Week 3

- ## Classification and Representation:
  - ### a. Classification:

    E.g. email:Spam/not spam?

    Online:Transcations:frauluent or not?

    Tumor:malignant/benign?

    $y \in \{0, 1\}$ → negative class {e.g. benign tumor}
    → positive class {e.g. malignant tumor}.

    How do we develop a classification algorithm?

    1.solution: linear regression:

    Classification:y=0 or 1 but the hypothesis can be lager than 1 or smaller than 0.

    2.Solution: logistic regression: $0 <= h_\theta <= 1$

  - ### b. Hypothesis Representation:

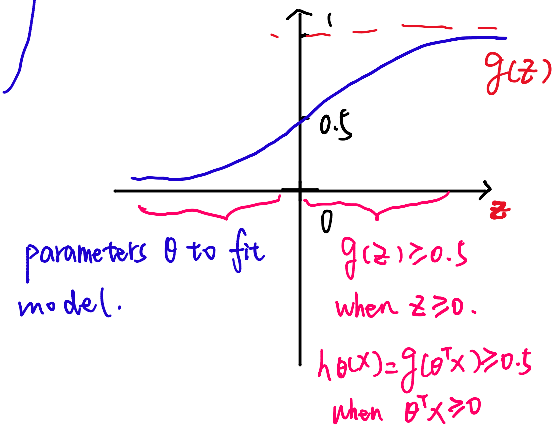
    Our aim :$0 <= h_\theta <= 1$

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

    sigmoid function / logistic function $\in$ $g(z) = \frac{1}{1+e^{-z}}$ ← $\theta^T x$

    $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ ⇒ $z \to \infty, e^{-\infty} \to 0 \Rightarrow g(z) = 1$.

    $z \to -\infty, e^{\infty} \to \infty \Rightarrow g(z) = 0$.

    parameters θ to fit model.

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

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

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

    $h_\theta(x) = p(y=1|x;\theta)$.

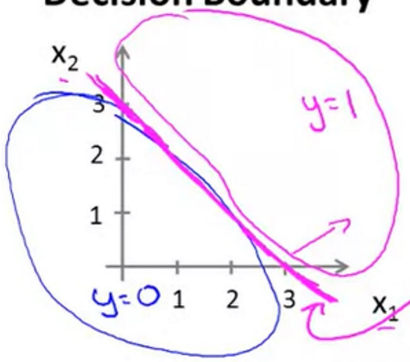
    $p(y=1|x;\theta) + p(y=0|x;\theta) = 1$.

  - ### c. Decision Boundary: → property of hypothesis under parameters.

    Suppose predict "y=1" if $h_\theta(x) \geq 0.5 \leftarrow \theta^T x \geq 0$

    predict "y=0" if $h_\theta(x) < 0.5 \leftarrow \theta x < 0$.

Decision Boundary



$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

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

$-3 \quad 1 \quad 1$

Decision boundary

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

$\theta^T x$

$x_1 + x_2 \geq 3$

x_1, x_2

$h_\theta(x) = 0.5$

$x_1 + x_2 = 3$

$x_1 + x_2 < 3$

$y = 0$

- **Logistic Regression Model:**
  - a. Cost Function:

    Training set:{$(x^{(1)},y^{(1)}),(x^{(2)},y^{(2)}),\ldots,(x^{(m)},y^{(m)})$}

    m examples

$x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$

$x_0 = 1, y \in \{0, 1\}$

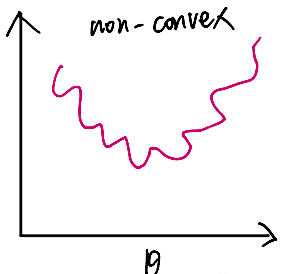
$h_\theta = \frac{1}{1+e^{-\theta^T x}}$

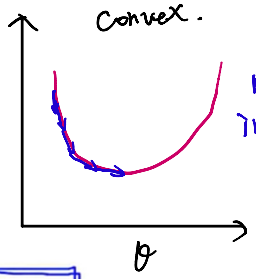
Linear regression: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$

