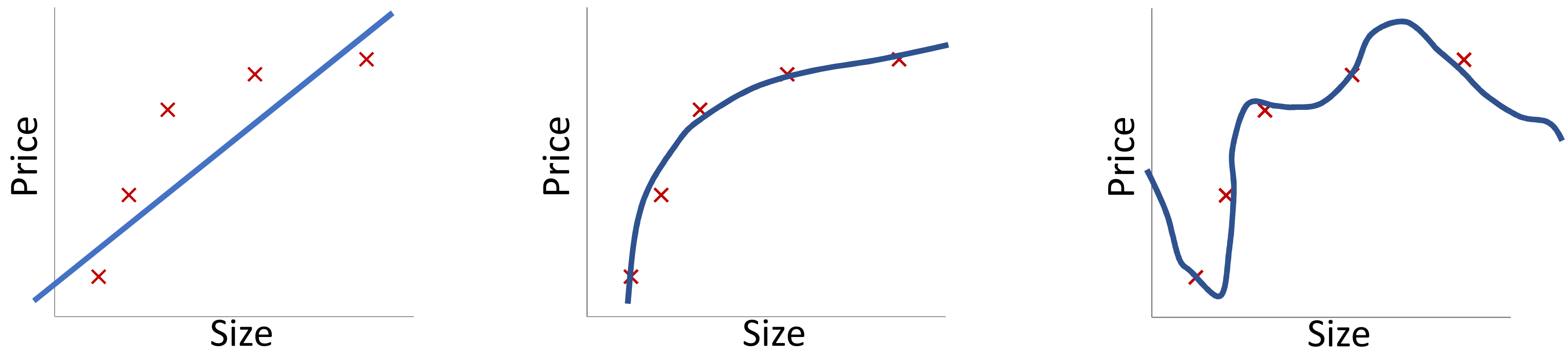
Regularization

Duration: 2 hrs

- The problem of overfitting
- Cost function
- Regularized linear regression
- Regularized logistic regression

Source: Machine Learning, Andrew Ng, coursera.org

Example: Linear regression (housing prices)



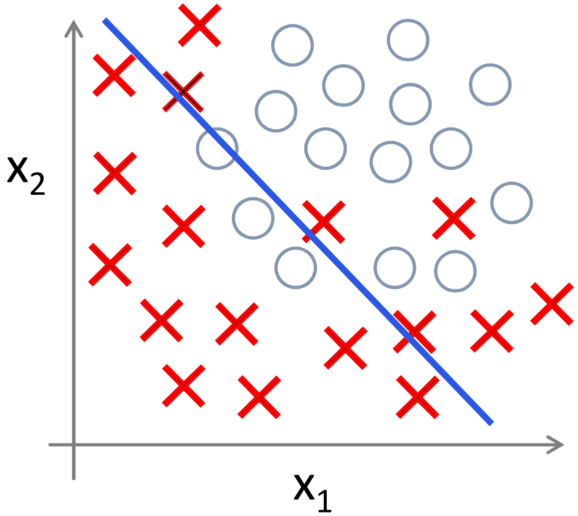
$\theta_0 + \theta_1 x$ — "Underfit" or "high bias"

$\theta_0 + \theta_1 x + \theta_2 x^2$ — "just right"

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ — "overfit" or "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).

