경사하강법을 이용한 최적화

$$cost(w,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

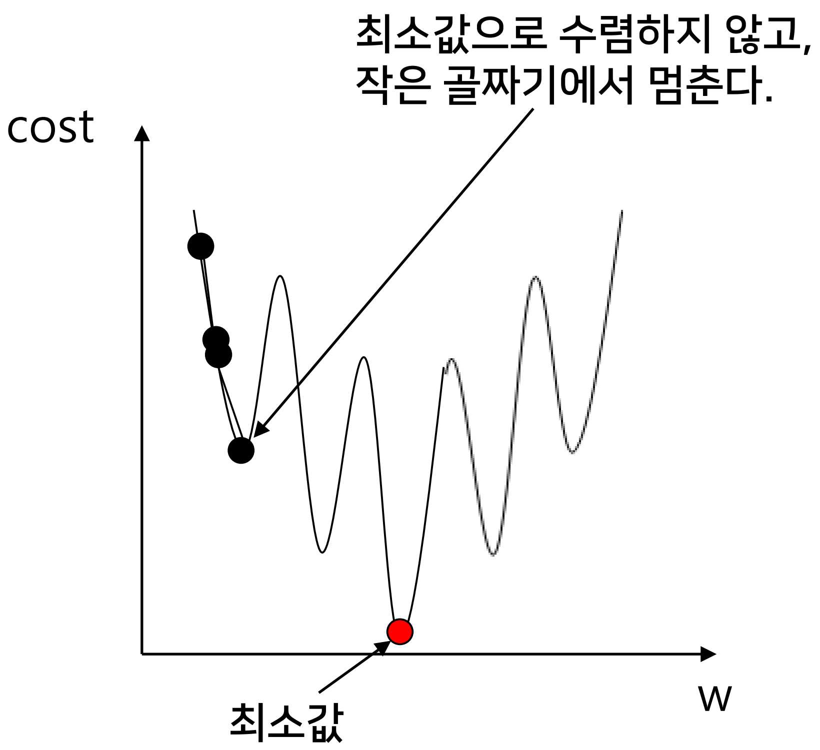
$$H(x) = \frac{1}{1 + e^{-(wx+b)}}$$

$$\underset{w,b}{minimize}\ cost(w,b)$$

- 경사하강법을 이용해 최적의 w,b 추정

시그모이드 함수 사용에 따른 cost 함수의 특성

- sigmoid 함수에 대한 cost함수는 지역적 최소점이 존재
  > cost 함수로 이용하기 부적절

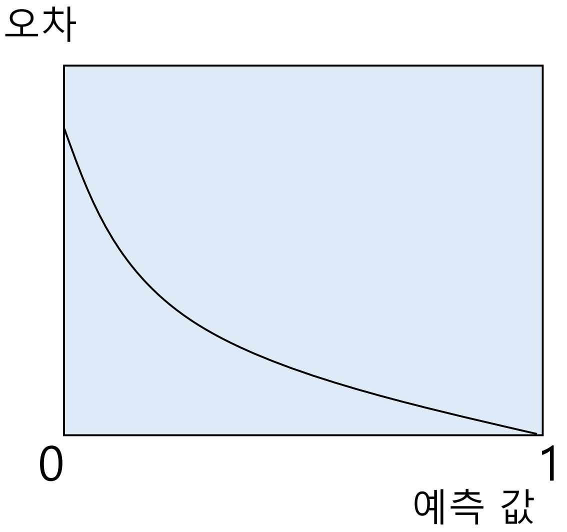


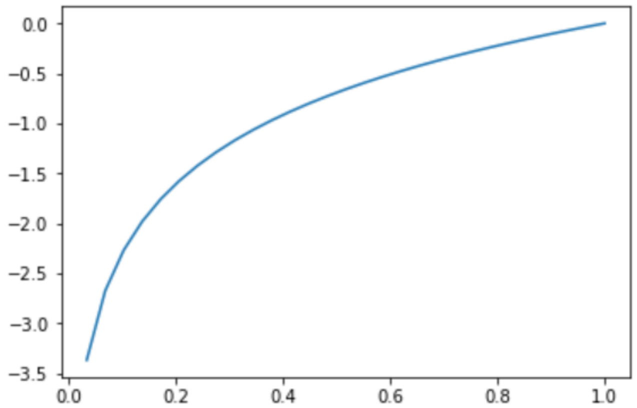
$$cost(w,b) = \frac{1}{m}\sum_{i=1}^{m}\left(H\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$H(x) = \frac{1}{1+e^{-(wx+b)}}$$

