CS 4104
APPLIED MACHINE LEARNING

Dr. Hashim Yasin

National University of Computer and Emerging Sciences,

Faisalabad, Pakistan.

LINEAR REGRESSION (RECALL)

Summary

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Objective: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Update rules:

$$\theta_0 \coloneqq \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_1 \coloneqq \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

Derivatives:

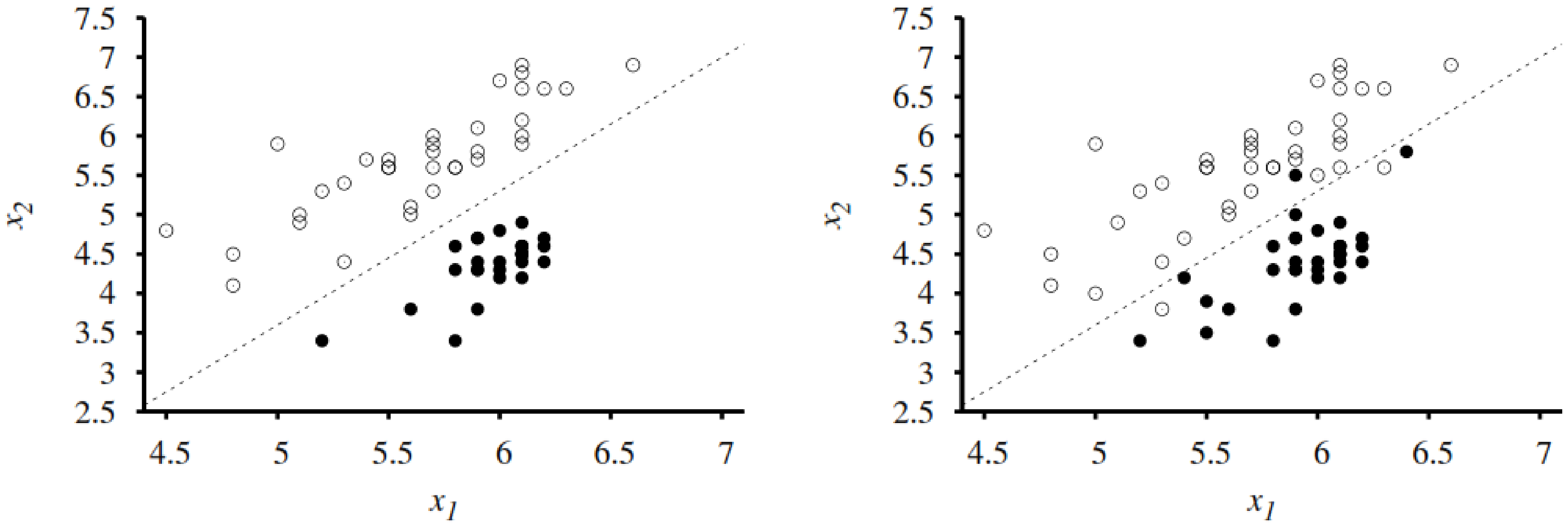
$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \cdot x^{(i)}$$

LINEAR FUNCTION

Linear Function

- **Linear functions** can be used to do classification as well as regression.



Linear Function

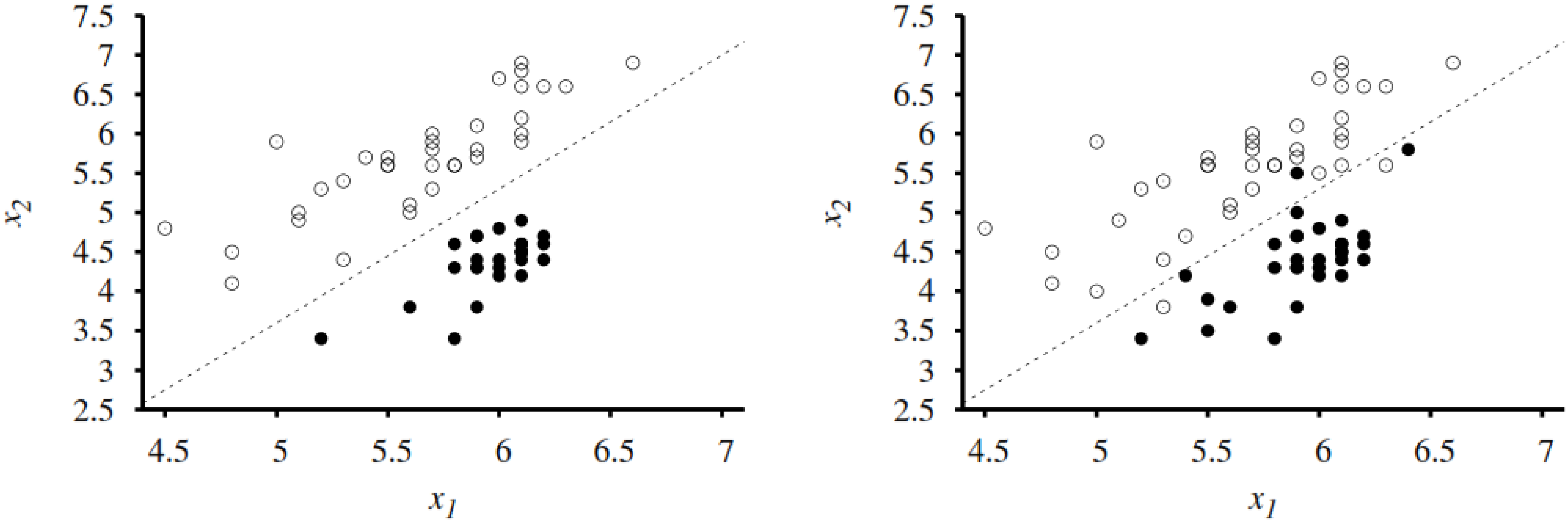
- Given these training data, the task of classification is to learn a hypothesis h.
- A **decision boundary** is a line (or a surface, in higher dimensions) that separates the two classes.
- A linear decision boundary is called a **linear separator** and data that admit such a separator are called **linearly separable.**

Linear Function

- In this case, the **linear separator** is

$$x_2 = 1.7x_1 - 4.9$$

$$-x_2 + 1.7x_1 - 4.9 = 0$$



Linear Function

- For **Class A**, the right of this line with higher values of x_1 and lower values of x_2,

$$-x_2 + 1.7x_1 - 4.9 > 0$$

- For **Class B**, the equation would be,

$$-x_2 + 1.7x_1 - 4.9 < 0$$

- The classification hypothesis can be written as

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \begin{cases} 1, & if\ \boldsymbol{\theta}.\mathbf{x} \geq 0 \\ 0, & otherwise \end{cases}$$

Linear Function

- **Alternatively,** we can think of h as the result of passing the linear function $\boldsymbol{\theta}.\mathbf{x}$ through a **threshold function:**

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = threshold(\boldsymbol{\theta}.\mathbf{x})$$

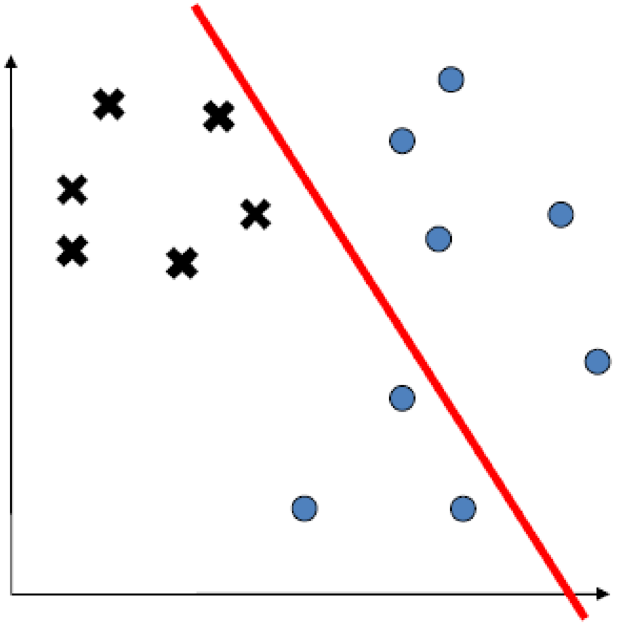
$$threshold(\boldsymbol{\theta}.\mathbf{x}) = \begin{cases} 1 & if \quad \boldsymbol{\theta}.\mathbf{x} \geq t \\ 0 & \quad otherwise \end{cases}$$

$$\begin{cases} \sum_{i=1}^{m} \theta_i x_i > t & output = 1 \\ else & output = 0 \end{cases}$$

