Logistic Regression

Fade-In

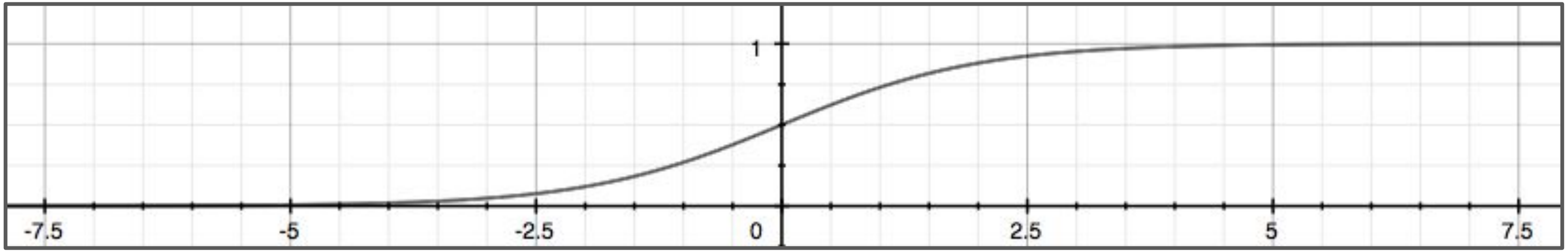
- Machine Learning concepts and Training set
- Training set type : supervised Learning (for prediction) / unsupervised (for clustering)
- Linear Regression (continuous value prediction)
- Logistic Regression (discrete value prediction) - Classification
- Neural Network

Logistic Regression (discrete value prediction) - Classification

Share

Classification: $y = 0$ or 1

$h_\theta(x)$ can be > 1 or < 0



Reminder: Hypothesis for Linear Regression

$$\boxed{h_\theta(x) = \theta_0 + \theta_1 x}$$

$$\boxed{h_\theta(x) = \theta^T x} = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Hypothesis representation

Logistic Regression Model

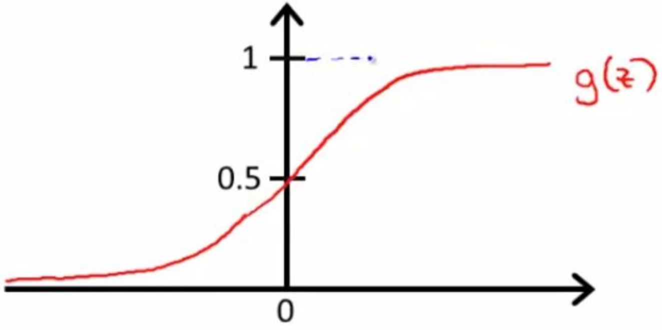
Want $0 \leq h_\theta(x) \leq 1$

$$\boxed{h_\theta(x) = g(\theta^T x)}$$

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

$\theta^T x$

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

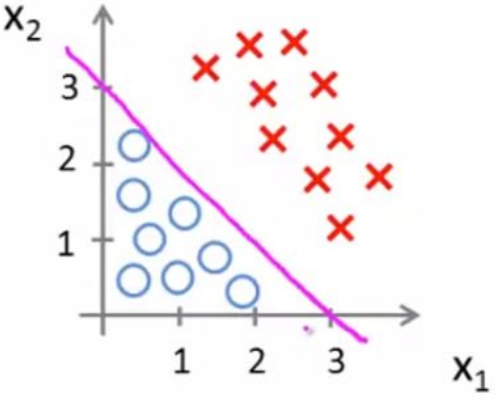


Sigmoid function

Logistic function

Andrew Ng

Decision Boundary



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

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

x_1, x_2

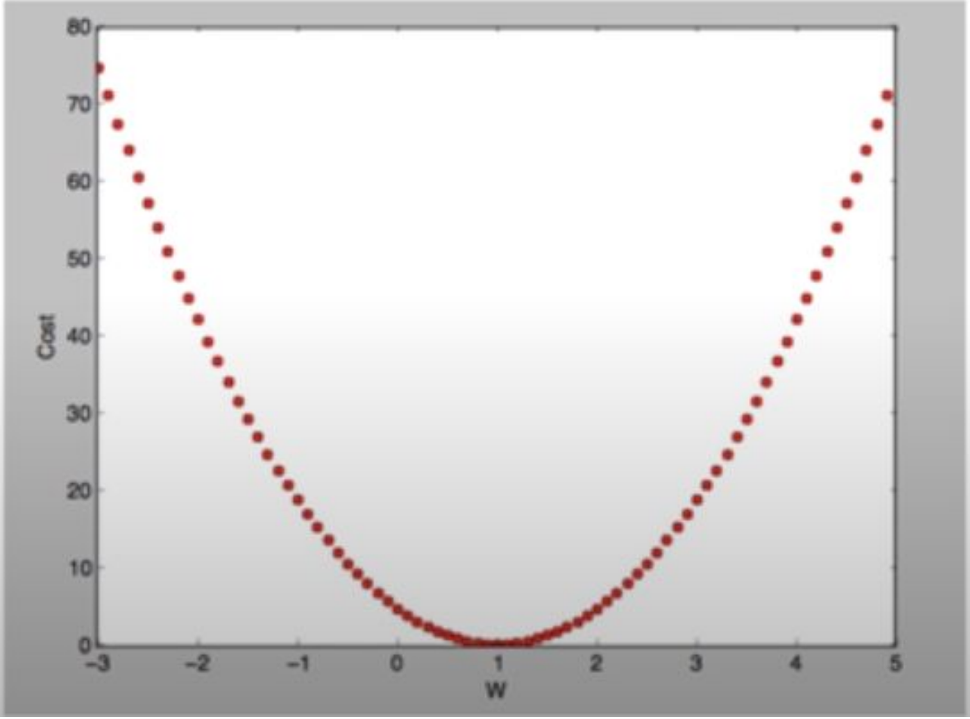
Predict "$y = 1$" if $\underbrace{-3 + x_1 + x_2 \geq 0}_{\theta^T x}$