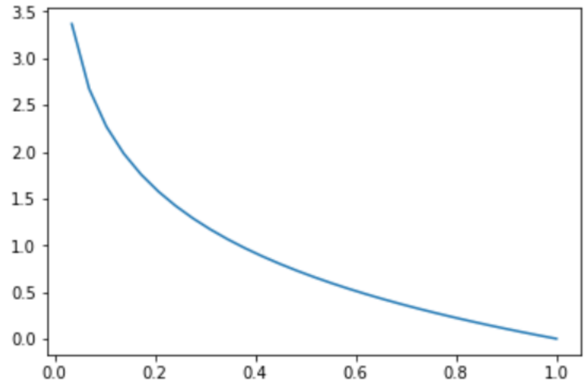
Log 기반 cost 함수

- **학습 데이터 레이블(Target) : 1일 때**
  - > 예측 값이 0이면 오차 큰 값(최대 ∞)
  - > 예측 값이 1이면 오차 작은 값(최소 0)





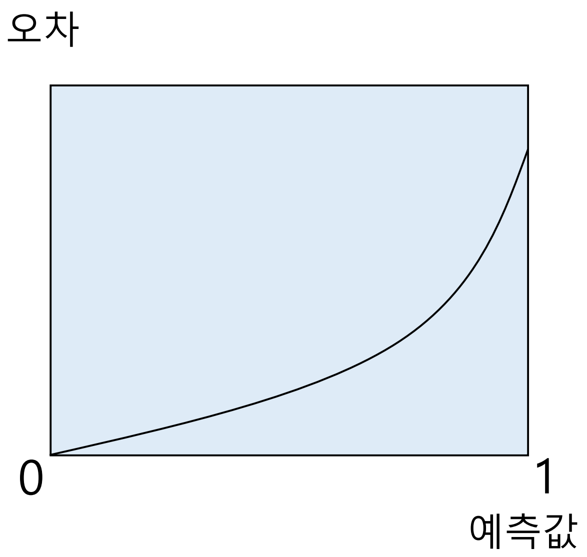
$\log(H(x))$

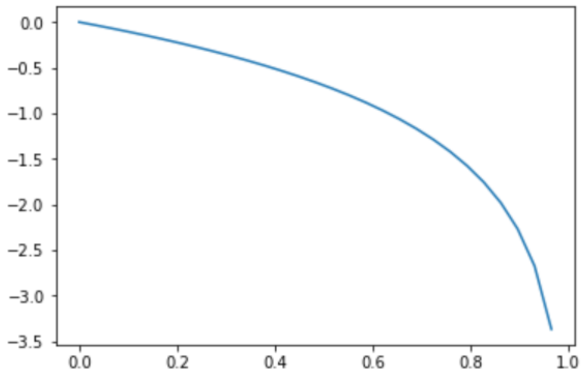


$-\log(H(x))$

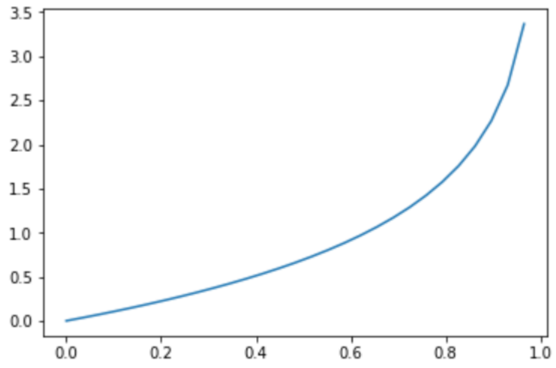
Log 기반 cost 함수

- **학습 데이터 레이블(Target) : 0일 때**
  - > 예측 값이 0이면 오차 큰 값(최소 0)
  - > 예측 값이 1이면 오차 작은 값(최대 ∞)





