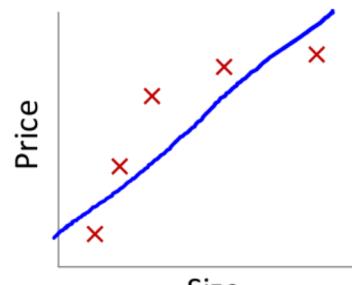


Machine Learning

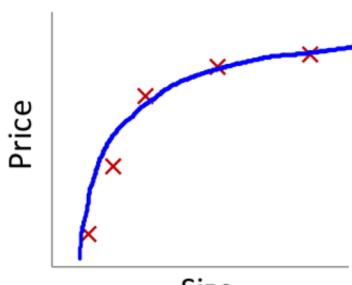
Regularization

The problem of overfitting

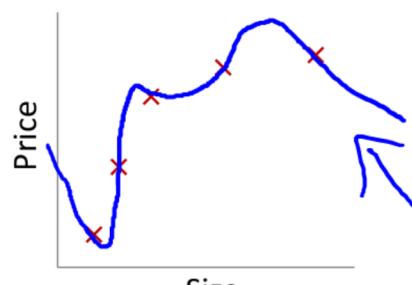
Example: Linear regression (housing prices)



$\rightarrow \theta_0 + \theta_1 x$
"Underfit" "High bias"



$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$
"Just right"

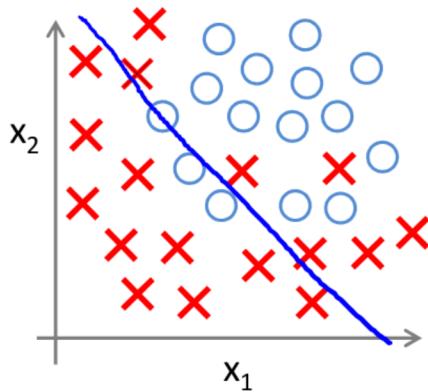


$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$
"Overfit" "High variance"

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).

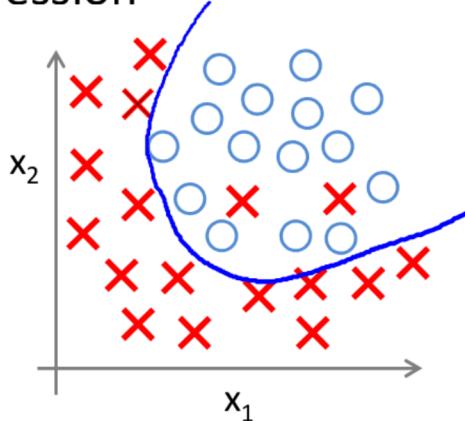
Andrew Ng

Example: Logistic regression



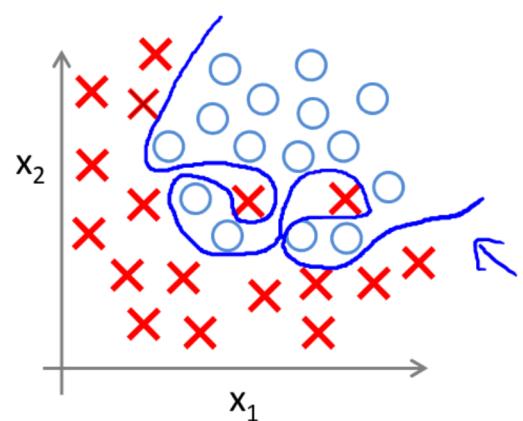
$\rightarrow h_{\theta}(x) = g(\underline{\theta_0 + \theta_1 x_1 + \theta_2 x_2})$
 $(g = \text{sigmoid function})$

↖ "Underfit"



$$g(\underline{\theta_0 + \theta_1 x_1 + \theta_2 x_2} \\ + \theta_3 \underline{x_1^2} + \theta_4 \underline{x_2^2} \\ + \theta_5 \underline{x_1 x_2})$$

