



CMSC726 Spring 2006: Evaluation + Bias/Variance Tradeoff

readings: Mitchell, ch. 5

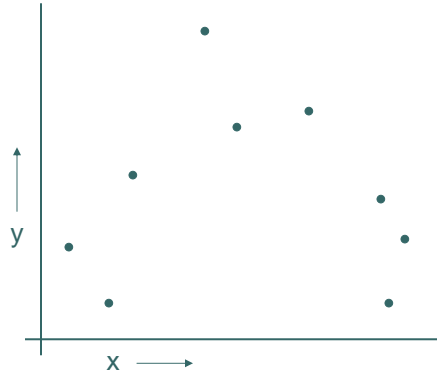
sources: course slides are based on material from a variety of sources, including **Tom Dietterich**, Carlos Guestrin, Terran Lane, Rich Maclin, Ray Mooney, **Andrew Moore**, Andrew Ng, Jude Shavlik, and others.

material from Andrew Moore <http://www.cs.cmu.edu/~awm/tutorials>

● ● ● | Outline

- Evaluating Hypotheses & Learning Algorithms
 - Motivation
 - Confusion Matrixes
 - Confidence Intervals
 - Cross Validation
- Bias-Variance Decomposition for Regression
- Summary and Conclusion

● ● ● | Motivation: A Regression Problem

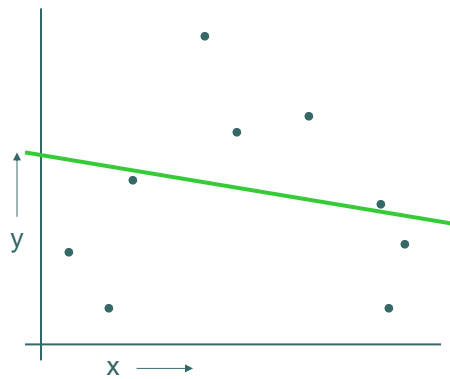


$$y = f(x) + \text{noise}$$

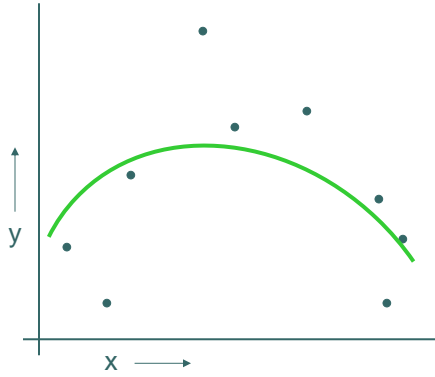
Can we learn f from this data?

Let's consider three methods...

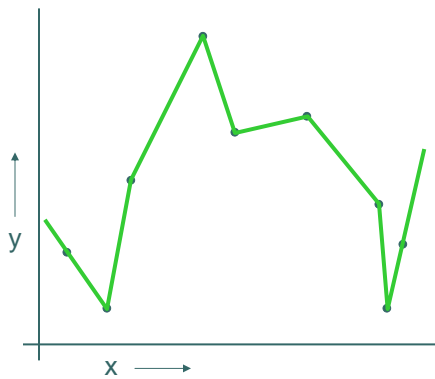
● ● ● | Linear Regression



● ● ● | Quadratic Regression

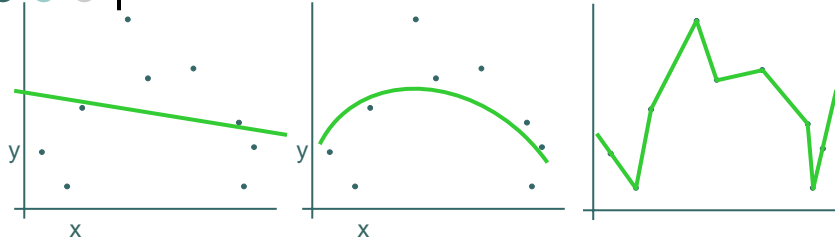


● ● ● | Join-the-dots



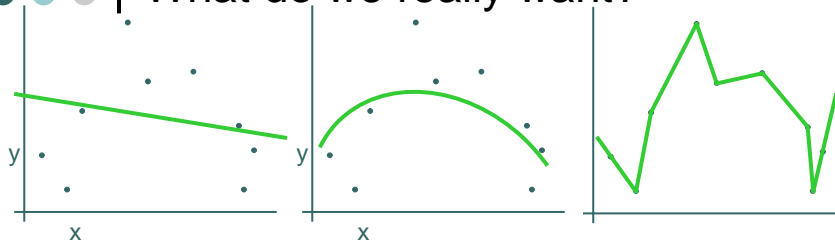
Also known as **piecewise linear nonparametric regression** if you think that sounds better

● ● ● | Which is best?



Why not choose the method with the best fit to the data?

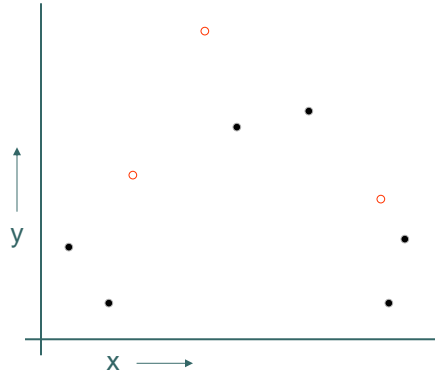
● ● ● | What do we really want?



Why not choose the method with the best fit to the data?

“How well are you going to **predict** future data drawn from the same distribution?”

