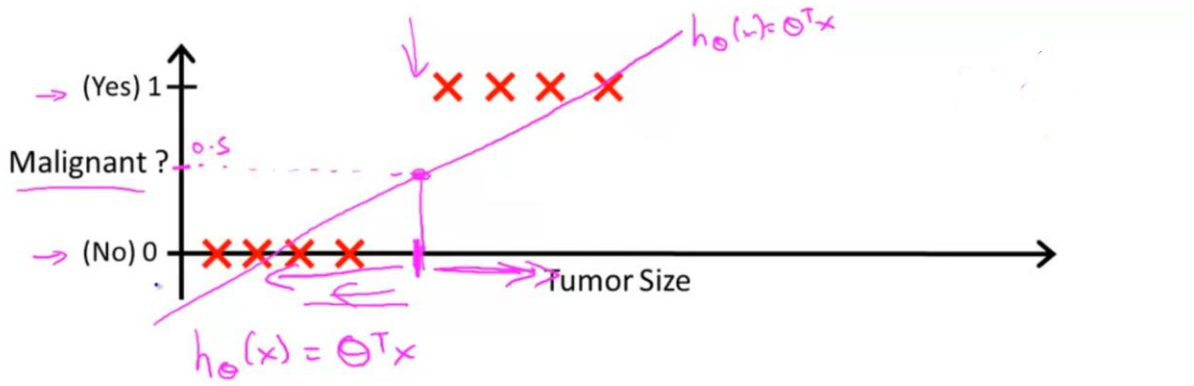
Logistic regression:

Logistic regression is used when we have a problem in which we need to classify the variable y.

To attempt classification, one method is to use linear regression and map all predictions greater than 0.5 as a 1 and all less than 0.5 as a 0. However, this method doesn't work well because classification is not actually a linear function.

For eg: if we say that all the people who have malignant tumor as classified as 1 and all those who have a benign tumor as classified as 0.

Now we will try to solve this problem linear regression:



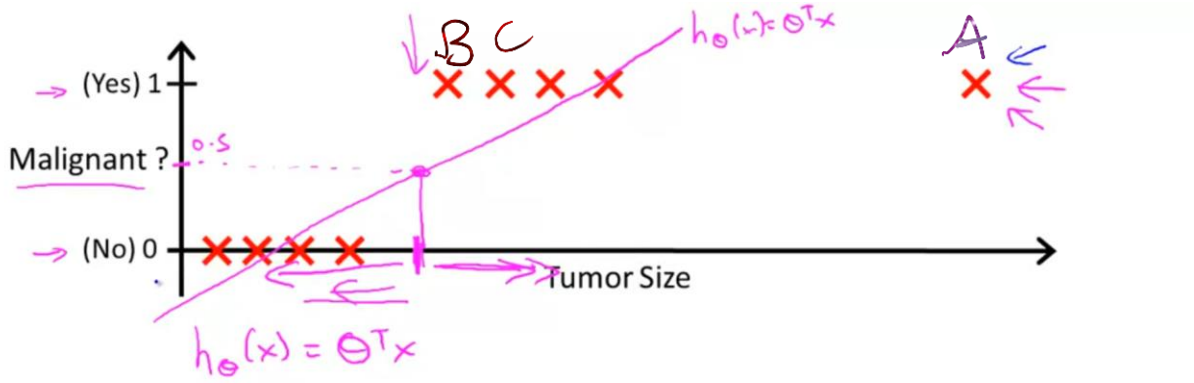
→ Threshold classifier output $h_\theta(x)$ at 0.5:

→ If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Andrew Ng

Now, suppose we add a point A in the graph:



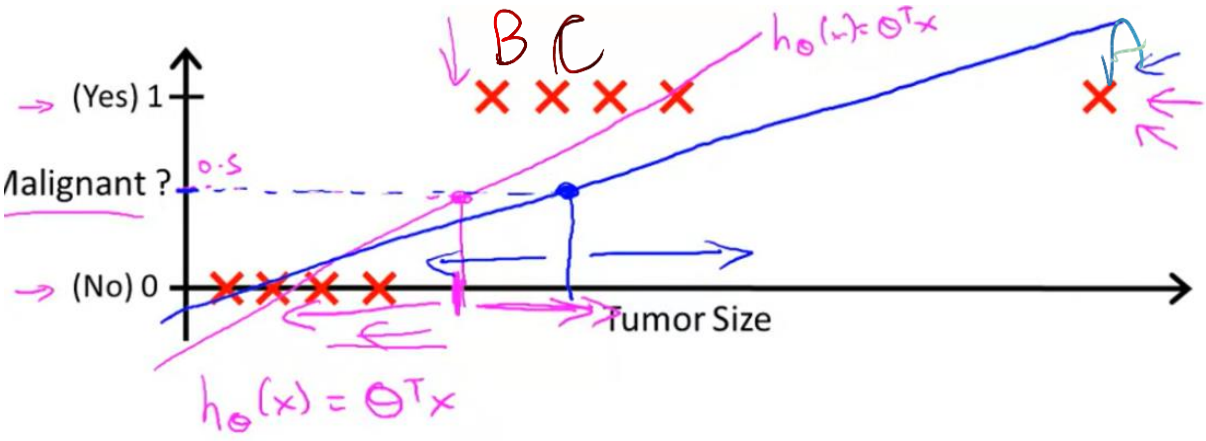
→ Threshold classifier output $h_\theta(x)$ at 0.5:

→ If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Andrew Ng

Now due to this point A , what would happen is that the line will bend more towards the axis, and a new regression line will be created:



→ Threshold classifier output $h_\theta(x)$ at 0.5:

→ If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

The new line also has the same criteria for classifying points as malignant and benign i.e. :

→ Threshold classifier output $h_\theta(x)$ at 0.5:

→ If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

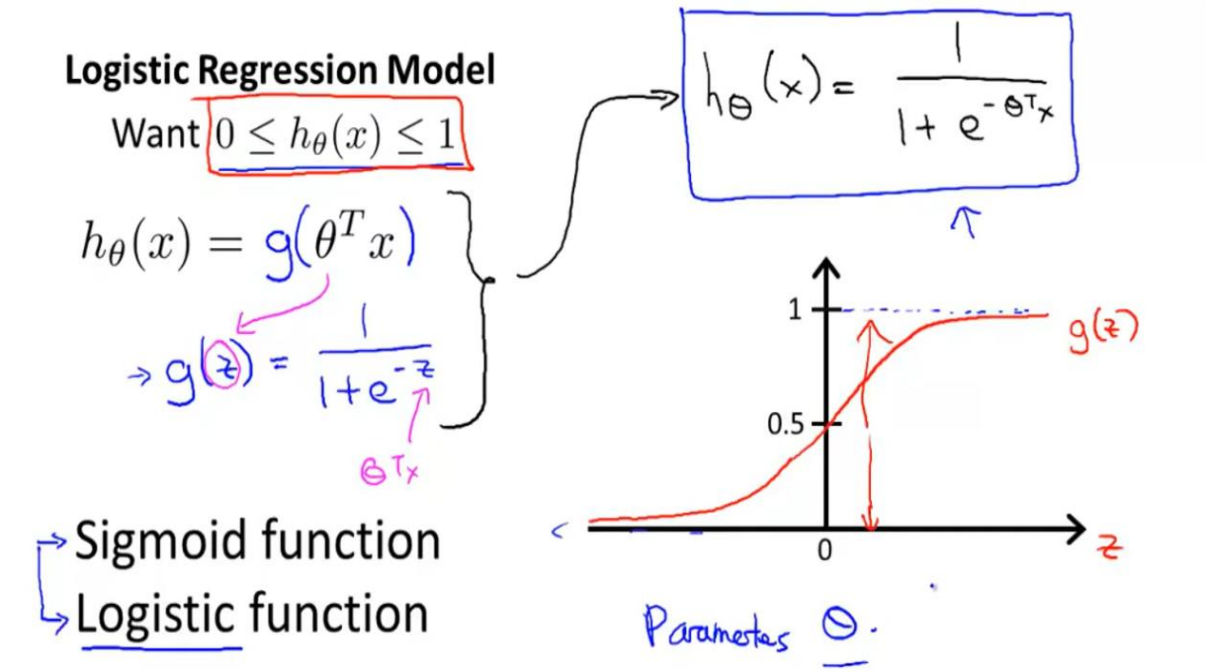
But, what happens is that points B and C which were earlier classified as malignant are tunow benign since the values of tumor predicted by them are <0.5 . This has caused an error in our model.

