D1EAD – Análise Estatística para Ciência de Dados

Logistic Regression

Lecture based on
Machine Learning course
by Andrew Ng
(Coursera)

Prof. Samuel Martins (Samuka)
samuel.martins@ifsp.edu.br

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
São Paulo
Campus Campinas

Binary Classification

Admission Exam: Pass / Fail

Email: Spam / Non-spam

Tumor: Malignant / Benign

Signature: Real / False

App: Free / Paid

....

Binary Classification

Admission Exam: Pass / Fail

Email: Spam / Non-spam

Tumor: Malignant / Benign

Signature: Real / False

App: Free / Paid

....

→ **Categorical variables with only 2 values**

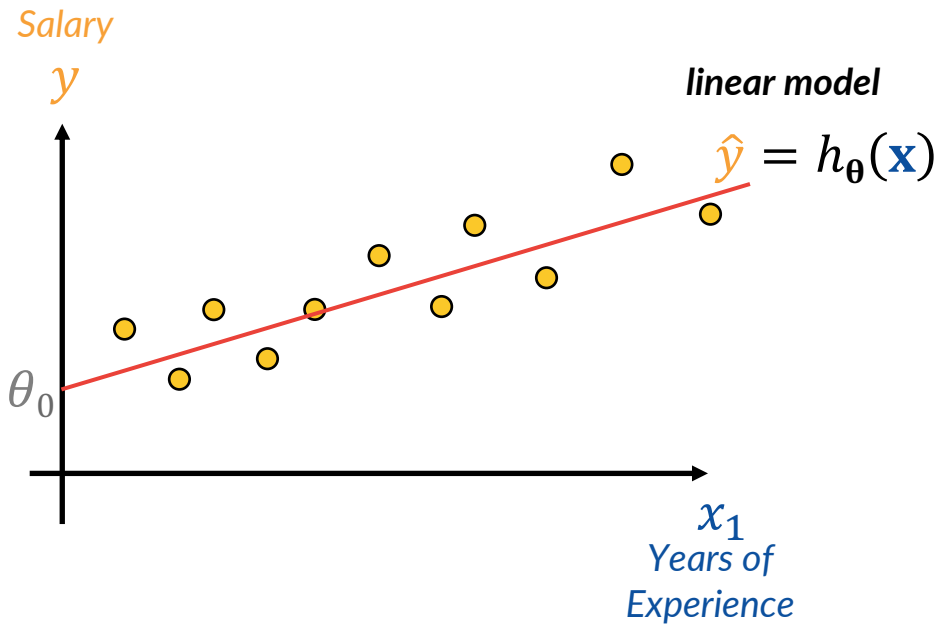
$$y \in \{0, 1\}$$

0: Negative class

1: Positive class

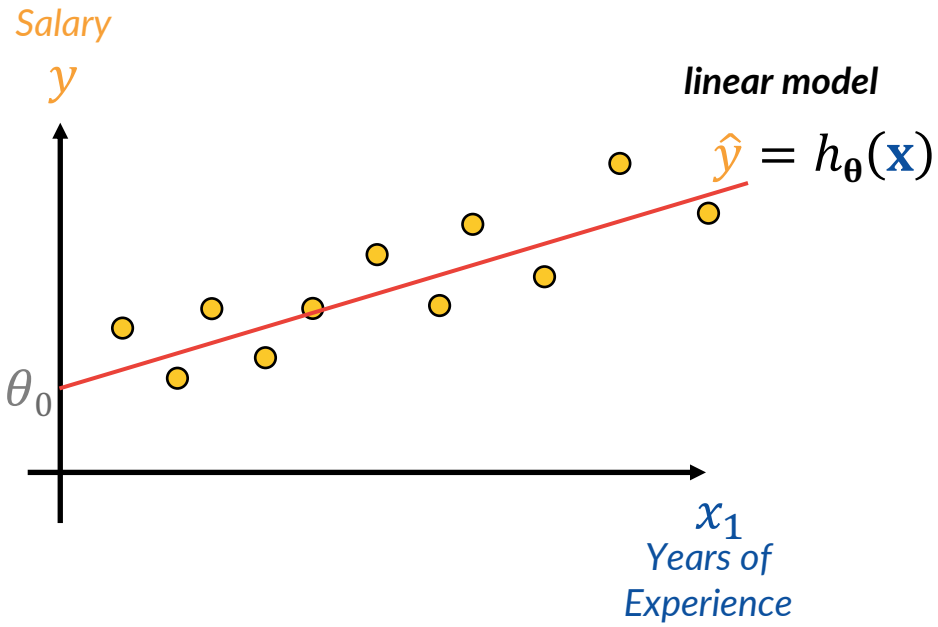
Linear Regression

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \theta_1 * x_1 + \cdots + \theta_n * x_n$$



Linear Regression

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \theta_1 * x_1 + \cdots + \theta_n * x_n$$



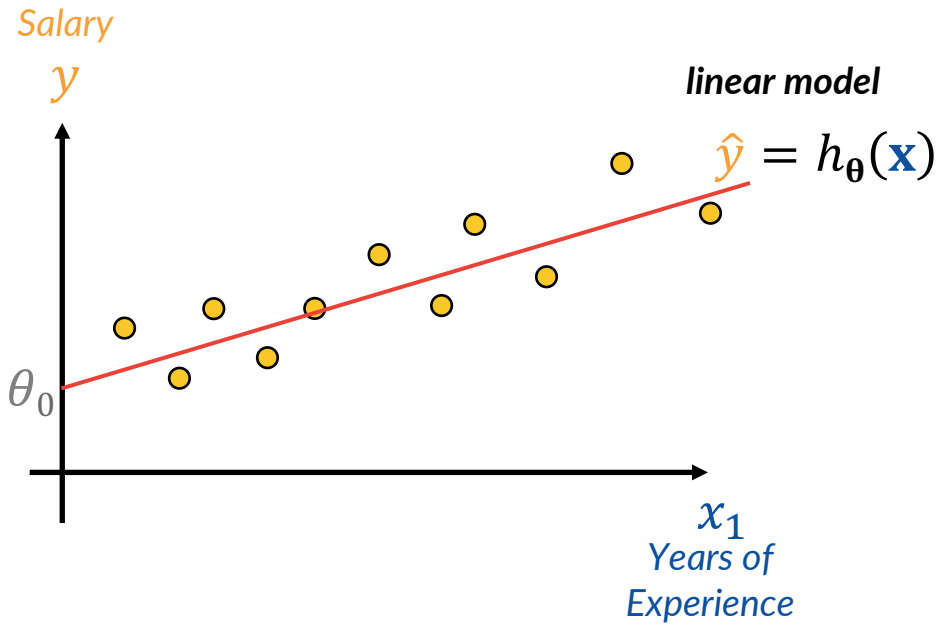
In Machine Learning, **vectors** are typically represented as **column vectors**.

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \cdots \\ \theta_n \end{bmatrix}_{n\times 1} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_1 \\ \cdots \\ x_n \end{bmatrix}_{n\times 1}$$

Linear Regression

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \theta_1 * x_1 + \cdots + \theta_n * x_n$$

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}$$



In Machine Learning, **vectors** are typically represented as **column vectors**.

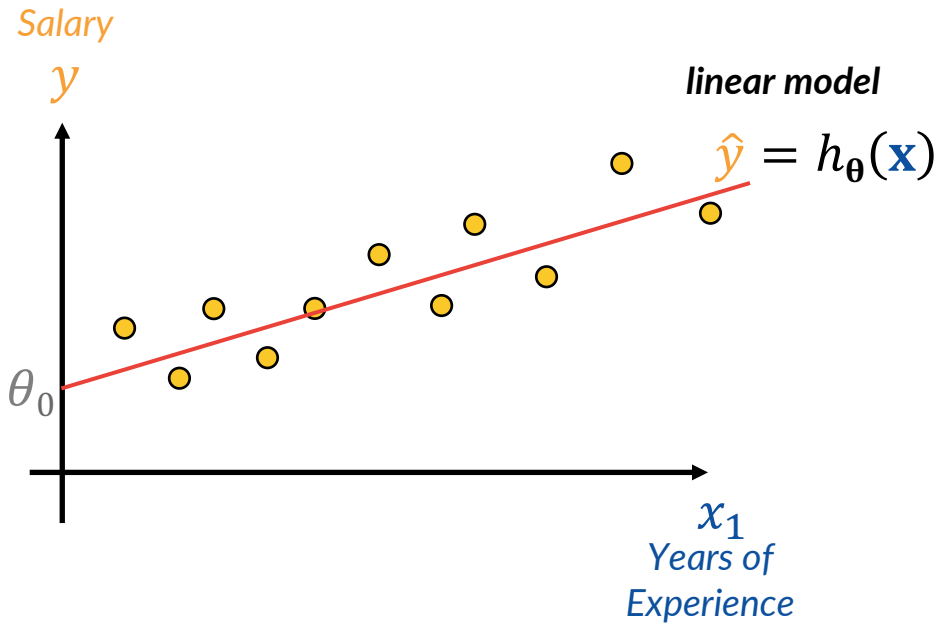
$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \cdots \\ \theta_n \end{bmatrix}_{n\times 1} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}_{n\times 1}$$

$$\boldsymbol{\theta}^{\mathbf{T}} = [\theta_1\ \theta_2 \cdots \theta_n]_{1\times n}$$

Linear Regression: Bias Trick

$$\hat{y} = h_{\mathbf{\theta}}(\mathbf{x}) = \theta_0 + \theta_1 * x_1 + \cdots + \theta_n * x_n$$

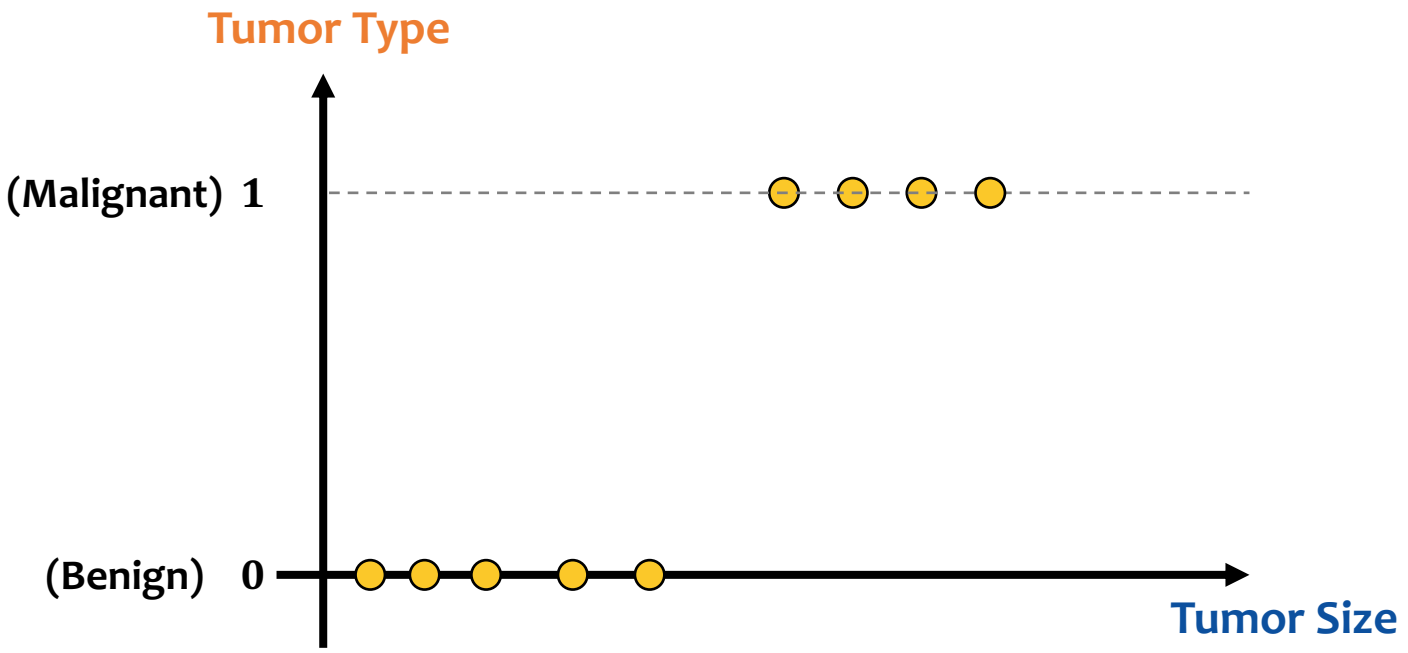
$$\hat{y} = h_{\mathbf{\theta}}(\mathbf{x}) = \mathbf{\theta^T} * \mathbf{x}$$



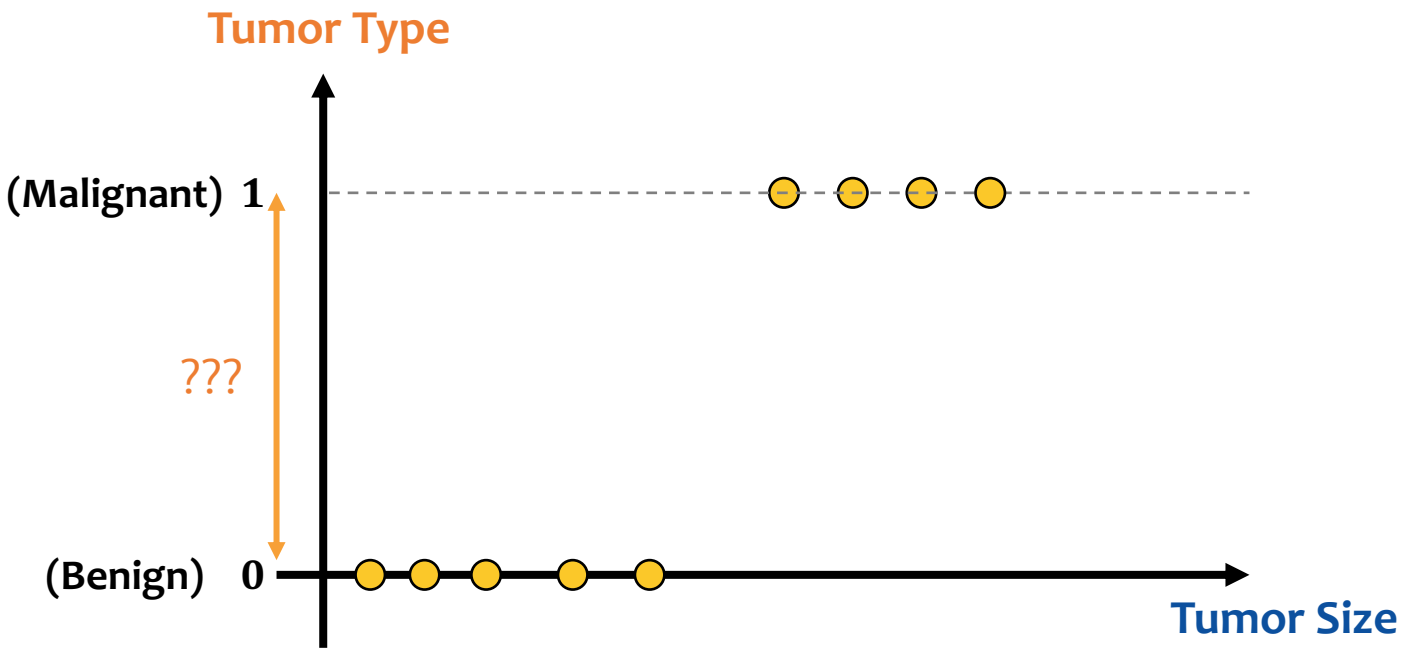
$$\mathbf{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \cdots \\ \theta_n \end{bmatrix}_{(n+1)\times 1} \qquad \mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}_{(n+1)\times 1}$$

$$\mathbf{\theta^T} = [\theta_0 \ \theta_1 \ \theta_2 \cdots \theta_n]_{1\times(n+1)}$$