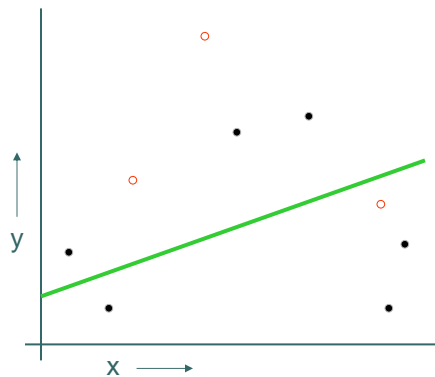
● ● ● | The test set method



1. Randomly choose 30% of the data to be in a **test set**

2. The remainder is a training set

● ● ● | The test set method



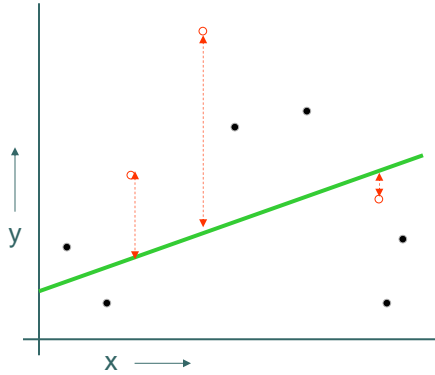
1. Randomly choose 30% of the data to be in a **test set**

2. The remainder is a training set

3. Perform your regression on the training set

(Linear regression example)

● ● ● | The test set method



(Linear regression example)

Mean Squared Error = 2.4

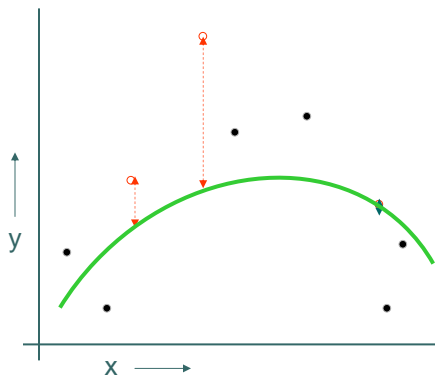
1. Randomly choose 30% of the data to be in a **test set**

2. The remainder is a training set

3. Perform your regression on the training set

4. Estimate your future performance with the test set

● ● ● | The test set method



(Quadratic regression example)

Mean Squared Error = 0.9

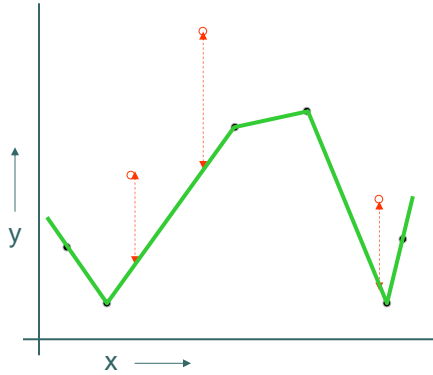
1. Randomly choose 30% of the data to be in a **test set**

2. The remainder is a training set

3. Perform your regression on the training set

4. Estimate your future performance with the test set

● ● ● | The test set method



(Join the dots example)
Mean Squared Error = 2.2

1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a training set
3. Perform your regression on the training set
4. Estimate your future performance with the **test set**

● ● ● | The test set method

Good news:

- Very very simple
- Can then simply choose the method with the best test-set score

Bad news:

- What's the downside?

● ● ● | The test set method

Good news:

- Very very simple
- Can then simply choose the method with the best test-set score

Bad news:

- Wastes data: we get an estimate of the best method to apply to 30% less data
- If we don't have much data, our test-set might just be lucky or unlucky

We say the "test-set estimator of performance has high variance"

● ● ● | Terminology and Evaluating Hypotheses

- Statistics
 - Basic terms
 - Sample error, true error
 - Distributions
 - Cost/utility
 - Tests for significance
- Comparing Learning Methods

● ● ● | Stats Refresher

- Given random variable Y , the **expected value** (or mean value), is

$$E[Y] = \sum_{i=1}^n y_i \Pr(Y = y_i)$$

- The **variance** of a random variable is

$$\text{Var}(Y) = E[(Y - E[Y])^2]$$

- The variance characterizes the width or dispersion of the distribution about its mean

- The **standard deviation** of Y is $\sigma_Y = \sqrt{\text{Var}(Y)}$

- Given a sample S , we can estimate the mean and variance. The estimate for the mean is:

$$\bar{Y} = \frac{\sum_{i \in S} Y_i}{N} \quad s^2 = \frac{\sum_{i \in S} (Y_i - \bar{Y})^2}{N - 1}$$

● ● ● | Evaluation: Data Sets

- **Training data set** – the set of data used to learn a model or hypothesis
- **Test data set** – the set of data used to estimate some value (often accuracy) related to a model
- **Validation set** – a set of data used to select parameters for a model, often as follows
 - Divide training data into a “sub” training set and validation set
 - For each possible set of parameters
 - Create a model using the “sub” training set
 - Evaluate the model on the validation set and pick the one that performs the best

● ● ● | Evaluating Models

- Need a measure of value – the cost (loss, utility) of a model
- Often use accuracy (or error)
 - Accuracy – how many examples we get “right”
 - Error – how many examples we get wrong
- Can be weighted
 - If examples are not equal, could count the cost (or utility) of mispredicted (correct) examples

● ● ● | Confusion Matrix

		Predicted		
		Positive	Negative	Total
Actual	Positive	True Positive (TP)	False Negative (FN)	#Positives
	Negative	False Positive (FP)	True Negative (TN)	#Negatives
Total		TP+FP	FN+TN	#Examples

- Accuracy = $(TP+TN) / \#Examples$
- Error = $(FP+FN) / \#Examples$
- Recall (sensitivity, true positive rate) = $TP / \#Positives$
- Precision = $TP / (FP+TP)$
- True Negative Rate (specificity) = $TN / \#Negatives$
- False Positive Rate = $FP / (FP+TP)$
- False Negative Rate = $FN / \#Negatives$

● ● ● | Confusion Matrix – Multi Class

- For many problems (especially multiclass problems), often useful to examine the sources of error
- Confusion matrix:

		Predicted			Total
		ClassA	ClassB	ClassC	
Expected	ClassA	25	5	20	50
	ClassB	0	45	5	50
	ClassC	25	0	25	50
Total		50	50	50	150

● ● ● | Results Analysis: Confusion Matrix

- Building a confusion matrix
 - Zero all entries
 - For each data point add one in row corresponding to actual class of problem under column corresponding to predicted class
- Perfect prediction has all non-zeros values down the diagonal
- Off diagonal entries can often tell us about what is being mispredicted

● ● ● | Two Definitions of Error

The **true error** of hypothesis h with respect to target function f and distribution D is the probability that h will misclassify an instance drawn at random according to D .

$$error_D(h) \equiv \Pr_{x \in D}[f(x) \neq h(x)]$$

The **sample error** of h with respect to target function f and data sample S is the proportion of examples h misclassifies

$$error_S(h) \equiv \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

where $\delta(f(x) \neq h(x))$ is 1 if $f(x) \neq h(x)$, and 0 otherwise

How well does $error_S(h)$ estimate $error_D(h)$?

