Machine Learning

1. Supervised – Input and output of each example specified
   1. Regression – Continuous output
   2. Classification – Discrete output
2. Unsupervised – Algorithm finds structures among dataset - clusters

Supervised Machine Learning

Classification

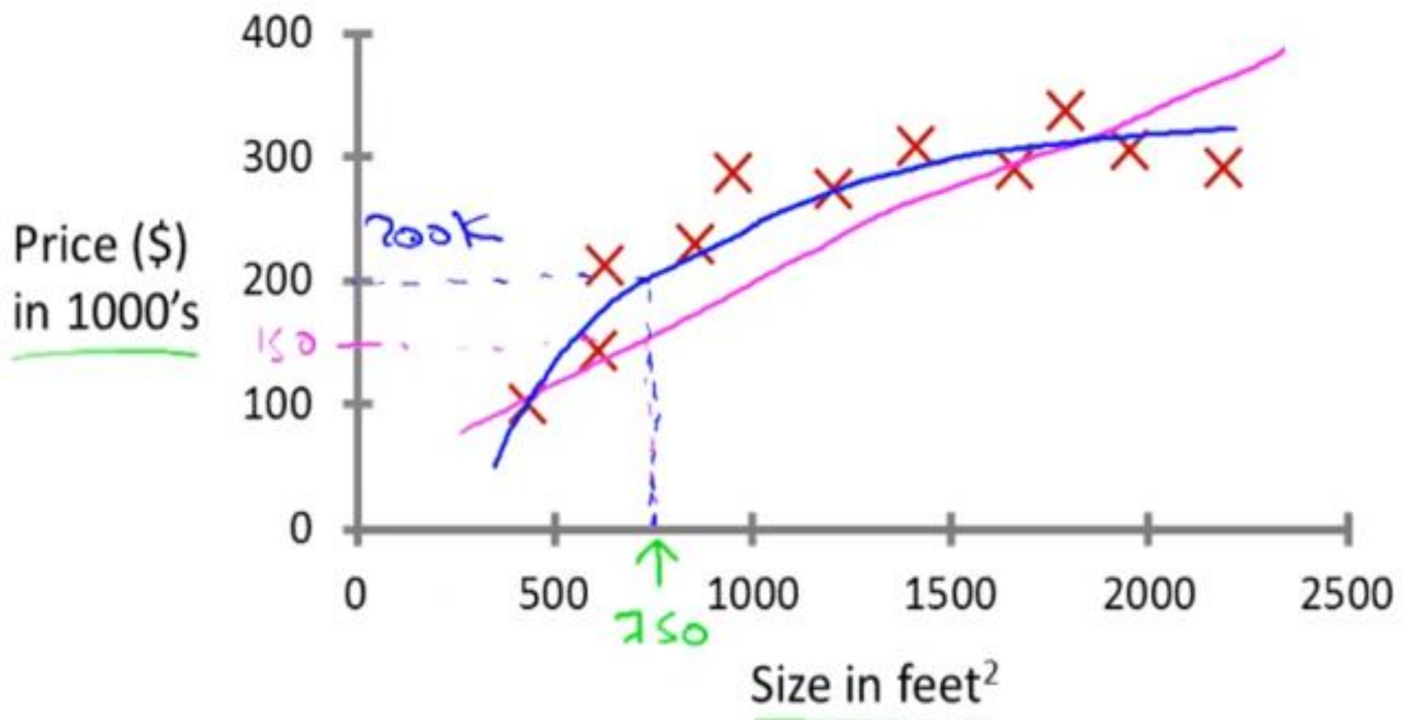
Am I a cat or a dog?

- Dataset consists of 100 pictures of 4 different animals
- Algorithm learns the features of the animals from the training set
- It can then classify the a random picture as one of the animals
- Called **multiclass classification**
- **Discrete output**

Supervised Machine Learning

Regression

What's the price of the house?