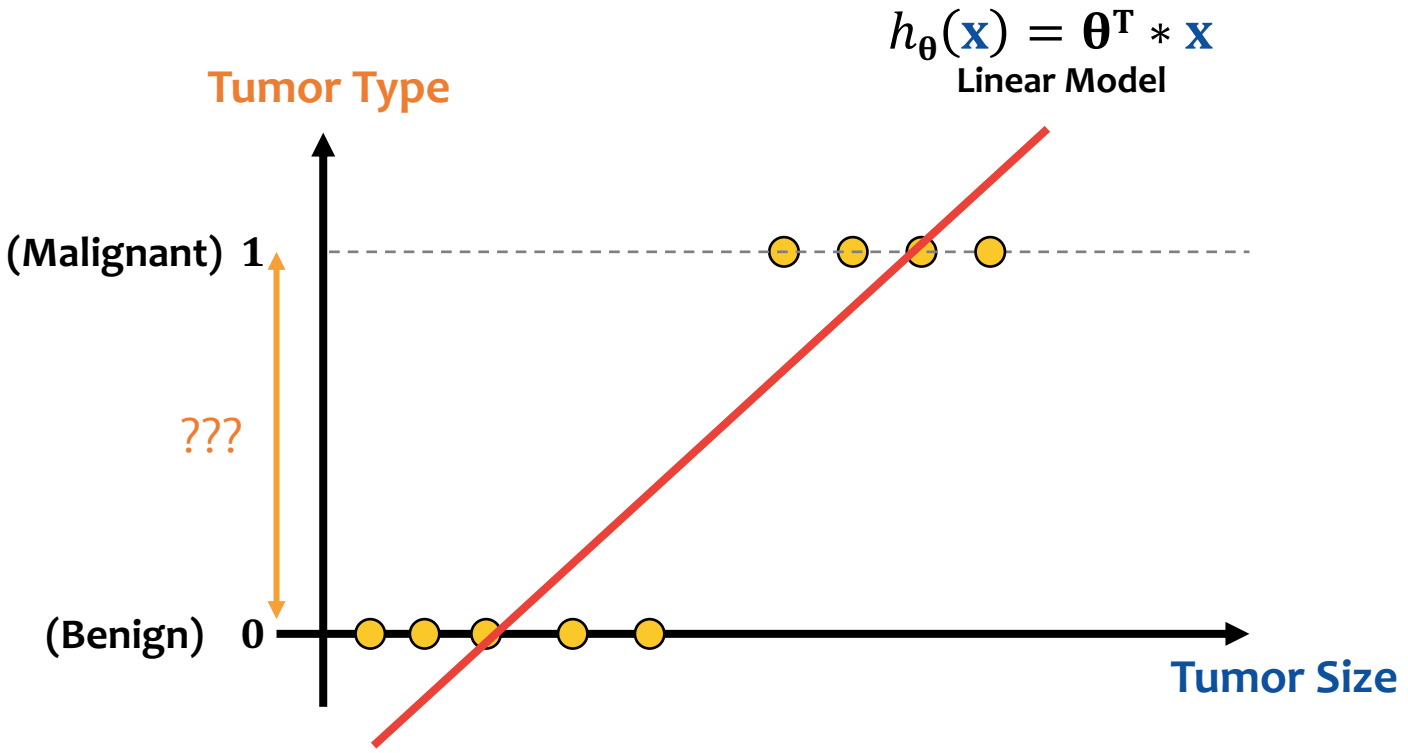
Binary Classification Problem



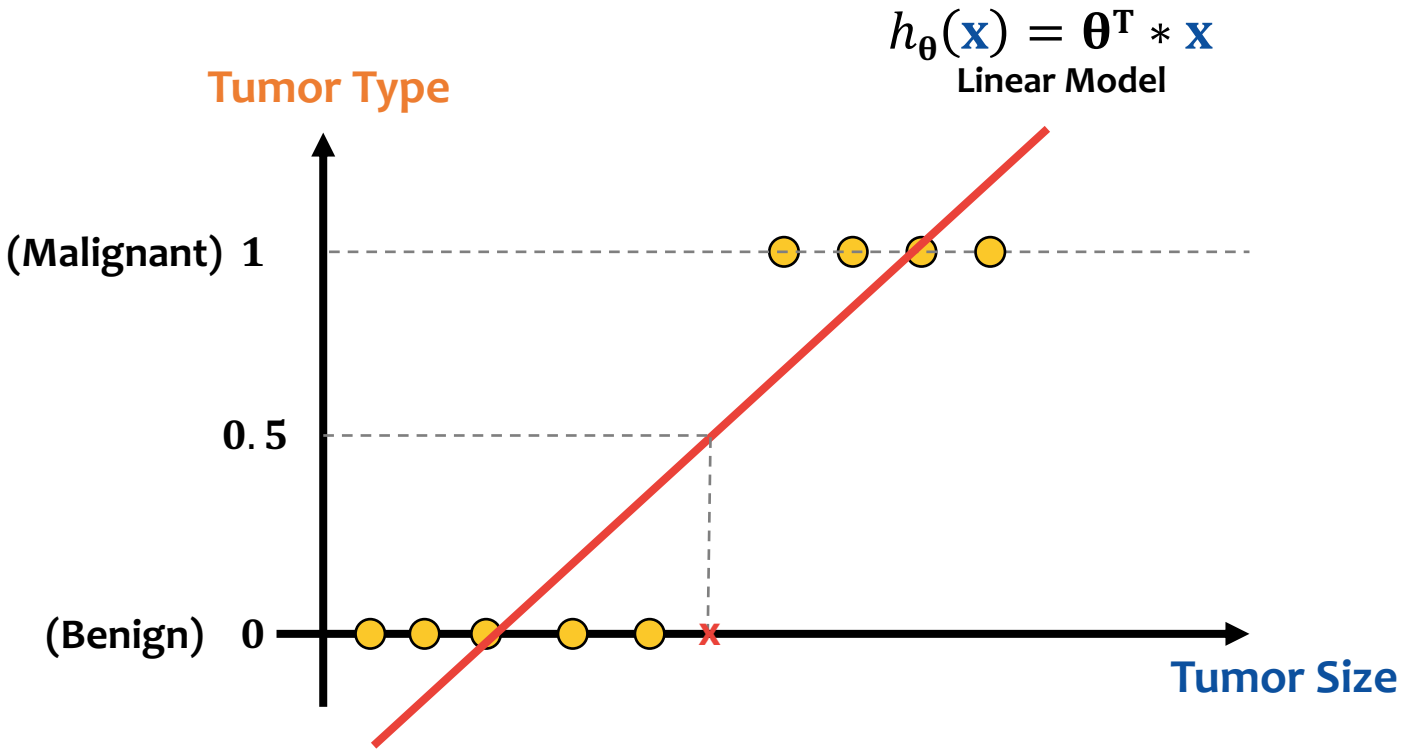
Binary Classification Problem



Binary Classification Problem



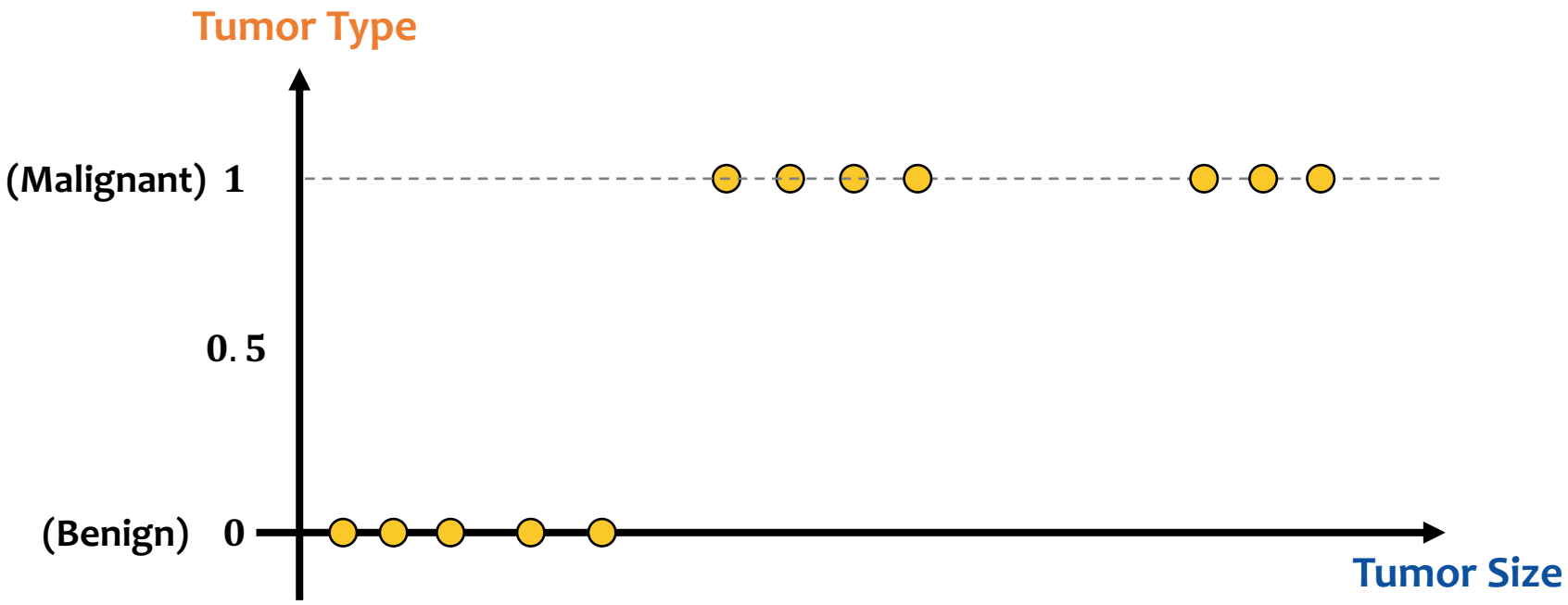
Binary Classification Problem



If $h_\theta(x) \geq 0.5$, classify as $\hat{y} = 1$

If $h_\theta(x) < 0$, classify as $\hat{y} = 0$

Binary Classification Problem



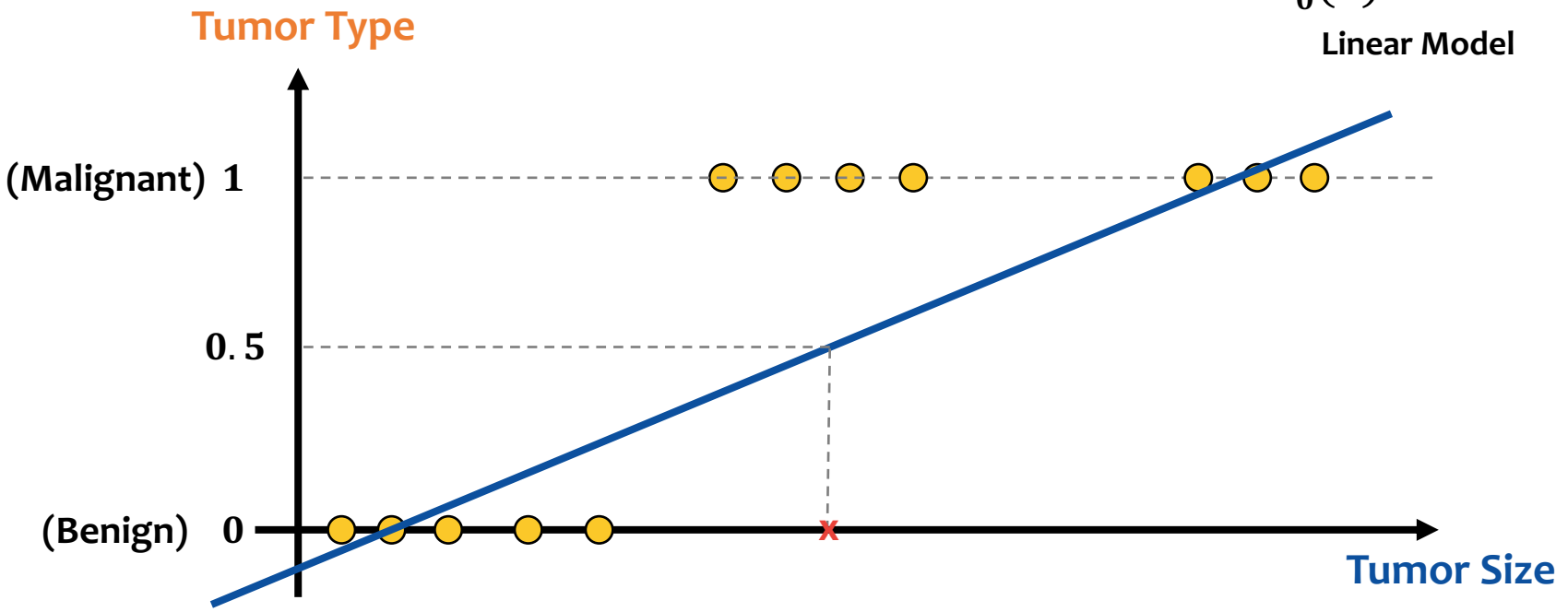
If $h_{\theta}(\mathbf{x}) \geq 0.5$, classify as $\hat{y} = 1$

If $h_{\theta}(\mathbf{x}) < 0$, classify as $\hat{y} = 0$

Binary Classification Problem

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

Linear Model



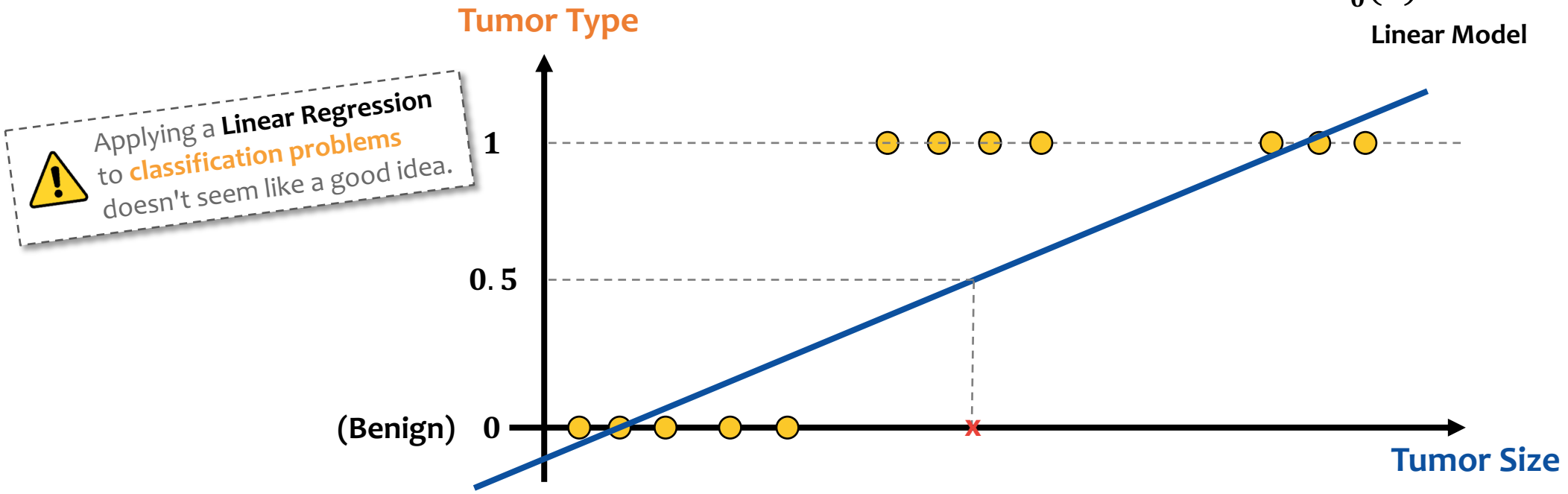
If $h_{\theta}(x) \geq 0.5$, classify as $\hat{y} = 1$

If $h_{\theta}(x) < 0$, classify as $\hat{y} = 0$

Binary Classification Problem

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

Linear Model



If $h_{\theta}(\mathbf{x}) \geq \mathbf{0.5}$, classify as $\hat{y} = 1$

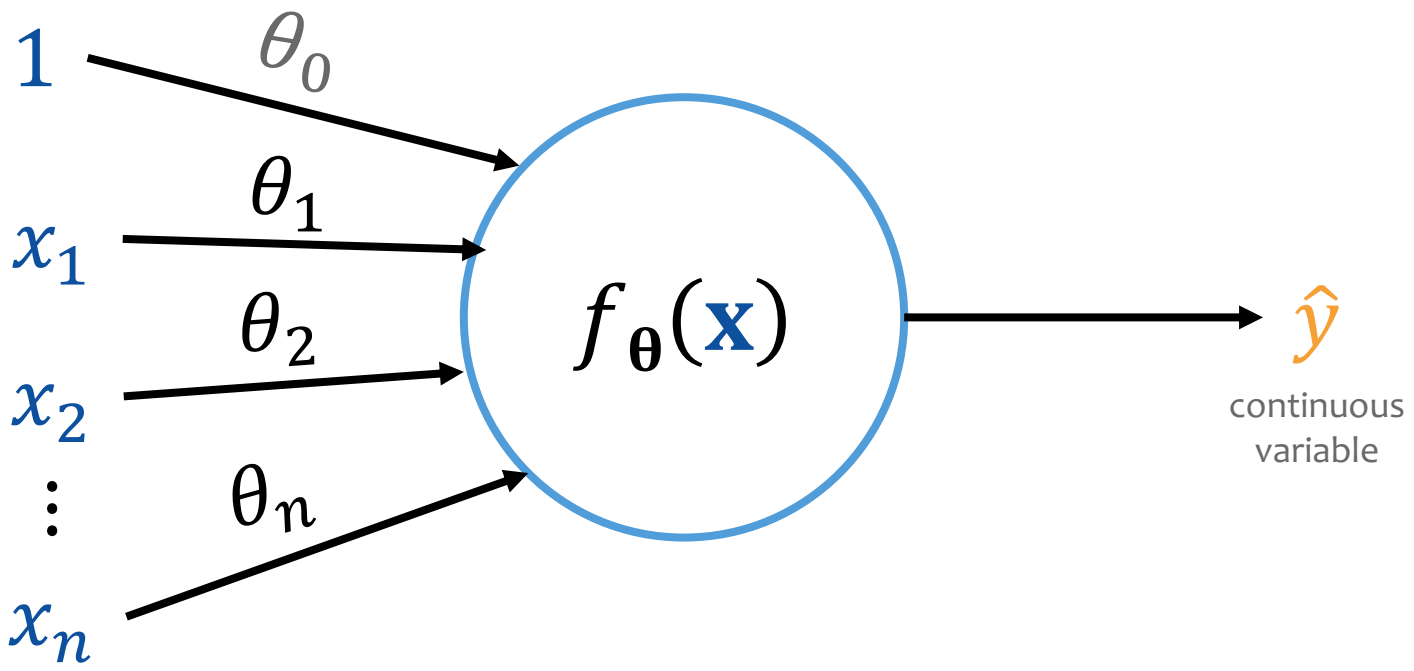
If $h_{\theta}(\mathbf{x}) < \mathbf{0}$, classify as $\hat{y} = 0$

From Linear Regression to Binary Linear Classification

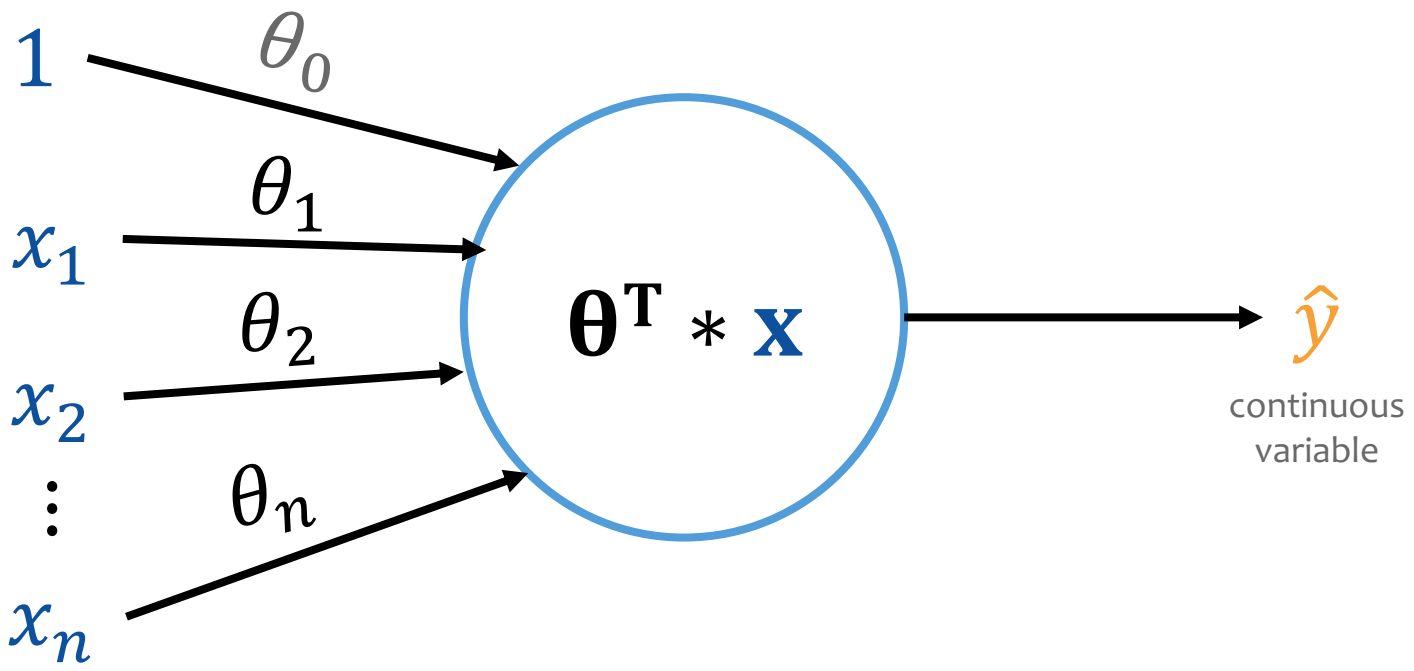
An Alternative Visualization for Linear Regression

Let's first rename our **Linear Regression model** $h_{\boldsymbol{\theta}}(\mathbf{x})$ to $f_{\boldsymbol{\theta}}(\mathbf{x})$.

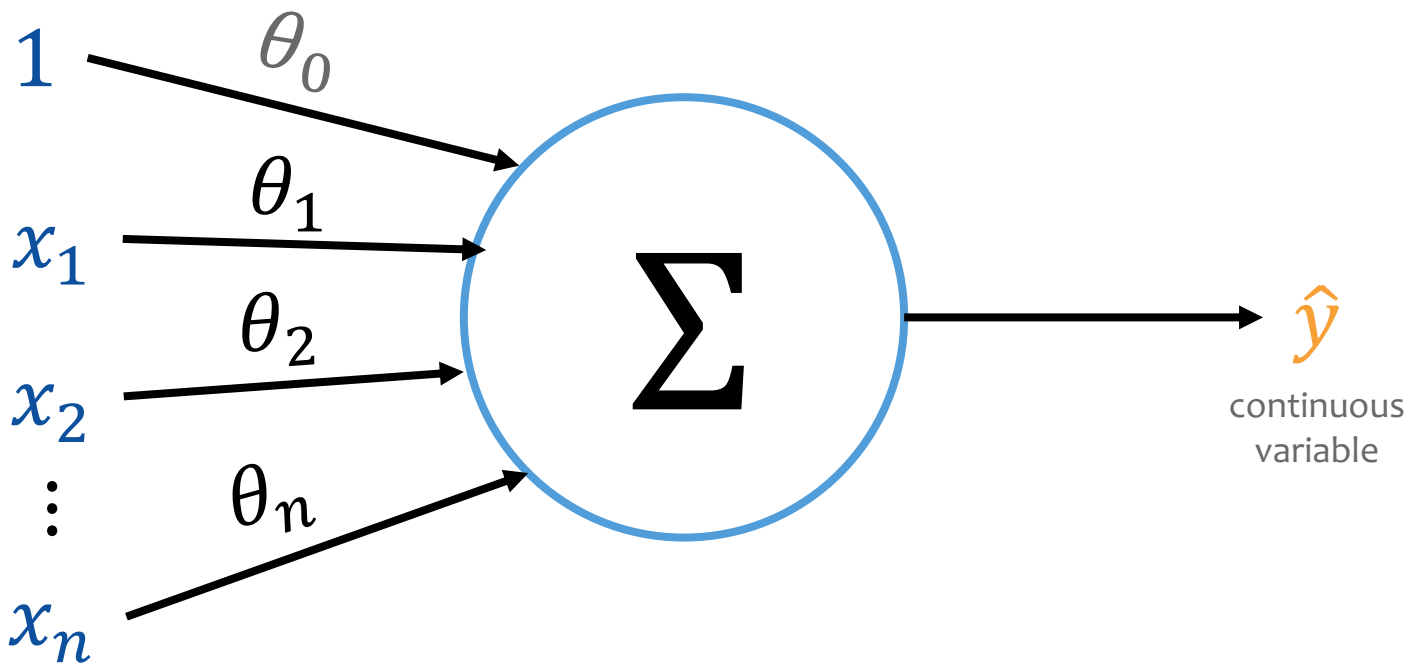
An Alternative Visualization for Linear Regression



An Alternative Visualization for Linear Regression

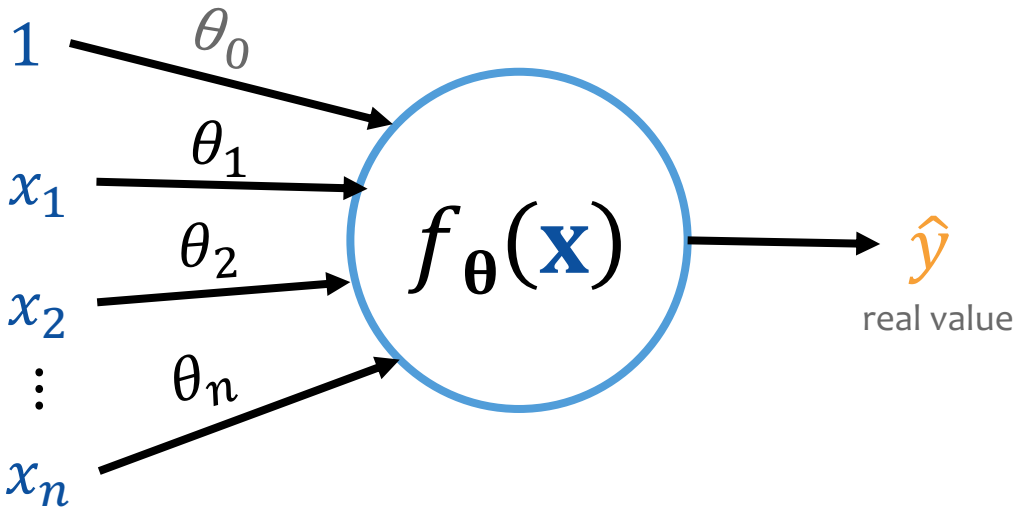


An Alternative Visualization for Linear Regression



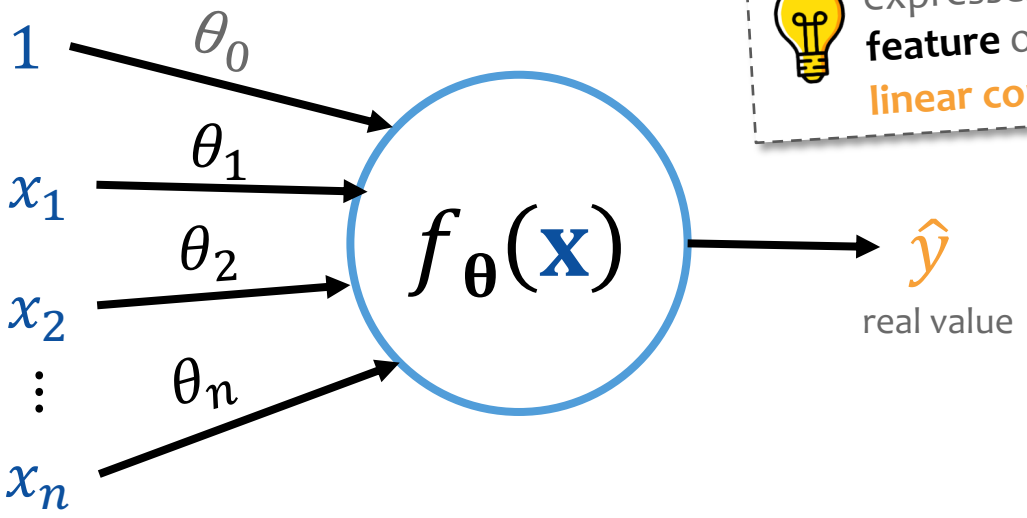
From Linear Regression to Bin. Linear Classification

Linear Regression



From Linear Regression to Bin. Linear Classification

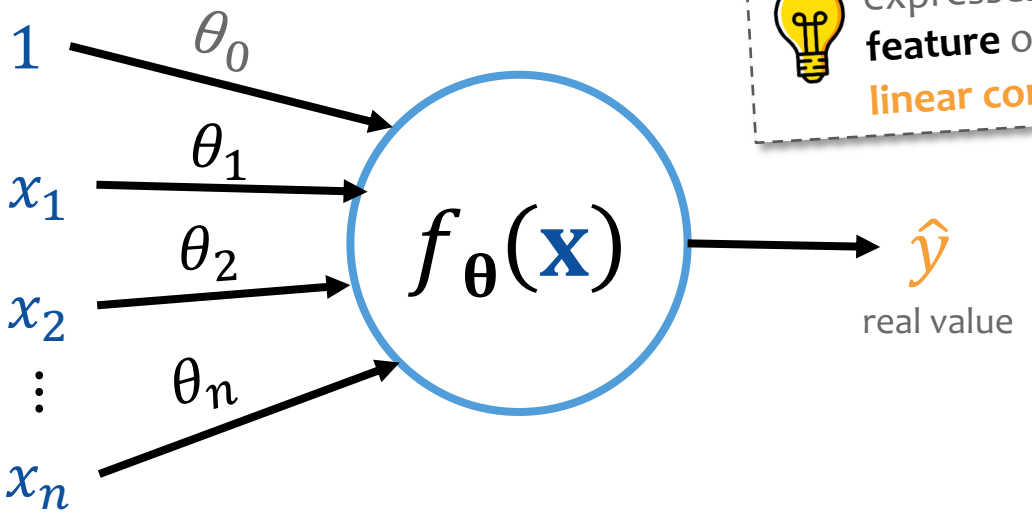
Linear Regression



This **real value** (**weighted sum**) expresses the **contribution of each feature** of the problem, according to a **linear combination**, for a given sample **x**.

