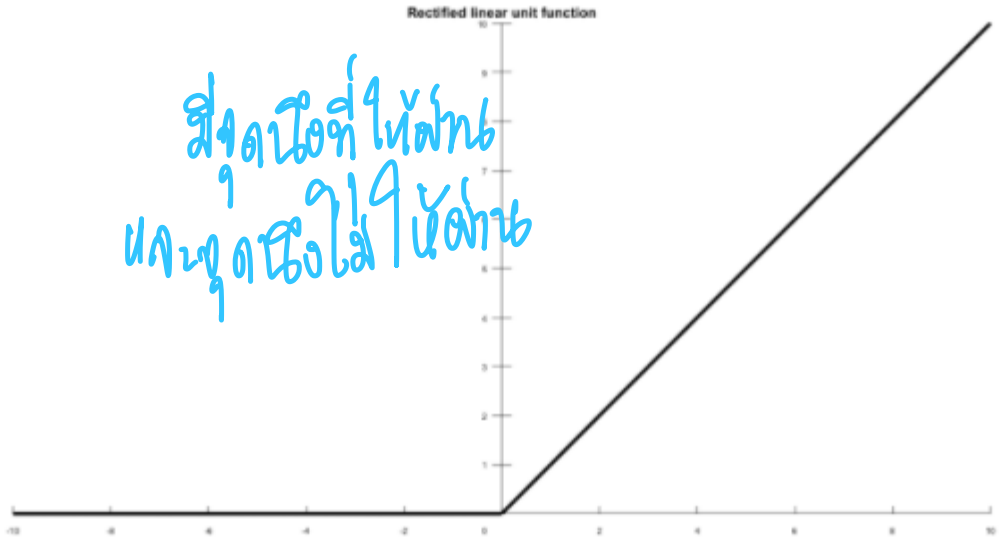
Activation function: Softmax function

- The output has from the normalization of the input data



Activation function: Rectified Linear Unit

- ReLU activation function
- The pass-through input to the output when the input value is in the positive part.
- The output can be gain or lose the value from the input.



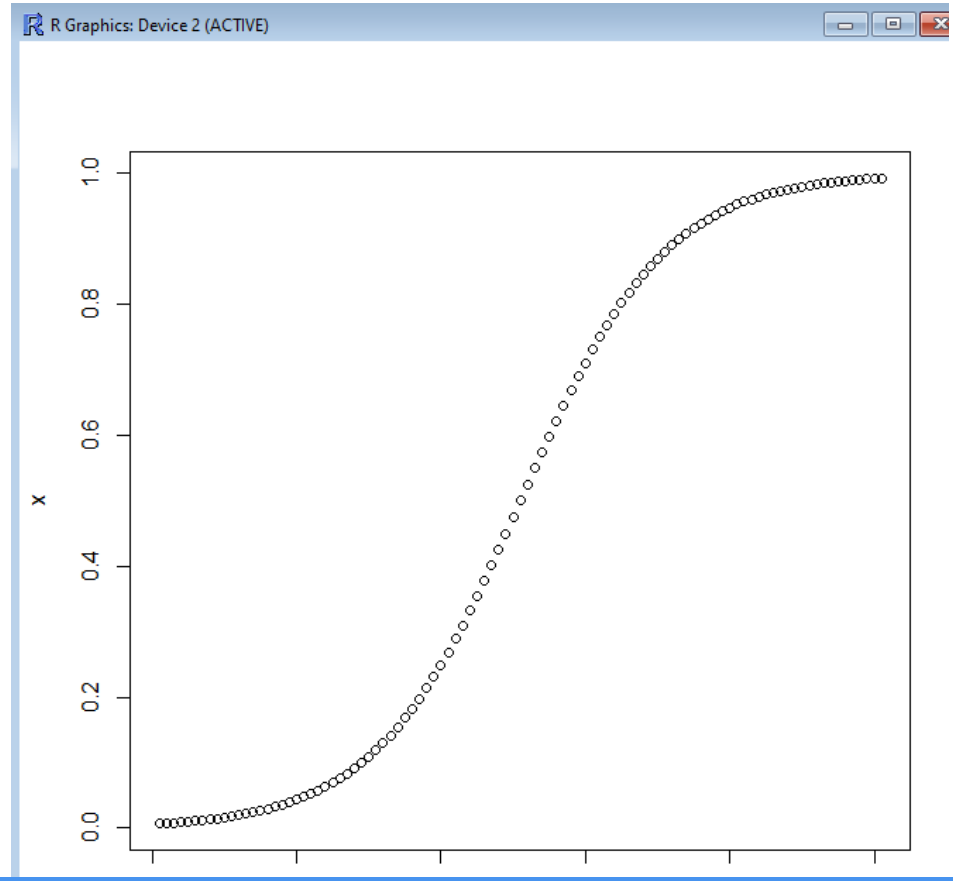
$$f(x) = o \; when \; x < o,$$

$$x \; when \; x >= o$$

Activity 3.1 Test activation function

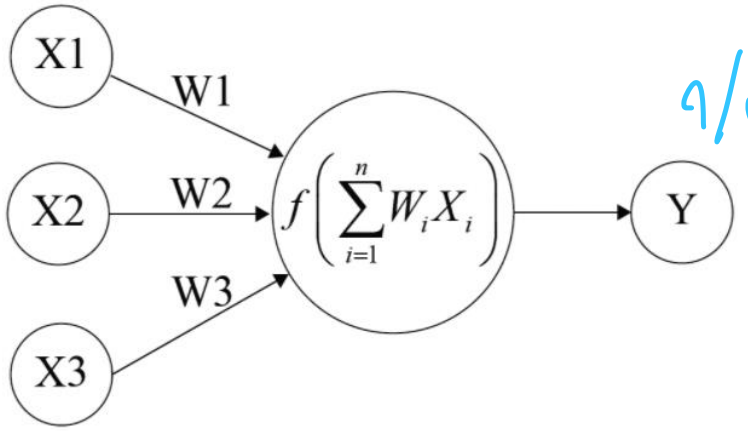
Given student plot output of four activation functions, by running R code below.

```
logistic = function (x)
{
    1/(1 + exp(-x))
}

x = seq(-5,5,0.1)
maxx = length(x)
y = 1:maxx
x = logistic(x)
plot(y, x)
```

