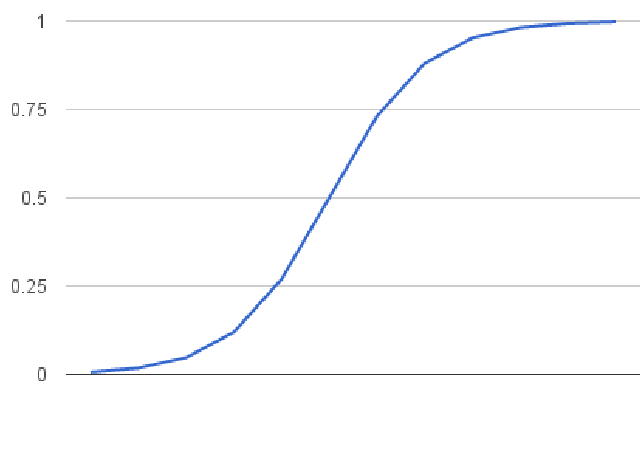
Logistic Regression

Logistic regression is a binary classification method based on logistic function.

Logistic function is a S-shape function whose output values are between 0 and 1 for any input values. It is defined as:

$$g(z) = \frac{1}{1+e^{-z}}$$

A typical logistic function looks like this:



Logistic Function

Logistic regression use a similar equation to predict an output value for any input value. A typical logistic regression equation is as follows:

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

Here θ is a vector representing the weights or coefficient values related to x. Keep in mind that this equation is different from linear regression where a linear relationship exists in between the input values (x) and output values (y or $h_\theta(x)$), as in

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

The output value $h_\theta(x)$ denotes the probability of y = 1 on input x. That is,

$$h_\theta(x) = P(y=1 | x, \theta)$$

This probability value is then transformed into a binary value (0 or 1) in order to make a probability prediction. Specifically, suppose we predict

$$y = 1 \text{ if } h_\theta(x) \geq 0.5$$

$$y = 0 \text{ if } h_\theta(x) < 0.5$$

That is equal to

$$y = 1 \text{ if } \theta^T x \geq 0$$

$$y = 0 \text{ if } \theta^T x < 0$$

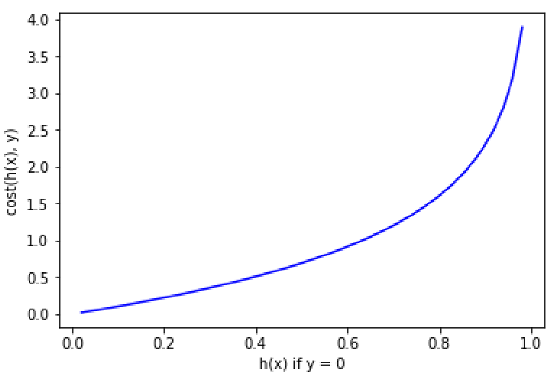
Now it is clear that $\theta^T x = 0$ defines the **decision boundary** for the binary classification.

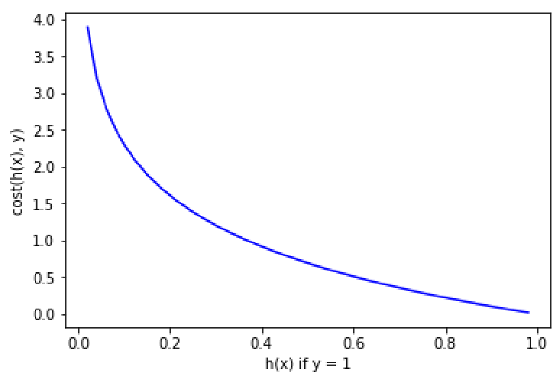
Similar to linear regression, we want to find the optimum θ that defines the decision boundary and mathematically we optimize the cost function defined as follows:

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

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

Here the cost function for each (x, y) pair is defined in a way that when y = 1 and ideally $h_\theta(x) = 1$ or when y = 0 and ideally $h_\theta(x) = 0$, the cost function is minimum (0). Below figures shows how the cost function change with h(x) for y = 0 and y = 1 respectively.





Similar to linear regression, the cost function $J(\theta)$ can be minimized using gradient descent. Below is the pseudo code:

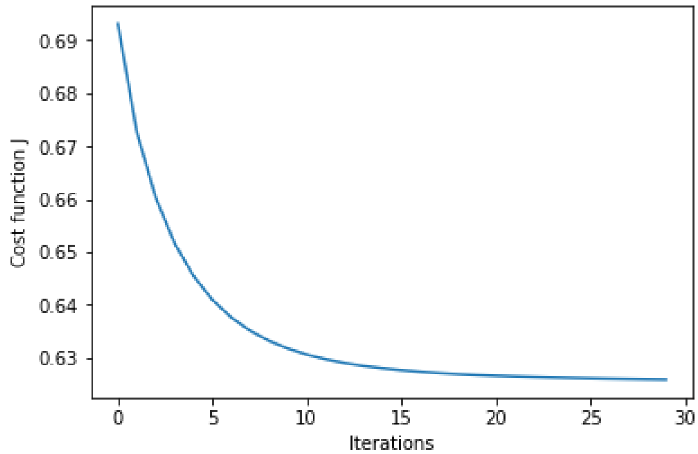
repeat until converge:

{

$$\theta := \theta - \alpha / m * \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

}

The figure below show how the cost function decrease with number of iterations for data in the Titanic Kaggle competition with slide modification (see Python code file).



References:

1, Machine Learning by Andrew Ng, from Coursera

2, Titanic Kaggle competition: https://www.kaggle.com/c/titanic/data