# Logistic Regression

# Step 1: Function Set

We want to find $P_{w,b}(C_1|x)$

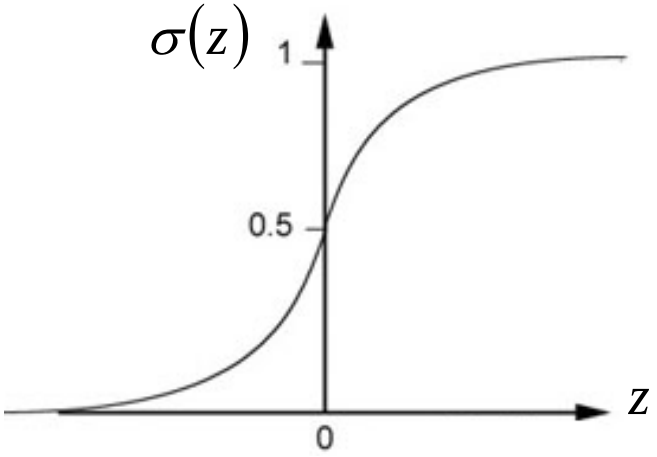
If $P_{w,b}(C_1|x) \geq 0.5$, output $C_1$

Otherwise, output $C_2$

$$P_{w,b}(C_1|x) = \sigma(z)$$

$$z = w \cdot x + b$$

$$\sigma(z) = \frac{1}{1 + exp(-z)}$$

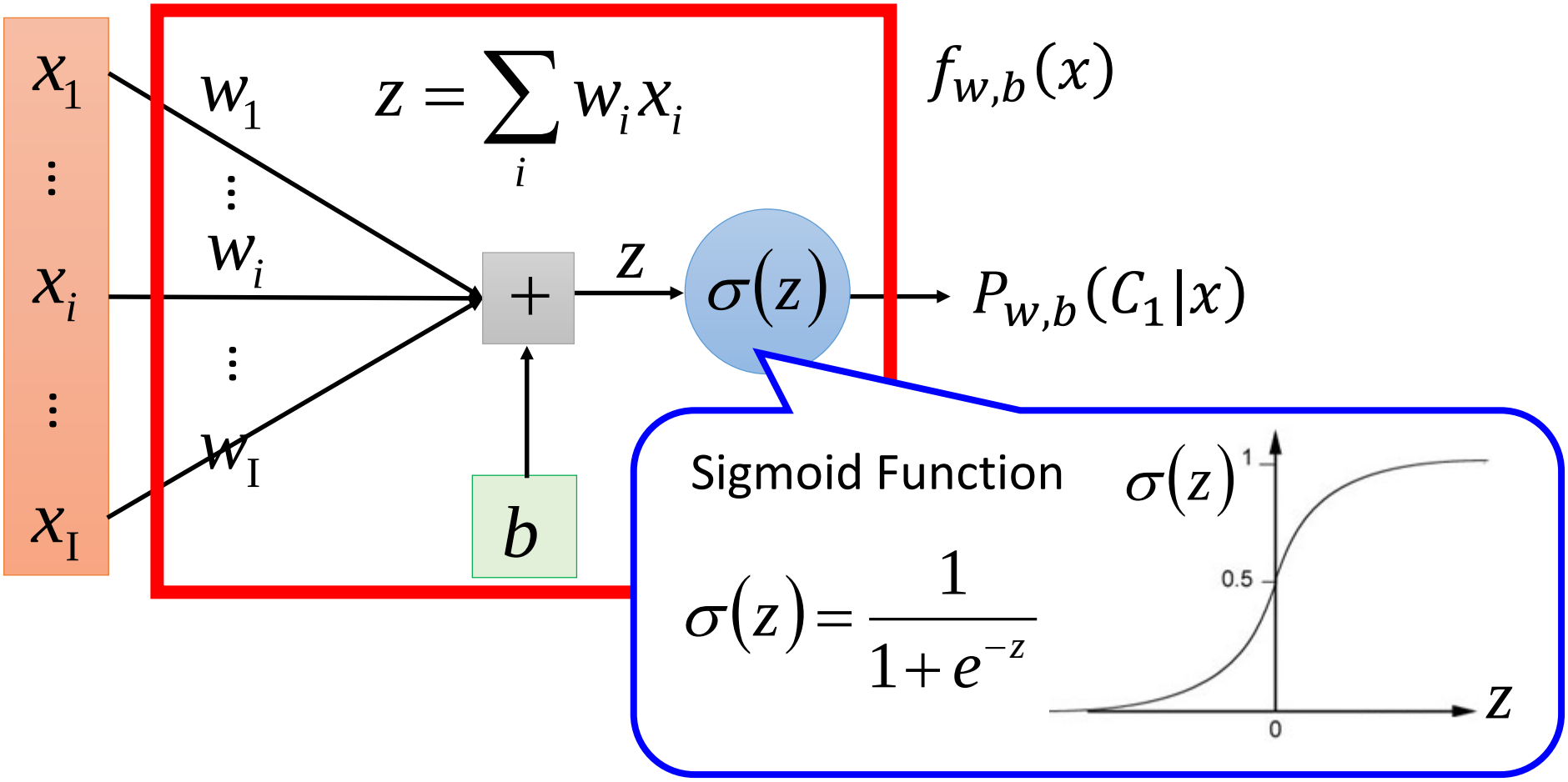



Function set:

$$f_{w,b}(x) = P_{w,b}(C_1|x)$$

Including all different w and b

# Step 1: Function Set



## ***Logistic Regression***

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Output: between 0 and 1

## ***Linear Regression***

$f_{w,b}(x) = \sum_i w_i x_i + b$

Output: any value

Step 2:

Step 3:

# Step 2: Goodness of a Function

Training Data

| $x^1$ | $x^2$ | $x^3$ | …… | $x^N$ |
|---|---|---|---|---|
| $C_1$ | $C_1$ | $C_2$ | | $C_1$ |

Assume the data is generated based on $f_{w,b}(x) = P_{w,b}(C_1|x)$

Given a set of w and b, what is its probability of generating the data?

$$L(w,b) = f_{w,b}(x^1) f_{w,b}(x^2) \left(1 - f_{w,b}(x^3)\right) \cdots f_{w,b}(x^N)$$

The most likely $w^*$ and $b^*$ is the one with the largest $L(w,b)$.

$$w^*, b^* = arg \max_{w,b} L(w,b)$$

| $x^1$ | $x^2$ | $x^3$ | …… |
|---|---|---|---|
| $C_1$ | $C_1$ | $C_2$ | |

| $x^1$ | $x^2$ | $x^3$ | …… |
|---|---|---|---|
| $\hat{y}^1 = 1$ | $\hat{y}^2 = 0$ | $\hat{y}^3 = 1$ | |

$\hat{y}^n$: 1 for class 1, 0 for class 2

$$L(w,b) = f_{w,b}(x^1) f_{w,b}(x^2) \left(1 - f_{w,b}(x^3)\right) \cdots$$

$$w^*, b^* = arg \max_{w,b} L(w,b) = w^*, b^* = arg \min_{w,b} -lnL(w,b)$$

$$-lnL(w,b)$$

$$= -lnf_{w,b}(x^1) \rightarrow -[1 \; lnf(x^1) + 0 \; ln(1 - f(x^1))]$$

$$-lnf_{w,b}(x^2) \rightarrow -[1 \; lnf(x^2) + 0 \; ln(1 - f(x^2))]$$

$$-ln\left(1 - f_{w,b}(x^3)\right) \rightarrow -[0 \; lnf(x^3) + 1 \; ln(1 - f(x^3))]$$

$$\vdots$$

# Step 2: Goodness of a Function

$$L(w,b) = f_{w,b}(x^1) f_{w,b}(x^2) \left(1 - f_{w,b}(x^3)\right) \cdots f_{w,b}(x^N)$$

$$-lnL(w,b) = lnf_{w,b}(x^1) + lnf_{w,b}(x^2) + ln\left(1 - f_{w,b}(x^3)\right) \cdots$$

$\hat{y}^n$: 1 for class 1, 0 for class 2

$$= \sum_n -\left[\hat{y}^n lnf_{w,b}(x^n) + (1-\hat{y}^n) ln\left(1 - f_{w,b}(x^n)\right)\right]$$

Cross entropy between two Bernoulli distribution

Distribution p:

$p(x=1) = \hat{y}^n$

$p(x=0) = 1 - \hat{y}^n$

cross entropy

Distribution q:

$q(x=1) = f(x^n)$

$q(x=0) = 1 - f(x^n)$

$$H(p,q) = -\sum_x p(x) ln\big(q(x)\big)$$

| | ***Logistic Regression*** | ***Linear Regression*** |
|---|---|---|
| Step 1: | $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$ | $f_{w,b}(x) = \sum_i w_i x_i + b$ |
| | Output: between 0 and 1 | Output: any value |
| | Training data: $(x^n, \hat{y}^n)$ | Training data: $(x^n, \hat{y}^n)$ |
| Step 2: | $\hat{y}^n$: 1 for class 1, 0 for class 2 | $\hat{y}^n$: a real number |
| | $L(f) = \sum_n C(f(x^n), \hat{y}^n)$ | $L(f) = \frac{1}{2}\sum_n (f(x^n) - \hat{y}^n)^2$ |

Cross entropy:

$$C(f(x^n), \hat{y}^n) = -\left[\hat{y}^n ln f(x^n) + (1 - \hat{y}^n) ln\left(1 - f(x^n)\right)\right]$$

Question: Why don't we simply use square error as linear regression?

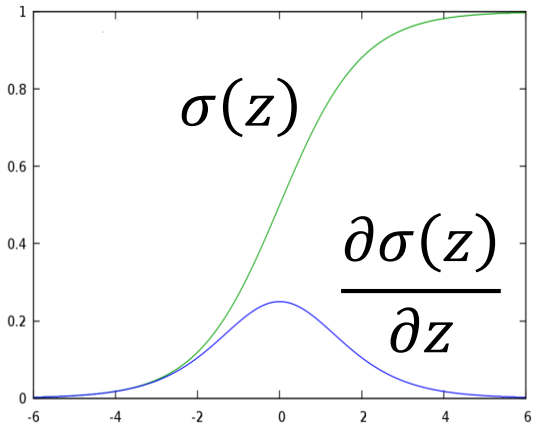
# Step 3: Find the best function

$$-\frac{lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{lnf_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n)\frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$$\left(1-f_{w,b}(x^n)\right)x_i^n$$

$$\frac{\partial lnf_{w,b}(x)}{\partial w_i} = \frac{\partial lnf_{w,b}(x)}{\partial z}\frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial ln\sigma(z)}{\partial z} = \frac{1}{\sigma(z)}\frac{\partial \sigma(z)}{\partial z} = \frac{1}{\sigma(z)}\sigma(z)\left(1-\sigma(z)\right)$$




$$f_{w,b}(x) = \sigma(z)$$
$$= 1/1 + exp(-z)$$

$$z = w \cdot x + b = \sum_i w_i x_i + b$$

# Step 3: Find the best function

$$-\frac{lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{ln f_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n) \frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$$\frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial w_i} = \frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial z}\frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial ln\left(1-\sigma(z)\right)}{\partial z} = -\frac{1}{1-\sigma(z)}\frac{\partial \sigma(z)}{\partial z} = -\frac{1}{1-\sigma(z)}\sigma(z)\left(1-\sigma(z)\right)$$

$$f_{w,b}(x) = \sigma(z)$$
$$= 1/1+exp(-z)$$

$$z = w \cdot x + b = \sum_i w_i x_i + b$$

# Step 3: Find the best function

$$-\frac{lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \frac{ln f_{w,b}(x^n)}{\partial w_i} + (1-\hat{y}^n) \frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$\left(1-f_{w,b}(x^n)\right)x_i^n$ $\qquad$ $f_{w,b}(x^n)x_i^n$

$$= \sum_n -\left[\hat{y}^n \left(1-f_{w,b}(x^n)\right)x_i^n - (1-\hat{y}^n) f_{w,b}(x^n)x_i^n\right]$$

$$= \sum_n -\left[\hat{y}^n - \hat{y}^n f_{w,b}(x^n) - f_{w,b}(x^n) + \hat{y}^n f_{w,b}(x^n)\right] x_i^n$$

$$= \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$$

Larger difference, larger update

$$w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$$

| | ***Logistic Regression*** | ***Linear Regression*** |
|---|---|---|
| Step 1: | $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$<br>Output: between 0 and 1 | $f_{w,b}(x) = \sum_i w_i x_i + b$<br>Output: any value |
| Step 2: | Training data: $(x^n, \hat{y}^n)$<br>$\hat{y}^n$: 1 for class 1, 0 for class 2<br>$L(f) = \sum_n C(f(x^n), \hat{y}^n)$ | Training data: $(x^n, \hat{y}^n)$<br>$\hat{y}^n$: a real number<br>$L(f) = \frac{1}{2}\sum_n (f(x^n) - \hat{y}^n)^2$ |

Step 3:

Logistic regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$

Linear regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$

# ***Logistic Regression + Square Error***

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: $(x^n, \hat{y}^n)$, $\hat{y}^n$: 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - \hat{y}^n\right)^2$$

Step 3:

$$\frac{\partial\, (f_{w,b}(x) - \hat{y})^2}{\partial w_i} = 2\left(f_{w,b}(x) - \hat{y}\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

$$= 2\left(f_{w,b}(x) - \hat{y}\right) f_{w,b}(x)\left(1 - f_{w,b}(x)\right) x_i$$

$\hat{y}^n = 1$ If $f_{w,b}(x^n) = 1$ (close to target) ➡ $\partial L/\partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (far from target) ➡ $\partial L/\partial w_i = 0$

# ***Logistic Regression + Square Error***

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: $(x^n, \hat{y}^n)$, $\hat{y}^n$: 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - \hat{y}^n\right)^2$$

Step 3:

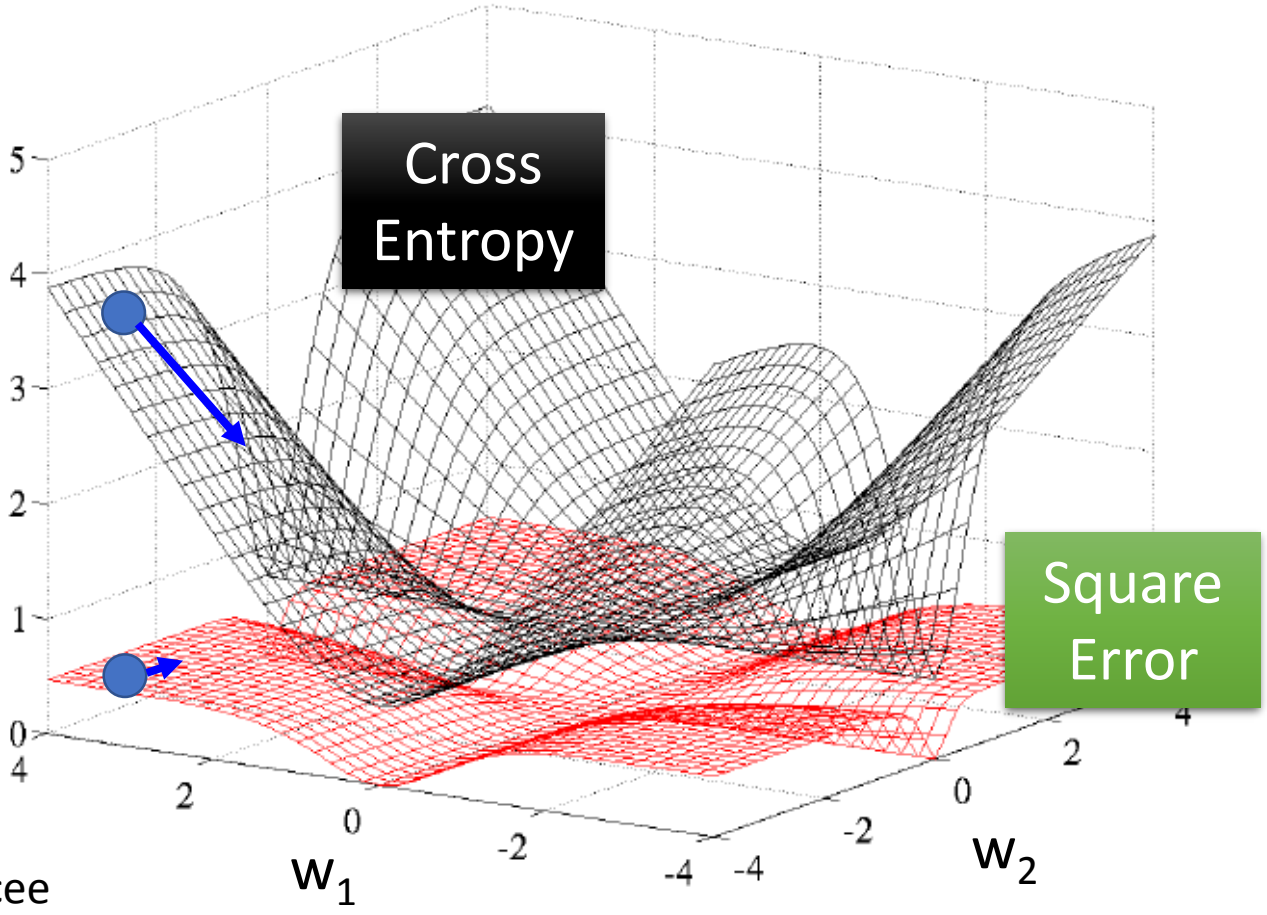
$$\frac{\partial\, (f_{w,b}(x) - \hat{y})^2}{\partial w_i} = 2\left(f_{w,b}(x) - \hat{y}\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

