Regularization and Overfitting; Assessing model effectiveness

Machine Learning:
Jay Urbain, PhD

Credits: Hastie and Tibshirani, Andrew Ng, Stanford;

Data has inherent variance that does not have predictive value

$$
\begin{aligned}
E(Y-\hat{Y})^2 &= E[f(X)+\epsilon-\hat{f}(X)]^2 \\
&= \underbrace{[f(X)-\hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{Irreducible}} \; ,
\end{aligned}
$$

Necessitates the need for training, validation, and test sets.

- Training set – Learn model
- Validation set – tune model
- Test set – evaluate tuned model

Assessing the accuracy of regression model coefficients

Linear regression with residual term. Represents what we can't explain with our model.

$$Y = \beta_0 + \beta_1 X + \epsilon.$$

RSS measures the amount of variability that is left unexplained after performing the regression

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2.$$

TSS (Total sum of squares) measures the total variance when measuring the response y.

$$\mathrm{TSS} = \sum (y_i - \bar{y})^2$$

R^2 amount of variance explained by our model

$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

The RSE is an estimate of the standard deviation of ε. It is basically the average amount that the response will deviate from the true regression line.

$$\mathrm{RSE} = \sqrt{\frac{1}{n-2}\mathrm{RSS}} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}.$$

Assessing the performance of classification

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

Accuracy measures the overall correctness of classification.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Sensitivity (also called the true positive rate, or recall) measures the proportion of positives that are correctly identified as such. E.g., people who have cancer.

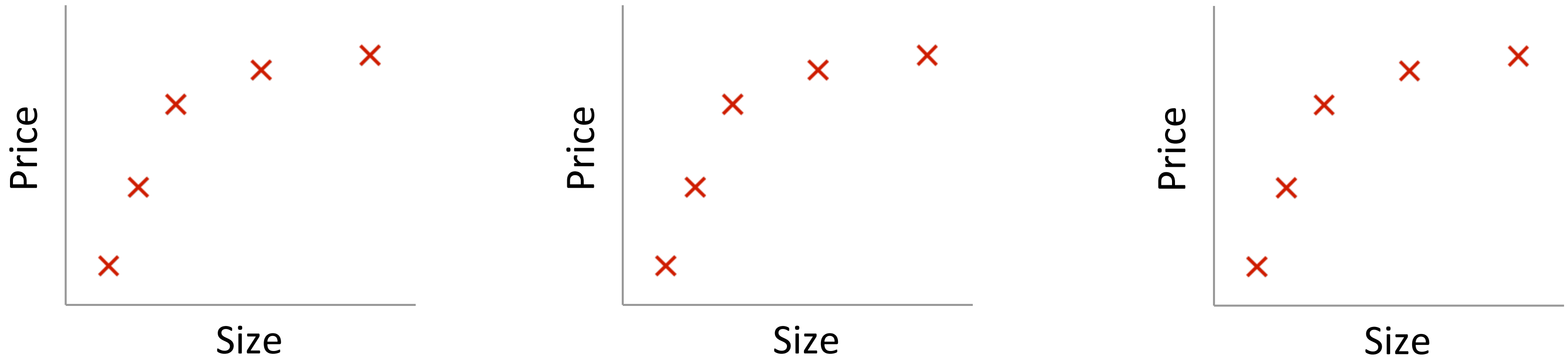
$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

Specificity (also called the true negative rate) measures the proportion of negatives that are correctly identified as such. E.g., people who do not have cancer.

$$\text{Specificity} = \frac{TN}{TN + FP}$$

Precision = TP/(TP + FN)
Recall = TP/(TP + FN)
F1 = 2*Precision*Recall/(Precision + Recall