From Linear Regression to Bin. Linear Classification

Linear Regression

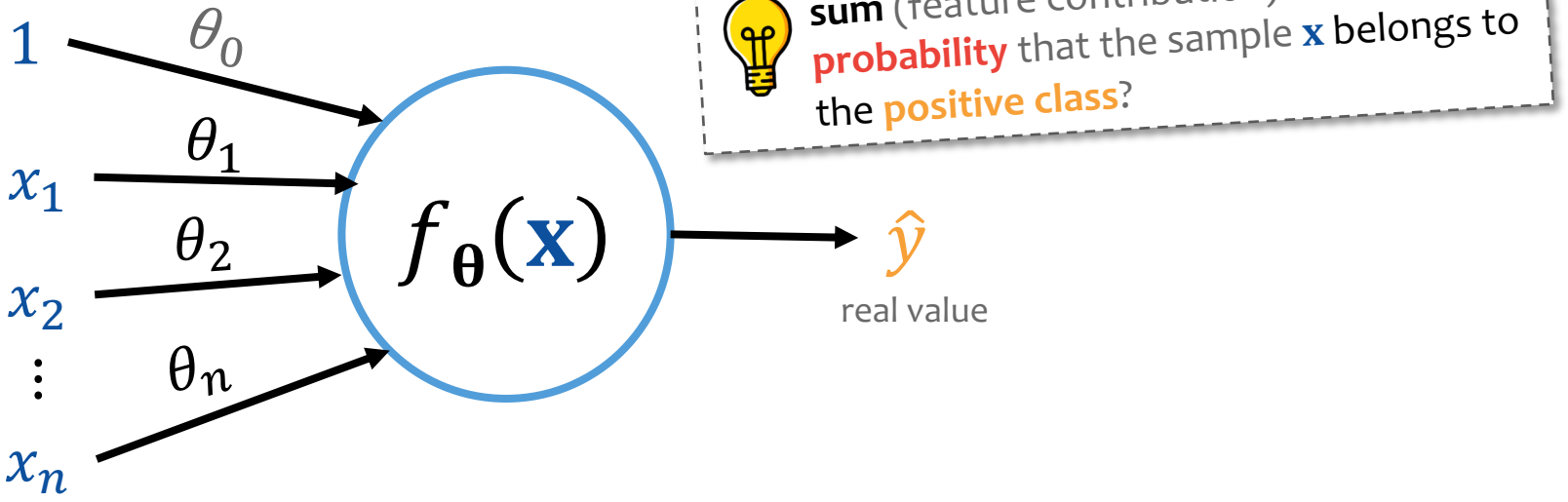


This **real value** (**weighted sum**) expresses the **contribution of each feature** of the problem, according to a **linear combination**, for a given sample **x**.

However, as we've seen previously, this **weighted sum** doesn't express a **class value** for a **binary classification problem**.

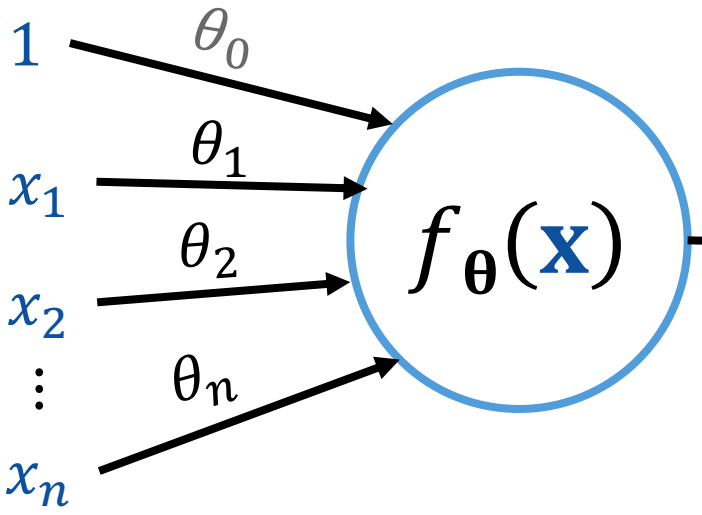
From Linear Regression to Bin. Linear Classification

Linear Regression



From Linear Regression to Bin. Linear Classification

Linear Regression



$\hat{y}$

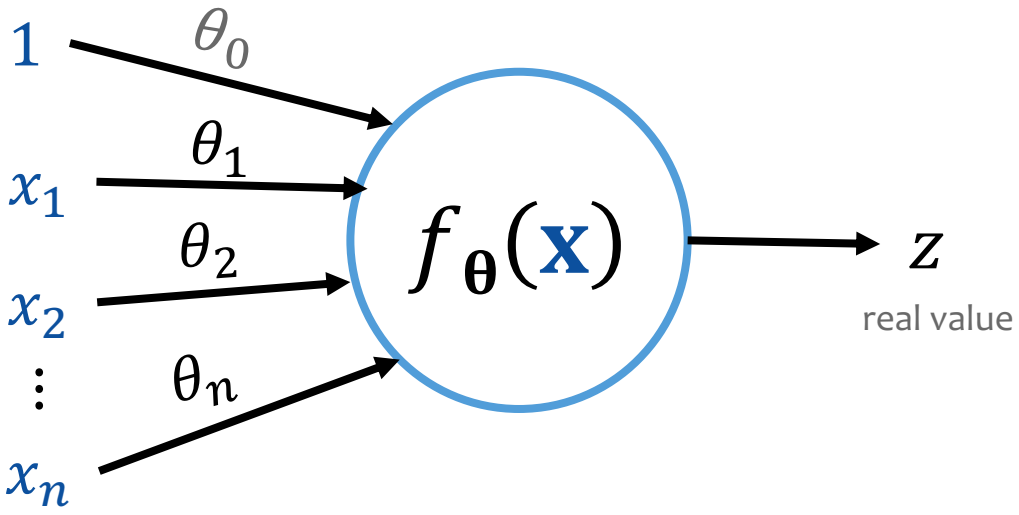
real value

How about **transforming** this **weighted sum** (feature contribution) into the **probability** that the sample **x** belongs to the **positive class**?

If the **probability** is >= 0.5, we consider that the **sample** belongs to the **positive class**. Otherwise, it belongs to the **negative class**.

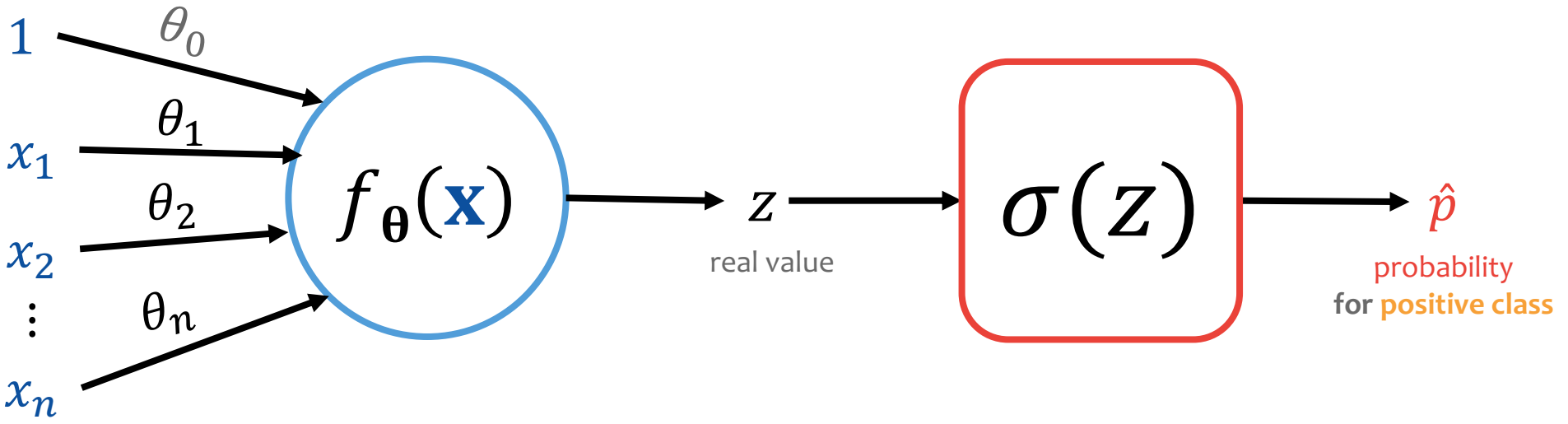
From Linear Regression to Bin. Linear Classification

Binary Linear Classifier



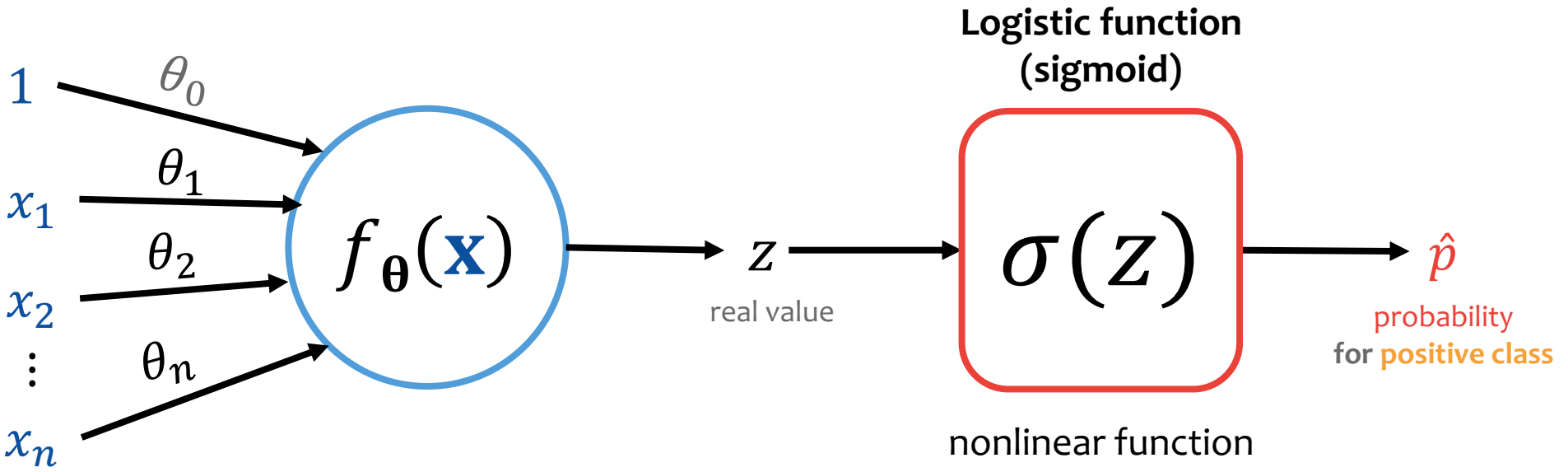
From Linear Regression to Bin. Linear Classification

Binary Linear Classifier



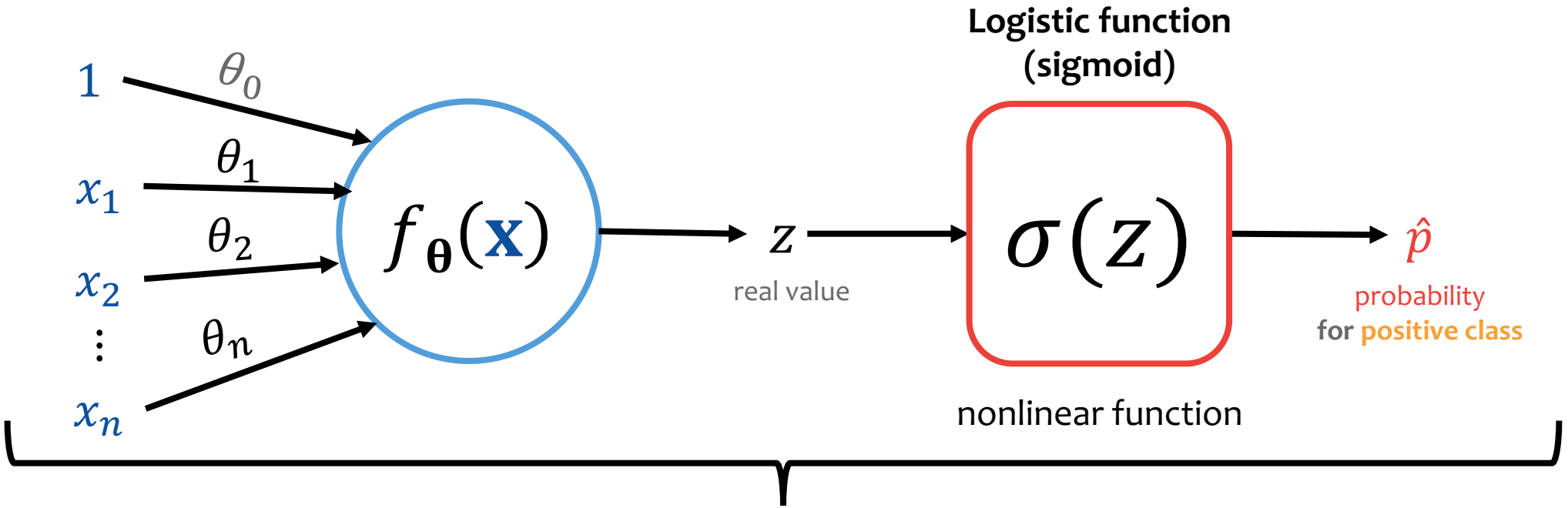
From Linear Regression to Bin. Linear Classification

Binary Linear Classifier



From Linear Regression to Bin. Linear Classification

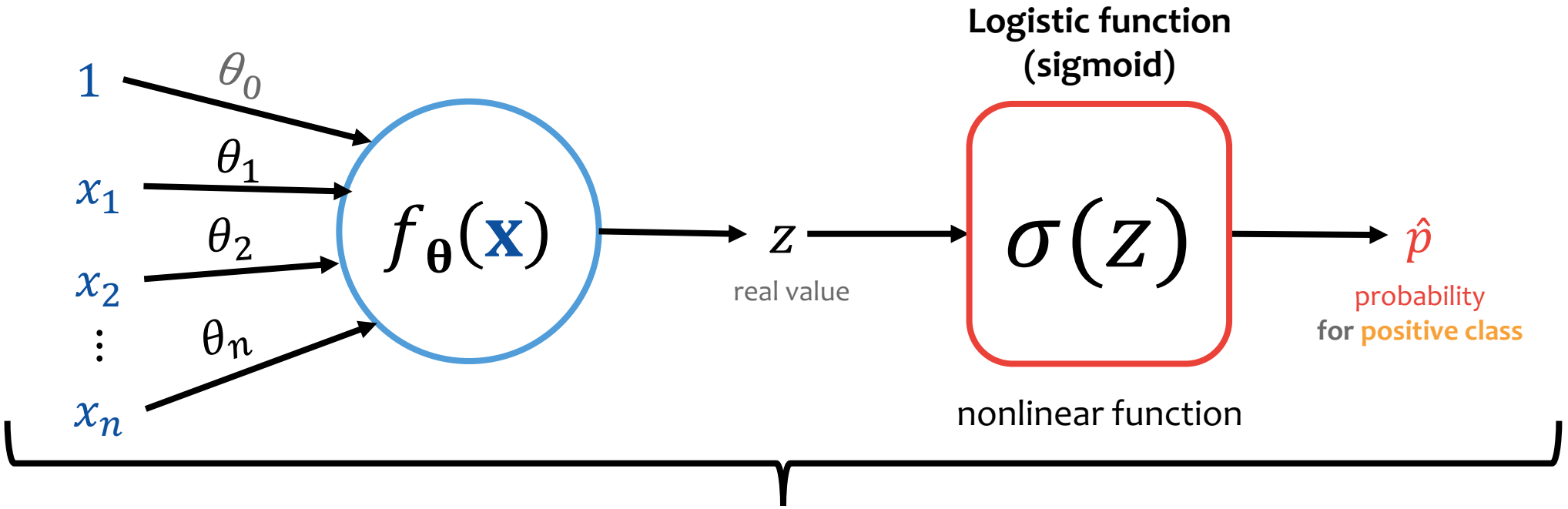
Binary Linear Classifier



This is our **new model** $h_{\boldsymbol{\theta}}(\mathbf{x})$ (**binary linear classifier**)

From Linear Regression to Bin. Linear Classification

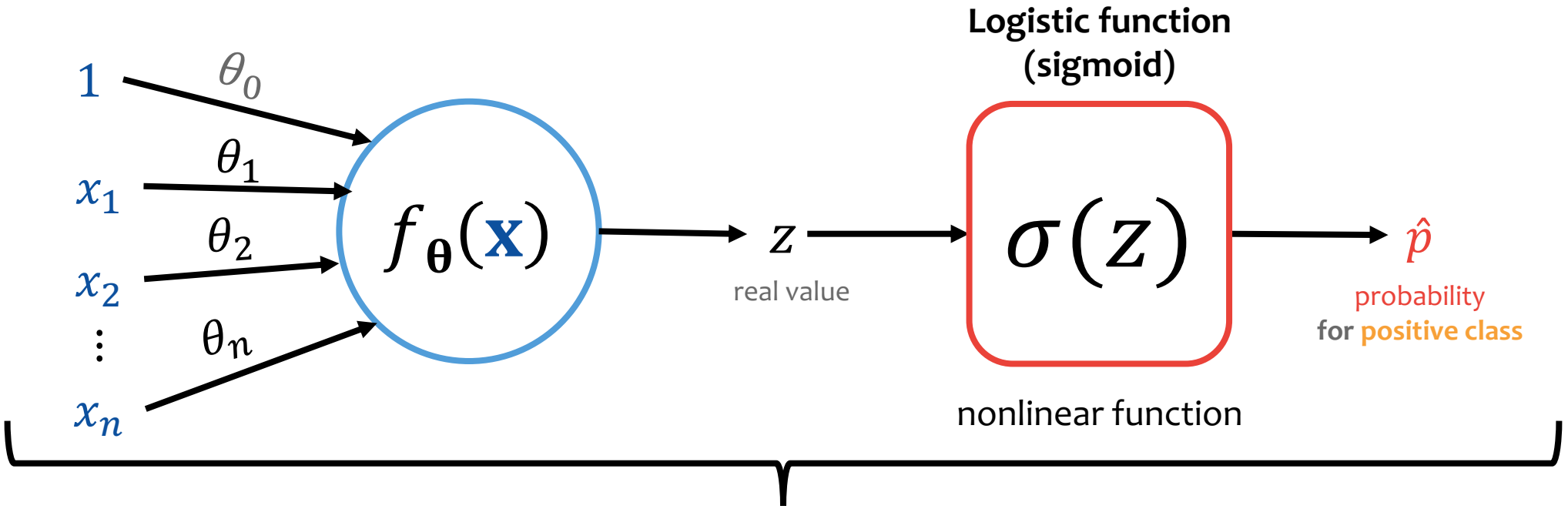
Binary Linear Classifier



$$\hat{p} = h_{\mathbf{\theta}}(\mathbf{x})$$

From Linear Regression to Bin. Linear Classification

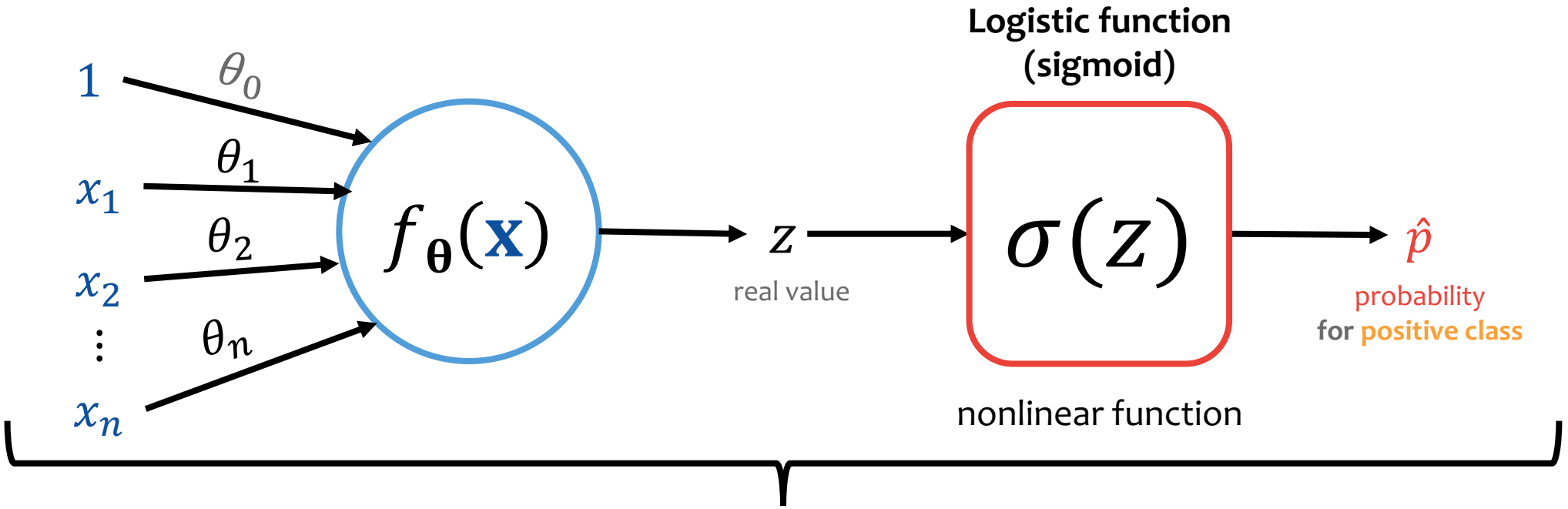
Binary Linear Classifier



$$\hat{p} = h_{\mathbf{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1 | \mathbf{x}; \mathbf{\theta})$$

From Linear Regression to Bin. Linear Classification

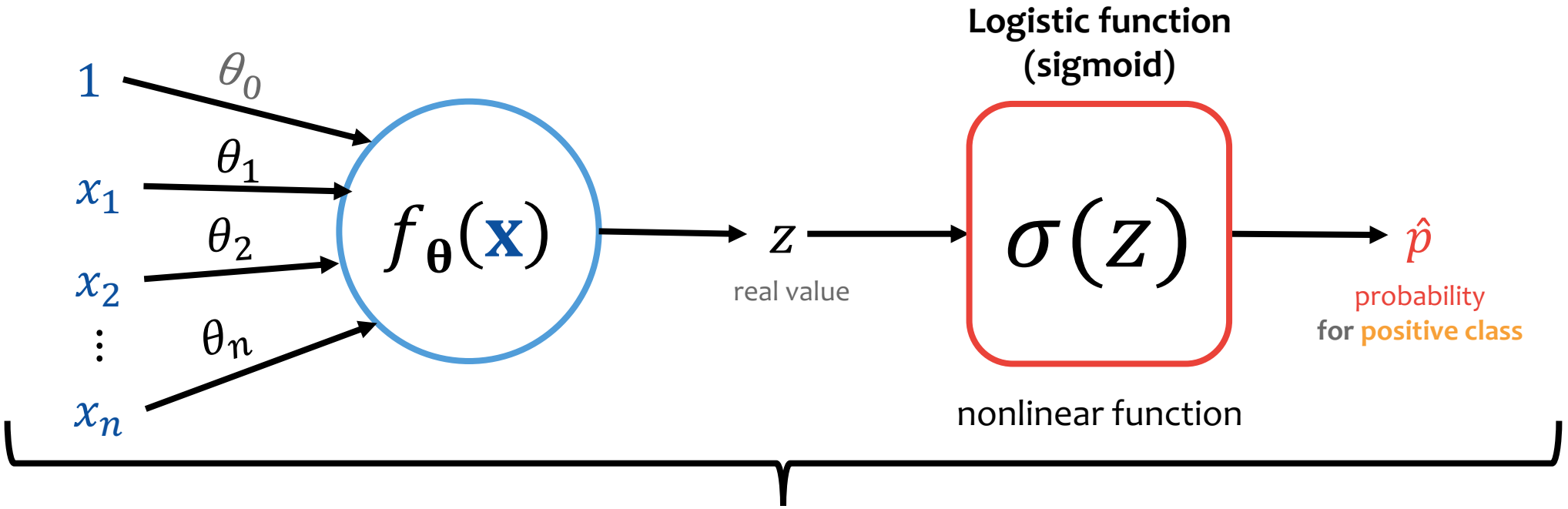
Binary Linear Classifier



$$\hat{p} = h_{\mathbf{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \mathbf{\theta}) = \sigma(z)$$