● ● ● | Problems Estimating Error

1. **Bias**: If S is training set, $error_S(h)$ is optimistically biased

$$bias \equiv E[error_S(h)] - error_D(h)$$

For unbiased estimate, h and S must be chosen independently

2. **Variance**: Even with unbiased S , $error_S(h)$ may still vary from $error_D(h)$

● ● ● | Example

Hypothesis h misclassifies 12 of 40 examples in S .

$$error_S(h) = \frac{12}{40} = .30$$

What is $error_D(h)$?

● ● ● | Estimators

Experiment:

1. Choose sample S of size n according to distribution D

2. Measure $error_S(h)$

$error_S(h)$ is a random variable (i.e., result of an experiment)

$error_S(h)$ is an unbiased estimator for $error_D(h)$

Given observed $error_S(h)$ what can we conclude about $error_D(h)$?

● ● ● | Confidence Intervals

If S contains n examples, $n \geq 30$, drawn independently of h and each other

Then with approximately $N\%$ probability, $error_D(h)$ lies in interval

$$error_S(h) \pm z_N \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

where

$N\%$:	50%	68%	80%	90%	95%	98%	99%
z_N :	0.67	1.00	1.28	1.64	1.96	2.33	2.53

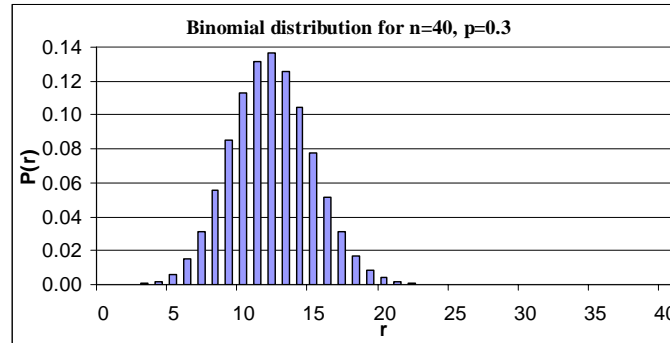
● ● ● | Confidence Intervals

If S contains n examples, drawn independently of h and each other

Then with approximately 95% probability, $error_D(h)$ lies in interval

$$error_S(h) \pm 1.96 \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

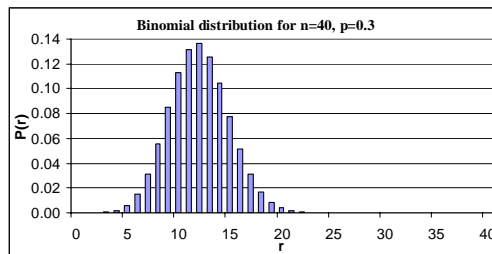
● ● ● | $error_S(h)$ is a Random Variable



- Rerun experiment with different randomly drawn S (size n)
- Probability of observing r misclassified examples:

$$P(r) = \frac{n!}{r!(n-r)!} error_D(h)^r (1 - error_D(h))^{n-r}$$

● ● ● | Binomial Probability Distribution

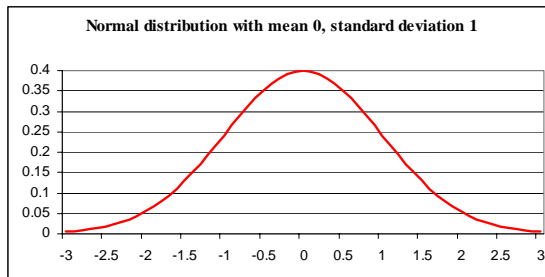


$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

Probability $P(r)$ of r heads in n coin flips, if $p = \Pr(\text{heads})$

- Expected, or mean value of X : $E[X] \equiv \sum_{i=0}^n iP(i) = np$
- Variance of X : $Var(X) \equiv E[(X - E[X])^2] = np(1-p)$
- Standard deviation of X : $\sigma_X \equiv \sqrt{E[(X - E[X])^2]} = \sqrt{np(1-p)}$

Normal Probability Distribution



$$P(r) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The probability that X will fall into the interval (a, b) is given by

$$\int_a^b p(x) dx$$

- Expected, or mean value of X : $E[X] = \mu$
- Variance of X : $Var(X) = \sigma^2$
- Standard deviation of X : $\sigma_x = \sigma$

Normal Distribution Approximates Binomial

$error_s(h)$ follows a Binomial distribution, with

- mean $\mu_{error_s(h)} = error_D(h)$
- standard deviation

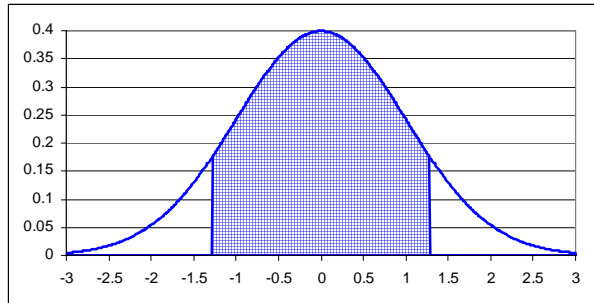
$$\sigma_{error_s(h)} = \sqrt{\frac{error_D(h)(1 - error_D(h))}{n}}$$

Approximate this by a Normal distribution with

- mean $\mu_{error_s(h)} = error_D(h)$
- standard deviation

$$\sigma_{error_s(h)} \approx \sqrt{\frac{error_s(h)(1 - error_s(h))}{n}}$$

● ● ● | Normal Probability Distribution



80% of area (probability) lies in $\mu \pm 1.28\sigma$

N% of area (probability) lies in $\mu \pm z_N\sigma$

N% :	50%	68%	80%	90%	95%	98%	99%
z_N :	0.67	1.00	1.28	1.64	1.96	2.33	2.53

● ● ● | Confidence Intervals, More Precisely

If S contains n examples, drawn independently of h and each other $n \geq 30$

Then with approximately 95% probability, $error_S(h)$ lies in interval

$$error_D(h) \pm 1.96 \sqrt{\frac{error_D(h)(1 - error_D(h))}{n}}$$

equivalently, $error_D(h)$ lies in interval

$$error_S(h) \pm 1.96 \sqrt{\frac{error_D(h)(1 - error_D(h))}{n}}$$

which is approximately

$$error_S(h) \pm 1.96 \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

● ● ● | Calculating Confidence Intervals

1. Pick parameter p to estimate $error_D(h)$
2. Choose an estimator $error_S(h)$
3. Determine probability distribution that governs estimator $error_S(h)$ governed by Binomial distribution, approximated by Normal when $n \geq 30$
4. Find interval (L, U) such that N% of probability mass falls in the interval
Use table of z_N values

● ● ● | Central Limit Theorem

Consider a set of independent, identically distributed random variables $Y_1 \dots Y_n$, all governed by an arbitrary probability distribution with mean μ and finite variance σ^2 . Define the sample mean

$$\bar{Y} \equiv \frac{1}{n} \sum_{i=1}^n Y_i$$

Central Limit Theorem. As $n \rightarrow \infty$, the distribution governing \bar{Y} approaches a Normal distribution, with mean μ and variance $\frac{\sigma^2}{n}$.

