Logistic Regression - Classification

Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

Classification (Intuition)

- Email: Spam / Not Spam ?
- Online Transactions: Fraudulent (Yes / No) ?
- Tumor: Malignant / Benign ?

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

0: ‘Negative Class’ (*e.g.*, benign tumor)

1: ‘Positive Class’ (*e.g.*, malignant tumor)

Classification (Intuition)

- Email: Spam / Not Spam ?
- Online Transactions: Fraudulent (Yes / No) ?
- Tumor: Malignant / Benign ?

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

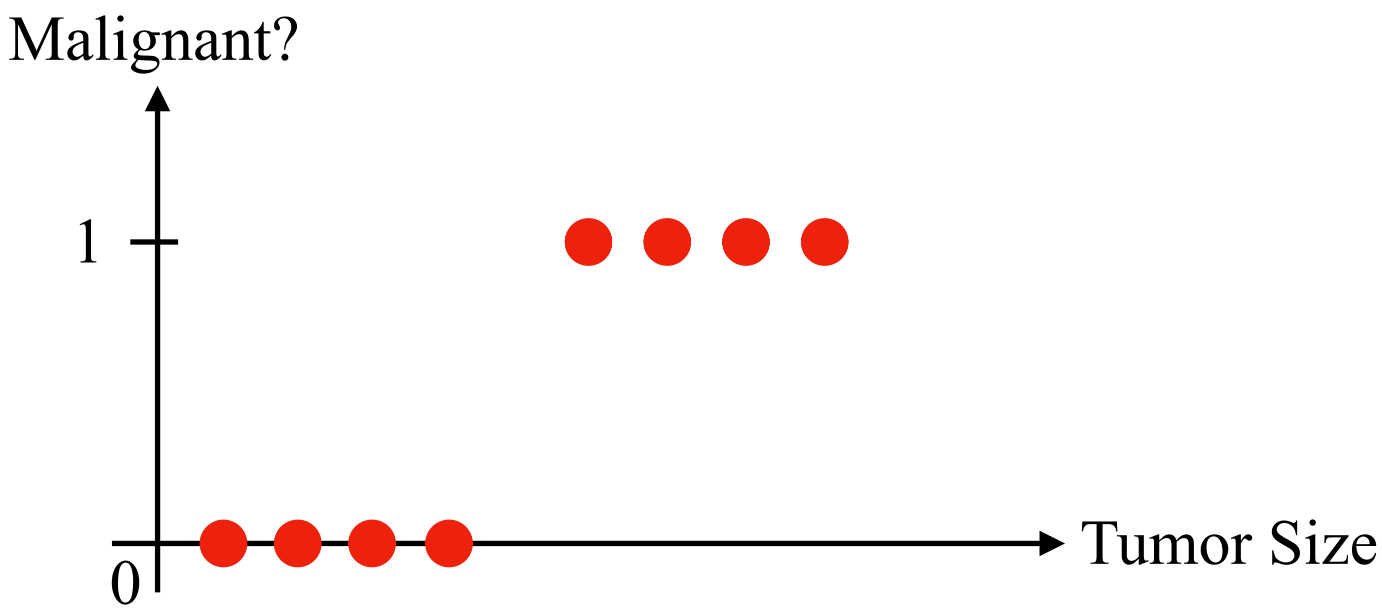
0: 'Negative Class' (*e.g.*, benign tumor)

1: 'Positive Class' (*e.g.*, malignant tumor)

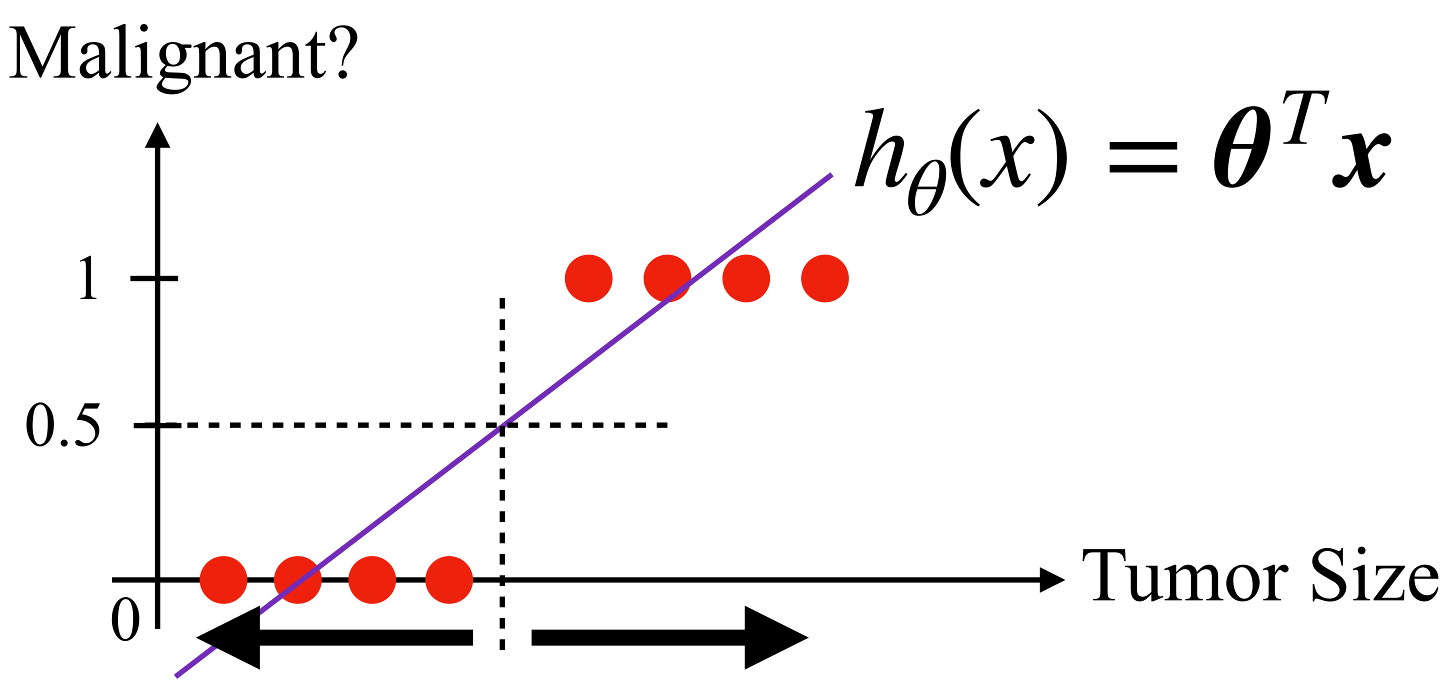
Later on, we will consider multi-classes *e.g.* $y \in \{0,1,2,3\}$

'multi-classes classification problem'

Classification (Intuition)



Classification (Intuition)



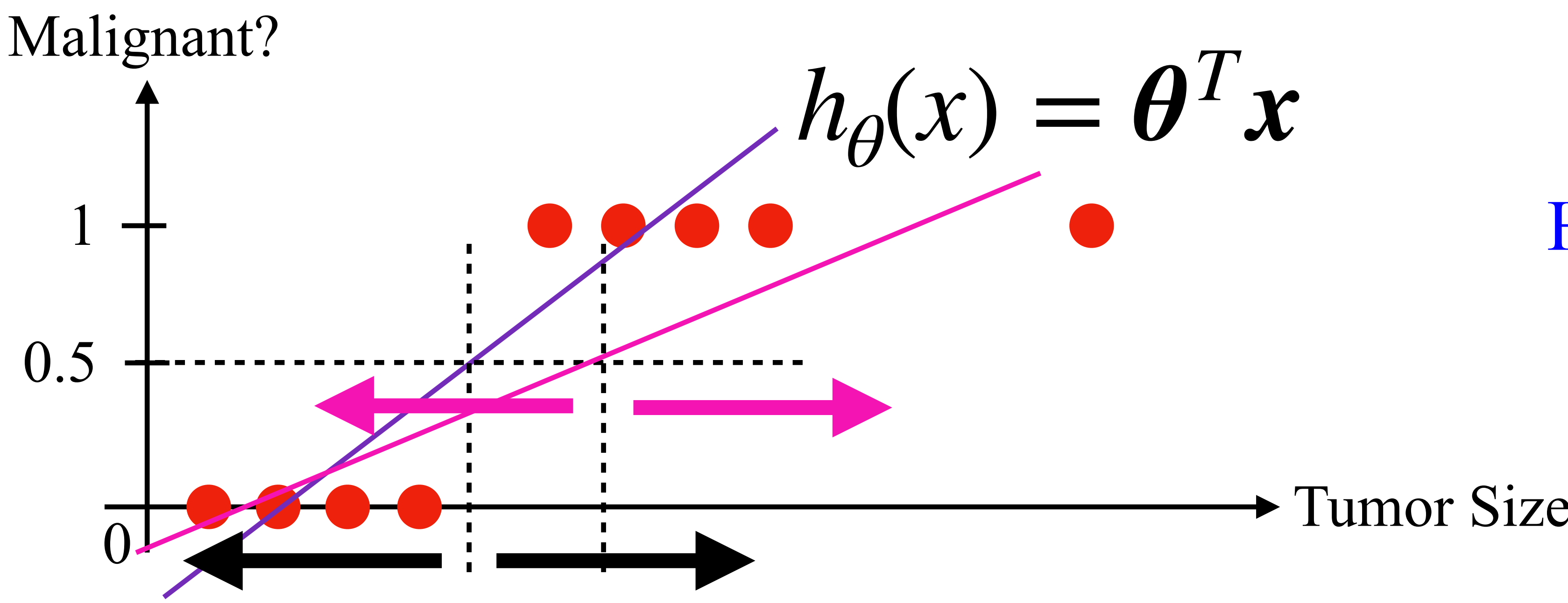
Ones may think like this !
But, this may not work well

Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict 'y = 1'

If $h_\theta(x) < 0.5$, predict 'y = 0'

Classification (Intuition)



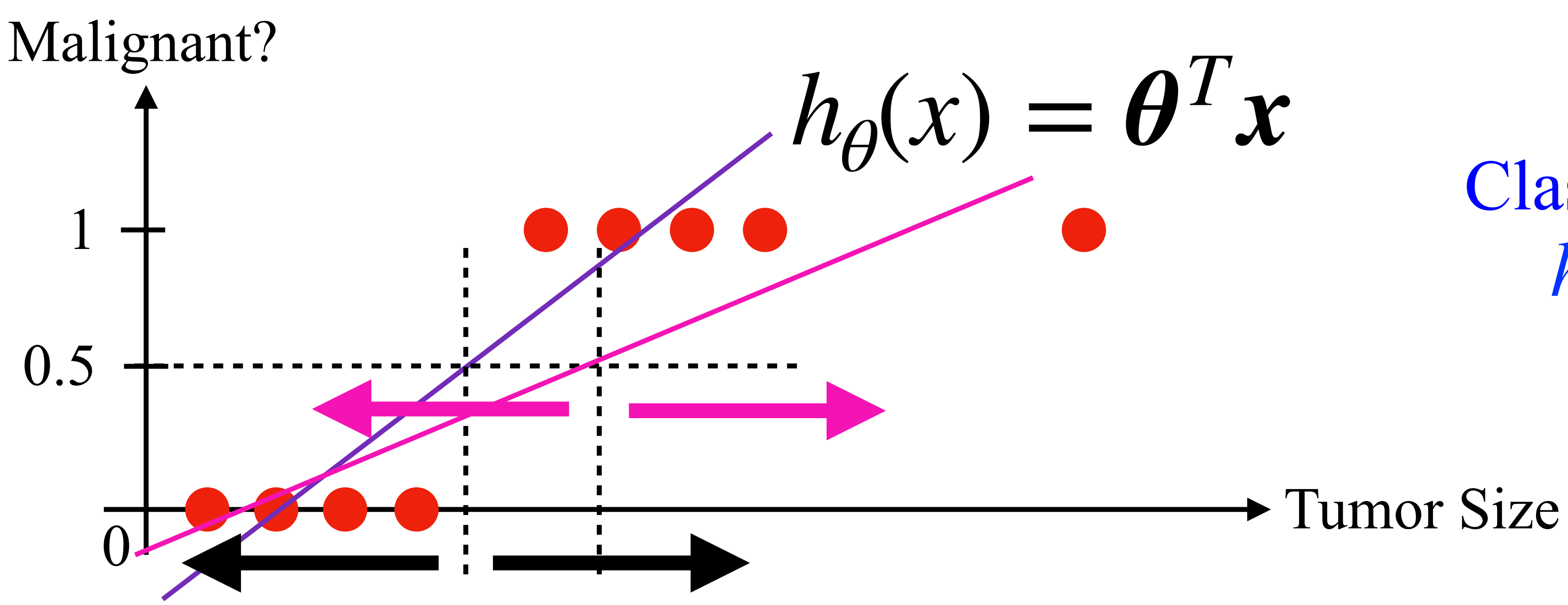
Here is a counter-example !

Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict 'y = 1'

If $h_\theta(x) < 0.5$, predict 'y = 0'

Classification (Intuition)



Classification: $y = 0$ or 1

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

Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict 'y = 1'

If $h_\theta(x) < 0.5$, predict 'y = 0'

Question

- Which of the following statements is true?

  (i) If linear regression doesn't work on a classification task as in the previous example, applying feature scaling may help.
  (ii) If the training set satisfies $0 \leq y^{(i)} \leq 1$ for every training example $(x^{(i)}, y^{(i)})$, then linear regression's prediction will also satisfy $0 \leq h_\theta(x) \leq 1$ for all values of x.
  (iii) If there is a feature x that perfectly predicts y *i.e.* if $y = 1$ when $x \geq c$ and $y = 0$ whenever $x < c$ (for some constant c), then linear regression will obtain zero classification error.
  (iv) None of the above statements are true.

