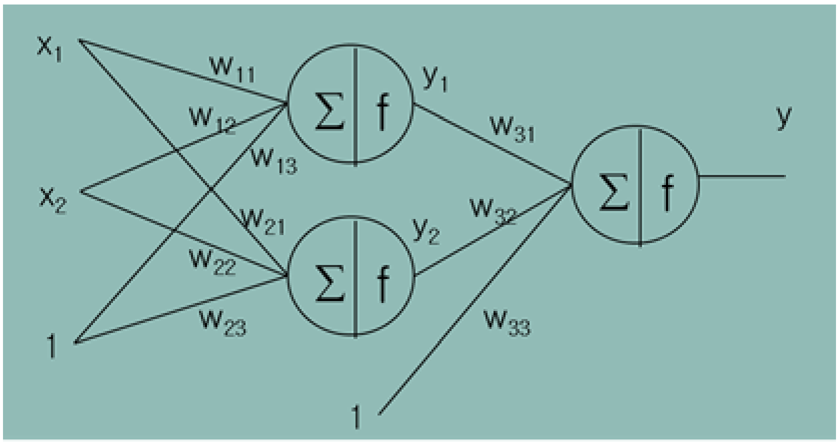
Neural Networks-Configuration

Regression

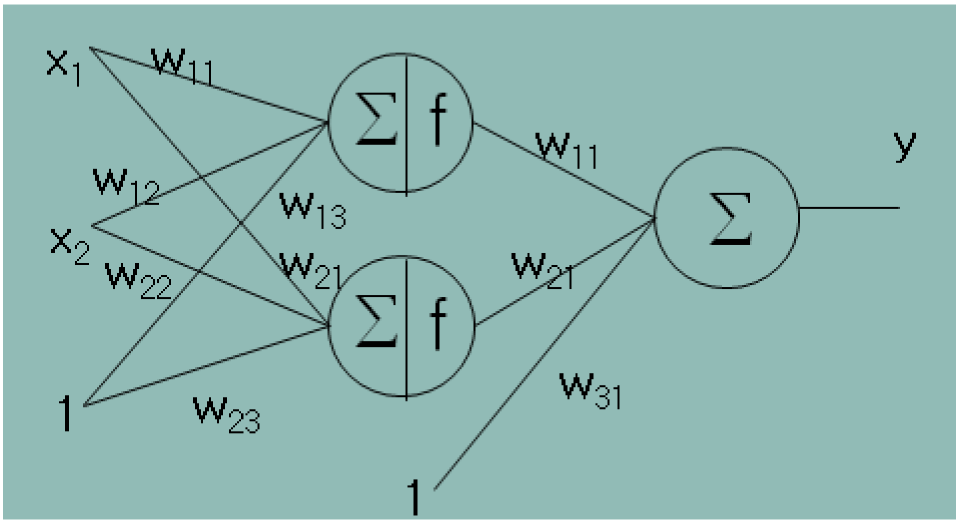


The activation functions produce a value between [0,1]

regression문제는 임의의 real value를 출력해야한다

Solution

- Normalize the outputs into [0,1]
- Or, Use a linear output node(=보통 항등함수)

