Regularization - The Problem of Overfitting

Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

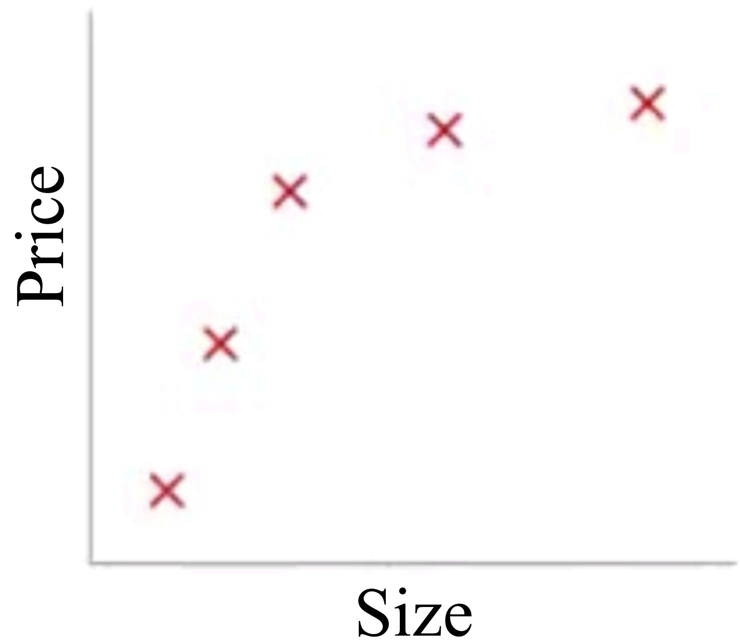
Summary about Supervised Learning

Our check-list before applying supervised learning techniques:

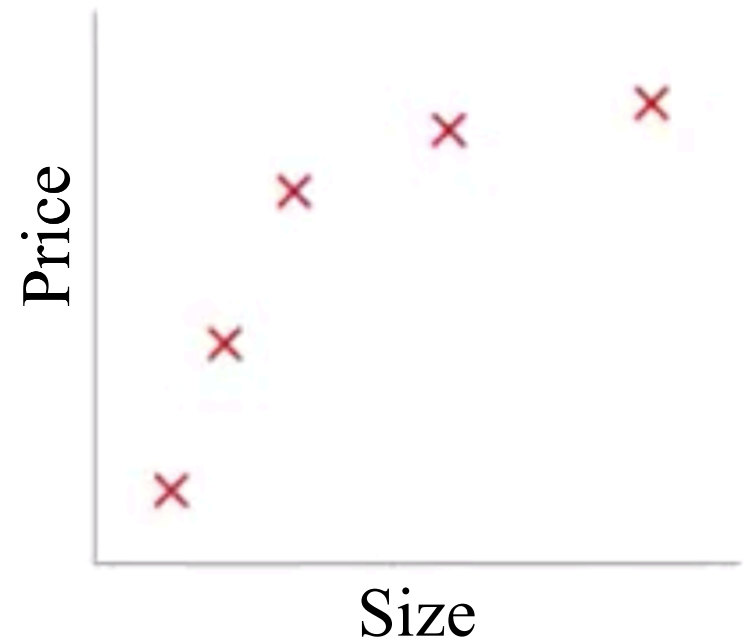
- If you have continuous $\mathscr{X}$ and continuous $\mathscr{Y}$, your first go-to model should be linear regression. Also, consider non-linear transformation of the inputs.
- If you have continuous $\mathscr{X}$ and discrete $\mathscr{Y}$ but don't know much about $p(\boldsymbol{x} \mid y)$, your first go-to model should be logistic or softmax regression, or may come up with a new GLM from scratch.
- If you have continuous $\mathscr{X}$ and discrete $\mathscr{Y}$ and know something about $p(\boldsymbol{x} \mid y)$, you should model the distribution accurately, as a Gaussian (GDA) or build a new generative model from scratch.
- If you have discrete $\mathscr{X}$ and $\mathscr{Y}$, you should probably start with naive Bayes and build up from there.

What is overfitting ?

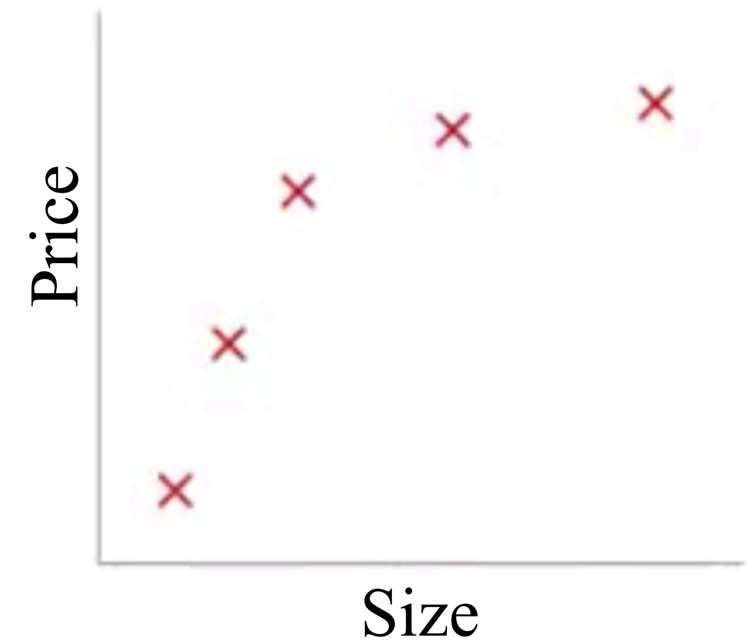
Example: Linear regression (housing prices)



$\theta_0 + \theta_1 x$



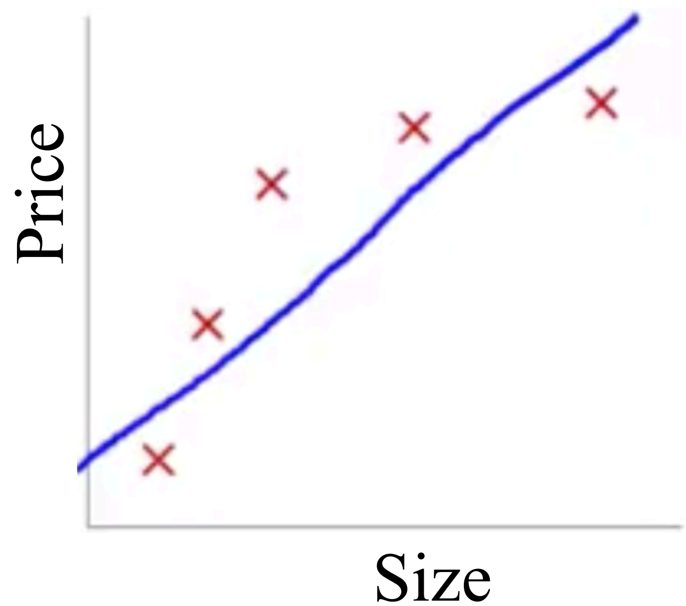
$\theta_0 + \theta_1 x + \theta_2 x^2$



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

What is overfitting ?