Logistic Regression - Hypothesis Representation

Logistic Regression Model

Improvement: let's change the form of h to constrain it to the range [0, 1]:

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

Here, the function

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

is called the logistic sigmoid function or just the sigmoid function.

Logistic Regression Model

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

Improvement: let's change the form of h to constrain it to the range [0, 1]:

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

<u>Goal:</u> Fit the parameter θ to the data !

Here, the function

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

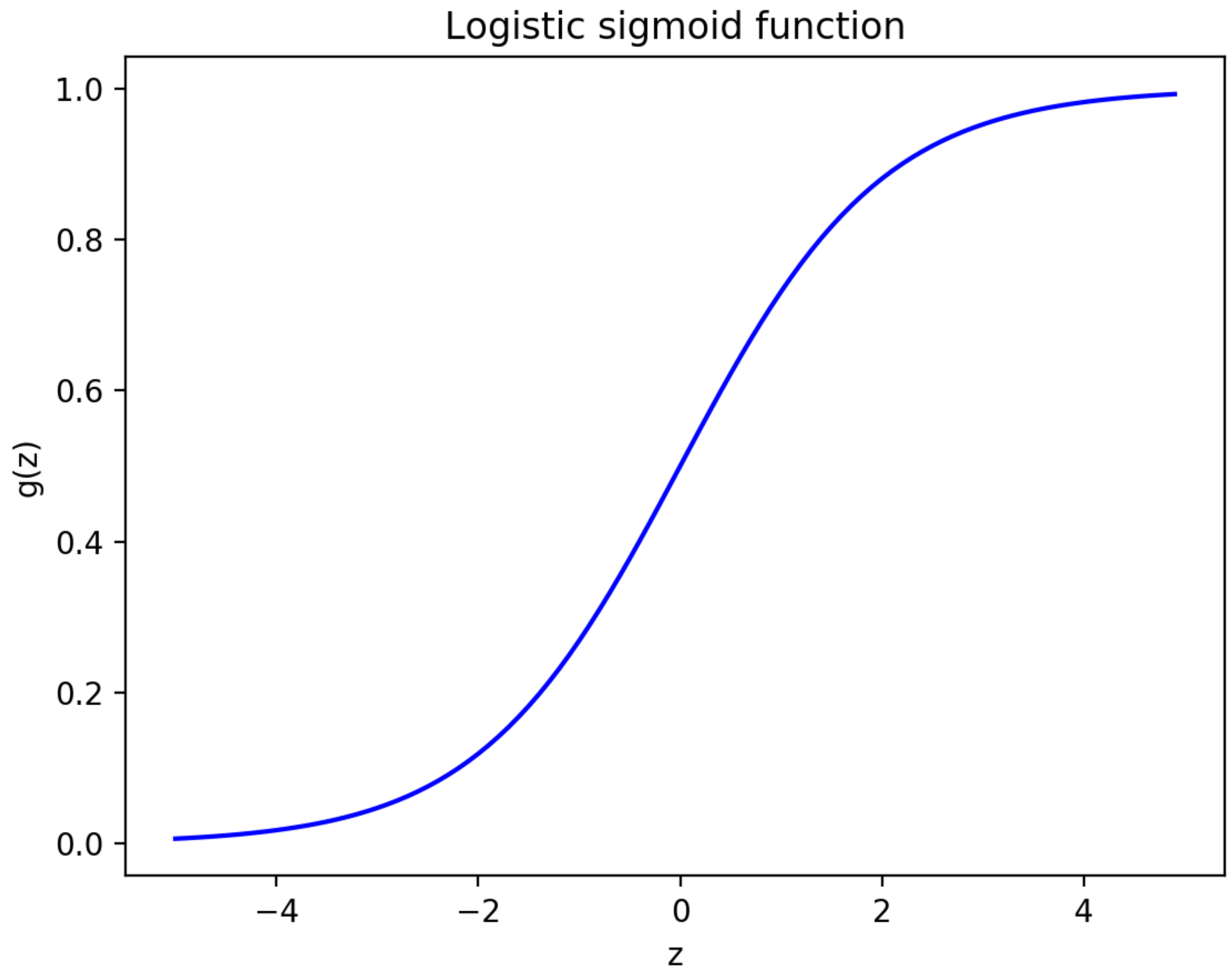
is called the logistic sigmoid function or just the sigmoid function.

Logistic Regression Model

```
import numpy
import matplotlib.pyplot as plt

def f(z):
    return 1/(1 + numpy.exp(-z))

z = numpy.arange(-5, 5, 0.1)
plt.plot(z, f(z), 'b')
plt.xlabel('z')
plt.ylabel('g(z)')
plt.title('Logistic sigmoid function')
```



Reading of Hypothesis Output

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

Estimated probability that $y = 1$ on input x

Reading of Hypothesis Output

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

Estimated probability that $y = 1$ on input x

Example: suppose $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \textbf{tumorSize} \end{bmatrix}$ and $h_\theta(x) = 0.7$

This means that there is a 70% chance of tumor being malignant !

Reading of Hypothesis Output

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

Estimated probability that $y = 1$ on input x

Example: suppose $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \textbf{tumorSize} \end{bmatrix}$ and $h_\theta(x) = 0.7$

This means that there is a 70% chance of tumor being malignant !

Mathematically, $h_\theta(x) = P(y = 1 \mid x; \theta)$

'probability that $y = 1$, given x, parameterized by θ'

Question

- Suppose we want to predict, from data x about a tumor, whether it is malignant ($y = 1$) or benign ($y = 0$). Our logistic regression classifier outputs, for a specific tumor, $h_\theta(x) = P(y = 1 \mid x; \theta) = 0.7$, so we estimate that there is a 70% chance of this tumor being malignant. What should be our estimate for $P(y = 0 \mid x; \theta)$, the probability the tumor is benign?

(i) $P(y = 0 \mid x; \theta) = 0.3$

(ii) $P(y = 0 \mid x; \theta) = 0.7$

(iii) $P(y = 0 \mid x; \theta) = 0.7^2$

(iv) $P(y = 0 \mid x; \theta) = 0.3 \times 0.7$

Hint:

$$P(y = 0 \mid x; \theta) + P(y = 1 \mid x; \theta) = 1$$

Decision Boundary

Decision Boundary (Intuition)

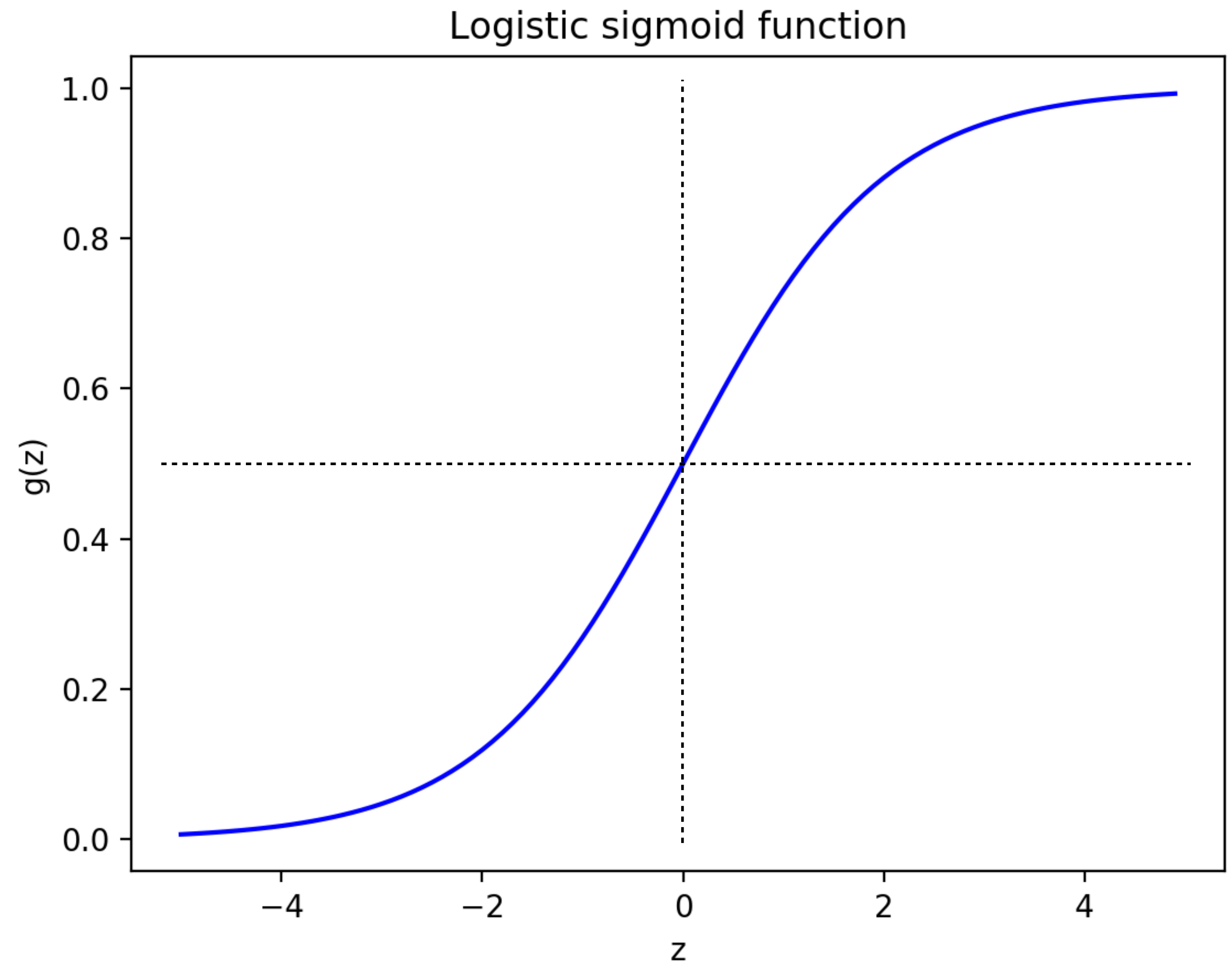
$P(y = 1 \mid x; \theta)$

Hypothesis:

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

$\because g(z) \geq 0.5$ when $z \geq 0$

$\therefore h_\theta(x) = g(\boldsymbol{\theta}^T \boldsymbol{x}) \geq 0.5$ when $\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$



Decision Boundary (Intuition)

$P(y = 1 \mid x; \theta)$

Hypothesis:

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

$\because g(z) \geq 0.5$ when $z \geq 0$

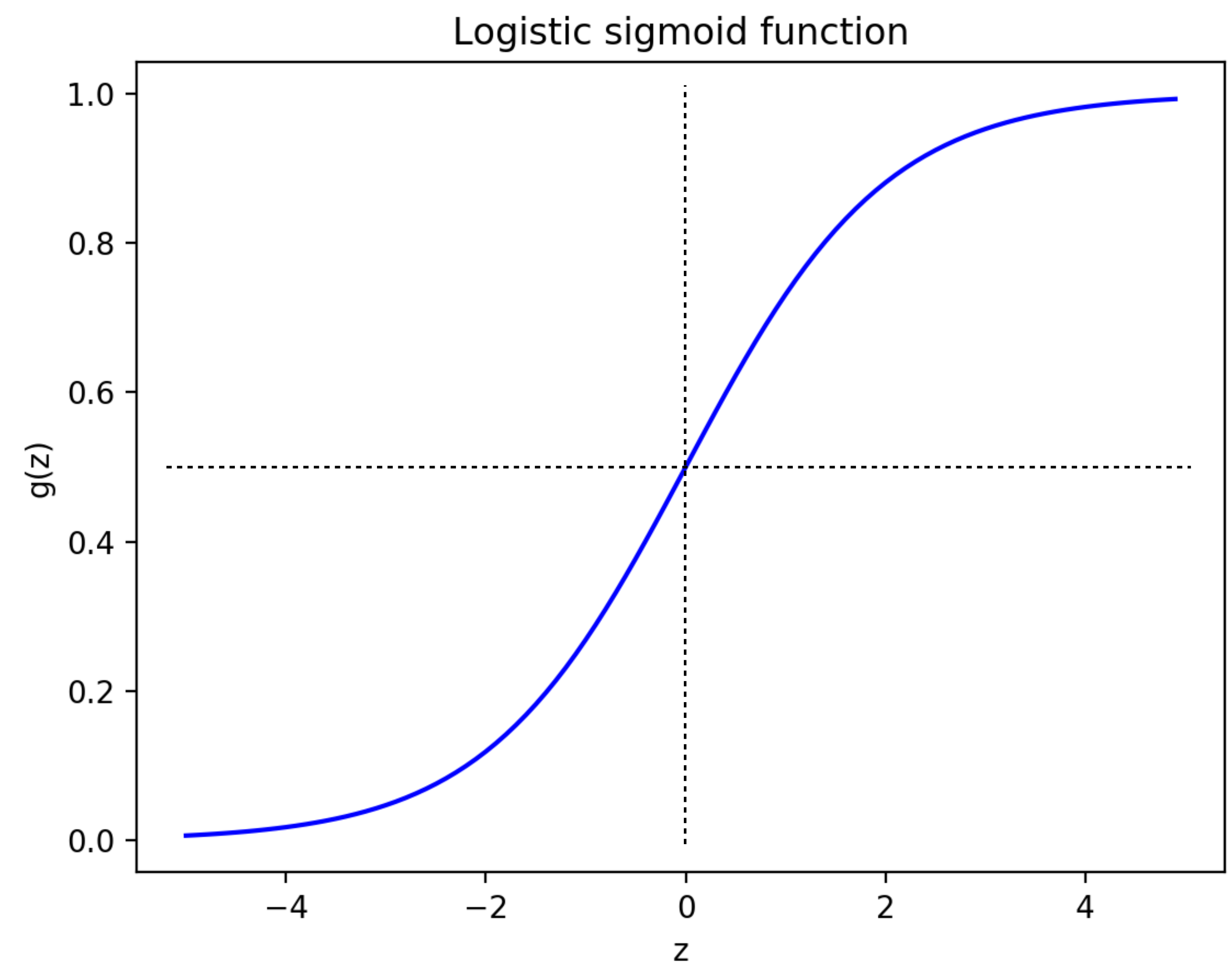
$\therefore h_\theta(x) = g(\boldsymbol{\theta}^T \boldsymbol{x}) \geq 0.5$ when $\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$