From Linear Regression to Bin. Linear Classification

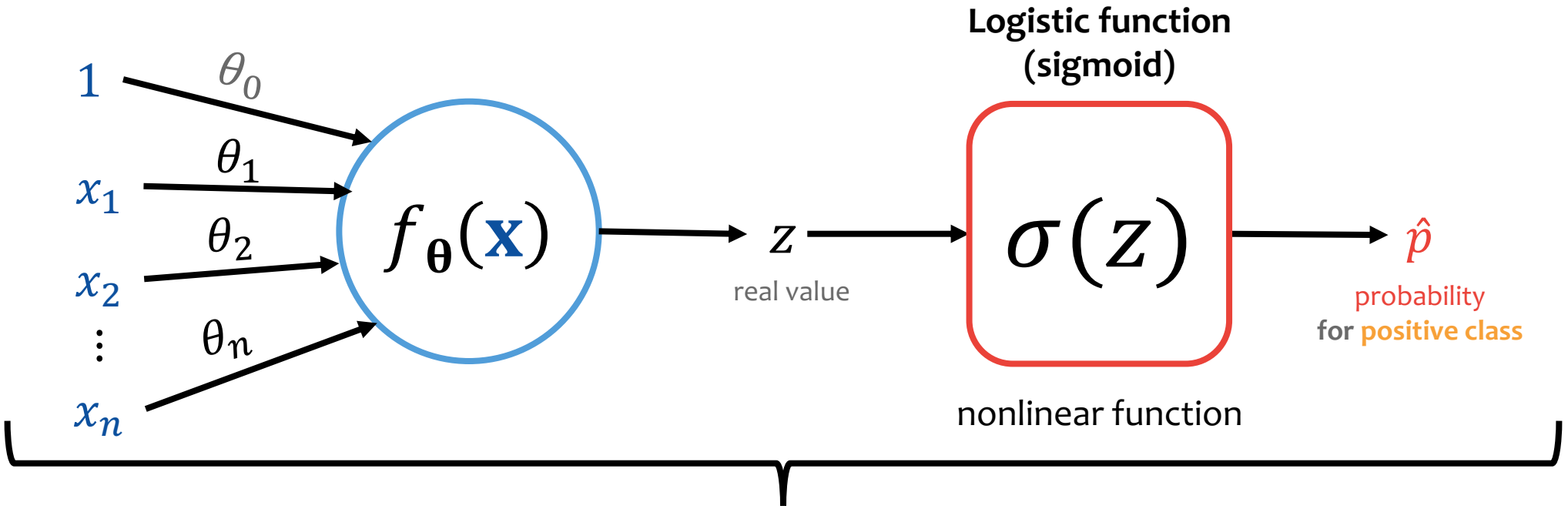
Binary Linear Classifier



$$\hat{p} = h_{\mathbf{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \mathbf{\theta}) = \sigma(z) = \sigma\big(f_{\mathbf{\theta}}(\mathbf{x})\big)$$

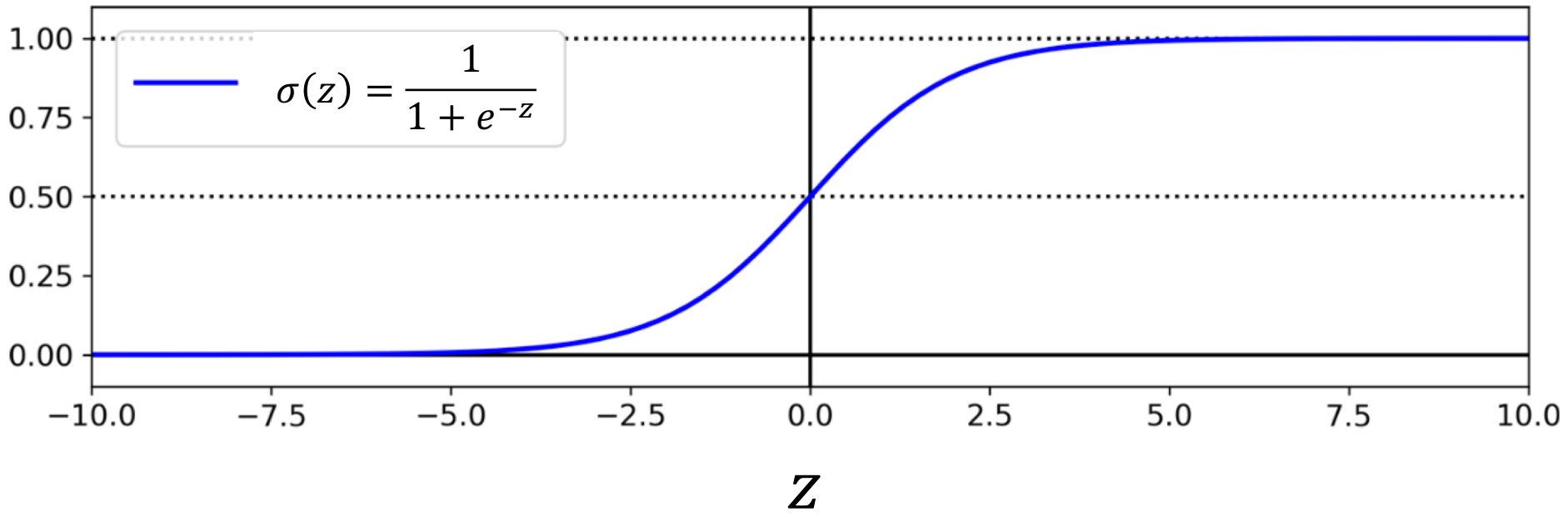
From Linear Regression to Bin. Linear Classification

Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\left(f_{\boldsymbol{\theta}}(\mathbf{x})\right) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x})$$

Logistic Function (Sigmoid)



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

nonlinear function

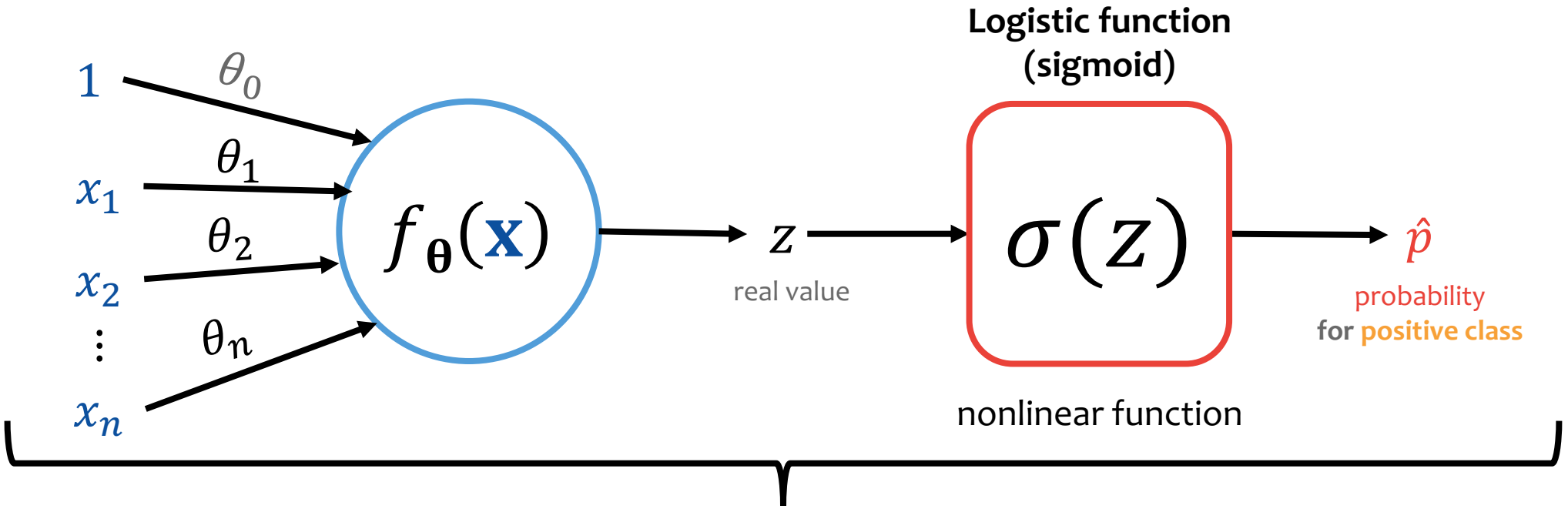
$\boldsymbol{\sigma(z)}$ returns a value between **0** and **1**

$\boldsymbol{\sigma(z) < 0.5}$ if $\boldsymbol{z < 0}$

$\boldsymbol{\sigma(z) \geq 0.5}$ if $\boldsymbol{z \geq 0}$

From Linear Regression to Bin. Linear Classification

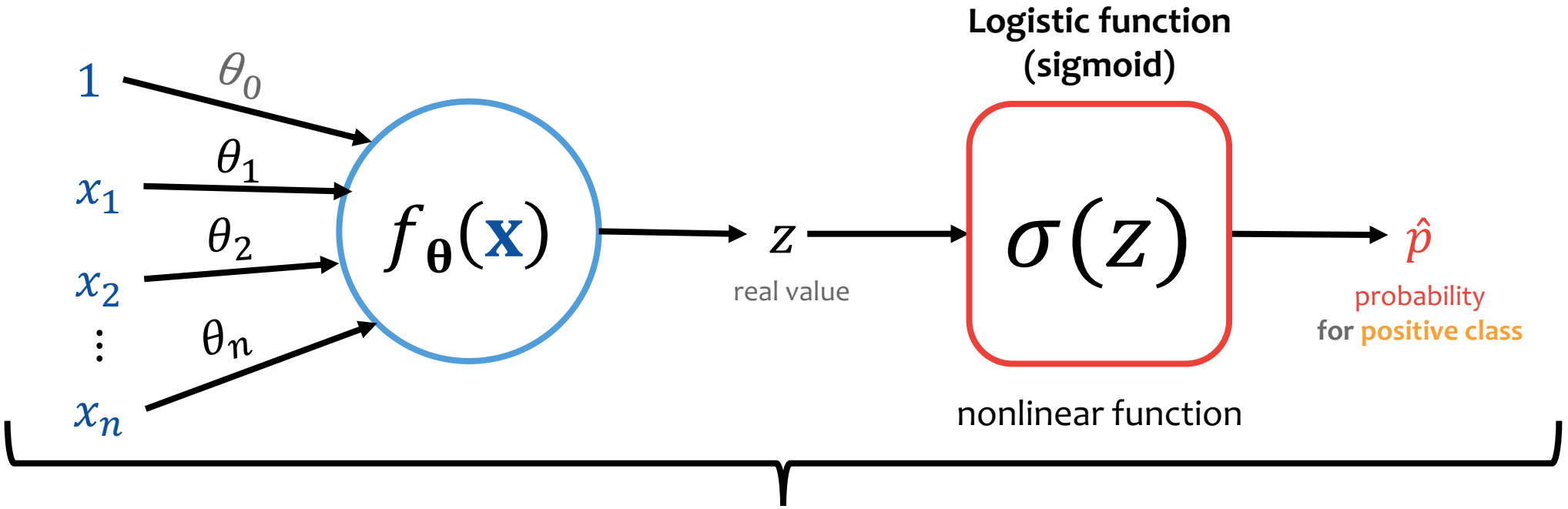
Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\big(f_{\boldsymbol{\theta}}(\mathbf{x})\big) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x})$$

From Linear Regression to Bin. Linear Classification

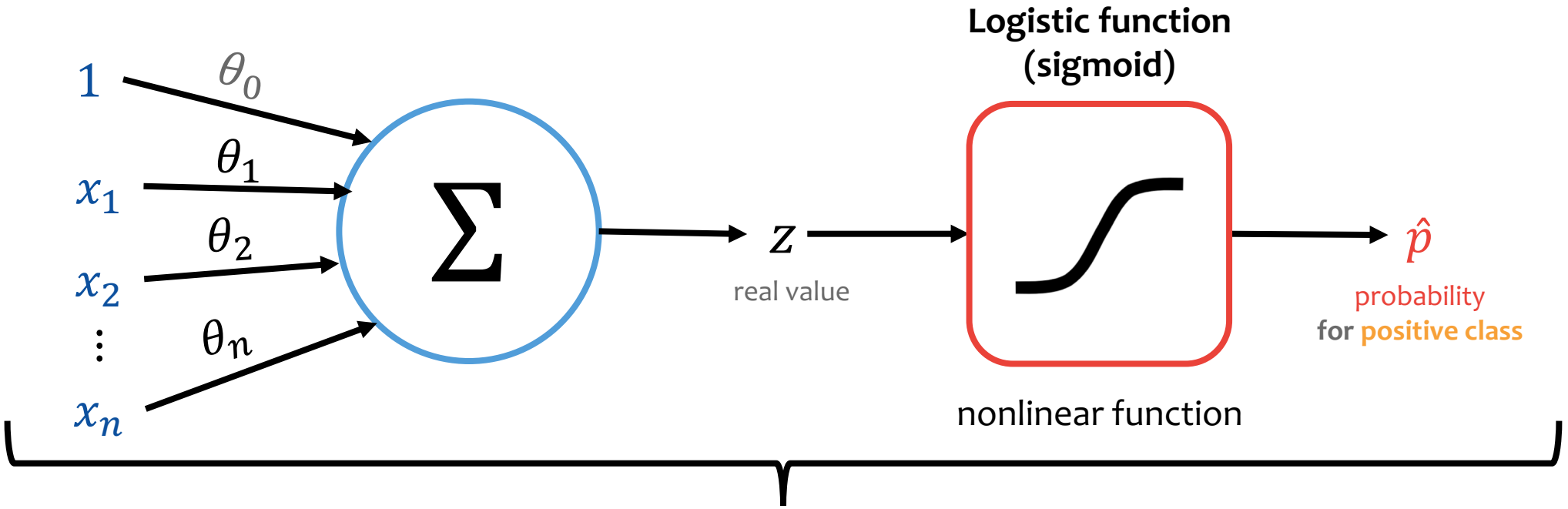
Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1 | \mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\left(f_{\boldsymbol{\theta}}(\mathbf{x})\right) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}}}$$

From Linear Regression to Bin. Linear Classification

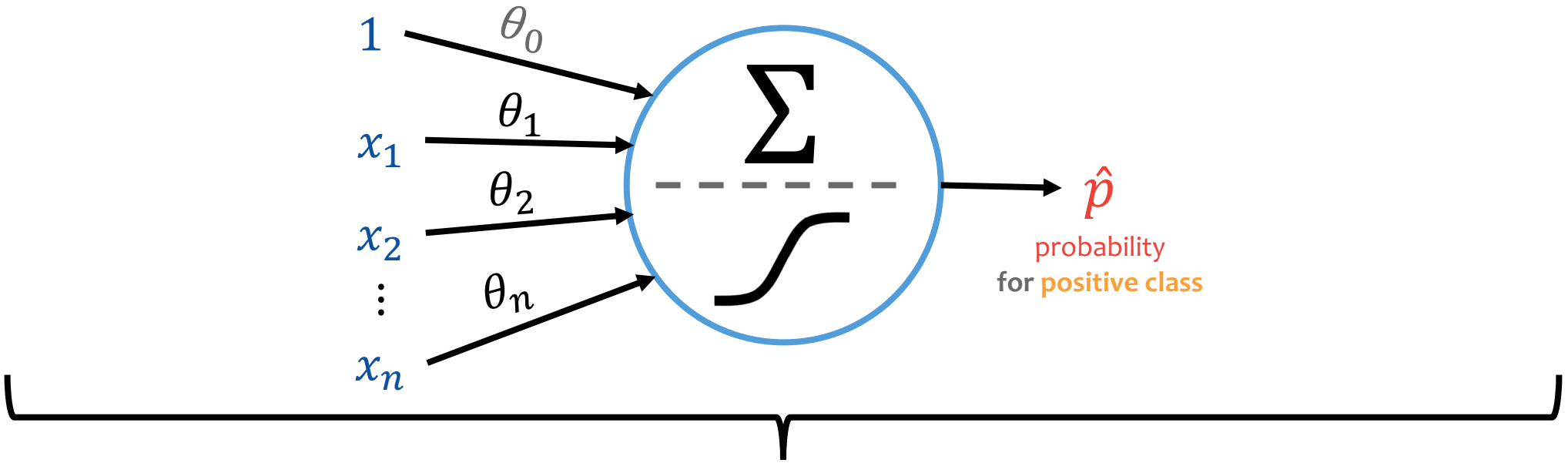
Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\big(f_{\boldsymbol{\theta}}(\mathbf{x})\big) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}}}$$

From Linear Regression to Bin. Linear Classification

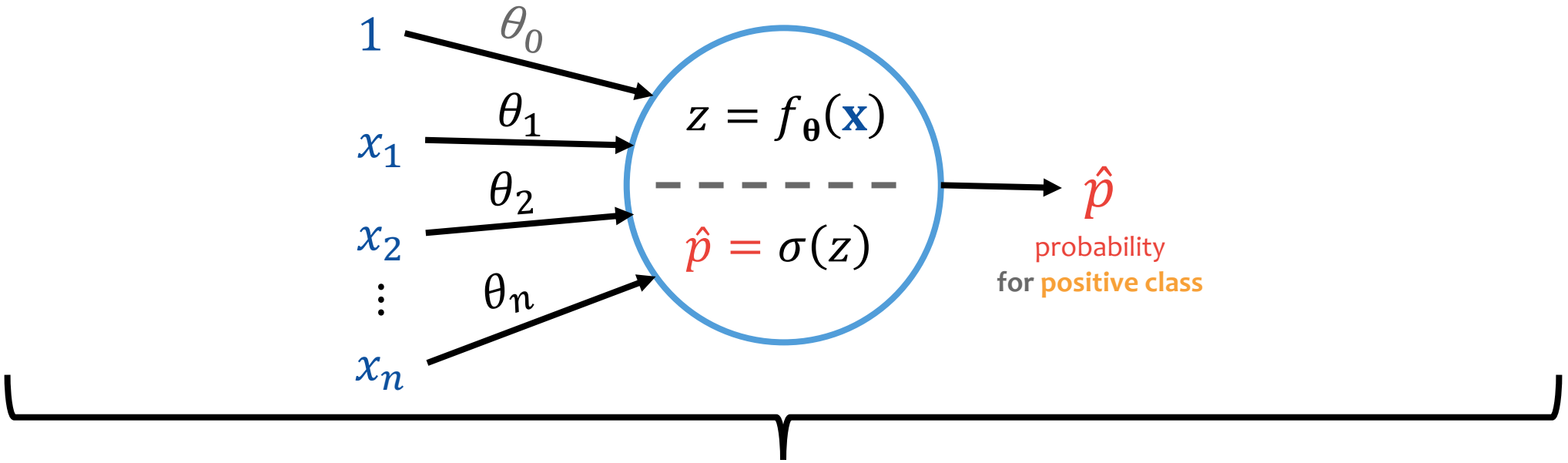
Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\big(f_{\boldsymbol{\theta}}(\mathbf{x})\big) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}}}$$

From Linear Regression to Bin. Linear Classification

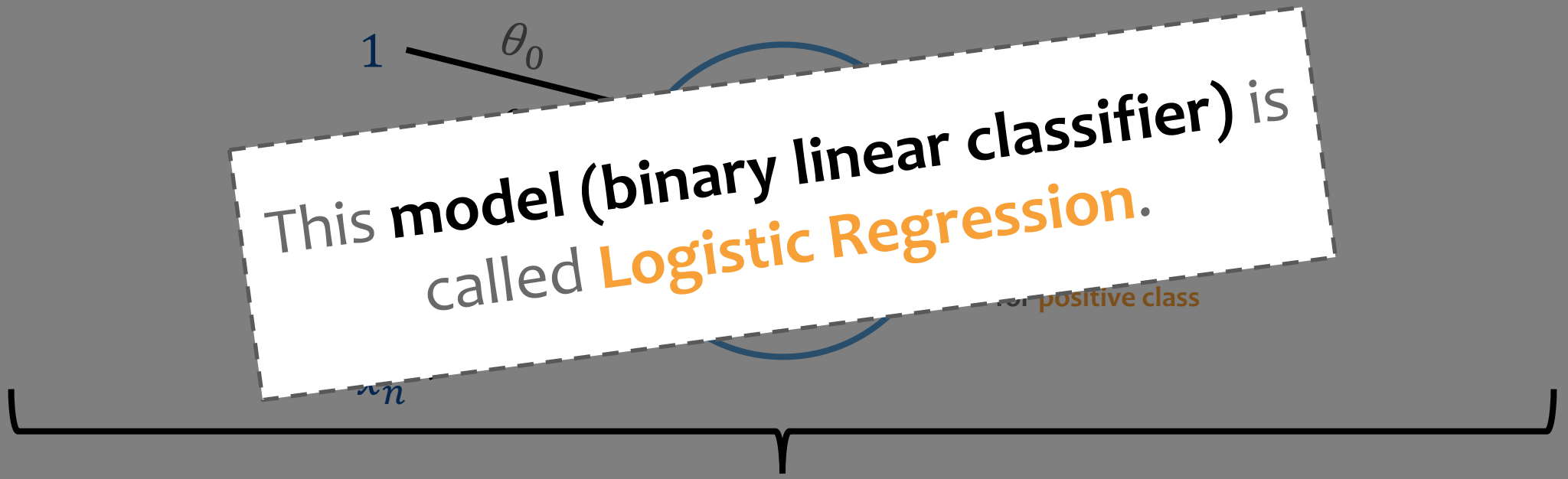
Binary Linear Classifier



$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1 | \mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\left(f_{\boldsymbol{\theta}}(\mathbf{x})\right) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}}}$$

From Linear Regression to Bin. Linear Classification

Binary Linear Classifier

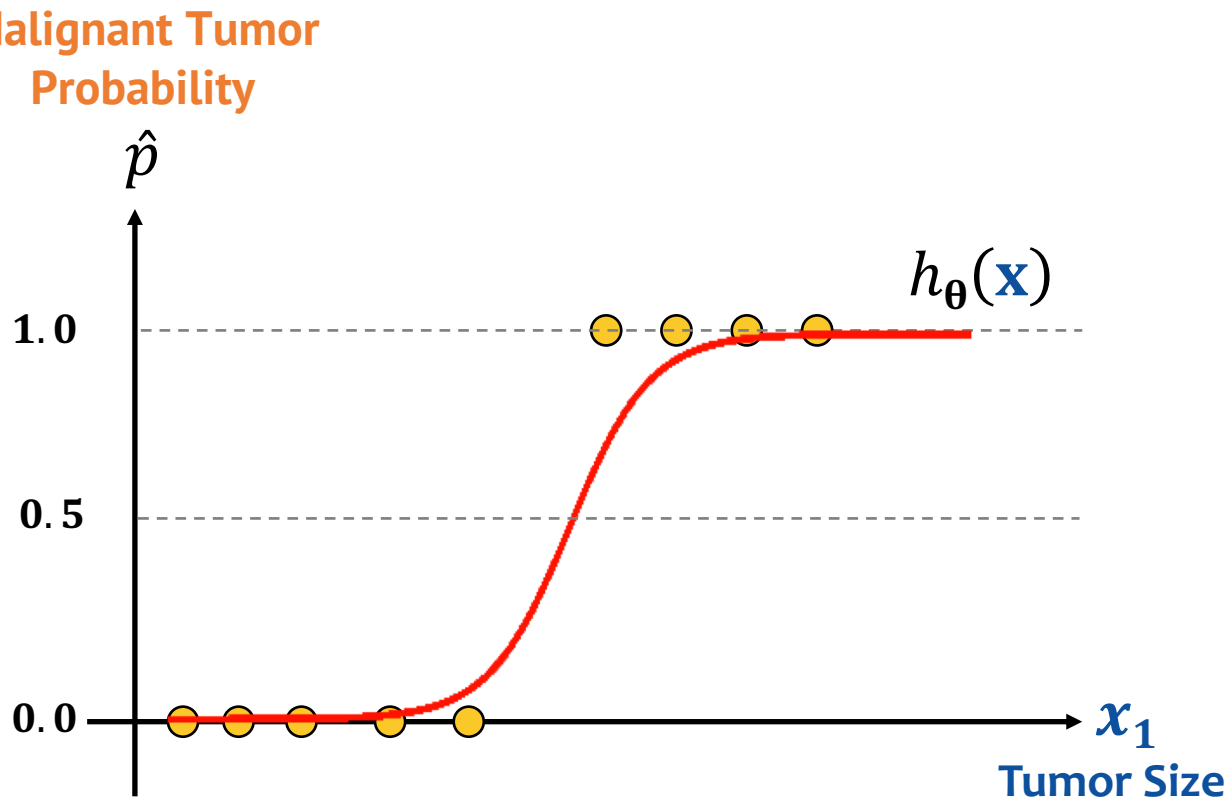


This **model (binary linear classifier)** is called **Logistic Regression**.

