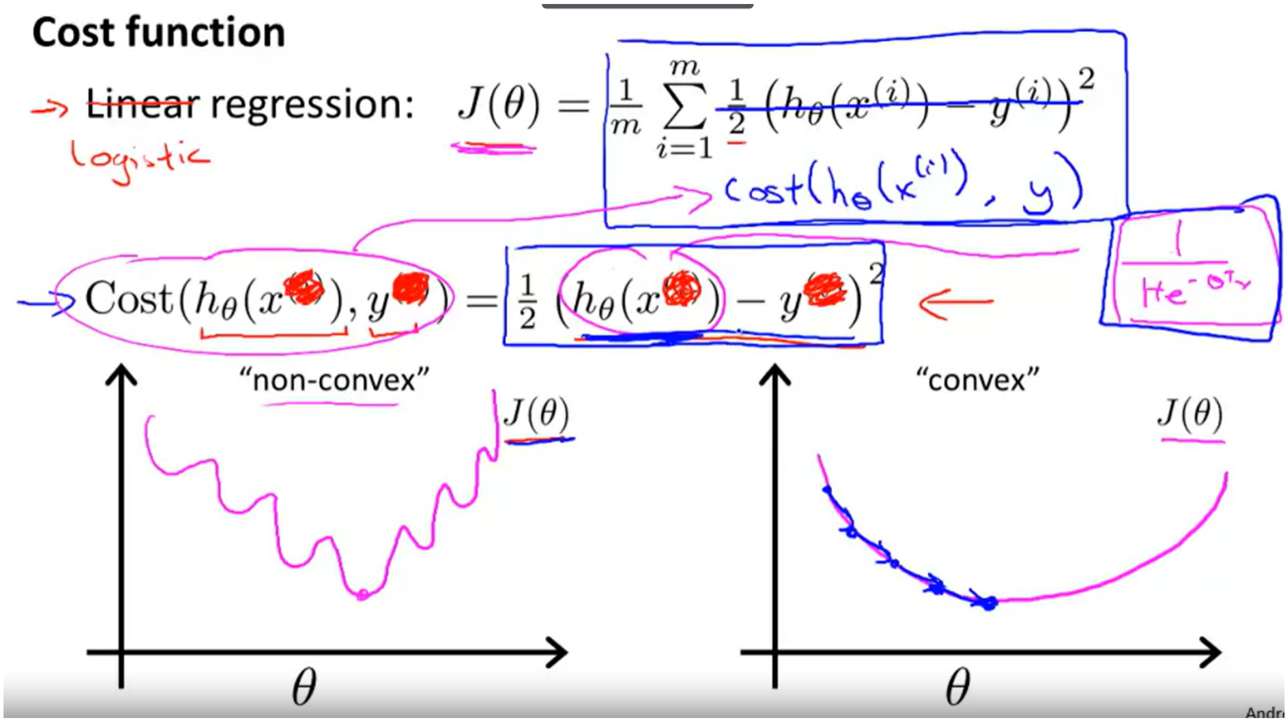
Week 3.2 Logistic Regression Mode

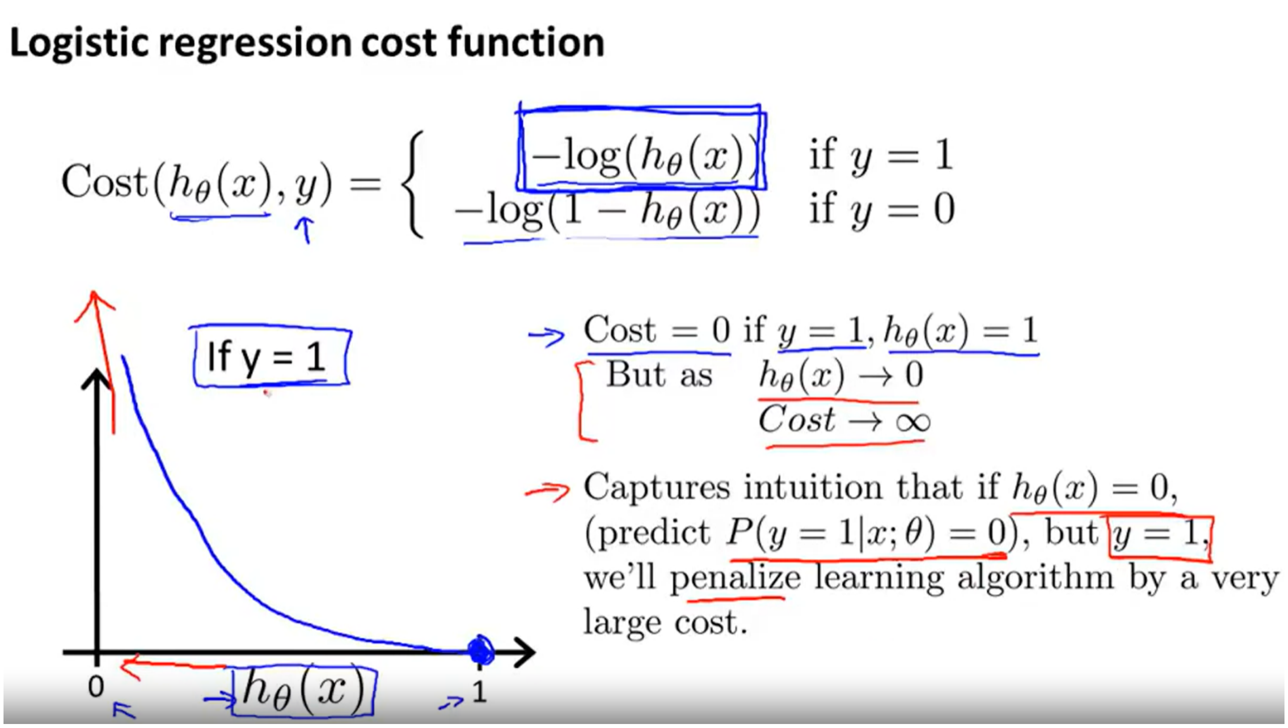
Cost Function

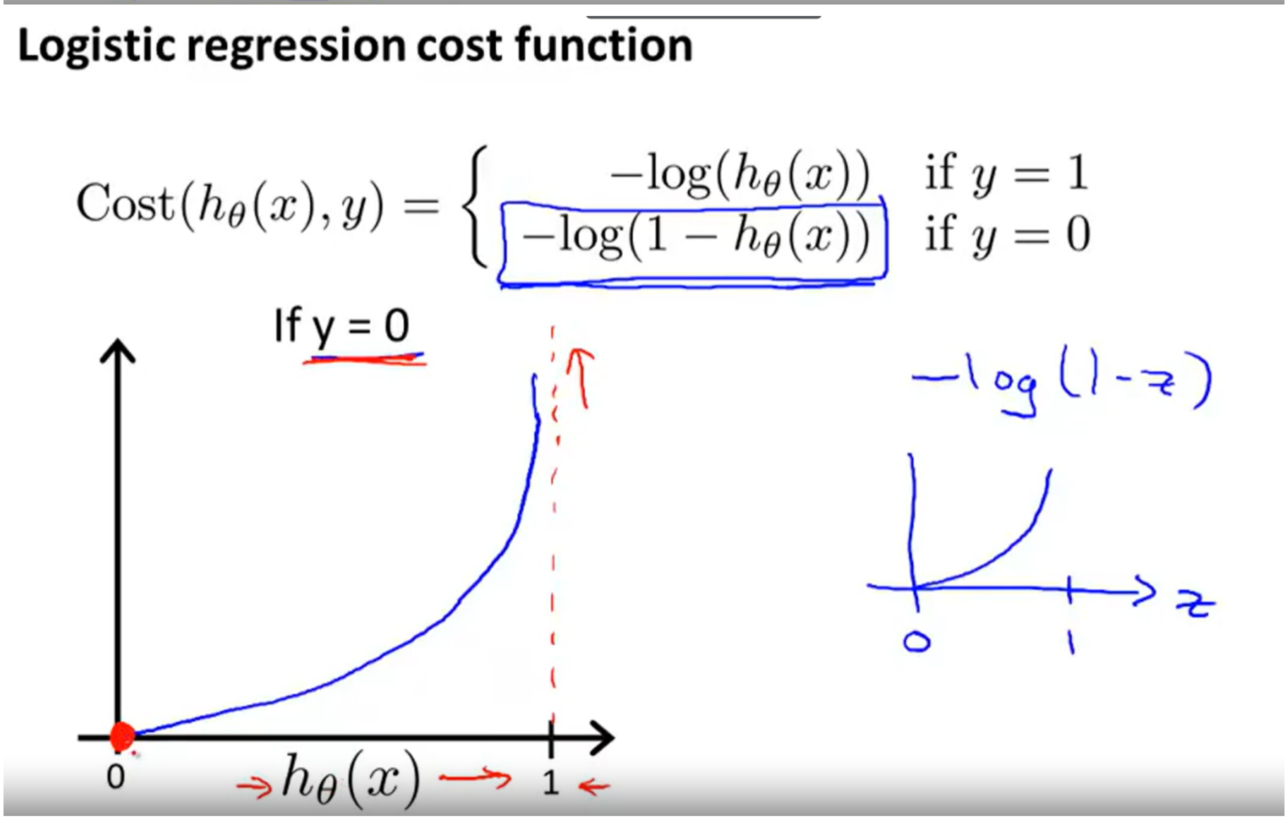
- convex problem about logistic regression when we use linear regression's cost function



- **Cost function of logistic regression**

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





Simplified Cost Function and Gradient Descent

- simplified cost function:
- $$Cost(h_\theta(x),y) = -ylog(h_\theta(x)) - (1-y)log(1-h_\theta(x))$$
- Entire cost function:
- $$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))]$$

Gradient Descent

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

Want $\min_\theta J(\theta)$:

$$\Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \quad \text{for } i = 0 \text{ to } n$$

Repeat {

$$\boxed{\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (\boxed{h_\theta(x^{(i)})} - y^{(i)}) x_j^{(i)}}$$

(simultaneously update all θ_j)

}

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

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

Algorithm looks identical to linear regression!

Advanved Optimization

Suppose we want to $min_\theta J(\theta)$
we have code to calculate $J(\theta)$ and $\frac{\partial}{\partial(\theta_j)}J(\theta)$

Optimization Algorithm:

- Gradient descent
- Conjugate gradient
- BFGS
- L-BFGS

- Advantages:

    1. no need to manually pick α
    2. often faster than gradient descent

- Disadvantages:

    - More complex

- Code in Octave:

Example: $\min_\theta J(\theta)$

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$\theta_1 = 5, \theta_2 = 5.$

$$\boxed{J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2}$$

$$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$$

$$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$$

```
function [jVal, gradient]
            = costFunction(theta)
  jVal = (theta(1)-5)^2 + ...
         (theta(2)-5)^2;
  gradient = zeros(2,1);
  gradient(1) = 2*(theta(1)-5);
  gradient(2) = 2*(theta(2)-5);
```

```
options = optimset('GradObj', 'on', 'MaxIter', '100');
initialTheta = zeros(2,1);
[optTheta, functionVal, exitFlag] ...
     = fminunc(@costFunction, initialTheta, options);
```