Linear Function

$$\begin{cases} \sum_{i=1}^{m} \theta_i x_i > t & output = 1 \\ else & output = 0 \end{cases}$$

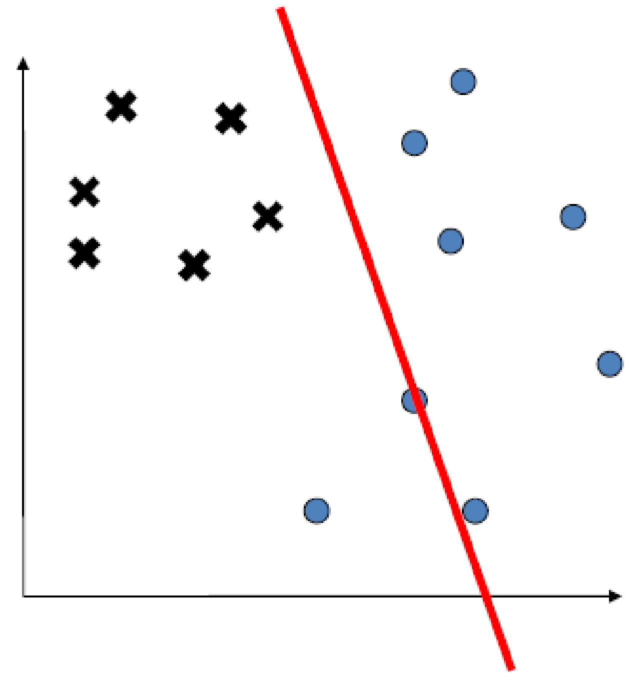
$$\theta_1 = 1, \theta_2 = 0.2, t = 0.05$$



Linear Function

$$\begin{cases} \sum_{i=1}^{m} \theta_i x_i > t & output = 1 \\ else & output = 0 \end{cases}$$

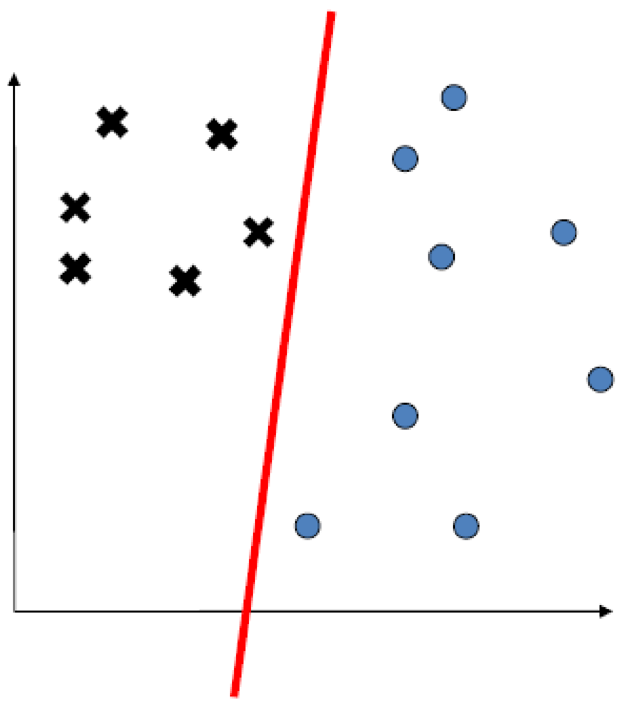
$$\theta_1 = 2.1, \theta_2 = 0.2, t = 0.05$$



Linear Function

$$\begin{cases} \sum_{i=1}^{m} \theta_i x_i > t & output = 1 \\ else & output = 0 \end{cases}$$

$$\theta_1 = -0.8, \theta_2 = 0.03, t = 0.05$$



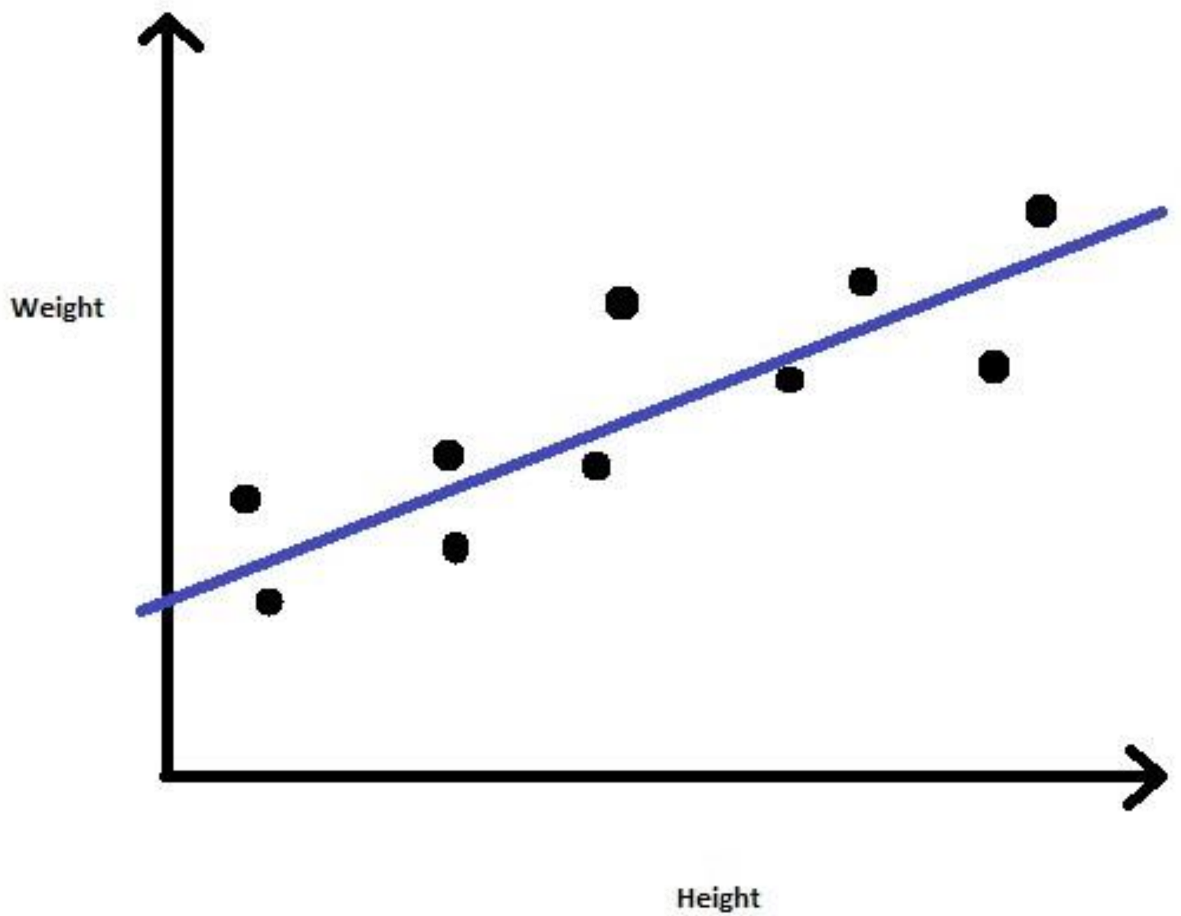
EXAMPLE

Linear Regression

- Let us consider a problem where we are given a dataset containing Height and Weight for a group of people.

- Our task is to predict the Weight for new entries in the Height column.