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{P}(\mathbf{y} = 1|\mathbf{x}; \boldsymbol{\theta}) = \sigma(z) = \sigma\big(f_{\boldsymbol{\theta}}(\mathbf{x})\big) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}}}$$

Logistic Regression Model

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathrm{T}} * \mathbf{x}}}$$



Logistic Regression Model

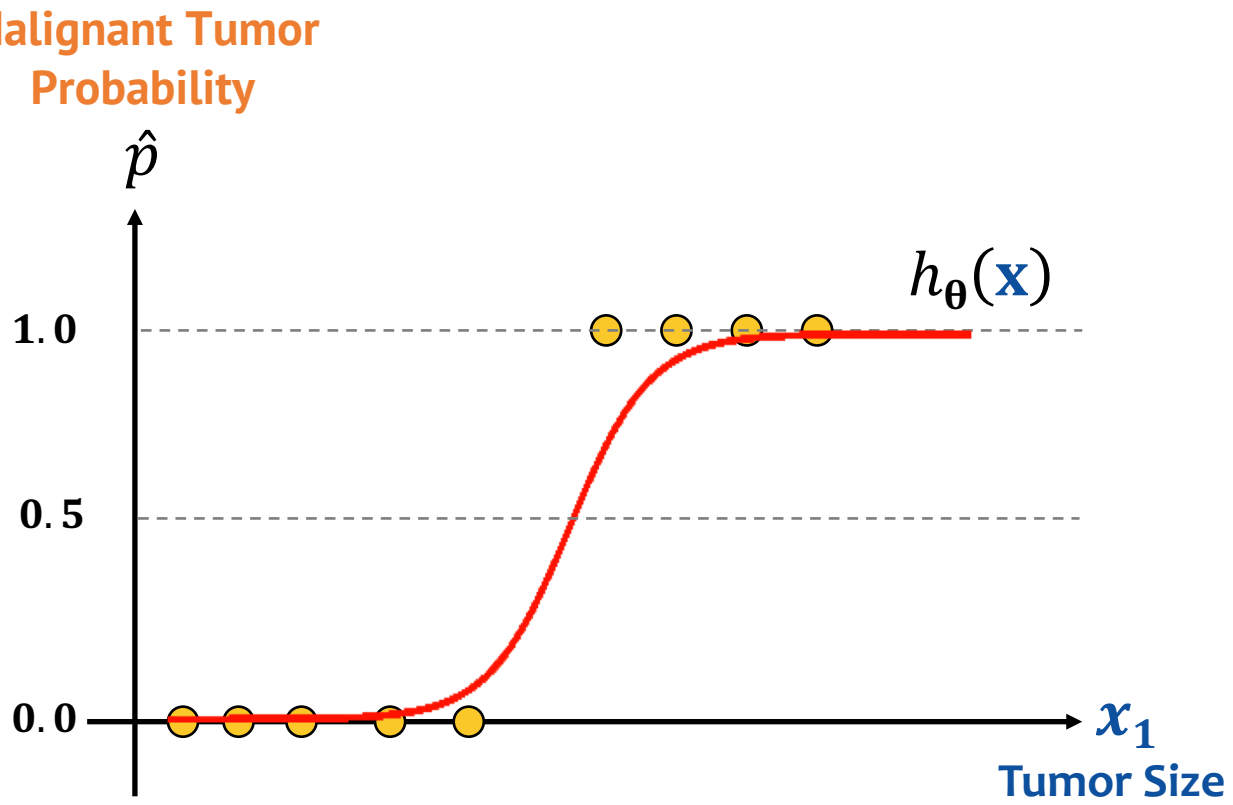
$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathrm{T}} * \mathbf{x}}}$$

E.g:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ tumor\ size \end{bmatrix}$$

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{0.7}$$

The **probability** of the tumor being **malignant (positive class)** is **70%.**



Logistic Regression Model

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

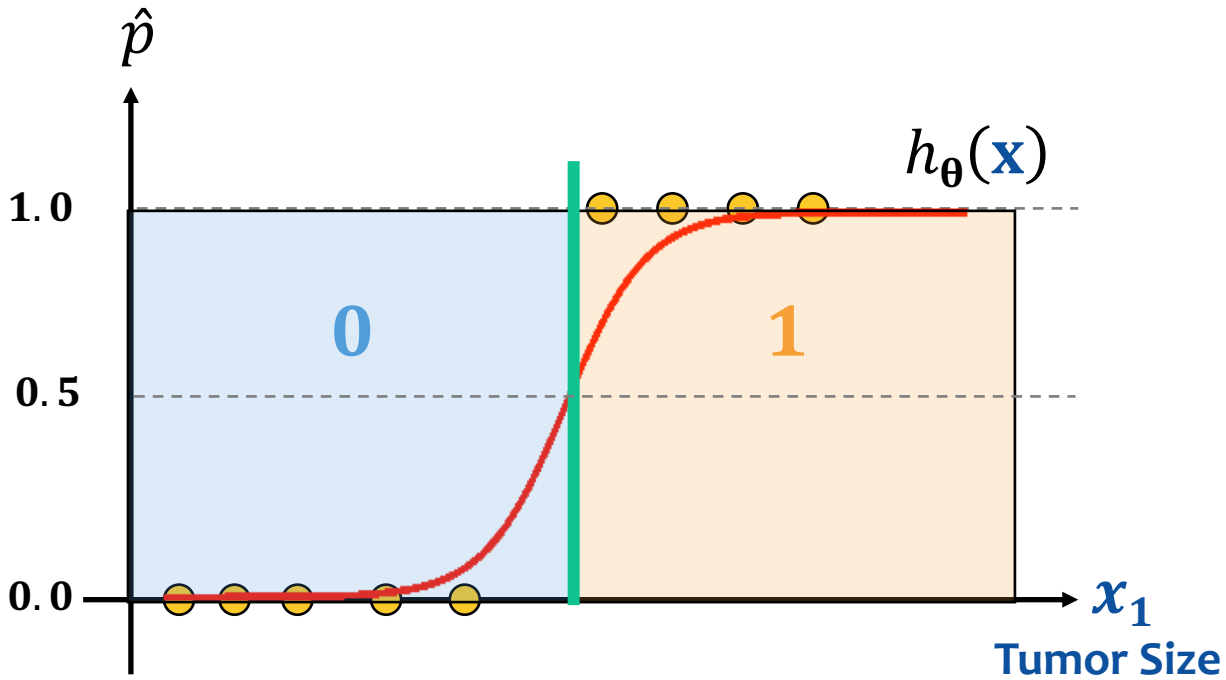
E.g:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ tumor\ size \end{bmatrix}$$

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{0.7}$$

The **probability** of the tumor being **malignant (positive class)** is **70%.**

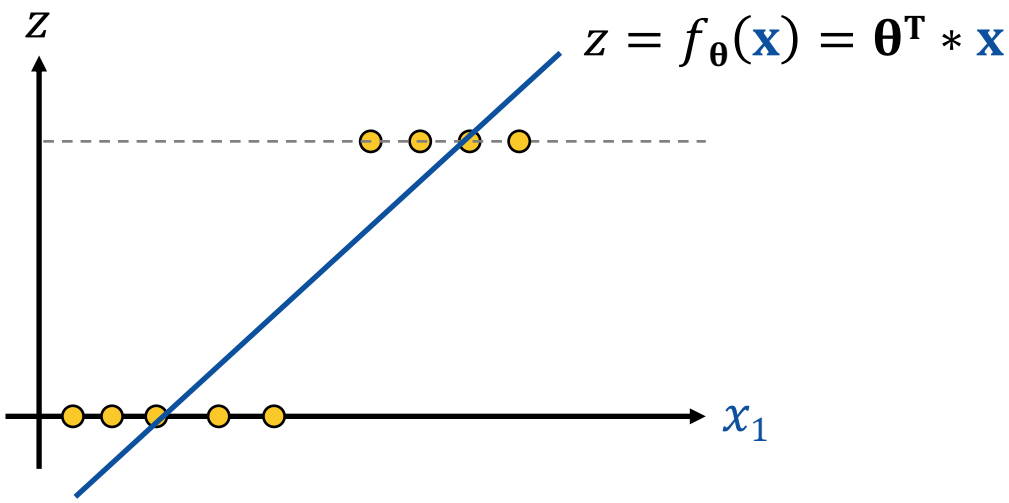
Malignant Tumor Probability



Classification

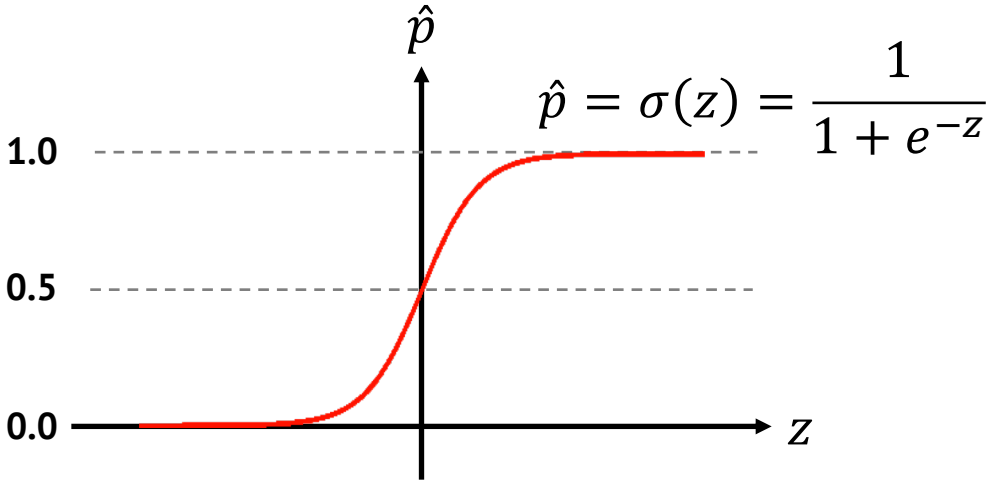
$$\hat{y} = \begin{cases} 1 & if\ \hat{p} \geq 0.5 \\ 0 & if\ \hat{p} < 0.5 \end{cases}$$

Linear Model



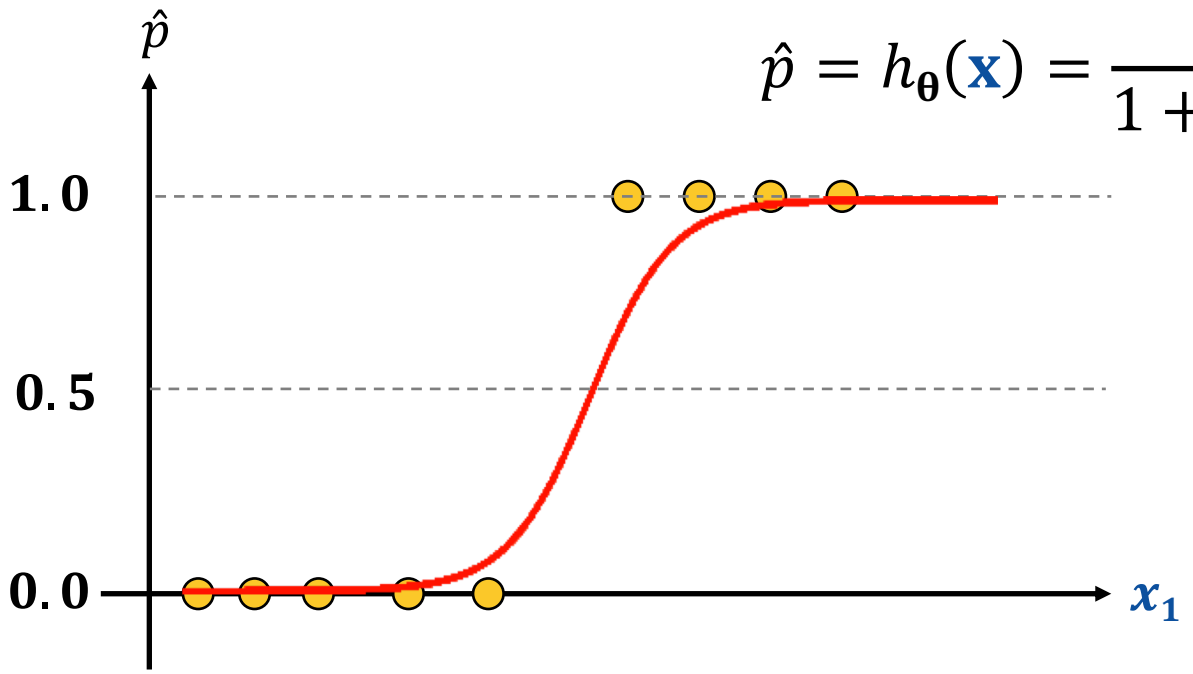
nonlinear

Logistic Function (Sigmoid)



Logistic Regression Model

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



Training

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

Training

objective cost function

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

Training

objective cost function

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

Find the **vector $\boldsymbol{\theta}$** (weights and bias) that **minimizes** a given **cost function** J

Training

objective cost function

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

Find the **vector $\boldsymbol{\theta}$** (weights and bias) that **minimizes** a given **cost function J**

high probabilities for **positive instances** $(y = 1)$;

low probabilities for **negative instances** $(y = 0)$;

Training

objective cost function

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

Find the **vector $\boldsymbol{\theta}$** (weights and bias) that **minimizes** a given **cost function J** → **Gradient Descent!**

high probabilities for **positive instances** $(y = 1)$;

low probabilities for **negative instances** $(y = 0)$;

Cost Function

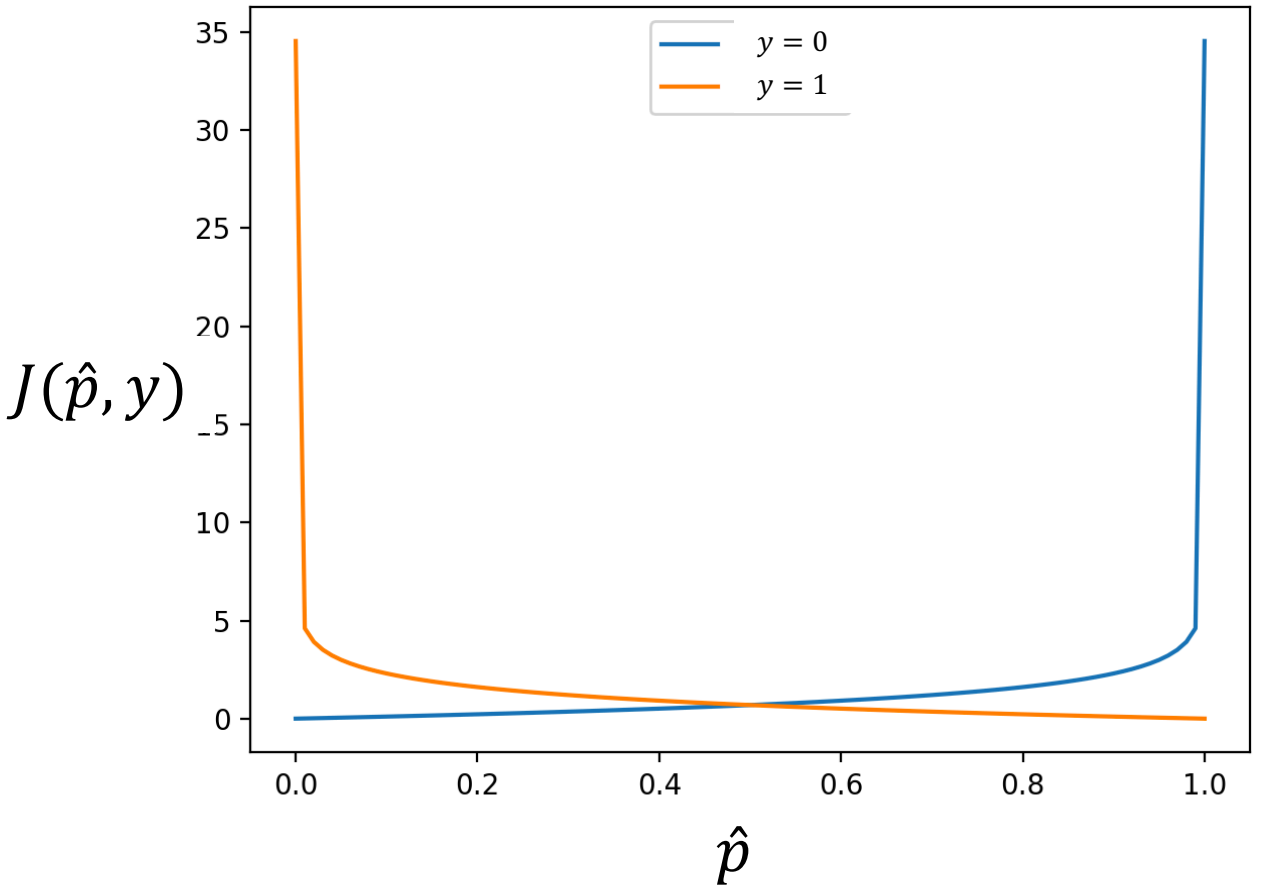
The common chosen loss function for **probabilities** is the **log loss cost function**, defined as:

Cost Function

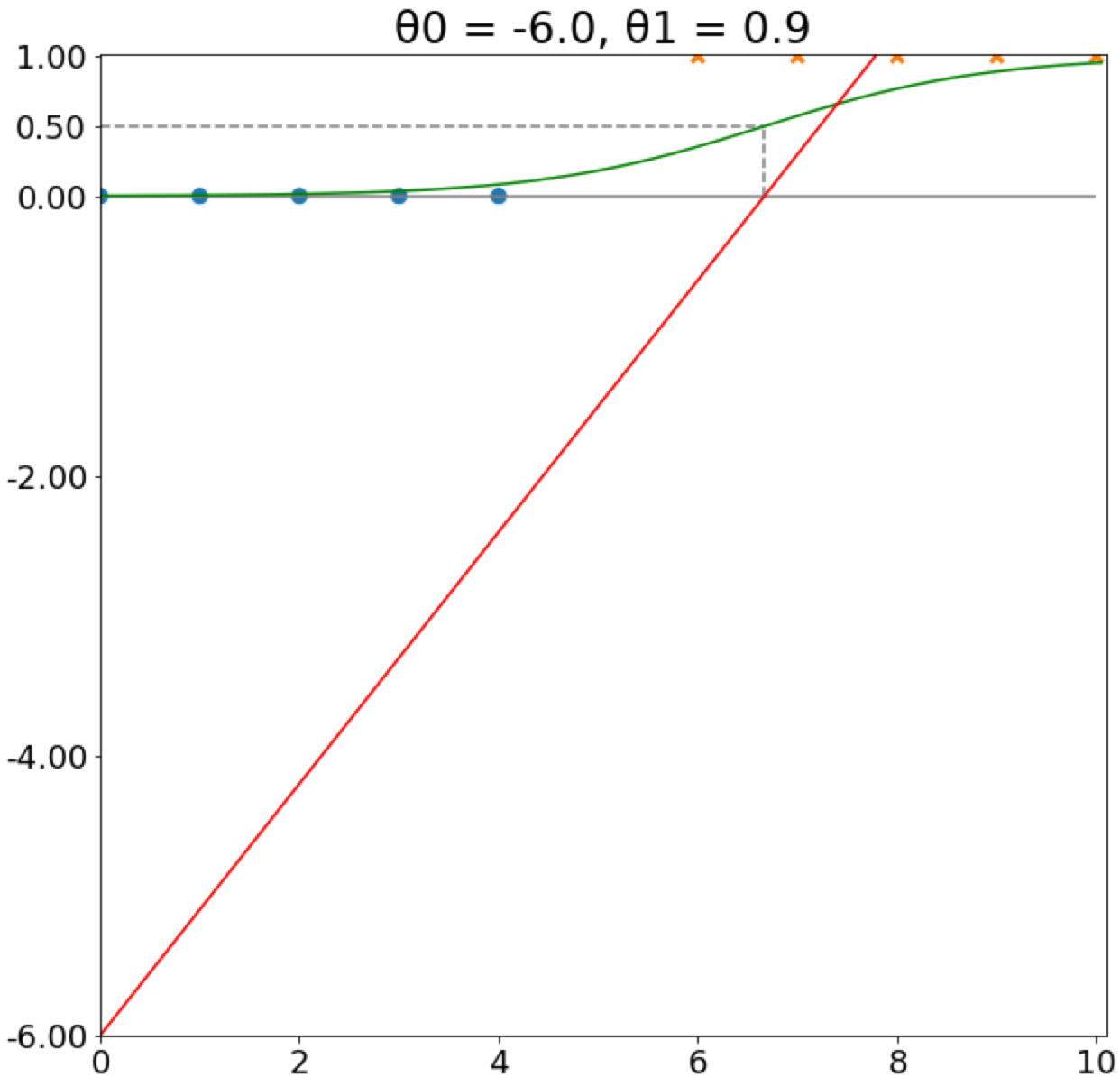
The common chosen loss function for **probabilities** is the **log loss cost function**, defined as:

