Logistic Regression

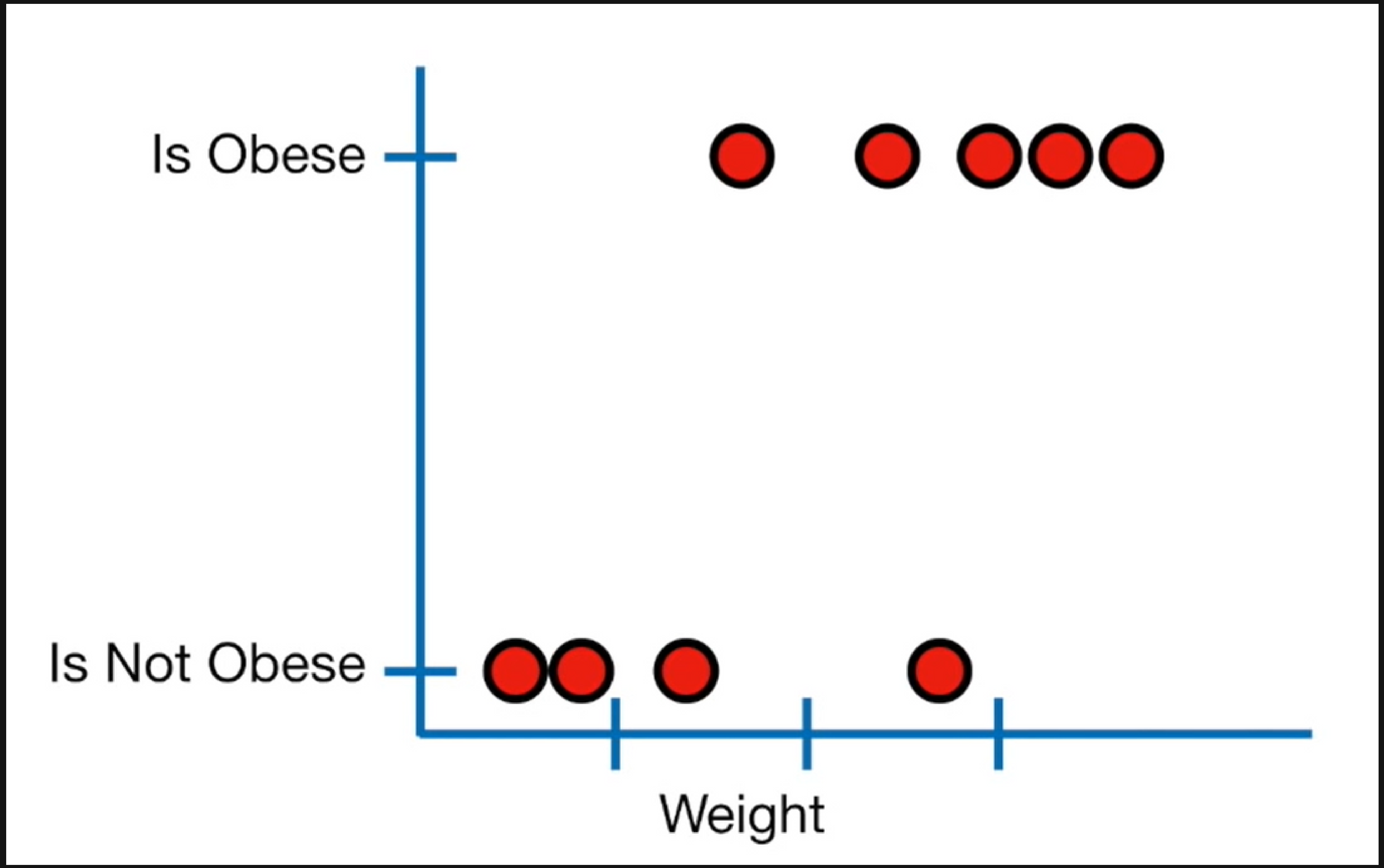
Unravelling the Power of Classification

CLASSIFICATION

A classification problem is where we predict the category of a given sample.eg-

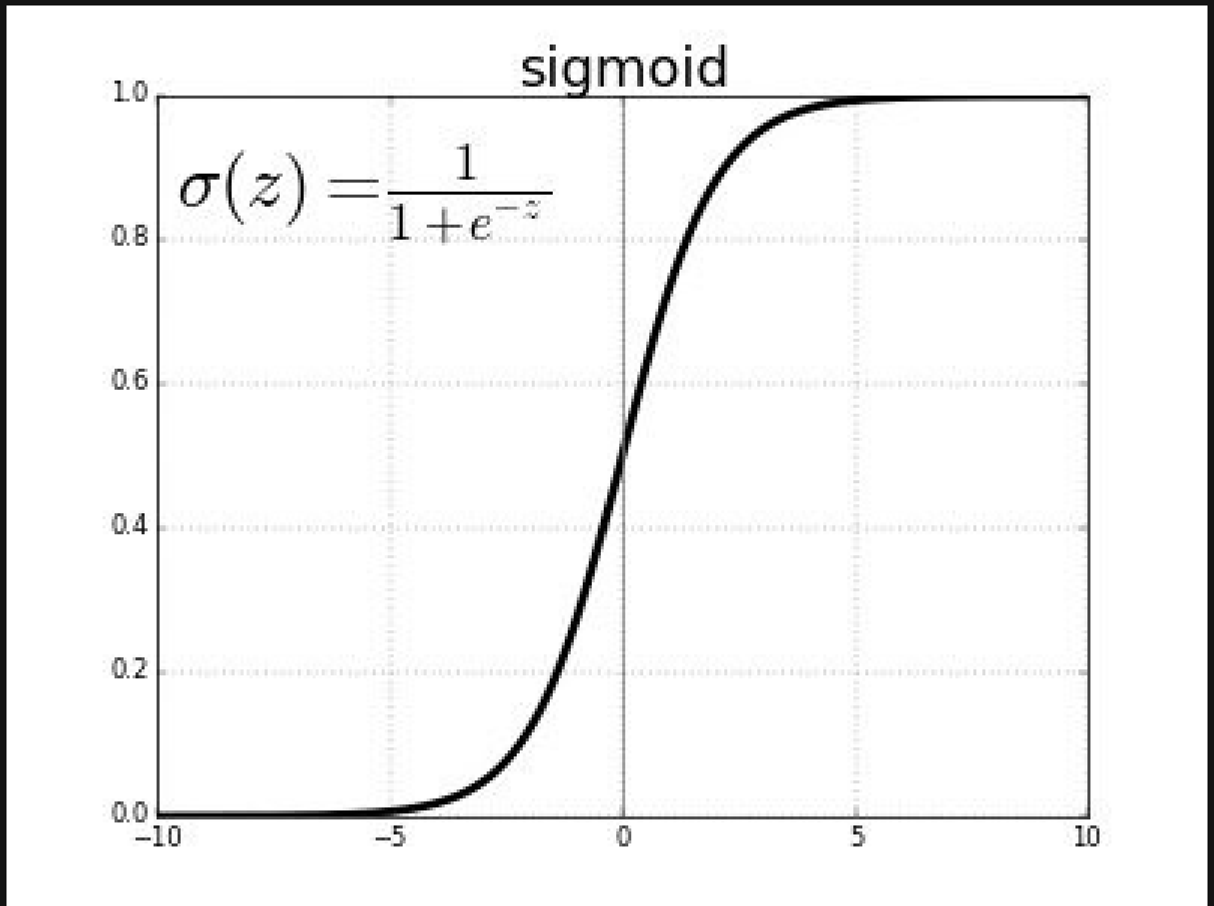
- SMS - spam or not spam [Binary classification]
- Cuisine preference - Indian , Chinese , Italian [Multinomial classification]
- Customer satisfaction - Low 😔, Medium 😐 , High 😊 [Ordinal classifcation]

Why not Linear regression



Sigmoid or Logistic Function

- The Linear regression model would predict a number between -infinity to +infinity
- However, we would like the predictions of our classification model to be between 0 and 1 since our output is either 0 or 1.
- This can be accomplished by using a "sigmoid function" which maps all input values to values between 0 and 1.



Logistic Regression

- The formula for a sigmoid function is as follows -

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

- In the case of logistic regression, z (the input to the sigmoid function), is the output of a linear regression model.

$$f_{\vec{w},b}(\vec{x}) = g(\underbrace{\vec{w}\cdot\vec{x}+b}_{z}) = \frac{1}{1+e^{-(\vec{w}\cdot\vec{x}+b)}}$$

"logistic regression"

DECISION BOUNDARY

$$f_{\mathbf{w},b}(\mathbf{x}^{(i)}) = g(\mathbf{w} \cdot \mathbf{x}^{(i)} + b)$$