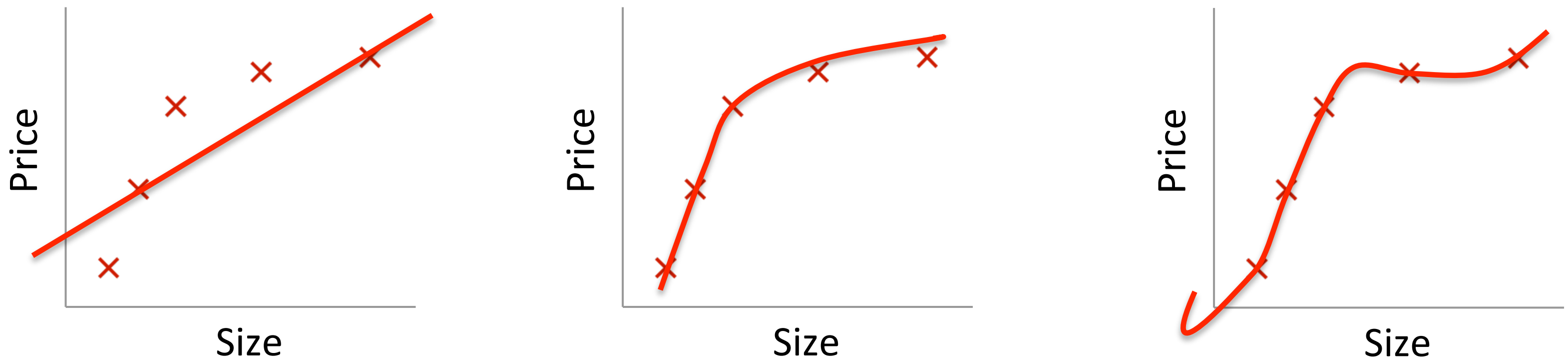
Example: Linear regression (housing prices)



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

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

Example: Linear regression (housing prices)



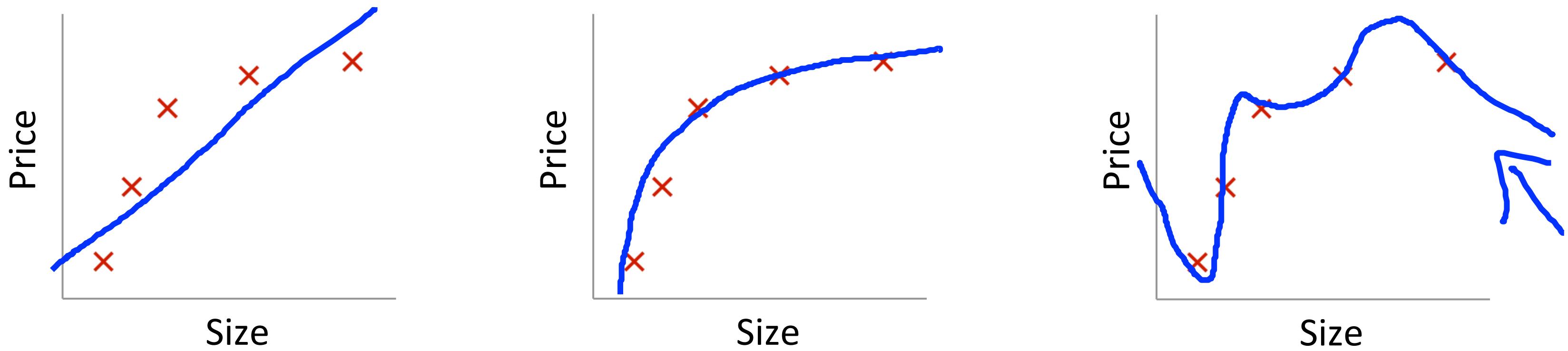
$\theta_0 + \theta_1 x$ | $\theta_0 + \theta_1 x + \theta_2 x^2$ | $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Underfit, high bias

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).

Example: Linear regression (housing prices)



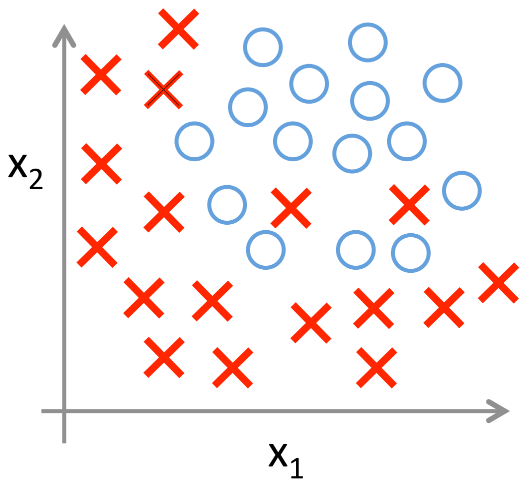
$\rightarrow \theta_0 + \theta_1 x$ "Underfit" "High bias"

