# Softmax

Activation Function으로 입력 받은 값의 출력으로 0~1 사이의 값으로 모두 정규화 하며 출력 값들의 총합은 항상 1이 되는 특성을 가진 함수이다. 분류하고 싶은 클래스의 수만큼 출력으로 구성한다. 가장 큰 출력 값을 부여 받은 클래스가 확률이 가장 높은 것으로 이용된다.

분류해야 할 범주 수가 n이고 softmax 함수의 i번째 입력값을 $x_i$, i번째 출력값을 $p_i$라고 할 때 softmax 함수는 다음과 같이 정의할 수 있다.

$$p_i = \frac{\exp(x_i)}{\sum_j^n \exp(x_j)}$$

추측하려는 분류의 $x_i$의 exponential값에 전체 $x_j$의 exponential 합을 나눠주는 방식이다.

예를 들어 x vector의 값이 $\begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}$라고 했을 때, softmax를 거친 p vector의 값은 $\begin{bmatrix} 0.7 \\ 0.2 \\ 0.1 \end{bmatrix}$가 된다. 이 값을 토대로 비용 함수를 구하는 방식이 Cross entropy이다.

# 정보 이론(Information Theory)

정보 이론에서는 정보라는 개념을 수학적으로 구체적으로 표현했다. 정보이론에서 정보는 '놀람의 정도'이다. 예상치 못한 데이터가 더 높은 가치를 매기는 것이다. 한마디로 정보 이론의 핵심은 **잘 일어나지 않는 사건의 정보는 자주 발생할 만한 사건보다 정보량이 많다고 하는 것**이다. 그러므로 정보량의 식은 아래와 같다고 볼 수 있다.

$$정보량 = I(x) = -\log(P(x))$$

P(x)는 확률이며 0~1 사이의 실수 이므로 P(x)는 항상 양수이다.

예를 들어 $x = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}$이고 $P(x) = \begin{bmatrix} 0.7 \\ 0.2 \\ 0.1 \end{bmatrix}$라는 softmax값이 있을 때, P(x) 벡터의 각 원소에 log를 취해주면 아래와 같은 결과가 나온다.

$$I(x) = \begin{bmatrix} 0.154 \\ 0.7 \\ 1 \end{bmatrix}$$

I(x)벡터의 각 원소의 값은 높을수록 정보량이 많다고 볼 수 있다. P(x)에서 $x_3$가 발생할 확률이 0.1로 제일 낮지만 I(x)의 값은 1로 제일 크며 $x_3$가 선택이 된다면 이것에 대한 정보량은 매우 높고 놀랍다고 보는 것이 정보이론이다.

## Shannon Entropy (Information Entropy)

Shannon entropy는 모든 사건 정보량의 기대값을 뜻한다. 전체 사건의 확률 분포의 불확실성의 양을 나타낼 때 쓴다. 즉 Shannon entropy는 정보량(놀람의 정도)들의 평균을 말하며 수식으로는 다음과 같다.

$$\begin{aligned} H(P) &= H(x) \\ &= E_{X\sim P}[I(x)] \\ &= -E_{X\sim P}\big[\log\big(P(x)\big)\big] \\ &= -\sum_{x} P(x) \times \log\big(P(x)\big) \end{aligned}$$

예를 들어 $x = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}$ 이고 $P(x) = \begin{bmatrix} 0.7 \\ 0.2 \\ 0.1 \end{bmatrix}$라는 softmax값이 있을 때, H(P)의 값은 다음과 같다.

$$\begin{aligned} H(P) &= 0.7 \times (-\log(0.7)) + 0.2 \times (-\log(0.2)) + 0.1 \times (-\log(0.1)) \\ &= 0.3482 \end{aligned}$$

즉 이 H(P)값의 의미는 확률 변수의 평균 정보량 = 놀람의 평균적인 정도이다.

## Kullback-Leibler divergence(KLD)

KL-divergence라는 것은 두 확률 분포 차이를 계산하는 데에 사용하는 함수이다.