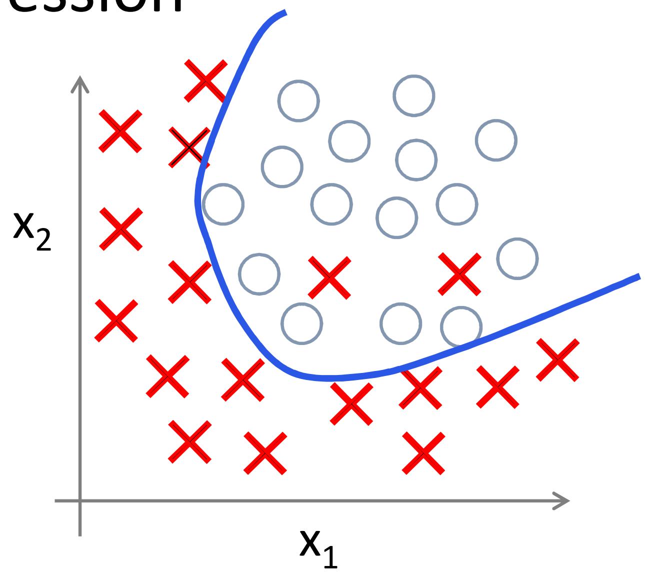
Example: Logistic regression



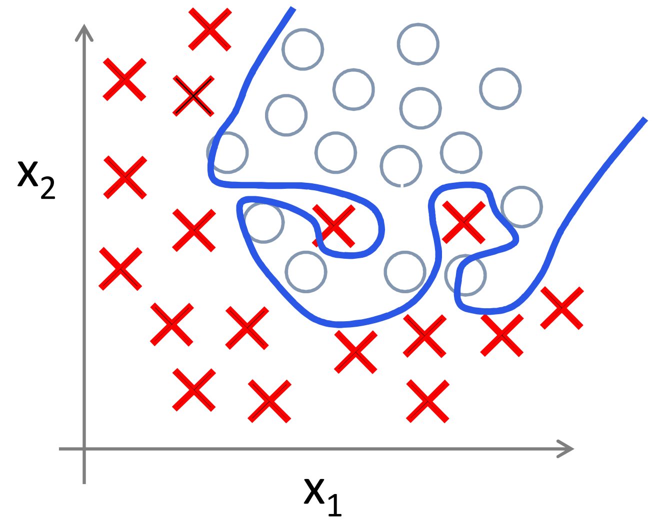
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

Underfit



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



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

overfit

Addressing overfitting:

$x_1 =$ size of house

$x_2 =$ no. of bedrooms

$x_3 =$ no. of floors

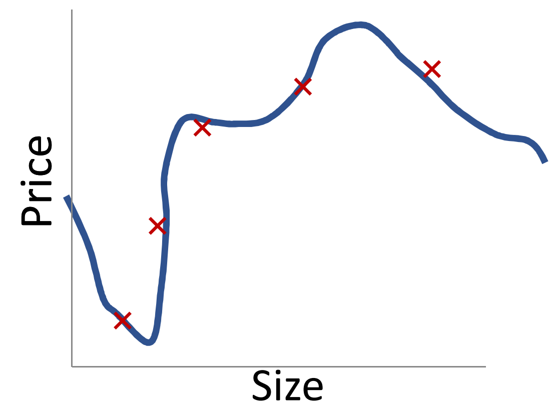
$x_4 =$ age of house

$x_5 =$ average income in neighborhood

$x_6 =$ kitchen size

$\vdots$

x_{100}



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.

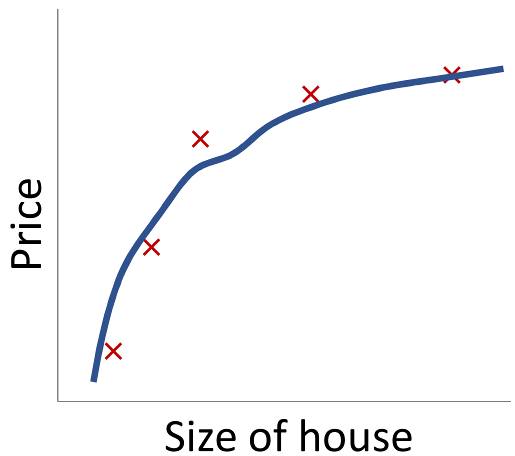
Regularization

Duration: 3 hrs

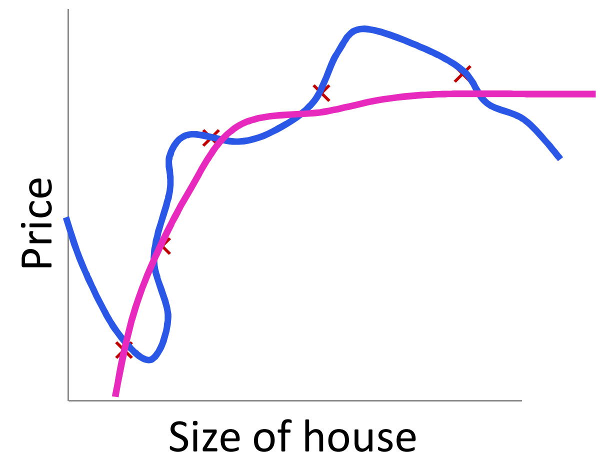
- The problem of overfitting
- Cost function
- Regularized linear regression
- Regularized logistic regression

Source: Machine Learning, Andrew Ng, coursera.org

Intuition



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



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

Suppose we penalize and make θ_3, θ_4 really small.

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

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

Regularization.