- Predict '$y = 1$' if $h_\theta(x) \geq 0.5$

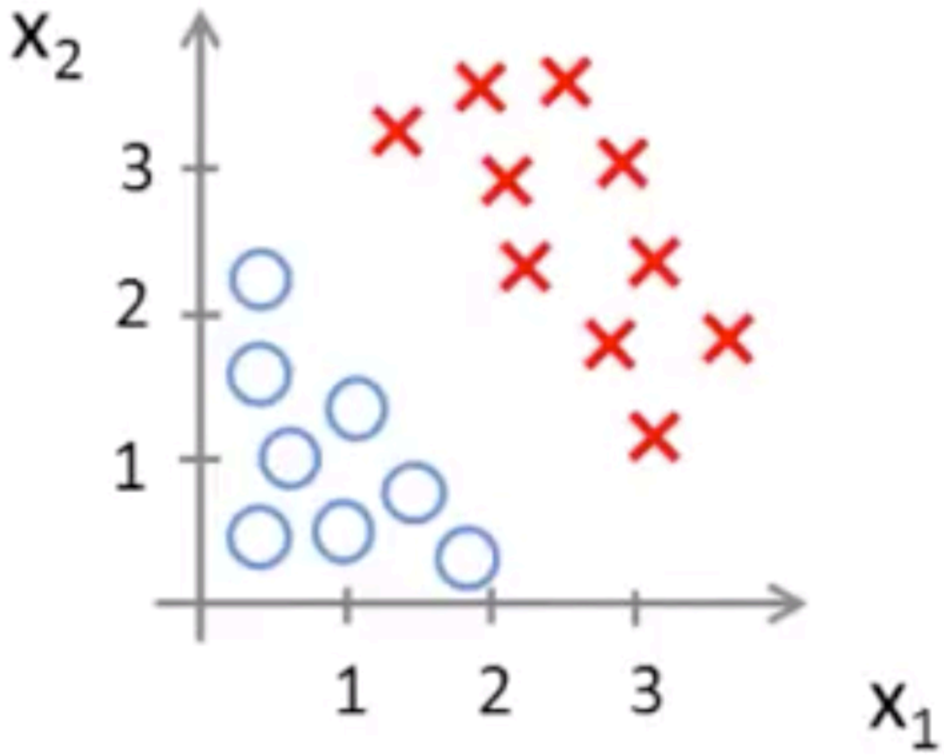
  $$\boldsymbol{\theta}^T \boldsymbol{x} \geq 0$$

- Predict '$y = 0$' if $h_\theta(x) < 0.5$

  $$\boldsymbol{\theta}^T \boldsymbol{x} < 0$$



Decision Boundary (Intuition)

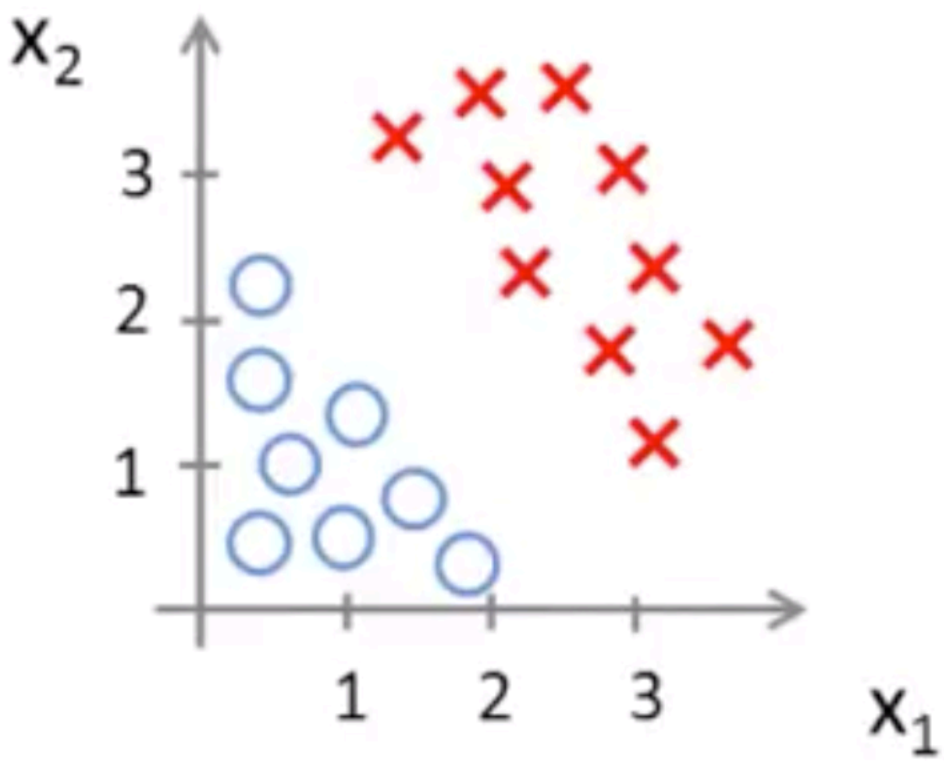


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

-3 1 1

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

Decision Boundary (Intuitively)



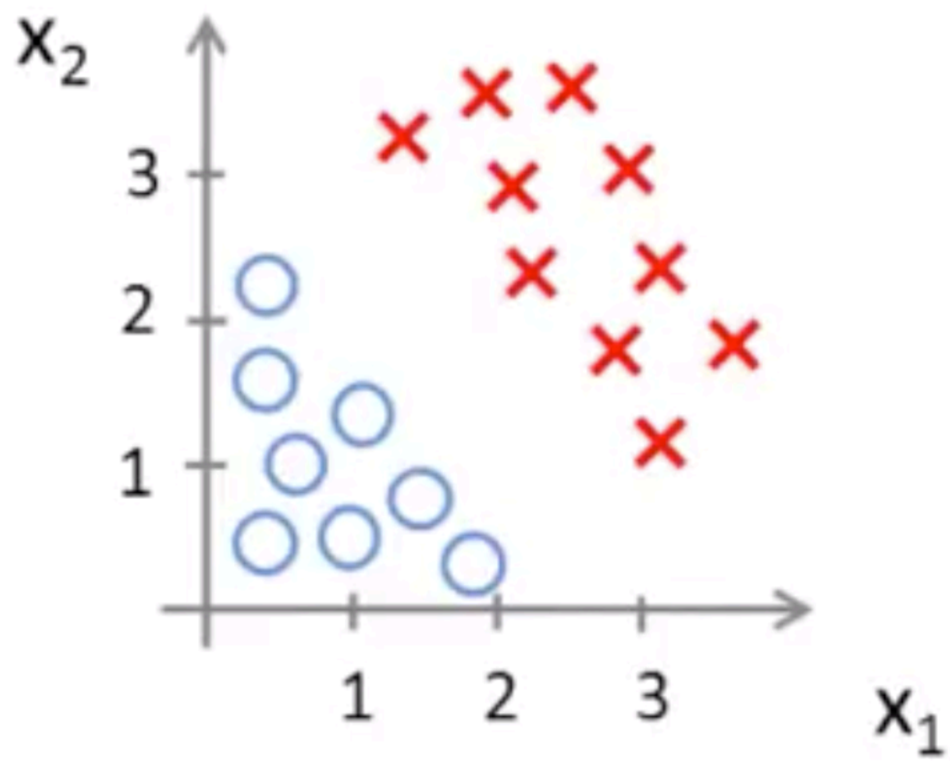
$$h_\theta(x) = g(\underset{-3}{\theta_0} + \underset{1}{\theta_1} x_1 + \underset{1}{\theta_2} x_2)$$

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

$$\because \theta^T x = [-3 \quad 1 \quad 1] \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = -3 + x_1 + x_2$$

$\therefore$ Predict '$y = 1$' if $-3 + x_1 + x_2 \geq 0$

Decision Boundary (Intuitively)



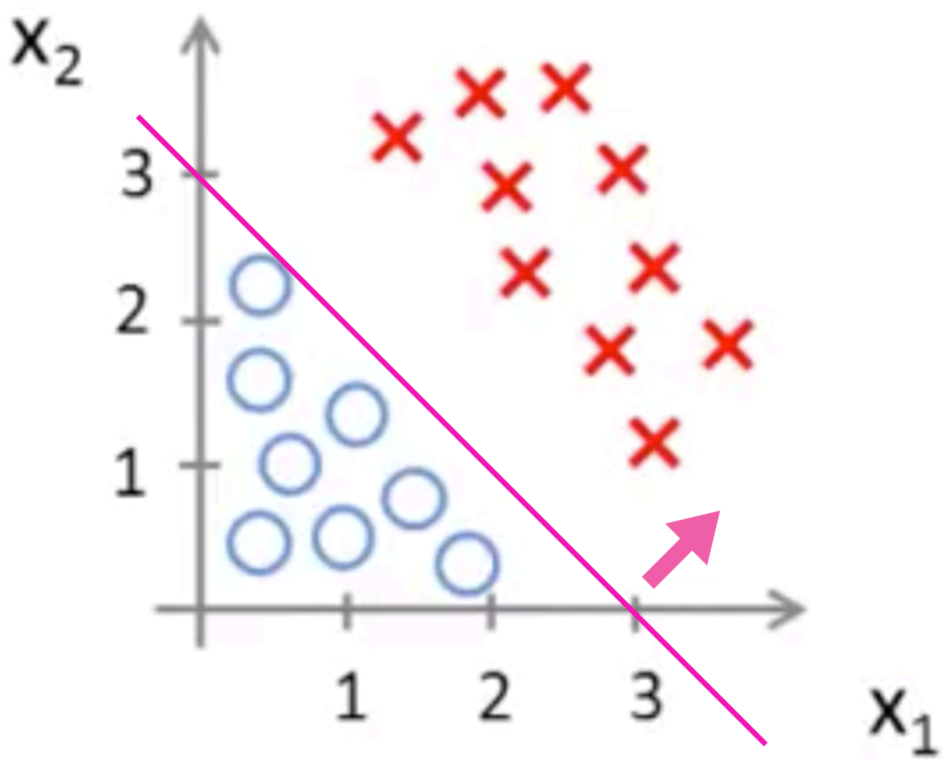
$$h_\theta(x) = g(\underbrace{\theta_0}_{-3} + \underbrace{\theta_1}_{1} x_1 + \underbrace{\theta_2}_{1} x_2)$$

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

$$\because \theta^T x = [-3 \quad 1 \quad 1] \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = -3 + x_1 + x_2$$

$\therefore$ Predict '$y = 1$' if $-3 + x_1 + x_2 \geq 0 \Longleftrightarrow x_1 + x_2 \geq 3$

Decision Boundary (Intuitively)



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

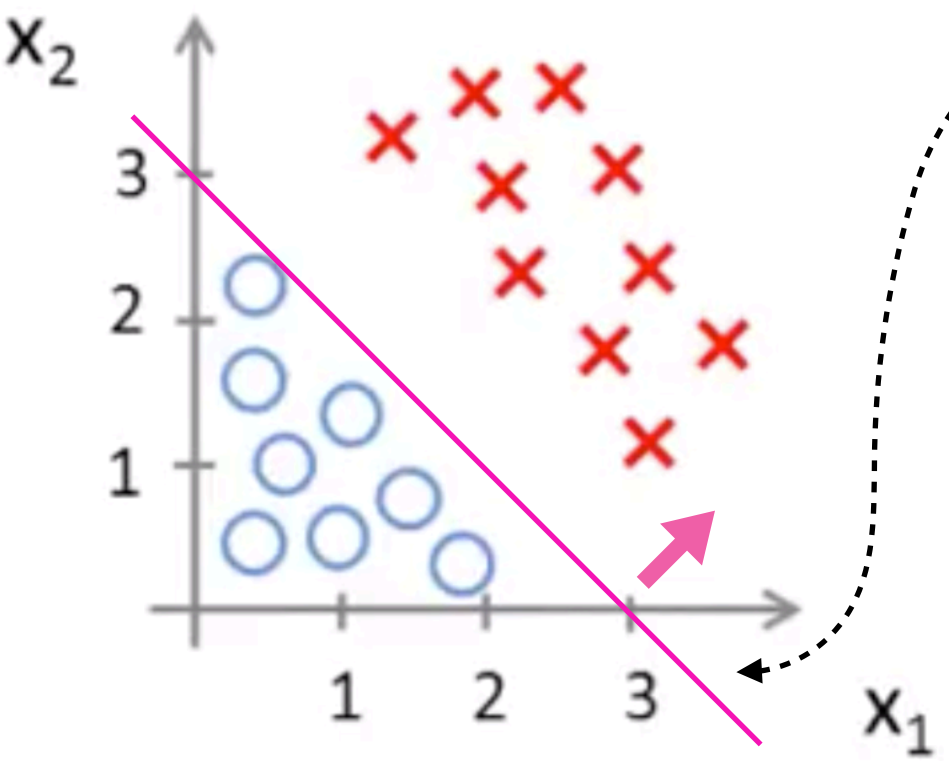
-3 1 1

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

$$\because\ \theta^T x = [-3 \quad 1 \quad 1] \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = -3 + x_1 + x_2$$

$\therefore$ Predict '$y = 1$' if $-3 + x_1 + x_2 \geq 0 \iff \boxed{x_1 + x_2 \geq 3}$

