10. Advice for applying machine learning

Debugging learning algorithms

Suppose you have implemented regularized linear regression to predict housing prices:

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

However you find out that your hypothesis on a new set of houses makes unacceptable large Errors on predictions.

What do you try next?

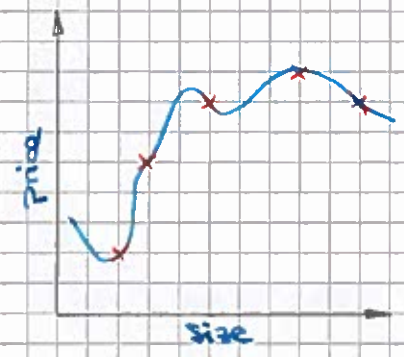
- get more training examples
- try smaller sets of features
- try getting additional features
- try add polynomial features
- try increasing / decreasing λ

} might lead to long Projects if you take the wrong road

Machine learning diagnostics:

Diagnostic: A ~~test~~ test that you can run to gain insight what is/isn't working with a learning algorithm, and gain guidance as to how best improve its performance.

Diagnostic can take time to implement, but doing so can be very good use of time.



Fails to generalize to new examples not in training set.

features

x_1 = size of house

⋮ = ⋮

x_{100} = kitchen size

} is hard or impossible to plot the hypothesis

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

Data set:

	Size	Price	
70%	2104	400	Training set
	⋮	⋮	
	1745	340	
30%	1375	212	Test set
	⋮	⋮	
	2473	365	

Training set $\Rightarrow$ $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ⋮

Test set $\Rightarrow$ $(x_{test}^{(1)}, y_{test}^{(1)})$, ⋮, $(x_{test}^{(n)}, y_{test}^{(n)})$

m_{test} = # of test example

$(x_{test}^{(1)}, y_{test}^{(1)})$

Training / testing procedure for linear regression

- Learn parameter θ from training data (minimizing training error $J(\theta)$) — 70%
- Compute test set error:

$$J_{test}(\theta) = \frac{1}{2m_{test}} \cdot \sum_{i=1}^{m_{test}} \underbrace{\left(h_\theta(x_{test}^{(i)}) - y_{test}^{(i)}\right)^2}_{\text{measure test set error}}$$

$\Rightarrow$ Randomly choose the training set (70%) and test set (30%)

Training / testing procedure for logistic regression

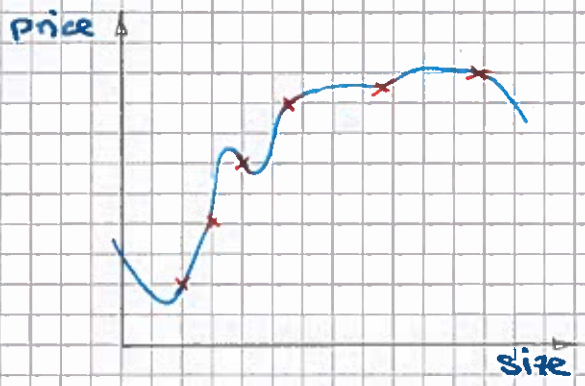
- Learn parameter θ from training data
- Compute test set error

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

- misclassification error (0/1 misclassification error)

$$err(h_\theta(x), y) = \begin{cases} 1 & \text{if } h_\theta(x) \geq 0.5, y = 0 \\ & \text{or if } h_\theta(x) < 0.5, y = 1 \\ 0 & \text{otherwise} \end{cases} \quad \} \text{ error}$$

$$\text{Test error} = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h_\theta(x_{test}^{(i)}), y_{test}^{(i)})$$



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

Once Parameter $\theta_0, \theta_1, \dots \theta_4$ were fit to some set of data (training set), the error of the parameters as measured on the data (training error $J(\theta)$) is likely to be lower than the actual generalization error.

