Configuration of Neural Networks

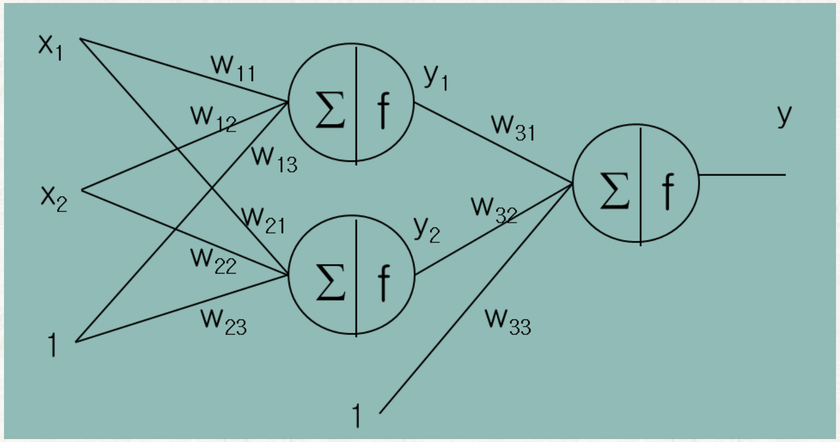
J.H. Lee, Dept. of Software, Sungkyunkwan Univ.

Contents

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

Regression

- Following Neural Network is OK for regression?

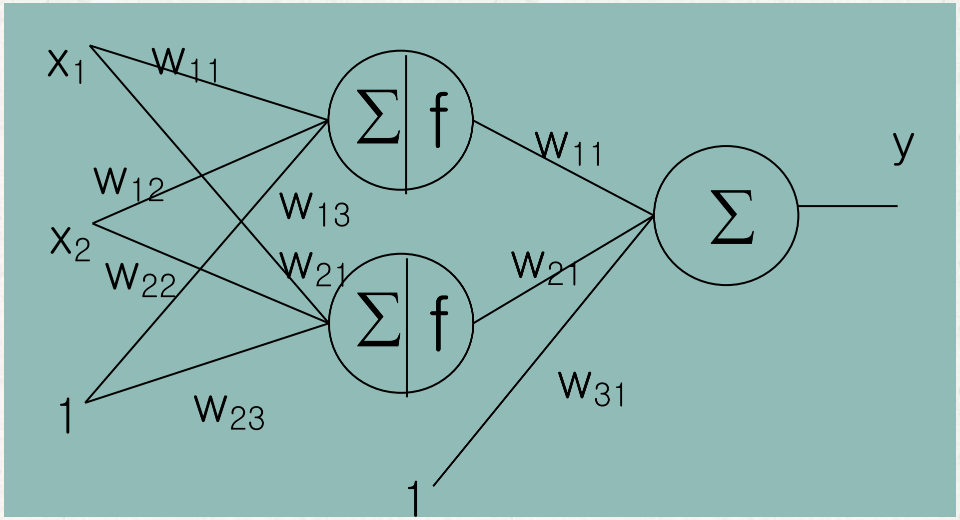


  - Maybe NO!! Why?
  - The activation functions produces a value between [0,1]

Regression

- Solution
  - Normalize the outputs into [0,1]

