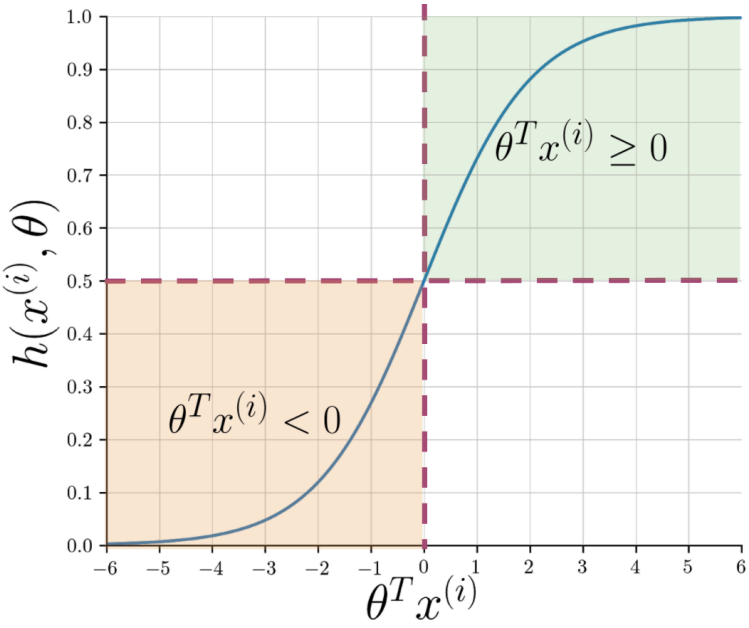
Logistic Regression Overview

Logistic regression makes use of the sigmoid function which outputs a probability between 0 and 1. The sigmoid function with some weight parameter θ and some input $x^{(i)}$ is defined as follows.

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



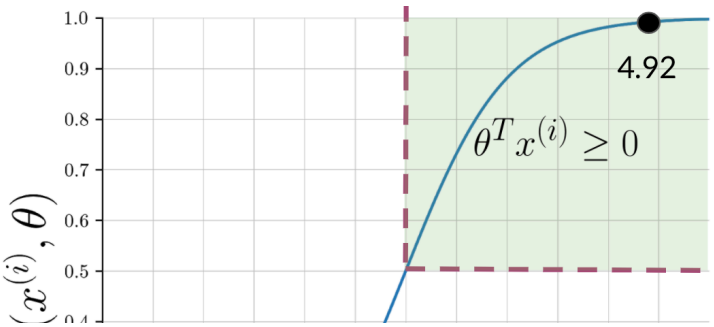
Note that as $\theta^T x^{(i)}$ gets closer and closer to $-\infty$ the denominator of the sigmoid function gets larger and larger and as a result, the sigmoid gets closer to 0. On the other hand, as $\theta^T x^{(i)}$ gets closer and closer to ∞ the denominator of the sigmoid function gets closer to 1 and as a result the sigmoid also gets closer to 1.

Now given a tweet, you can transform it into a vector and run it through your sigmoid function to get a prediction as follows:

@YMourri and @AndrewYNg are tuning a GREAT AI model

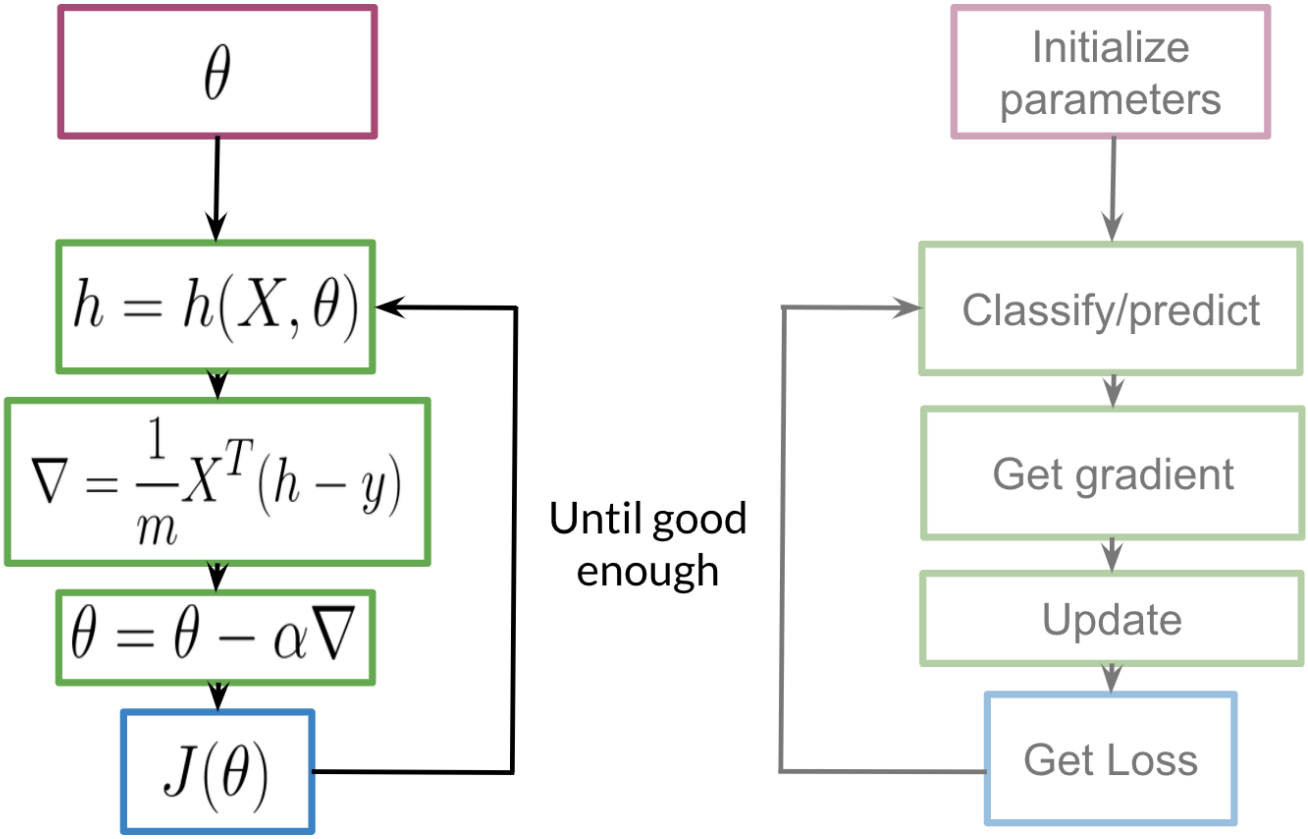
↓

[tun, ai, great, model]



Logistic Regression: Training

To train your logistic regression function, you will do the following:



You initialize your parameter θ, that you can use in your sigmoid, you then compute the gradient that you will use to update θ, and then calculate the cost. You keep doing so until good enough.

Note: If you do not know what a gradient is, don't worry about it. I will show you what it is at then end of this week in an optional reading. In a nutshell, the gradient allows you to learn what θ is so that you can predict your tweet sentiment accurately.

Usually you keep training until the cost converges. If you were to plot the number of iterations versus the cost, you should see something like this:

Logistic Regression: Testing

To test your model, you would run a subset of your data, known as the validation set, on your model to get predictions. The predictions are the outputs of the sigmoid function. If the output is $\geq = 0.5$, you would assign it to a positive class. Otherwise, you would assign it to a negative class.

