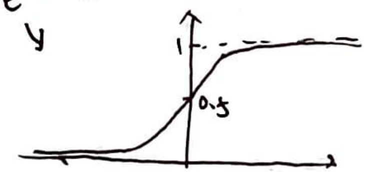
Logistic Regression Model

want $0 \le h_\theta(X) \le 1$

$h_\theta(x) = g(\theta^T x)$

Sigmoid / Logistic function $g(z) = \dfrac{1}{1+e^{-z}}$

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



Interpretation of Hypothesis Output

$h_\theta(x)$ estimated probability that $y=1$ on input X

$h_\theta(x) = p(y=1 \mid x;\theta)$ probability that $y=1$. given X, parametrized by θ.

$p(y=0 \mid x;\theta) = 1 - p(y=1 \mid x;\theta)$

Decision Boundary

Suppose predict $y=1$ if $h_\theta(x) \ge 0.5$ $\quad g(z) \ge 0.5$ when $z \ge 0$

predict $y=0$ if $h_\theta(x) < 0.5$ $\quad g(\theta^T x) \ge 0.5$ when $\theta^T X \ge 0$

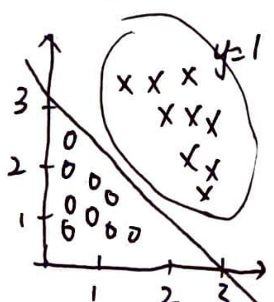
$h_\theta(x) = g(\underset{-3}{\theta_0} + \underset{1}{\theta_1} x_1 + \underset{1}{\theta_2} x_2)$

$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

predict "$y=1$" if $\underbrace{-3 + x_1 + x_2}_{\theta^T X} \ge 0$.

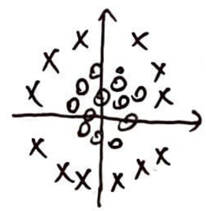
or

$x_1 + x_2 \ge 3$



$\rightarrow$ decision boundary (is a property of hypothesis)

Nonlinear decision boundaries.



$h_\theta(x) = g(\overset{-1}{\theta_0} + \overset{0}{\theta_1} x_1 + \overset{0}{\theta_2} x_2 + \overset{1}{\theta_3} x_1^2 + \overset{1}{\theta_4} x_2^2)$

$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$ predict $y=1$ if $-1 + x_1^2 + x_2^2 \ge 0$

$x_1^2 + x_2^2 = 1$. Decision Boundary

Cost Function For Logistic Regression Intuition

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

m examples $\quad x \in \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix}, \ x_0 = 1, \ y \in \{0, 1\}$

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

How to choose parameters θ?

Linear Regression $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (h_\theta(x^{(i)}) - y^{(i)})^2$ } for logistic

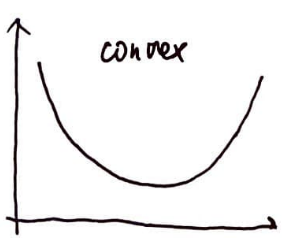
$\mathrm{cost}(h(x^{(i)}), y)$

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

$$\frac{1}{1 + e^{-\theta^T x}}$$

non-convex

convex



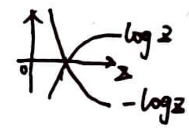
Logistic regression cost Function

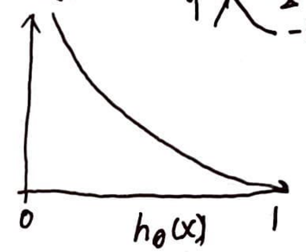
$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

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

Note $y = 0$ or 1 only

$y = 1$

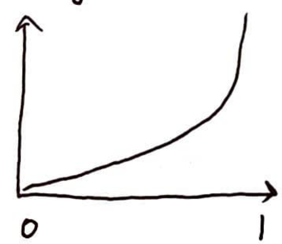




Cost = 0 if $y = 1$.

as $h_\theta(x) \to 0$, Cost $\to \infty$

$y = 0$



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

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

principles of maximum likelihood analysis

To fit parameters θ:

$$\min_\theta J(\theta)$$

To make a prediction given new x:

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

Want $\min_\theta J(\theta)$:

Repeat {

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

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

Simplified Cost Function and Gradient Descent

Note: [6:53 - the gradient descent equation should have a 1/m factor]

We can compress our cost function's two conditional cases into one case:

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

Notice that when y is equal to 1, then the second term $(1-y)\log(1-h_\theta(x))$ will be zero and will not affect the result. If y is equal to 0, then the first term $-y\log(h_\theta(x))$ will be zero and will not affect the result.

We can fully write out our entire cost function as follows:

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

A vectorized implementation is:

$$\begin{aligned} h &= g(X\theta) \\ J(\theta) &= \frac{1}{m}\cdot\left(-y^T\log(h) - (1-y)^T\log(1-h)\right) \end{aligned}$$

Gradient Descent

Remember that the general form of gradient descent is:

$$\begin{aligned} & Repeat\ \{ \\ & \ \theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) \\ & \} \end{aligned}$$

We can work out the derivative part using calculus to get:

$$\begin{aligned} & Repeat\ \{ \\ & \ \theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \\ & \} \end{aligned}$$

Notice that this algorithm is identical to the one we used in linear regression. We still have to simultaneously update all values in theta.

A vectorized implementation is:

$$\theta := \theta - \frac{\alpha}{m}X^T(g(X\theta) - \vec{y})$$

One iteration of gradient descent simultaneously performs these updates:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)}$$

$$\vdots$$

$$\theta_n := \theta_n - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_n^{(i)}$$

We would like a vectorized implementation of the form $\theta := \theta - \alpha\delta$ (for some vector $\delta \in \mathbb{R}^{n+1}$).

What should the vectorized implementation be?

- $\theta := \theta - \alpha \frac{1}{m} \sum_{i=1}^{m} [(h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}]$