$\rightarrow x_1 + x_2 \geq 3$

$x_1 + x_2 = 3$

Cost function

$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2 \quad \text{when} \quad H(x) = Wx + b$$

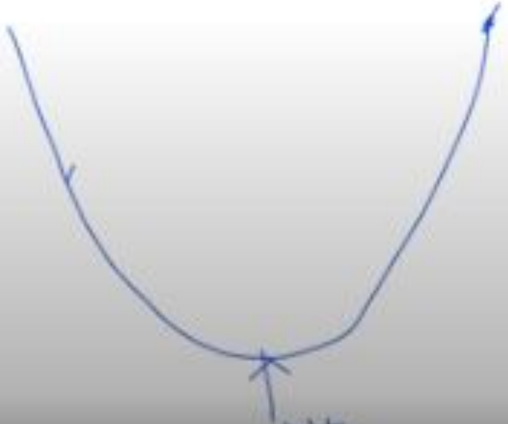


$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

0< ~ <1

$$H(x) = Wx + b$$

$$H(X) = \frac{1}{1+e^{-W^TX}}$$



최소

New cost function for logistic

$$cost(W) = \frac{1}{m} \sum c(H(x), y)$$

$$\boxed{c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}}$$

$$c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}$$

Cost $\boxed{y=1}$

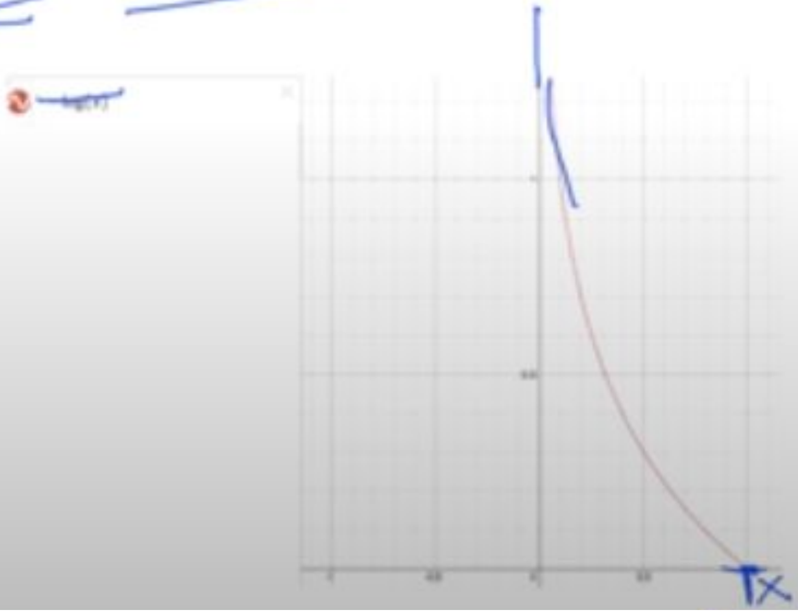
$H(x) = 1 \rightarrow cost(1) = 0$

$H(x) = 0 \rightarrow cost = \infty \uparrow$

cost

||

$g(z) = -log(z)$



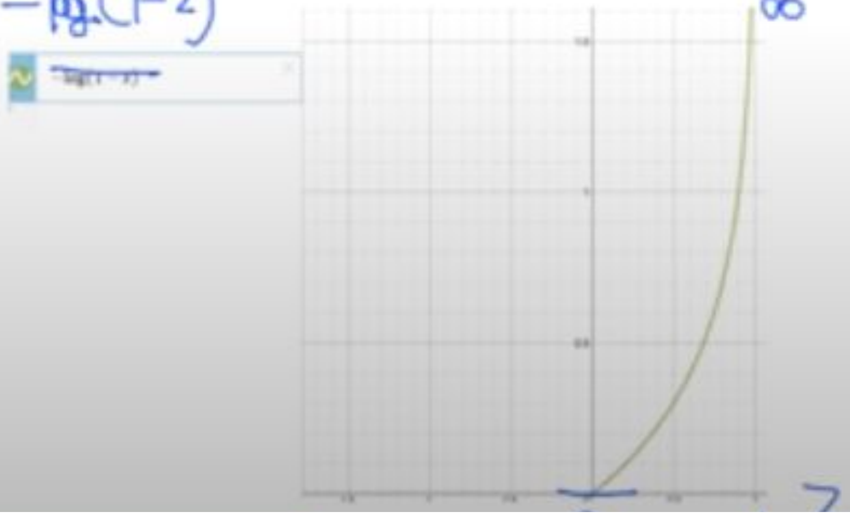
$$c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}$$

$y = 0$

$H(x) = 0$, cost $= 0$

$H(x) = 1$, cost $= \infty \uparrow$

$-\log(1-z)$



Cost function of Logistic Regression

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

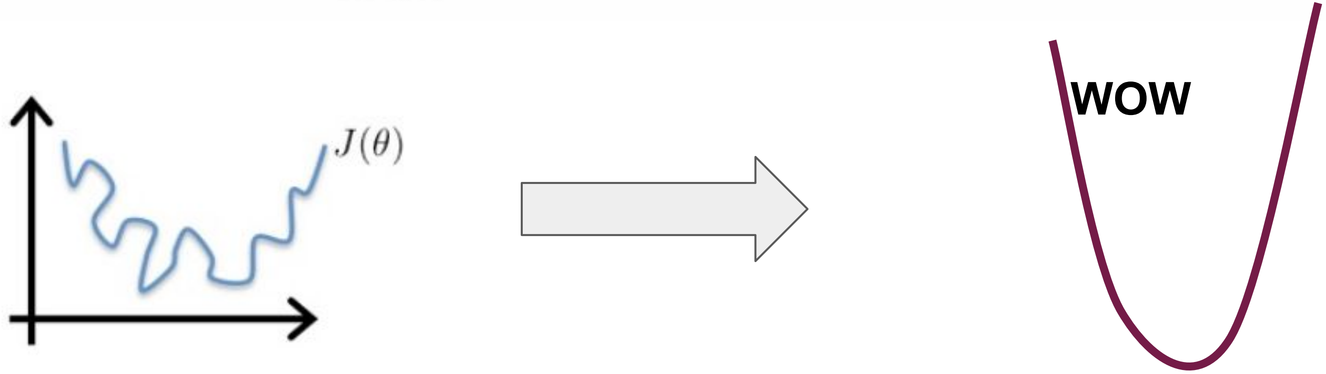
$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Note: $y = 0$ or 1 always