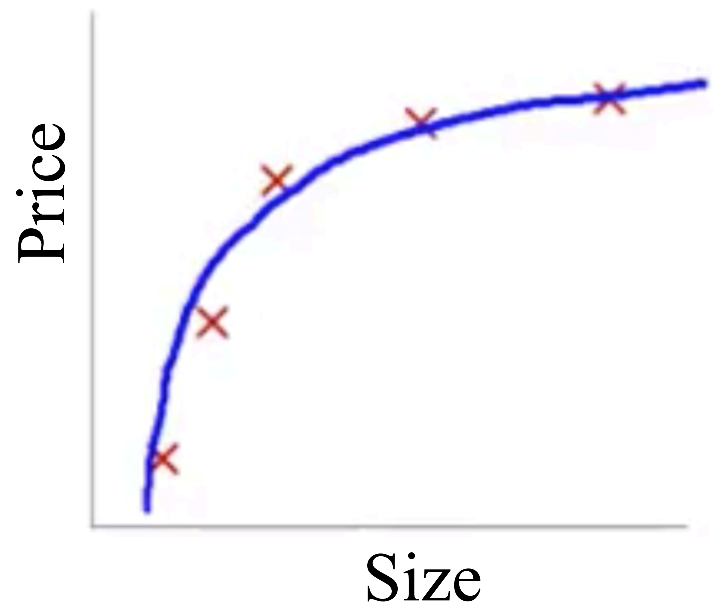
Example: Linear regression (housing prices)



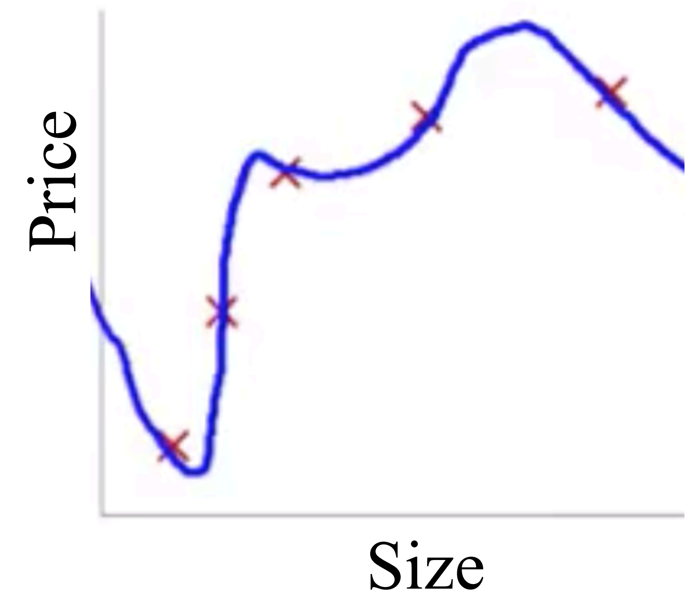
$\theta_0 + \theta_1 x$

'Underfit'

'High bias'



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



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

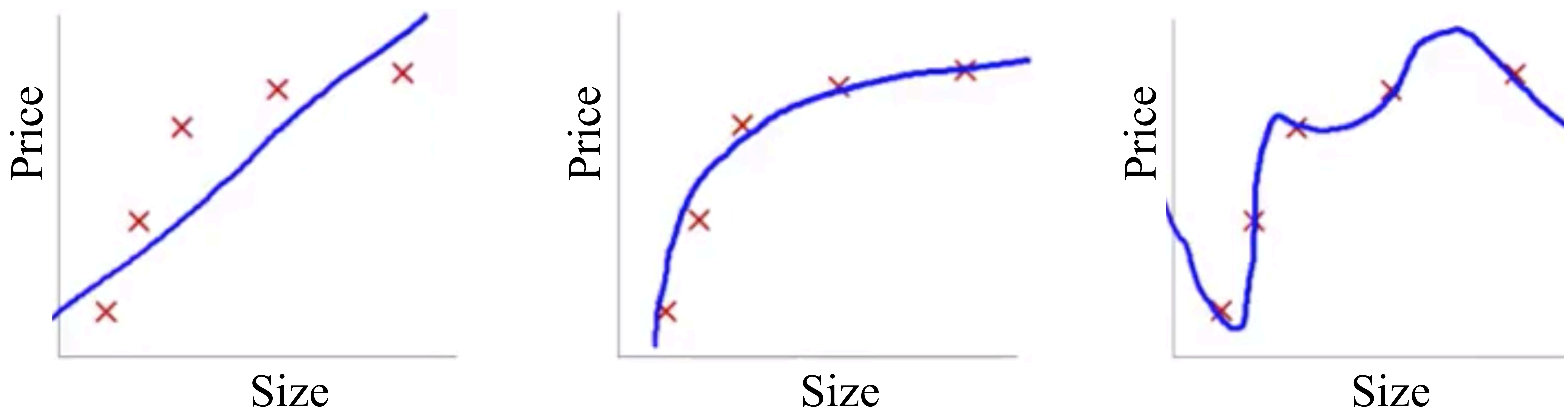
'Overfit'

'High variance'

