Still An intuation of Machine Learning

A thing called neural network
which I am trying to do some trival thing with

Eric Wang Richard Zhong

November 3, 2017

SHUOSC

log-linear regression

Costfuction of linear regression

$$y = W^T x + b$$

More generally

$$\ln y = W^T x + b$$

$$y = e^{W^T x + b}$$

Generalized linear model

$$y = g(W^T x + b)$$

Why Logistic Regression

Considering a binary classfication problem

$$y \in \{0, 1\}$$

while

$$z = W^T x + b, z \in \{-\infty, +\infty\}$$

trying to contact y with our predicted result

$$z = W^T x + b$$

Logistic Function

First considering the unit-step function

$$y = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$$

Not a continuous function,considering logistic function

$$y = \frac{1}{1 + e^{-z}}$$

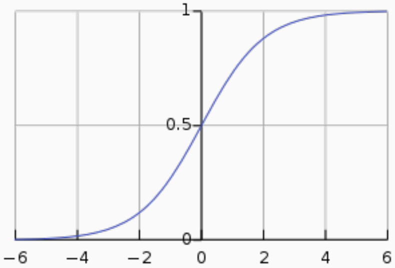


Figure 1: Sigmoid Function

Cost Function

In Linear Regression

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

We Define Costfunction

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

In Logistic Regression

$h(x)$ can be a non-linear function
and have many local optima

Cost Function

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

To Simplify

$$J(h_\theta(x), y) = -ylog(h_\theta(x)) - (1 - y)log(1 - h_\theta(x))$$

To sum us

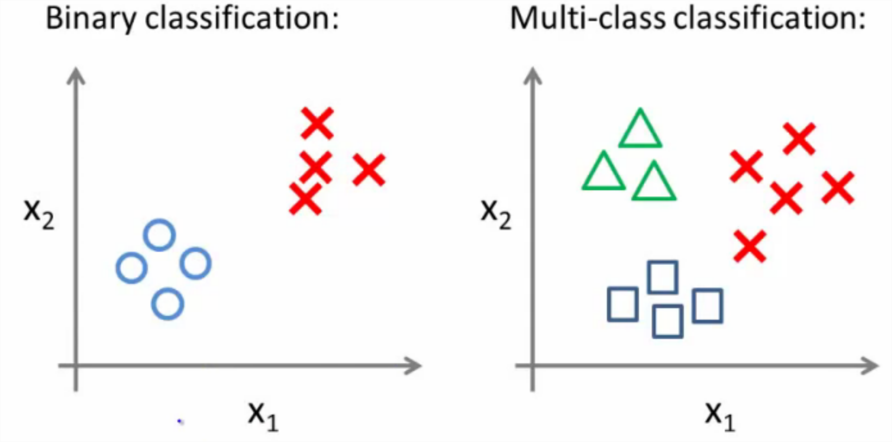
$$J(h_\theta(x), y) = -\frac{1}{m}\sum_{i=1}^{m} y^i log(h_\theta(x^i)) - (1 - y^i)log(1 - h_\theta(x^i))$$

Gradient Desent

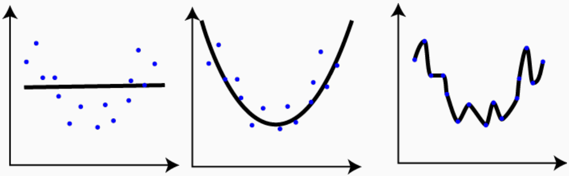
Repeat

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

One VS all


Binary classification:
Multi-class classification:
x_2
x_1
x_2
x_1

Overfitting



To solve this problem we shall reduce the number of features or used regularization

Regularzed Linear Regression

In Gradient Decent

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

We plus an regularization term $\lambda \theta_j$ to each θ (except the bias one)

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

Finally the Costfunction goes to be

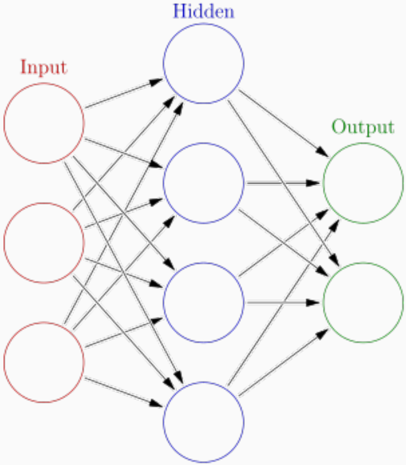
$$J(\theta) = \frac{1}{2m} [\sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2 + \lambda \sum_{j=1}^{m} \theta^2]$$

Regularzied Logistic Regression

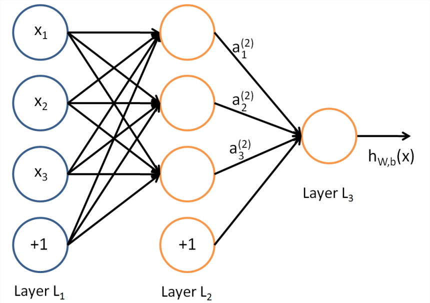
$$\theta_0 = \theta_0 - \alpha \frac{1}{m} (\sum_{i=1}^{m} (h_\theta(x^i) - y^i) x_0^i$$

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

Simple neural network



Simple neural network



$$
\begin{aligned}
a_1^{(2)} &= f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \\
a_2^{(2)} &= f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \\
a_3^{(2)} &= f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)}) \\
h_{W,b}(x) = a_1^{(3)} &= f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})
\end{aligned}
$$

- a_i^j=activation of unit i in Layer j
- W=the weight from Layer j to Layer j+1

Vectorization

$$X = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} W = \begin{bmatrix} W_{10} & W_{11} & W_{12} & W_{13} \\ W_{20} & W_{21} & W_{22} & W_{23} \\ W_{30} & W_{31} & W_{32} & W_{33} \end{bmatrix}$$

$$z = WX$$

$$a_1^2 = f(z_1^2)$$

$$a_2^2 = f(z_2^2)$$

$$a_3^2 = f(z_3^2)$$

$$h_w(x) = a_1^3 = f(z_1^3)$$

Example

Table 1: And

x_1	x_2	$h_w(x)$
0	0	0
0	1	0
1	0	0
1	1	1

Cost function

$$J(W,b;x,y) = \frac{1}{2}\left\| h_{W,b}(x) - y \right\|^2 .$$

$$J(\theta) = \ y \ \log h_\theta(x\ ) + (1-y\ ) \log(1 - h_\theta(x\ ))$$