↖

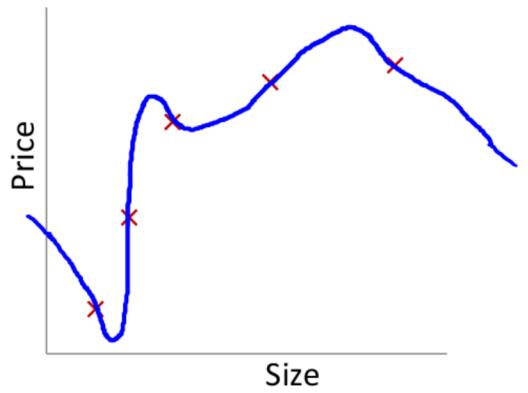


$$g(\underline{\theta_0 + \theta_1 x_1 + \theta_2 x_1^2} \\ + \theta_3 \underline{x_1^2 x_2} + \theta_4 \underline{x_2^2 x_2^2} \\ + \theta_5 \underline{x_1^2 x_2^3} + \theta_6 \underline{x_1^3 x_2} + \dots)$$

↖ "Overfit"

Addressing overfitting:

x_1 = size of house
 x_2 = no. of bedrooms
 x_3 = no. of floors
 x_4 = age of house
 x_5 = average income in neighborhood
 x_6 = kitchen size
:
 x_{100}

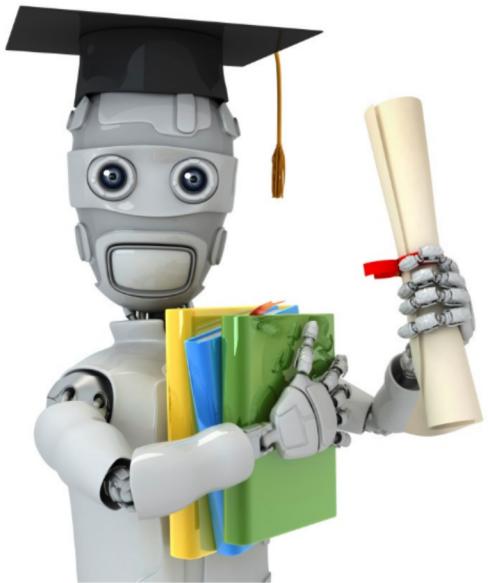


Andrew Ng

Addressing 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 .

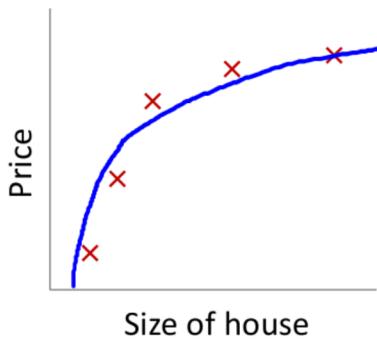


Machine Learning

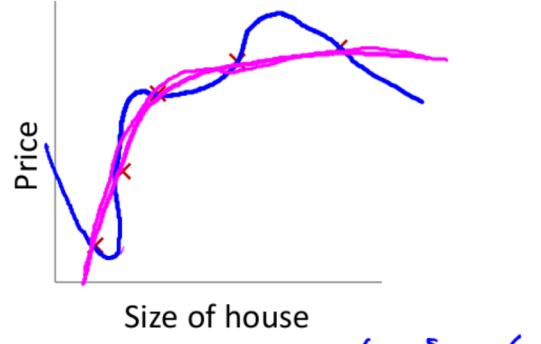
Regularization

Cost function

Intuition



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



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

Suppose we penalize and make θ_3, θ_4 really small.

$$\rightarrow \min_{\theta} \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + 1000 \underline{\theta_3^2} + 1000 \underline{\theta_4^2}$$

$\underline{\theta_3 \approx 0}$ $\underline{\theta_4 \approx 0}$

Andrew Ng

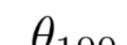
Regularization.