Cannot use in real life

Overfitting in Linear Regression

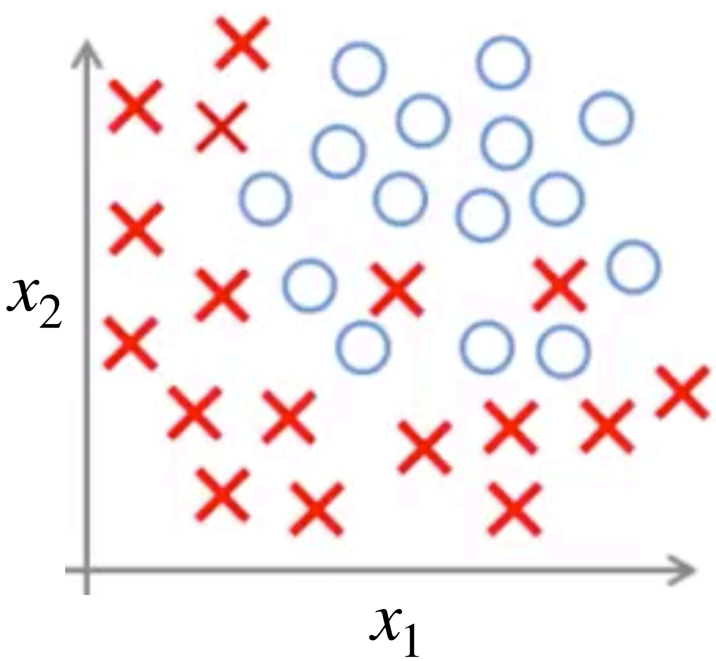
Example: Linear regression (housing prices)



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

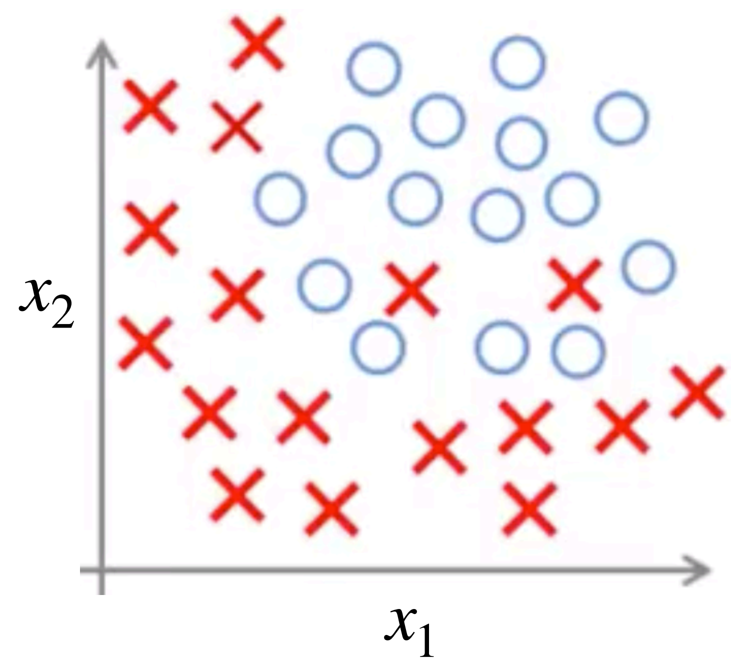
Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\Sigma_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to 'generalize' to new examples (*i.e.* predict prices on new examples).

Overfitting in Logistic Regression

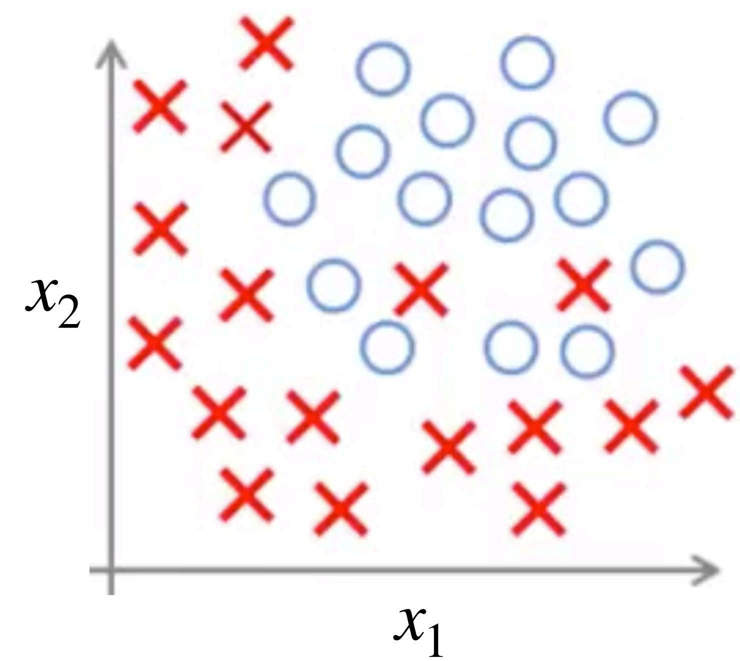


$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g is a sigmoid function)