Another thing that can happen is that though we want of our hypothesis to be 0 or 1 , our hypothesis may predict values of y to be greater than 1 or less than 0.

Logistic Regression: $0 \leq h_\theta(x) \leq 1$

↳ Classification

The logistic regression model:

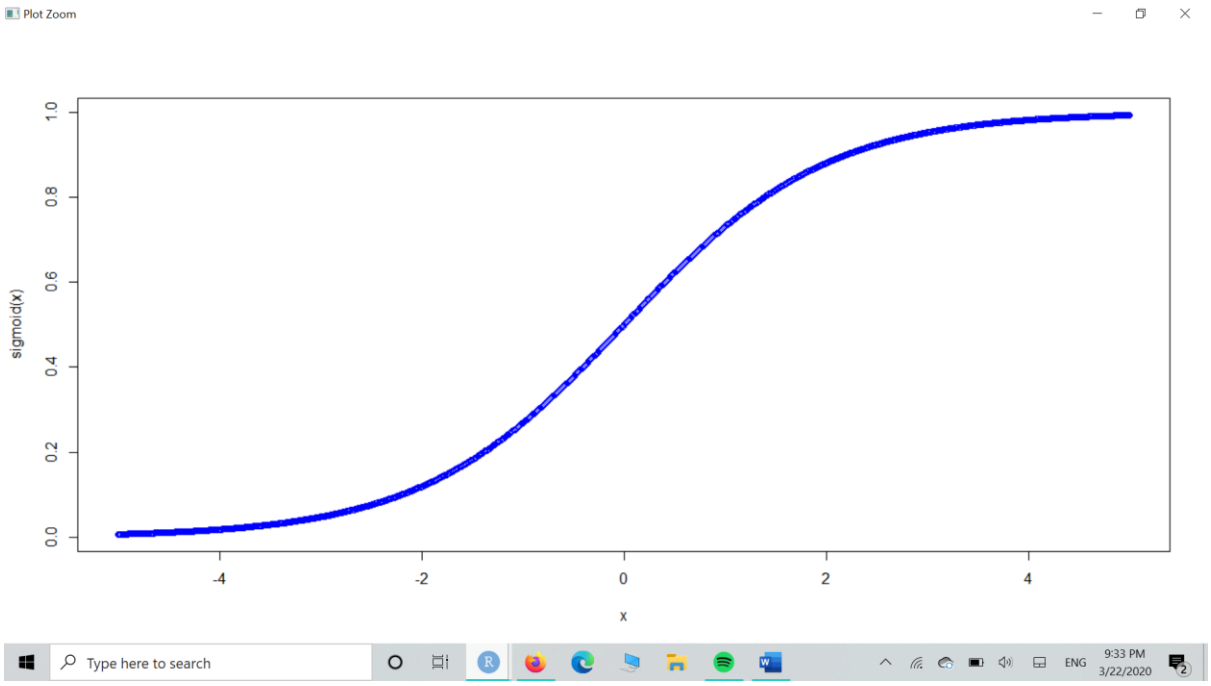


Example of sigmoid function in R:

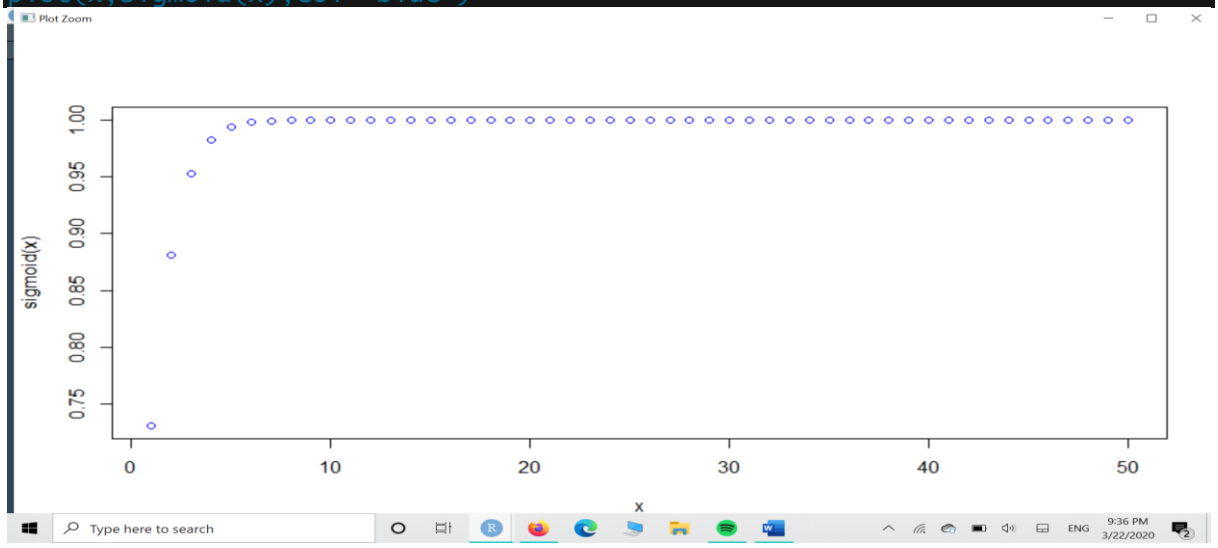
```
sigmoid = function(x) {
    1 / (1 + exp(-x))
}
plot(x,sigmoid(x))

x <- seq(-5, 5, 0.01)

plot(x,sigmoid(x),col='blue')
```