● ● ● | Difference Between Hypotheses

Test h_1 on sample S_1 , test h_2 on S_2

1. Pick parameter to estimate

$$d \equiv \text{error}_D(h_1) - \text{error}_D(h_2)$$

2. Choose an estimator

$$d \equiv \text{error}_{S_1}(h_1) - \text{error}_{S_2}(h_2)$$

3. Determine probability distribution that governs estimator

$$\sigma_d \approx \sqrt{\frac{\text{error}_{S_1}(h_1)(1 - \text{error}_{S_1}(h_1))}{n_1} + \frac{\text{error}_{S_2}(h_2)(1 - \text{error}_{S_2}(h_2))}{n_2}}$$

4. Find interval (L, U) such that $N\%$ of probability mass falls in the interval

$$\hat{d} \pm z_N \sqrt{\frac{\text{error}_{S_1}(h_1)(1 - \text{error}_{S_1}(h_1))}{n_1} + \frac{\text{error}_{S_2}(h_2)(1 - \text{error}_{S_2}(h_2))}{n_2}}$$

● ● ● | Paired t test to Compare L_A, L_B

1. Partition data into k disjoint test sets T_1, T_2, \dots, T_k of equal size, where this size is at least 30.

2. For i from 1 to k do

$$S_i = \{D - T_i\}$$

$$h_A = L_A(S_i)$$

$$h_B = L_B(S_i)$$

$$\delta_i \leftarrow \text{error}_{T_i}(h_A) - \text{error}_{T_i}(h_B)$$

3. Return the value d , where

$$\bar{\delta} \equiv \frac{1}{k} \sum_{i=1}^k \delta_i$$

$N\%$ confidence interval estimate for d :

$$\bar{\delta} \pm t_{N, k-1} s_{\bar{\delta}}$$

$$s_{\bar{\delta}} \equiv \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^k (\delta_i - \bar{\delta})^2}$$

Note δ_i approximately Normally distributed

● ● ● | N-Fold Cross Validation

- Popular testing methodology
- Divide data into N even-sized random folds
- For $n = 1$ to N
 - Train set = all folds except n
 - Test set = fold n
 - Create learner with train set
 - Count number of errors on test set
- Accumulate number of errors across N test sets and divide by N (result is error rate)
- For comparing algorithms, use the same set of folds to create learners (results are paired)

● ● ● | N-Fold Cross Validation

- Advantages/disadvantages
 - Estimate of error within a single data set
 - Every point used once as a test point
 - At the extreme (when $N =$ size of data set), called leave-one-out testing
 - Results affected by random choices of folds (sometimes answered by choosing multiple random folds – Dietterich in a paper expressed significant reservations)

● ● ● | The test set method

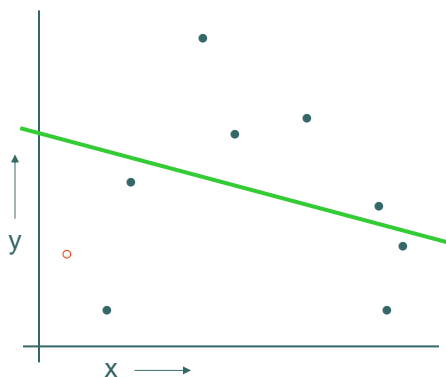
Good news:

- Very very simple
- Can then simply choose the method with the best test-set score

Bad news:

- **Wastes data:** we get an estimate of the best method to apply to 30% less data
- If we don't have much data, our test-set might just be lucky or unlucky