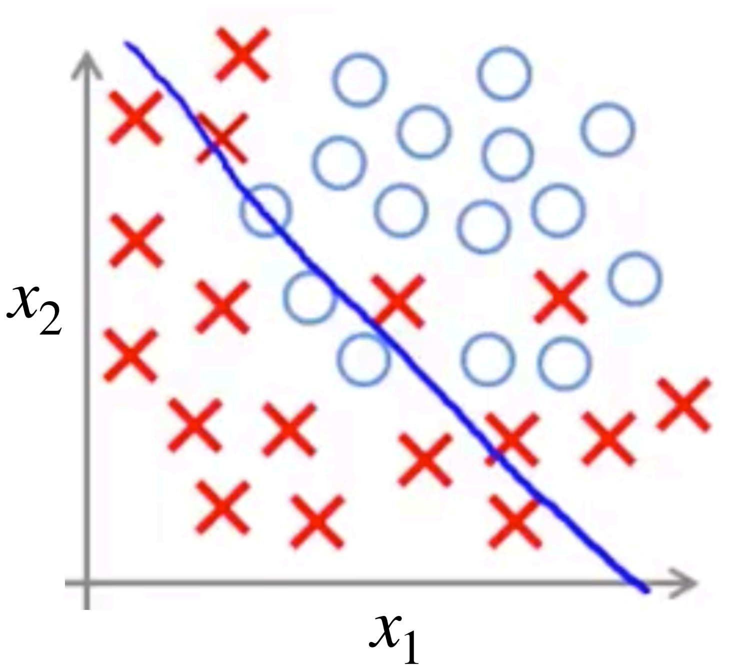


$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$$



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

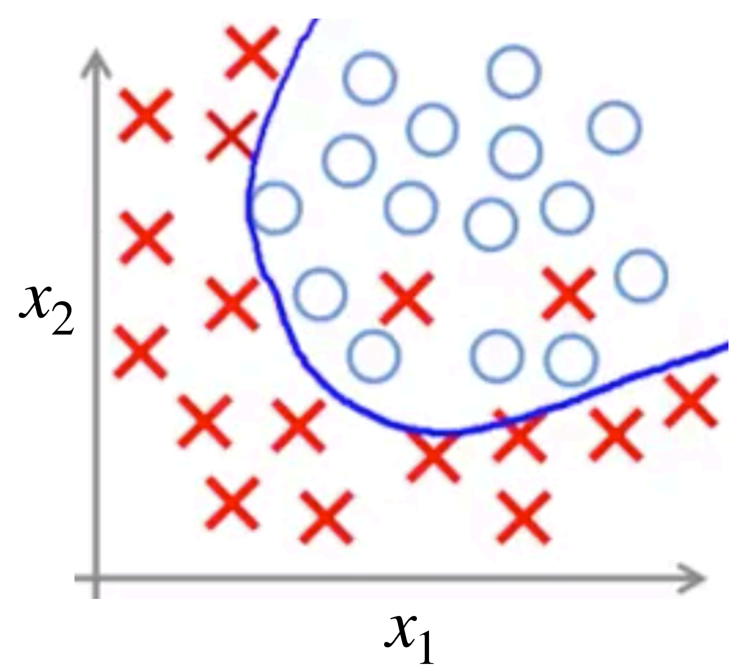
Overfitting in Logistic Regression



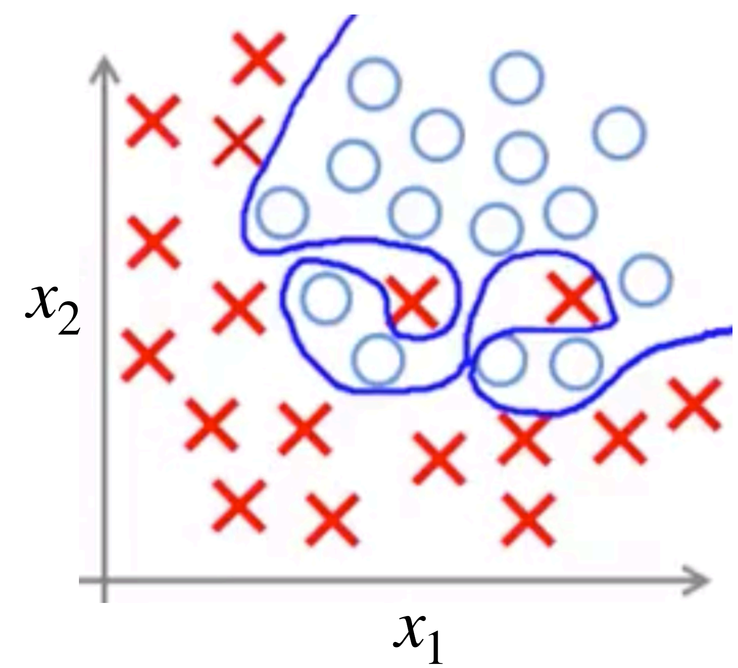
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

(g is a sigmoid function)

‘Underfit’
‘High bias’



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



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

‘Overfit’
‘High variance’

Question

Consider the medical diagnosis problem of classifying tumors as malignant or benign. If a hypothesis $h_\theta(x)$ has overfit the training set, it means that:

(i) It makes accurate predictions for examples in the training set and generalizes well to make accurate predictions on new, previously unseen examples.

(ii) It does not make accurate predictions for examples in the training set, but it generalizes well to make accurate predictions on new, previously unseen examples.

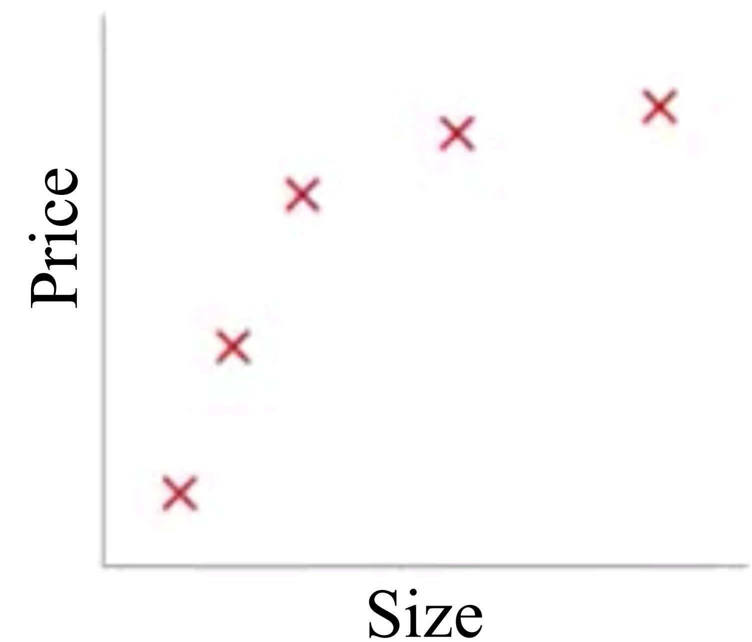
(iii) It makes accurate predictions for examples in the training set, but it does not generalize well to make accurate predictions on new, previously unseen examples.

(iv) It does not make accurate predictions for examples in the training set and does not generalize well to make accurate predictions on new, previously unseen examples.

Addressing Overfitting

Too many features may lead to overfitting !

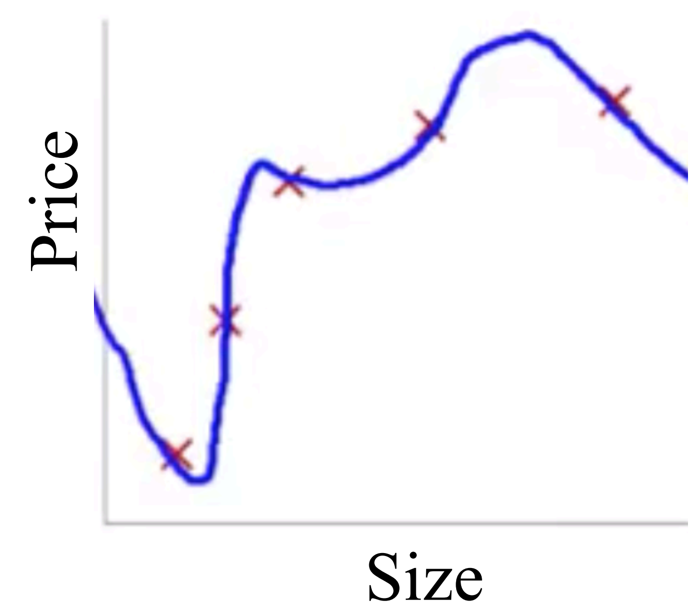
x_1 = size of house
x_2 = #bedrooms
x_3 = #floors
x_4 = age of house
x_5 = average income in neighborhood
x_6 = kitchen size
$\vdots$
x_{100}



