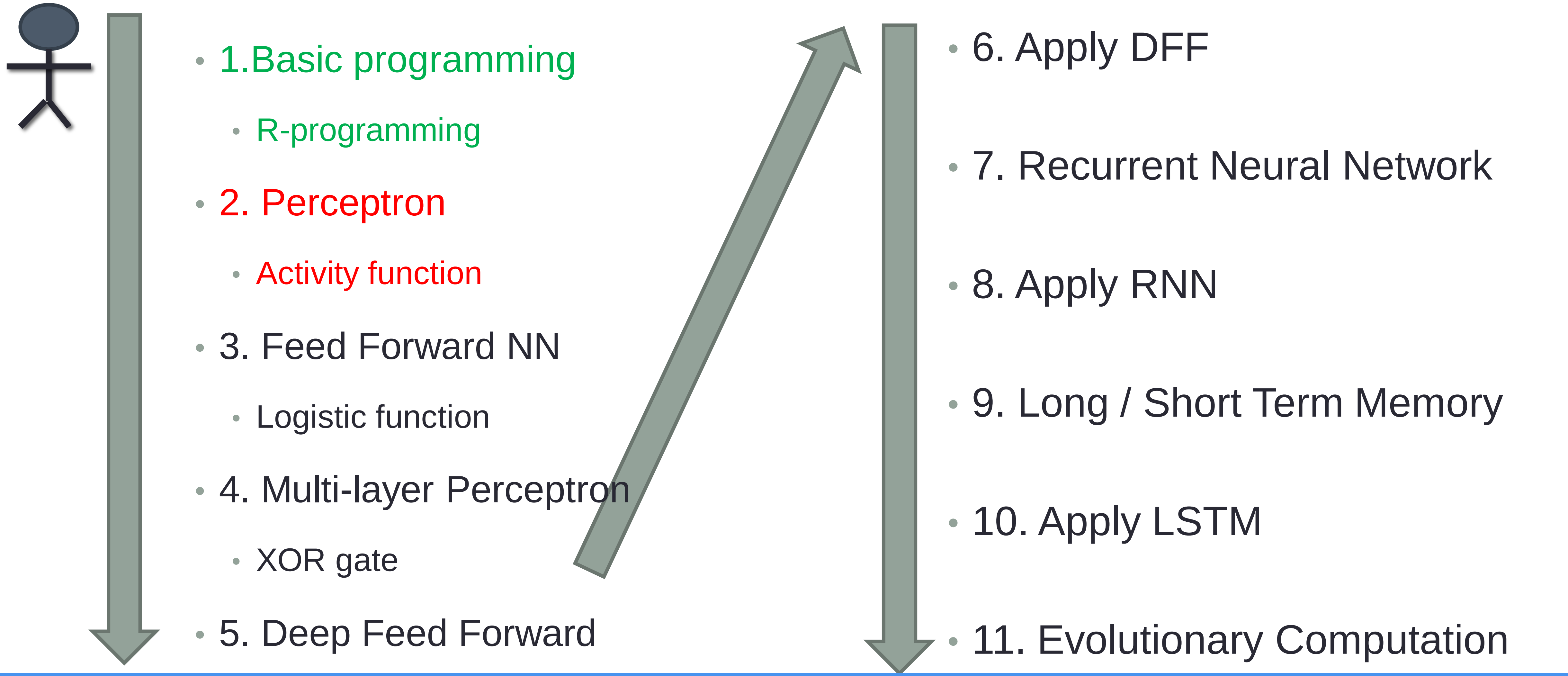
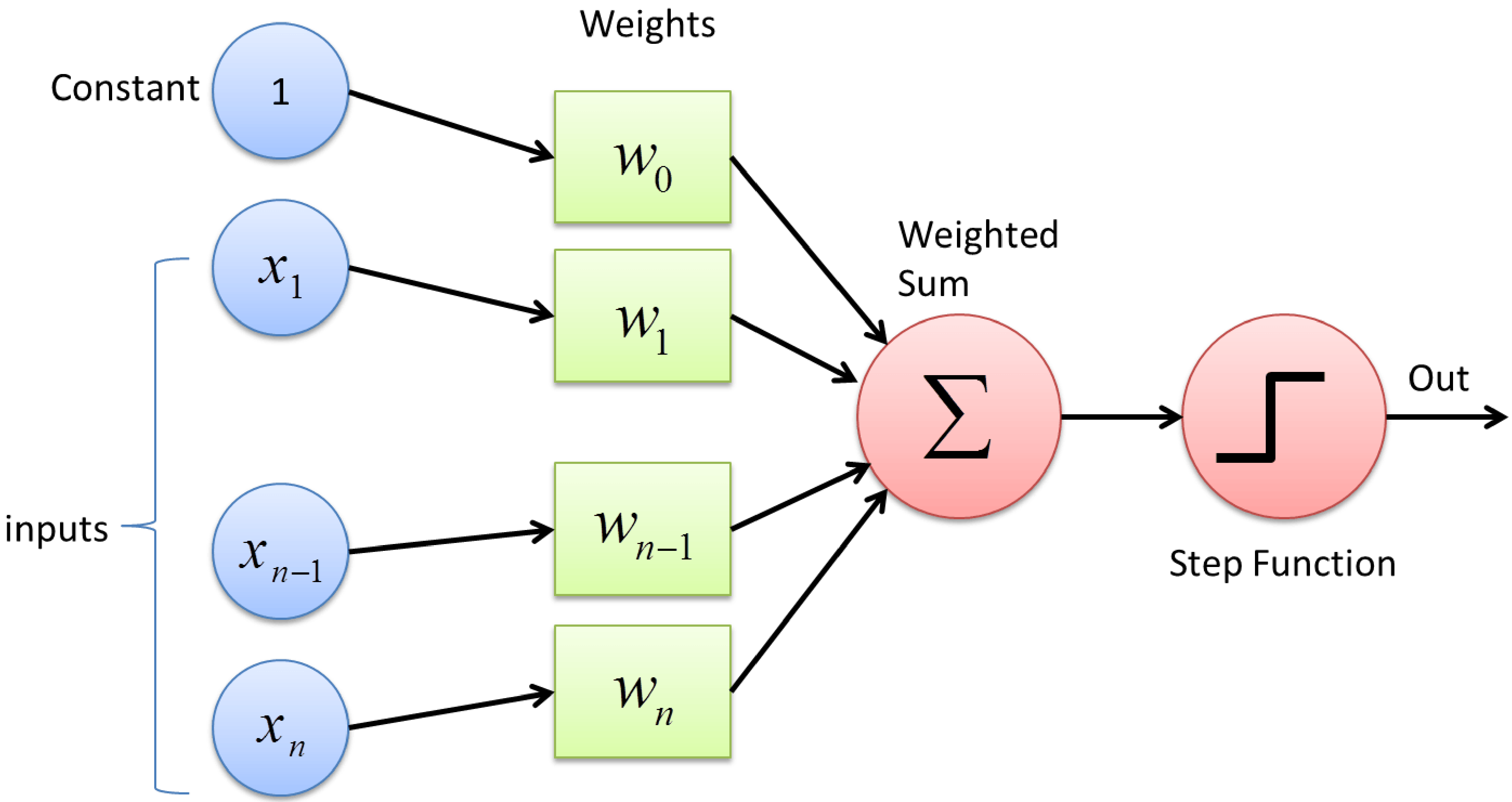
PERCEPTRON