Decision Boundary



Decision Boundary

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

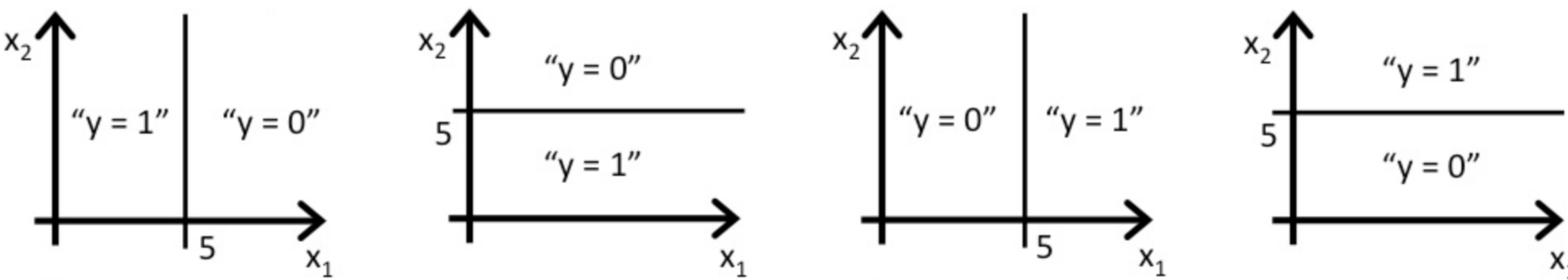
$$i.e.\ \ \theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

$$\because\ \theta^T x = [-3 \quad 1 \quad 1] \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = -3 + x_1 + x_2$$

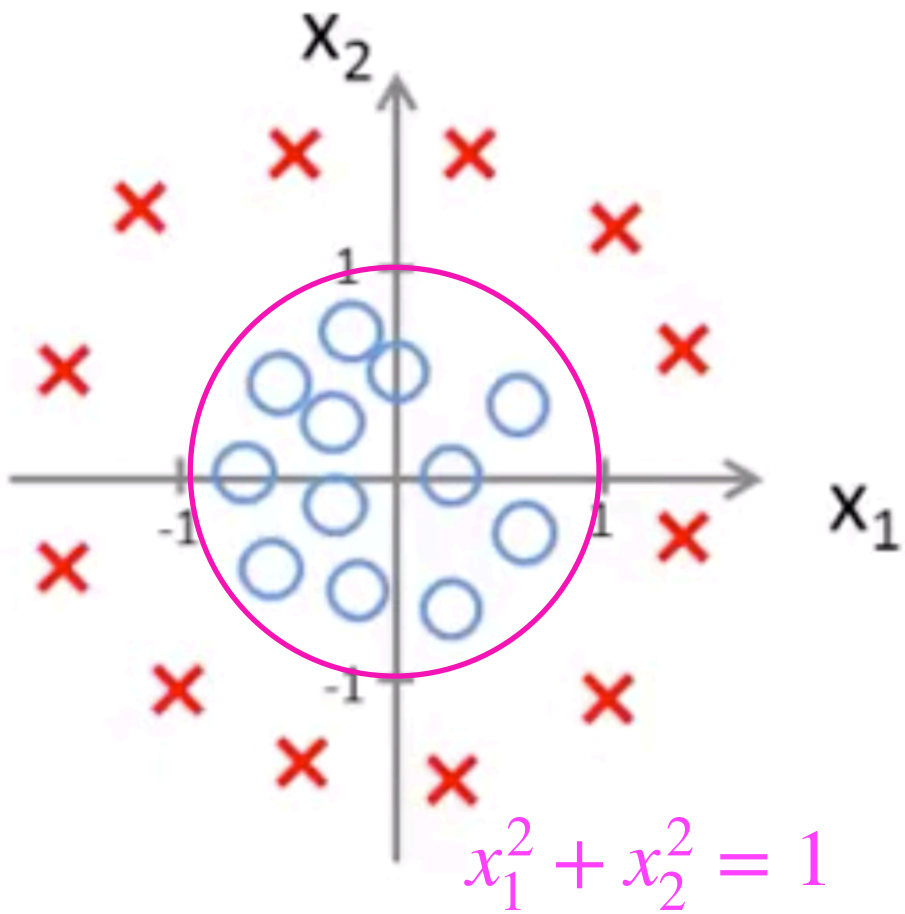
$$\therefore\ \text{Predict '}y = 1\text{' if } -3 + x_1 + x_2 \geq 0 \Longleftrightarrow \boxed{x_1 + x_2 \geq 3}$$

Question

- Consider logistic regression with two features x_1 and x_2. Suppose $\theta_0 = 5, \theta_1 = -1, \theta_2 = 0$, so that $h_\theta(x) = g(5 - x_1)$. Which of these shows the decision boundary of $h_\theta(x)$?



Non-linear Decision Boundary



$$h_\theta(x) = g(\underset{-1}{\theta_0} + \underset{0}{\theta_1} x_1 + \underset{0}{\theta_2} x_2 + \underset{1}{\theta_3} x_1^2 + \underset{1}{\theta_4} x_2^2)$$

$$i.e.\ \theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$\therefore$ Predict '$y = 1$' if $-1 + x_1^2 + x_2^2 \geq 0$

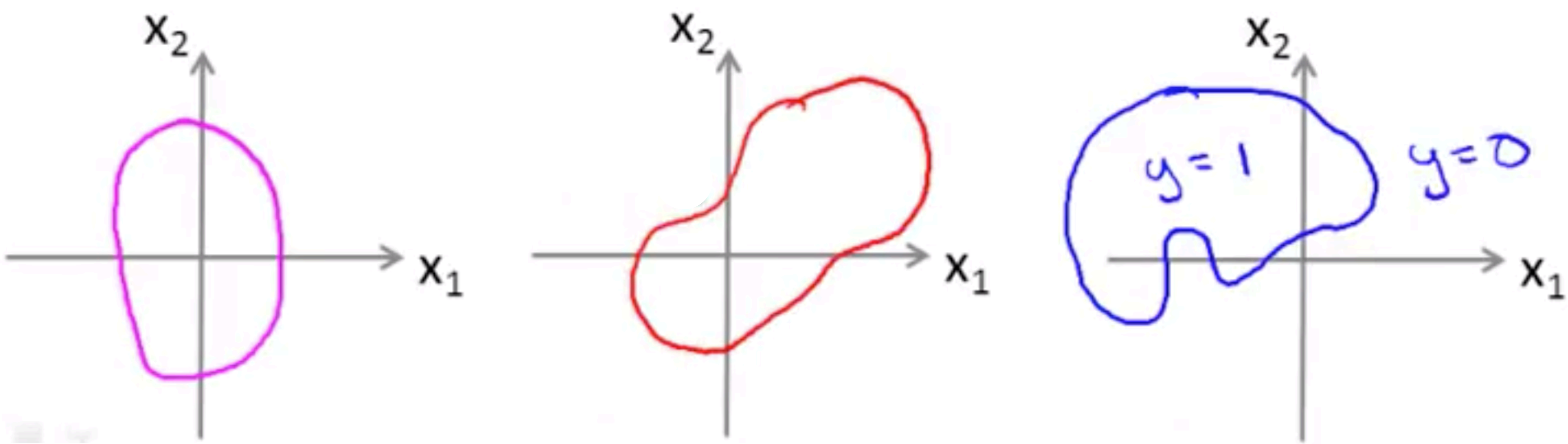
$$\iff x_1^2 + x_2^2 \geq 1$$

x : positive example

o : negative example

Non-linear Decision Boundary

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \ldots)$$



Cost Function
(Probabilistic Interpretation)

Cost Function

So far, we have

1. Training set: let's first restrict ourselves to binary classification
2. Hypothesis function:

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

What else do we need ?

Cost Function

So far, we have

1. Training set: let's first restrict ourselves to binary classification
2. Hypothesis function:

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

What else do we need ?

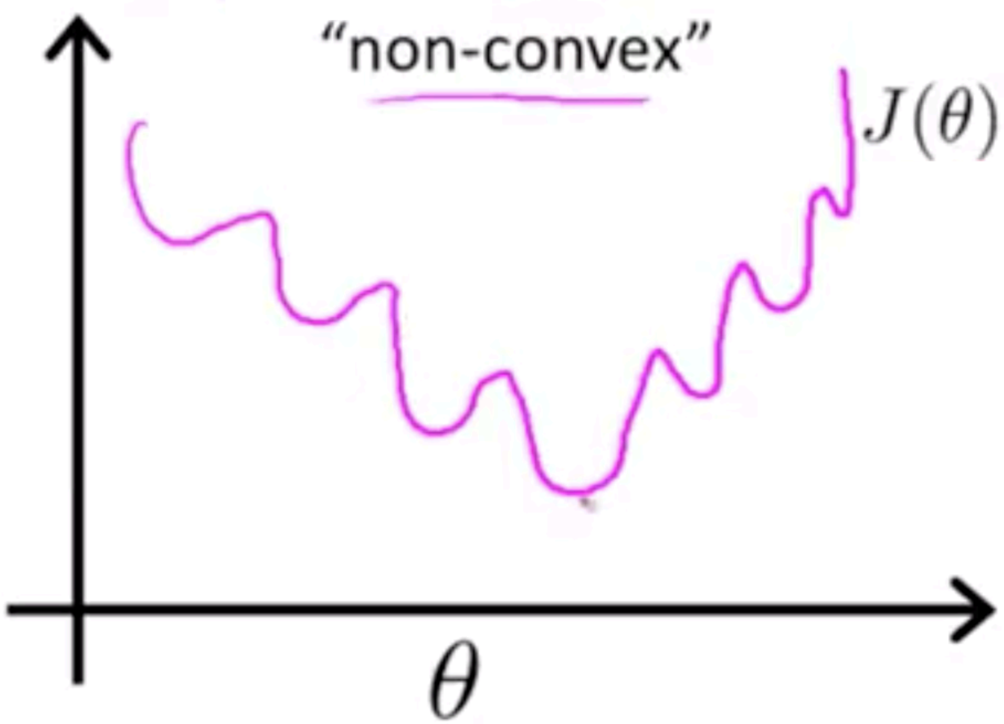
3. Cost function
4. Methods to minimize that cost

Cost Function

Now, least square does not make sense for classification. (Why ?)

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

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



Cost Function

Let's instead maximize the likelihood (actually, the log likelihood) of $\boldsymbol{\theta}$

The likelihood of $\boldsymbol{\theta}$ is:

$$L(\boldsymbol{\theta}) = p(\boldsymbol{y} \mid \boldsymbol{X}; \boldsymbol{\theta})$$
$$= \Pi_{i=1}^{m} p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$$

How should we model $p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$?

It should be maximal when our prediction $h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})$ is correct and minimal when the prediction is incorrect.

Cost Function

In logistic regression, we assume

$$p(y\,|\,\boldsymbol{x};\boldsymbol{\theta}) = \begin{cases} h_\theta(\boldsymbol{x}) & \textbf{if } y = 1 \\ 1 - h_\theta(\boldsymbol{x}) & \textbf{otherwise} \end{cases}$$

That is, $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ models $p(y = 1\,|\,\boldsymbol{x})$.

(Why is this reasonable ?)

A compact representation of this form is:

$$p(y\,|\,\boldsymbol{x};\boldsymbol{\theta}) = (h_\theta(\boldsymbol{x}))^y(1 - h_\theta(\boldsymbol{x}))^{1-y}$$

Cost Function

We thus obtain

$$
\begin{aligned}
L(\boldsymbol{\theta}) &= \Pi_{i=1}^{m} p(y^{(i)} \,|\, \boldsymbol{x}^{(i)}; \boldsymbol{\theta}) \\
&= \Pi_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1 - h_\theta(\boldsymbol{x}))^{1-y^{(i)}}
\end{aligned}
$$

and the log likelihood is:

$$
\begin{aligned}
l(\boldsymbol{\theta}) &= \log L(\boldsymbol{\theta}) \\
&= \Sigma_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(\boldsymbol{x}^{(i)}))
\end{aligned}
$$

Note: the log of a product is the sum of logs of the things inside the product !

Cost Function

To find the maximum of $l(\boldsymbol{\theta})$, we first need to find $\nabla_l(\boldsymbol{\theta})$.

Since we don't have a closed form solution for $\nabla_l(\boldsymbol{\theta}) = 0$, so instead we have to use gradient ascent.

For simplicity, let us try the stochastic version of gradient ascent *i.e.* we assume the training set is just a single pair $(\boldsymbol{x}, y)$.

Note that gradient descent aims at minimizing an objective function whereas gradient ascent aims at maximizing an objective function.

Cost Function

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \left(y \cdot \frac{1}{g(\boldsymbol{\theta}^T \boldsymbol{x})} - (1-y) \cdot \frac{1}{1 - g(\boldsymbol{\theta}^T \boldsymbol{x})} \right) \boxed{\frac{\partial}{\partial \theta_j} g(\boldsymbol{\theta}^T \boldsymbol{x})}$$

We need some tricks to deal with this !

Here is a trick dealing with a function involving sigmoid :

$$g'(z) = \frac{d}{dz} \frac{1}{1+e^{-z}} = \frac{1}{(1+e^{-z})^2} (e^{-z})$$

$$= \frac{1}{1+e^{-z}} \cdot \left(\frac{1+e^{-z}-1}{1+e^{-z}} \right)$$

$$= g(z)(1-g(z))$$

Cost Function

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \left(y \cdot \frac{1}{g(\boldsymbol{\theta}^T \boldsymbol{x})} - (1-y) \cdot \frac{1}{1 - g(\boldsymbol{\theta}^T \boldsymbol{x})} \right) \frac{\partial}{\partial \theta_j} g(\boldsymbol{\theta}^T \boldsymbol{x})$$

Since

$$\frac{\partial}{\partial \theta_j} g(\boldsymbol{\theta}^T \boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})(1 - g(\boldsymbol{\theta}^T \boldsymbol{x})) \frac{\partial}{\partial \theta_j} (\boldsymbol{\theta}^T \boldsymbol{x})$$

we then obtain

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \Big(y(1 - g(\boldsymbol{\theta}^T \boldsymbol{x})) - (1-y) g(\boldsymbol{\theta}^T \boldsymbol{x}) \Big) x_j = (y - h_\theta(\boldsymbol{x})) x_j$$

Therefore, $\nabla_l(\boldsymbol{\theta}) = (y - h_\theta(\boldsymbol{x}))\boldsymbol{x}$

Cost Function

We thus get the stochastic gradient rule for logistic regression:

$$\boldsymbol{\theta}^{(n+1)} := \boldsymbol{\theta}^{(n)} + \alpha(y^{(i)} - h_\theta(\boldsymbol{x}^{(i)}))\boldsymbol{x}^{(i)}$$

Does this look familiar? Take a look at slide no. 25 of Lecture 2.1 !

