Maximum Likelihood Estimation & Logistic Regression

Slides adapted from David Sontag and Andrew Ng.

Maximum Likelihood Estimation (MLE)

Framework:

- Observed data *D* (observations)
- Hypothesize data has a specific probability distribution parameterized by unknown parameter values $\boldsymbol{\theta}$: i.e., distribution $P_{\boldsymbol{\theta}}(D)$ is known
- Goal: estimate (learn) the parameter values $\boldsymbol{\theta}$.
- MLE: Choose parameter values $\boldsymbol{\theta}$ that maximize $P_{\boldsymbol{\theta}}(D)$

Thumbtack example

- P_θ(Heads) = θ, P_θ(Tails) = 1-θ. What is θ?

- Flips are *i.i.d.*:

$$D = \{x_i \mid i = 1, \ldots, m\}, \quad P_\theta(D) = \Pi_i P_\theta(x_i)$$

x_i = H or T

  - Independent and Identically distributed
- Observe α_H Heads and α_T Tails: $\alpha_H + \alpha_T = n$
- Probability of D occurring (given θ) is:

$$P_\theta(D) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

Called the "likelihood" of the data under the model. It is our "model".

Maximum Likelihood Estimation

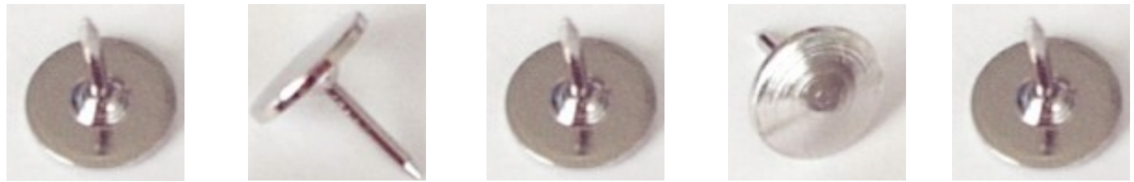
- **Data:** Observed set D: sequence consisting of α_H Heads and α_T Tails.
- **Model:** $$P_{\boldsymbol{\theta}}(D) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$
- **Learning:** find θ that maximizes the probability of the observation D, i.e., find:

$$\widehat{\theta} = \arg\max_{\theta} P_{\boldsymbol{\theta}}(D)$$

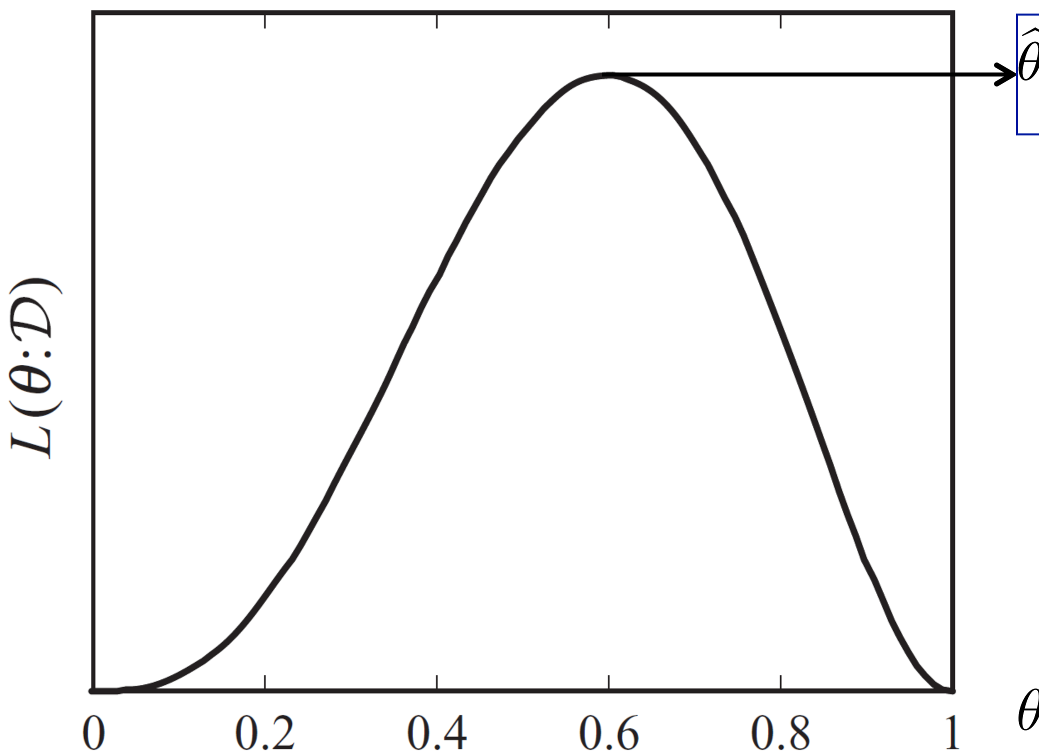
- **Taking derivative and setting to zero, get:**