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

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

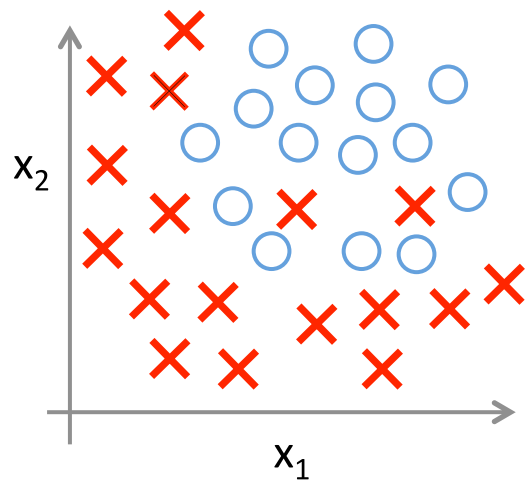
Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Example: Logistic regression

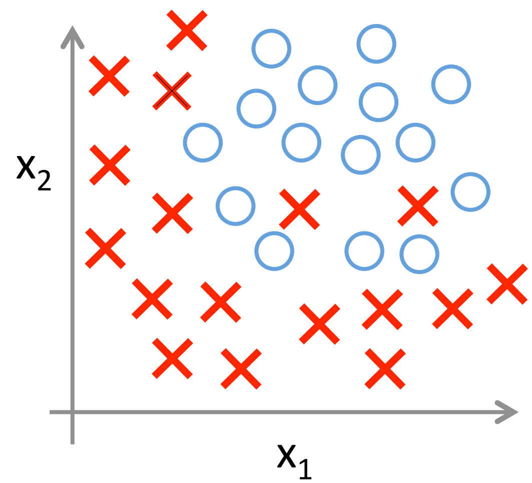


$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

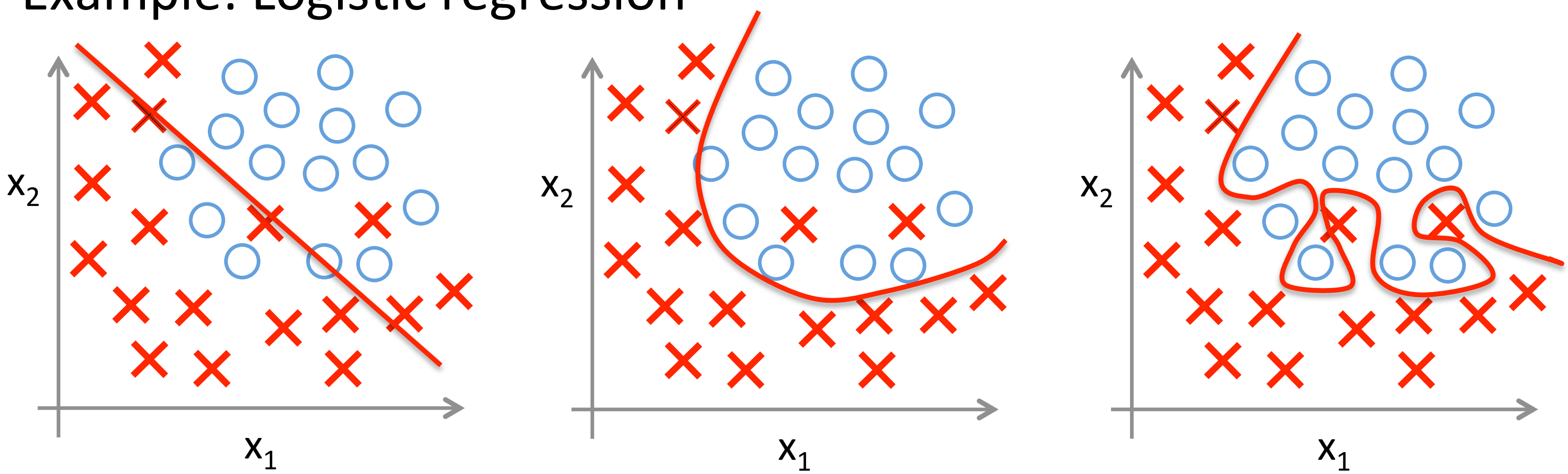


$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 \\ + \theta_3 x_1^2 + \theta_4 x_2^2 \\ + \theta_5 x_1 x_2)$$