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

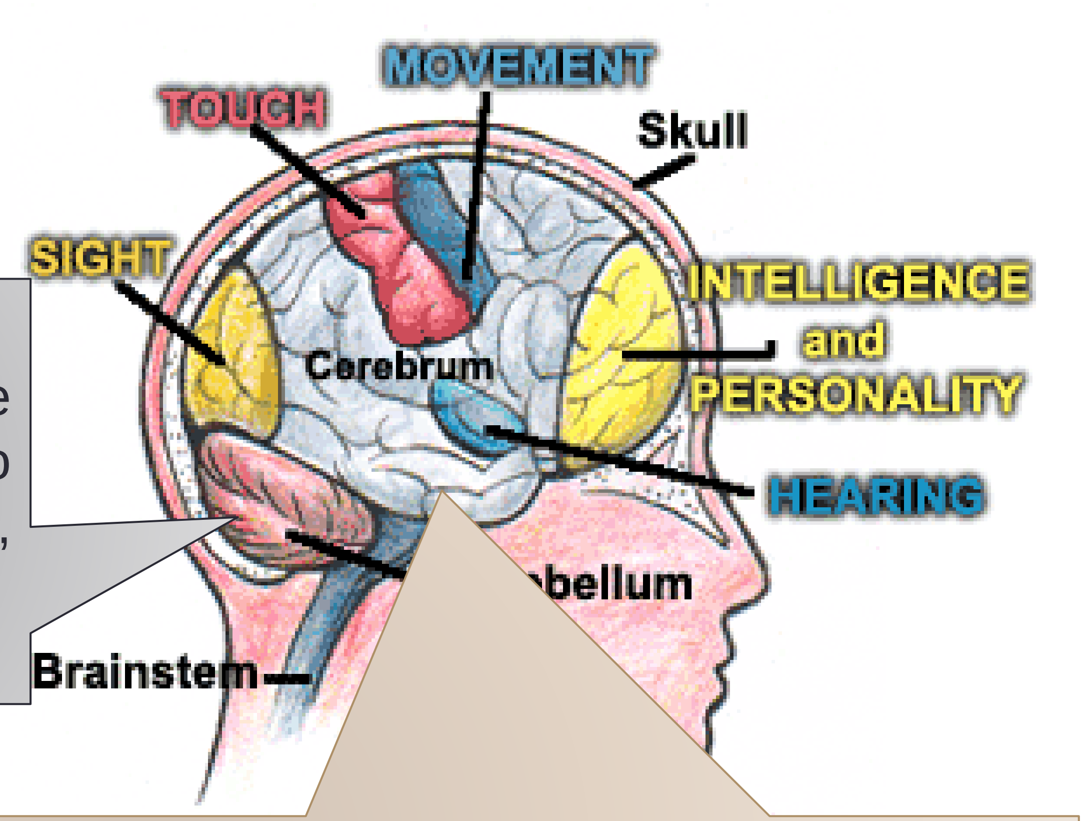

Weights
Constant
1
w_0
x_1
w_1
Weighted Sum
Σ
Out
inputs
x_{n-1}
w_{n-1}
x_n
w_n
Step Function

Learning Objective

- To understand mechanism calculation in perceptron.
- To know binary classification with perceptron.
- To implement a perceptron solving a small problem.

2.1 Brain

A I

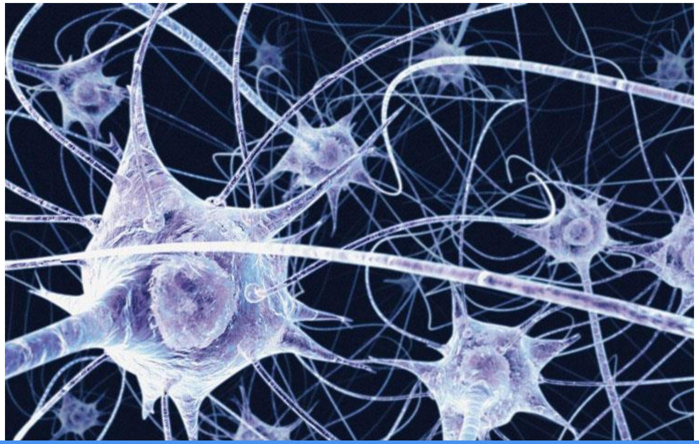


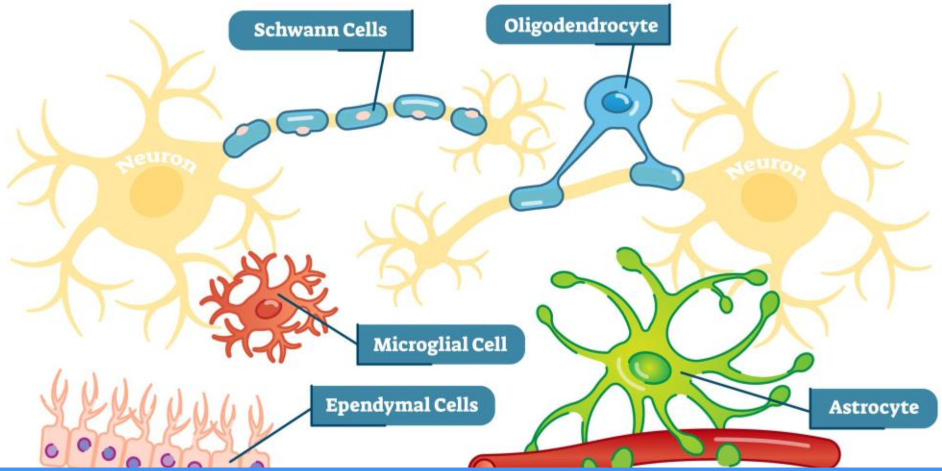
Cerebellum is located under the cerebrum. Its function is to coordinate muscle movements, maintain posture, and balance.