Addressing Overfitting

Too many features may lead to overfitting !

x_1 = size of house
x_2 = #bedrooms
x_3 = #floors
x_4 = age of house
x_5 = average income in neighborhood
x_6 = kitchen size
$\vdots$
x_{100}



Addressing Overfitting

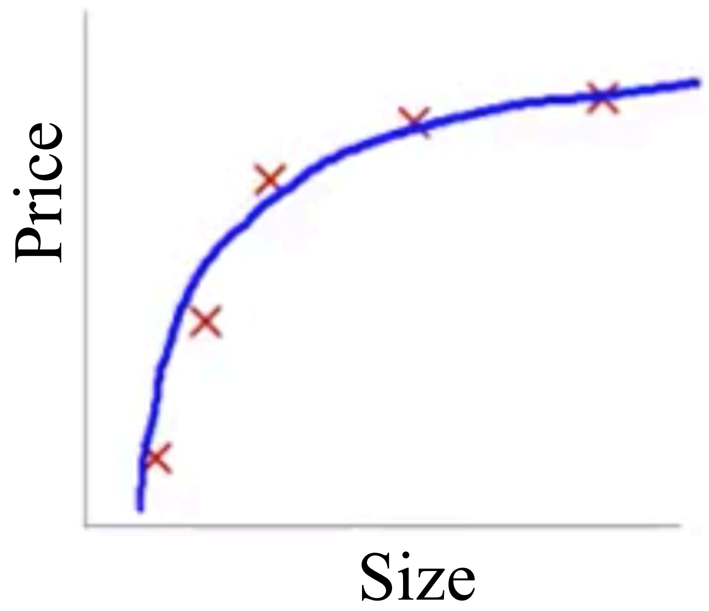
How to address overfitting:

1. Reduce number of features
   - Manually select which features to keep
   - Model selection algorithm (later in this class)

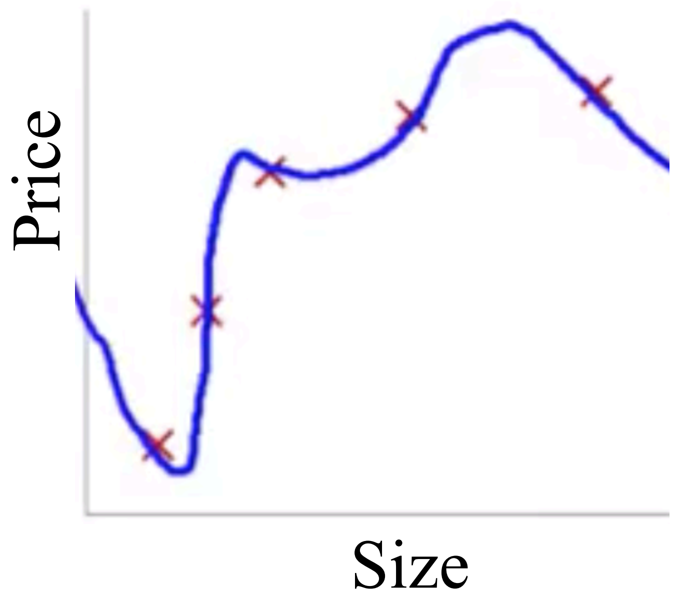
2. Regularization
   - Keep all the features, but reduce magnitude/values of parameters θ_j.
   - Works well when we have a lot of features, each of which contributes a bit to predicting y.

Regularization - Cost Function

Regularization (Intuition)

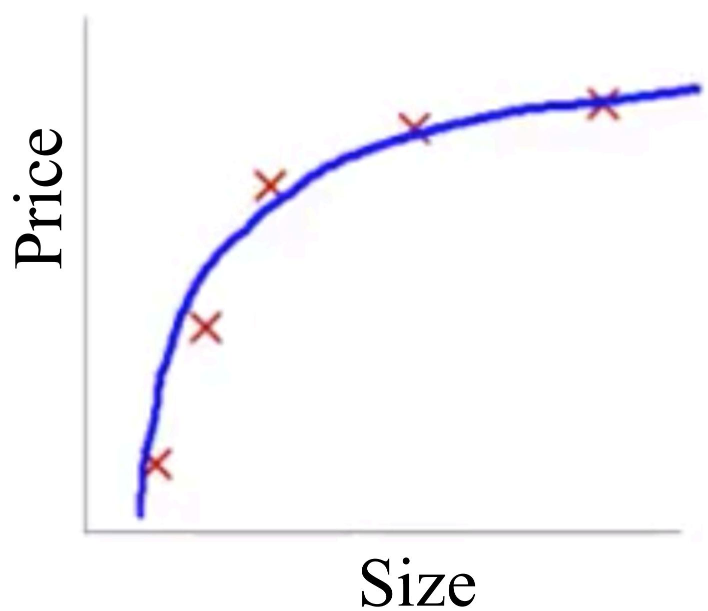


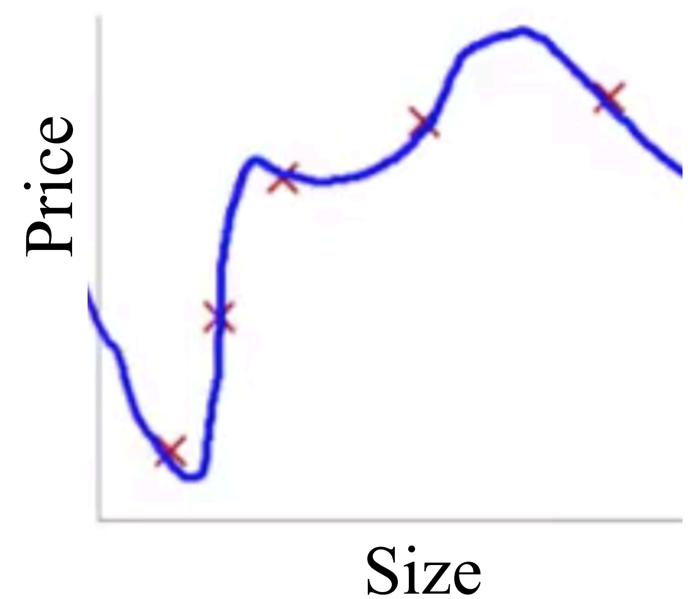
$\theta_0 + \theta_1 x + \theta_2 x^2$



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

Regularization (Intuition)