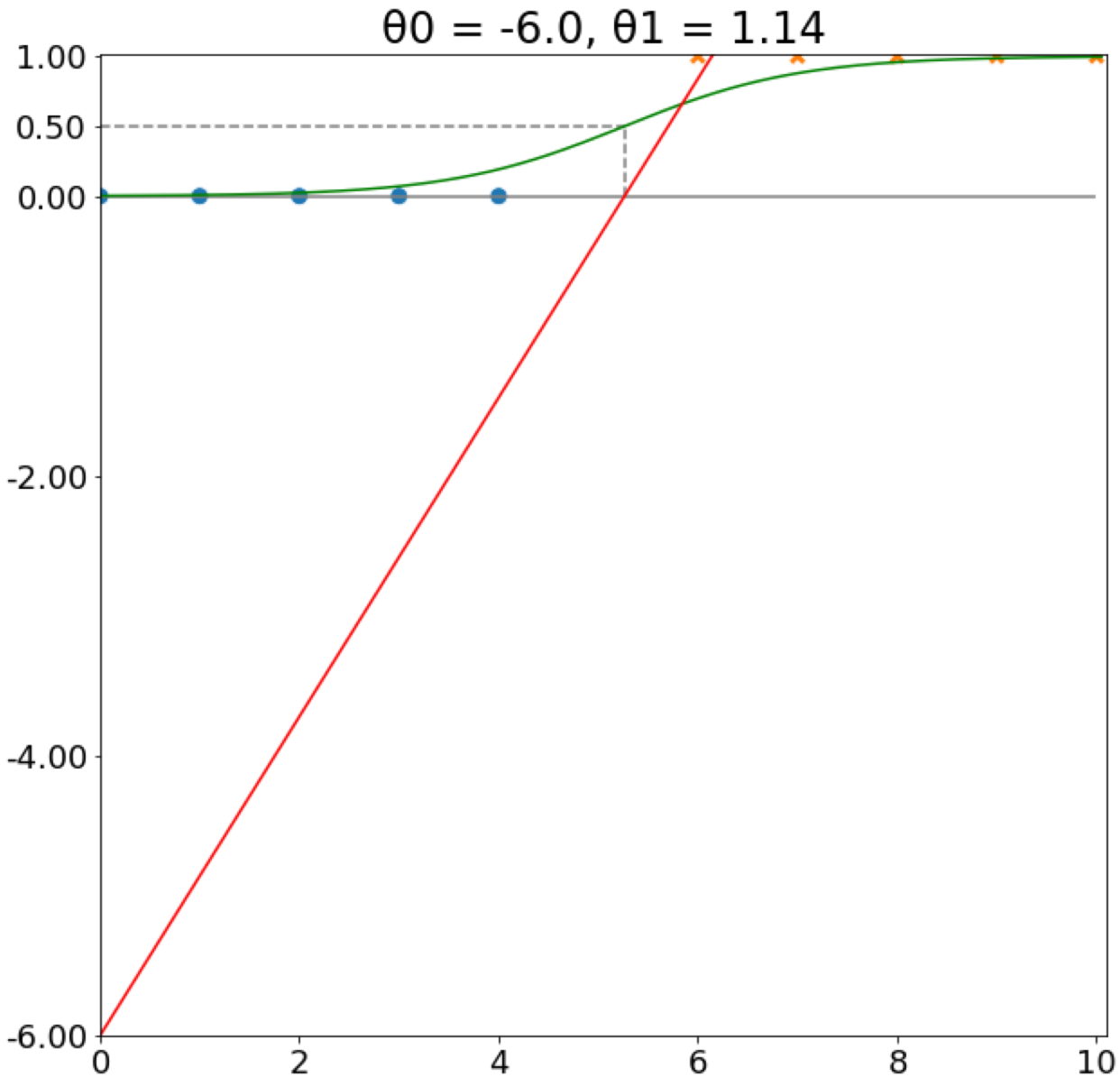
ground-truth (true label)

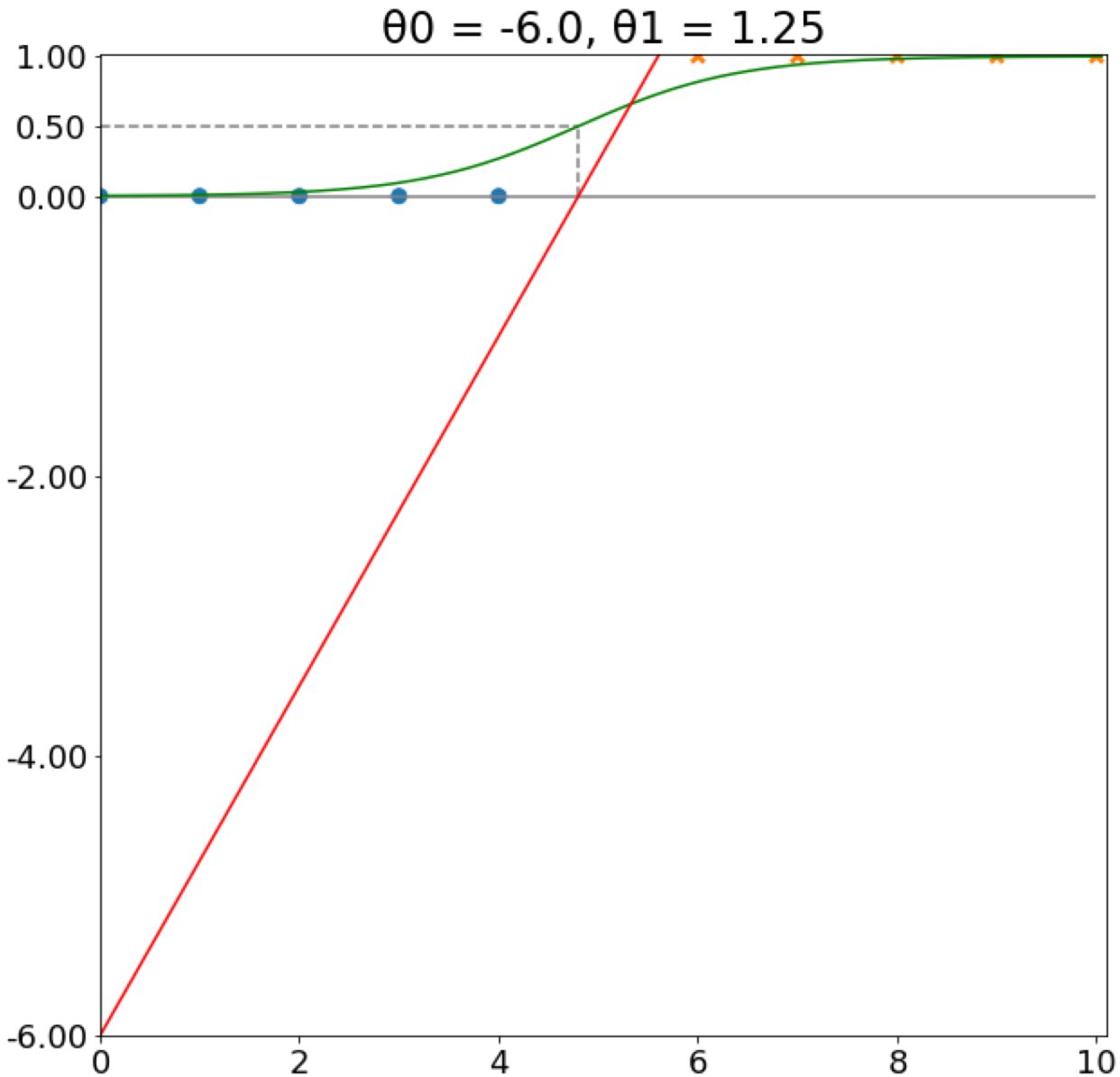
$$J(\hat{p}, y) = \begin{cases} -\log(\hat{p}) & \text{if } y = 1 \\ -\log(1 - \hat{p}) & \text{if } y = 0 \end{cases}$$

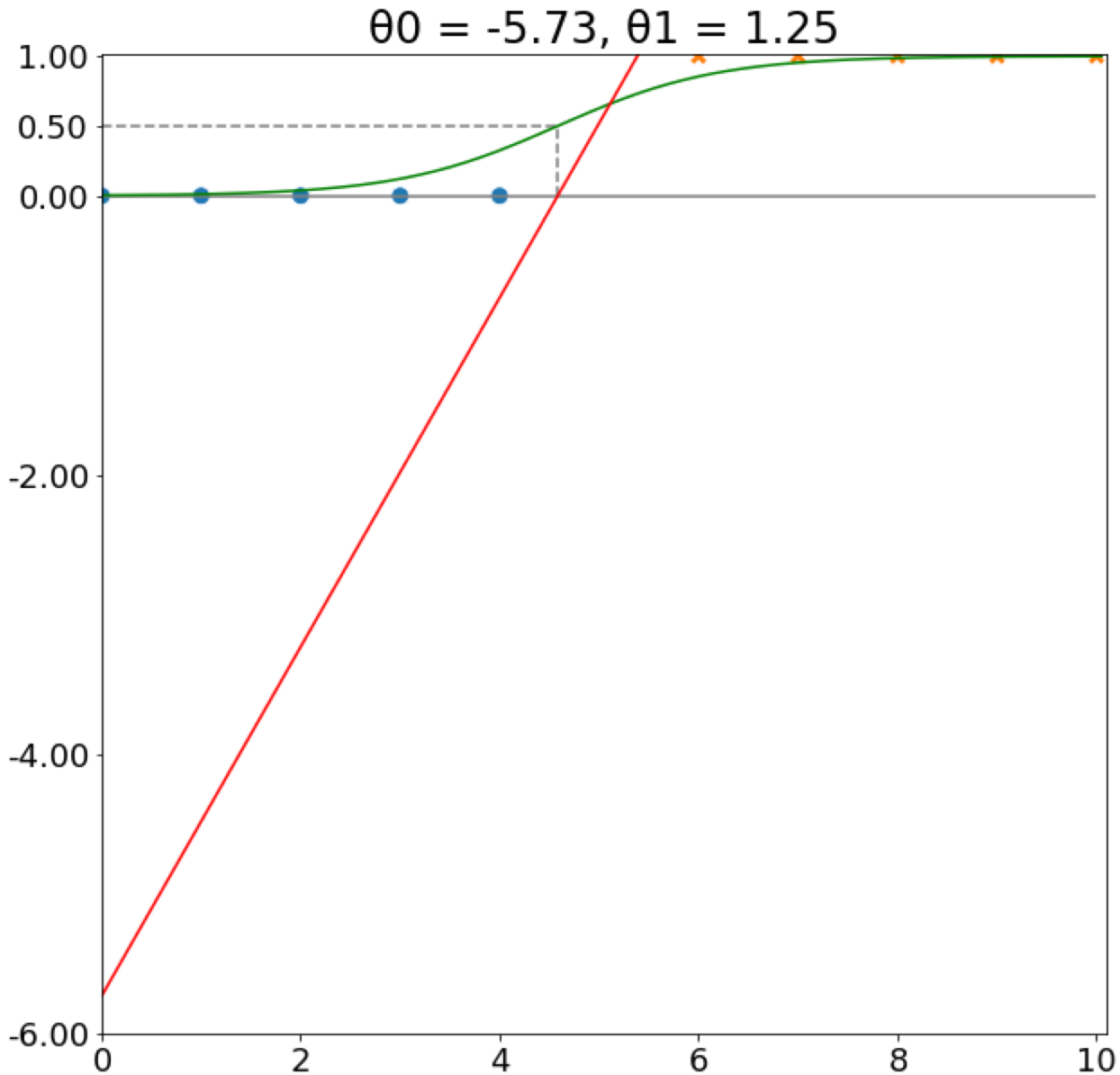


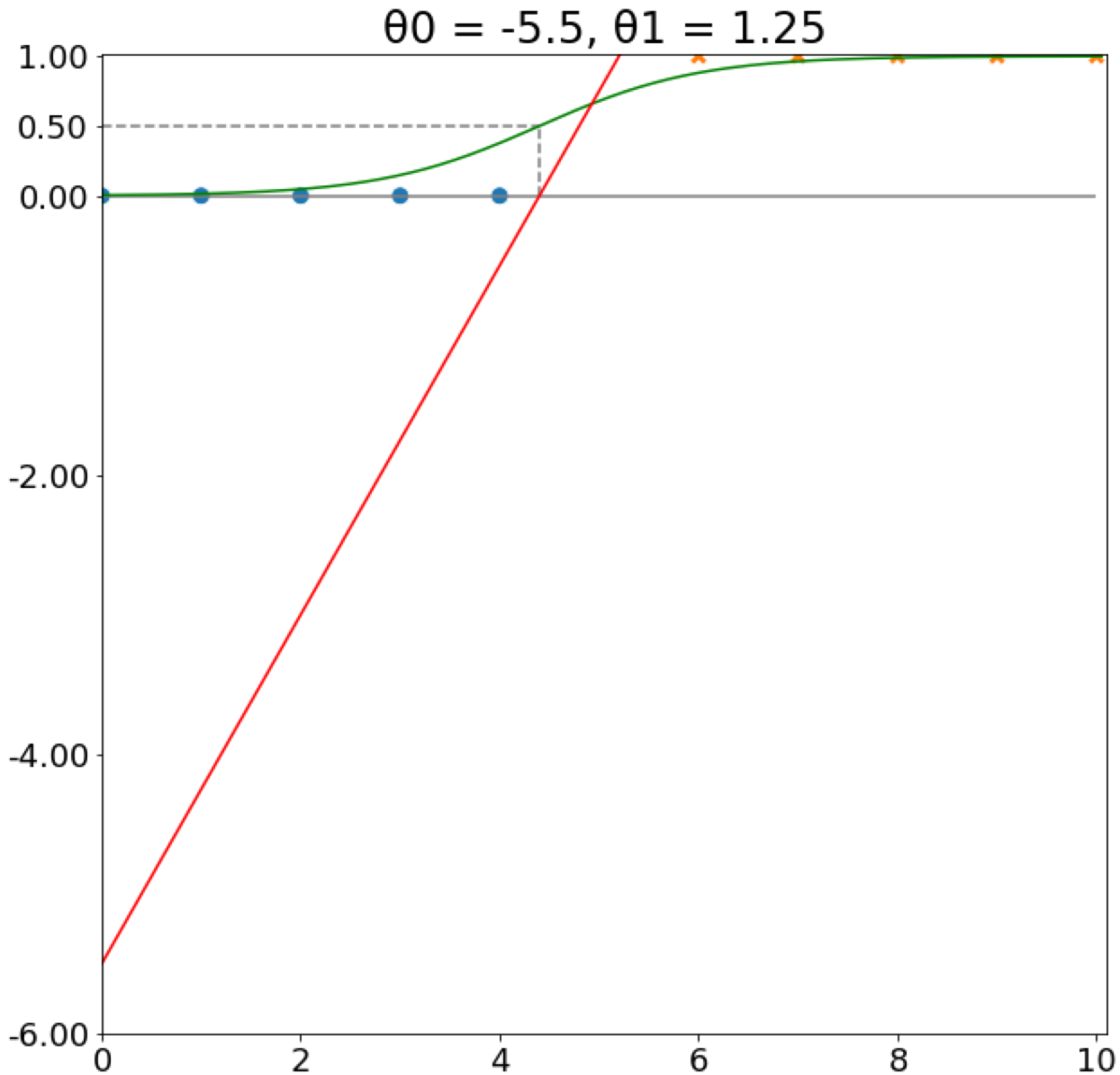
Adapted from: https://paperswithcode.com/paper/improving-pairwise-ranking-for-multi-label