Cerebrum is the largest part of the brain and is composed of right and left hemispheres. It performs higher functions like interpreting touch, vision and hearing, as well as speech, reasoning, emotions, learning, and fine control of movement.

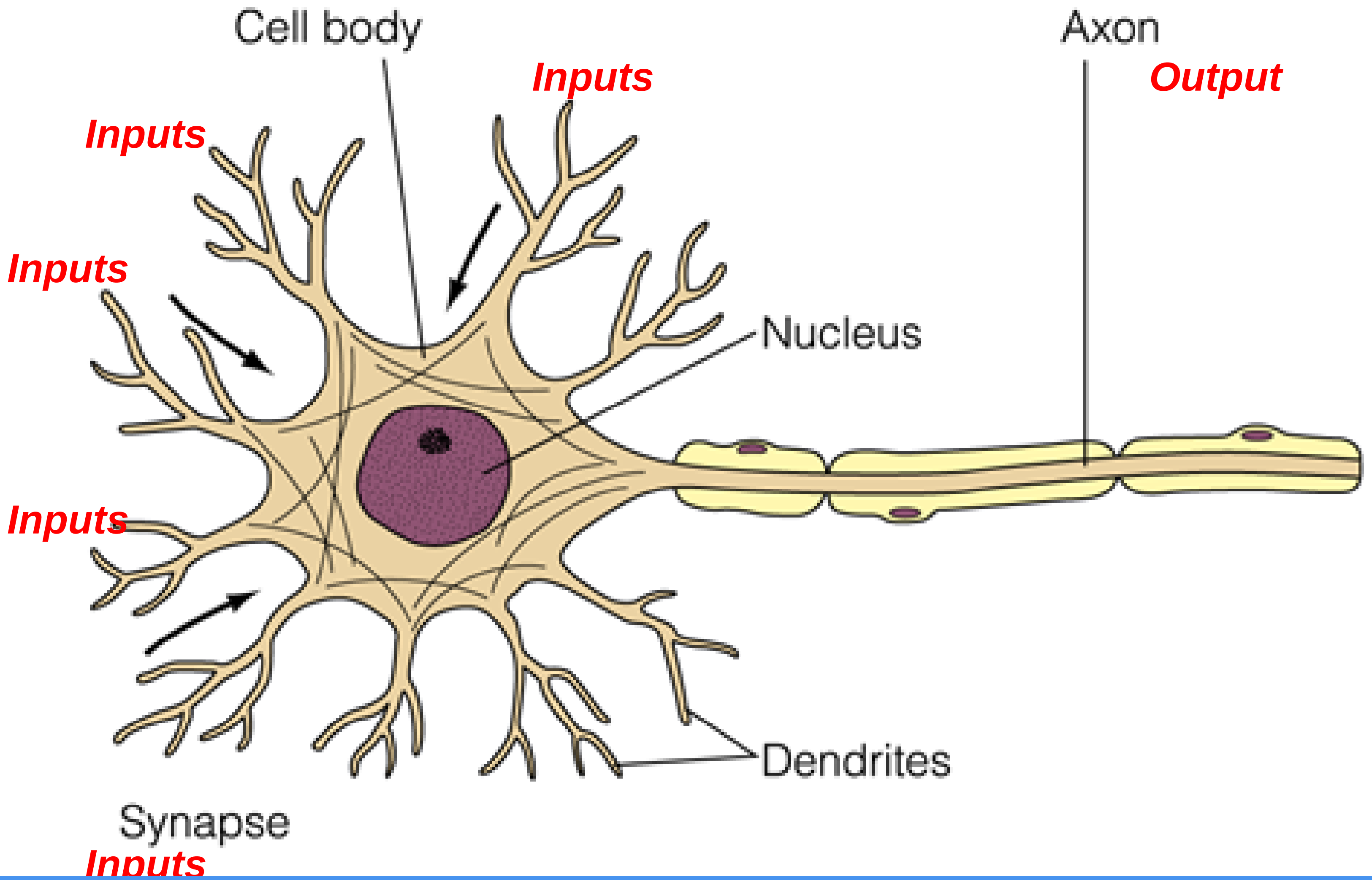
2.2 Cells of the brain

- The brain is made up of two types of cells: nerve cells and glia cells
  - Nerve cells There are many sizes and shapes of neurons, but all consist of a cell body, dendrites and an axon. The neuron conveys information through electrical and chemical signals.
  - Glial cells provide support for the neurons





2.3 Biological Neuron



2.4 Brain vs Computers

av

- 200 billion neurons, 32 trillion synapses
- Element size: 10^{-6} m
- Energy use: 25W
- Processing speed: 100 Hz
- Parallel, Distributed
- Fault Tolerant
- Learns: Yes
- Intelligent/Conscious: Usually

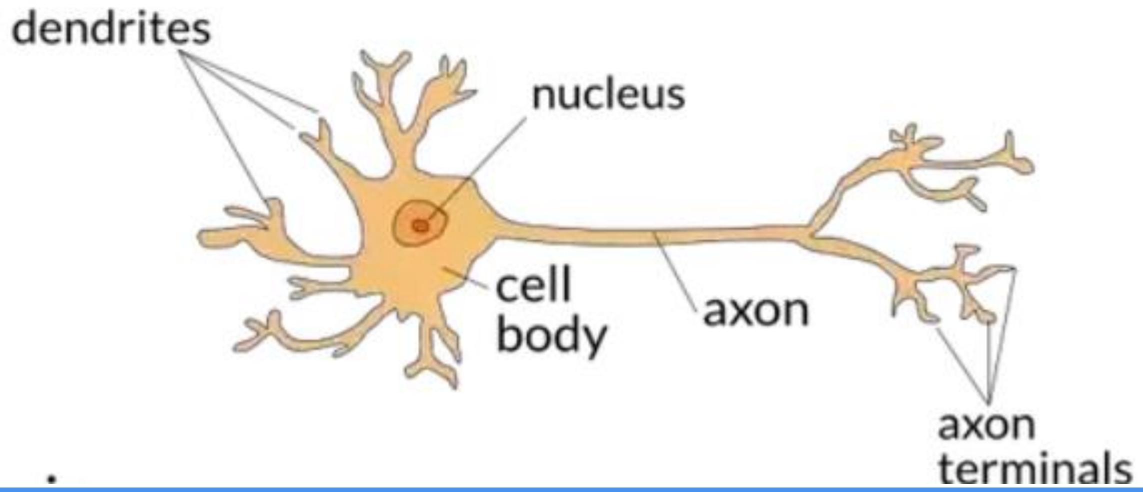
computer

- 1 billion bytes RAM but trillions of bytes on disk
- Element size: 10^{-9} m
- Energy watt: 30-90W (CPU)
- Processing speed: >1GHz
- Serial, Centralized
- Generally, not Fault Tolerant
- Learns: Some
- Intelligent/Conscious: Generally, No