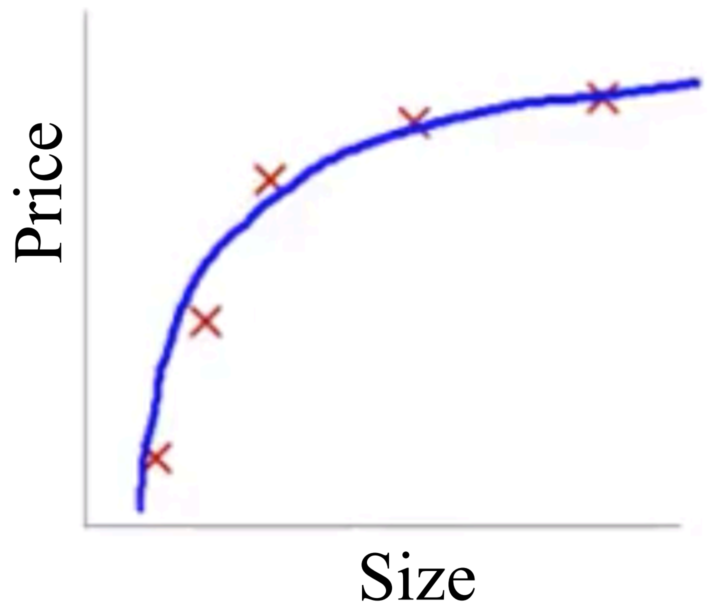
$\theta_0 + \theta_1 x + \theta_2 x^2$

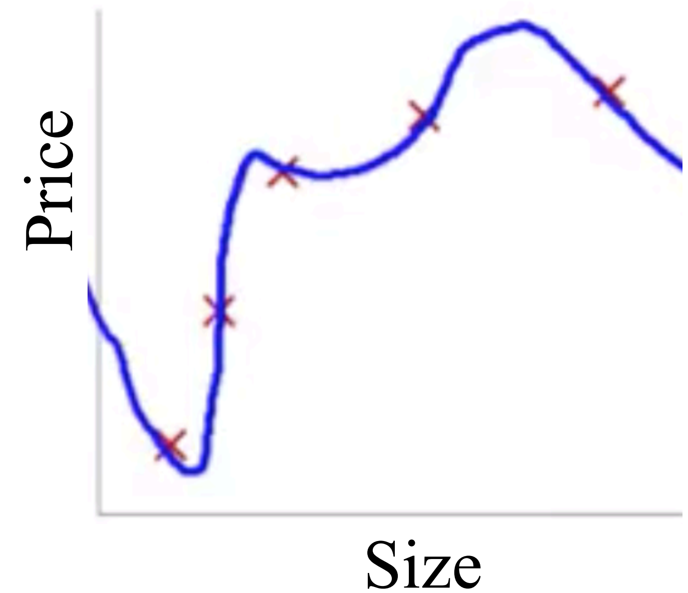
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Suppose we penalize and make θ_3, θ_4 really small.

Optimization objective: $\min_{\theta} \frac{1}{2m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

Regularization (Intuition)





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

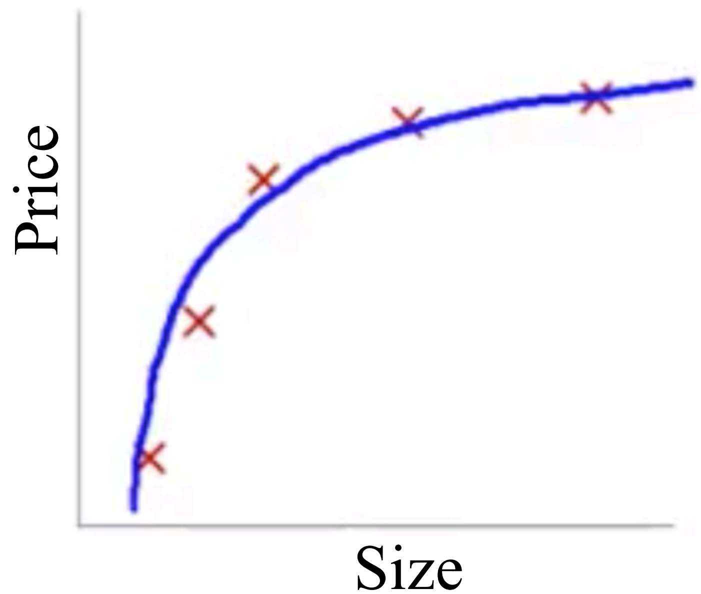
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

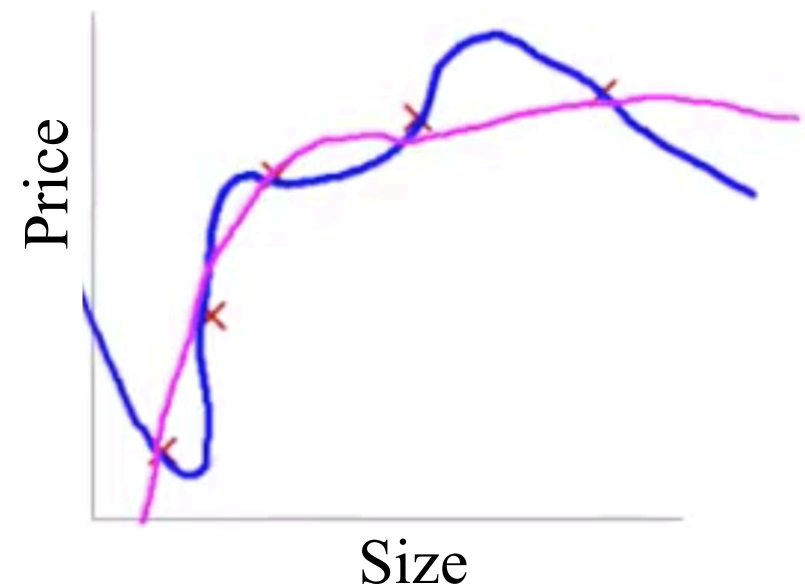
Suppose we penalize and make θ_3, θ_4 really small.

Optimization objective: $\min\limits_{\theta} \dfrac{1}{2m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\theta_3^2 + 1000\theta_4^2$

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

Regularization (Intuition)





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

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

Suppose we penalize and make θ_3, θ_4 really small.

