Cmput 466 / 551

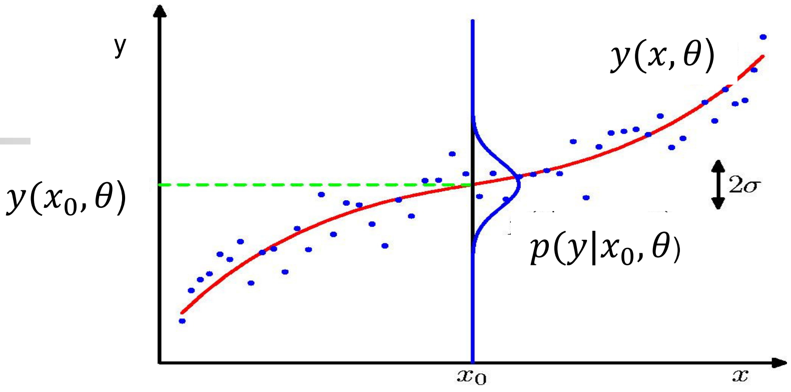
Linear Models, Regularization Bias-Variance Tradeoff

Covering chapters ***HTF: Ch3, 7***

R Greiner
Department of Computing Science
University of Alberta

Thanks to C Guestrin, T Dietterich, R Parr, N Ray, H L Størvold, R Salakhutdinov

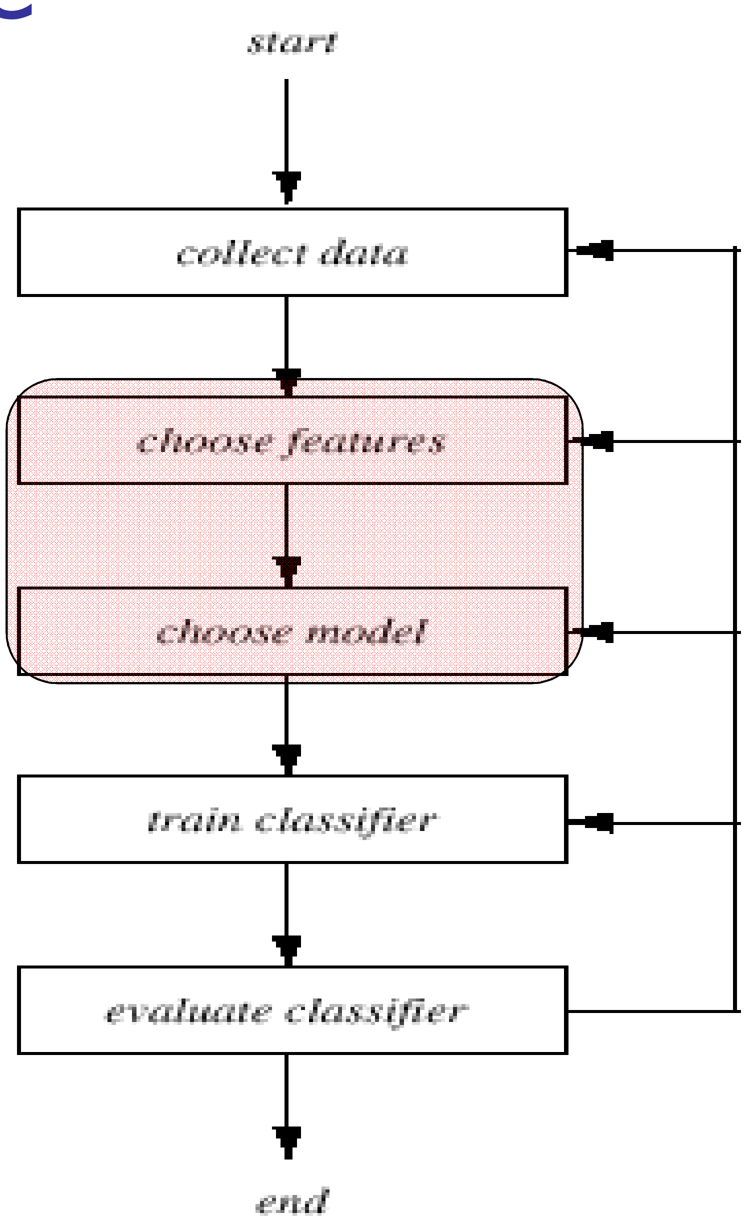
Outline



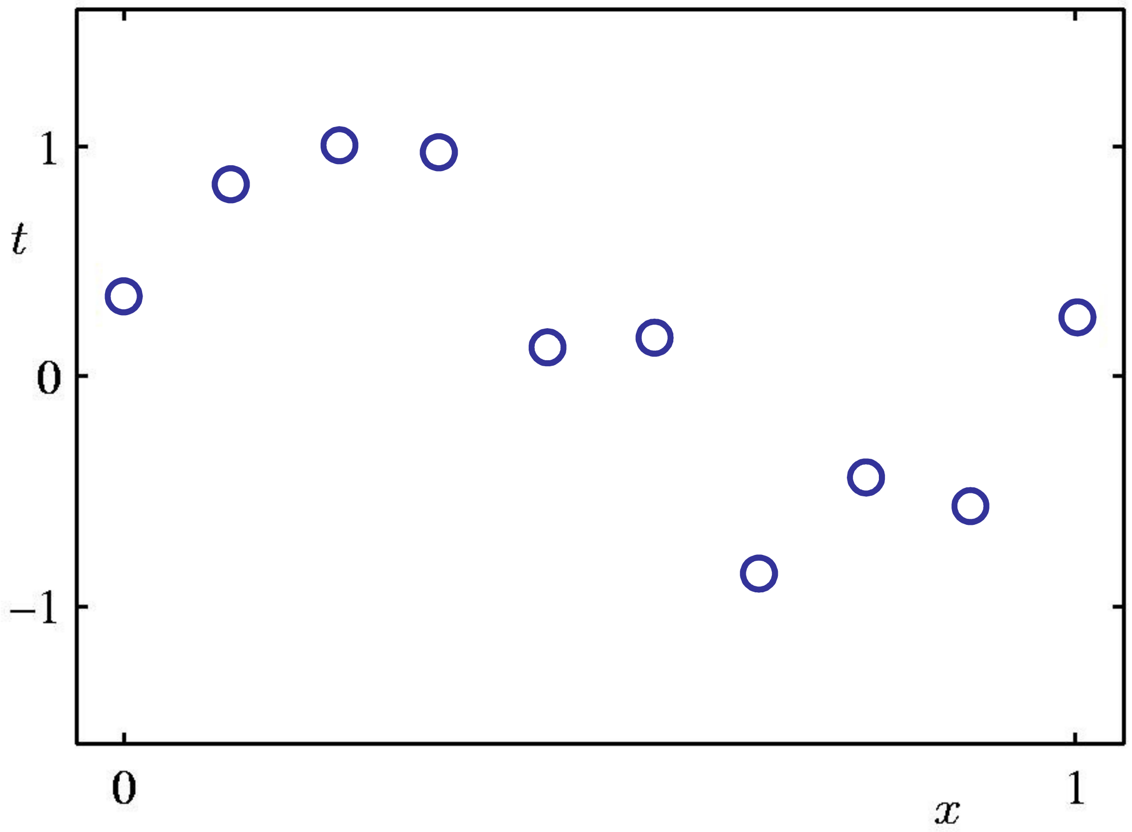
- Linear Regression
- Evaluating Predictors
  - Training set error vs Test set error
  - Cross Validation
- Overfitting
  - Bias-Variance analysis
  - Feature Selection
  - L2 Regularization
  - Setting parameters … internal C-V
  - Bayesian Model
  - L1-Regularization (Lasso)
- Linear Classification

The Design Cycle

How to do this (based on data) …

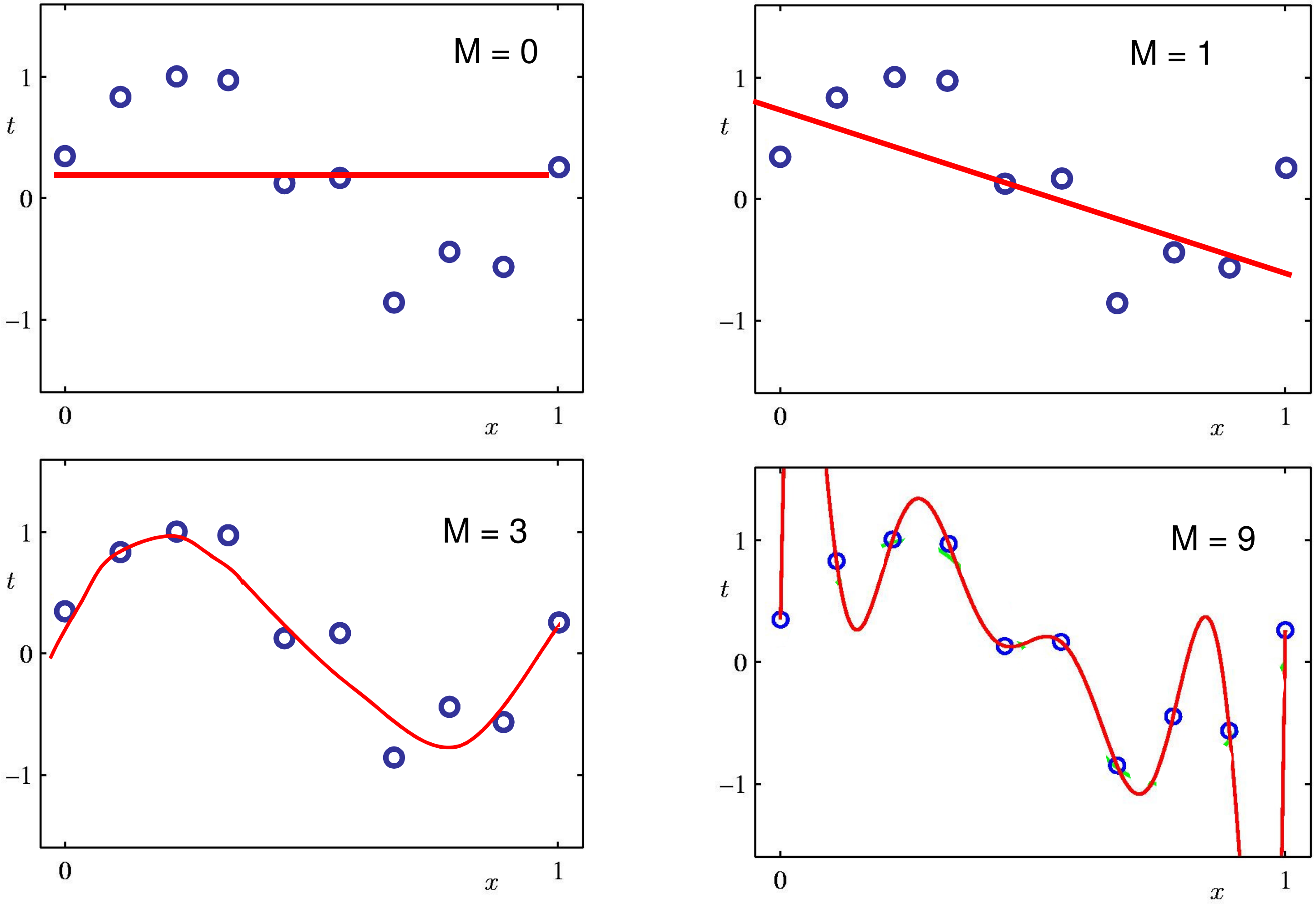


What is best choice of Polynomial?

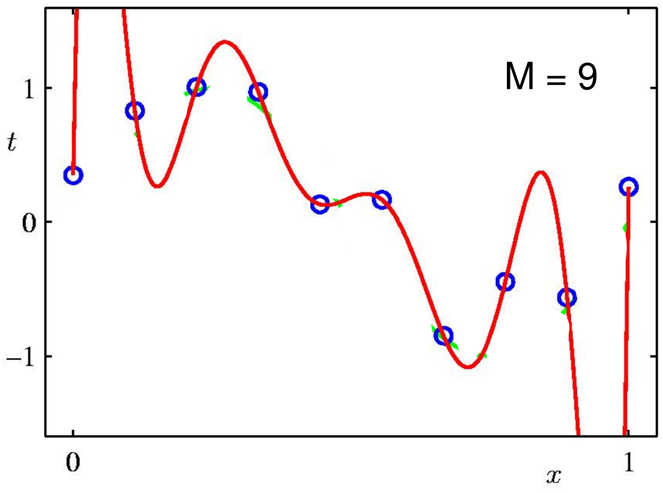


Noisy Source Data

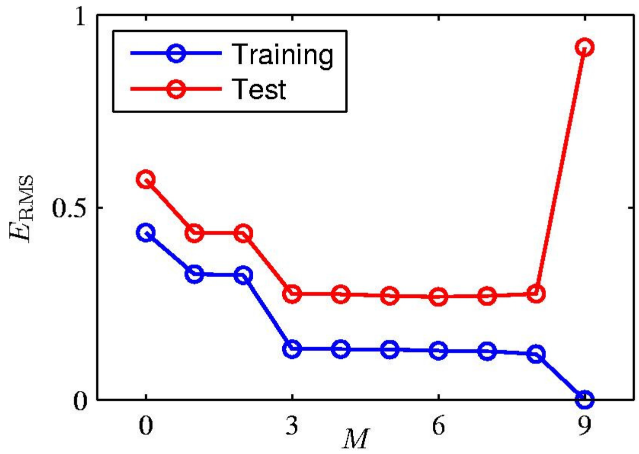
Fit using Degree: 0, 1, 3, 9



True error …



- Generate a new TEST SET:
  - 100 new (x,t) values
  - [same process that generated original training data]
- For M=9 (9th degree poly):
  - Training error is 0
    - … 10 degrees of freedom, can exactly fit 10 datapoints
  - Why test error so large?