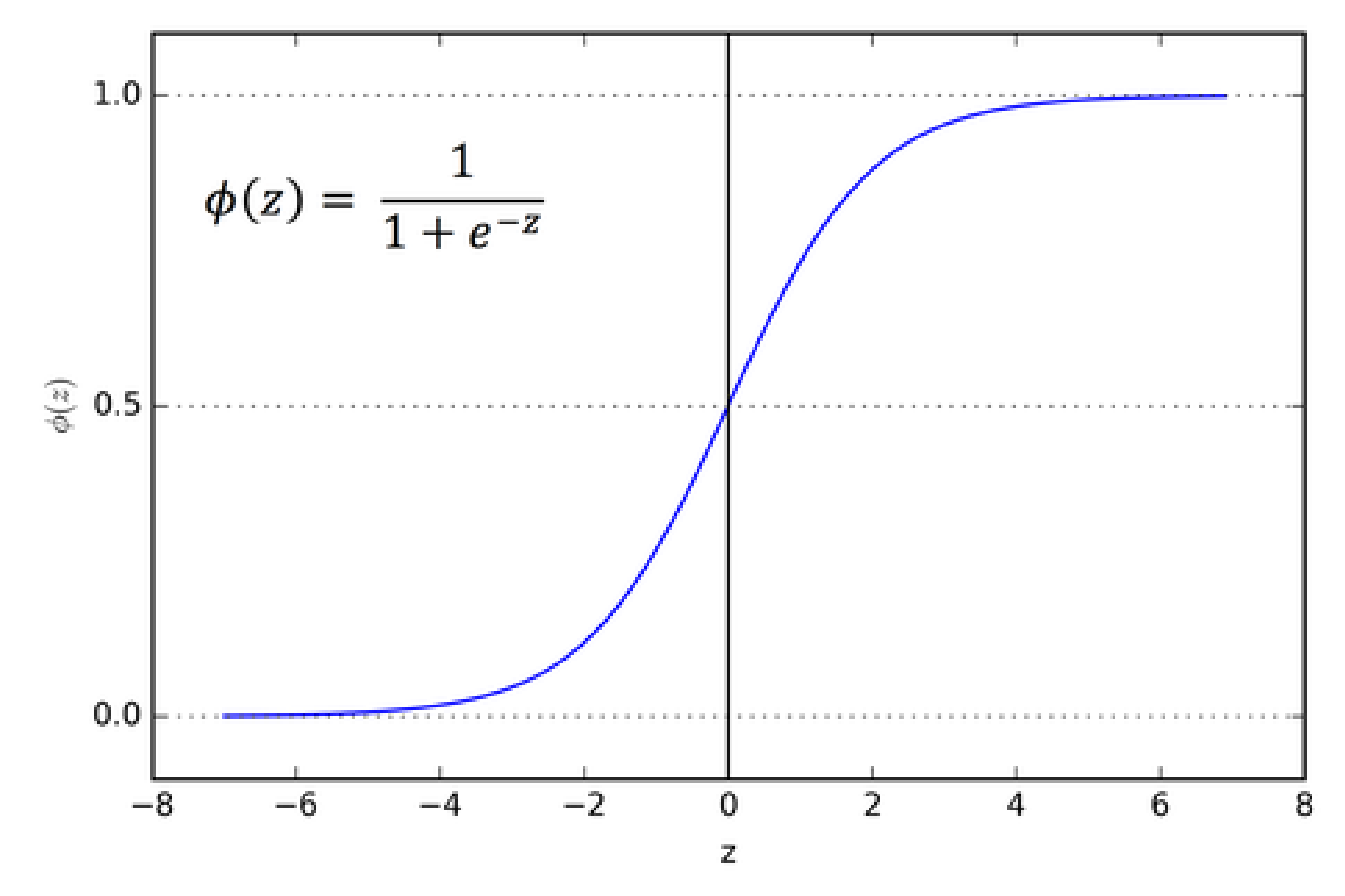
$$g(z) = \frac{1}{1+e^{-z}}$$

to get a final prediction (y=0 or y=1)
from the logistic regression model, we can use the following heuristic -

if $f_{\mathbf{w},b}(x)>=0.5$
, predict y=1

if $f_{\mathbf{w},b}(x)<0.5$
, predict y=0

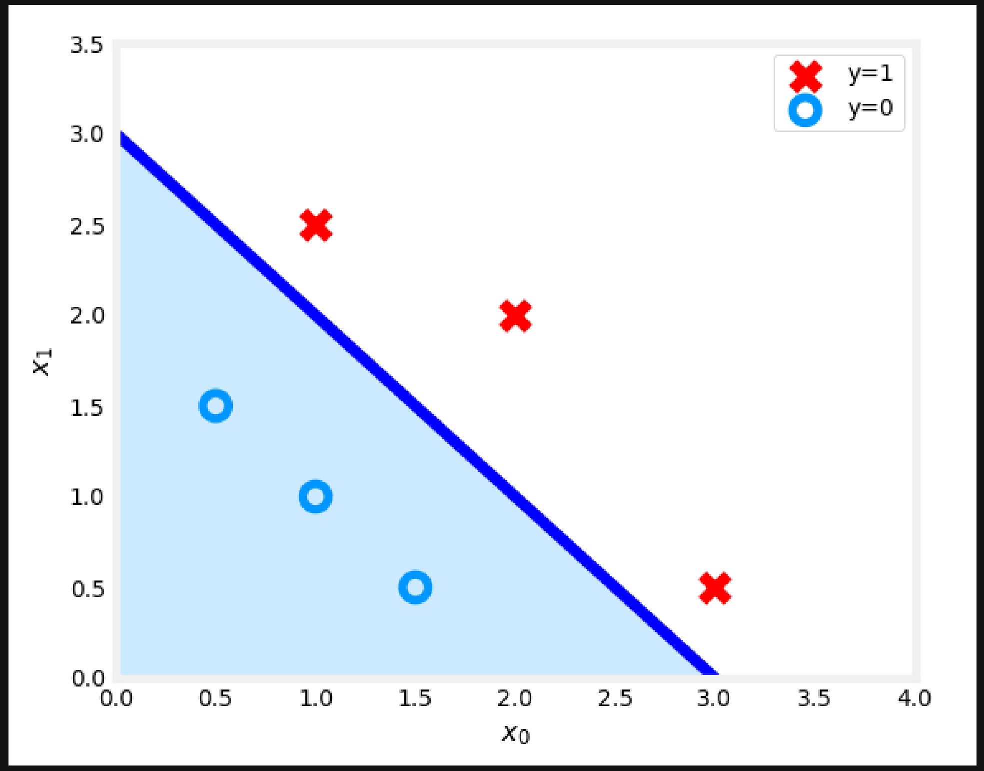
for $f_{\mathbf{w},b}(x)>=0.5$
=> g(z)>=0.5
=> z>0