Linear Regression



Linear Regression

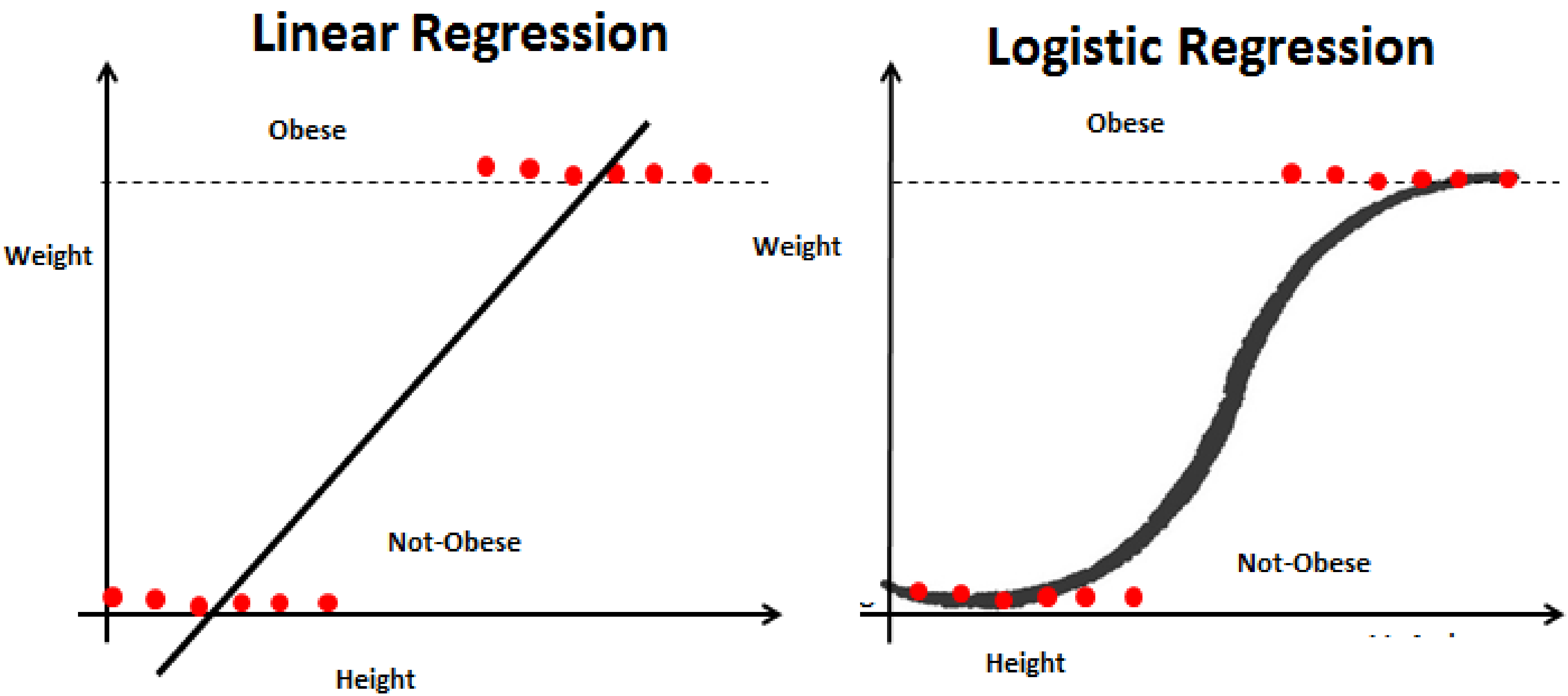
- Let us consider a problem where we are given a dataset containing Height and Weight for a group of people.

- Our task is to predict the Weight for new entries in the Height column.

- Now suppose, we have to classify whether a person is obese or not depending on their provided height and weight.