2.5 Artificial Neural Networks

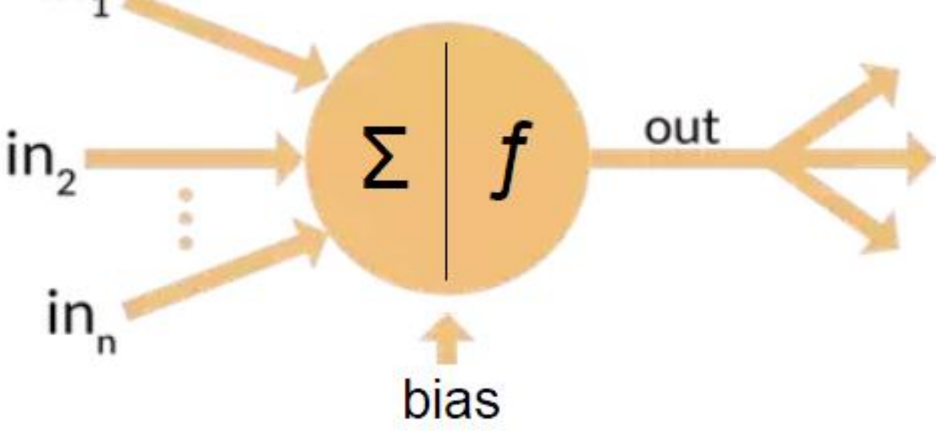
ANN or Neural Network (NN) is analogous to a biological neuron

- Input cells = "dendrites"
- a single output value = "axon" produced either 0 or 1.



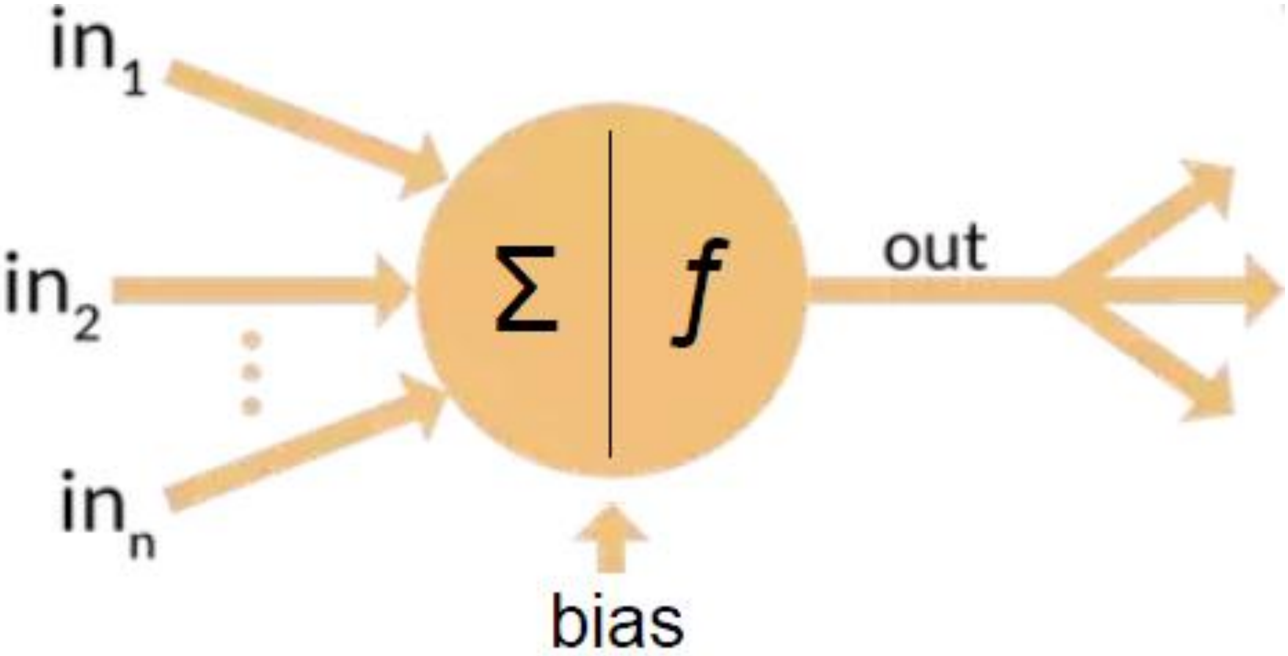
→

จดจำ



2.6 Perceptron

- Frank Rosenblatt conducted Perceptron in 1950.
- Perceptron is supervised learning of binary classifiers. 1/0 , T/F



2.6 Perceptron

- Frank Rosenblatt conducted Perceptron in 1950.
- Perceptron is supervised learning of binary classifiers.

