

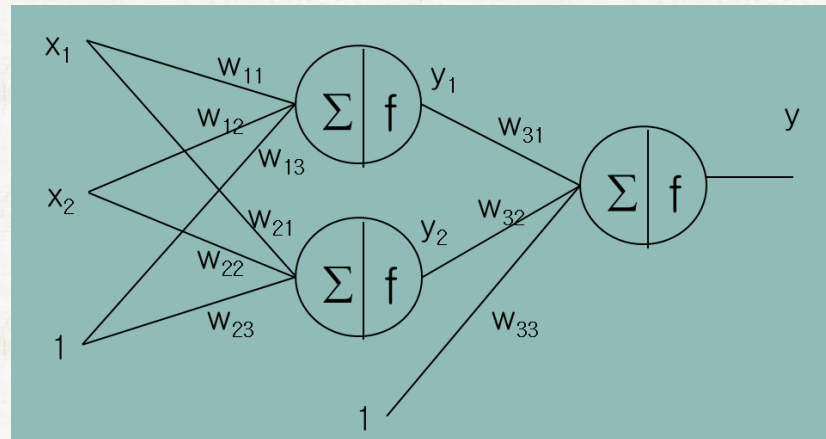
# Configuration of Neural Networks

# Contents

- Regression
- Binary-Class Classification
- Multi-Class Classification
- Nominal Input

# Regression

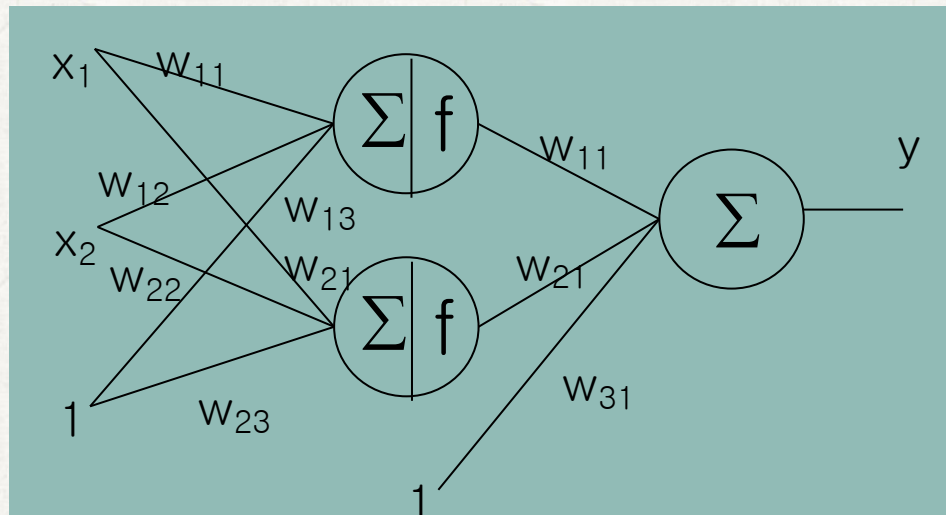
- Following Neural Network is OK for regression?



- Maybe NO!! Why?
- The activation functions produces a value between  $[0,1]$

# Regression

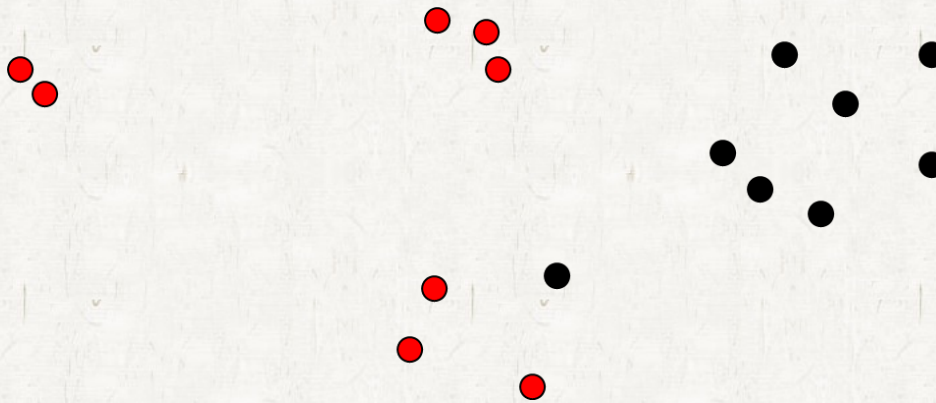
- Solution
  - Normalize the outputs into  $[0,1]$
  - Or, use a linear output node