Model selection:

d = degree of polynomio

1. $h_\theta(x) = \theta_0 + \theta_1 x$ → $\theta^{(1)}$ by fitting → $J_{Test}(\theta^{(1)})$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ → $\theta^{(2)}$ → $J_{Test}(\theta^{(2)})$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$ → $\theta^{(3)}$ → $J_{Test}(\theta^{(3)})$

⋮

10. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_{10} x^{10}$ → $\theta^{(10)}$ → $J_{Test}(\theta^{(10)})$

Choose (e.g.) $\theta_0 + \dots + \theta_5 x^5$

How well does the model generalize? Report test set error $J_{Test}(\theta^{(5)})$

Problem:

$J(\theta^{(5)})$ is likely to be an optimistic estimate of generalization error. I.e. our extra parameter (d = degree of polynomial) is fit to the test set.

Fitting $\theta_0, \theta_1, \theta_2$ does not give a adequat answer how good a prediction is. → How good it is generalizing

Evaluating your test set

Data set:

	Size	price	
60%	2104	400	Training set
	1600	330	
	2400	369	
	1416	232	
	3000	540	
	1985	300	
20%	1534	315	Cross validation set (CV)
	1427	199	
20%	1380	212	test set
	1494	243	

$(x^{(1)}, y^{(1)})$
$(x^{(2)}, y^{(2)})$
$\vdots$
$(x^{(m)}, y^{(m)})$

$(x_{cv}^{(1)}, y_{cv}^{(1)})$ m_{cv} = # of cross val. example
$(x_{cv}^{(2)}, y_{cv}^{(2)})$
$\vdots$
$(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$

$(x_{test}^{(1)}, y_{test}^{(1)})$ m_{test} = # of test example.
$(x_{test}^{(2)}, y_{test}^{(2)})$
$\vdots$
$(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

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:

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

Test error:

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

Model selection:

1. $h_\theta(x) = \theta_0 + \theta_1 x$ $\longrightarrow \min_\theta J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{CV}(\theta^{(1)})$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ $\longrightarrow \theta^{(2)} \rightarrow J_{CV}(\theta^{(2)})$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$ $\theta^{(3)} \rightarrow J_{CV}(\theta^{(3)})$

$\Rightarrow$ $\theta^{(4)} \rightarrow J_{CV}(\theta^{(4)})$ best fit

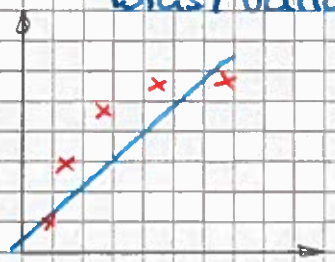
$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_{10} x^{10}$ $\theta^{(10)} \rightarrow J_{CV}(\theta^{(10)})$

Pick $\theta_0 + \theta_1 x + \dots + \theta_4 x^4$

Estimate generalization error for test set $J_{test}(\theta^{(4)})$

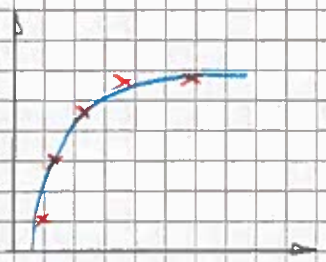
Bias / Variance



$\Theta_0 + \Theta_1 x$

high bias (underfit)