Simplified cost function

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

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

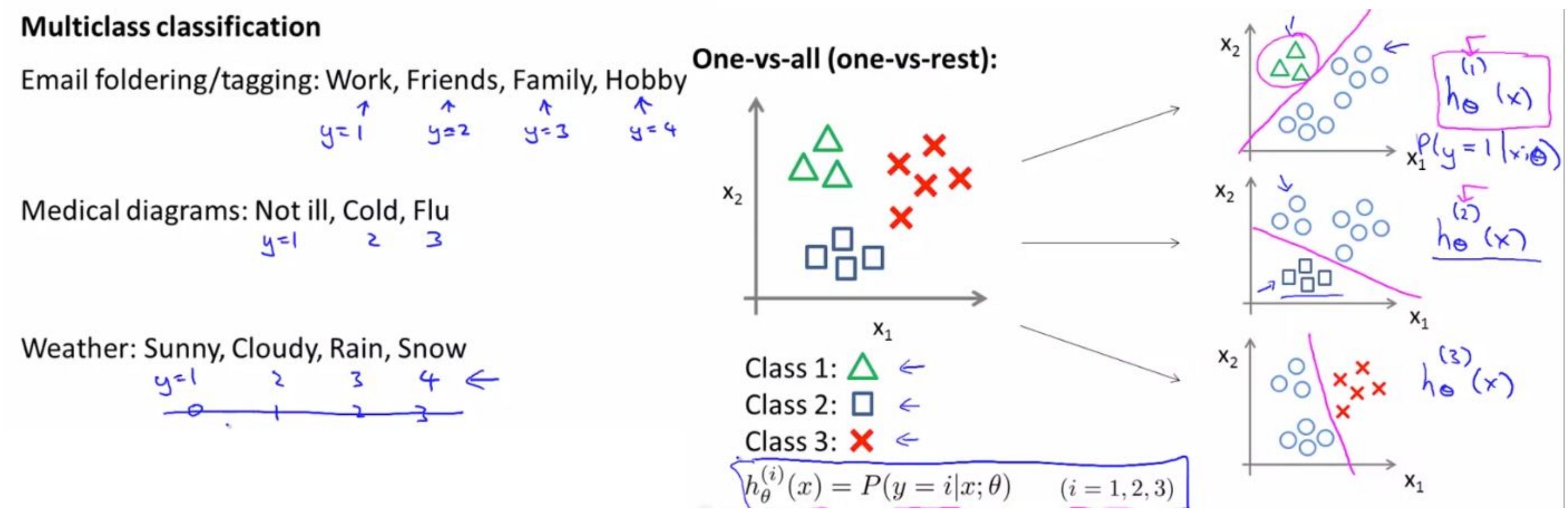


Logistic Regression : Gradient descent

$$W := W - \alpha \frac{\partial}{\partial W} cost(W)$$

Same method - Gradient descent of Linear Regression

Multiclass Classification



Multiclass Classification (cont'd)

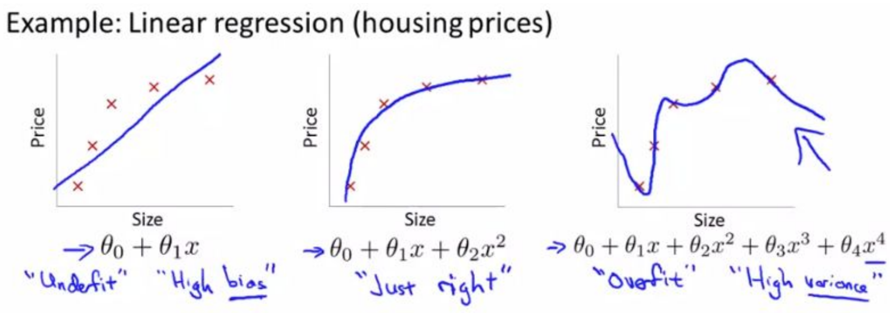
One-vs-all

Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$.

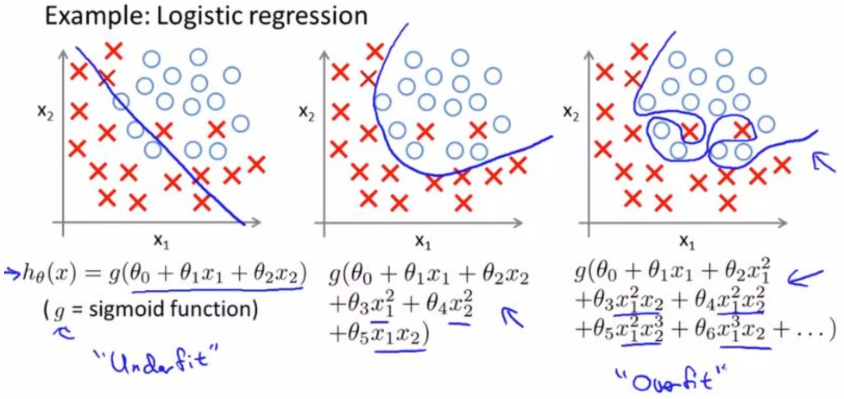
On a new input x, to make a prediction, pick the class i that maximizes

$$\max_i h_\theta^{(i)}(x)$$

Solving the problem of overfitting

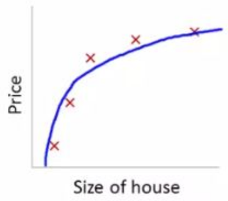


Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

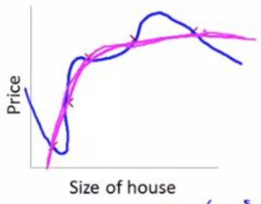


Solving the problem of overfitting (cont'd)