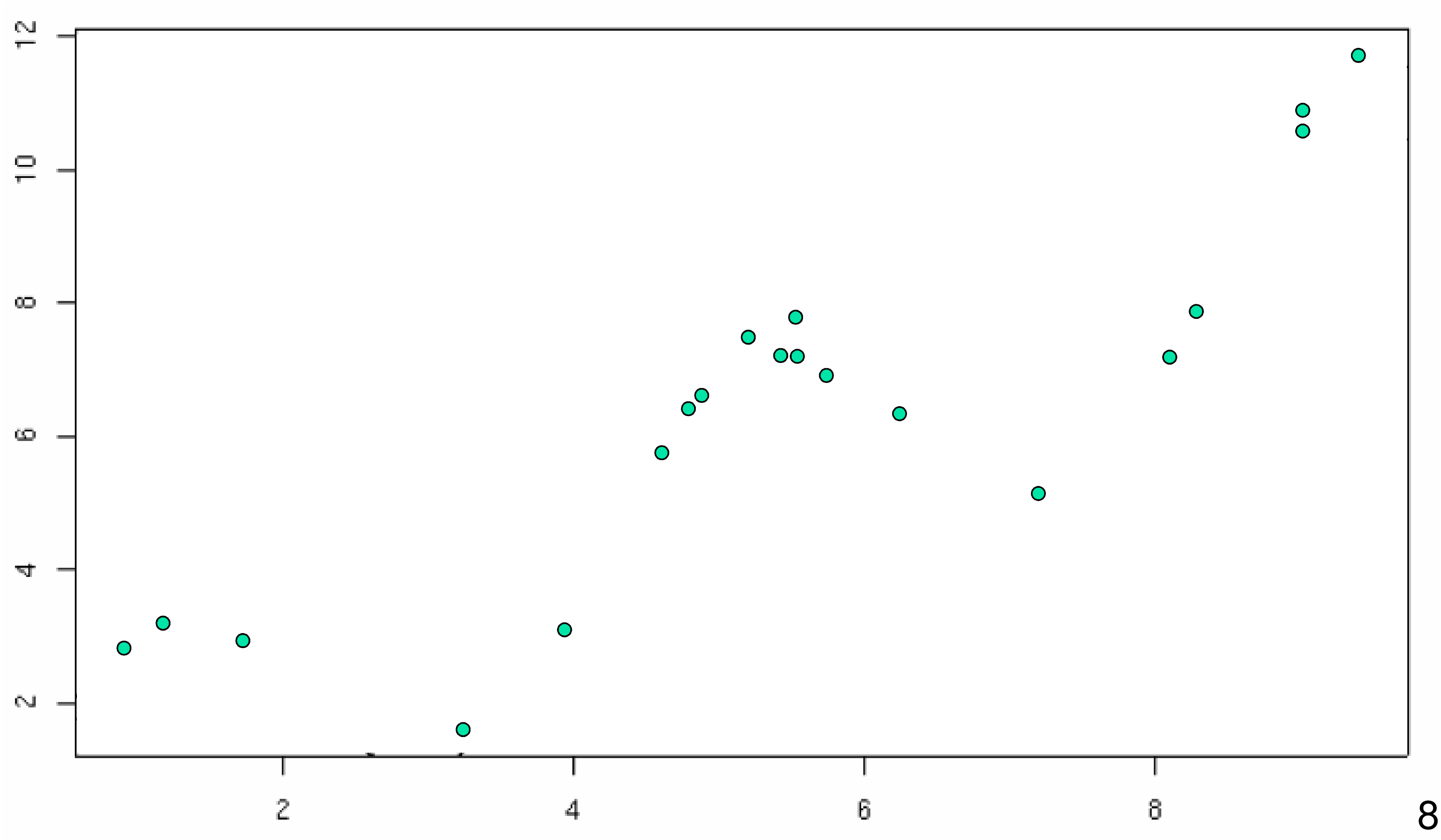


Bias-Variance Analysis in Regression

- Observed value is $t(\mathbf{x}) = h(\mathbf{x}) + \varepsilon$
  - Typically $\varepsilon \sim N(0, \sigma^2)$
    - normally distributed: mean 0, std deviation σ^2
  - Note: $h(\mathbf{x}) = E[\, t(\mathbf{x}) \mid \mathbf{x}\,]$
- Given training examples, $d = \{ (\mathbf{x}_i, t_i) \}$, let
  $y(\cdot\, ;\, d)$
  be predicted function,
  based on model learned using d
    - Here, linear model $\mathbf{y}(\mathbf{x}\, ;\, \mathbf{d}) = \boldsymbol{\theta}_{\mathbf{d}}^{\mathbf{T}}\mathbf{x}$
      using $\hat{\theta}_{\mathbf{d}} = \text{MLE}(\ d\ )$

