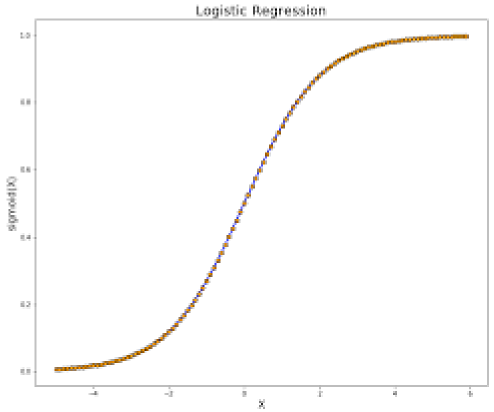
Machine Learning

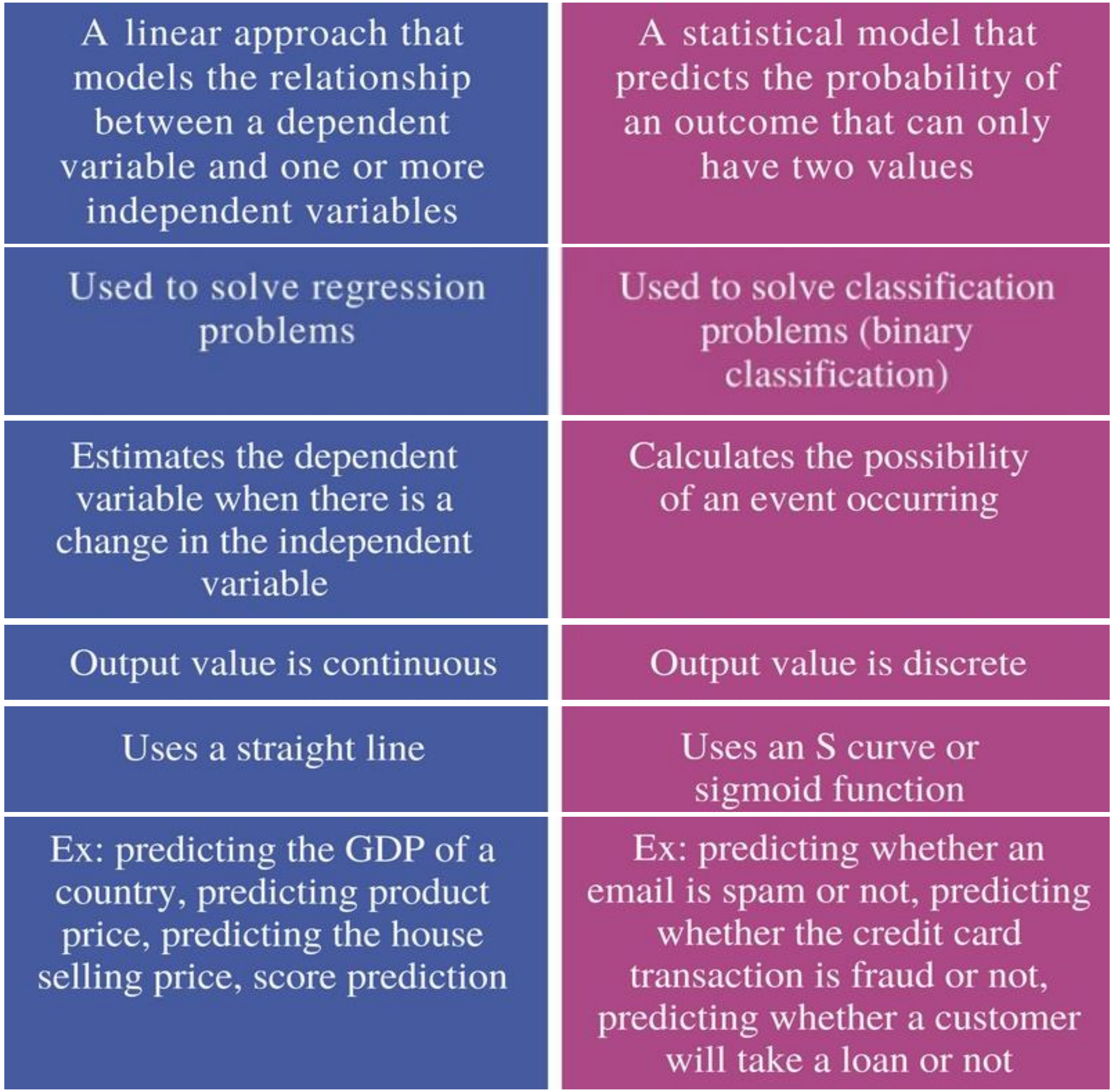
Logistic Regression



What is Logistic Regression?

- **Logistic regression** is basically a supervised **classification** algorithm.
- **Regression and classification** are categorized under the same umbrella of supervised machine learning.
- **Logistic regression** becomes a **classification** technique only when a decision threshold is brought into the picture.

Linear Regression Vs Logistic Regression



A linear approach that models the relationship between a dependent variable and one or more independent variables	A statistical model that predicts the probability of an outcome that can only have two values
Used to solve regression problems	Used to solve classification problems (binary classification)
Estimates the dependent variable when there is a change in the independent variable	Calculates the possibility of an event occurring
Output value is continuous	Output value is discrete
Uses a straight line	Uses an S curve or sigmoid function
Ex: predicting the GDP of a country, predicting product price, predicting the house selling price, score prediction	Ex: predicting whether an email is spam or not, predicting whether the credit card transaction is fraud or not, predicting whether a customer will take a loan or not

Classification

Two types

- Two Class Classification/Binary Classification :

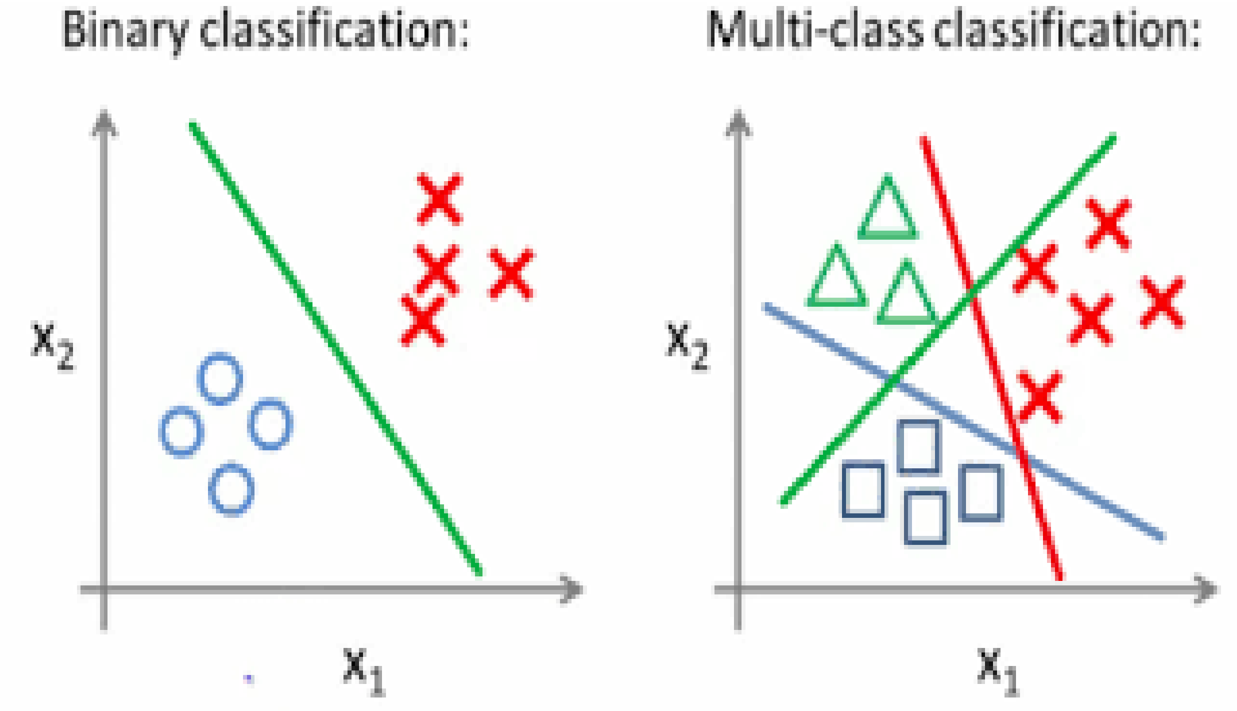
  takes two values 0 or 1

  - Examples
    - Spam or Not Spam
    - Cancer Malignant or Benign
    - Rain Yes or No
    - Admission Yes or No

- Multi Class Classification Problem :

  takes more than 2 values

  - Examples
    - Identifying Digits
    - Identifying some images
    - Identifying some categories

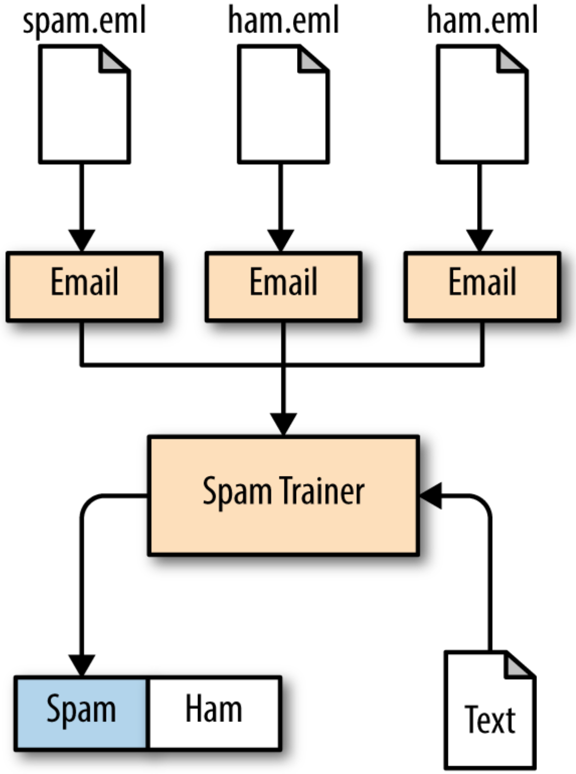


Classification

- The variable to be predicted, will take two values either 1 or 0
  - 1 Positive Class
  - 0 Negative Class

Example

- For instance, if we are trying to build a spam classifier for email, then x^i may be some features of a piece of email, and y may be 1 if it is a piece of spam mail, and 0 otherwise.