$$\log(1 - H(x))$$

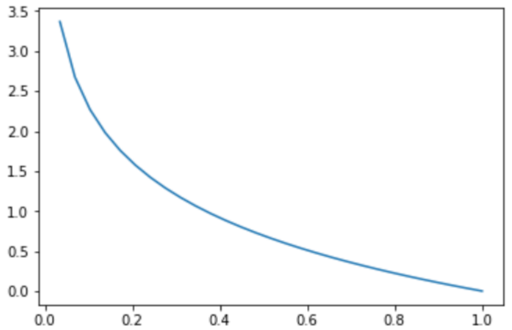


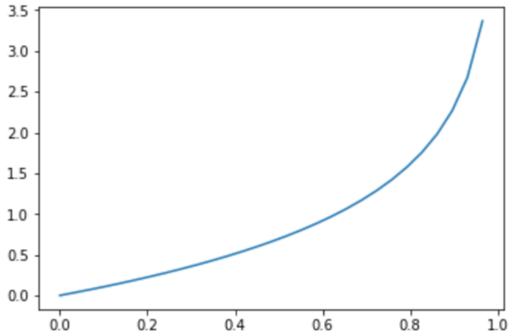
$$-\log(1 - H(x))$$

Log 기반 cost 함수

로지스틱 회귀에서 오차

$$c(H(x),y)=\begin{cases}-\log\left(H(x)\right) & : y=1\\ -\log\left(1-H(x)\right) & : y=0\end{cases}$$





$\Downarrow$

y=1일때, y=0일 때를 모두 고려한 cost 함수

$$c(H(x),y)=-\left(y\log(H(x))+(1-y)\log(1-H(x))\right)$$

로지스틱 회귀 cost 함수

$$cost(w)=-\frac{1}{m}\sum ylog(H(x))+(1-y)log(1-H(x))$$

Cost 함수 직접 계산하기

x	y
1	0
2	0
7	1
8	1

$$cost(w) = -\frac{1}{m}\sum ylog(H(x)) + (1-y)log(1-H(x))$$

cost(w = -3)
= - (log(1-H(1)) + log(1-H(2)) + log(H(7)) + log(H(8)) / 4)
= - (log(1- 0.04742) + log(1- 0.0024726) + log(7.5e-10) + log(3.77e-11)) / 4
= - (-0.0485 + -0.0024 + -21.010 + -24.0013) / 4
= 11.26

H(1)=1/(1+exp(-3×1)=0.04742
H(2)=1/(1+exp(-3×2)=0.0024626
H(7)=1/(1+exp(-3×7)=7.5e-10
H(8)=1/(1+exp(-3×8)=3.77e-11

로지스틱 회귀 경사하강법

로지스틱 회귀

$$H(x) = \frac{1}{1 + e^{-(wx+b)}}$$

$$cost(w, b) = -\frac{1}{m}\sum ylog\big(H(x)\big) + (1 - y)\log(1 - H(x))$$

선형 회귀

$$H(x) = wx + b$$

$$cost(w, b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

경사하강법

$$w = w - \alpha\frac{\partial}{\partial w}cost(w)$$

로지스틱 회귀 학습

- 학습 데이터
  - > 학습데이터 x와 출력 레이블 → $(x, [0|1])$
- 학습 모델 정의
  - > 시그모이드(기울기 w, x축 이동량 b) → $H(x) = \frac{1}{1 + e^{-(wx+b)}}$
- 학습방법
  - > cost 함수 정의 → $cost(w,b) = -\frac{1}{m}\sum ylog\big(H(x)\big) + (1-y)\log(1-H(x))$
  - > 비용이 최소화 되도록 최적화 → $\underset{w,b}{minimize}\ cost(w,b)$
  - > 경사하강법을 이용한 w, b 추정 → $w = w - \alpha \frac{\partial}{\partial w} cost(w)$
- 예측
  - > w,b를 사용해 x 분류 → $true, if\ \mathrm{H}(x) > 0.5$

다중 클래스에서 cost함수 정의

- 다중 class에 맞는 별도의 cost 함수 필요

목적값 [1 0 0]
추정값 [0.9 0.05 0.05]

최대값

cost = 낮은값(최소 0)