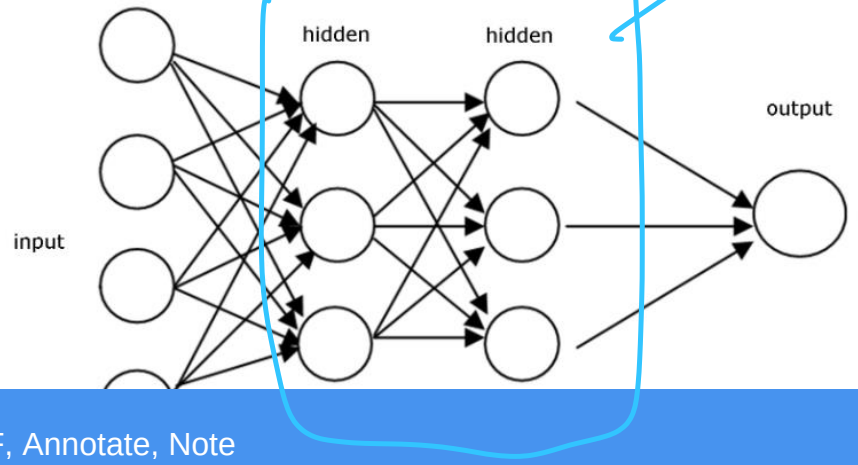


Perceptron vs Multi-Layer Perceptron

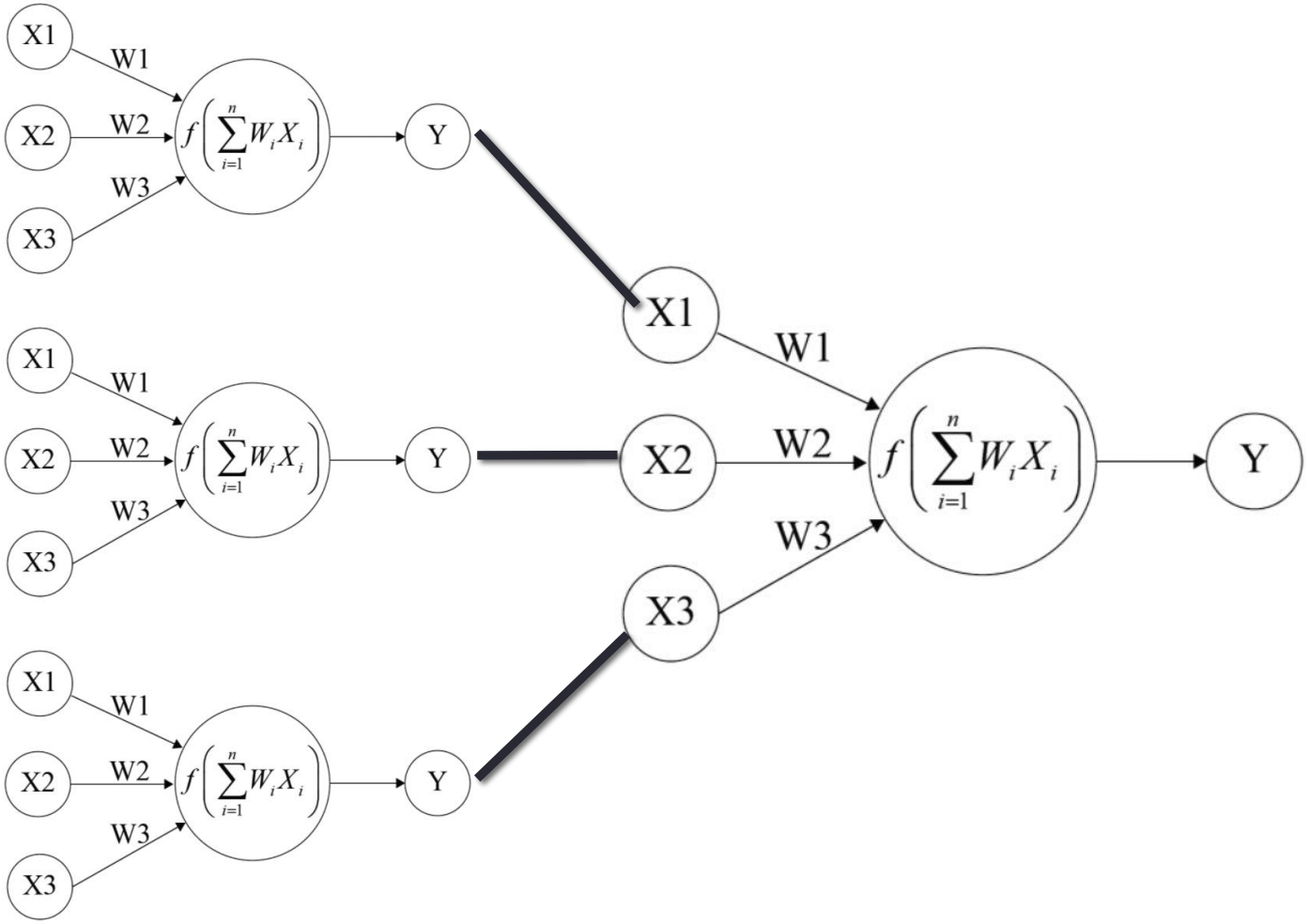
- Perceptron
  - A single neuron
  - Set of input is two categories
  - Usually, it uses step function



- Multi-Layer Perceptron
  - Combined many layer of the perceptron
  - Most of widely used architecture for Neural Network.



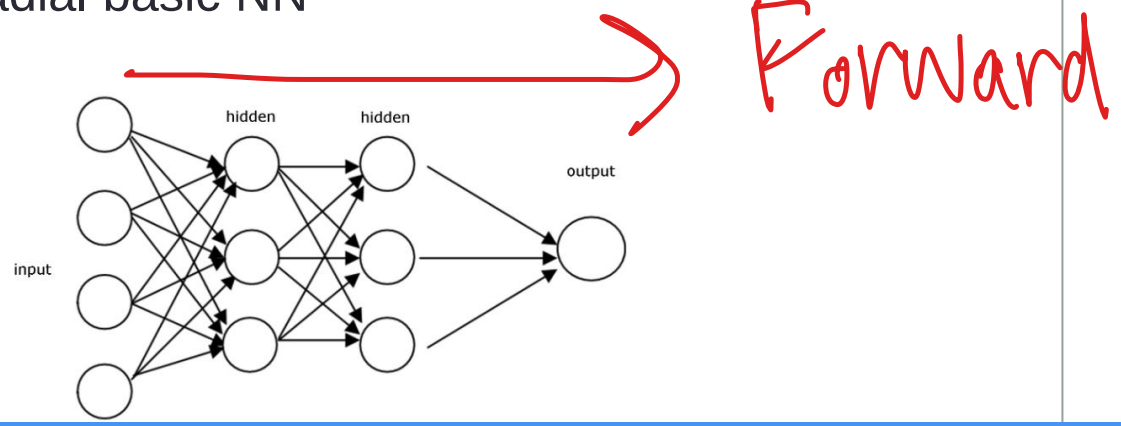
A large model is created from small models



Feed-**Forward** and Feed-**Back** networks

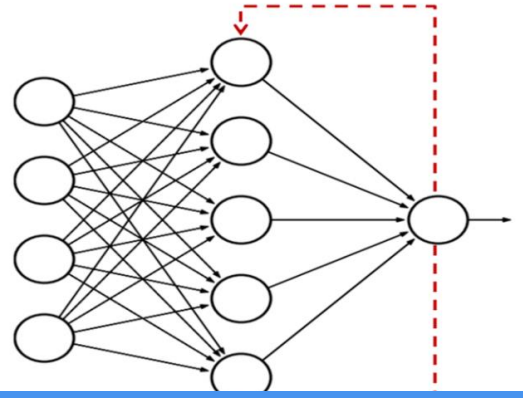
Feed-forward neural network has

- One direction ทิศทางเดียว
- Input signals are feed to the input layer.
- Most of feed-forward network
  - MLP
  - Radial basic NN