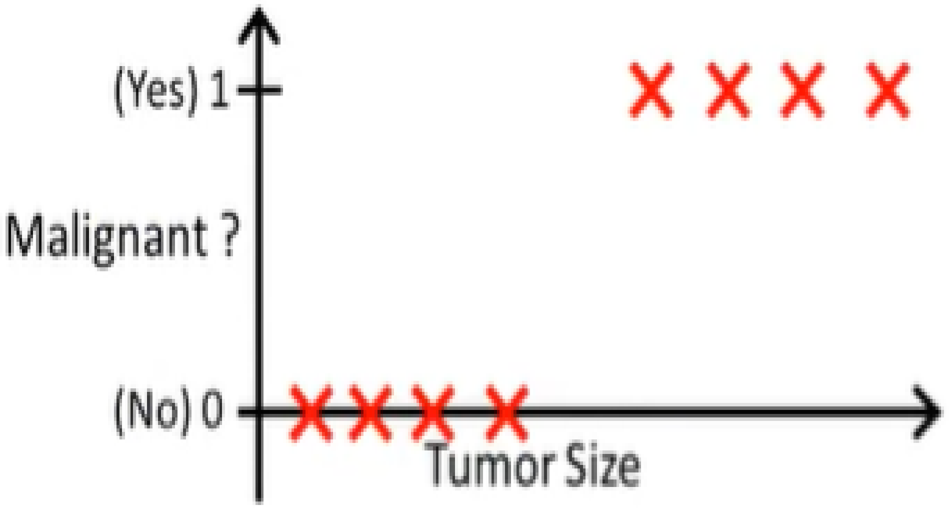


Classification Problem

- The classification problem is just like the regression problem, except that the values we now want to predict take on only a small number of discrete values.
- 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.
- Lets see why?

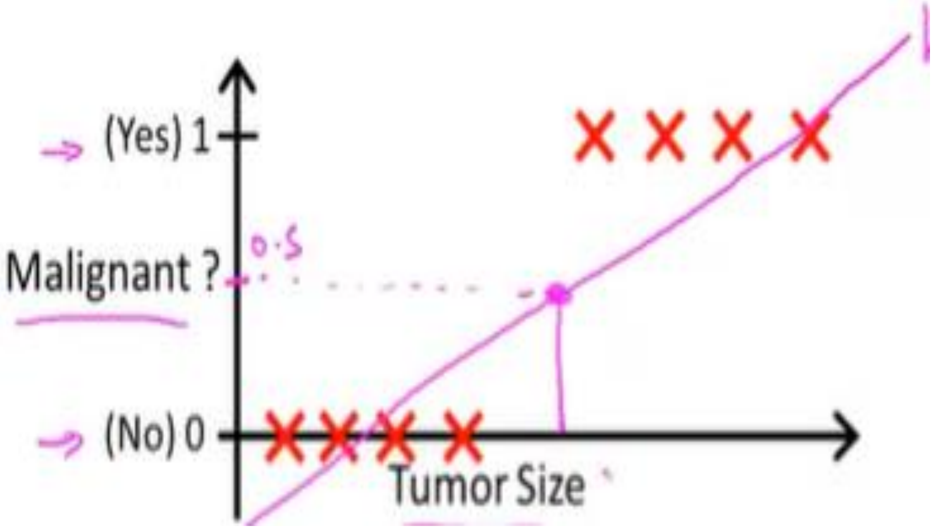
Classification Problem

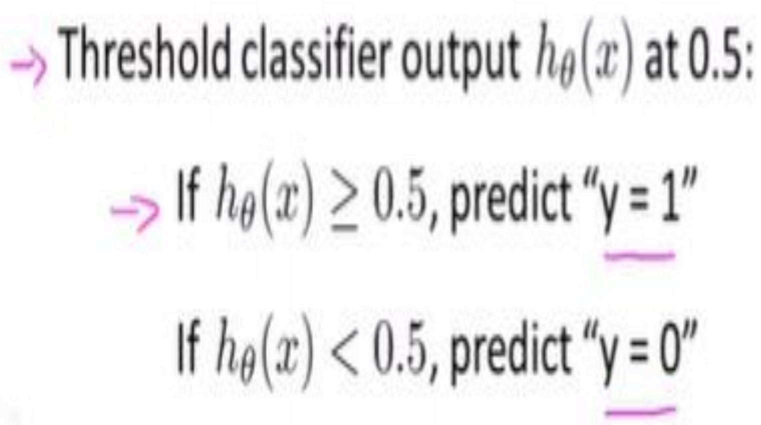
- Malignancy takes only two values 0 or 1



Applying Linear Regression

- If we apply linear regression to this problem, and fit straight line to the data, we may get a hypothesis like this.





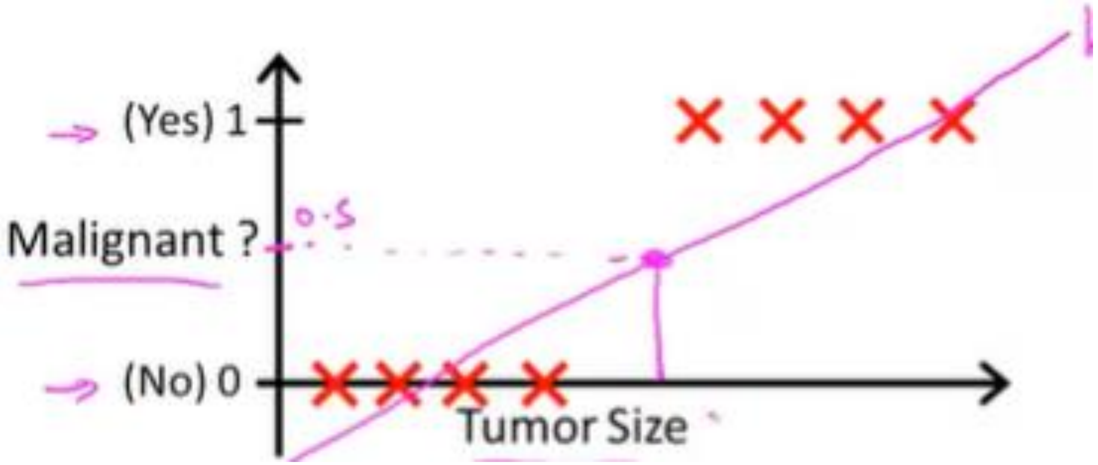
Threshold classifier output $h_\theta(x)$ at 0.5:

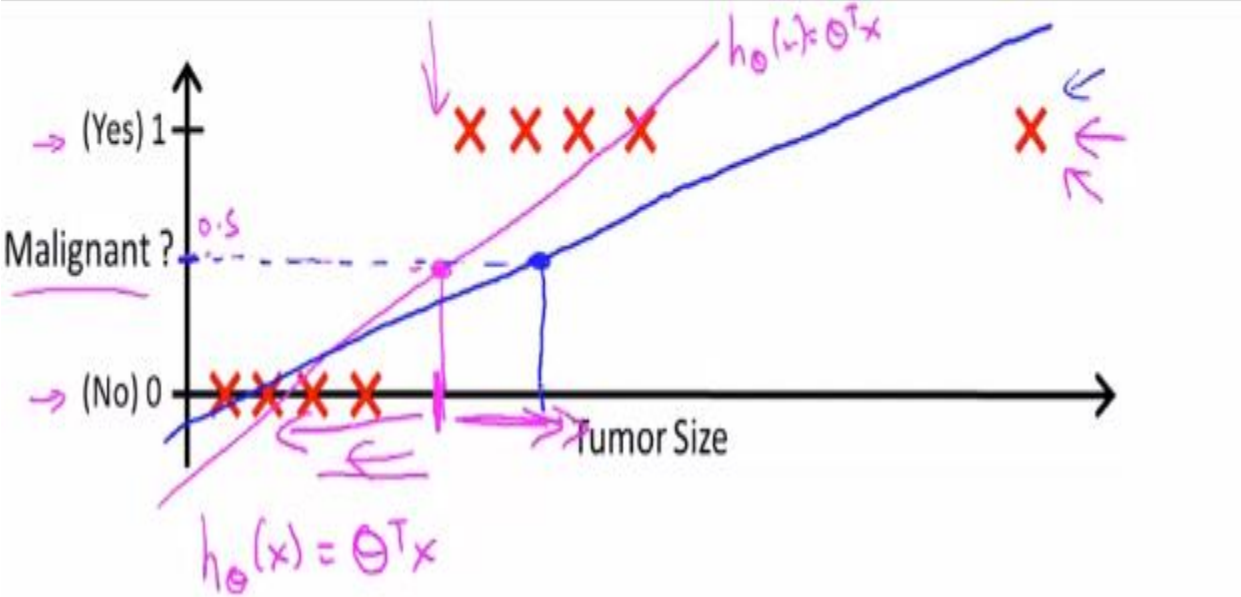
If $h_\theta(x) \geq 0.5$, predict "y = 1"

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

Applying Linear Regression

- But when we have one more example, Linear regression line changes, and observe the dividing line. Now this hypothesis is unfit and not correct.
- Therefore linear regression cannot be applied to classification problems.