Example of Fitted Hypothesis

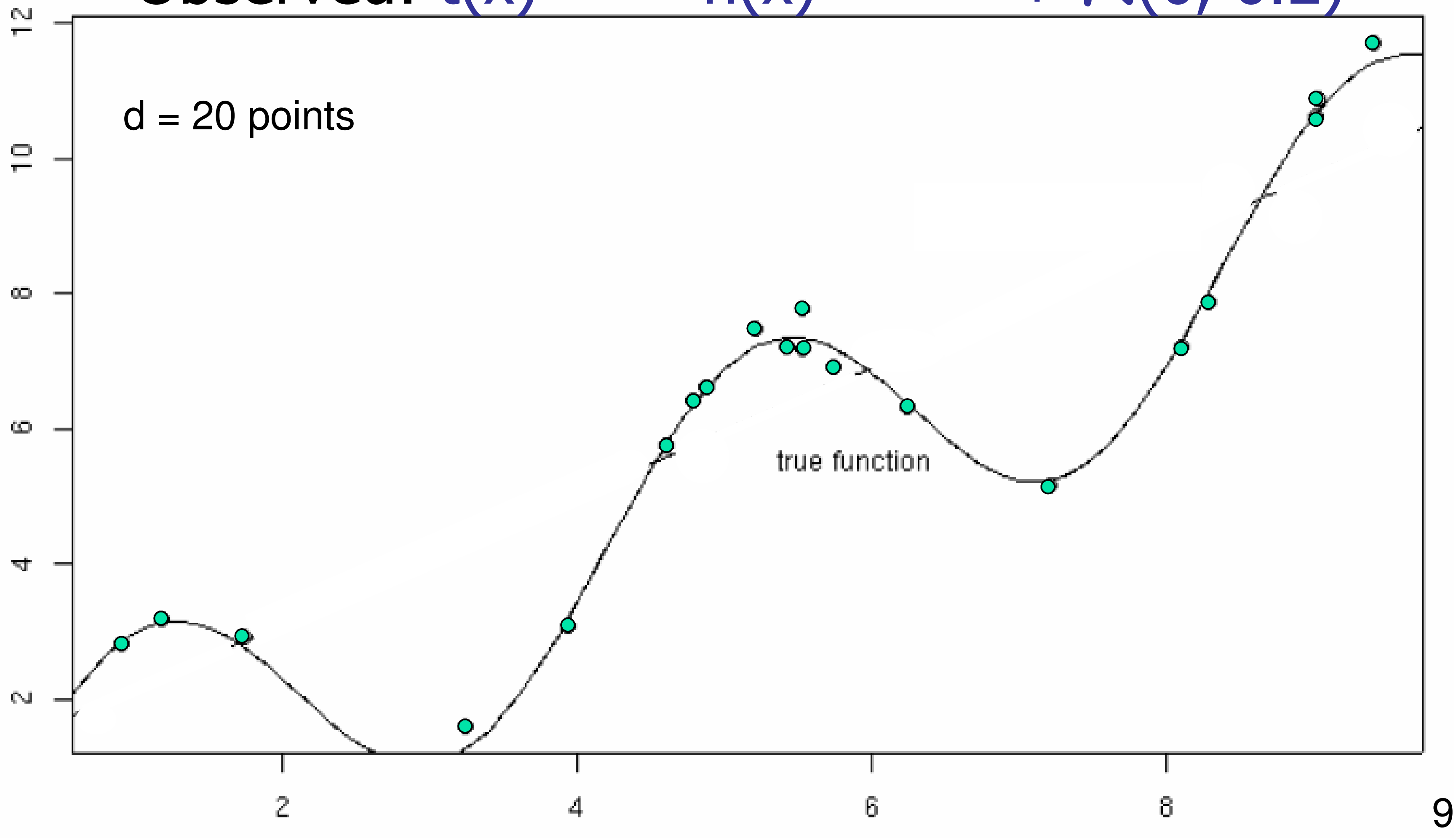
D = 20 points



Example of Fitted Hypothesis

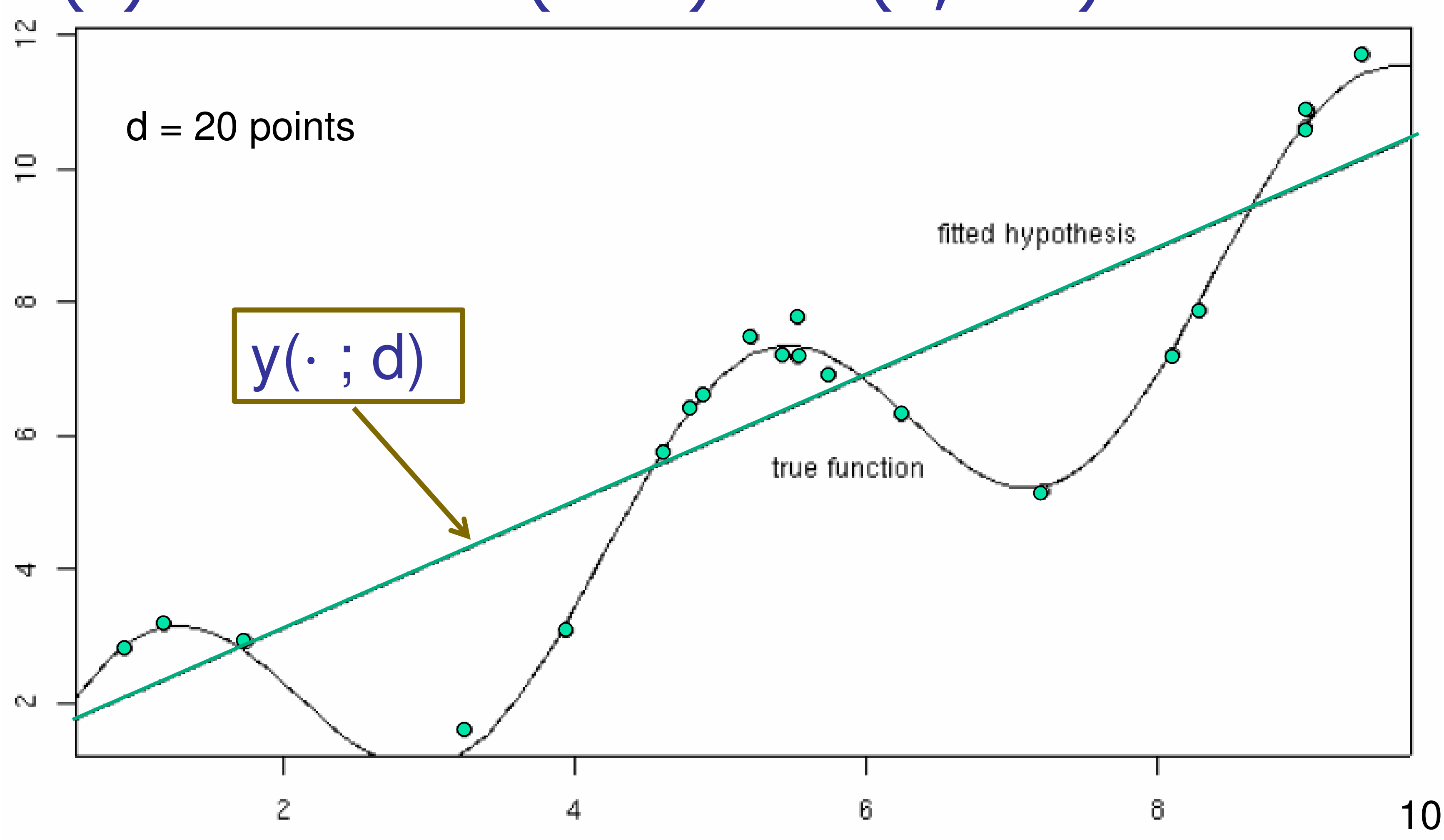
Truth: $h(x) = x + 2\sin(1.5x)$

Observed: $t(x) = h(x) + N(0, 0.2)$



Example of Fitted Hypothesis

$t(x) = x + 2\sin(1.5x) + N(0, 0.2)$



Bias-Variance Analysis

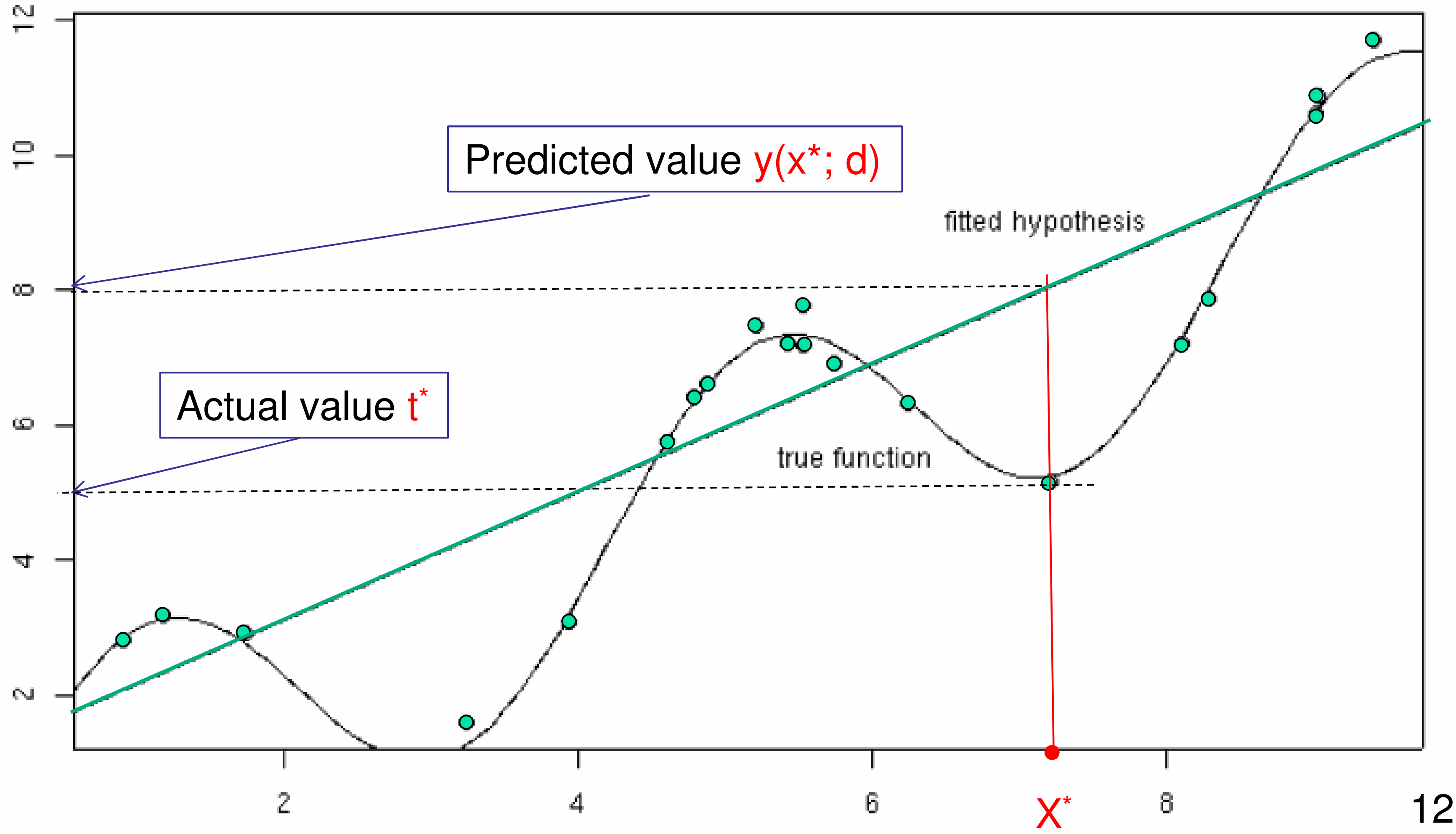
- Fix a dataset d
- Given a *new* data point $\mathbf{x}^*$
  - return predicted response: $y(\mathbf{x}^*; d)$
  - observed response: $t^* = h(\mathbf{x}^*) + \varepsilon$
- The *expected prediction error* is …

$$\boxed{\mathrm{Eerr}(d) = E_{(x^*,t^*)}[\, (t^* - y(\mathbf{x}^*; d))^2 \,]}$$

Example of Fitted Hypothesis

$t(x) = x + 2\sin(1.5x) + \mathcal{N}(0, 0.2)$

d = 20 points



Predicted value $y(x^*; d)$

Actual value t^*

X^*

Fixed dataset d: $y(\mathbf{x}) = y(\mathbf{x}, d)$

Expected Squared Loss

- $\boxed{[y(\mathbf{x}) - t]^2} = [y(\mathbf{x}) - h(\mathbf{x}) + h(\mathbf{x}) - t]^2 =$
  $[y(\mathbf{x}) - h(\mathbf{x})]^2$
  $+ 2\,[y(\mathbf{x}) - h(\mathbf{x})]\,\boxed{[h(\mathbf{x}) - t]}$
  $+ [h(\mathbf{x}) - t]^2$

Expected value (wrt t) is 0 as
$h(\mathbf{x}) = E[\, t \mid \mathbf{x}\,]$

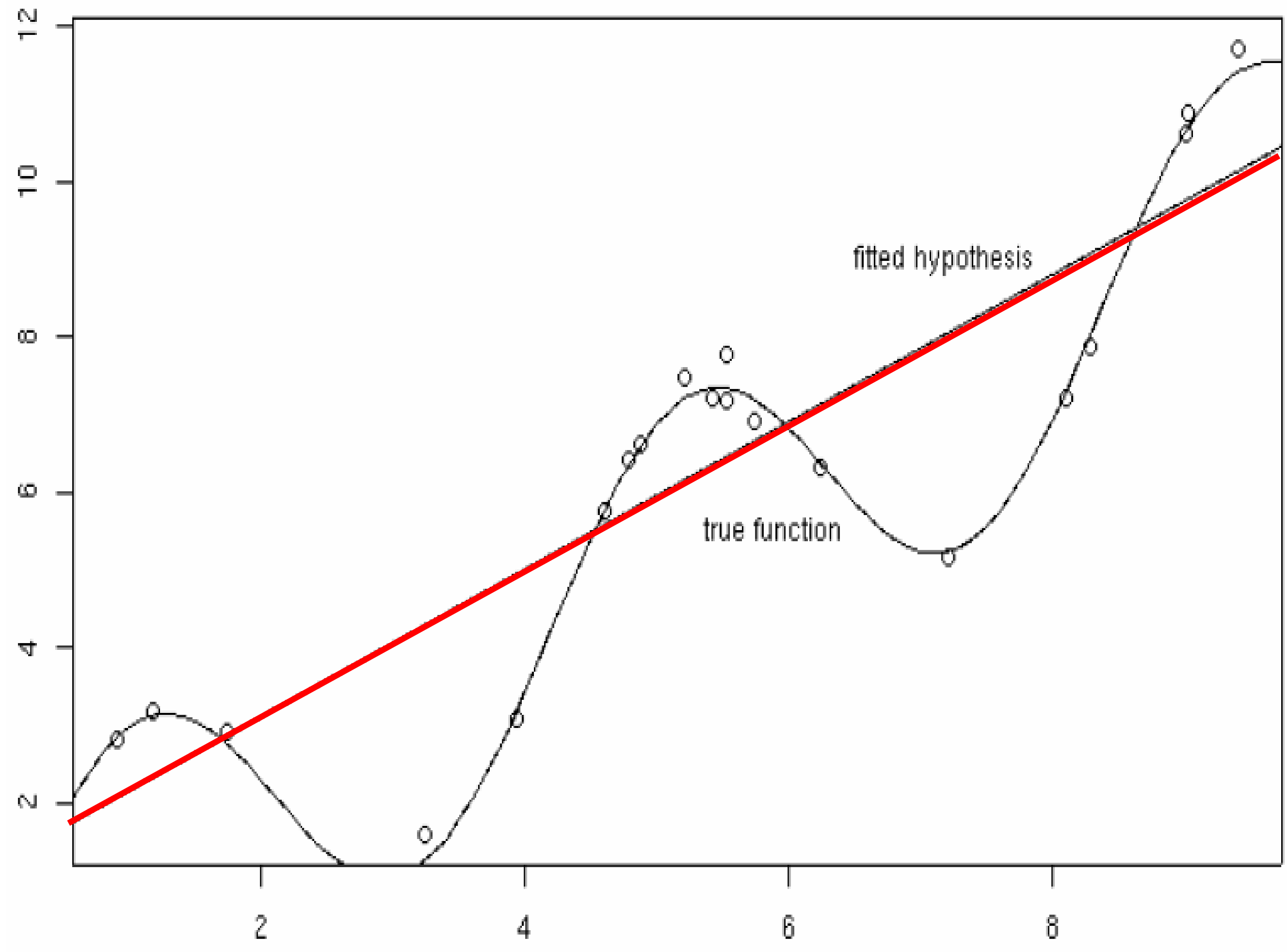
- $Eerr(d) = \int \boxed{[y(\mathbf{x}) - t]^2}\, p(\mathbf{x},t)\; dt\; d\mathbf{x}$

$$= \int \{\underbrace{y(\mathbf{x}) - h(\mathbf{x})\}^2\, p(\mathbf{x})\; d\mathbf{x}} \;+\; \int \{\underbrace{h(\mathbf{x}) - t\}^2\, p(\mathbf{x}, t)\; d\mathbf{x}\; dt}$$

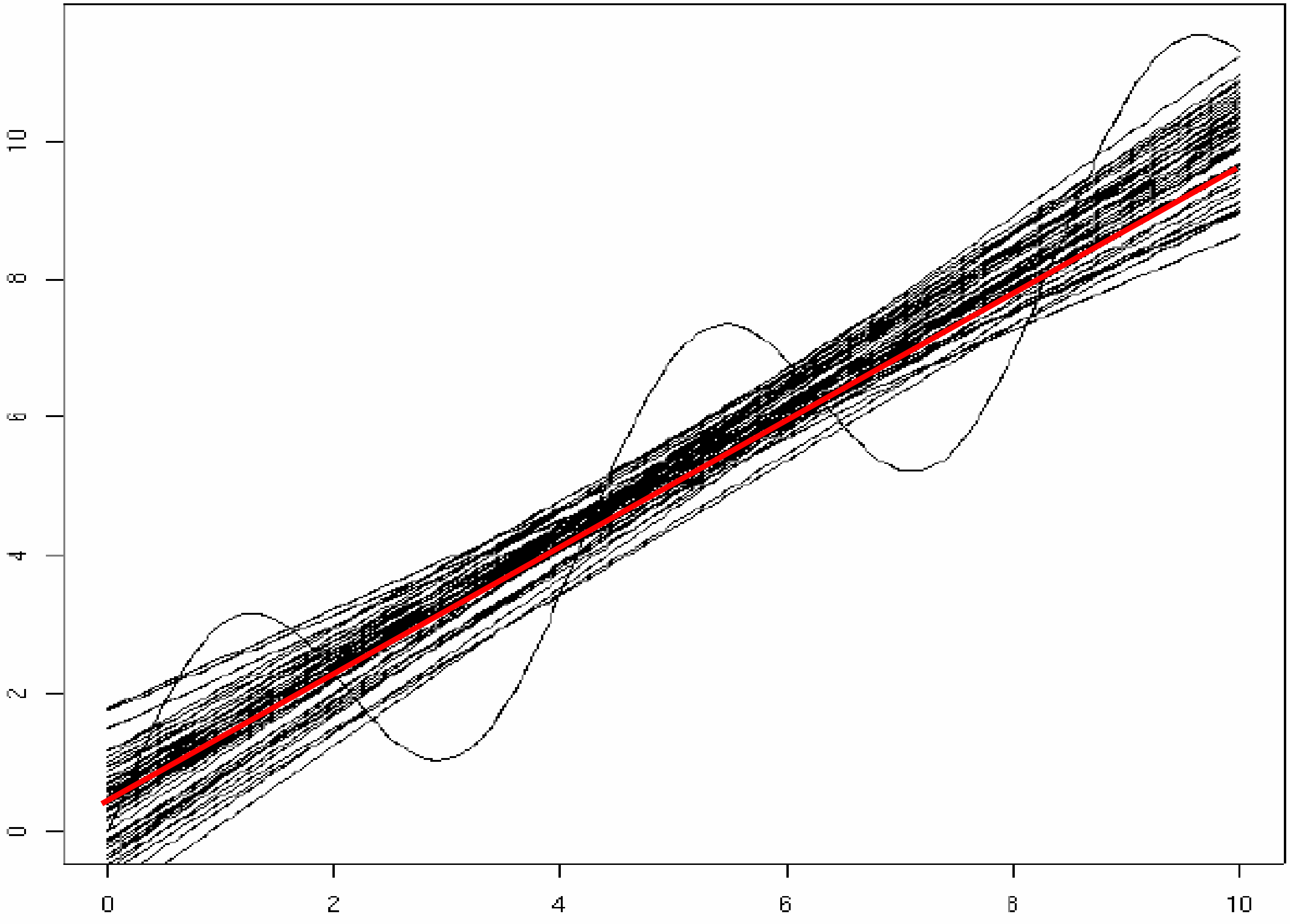
Mismatch between OUR hypothesis y(.) & target h(.)
… we can influence this, as we can change h(.)

