May 14, 2021

e.g. Given

Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \dots (x^{(m)}, y^{(m)})\}$

and $x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$, $x_0 = 1$, $y \in \{0, 1\}$

and $h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$,

How do we choose parameters θ?

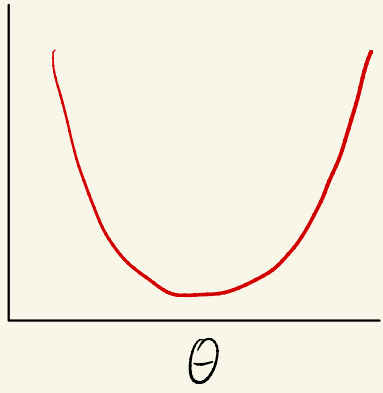
If we use $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (h_\theta(x^{(i)}) - y^{(i)})^2$,

where $h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$, then $J(\theta)$ will be non-convex,

Non-Convex

$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$

Convex



$h_\theta(x) = \theta_0 + \theta_1 x$

↳ many local minima, accuracy may not converge

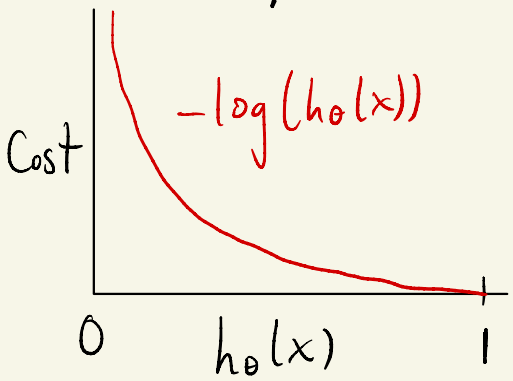
Logistic Regression Cost Function:

Originally: $Cost(h_\theta(x), y) = \frac{1}{2}(h_\theta(x) - y)^2$

Now: $$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

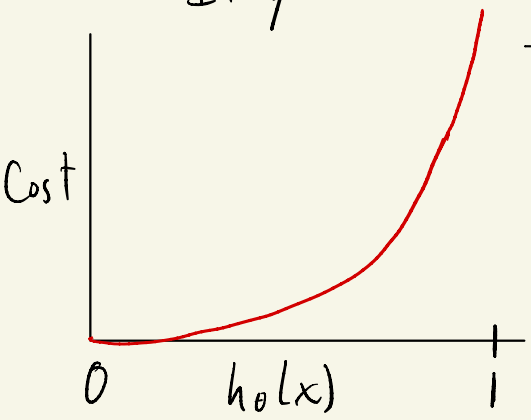
* Guarantees convex cost function

If $y = 1$



→ For $y = 1$, if $h_\theta(x) = 0$ (i.e. prediction is 0 but $y = 1$), then the learning algorithm is heavily penalized

If $y = 0$



→ Same thing as $y = 1$, but opposite.

Simplified:

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Since $y = 0$ or 1 always:

↳ $$\boxed{\text{Cost}(h_\theta(x), y) = -y \log(h_\theta(x)) - (1 - y) \log(1 - h_\theta(x))}$$

↳ $y = 1$; $\text{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) - 0$

$y = 0$; $\text{Cost}(h_\theta(x), y) = 0 - \log(1 - h_\theta(x))$

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$\therefore$ $$\boxed{J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right]}$$

So how do we get $\min J(\theta)$?

Gradient Descent For Logistic Regression:

$$\frac{\partial}{\partial \theta} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

repeat {

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

} simultaneously update all θ_j for $j = 0 \ldots n$

↳ same as for linear regression, but $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

There are also more complex optimization algos

↳ conjugate gradient
↳ BFGS
↳ L-BFGS

- no need to manually pick α
- often faster than gradient descent