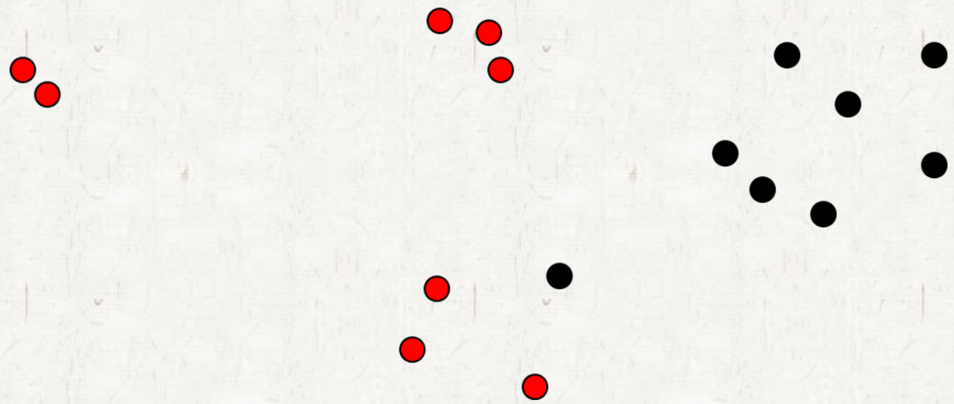
  - Or, use a linear output node



Binary-Class Classification

- You Have Two Problems

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



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

Binary-Class Classification

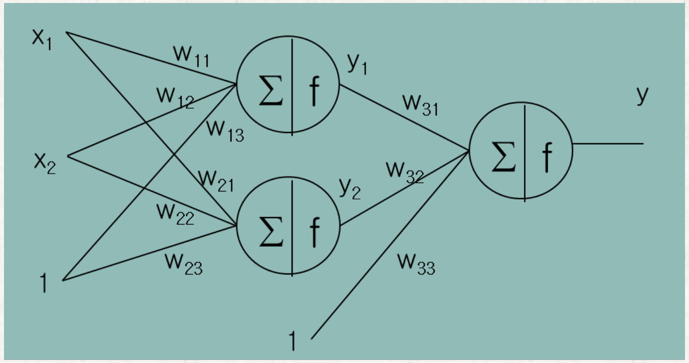
- P1: Handling Nominal Values
  - Use 0 and 1 for class labels

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

$$\Downarrow$$

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

  - Use Sigmoid

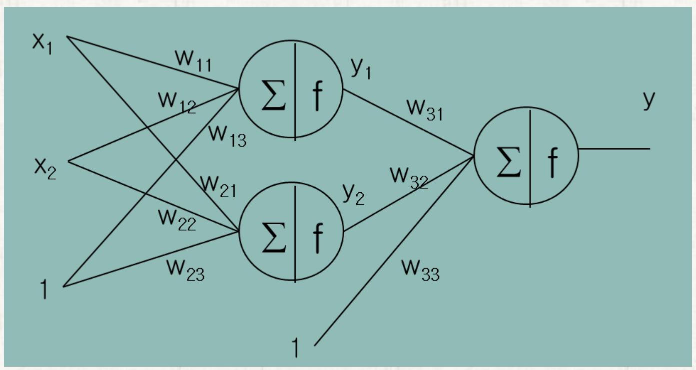


J.H. Lee, Dept. of Software, Sungkyunkwan Univ.

Binary-Class Classification

- P2: Error Function for Training

Sigmoid at output node + MSE



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

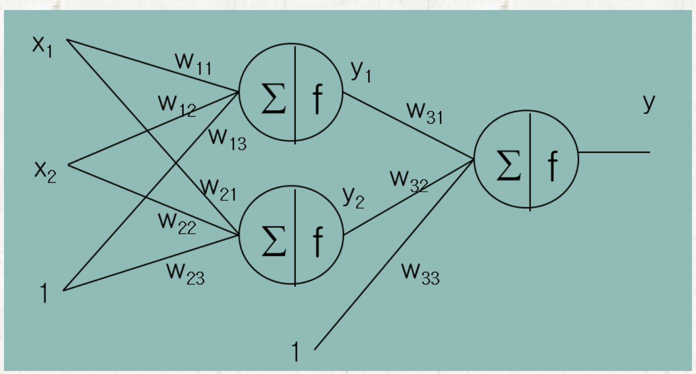
NNs will not be trained in some special cases!!