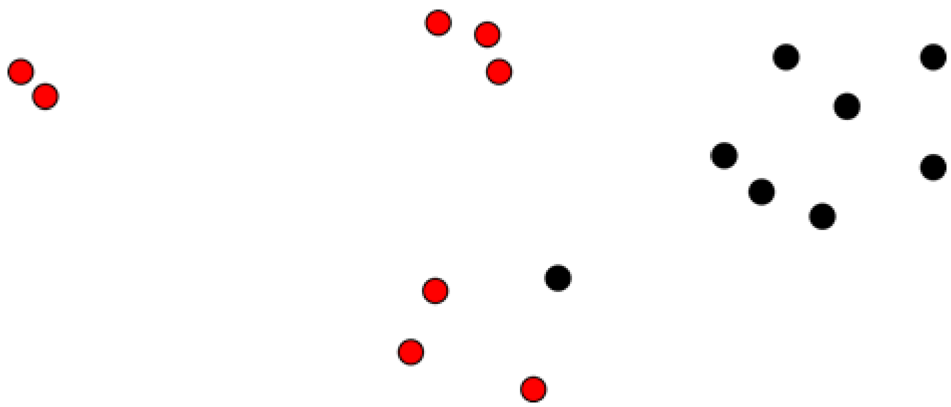


activation func : Non-linear transformation으로 해야 deep 하게 쌓는 의미가 있는것이다

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), $(x_{51}, x_{52}, Black)$, $\cdots$



output이 Two class

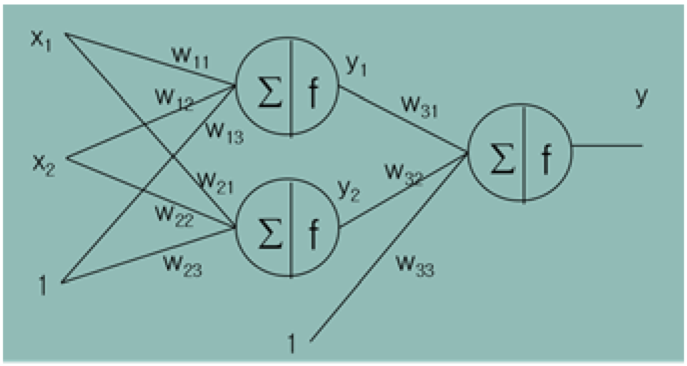
- P1 : NN cannot produces nominal values
- P2 : Error Function for training—>XMSE

P1 : Handling Nominal Values

- Use 0 and 1 for class labels

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

- Use sigmoid



P2 : Error Function for Training

NNs will not be trained in some special cases!!>>MSE는 잘안탬 ,그래서 cross entropy사용

Cross Entropy

$$H_p(q) = -\sum_{i=1}^{n} q(x_i) log p(x_i)$$

크로스 엔트로피는 실제 분포q에 대하여 알지 못하는 상태에서, 모델링을 통하여 구한 분포인p를 통하여 q를 예측하는 것이다.

q와 p가 모두 들어가서 크로스 엔트로피라고 한다.

머신러닝을 하는 경우에 실제 환경의 값과 q를, 예측값(관찰값) p를 모두 알고 있는 경우가 있다. **머신러닝의 모델은 몇%의 확률로 예측했는데, 실제 확률은 몇%야!**라는 사실을 알고 있을 때 사용한다.

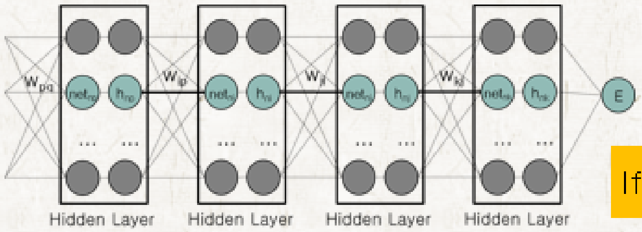
—>크로스 엔트로피는 출력을 확률로 가정하고 만들어진 error func

크로스 엔트로피에서는 실제값과 예측값이 맞는 경우에는 0으로 수렴하고, 값이 틀릴경우에는 값이 커지기 때문에, `실제 값과 예측 값의 차이를 줄이기 위한 엔트로피`라고 보시면 될 것 같습니다.

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

그럼 왜 Error func로 MSE를 안쓰고 cross entropy로 쓸까??