Small values for parameters $\theta_0, \theta_1, \dots, \theta_n$ 

- “Simpler” hypothesis 
- Less prone to overfitting 

θ_3, θ_4 
 ≈ 0 

Housing:

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

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

~~$\theta_1, \theta_2, \theta_3, \dots, \theta_{100}$~~ 

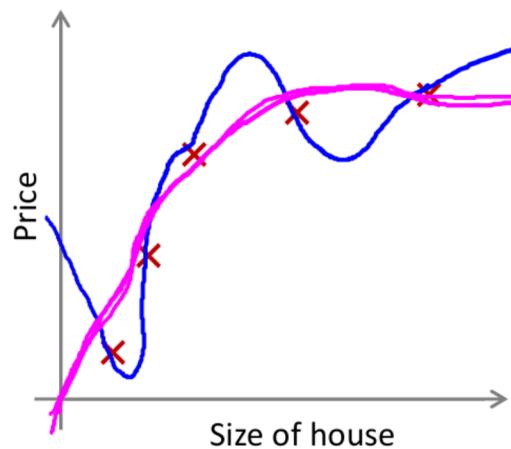
Andrew Ng

Regularization.

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

$\min_{\theta} J(\theta)$

regularization parameter



Andrew Ng

In regularized linear regression, we choose θ to minimize

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

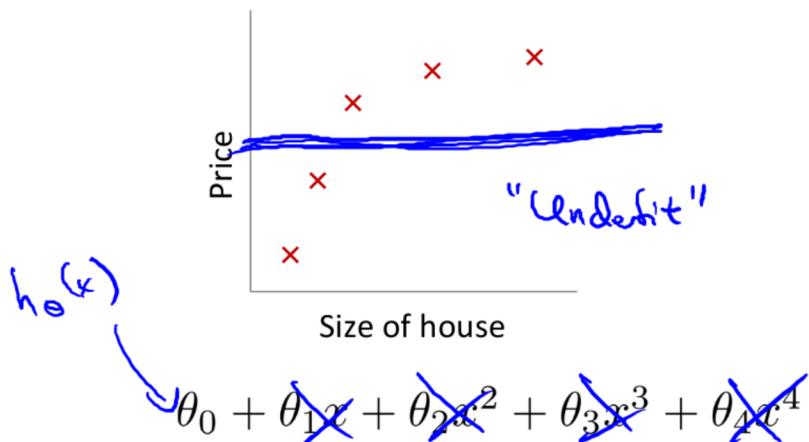
What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

- Algorithm works fine; setting λ to be very large can't hurt it
- Algorithm fails to eliminate overfitting.
- Algorithm results in underfitting. (Fails to fit even training data well).
- Gradient descent will fail to converge.

In regularized linear regression, we choose θ to minimize

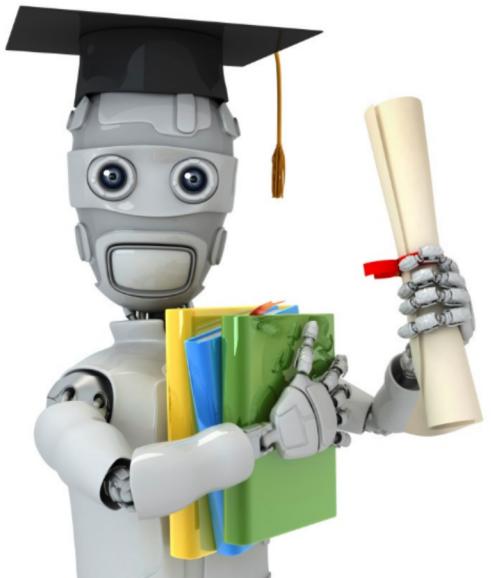
$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

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



$$\begin{aligned}\theta_1, \theta_2, \theta_3, \theta_4 \\ \theta_1 \approx 0, \theta_2 \approx 0 \\ \theta_3 \approx 0, \theta_4 \approx 0 \\ h_\theta(x) = \theta_0\end{aligned}$$

Andrew Ng



Regularization

Regularized linear regression

Machine Learning

Regularized linear regression

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

$\min_{\theta} J(\theta)$

↑ —

Gradient descent

Repeat {

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

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

$$\theta_j := \theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} + \frac{\lambda}{m} \theta_j \right] \quad (j = 1, 2, 3, \dots, n)$$