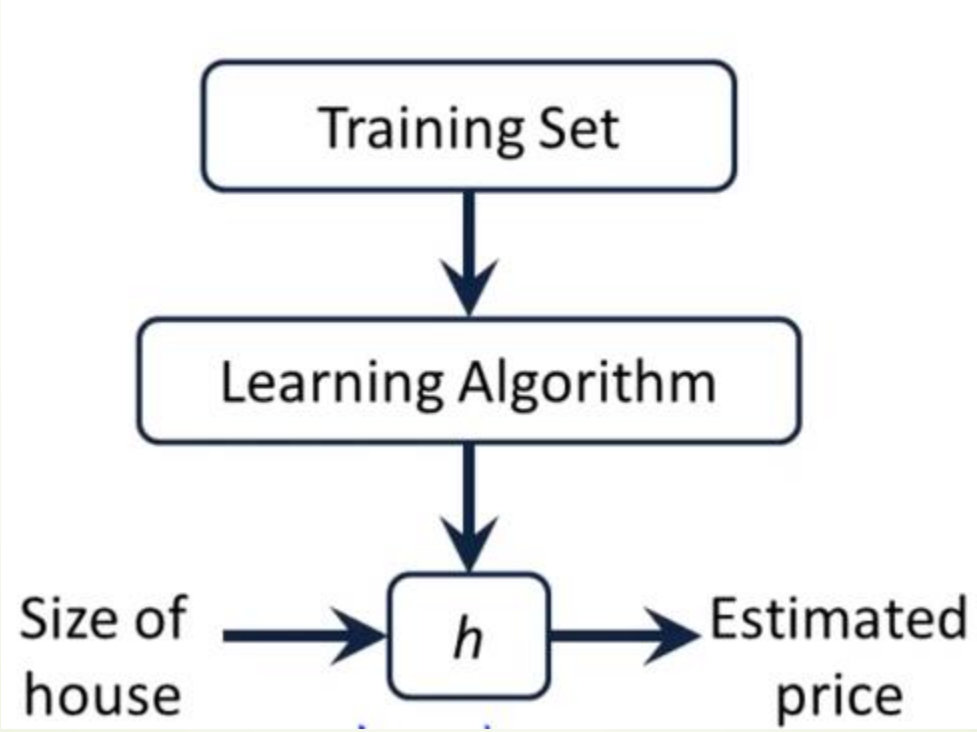
- **Learning algorithm** maps the size of the house to the price
- Predicts the price for the given input house size

Linear Regression – Univariate

- m –> training examples
- x -> input variables / features
- y -> output features

h -> Hypothesis function

→ X

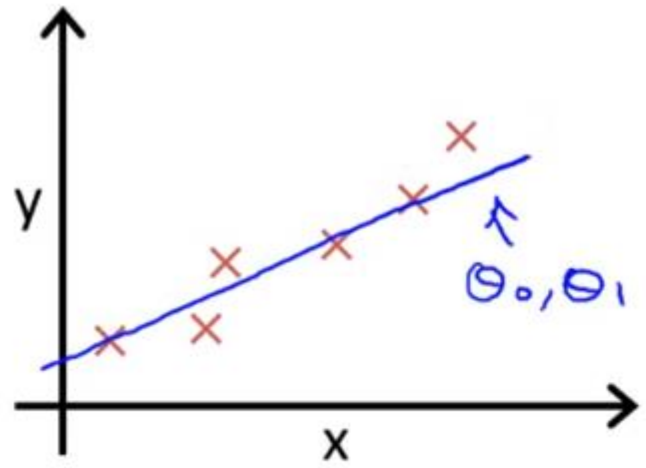


← Y

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

θ_0 **BIAS** – Offset all predictions made

θ_1 **SLOPE** of the line depicted by hypothesis function



Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training examples (x, y)

COST FUNCTION

- Error – [**h(x) - y**]