Noise in distribution of target
… nothing we can do…

1 fit, based on 20 examples



50 fits, each based on 20 examples



Each dataset d drawn from distribution over datasets $\mathcal{D}$ .. of size 20

Terms

- $\mathbf{x}$ – input variable
  - $\mathbf{x}^*$ – new input variable
- $h(\mathbf{x})$ – "truth" – underlying response function
- $t = t(\mathbf{x}) = h(\mathbf{x}) + \varepsilon$ – actual observed response
- $y(\mathbf{x}; d)$ – predicted response,
  based on model learned from dataset d
- $\hat{y}(\mathbf{x}) = E_{d \sim \mathcal{D}}[\ y(\mathbf{x}; d)\ ]$ – expected response,
  averaged over (models based on) all datasets
- $Eerr = E_{d \sim \mathcal{D},\ \mathbf{x}^*}[\ (h(\mathbf{x}^*) - y(\mathbf{x}^*, d))^2\ ]$
  – expected L_2 error on new instance $\mathbf{x}^*$
  … over all $d \sim \mathcal{D}$

$$Err(d) = \int\{h(\mathbf{x}) - y(\mathbf{x};\, \mathbf{d})\}^2\, p(\mathbf{x})\, d\mathbf{x} \;+\; \cancel{\int\{h(\mathbf{x}) - t\}^2\, p(\mathbf{x},t)\, d\mathbf{x}\, dt}$$

Relevant Part of Loss

- $y(\mathbf{x}) = y(\mathbf{x};\, d)$ is fit to data d…
  so consider *expectation over data sets* $d \sim \mathcal{D}$
  - Let $\hat{y}(\mathbf{x}) = E_{d\sim\mathcal{D}}[y(\mathbf{x};\, d)]$
- $E_{d\sim\mathcal{D}}[\, \{ h(\mathbf{x}) - y(\mathbf{x};\, d) \}^2 \,]$

Does not depend on d
… so pull out from $E_d[\ \ldots]$

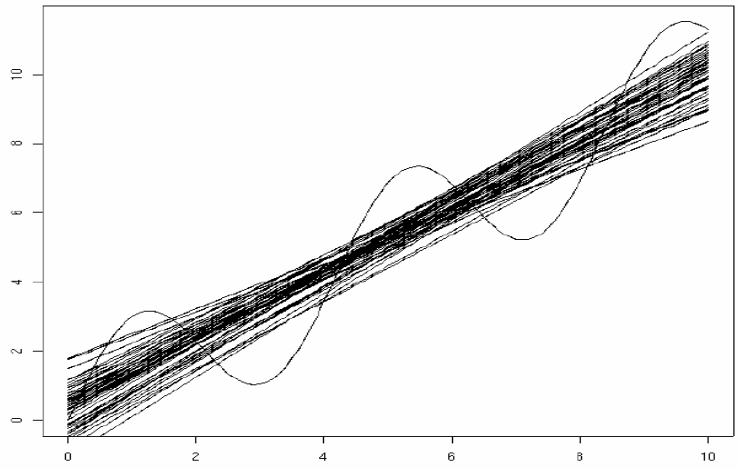
$$= E_{\mathcal{D}}[\, \{ h(\mathbf{x}) - \hat{y}(x) \;+\; \hat{y}(x) - y(\mathbf{x};\, d) \}^2 \,]$$

$$= E_{\mathcal{D}}[\, \boxed{\{ h(\mathbf{x}) - \hat{y}(x)\}^2} \,] \;+\; 2E_{\mathcal{D}}[\, \boxed{\{h(\mathbf{x}) - \hat{y}(x)\}}\, \cancelto{0}{\{\hat{y}(x) - y(\mathbf{x};\, d)\}} \,]$$

$$+\; E_{\mathcal{D}}[\{\, y(\mathbf{x};\, d) - E_{\mathcal{D}}[y(\mathbf{x};\, d)] \,\}^2\,]$$

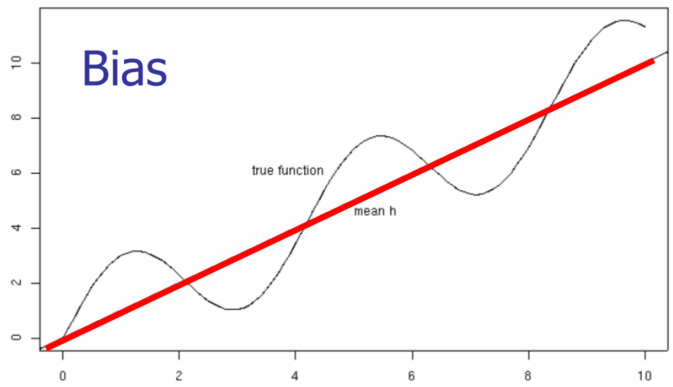
$$= \underbrace{\{h(\mathbf{x}) - \hat{y}(\mathbf{x})\}^2}_{\text{Bias}^2} \;+\; \underbrace{E_{\mathcal{D}}[\, \{ y(\mathbf{x};\, d) - \hat{y}(\mathbf{x}) \}^2 \,]}_{\text{Variance}}$$

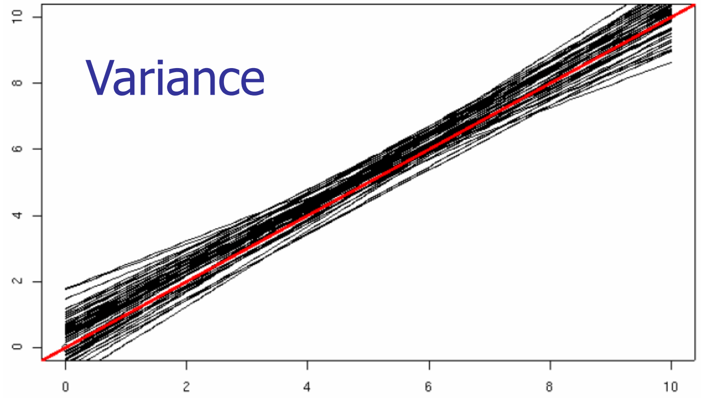
Bias, Variance, Noise

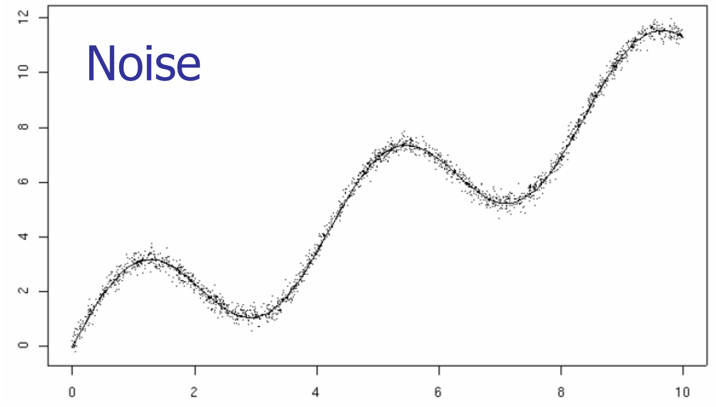


50 fits (20 examples each)

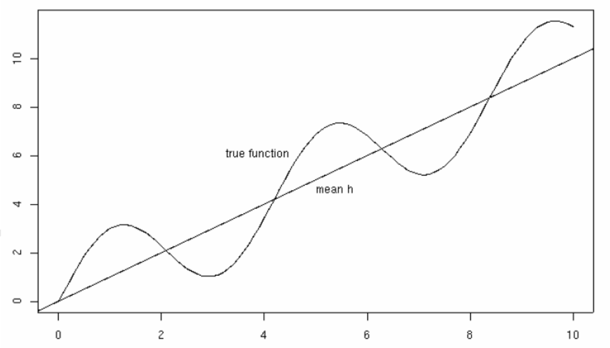
=







Understanding Bias



$$\boxed{\{\ \hat{y}(\mathbf{x}) - h(\mathbf{x})\ \}^2}$$

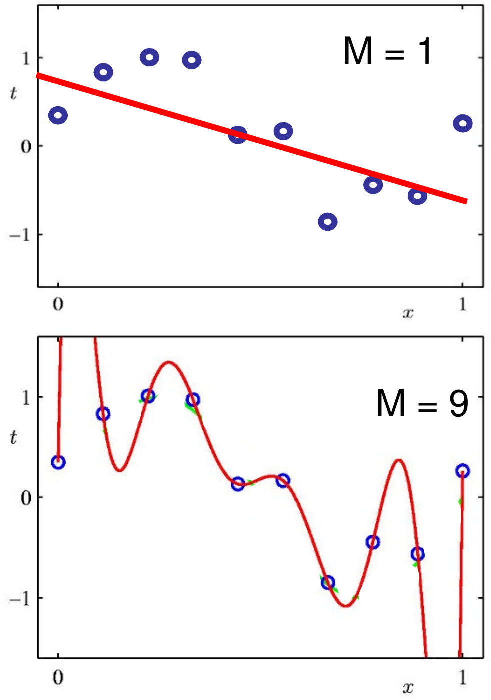
- Measures how well
  our approximation architecture (all of models)
  can fit the data

- Weak approximators
  - (eg, low-degree polynomials)

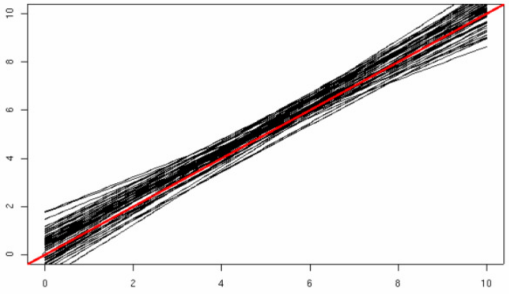
  will have high bias

- Strong approximators
  - (eg, high-degree polynomials)

  will have lower bias



Understanding Variance



$$E_{d\sim\mathcal{D}}[\, \{\, y(\mathbf{x}; d) - \hat{y}(\mathbf{x})\, \}^2\,]$$

- No *direct* dependence on target values
- For a fixed size $|d|$:
  - **Strong** approximators tend to have **more variance**
    … different datasets will lead to DIFFERENT predictors
  - **Weak** approximators tend to have **less variance**
    … slightly different datasets may lead to SIMILAR predictors
- Variance will typically disappear as $|d| \rightarrow \infty$

Summary of Bias,Variance,Noise

- $\text{Eerr} \;=\; E[\,(t^{*} - y(\mathbf{x}^{*}))^{2}\,] \;=$

$$(\hat{y}(\mathbf{x}^{*}) - h(\mathbf{x}^{*}))^{2}$$

$$+\; E[\,(y(\mathbf{x}^{*}) - \hat{y}(\mathbf{x}^{*}))^{2}\,]$$

$$+\; E[\,(t^{*} - h(\mathbf{x}^{*}))^{2}\,]$$

$$= \text{Bias}(\;h(\cdot)\;)^{2} + \text{Var}(\;h(\cdot)\;) + \text{Noise}$$

Expected prediction error

= Bias^{2} + Variance + Noise

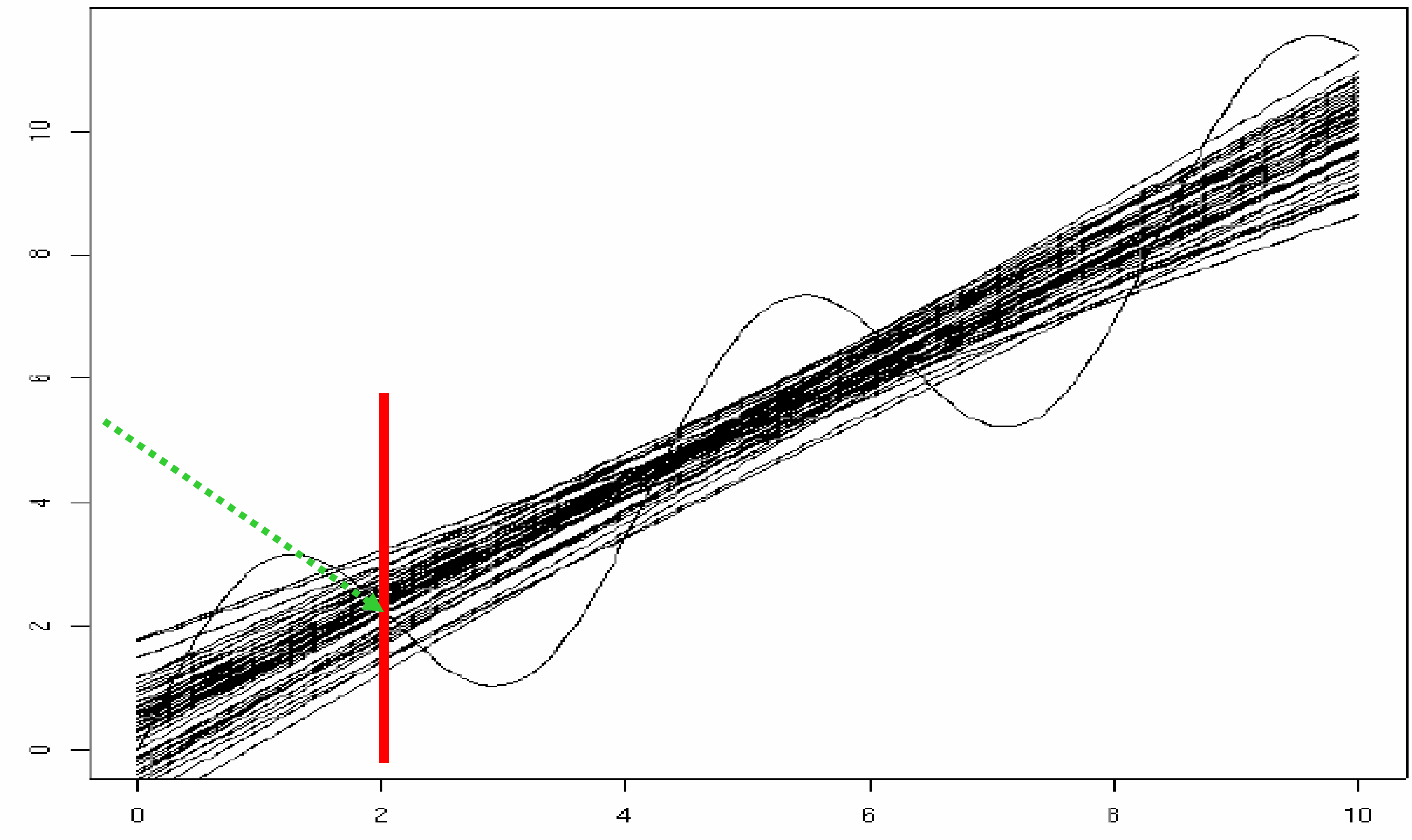
Bias, Variance, and Noise

- **Bias**: $\hat{y}(\mathbf{x}^*)- h(\mathbf{x}^*)$
  - the best error of model $\hat{y}(x^*)$ [average over datasets]