# Binary-Class Classification

## • You Have Two Problems

$(x_{11}, x_{12}, Red), (x_{21}, x_{22}, Red), (x_{31}, x_{32}, Black), (x_{41}, x_{42}, Red), (x_{51}, x_{52}, Black), \dots$



- P1: NN cannot produce nominal values
- P2: Error Function for training



# Binary-Class Classification

## ● P1: Handling Nominal Values

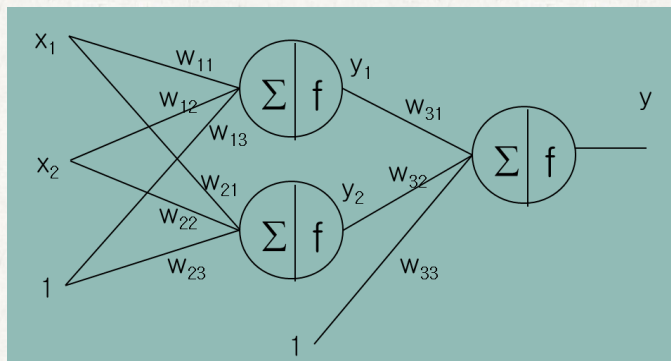
- Use 0 and 1 for class labels

$(x_{11}, x_{12}, Red), (x_{21}, x_{22}, Red), (x_{31}, x_{32}, Black), (x_{41}, x_{42}, Red), (x_{51}, x_{52}, Black), \dots$



$(x_{11}, x_{12}, 1), (x_{21}, x_{22}, 1), (x_{31}, x_{32}, 0), (x_{41}, x_{42}, 1), (x_{51}, x_{52}, 0), \dots$

- Use Sigmoid



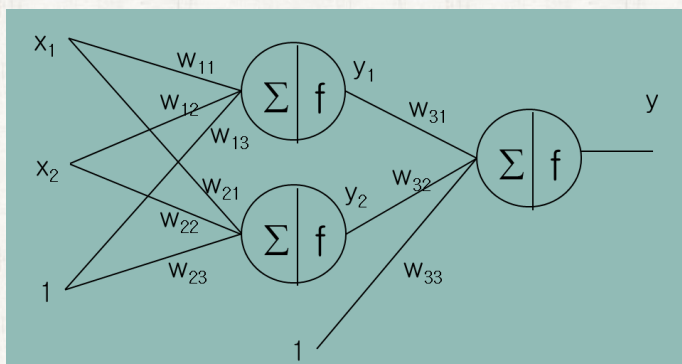
# Binary-Class Classification

## ● P2: Error Function for Training

Sigmoid at output node

+

MSE



~~$$E = \frac{1}{2} \sum_{n=1}^N (x_n - y_n)^2$$~~

NNs will not be trained in some special cases!!

# Binary-Class Classification

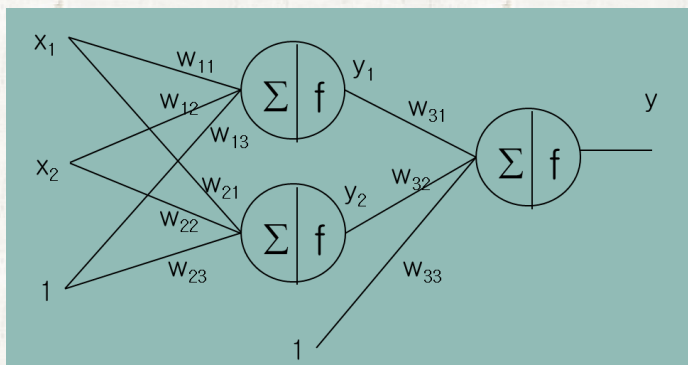
## ● P2: Error Function for Training

$(x_{11}, x_{12}, 1), (x_{21}, x_{22}, 1), (x_{31}, x_{32}, 0), (x_{41}, x_{42}, 1), (x_{51}, x_{52}, 0), \dots$