Intuition



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Suppose we penalize and make θ_3, θ_4 really small.

$$\rightarrow \min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0 \qquad \theta_4 \approx 0$

In regularized linear regression, we choose θ to minimize

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

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

$\theta_1, \theta_2, \theta_3, \theta_4$

$\theta_1 \approx 0, \ \theta_2 \approx 0$

$\theta_3 \approx 0, \ \theta_4 \approx 0$

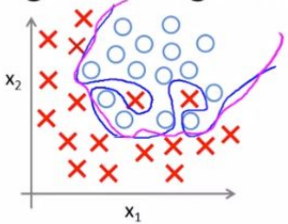
$\boxed{h_\theta(x) = \theta_0}$

$h_\theta(x)$

$\theta_0 + \cancel{\theta_1 x} + \cancel{\theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Solving the problem of overfitting (cont'd)

Regularized logistic regression.



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

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

$\boxed{\theta_1, \theta_2, \dots, \theta_n}$

Advanced optimization

fminunc(@costFunction...)

$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$ theta(1), theta(2), theta(n+1)

```
function [jVal, gradient] = costFunction(theta)
    jVal = [code to compute J(θ)];
```

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

```
    gradient(1) = [code to compute ∂/∂θ0 J(θ)];
```

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

```
    gradient(2) = [code to compute ∂/∂θ1 J(θ)];
```

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

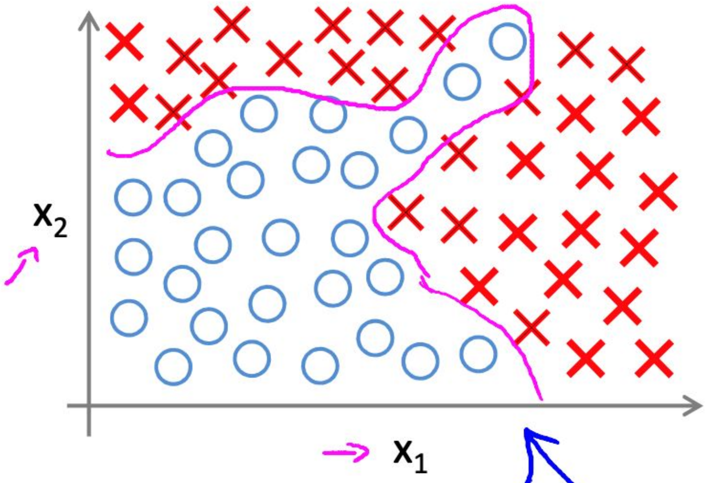
```
    gradient(3) = [code to compute ∂/∂θ2 J(θ)];
```

$$\left(\frac{1}{m}\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}\right) + \frac{\lambda}{m}\theta_2$$

```
    ⋮
    gradient(n+1) = [code to compute ∂/∂θn J(θ)];
```

Neural Networks

Non-linear Classification



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \dots)$$

$x_1 =$ size

$x_2 =$ # bedrooms

$x_3 =$ # floors

$x_4 =$ age

$\dots$

x_{100}

$n = 100$

$\rightarrow x_1^2, x_1x_2, x_1x_3, x_1x_4 \dots x_1x_{100}$

$x_2^2, x_2x_3 \dots$

≈ 5000 feature $\quad O(n^2)$ $\quad \approx \frac{n^2}{2}$

$\rightarrow x_1^2, x_2^2, x_3^2, \dots, x_{100}^2$ $\quad 10$

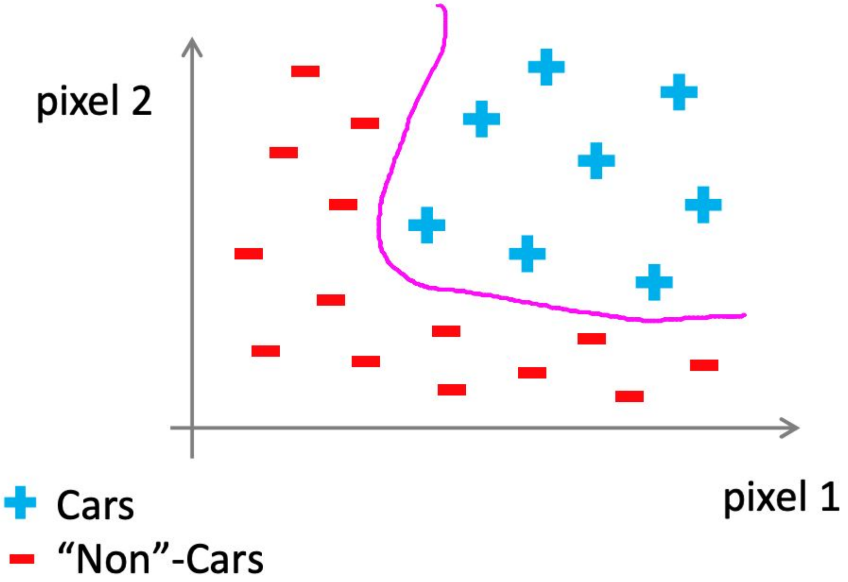
$\rightarrow x_1x_2x_3, x_1^2x_2, x_{10}x_{11}x_{17}, \dots$

$O(n^3)$

170,000

Andrew Ng

→ **Learning Algorithm**



50 x 50 pixel images→ 2500 pixels

$\boxed{n = 2500}$ (7500 if RGB)