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

- Aim – Minimize the cost J
- Mean Squared Error cost function

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$

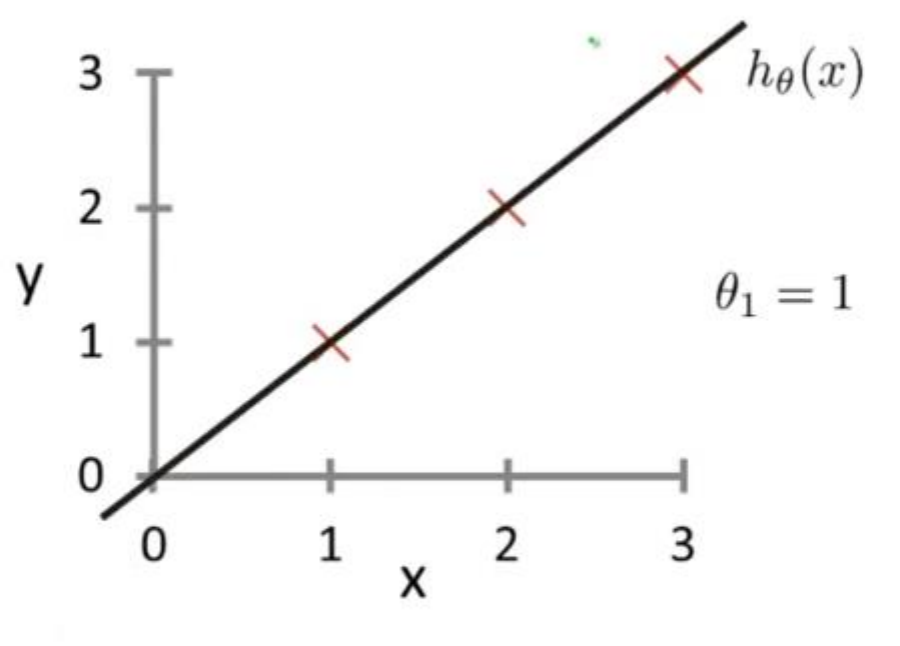
Cost Function:

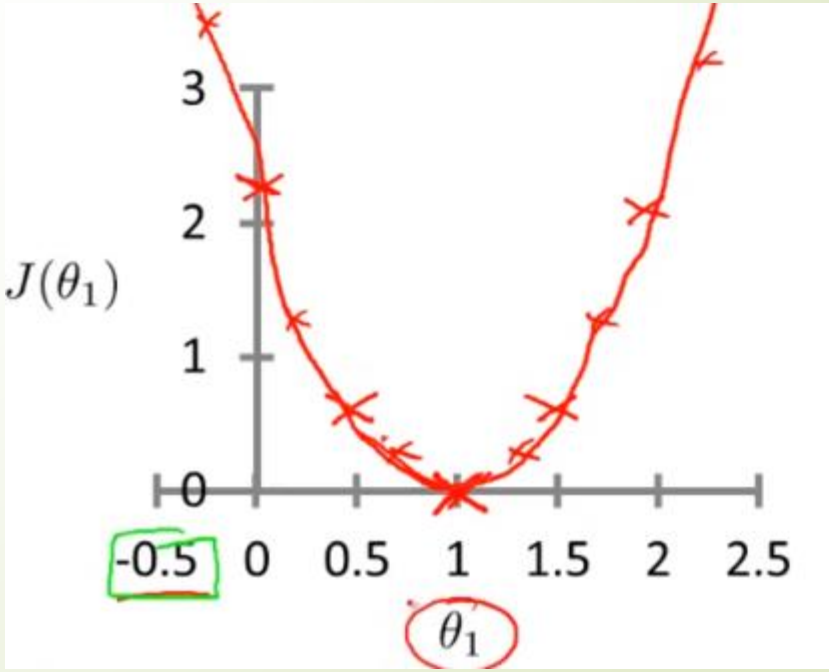
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

Simplified Model

- **$h(x) = \theta_1 x$ … $(\theta_0 = 0)$**
- When $\theta_1 = 0$
- When $\theta_1 = 0.5$
- When $\theta_1 = 1$
- **Find the value of $J(\theta)$ for each value of θ_1**





Examples

- Lemonade business – Revenue is dependent on temperature
- Amazon – recommends items to users based on user behavior
- Predict number of shoppers passing in front of billboard – predict maximum to bid for advertisement

