



Logistic Regression

**Jeremy Irvin and
Daniel Spokoyny**

**Created from Andrew Ng's
Stanford CS229 Notes**



Classification

- Recall that we we're trying to predict continuous values using regression.
- If we're trying to predict the values y which only take on a small amount of discrete values, it is called classification.
- For now we will focus on binary classification, ie, predicting either a **0** or a **1**.
- **0** will be called the negative class, and **1** will be called the positive class.

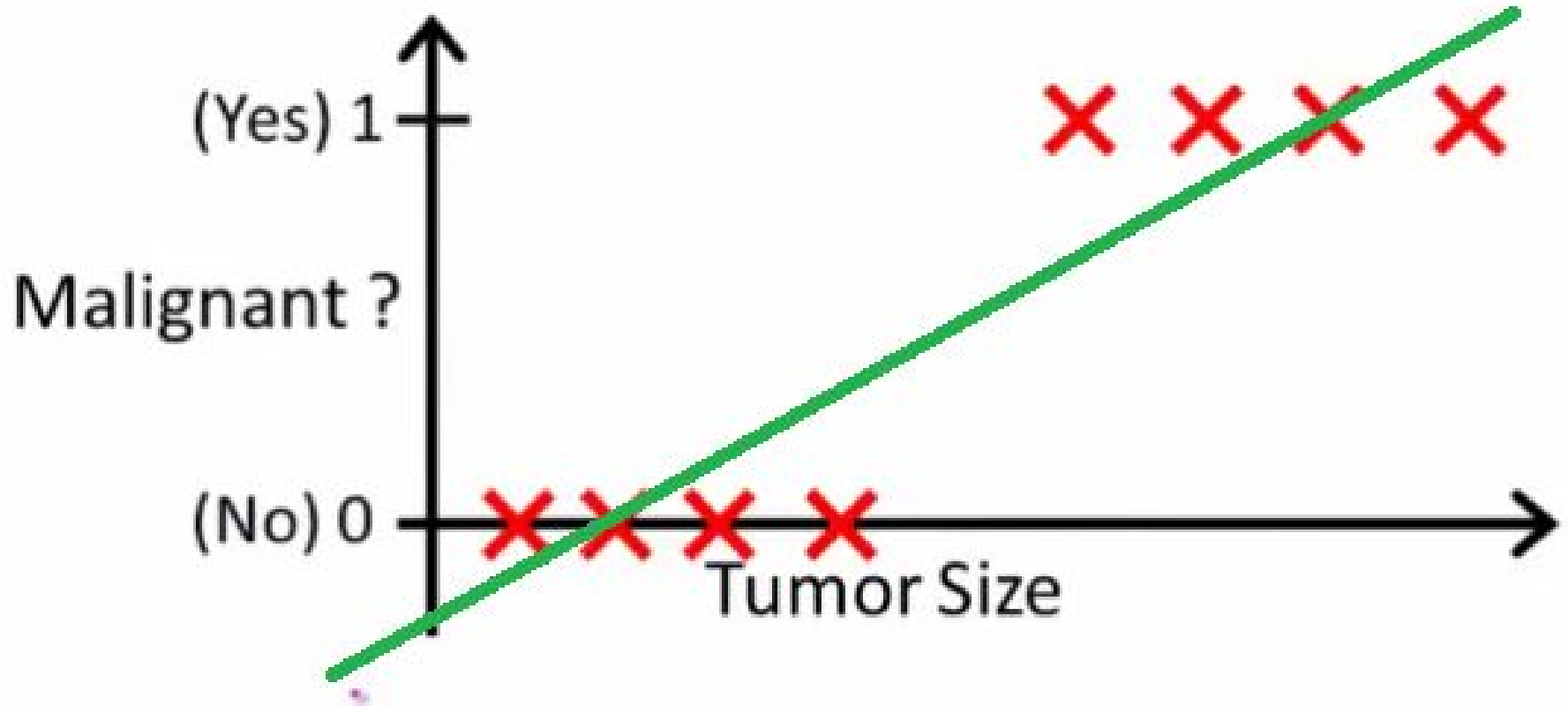
Logistic Regression

- We could attempt to tackle this classification problem with the linear regression algorithm.
- However, it is easy to construct an example where this performs poorly, as we will see on the next slide.
- For initial intuition, it does not make sense for the hypothesis function to output values greater than 1 or less than 0 when $y \in \{0, 1\}$.