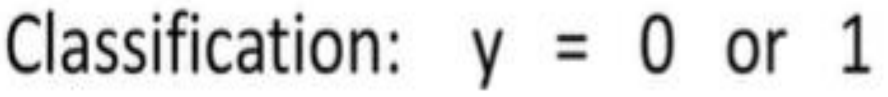
Applying Linear Regression

- Therefore Linear regression is not suitable for Classification Problem.

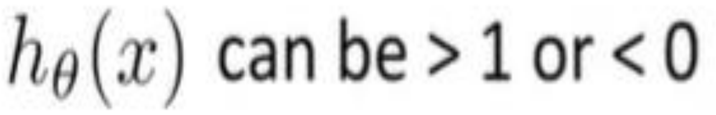
- Another problem of applying Linear Regression hypothesis is that it may give answer > 1 or < 0 which is not required in classification problem.

Logistic Regression

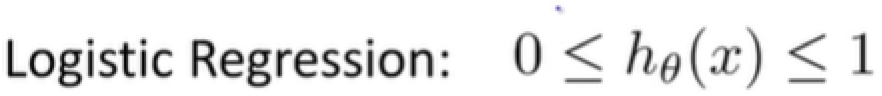
- Logistic Regression is the algorithm having the property that its predictions are always between 0 and 1.



Classification: y = 0 or 1



$h_\theta(x)$ can be > 1 or < 0



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

Logistic Function / Sigmoid Function

- Form of hypothesis in Logistic Regression is :

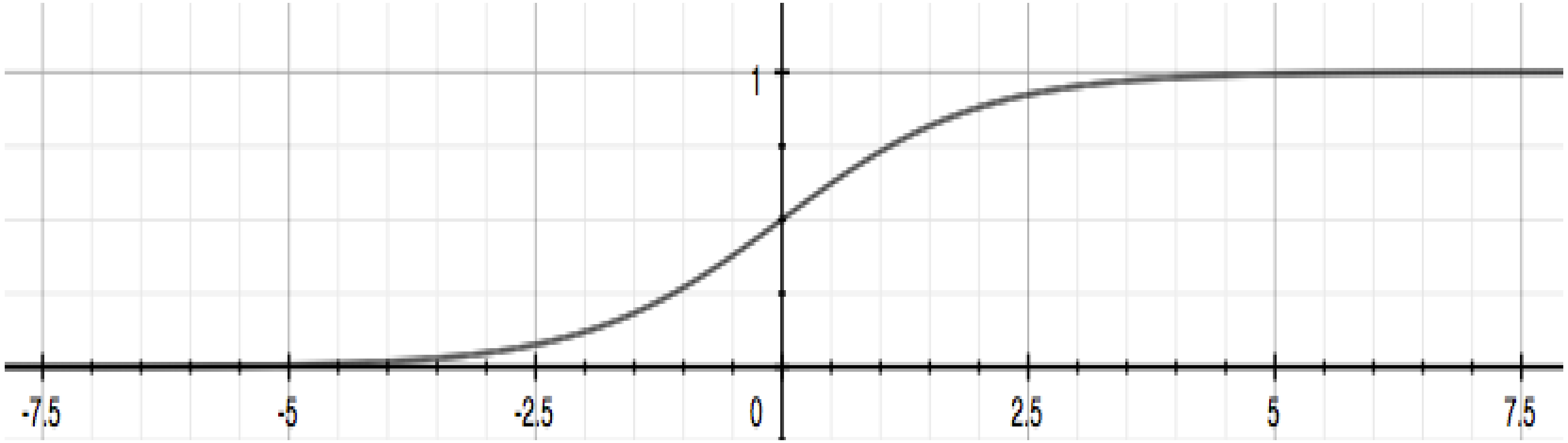
$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) = g(\theta^T x)$$

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

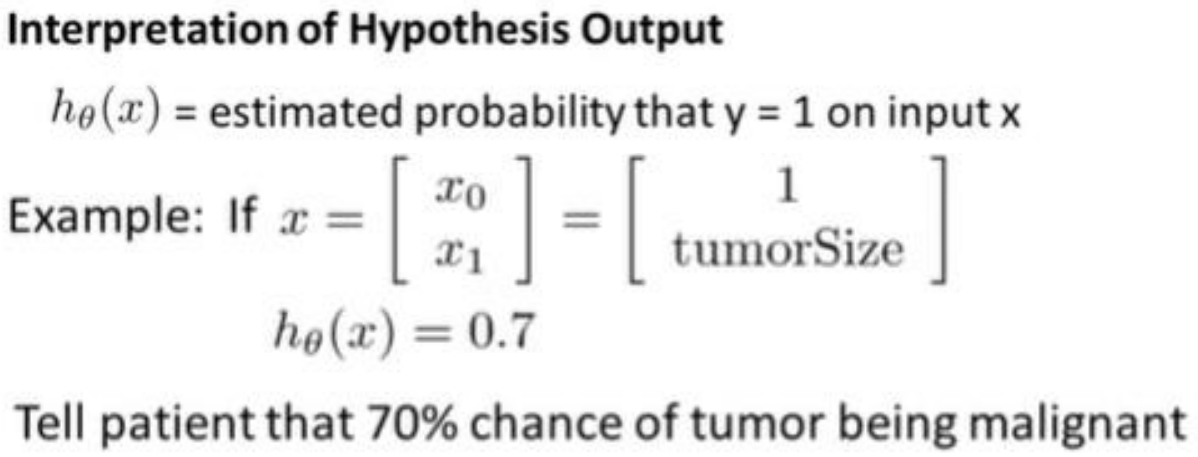
$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

Logistic Function / Sigmoid Function



- The function g(z) maps any real number to the (0, 1) interval, making it useful for transforming an arbitrary-valued function into a function better suited for classification.
- It slowly increases from 0 to 1 asymptoting at 1

Hypothesis Output



Interpretation of Hypothesis Output

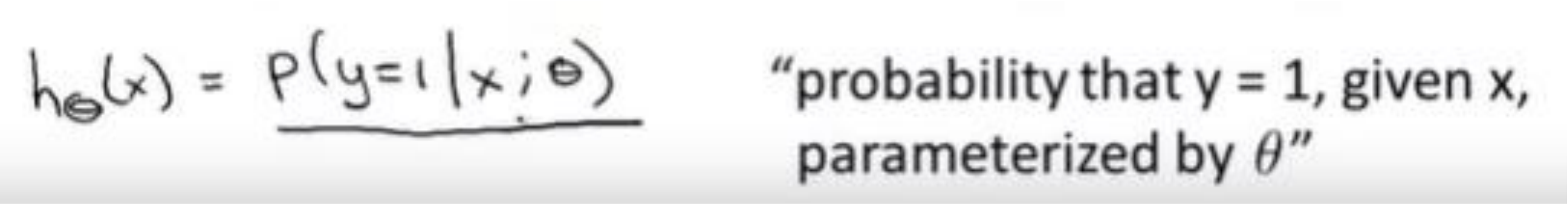
$h_\theta(x)$ = estimated probability that y = 1 on input x

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

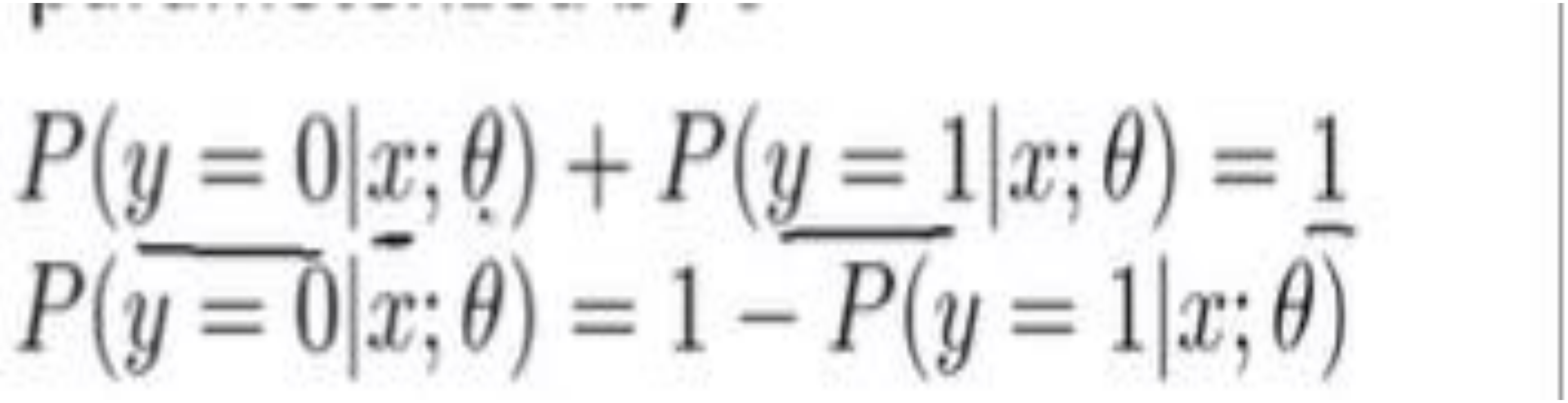
For a patient with features x, the probability that y equals 1 is 0.7, ie 70% chance or 0.7 chance of being malignant. Written mathematically as follows:



$$h_\theta(x) = \underline{P(y=1|x;\theta)}$$

"probability that y = 1, given x, parameterized by θ"