}

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

$$J(\theta)$$

$$\theta_j$$

$$\left| -\alpha \frac{\lambda}{m} \right| < 1 \quad 0.99 \quad \theta_j \times 0.99$$

Andrew Ng

Normal equation

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} \leftarrow \quad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbb{R}^m$$

$\rightarrow \min_{\theta} J(\theta)$

$\frac{\partial}{\partial \theta_j} J(\theta) \stackrel{\text{set}}{=} 0 \quad \text{when}$

$$\Rightarrow \theta = (X^T X + \lambda \underbrace{\begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}}_{(n+1) \times (n+1)})^{-1} X^T y$$

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

Andrew Ng

Non-invertibility (optional/advanced).

Suppose $m \leq n$, \leftarrow

(#examples) (#features)

$$\theta = \underbrace{(X^T X)^{-1}}_{\text{non-invertible / singular}} X^T y$$

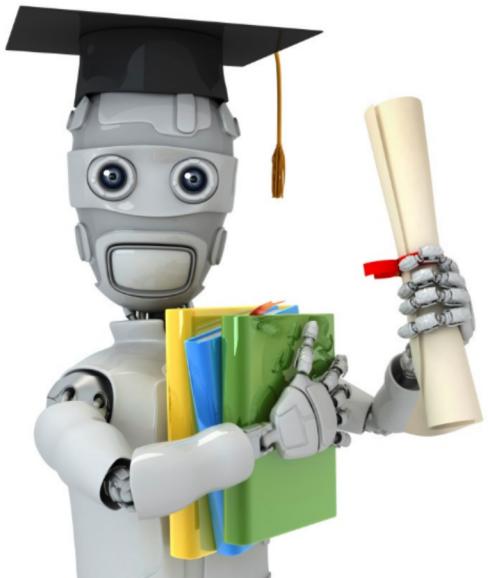
pinv

inv
 $\frac{1}{n}$

If $\lambda > 0$,

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

invertible.

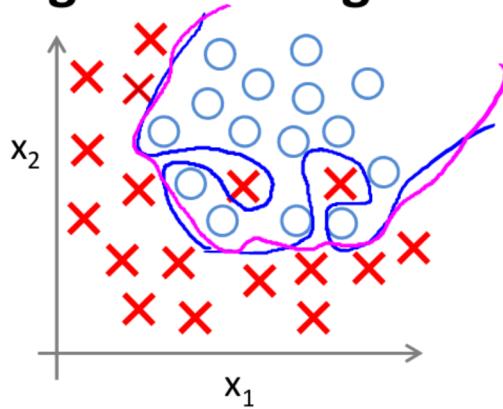


Regularization

Regularized logistic regression

Machine Learning

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:

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

$\boxed{\theta_1, \theta_2, \dots, \theta_n}$

Andrew Ng

Gradient descent

Repeat {

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

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

$(j = \cancel{0}, 1, 2, 3, \dots, n)$
 $\theta_0, \dots, \theta_n$

}

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

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

Andrew Ng

Advanced optimization

function [jVal, gradient] = costFunction(theta)

jVal = [code to compute $J(\theta)$];

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

$\rightarrow \text{gradient}(1) = [\text{code to compute } \frac{\partial}{\partial \theta_0} J(\theta)]$; $J(\theta)$

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

$\rightarrow \text{gradient}(2) = [\text{code to compute } \frac{\partial}{\partial \theta_1} J(\theta)]$; $J(\theta)$

$$\left(\frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)} \right) - \frac{\lambda}{m} \theta_1$$

$\rightarrow \text{gradient}(3) = [\text{code to compute } \frac{\partial}{\partial \theta_2} J(\theta)]$;

$$\vdots \quad \left(\frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)} \right) - \frac{\lambda}{m} \theta_2$$

$\text{gradient}(n+1) = [\text{code to compute } \frac{\partial}{\partial \theta_n} J(\theta)]$;

