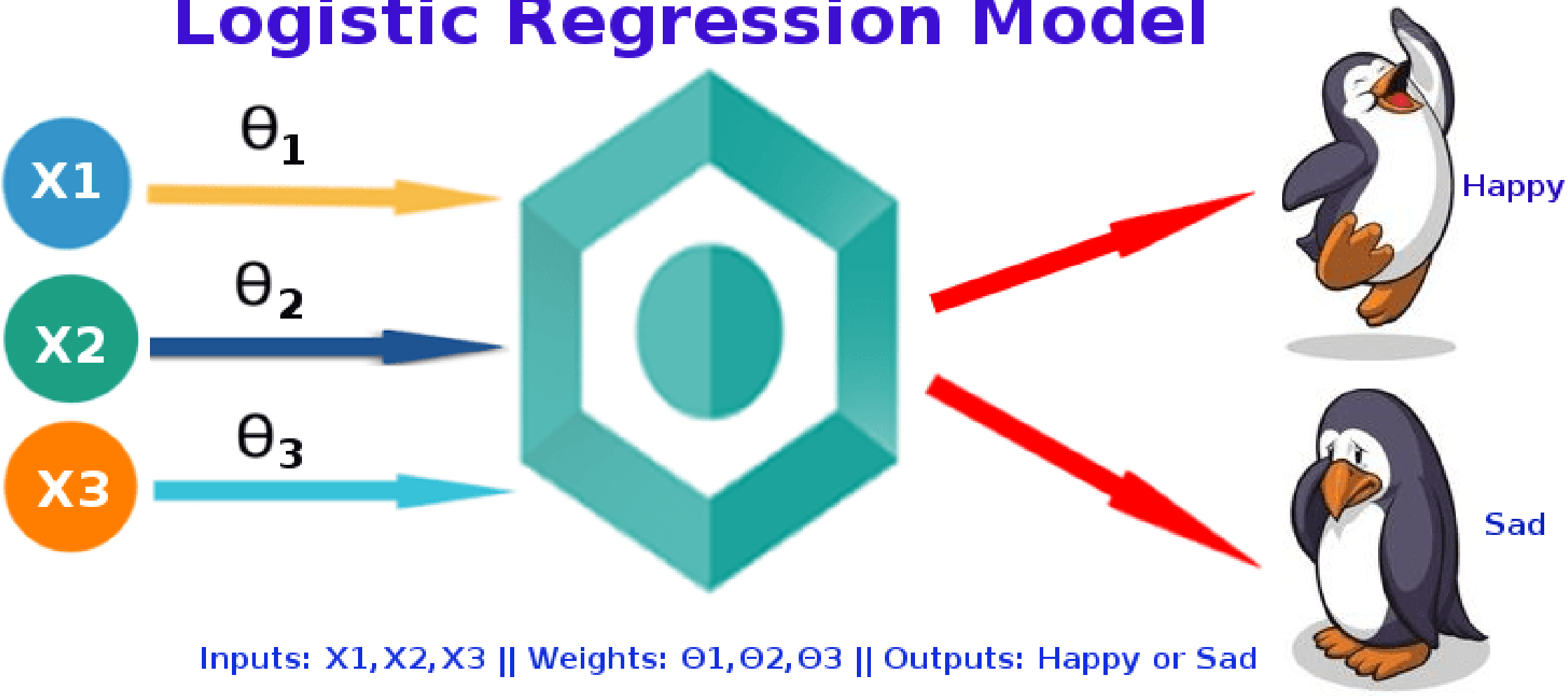
Logistic Regression Model



Inputs: X1, X2, X3 || Weights: Θ1, Θ2, Θ3 || Outputs: Happy or Sad

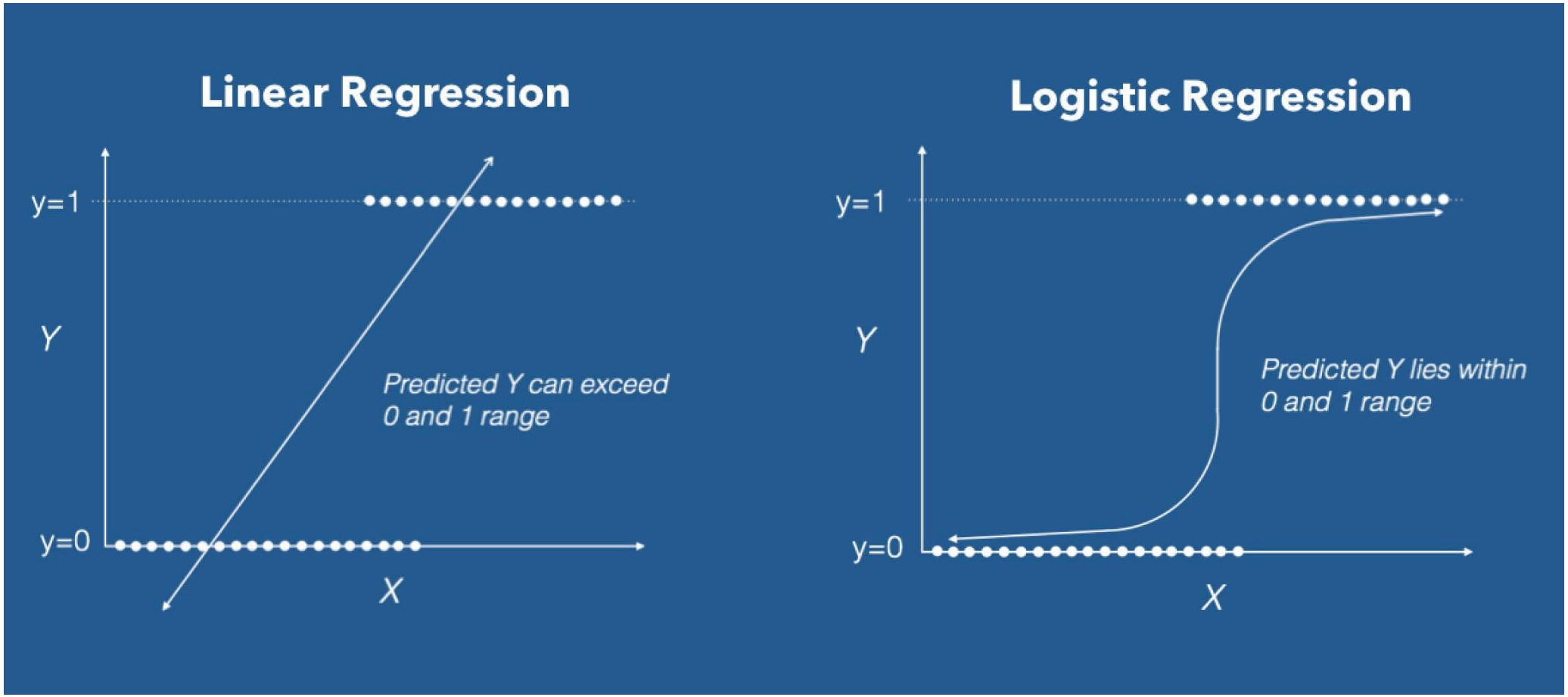
@dataaspirant.com

Some real world examples of binary classification problems

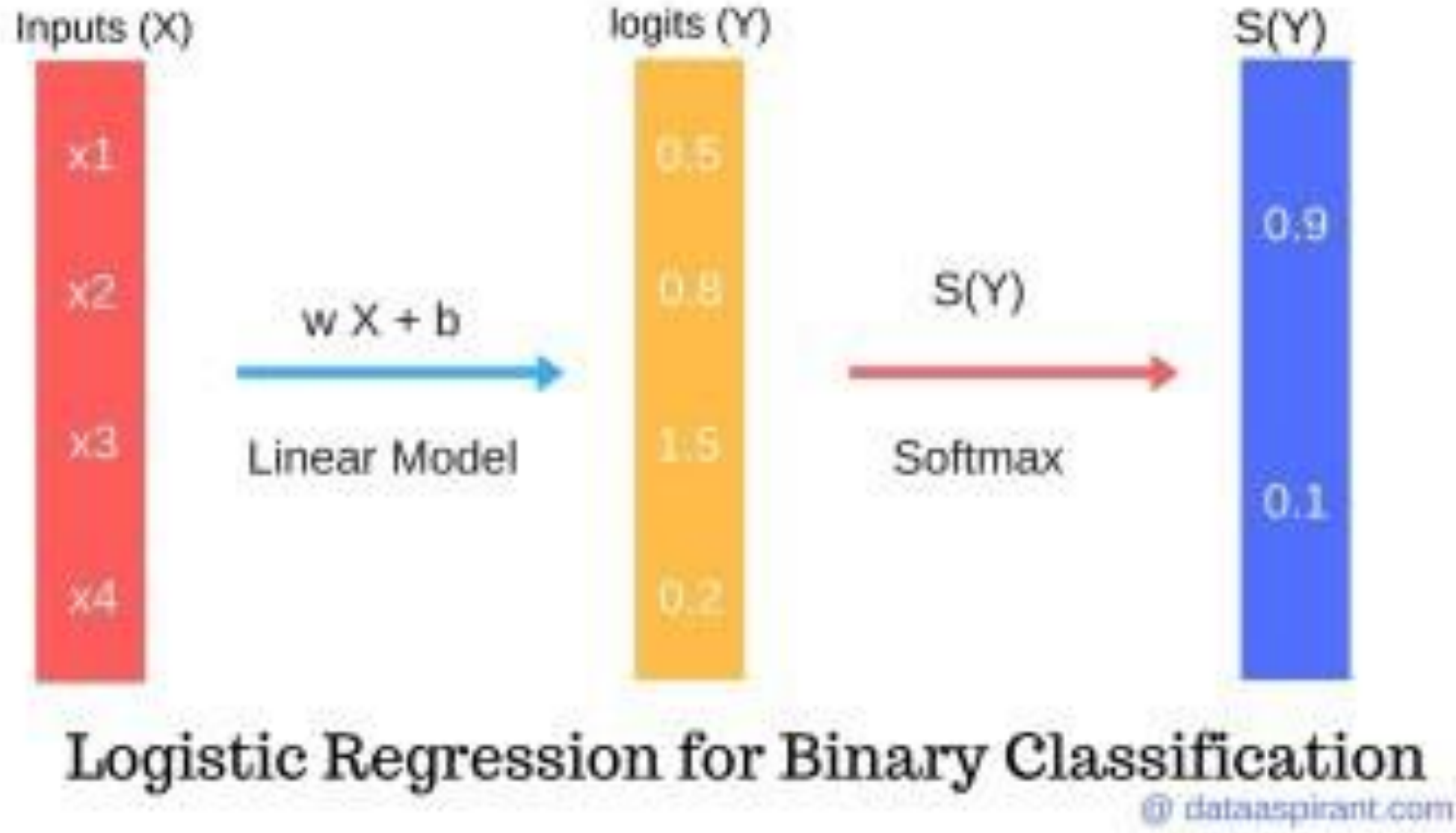
- **Spam Detection** : Predicting if an email is Spam or not
- **Credit Card Fraud** : Predicting if a given credit card transaction is fraud or not
- **Health** : Predicting if a given mass of tissue is benign or malignant
- **Marketing** : Predicting if a given user will buy an insurance product or not
- **Banking** : Predicting if a customer will default on a loan.

Why not linear regression?

- When the response variable has only 2 possible values, it is desirable to have a model that predicts the value either as 0 or 1 or as a probability score that ranges between 0 and 1.
- Linear regression does *not* have this capability. Because, If you use linear regression to model a binary response variable, the resulting model may not restrict the predicted Y values within 0 and 1.



This is where logistic regression comes into play. In logistic regression, you get a probability score that reflects the probability of the occurence of the event.
An event in this case is each row of the training dataset. It could be something like classifying if a given email is spam, or mass of cell is malignant or a user will buy a product and so on.



Logistic Regression for Binary Classification

- **Binary logistic regression** is estimated using **Maximum Likelihood Estimation (MLE)**, unlike linear regression which uses the Ordinary Least Squares (OLS) approach.

Types of Logistic Regression

1. Binary Logistic Regression

The categorical response has only two 2 possible outcomes. Example: Spam or Not

2. Multinomial Logistic Regression

Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan)