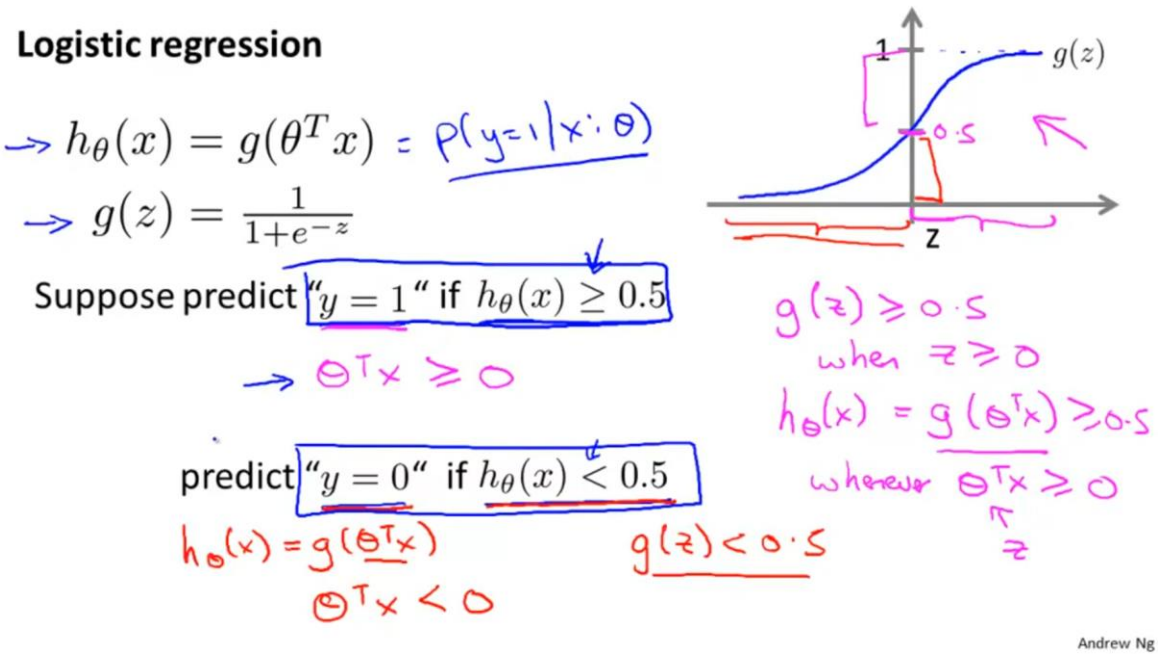

```
x <- seq(1,50)
plot(x,sigmoid(x),col='blue')
```



```
sigmoid(x)
 [1] 0.7310586 0.8807971 0.9525741 0.9820138 0.9933071 0.9975274 0.9990889
0.9996646
 [9] 0.9998766 0.9999546 0.9999833 0.9999939 0.9999977 0.9999992 0.9999997
0.9999999
[17] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
1.0000000
[25] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
1.0000000
[33] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
1.0000000
[41] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
1.0000000
[49] 1.0000000 1.0000000

exp(-20)
[1] 2.061154e-09
```

We see that as values of x keep on increasing , e^{-x} keeps on becoming close to 0 and $1+ e^{-x}$ keeps getting close to 1, hence we see many 1's on the above plot.



Examples predicting the above:

```
sigmoid(-0.23)  ##z<=0 , which => probability of being close to 1 is low.
[1] 0.4427521

 sigmoid(0.3)   ##z>=0 , which => probability of being close to 1 is low.
[1] 0.5744425

 sigmoid(0.82)  ##z>=0 , which => probability of being close to 1 is low.
[1] 0.6942363
```

Interpretation of Hypothesis Output

$h_\theta(x)$ = estimated probability that y = 1 on input x

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x) = 0.7$

Tell patient that 70% chance of tumor being malignant

$h_\theta(x) = P(y=1|x;\theta)$ "probability that y = 1, given x, parameterized by θ"

$y = 0$ or 1

$$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

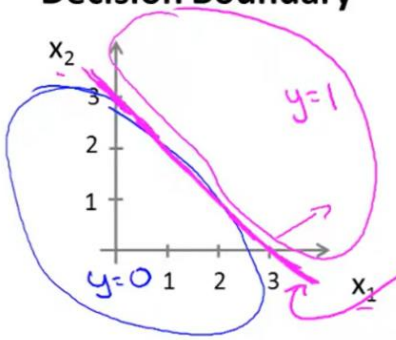
Andrew Ng

$h_\theta(x)$ will give us the **probability** that our output is 1. For example, $h_\theta(x) = 0.7$ gives us a probability of 70% that our output is 1. Our probability that our prediction is 0 is just the complement of our probability that it is 1 (e.g. if probability that it is 1 is 70%, then the probability that it is 0 is 30%).

DECISION BOUNDARY:

Example 1:

Decision Boundary



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

Decision boundary

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

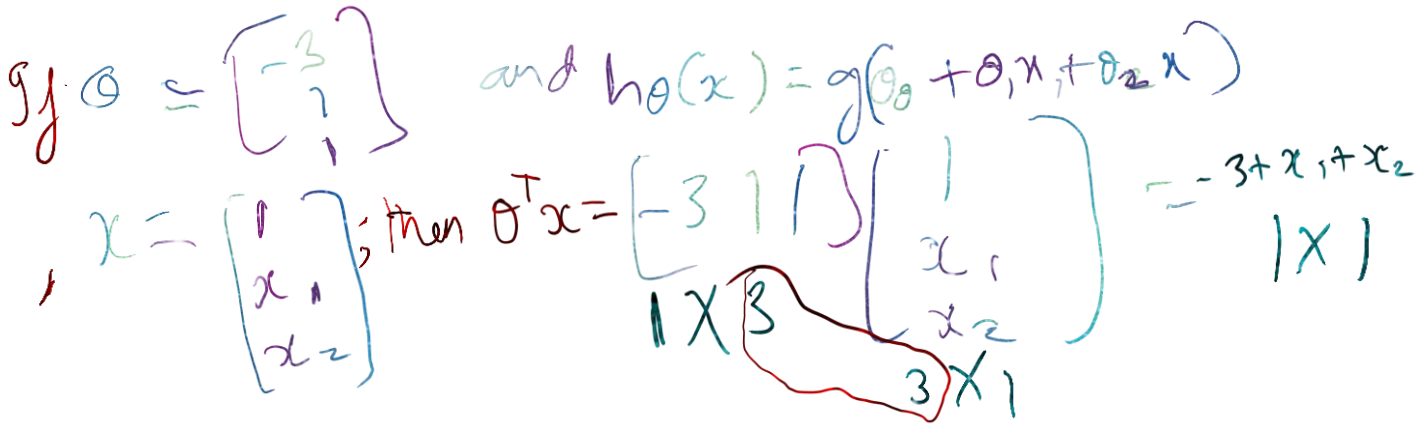
$\theta^T x$

$x_1 + x_2 \geq 3$

x_1, x_2 → $h_\theta(x) = 0.5$ → $x_1 + x_2 = 3$

$x_1 + x_2 < 3$ → $y = 0$

Andrew Ng



The **decision boundary** is the line that separates the area where y = 0 and where y = 1. It is created by our hypothesis function.

Decision boundary gives us a way to segregate values which when fitted into logistic regression equation will have a probability of giving y =1 and y=0. In the above equation, values for which x1 + x2 >= 3 will have a greater probability of giving y=1.

Note: the decision boundary is a property of the hypothesis i.e. the decision boundary is made using the equation obtained from variables in the hypothesis equation , and not by seeing how the data/training set looks on the graph.

Decision Boundary

In order to get our discrete 0 or 1 classification, we can translate the output of the hypothesis function as follows:

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

The way our logistic function g behaves is that when its input is greater than or equal to zero, its output is greater than or equal to 0.5:

$$g(z) \geq 0.5$$
$$when\ z \geq 0$$

Remember.

$$z = 0, e^0 = 1 \Rightarrow g(z) = 1/2$$
$$z \rightarrow \infty, e^{-\infty} \rightarrow 0 \Rightarrow g(z) = 1$$
$$z \rightarrow -\infty, e^{\infty} \rightarrow \infty \Rightarrow g(z) = 0$$

So if our input to g is $\theta^T X$, then that means:

$$h_\theta(x) = g(\theta^T x) \geq 0.5$$
$$when\ \theta^T x \geq 0$$

From these statements we can now say:

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

The **decision boundary** is the line that separates the area where y = 0 and where y = 1. It is created by our hypothesis function.