Hypothesis Output



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

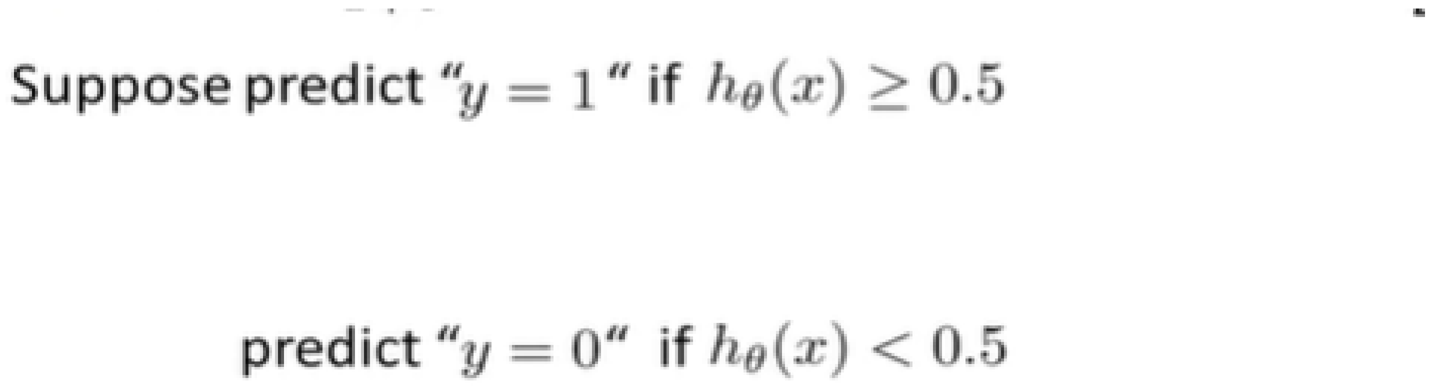
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

Probabiity of y=0 and probability of y=1 must add upto 1 .

First equation says probability of y=0 for a particular patient with features x and given parameters Theta plus the probability of y=1 for that same patient with features x and given theta parameters, theta must add upto 1.

Decision Boundary

- Decision Boundary gives an idea of what Regression hypothesis function is computing

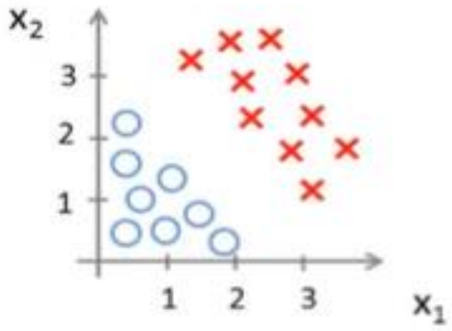


Suppose predict “$y = 1$“ if $h_\theta(x) \geq 0.5$

predict “$y = 0$“ if $h_\theta(x) < 0.5$

- g(z) equal to 0.5 whenever z >= 0
- So when z is positive, g of z , sigmoid function is >= 0.5
- Hypothesis will predict y =1 whenever theta transpose x is >=0
- Hypothesis will predict y=0 whenever theta transpose x is < 0

Applying Logistic Regression : Decision Boundary

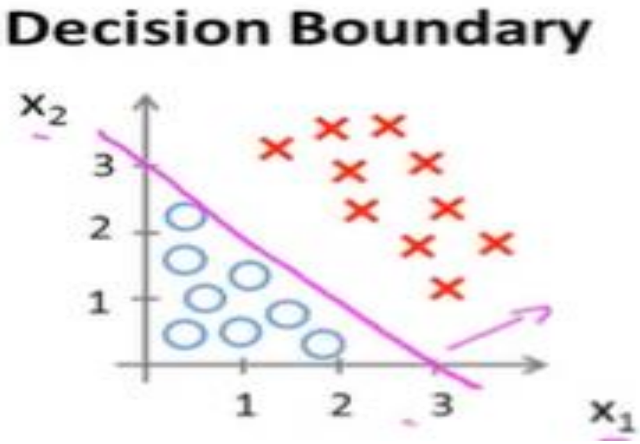


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

Choose theta0 = -3, theta1 = 1, theta2 = 1. Therefore parameter vector is theta equals -3, 1, 1.

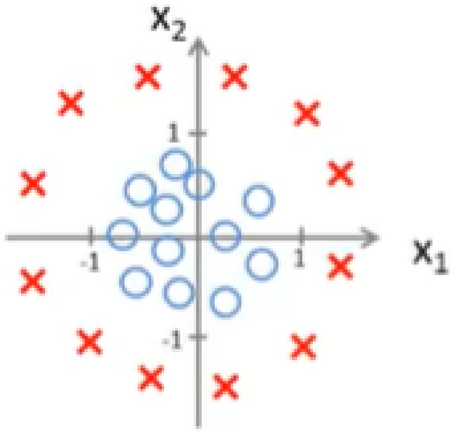
So for any example, which features x1 and x2 that satisfy this equation, that -3 + x1 + x2 >=0, our hypothesis will think that y equals 1

We can rewrite x1+x2 >= 3, therefore y=1 whenever x1+x2 >= 3


Decision Boundary

Non Linear Decision boundaries

- If Theta0 =-1 and Theta1=0 and Theta2=0 and Theta3=1 and Theta4=1



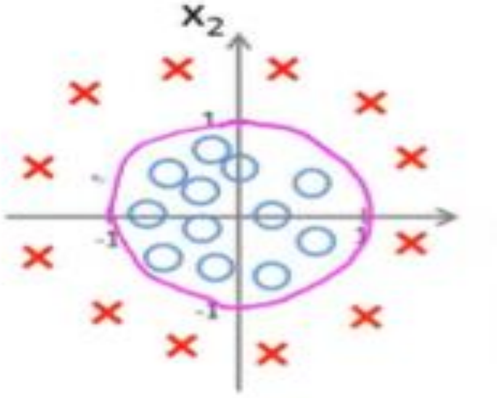
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 \\ + \theta_3 x_1^2 + \theta_4 x_2^2)$$

Predict “$y = 1$“ if $-1 + x_1^2 + x_2^2 \geq 0$

- this means that my hypothesis will predict that
- y=1 whenever -1 + x1 squared + x2 squared is greater than or equal to 0.
- This is whenever theta transpose times my theta transfers,
- my features is greater than or equal to zero.

Non Linear Decision boundaries

- On plotting the curve for x1 squared plus x2 squared=1
- This is the equation for circle of radius one, centered around the origin.
- So that is the decision boundary, everything outside the circle is predicted as y=1 everything inside the circle is predicted as y=0.
- So by adding these more complex, or these polynomial terms to the features , you get more complex decision boundaries like a circle.



Non Linear Decision boundaries

But the training set is not what we use to define the decision boundary.

The training set may be used to fit the parameters theta.

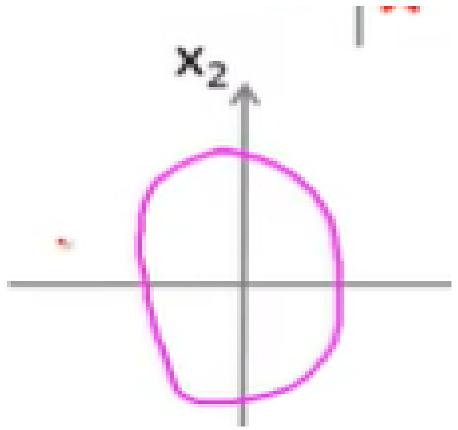
Once you have the parameters theta,one can define the decisions boundary.

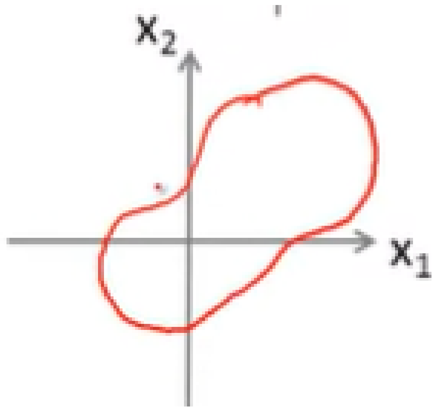
The training set just for visualization.