Note that the update is not exactly the same since $h_\theta(\boldsymbol{x})$ is not the same.

Cost Function

We thus get the stochastic gradient rule for logistic regression:

$$\boldsymbol{\theta}^{(n+1)} := \boldsymbol{\theta}^{(n)} + \alpha(y^{(i)} - h_\theta(\boldsymbol{x}^{(i)}))\boldsymbol{x}^{(i)}$$

Does this look familiar? Take a look at slide no. 25 of Lecture 2.1 !

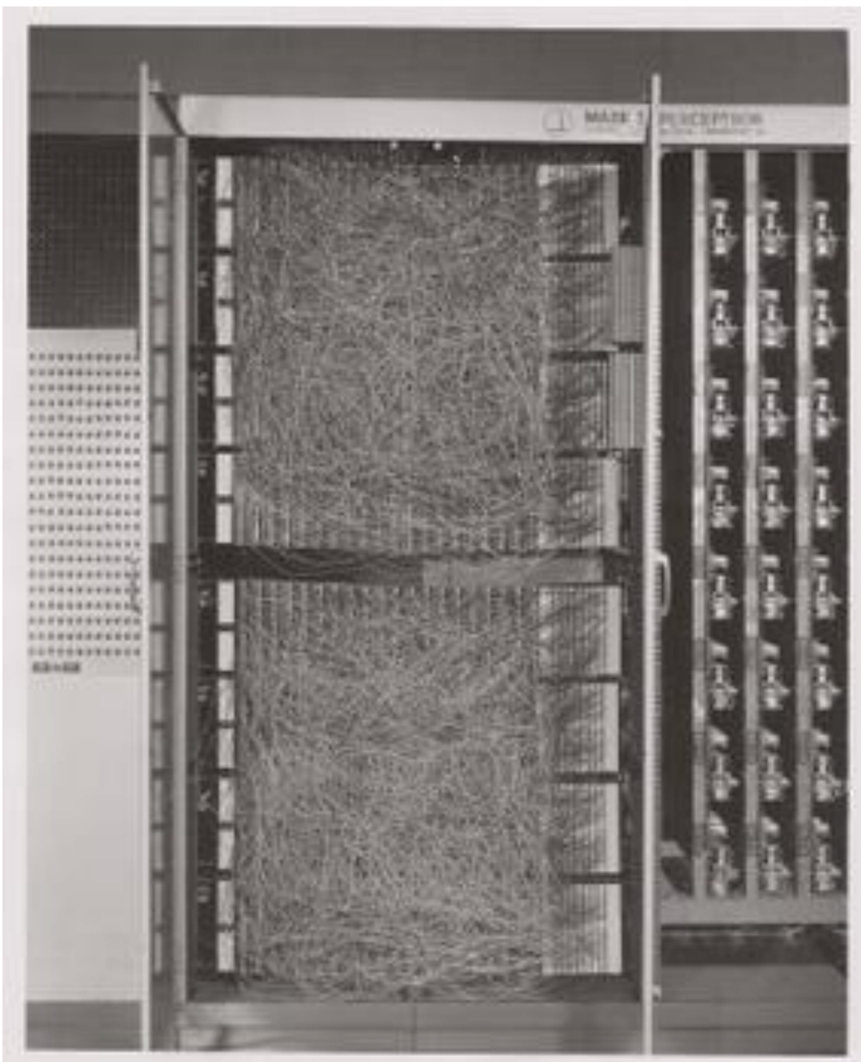
Note that the update is not exactly the same since $h_\theta(\boldsymbol{x})$ is not the same.

Still, it is amazing that we get similar update rules for:

- Linear regression with least squares
- Linear regression with maximum likelihood
- Logistic regression with maximum likelihood

Relationship to the perceptron

The 1^{st} Neural Network



Mark I Perceptron Machine
(Wikipedia)

In 1957, Rosenblatt conceived of the perceptron, a physical machine implementing the classification function

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

with

$$g(z) = \begin{cases} 1 & \textbf{if } z \geq 0 \\ 0 & \textbf{otherwise} \end{cases}$$

The 1st Neural Network

The perceptron learning algorithm also used the update rule:

$$\boldsymbol{\theta}^{(n+1)} := \boldsymbol{\theta}^{(n)} + \alpha(y^{(i)} - h_\theta(\boldsymbol{x}^{(i)}))\boldsymbol{x}^{(i)}$$

However, this is different from the logistic and linear regression rules since $h_\theta(\boldsymbol{x})$ is in this case a hard threshold classifier without any probabilistic interpretation.

The 1st Neural Network

Rosenblatt was extremely optimistic about the perceptron. It was believed that the machine may learn and translate languages in the future.

Minsky and Papert's (1969) Perceptrons [1] critiqued the power of the perceptron and helped kill neural network for many years.

Neural network research later went through two 'rebirths':

1. Rumelhart, Hinton, and Williams (1986): back propagation
2. Krizhevsky, Sutskever, and Hinton (2012): AlexNet

Now, we are in the 3rd cycle of possibly over-hyped expectations about neural networks !

[1] Minsky, Marvin, and Seymour Papert. "Perceptrons: An essay in computational geometry." (1969).

Summary (Part 1)

We are now familiar with solving supervised learning problems through:

- Direct solution of $\nabla_J(\boldsymbol{\theta}) = \mathbf{0}$ or $\nabla_l(\boldsymbol{\theta}) = \mathbf{0}$
- Gradient descent on a cost function $\boldsymbol{\theta}^{(n+1)} := \boldsymbol{\theta}^{(n)} - \alpha \nabla_J(\boldsymbol{\theta})$
- Gradient ascent on a log likelihood function $\boldsymbol{\theta}^{(n+1)} := \boldsymbol{\theta}^{(n)} + \alpha \nabla_l(\boldsymbol{\theta})$

Cost Function
(*w.r.t.* Mathematical Sense)

Things we have

Training set: $\{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(m)}, y^{(m)})\}$

$$\text{such that} \quad \boldsymbol{x}^{(i)} \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}_{\mathbb{R}^{n+1}}, \; x_0 = 1, \; y^{(i)} \in \{0,1\}$$

Hypothesis: $h_\theta(\boldsymbol{x}) = \dfrac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$

Goal: how to choose parameters $\boldsymbol{\theta}$ to minimize the cost error ?

What does the cost error function look like ?

Things we have

Training set: $\{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(m)}, y^{(m)})\}$

such that $\boldsymbol{x}^{(i)} \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}_{\mathbb{R}^{n+1}}$, $x_0 = 1,\ y^{(i)} \in \{0,1\}$

Hypothesis: $h_\theta(\boldsymbol{x}) = \dfrac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$

Goal: how to choose parameters $\boldsymbol{\theta}$ to minimize the cost error ?

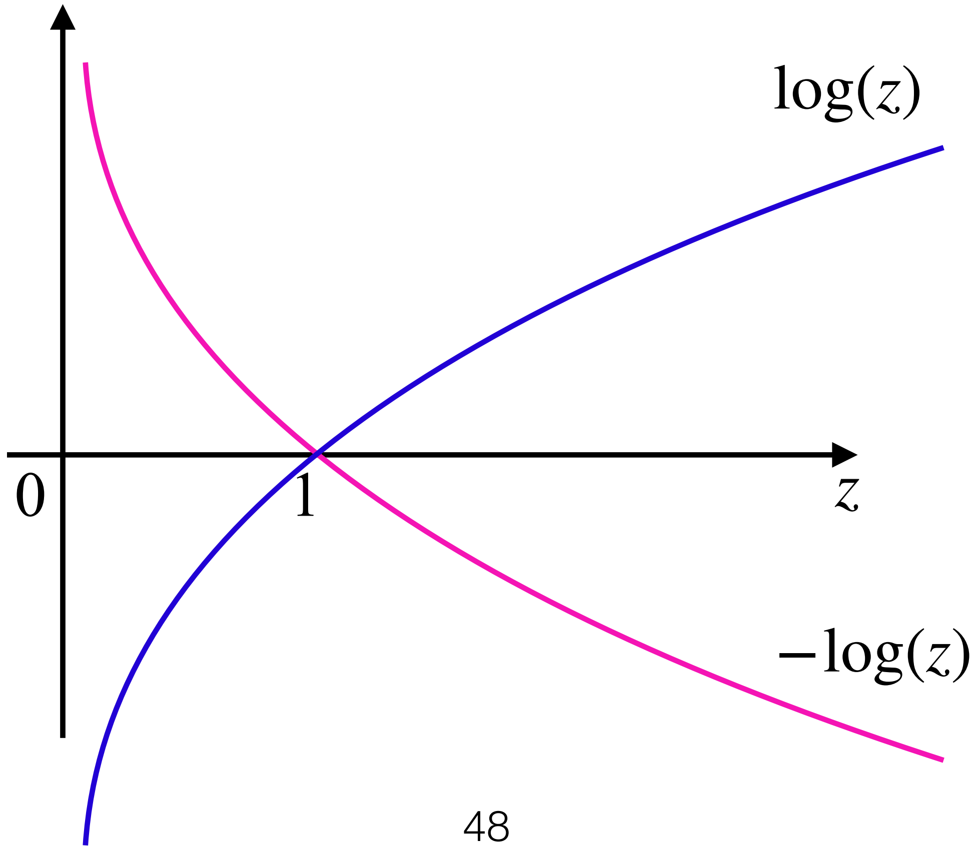
What does the cost error function look like ?

(Should it be $\textbf{cost}(h_\theta(\boldsymbol{x}^{(i)}), y^{(i)})$?)

Cost Function

What we want:

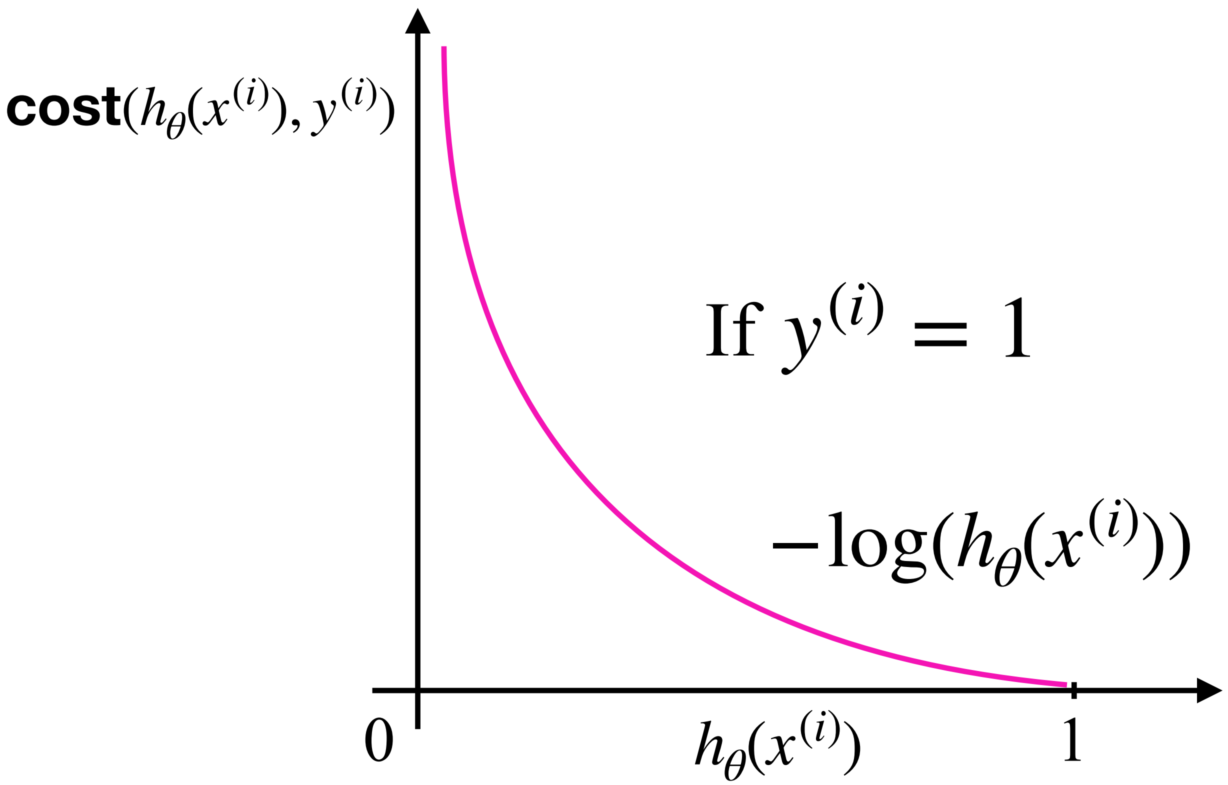
- if $y = 1$ and $h_\theta(\boldsymbol{x}) = 1$, then our cost must be 0.
- if $y = 1$ and $h_\theta(\boldsymbol{x}) = 0$, then our cost should be very huge.
- if $y = 0$ and $h_\theta(\boldsymbol{x}) = 1$, then our cost should be very huge.
- if $y = 0$ and $h_\theta(\boldsymbol{x}) = 0$, then our cost should be 0.