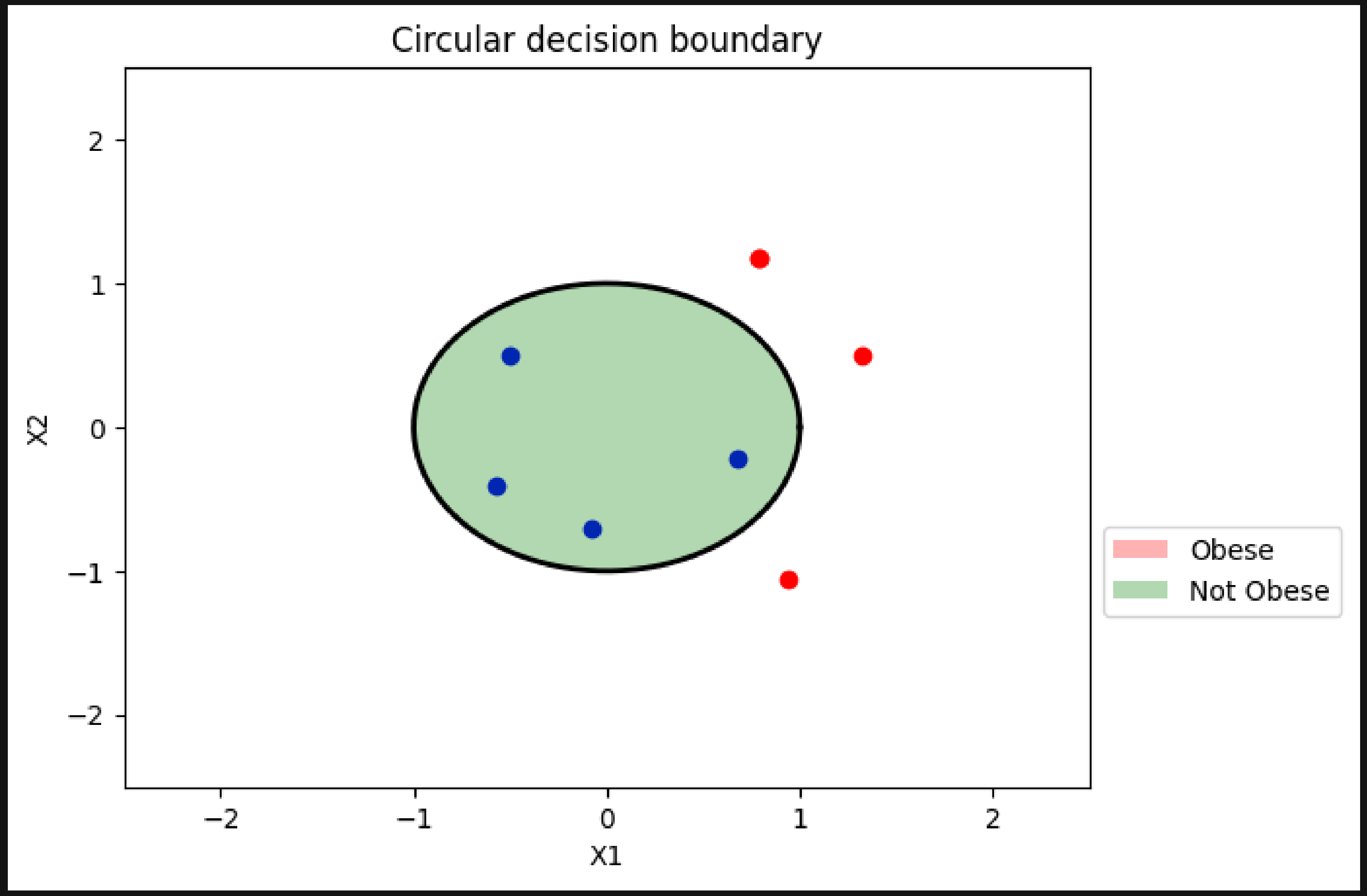
DECISION BOUNDARY

Decision boundary is given by z=0

- Say you have a logistic regression model of the form $f(x)=g(w0x0+w1x1+b)$
- Let's say that you trained the model and get the parameters as $b=-3, w0=1, w1=1$. That is, $f(x)=g(x0+x1-3)$
  then decision boundary is given by $x0+x1-3=0$, which is a line



