



Deep Learning

Dr. Mehran Safayani

safayani@iut.ac.ir

safayani.iut.ac.ir



<https://www.aparat.com/mehran.safayani>

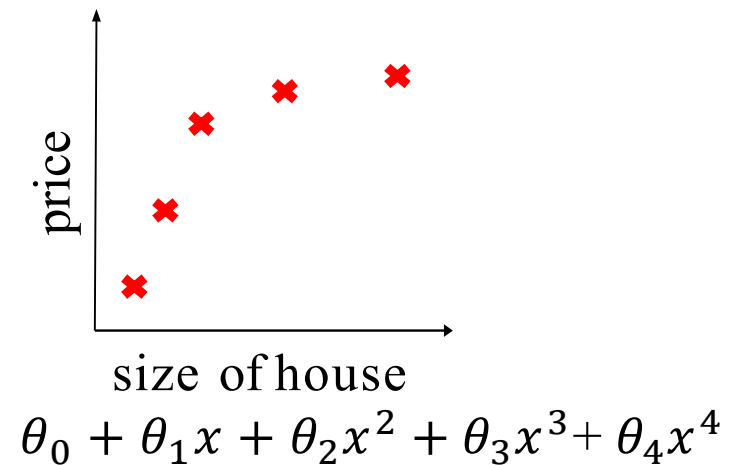
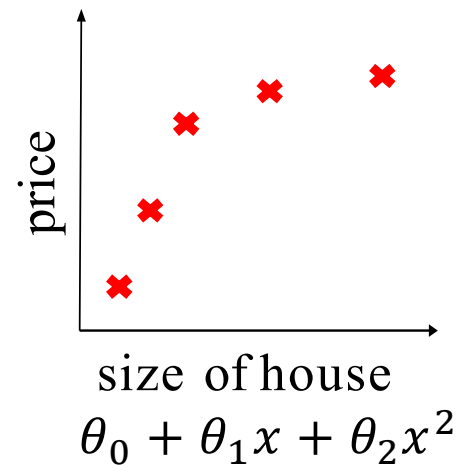
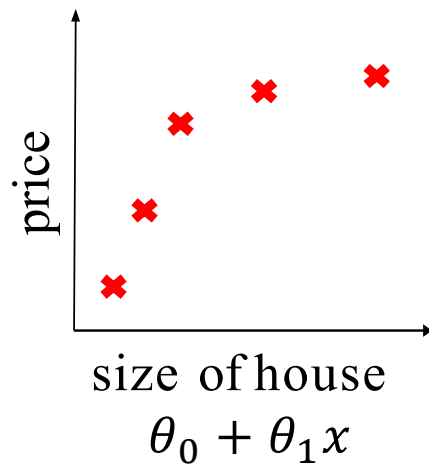


https://github.com/safayani/deep_learning_course



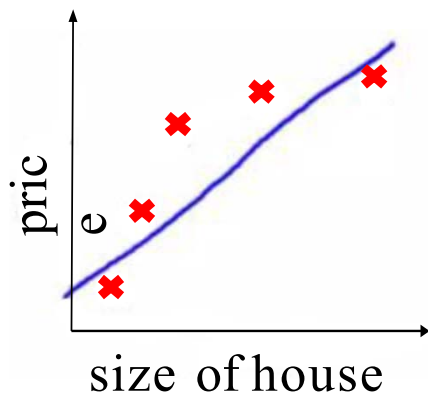
Department of Electrical and computer engineering, Isfahan university of technology, Isfahan, Iran

Example: Linear regression (housing prices)

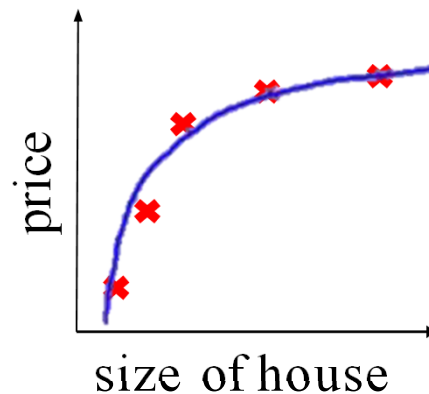


The slides are modified, based on original slides by [Andrew NG, Stanford university]