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

- "Simpler" hypothesis
- Less prone to overfitting

Housing:

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

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

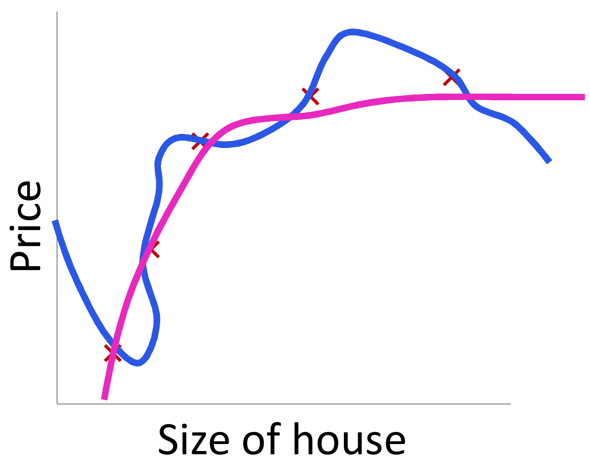
$\theta_1, \theta_2, \theta_3, \ldots, \theta_{100}$
Not θ_0

Regularization.

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

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

$\lambda: regularization\ parameter$



In regularized linear regression, we choose θ to minimize

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

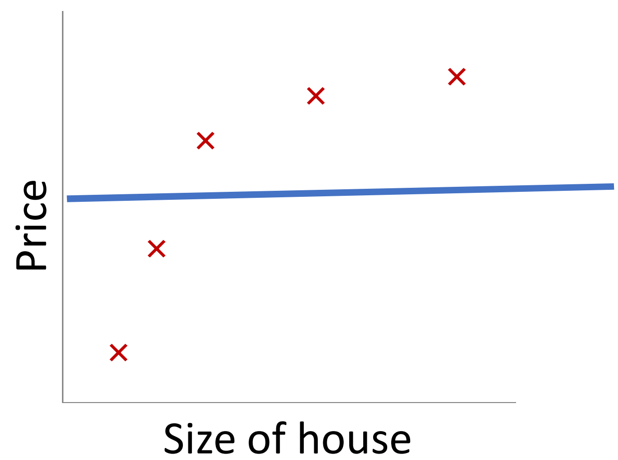
What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

- 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 $\lambda = 10^{10}$)?



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

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

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

$h_\theta(x) = \theta_0$

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

Regularization

Duration: 2 hrs

- The problem of overfitting
- Cost function
- Regularized linear regression
- Regularized logistic regression

Source: Machine Learning, Andrew Ng, coursera.org

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 $\{$

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

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

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

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

$\}$

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

$$1 - \alpha \frac{\lambda}{m} < 1 \qquad \theta_j \times 0.99$$

Normal equation

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

$m \times (n+1)$

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

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

Set $\frac{\partial}{\partial \theta_j} J(\theta) = 0$

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

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

Non-invertibility (optional/advanced).

Suppose $m \leq n$,

(#examples) (#features)

$$\theta = \underline{(X^T X)}^{-1} X^T y$$

Non-invertible/singular -> use pinv

If $\lambda > 0$,

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

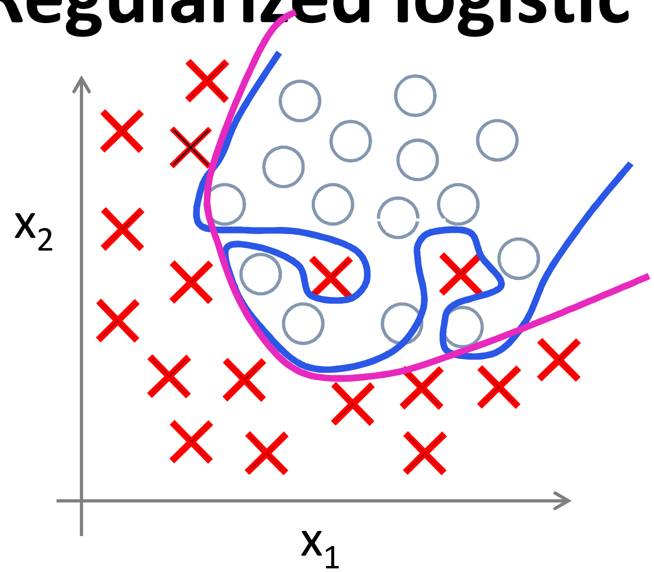
Regularization

Duration: 2 hrs

- The problem of overfitting
- Cost function
- Regularized linear regression
- Regularized logistic regression

Source: Machine Learning, Andrew Ng, coursera.org

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:

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

$$+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Gradient descent

Repeat $\{$

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

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

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

$\theta_1, \dots, \theta_n$

$\}$

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

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

Advanced optimization

```
function [jVal, gradient] = costFunction(theta)
```

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

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

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

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

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

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

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

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

$\vdots$

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