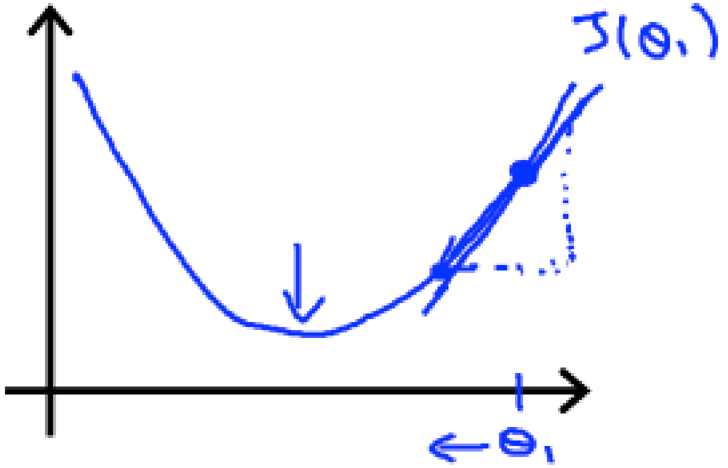
Gradient Descent

- algorithm to minimize cost function

- Start with some parameter values (θ_0, θ_1)
- Change values to reduce J(θ_0, θ_1) until minimum

$$\begin{aligned}&\text{repeat until convergence } \{\\ &\quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad (\text{for } j = 0 \text{ and } j = 1)\\ &\}\end{aligned}$$

- α - learning rate (controls how big a step is taken downhill)
- Simultaneous Update – Compute the RHS for both and assign to new (θ_0, θ_1)
- Converges even with a **fixed rate α** - the derivative approaches zero as we approach the bottom of the function – gradient descent will take smaller steps

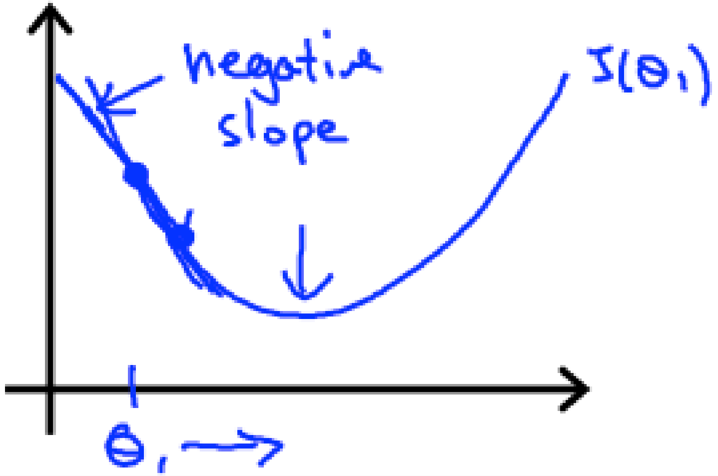


$(\theta_1 \in \mathbb{R})$

$\frac{\partial}{\partial \theta_1}$

$$\theta_1 := \theta_1 - \alpha \underbrace{\boxed{\frac{d}{d\theta_1} J(\theta_1)}}_{\geq 0}$$