Non Linear Decision boundaries

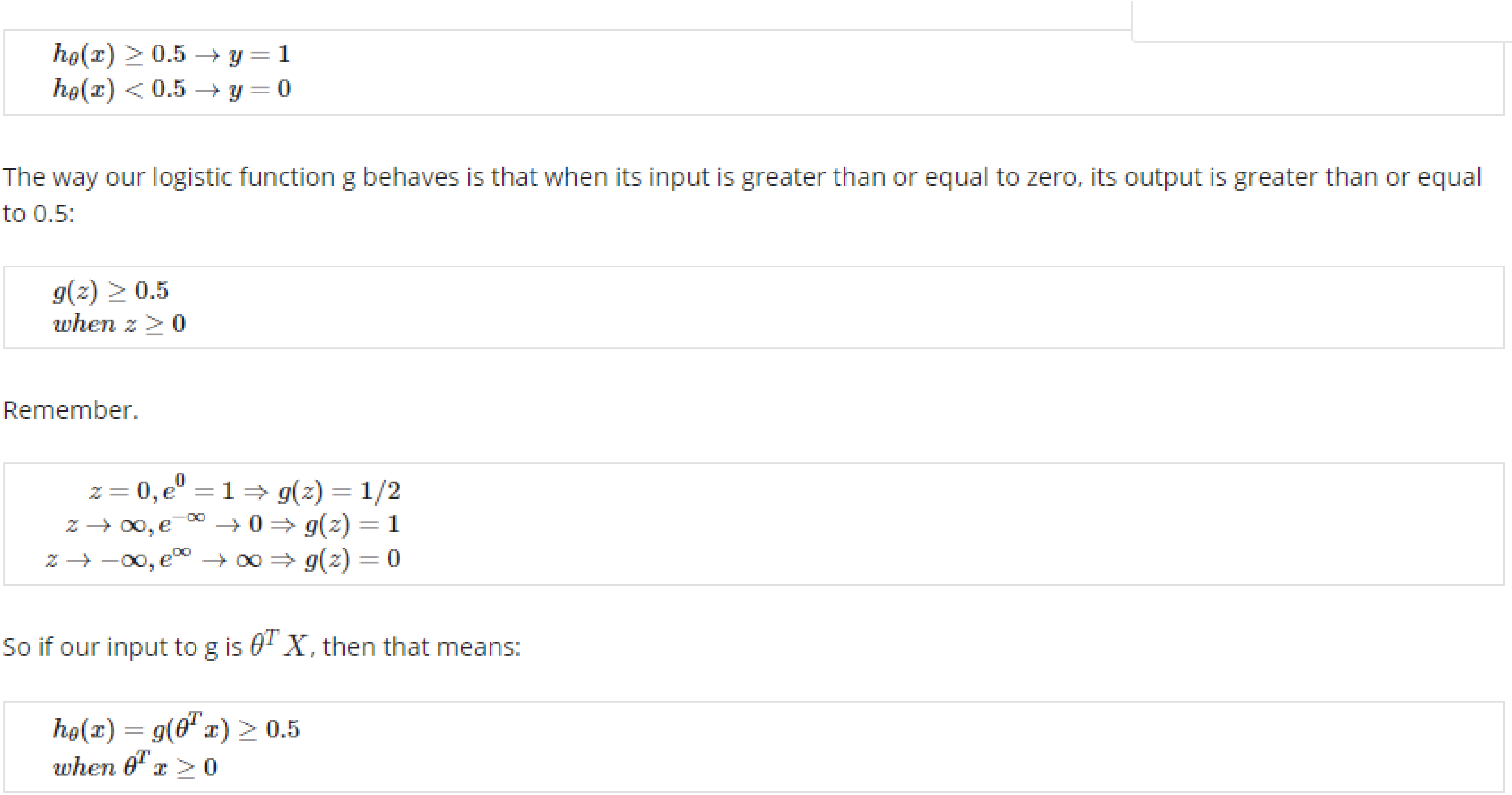
There can be higher polynomials, then it's possible to show that you can get even more complex decision boundaries and the regression can be used to find decision boundaries that may, for example, be an ellipse. For even more complete examples maybe you can also get this decision boundaries that could look like more complex shapes.

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 \underline{x_1^2}$$
$$+\theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \dots)$$





Conclusion : Non Linear Decision boundaries



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

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:

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

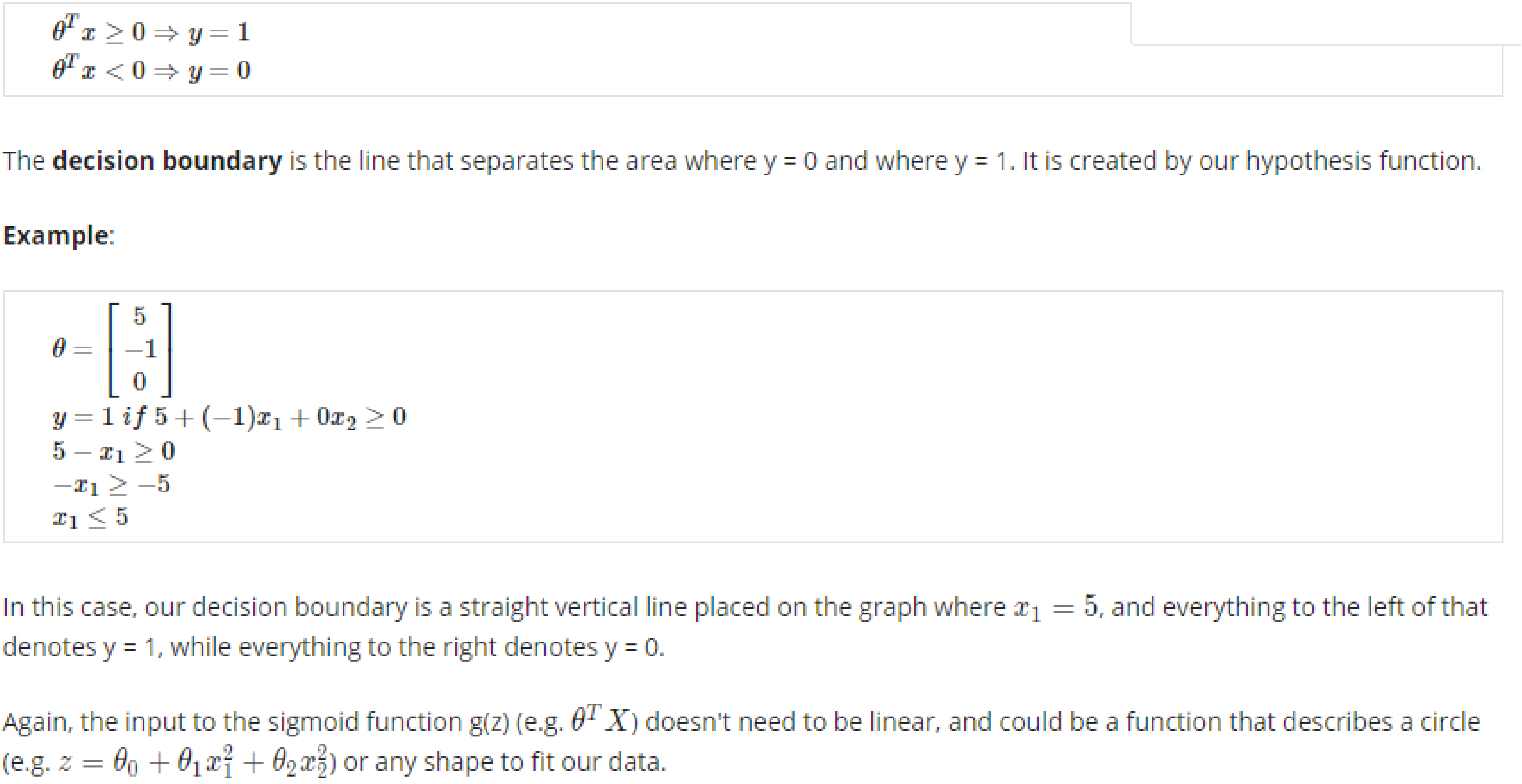
Remember.

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

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

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

Non Linear Decision boundaries



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

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:

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

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.

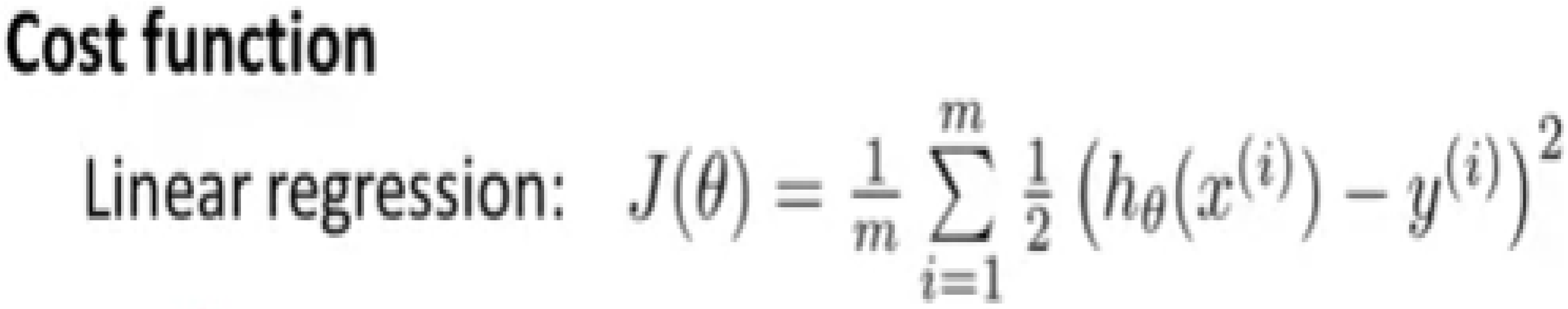
Again, the input to the sigmoid function g(z) (e.g. $\theta^T X$) doesn't need to be linear, and could be a function that describes a circle (e.g. $z = \theta_0 + \theta_1 x_1^2 + \theta_2 x_2^2$) or any shape to fit our data.