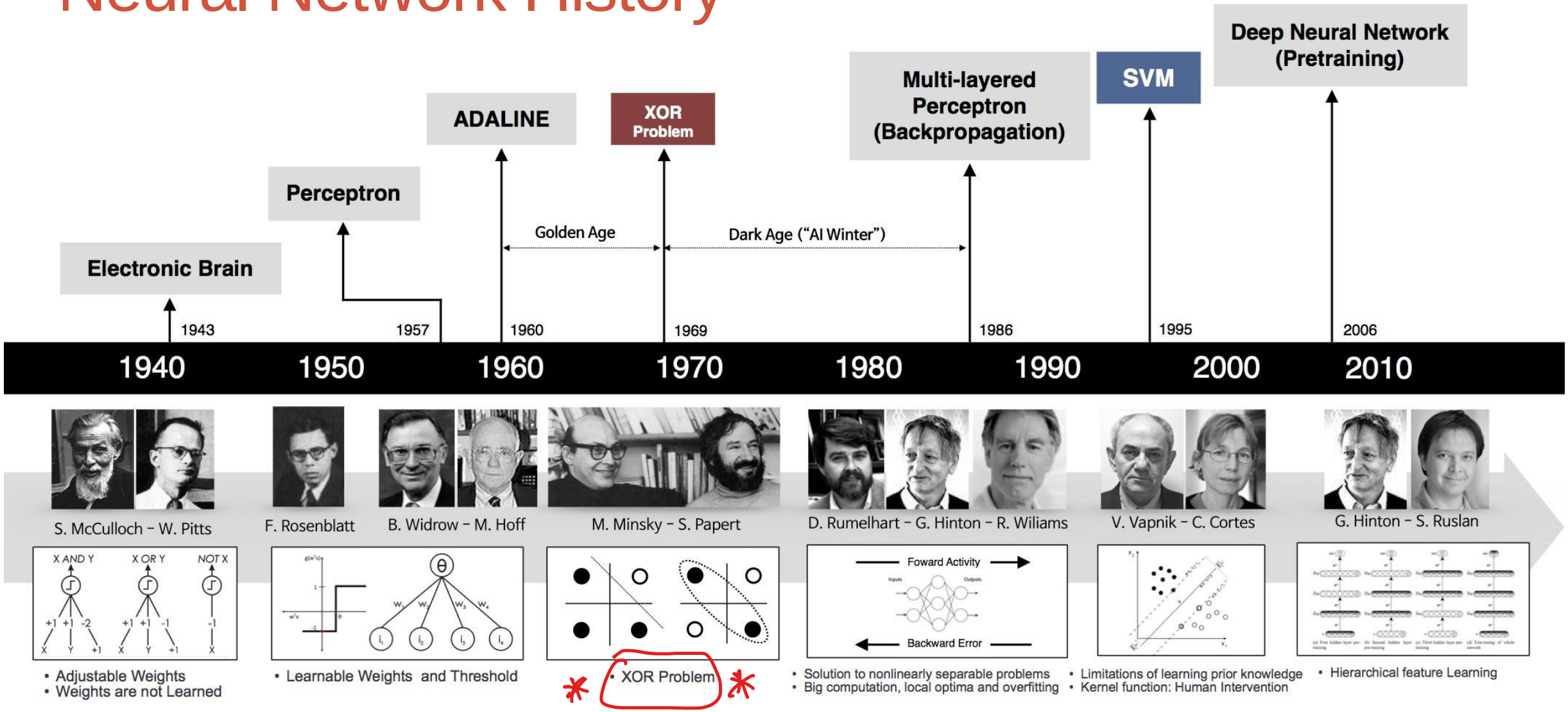


Feedback networks

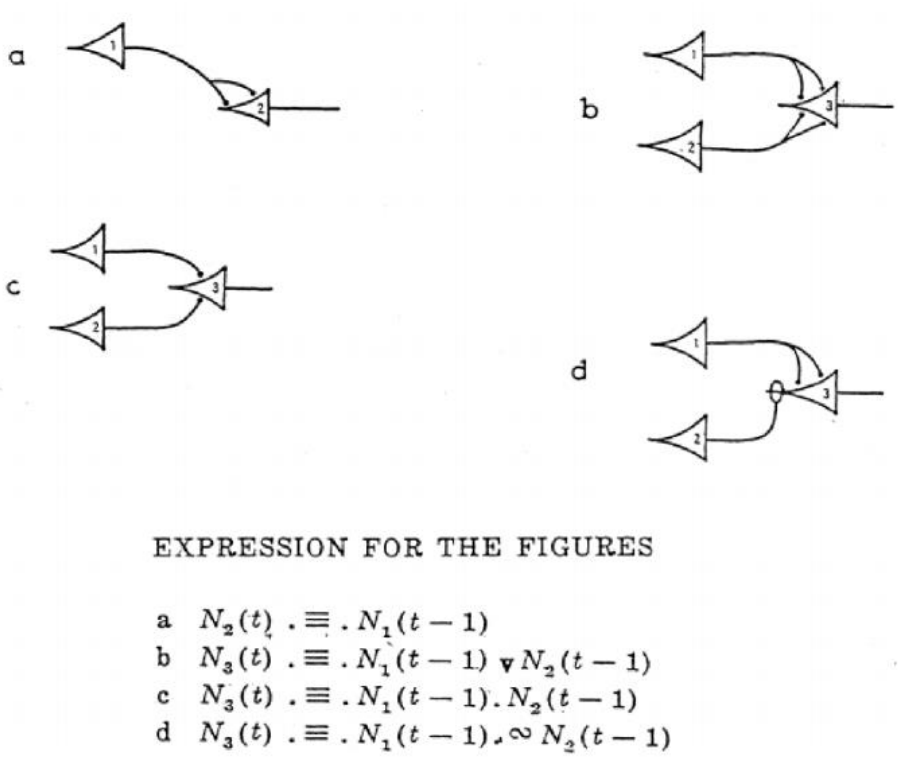
- A neural or layer has received the processed signal from the next.
- Most of feedback network
  - Elman
  - Hopfield



Neural Network History



1940, Electronic Brain



AND, OR, NOT

- In 1943, McCulloch and Pitts tried to understand how the brain could produce highly complex patterns by using many basic cells that are connected.
- These basic brain cells are called neurons.
- The McCulloch and Pitts model of a neuron, called an MCP neuron