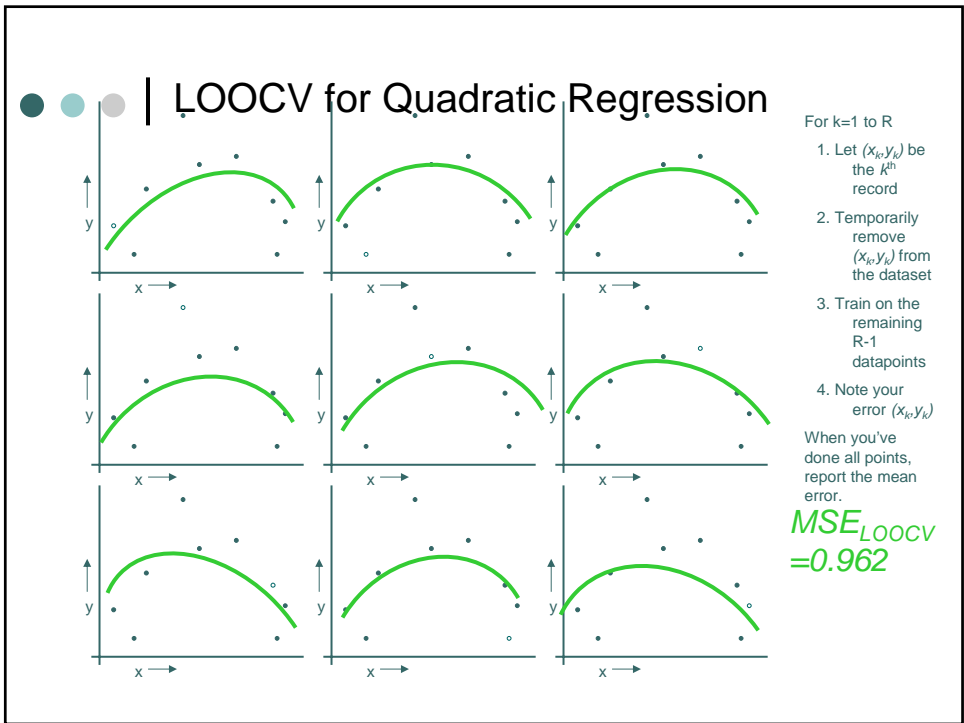
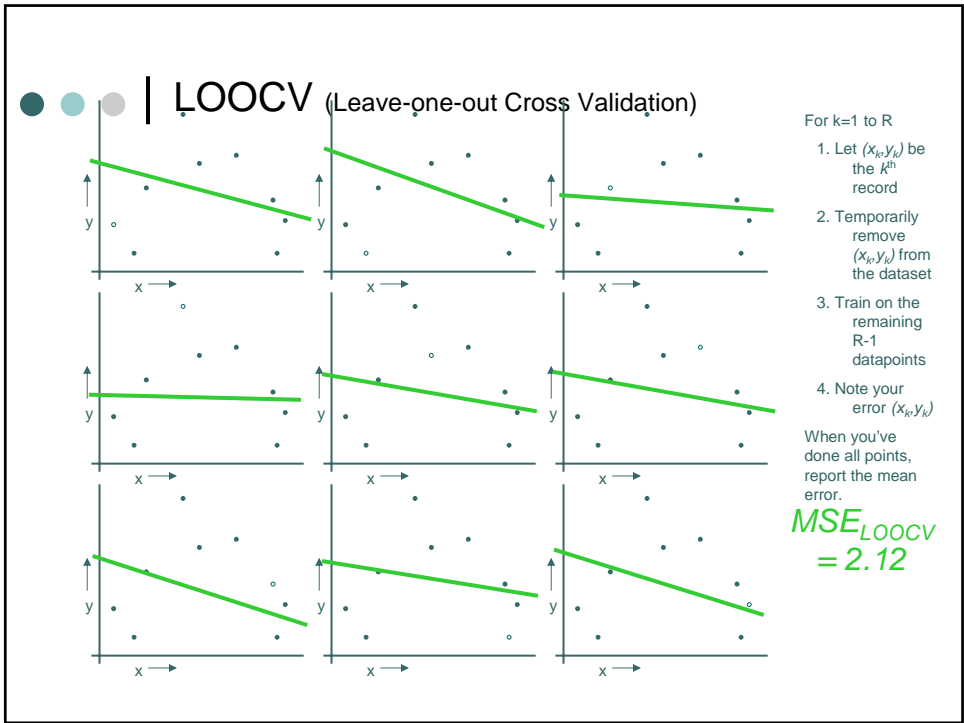
● ● ● | LOOCV (Leave-one-out Cross Validation)

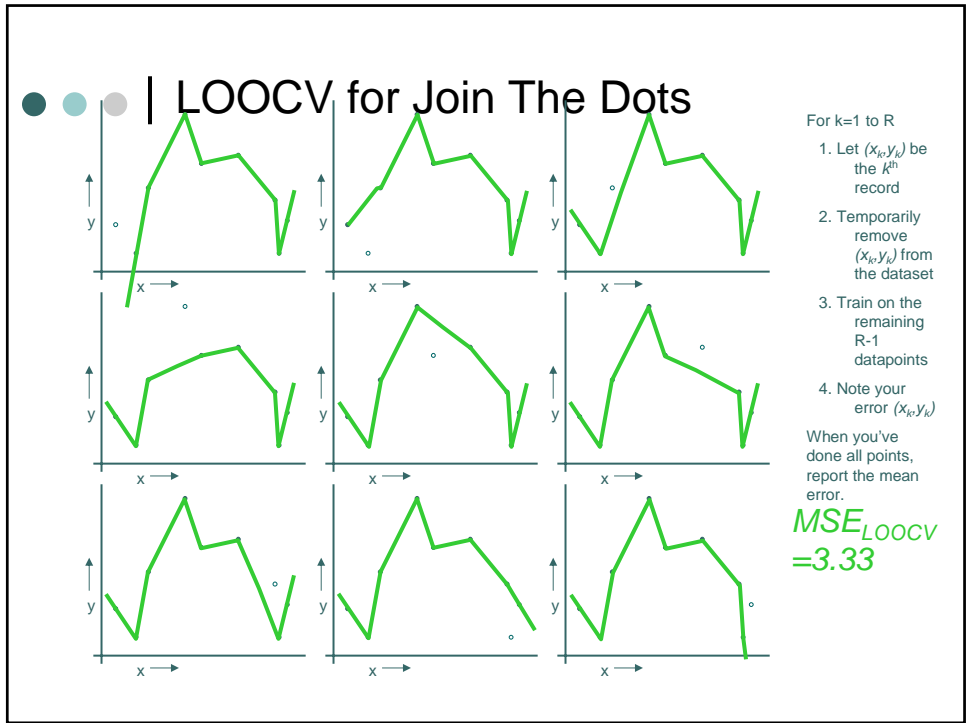


For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

When you've done all points, report the mean error.