$$= 2\left(f_{w,b}(x) - \hat{y}\right) f_{w,b}(x)\left(1 - f_{w,b}(x)\right) x_i$$

$\hat{y}^n = 0$ If $f_{w,b}(x^n) = 1$ (far from target) ➡ $\partial L/\partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (close to target) ➡ $\partial L/\partial w_i = 0$

# Cross Entropy v.s. Square Error



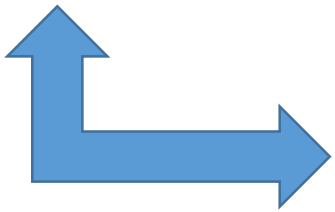

http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf

# Discriminative v.s. Generative

$$P(C_1|x) = \sigma(w \cdot x + b)$$

directly find ***w*** and b

Find $\mu^1, \mu^2, \Sigma^{-1}$



Will we obtain the same set of w and b?

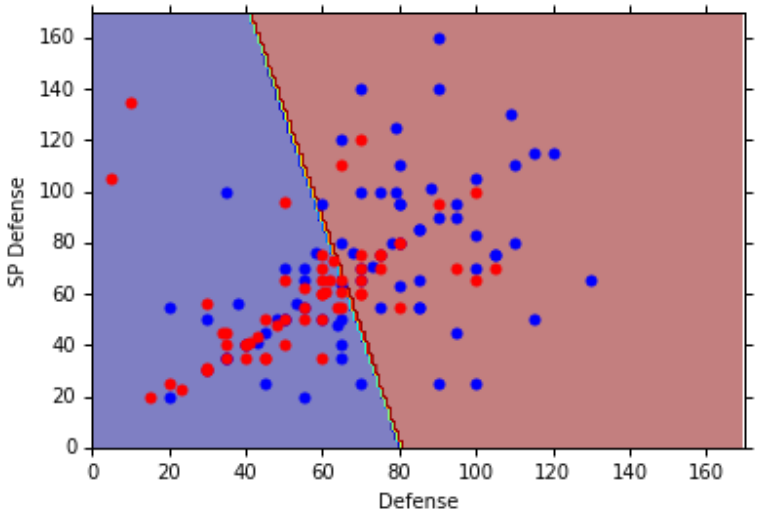
$$w^T = (\mu^1 - \mu^2)^T \Sigma^{-1}$$

$$b = -\frac{1}{2}(\mu^1)^T (\Sigma^1)^{-1} \mu^1 + \frac{1}{2}(\mu^2)^T (\Sigma^2)^{-1} \mu^2 + ln\frac{N_1}{N_2}$$

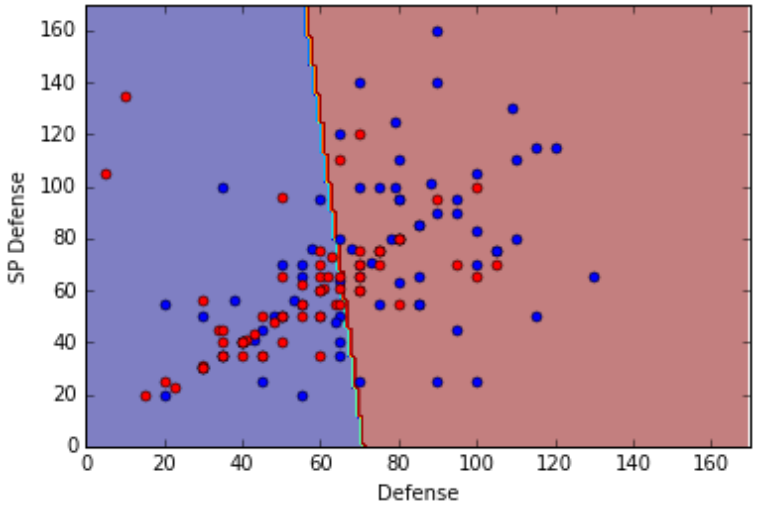
The same model (function set), but different function is selected by the same training data.

# Generative v.s. Discriminative

***Generative***




***Discriminative***




All: total, hp, att, sp att, de, sp de, speed

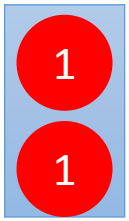
73% accuracy

79% accuracy

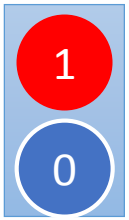
# Generative v.s. Discriminative

- Example

Training Data

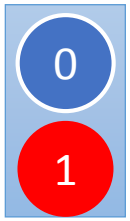



Class 1



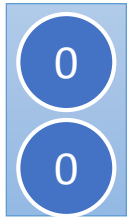

X 4

Class 2




X 4

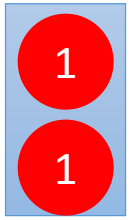
Class 2




X 4

Class 2

Testing Data




Class 1?
Class 2?

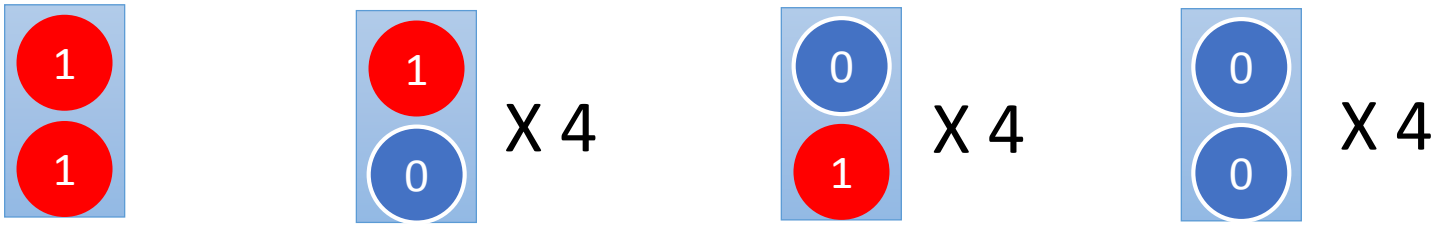
How about Naïve Bayes?

$$P(x|C_i) = P(x_1|C_i)P(x_2|C_i)$$

# Generative v.s. Discriminative

- Example

Training Data

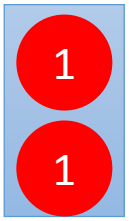



Class 1 Class 2 Class 2 Class 2

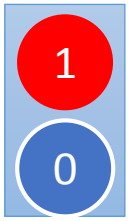
$$P(C_1) = \frac{1}{13} \qquad P(x_1 = 1|C_1) = 1 \qquad P(x_2 = 1|C_1) = 1$$

$$P(C_2) = \frac{12}{13} \qquad P(x_1 = 1|C_2) = \frac{1}{3} \qquad P(x_2 = 1|C_2) = \frac{1}{3}$$

Training Data

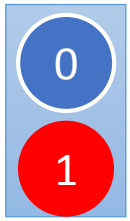



Class 1



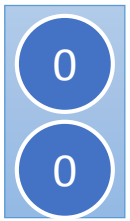

X 4

Class 2




X 4

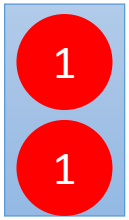
Class 2




X 4

Class 2

Testing Data




<0.5

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

$P(x|C_1)$: $1 \times 1$

$P(C_1)$: $\frac{1}{13}$

$P(x|C_2)$: $\frac{1}{3} \times \frac{1}{3}$

$P(C_2)$: $\frac{12}{13}$

$$P(C_1) = \frac{1}{13} \qquad P(x_1 = 1|C_1) = 1 \qquad P(x_2 = 1|C_1) = 1$$

$$P(C_2) = \frac{12}{13} \qquad P(x_1 = 1|C_2) = \frac{1}{3} \qquad P(x_2 = 1|C_2) = \frac{1}{3}$$

# Generative v.s. Discriminative

- Benefit of generative model
  - With the assumption of probability distribution, less training data is needed
  - With the assumption of probability distribution, more robust to the noise
  - Priors and class-dependent probabilities can be estimated from different sources.