- similarly , if $f(x)=g((x0)^2+(x1)^2-1)$ then decision boundary is given by $(x0)^2+(x1)^2-1=0$, which is a circle



QUIZ-1

COST FUNCTION

- for Linear Regression we have used the squared error cost function: The equation for the squared error cost with one variable is:

$$J(w,b) = \frac{1}{2m}\sum_{i=0}^{m-1}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

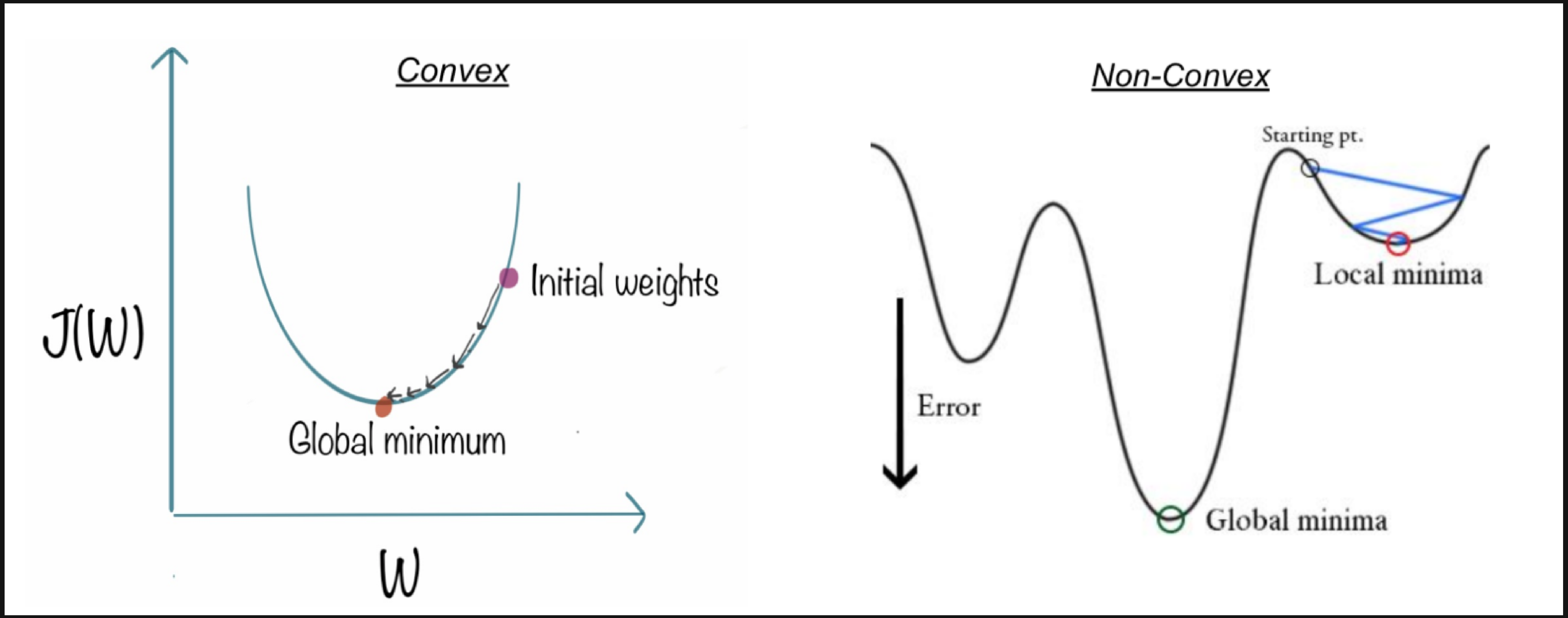
$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