0-255

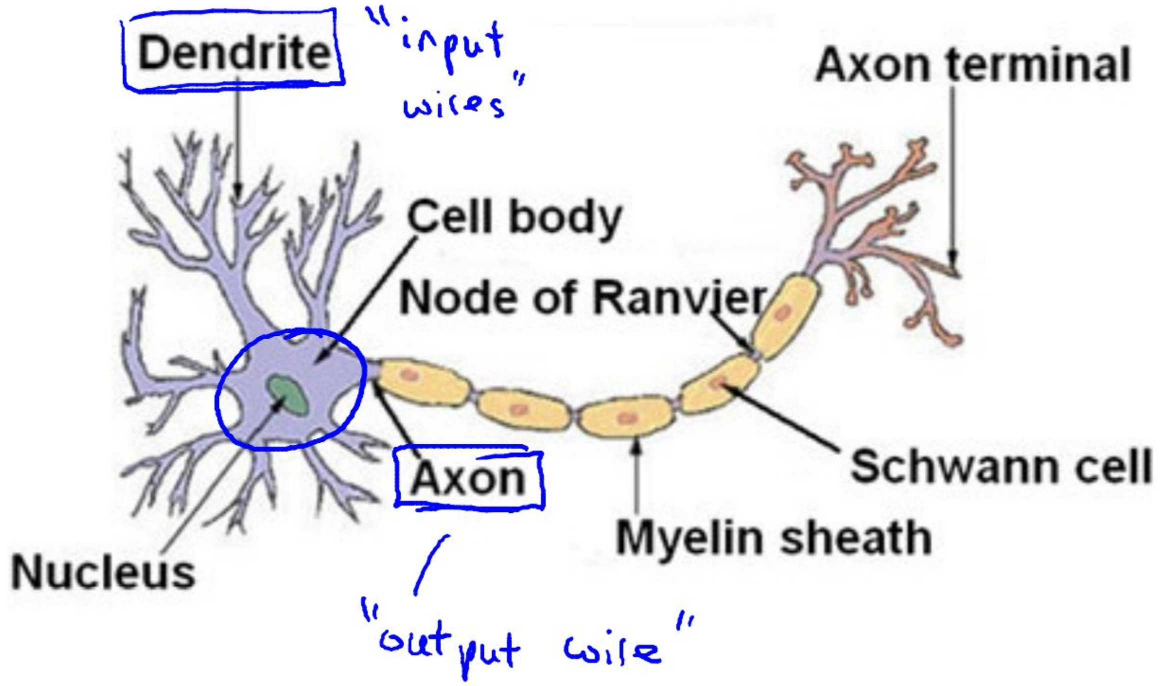
$$x = \begin{bmatrix} \text{pixel 1 intensity} \\ \text{pixel 2 intensity} \\ \vdots \\ \text{pixel 2500 intensity} \end{bmatrix}$$

Quadratic features ($x_i \times x_j$): ≈3 million features

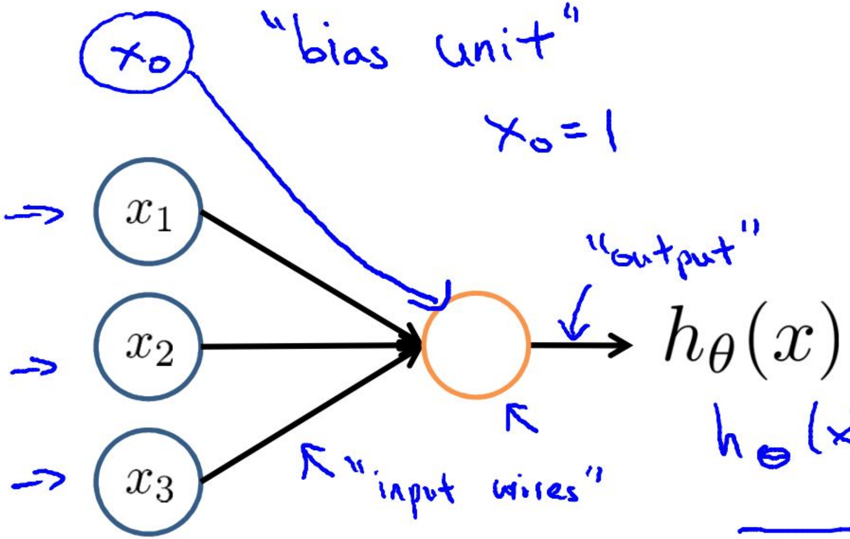
Andrew Ng

Neural Networks

Neuron in the brain



Neuron model: Logistic unit



$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

"weights"