[Bishop, P209-210]

# Multi-class Classification (3 classes as example)

$C_1$: $w^1, b_1$ $\quad z_1 = w^1 \cdot x + b_1$

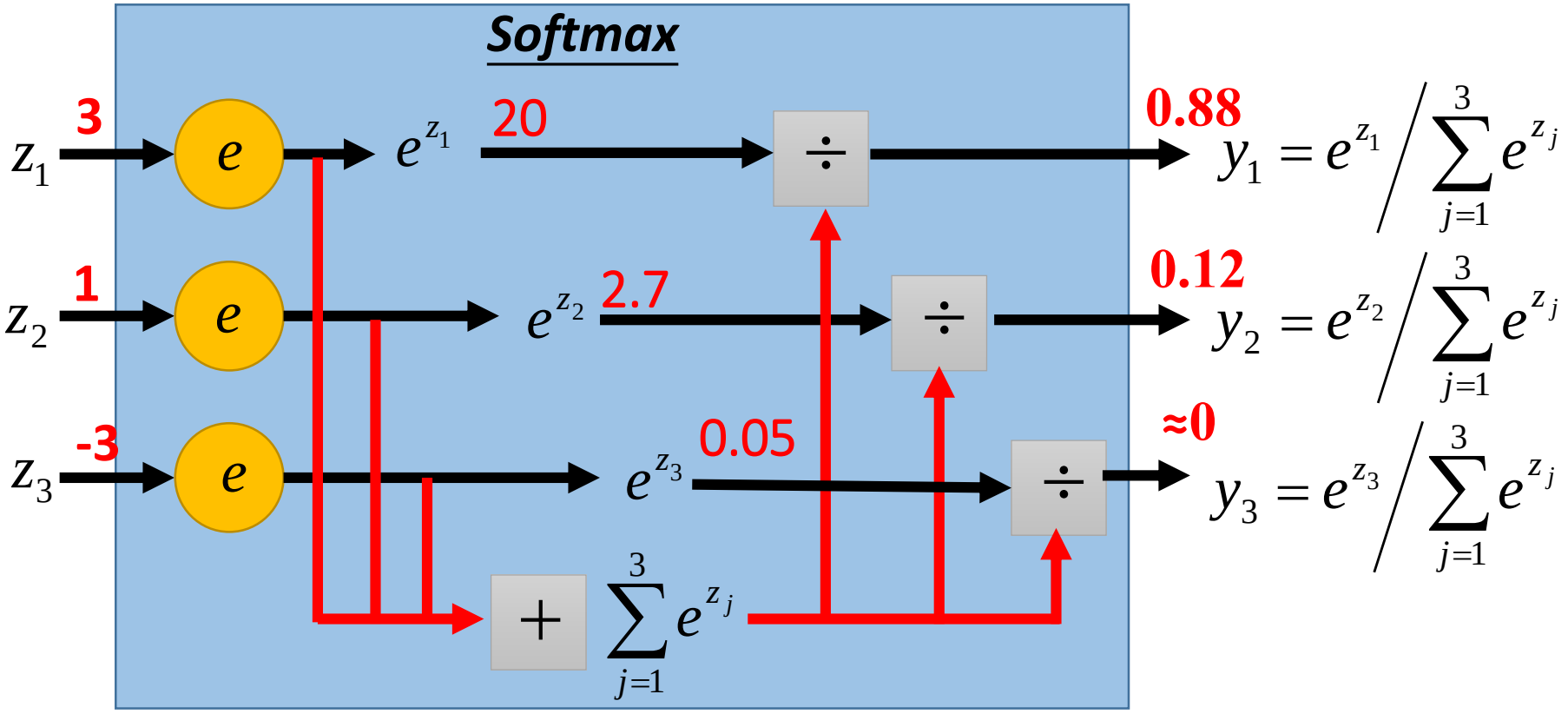
$C_2$: $w^2, b_2$ $\quad z_2 = w^2 \cdot x + b_2$

$C_3$: $w^3, b_3$ $\quad z_3 = w^3 \cdot x + b_3$

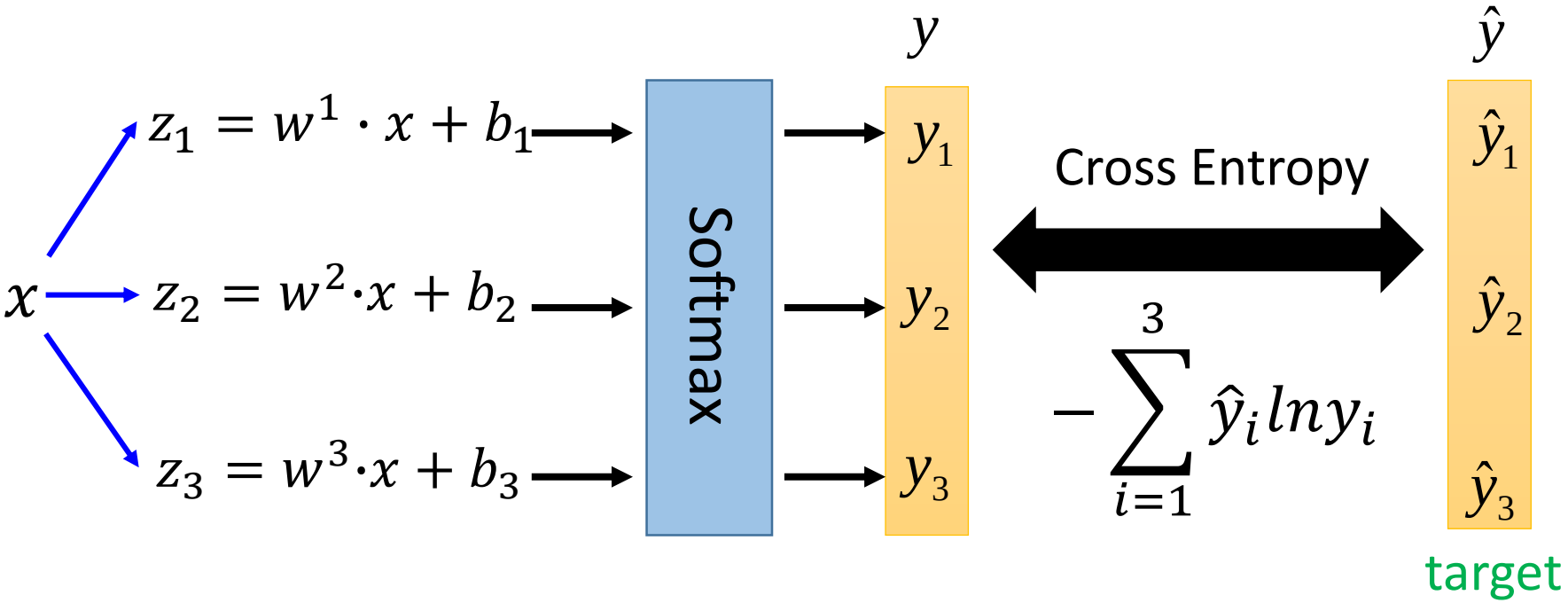
**Probability**:

- $1 > y_i > 0$
- $\sum_i y_i = 1$

$$y_i = P(C_i \mid x)$$



# ***Multi-class Classification*** (3 classes as example)




If x ∈ class 1

$$\hat{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

If x ∈ class 2

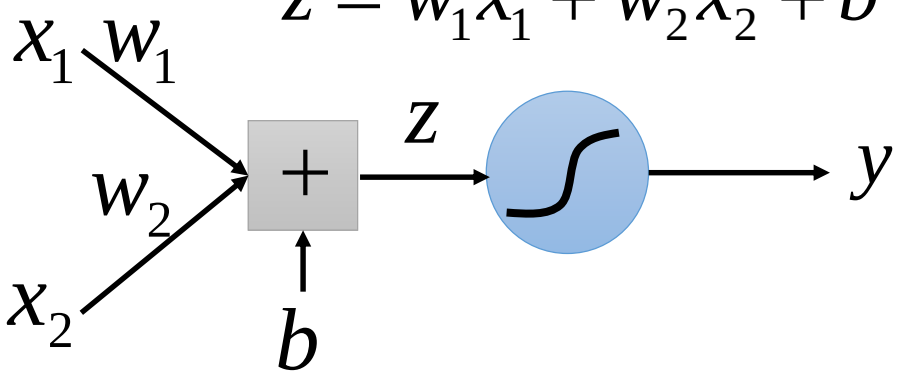
$$\hat{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

If x ∈ class 3

$$\hat{y} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

# Limitation of Logistic Regression

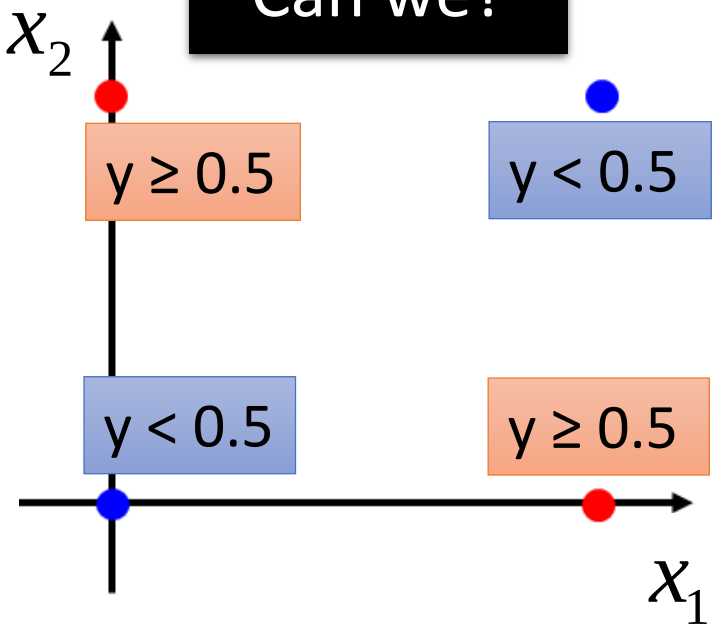
$$z = w_1 x_1 + w_2 x_2 + b$$




$$\begin{cases} Class1 & y \geq 0.5 \\ Class2 & y < 0.5 \end{cases}$$

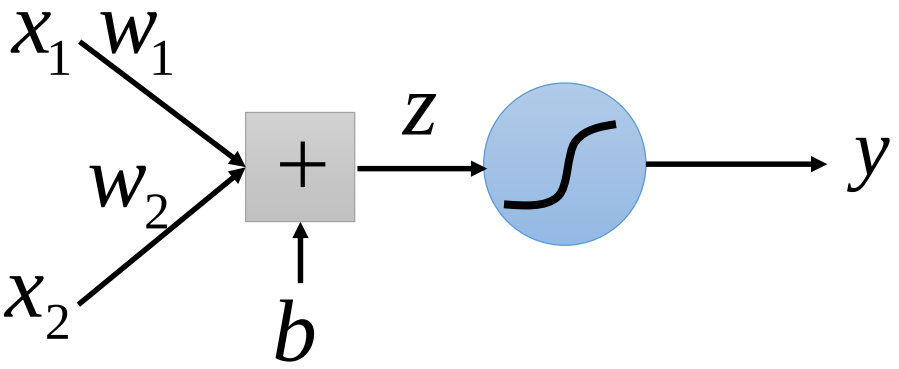
| Input Feature | | Label |
|---|---|---|
| $x_1$ | $x_2$ | |
| 0 | 0 | Class 2 |
| 0 | 1 | Class 1 |
| 1 | 0 | Class 1 |
| 1 | 1 | Class 2 |

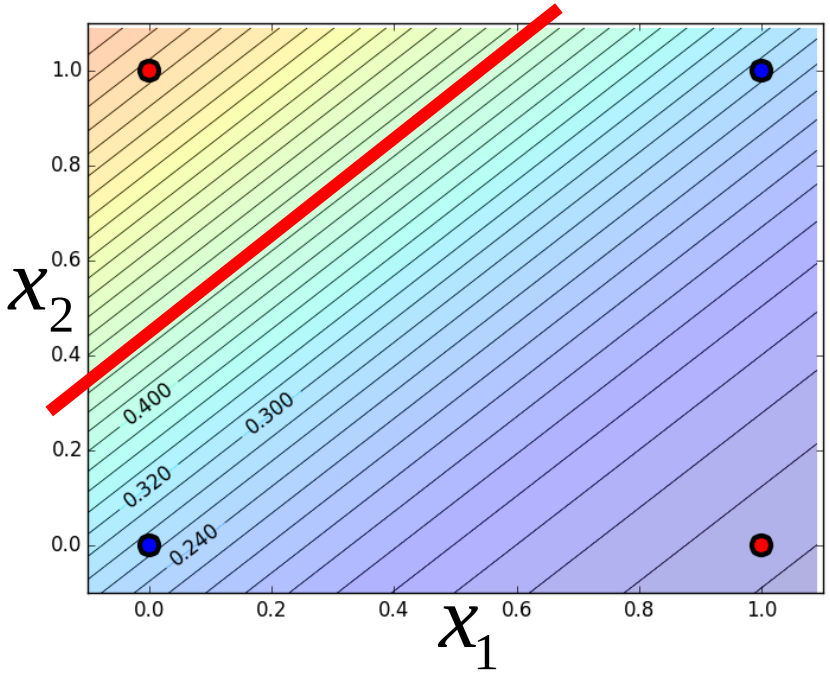
Can we?