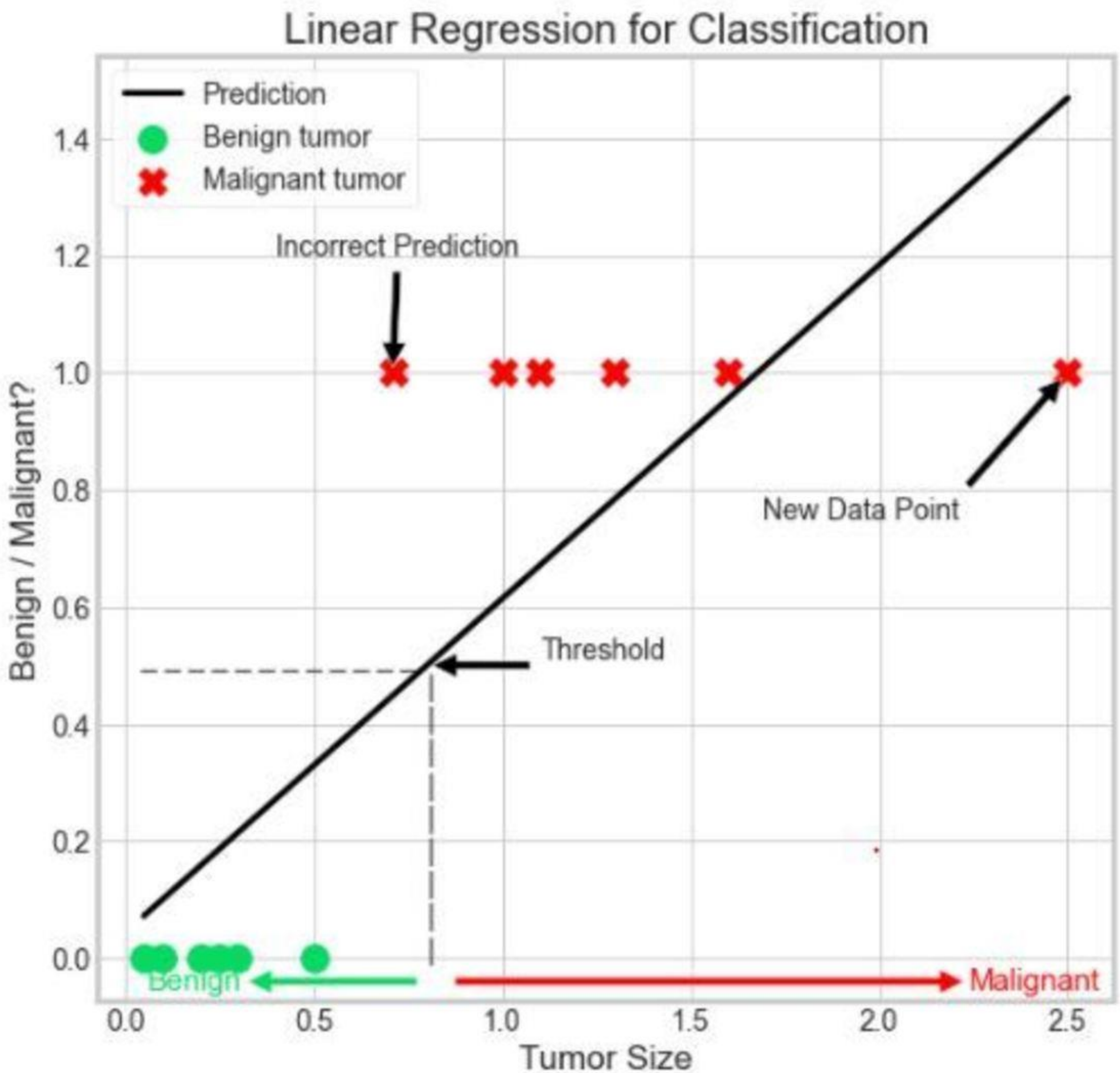
Linear Regression

Find Obese (fat): Yes/No

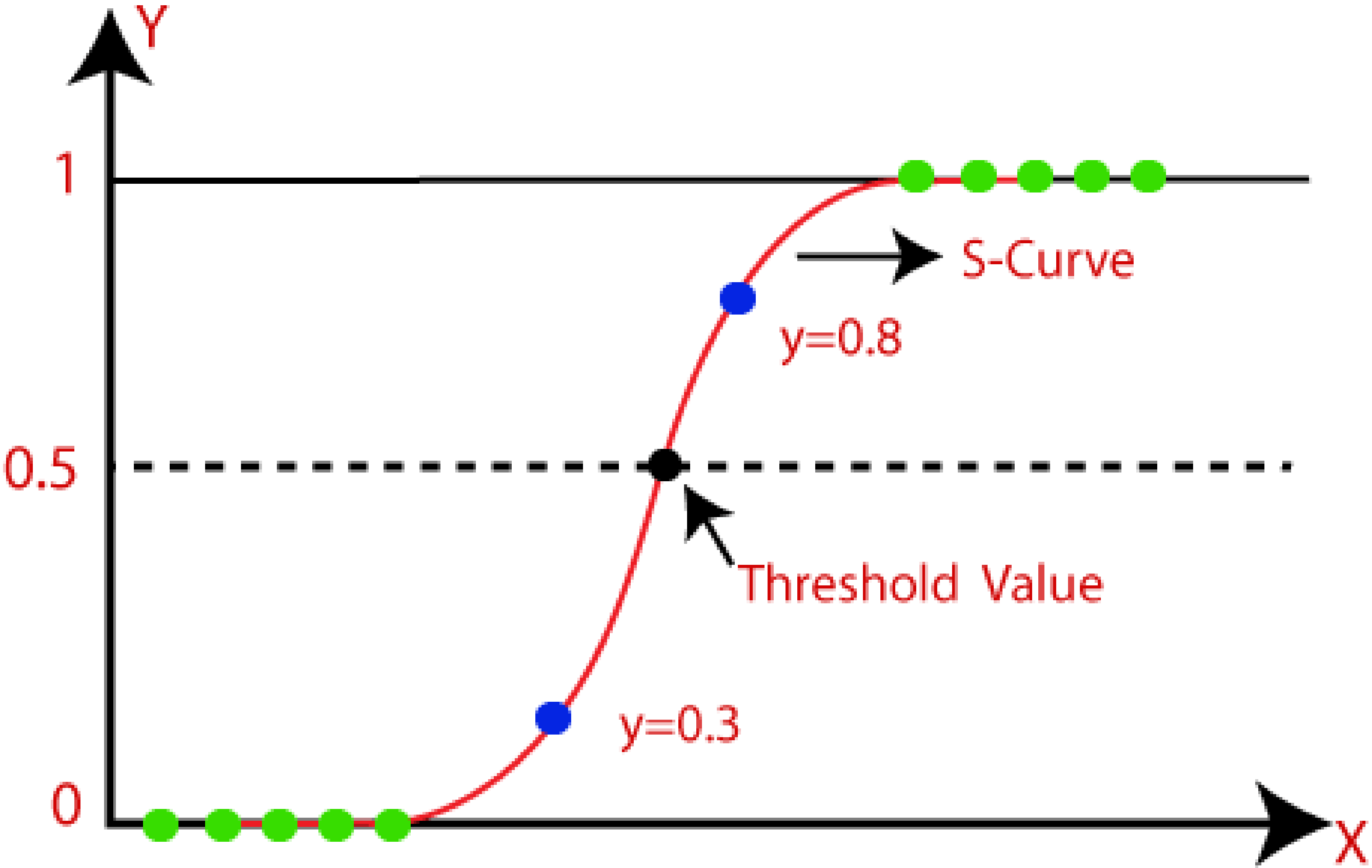


Linear Regression

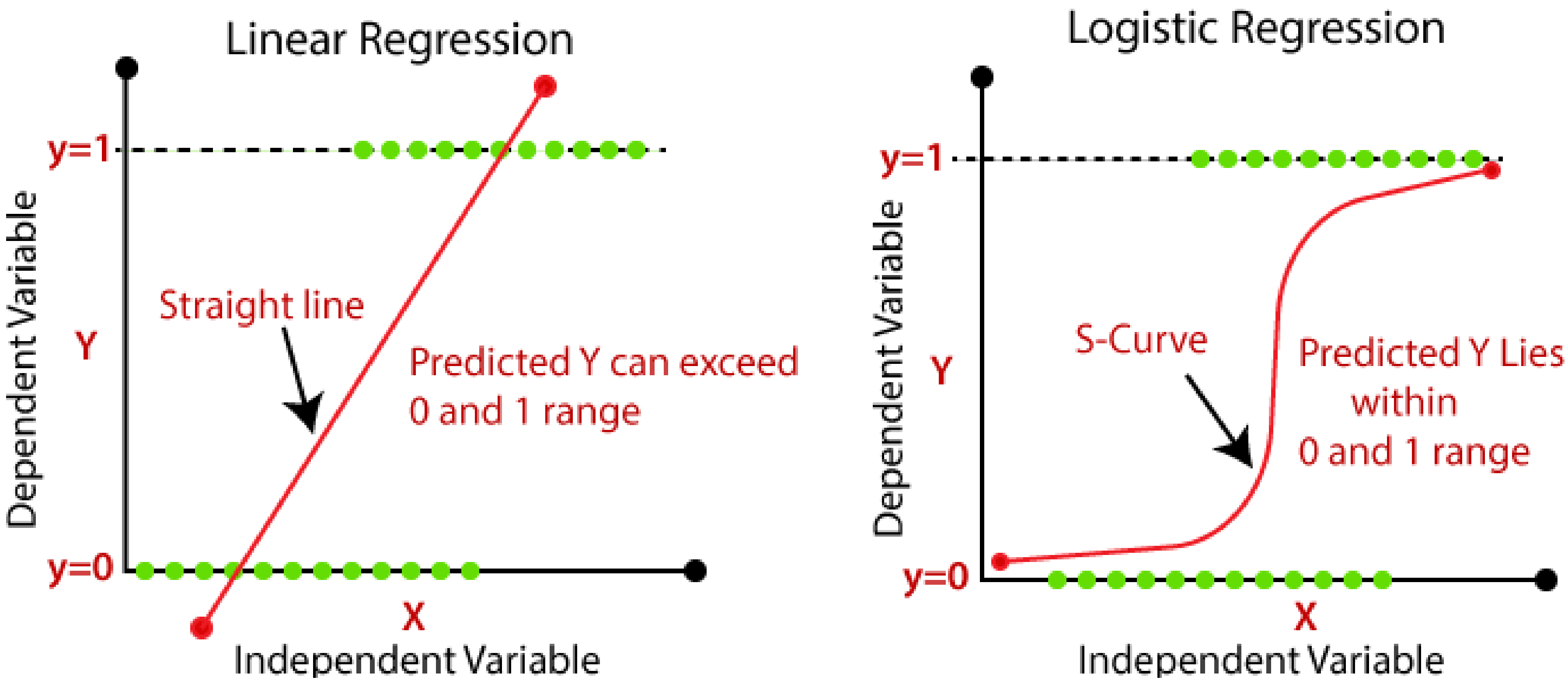
We want the output to be in between 0 and 1



Linear Regression



Linear Regression



LOSS FUNCTIONS

Loss Functions

- **Root Mean Square Error (RMSE)**

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

m - number of samples
x^i - i^th^ sample from dataset
$h_{\boldsymbol{\theta}}(x^i)$ - prediction for i^th^ sample (thesis)
y^i - ground truth label for i^th^ sample