Cost Function

What we want:

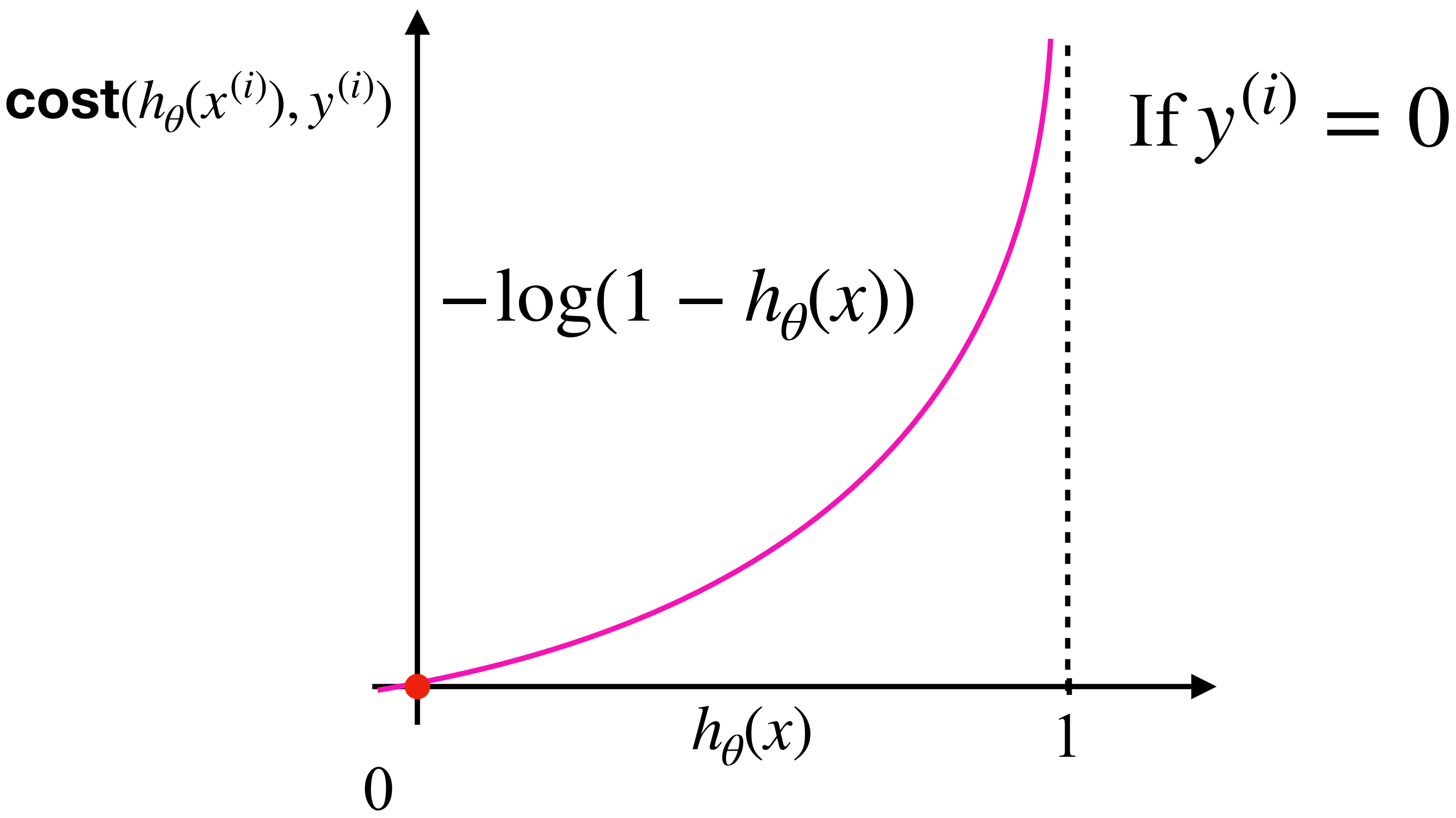
- if $y = 1$ and $h_\theta(\boldsymbol{x}) = 1$, then our cost must be 0.
- if $y = 1$ and $h_\theta(\boldsymbol{x}) = 0$, then our cost should be very huge.



Cost Function

What we want:

- if $y = 0$ and $h_\theta(\boldsymbol{x}) = 1$, then our cost should be very huge.
- if $y = 0$ and $h_\theta(\boldsymbol{x}) = 0$, then our cost should be 0.



Cost Function

$$\textbf{cost}(h_\theta(\boldsymbol{x}^{(i)}), y^{(i)}) = \begin{cases} -\log(h_\theta(\boldsymbol{x}^{(i)})) & \textbf{if } y^{(i)} = 1 \\ -\log(1 - h_\theta(\boldsymbol{x}^{(i)})) & \textbf{if } y^{(i)} = 0 \end{cases}$$

In a compact form, this is rewritten as:

$$\textbf{cost}(h_\theta(\boldsymbol{x}^{(i)}), y^{(i)}) = -\, y^{(i)} \log(h_\theta(\boldsymbol{x}^{(i)})) - (1 - y^{(i)})\log(1 - h_\theta(\boldsymbol{x}^{(i)}))$$

Cost Function

Since

$$\textbf{cost}(h_\theta(\boldsymbol{x}^{(i)}), y^{(i)}) = -y^{(i)}\log(h_\theta(\boldsymbol{x}^{(i)})) - (1 - y^{(i)})\log(1 - h_\theta(\boldsymbol{x}^{(i)}))$$

Therefore

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

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

Remind that $J(\theta)$ can be derived from 'maximum likelihood estimation'

Things we have

Cost Function:

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

Goal: $\min_{\theta} J(\theta)$

To make a prediction given new $\boldsymbol{x}$:

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

i.e. the probability that $y = 1$ given that $\boldsymbol{x}$ parameterized by $\boldsymbol{\theta}$

Things we have

Goal: $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

How to minimize it with gradient descent ?

Repeat until converged {

$$\theta_j := \theta_j - \alpha \boxed{\left[\frac{1}{m}\Sigma_{i=1}^{m}(h_\theta(\boldsymbol{x}^{(i)}) - y^{(i)})\boldsymbol{x}_j^{(i)}\right]} \quad \text{(simultaneously update all } \theta_j \text{)}$$

}

$\nabla_J(\boldsymbol{\theta})$

Question

- Suppose you are running gradient descent to fit a logistic regression model with parameter $\theta \in \mathbb{R}^{n+1}$. Which of the following is a reasonable way to make sure the learning rate α is set properly and that gradient descent is running correctly?

(i) Plot $J(\theta) = (1/m)\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$ as a function of the number of iterations (*i.e.* the horizontal axis is the iteration number) and make $J(\theta)$ is decreasing on every iteration.

(ii) Plot $J(\theta) = (-1/m)\Sigma_{i=1}^{m}[y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))]$ as a function of the number of iterations and make sure $J(\theta)$ is decreasing on every iteration.

(iii) Plot $J(\theta)$ as a function of θ and make sure it is decreasing on every iteration.

(iv) Plot $J(\theta)$ as a function of θ and make sure it is convex.

Multiclass Classification

Multiclass (Intuition)