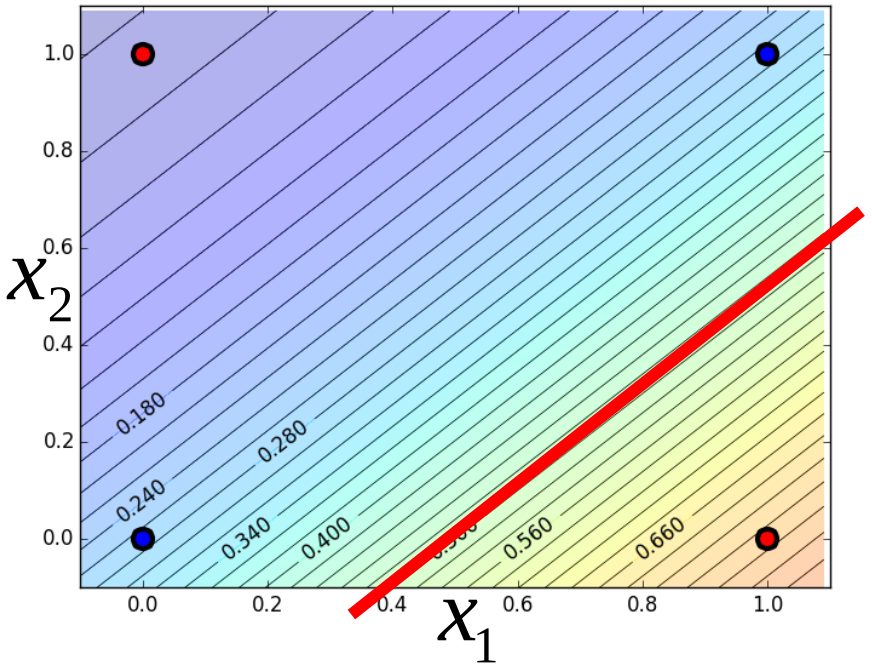


# Limitation of Logistic Regression

- No, we can't ……



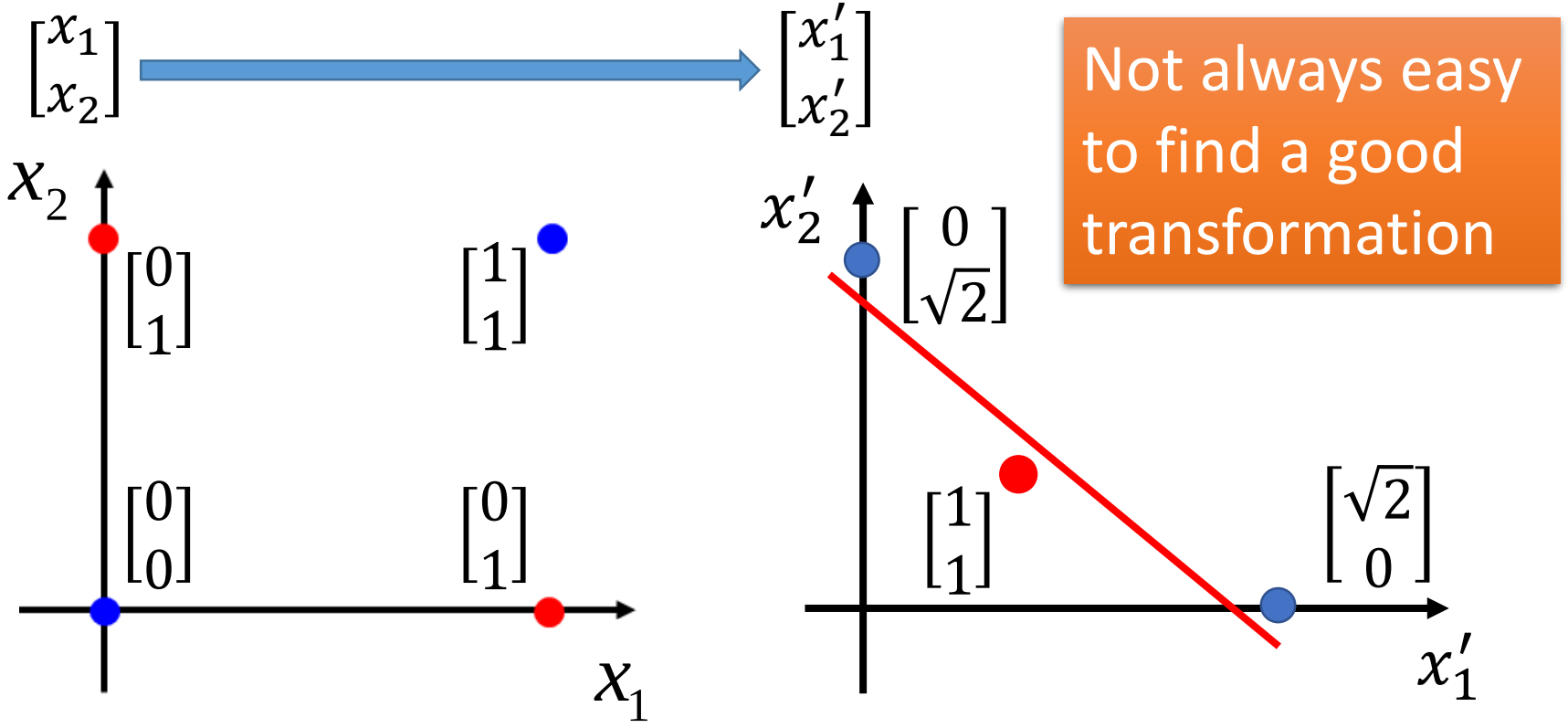



# Limitation of Logistic Regression

- Feature Transformation

$x_1'$: distance to $\begin{bmatrix}0\\0\end{bmatrix}$

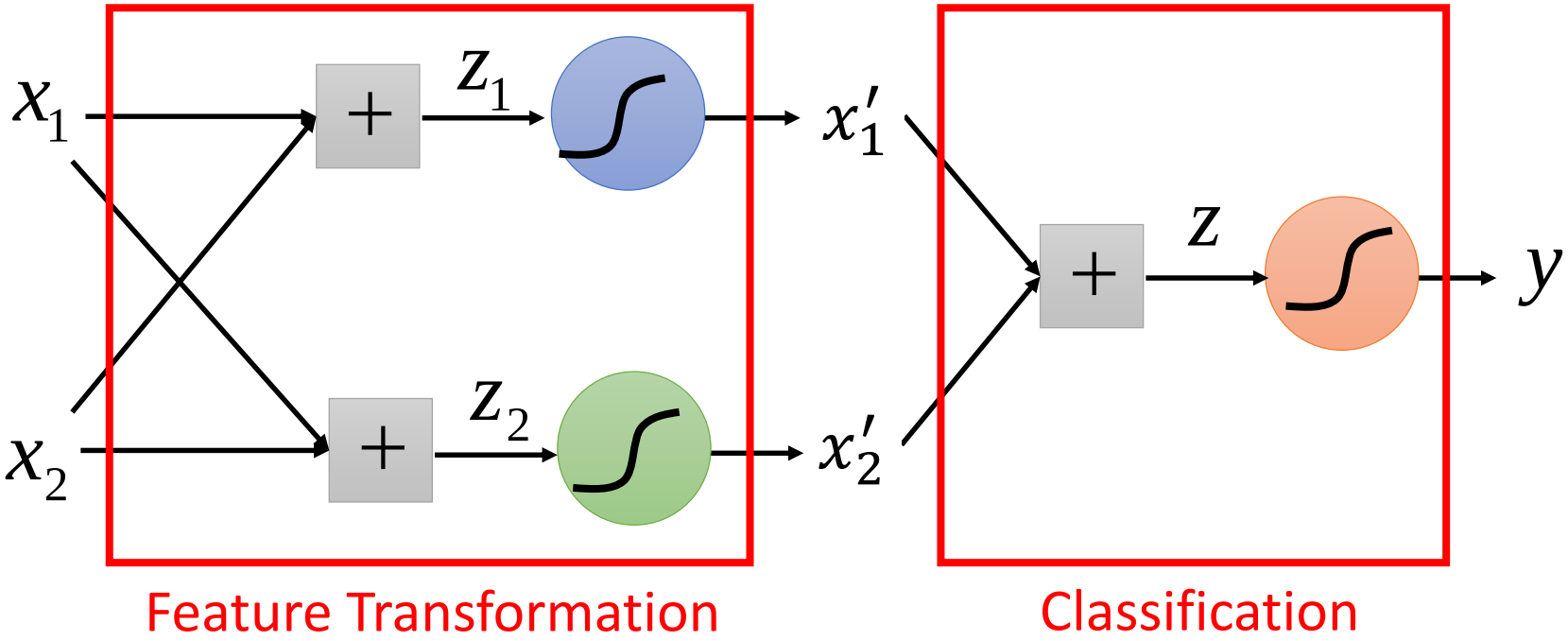
$x_2'$: distance to $\begin{bmatrix}1\\1\end{bmatrix}$