Sigmoid at output node

+

Cross Entropy



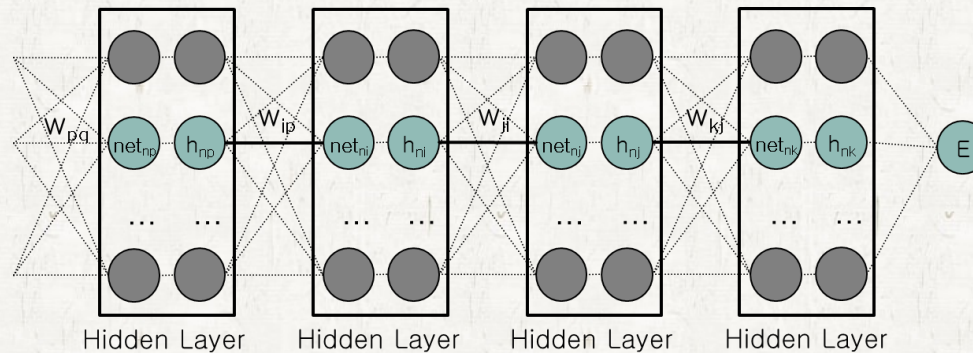
$$E = - \sum_{n=1}^N (t_n \log(y_n) + (1 - t_n) \log(1 - y_n))$$

where  $t_n \in \{0,1\}$  and  $y_n \in [0,1]$



# Binary-Class Classification

- Reminding: Weights between deep layers



$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial net_{nk}} \frac{\partial net_{nk}}{\partial w_{kj}} = -(t_n - h_{nk}) h_{nk} (1 - h_{nk}) h_{nj}$$

If  $h = \text{Sigmoid}(net)$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_{nj}} \frac{\partial net_{nj}}{\partial w_{ji}} = \left( \sum_{k=1}^K -(t_n - h_{nk}) h_{nk} (1 - h_{nk}) w_{kj} \right) \frac{\partial net_{nj}}{\partial w_{ji}}$$

$$\frac{\partial E}{\partial w_{ip}} = \frac{\partial E}{\partial net_{ni}} \frac{\partial net_{ni}}{\partial w_{ip}} = \left( \sum_{j=1}^J \left( \sum_{k=1}^K -(t_n - h_{nk}) h_{nk} (1 - h_{nk}) w_{kj} \right) \frac{\partial net_{nj}}{\partial w_{ji}} \right) \frac{\partial net_{ni}}{\partial w_{ip}}$$

# Binary-Class Classification

## Not Good...

- If  $h_{nk}$  is close to 1 or 0, all gradients for  $n$ -th training data is 0
- What if  $h_{nk}$  is close to 1 or 0 but it's wrong?

If most of data are mis-learned,  
NN cannot escape from the mis-learning

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial net_{nk}} \frac{\partial net_{nk}}{\partial w_{kj}} = -(t_n - h_{nk}) h_{nk}(1 - h_{nk}) h_{nj}$$

If  $h = \text{Sigmoid}(net)$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_{nj}} \frac{\partial net_{nj}}{\partial w_{ji}} = \left( \sum_{k=1}^K -(t_n - h_{nk}) h_{nk}(1 - h_{nk}) w_{kj} \right) \frac{\partial net_{nj}}{\partial w_{ji}}$$

$$\frac{\partial E}{\partial w_{ip}} = \frac{\partial E}{\partial net_{ni}} \frac{\partial net_{ni}}{\partial w_{ip}} = \left( \sum_{j=1}^J \left( \sum_{k=1}^K -(t_n - h_{nk}) h_{nk}(1 - h_{nk}) w_{kj} \right) \frac{\partial net_{nj}}{\partial w_{ji}} \right) \frac{\partial net_{ni}}{\partial w_{ip}}$$

# Binary-Class Classification

## ● Cross Entropy

$$E = - \sum_{n=1}^N (t_n \log(y_n) + (1 - t_n) \log(1 - y_n))$$

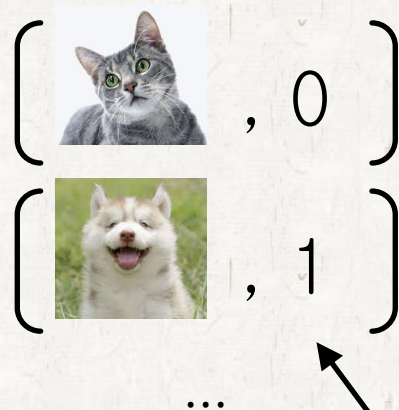
where  $t_n \in \{0,1\}$  and  $y_n \in [0,1]$   $y_n = \text{Sigmoid}(\text{net})$

$$\begin{aligned} \delta_{nk} &= \frac{\partial E}{\partial h_{nk}} \frac{\partial h_{nk}}{\partial \text{net}_{nk}} = - \left( \sum_{n=1}^N \frac{t_n}{y_n} + \frac{(1 - t_n)}{(1 - y_n)} \right) y_n (1 - y_n) \\ &= - \left( \sum_{n=1}^N t_n (1 - y_n) + (1 - t_n) y_n \right) \end{aligned}$$