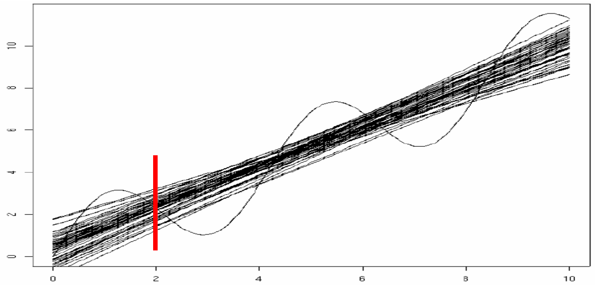
- **Variance**: $E_{d\sim\mathcal{D}}[\ (\ y_d(\mathbf{x}^*) - \hat{y}(\mathbf{x}^*)\ )^2\ ]$
  - How much $y_d(x^*)$ varies from one training set d to another

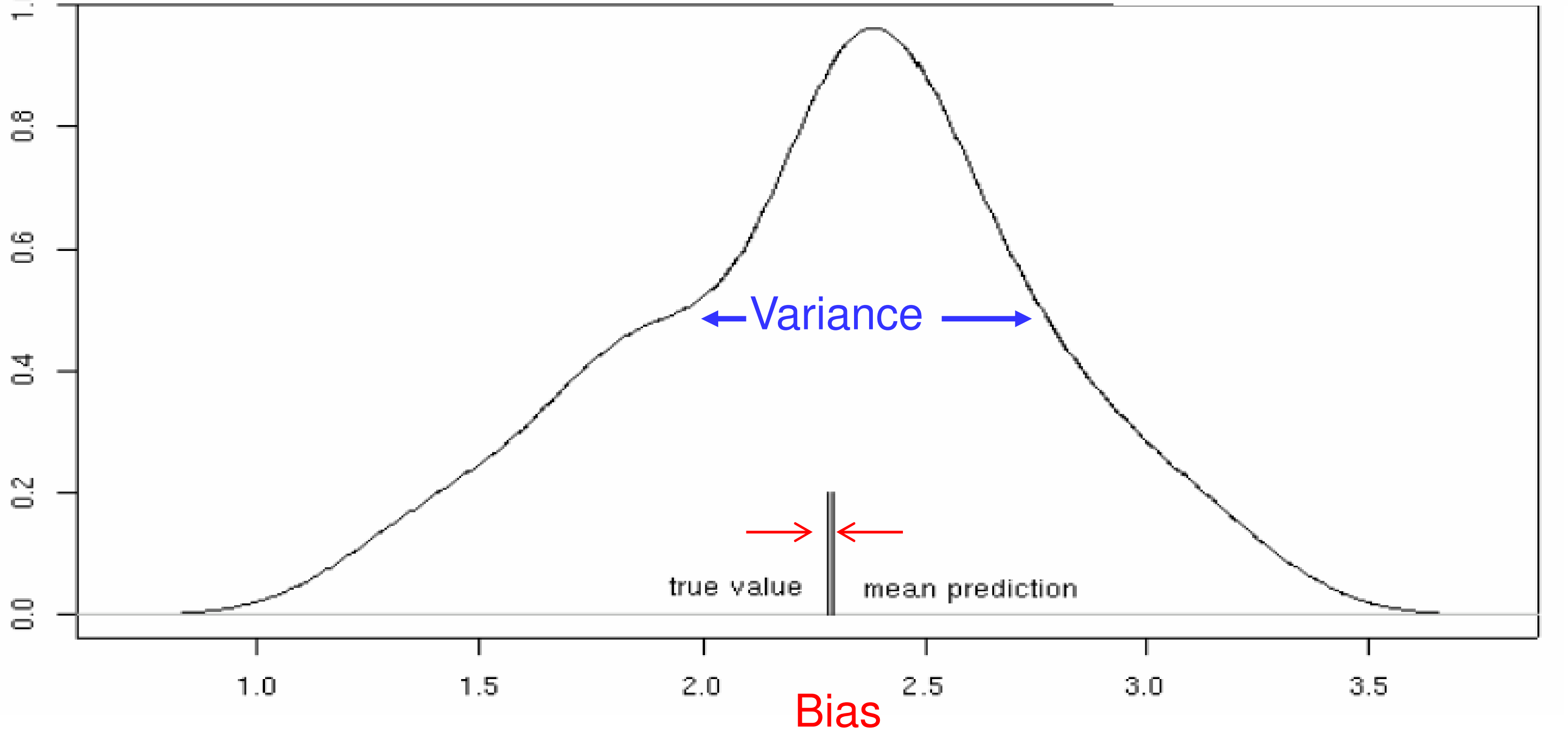
- **Noise**: $E[\ (t^* - h(\mathbf{x}^*))^2\ ] = E[\varepsilon^2] = \sigma^2$
  - How much t^* varies from $h(\mathbf{x}^*) = t^* + \varepsilon$
  - Unavoidable error,
    even given PERFECT model, and ∞ data

50 fits (20 examples each)

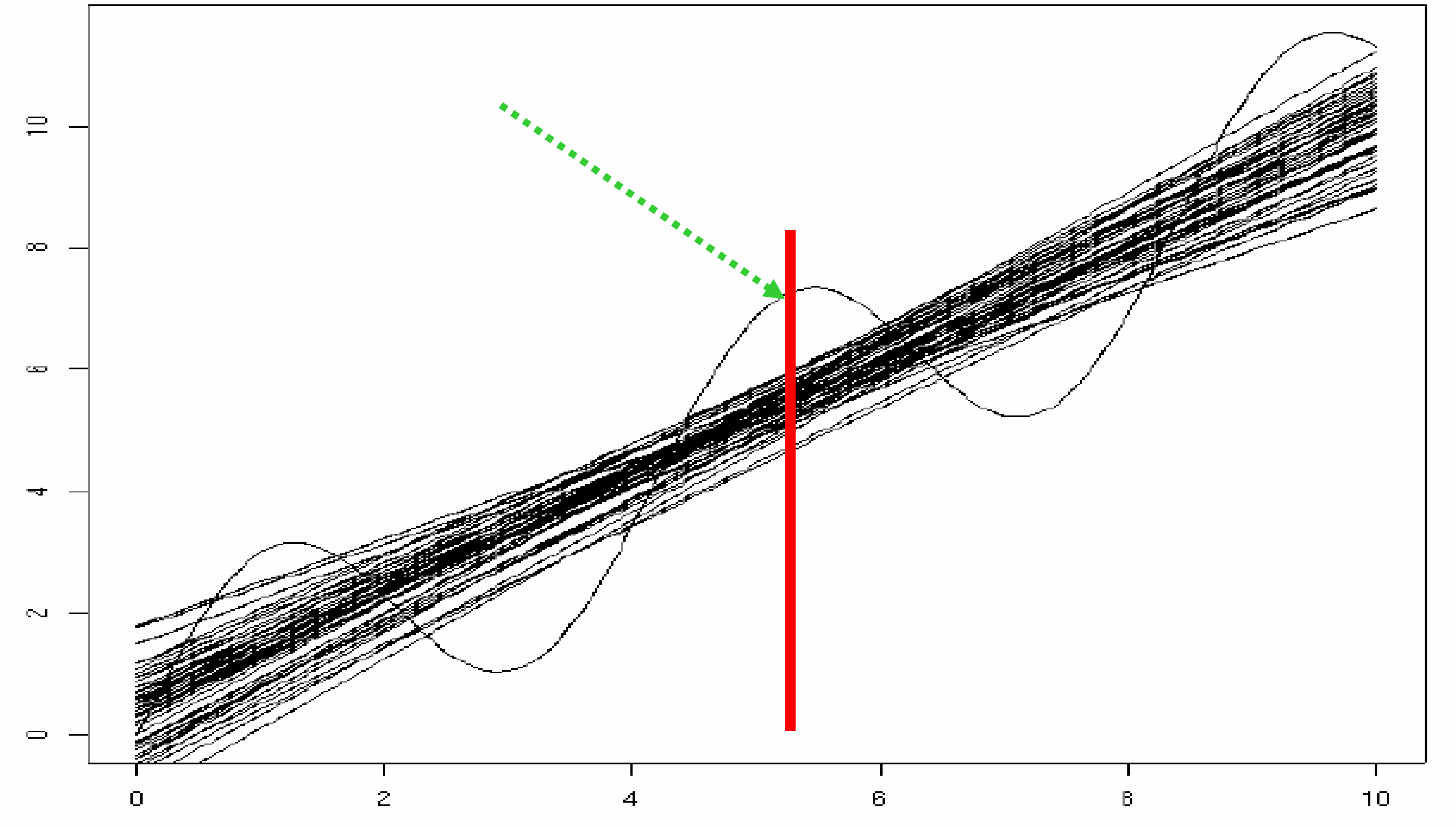


Predictions at x=2.3

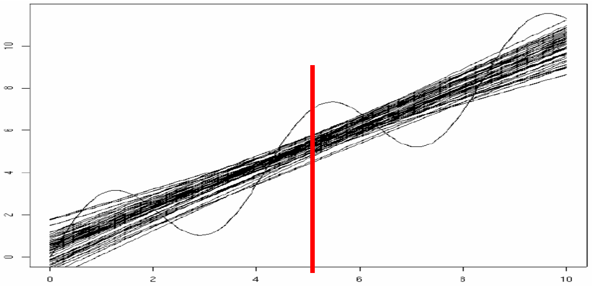


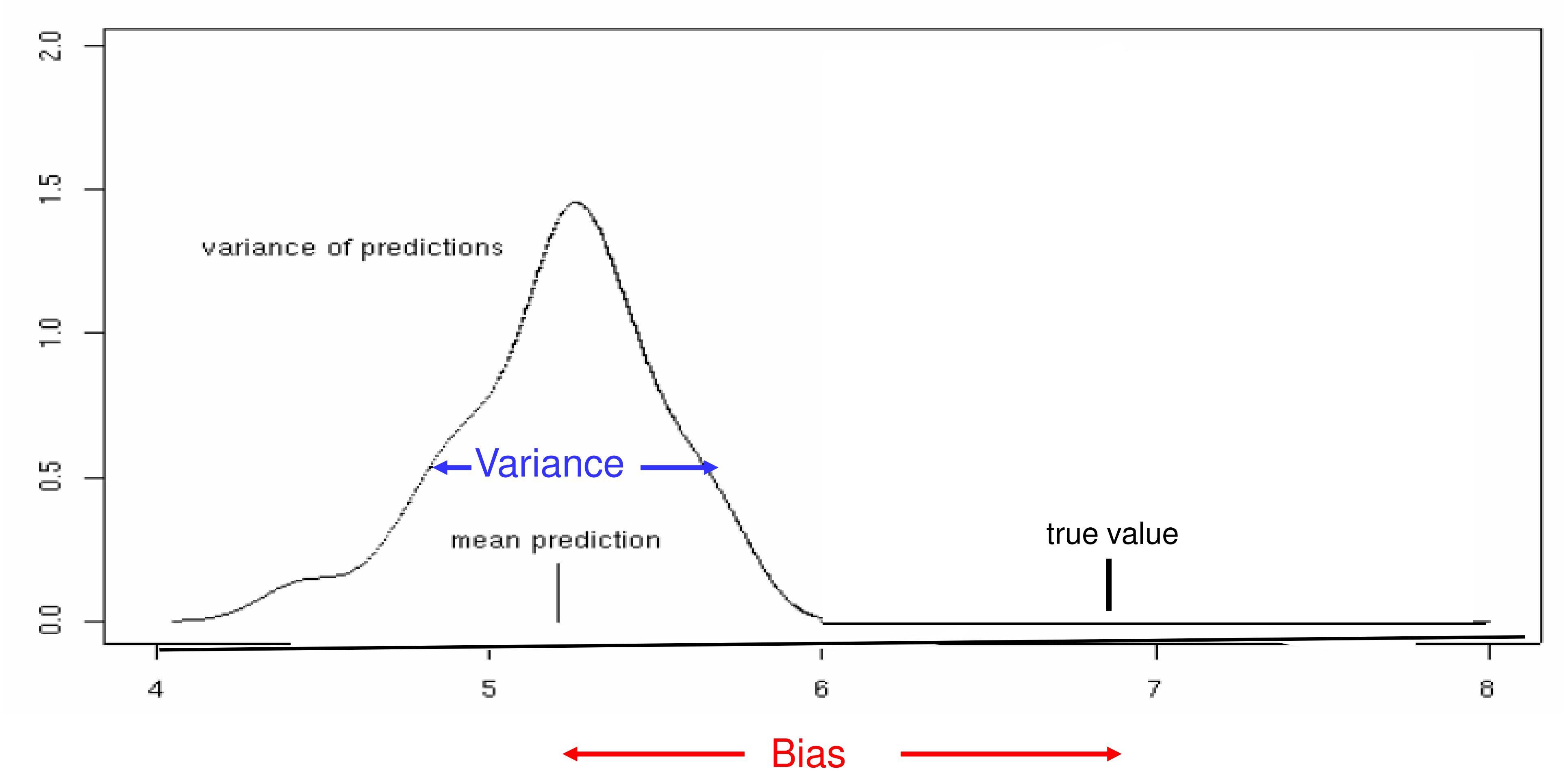


50 fits (20 examples each)