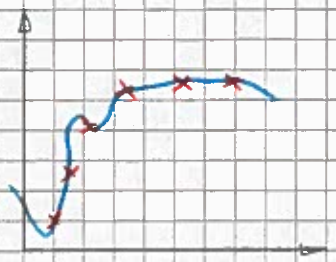
$d = 1$



$\Theta_0 + \Theta_1 x + \Theta_2 x^2$

Just right

$d = 2$



$\Theta_0 + \Theta_1 x + \Theta_2 x^2 + \Theta_3 x^3 + \Theta_4 x^4$

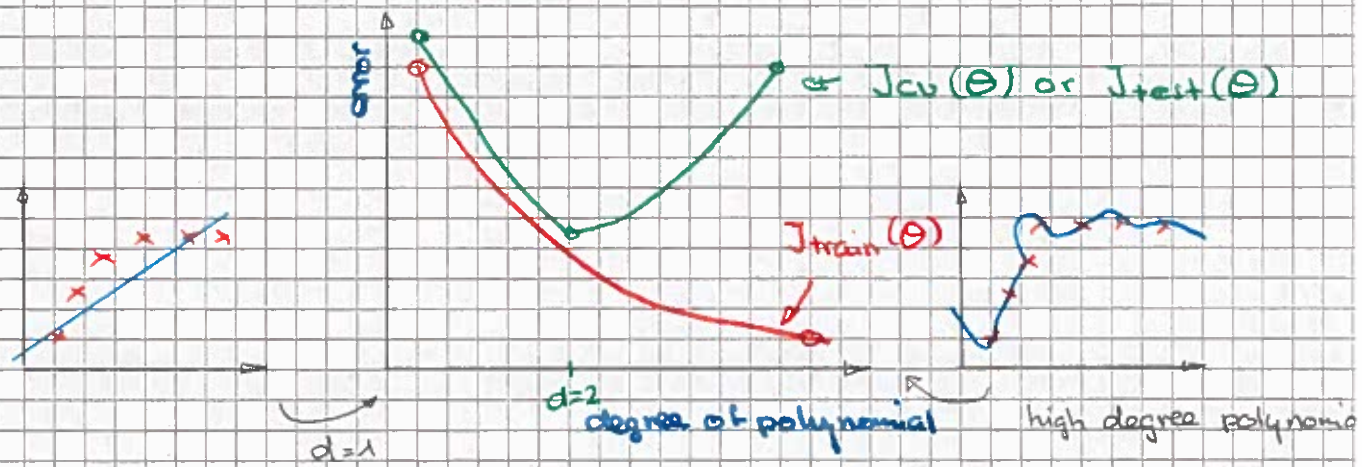
High variance (overfit)

$d = 4$

Bias / Variance

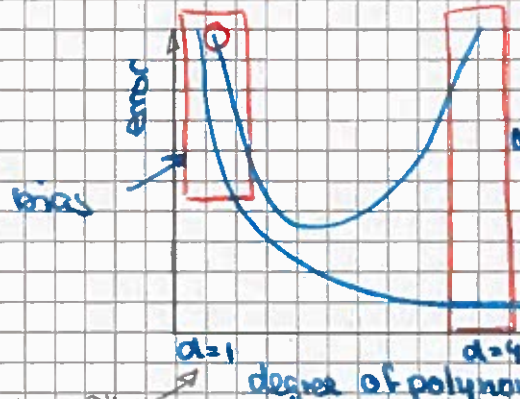
Training error: $J_{train}(\Theta) = \frac{1}{2m} \cdot \sum_{i=1}^{m} (h_\Theta(x^{(i)}) - y^{(i)})^2$

Cross validation error: $J_{cv}(\Theta) = \frac{1}{2m_{cv}} \cdot \sum_{i=1}^{m_{cv}} (h_\Theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$ 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 or variance problem?



Bias (underfit)

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

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

Variance (overfit)

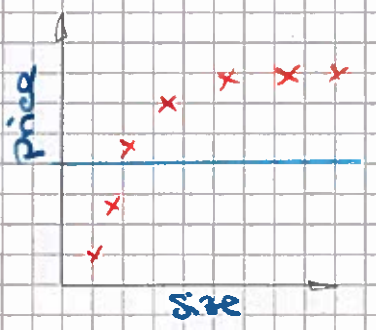
$J_{train}(\Theta)$ will be low

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

Linear Regression with regularization:

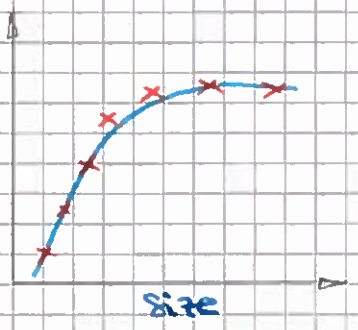
Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \underbrace{\frac{\lambda}{2m} \cdot \sum_{j=1}^{m} \theta_j^2}_{\text{prevent from overfitting}}$$

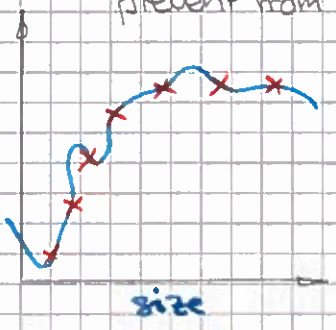


Large λ
High bias (underfit)

$\lambda = 10'000$ $\theta_1 \approx 0, \theta_2 \approx 0$
$h_\theta(x) \approx \theta_0$



intermediate λ
"Just right"



Small λ
High variance (overfit)

$\lambda = 0$ => no regularizatio

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

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

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

J_{train} average
J_{cv} squared
J_{test} error
without regularization term

1. try $\lambda = 0$ → $\min_\theta J(\theta)$ → $\theta^{(1)}$ → $J_{cv}(\theta^{(1)})$
2. $\lambda = 0.01$ → $\theta^{(2)}$ → $J_{cv}(\theta^{(2)})$
3. $\lambda = 0.02$ → $\theta^{(3)}$
4. $\lambda = 0.04$
5. $\lambda = 0.08$
- ⋮
10. $\lambda = 10.24$ → $\theta^{(10)}$ → $J_{cv}(\theta^{(10)})$

pick the one with lowest error → e.g. $\theta^{(5)}$ → $J_{cv}(\theta^{(5)})$

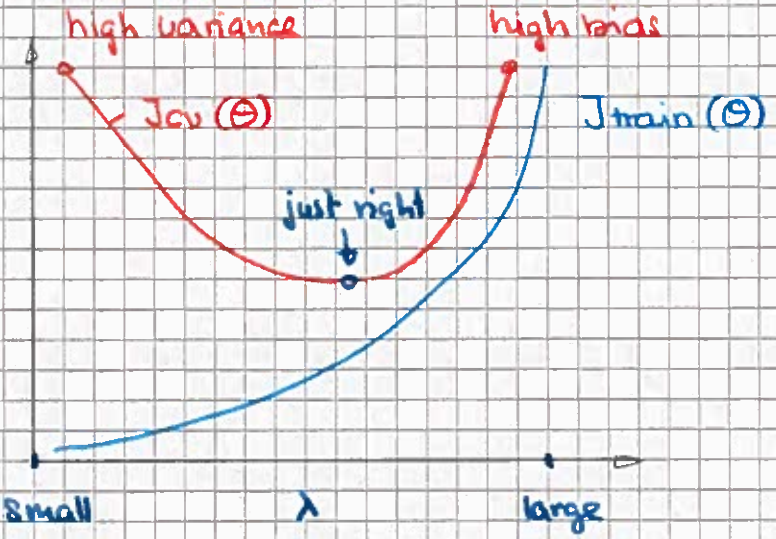
Pick (let's say) $\theta^{(5)}$ Test error: $J_{test}(\theta^{(5)})$

use it on Test set

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

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

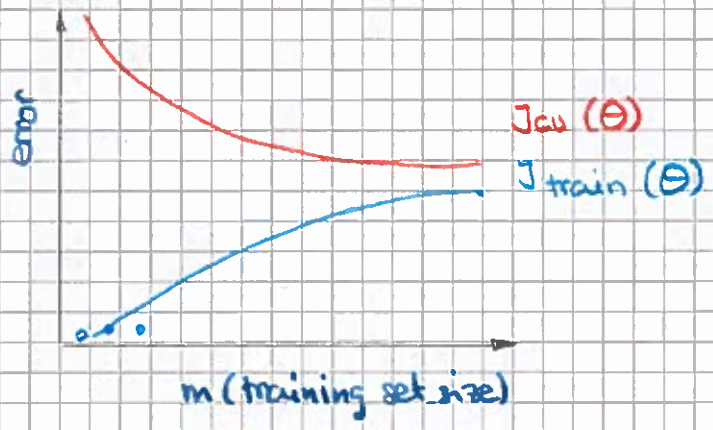
$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \cdot \sum_{i=1}^{m_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