Train, test

Supervised Learning
vs
Unsupervised Learning

Data → จัดกลุ่ม

```
if (a>10)
        output = 1
else
        output = 0
```

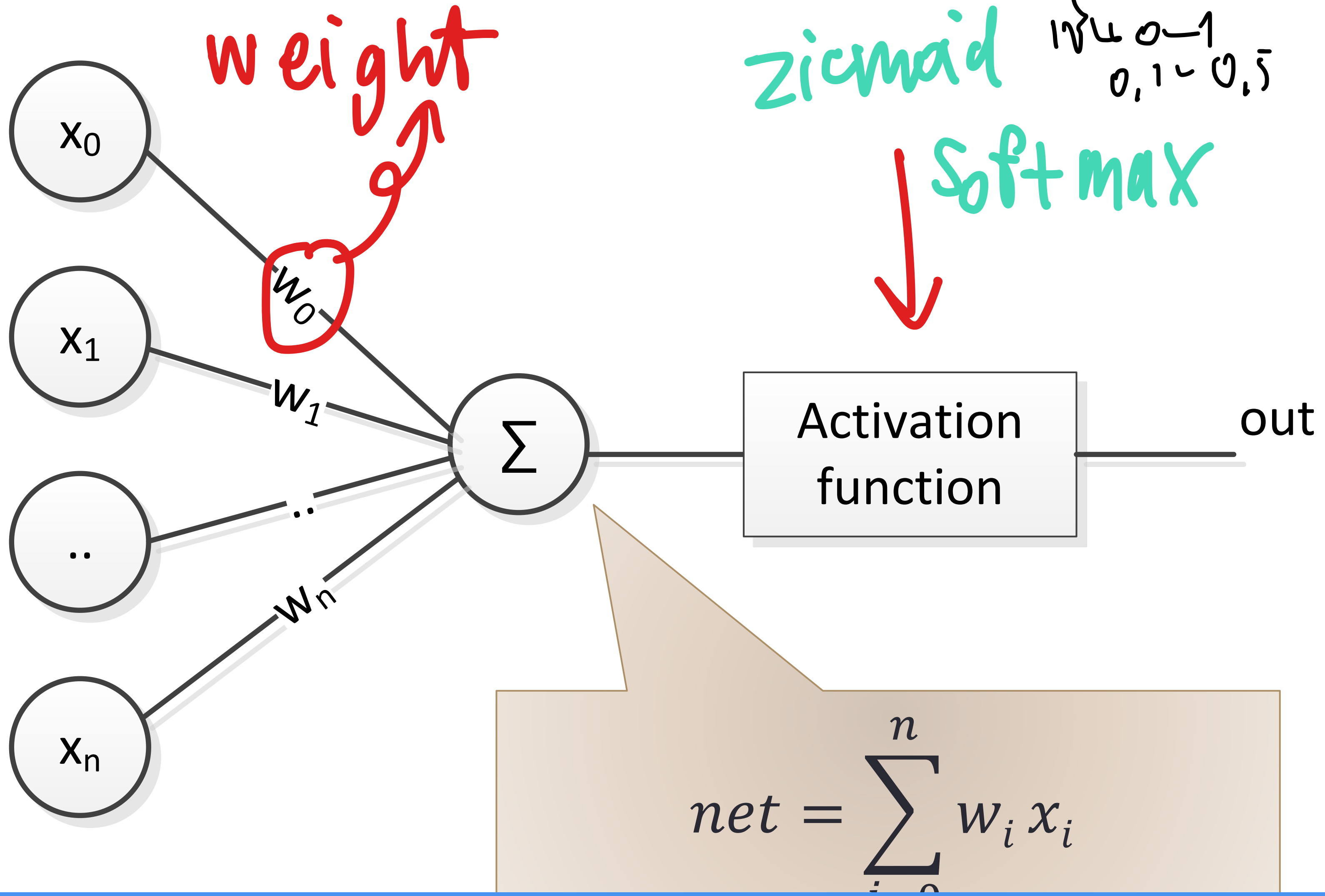
2.7 Create Perceptron

มักเหมาะกับงาน
ของจำแนก

zicmaid

เช่น 0-1
0,1 - 0,5

Soft max



$$net = \sum_{i=0}^{n} w_i x_i$$

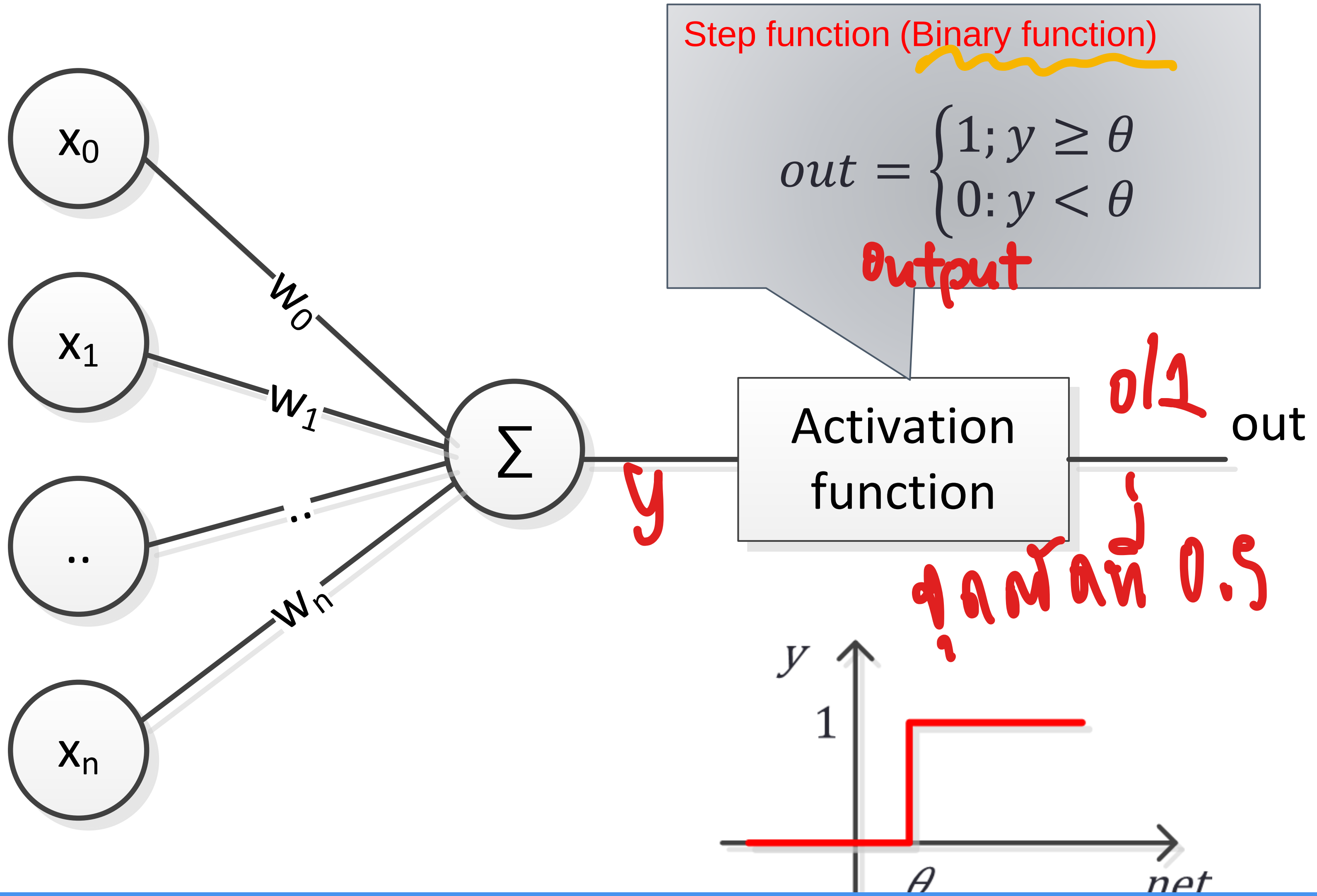
#R

```
x = c(1,2,3,4)
w = c(0.1,0.2,0.3,0.4)
print(w*x)
net = sum(w*x)
print(net)
```

```
> x = c(1,2,3,4)
> w = c(0.1,0.2,0.3,0.4)
> print(w*x)
[1] 0.1 0.4 0.9 1.6
> net = sum(w*x)
> print(net)
[1] 3
```

$$\begin{cases} 1 ; x \geq 0 \\ 0 ; x > 0 \\ 1 ; x \leq 0 \end{cases}$$

2.7 Create Perceptron



Step function (Binary function)