Optimization objective: $\min_\theta \frac{1}{2m} \Sigma_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\theta_3^2 + 1000\theta_4^2$

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

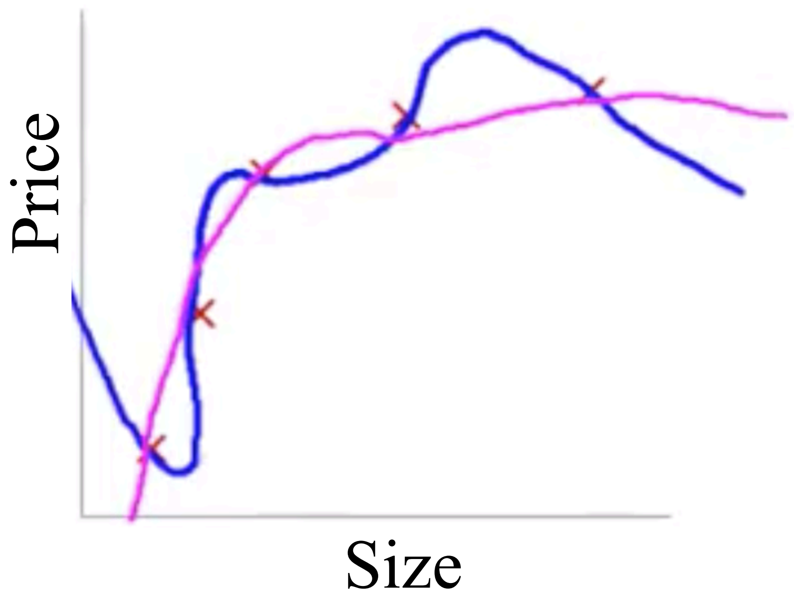
Regularization (Formally)

Small values for parameters $\theta_1, \theta_2, \ldots, \theta_n$

- 'Simpler' hypothesis
- Less prone to overfitting

Housing:

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



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

Cost function (in linear regression) :

$$J(\theta) = \frac{1}{2m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

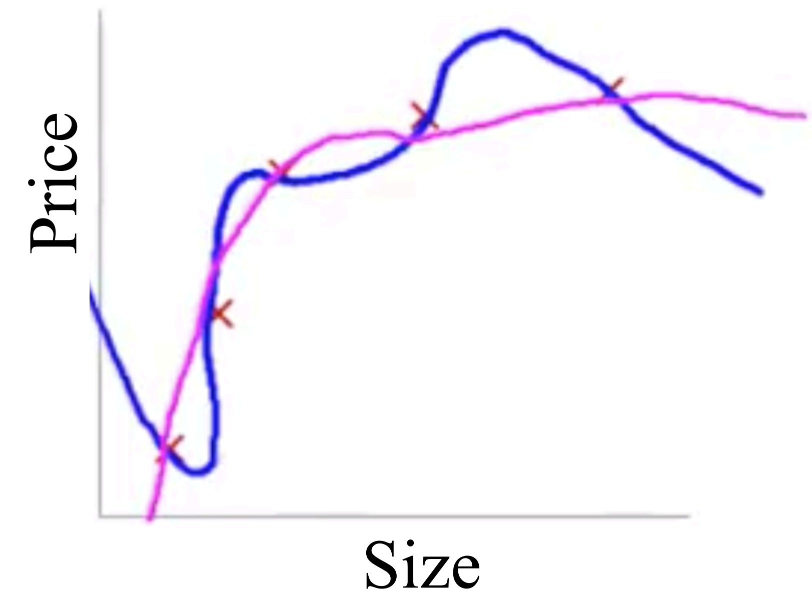
Regularization (Formally)

Small values for parameters $\theta_1, \theta_2, \ldots, \theta_n$

- 'Simpler' hypothesis
- Less prone to overfitting

Housing:

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



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

Cost function (in linear regression) :

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

j=1

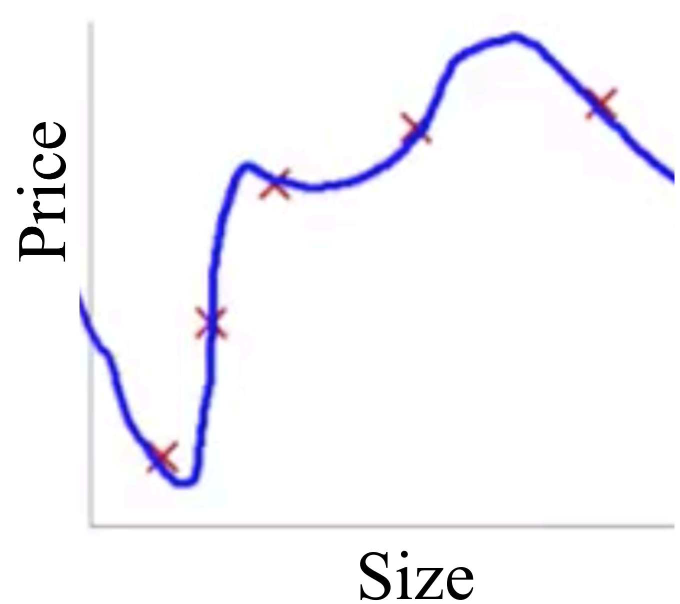
Regularization (Formally)

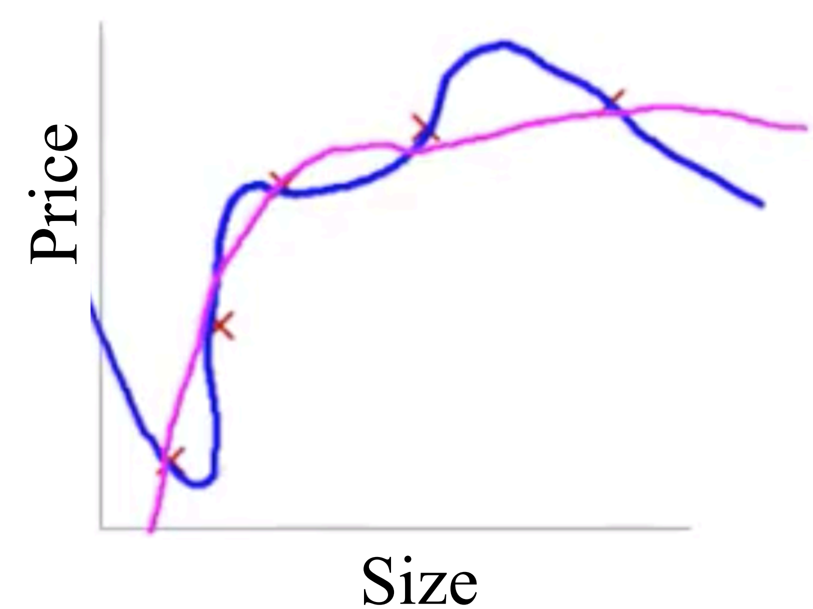
Regularized cost function:

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

regularization parameter

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





Question

In regularized linear regression, we choose θ to minimize:

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

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

(i) Algorithm works fine; setting λ to be very large can't hurt it.
(ii) Algorithm fails to eliminate overfitting.
(iii) Algorithm results is underfitting (fails to fit even the training set).
(iv) Gradient descent will fail to converge.