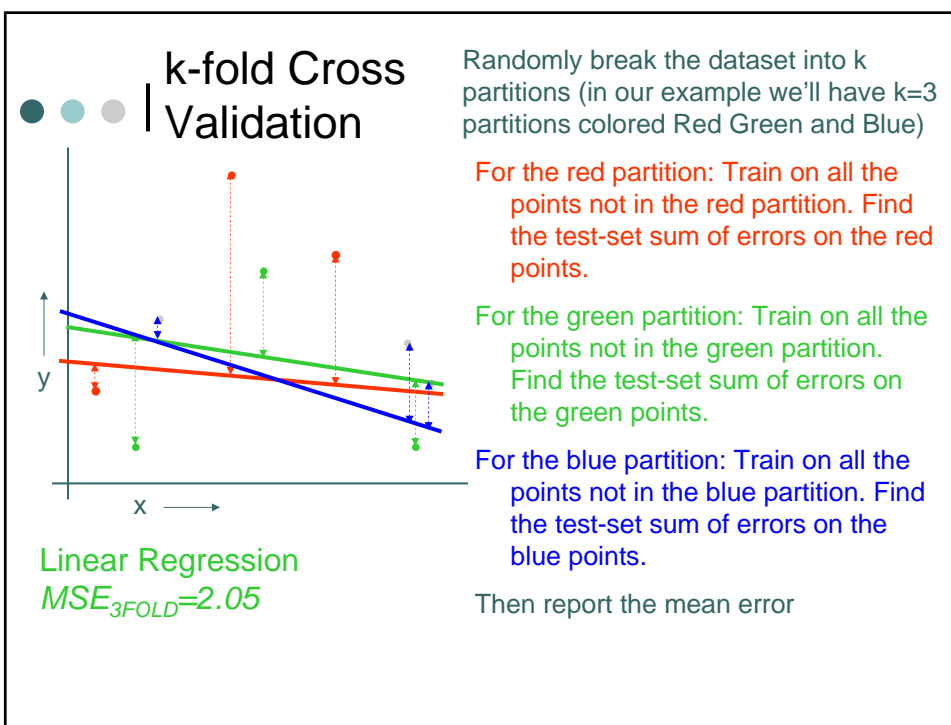
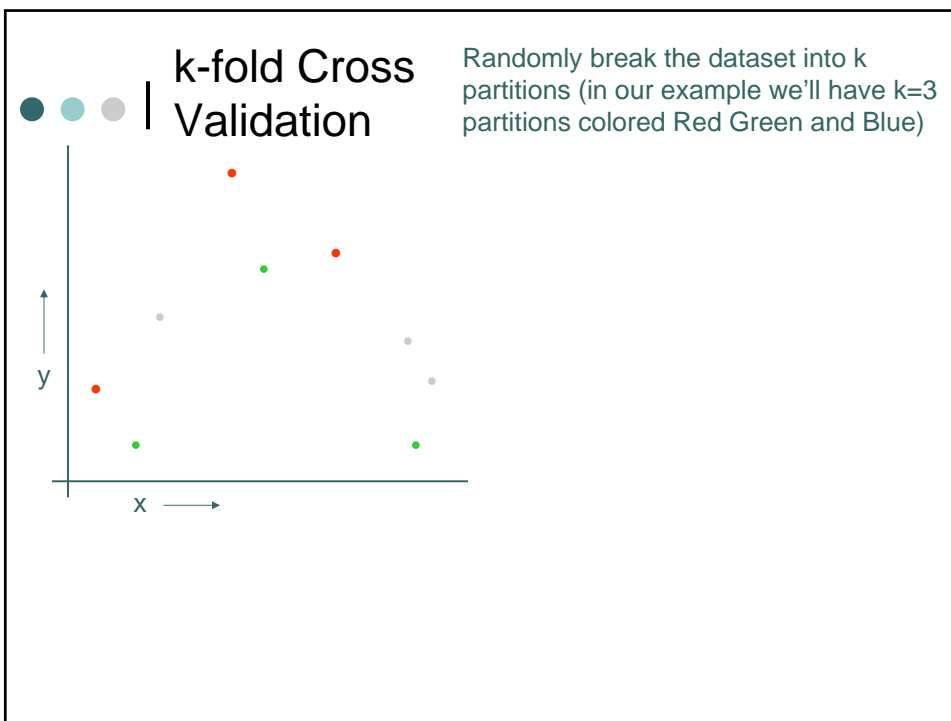




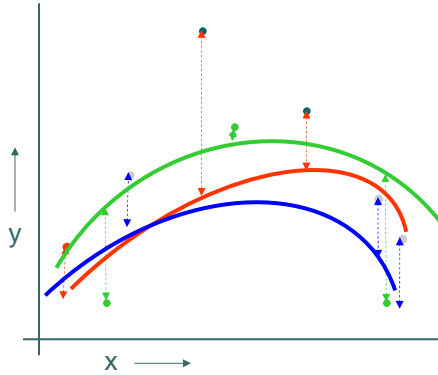
● ● ● | Which kind of Cross Validation?

	Downside	Upside
Test-set	Variance: unreliable estimate of future performance	Cheap
Leave-one-out	Expensive. Has some weird behavior	Doesn't waste data

..can we get the best of both worlds?



k-fold Cross Validation



Quadratic Regression
 $MSE_{3FOLD}=1.11$

Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

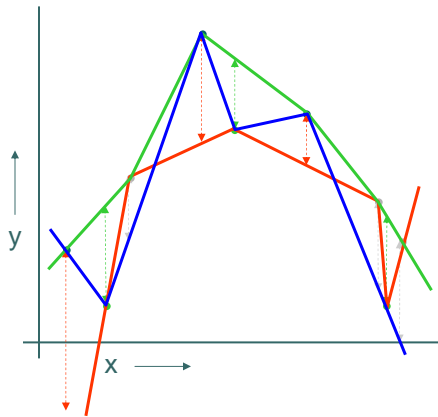
For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error

k-fold Cross Validation



Joint-the-dots
 $MSE_{3FOLD}=2.93$

Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error

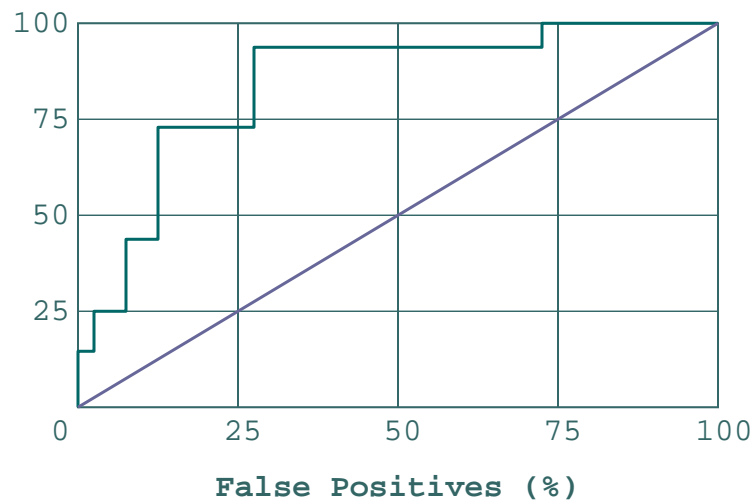
● ● ● | Which kind of Cross Validation?

