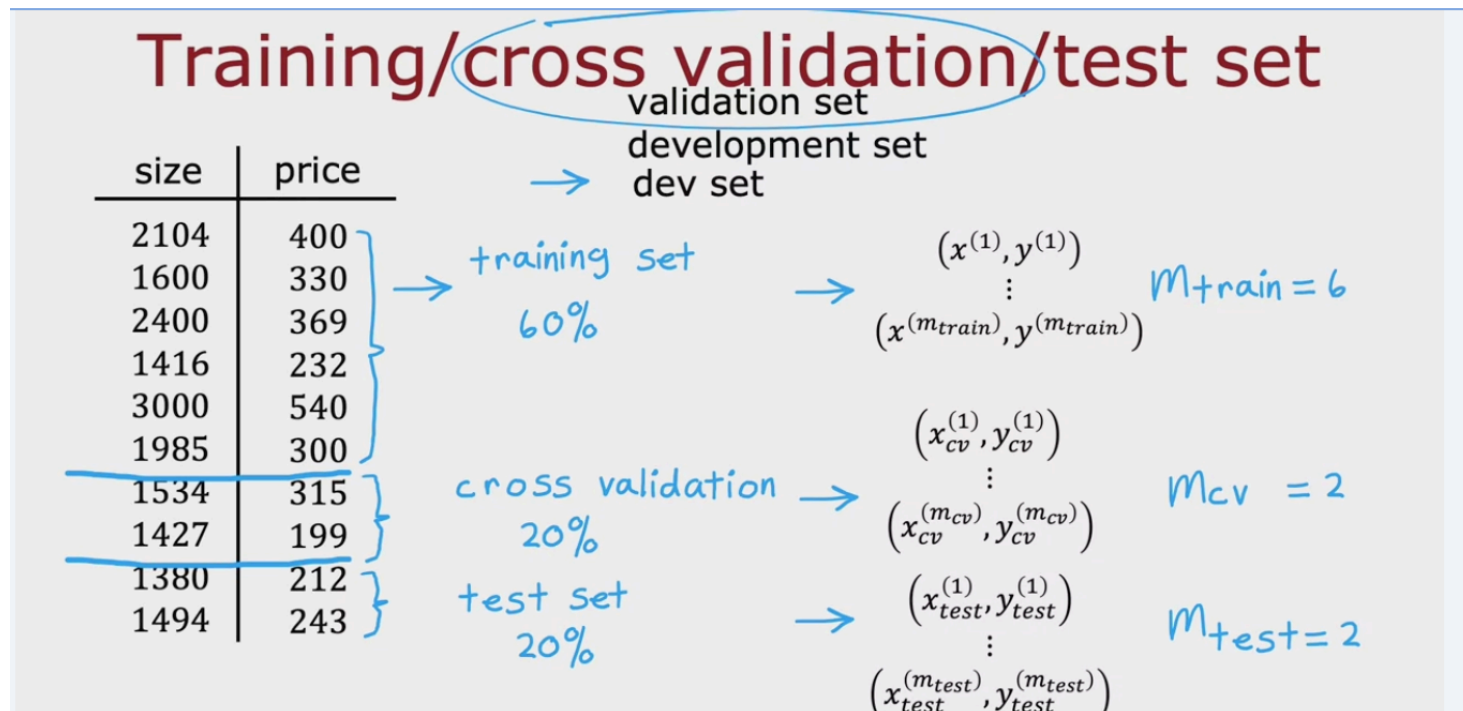


# Advanced Learning Algorithms - Week 3

ML Diagnostics:

Validation / Test:



## Training/cross validation/test set

Training error:  $J_{train}(\vec{w}, b) = \frac{1}{2m_{train}} \left[ \sum_{i=1}^{m_{train}} (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})^2 \right]$

Cross validation error:  $J_{cv}(\vec{w}, b) = \frac{1}{2m_{cv}} \left[ \sum_{i=1}^{m_{cv}} (f_{\vec{w}, b}(\vec{x}_{cv}^{(i)}) - y_{cv}^{(i)})^2 \right]$  (validation error, dev error)

Test error:  $J_{test}(\vec{w}, b) = \frac{1}{2m_{test}} \left[ \sum_{i=1}^{m_{test}} (f_{\vec{w}, b}(\vec{x}_{test}^{(i)}) - y_{test}^{(i)})^2 \right]$

The **cross validation** set is used to evaluate different models during training to choose the best one, while the test set is typically reserved for final evaluation after the model has been selected.