Binary-Class Classification

- P2: Error Function for Training

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

Sigmoid at output node + Cross Entropy

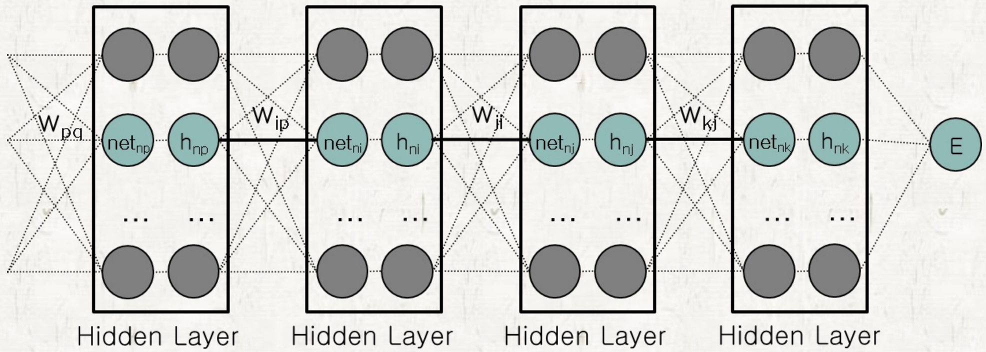


$$E = -\sum_{n=1}^{N} \left(t_n \log(y_n) + (1 - t_n)\log(1 - y_n)\right)$$

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 $h = Sigmoid(net)$

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

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

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 $h = Sigmoid(net)$

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

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

Binary-Class Classification

- Cross Entropy

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

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

$$\delta_{nk} = \frac{\partial E}{\partial h_{nk}}\frac{\partial h_{nk}}{\partial net_{nk}} = -\left(\sum_{n=1}^{N}\frac{t_n}{y_n} + \frac{(1 - t_n)}{(1 - y_n)}\right) y_n(1 - y_n)$$

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

J.H. Lee, Dept. of Software, Sungkyunkwan Univ.

Binary-Class Classification

- What Cross Entropy Is ?

Training Data

 , 0

 , 1

...

Test Data

 → NN → 0.7

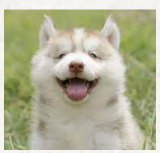
I call this "Label", then How can we interpret this?

Binary-Class Classification

- What Cross Entropy Is ?

Training Data

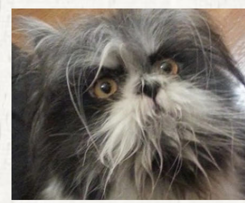
, 0



, 1

...

Test Data



NN

0.7

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

J.H. Lee, Dept. of Software, Sungkyunkwan Univ.

Binary-Class Classification

- What Cross Entropy Is?
  - Output of Classification -> Probability

Find w so that NN correctly predicts all training data

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

$\Leftrightarrow$ Find w which maximizes the following:

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

Probability to be DOG

Probability to be CAT

Binary-Class Classification

- Cross Entropy

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

$$= \underset{\mathrm{w}}{\operatorname{argmax}} \log\left(\prod_{(x,1)\in Data} NN(\mathrm{x};\mathrm{w}) \times \prod_{(x,0)\in Data} (1 - NN(\mathrm{x};\mathrm{w}))\right)$$

$$= \underset{\mathrm{w}}{\operatorname{argmax}}\left(\sum_{(x,1)\in Data} \log NN(\mathrm{x};\mathrm{w}) + \sum_{(x,0)\in Data} \log(1 - NN(\mathrm{x};\mathrm{w}))\right)$$

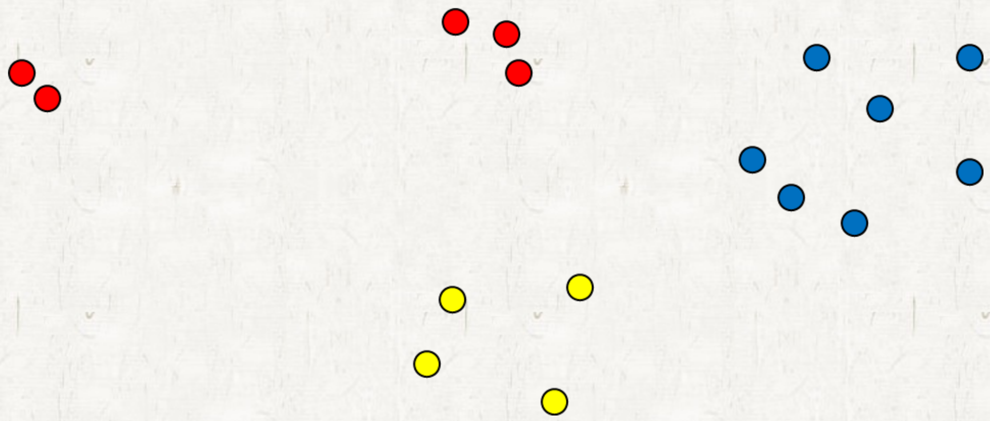
$$= \underset{\mathrm{w}}{\operatorname{argmax}}\left(\sum_{(x,1)\in Data} y \log NN(\mathrm{x};\mathrm{w}) + (1-y)\log(1 - NN(\mathrm{x};\mathrm{w})) + \sum_{(x,0)\in Data} y \log NN(\mathrm{x};\mathrm{w}) + (1-y)\log(1 - NN(\mathrm{x};\mathrm{w}))\right)$$