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

Example: Logistic regression



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

(g = sigmoid function)

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

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

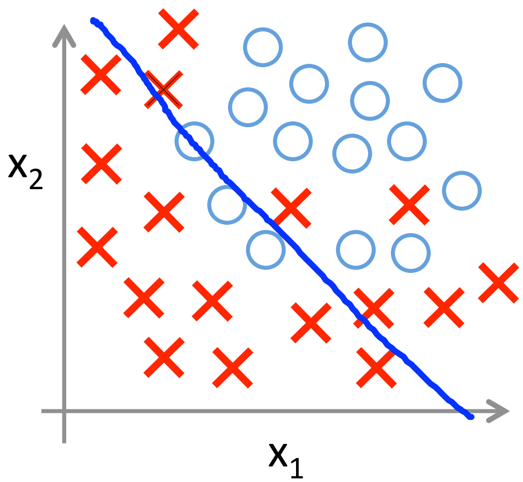
Underfit, high bias

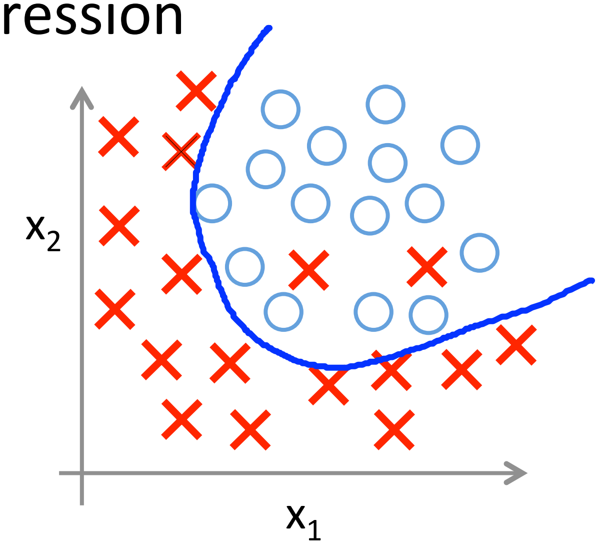
Overfit, high variance

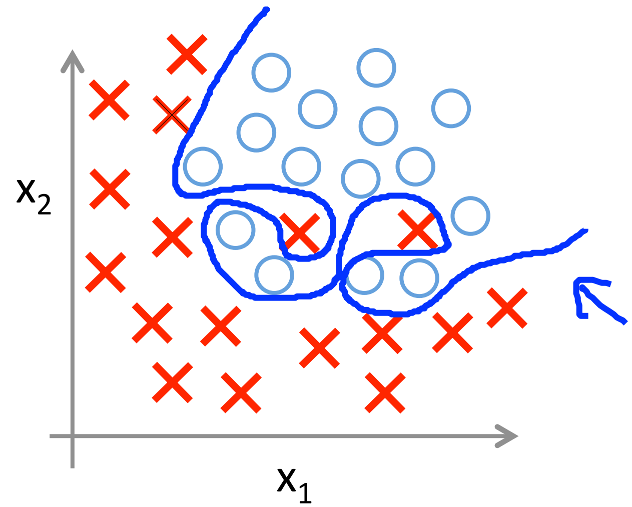
Low model complexity ←

→ High model complexity

Example: Logistic regression







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

(g = sigmoid function)

"Underfit"

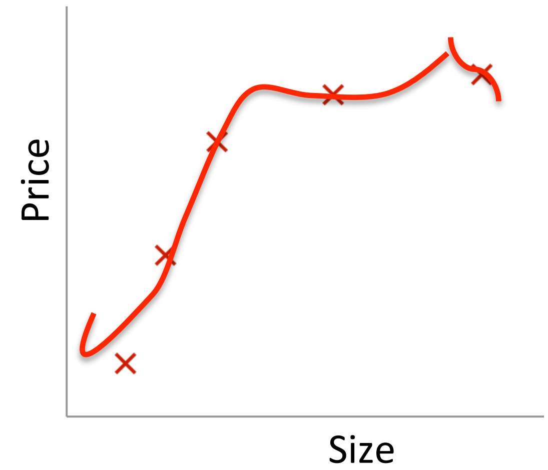
$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$$