3. Ordinal Logistic Regression

Three or more categories with ordering. Example: Movie rating from 1 to 5

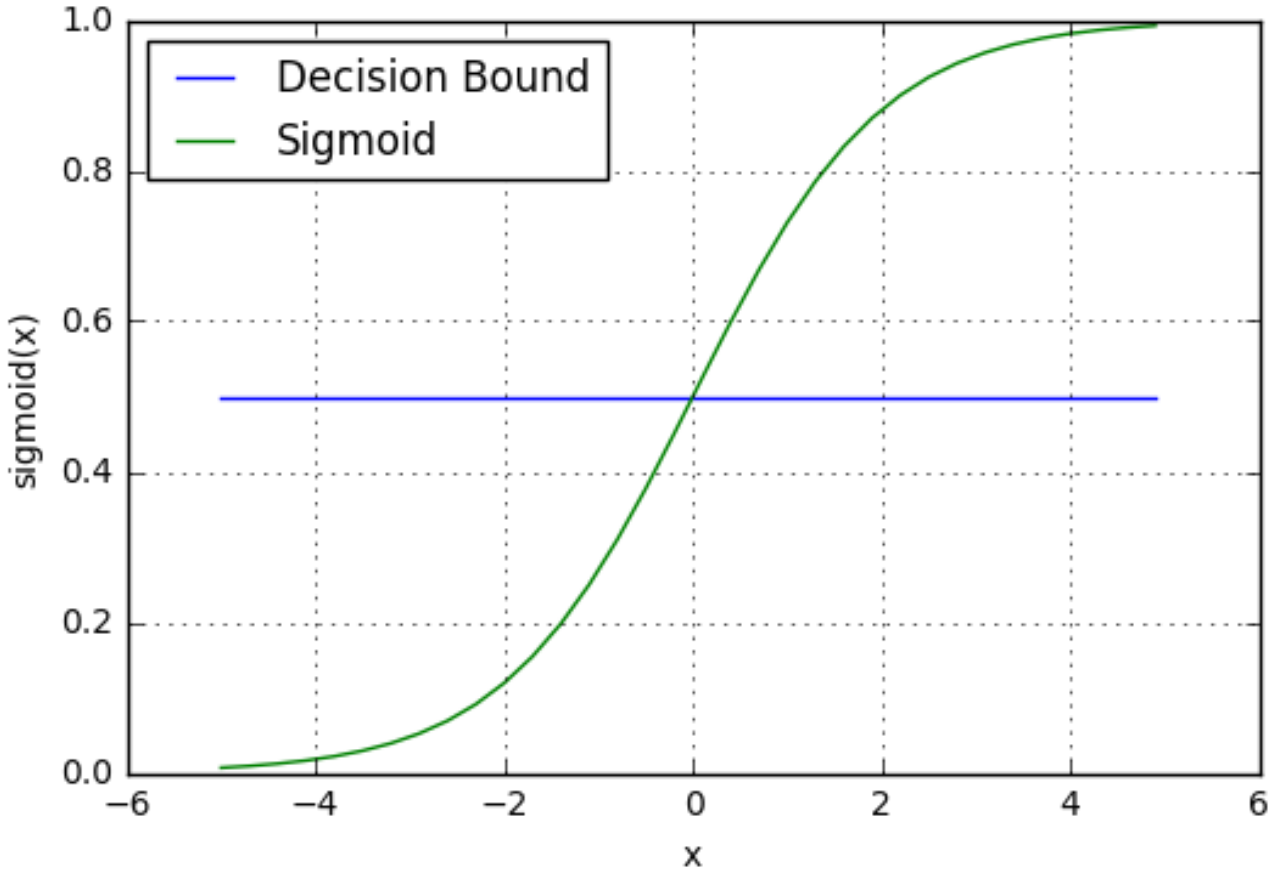
Decision Boundary

$$p \geq 0.5, class = 1$$
$$p < 0.5, class = 0$$

- To predict which class a data belongs, a threshold can be set. Based upon this threshold, the obtained estimated probability is classified into classes.
- Say, if predicted_value ≥ 0.5, then classify email as spam else as not spam.
- Our current prediction function returns a probability score between 0 and 1.
- In order to map this to a discrete class (true/false, cat/dog), we select a threshold value or tipping point above which we will classify values into class 1 and below which we classify values into class 2.

- Decision boundary can be linear or non-linear. Polynomial order can be increased to get complex decision boundary.

- For example, if our threshold was .5 and our prediction function returned .7, we would classify this observation as positive. If our prediction was .2 we would classify the observation as negative. For logistic regression with multiple classes we could select the class with the highest predicted probability.