KL-divergence라는 것은 relative entropy 즉 상대적인 entropy를 말한다. 딥러닝 모델을 만들 때 예로 들면 우리가 가지고 있는 데이터의 분포 P(x)와 모델이 추정한 데이터 분포 Q(x)간의 차이를 KLD를 활용해 구할 수 있다. KLD식은 다음과 같다.

$$\begin{aligned} D_{KL}(P||Q) &= E_{X\sim P}\left[\log \frac{P(x)}{Q(x)}\right] \\ &= E_{X\sim P}[\log\big(P(x)\big) - \log\big(Q(x)\big)\ ] \end{aligned}$$

개념상 두 확률 분포 사이의 distance라고도 볼 수 있다. (실제 distance는 아님)

P(x)와 Q(x)가 동일한 확률 분포일 경우 KLD는 정의에 따라 그 값이 0이 된다.

# Cross entropy

KLD는 cross entropy와 깊은 연관이 있다. P와 Q에 대한 cross entropy H(P,Q)와 KLD와의 관계식은 다음과 같다.

$$H(P,Q) = H(P) + D_{KL}(\mathrm{P}||\mathrm{Q})$$

Cross entropy식은 loss function이므로 최소화 시키는 방향으로 가중치를 업데이트 해야 한다.

$$D_{KL}(\mathrm{P}||\mathrm{Q}) = E_{X\sim P}[\log(\mathrm{P(x)}) - \log(\mathrm{Q(x)})]$$

$D_{KL}(\mathrm{P}||\mathrm{Q})$식은 위와 같다. 그러므로 $H(P,Q)$는 아래와 같다.

$$H(P,Q) = H(P) + E_{X\sim P}[\log(\mathrm{P(x)}) - \log(\mathrm{Q(x)})]$$

위 식에서 P(x)의 값은 현재 가지고 있는 데이터의 분포를 가리키는데 이것은 학습과정에서 바뀌는 것이 아니므로 Q를 최소화 하는 것이 loss function cross entropy를 최소화 하는 것이 된다. 이 뜻은 cross entropy를 최소화 하는 것은 KLD를 최소화 하는것과 같은 의미이며, 현재 가지고 있는 데이터의 분포 P(x)와 모델이 추정한 데이터의 분포 Q(x) 간에 차이를 최소화 한다고 해석할 수 있다. $-E_{X\sim P}[\log(\mathrm{Q(x)})] = -\sum_x P(x) * log(Q(x))$라고 볼 수 있으며 이것을 cross entropy라고 부른다.

$$H(P,Q) = -\sum_x P(x) \times log(Q(x))$$

예를 들어 $\mathrm{x} = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}$ 이고, 실제 데이터 y=P(x)=$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ 이고, x데이터에 대한 softmax 값을 $\mathrm{Q(x)} = \begin{bmatrix} 0.7 \\ 0.2 \\ 0.1 \end{bmatrix}$라고 했을 때, H(P,Q)값을 구해보면 다음과 같다.

$$\begin{aligned} H(P,Q) &= -\sum_x P(x) \times log(Q(x)) \\ &= -(1 \times \log(0.7) + 0 \times \log(0.2) + 0 \times \log(0.1)) \\ &= 0.155 \end{aligned}$$

실제 데이터의 y=P(x)=$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$이라 할 때, H(P,Q)값을 구해보면 다음과 같다.

$$\begin{aligned} H(P,Q) &= -\sum_x P(x) \times log(Q(x)) \\ &= -(0 \times \log(0.7) + 1 \times \log(0.2) + 0 \times \log(0.1)) \\ &= 0.7 \end{aligned}$$

두 가지 경우를 보면 sofmax Q(x)벡터와 실제 y값 P(x) 벡터의 element wise 곱인 것을 알 수 있다. Element wise 곱을 한 모든 원소들의 합에 마이너스를 붙여준 값이 $H(P,Q)$이며 최종 Loss이다.