Cost Function in Logistic Regression

- After knowing what is h(x) like, our goal is to automatically choose the parameters Theta, so that given a training set, we can automatically fit the parameters to our data.

- We cannot use the same cost function that we use for linear regression because the Logistic Function will cause the output to be wavy, causing many local optima. In other words, it will not be a convex function.

Cost Function in Logistic Regression



Cost function

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

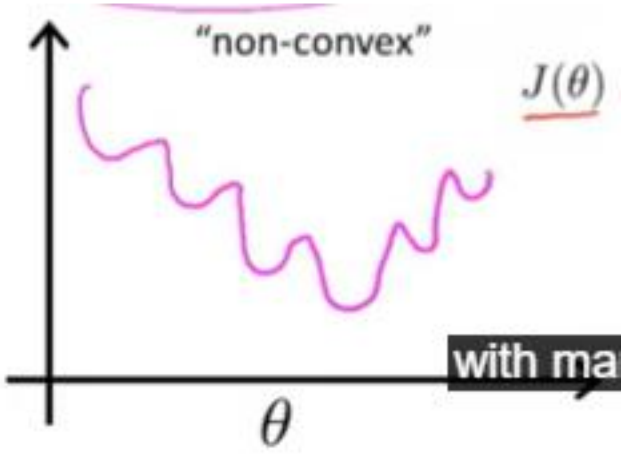
Instead of 1/2m, taken slightly differently 1/m outside. We write alternately.

$$\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

If we use this particular cost function, it would be a non-convex function of the parameters data.

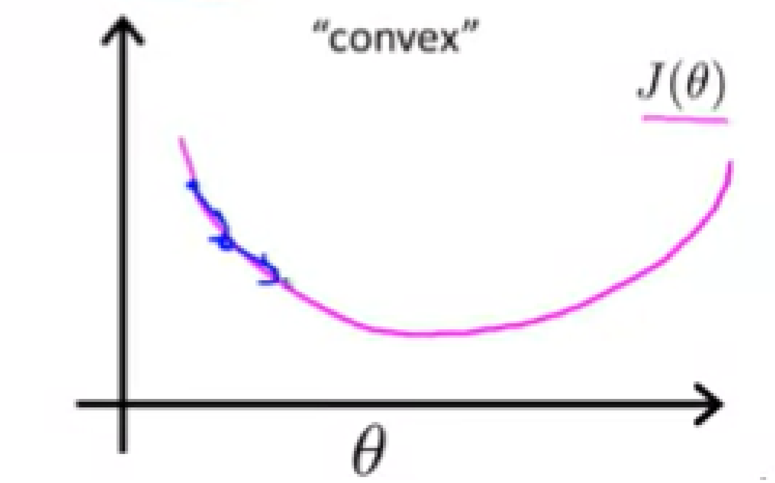
Cost Function in Logistic Regression

- Actually the z in g(z) or h(x) is has non-linearity
- J of Theta will look like



- Will have many local minimum, non-convex, on running GD, not guaranateed that it will converge to global minimum

Cost Function in Logistic Regression

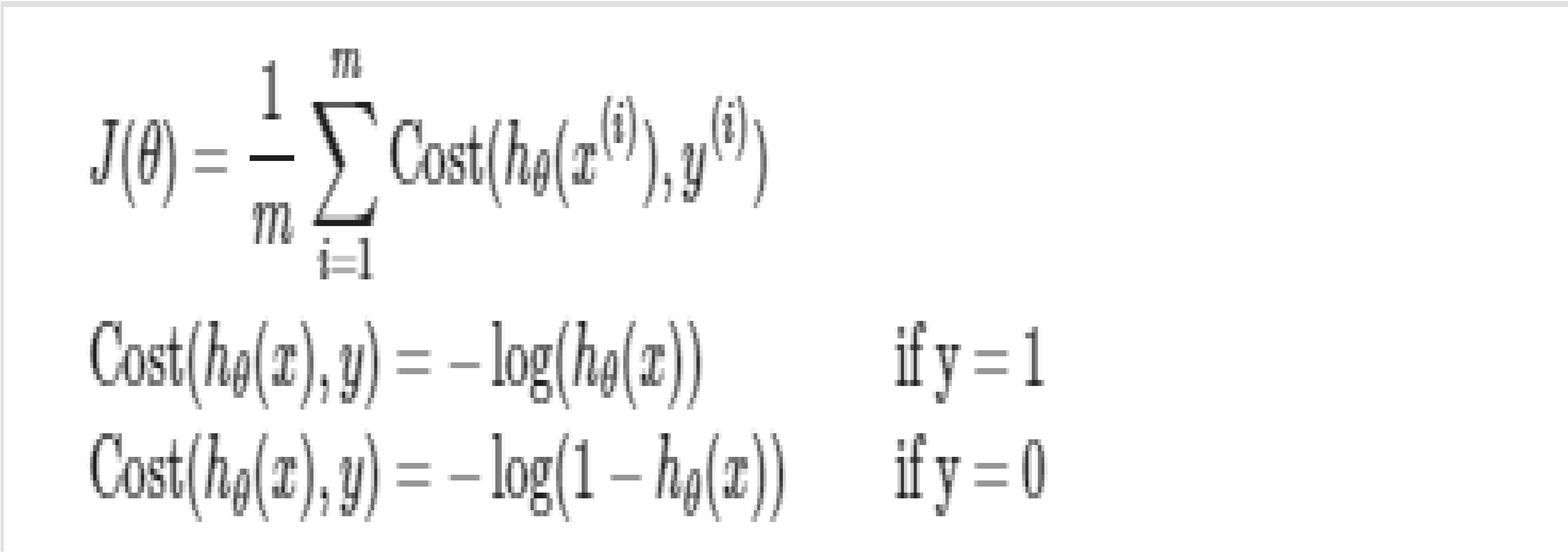


In the cost function J of Theta, which is convex, on running GD, it is guaranteed that GD will converge to global function.

Instead of 1/2m, taken slightly differently 1/m outside. We write alternately.

If we use this particular cost function, it would be a non-convex function of the parameters data.