- **Email foldering / tagging:** Work, Friends, Family, Hobby

  $y = 1$ $y = 2$ $y = 3$ $y = 4$

- **Medical diagnosis:** Not ill, Cold, Flu

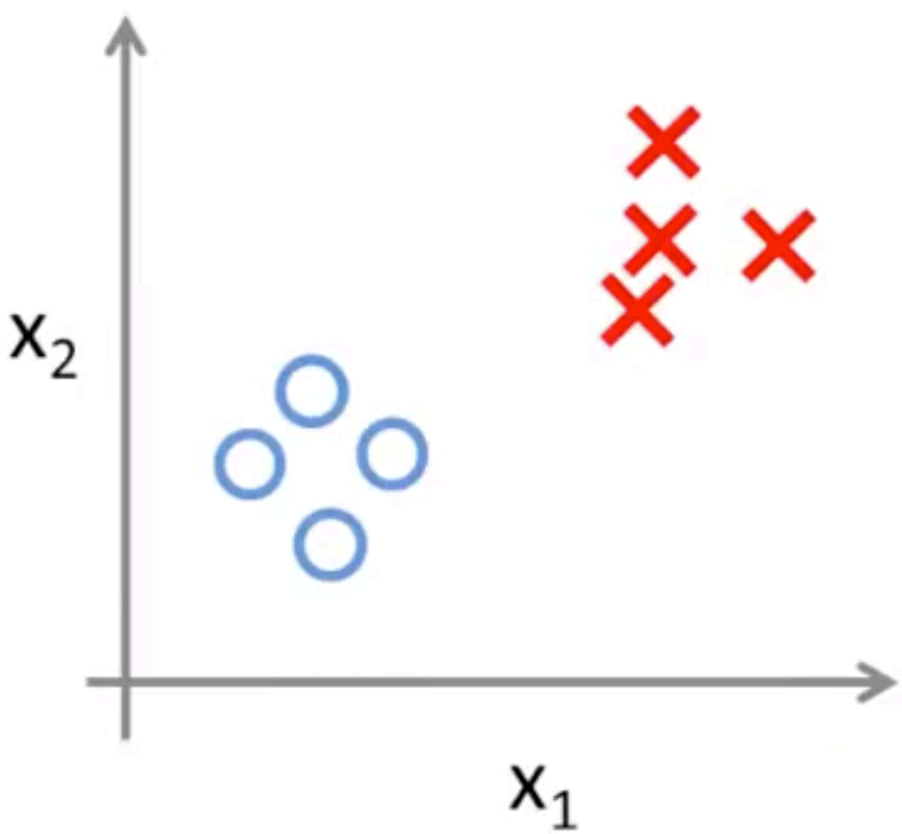
  $y = 1$ $y = 2$ $y = 3$

- **Weather:** Sunny, Cloudy, Rain, Snow

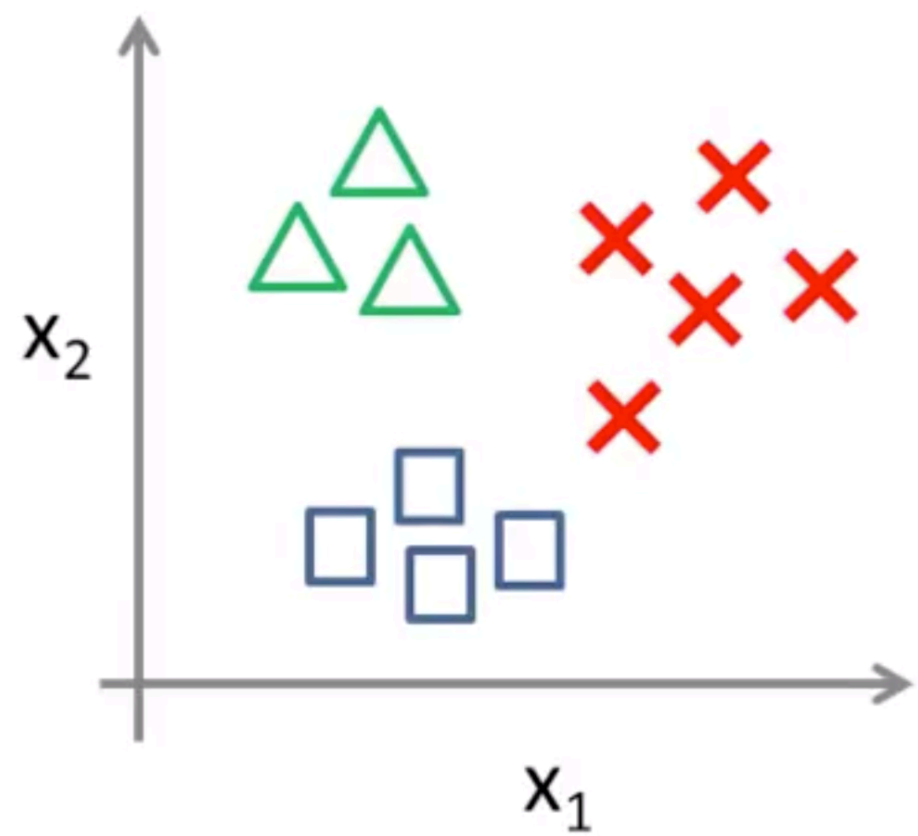
  $y = 1$ $y = 2$ $y = 3$ $y = 4$

Binary *vs.* Multiclass

Binary classification

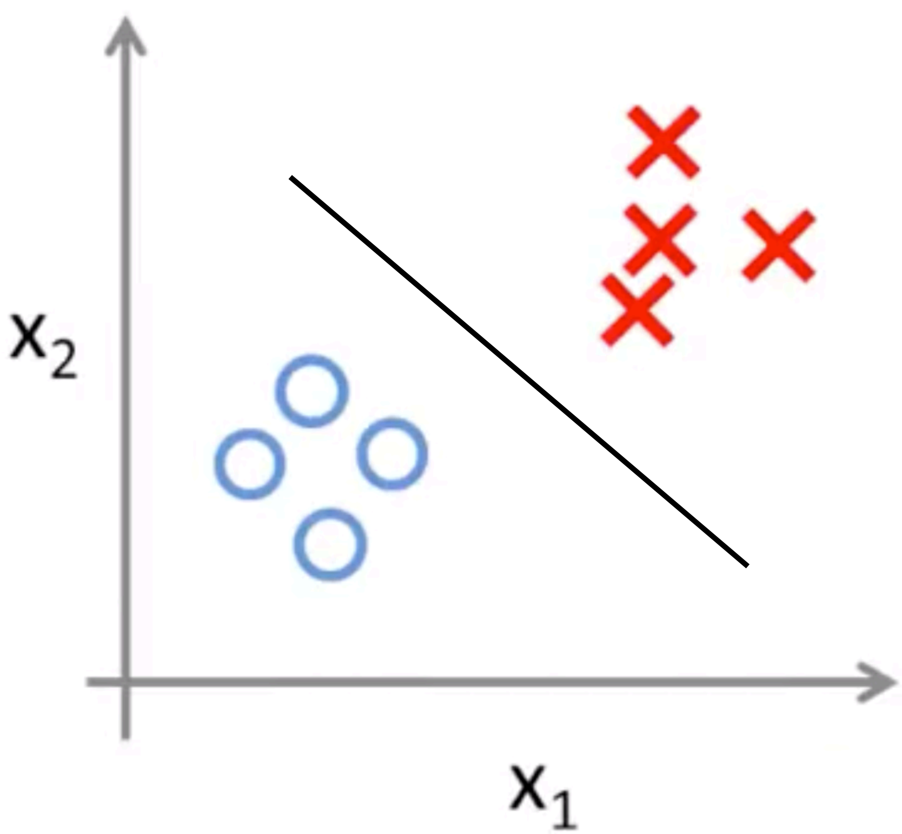


Multiclass classification



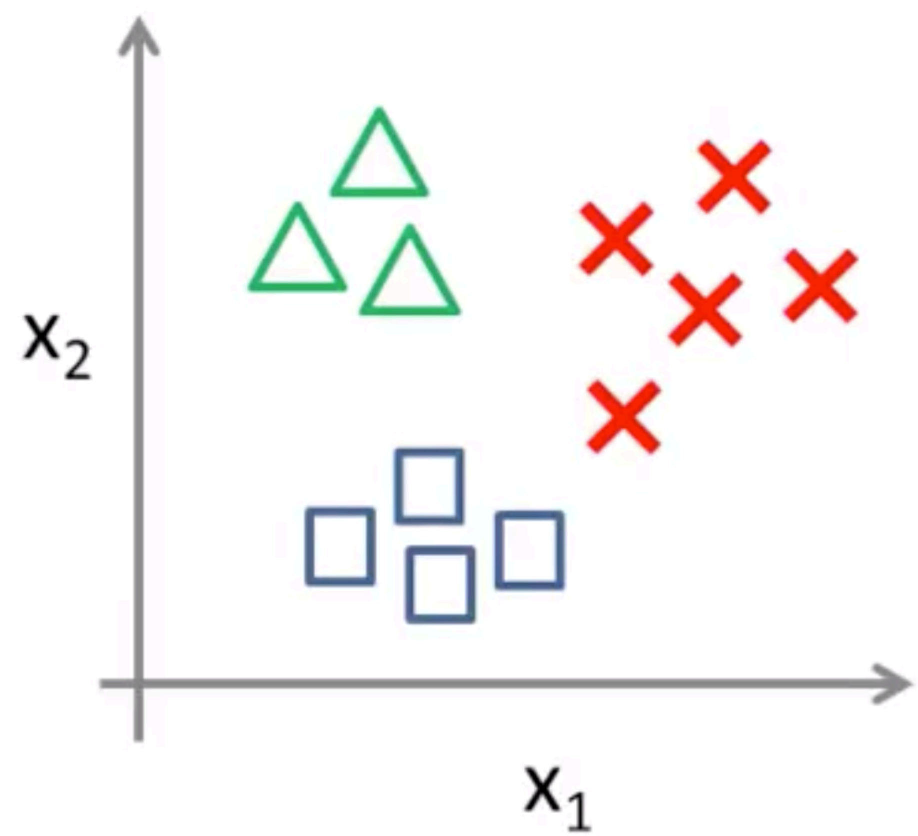
Binary *vs.* Multiclass

Binary classification



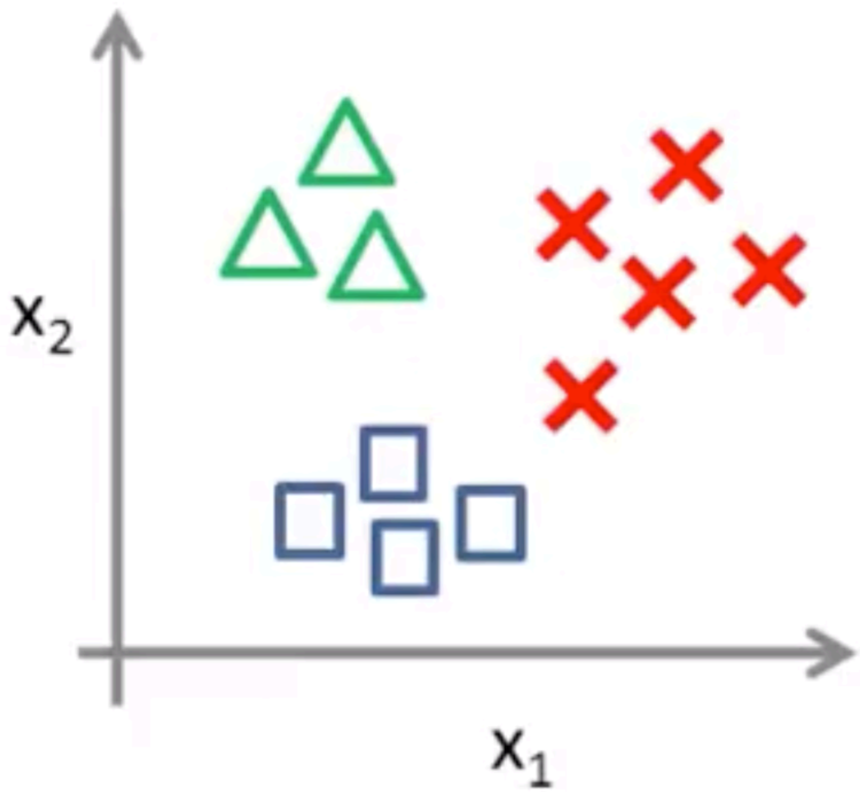
Regression algorithm !

Multiclass classification



One-vs-all algorithm !

One-vs-all (One-vs-rest)

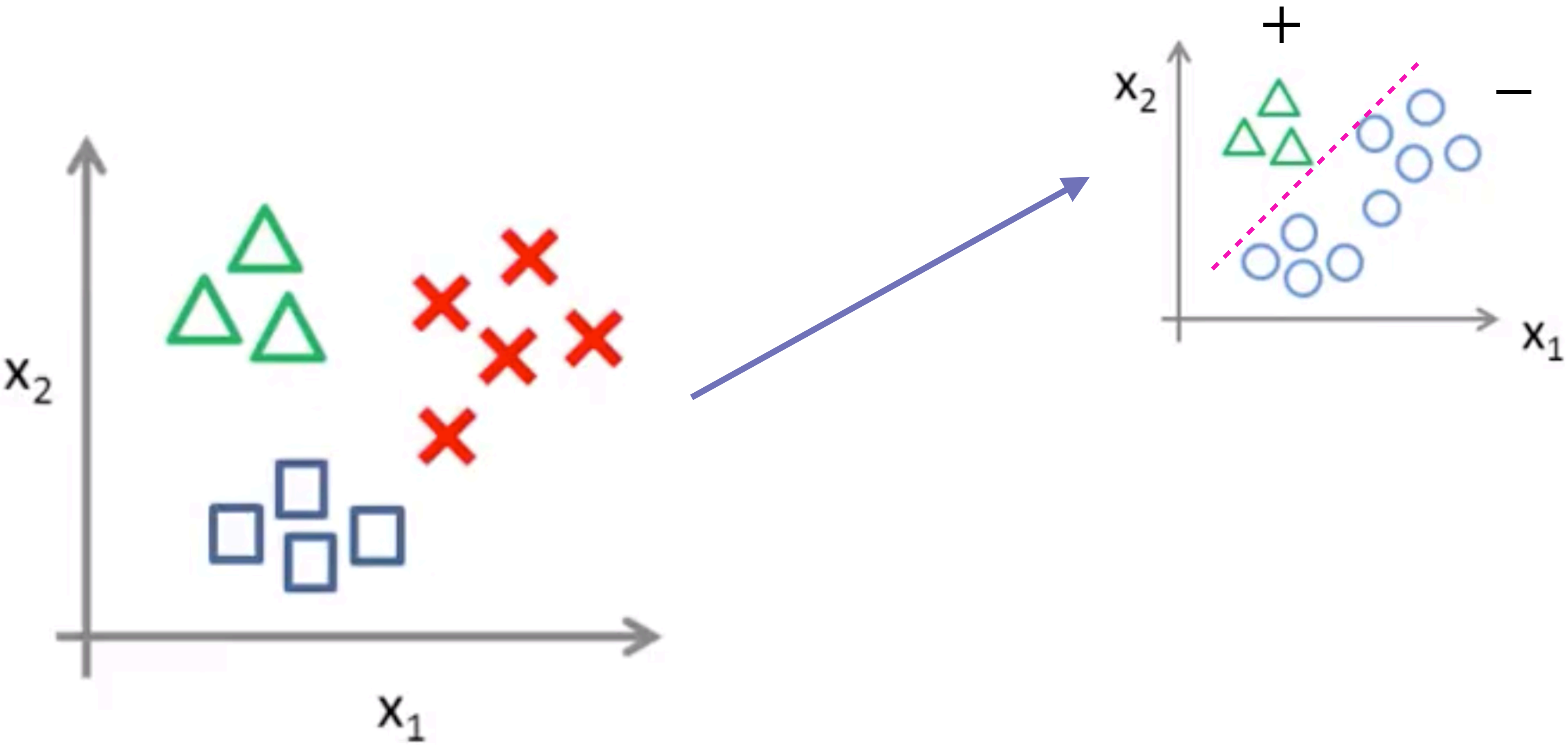


Class 1: △ $(y = 1)$

Class 2: □ $(y = 2)$

Class 3: ✕ $(y = 3)$

One-vs-all (One-vs-rest)



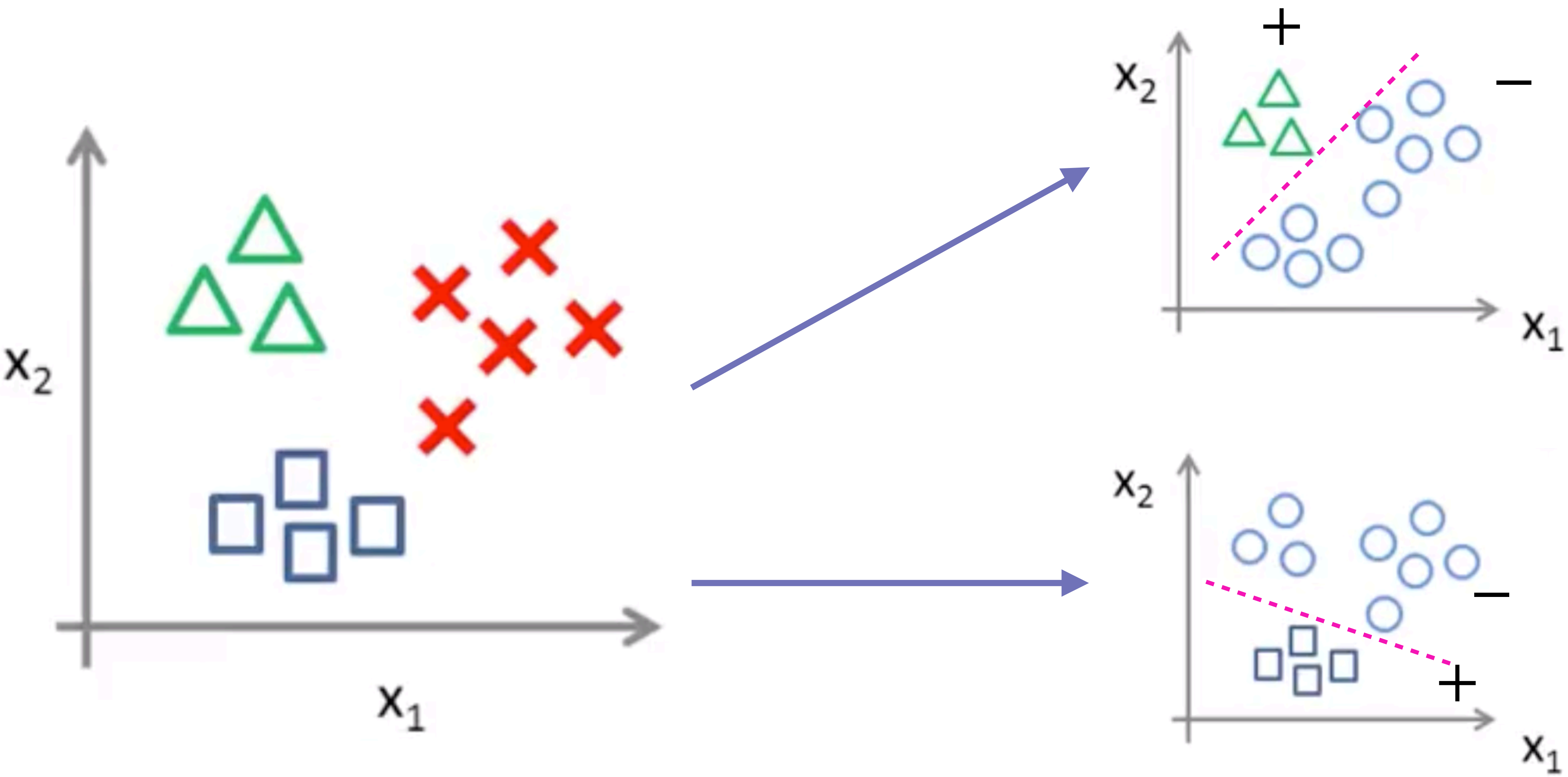
$h_\theta^{(1)}(\boldsymbol{x})$

Class 1: △ $(y = 1)$

Class 2: □ $(y = 2)$

Class 3: ✕ $(y = 3)$

One-vs-all (One-vs-rest)