$\text{cost}(h_\theta(x^{(i)}), y)$

Simplify

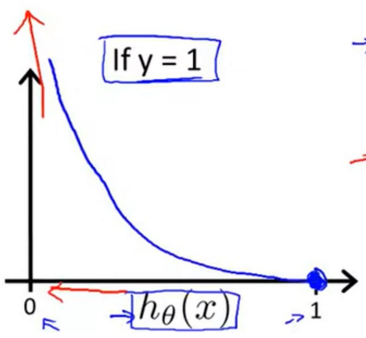
$\text{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$





non-convex in logistic regression

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

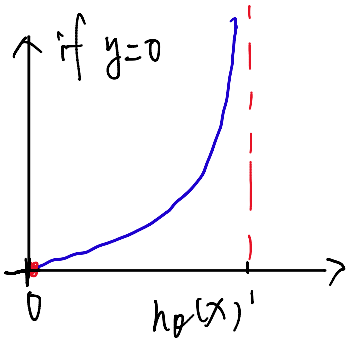


→ Cost = 0 if $y = 1, h_\theta(x) = 1$

But as $h_\theta(x) \to 0$

$Cost \to \infty$

→ Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x;\theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.



predict $P(y=0|x;\theta)=0$, but $y=0$

cost $\to \infty$. as $h_\theta(x) \to 0$

cost $\to \infty$.

b.Simplified Cost Function and Gradient Descent:

Simplified form:

$\text{cost}(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$.

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

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

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

$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$

Repeat {

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

(simultaneously update all θ_j)

}

linear regression

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

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

logistic regression.

Algorithm looks identical to linear regression!

c.Advanced Optimization:

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

Advantages:

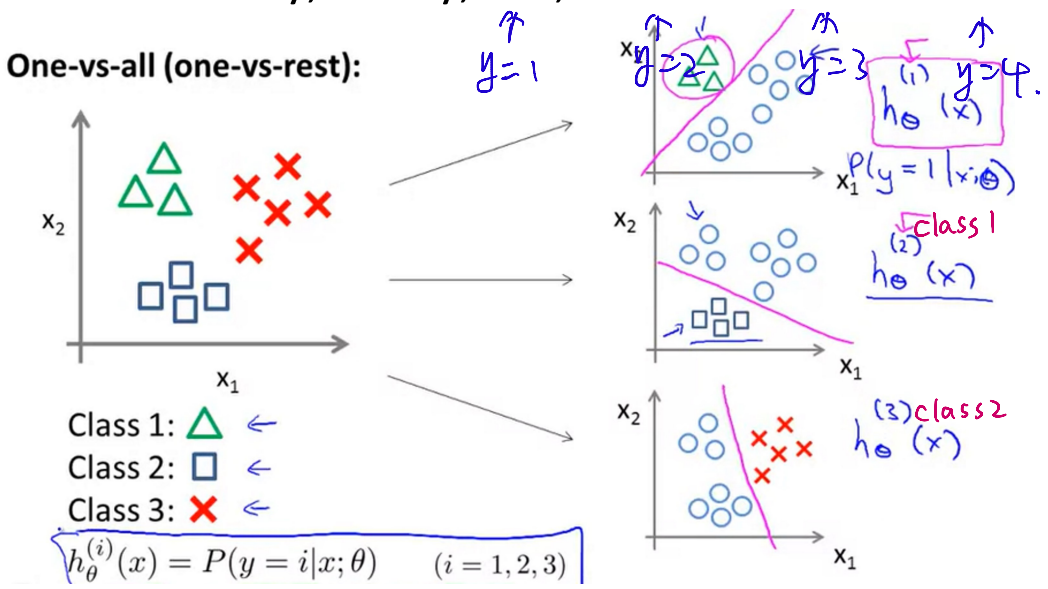
- no need to manually pick α
- Often faster than GD

Disadvantages:

- more complex

- # Multiclass Classification:
  a. One-vs-all:
  Email foldering/tagging:work,friends, family, hobby