Example:

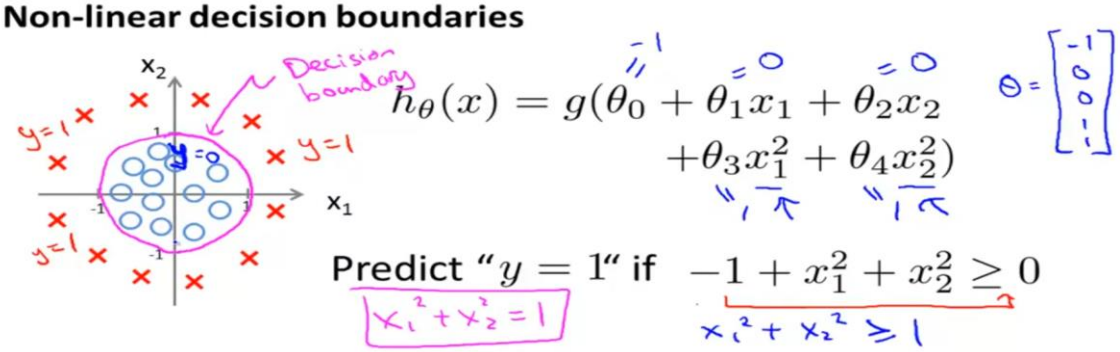
$$\theta = \begin{matrix} 5 \\ -1 \\ 0 \end{matrix}$$
$$y = 1\ if\ 5 + (-1)x_1 + 0x_2 \geq 0$$
$$5 - x_1 \geq 0$$
$$-x_1 \geq -5$$
$$x_1 \leq 5$$

In this case, our decision boundary is a straight vertical line placed on the graph where $x_1 = 5$, and everything to the left of that denotes y = 1, while everything to the right denotes y = 0.

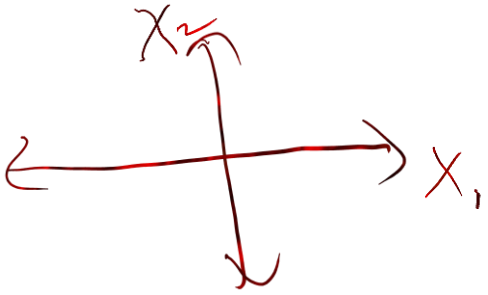
NOTE:

Again, the input to the sigmoid function g(z) (e.g. $\theta^T X$) doesn't need to be linear, and could be a function that describes a circle (e.g. $z = \theta_0 + \theta_1 x_1^2 + \theta_2 x_2^2$) or any shape to fit our data.

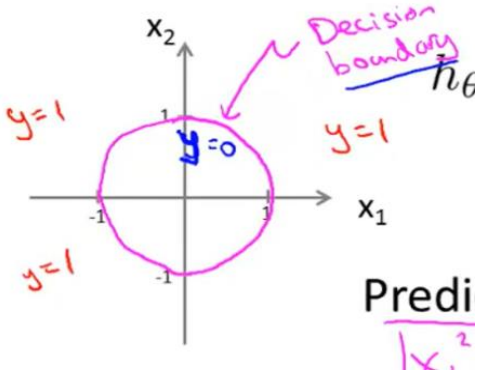
Example 2:



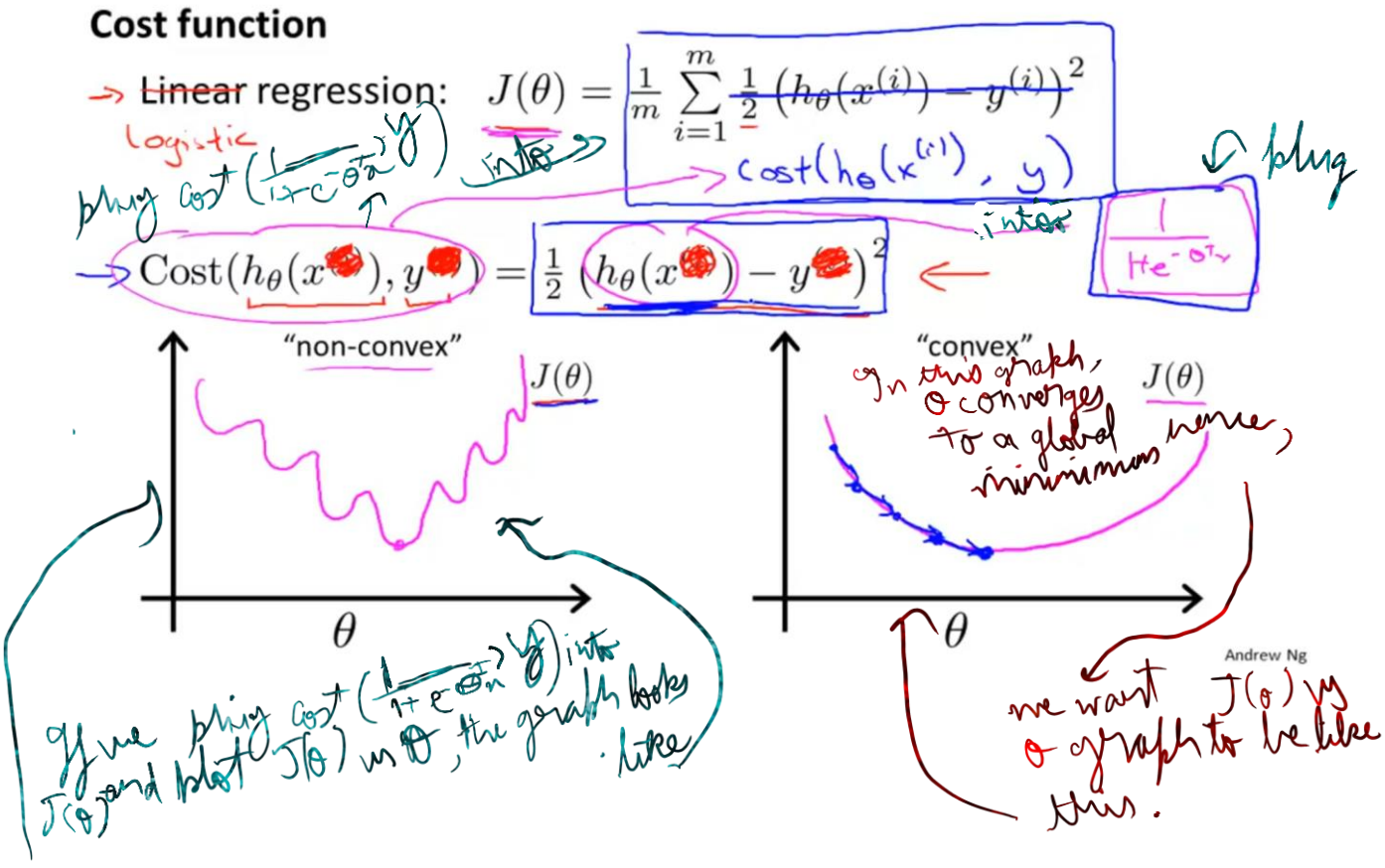
In the above plot, the decision boundary is not made by seeing how the training set/ data looks on the graph but it is made by using the hypothesis equation. Even if we had a plain graph like this:



We could still plot a decision boundary using the equation of hypothesis $x_1^2 + x_2^2 >= 1$:



Cost Function For Logistic Regression:

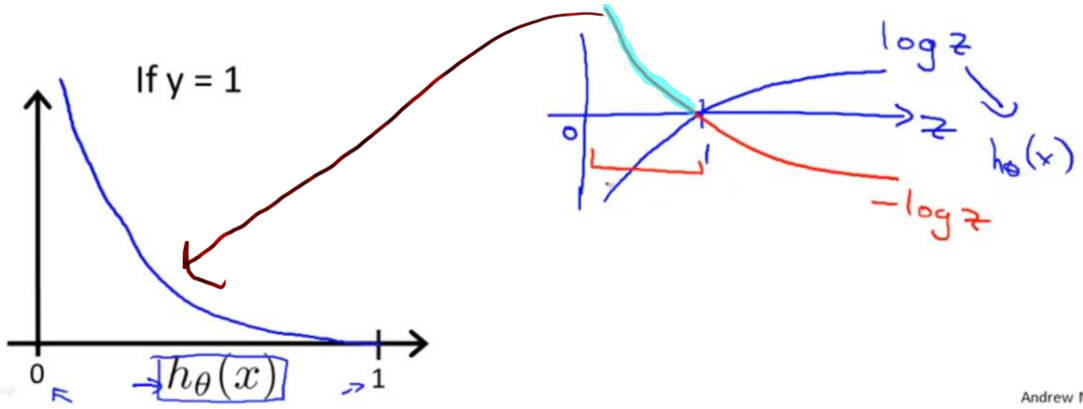


If we were to run gradient descent on this sort of function It is not guaranteed to converge to the global minimum.

Actual cost function for logistic regression:

Logistic regression cost function

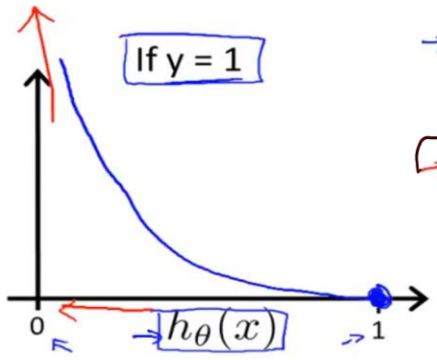
$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Andrew Ng

Logistic regression cost function

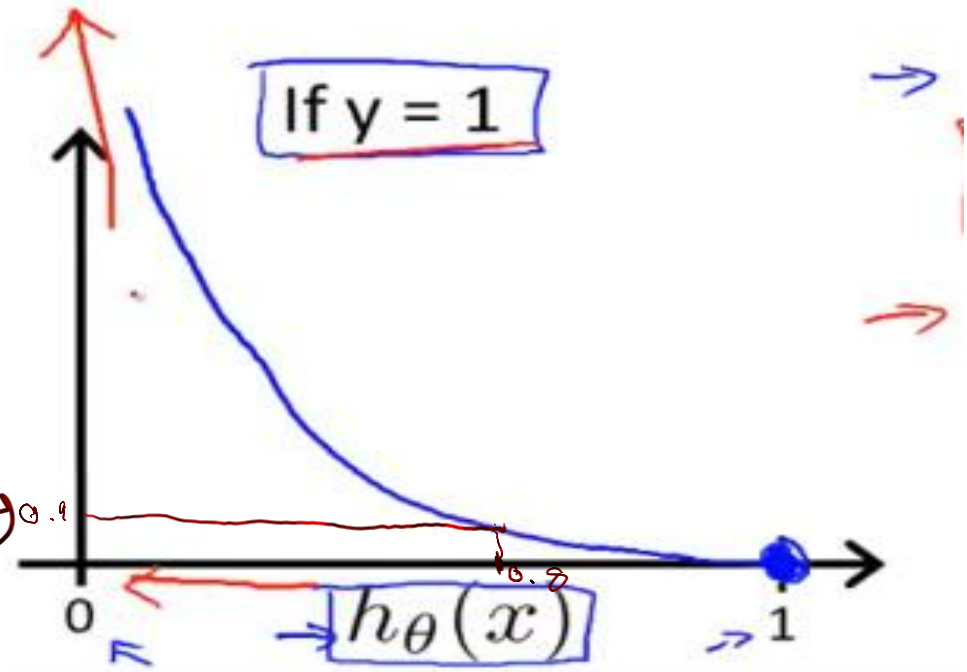
$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Cost = 0 if $y = 1, h_\theta(x) = 1$
But as $h_\theta(x) \to 0$
$Cost \to \infty$

Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x; \theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.

Say we predict that probability that a person has a malignant tumor is 0, that is, P(y=1|x;θ)=0 . However, the actual test results show that the person does have a tumor. In such a case, where our prediction is completely wrong, we penalize our cost function by a very large amount.

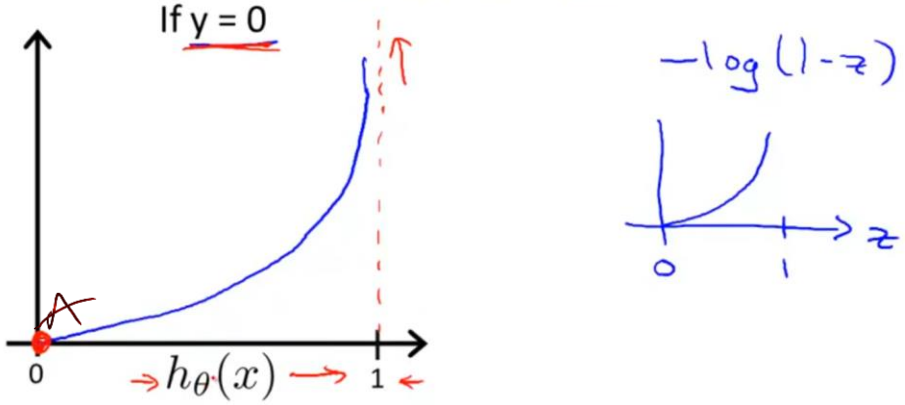


Say we predict that probability that a person has a malignant tumor is 0.8, that is, P(y=1| x;θ)=0.80. Now, the actual test results also show that the person does have a tumor. In such a case, where our prediction is 80% right, we penalize our cost function by a very less amount.