$$\boxed{\widehat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}}$$

Data



$$L(\boldsymbol{\theta};D) = \ln P_{\boldsymbol{\theta}}(D)$$



$$\boxed{\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}}$$

Logistic Regression

- Popular type of supervised machine learning for classification
- Classification, not regression!
- Gives probabilities for classification, e.g., email is spam with probability 0.86
- Can be viewed as a MLE estimator
- Often used in neural networks

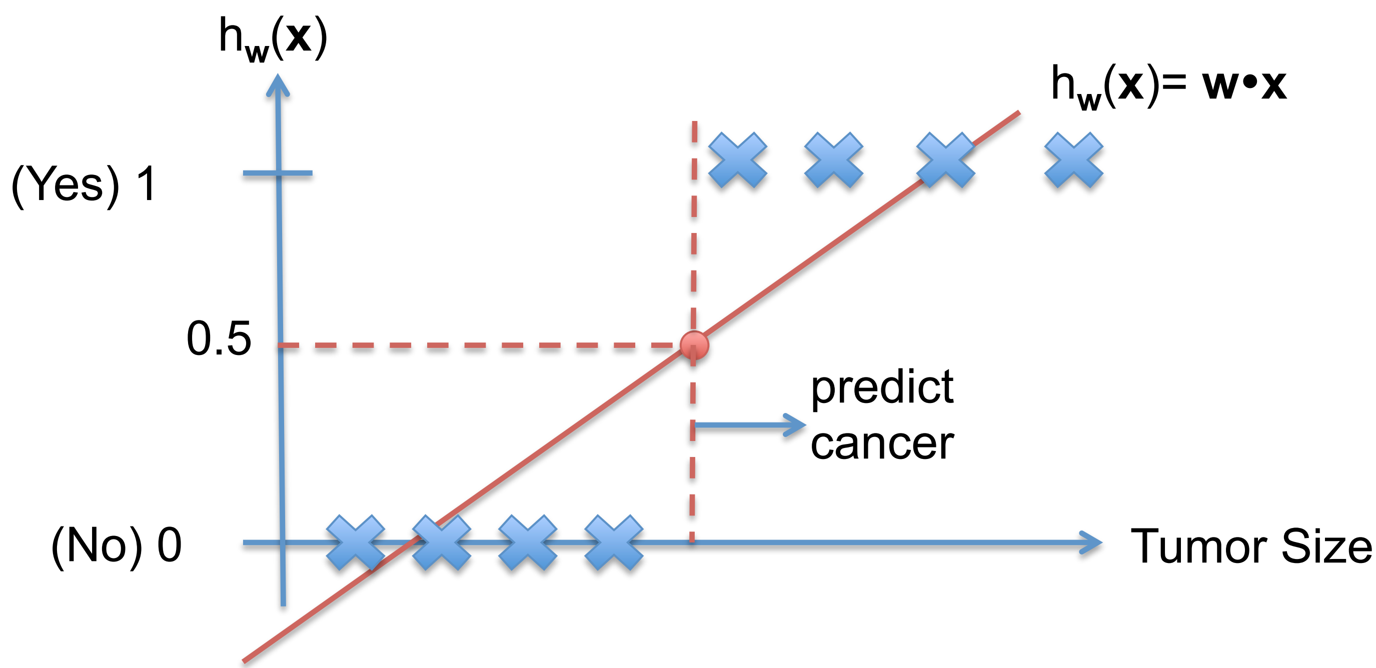
Classification

- Email: Spam/ Not Spam?
- Online Transactions: Fraudulent (Yes/ No?)
- Tumor: Malignant/ Benign?