  Medical diagrams:not ill, cold,flu.
  Weather:sunny,cloudy,rain,snow

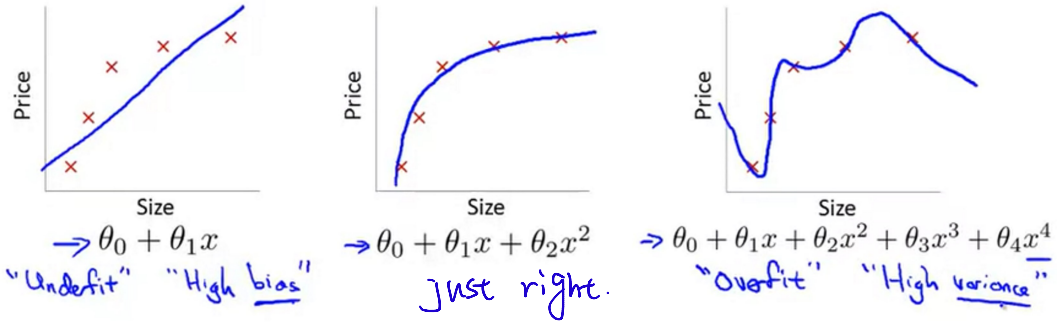


On a new input x, to make a prediction, pick the class i that maximizes

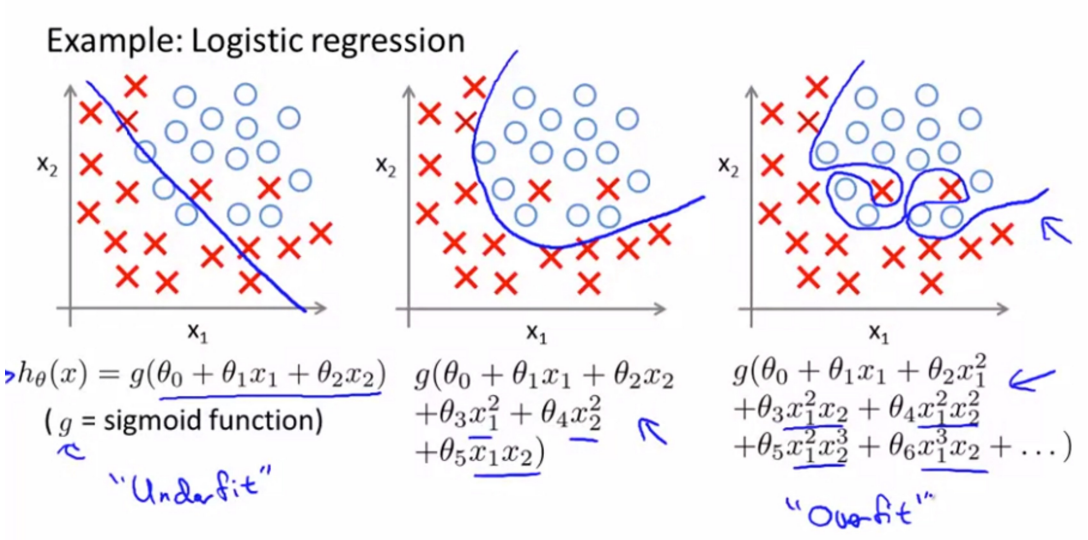
$$\max_i h_\theta^{(i)}(x)$$

- # Solving the Problem of Overfitting:
  a. The problem of Overfitting:

Example: Linear regression (housing prices)



Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).



A lot of features and little training examples may result in overfitting.

Options:

1. Reduce number of features.
   - — Manually select which features to keep.
   - — Model selection algorithm (later in course).
2. Regularization.
   - — Keep all the features, but reduce magnitude/values of parameters θ_j.
   - — Works well when we have a lot of features, each of which contributes a bit to predicting y.

b. Cost Function:

Regularization.

Small values for parameters $\theta_0, \theta_1, \ldots, \theta_n$
- — "Simpler" hypothesis
- — Less prone to overfitting

$\theta_3, \theta_4 \approx 0$

Housing:
- — Features: $x_1, x_2, \ldots, x_{100}$
- — Parameters: $\theta_0, \theta_1, \theta_2, \ldots, \theta_{100}$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

regularization parameter. regularization term.