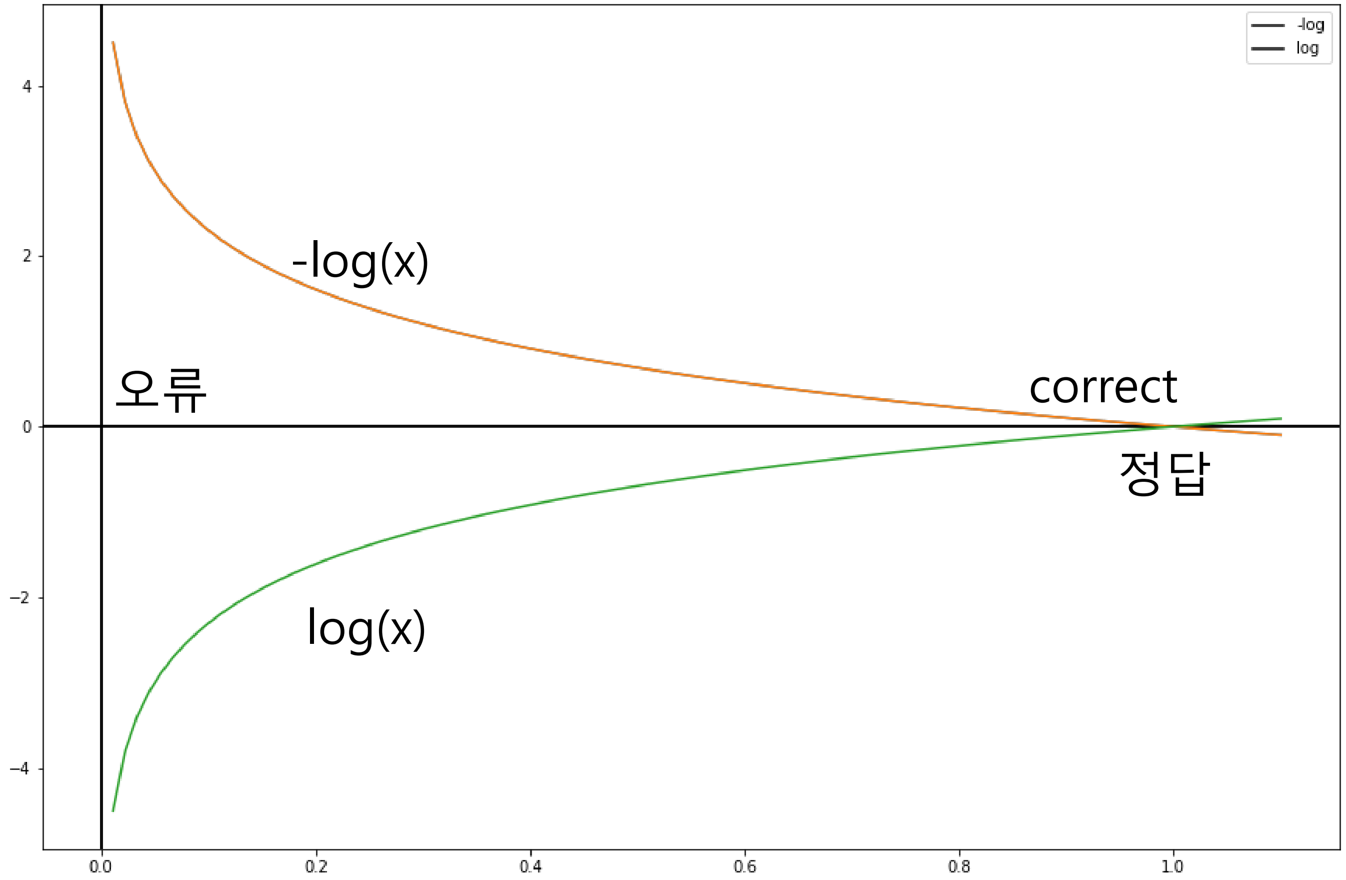
목적값 [1 0 0]
추정값 [0.05 0.9 0.05]

최대값

cost = 높은값(최대∞)

다중 클래스에서 cost함수 정의

- -log(x) 함수 : 정답에 가까우면 출력값이 0, 오류일 경우에는 무한대



다중 클래스에서 cost함수 정의

정답인 경우

$$\begin{array}{ll} \text{목적값} & [1 \quad 0 \quad 0] \\ \text{추정값} & [0.9 \quad 0.05 \quad 0.05] \end{array}$$

$$cost = -\underbrace{p(x)}_{\text{목적값}}\ \underbrace{logq(x)}_{\text{추정값}}$$

$$\begin{array}{ll} \text{목적값} & [1 \quad 0 \quad 0] \\ -log(\text{추정값}) & [0.1 \quad 2.99 \quad 2.99] \end{array}$$