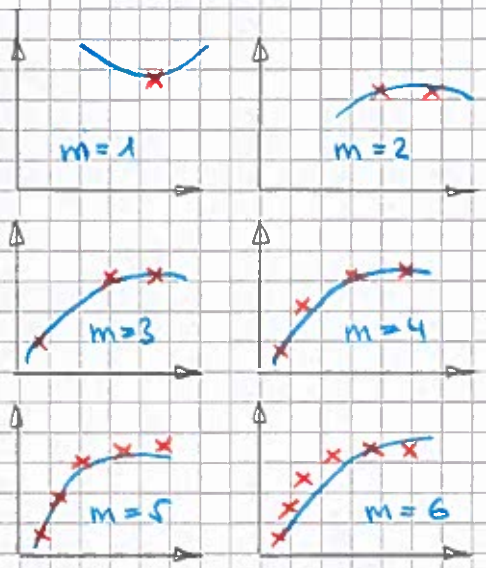


$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

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

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$



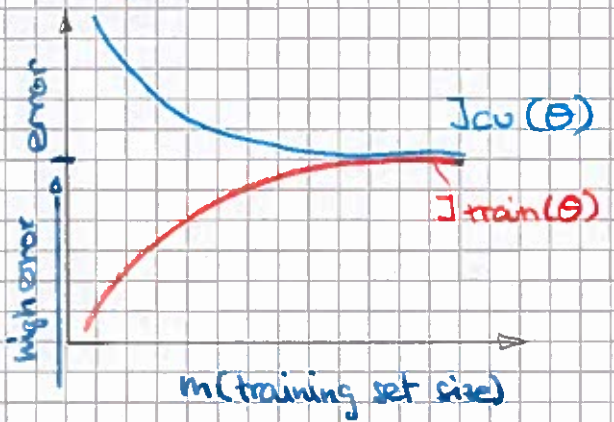


Small amount of training sets

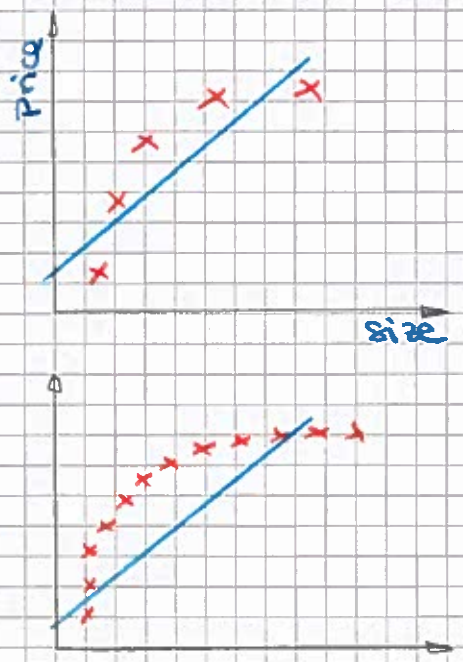
J_{cv} does not generalize well

only when training set gets larger (better)