$$\theta_1 := \theta_1 - \underline{\alpha} \cdot (\text{positive number})$$



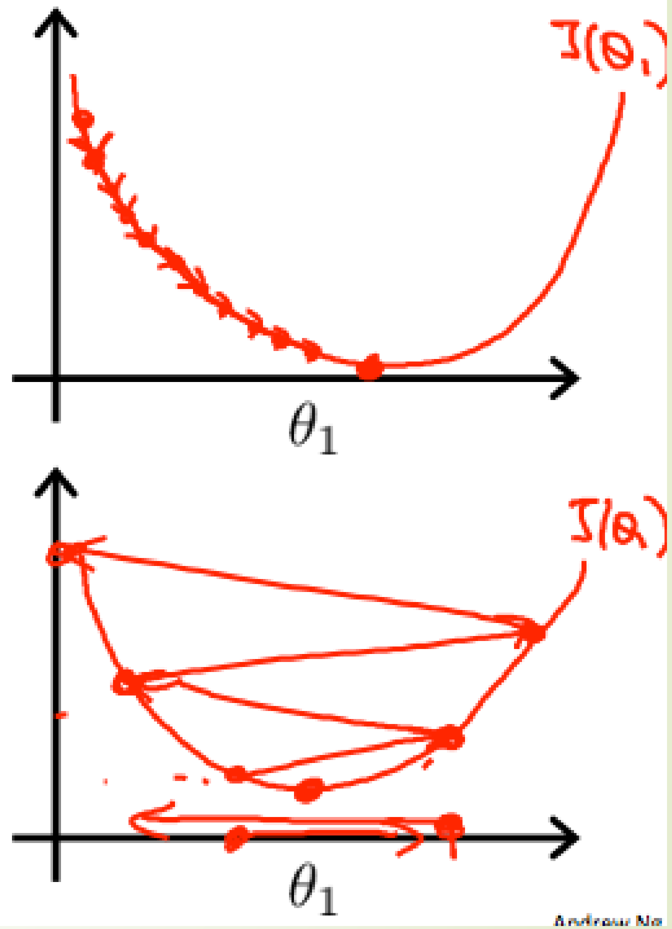
$$\underbrace{\frac{d}{d\theta_1} J(\theta_1)}_{\leq 0}$$

$$\theta_1 := \underline{\theta_1} - \alpha \, (\text{negative number})$$

$$\boxed{\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)}$$

If α is too small, gradient descent can be slow.

If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2 \\
&= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y\right) \\
&= \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y\right) \\
&= \left(h_\theta(x) - y\right) x_j
\end{aligned}
$$

repeat until convergence: {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i)$$

}

Multivariate Linear Regression

- $x_{1,}$ $x_{1,}$ x_1……….. x_n -> input
- Y -> output
- E.g. size, number of rooms, age of home, number of floors -> price

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

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

- Convenience – introduce $x_0 = 1$, such that **'x' and** '**θ**' have **'n+1'** elements each

$$h_\theta(x) = \begin{bmatrix} \theta_0 & \theta_1 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \theta^T x$$

Gradient Descent - Multivariate

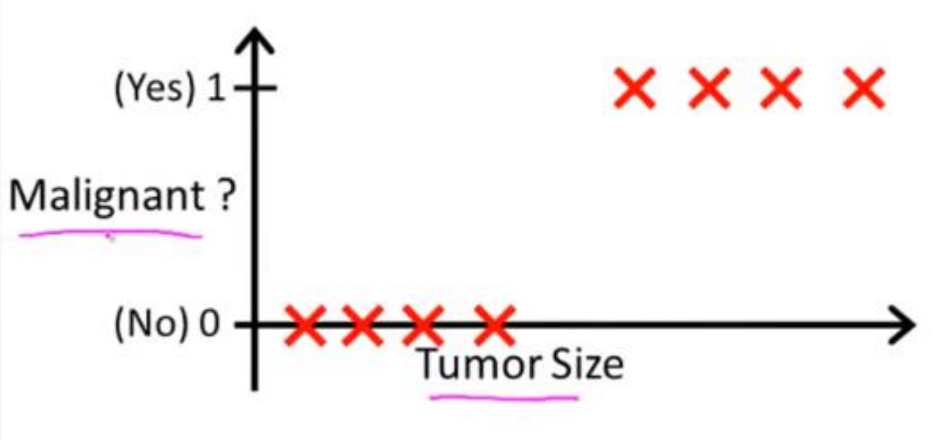
repeat until convergence: {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)} \qquad \text{for j := 0...n}$$