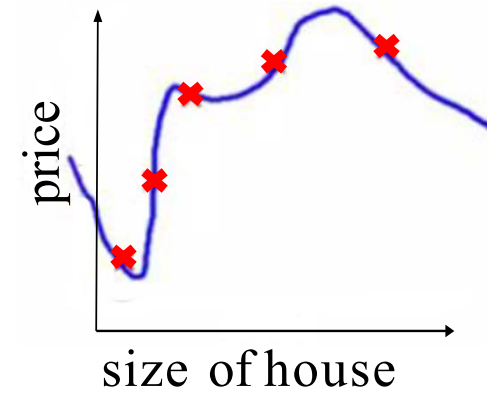
Example: Linear regression (housing prices)



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



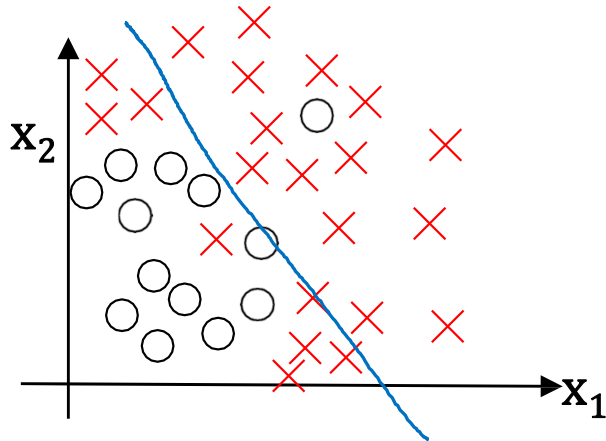
$\theta_0 + \theta_1 x + \theta_2 x^2$
"Just right"



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

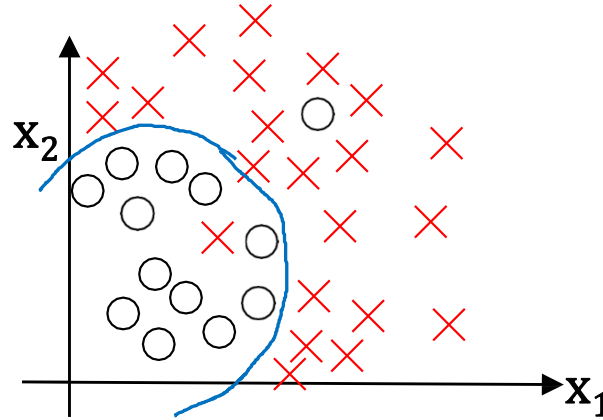
Example: Logistic regression



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

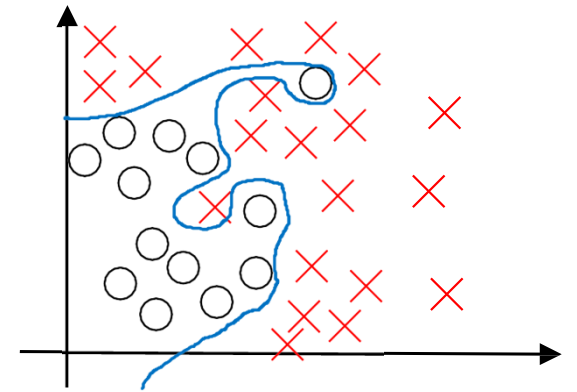
high bias

underfitting



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2) \quad g\left(\begin{array}{l} \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 \\ + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots \end{array}\right)$$

“just right”



high variance

overfitting

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

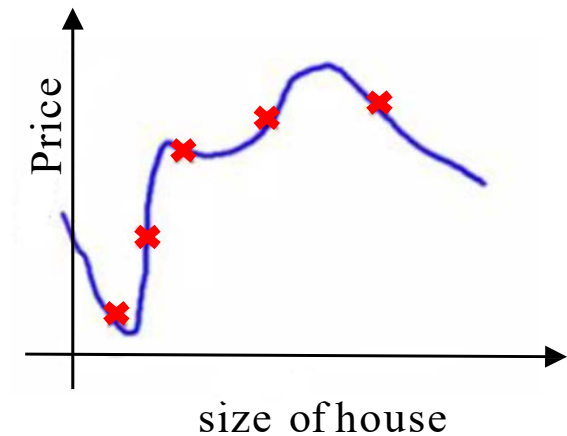
x_6 = kitchen size

.