Loss Functions

- **Mean Absolute Error (MAE) (L1 norm)**

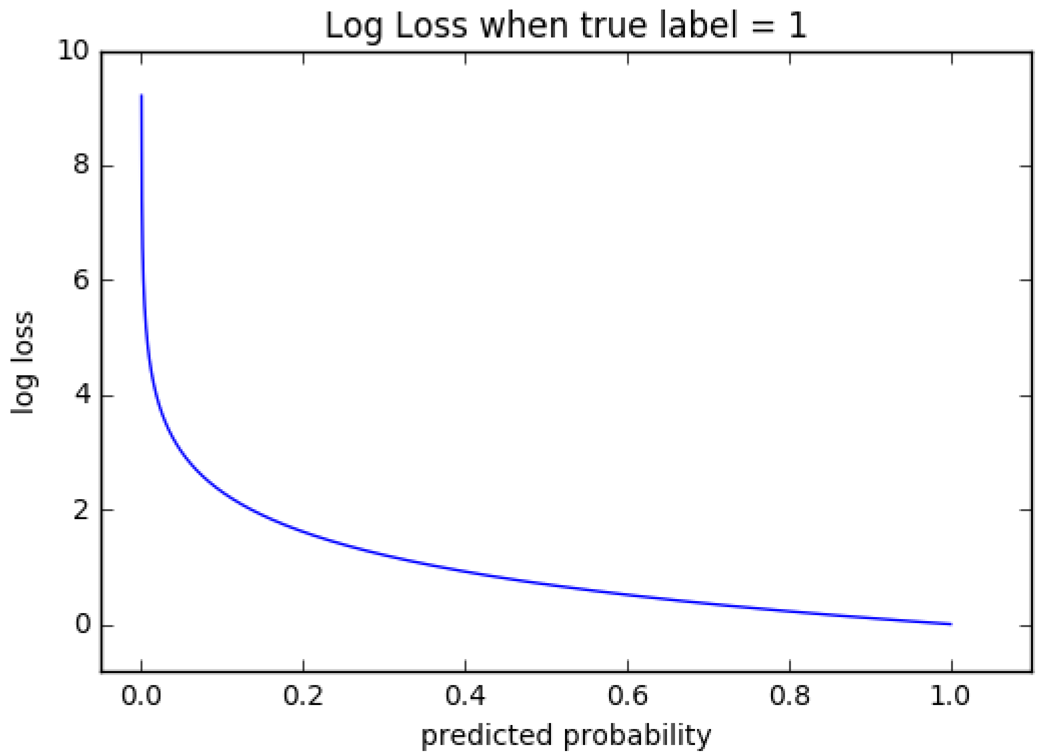
$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left|\left(h_{\boldsymbol{\theta}}\left(x^{i}\right) - y^{i}\right)\right|$$

- **Mean Square Error (MSE) (L2 norm)**

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}\left(x^{i}\right) - y^{i}\right)^{2}$$

Loss Functions

- **Cross-entropy loss error, or log loss,** measures the performance of a classification model whose output is a probability value between 0 and 1.



Loss Functions

- **Cross-entropy loss error, or log loss,** measures the performance of a classification model whose output is a probability value between 0 and 1.

$$-(y\log(p) - (1-y)\log(1-p))$$

- log - the natural log
- y - binary indicator (0 or 1) if class label c is the correct classification for observation o
- p - predicted probability observation o is of class c

Loss Functions

- If there are multiple classes ($M > 2$) (multiclass classification), then

$$-\sum_{c=1}^{M}\left(y_{o,c}\log\left(p_{o,c}\right)\right)$$

- M - number of classes (dog, cat, fish)
- log - the natural log
- y - binary indicator (0 or 1) if class label c is the correct classification for observation o
- p - predicted probability observation o is of class c

LOGISTIC REGRESSION

Logistic Regression

- In previous example, the hypothesis $h(\mathbf{x})$ with some hard threshold is **not differentiable**
- It is in fact a **discontinuous function** of its inputs and its weights.
- As a result, learning becomes very ***unpredictable*** adventure.
- The **linear classifier** always announces a completely **confident prediction of 1 or 0**, even for examples that are very close to the boundary.

Logistic Regression

- All of these issues can be resolved to a large extent by **softening the threshold function**
- In case of **logistic regression**, we approximate the threshold with a ***continuous***, ***differentiable*** function that generate output between 0 and 1

$$0 \leq h_{\boldsymbol{\theta}}(\mathbf{x}) \leq 1$$

Logistic Regression

- The **logistic function** is

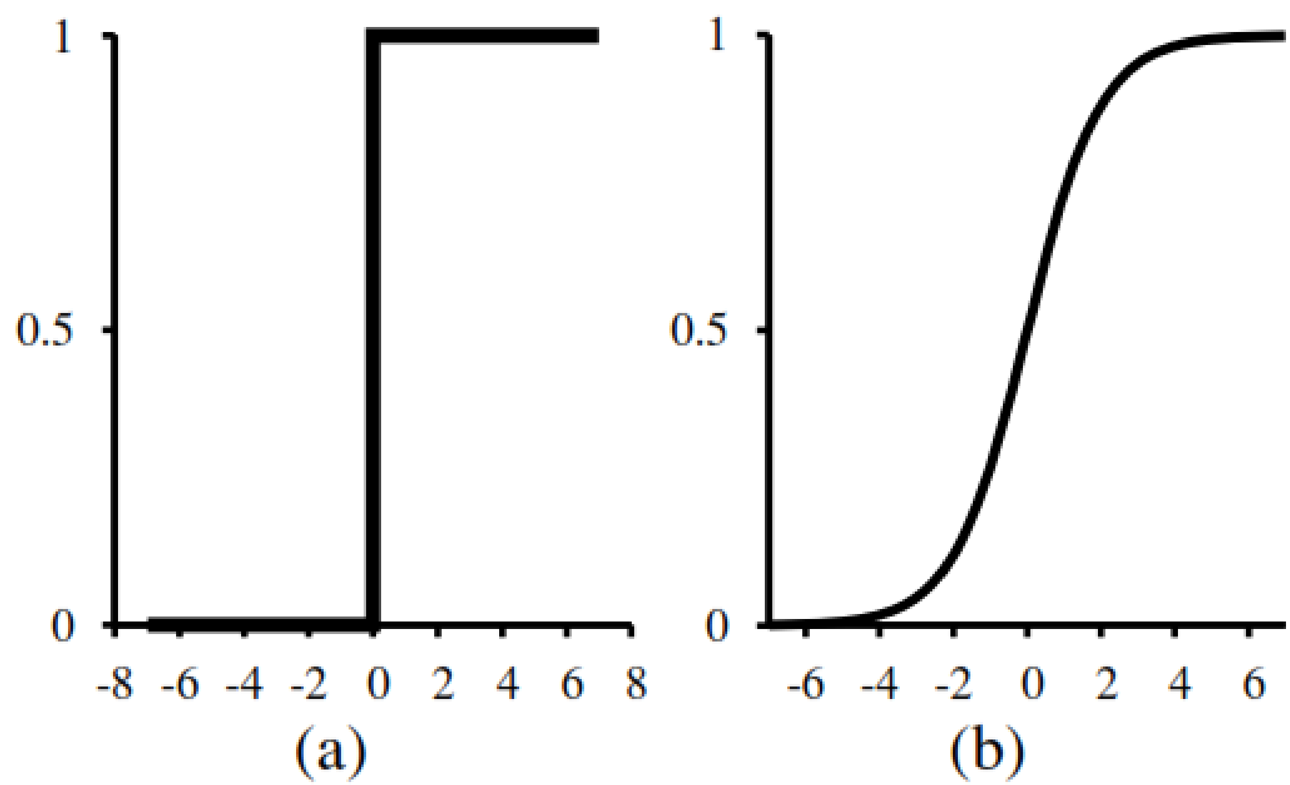
$$g(z) = \frac{e^{z}}{1+e^{z}} = \frac{1}{1+e^{-z}}$$