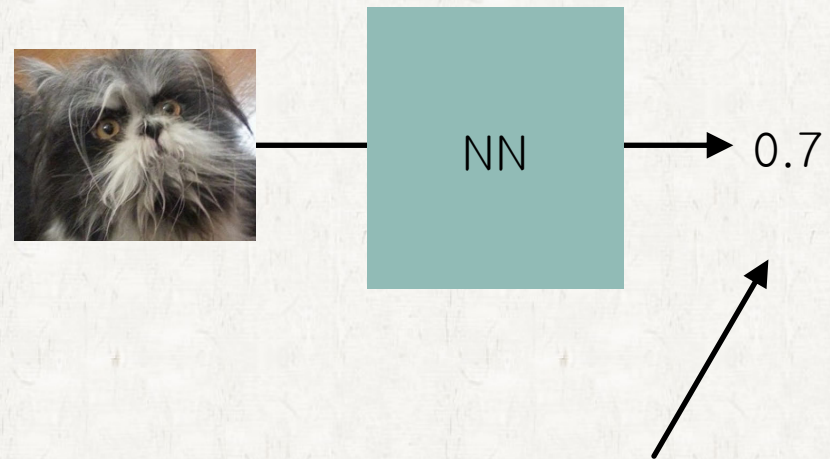
# Binary-Class Classification

- What Cross Entropy Is ?

Training Data



Test Data



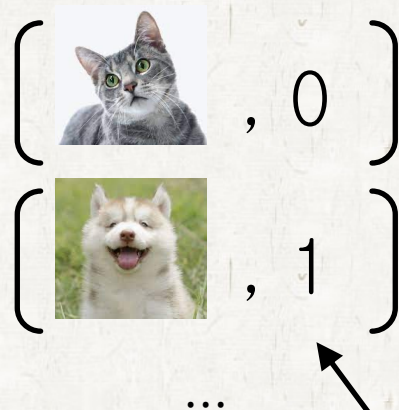
I call this “**Label**”, then How can we interpret this?



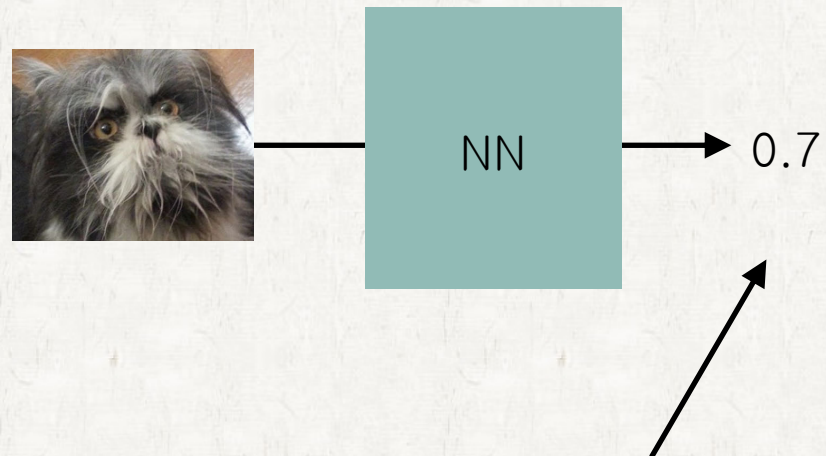
# Binary-Class Classification

- What Cross Entropy Is ?