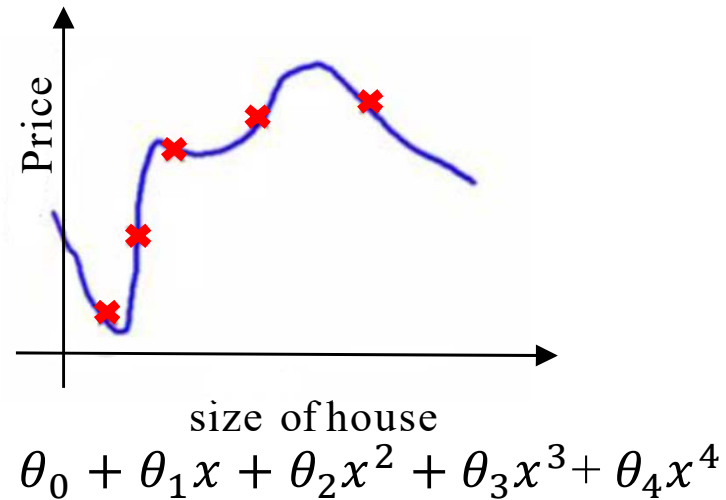
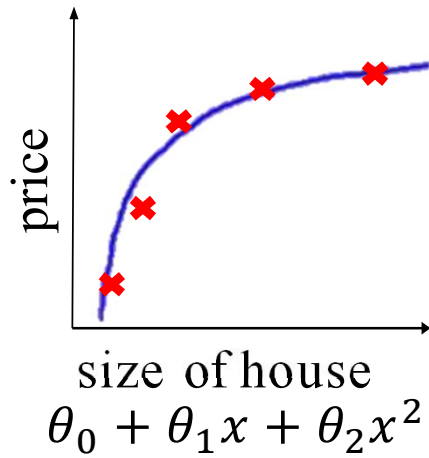
.

.

x_{100}



Intuition



- Suppose we penalize and make θ_3, θ_4 really small.

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \underbrace{1000\theta_3^2}_{\theta_3 \approx 0} + \underbrace{1000\theta_4^2}_{\theta_4 \approx 0}$$

Regularization

- Small values for parameters $\theta_0, \theta_1, \dots, \theta_n$
 - "Simpler" hypothesis
 - Less prone to overfitting
- Housing:
 - Features: x_1, x_2, \dots, x_{100}
 - Parameters: $\theta_0, \theta_1, \theta_2, \dots, \theta_{100}$

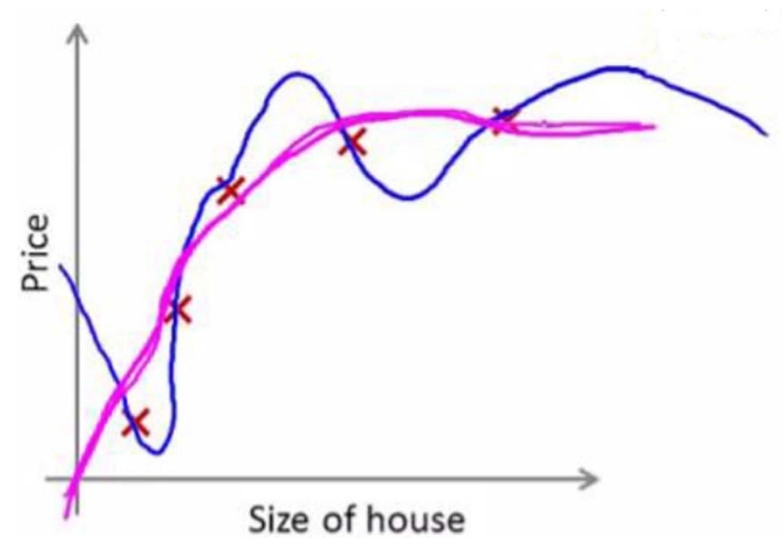
$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