Plot for $-\log(1-h_\theta(x))$ used when $y=0$

Logistic regression cost function

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

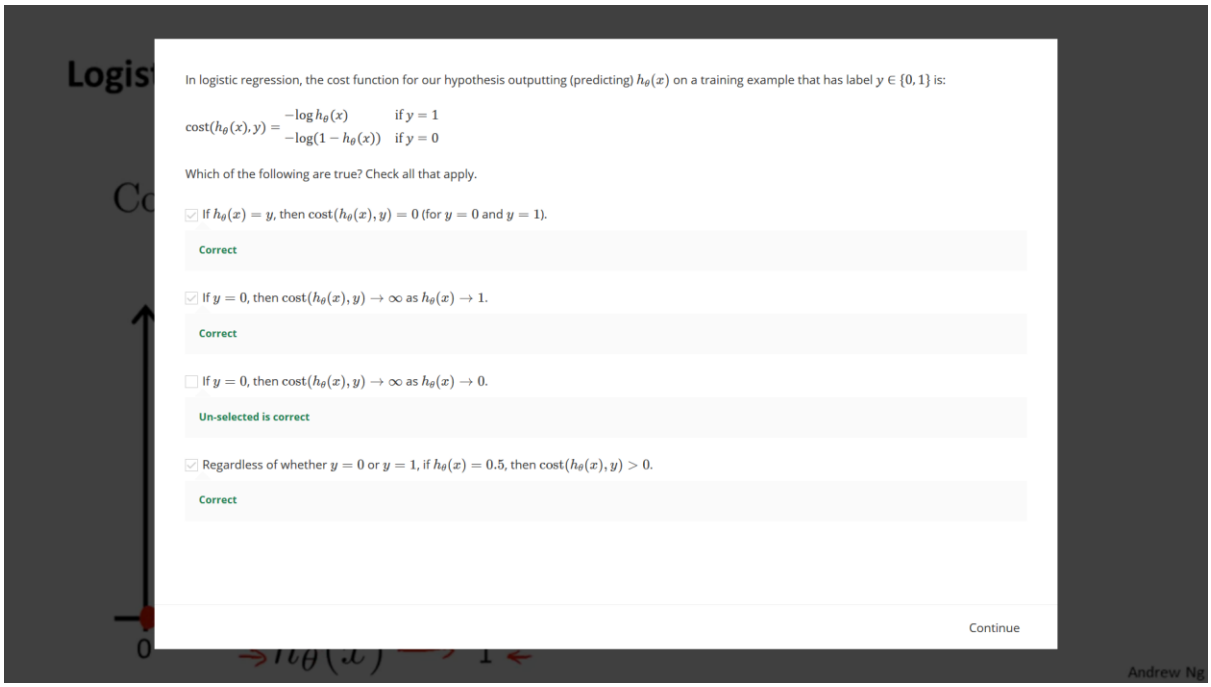


At point A, value of cost fnc^n = 0

Say we predict that probability that a person has a malignant tumor is 0, that is, P(y=0|x;θ)= 1 . The actual test results also show that the person does not have a tumor. In such a case, where our prediction is completely right, we do not penalize our cost function at all.

NOTE: In both the graphs above , h(θx) tells us the probability that we predicted an outcome to be 0 and 1 . If plot a curve of -log(theta$_x$) i.e. y=1 and predict probability y=1 to be 0.9 , then our prediction is quite right and value of cost function very less.

QUESTION:



Other way of writing the cost function:

$$Cost(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y)$$

Logistic regression cost function

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

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

Note: $y = 0$ or 1 always

$$Cost(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y) \log(1 - h_\theta(x))$$

If $y=1$: $Cost(h_\theta(x), y) = -\log h_\theta(x)$

If $y=0$: $Cost(h_\theta(x), y) = -\log(1-h_\theta(x))$

Final Cost function:

Logistic regression cost function

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

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

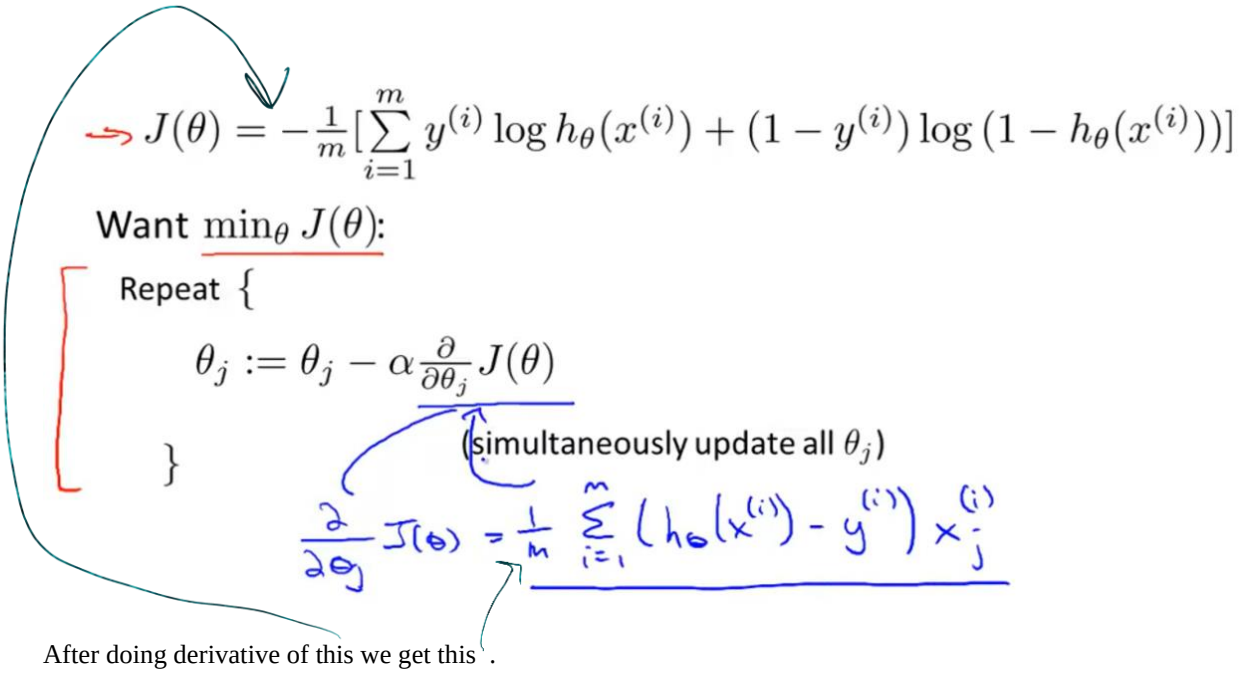
To fit parameters θ:

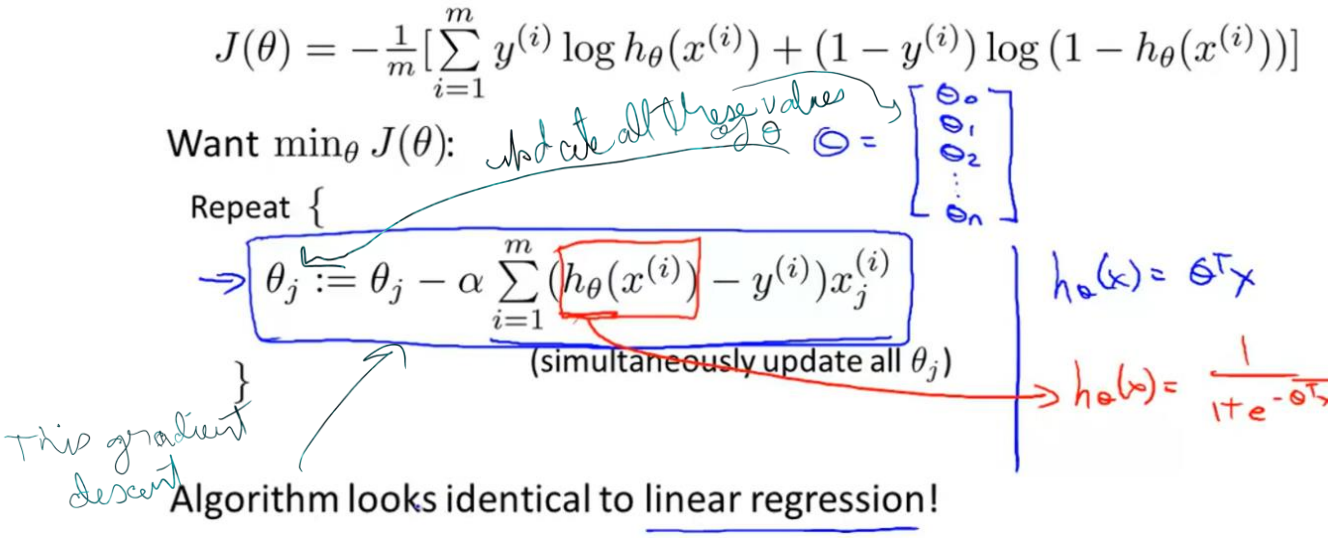
$$\min_\theta J(\theta)$$ Get θ

To make a prediction given new x: we use:

Output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ $\quad p(y=1 \mid x; \theta)$

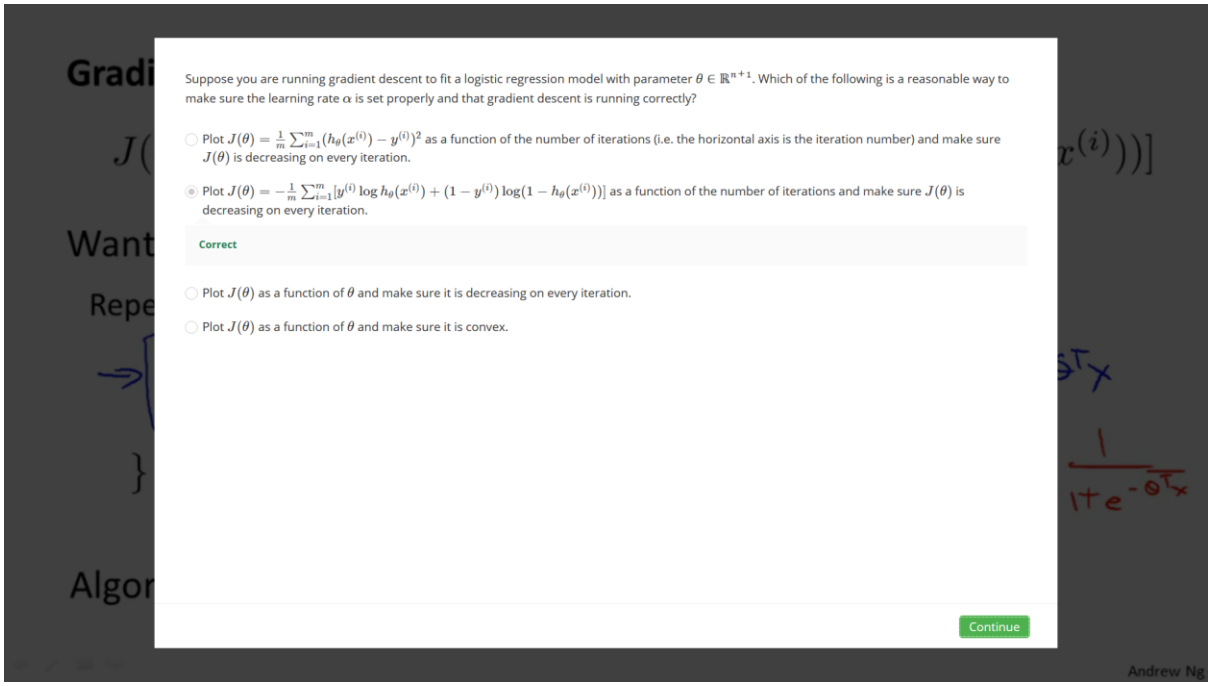
This output gives the probability of getting 1 given x and a parameter theta corresponding to it.


$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$
Want $\min_\theta J(\theta)$:
Repeat {
$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$
}
(simultaneously update all θ_j)
$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$
After doing derivative of this we get this .

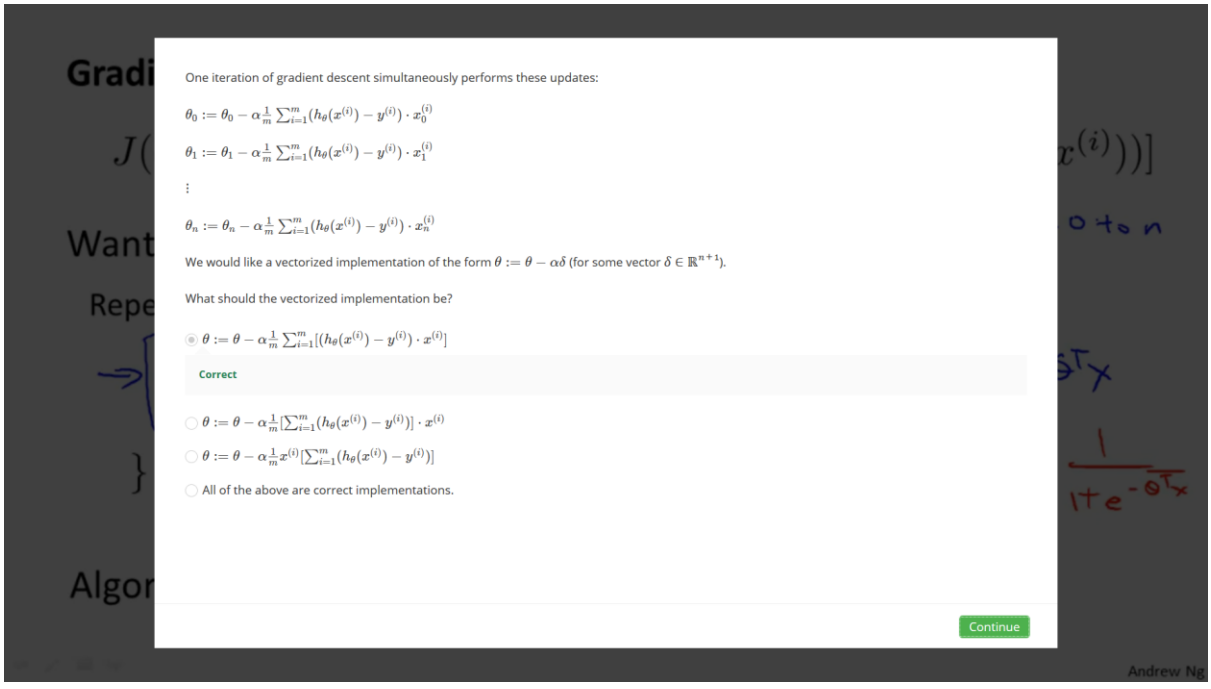

$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$
Want $\min_\theta J(\theta)$:
Repeat {
$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$
(simultaneously update all θ_j)
}
update all these values
$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$
$h_\theta(x) = \theta^T x$
$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$
this gradient descent
Algorithm looks identical to linear regression!

Questions:

Q1:



Q2:



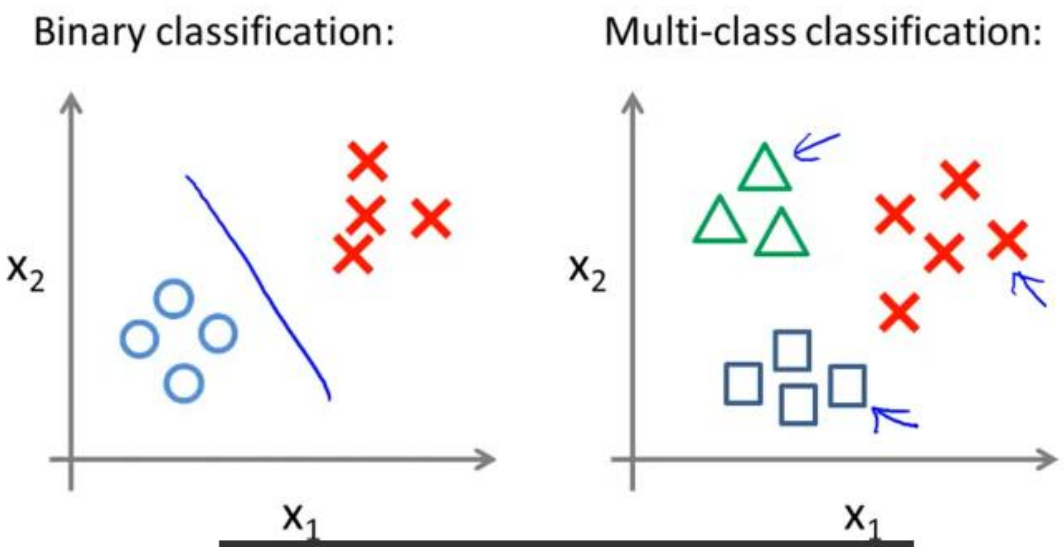
Multiclass Classification: One – vs- all classification

In this topic , we try to classify multivariate data , i.e. when we have to classify data into 3 or more types.