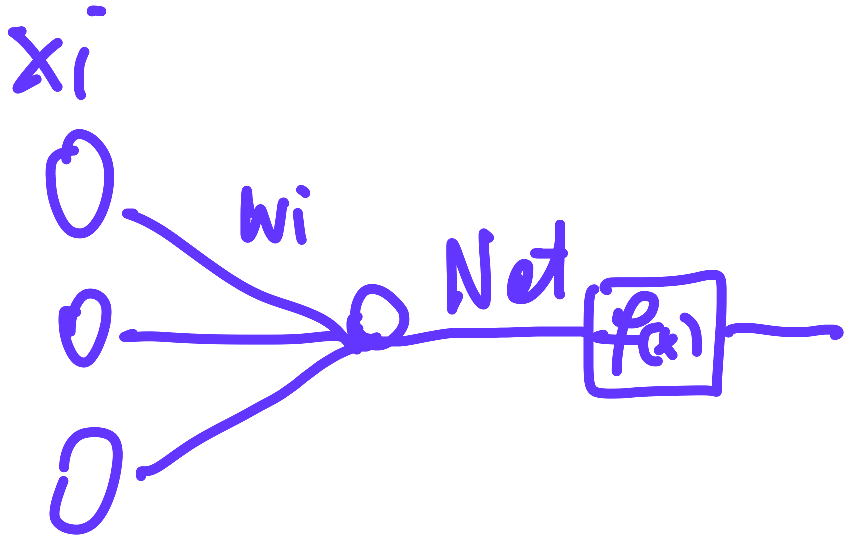
$$out = \begin{cases} 1; y \geq \theta \\ 0: y < \theta \end{cases}$$

Output

จุดตัดที่ 0.5

```
#R
x = c(1,2,3,4)
w = c(0.1,0.2,0.3,0.4)
print(w*x)
net = sum(w*x)
print(net)
threshold = 1
if(net > 1)
{
        out = 1
}else
{
        out = 0
}
print(out)
```

2.8 Perceptron training



This is sequence of training perceptron

1) $\boxed{net = \sum_{i=0}^{n} w_i x_i}$

2) $out = \begin{cases} 1; y \geq \theta \\ 0: y < \theta \end{cases}$ or $out = \frac{1}{1+e^{-net}}$

3) Set the desired output that is *out'* at 0 and applies an equation below for updating w_i

4) $w'_i = w_i r(out' - out)x_i$ Slope

5) Uses eq.(1) and (2) to calculate a new *out*, when the new *out* is “0”, stop the process and keeps the new weight. In the case of a new out does not be “0”, the process moves to do step 3, until getting “0” at *out*.

out = out ปกติ $\overline{out}$ = target ที่ต้องการ

test
ไม่เปลี่ยน
เหมือน
train

Algorithm 5 PerceptronTrain(**D**, *MaxIter*)

```
$w_d \leftarrow 0$, for all $d = 1 \dots D$                    // initialize weights
$b \leftarrow 0$                                                // initialize bias
for $iter = 1 \dots MaxIter$ do
    for all $(x,y) \in \mathbf{D}$ do
        $a \leftarrow \sum_{d=1}^{D} w_d\, x_d + b$            // compute activation for this example
        if $ya \leq 0$ then
            $w_d \leftarrow w_d + y x_d$, for all $d = 1 \dots D$   // update weights
            $b \leftarrow b + y$                                 // update bias
        end if
    end for
end for
return $w_0, w_1, \dots, w_D, b$
```

bias อีกตัว

ปรับ weight

train

Output
↓
algorithm
↓
Cari fly

Algorithm 6 PerceptronTest($w_0, w_1, \dots, w_D, b, \hat{x}$)

```
$a \leftarrow \sum_{d=1}^{D} w_d\, \hat{x}_d + b$     // compute activation for the test example
return sign$(a)$
```

Activity 2.1 Checking data before classification

ง่าย

- Load 2D data group_a.txt and group_b.txt
- Plot the data in R
- Consider classification point