Training Data



Test Data



Let's **regard** this as “**Probability** of Dog”, then it is easy to interpret



# Binary-Class Classification

- What Cross Entropy Is?
  - Output of Classification  $\rightarrow$  Probability

Find  $w$  so that NN correctly predicts all training data



Find  $w$  which maximizes the probability that NN correctly predicts all training data



Find  $w$  which maximizes the following:

$$\left( \prod_{(x,1) \in Data} NN(x; w) \right) \times \left( \prod_{(x,0) \in Data} (1 - NN(x; w)) \right)$$

Probability to be DOG

Probability to be CAT

# Binary-Class Classification

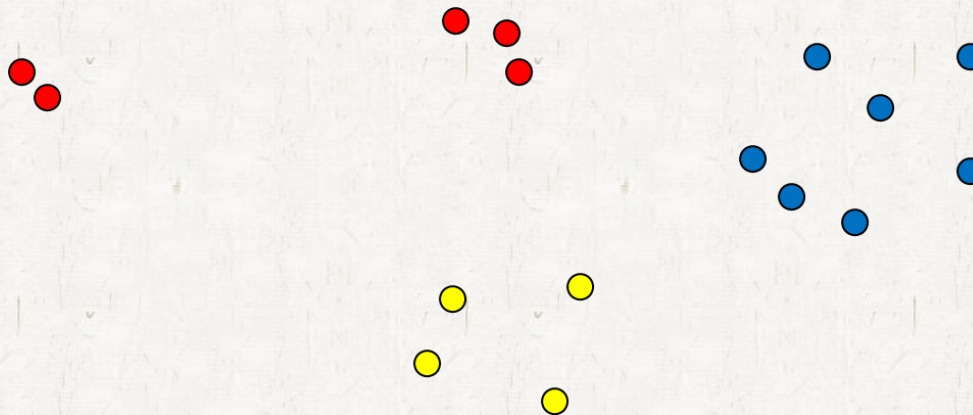
## ● Cross Entropy

$$\begin{aligned}
 & \operatorname{argmax}_w \left( \prod_{(x,1) \in \text{Data}} NN(x; w) \times \prod_{(x,0) \in \text{Data}} (1 - NN(x; w)) \right) \\
 &= \operatorname{argmax}_w \log \left( \prod_{(x,1) \in \text{Data}} NN(x; w) \times \prod_{(x,0) \in \text{Data}} (1 - NN(x; w)) \right) \\
 &= \operatorname{argmax}_w \left( \sum_{(x,1) \in \text{Data}} \log NN(x; w) + \sum_{(x,0) \in \text{Data}} \log(1 - NN(x; w)) \right) \\
 &= \operatorname{argmax}_w \left( \sum_{(x,1) \in \text{Data}} \log NN(x; w) + \sum_{(x,0) \in \text{Data}} \log(1 - NN(x; w)) \right) \\
 &= \operatorname{argmax}_w \left( \sum_{(x,1) \in \text{Data}} y \log NN(x; w) + (1 - y) \log(1 - NN(x; w)) + \sum_{(x,0) \in \text{Data}} y \log NN(x; w) + (1 - y) \log(1 - NN(x; w)) \right) \\
 &= \operatorname{argmax}_w \left( \sum_{(x,1) \in \text{Data}} y \log NN(x; w) + (1 - y) \log(1 - NN(x; w)) \right)
 \end{aligned}$$

# Multi-Class Classification

## ● Problem

$(\mathbf{x}_1, \text{Red}), (\mathbf{x}_2, \text{Yellow}), (\mathbf{x}_3, \text{Blue}), (\mathbf{x}_4, \text{Red}), (\mathbf{x}_5, \text{Blue}), \dots$



## ● Easy...

# Multi-Class Classification

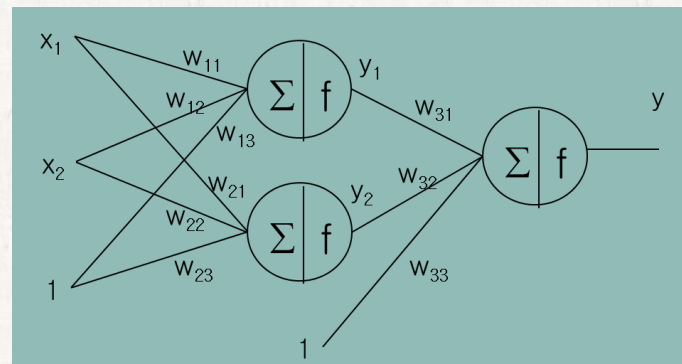
- Nominal Value Handling: Linear conversion of class labels

$(x_{11}, x_{12}, Red), (x_{21}, x_{22}, Yellow), (x_{31}, x_{32}, Blue), (x_{41}, x_{42}, Red), (x_{51}, x_{52}, Blue), \dots$



