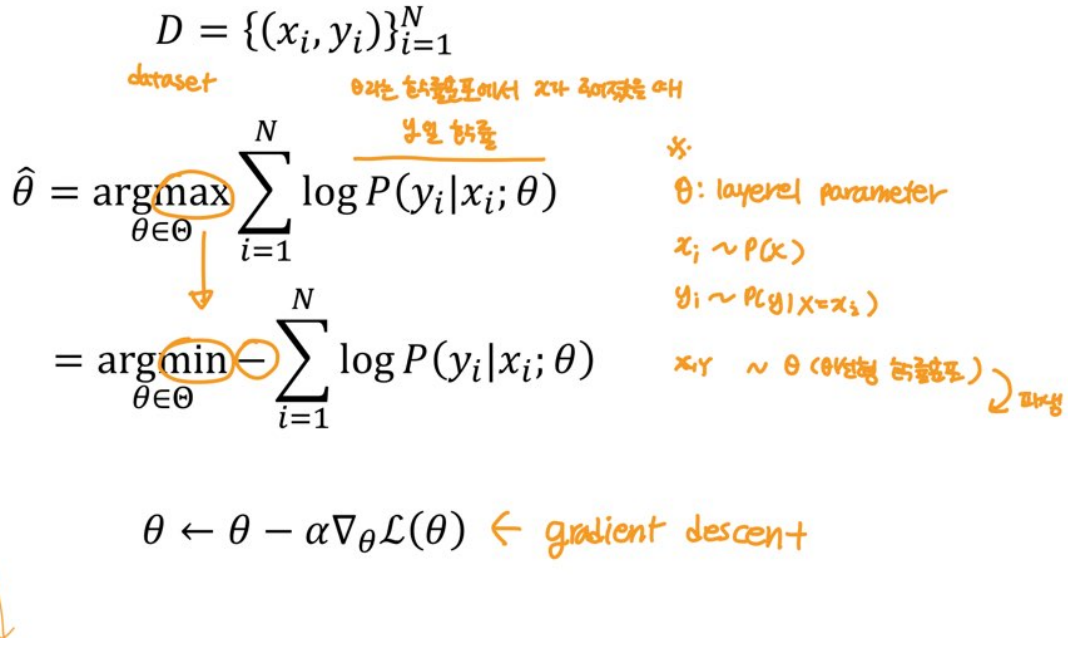
4. Deep Nerual Network with MLE : Equations

MLE Equations

dataset x, y에서 negative log likelihood를 최소화하는 seta를 gradient descent를 통해 찾는 일련의 과정을 식을 통해 표현하였다.



MLE를 신경망에서 구현하는 방법

신경망에 x_i를 넣어서 y_i에 대한 추정값을 구할 수 있다면,

앞에서 구했던 Negative Log Likelihood 식을 cross entropy error 식으로 볼 수 있다.

(softmax layer를 거쳤으며 참값은 one-hot encoding 되어 있다고 가정한다.)

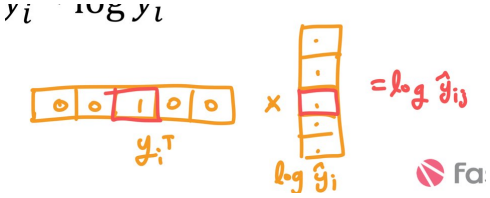
$$D = \{(x_i, y_i)\}_{i=1}^{N}$$

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmin}} - \sum_{i=1}^{N} \log P(y_i | x_i; \theta)$$

$$\hat{y}_i = f_\theta(x_i)$$ <⇒ Neural network 라는 함수에 입력값 x_i를 넣어서 softmax layer의 결과값을 반환

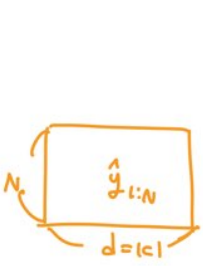
$$-\sum_{i=1}^{N} \log P(y_i | x_i; \theta) = -\sum_{i=1}^{N} y_i^T \cdot \log \hat{y}_i$$

위의 점곱이 어떻게 이루어지는지에 대해 그림으로 표현해 놓았다.



Cross Entropy error

최종적으로 classification에서 softmax layer을 사용해 손실함수를 Cross Entropy error를 사용하여 손실을 최소로 만드는 것은 MLE 관점에서 주어진 데이터를 가장 잘 나타내는 파라미터값들을 찾는 것으로 볼 수 있다.



$$\mathrm{CE}(y_{1:N}, \hat{y}_{1:N}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{d} y_{i,j} \log \hat{y}_{i,j}$$ (d: 클래스의 개수)

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i^T \cdot \log \hat{y}_i,$$ → 위 슬라이드와 $\frac{1}{N}$ 차이가 있지만 gradient-descent 해서 이 때문에 상수이므로 미차피/ 나아가서 상관X

where $y_{1:N} \in \mathbb{R}^{N \times d}, \hat{y}_{1:N} \in \mathbb{R}^{N \times d}$.