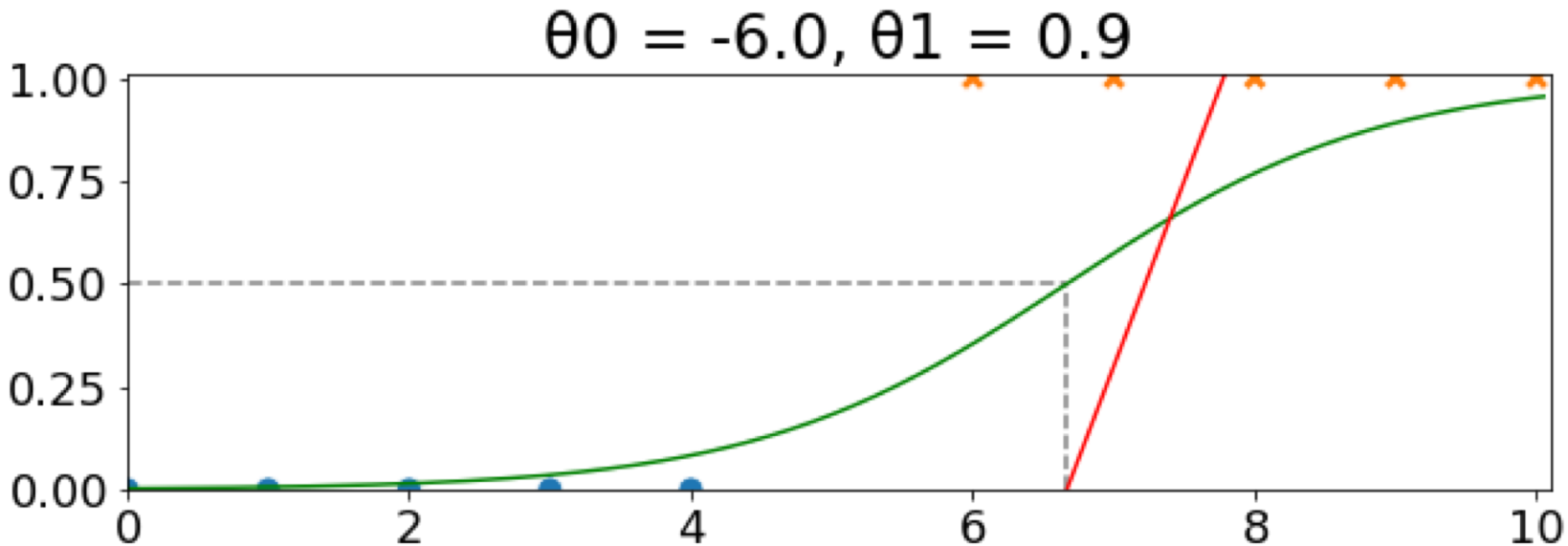

θ0 = -6.0, θ1 = 0.9
1.00
0.50
0.00
-2.00
-4.00
-6.00
0
2
4
6
8
10

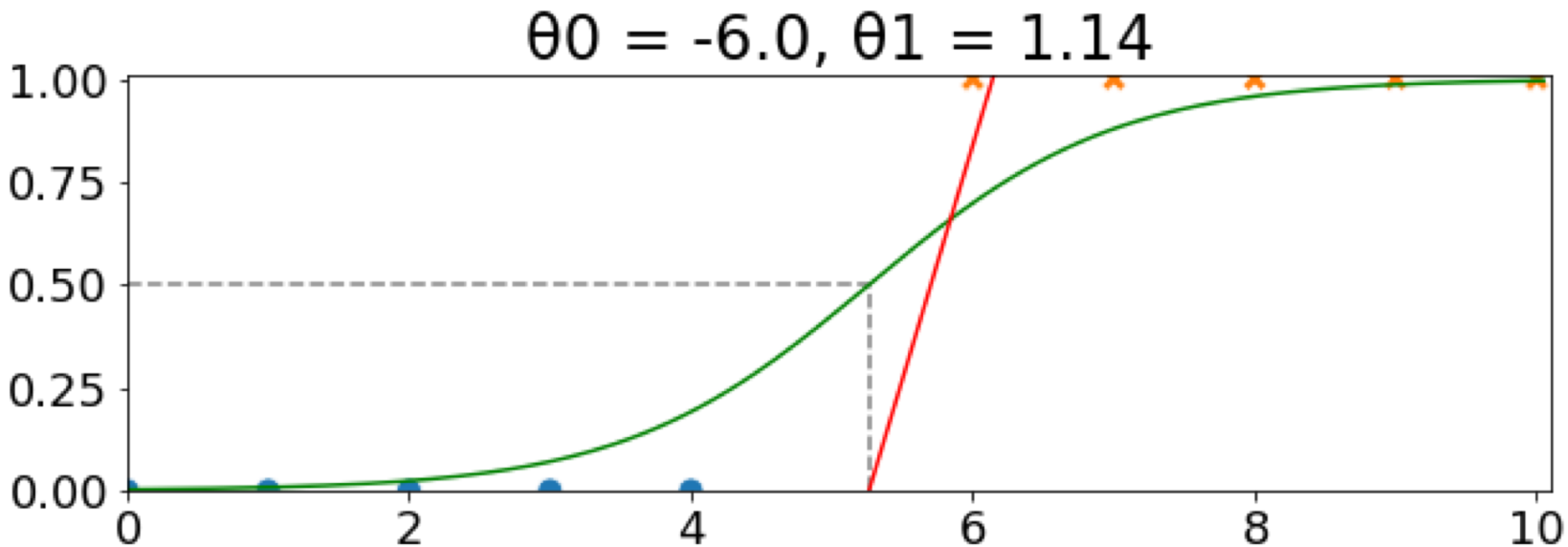

θ0 = -6.0, θ1 = 1.14
1.00
0.50
0.00
-2.00
-4.00
-6.00
0
2
4
6
8
10

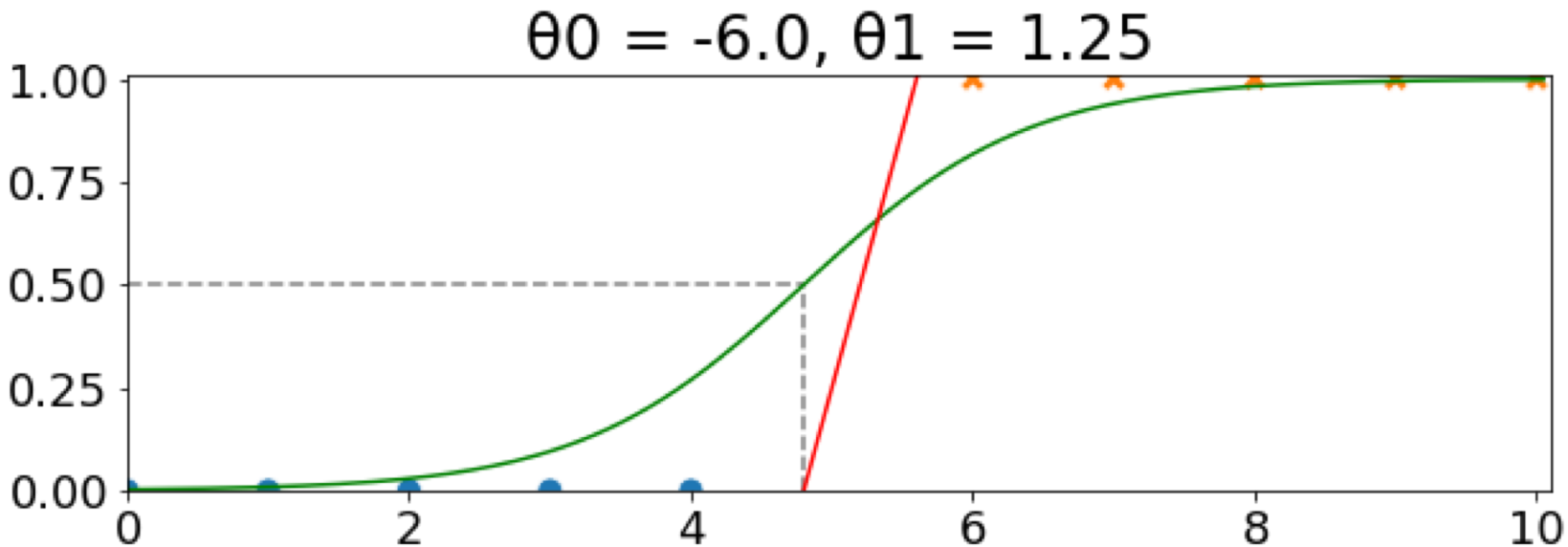

θ0 = -6.0, θ1 = 1.25
1.00
0.50
0.00
-2.00
-4.00
-6.00
0
2
4
6
8
10

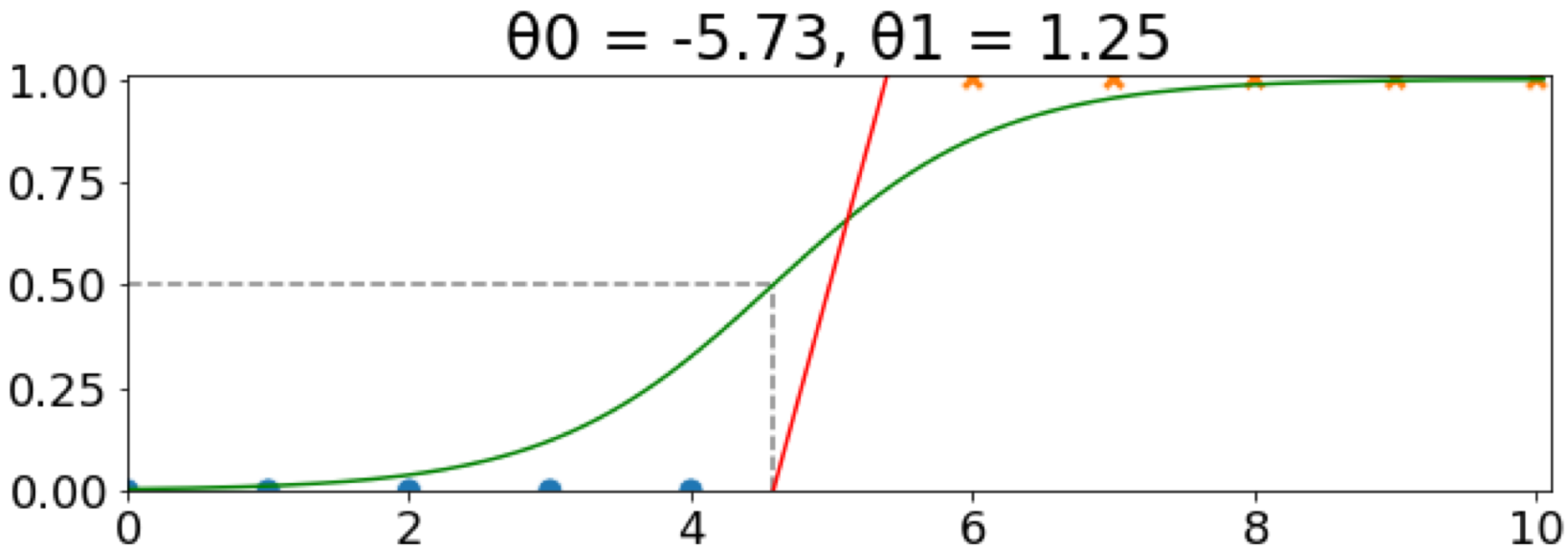

θ0 = -5.73, θ1 = 1.25
1.00
0.50
0.00
-2.00
-4.00
-6.00
0
2
4
6
8
10

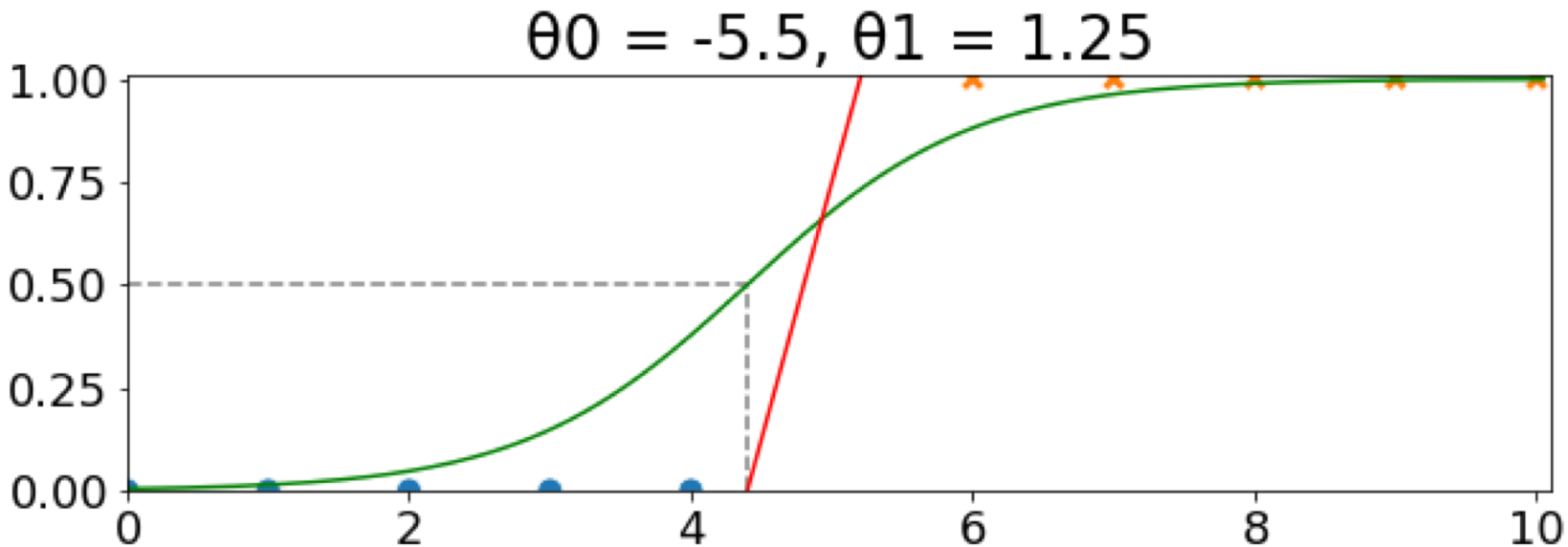

θ0 = -5.5, θ1 = 1.25
1.00
0.50
0.00
-2.00
-4.00
-6.00
0
2
4
6
8
10


θ0 = -6.0, θ1 = 0.9
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10

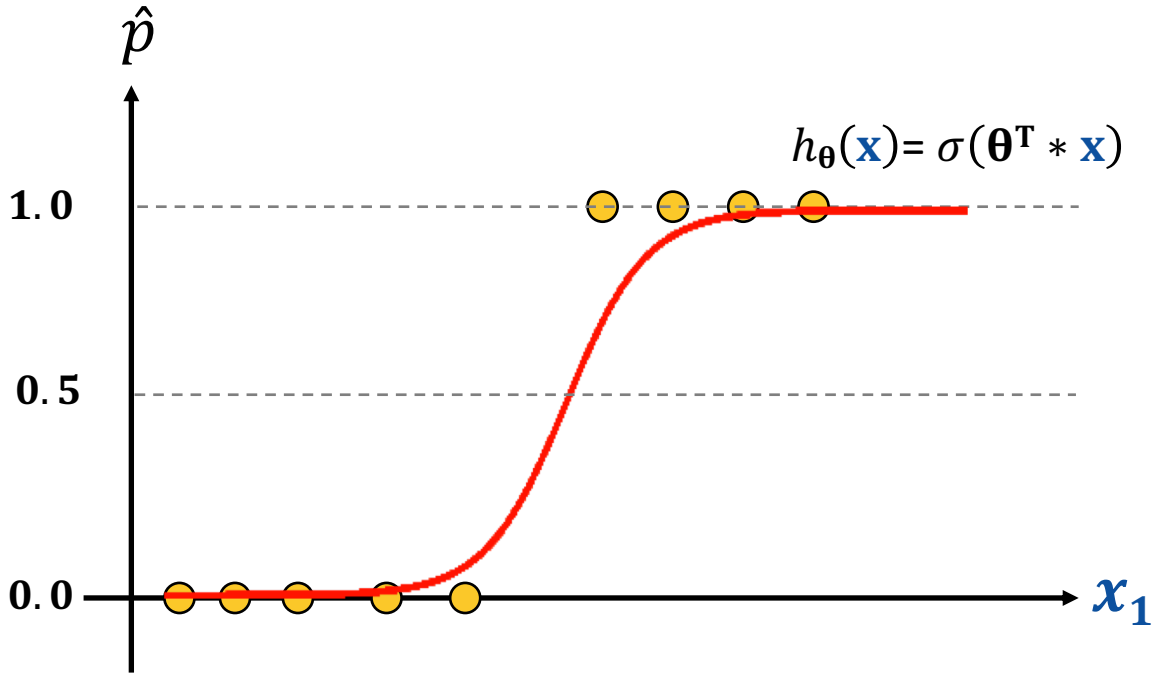

θ0 = -6.0, θ1 = 1.14
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10


θ0 = -6.0, θ1 = 1.25
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10

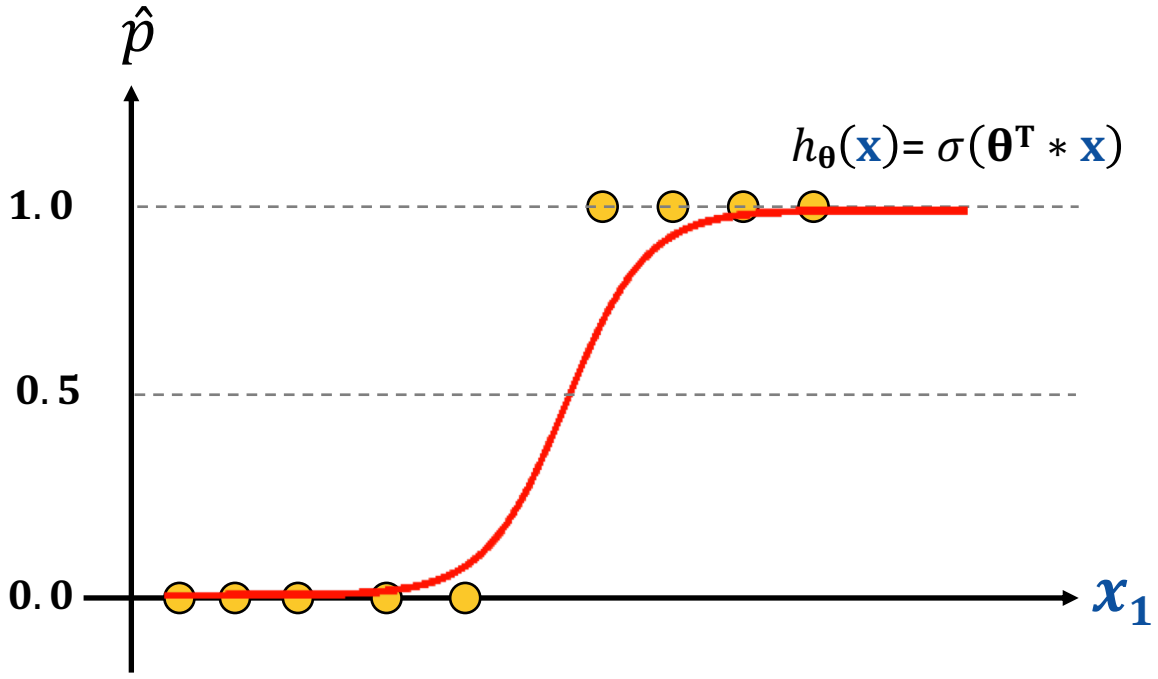

θ0 = -5.73, θ1 = 1.25
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10


θ0 = -5.5, θ1 = 1.25
1.00
0.75
0.50
0.25
0.00
0
2
4
6
8
10

Decision Boundary



Decision Boundary



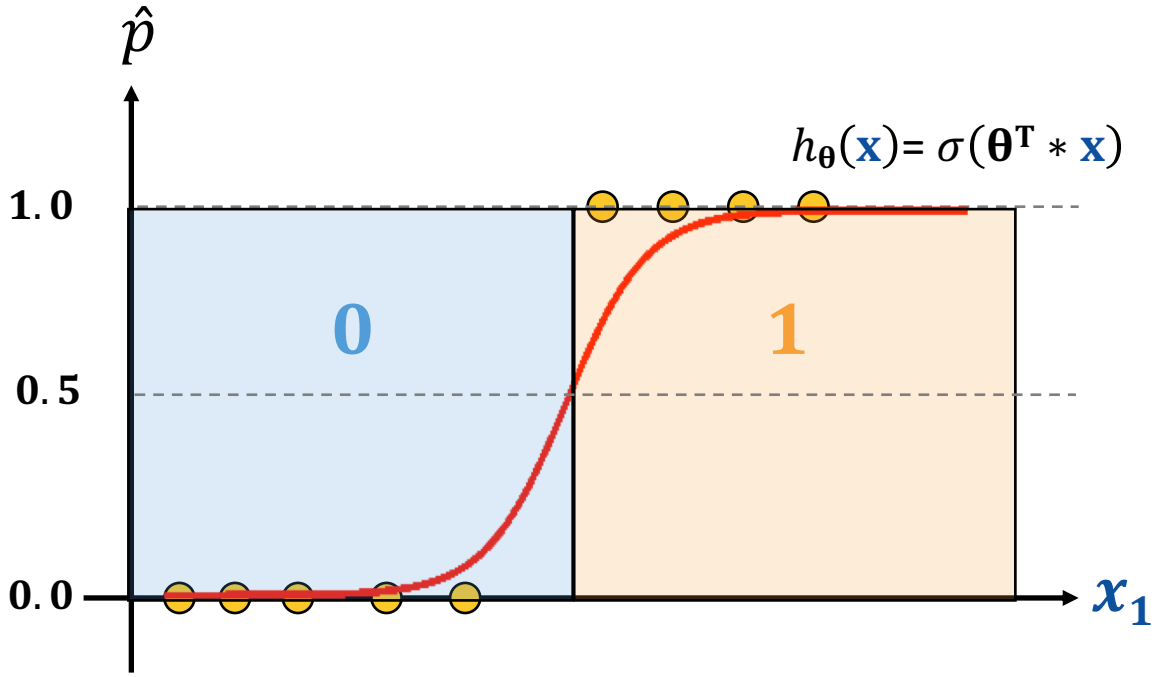
$\sigma(z)$ returns a value between **0** and **1**

$\sigma(z) < 0.5$ if $z < 0$ (negative class)

$\sigma(z) \geq 0.5$ if $z \geq 0$ (positive class)

$\sigma(z) = 0.5$ if $z = 0$

Decision Boundary



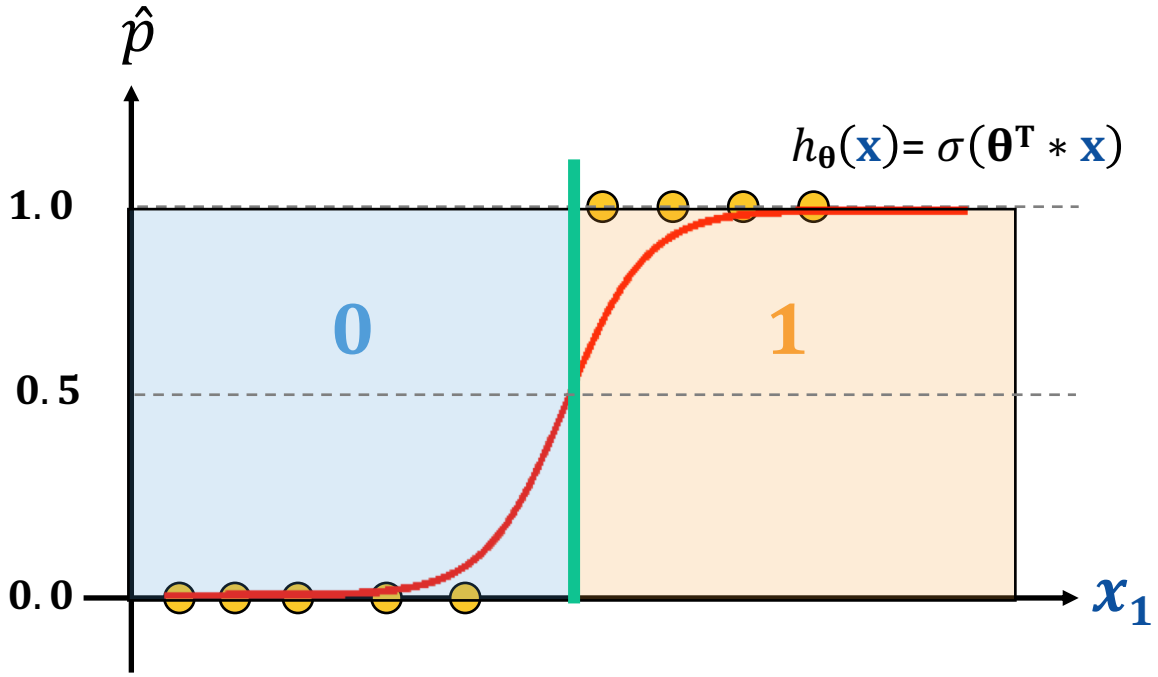
$\sigma(z)$ returns a value between **0** and **1**

$\sigma(z) < 0.5$ if $z < 0$ (negative class)

$\sigma(z) \geq 0.5$ if $z \geq 0$ (positive class)

$\sigma(z) = 0.5$ if $z = 0$

Decision Boundary



$\sigma(z)$ returns a value between **0** and **1**

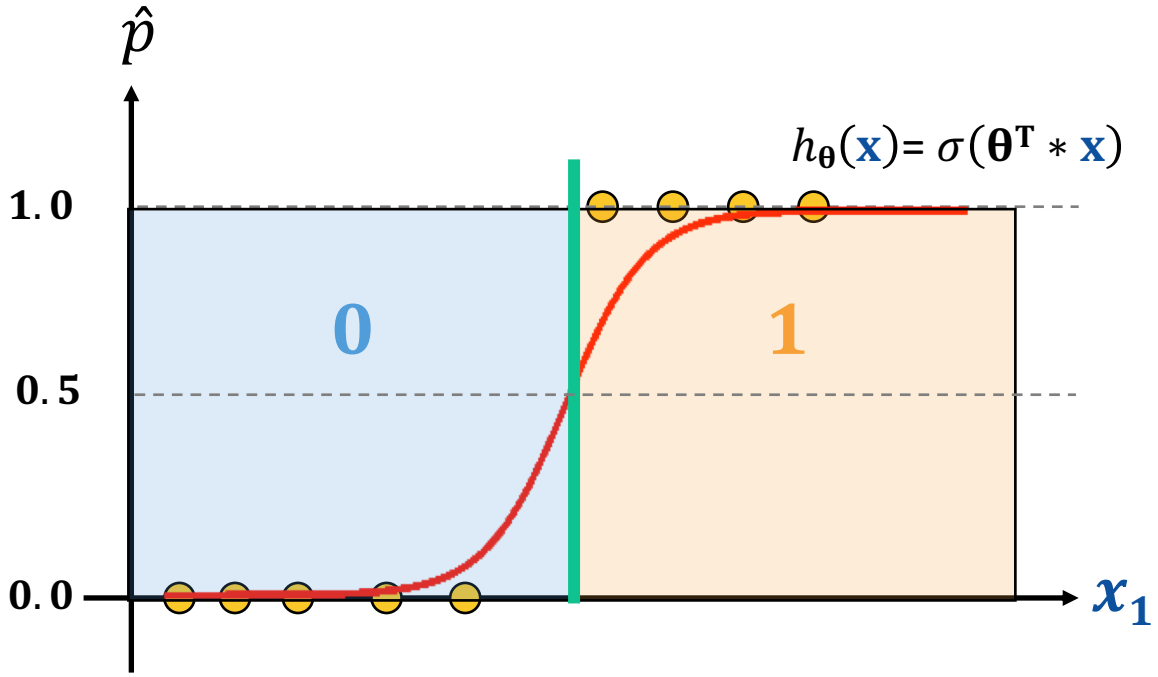
$\sigma(z) < 0.5$ if $z < 0$ (negative class)

$\sigma(z) \geq 0.5$ if $z \geq 0$ (positive class)

$\sigma(z) = 0.5$ if $z = 0$

Thus, the hyperplane $(\theta^T * x) = 0$ is the ***decision boundary*** of the classifier..

Decision Boundary



$\sigma(z)$ returns a value between **0** and **1**

$\sigma(z) < 0.5$ if $z < 0$ **(negative class)**

$\sigma(z) \geq 0.5$ if $z \geq 0$ **(positive class)**

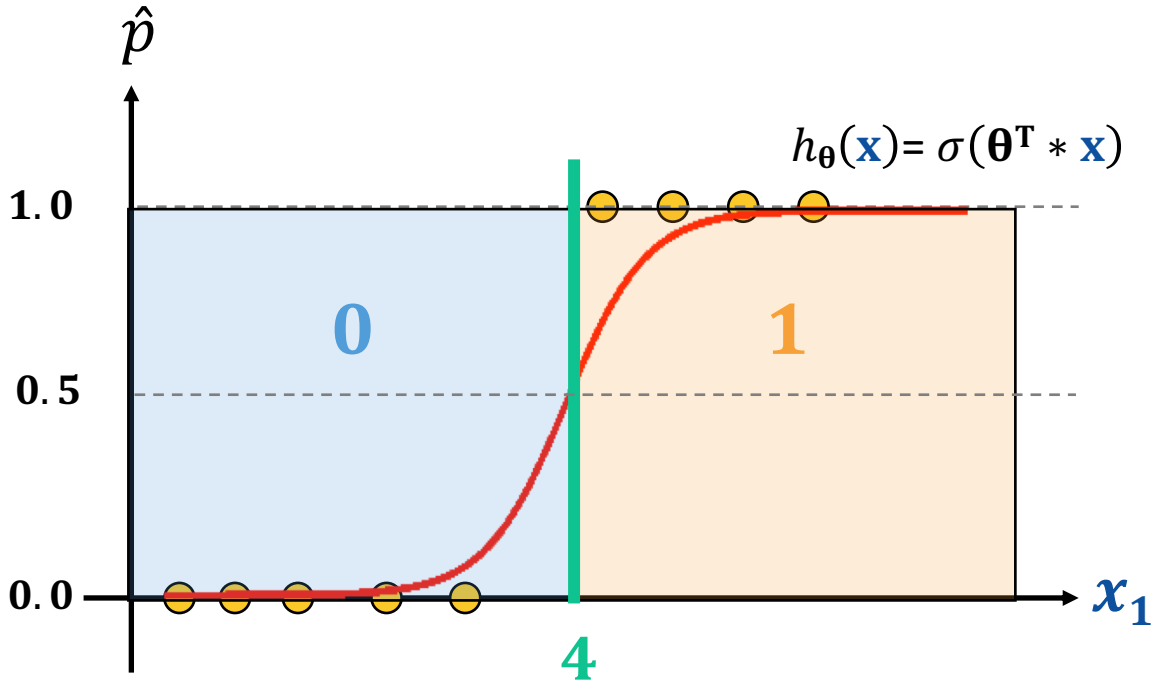
$\sigma(z) = 0.5$ if $z = 0$

Thus, the hyperplane $(\theta^T * x) = 0$ is the ***decision boundary*** of the classifier..