$h_\theta^{(1)}(\boldsymbol{x})$

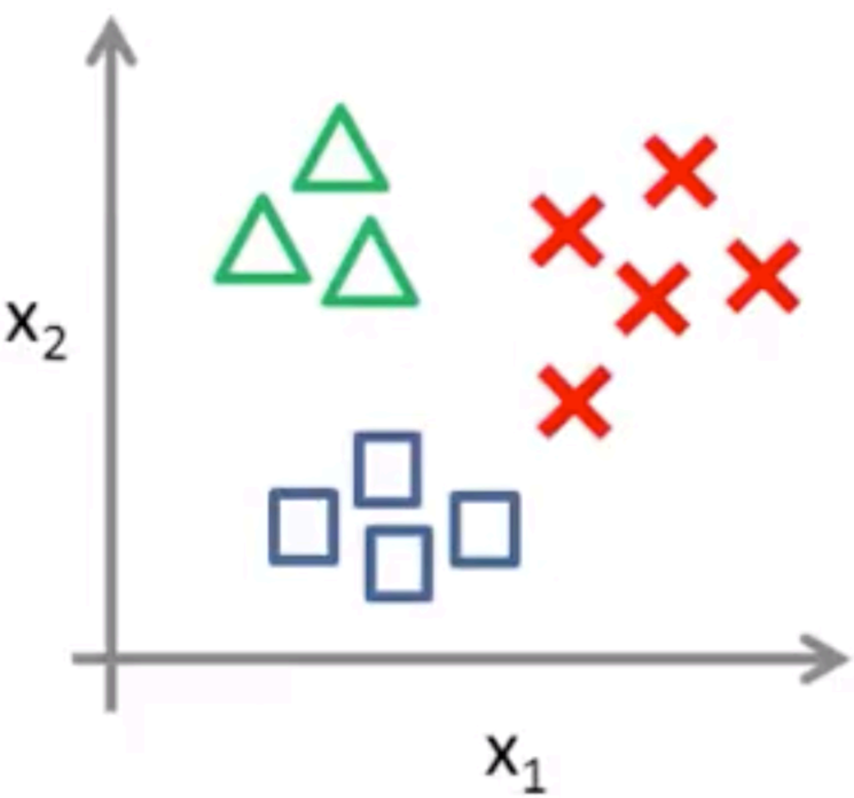
$h_\theta^{(2)}(\boldsymbol{x})$

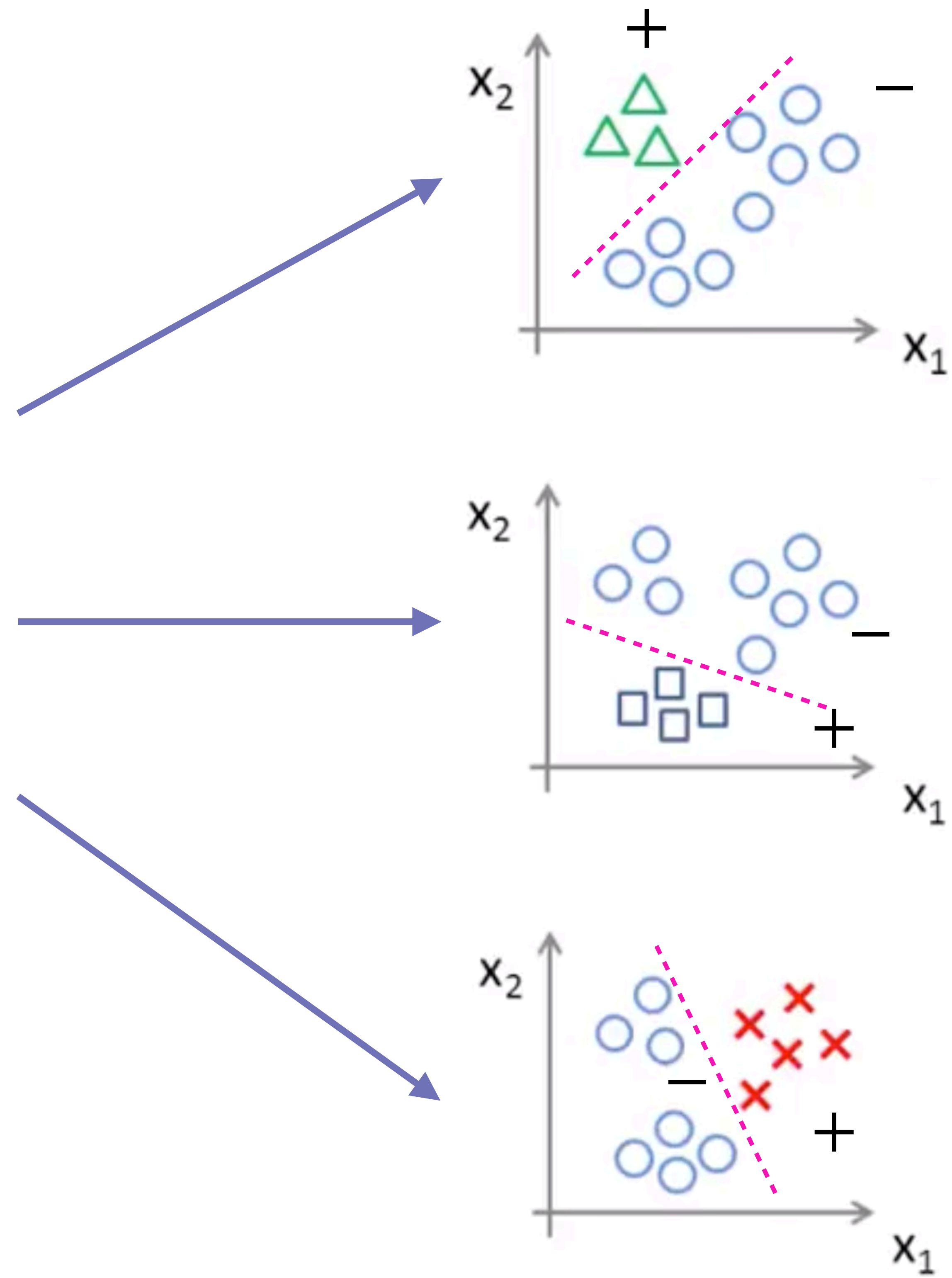
Class 1: △ $(y = 1)$

Class 2: □ $(y = 2)$

Class 3: ✕ $(y = 3)$

One-vs-all (One-vs-rest)





$h_\theta^{(1)}(\boldsymbol{x})$

$h_\theta^{(2)}(\boldsymbol{x})$

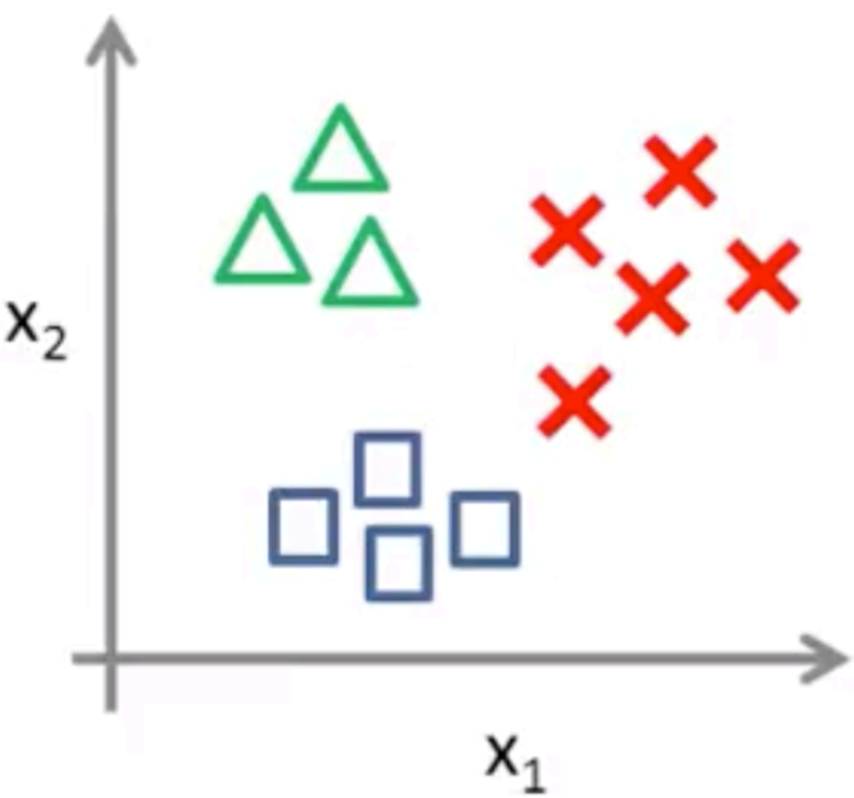
$h_\theta^{(3)}(\boldsymbol{x})$

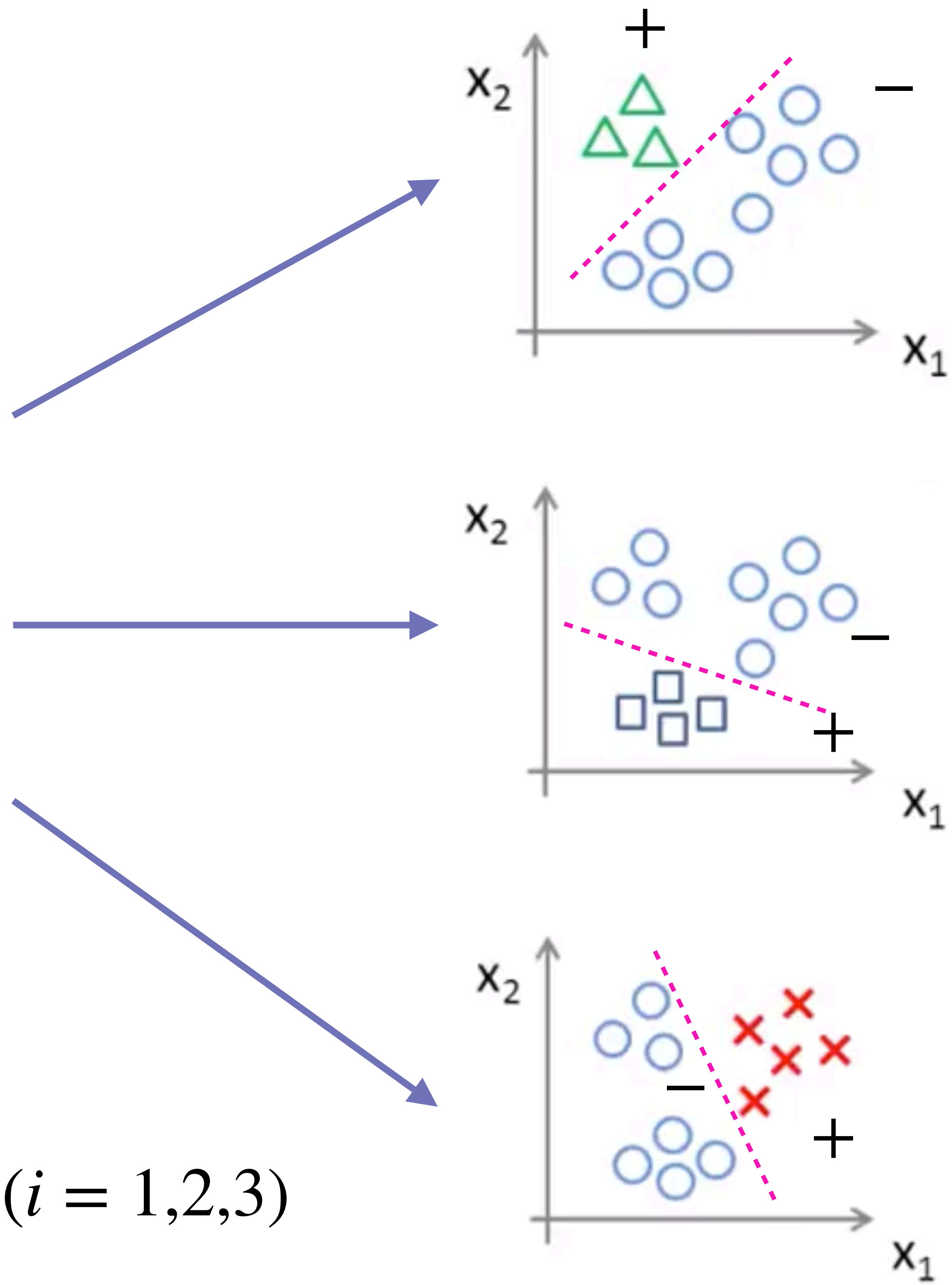
Class 1: △ $(y = 1)$

Class 2: □ $(y = 2)$

Class 3: ✕ $(y = 3)$

One-vs-all (One-vs-rest)





$h_\theta^{(1)}(\boldsymbol{x})$

$h_\theta^{(2)}(\boldsymbol{x})$

$h_\theta^{(3)}(\boldsymbol{x})$

Class 1: △ $(y = 1)$

Class 2: □ $(y = 2)$

Class 3: ✕ $(y = 3)$

$h_\theta^{(i)}(x) = P(y = i \mid x; \theta) \qquad (i = 1,2,3)$

One-vs-all

Idea:

- Train a logistic regression classifier $h_\theta^{(i)}(\boldsymbol{x})$ for each class *i* to predict the probability that $y = i$.
- To make a prediction on a new input $\boldsymbol{x}$, pick a class *i* that maximizes $h_\theta^{(i)}(\boldsymbol{x})$ *i.e.*

$$y = \arg\max_i h_\theta^{(i)}(\boldsymbol{x})$$

Question

- Suppose you have a multi-class classification problem with k classes (so $y \in \{1,2,\ldots,k\}$). Using the one-vs-all method, how many different logistic regression classifiers will you end up training?

(i) $k - 1$
(ii) k
(iii) $k + 1$
(iv) Approximately $\log_2(k)$

Summary (Part 2)

Last reminder !

1. If you have continuous $\mathscr{X}$ and want to predict continuous $\mathscr{Y}$, then your first go-to model is linear regression !
   - You may also consider non-linear transformation.
2. If you have continuous $\mathscr{X}$ and want to predict discrete $\mathscr{Y}$, then your first go-to model is logistic regression !

Last note !

So far, the methods we have tried are to learn $p(y\,|\,\boldsymbol{x})$ directly
i.e. given an input $\boldsymbol{x}$, we map directly to the target y.
These methods are called discriminative learning algorithms.