- In the case of Logistic regression

$$f_{w,b}(x^{(i)}) = sigmoid(wx^{(i)} + b)$$

- Lets look at the graph of the cost function

SQUARED ERROR COST FUNCTION



$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

$$f_{w,b}(x^{(i)}) = sigmoid(wx^{(i)} + b)$$

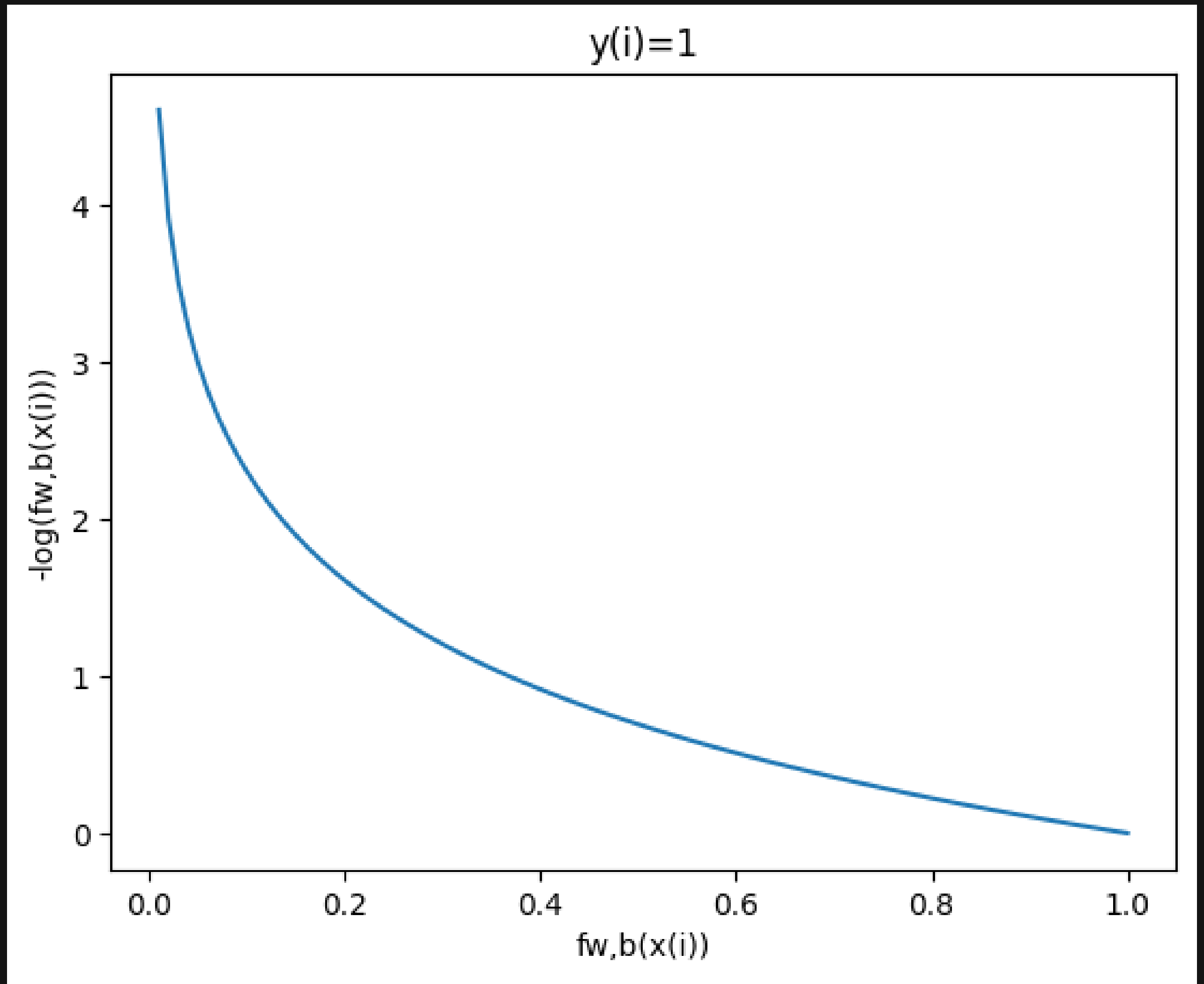
- Its clear that the squared error cost function in the case of logistic regression is a non convex
- So the gradient descent algorithm might get stuck in a local minimum point before reaching the global minima
- So we need a different cost function which can give us a convex graph like in the case of squared error cost for linear regression