Cost Function for Logistic Regression

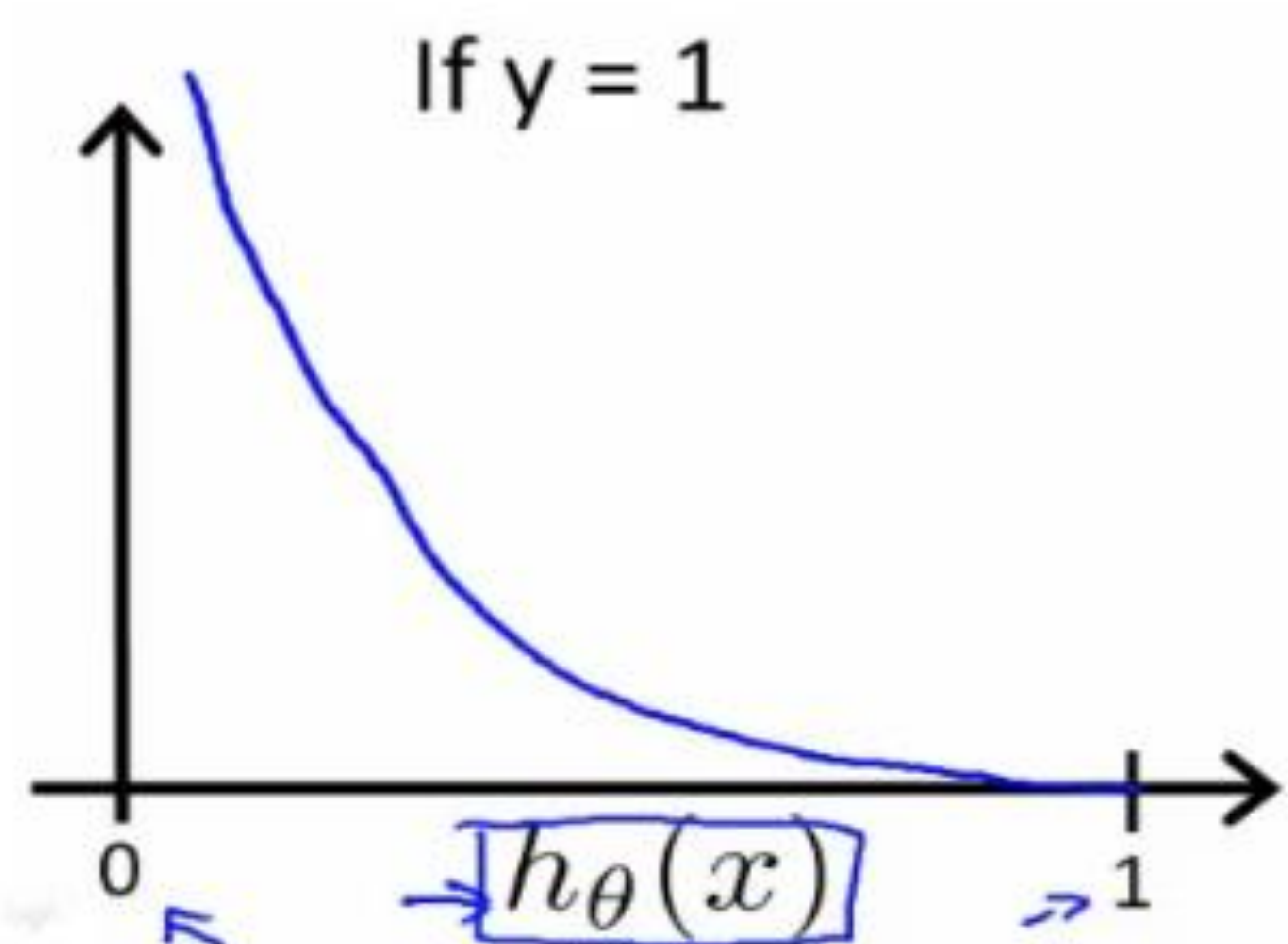


$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

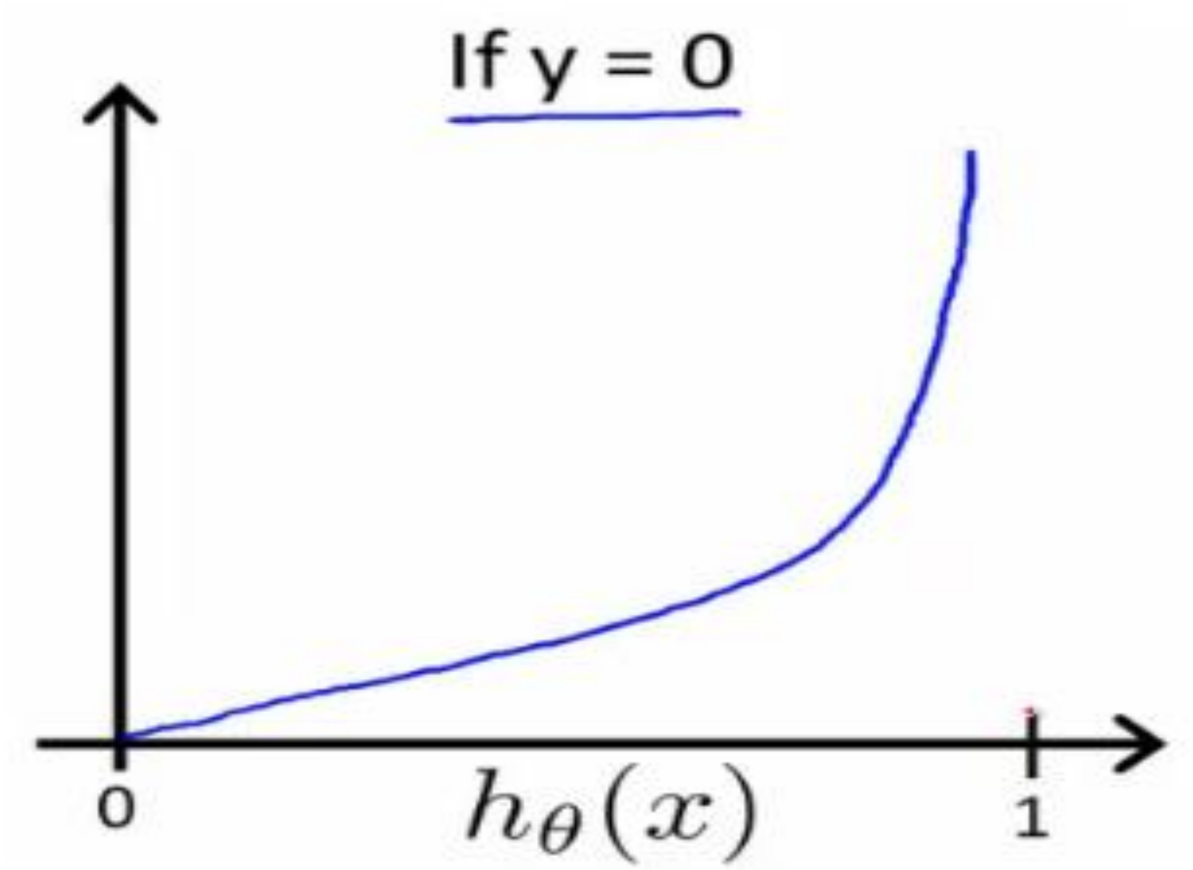
$$\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \qquad \text{if y} = 1$$

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

When y = 1, we get the following plot for $J(\theta)$ vs $h_\theta(x)$:



Similarly, when y = 0, we get the following plot for $J(\theta)$ vs $h_\theta(x)$:



$$
\begin{aligned}
&\mathrm{Cost}(h_\theta(x), y) = 0 \text{ if } h_\theta(x) = y \\
&\mathrm{Cost}(h_\theta(x), y) \rightarrow \infty \text{ if } y = 0 \;\mathrm{and}\; h_\theta(x) \rightarrow 1 \\
&\mathrm{Cost}(h_\theta(x), y) \rightarrow \infty \text{ if } y = 1 \;\mathrm{and}\; h_\theta(x) \rightarrow 0
\end{aligned}
$$

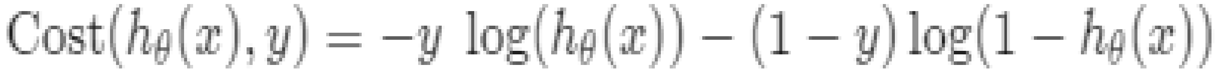
If our correct answer 'y' is 0, then the cost function will be 0 if our hypothesis function also outputs 0. If our hypothesis approaches 1, then the cost function will approach infinity.

If our correct answer 'y' is 1, then the cost function will be 0 if our hypothesis function outputs 1. If our hypothesis approaches 0, then the cost function will approach infinity.

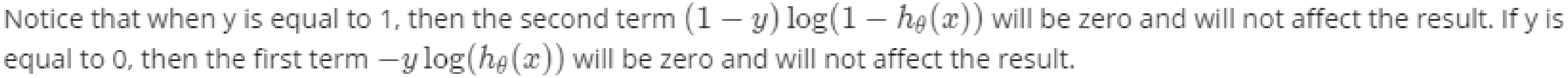
Note that writing the cost function in this way guarantees that J(θ) is convex for logistic regression.

Simplified Cost Function and Gradient Descent

- We can compress our cost function's two conditional cases into one case:

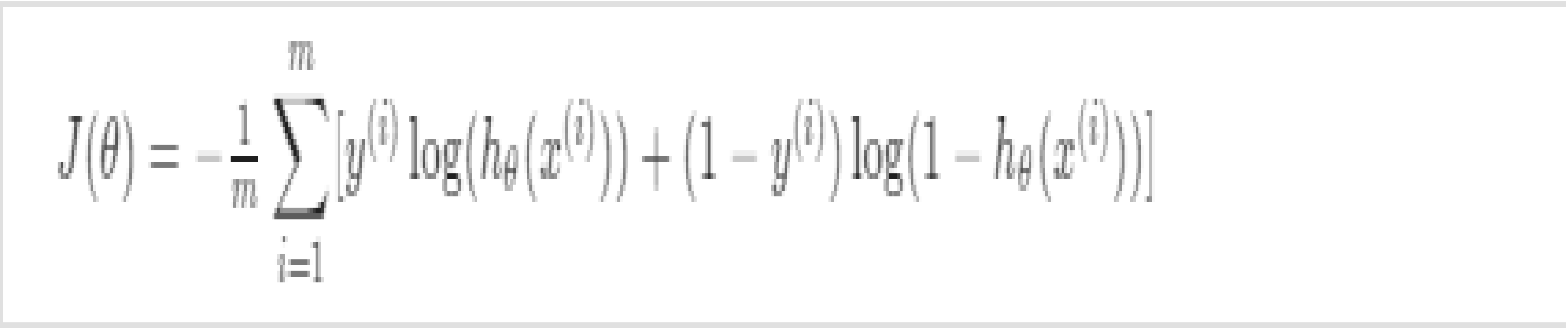


$$\mathrm{Cost}(h_\theta(x), y) = -y\ \log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$



Notice that when y is equal to 1, then the second term $(1-y)\log(1-h_\theta(x))$ will be zero and will not affect the result. If y is equal to 0, then the first term $-y\log(h_\theta(x))$ will be zero and will not affect the result.