- Research Questions/Answers
  - Adjustable weights
  - Weights are not learned

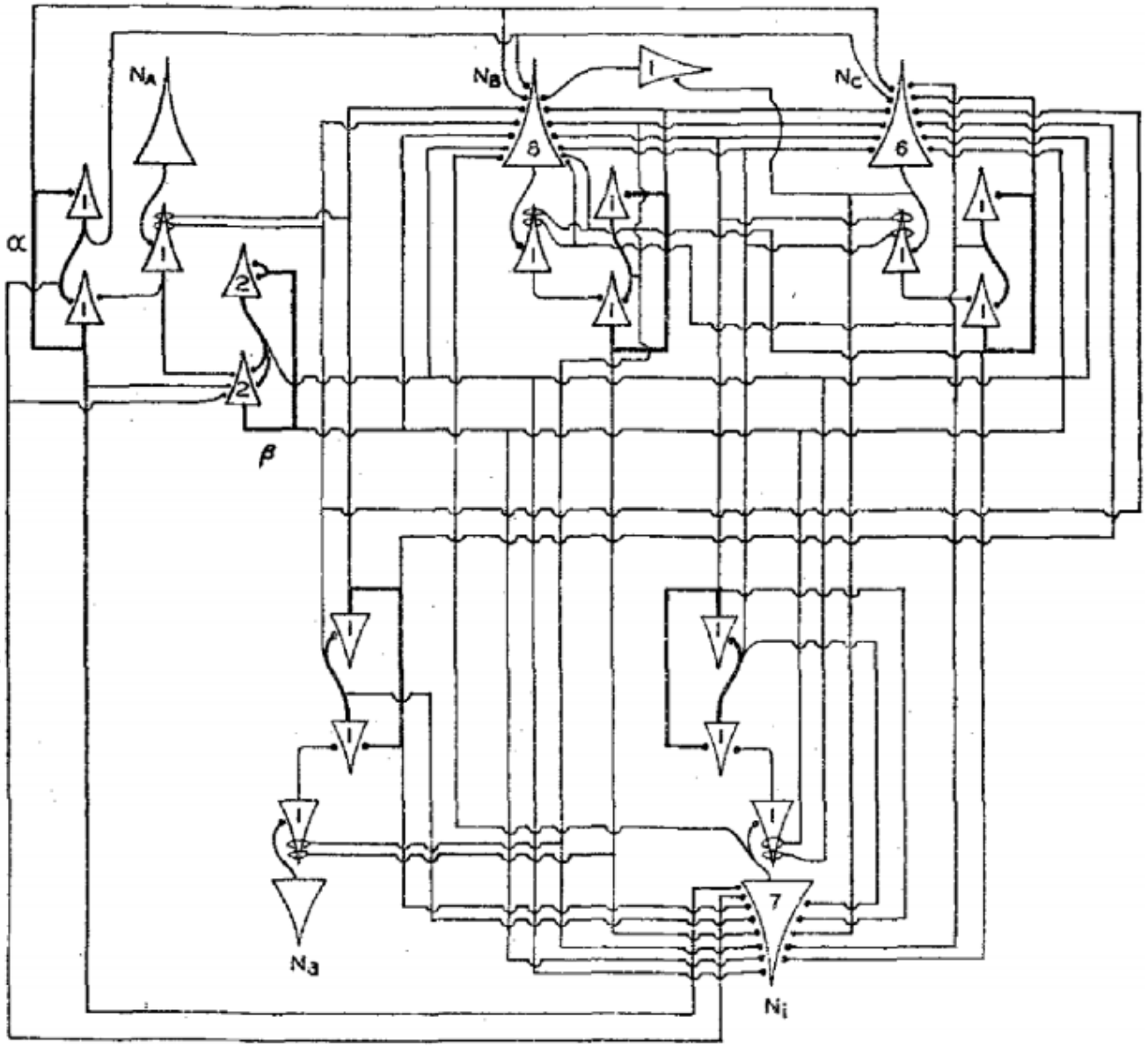
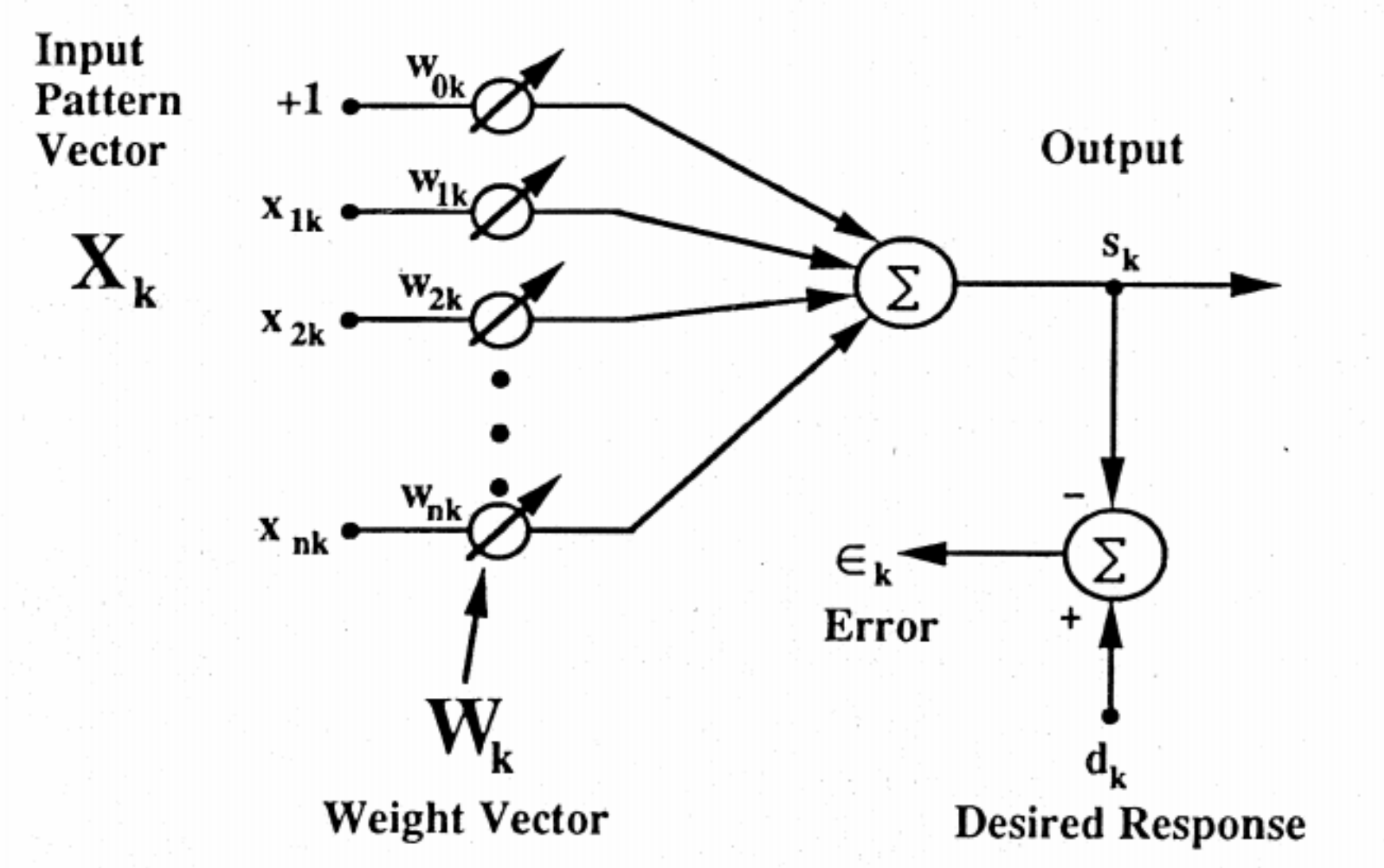


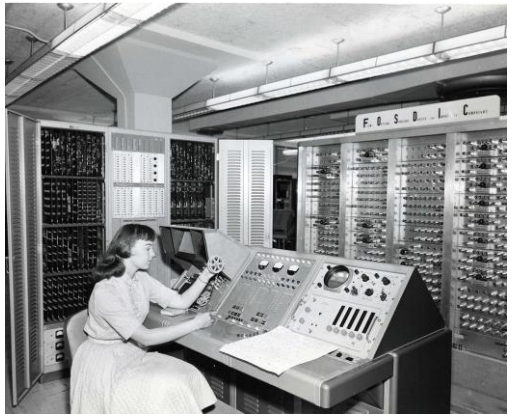
Fig. 7. Nicolas Rashevsky's complex neural network example (from [46, p. 32])

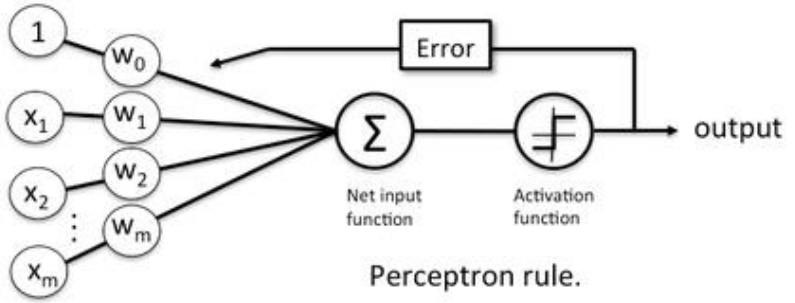
1950, Perceptron



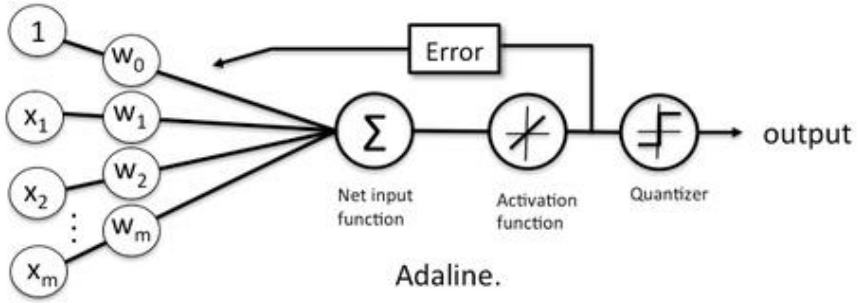
History: 1960s, ADALINE

- ADALINE is aberration from ADAptive LInear NEural.
- In the past, the computer is different from today.
- Main problem is how to adjust weight of the perceptron.
- ADALINE uses a feedback after the activation function and before a quantize function comparing for weights adjustment