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

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

Logistic Regression

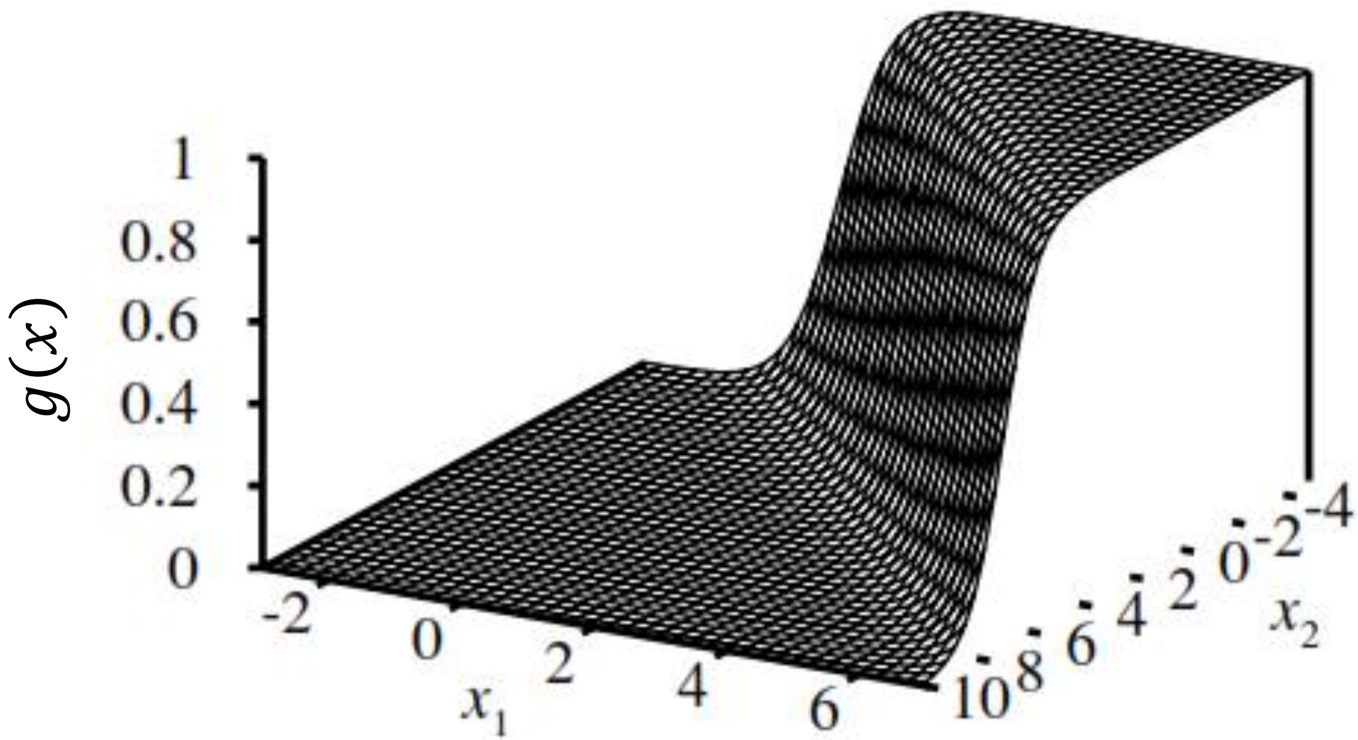
The function is non-differentiable at z=0.



(a) The hard threshold function Threshold with 0/1 output.

(b) The logistic function,

Logistic Regression



Plot of a logistic regression hypothesis

Logistic Regression

Logistic Function

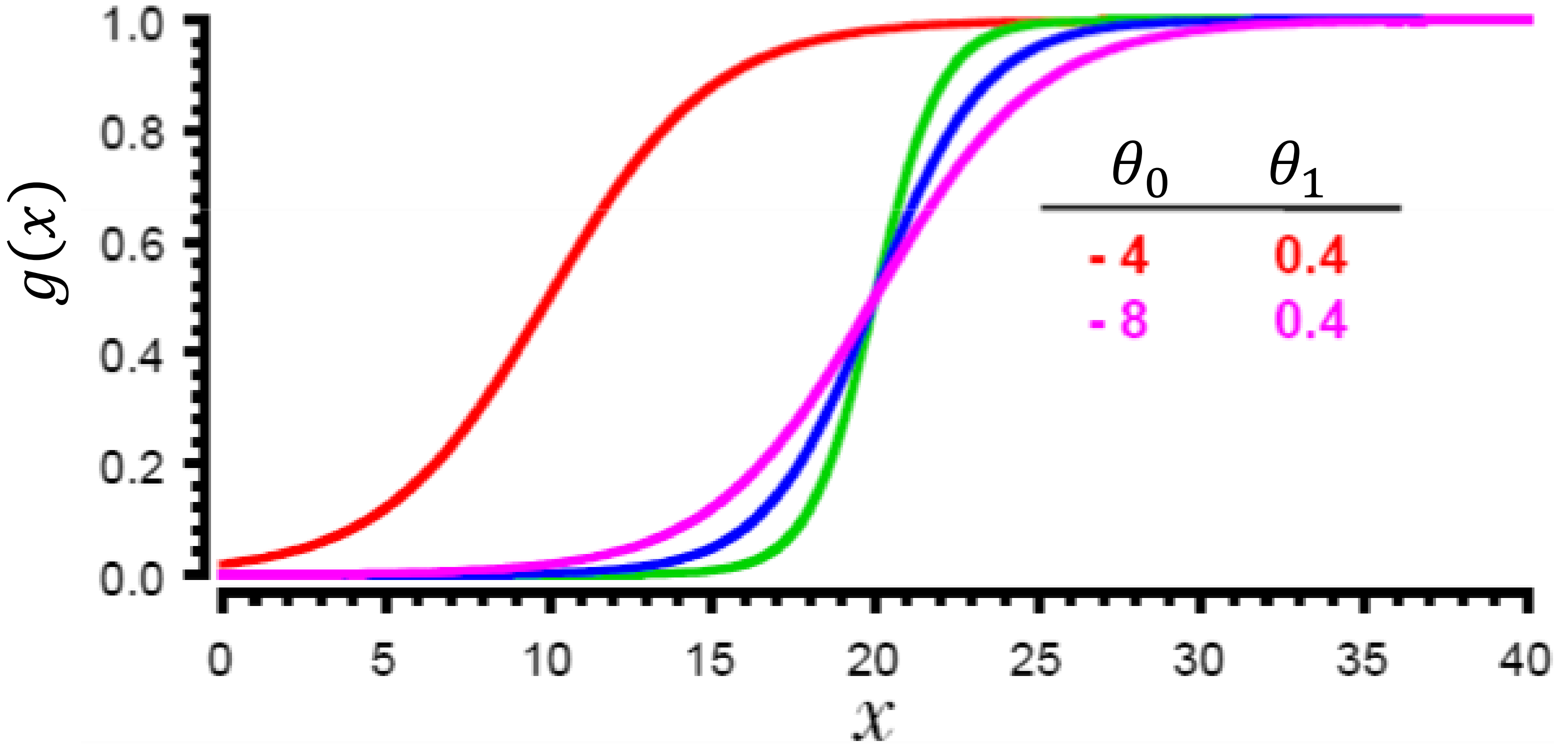
$$\frac{1}{1+e^{-\theta . x}} = h_\theta(\mathbf{x})(1 - h_\theta(\mathbf{x}))$$

$$\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right) \\
&= \sigma(x) \cdot (1 - \sigma(x))
\end{aligned}$$

Logistic Regression

$$g(z) = \frac{e^z}{1+e^z} = \frac{e^{(\theta_0+\theta_1 x)}}{1+e^{(\theta_0+\theta_1 x)}}$$

θ_0 … controls location of the midpoint
θ_1 … controls slope of rise



Logistic Regression

- Parameters control shape and location of logistic/sigmoid curve
- θ_0 … controls location of the midpoint
- θ_1 … controls slope of rise

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

$$= \frac{e^{(\theta_0 + \theta_1 x)}}{1 + e^{(\theta_0 + \theta_1 x)}}$$

Logistic Regression

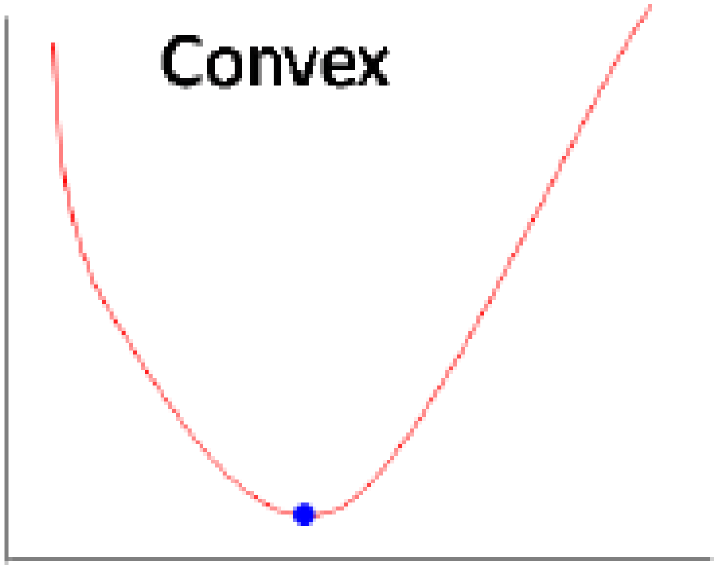
- Notice that the output, *being a number between 0 and 1*, can be interpreted as a ***probability*** of belonging to the class labeled 1.
- If $h(x) \geq 0.5$, the output is "1" otherwise $h(x) < 0.5$ the output is "0".
- The process of fitting the parameters of this model to minimize loss on a dataset is called **logistic regression.**
- There is ***NO easy closed-form solution*** for this model, but the gradient descent computation is straight-forward.