High bias



$$h_\theta(x) = \theta_0 + \theta_1 x$$

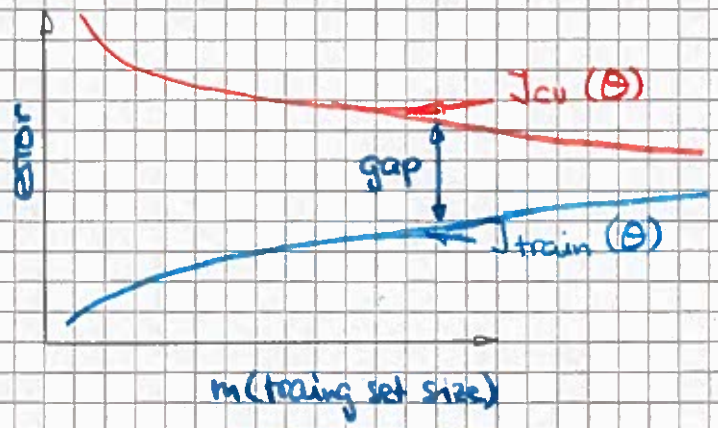


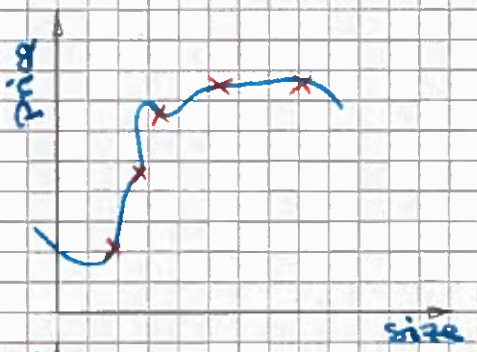
if a learning algorithm is suffering from high bias, getting more data will not help much.

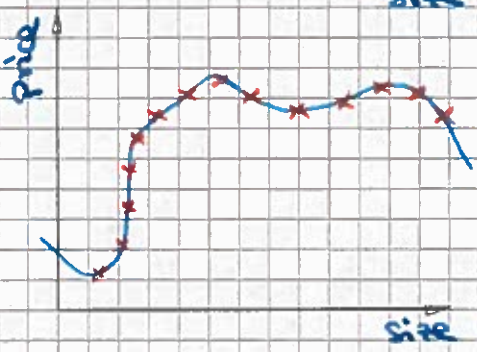
High variance

$h_\theta(x) = \theta_0 + \theta_1 x + \dots + \theta_{100} x^{100}$

(and small λ)







If a learning algorithm is suffering from high variance, getting more training data is likely to help.

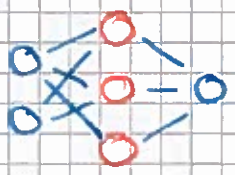
Debugging a learning algorithm

Suppose you have implemented regularized linear regression to predict housing prices. However when you test your hypothesis in a new set of houses, you find that it makes unacceptable large errors in it's prediction. What should you do?

- get more training data → fix high variance
- try smaller sets of features → fix high variance
- try getting additional features → fixing high bias problem
- try add polynomio features → fix high bias
- decrease λ → fixes high bias
- increase λ → fixes high variance

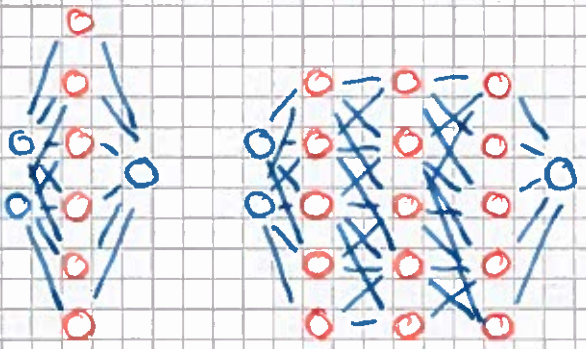
Neural networks and overfitting

Small neural networks
(fewer params, more prone to underfitting)



Computationally cheaper

Large neural network
(more params, more prone to overfitting)



Computationally more expensive

use regularization to adress overfitting

$J_{cv}(\theta)$

using multiple hidden layer is better even if there is the chance to overfit, but then play with lamda to reduce oberfitting