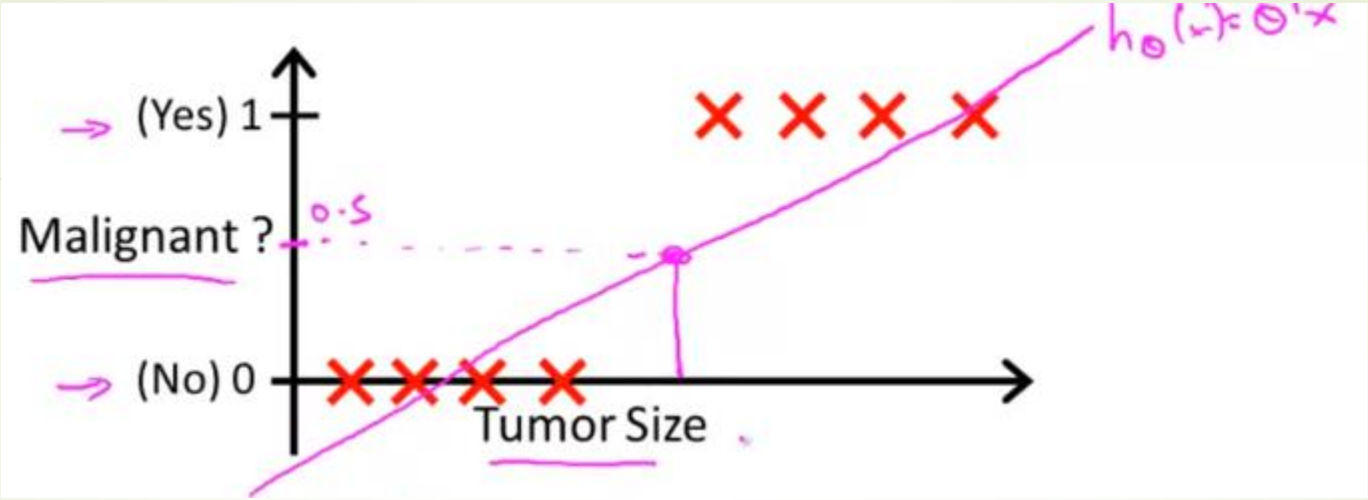
}

Logistic Regression

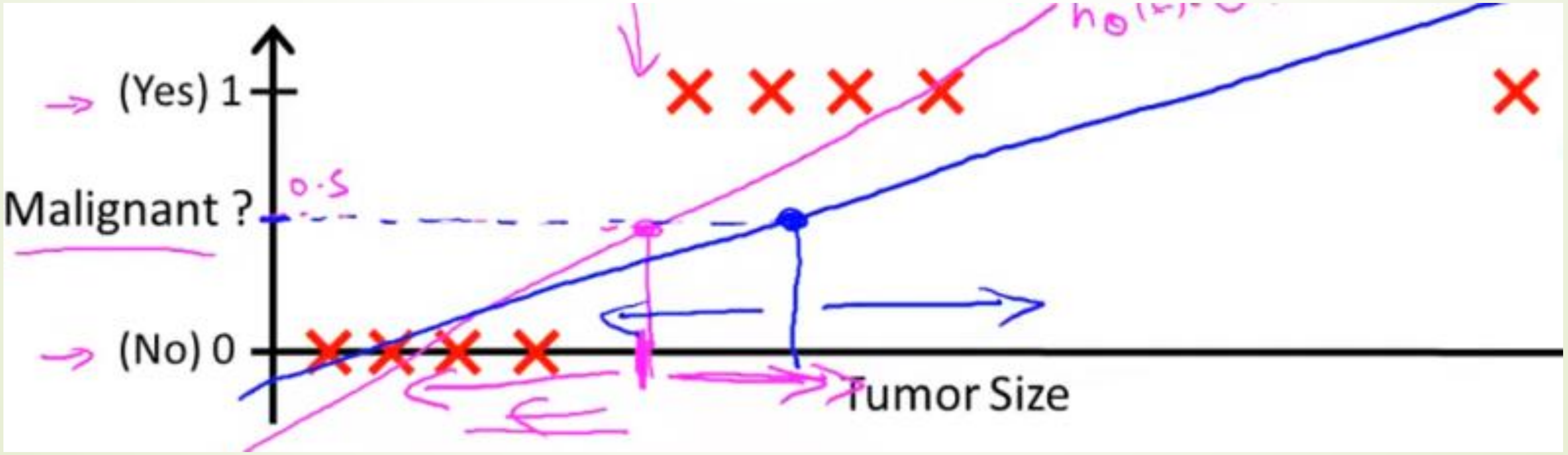
- Method for classifying data into **discrete** outcomes
- **Classification**, not regression
- Purpose of logistic regression – because linear regression is not useful for classification problems.
- y = 0 or y = 1, but using Linear Regression, h(x) can be > 1 or < 0 too.
- Using log regression, **0 <= h(x) < 1**