$(x_{11}, x_{12}, 1), (x_{21}, x_{22}, 0.5), (x_{31}, x_{32}, 0), (x_{41}, x_{42}, 1), (x_{51}, x_{52}, 0), \dots$

- Use Sigmoid at output node
- Prediction



$$class(\mathbf{x}) = \begin{cases} 1 & NN(\mathbf{x}) \geq 2/3 \\ 0.5 & 2/3 \geq NN(\mathbf{x}) \geq 1/3 \\ 0 & Otherwise \end{cases}$$

# Multi-Class Classification

## ● Not Good... why?

- There is no order between Red, Yellow, Blue
- They are just names. We cannot say that  $\text{Red} > \text{Yellow} > \text{Blue}$
- Linear conversion changes the original problem.

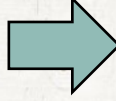
Red	→	1.0
Yellow	→	0.5
Blue	→	0.0



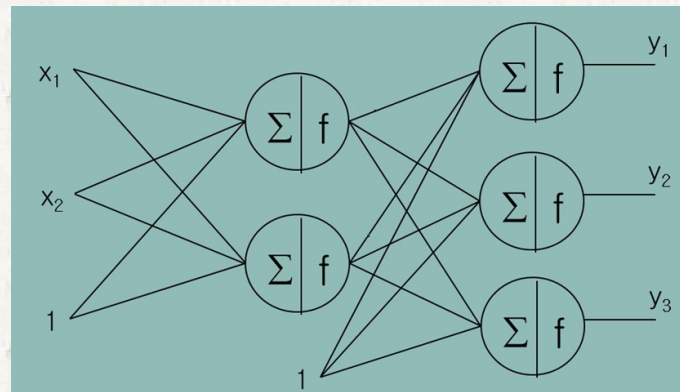
# Multi-Class Classification

Then?

- Create virtual outputs

$(x_{11}, x_{12}, Red),$		$(x_{11}, x_{12}, 1, 0, 0),$
$(x_{21}, x_{22}, Yellow),$		$(x_{21}, x_{22}, 0, 1, 0),$
$(x_{31}, x_{32}, Blue),$		$(x_{31}, x_{32}, 0, 0, 1),$
$(x_{41}, x_{42}, Red),$		$(x_{41}, x_{42}, 1, 0, 0),$
$(x_{51}, x_{52}, Blue),$		$(x_{51}, x_{52}, 0, 0, 1),$
...		...

- Place nodes at the output layer as many as virtual outputs



# Multi-Class Classification

## • How about Activation Function?

- Outputs of Multi-Class Classification satisfies

$$1 = \sum_{k=1}^{Class} t_{nk} \quad \text{for } n\text{-th training data}$$

- But, Sigmoid does not satisfy this (Not a Probability)

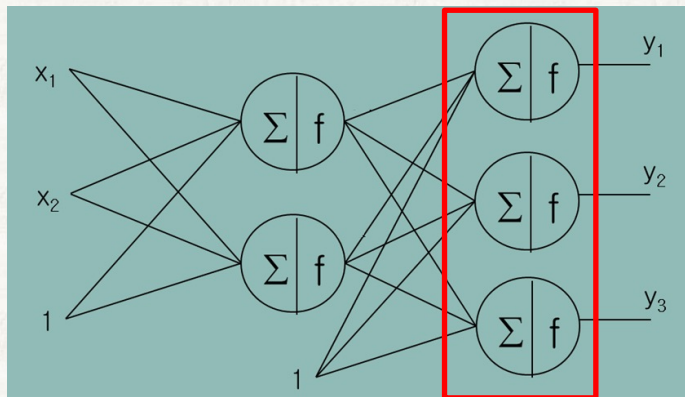
$$1 \neq \sum_{k=1}^{Class} \text{Sigmoid}(net_{nk}) \quad \text{for } n\text{-th training data}$$

