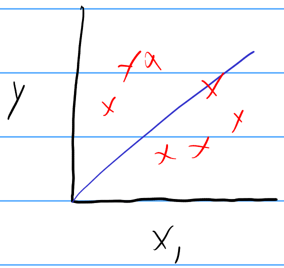
Linear Regression

- Used for simple prediction
- Sophisticated line of best fit
- Establish hypothesis function



$$h_\theta(x) = \theta_0 + \theta_1 x_1 = \theta^T X$$

- Objective is to minimize distance of all points from this line
- We can evaluate this with a cost function

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

- Utilizing the cost function, we can adjust the parameters through gradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$ Simultaneously update all θ

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

Logistic Regression

- Used to solve classification problems, where y is discrete
- We would like $0 \le h_\theta(x) \le 1$
- In linear regression, $h_\theta(x) = \theta^T X$
- In logistic regression, we use an activation function: sigmoid/logistic

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

- We establish a decision boundary to classify the output

Assume $h_\theta X = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ $\quad \theta = [-3, 1, 1]$

We predict $y=1$ if $-3 + x_1 + x_2 \ge 0$ Why?

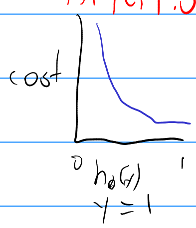
$$x_1 + x_2 \ge 3$$

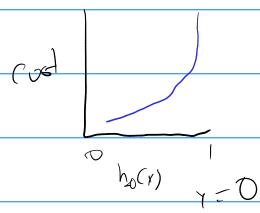
$x_1 + x_2 = 3$ ← Decision boundary line

- Cost function has cool intuition

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

Intuition

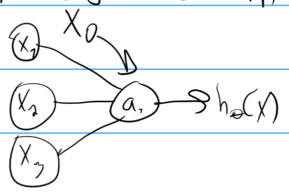




$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^{m} y^i \log(h_\theta(x^i)) + (1 - y^i) \log(1 - h_\theta(x^i)) \right]$$

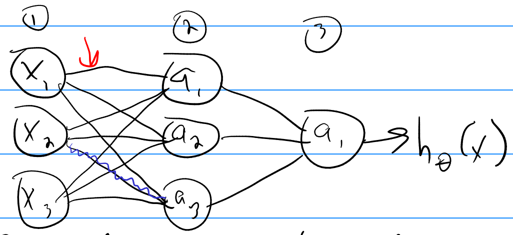
Neural Networks!

- Useful when input space is way too fucking large, your hypothesis is non-linear, and you want to be able to have super wack separating line
- A logistic unit



- All input X modified by respective weights which make up $\Theta^T X$, the input to sigmoid, whose output is referred to as the node's activation

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$



Θ^1_{11} Θ^1_{23}

- Consider a vectorized implementation!

$a_1^2 = g(z_1^2)$ $z_1^2 = \Theta_1^{1T} X$

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

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

$z^2 = \Theta^{1T} X$

$a^2 = g(z^2)$

$h_\theta(X) = a^3 = g(z^3) = g(\Theta^2 a^2)$

- Some intuition