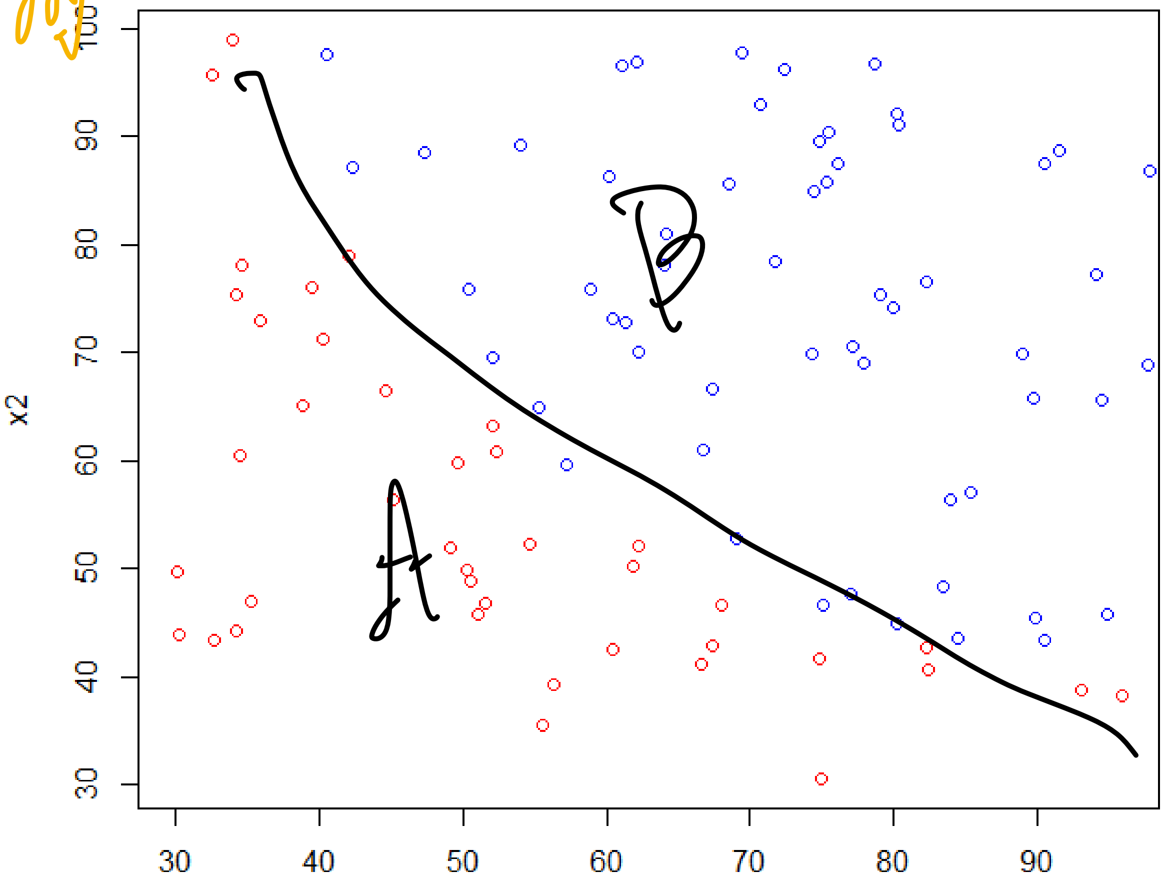
มีความเห็นข้อมูลก่อน

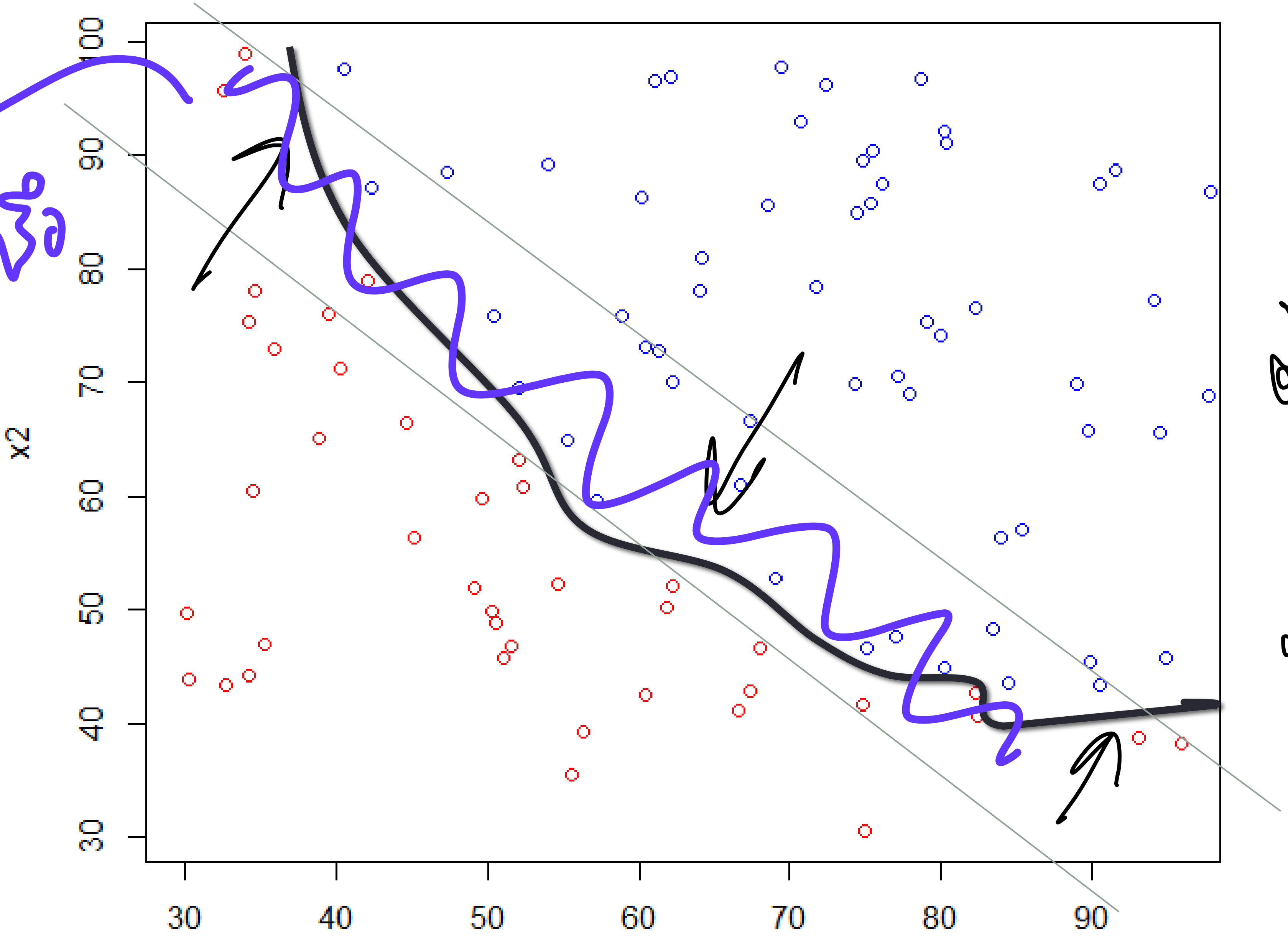
Supervised learning

```
#LOAD DATA
ga = read.csv("group_a.txt", header=T, sep="\t")
gb = read.csv("group_b.txt", header=T, sep="\t")

#PLOT DATA
plot.new()
plot(ga[,1:2],col="red")
lines(gb[,1:2],type="p",col="blue")

#CHECK RANGE OF DATA
summary(ga)
```





ค่าคลุมเครือ

ปนกันมาก

ต้อง add
ข้อมูลเพิ่ม
มาช่วยแบ่ง

Activity 2.2 Training perceptron

```
perceptron = function(x)
{
          err = 1
          iter = 0
          maxi = length(x$x1)
          while(err>0.01 && iter<5000)
          {
                    w1 = runif(1)      #RANDOM VALUE
                    w2 = runif(1)      #RANDOM VALUE

                    err = 0
                    for(i in 1:maxi)
                    {
                              #print(x[i,1])
                              x1 = x[i,1]
                              x2 = x[i,2]
                              sum = x1*w1 + x2*w2
                              #print(sum)
```