- **Math**
- Let’s use the same multiple linear regression equation from our linear regression tutorial.
    - $z=W0+W1Studied+W2Slept$

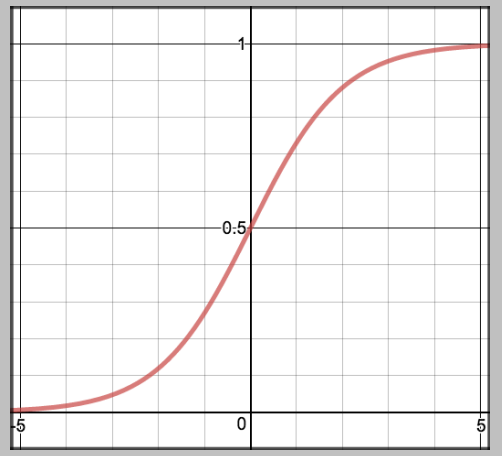
- This time however we will transform the output using the sigmoid function to return a probability value between 0 and 1.

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

- If the model returns .4 it believes there is only a 40% chance of passing. If our decision boundary was .5, we would categorize this observation as “Fail.”“

Sigmoid Activation

- In order to map predicted values to probabilities, we use the sigmoid function.
- The function maps any real value into another value between 0 and 1.
- In machine learning, we use sigmoid to map predictions to probabilities.



Math

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

Note

- $s(z)$ = output between 0 and 1 (probability estimate)
- z = input to the function (your algorithm's prediction e.g. mx + b)
- e = base of natural log

- Sigmoid Function (Logistic Function)

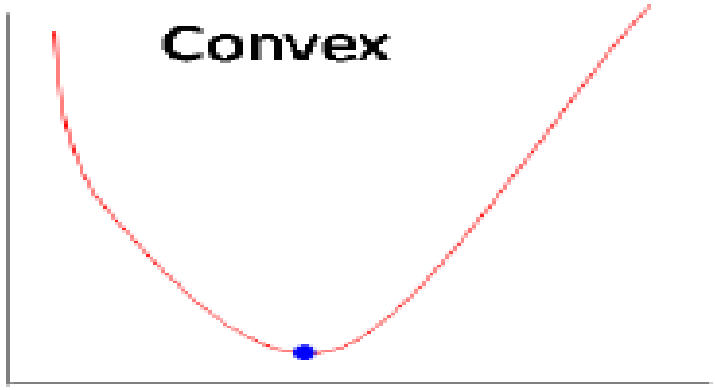
Cost Function:

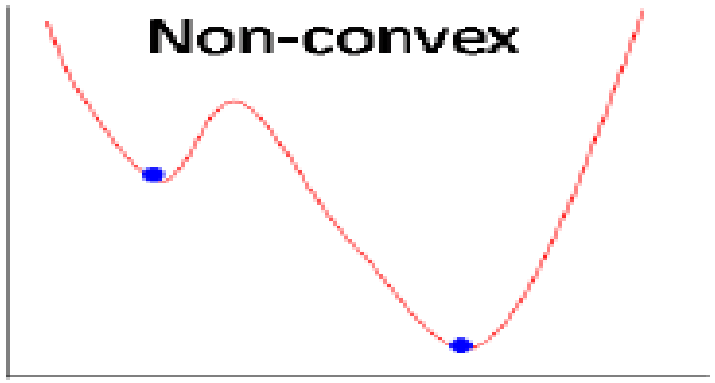
$$\text{Cost}(h_\Theta(x), Y(\text{actual})) = -\log(h_\Theta(x)) \text{ if } y=1$$

$$-\log(1- h_\Theta(x)) \text{ if } y=0$$

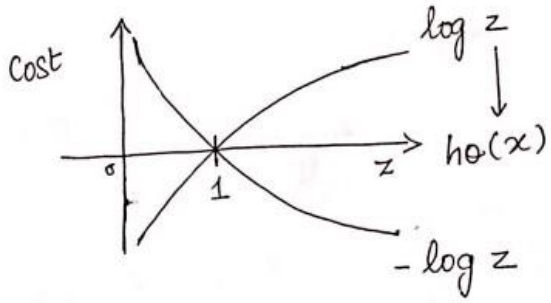
Cost Function

- Why cost function which has been used for linear can not be used for logistic?
- Linear regression uses mean squared error as its cost function. If this is used for logistic regression, then it will be a non-convex function of parameters (theta). Gradient descent will converge into global minimum only if the function is convex.