Say we have to classify a weather as : Sunny , Cloudy or Rainy

Here we represent these weather types by :-

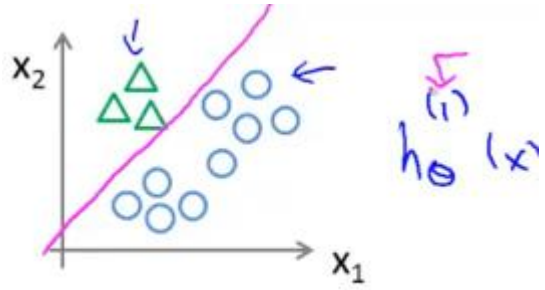
Binary classification vs Multi-class classification:



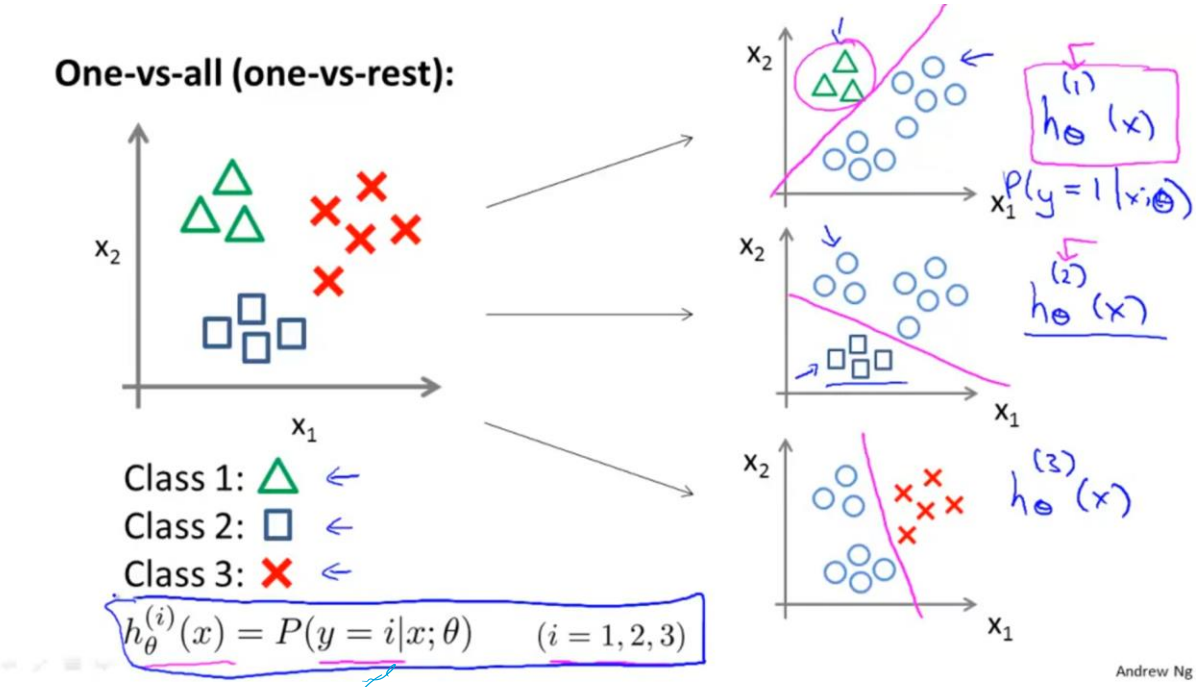
Using the one vs all method:

We turn above multi-class classification problem into three separate two class classification problems.

So let's start with class one which is the triangle. We're going to essentially create a new sort of fake training set where classes two and three get assigned to the negative class. And class one gets assigned to the positive class. You want to create a new training set like that shown on the right, and we're going to fit a classifier which I'm going to call h subscript theta superscript one of x where here the triangles are the positive examples and the circles are the negative examples. So think of the triangles being assigned the value of one and the circles and squares assigned the value of zero. And we're just going to train a standard logistic regression classifier and maybe that will give us a position boundary that looks like this:

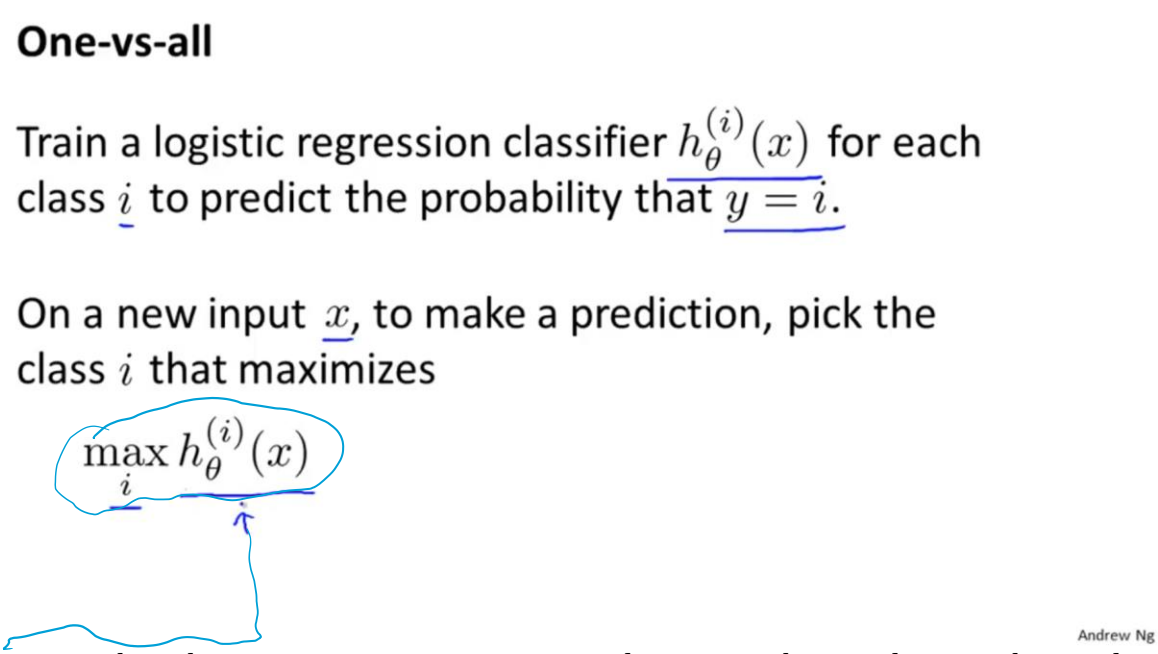


We do the same for squares and crosses and obtain the following diagram:



This equation represents that we now have three classifiers, each of which was trained to recognize one of the three classes.

Summarising one-vs-all method:



This means that when we're given a new input x, and want to make a prediction, what we do is we just run all of our classifiers on the input x and we then pick the class i that maximizes the three. So, we just basically pick the classifier, which is the most confident. So whichever value of i gives us the highest probability we then predict y to be that value.