- So, we use Softmax layer

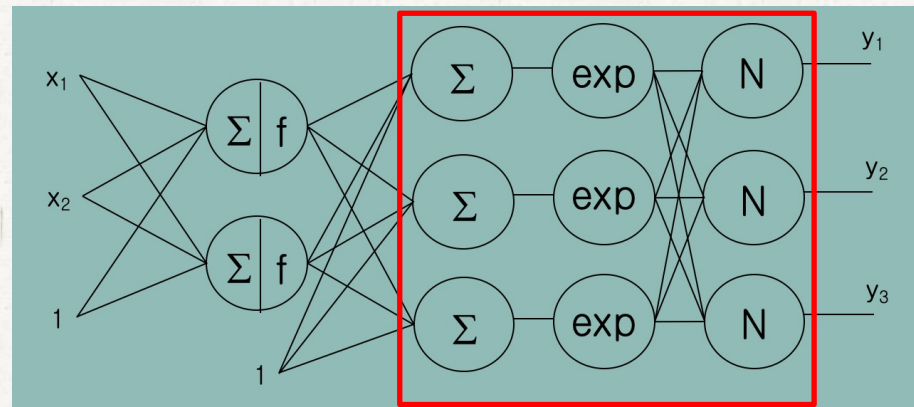
$$y_{nk} = \frac{\exp(net_{nk})}{\sum_{i=1}^{Class} \exp(net_{ni})}$$

# Multi-Class Classification

## ● Softmax layer



Regular layer



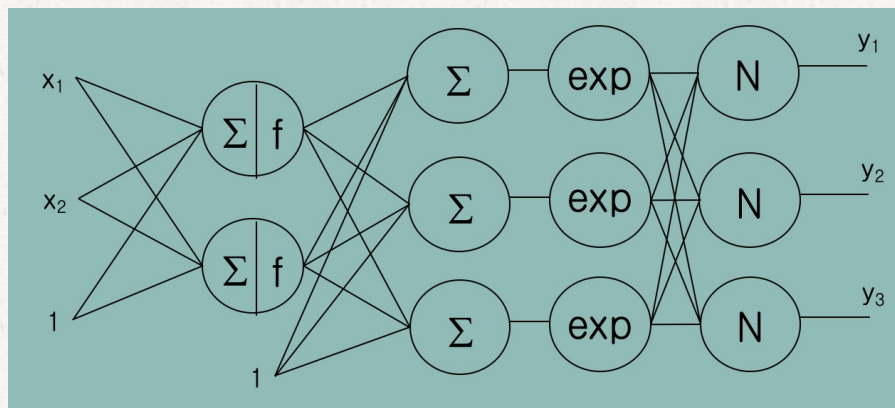
Softmax layer

$$y_{nk} = \frac{\exp(\text{net}_{nk})}{\sum_{i=1}^{\text{Class}} \exp(\text{net}_{nj})}$$

$$y_{n2} = \frac{\exp(\text{net}_{n2})}{\exp(\text{net}_{n1}) + \exp(\text{net}_{n2}) + \exp(\text{net}_{n3})}$$

# Multi-Class Classification

## Loss Function



$$E = \sum_{n=1}^{Data} \sum_{k=1}^{Class} -t_{nk} \log(y_{nk})$$

$$\text{Hmm?? } -(t_n \log(y_n) + (1 - t_n) \log(1 - y_n))$$



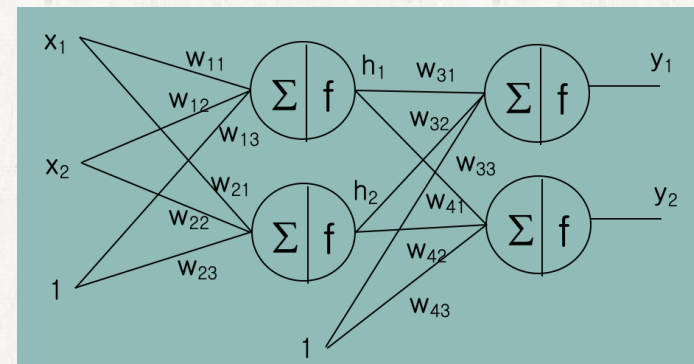
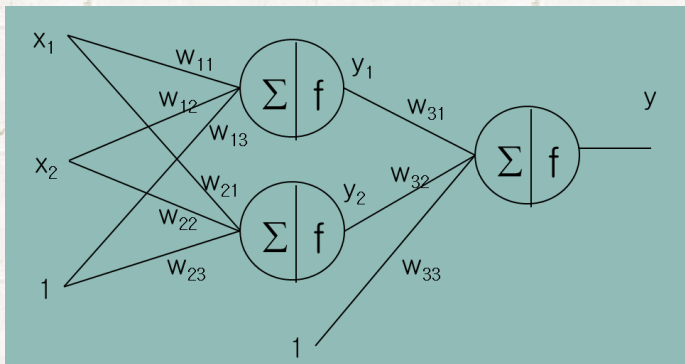
# Multi-Class Classification