$\theta_0, \theta_1, \theta_2, \dots, \theta_{100}$

λ = regularization parameter
lambda lambda

Regularization

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

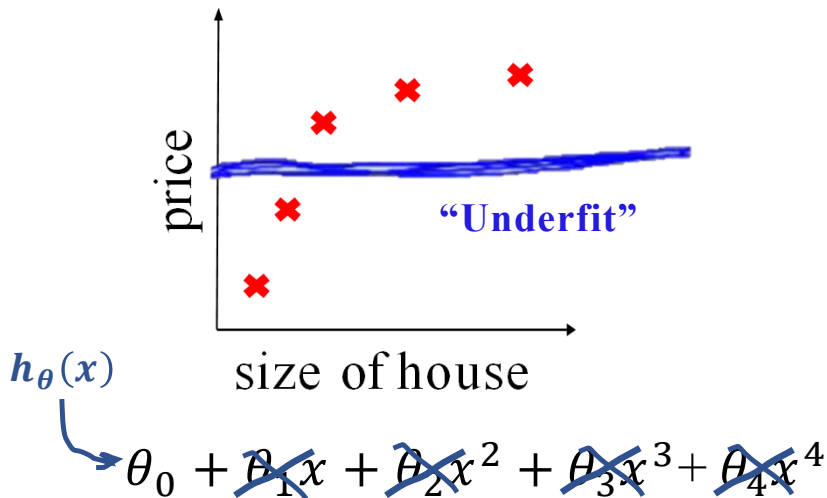


Regularization

- 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 far too large for our problem, say $\lambda = 1010$)?



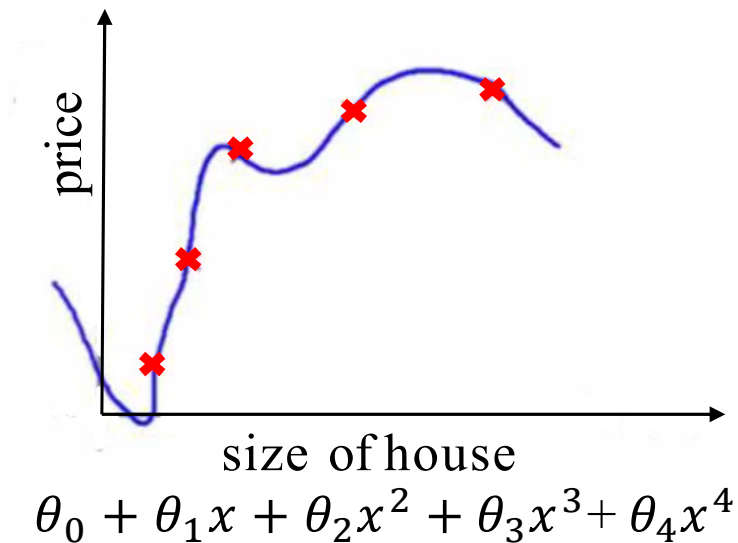
$$\theta_1, \theta_2, \theta_3, \theta_4$$

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

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

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

Evaluating your hypothesis



- Fails to generalize to new examples not in training set.

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}

Evaluating your hypothesis

- Dataset:

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

60%
Training set

20%
Cross validation set (cv)

20%
Test set

 $(x^{(1)}, y^{(1)})$ $(x^{(2)}, y^{(2)})$ \vdots \vdots \vdots $(x^{(m)}, y^{(m)})$ $(x_{cv}^{(1)}, y_{cv}^{(1)})$ $(x_{cv}^{(2)}, y_{cv}^{(2)})$ \vdots \vdots \vdots $(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$ $(x_{test}^{(1)}, y_{test}^{(1)})$ $(x_{test}^{(2)}, y_{test}^{(2)})$ \vdots \vdots \vdots $(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

Train/validation/test error

- Training error:

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

- Cross Validation error:

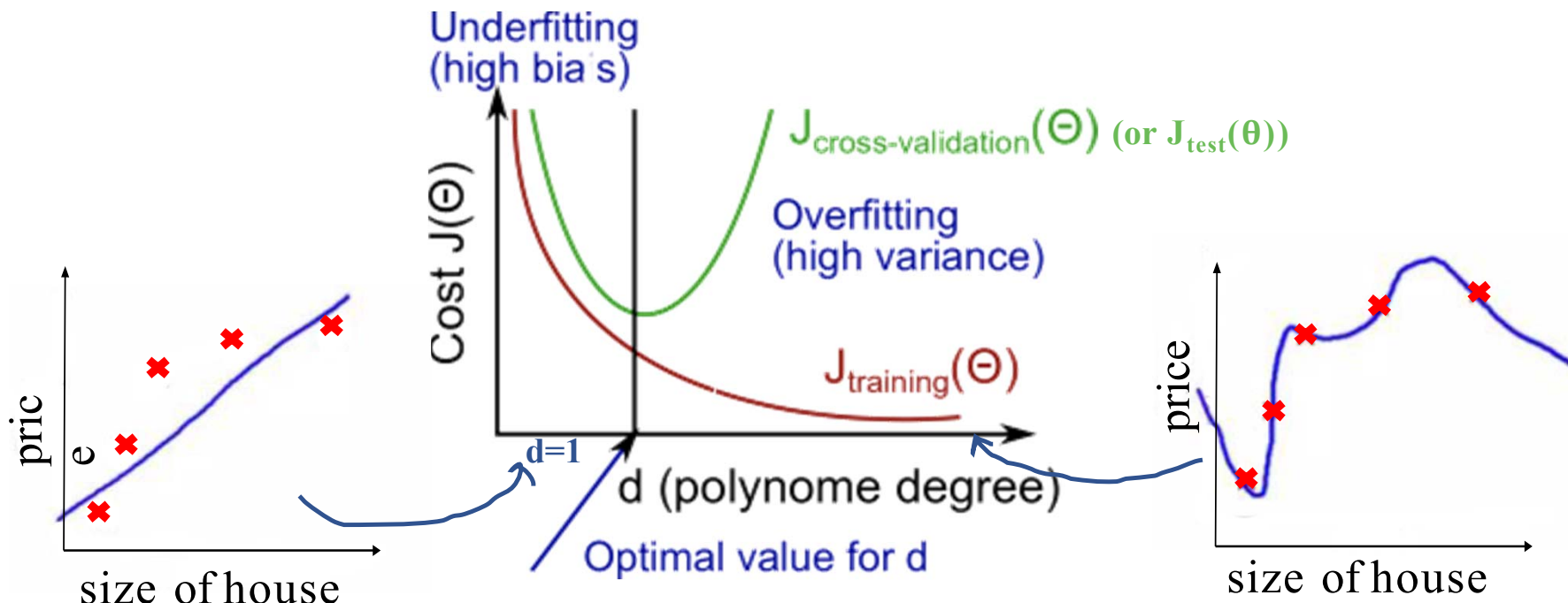
$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

- Test error:

$$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_{\theta}(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

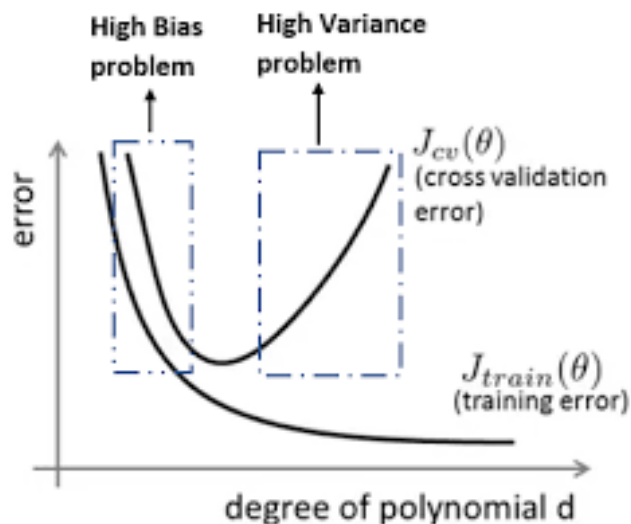
Bias/variance

- Training error: $J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$
- Cross validation error: $J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$ (or $J_{test}(\theta)$)



Diagnosing bias vs. variance

- Suppose your learning algorithm is performing less well than you were hoping. ($J_{cv}(\theta)$ or $J_{test}(\theta)$) is high.) Is it a bias problem or a variance problem?



<https://towardsdatascience.com/>

Bias (underfit):

$J_{train}(\theta)$ will be high

$$J_{cv}(\theta) \approx J_{train}(\theta)$$

Variance (overfit):

$J_{train}(\theta)$ will be low

$$J_{cv}(\theta) \gg J_{train}(\theta)$$

Linear regression with regularization

Model: $h_{\theta}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

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

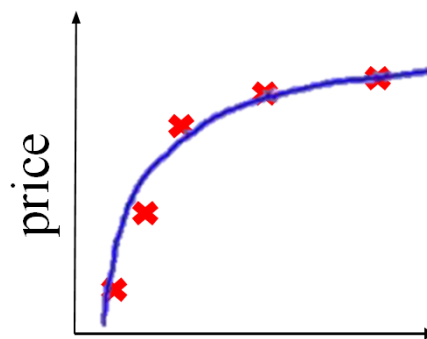


size of house

Large λ

High bias (underfit)