Suppose the **best parameter vector θ** is:

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} -4 \\ 1 \end{bmatrix} \qquad x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \end{bmatrix}$$

Decision Boundary



$\boldsymbol{\sigma(z)}$ returns a value between **0** and **1**

$\boldsymbol{\sigma(z) < 0.5}$ if $\mathbf{z < 0}$ **(negative class)**

$\boldsymbol{\sigma(z) \geq 0.5}$ if $\boldsymbol{z \geq 0}$ **(positive class)**

$\boldsymbol{\sigma(z) = 0.5}$ if $\boldsymbol{z = 0}$

Thus, the hyperplane $(\boldsymbol{\theta}^T * \mathbf{x}) = 0$ is the ***decision boundary*** of the classifier..

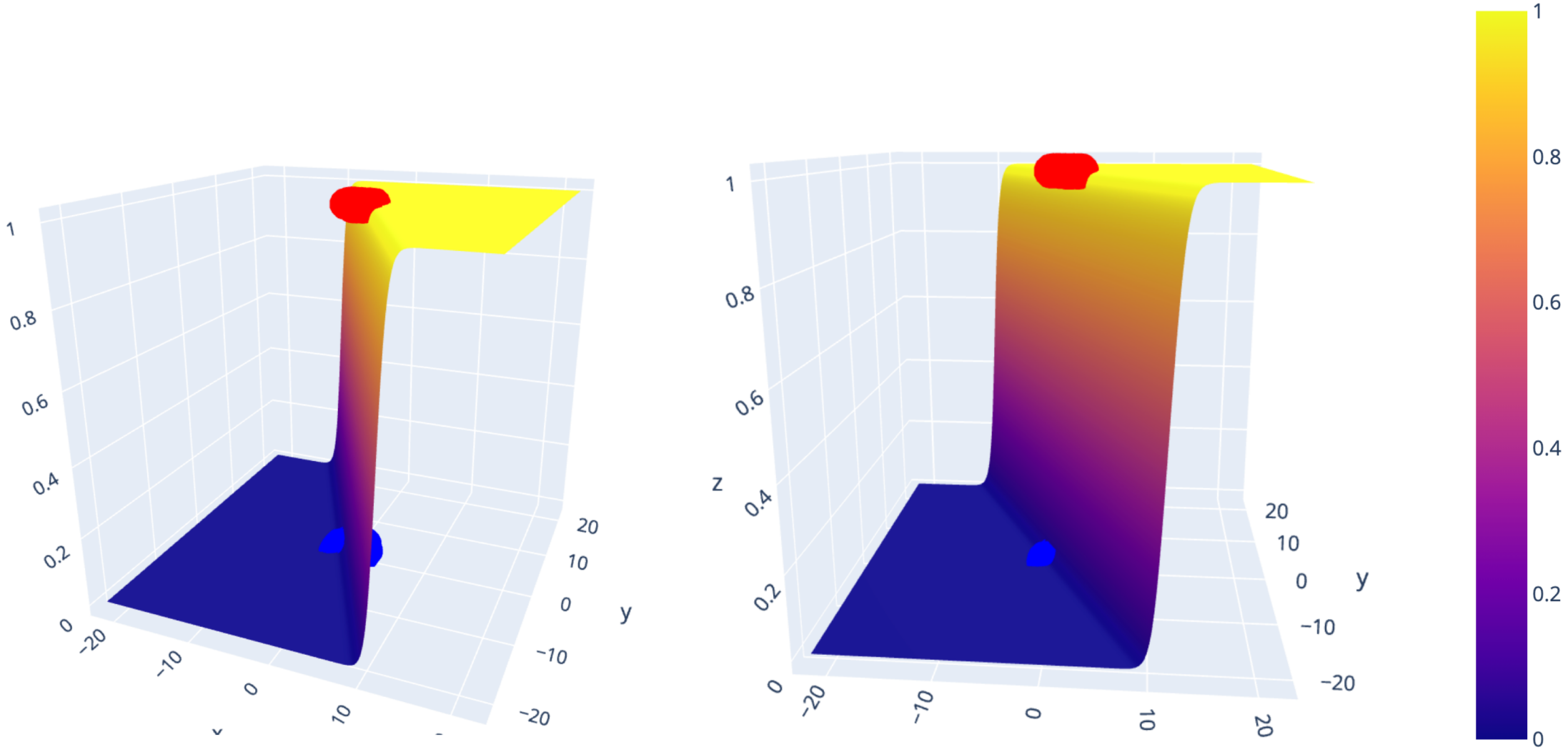
Suppose the **best parameter vector θ** is:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} -4 \\ 1 \end{bmatrix} \qquad \mathrm{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \end{bmatrix}$$

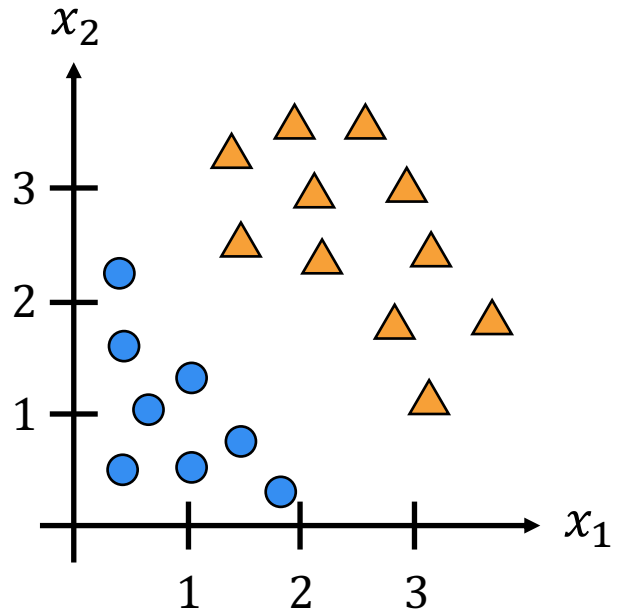
$$\begin{aligned} \boldsymbol{\theta}^T * \mathbf{x} &= 0 \\ -4 + 1 * x_1 &= 0 \\ x_1 &= 4 \end{aligned}$$

decision boundary

Decision Boundary for two features



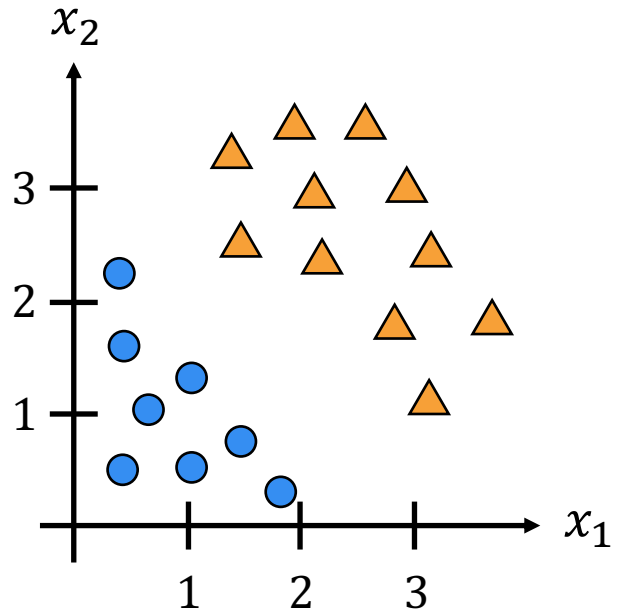
Decision Boundary for two features



$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$

The hyperplane $(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = 0$ is the ***decision boundary*** of the classifier.

Decision Boundary for two features



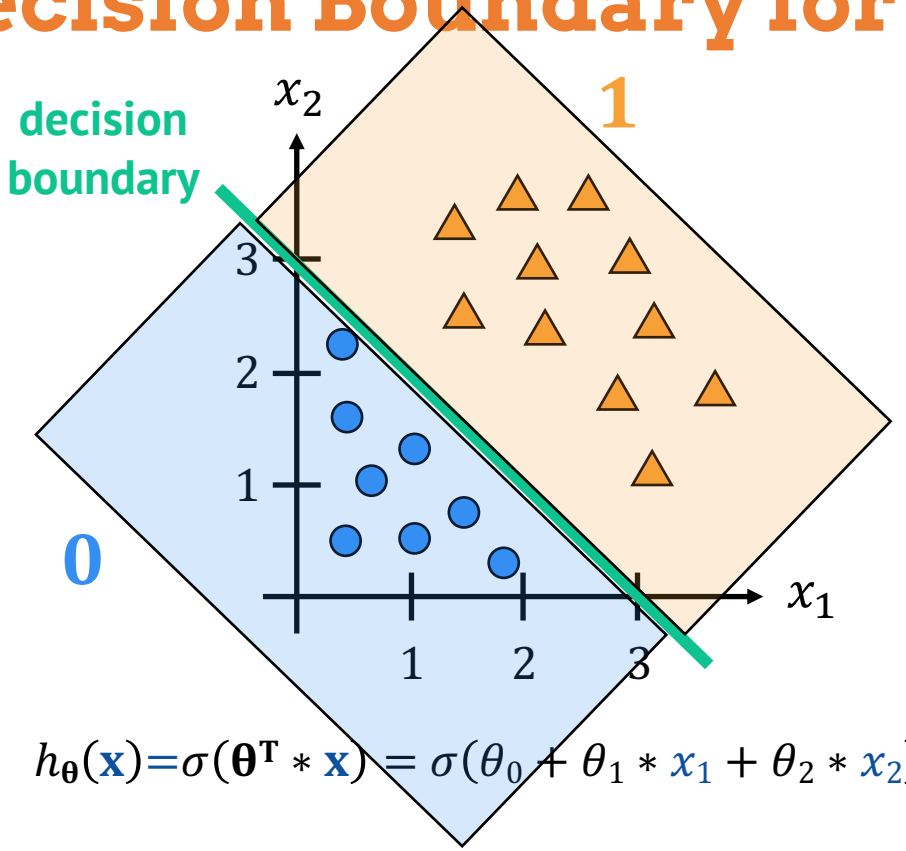
The hyperplane $(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = 0$ is the ***decision boundary*** of the classifier.

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$$

Suppose the **best parameter vector θ** is:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

Decision Boundary for two features



$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$

Suppose the **best parameter vector θ** is:

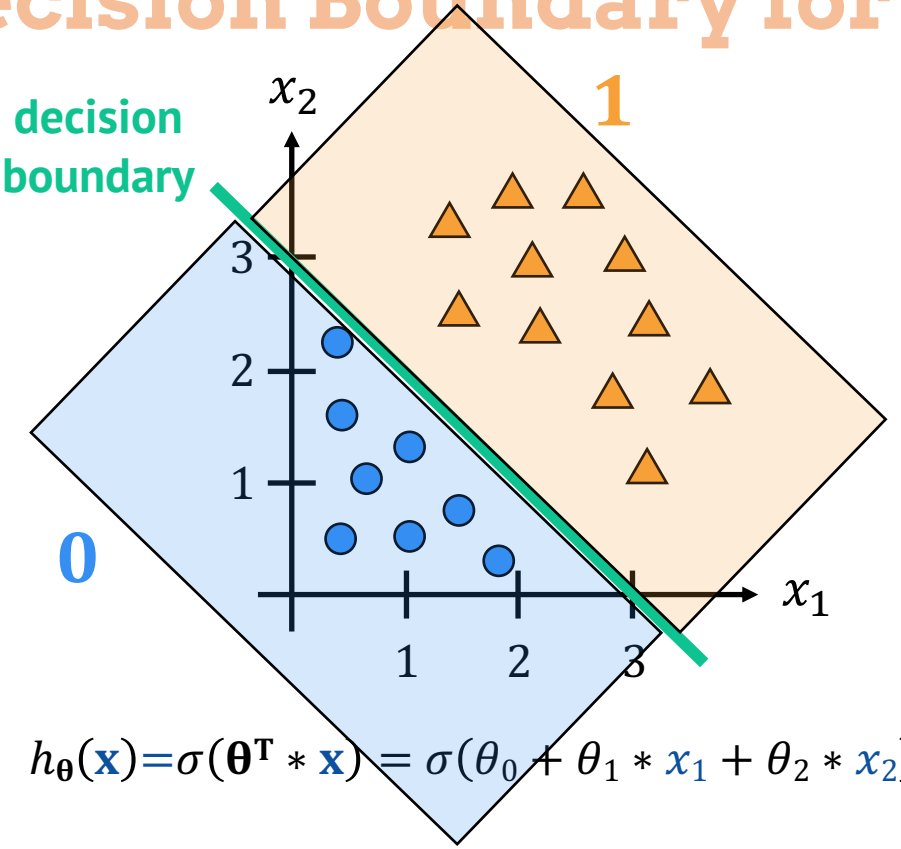
$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

The hyperplane $(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = 0$ is the ***decision boundary*** of the classifier.

$$\begin{aligned} \boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x} &= 0 \\ -3 + 1 * x_1 + 1 * x_2 &= 0 \\ x_1 + x_2 &= 3 \end{aligned}$$

decision boundary

Decision Boundary for two features



$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$

Suppose the **best parameter vector θ** is:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

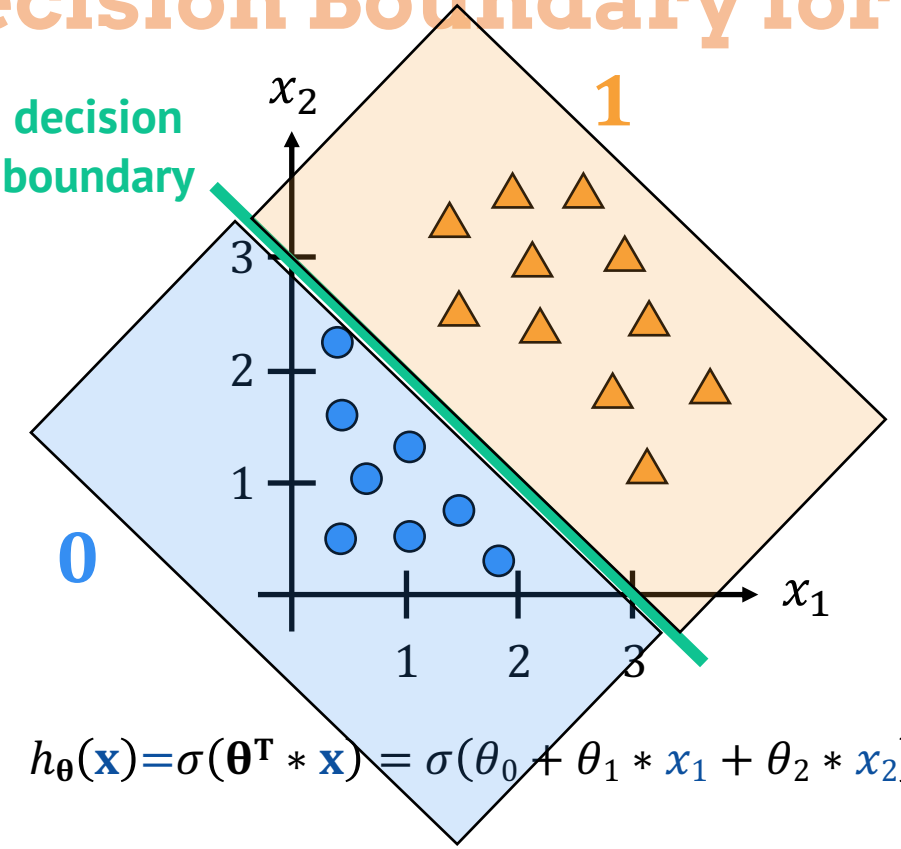
Logistic Regression is a **linear classifier** since its estimated **decision boundary** is **linear**.

fier.

$$\begin{aligned} \boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x} &= 0 \\ -3 + 1 * x_1 + 1 * x_2 &= 0 \\ x_1 + x_2 &= 3 \end{aligned}$$

decision boundary

Decision Boundary for two features



$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$

Suppose the **best parameter vector θ** is:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

Logistic Regression is a **linear classifier** since its estimated **decision boundary** is **linear**.

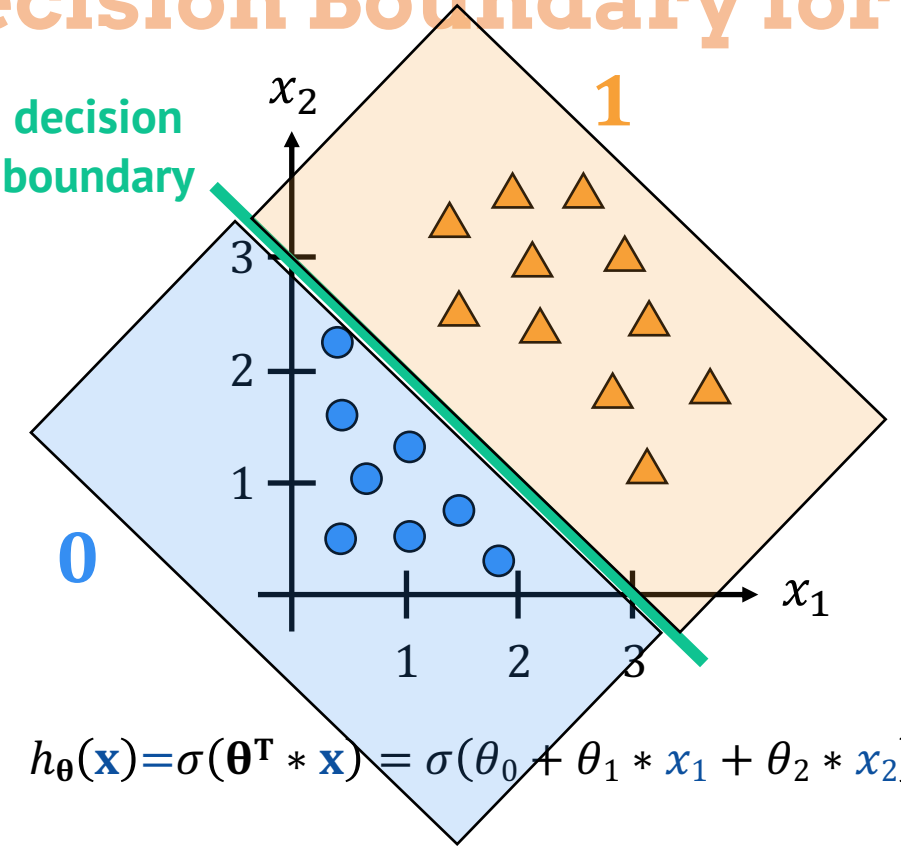
fier.

Logistic Regression is used for **BINARY classification problems.**

$$\begin{aligned} \boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x} &= 0 \\ -3 + 1 * x_1 + 1 * x_2 &= 0 \\ x_1 + x_2 &= 3 \end{aligned}$$

decision boundary

Decision Boundary for two features



$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\mathbf{T}} * \mathbf{x}) = \sigma(\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$

Suppose the **best parameter vector θ** is:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$$

Logistic Regression is a **linear classifier** since its estimated **decision boundary** is **linear**.

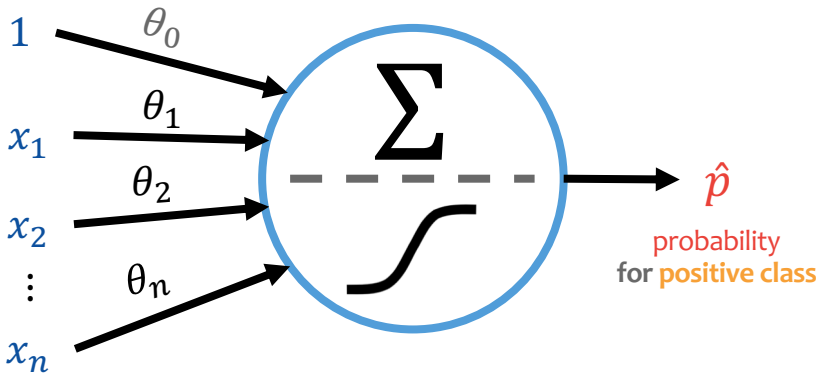
Logistic Regression is used for **BINARY classification problems.**

There are some **strategies** to **extend Logistic Regression** for **multiclass classification.** We'll see them soon!

Logistic Regression and Artificial Neurons

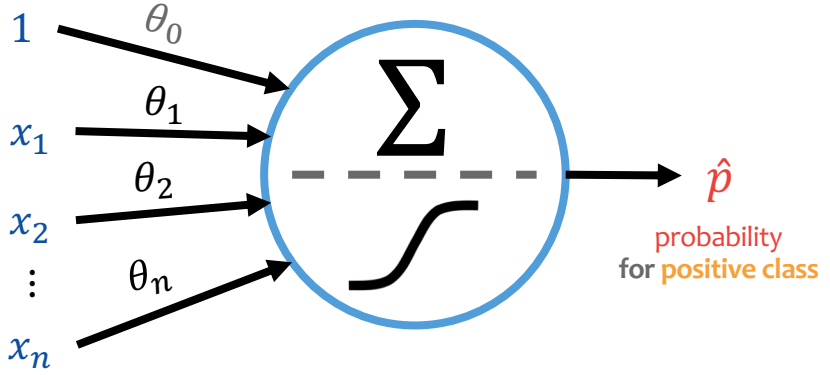
Logistic Regression and Artificial Neurons

A **Logistic Regression** classifier is exactly an **Artificial Neuron (Perceptron)** with the **sigmoid** as activation function.

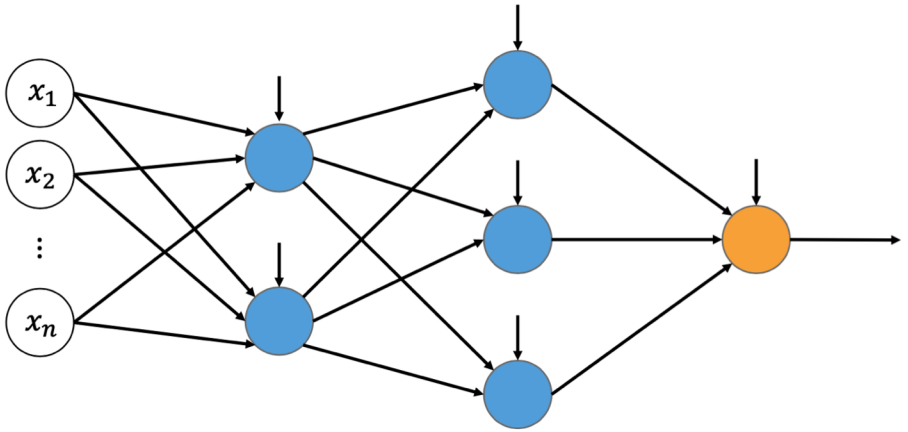


Logistic Regression and Artificial Neurons

A **Logistic Regression** classifier is exactly an **Artificial Neuron (Perceptron)** with the **sigmoid** as activation function.



Artificial Neurons are the basic units of **Artificial Neural Networks**.



Hands-on

D1EAD – Análise Estatística para Ciência de Dados

Logistic Regression

Lecture based on
Machine Learning course
by Andrew Ng
(Coursera)

Prof. Samuel Martins (Samuka)
samuel.martins@ifsp.edu.br

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
São Paulo
Campus Campinas