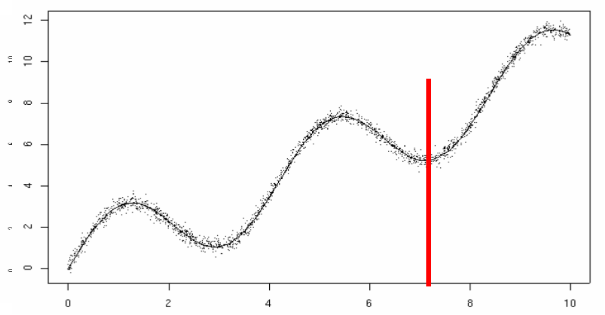


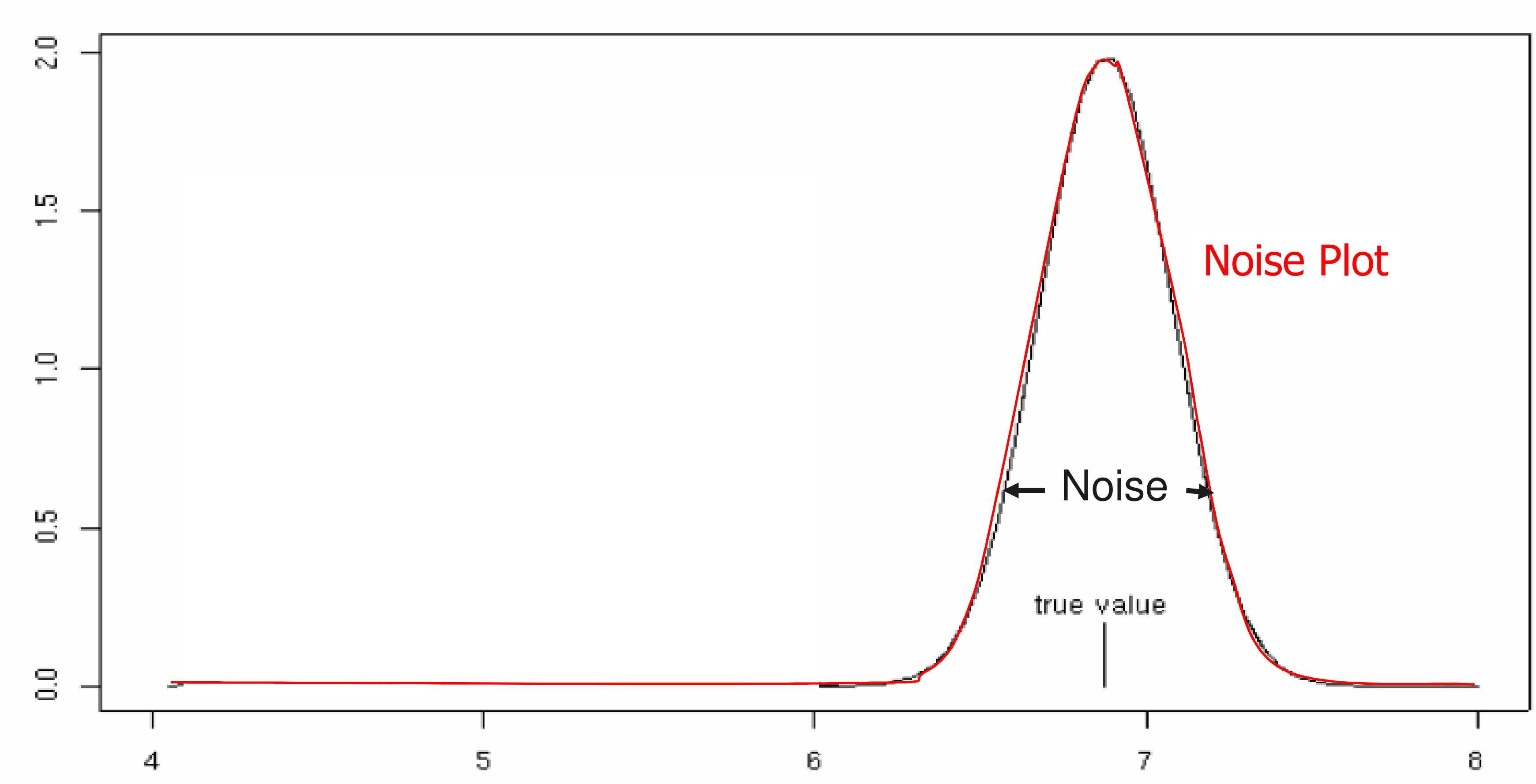
Predictions at x=5.2



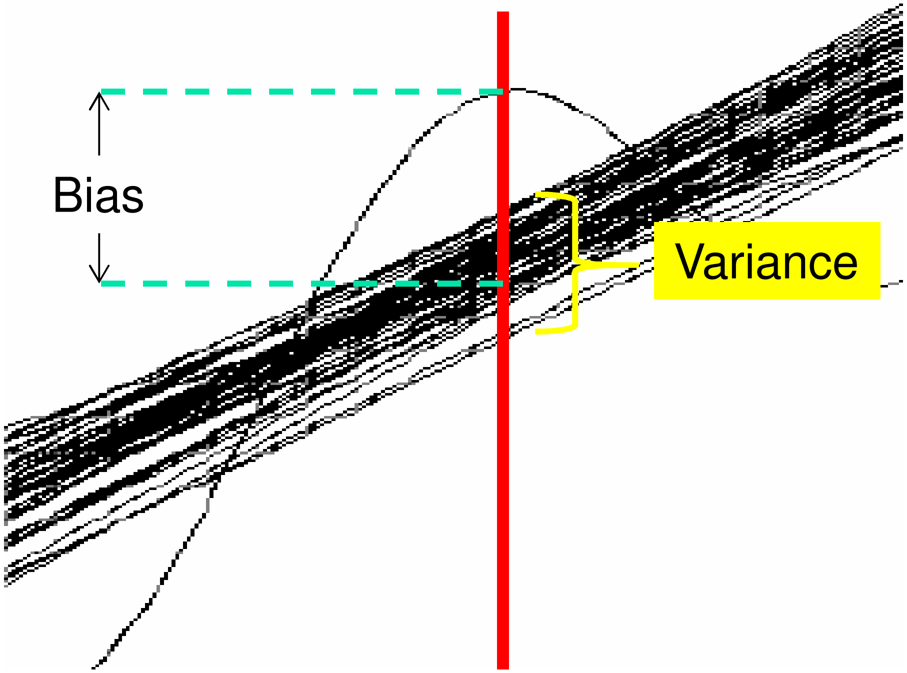


y-values observed at x=6.8

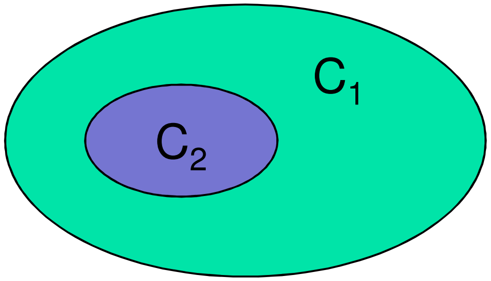




Bias and Variance



Model Selection: Bias-Variance



- C_1 "more expressive than" C_2

  iff

  representable in $C_2 \Rightarrow$ representable in C_1

  iff

  "$C_2 \subset C_1$"

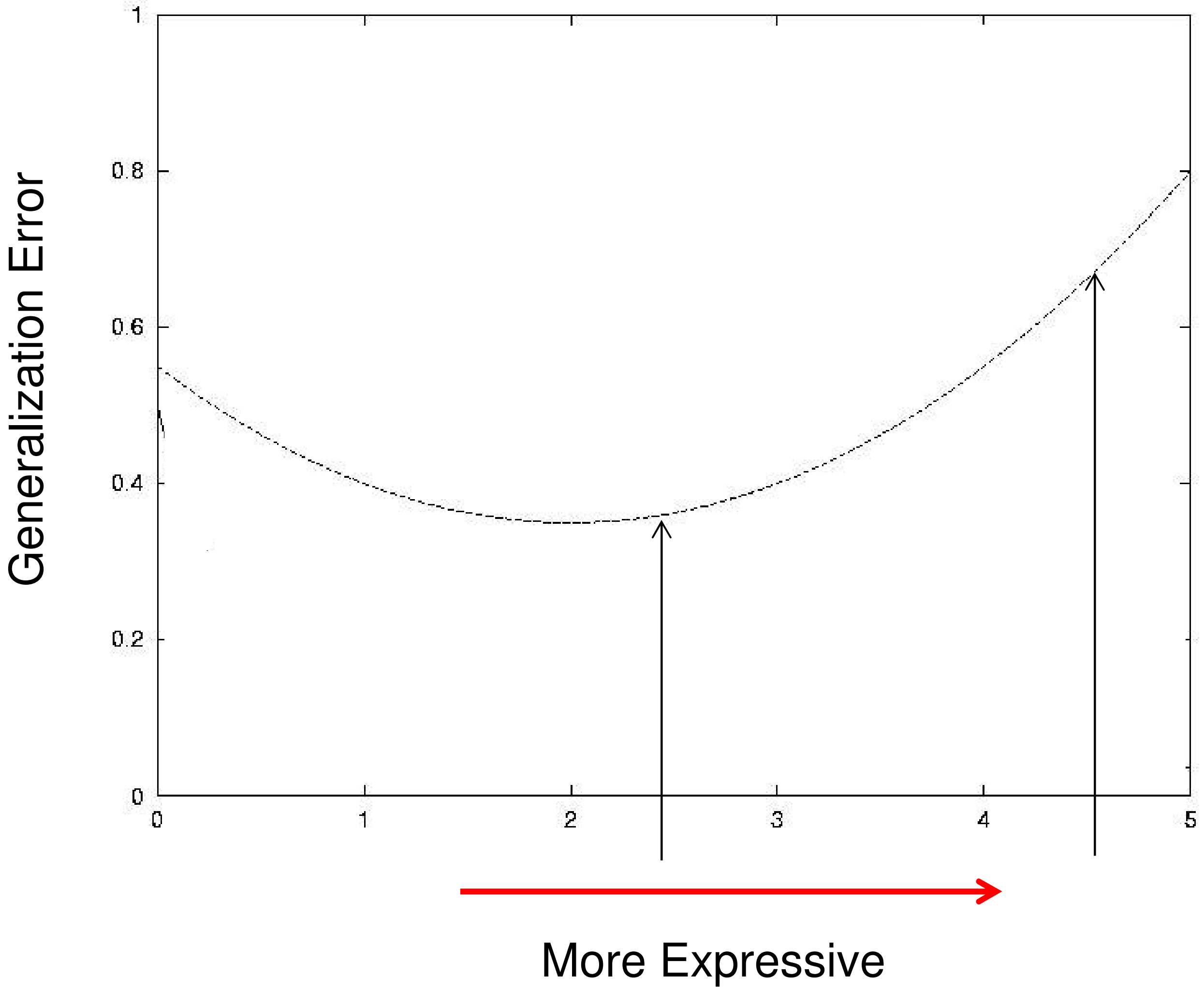
- Eg, LinearFns $\subset$ QuadraticFns

  0-HiddenLayerNNs $\subset$ 1-HiddenLayerNNs

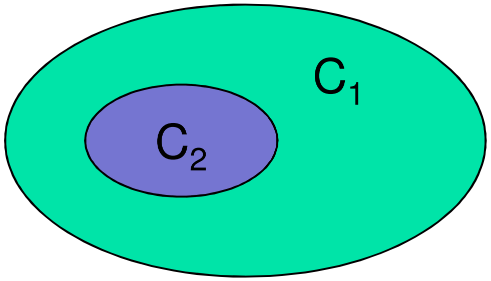
$\Rightarrow$ can ALWAYs get better fit using C_1, over C_2

- But … sometimes better to look for $y \in C_2$

Standard Plot…



Why?



- $C_2 \subset C_1 \Rightarrow$
  $\forall\, y \in C_2$
  $\exists\, x^* \in C_1$ that is at-least-as-good-as y

- But given *limited sample,* might not find this best x^*

- Approach: consider Bias^2 + Variance!!