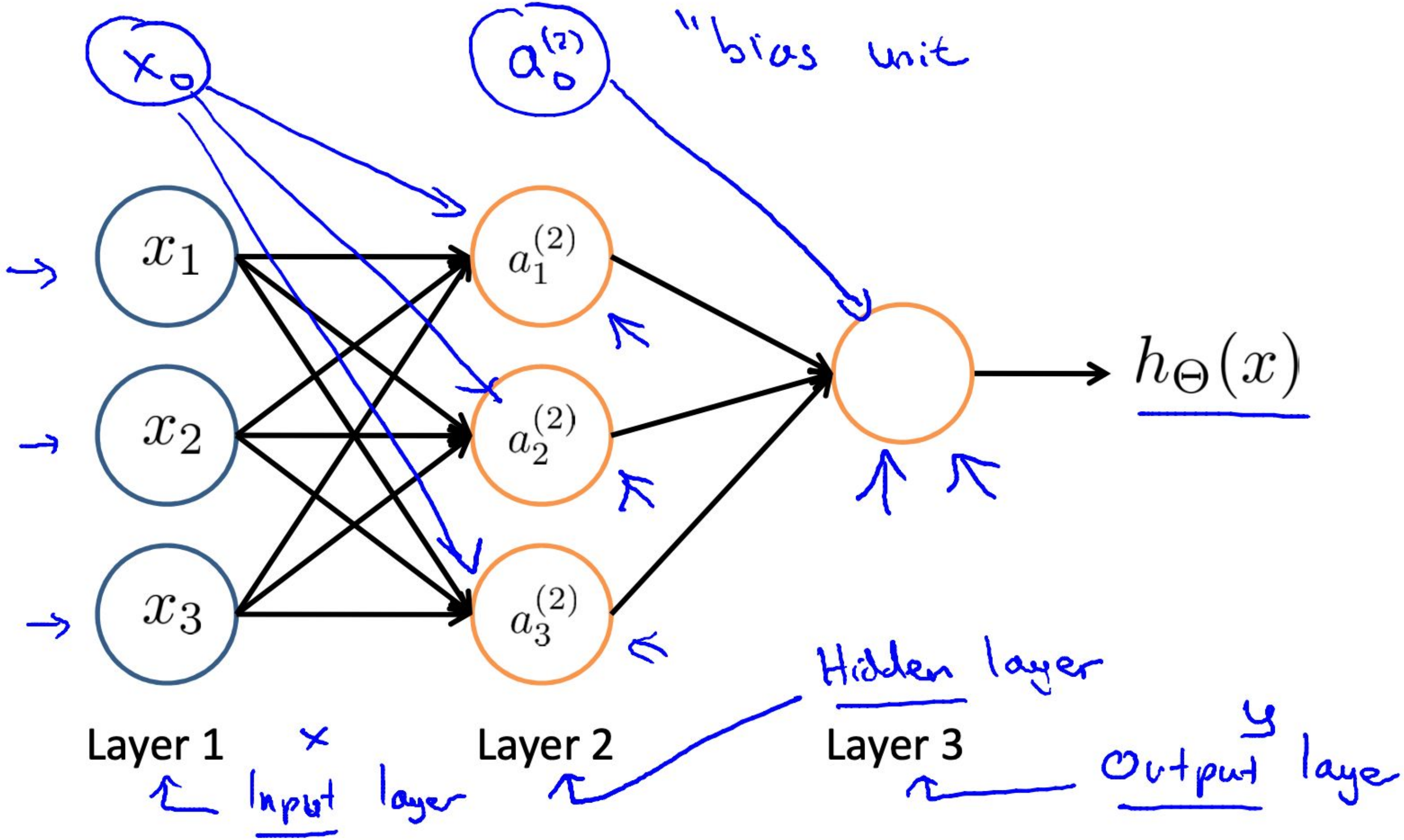
(parameters

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

Sigmoid (logistic) activation function.

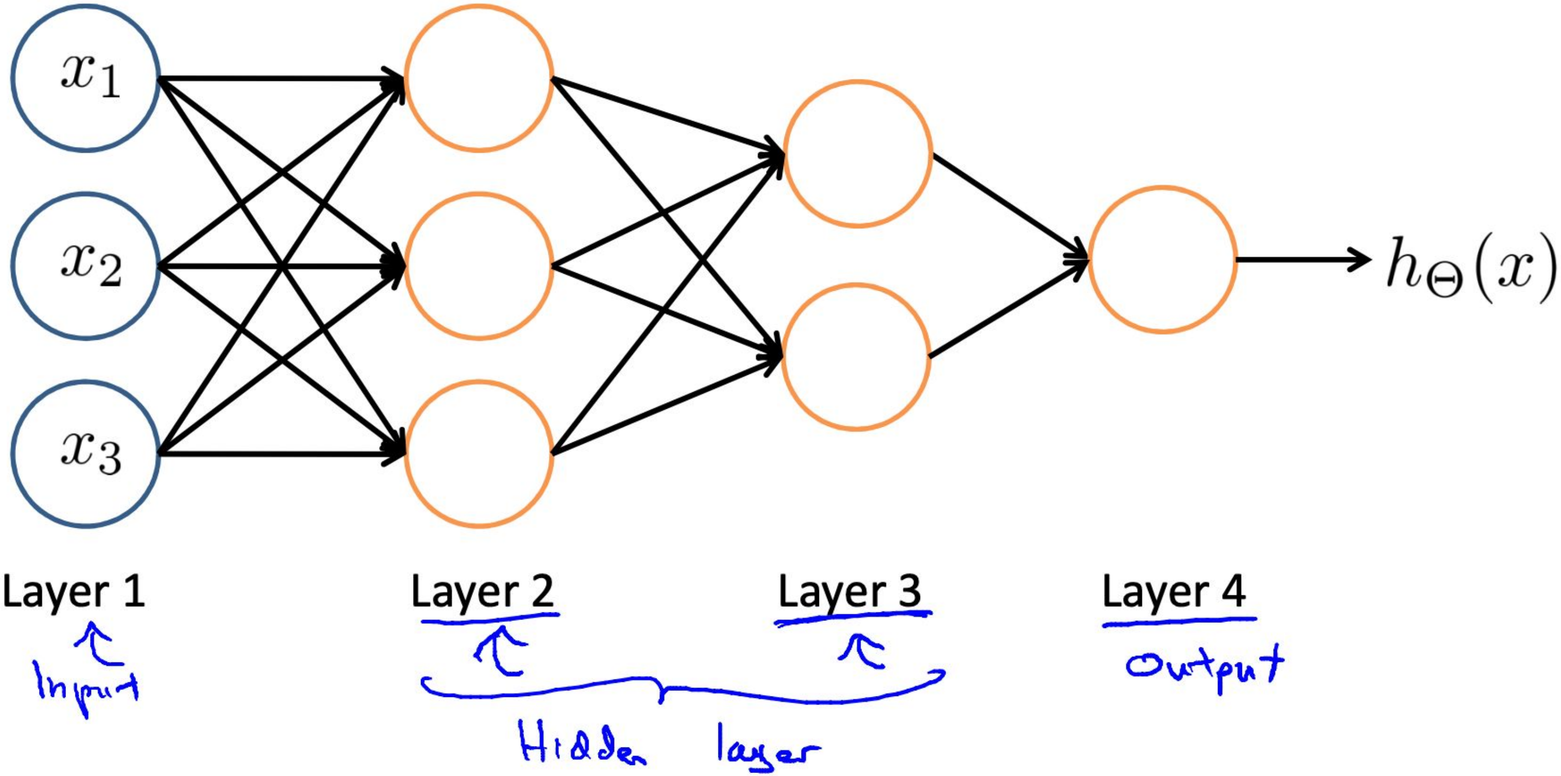
$$g(z) = \frac{1}{1+e^{-z}}$$

Neural Network



Andrew Ng

Other network architectures



Andrew Ng

Applications

Applications

How a neural network can compute a compute non-linear function of the input

Non-linear 분류의 대표적인 예:

XOR / XNOR

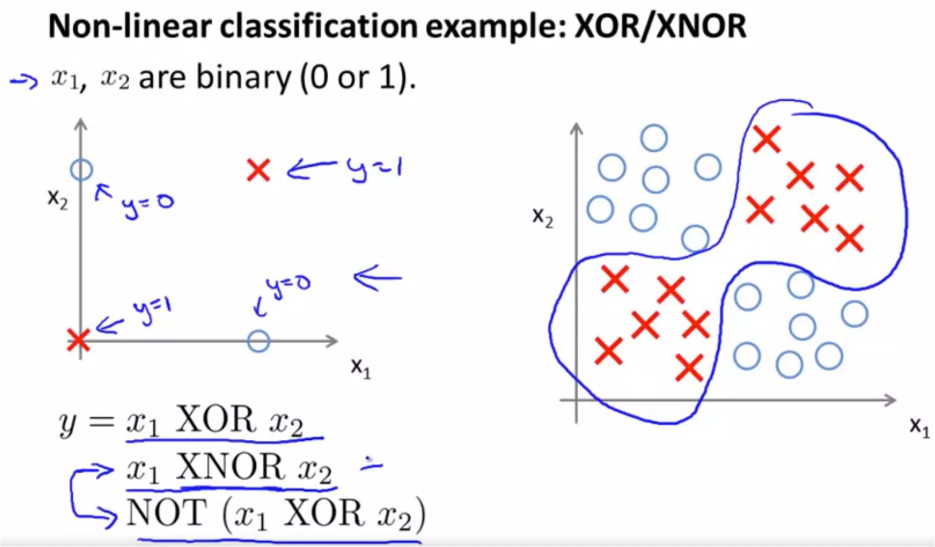
XOR	0	1
0	0	1
1	1	0

XOR 게이트는 exclusive OR 게이트의 줄임말로 한국어로는 상호 배제적인 OR 게이트이다. 상호 배제적 OR 게이트란 이름의 의미는 OR 게이트와 동일하게 작동하지만 입력값이 동일한 경우에는 1을 출력하지 않는다는 의미이다. 입력값이 서로 다르면 1을 출력하고, 같으면 0을 출력한다.

XNOR	0	1
0	1	0
1	0	1

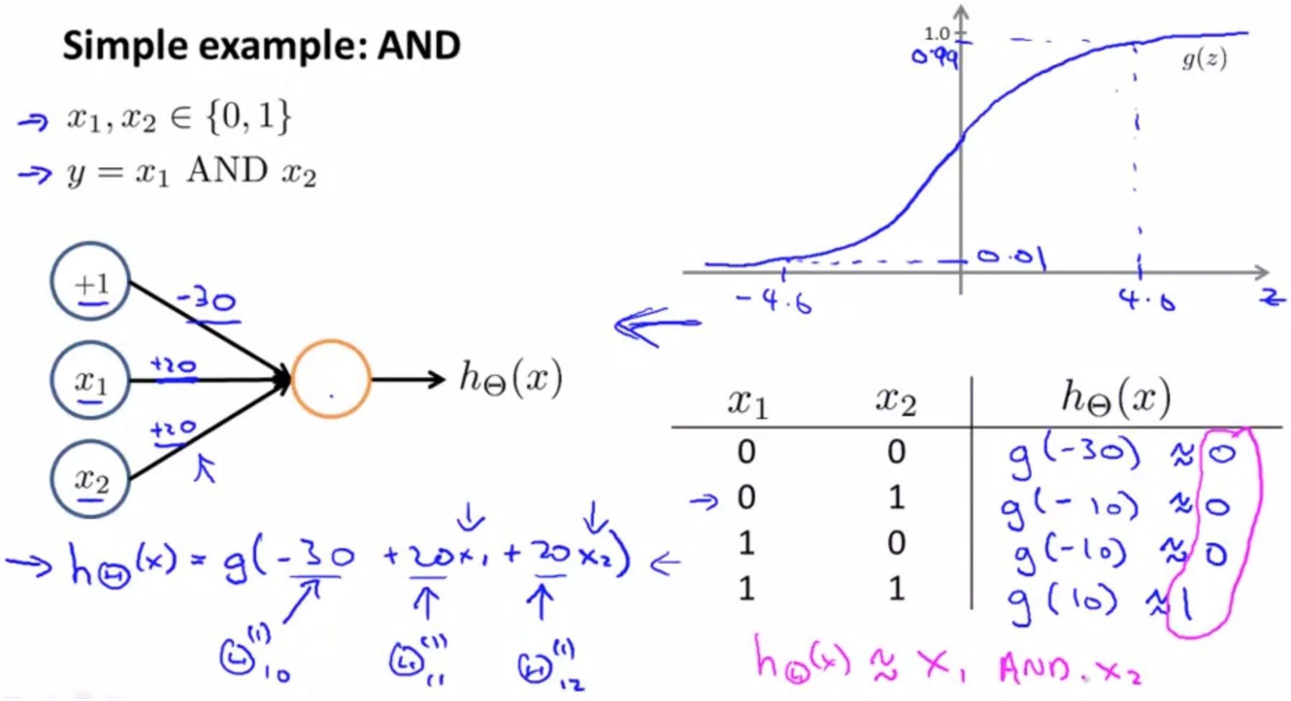
XNOR 게이트(XNOR gate 또는 EXNOR, ENOR, NXOR, XAND gate)는 XOR 게이트 뒤에 NOT 게이트를 붙여 출력값을 반대로 만들어놓은 것이다. 입력값이 서로 같으면 1을 출력하고, 다르면 0을 출력한다. 그래서 비교 게이트나 일치 확인 게이트라고도 불린다.