- Threshold the hypothesis output at 0.5 – **if h(x) > 0.5, y = 1; if h(x) < 0.5, y = 0**



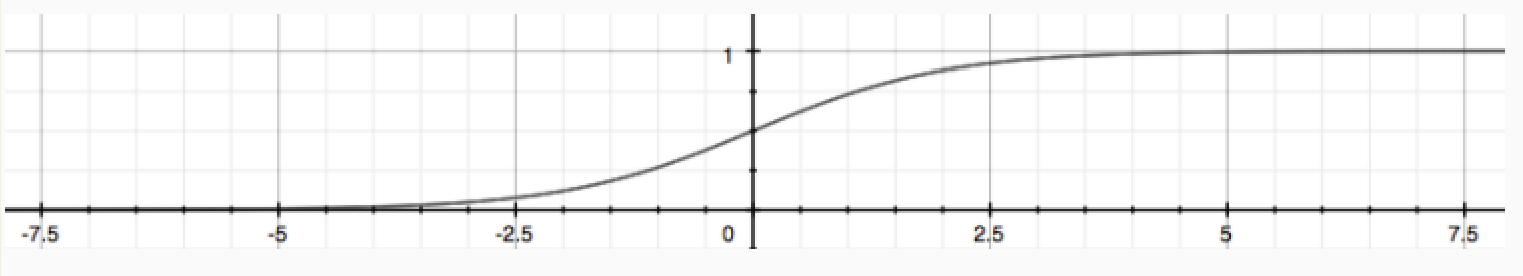
(Yes) 1
Malignant ?
0.5
(No) 0
Tumor Size



(Yes) 1
Malignant ?
0.5
(No) 0
Tumor Size

- $h(x) \rightarrow \theta^T x$ … linear regression
- **$h(x) \rightarrow g(\theta^T x)$ … logistic regression**

- **$g(z) = 1 / (1 + e^{-z})$**
  - This is the sigmoid or the logistic function
  - **Therefore, $h(x) \rightarrow 1 / (1 + e^{-\theta^T x})$**



  - **This function maps any real number to interval (0,1)**
  - **$h(x)$ – probability that output is 1 – example,**

Decision Boundary

$$h_\theta(x) \geq 0.5 \rightarrow y = 1$$
$$h_\theta(x) < 0.5 \rightarrow y = 0$$

Sigmoid function –
When **input (z)** is greater or equal to 0, output of function i.e. **h(x)** is greater than 0.5

If input to 'g' is $\theta^T x$ →

$$\theta^T x \geq 0 \Rightarrow y = 1$$
$$\theta^T x < 0 \Rightarrow y = 0$$

Decision Boundary – Line separating area where y = 0 and y = 1. Created by our hypothesis function

Cost Function – Logistic regression

- It is the penalty the learning algorithm has to pay if it outputs **h(x)** and actual label is **y**.
- If we use the same cost function as linear regression, the cost function will be a non-convex function (gradient descent won't work as it will have many local optimums) as h(x) is non linear.

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

$$\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \qquad \text{if y} = 1$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \qquad \text{if y} = 0$$

- If y = 1, cost -> ∞ if prediction is h(x) = 0 (**high penalty**) and minimum cost if h(x) = 1.
- If y = 0, cost -> ∞ if prediction is h(x) = 1 (**high penalty**) and minimum cost if h(x) = 0.