예를 들어, target값=1이고, 출력값(y)=0일경우

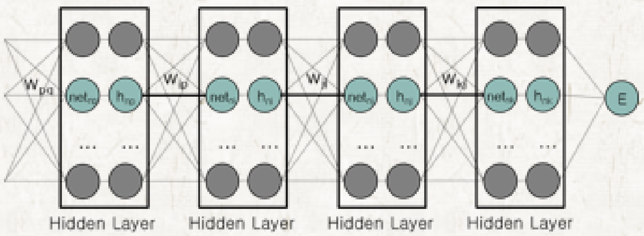


If $h = Sigmoid(net)$

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

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

$$\frac{\partial E}{\partial w_{ip}} = \frac{\partial E}{\partial net_{ni}} \frac{\partial net_{ni}}{\partial w_{ip}} = \delta_{ni} h_{np} \qquad \delta_{ni} = \left(\sum_{j=1}^{J} \delta_{nj} w_{ji}\right) \frac{\partial h_{ni}}{\partial net_{ni}}$$



If $h = Sigmoid(net)$

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

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

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

If hnk is close to 1 or 0, all gradients for n-th training data is 0

what if hnk is close to 1 or 0 but it's wrong?

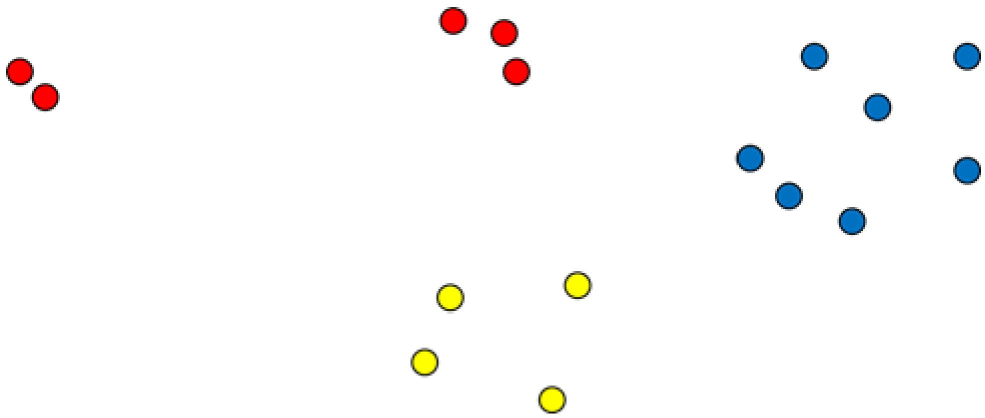
요약: Binary -class classification은 class output가 2개로 주어지며, output activation func는 sigmoid

Error_func는 cross entropy를 사용한다!!

Multi-Class Classification

Problem

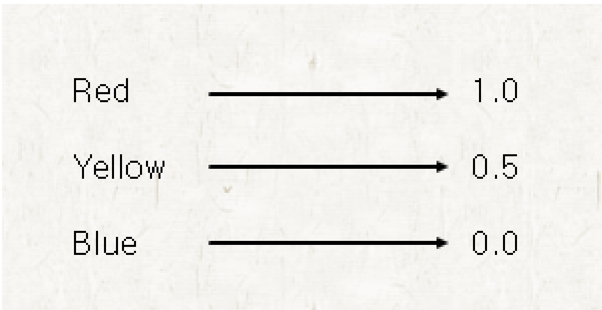
$(\mathbf{x}_1, Red),\ (\mathbf{x}_2, Yellow),\ (\mathbf{x}_3, Blue),\ (\mathbf{x}_4, Red), (\mathbf{x}_5, Blue), \cdots$



Nominal value를 다루는데 예를들어, Red:1,Yello:0.5,Blue:0으로 하면될까??

결론만 따지면 별로다.

왜냐?