Regularized Linear Regression

Cost Function (Recap)

Cost function:

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

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

Gradient Descent (Original)

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \Sigma_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \qquad (j = 0,1,2,\ldots,n)$$

}

Gradient Descent for Regularized Linear Regression

Repeat {

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

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

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

}

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

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

Gradient Descent for Regularized Linear Regression

Repeat {

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

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

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

}

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

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

$$\Updownarrow$$

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

Question

- Suppose you are doing gradient descent on a training set of $m > 0$ examples, using a fairly small learning rate $\alpha > 0$ and some regularization parameter $\lambda > 0$. Consider the update rule:

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

Which of the following statements about the term $(1 - \alpha\frac{\lambda}{m})$ must be true?

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

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

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

None of these

Normal Equation (Recap)

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}_{m \times (n+1)} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbb{R}^m$$

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

Normal Equation (Recap)

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}_{m \times (n+1)} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbb{R}^m$$

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

Solution: $\theta = (X^T X \qquad)^{-1} X^T y$

Normal Equation for Regularized Linear Regression

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}_{m \times (n+1)} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \in \mathbb{R}^m$$

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

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

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

Non-invertibility

(#examples) (#features)

- If $m < n$, then $(X^T X)$ is non-invertible / singular.
- If $m = n$, then $(X^T X)$ may be non-invertible.

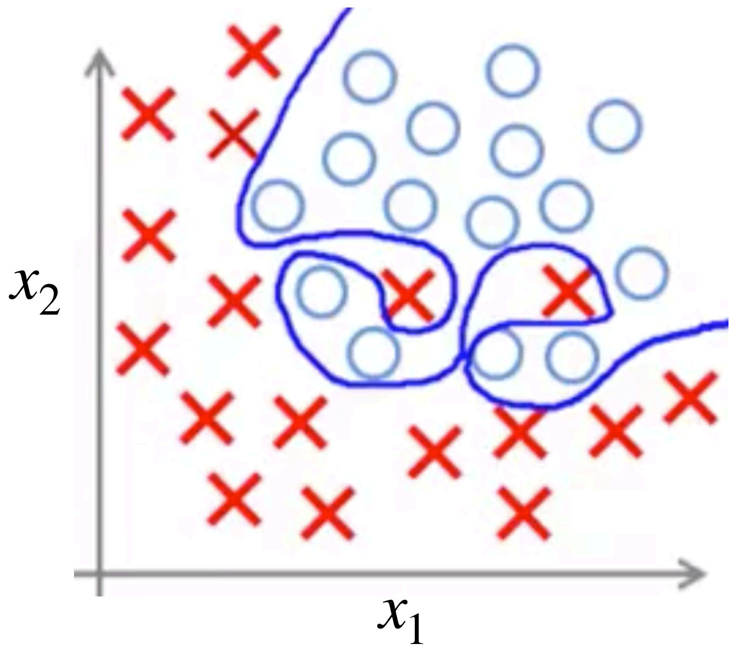
However, using regularization can take care of any non-invertibility issues.

If $\lambda > 0$, then:

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

Regularized Logistic Regression

Regularized Logistic Regression

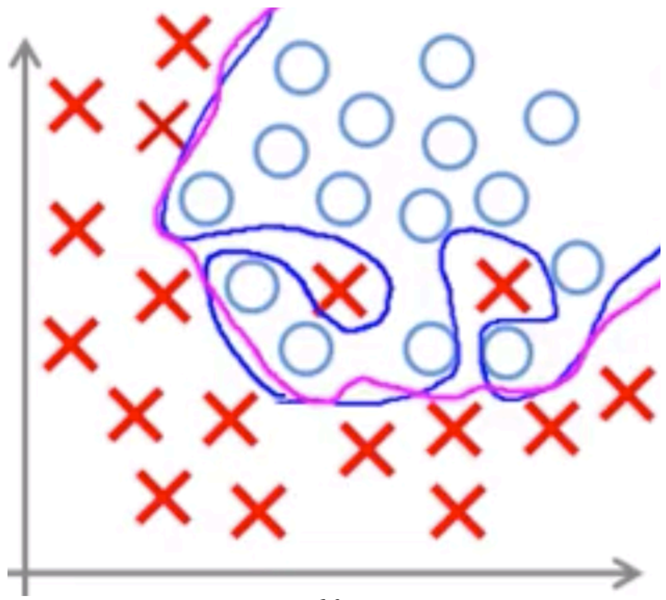


$$
\begin{aligned}
h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 \\
+ \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 \\
+ \theta_6 x_1^3 x_2 + \ldots)
\end{aligned}
$$

Cost function:

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

Regularized Logistic Regression



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \ldots)$$

Cost function:

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

Gradient Descent for Regularized Logistic Regression

Repeat {

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

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

} $\qquad (j = \cancel{0}, 1, 2, \ldots, n)$

This is not exactly the same algorithm as gradient descent for regularized linear regression ! $\because h_\theta(x) = \dfrac{1}{1 + e^{-\theta^T x}}$

Question

- When using regularized logistic regression, which of these is the best way to monitor whether gradient descent is working correctly?

(i) Plot $-[\frac{1}{m}\Sigma_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$ as a function of the number of iterations and make sure it is decreasing.

(ii) Plot $-[\frac{1}{m}\Sigma_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\Sigma_{j=1}^{n}\theta_j^2$ as a function of the number of iterations and make sure it is decreasing.

(iii) Plot $\Sigma_{j=1}^{n}\theta_j^2$ as a function of the number of iterations and make sure it is decreasing.