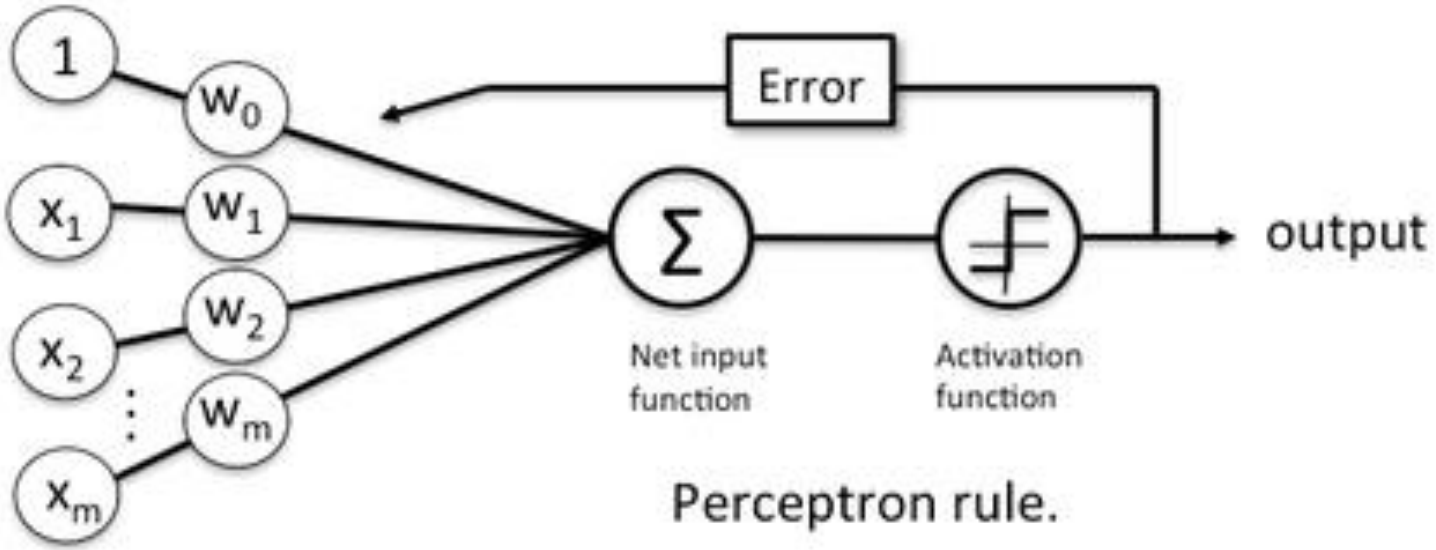




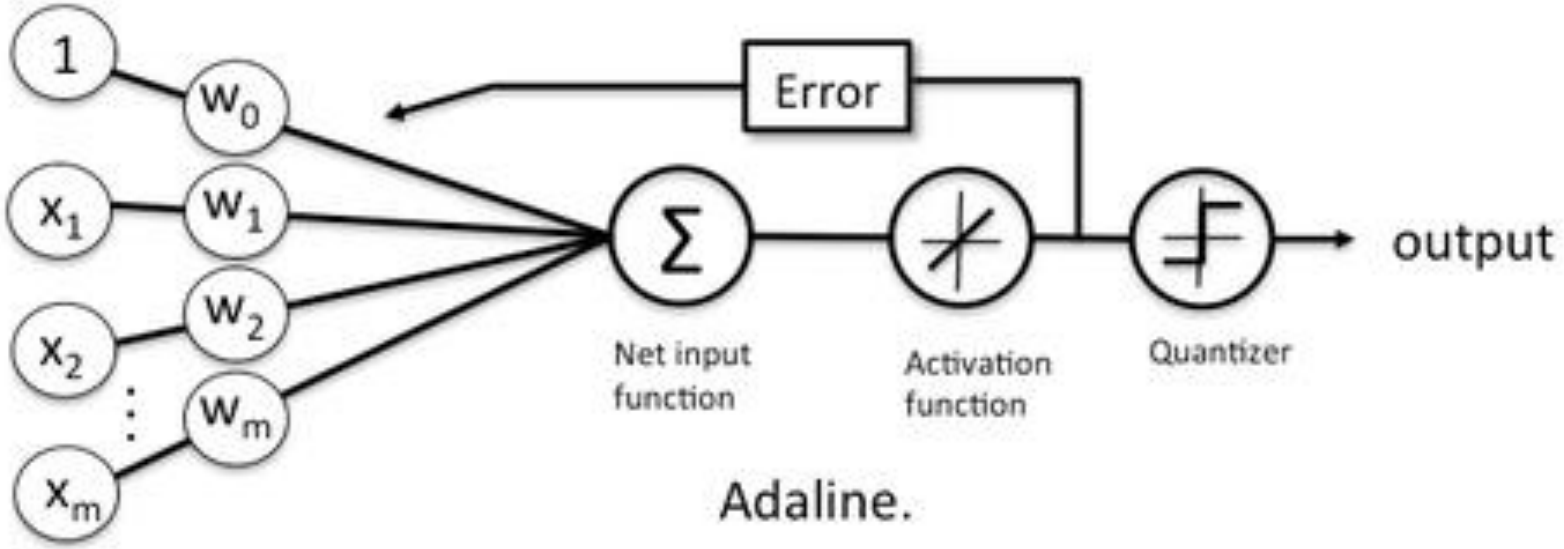
Perceptron rule.



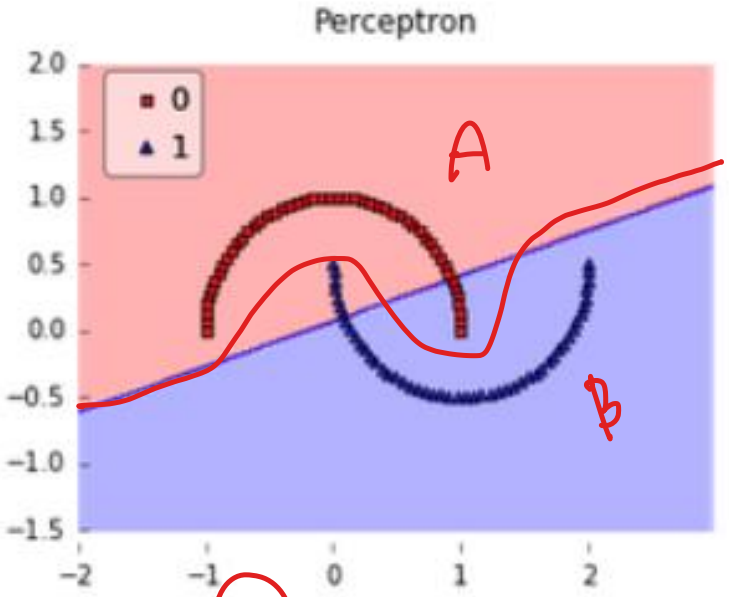
Adaline.

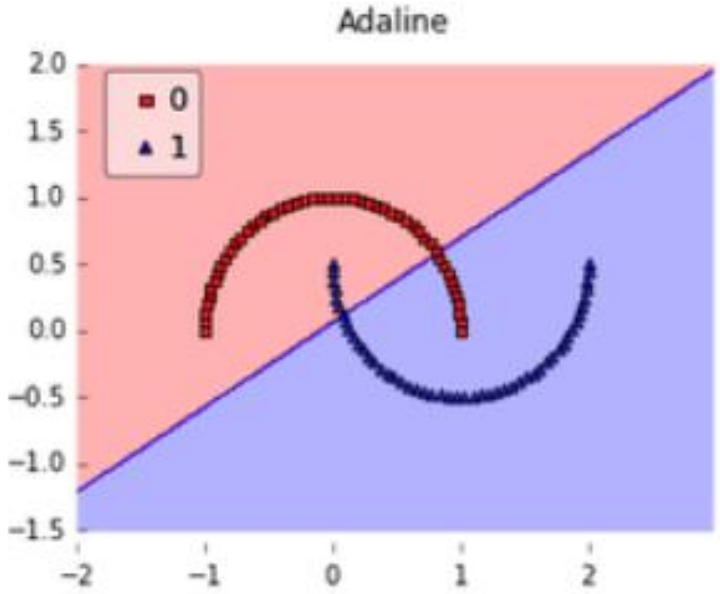


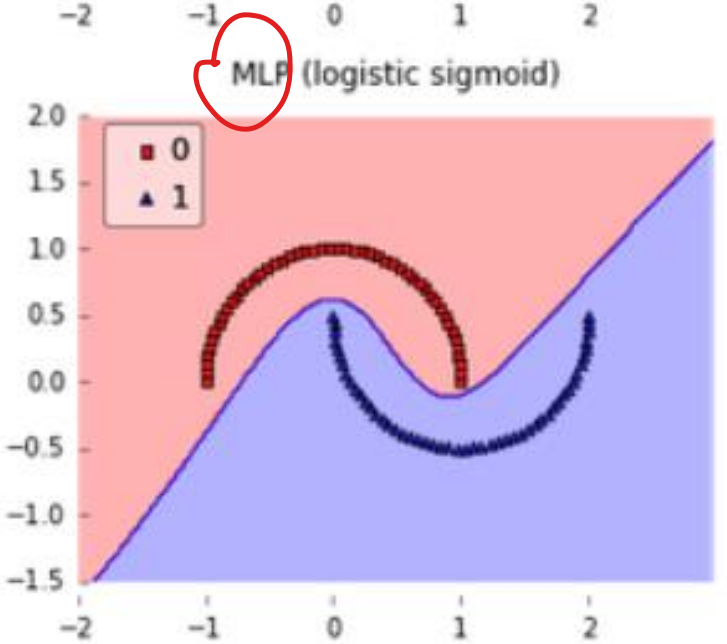
Perceptron rule.