# Softmax Backpropagation

Softmax function의 식은 아래와 같고,

$$p_i = \frac{\exp(x_i)}{\sum_k^n \exp(x_k)}$$

Loss function의 식은 Cross entropy에 의해 아래와 같이 표현할 수 있다.

$$L = -\sum_j y_j \times \log p_j$$

$\frac{\partial L}{\partial x_i}$을 chain rule을 사용해 구하자.

$y_j$는 정답 벡터의 j번째 요소라는 의미이고, $p_i$는 softmax function의 i번째 즉 정답이 아닌 모든 softmax function의 출력값이라는 의미이다. $p_j$는 softmax function의 함수 j번째 즉 정답 벡터를 잘 예측했을 때의 출력값이라는 의미이다. 그러므로 i=j인 경우는 예측을 잘했을 때의 경우이고, $i \neq j$인 경우는 예측을 잘 못했을 때의 경우이다.

$$\frac{\partial p_j}{\partial x_i} = \frac{\partial}{\partial x_i}\frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

위 식에서 i=j 일 경우와, i≠j인 경우에 대해서 $\frac{\partial p_j}{\partial x_i}$ 를 구해준다.

i = j인 경우)

$$\begin{aligned}
\frac{\partial p_i}{\partial x_i} &= \frac{\exp(x_i) \times \sum_k \exp(x_k) - \exp(x_i) \times \exp(x_i)}{(\sum_k exp(x_k))^2} \\
&= \frac{\exp(x_i) \times (\sum_k \exp(x_k) - \exp(x_i))}{(\sum_k exp(x_k))^2} \\
&= \frac{\exp(x_i)}{\sum_k \exp(x_k)} \times \frac{\sum_k \exp(x_k) - \exp(x_i)}{\sum_k \exp(x_k)} \\
&= \frac{\exp(x_i)}{\sum_k \exp(x_k)} \times \left(1 - \frac{\exp(x_i)}{\sum_k \exp(x_k)}\right) \\
&= p_i \times (1 - p_i)
\end{aligned}$$

$i \neq j$인 경우)

$$\frac{\partial p_i}{\partial x_j} = \frac{0 - \exp(\mathrm{x_i}) \times \exp(\mathrm{x_j})}{(\sum_k exp(\mathrm{x}_k))^2}$$

$$= -\frac{\exp(\mathrm{x_i})}{\sum_\mathrm{k} \exp(\mathrm{x_k})} \times \frac{\exp(\mathrm{x_j})}{\sum_\mathrm{k} \exp(\mathrm{x_k})}$$

$$= -p_i \times p_j$$

$\frac{\partial \mathrm{L}}{\partial x_i}$를 구해주면 다음과 같다.
$\frac{\partial p_j}{\partial x_i}$는 i와 j가 같을 때와 다를 때 마다 값이 다르므로 다음과 같이 식을 유도할 수 있다.

$$\frac{\partial \mathrm{L}}{\partial x_i} = \frac{\partial}{\partial x_i}\left(-\sum_j y_j \times \log p_j\right)$$

$$= -\sum_j y_j \times \frac{\partial \log p_j}{\partial x_i}$$

$$= -\sum_j y_j \times \frac{1}{p_j} \times \frac{\partial p_j}{\partial x_i}$$

$$= -y_i \times \frac{1}{p_i} \times p_i \times (1 - p_i) - \sum_{i \neq j}\left(y_j \times \frac{1}{p_j} \times (-p_i \times p_j)\right)$$

$$= -y_i + y_i \times p_i + \sum_{i \neq j}(y_j \times p_i)$$

$$= -y_i + \sum_\mathrm{j} y_j \times p_i$$

$$= -y_i + p_i \times \sum_\mathrm{j} y_j$$

$$= -y_i + p_i \times 1$$

$$= -y_i + p_i$$

$\frac{\partial \mathrm{L}}{\partial x_i} = o$이다.