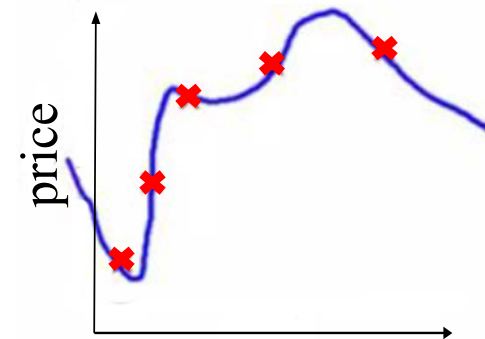
$$\lambda = 10000. \theta_1 \approx 0, \theta_2 \approx 0, \dots$$
$$h_{\theta}(x) \approx \theta_0$$



size of house

Intermediate λ

"Just right"



size of house

Small λ

High variance (overfit)

$$\lambda = 0$$

Choosing the regularization parameter λ

Model: $h_{\theta}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

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

1. Try $\lambda = 0 \rightarrow \min J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{cv}(\theta^{(1)})$
2. Try $\lambda = 0.01 \rightarrow \min J(\theta) \rightarrow \theta^{(2)} \rightarrow J_{cv}(\theta^{(2)})$
3. Try $\lambda = 0.02 \rightarrow \min J(\theta) \rightarrow \theta^{(3)} \rightarrow J_{cv}(\theta^{(3)})$
4. Try $\lambda = 0.04$
5. Try $\lambda = 0.08 \rightarrow \min J(\theta) \rightarrow \theta^{(5)} \rightarrow J_{cv}(\theta^{(5)})$
-
-
-
12. Try $\lambda = 10$ $\rightarrow \min J(\theta) \rightarrow \theta^{(12)} \rightarrow J_{cv}(\theta^{(12)})$
10.24 Pick (say) $\theta^{(5)}$. Test error: $J_{test}(\theta^{(5)})$

Bias/variance as a function of the regularization parameter λ

- **Training error:**

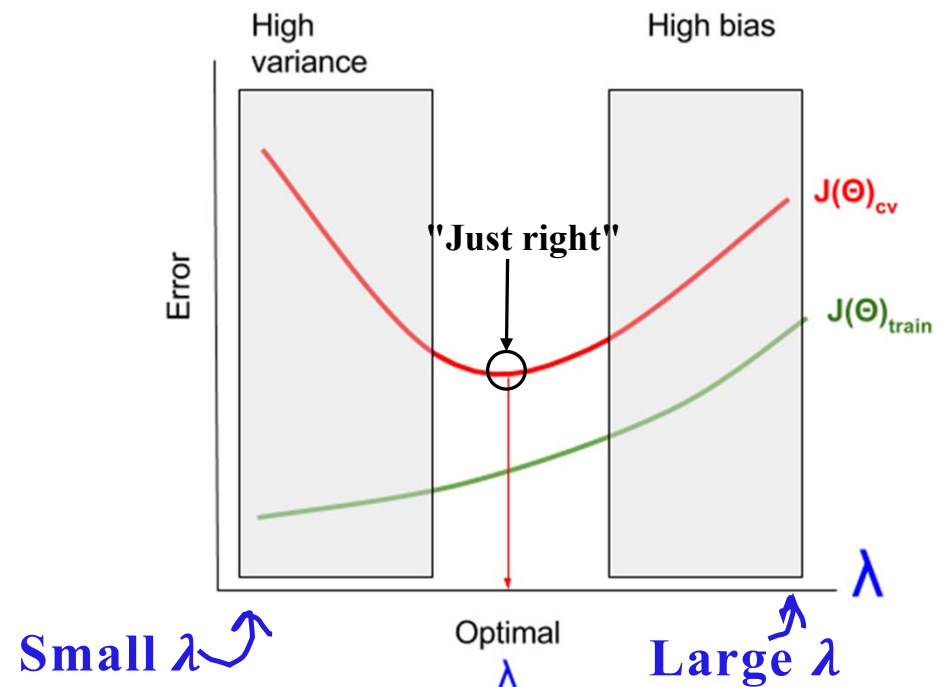
$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2$$

- **Cross Validation error:**

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

- **Test error:**

$$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_{\theta}(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

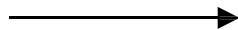


Regularization

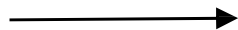
Bias/variance

- High bias \rightarrow Bigger network
- High variance \rightarrow Regularization
More data