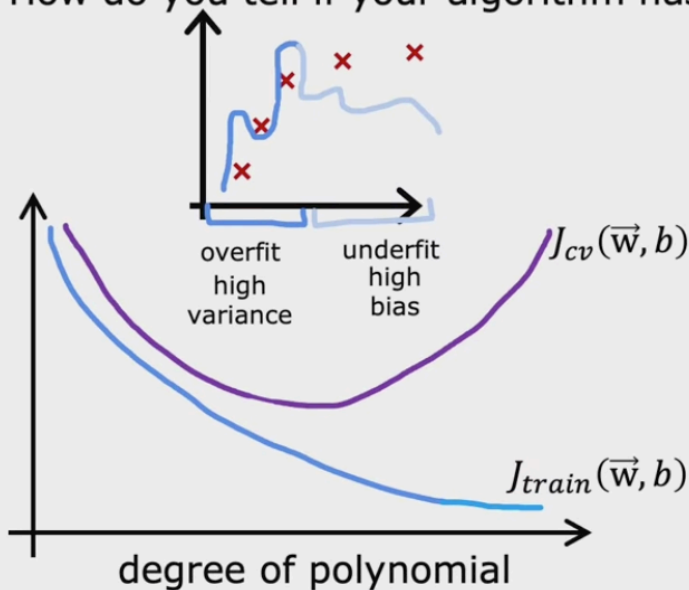
High Bias - Underfit

High Variance - Overfit

Both  $J_{train}$  and  $J_{cv}$  should be low - how a better model performs.

# Diagnosing bias and variance

How do you tell if your algorithm has a bias or variance problem?



High bias (underfit)

→  $J_{train}$  will be high  
( $J_{train} \approx J_{cv}$ )

High variance (overfit)

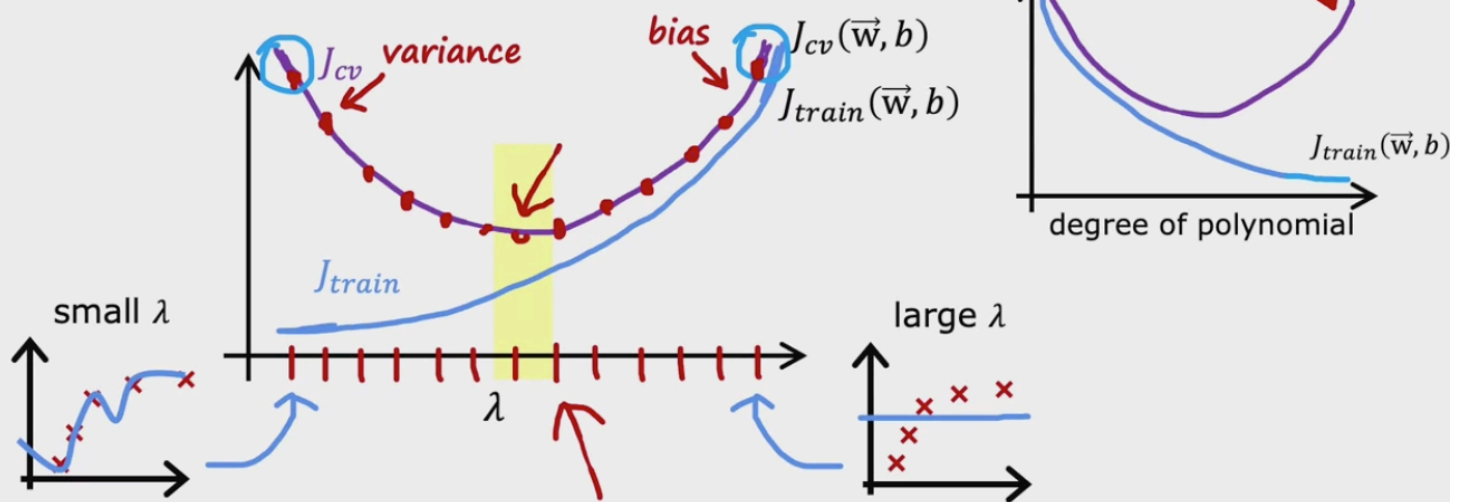
→  $J_{cv} \gg J_{train}$   
( $J_{train}$  may be low)