$$= \underset{\mathrm{w}}{\operatorname{argmax}}\left(\sum_{(x,1)\in Data} y \log NN(\mathrm{x};\mathrm{w}) + (1-y)\log(1 - NN(\mathrm{x};\mathrm{w}))\right)$$

Multi-Class Classification

- Problem

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



  - Easy…

Multi-Class Classification

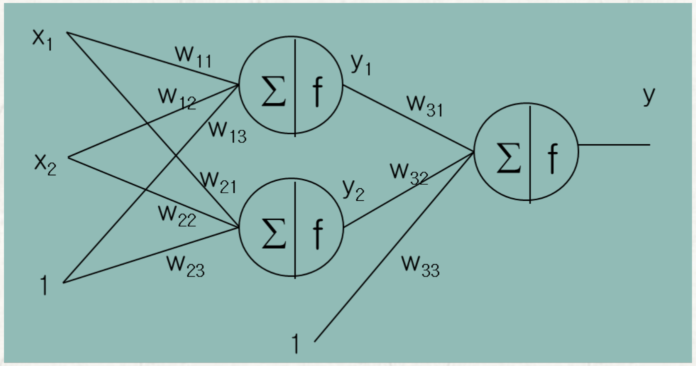
- Nominal Value Handling: Linear conversion of class labels

$$(\mathrm{x}_{11}, \mathrm{x}_{12}, Red),\ (\mathrm{x}_{21}, \mathrm{x}_{22}, Yellow),\ (\mathrm{x}_{31}, \mathrm{x}_{32}, Blue),\ (\mathrm{x}_{41}, \mathrm{x}_{42}, Red), (\mathrm{x}_{51}, \mathrm{x}_{52}, Blue), \cdots$$

$$\Downarrow$$

$$(\mathrm{x}_{11}, \mathrm{x}_{12}, 1),\ (\mathrm{x}_{21}, \mathrm{x}_{22}, 0.5),\ (\mathrm{x}_{31}, \mathrm{x}_{32}, 0),\ (\mathrm{x}_{41}, \mathrm{x}_{42}, 1), (\mathrm{x}_{51}, \mathrm{x}_{52}, 0), \cdots$$

- 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 Red > Yellow > Blue
  - Linear conversion changes the original problem.

Red → 1.0

Yellow → 0.5

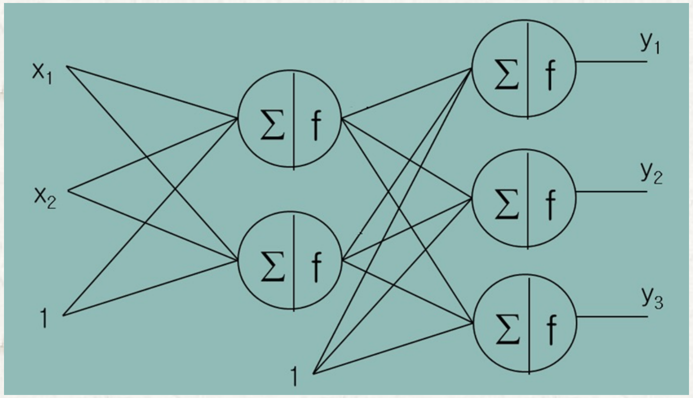
Blue → 0.0

Multi-Class Classification

- Then?
  - Create virtual outputs

$$
\begin{array}{lcl}
(\mathrm{x}_{11}, \mathrm{x}_{12}, Red), & & (\mathrm{x}_{11}, \mathrm{x}_{12}, 1, 0, 0), \\
(\mathrm{x}_{21}, \mathrm{x}_{22}, Yellow), & & (\mathrm{x}_{21}, \mathrm{x}_{22}, 0, 1, 0), \\
(\mathrm{x}_{31}, \mathrm{x}_{32}, Blue), & \Rightarrow & (\mathrm{x}_{31}, \mathrm{x}_{32}, 0, 0, 1), \\
(\mathrm{x}_{41}, \mathrm{x}_{42}, Red), & & (\mathrm{x}_{41}, \mathrm{x}_{42}, 1, 0, 0), \\
(\mathrm{x}_{51}, \mathrm{x}_{52}, Blue), & & (\mathrm{x}_{51}, \mathrm{x}_{52}, 0, 0, 1), \\
\ldots & & \ldots
\end{array}
$$