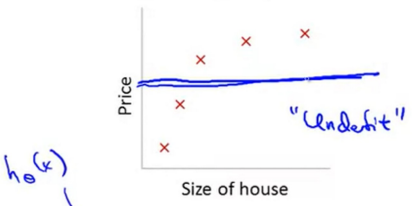
$\theta_0, \theta_1, \theta_2, \theta_3, \ldots, \theta_{100}$

fit training data; keep θ small

control a trade off between two different goals.

modify cost function to shrink all parameters

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?



$h_\theta(x)$

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$\theta_1, \theta_2, \theta_3, \theta_4$

$\theta_1 \approx 0, \theta_2 \approx 0$

$\theta_3 \approx 0, \theta_4 \approx 0$

$h_\theta(x) = \theta_0$

c. Regularized Linear Regression:

Gradient descent

$\theta_0, \theta_1, \theta_2, \ldots, \theta_n$

Repeat {

we don't penalise θ_0

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$\frac{\partial}{\partial \theta_0} J(\theta)$

$$\theta_j := \theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} + \frac{\lambda}{m} \theta_j \right]$$

$(j = 1, 2, 3, \ldots, n)$

update as usual.

}

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

shrink θ_j a little bit

$1 - \alpha \frac{\lambda}{m} < 1$

0.99

$\theta_j \times 0.99$

θ_j

Normal equation

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} \quad m \times (n+1)$$

$$y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \quad \mathbb{R}^m$$

$\min_\theta J(\theta)$

$\frac{\partial}{\partial \theta_j} J(\theta) \overset{\text{set}}{=} 0$

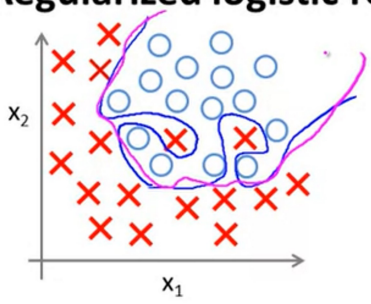
$$\theta = \left(X^T X + \lambda \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & 1 & \\ & & & \ddots \\ & & & & 1 \end{bmatrix} \right)^{-1} X^T y$$

$(n+1) \times (n+1)$

Eg. $n = 2$ $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

d. Regularized Logistic Regression:

Regularized logistic regression.



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

Cost function:

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

$\theta_1, \theta_2, \dots, \theta_n$

Gradient descent

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \left[\frac{1}{m}\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$(j = 1, 2, 3, \dots, n)$

$\theta_1 \dots \theta_n$

} treat θ_0 seperately.

$\frac{\partial}{\partial \theta_j} J(\theta)$

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

Advanced optimization

fminunc (@costFunction, $\theta = \begin{bmatrix}\theta_0\\ \theta_1\\ \vdots\\ \theta_n\end{bmatrix}$) — theta(1), theta(2), theta(n+1)

```
function [jVal, gradient] = costFunction(theta)
    jVal = [code to compute J(θ)];
        J(θ) = [ -(1/m) Σ_{i=1}^{m} y^(i) log(h_θ(x^(i))) + (1 - y^(i)) log 1 - h_θ(x^(i)) ] + (λ/2m) Σ_{j=1}^{n} θ_j^2
    gradient(1) = [code to compute ∂/∂θ_0 J(θ)];
        (1/m) Σ_{i=1}^{m} (h_θ(x^(i)) - y^(i)) x_0^(i)
    gradient(2) = [code to compute ∂/∂θ_1 J(θ)];
        ( (1/m) Σ_{i=1}^{m} (h_θ(x^(i)) - y^(i)) x_1^(i) ) + (λ/m) θ_1
    gradient(3) = [code to compute ∂/∂θ_2 J(θ)];
        ( (1/m) Σ_{i=1}^{m} (h_θ(x^(i)) - y^(i)) x_2^(i) ) + (λ/m) θ_2
    ⋮
    gradient(n+1) = [code to compute ∂/∂θ_n J(θ)];
```

J(θ)