$$y \in \{0,1\}$$

0: "Negative Class" (e.g., benign tumor)
1: "Positive Class" (e.g., malignant tumor)

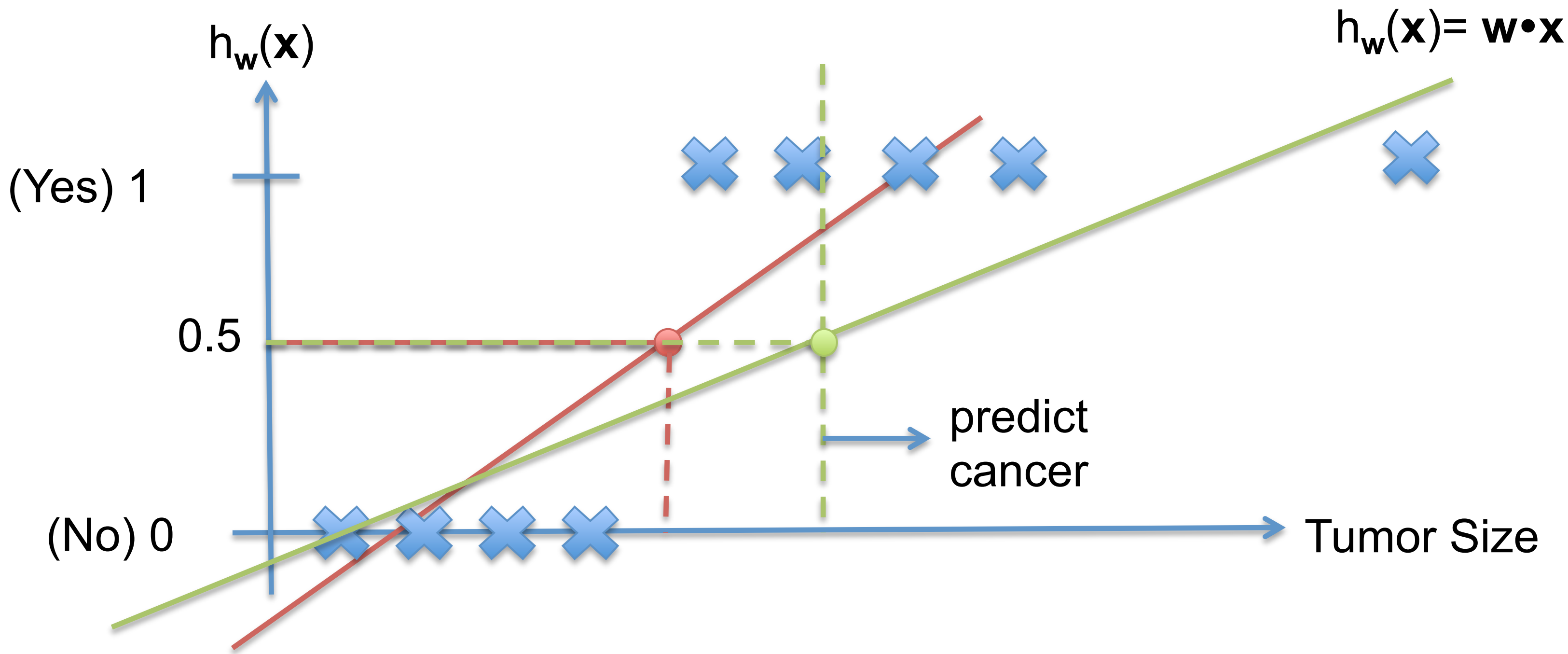
Let's try to predict with ordinary regression



Natural threshold classifier:

- If $h_w(\mathbf{x}) \geq 0.5$, predict "y=1"
- If $h_w(\mathbf{x}) < 0.5$, predict "y=0"