Bias-Variance tradeoff – Intuition

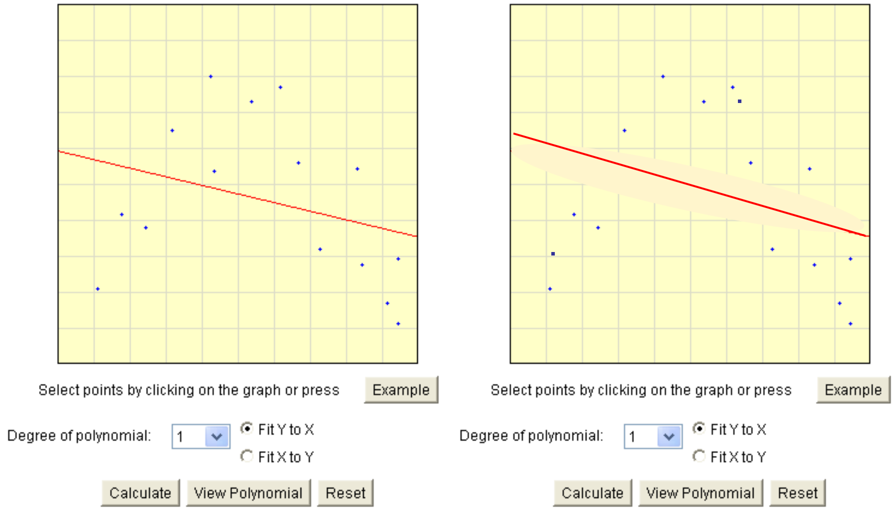
- Model too "simple" ⇒ does *not* fit the data well
  - A biased solution

- Model too "complex" ⇒ small changes to the data, changes predictor a lot
  - A high-variance solution

Bias-Variance Tradeoff

- Choice of hypothesis class introduces learning bias
  - Simple class ⇒ high bias
  - Complex class ⇒ high variance

Degree 1: High Bias, Low Variance



Select points by clicking on the graph or press

Degree of polynomial: 1

● Fit Y to X

○ Fit X to Y

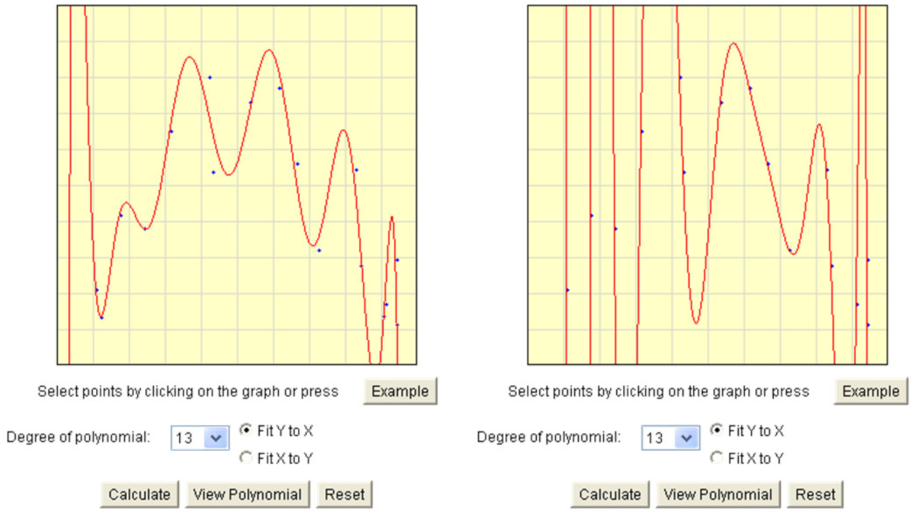
Select points by clicking on the graph or press

Degree of polynomial: 1

● Fit Y to X

○ Fit X to Y

Degree 13: Low Bias, High Variance



Select points by clicking on the graph or press

Degree of polynomial: 13

● Fit Y to X

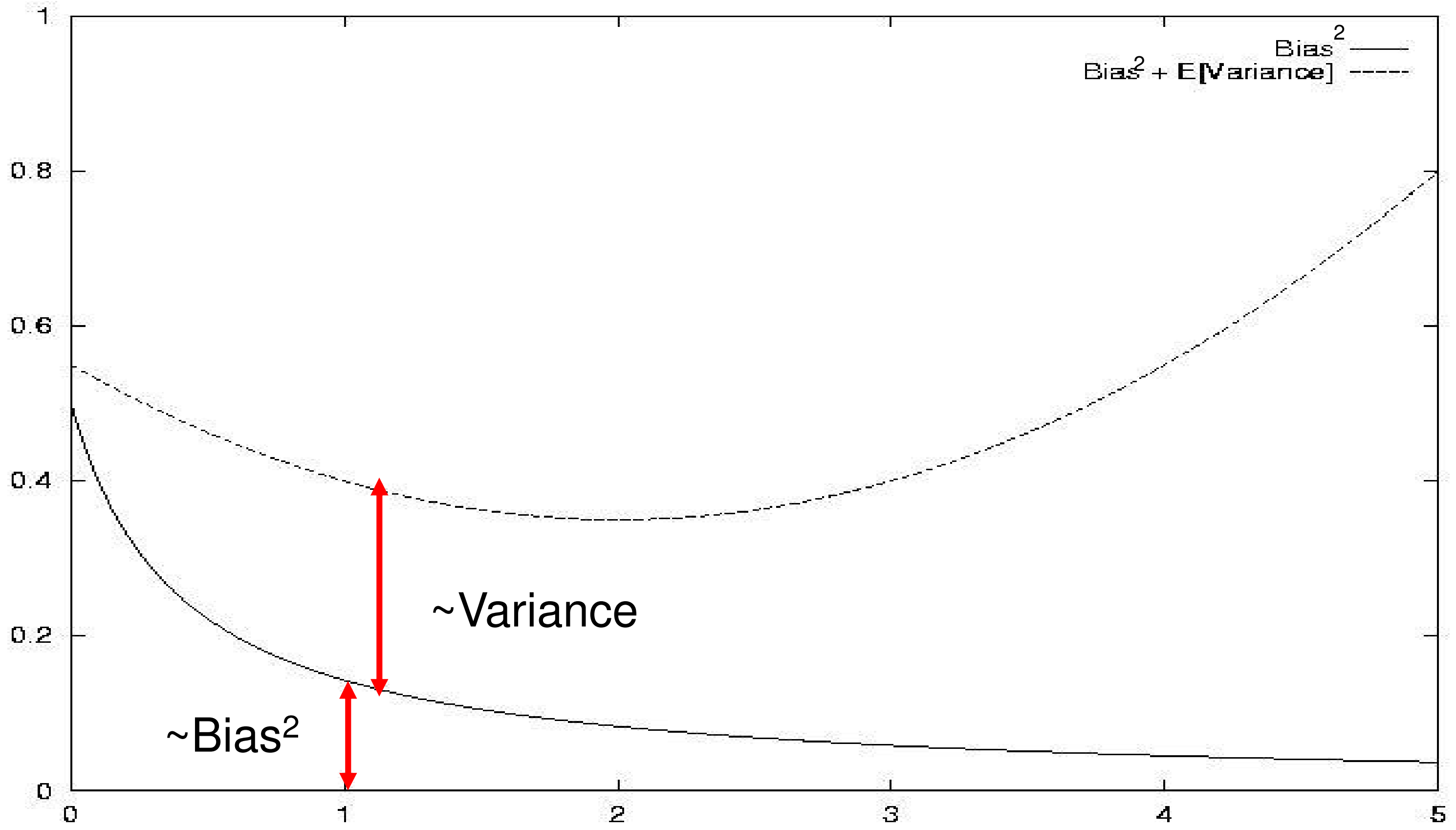
○ Fit X to Y

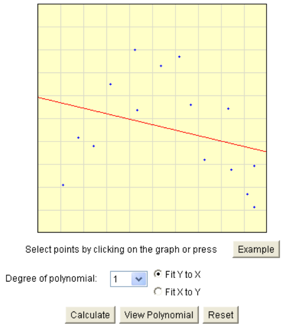
Select points by clicking on the graph or press

Degree of polynomial: 13

● Fit Y to X

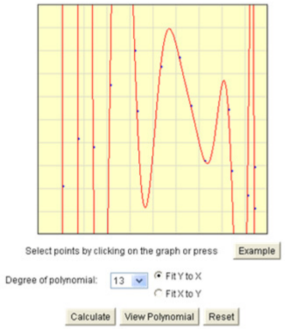
○ Fit X to Y




Select points by clicking on the graph or press Example

Degree of polynomial: 1

● Fit Y to X
○ Fit X to Y


Select points by clicking on the graph or press Example

Degree of polynomial: 13

● Fit Y to X
○ Fit X to Y

Effect of Algorithm Parameters on Bias and Variance

Skip Estimating B & V

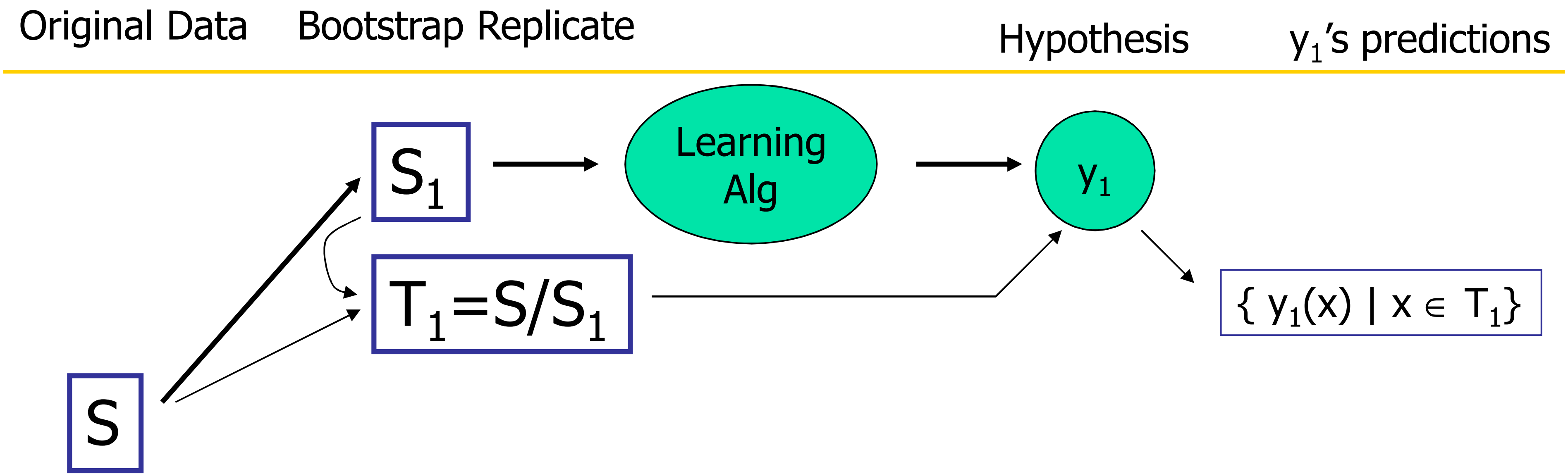
- linear regressors over r features
  - increasing number of features r reduces bias and increases variance
  - increasing range of values for parameters { θ_i } reduces bias and increases variance
- k-nearest neighbor:
  - increasing k typically increases bias and reduces variance
- decision trees of depth d:
  - increasing d typically reduces bias and increases variance
- RBF SVM with parameter σ:
  - increasing σ typically increases bias and reduces variance

Estimating Bias and Variance

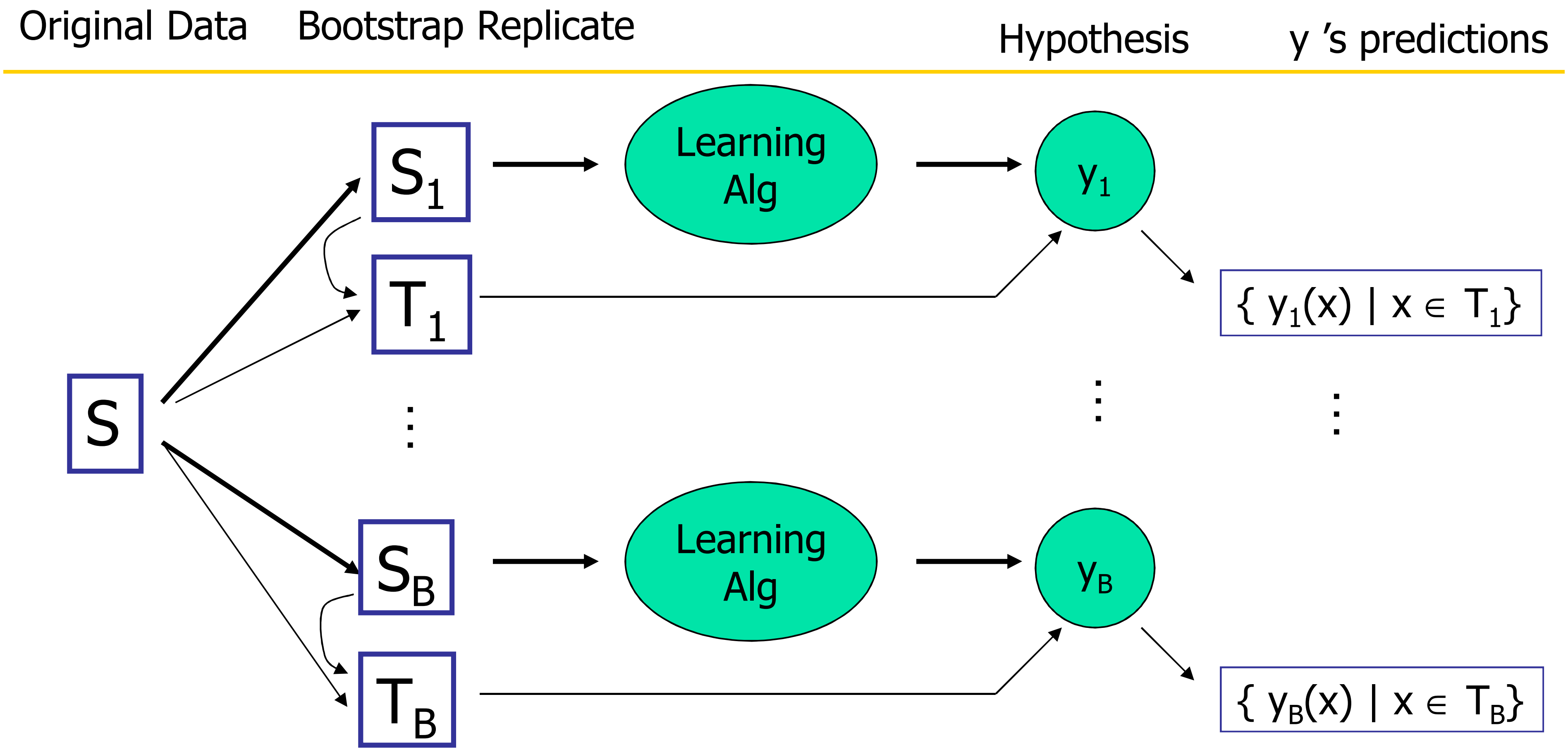
- In practice (unlike in theory), only *ONE training set* d
- Simulate multiple training sets by bootstrap replicates
  - d’ = {x | x is drawn at random, with replacement, from d }
  - |d’| = |d|
  - E[| d’ ∩ d |] / |d| $\approx 1 - \frac{1}{e} \approx 0.632$

$$\lim \left(1 - \frac{1}{n}\right)^n = \frac{1}{e}$$

Estimating Bias / Variance



Estimating Bias / Variance



Original Data

Bootstrap Replicate

Hypothesis

y 's predictions

S

S_1

T_1

S_B

T_B

Learning Alg

Learning Alg

y_1

y_B

$\{ y_1(x) \mid x \in T_1 \}$

$\{ y_B(x) \mid x \in T_B \}$

- Each S_i is bootstrap replicate
- $T_i = S / S_i$
- y_i = hypothesis, based on S_i