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

"Overfit"

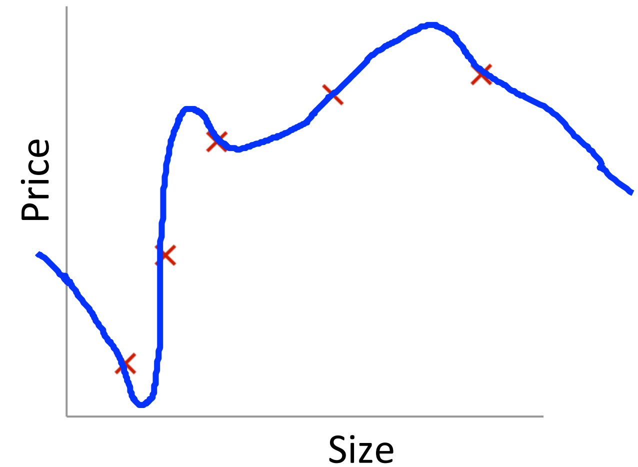
Addressing overfitting:

$x_1 =$ 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:

$x_1 =$ 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}

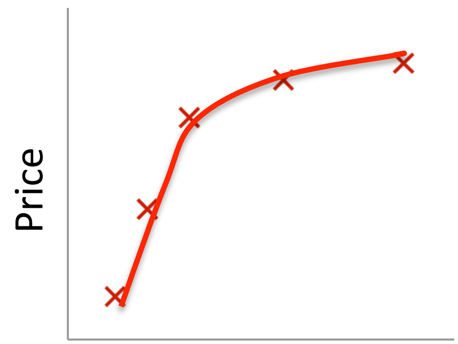


Addressing overfitting:

Options:

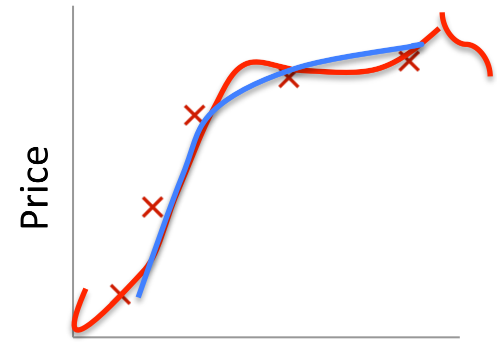
1. Reduce number of features.
   - Manually select which features to keep.
   - Model selection algorithm.
2. Regularization.
   - Keep all the features, but reduce magnitude/values of parameters θ_j.
   - Works well when we have a lot of features, each of which contributes a bit to predicting y.

Intuition



Size of house

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



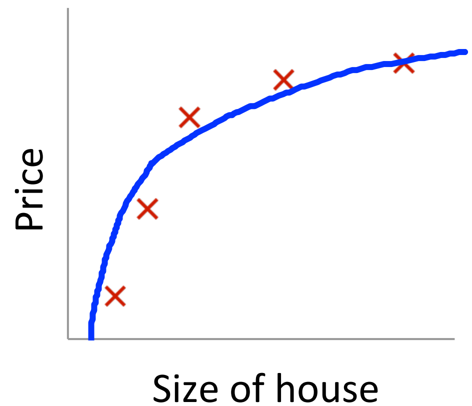
Size of house

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

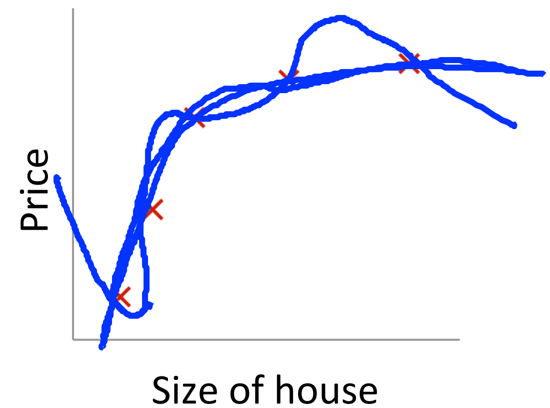
Suppose we penalize and make θ_3, θ_4 really small.

$$\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000 * \Theta_3 + 1000 * \Theta_4$$

Intuition



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



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

Suppose we penalize and make θ_3, θ_4 really small.