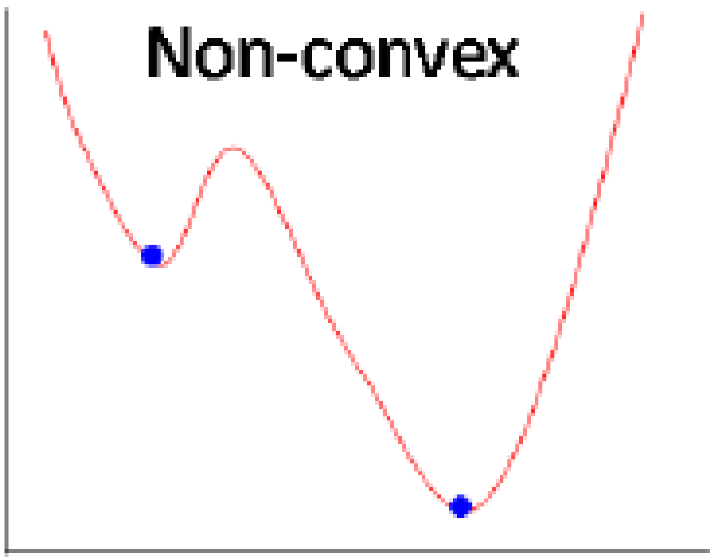
LOGISTIC REGRESSION COST FUNCTION

Cost Function

Linear Regression Cost Function

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





Cost Function

Logistic Regression Cost Function

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

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

For y = 1;

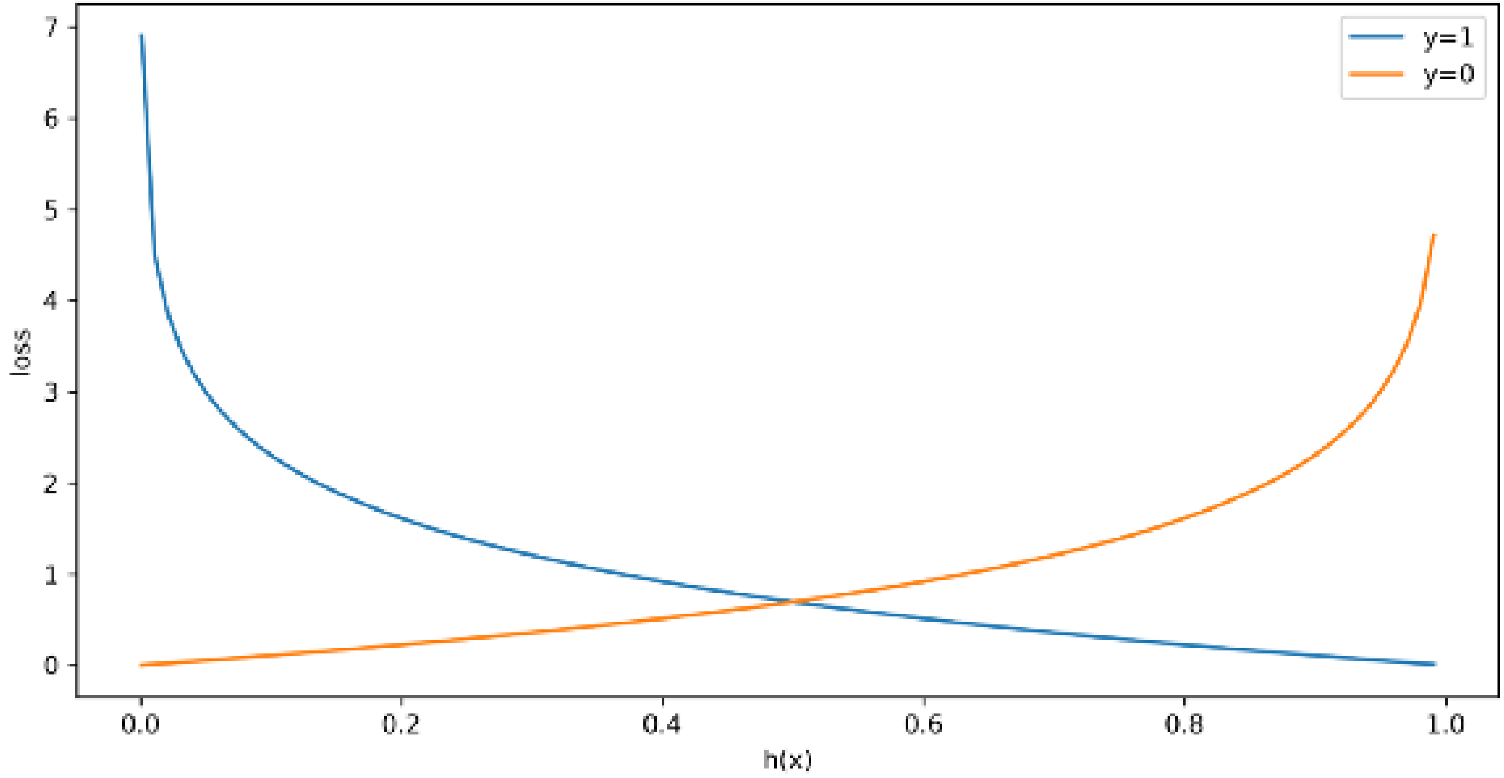
$$cost(h_\theta(x^i), y^i) = -\log(h_\theta(x^i))$$

For y = 0;

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

Cost Function

Logistic Regression Cost Function



Cost Function

Logistic Regression Cost Function

$$\text{cost}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}),\ y\right) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } \ y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } \ y = 0 \end{cases}$$

$$\text{cost}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}),\ y\right) = -y\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) - (1-y)\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))$$

$$\begin{aligned} &\text{if } y = 1: \quad \text{cost}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y\right) = -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) \\ &\text{if } y = 0: \quad \text{cost}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y\right) = -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) \end{aligned}$$

$h_{\theta}(x)$ $h_{\theta}(x)$

Cost Function

Logistic Regression Cost Function

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

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

$$m = number\ of\ samples$$

Gradient Descent

Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

Gradient Descent

Gradient descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(simultaneously update } j = 0 \text{ and } j = 1\text{)}$$

}

Notice : α is the learning rate.

?

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Gradient Descent

Gradient descent algorithm

$$\text{repeat until convergence } \{$$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j = 0 \text{ and } j = 1)$$

$$\}$$

Correct: Simultaneous update

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\theta_1 := \text{temp1}$$

Incorrect:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_1 := \text{temp1}$$

Gradient Descent Search

$\boldsymbol{\theta} \leftarrow$ any point in the parameter space

loop until convergence **do**

for each θ_j **in** $\boldsymbol{\theta}$ **do**

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

- The parameter α is the **step size** or the **learning rate** in a learning problem.
- It can be a fixed constant, or it can decay over time as the learning process proceeds.

Gradient Descent Search

$\boldsymbol{\theta} \leftarrow$ any point in the parameter space

loop until convergence **do**

for each θ_j **in** $\boldsymbol{\theta}$ **do**

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

- **Surprisingly*****, the update rule is the same as the one derived by using the sum of the squared errors in linear regression***.
- As a result, we can use the same gradient descent formula for logistic regression as well.