High bias and high variance

→  $J_{train}$  will be high  
→ and  $J_{cv} \gg J_{train}$

## Bias and variance as a function of regularization parameter $\lambda$

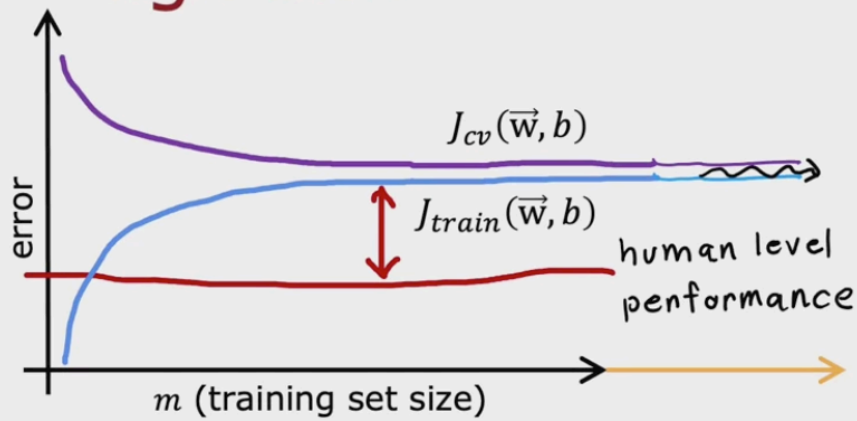
$$J(\vec{w}, b) = \frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^n w_j^2$$



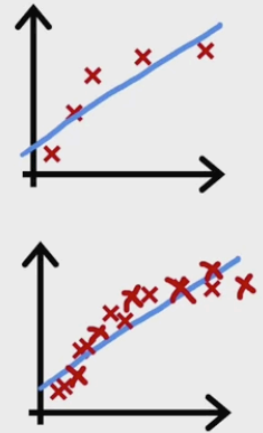
Gap between the baseline performance & training error tells you if you have high bias.

Gap between training error & cross validation error tells you if you have high variance.

# High bias



$$f_{\vec{w},b}(x) = w_1x + b$$

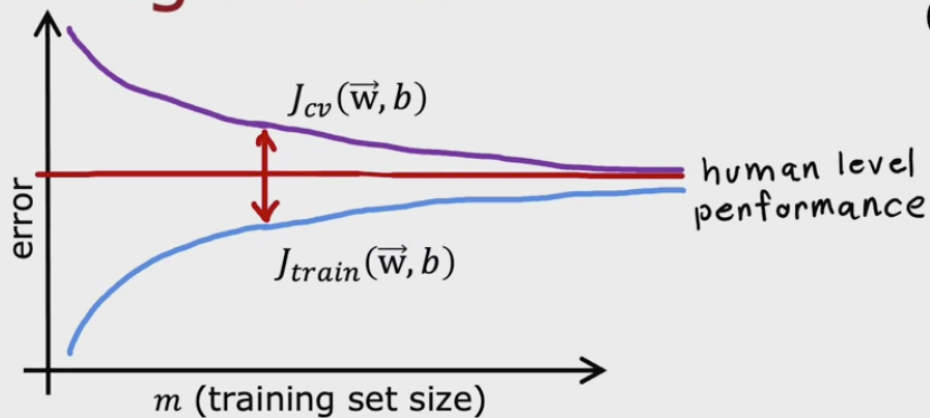


if a learning algorithm suffers from high bias, getting more training data will not (by itself) help much.

your learning algorithm has

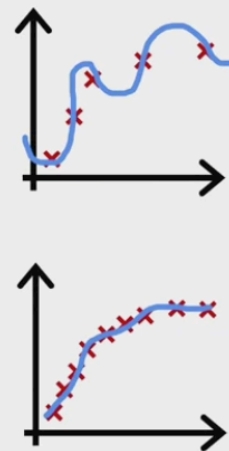
When high bias - throwing more training data generally does not help.

# High variance



$$f_{\vec{w},b}(x) = w_1x + w_2x^2 + w_3x^3 + w_4x^4 + b$$

(with small  $\lambda$ )



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

the training set

# Debugging a learning algorithm

You've implemented regularized linear regression on housing prices

$$J(\vec{w}, b) = \underbrace{\frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})^2}_{\text{residual sum of squares}} + \underbrace{\frac{\lambda}{2m} \sum_{j=1}^n w_j^2}_{\text{regularization term}}$$

But it makes unacceptably large errors in predictions. What do you try next?