Adaline.







perceptron Node 1ea

Multilayer

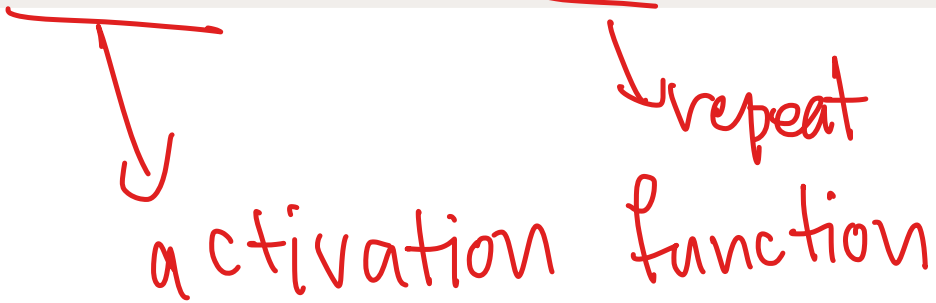
XOR problems

Library nerualnet()

Package	neuralnet
Description	Training of NN using backpropagation Allow user adjust activation function,

Usage

- neuralnet(formula, data, hidden = 1, threshold = 0.01, stepmax = 1e+05, act.fct = "logistic", rep=1 …)



Installation neuralnet()

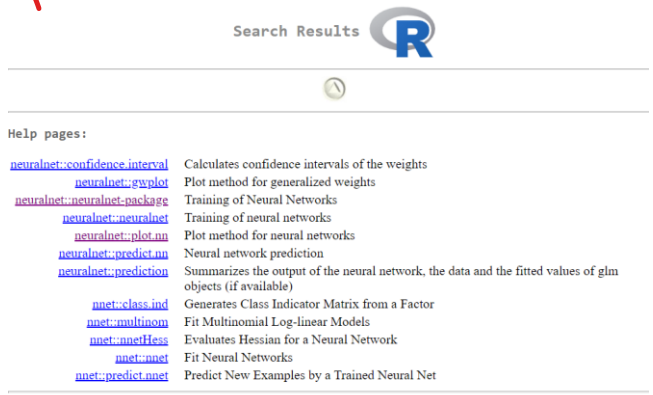
ใช้แค่ครั้งแรก ครั้งเดียว

There are command steps in R

1. install.packages("neuralnet") #Install "nerualnet" in the system.
2. library("neuralnet") #call "neuralnet"
3. ??neuralnet #open help

เรียกเมื่อต้องการใช้ เรียกทุกครั้งที่เปิดโปรแกรม

```
> install.packages("neuralnet")
Warning: package 'neuralnet' is in use and will not be installed
> update.packages("neuralnet")
> library("neuralnet")
> ??neuralnet
```


Search Results

Help pages:

neuralnet::confidence.interval	Calculates confidence intervals of the weights
neuralnet::gwplot	Plot method for generalized weights
neuralnet::neuralnet-package	Training of Neural Networks
neuralnet::neuralnet	Training of neural networks
neuralnet::plot.nn	Plot method for neural networks
neuralnet::predict.nn	Neural network prediction
neuralnet::prediction	Summarizes the output of the neural network, the data and the fitted values of glm objects (if available)
nnet::class.ind	Generates Class Indicator Matrix from a Factor
nnet::multinom	Fit Multinomial Log-linear Models
nnet::nnetHess	Evaluates Hessian for a Neural Network
nnet::nnet	Fit Neural Networks
nnet::predict.nnet	Predict New Examples by a Trained Neural Net

Activity 3.2 Train AND gate with neuralnet()

อ้างอิงจาก Data frame

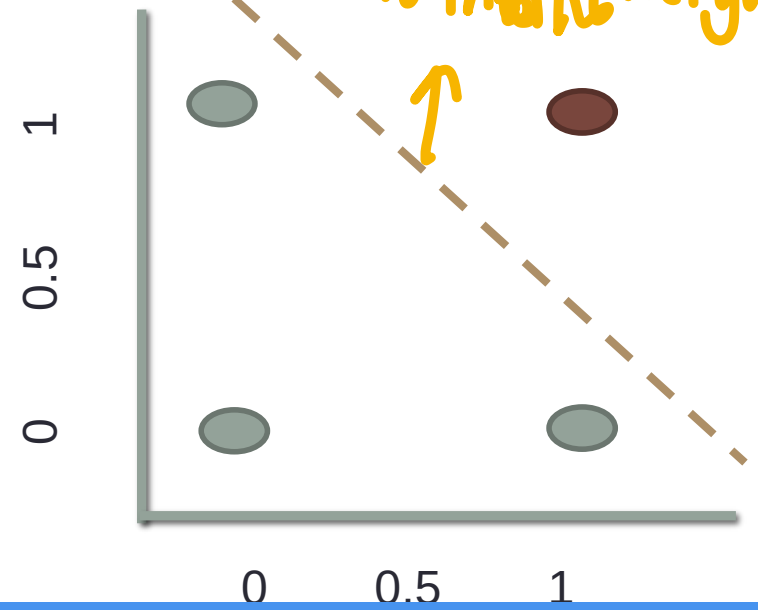
A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

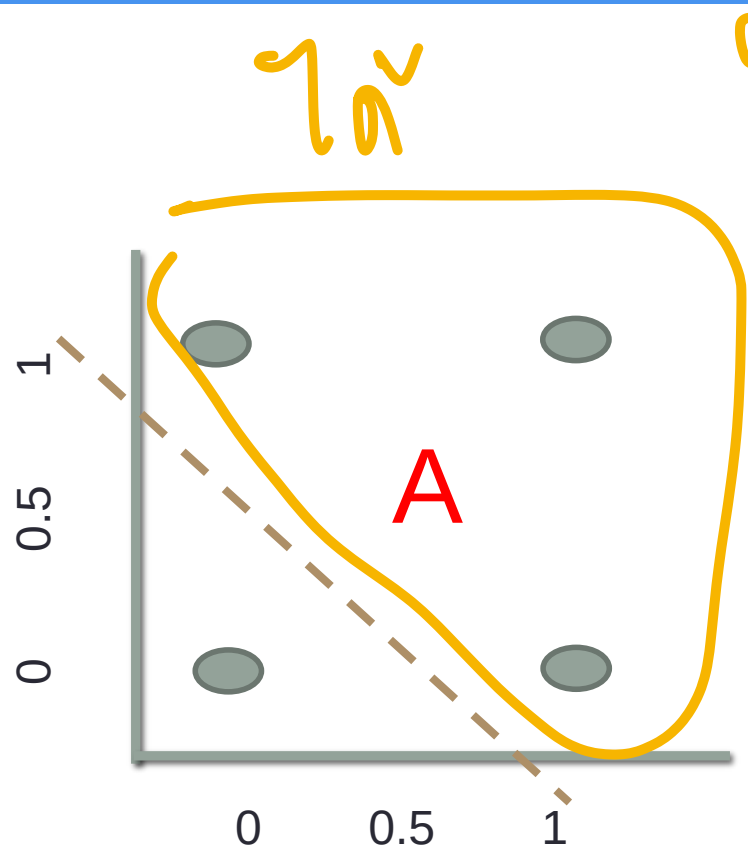
R Console

```
> AND = c(0,0,0,1)
> truthtable = expand.grid(c(0,1), c(0,1))
> AND.data <- data.frame(truthtable, AND)
> print(AND.data)
  Var1 Var2 AND
1    0    0   0
2    1    0   0
3    0    1   0
4    1    1   1
```