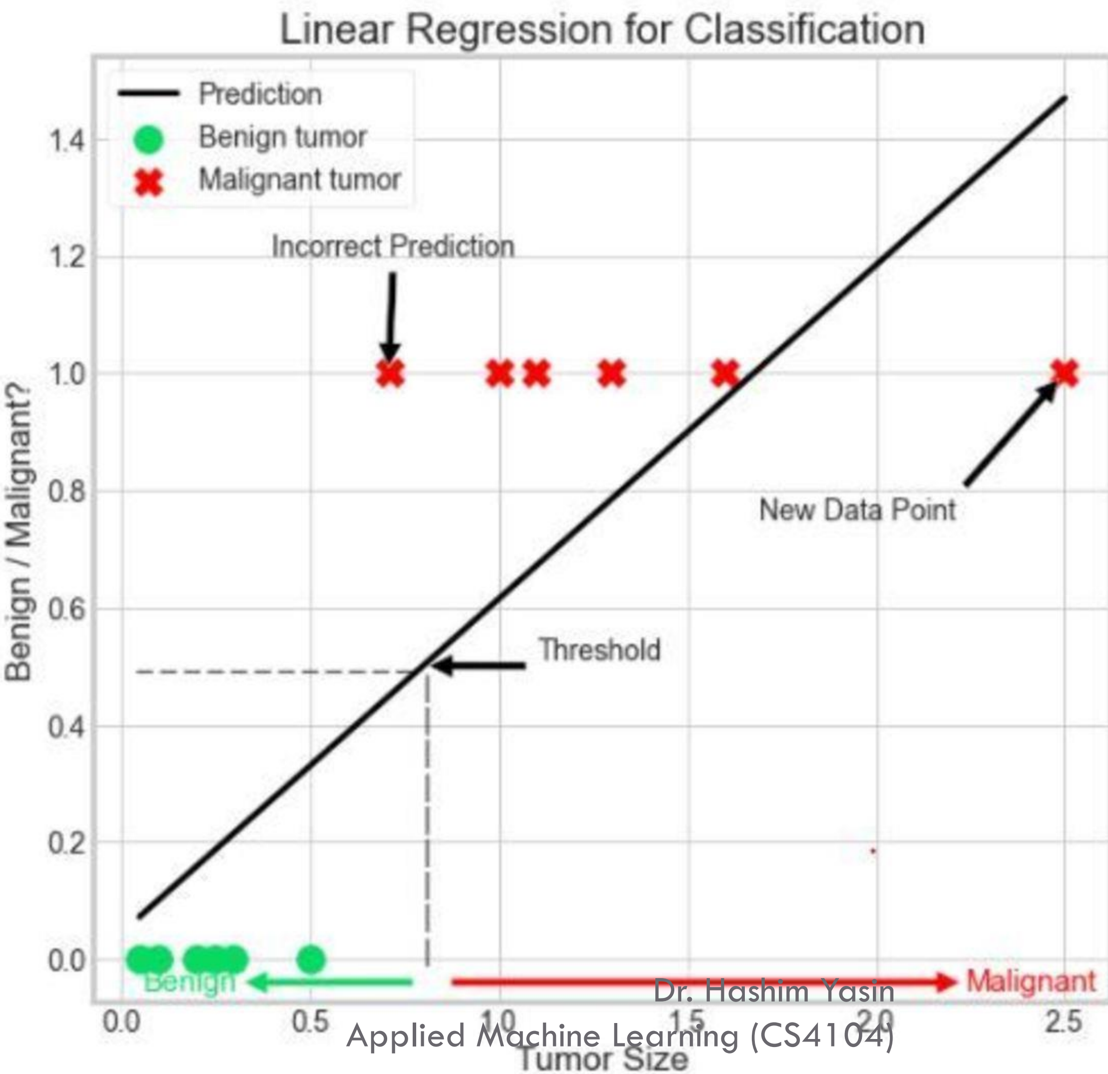
LINEAR VS LOGISTIC REGRESSION

Linear vs Logistic Regression

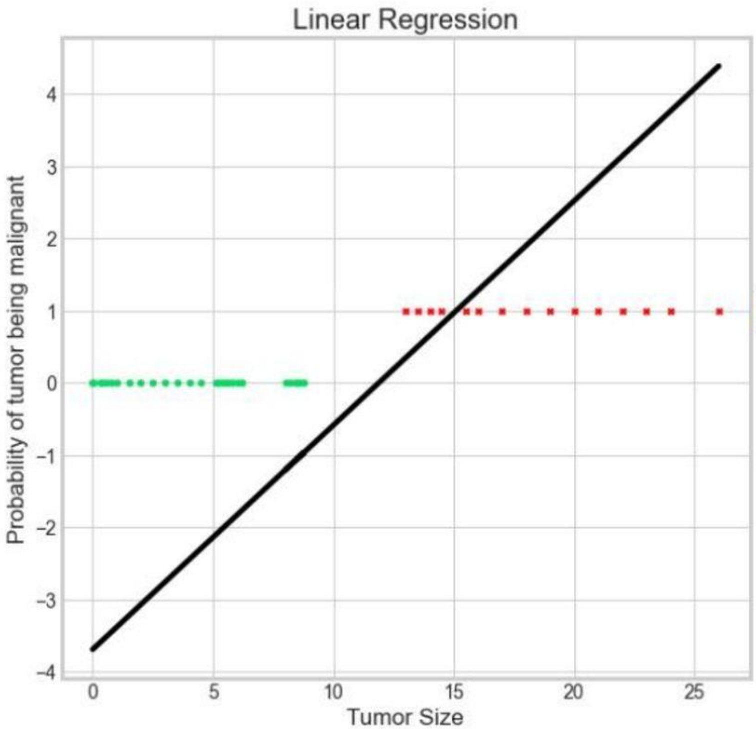
- Linear Regression is Not Suitable for Classification
- Linear regression faces two major problems:
  - Regression outputs continuous values which ***CANNOT be treated as pure probabilistic***.
  - There is an *unwanted shift in the threshold value* when new data points are added.

Linear vs Logistic Regression

We want the output to be in between 0 and 1

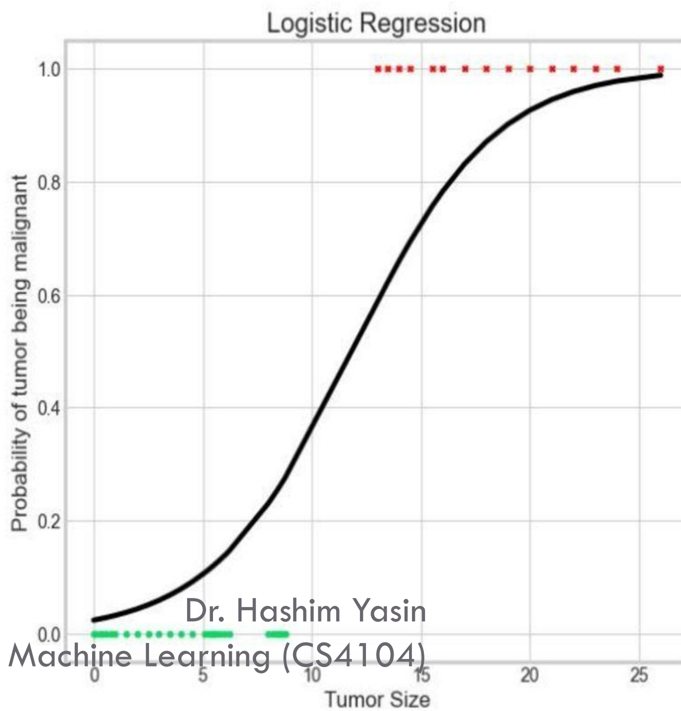


Linear vs Logistic Regression



Output:

$$h_\theta(x) = \theta_0 + \theta_1 x$$



Output:

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

$$= \frac{1}{1 + e^{-(\theta_0 + \theta_1 x)}}$$

Linear vs Logistic Regression

- Linear regression does not work well with classification problems.
- Logistic regression uses the logistic function which squashes the output range between 0 and 1.
- Logistic regression makes use of hypothesis function of the linear regression algorithm.

Linear vs Logistic Regression

- **Linear Regression** is used to handle regression problems whereas **Logistic regression** is used to handle classification problems.

- **Linear regression** provides a continuous output, but **Logistic regression** provides probabilities (discreet output).

- The method for calculating loss function in linear regression is the mean squared error whereas for logistic regression it is **maximum likelihood estimation**.

Acknowledgement

Tom Mitchel, Russel & Norvig, Andrew Ng, Alpydin & Ch. Eick.