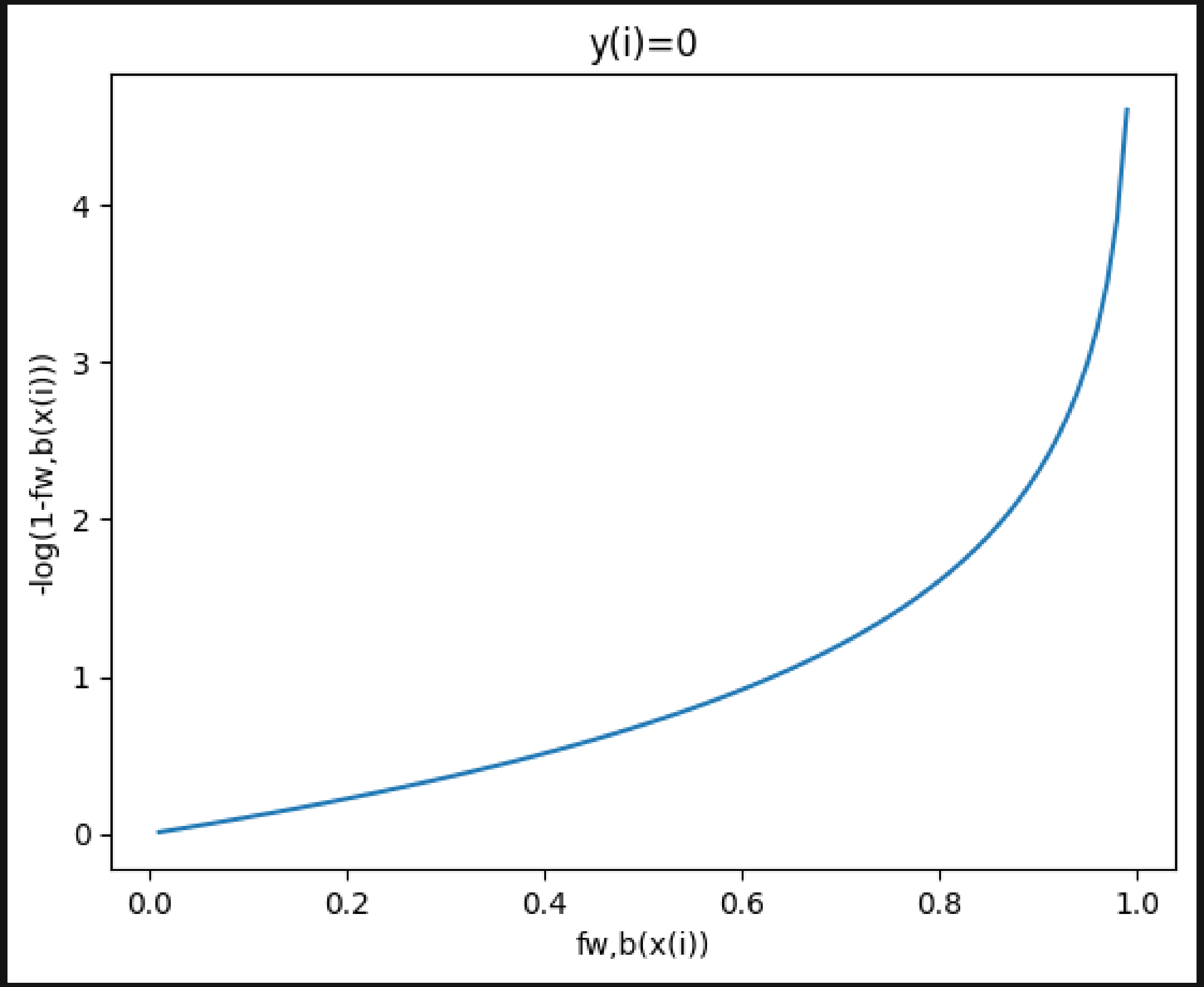
LOGISTIC LOSS FUNCTION

$$loss(f_{\mathbf{w},b}(\mathbf{x}^{(i)}), y^{(i)}) = \begin{cases} -\log\left(f_{\mathbf{w},b}\left(\mathbf{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - f_{\mathbf{w},b}\left(\mathbf{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$

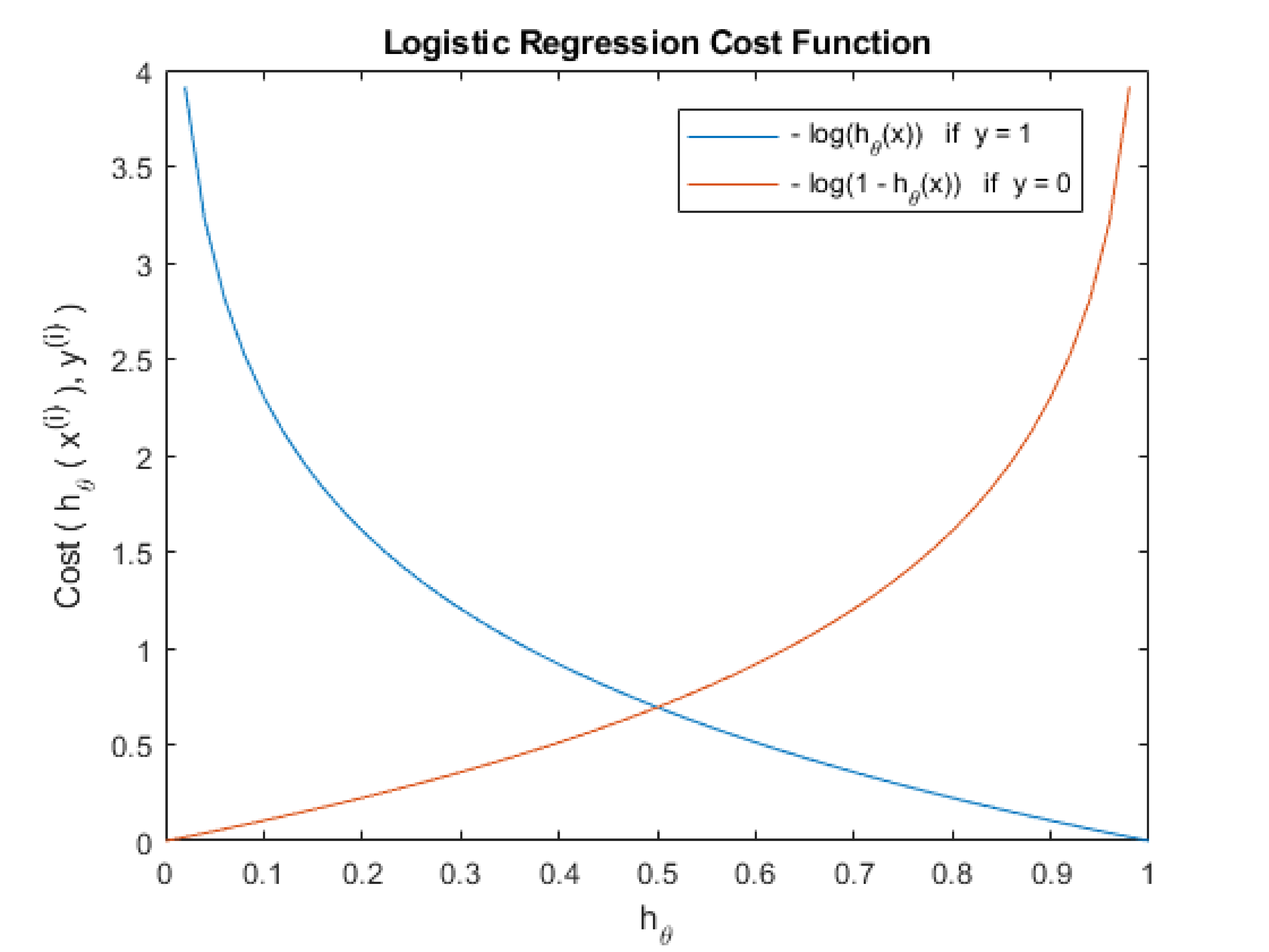
- $f\mathbf{w},b(\mathbf{x}(i)$ is the model's prediction, while $y(i)$ is the target value.