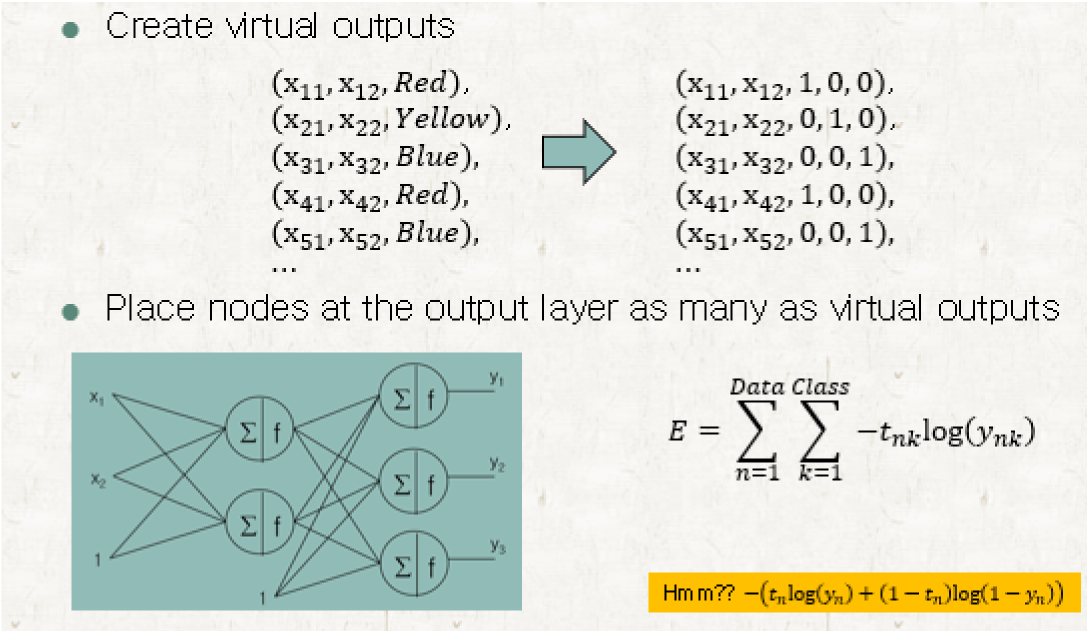
Red가 1인데 0.5를 선택하면 틀린거다. 0을 선택해도 마찬가지다.

그러나 0.5일때 MSE가 더작다...

They are just names. We cannot say that Red>yellow>Blue

그러면 어떻게 해야하나?

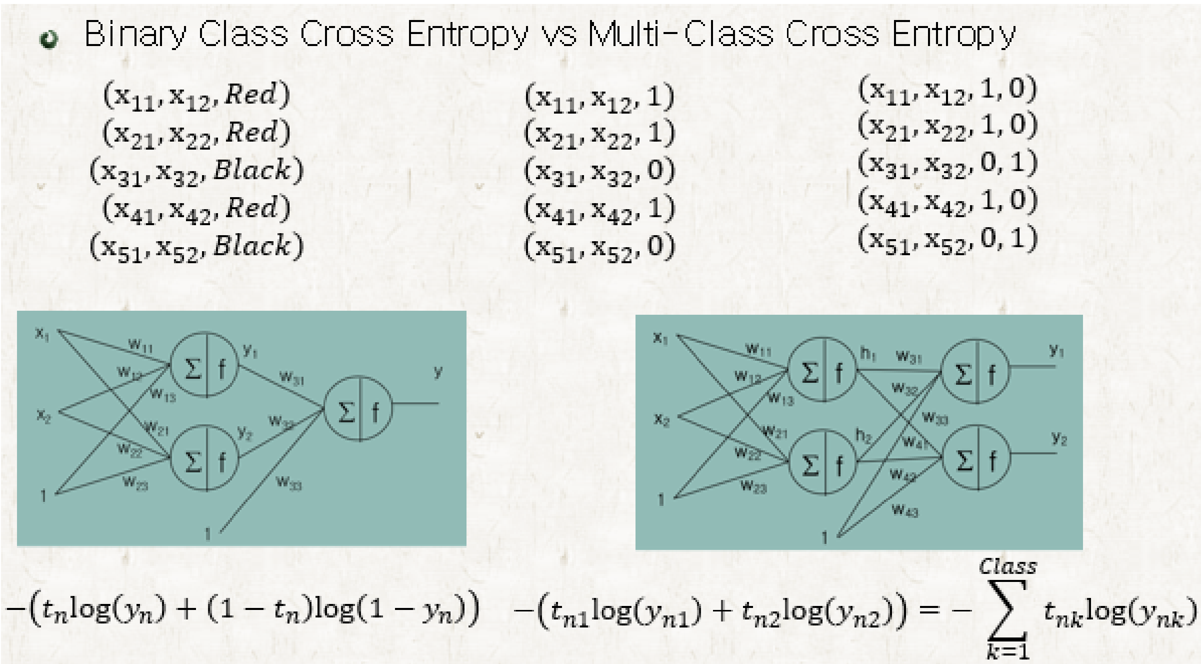
Create virtual outputs!!