มากกว่า 5,000

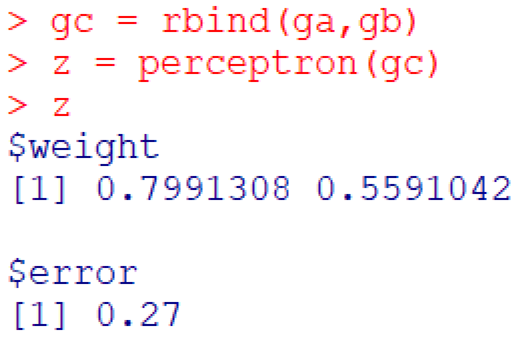
stop

สุ่มค่า weight

```
                              yh = step_func(sum,100)

                              yt = x[i,3]
                              if(yh==yt)
                              {
                                        #print("pass")
                              }else
                              {
                                        #print("not pass")
                                        err = err+1
                              }
                    }
                    err = err/maxi
                    #print(err)
                    iter = iter+1
                    #print(iter)
          }
          #print(iter)
          #print(c(w1,w2))
          listReturn = list("weight" = c(w1,w2), "error" = err)
          return(listReturn)
```

gc = rbind(ga,gb)
perceptron(gc)



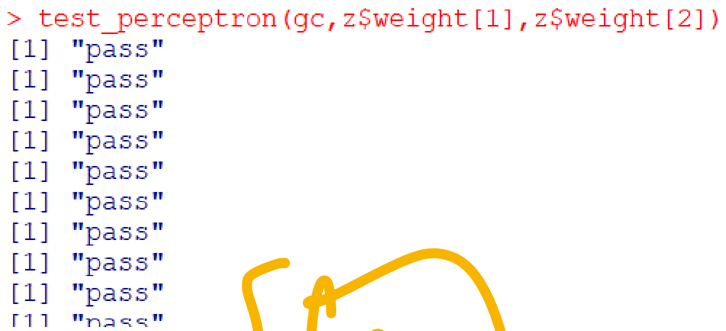
```
> gc = rbind(ga,gb)
> z = perceptron(gc)
> z
$weight
[1] 0.7991308 0.5591042

$error
[1] 0.27
```

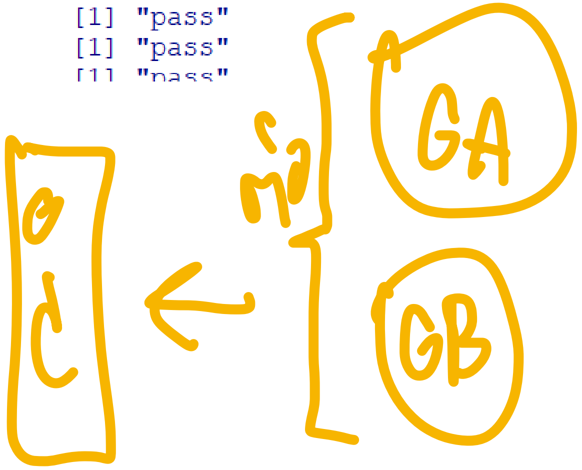
คือ codeเดียวกัน
กับก่อนหน้า

```
test_perceptron = function(x,w1,w2)
{
        err = 1
        iter = 0
        maxi = length(x$x1)
        err = 0
        for(i in 1:maxi)
        {
                x1 = x[i,1]
                x2 = x[i,2]
                sum = x1*w1 + x2*w2
                yh = step_func(sum,100)
                yt = x[i,3]
                if(yh==yt)
                {       print("pass")
                }else
                {       print("not pass")
                        err = err+1
                }
        }
        print(err/maxi)
```

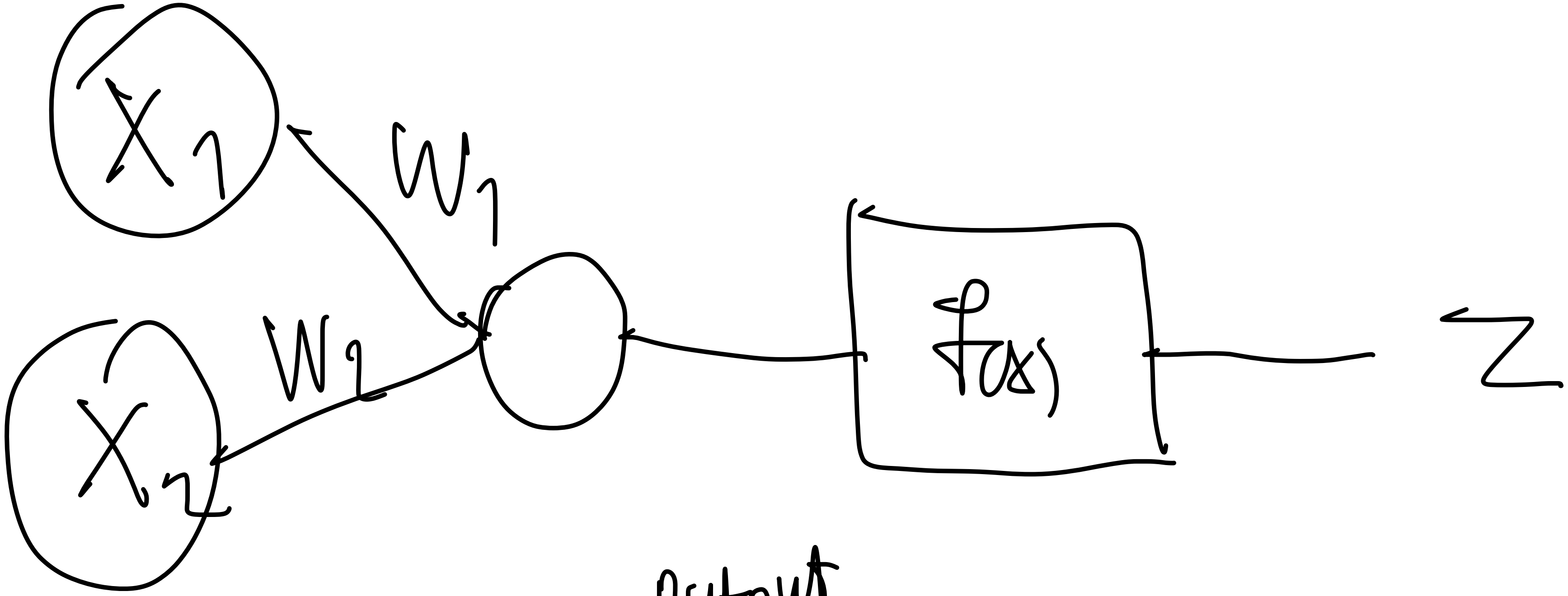
gc = rbind(ga,gb)
perceptron(gc)



```
> test_perceptron(gc,z$weight[1],z$weight[2])
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
[1] "pass"
```



Activity 2.2 all code



output

pass = 0   ข้อมูล 0 → 0

not pass   ไมตรง