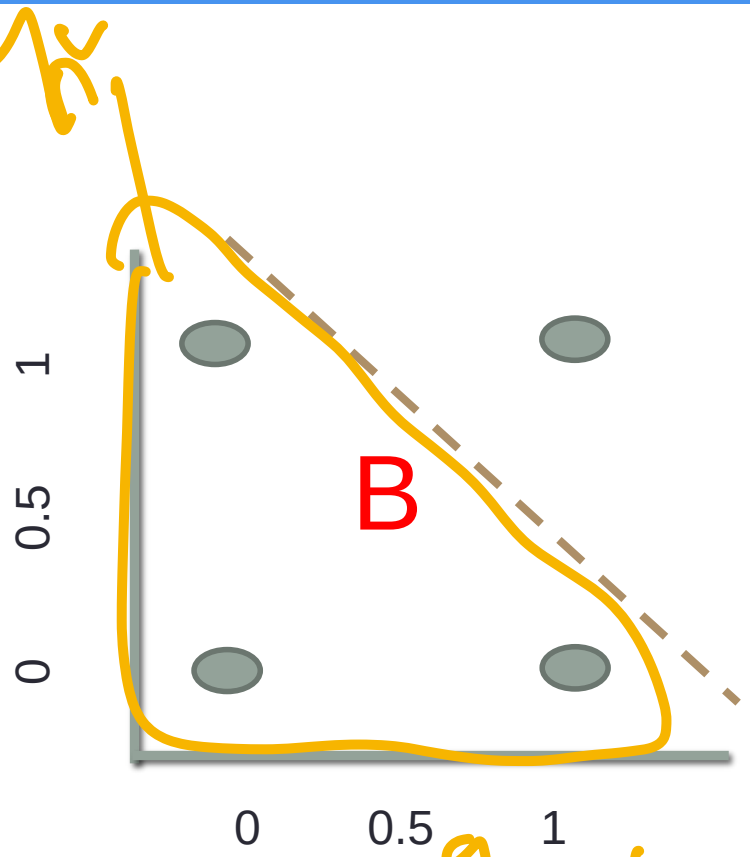
- AND = c(0,0,0,1)
- truthtable = expand.grid(c(0,1), c(0,1))
- AND.data <- data.frame(truthtable, AND)
- print(AND.data)

เส้นที่ได้เกิดจาก weight



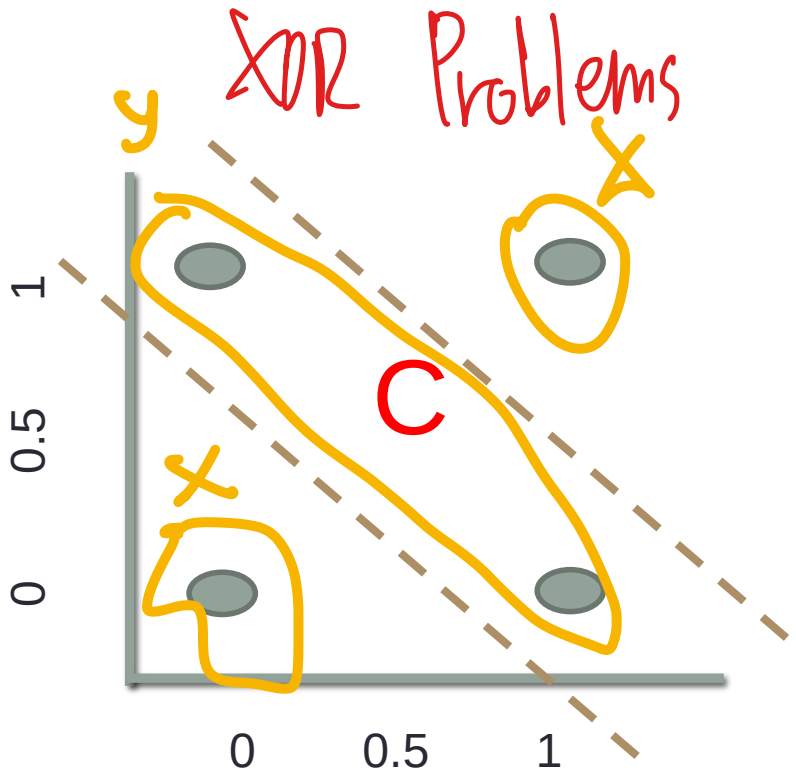


ได้



ก็ได้

perceptron จำใช้ดี



perceptron error ของข้าง

Activity 3.2(2) Train AND gate with neuralnet()

- net <- neuralnet(AND~Var1+Var2, AND.data, hidden=0, rep=1)
- print(net)

```
> print(net)
$call
neuralnet(formula = AND ~ V
    rep = 1)

$response
  AND
1   0
2   0
3   0
4   1

$covariate
     Var1 Var2
[1,]    0    0
[2,]    1    0
[3,]    0    1
[4,]    1    1

$model.list
$model.list$response
[1] "AND"
```

```
$model.list$variables
[1] "Var1" "Var2"


$err.fct
function (x, y)
{
    1/2 * (y - x)^2
}
<bytecode: 0x00000000151ae128>
<environment: 0x000000000ee42798>
attr(,"type")
[1] "sse"

$act.fct
function (x)
{
    1/(1 + exp(-x))
}
<bytecode: 0x00000000151b29b0>
<environment: 0x000000000ee3ee00>
attr(,"type")
```

```
$data
  Var1 Var2 AND
1    0    0   0
2    1    0   0
3    0    1   0
4    1    1   1

$exclude
NULL

$net.result
$net.result[[1]]
            [,1]
[1,] -0.2408318
[2,]  0.2506263
[3,]  0.2510820
[4,]  0.7425401
```

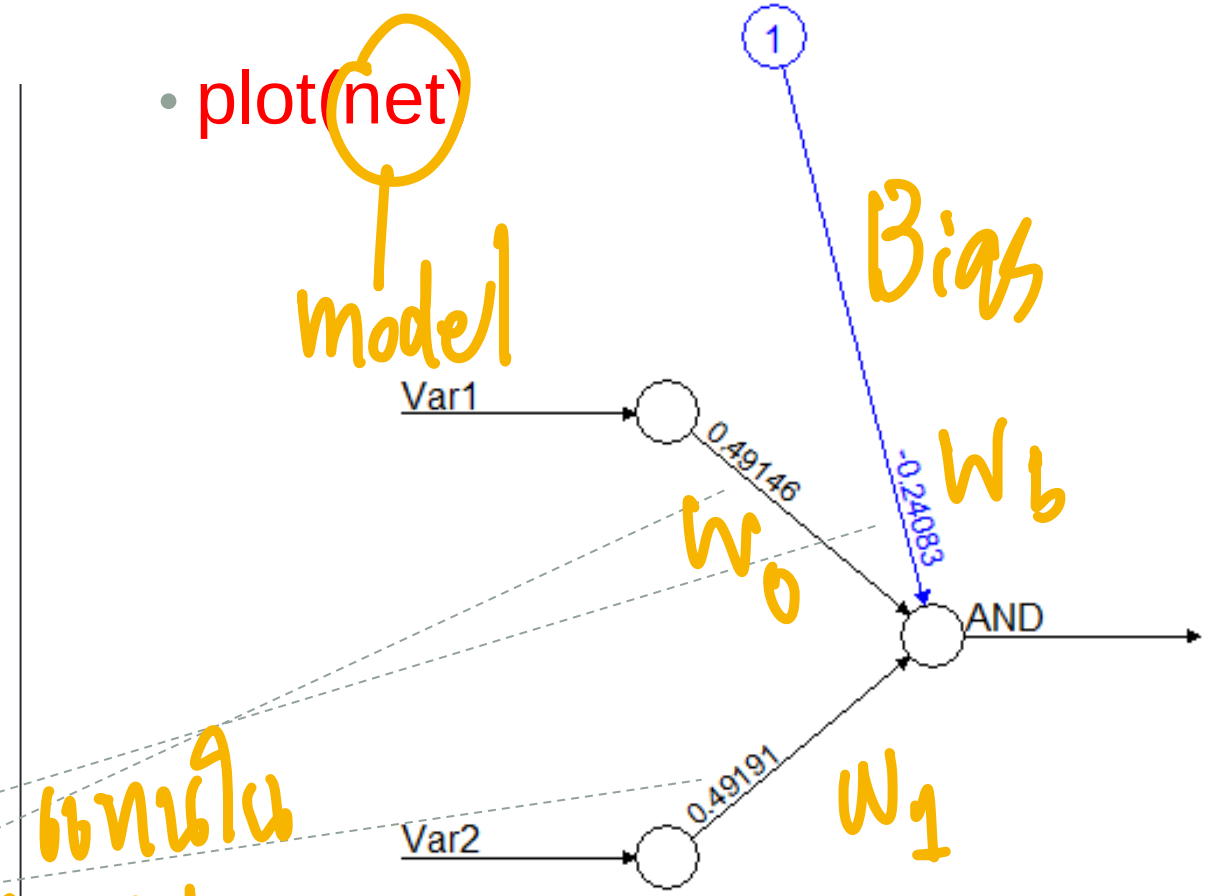
Activity 3.2(3) Train AND gate with neuralnet()

```
$startweights
$startweights[[1]]
$startweights[[1]][[1]]
            [,1]
[1,] -0.1119942
[2,] -0.7498539
[3,] -0.7493983
```

```
$result.matrix
                              [,1]
error                 0.125070634
reached.threshold     0.006833508
steps                65.000000000
Intercept.to.AND     -0.240831818
Var1.to.AND           0.491458166
Var2.to.AND           0.491913796

attr(,"class")
[1] "nn"
```