- Get more training examples
  - Try smaller sets of features  $x, x^2, \cancel{x^3}, \cancel{x^4}, \cancel{x^5} \dots$
  - Try getting additional features
  - Try adding polynomial features  $(\underline{x_1^2}, \underline{x_2^2}, \underline{x_1 x_2}, \text{etc})$
  - Try decreasing  $\lambda$
  - Try increasing  $\lambda$
- fixes high variance  
fixes high variance  
fixes high bias  
fixes high bias  
fixes high bias  
fixes high variance

**Tradeoff** between bias and variance

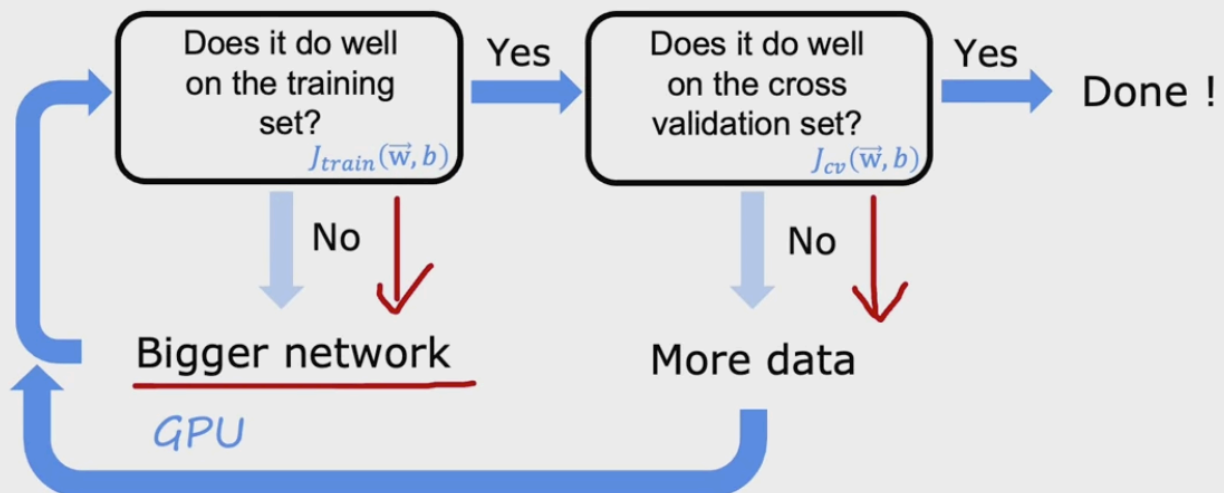
Simple model - high bias

Complex model - high variance

Large neural networks are low bias machines.

## Neural networks and bias variance

Large neural networks are low bias machines



## Machine Learning Development Process

Choose Architecture -> Train -> Diagnostics

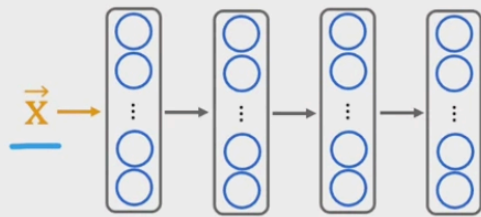
**Data Augmentation:**

Letter A - symmetric - distortion - color change - rotation - combination (noises)

**Transfer Learning:**

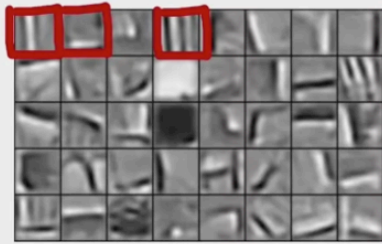
Copying levels

# Why does transfer learning work?

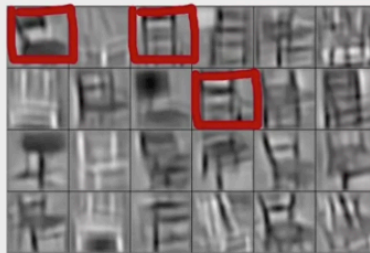


*use the same input type*

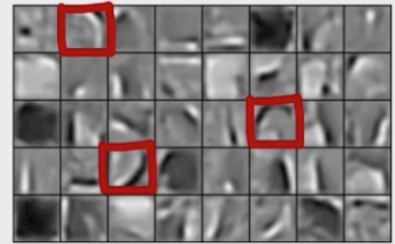
detects edges    detects corners    detects curves/basic shapes



Edges



Corners



Curves / basic shapes

- 1- download a neural network - trained with a large dataset - in your application
- 2- further train (finetune) the network on your data.