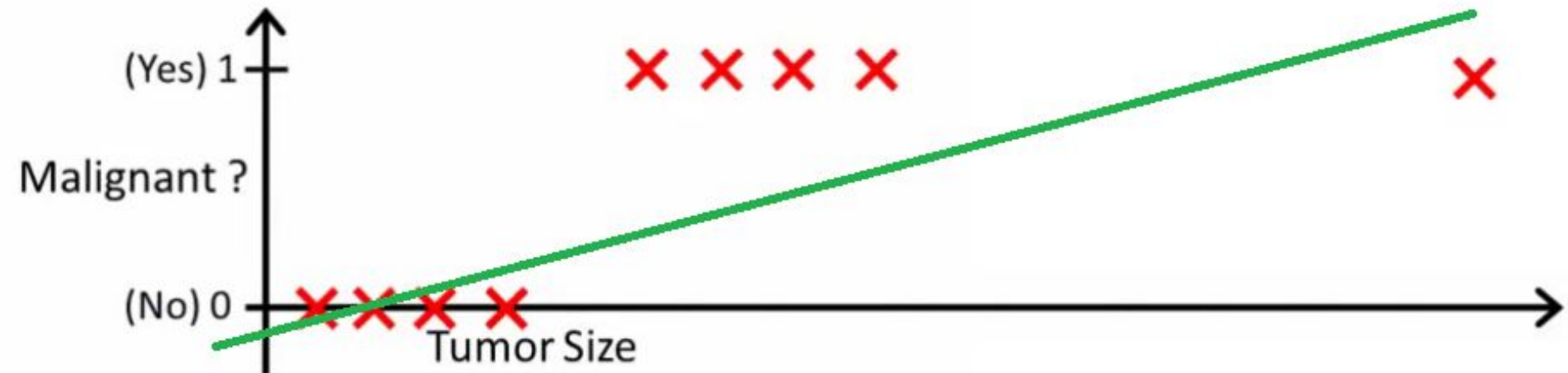
Linear Regression Binary Classification Example

- Suppose we are trying to predict whether a tumor is malignant based on its size.
- Malignant tumors are labeled **1**, and benign tumors are labeled **0**.
- To make predictions using linear regression, we could say if $h(x)$ outputs a value larger than 0.5, predict malignant, otherwise predict benign.

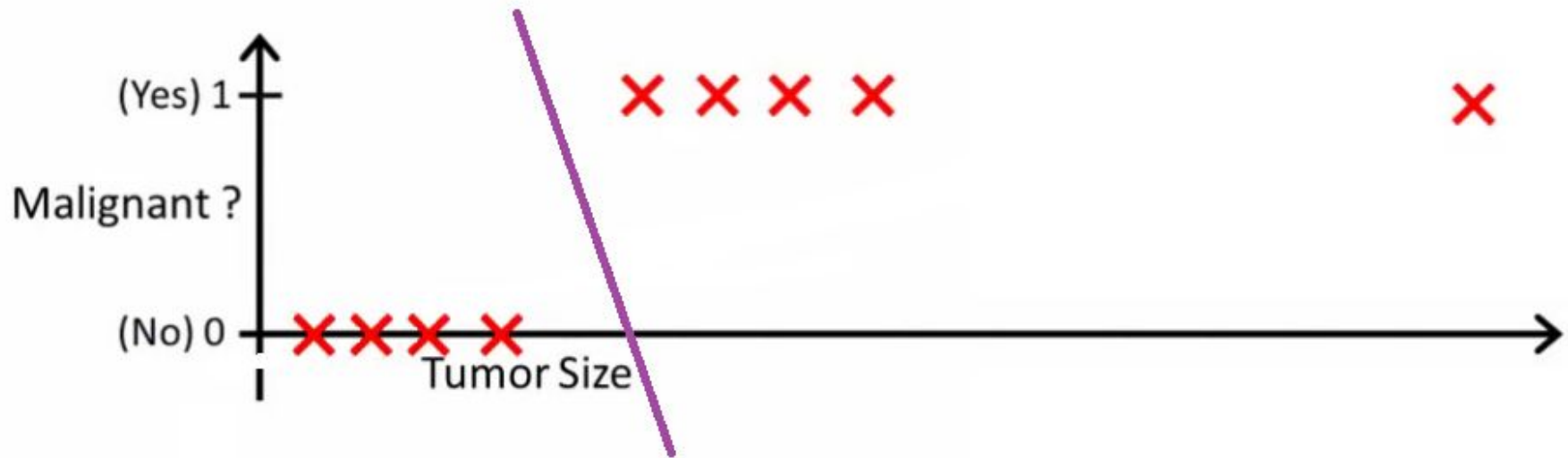
Example Cont.



Example Cont.



Example Cont.



- Both linear and logistic predict straight lines.
- Linear interpolates the output and predicts the value for x we haven't seen.
- Logistic says all points sitting to the right of the classifier line belong to one class, and the left belong to the other.
 - In this case, $h(x)$ represents the probability that x belongs to the positive class.

Sigmoid Function

- We will change the form of $h_{\theta}(x)$:

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

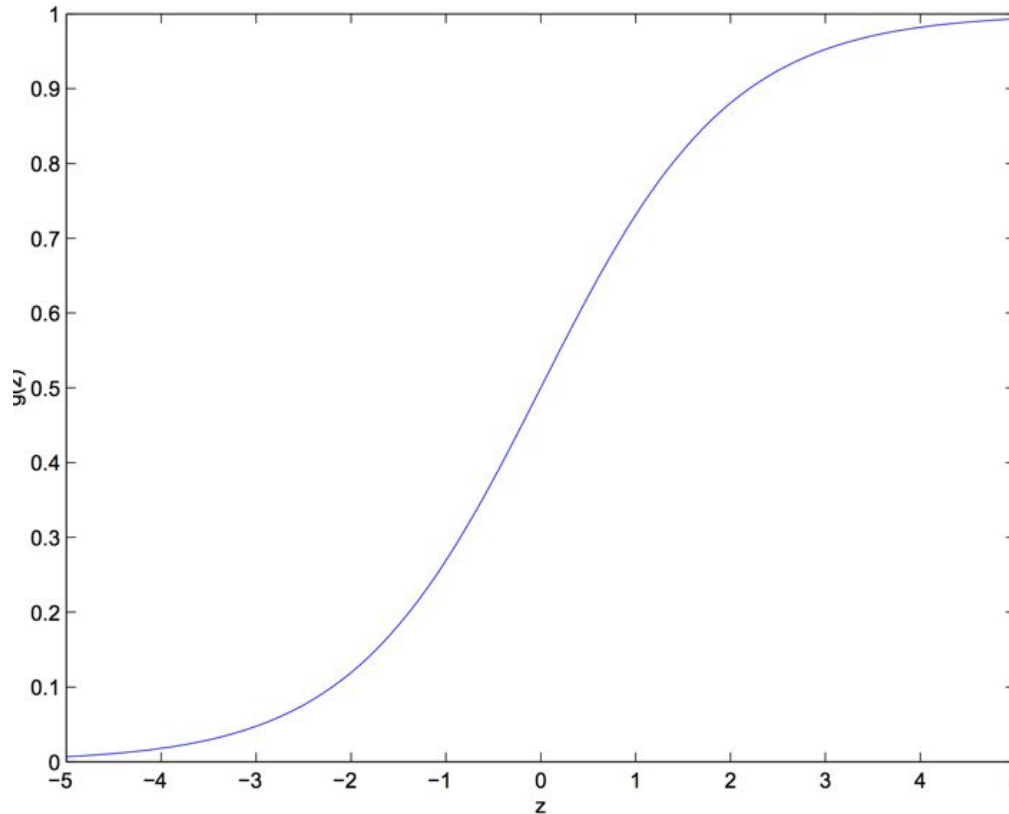
where

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

is called the logistic or sigmoid function.

Sigmoid Function

- Here is a plot of $g(z)$: (visualize scaling)



$$\lim_{z \rightarrow \infty} g(z) = 1, \quad \lim_{z \rightarrow -\infty} g(z) = 0$$

Sigmoid Function

- So we kind of arbitrarily chose this g due to the fact that it increases from 0 to 1.
- In fact, there are many reasons why we use the logistic function, the first being its *smoothness* and easily computable derivative:

$$\begin{aligned} g'(z) &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\ &= \frac{1}{(1 + e^{-z})^2} (e^{-z}) \\ &= \frac{1}{(1 + e^{-z})} \cdot \left(1 - \frac{1}{(1 + e^{-z})} \right) \\ &= g(z)(1 - g(z)). \end{aligned}$$

Fitting the Logistic Regression Model

- Similar to linear regression, we want to find θ to *best* fit our data for future predictions.
- Let's endow the classification problem with some probabilistic assumptions (as we did with linear regression), and then fit the parameters through maximum likelihood!

Fitting the Logistic Regression Model

- Assume that

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

$$P(y = 0 \mid x; \theta) = 1 - h_{\theta}(x)$$

- Or more compactly, that

$$p(y|x; \theta) = (h_{\theta}(x))^y (1 - h_{\theta}(x))^{1-y}$$

Fitting the Logistic Regression Model

- Assuming the m training examples were generated independently, the likelihood of the parameters is

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X; \theta) \\ &= \prod_{i=1}^m p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^m (h_{\theta}(x^{(i)}))^{y^{(i)}} (1 - h_{\theta}(x^{(i)}))^{1-y^{(i)}} \end{aligned}$$

Fitting the Logistic Regression Model

- Maximizing the log likelihood will again be easier:

$$\begin{aligned}\ell(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^m y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))\end{aligned}$$

- We will maximize this function using gradient descent, thus we need to find its gradient. Let's do it component-wise on a single training example (x, y) :

Fitting the Logistic Regression Model

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

$$\ell(\theta) = y \log h_{\theta}(x) + (1 - y) \log(1 - h_{\theta}(x))$$

Therefore:

$$\begin{aligned} \frac{\partial}{\partial \theta_j} \ell(\theta) &= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) \frac{\partial}{\partial \theta_j} g(\theta^T x) \\ &= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) g(\theta^T x)(1 - g(\theta^T x)) \frac{\partial}{\partial \theta_j} \theta^T x \\ &= (y(1 - g(\theta^T x)) - (1 - y)g(\theta^T x)) x_j \\ &= (y - h_{\theta}(x)) x_j \end{aligned}$$

since $g'(z) = g(z)(1 - g(z))$.

Fitting the Logistic Regression Model

- This gives us the stochastic gradient *ascent* rule:

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_{\theta}(x^{(i)}))x_j^{(i)}$$

- This looks identical to the LMS update rule!
- But this is a completely different algorithm.
- Is this a coincidence?
- No!! It is because they are both types of Generalized Linear Models (GLM's).