	Downside	Upside
Test-set	Variance: unreliable estimate of future performance	Cheap
Leave-one-out	Expensive. Has some weird behavior	Doesn't waste data
10-fold	Wastes 10% of the data. 10 times more expensive than test set	Only wastes 10%. Only 10 times more expensive instead of R times.
3-fold	Wastes more than 10-fold. more expensive than test set	Slightly better than test-set
R-fold	Identical to Leave-one-out	

● ● ● | Receiver Operator Characteristic (ROC) Curves

- Originally from signal detection
- Becoming very popular for ML
- Used in:
 - Two class problems
 - Where predictions are ordered in some way (e.g., neural network activation is often taken as an indication of how strong or weak a prediction is)
- Plotting an ROC curve:
 - Sort predictions (right) by their predicted strength
 - Start at the bottom left
 - For each positive example, go up $1/P$ units where P is the number of positive examples
 - For each negative example, go right $1/N$ units where N is the number of negative examples

● ● ● | ROC Curve



● ● ● | ROC Properties

- Can visualize the tradeoff between coverage and accuracy (as we lower the threshold for prediction how many more true positives will we get in exchange for more false positives)
- Gives a better feel when comparing algorithms
 - Algorithms may do well in different portions of the curve
- A perfect curve would start in the bottom left, go to the top left, then over to the top right
 - A random prediction curve would be a line from the bottom left to the top right
- When comparing curves:
 - Can look to see if one curve dominates the other (is always better)
 - Can compare the area under the curve (very popular – some people even do t-tests on these numbers)

● ● ● | Bias-Variance Tradeoff: Intuition 1

- The goal in learning is not to learn an exact representation of the training data itself, but to build a statistical model of the process which generates the data. This is important if the algorithm is to have good generalization performance
- We saw that
 - models with too few parameters can perform poorly
 - models with too many parameters can perform poorly
- Need to optimize the complexity of the model to achieve the best performance
- One way to get insight into this tradeoff is the decomposition of generalization error into $\text{bias}^2 + \text{variance}$
 - a model which is too simple, or too inflexible, will have a large bias
 - a model which has too much flexibility will have high variance

● ● ● | Intuition

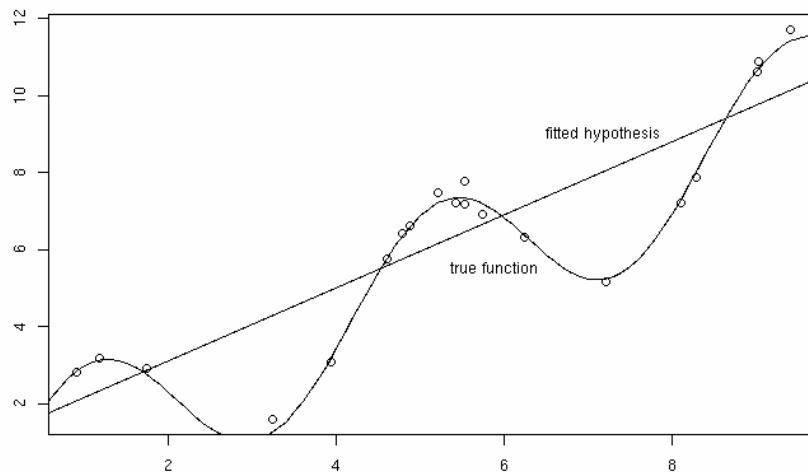
- bias:
 - measures the accuracy or quality of the algorithm
 - high bias means a poor match
- variance:
 - measures the precision or specificity of the match
 - a high variance means a weak match
- We would like to minimize each of these
- Unfortunately, we can't do this independently, there is a trade-off

● ● ● | Bias-Variance Analysis in Regression

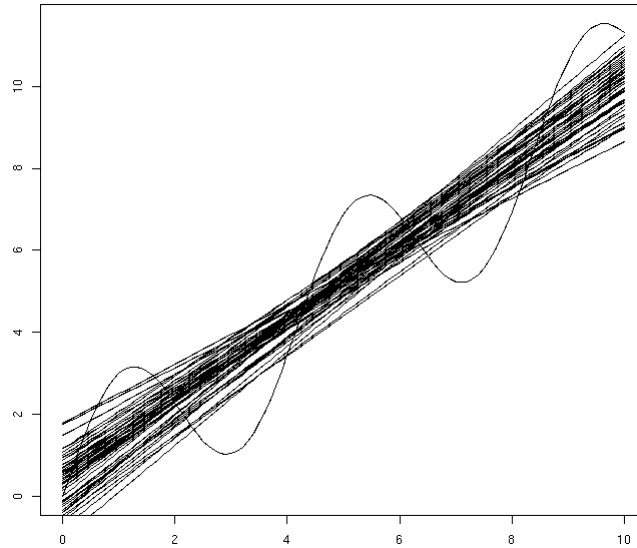
- True function is $y = f(x) + \varepsilon$
 - where ε is normally distributed with zero mean and standard deviation σ .
- Given a set of training examples, $\{(x_i, y_i)\}$, we fit an hypothesis $h(x) = w \cdot x + b$ to the data to minimize the squared error

$$\sum_i [y_i - h(x_i)]^2$$

● ● ● | Example: 20 points $y = x + 2 \sin(1.5x) + N(0,0.2)$



● ● ● | 50 fits (20 examples each)



● ● ● | Bias-Variance Analysis

- Now, given a new data point x^* (with observed value $y^* = f(x^*) + \varepsilon$, we would like to understand the expected prediction error

$$E[(y^* - h(x^*))^2]$$

● ● ● | Classical Statistical Analysis

- Imagine that our particular training sample S is drawn from some population of possible training samples according to $P(S)$.
- Compute $E_P [(y^* - h(x^*))^2]$
- Decompose this into “bias”, “variance”, and “noise”

● ● ● | Lemma

- Let Z be a random variable with probability distribution $P(Z)$
- Let $\bar{Z} = E_P[Z]$ be the average value of Z .
- Lemma : $E[(Z - \bar{Z})^2] = E[Z^2] - \bar{Z}^2$

$$\begin{aligned} E[(Z - \bar{Z})^2] &= E[Z^2 - 2Z\bar{Z} + \bar{Z}^2] \\ &= E[Z^2] - 2E[Z]\bar{Z} + \bar{Z}^2 \\ &= E[Z^2] - 2\bar{Z}^2 + \bar{Z}^2 \\ &= E[Z^2] - \bar{Z}^2 \end{aligned}$$