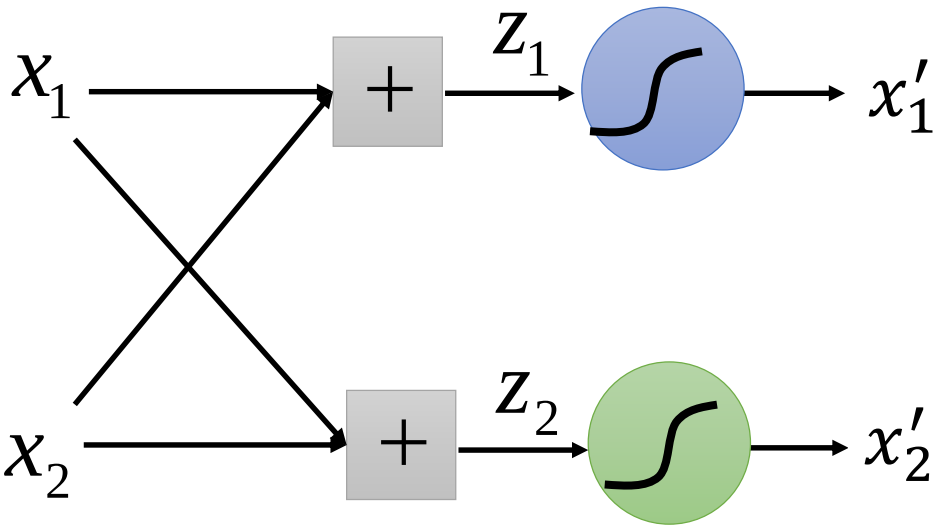

Not always easy to find a good transformation

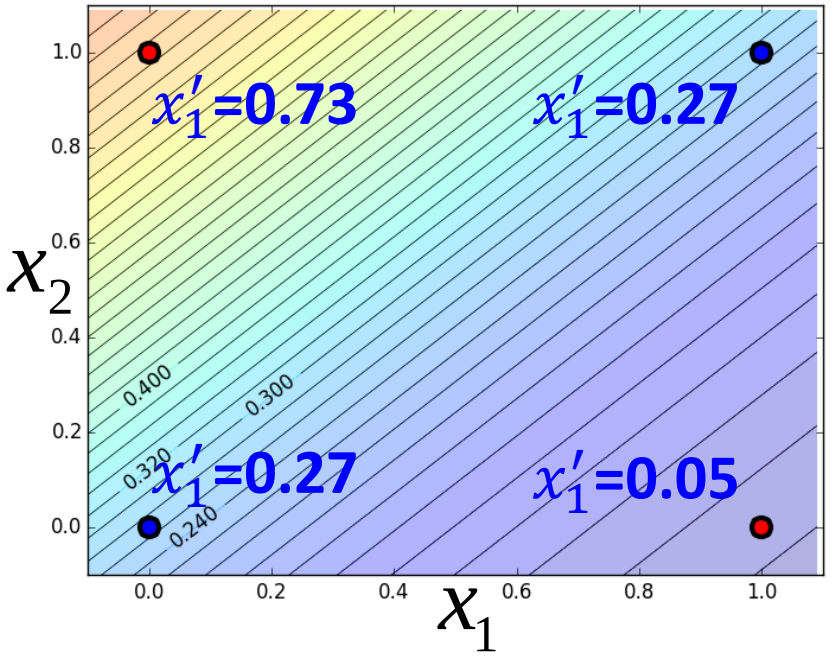
# Limitation of Logistic Regression

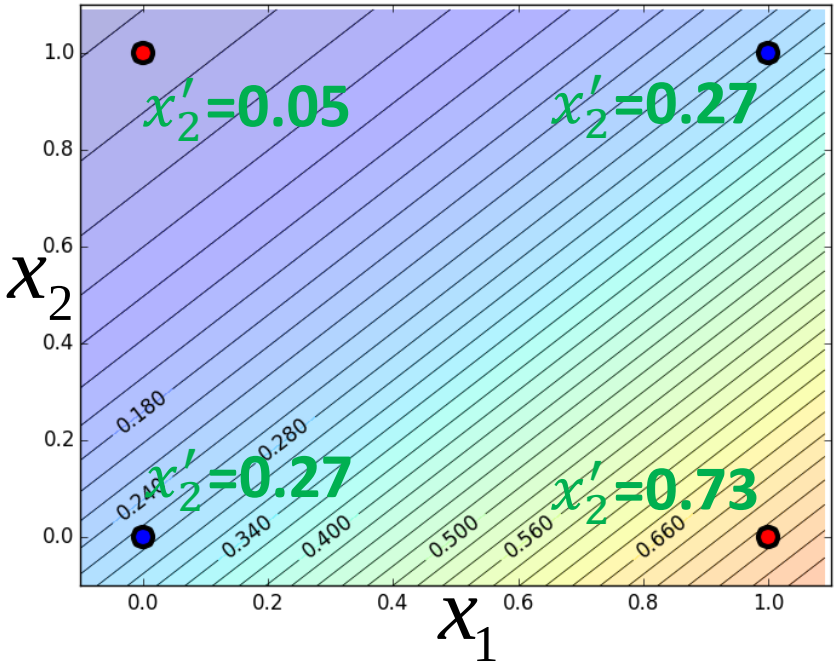
- Cascading logistic regression models

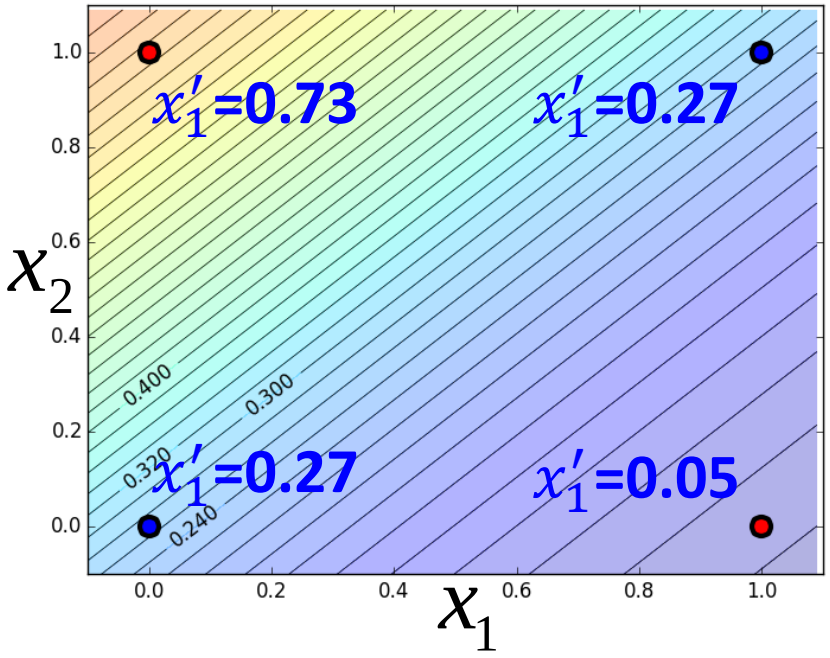


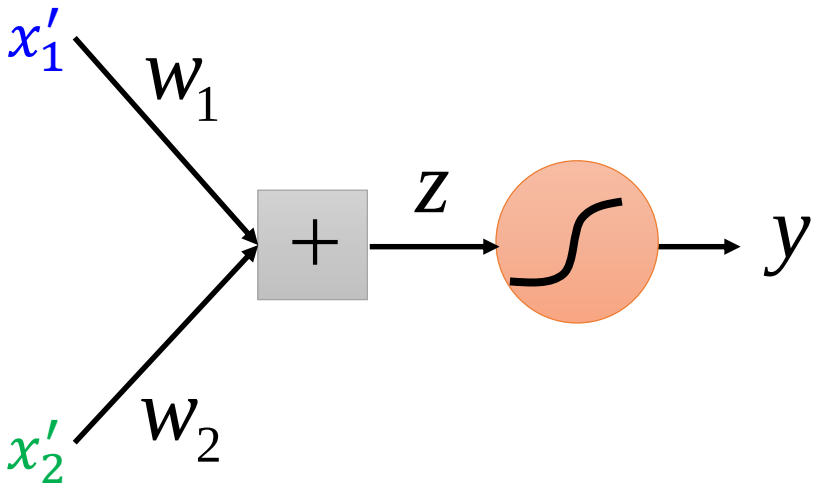

(ignore bias in this figure)