- Loss is a measure of the difference of a single example to its target value while the
- Cost is a measure of the losses over the training set





Logistic Loss Function



Gradient descent

$$J(\vec{\mathrm{w}}, b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right)\right)\right]$$

repeat {

$$w_j = w_j - \alpha\frac{\partial}{\partial w_j}J(\vec{\mathrm{w}}, b)$$

$$b = b - \alpha\frac{\partial}{\partial b}J(\vec{\mathrm{w}}, b)$$

}

$$\frac{\partial}{\partial w_j}J(\vec{\mathrm{w}}, b) = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

$$\frac{\partial}{\partial b}J(\vec{\mathrm{w}}, b) = \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) - y^{(i)}\right)$$

gradient descent

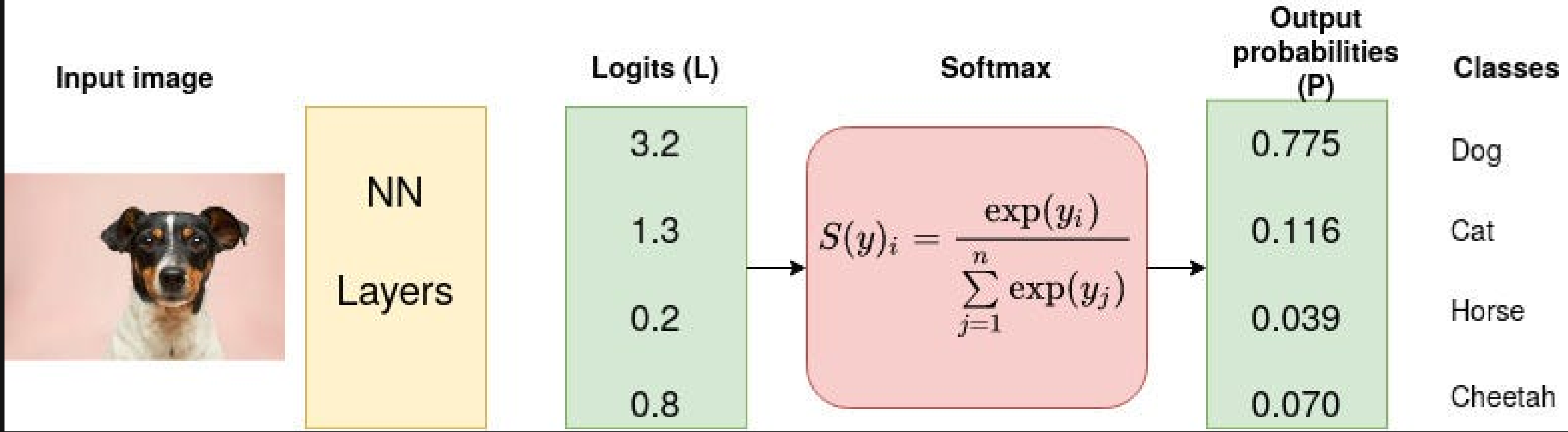
Now that we have found the cost function we can now use the gradient descent to update the weights(wj) and biases(b) to minmise the error in the prediction

Logistic Regression code implementation

MULTICLASS CLASSIFICATION

- we used sigmoid for binary classification
- For multi class classifications we can use the softmax function
- Softmax converts the outputs from the model into probabilities

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$



LIMITATIONS OF LOGISTIC REGRESSION

Logistic Regression assumes linearity between the input features and the binary outcome :
For example, predicting the likelihood of a customer making a purchase based on their age and income may not have a linear relationship. In such cases, logistic regression may not capture the complex non-linear patterns in the data, and its performance may be limited.

QUIZ-2