- Corollary : $E[Z^2] = E[(Z - \bar{Z})^2] + \bar{Z}^2$

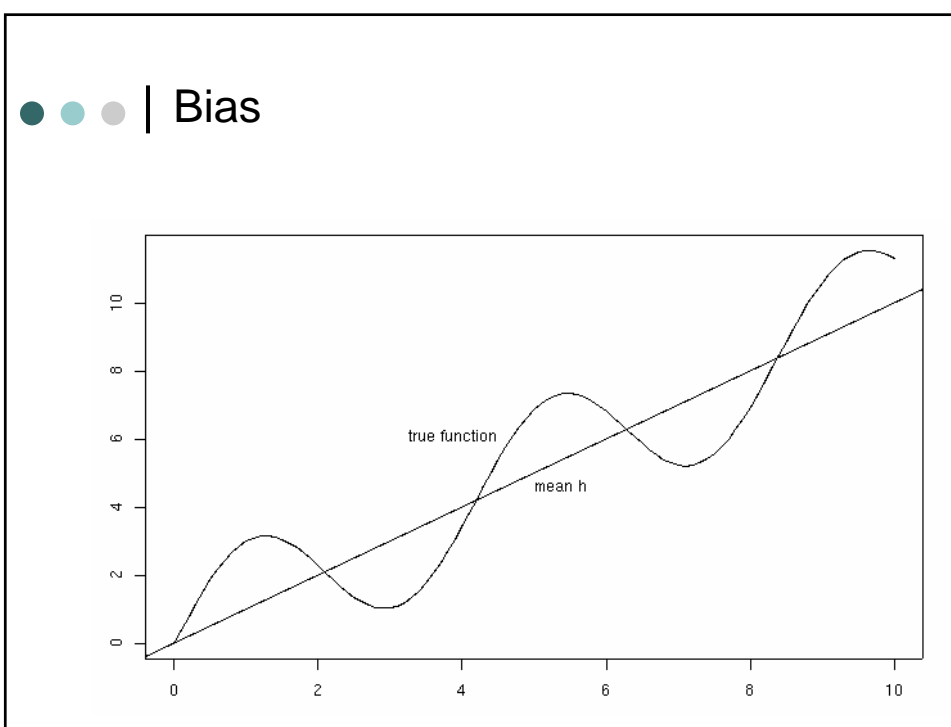
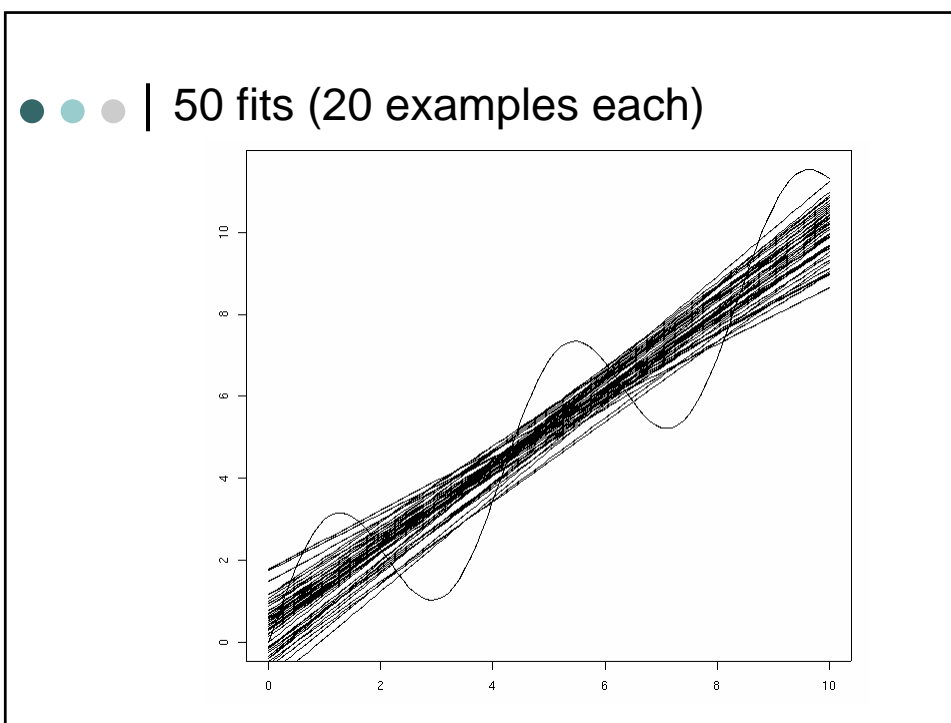
● ● ● | Bias-Variance-Noise Decomposition

$$\begin{aligned}
 E[(h(x^*) - y^*)^2] &= E[h(x^*)^2 - 2h(x^*)y^* + y^{*2}] \\
 &= E[h(x^*)^2] - 2E[h(x^*)]E[y^*] + E[y^{*2}] \\
 &= E[(h(x^*) - \overline{h(x^*)})^2] - \overline{h(x^*)}^2 \\
 &\quad - 2\overline{h(x^*)}f(x^*) \\
 &\quad + E[(y^* - f(x^*))^2] + f(x^*)^2 \\
 &= E[(h(x^*) - \overline{h(x^*)})^2] + \text{VARIANCE} \\
 &\quad \left(\overline{h(x^*)}^2 - f(x^*)^2 \right) + \text{BIAS} \\
 &\quad + E[(y^* - f(x^*))^2] + \text{NOISE} \\
 &= \text{Var}(h(x^*)) + \text{Bias}(h(x^*))^2 + E[\varepsilon^2] \\
 &= \text{Var}(h(x^*)) + \text{Bias}(h(x^*))^2 + \sigma^2
 \end{aligned}$$

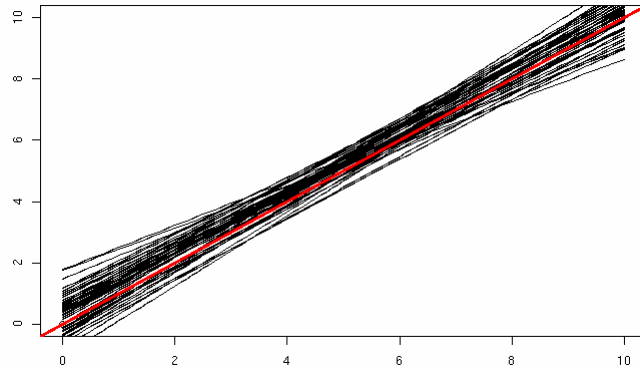
Expected prediction error = Variance + Bias² + Noise²

● ● ● | Bias, Variance, and Noise

- Variance: $E[(h(x^*) - \overline{h(x^*)})^2]$
Describes how much $h(x^*)$ varies from one training set S to another
- Bias: $[\overline{h(x^*)} - f(x^*)]$
Describes the average error of $h(x^*)$.
- Noise: $E[(y^* - f(x^*))^2] = E[\varepsilon^2] = \sigma^2$
Describes how much y^* varies from $f(x^*)$



● ● ● | Variance



● ● ● | Noise