Data augmentation



4



4

4

4

Regularization

- Logistic regression

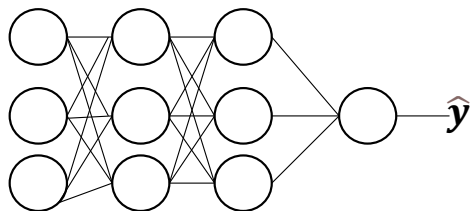
$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \|w\|_2^2$$

$$L2: \|w\|_2^2 = \sum_{j=1}^{n_x} w_j^2 = w^T w$$

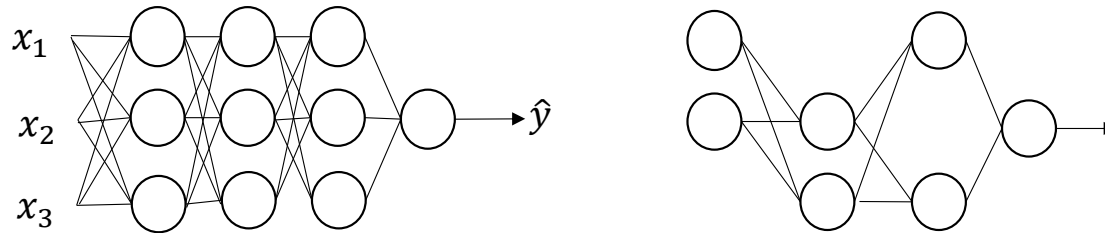
$$L1: \|w\|_1 = \sum_{j=1}^{n_x} |w_j|$$

Neural Network

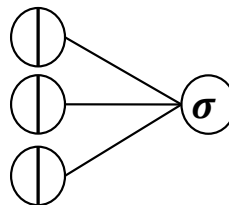
- $J(w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}, \dots, w^{[L]}, b^{[L]}) =$
- $J(w, x) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) + \underbrace{\frac{\lambda}{2m} \sum_{l=1}^L \|w^{[l]}\|^2}_E$
- $dw^{[l]} = (\text{from backprop}) + \frac{\lambda}{m} w^{[l]}$
- $w^{[l]} = w^{[l]} - \alpha dw^{[l]}$



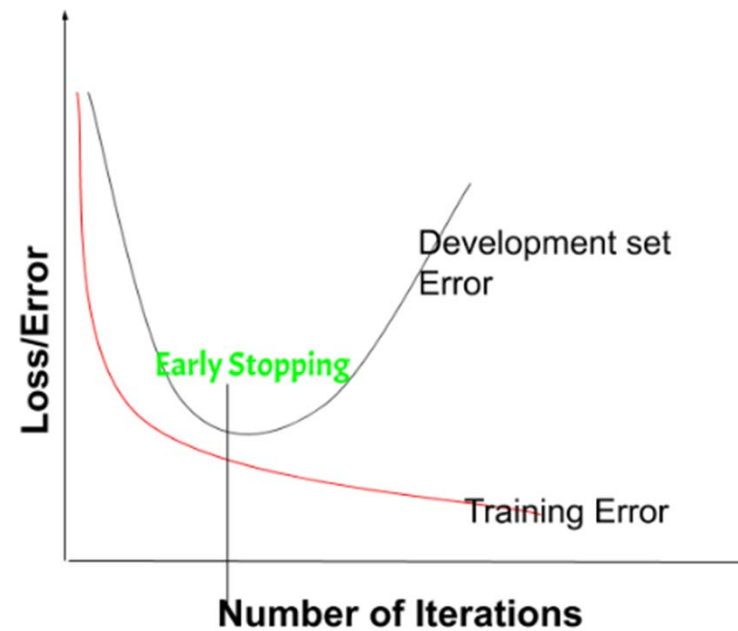
Dropout Regularization



- $d_3 = np.random.rand(a3.shape[0], a3.shape[1]) < keep_prob = 0.8$
- $a_3 = np.multiply(a_3, d_3)$
- $a_3 \neq keep_prob$



Early stopping



<https://www.geeksforgeeks.org/regularization-by-early-stopping/>

Old way of splitting data

- Deep learning

train			dev	test
60%	20%	20%		
1000,000 data			1%	10,000
98%	1%	1%		
99.5%	0.4%	0.1%		

- K-fold cv