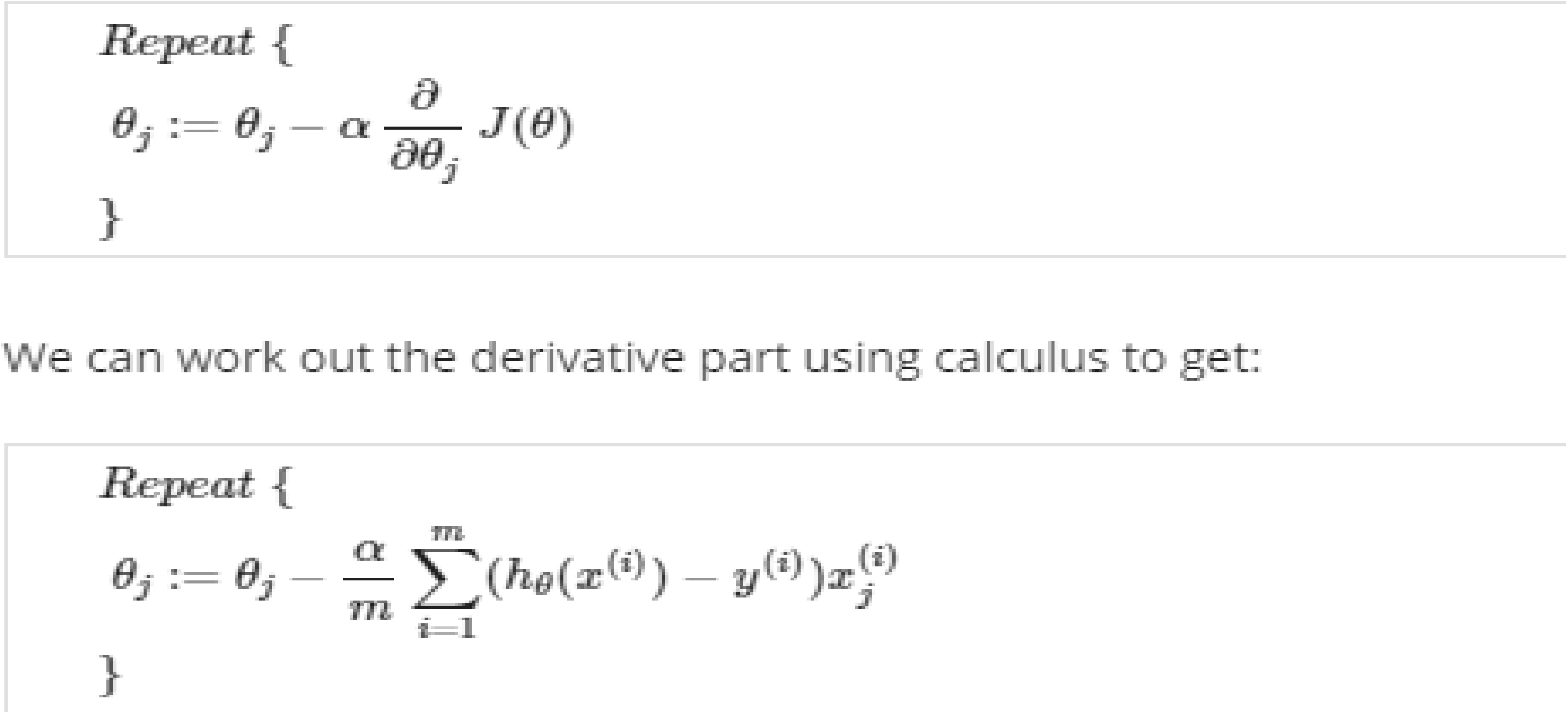
- We can fully write out our entire cost function as follows:



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

Gradient Descent

- Remember that the general form of gradient descent is:



$$\begin{aligned}&Repeat\ \{\\ &\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)\\ &\}\end{aligned}$$

We can work out the derivative part using calculus to get:

$$\begin{aligned}&Repeat\ \{\\ &\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}\\ &\}\end{aligned}$$

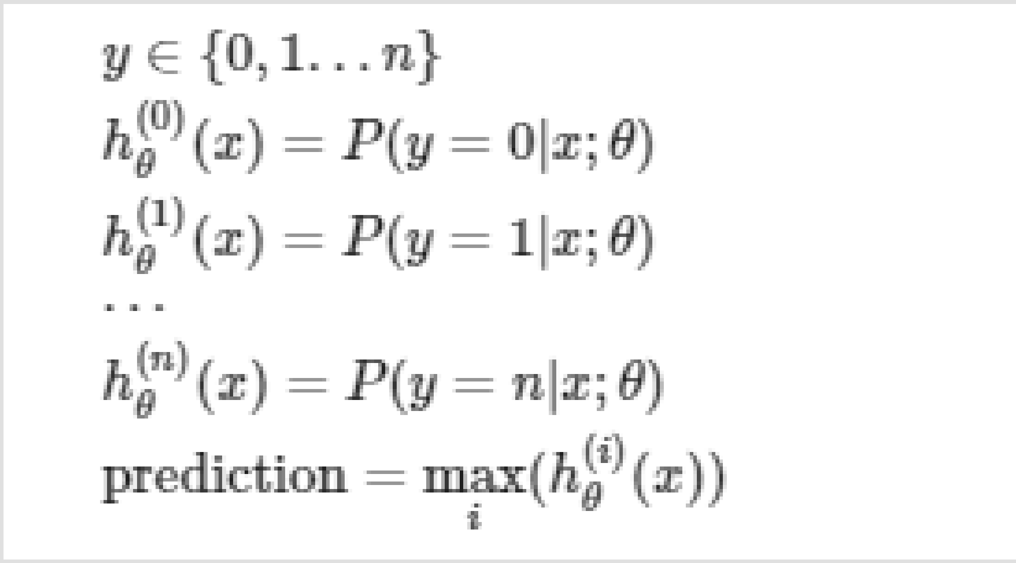
Gradient Descent

- Notice that this algorithm is identical to the one we used in linear regression. We still have to simultaneously update all values in theta.
- A vectorized implementation is:

$$\theta := \theta - \frac{\alpha}{m} X^T (g(X\theta) - \vec{y})$$

Multi Class Classification : One Vs All

- Now we will approach the classification of data when we have more than two categories. Instead of y = {0,1} we will expand our definition so that y = {0,1...n}.
- Since y = {0,1...n}, we divide our problem into n+1 (+1 because the index starts at 0) binary classification problems; in each one, we predict the probability that 'y' is a member of one of our classes.



$$
\begin{aligned}
&y \in \{0, 1 \ldots n\} \\
&h_\theta^{(0)}(x) = P(y = 0 | x; \theta) \\
&h_\theta^{(1)}(x) = P(y = 1 | x; \theta) \\
&\cdots \\
&h_\theta^{(n)}(x) = P(y = n | x; \theta) \\
&\mathrm{prediction} = \max_i (h_\theta^{(i)}(x))
\end{aligned}
$$

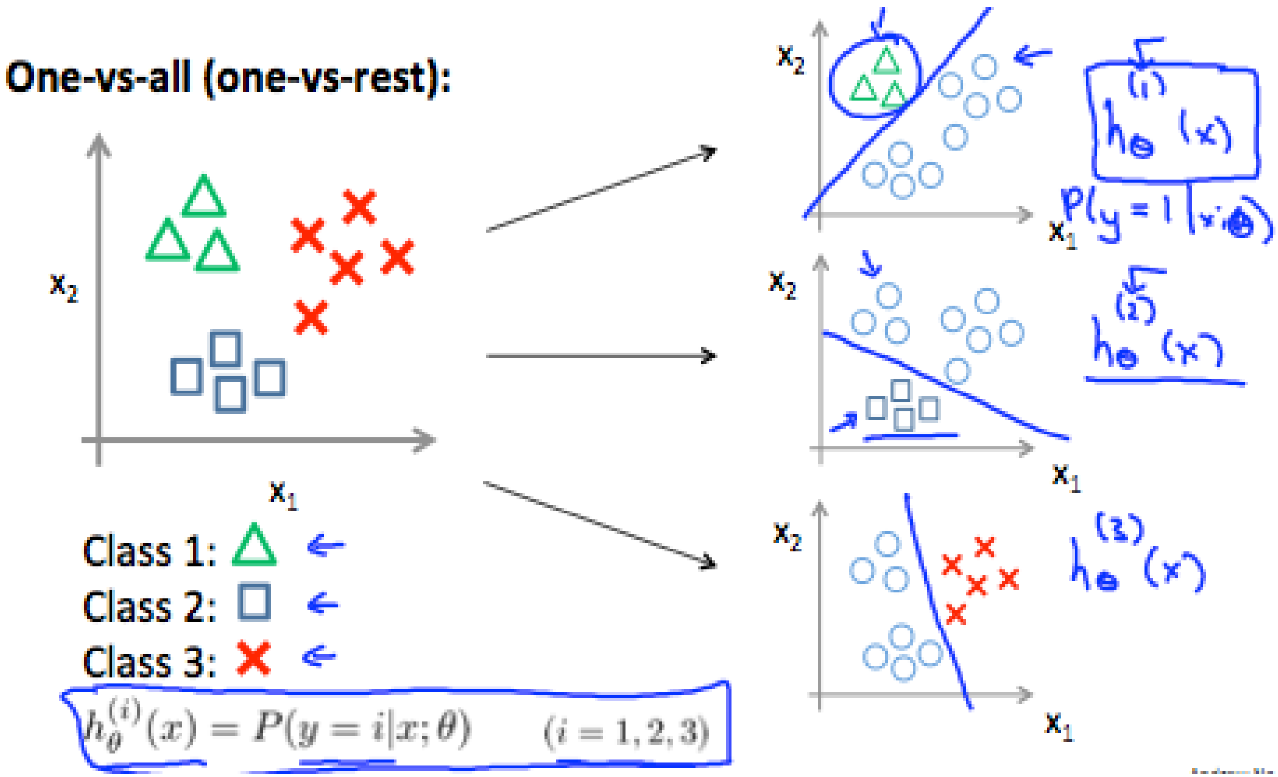
Multi Class Classification

- Example:
  - Want to tag my emails into four groups

Work	y=1
Friends	y=2
Family	y=3
Hobby	y=4

- We are basically choosing one class and then lumping all the others into a single second class. We do this repeatedly, applying binary logistic regression to each case, and then use the hypothesis that returned the highest value as our prediction.

he following image shows how one could classify 3 classes:



Conclusion

- Train a logistic regression classifier hTheta(x) for each class to predict the probability.
- To make a prediction on a new x, pick the class that maximizes htheta(x).