- plot(net)

model



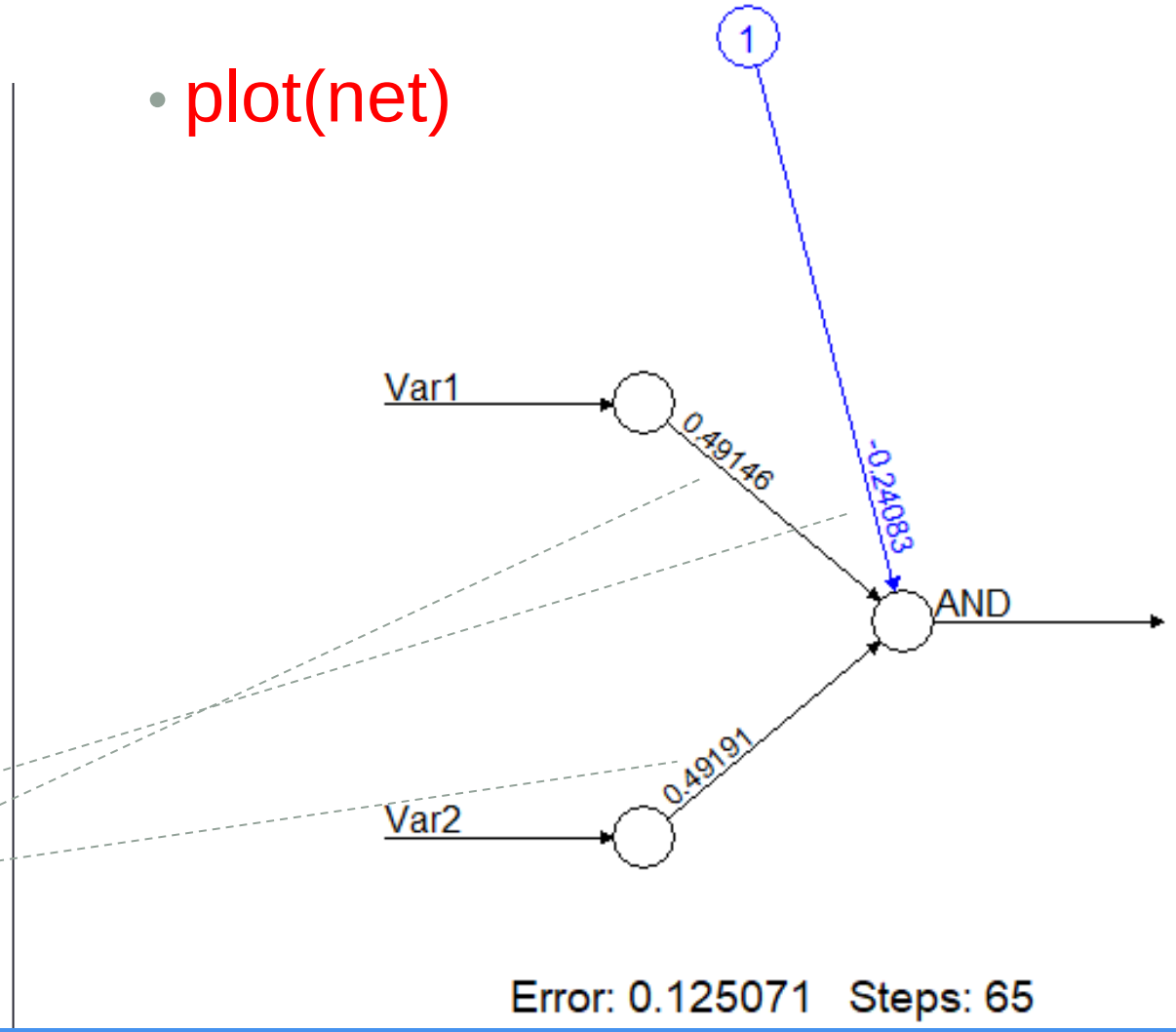
w_0 w_1 w_2

แทนใน Model ใช้งานได้เลยจ้า

Activity 3.2(4) Train AND gate with neuralnet()

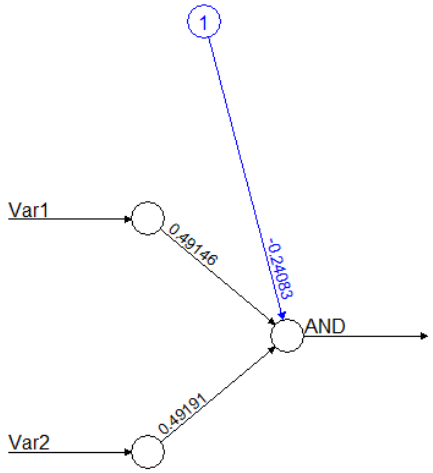
```
$startweights
$startweights[[1]]
$startweights[[1]][[1]]
            [,1]
[1,] -0.1119942
[2,] -0.7498539
[3,] -0.7493983



$result.matrix
                              [,1]
error                 0.125070634
reached.threshold     0.006833508
steps                65.000000000
Intercept.to.AND     -0.240831818
Var1.to.AND           0.491458166
Var2.to.AND           0.491913796

attr(,"class")
[1] "nn"
```

- plot(net)



Activity 3.3 Predict the model AND gate

```
$startweights
$startweights[[1]]
$startweights[[1]][[1]]
            [,1]
[1,] -0.1119942
[2,] -0.7498539
[3,] -0.7493983
```



Error: 0.125071 Steps: 65

```
$result.matrix
                          [,1]
error                0.125070634
reached.threshold    0.006833508
steps               65.000000000
Intercept.to.AND    -0.240831818
Var1.to.AND          0.491458166
Var2.to.AND          0.491913796

attr(,"class")
[1] "nn"
```

```
var1 = c(0,1,0,1)
var2 = c(0,0,1,1)
datatest = data.frame(var1,var2)
pred <- predict(model, datatest)
pred
```

สมมุติ

net

Activity 3.2 – 3.3 ALL CODE

```
library("neuralnet")


AND = c(0,0,0,1)
truthtable = expand.grid(c(0,1), c(0,1))
AND.data <- data.frame(truthtable, AND)
print(AND.data)
net <- neuralnet( AND~Var1+Var2, AND.data,
hidden=0, rep=5)
print(net)
plot(net)
```

```
logistic = function (x)
{
    1/(1 + exp(-x))
}
var1 = c(0,1,0,1)
var2 = c(0,0,1,1)
n = (net$result.matrix[4] +
            net$result.matrix[5] * var1 +
            net$result.matrix[6] * var2)
logistic(n)
```

not OR not AND

Activity 3.4 Test OR, NOR, NAND, XOR

ขอหน้าเดียวพอ!!

ต้องทำส่ง

รหัสนักศึกษา

Key nndL64 submit

modifies ได้จาก 3.1, 3.2 จะรวมหรือแยกก็ได้

เขียนให้เสร็จ หลัง แคป ผลลัพธ์ + สรุปลง word

ส่ง pdf กำหนดส่งก่อนเรียนครั้งหน้า

ส่งที่ 161.246.38.75 ตรงช่อง NNDL upload

ชื่อ

XOR NAND

OR NOR

สรุป

Summary