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

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

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

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

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

Do not penalize bias term

Regularization.

Small values for parameters $\boxed{\theta_0, \theta_1, \ldots, \theta_n}$

- "Simpler" hypothesis
- Less prone to overfitting

$\boxed{\theta_3, \theta_4} \approx 0$

Housing:

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

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

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

Regularized linear regression.

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

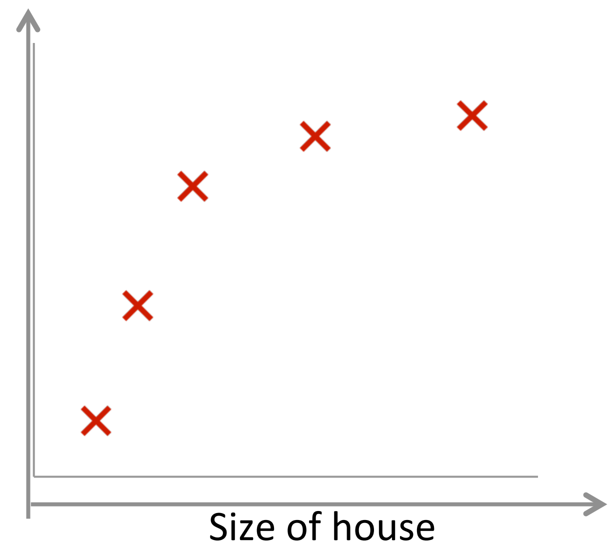
$$\min_\theta J(\theta)$$

Repeat $\{$

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

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

$(1 - \alpha\frac{\lambda}{m})$ Usually < 1

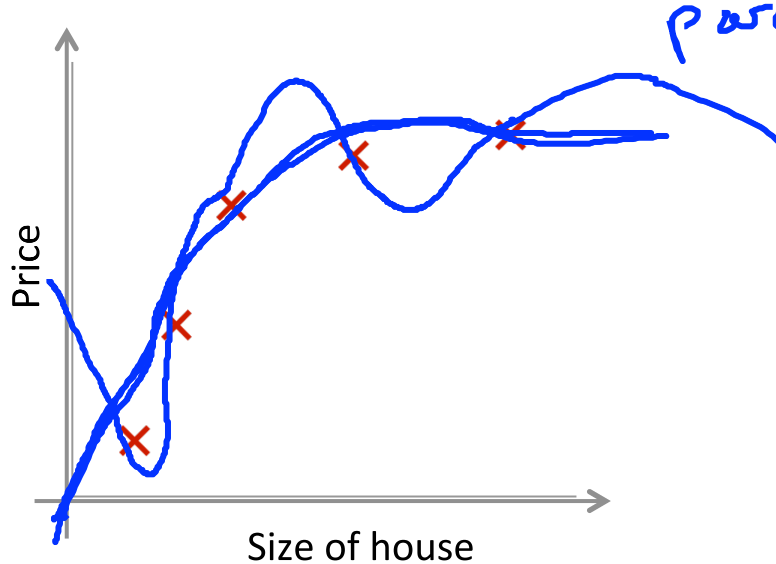


Regularization.

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

regularization parameter

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



In regularized linear regression, we choose θ to minimize

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

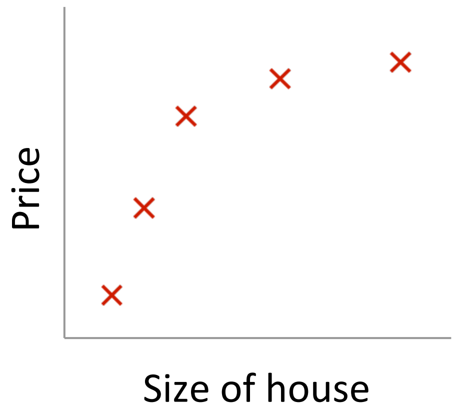
What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