Additional data point



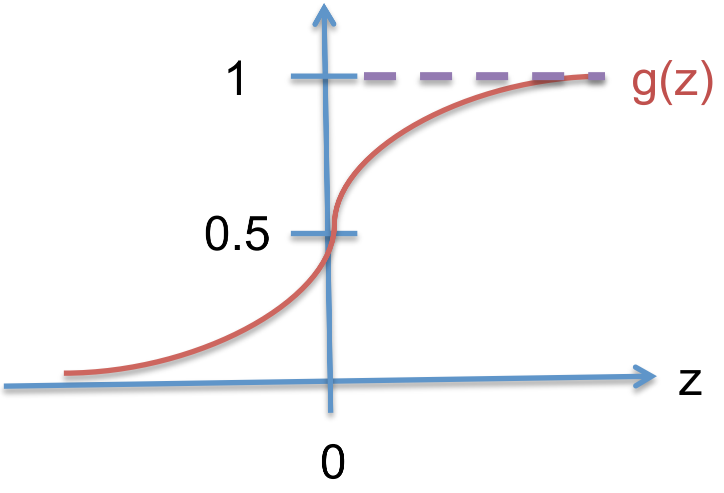
Linear regression with natural 0.5 threshold does not look good here.

Graphically, what kind of function would be a good fit?

Sigmoid function

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

Sigmoid function = Logistic function

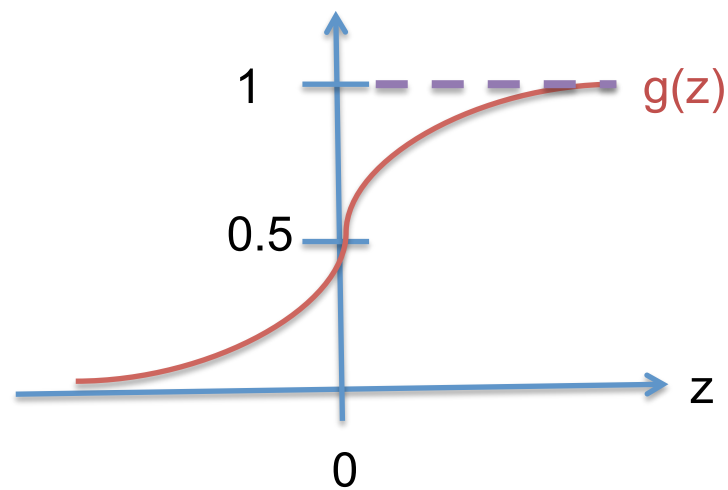


Logistic regression

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

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = g(\boldsymbol{w} \cdot \boldsymbol{x})$$

- Note that $0 \leq h_{\mathbf{w}}(\mathbf{x}) \leq 1$
- Can be interpreted as a probability.
- Can choose **w** to optimize the fit to data (later).
- How might we fit the tumor data with logistic regression?



- Suppose we have learned **w**. Observe **x** for new patient and want to predict if patient has cancer
- $h_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w} \cdot \mathbf{x})$ estimated probability that patient has cancer
- Example:

$$\boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ tumorSize \end{bmatrix}$$

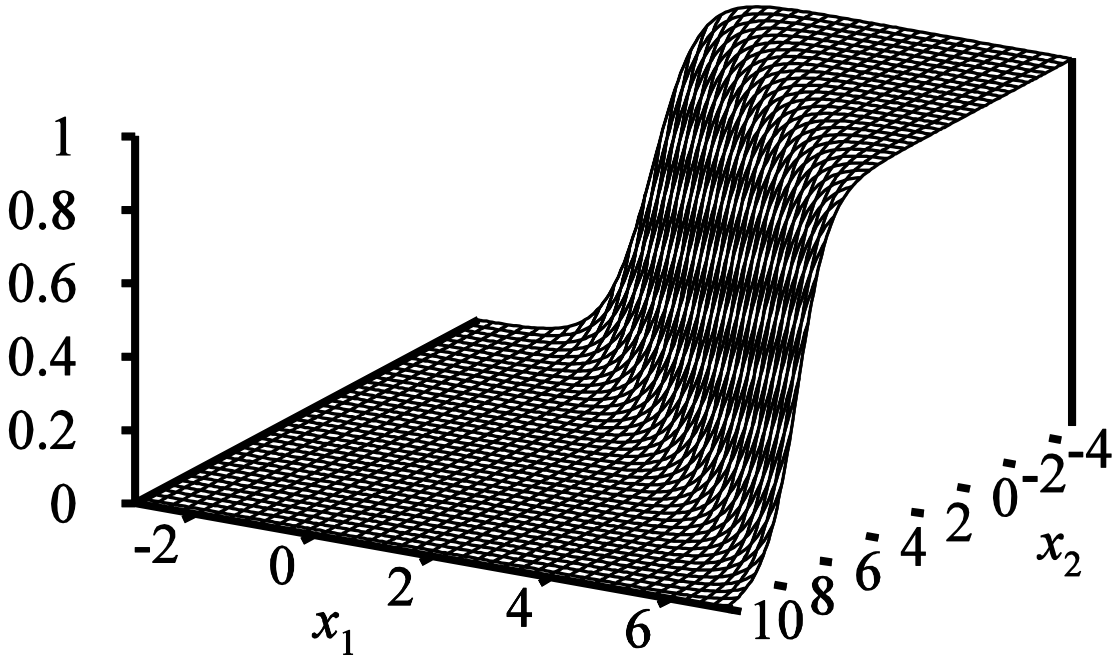
$$h_{\boldsymbol{w}}(\boldsymbol{x}) = 0.7$$