원핫인코딩이란(one-hot encoding)?

0으로 이루어진 벡터에 단 한개의 1의 값으로 해당 데이터의 값을 구별하는 것이 원핫 인코딩이다.

Binary Class Cross Entropy vs Multi Class Cross Entropy



binary나 multi나 cross entropy가 다르다 생각할 수 있지만, 같다!

그렇다면 Activation function??

- Outputs of Multi-Class Classification satisfies

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

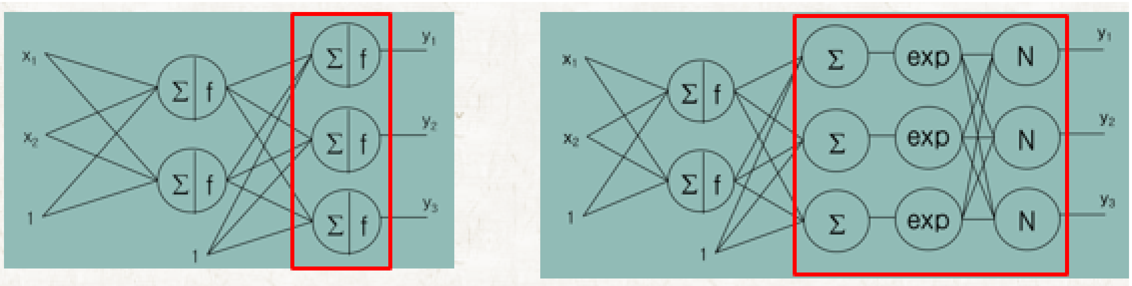
- But, Sigmoid does not satisfy this

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

- So, We use Softmax layer

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

Softmax layer



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

How about Error func?

- Softmax layer + Cross Entropy

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

Multi-Label Classification

Multi-Label:Multi-output(with binary or multi values)

$$\begin{aligned}
&(\mathrm{x}_{11}, \mathrm{x}_{12}, Yes, Yes, Male) \\
&(\mathrm{x}_{21}, \mathrm{x}_{22}, No, Yes, Male) \\
&(\mathrm{x}_{31}, \mathrm{x}_{32}, Yes, No, Female) \\
&(\mathrm{x}_{41}, \mathrm{x}_{42}, No, No, Female) \\
&(\mathrm{x}_{51}, \mathrm{x}_{52}, No, Yes, Male) \\
&\ldots
\end{aligned}$$

위의그림에서

(x11,x12,1,1,1)

(x21,x22,0,1,1) 이런식으로 원핫 인코딩을 적용하면 된다.

output들의 각 label은 독립적이다.

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

$$CE(t_{nk}, y_{nk}) = -\left(t_{nk}\log(y_{nk}) + (1 - t_{nk})\log(1 - y_{nk})\right)$$

Nominal Inputs

Two inputs and one output

$$x_1 \in R$$
$$x_2 \in \{Red, Yellow, Blue\}$$
$$y \in \{0,1\}$$

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

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
Multi-label Classification	ReLU	Sigmoid	CE