  - 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)

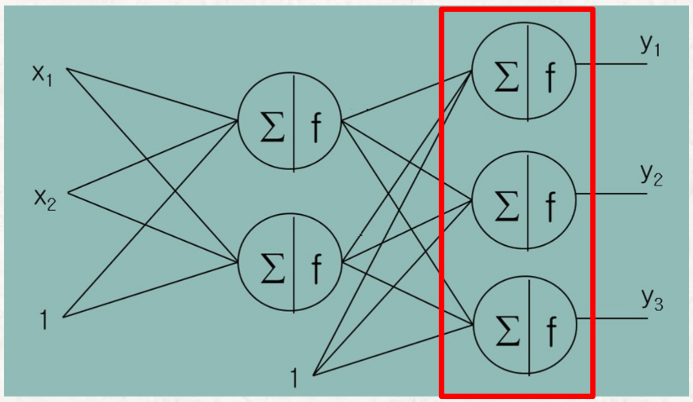
$$1 \neq \sum_{k=1}^{Class} Sigmoid(net_{nk}) \quad \text{for } n\text{−th training data}$$

  - So, we use Softmax layer

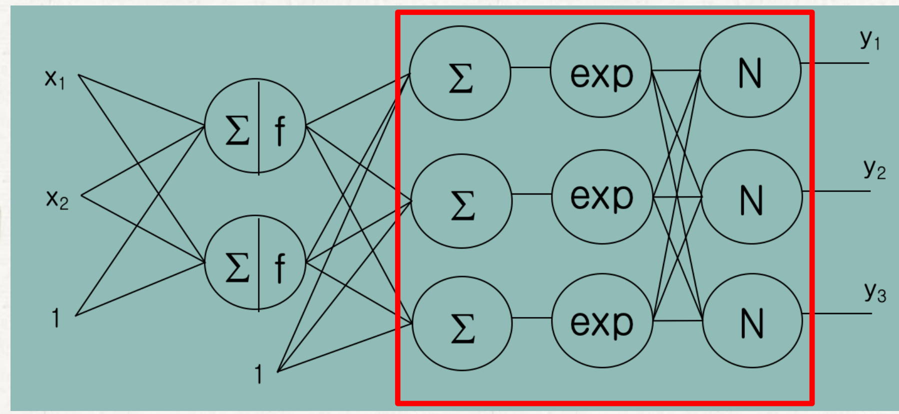
$$y_{nk} = \frac{exp(net_{nk})}{\sum_{i=1}^{Class} exp(net_{nj})}$$