Tell patient that 70% chance of tumor being malignant

$$P_{\mathbf{w}}(y=1|\mathbf{x}) = h_{\mathbf{w}}(\mathbf{x})$$

“Probability that y=1, given **x**, parameterized by **w**”

Logistic Function in n Dimensions



Summary

- Use labeled data to learn **w**
- Observe new **x**
- Given **x**, we say y=1 with estimated probability $h_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w} \bullet \mathbf{x})$, where

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

- Alternatively way of saying it: $P_{\mathbf{w}}(y=1|\mathbf{x}) = h_{\mathbf{w}}(\mathbf{x})$.
- But how do we learn **w** ?

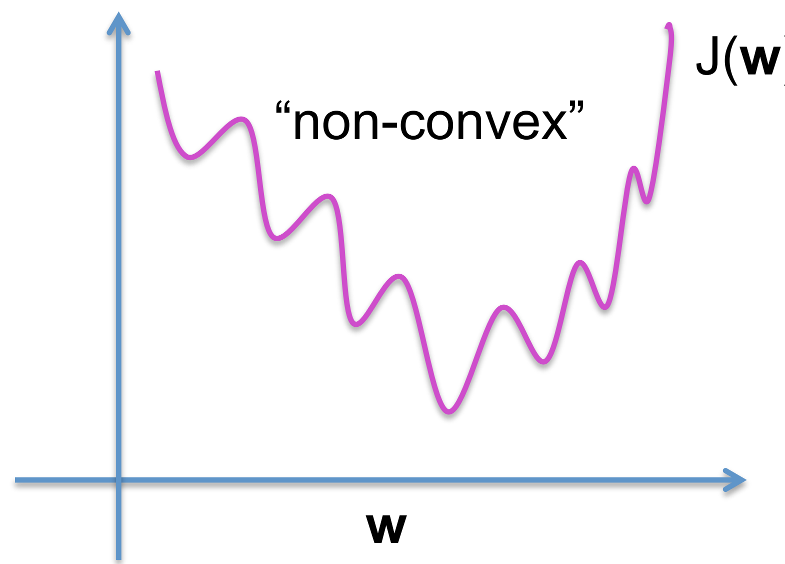
Learning the Parameters **w**

- Training set: $\{ (\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}),\ldots, (\mathbf{x}^{(m)},y^{(m)})\}$
- How about choosing w to minimize MSE (as usual)?

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

$$Cost(h_{\boldsymbol{w}}(\boldsymbol{x}), y) = \frac{1}{2}(h_{\boldsymbol{w}}(\boldsymbol{x}) - \mathrm{y})^2$$

$$h_{\boldsymbol{w}}(x) = \frac{1}{1+e^{-\boldsymbol{w}\cdot\boldsymbol{x}}}$$



Learning the Parameters **w**

- Instead try using MLE. Find **w** that maximizes the probability of the observation. Maximize:

  $$P_{\mathbf{w}}(y^{(1)},y^{(2)},\ldots,y^{(m)} \mid \mathbf{x}^{(1)}, \mathbf{x}^{(2)},\ldots, \mathbf{x}^{(m)})$$

- Assume each observed data point is conditionally independent:
- $P_{\mathbf{w}}(y^{(1)},y^{(2)},\ldots,y^{(m)} \mid \mathbf{x}^{(1)}, \mathbf{x}^{(2)},\ldots, \mathbf{x}^{(m)}) =$
  $P_{\mathbf{w}}(y^{(1)} \mid \mathbf{x}^{(1)}) \times P_{\mathbf{w}}(y^{(2)} \mid \mathbf{x}^{(2)}) \times \ldots \times P_{\mathbf{w}}(y^{(m)} \mid \mathbf{x}^{(m)})$
- Assume logistic function probabilities:

  $P_{\mathbf{w}}(y^{(i)}=1 \mid \mathbf{x}^{(i)}) = h_{\mathbf{w}}(\mathbf{x}^{(i)})$
  $P_{\mathbf{w}}(y^{(i)}=0 \mid \mathbf{x}^{(i)}) = 1- h_{\mathbf{w}}(\mathbf{x}^{(i)})$

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}\cdot\boldsymbol{x}}}$$

Choose **w** to maximize:

- $P_{\mathbf{w}}(y^{(1)},y^{(2)},\ldots,y^{(m)} \mid \mathbf{x}^{(1)}, \mathbf{x}^{(2)},\ldots, \mathbf{x}^{(m)}) =$
  $P_{\mathbf{w}}(y^{(1)} \mid \mathbf{x}^{(1)}) \times P_{\mathbf{w}}(y^{(2)} \mid \mathbf{x}^{(2)}) \times \ldots \times P_{\mathbf{w}}(y^{(m)} \mid \mathbf{x}^{(m)})$
- Suppose f(z) is a monotonically increasing function of z. Suppose t(**w**) is some function of **w**. Then if **w*** is optimal for f(t(**w**)), it is also optimal for t(**w**).
- So we can instead maximize
  $\log (P_{\mathbf{w}}(y^{(1)},y^{(2)},\ldots,y^{(m)} \mid \mathbf{x}^{(1)}, \mathbf{x}^{(2)},\ldots, \mathbf{x}^{(m)})) =$
  $\log [P_{\mathbf{w}}(y^{(1)} \mid \mathbf{x}^{(1)}) \times P_{\mathbf{w}}(y^{(2)} \mid \mathbf{x}^{(2)}) \times \ldots \times P_{\mathbf{w}}(y^{(m)} \mid \mathbf{x}^{(m)})] =$
  $\log P_{\mathbf{w}}(y^{(1)} \mid \mathbf{x}^{(1)}) + \log P_{\mathbf{w}}(y^{(2)} \mid \mathbf{x}^{(2)}) + \ldots + \log P_{\mathbf{w}}(y^{(m)} \mid \mathbf{x}^{(m)})$
- $\log P_{\mathbf{w}}(y^{(i)} = 1 \mid \mathbf{x}^{(i)}) = \log h_{\mathbf{w}}(\mathbf{x}^{(i)})$
  $\log P_{\mathbf{w}}(y^{(i)} = 0 \mid \mathbf{x}^{(i)}) = \log (1 - h_{\mathbf{w}}(\mathbf{x}^{(i)}))$
- So $P_{\mathbf{w}}(y^{(i)} \mid \mathbf{x}^{(i)}) = y^{(i)} \log h_{\mathbf{w}}(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\mathbf{w}}(\mathbf{x}^{(i)}))$

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

Convex function!

Gradient Descent

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

Want $\min_{\mathbf{w}}$ J(**w**):

Repeat {

$$w_j := w_j - \alpha \frac{\partial}{\partial w_j} J(\boldsymbol{w})$$

} (simultaneously update all $\mathbf{w}_j$)

$$\frac{\partial}{\partial w_j} J(\boldsymbol{w}) = \frac{1}{m}\sum_{i=1}^{m}(h_{\boldsymbol{w}}\left(\boldsymbol{x}^{(\boldsymbol{i})}\right) - y^{(i)})x_j^{(i)}$$

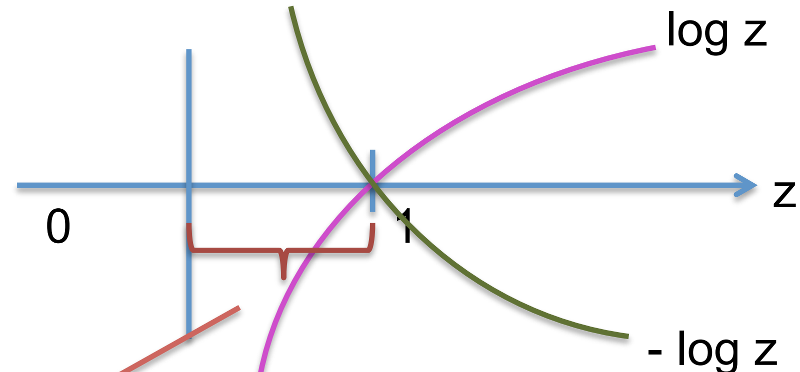
$$w_j := w_j - \alpha \sum_{i=1}^{m}(h_{\boldsymbol{w}}\left(\boldsymbol{x}^{(\boldsymbol{i})}\right) - y^{(i)})x_j^{(i)}$$