## ● Cross Entropy for Multi-Class

$(x_{11}, x_{12}, Red)$   
 $(x_{21}, x_{22}, Red)$   
 $(x_{31}, x_{32}, Black)$   
 $(x_{41}, x_{42}, Red)$   
 $(x_{51}, x_{52}, Black)$

$(x_{11}, x_{12}, 1)$   
 $(x_{21}, x_{22}, 1)$   
 $(x_{31}, x_{32}, 0)$   
 $(x_{41}, x_{42}, 1)$   
 $(x_{51}, x_{52}, 0)$

$(x_{11}, x_{12}, 1, 0)$   
 $(x_{21}, x_{22}, 1, 0)$   
 $(x_{31}, x_{32}, 0, 1)$   
 $(x_{41}, x_{42}, 1, 0)$   
 $(x_{51}, x_{52}, 0, 1)$



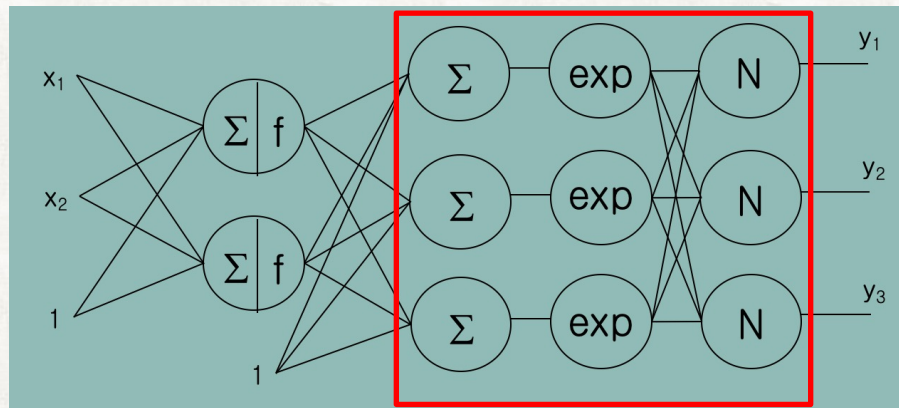
$$-(t_n \log(y_n) + (1 - t_n) \log(1 - y_n)) \quad -(t_{n1} \log(y_{n1}) + t_{n2} \log(y_{n2})) = - \sum_{k=1}^{Class} t_{nk} \log(y_{nk})$$



# Multi-Class Classification

- How about Error Function?
  - Softmax layer + Cross Entropy

$$E = \sum_{n=1}^{Data} \sum_{k=1}^{Class} CE(t_{nk}, y_{nk})$$



Softmax layer

$$y_{nk} = \frac{\exp(net_{nk})}{\sum_{i=1}^{Class} \exp(net_{nj})}$$

# Nominal Inputs

- What if you have categorical inputs

- Two inputs and one output

$$x_1 \in R$$

$$x_2 \in \{Red, Yellow, Blue\}$$

$$y \in \{0,1\}$$

- Create a new input variable for each categorical value

$$x_2 = \begin{cases} 1 & \text{if original } x_2 \text{ is Yellow} \\ 0 & \text{Otherwise} \end{cases}$$

$$x_3 = \begin{cases} 1 & \text{if original } x_2 \text{ is Red} \\ 0 & \text{Otherwise} \end{cases}$$

$$x_4 = \begin{cases} 1 & \text{if original } x_2 \text{ is Blue} \\ 0 & \text{Otherwise} \end{cases}$$

(0.1, Red, 0)

(0.2, Blue, 1)

(0.3, Yellow, 0)

(0.4, Red, 1)



(0.1, 1, 0, 0, 0)

(0.2, 0, 0, 1, 1)

(0.3, 0, 1, 0, 0)

(0.4, 1, 0, 0, 1)

# Summary

Problem	Activation Function		Loss function
	Hidden Layer	Output Layer	
Regression	ReLU	Linear	MSE
2-class Classification	ReLU	Sigmoid	CE
Multi-class Classification	ReLU	Softmax layer	CE