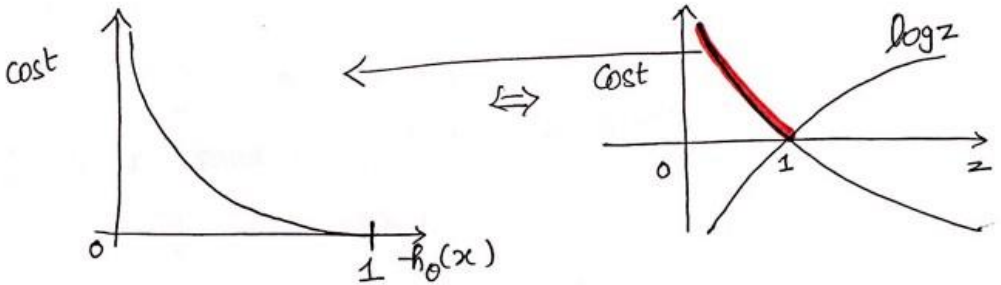
Cost function explanation



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

If $y = 1$,

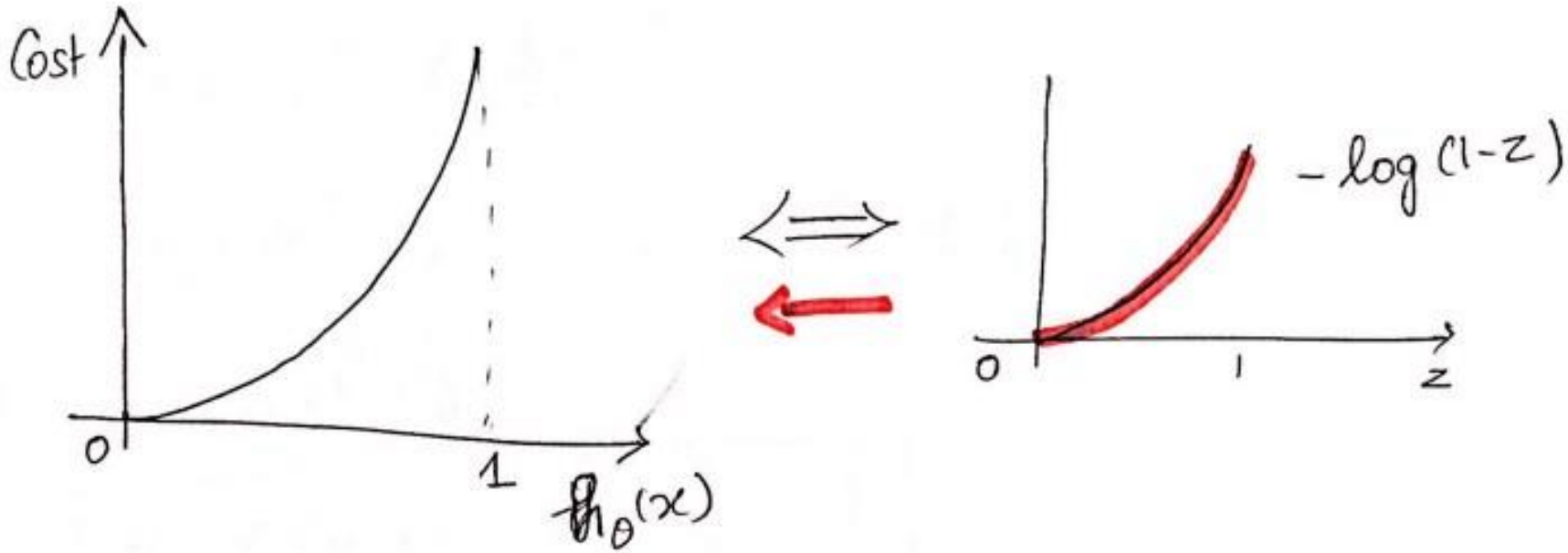
$$\text{Cost}(h_\theta(x), y) = -\log(h_\theta(x))$$



If, Cost = 0 $\Rightarrow$ $y = 1$ $\Rightarrow$ $h_\theta(x) = 1$

Cost = unfinity for $h_\theta(x) = 0$

If $h_\theta(x) = 0$, it is similar to predicting $P(y = 1 | x; \theta) = 0$



If $\text{cost} = 0 \Rightarrow h_\theta(x) = 0 \Rightarrow y = 0$

$\text{cost} = \infty \Rightarrow h_\theta(x) = 1$

If $h_\theta(x) = 1$, it is similar to predicting

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

Simplified cost function

$\text{Cost}(h_\Theta(x), y) = -y \log(h_\Theta(x)) - (1-y) \log(1- h_\Theta(x))$

If y = 1, (1-y) term will become zero, therefore $-\log(h_\Theta(x))$ alone will be present

If y = 0, (y) term will become zero, therefore $-\log(1- h_\Theta(x))$ alone will be present

Why this cost function?

