

Gradient descent.

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

$$\theta_j = \theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} + \frac{\lambda}{m} \theta_j \right]$$

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

Normal Equation.

$$\theta = (X^T X + \lambda \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix})^{-1} X^T y.$$

$n+1 \times n+1$

(n features)

it solves non-invertible case.
for $m \leq n$

Logistic Regression

$$J(\theta) = - \left[\frac{1}{m} \sum_{i=1}^m y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\theta}(x^{(i)})) \right] + \frac{\lambda}{2m} \sum_{j=1}^m \theta_j^2$$

Hypothesis is different

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

Obj = ~~cost~~ Advanced optimization

$$\begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

function [jVal, gradients] = costFunction(theta)

jVal = code to compute $J(\theta)$

$$J(\theta) = \left[-\frac{1}{m} \sum_{i=1}^m y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) \right] + \frac{\lambda}{2m} \sum_{j=1}^n \theta_j^2$$

gradients(1) = code to compute $\frac{\partial}{\partial \theta_0} J(\theta)$

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

gradients(2) = code to compute $\frac{\partial}{\partial \theta_1} J(\theta)$

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