$\rightarrow$ cost = 0.1

오답인 경우

$$\begin{array}{ll} \text{목적값} & [1 \quad 0 \quad 0] \\ \text{추정값} & [0.05 \quad 0.9 \quad 0.05] \end{array}$$

$$cost = -p(x)\ logq(x)$$

$$\begin{array}{ll} \text{목적값} & [1 \quad 0 \quad 0] \\ -log(\text{추정값}) & [2.99 \quad 0.1 \quad 2.99] \end{array}$$

$\rightarrow$ cost = 2.99

다중 클래스에서 cost함수 정의

$$H(p,q) = -\sum_{x} p(x)\, log q(x)$$

타겟모델과 예측 확률 모델의 차를 의미하는 Cross Entropy

모델 학습이 잘 된 경우

$$-P(x)logQ(x) = -\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}log0\\log1\end{bmatrix} = -\begin{bmatrix}1\times log0\\0\times log1\end{bmatrix} = -(-\infty + 0) = \infty$$

$$\log 0 = \infty, \qquad \log 1 = 0$$

모델 학습이 잘 안 된 경우

$$-P(x)logQ(x) = -\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}log1\\log0\end{bmatrix} = -\begin{bmatrix}1\times log1\\0\times log0\end{bmatrix} = -(0 + 0) = 0$$

다중 클래스에서 cost함수 정의

s(x)$_i$: 0~1 사이의 값을 갖는 확률 벡터 값

$$S(x) = softmax(wx + b) \xrightarrow{\text{확률벡터}} S(x) = [S_1\ S_2\ S_3\ S_4\ S_5]$$

ex) [0.05 0.05 0.8 0.03 0.07]

$$L = [L_1\ L_2\ L_3\ L_4\ L_5]$$

ex) [0 0 1 0 0]

$$D(s(x), L) = Loss = -\sum_i L_i \log(S(x)_i) \longrightarrow$$ cost

$$\mathcal{L} = \frac{1}{N}\sum D(S(x), L) \longrightarrow$$ 전체 cost

오차 함수 직접 계산하기

x	y
1	0 1
2	0 1
7	1 0
8	1 0

- 초기값
b=0
w=[-1, 1]

wx

--1. 1.
--2. 2.
--7. 7.
--8. 8.

s = sofmax

0.11, 0.88
0.01, 0.98
0.00, 0.99
0.00, 0.99

S

L

0	1
0	1
1	0
1	0

log(s)

-2.12	-0.12
-4.01	-0.018
-14	0
-16	0

L log(s)

-0.	-0.12
-0.	-0.018
-14	0.
-16	0.

-sum(L log(s))

D

[0.12
0.018
14.0
16.0]

mean(-sum(L log(s)))

[7.53]

$\mathcal{L}$

cost= 7.53

클래스 레이블 예측

- 예측값은 세 개 중에서 가장 확률값이 높은 class의 위치 값

$$h = softmax(wx + b)$$

$$\hat{y} = argmax(h)$$

Ex)

$$h = [0.05 \quad 0.8 \quad 0.15]$$

$$\hat{y} = 1$$

Softmax 다중 클래스 학습

- **학습 데이터**
  - 학습데이터 x와 one-hot 코딩 레이블

$$(x, [1\ 0\ 0\ 0\ 0\ 0])$$

- **학습 모델 정의**
  - softmax

$$s(x) = softmax(wx + b)$$

- **학습방법**
  - 크로스 엔트로피 기반 cost 함수 정의

$$D(s(x), L) = -\sum_i L_i \log(S(x)_i) \qquad \mathcal{L} = \frac{1}{N}\sum D(S(x), L)$$

  - 비용이 최소화 되도록 최적화

$$\underset{w,b}{minimize}\ cost(w, b)$$

  - 경사하강법을 이용한 w, b 추정

$$w = w - \alpha \frac{\partial}{\partial w} cost(w)$$

- **예측**
  - w,b를 사용해 함수의 최대값 위치

$$h = softmax(wx + b)$$

$$y = argmax(h)$$