Let us consider,

* $\hat{y} = P(y=1|x)$

  $\hat{y}$ is the probability that $y=1$, given x

* $1-\hat{y} = P(y=0|x)$

* $P(y|x) = \hat{y}^{y} \cdot (1-\hat{y})^{(1-y)}$

If $y=1 \Rightarrow P(y|x) = \hat{y}$

$\Rightarrow \log(\hat{y}^{y} \cdot (1-\hat{y})^{(1-y)})$

$\Rightarrow y \log \hat{y} + (1-y) \log (1-\hat{y})$

$\Rightarrow -L(\hat{y}, y)$

$$\boxed{\log P(y|x) = -L(\hat{y}, y)}$$

- This negative function is because when we train, we need to maximize the probability by minimizing loss function. Decreasing the cost will increase the maximum likelihood assuming that samples are drawn from an identically independent distribution.

Deriving the formula for Gradient Descent Algorithm

$$\frac{\partial L}{\partial w_1} = \left(\left(\frac{-y}{a} + \frac{(1-y)}{1-a} \right) \cdot (a)(1-a) \right) \cdot x_1$$

$$= (a-y) \cdot x_1$$

Update for w_1,

$$\frac{\partial L}{\partial w_1} = (a-y) \cdot x_1$$

Here, $(a-y) = \frac{\partial L}{\partial z}$

$$\boxed{w_1 = w_1 - \alpha \frac{\partial L}{\partial w_1}}$$

Similarly, for all parameters

$$\boxed{w_i = w_i - \alpha \frac{\partial L}{\partial w_i}} \qquad i = 1, 2, \ldots, m$$

m = no of parameters

$$\boxed{b = b - \alpha \frac{\partial L}{\partial b}}$$

Where, $\frac{\partial L}{\partial b} = (a-y)$

Gradient

$$\boxed{z = w_1 x_1 + w_2 x_2 + b} \rightarrow \boxed{\hat{y} = a = \sigma(z)} \rightarrow \boxed{L(\hat{y}, y)}$$

$$\Leftrightarrow \boxed{a = \hat{y}}$$

$$\boxed{w_1} \Rightarrow \frac{\partial (L)}{\partial w_1} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial (z)}{\partial w_1}$$

$$\frac{\partial L}{\partial a} = \frac{\partial}{\partial a} \left(-y \log a - (1-y) \log (1-a) \right)$$

$$= -y \left(\frac{1}{a} \right) - (-1) \frac{(1-y)}{(1-a)}$$

$$\boxed{\frac{\partial L}{\partial a} = \left(\frac{-y}{a} \right) + \left(\frac{1-y}{1-a} \right)}$$

$$\boxed{\frac{\partial a}{\partial z} = a(1-a)}$$

$$\boxed{\frac{\partial z}{\partial w_1} = x_1}$$

How to deal with Class Imbalance?

- The classes 'benign' and 'malignant' are split approximately in 1:2 ratio.
- Clearly there is a class imbalance. So, before building the logit model, you need to build the samples such that both the 1's and 0's are in approximately equal proportions.
- This concern is normally handled with a couple of techniques called:
- **Down Sampling**
- **Up Sampling**
- **Hybrid Sampling using SMOTE and ROSE**.

10. Why handling with class imbalance is important?

- Alright I promised I will tell you why you need to take care of class imbalance earlier. To understand that lets assume you have a dataset where 95% of the Y values belong to benign class and 5% belong to malignant class.
- Had I just blindly predicted all the data points as benign, I would achieve an accuracy percentage of 95%. Which sounds pretty high. But obviously that is flawed. What matters is how well you predict the malignant classes.
- So that requires the benign and malignant classes are balanced AND on top of that I need more refined accuracy measures and model evaluation metrics to improve my prediction model.