논리회로: 출처 나무위키

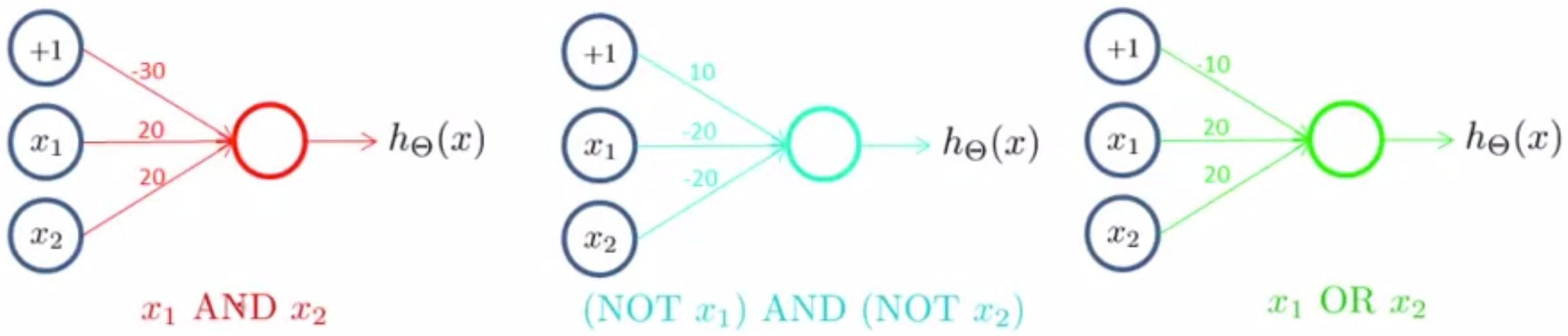


Simple example: And Function

And function 계산
(1 X -30
x1 X 20
x2 X 20)
에다가 Sigmoid
function 적용



Putting it together: x_1 XNOR x_2



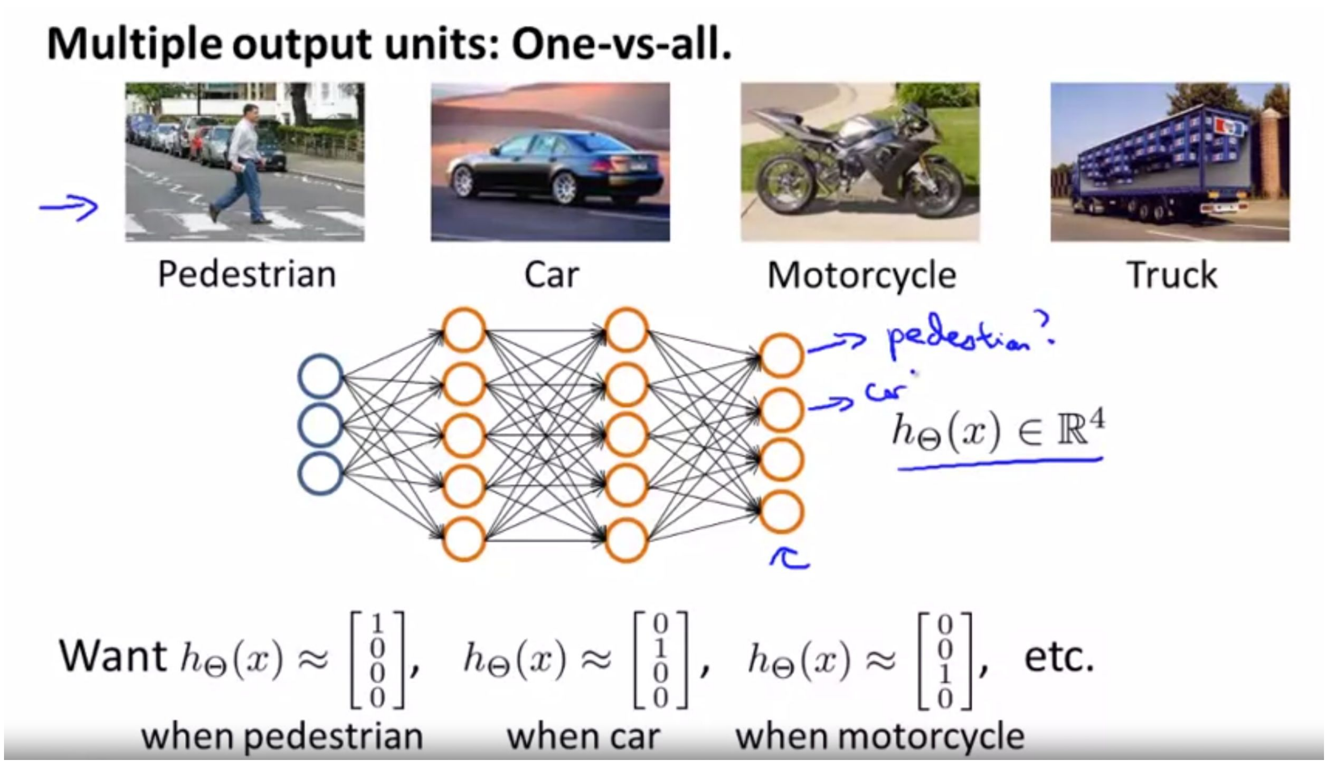
x1 XNOR x2 =

x1 AND x2
(NOT x1) AND (NOT x2)
x1 OR x2

XNOR 이나 XOR 과 같이
Linear function으로 Classification이 되지 않는 경우에도
Neural Network를 사용해 Classification이 가능함

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

Multiclass Classification



One vs all 의 확장판쯤

Pedestrian? Yes or No
[1, 0, 0, 0]
Car? Yes or No
[0, 1, 0, 0]
Motocycle? Yes or No
[0, 0, 1, 0]
Truck? Yes or No
[0, 0, 0, 1]

다른 예시: MNIST 10 digits

```
print(Y_train[:5])
```

```
[[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```