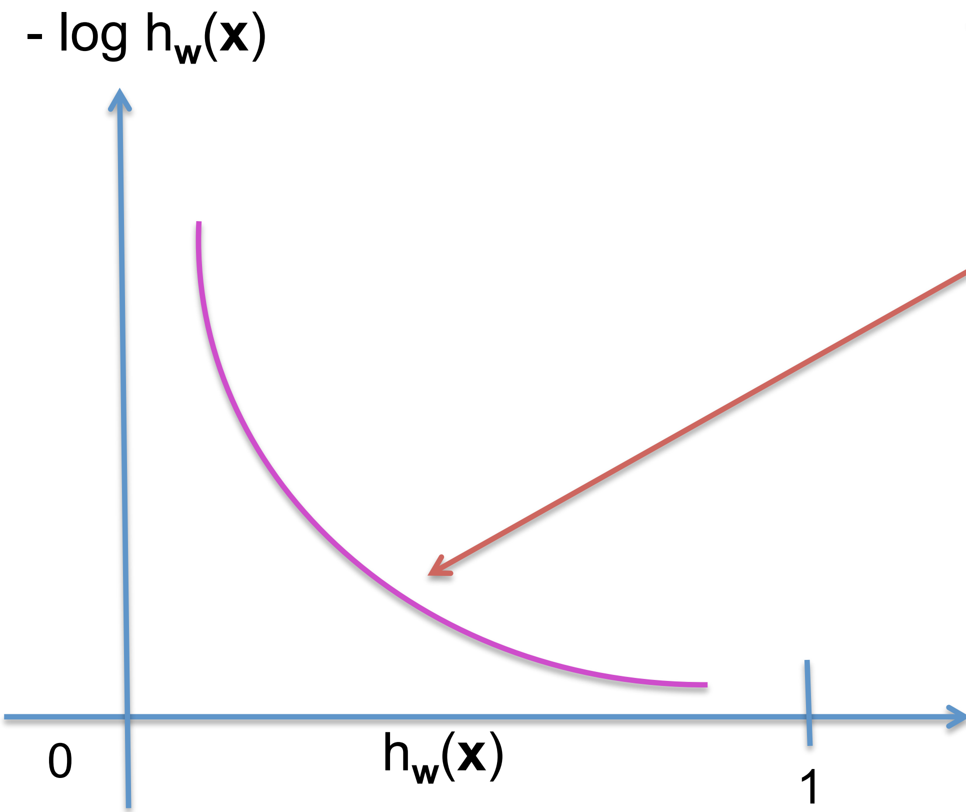
Algorithm looks identical to linear regression! But it is not!

Some intuition into cost function

$$Cost(h_{\boldsymbol{w}}(\boldsymbol{x}), y) = \begin{cases} -\log\left(h_{\boldsymbol{w}}(\boldsymbol{x})\right) \; if \; y = 1 \\ -\log\left(1 - h_{\boldsymbol{w}}(\boldsymbol{x})\right) \; if \; y = 0 \end{cases}$$

Consider y=1





Cost = 0 if y=1, $h_w(\mathbf{x})$ = 1.
But as $h_w(\mathbf{x}) \rightarrow 0$, Cost $\rightarrow \infty$
Captures intuition that if $h_w(\mathbf{x}) = 0$, (predict $P_w(y=1 | \mathbf{x})=0$), we'll penalize learning algorithm by a very large cost.

Summary: Logistic regression cost function

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

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

To fit parameters **w**:

$$\min_{\boldsymbol{w}} J(\boldsymbol{w})$$

To make a prediction given new **x**:

Output $$h_{\boldsymbol{w}}(x) = \frac{1}{1 + e^{-\boldsymbol{w}\cdot\boldsymbol{x}}}$$