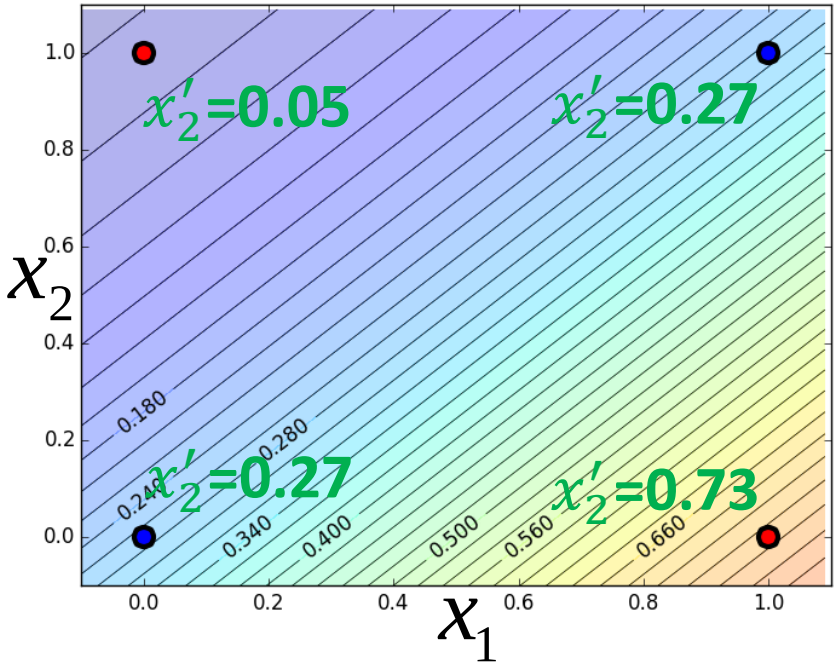

$x_1$
$x_2$
+
+
$z_1$
$z_2$
$x_1'$
$x_2'$

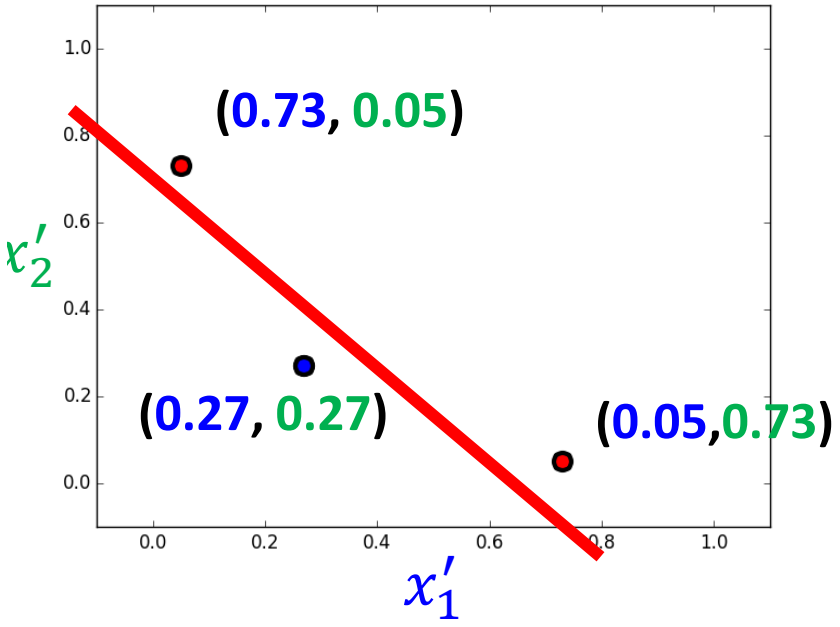

$x_1'$=0.73
$x_1'$=0.27
$x_1'$=0.27
$x_1'$=0.05
$x_2$
$x_1$
0.0
0.2
0.4
0.6
0.8
1.0
0.400
0.300
0.320
0.240


$x_2'$=0.05
$x_2'$=0.27
$x_2'$=0.27
$x_2'$=0.73
$x_2$
$x_1$
0.0
0.2
0.4
0.6
0.8
1.0
0.180
0.280
0.240
0.340
0.400
0.500
0.560
0.660

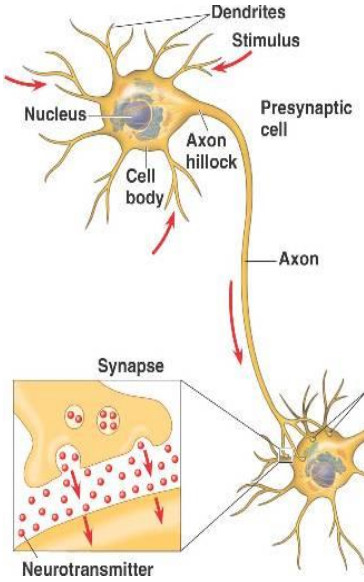

$x_1'$=0.73
$x_1'$=0.27
$x_1'$=0.27
$x_1'$=0.05
0.400
0.300
0.320
0.240
$x_2$
$x_1$

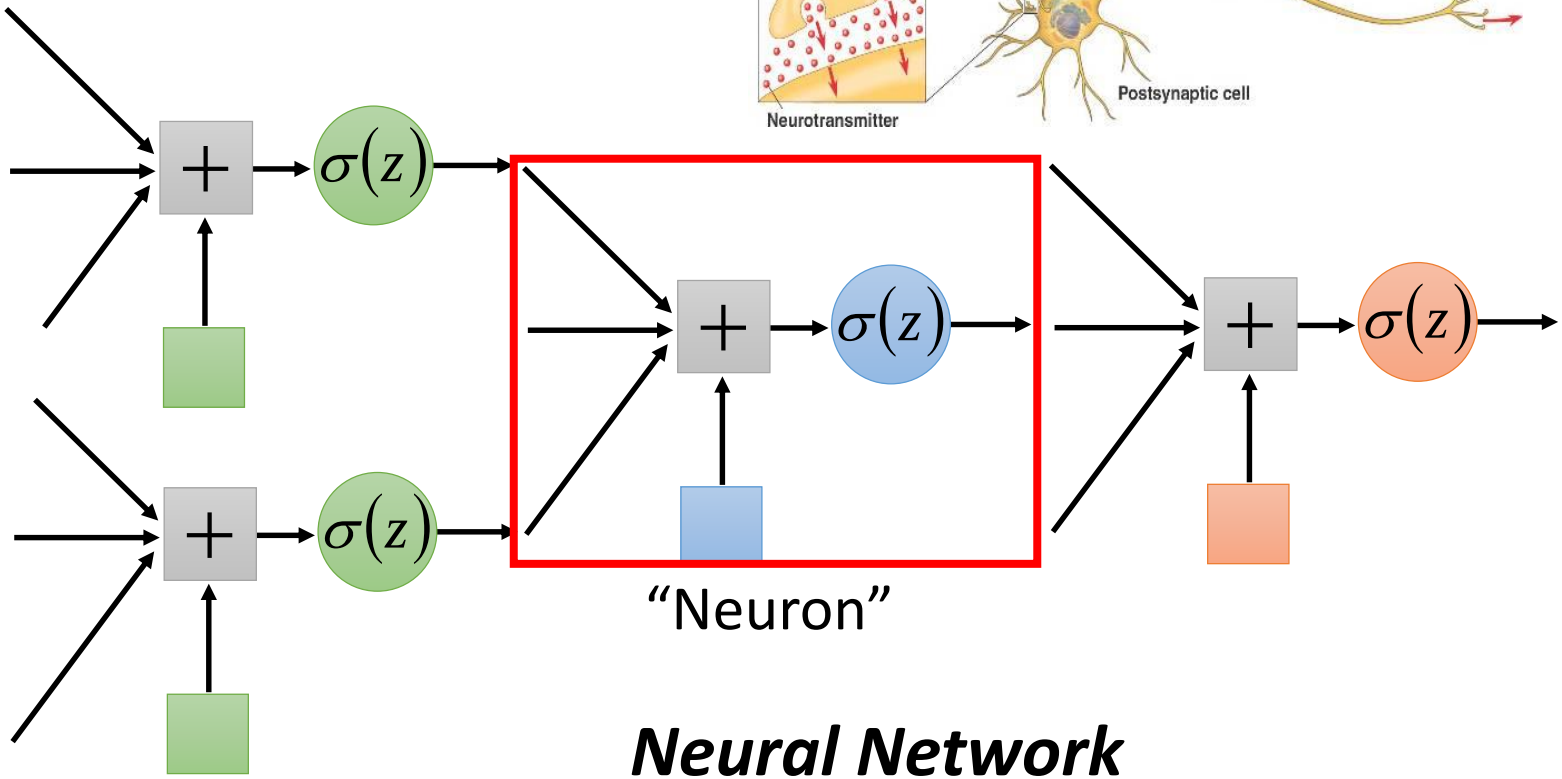

$x_1'$
$w_1$
$z$
$y$
$x_2'$
$w_2$


$x_2'$=0.05
$x_2'$=0.27
$x_2'$=0.27
$x_2'$=0.73
0.180
0.280
0.240
0.340
0.400
0.500
0.560
0.660
$x_2$
$x_1$


(0.73, 0.05)
(0.27, 0.27)
(0.05,0.73)
$x_2'$
$x_1'$

# Deep Learning!






"Neuron"

***Neural Network***

# Reference

- Bishop: Chapter 4.3