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

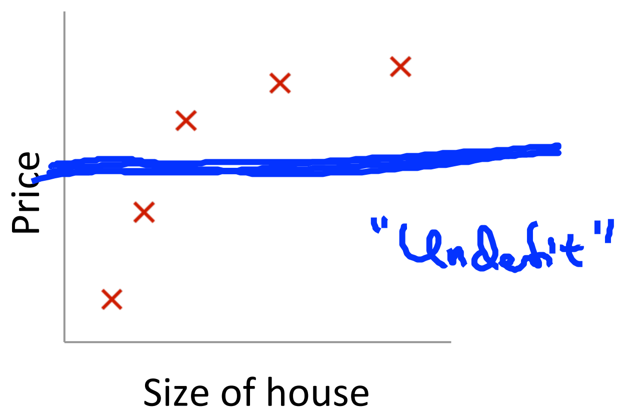


$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

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



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

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

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

$$\boxed{h_\theta(x) = \theta_0}$$

$h_\theta(x)$

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

Gradient descent

Repeat $\{$

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

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

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

$\}$

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

Gradient descent

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

Repeat {

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

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

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

}

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

$1 - \alpha \frac{\lambda}{m} < 1$ $\quad 0.99 \quad \theta_j \times 0.99 \quad \boxed{\theta_j^{2}}$

Non-invertibility (optional/advanced).

Suppose $m \leq n$,

(#examples) (#features)

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

If $\lambda > 0$,

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

Non-invertibility (optional/advanced).

Suppose $m \leq n$,

(#examples) (#features)

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

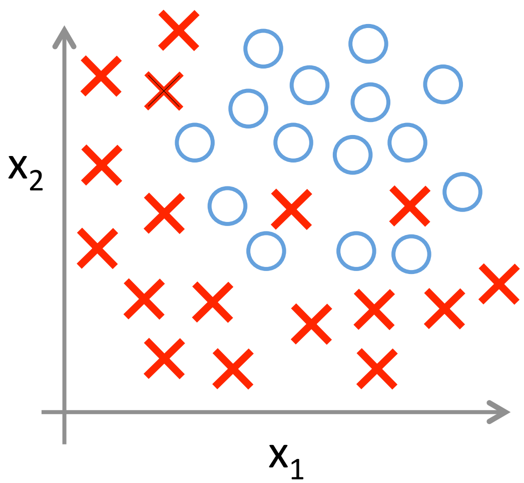
pinv

inv

If $\lambda > 0$,

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

Regularized logistic regression.

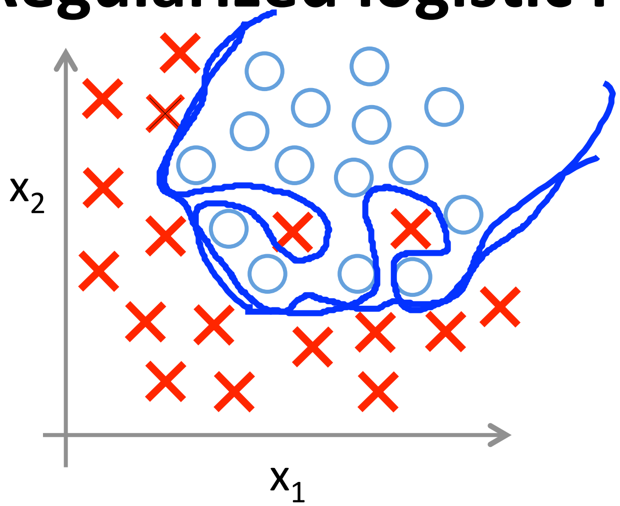


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

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

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:

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

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

Gradient descent

Repeat $\{$

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

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

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

$\}$

Gradient descent

Repeat $\{$

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

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

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

$\theta_1, \ldots, \theta_n$

$\}$

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

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 $\frac{\partial}{\partial\theta_2}J(\theta)$ `];`

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

⋮

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

fminunc(@costFunction...)

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

Advanced optimization

function [jVal, gradient] = costFunction(theta)

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

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

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

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

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

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

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

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

$\vdots$

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

$J(\theta)$