Multi-Class Classification

- Softmax layer



Regular layer



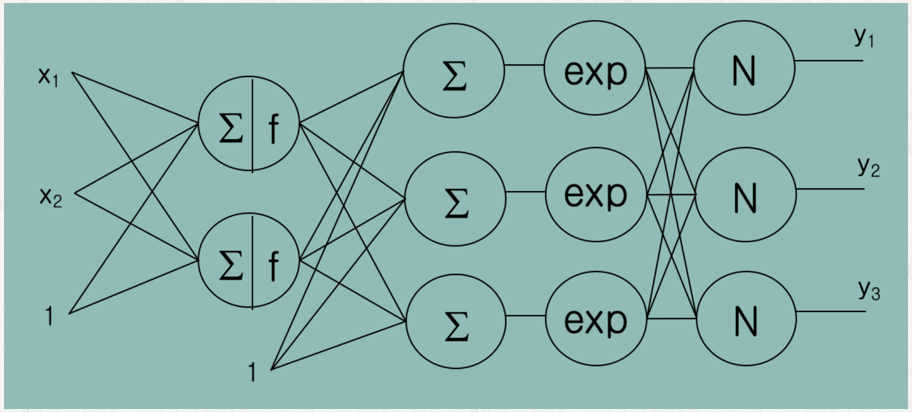
Softmax layer

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

$$y_{n2} = \frac{exp(net_{n2})}{exp(net_{n1}) + exp(net_{n2}) + exp(net_{n3})}$$

Multi-Class Classification

- Loss Function



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

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

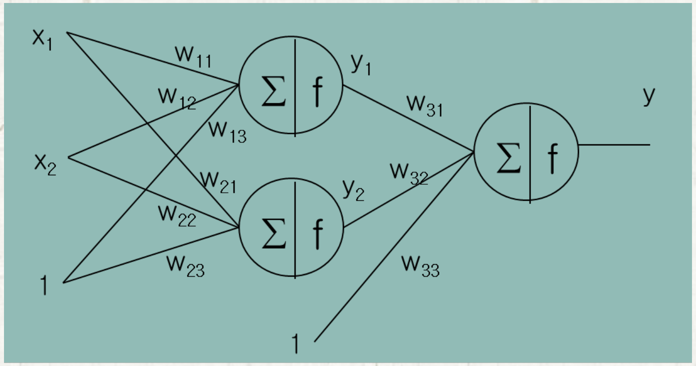
Multi-Class Classification

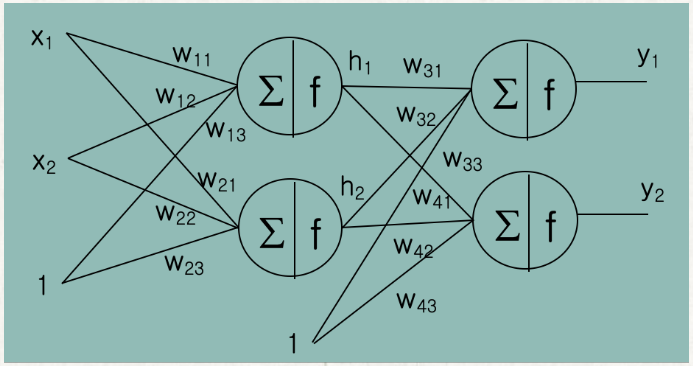
- Cross Entropy for Multi-Class

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

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

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





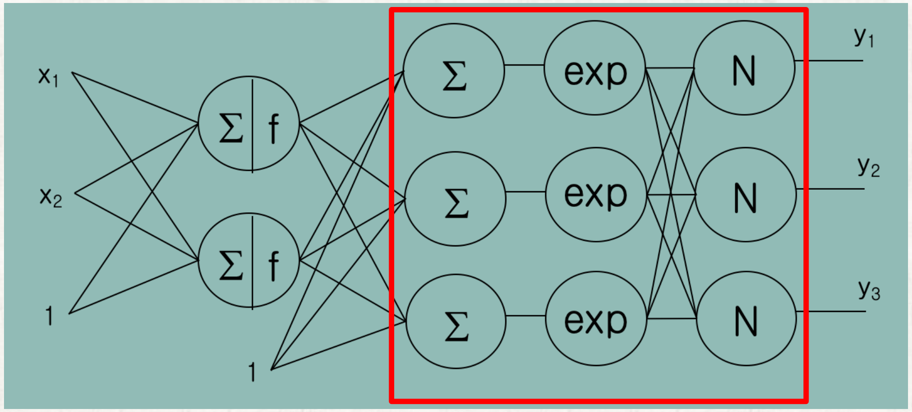
$$-(t_n \log(y_n) + (1 - t_n)\log(1 - y_n))$$

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

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

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

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

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

$$\begin{array}{l} (0.1, \text{Red}, 0) \\ (0.2, Blue, 1) \\ (0.3, Yellow, 0) \\ (0.4, Red, 1) \end{array} \Rightarrow \begin{array}{l} (0.1, 1,0,0, 0) \\ (0.2, 0,0,1, 1) \\ (0.3, 0,1,0, 0) \\ (0.4, 1,0,0, 1) \end{array}$$

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

J.H. Lee, Dept. of Software, Sungkyunkwan Univ.