- X_{val} Y_{val} θ

$$h(X_{val}, \theta)$$

$$pred = h(X_{val}, \theta) \geq 0.5$$

$$\begin{bmatrix} 0.3 \\ 0.8 \\ 0.5 \\ \vdots \\ h_m \end{bmatrix} \geq 0.5 = \begin{bmatrix} 0.3 \geq 0.5 \\ 0.8 \geq 0.5 \\ 0.5 \geq 0.5 \\ \vdots \\ pred_m \geq 0.5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ \vdots \\ pred_m \end{bmatrix}$$

In the video, I briefly mentioned X validation. In reality, given your X data you would usually split it into three components. $X_{train}, X_{val}, X_{test}$. The distribution usually varies depending on the size of your data set. However, an $80, 10, 10$ split usually works fine.

To compute accuracy, you solve the following equation:

$$\text{Accuracy} \longrightarrow \sum_{i=1}^{m} \frac{(pred^{(i)} == y_{val}^{(i)})}{m}$$

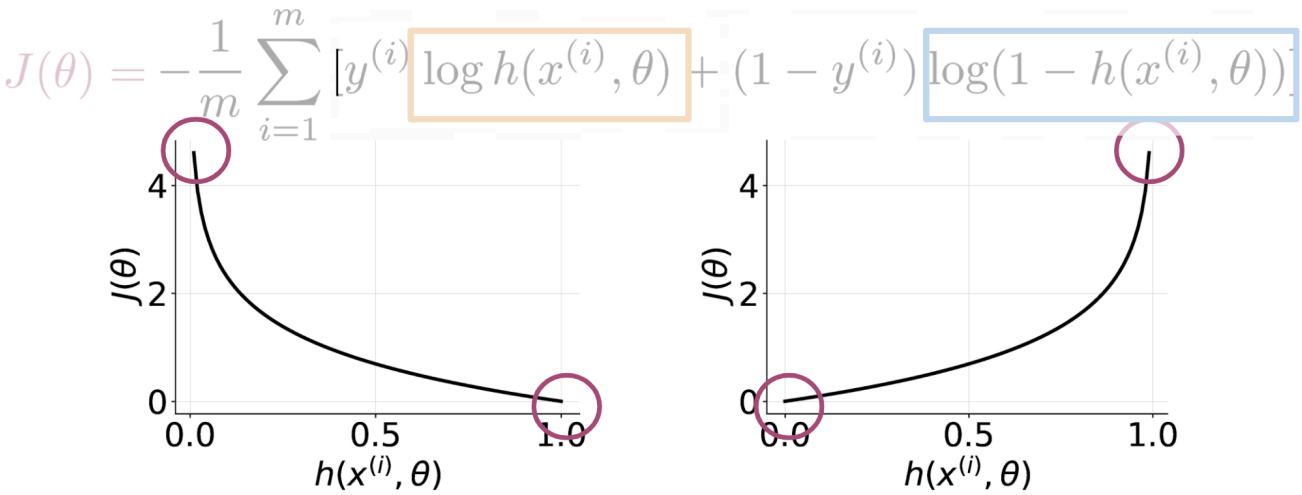
In other words, you go over all your training examples, m of them, and then for every prediction, if it was right you add a one. You then divide by m.

Optional Logistic Regression: Cost Function

This is an advanced optional reading where we delve into the details.. If you do not get the math, do not worry about it - you will be just fine by moving onto the next component. In this part, I will tell you about the intuition behind why the cost function is designed the way it is. I will then show you how to take the derivative of the logistic regression cost function to get the gradients.

The logistic regression cost function is defined as

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



As you can see in the picture above, if $y = 1$ and you predict something close to 0, you get a cost close to ∞. The same applies for then $y = 0$ and you predict something close to 1. On the other hand if you get a prediction equal to the label, you get a cost of 0. In either, case you are trying to minimize $J(\theta)$.

Math Derivation

To show you why the cost function is designed that way, let us take a step back and write up a function that compresses the two cases into one case.

$$P(y|x^{(i)},\theta) = h(x^{(i)},\theta)^{y^{(i)}}(1-h(x^{(i)},\theta)^{(1-y^{(i)})}$$