Average Response for each x_i

	x_1	…	x_r
$\in^? T_1$	$y_1(x_1)$	…	
$\in^? T_2$	--	…	$y_2(x_r)$
⋮			
$\in^? T_B$	$y_B(x_1)$	…	$y_B(x_r)$

$\hat{y}(x_1) = \frac{1}{k_1}\sum_i y_i(x_1)$ … $\hat{y}(x_r) = \frac{1}{k_r}\sum_i y_i(x_r)$

$$\hat{y}(x_j) = \frac{1}{|\{\, i : x \in T_i \}|} \sum_{\{i : x \in T_i\}} y_i(x_j)$$

Procedure for Measuring Bias and Variance

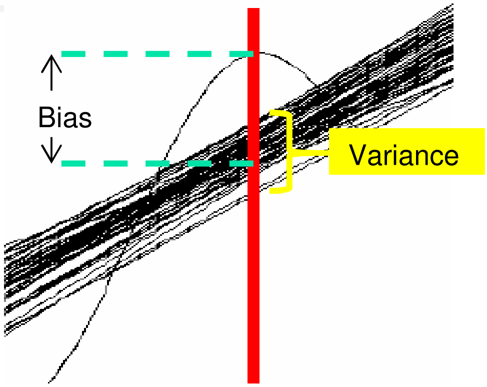
- Construct B bootstrap replicates of S :

  $S_1, \ldots, S_B$
- Apply learning alg to each replicate S_b to obtain hypothesis y_b
- Let $T_b = S \setminus S_b$ = data points not in S_b (*out of bag* points)
- Compute predicted value

  $y_b(x)$

  for each $x \in T_b$

Estimating Bias and Variance



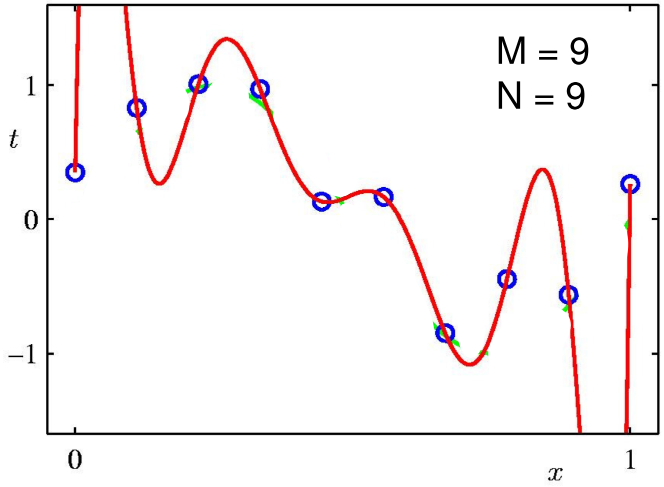
- For each $x \in S$,
  - observed response $t = t(x)$
  - predictions $y_1, \ldots, y_k$
- Compute average prediction $\hat{y}(x) = \text{ave}_i \{ y_i \}$
- Estimate bias: $\hat{y}(x) - t(x)$
- Estimate variance:

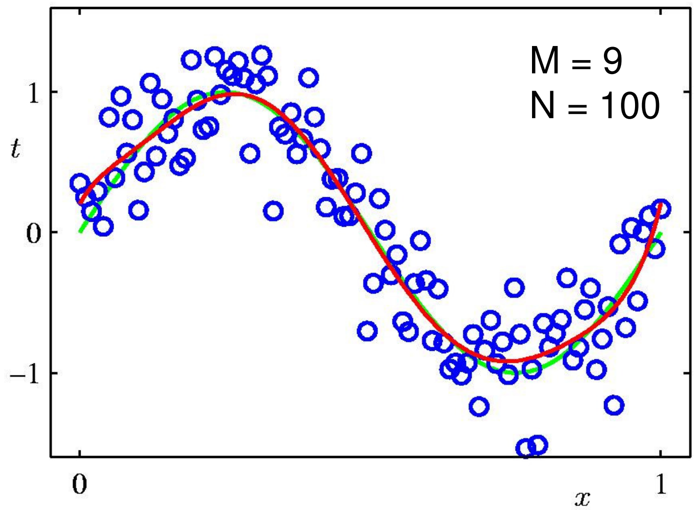
$$\frac{1}{|\{ i{:}x{\in}T_i\}|-1} \sum_{\{i:x\in T_i\}} (\,\hat{y}(x) \; - y_i(x))^2$$

- Consider Average Bias, Average Variance
- Assume noise is 0

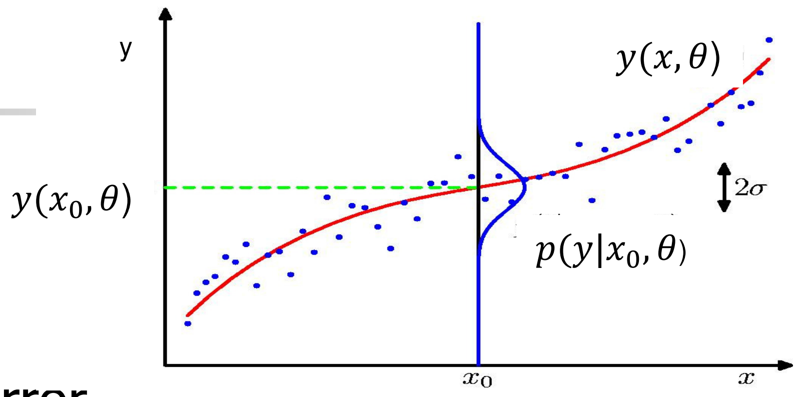
Sample Size

- This analysis considered FIXED sample size
  - N = 15
- Overfitting is less problematic, as #datapoints increases…





Outline

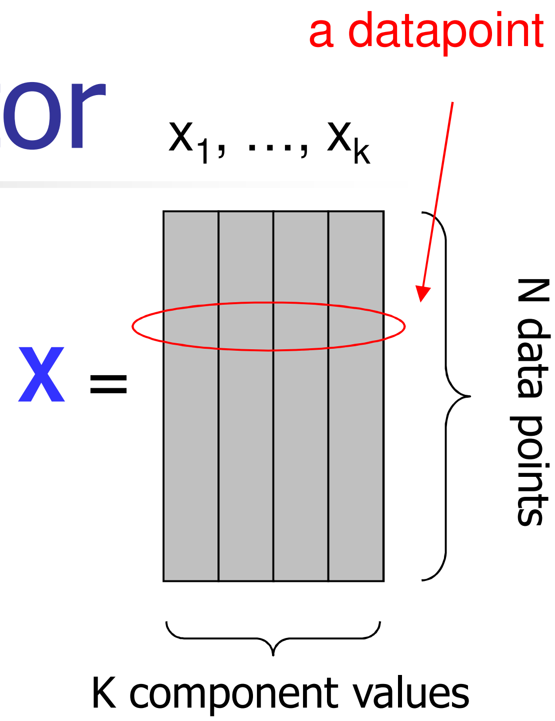


- Linear Regression
- Evaluating Predictors
  - Training set error vs Test set error
  - Cross Validation
- Overfitting
  - Bias-Variance analysis
  - Feature Selection
  - L2 Regularization
  - Setting parameters … internal C-V
  - Bayesian Model
  - L1 Regularization (Lasso)
- Linear Classification

Least Squares Estimator

- If truth: $f(x) = x^T\theta$

  Observed: $y = f(x) + \varepsilon;\ E[\,\varepsilon\,] = 0$



- Least squares estimator

  $$\hat{f}(x_0) = x_0^T\hat{\theta} \qquad \hat{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

  - $\hat{f}(\cdot)$ unbiased iff $f(x_0) = E[\hat{f}(x_0)\,]$

$$f(x_0) - E[\hat{f}(x_0)\,]$$

$$= x_0^T\theta - E[\, x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T \quad \mathbf{y} \qquad]$$

$$= x_0^T\theta - E[\, x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\theta + \varepsilon)\,]$$

$$= x_0^T\theta - E[\, x_0^T\cancel{(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{X})}\,\theta + x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\varepsilon\,]$$

$$= \boxed{x_0^T\theta \;-\; x_0^T\theta} + x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\,\cancel{E[\varepsilon]} \;=\; 0$$

Gauss-Markov Theorem

- Least squares estimator $\hat{f}(x) = x^T \boxed{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y}$
  - … is unbiased: $E[\hat{f}(x)] = f(x)$
  - … is linear in x … $\hat{f}(x) = x^T c_0$ where c_0
- Gauss-Markov Theorem:
  OLS (*Ordinary Least Square*) *estimate* has
  the minimum variance
  among all linear unbiased estimators.
  - BLUE: Best Linear Unbiased Estimator
- Interpretation: Let $g(x)$ be any other …
  - unbiased estimator of $f(x)$ … ie, $E[\, g(x)\,] = f(x)$
  - that is linear in x … ie, $g(x) = x^T c$

  then $\mathrm{Var}[\hat{f}(\cdot)] \leq \mathrm{Var}[\, g(\cdot)\,]$

However…

- Gauss-Markov Theorem:

  OLS (*Ordinary Least Square*) *estimate* has
  the minimum variance
  among all linear unbiased estimators.

- However, there may be a **biased** estimator with lower Mean Square Error

$$MSE(\hat{\theta}) = E(\hat{\theta} - \theta)^2$$

$$= \mathrm{Var}(\hat{\theta}) + \overbrace{\left[E(\hat{\theta}) - \theta\right]^2}^{\text{this is 0 for OLS}}$$

Increase Bias (>0), to reduce Variance!

Better Linear Model… reduce MSE

- Bias–variance trade off:
  - Goal: choose a model to minimize MSError

$$\boxed{MSE(\hat{\theta}) = \mathrm{Var}(\hat{\theta}) + \mathrm{Bias}(\hat{\theta})^2}$$

  - Method: sacrifice a little bit of bias to reduce the variance

- Feature selection (remove variables)

  … variance is O(p)
  … so fewer features, reduce variance… but biased…

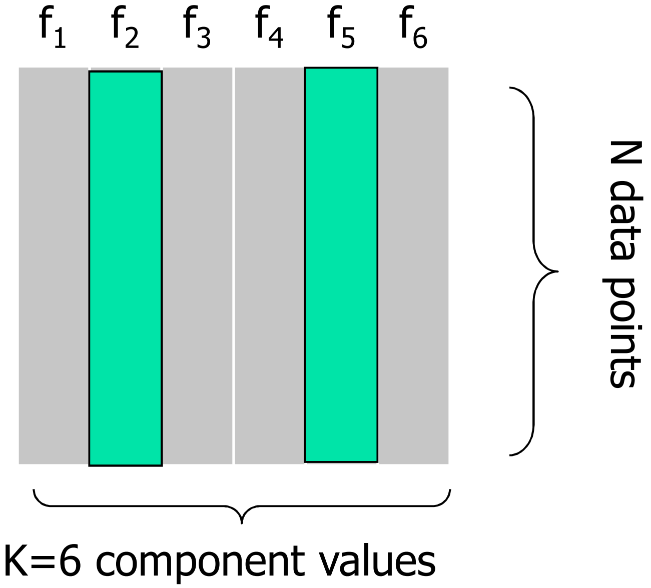
  - Subset selection
  - Forward selection
  - Backward selection
  - (L1-regularization)

- Reduce "range of values" of parameters
  - Shrinkage methods (L2-regularization)

Notation

- Set of all features F
- Given SUBSET of indices $\mathrm{R} \subset \{1,2,\ldots,|F|\}$
  - $X_R \subset X$ subset of data involving just R
  - $\theta_R \subset \theta$ subset of parameters involving just R

X =



$$R = \{\,2,5\,\}$$
$$X_R = \{\,X_2, X_5\}$$
$$\Theta_R = \{\,\theta_2, \theta_5\,\}$$

Subset selection

- Goal: to eliminate unnecessary variables from the model.
- Three approaches:
  - Best subset regression
    - Choose subset of size *k* that gives lowest MSE
  - Forward stepwise selection
    - Sequentially add “best” features
  - Backward stepwise selection
    - Sequentially remove “worst” features

Greedy techniques – not guaranteed to find the best model

Best Subset Regression

- For each $k \in \{0, 1, ..., p\}$,
  find subset of size k that gives smallest MSE (validation)

- *Leaps and bounds* procedure works with $p \leq 40$

- How to choose k ?
  Choose model that minimizes prediction error

- For larger p, searching through all subsets is not feasible.
  Instead: seek a good path through subsets...

Forward Stepwise Selection

$$\boxed{m_R = MSE(\,\theta_R\,) = (t - X_R\,\theta_R)^T\,(t - X_R\theta_R)}$$

- Sequential Greedy Method:
  - Start with trivial model: $R = \{\}$
  - Sequentially include "best" variable f
    - $R \leftarrow R + f$
  - Stop when no new variable improves fit significantly
    ... and use that final R
- "Best"??
  - Highest correlation with the residual error?
    - $f = argmax_j\,\{\,|\,correlation(\,X_j\,, m_R)|\,\}$
  - "F test": $argmax_f\,F(f) = \frac{MSE(\theta_R) - MSE(\theta_{R+f})}{\frac{MSE(\theta_{R+f})}{n-k-2}}$

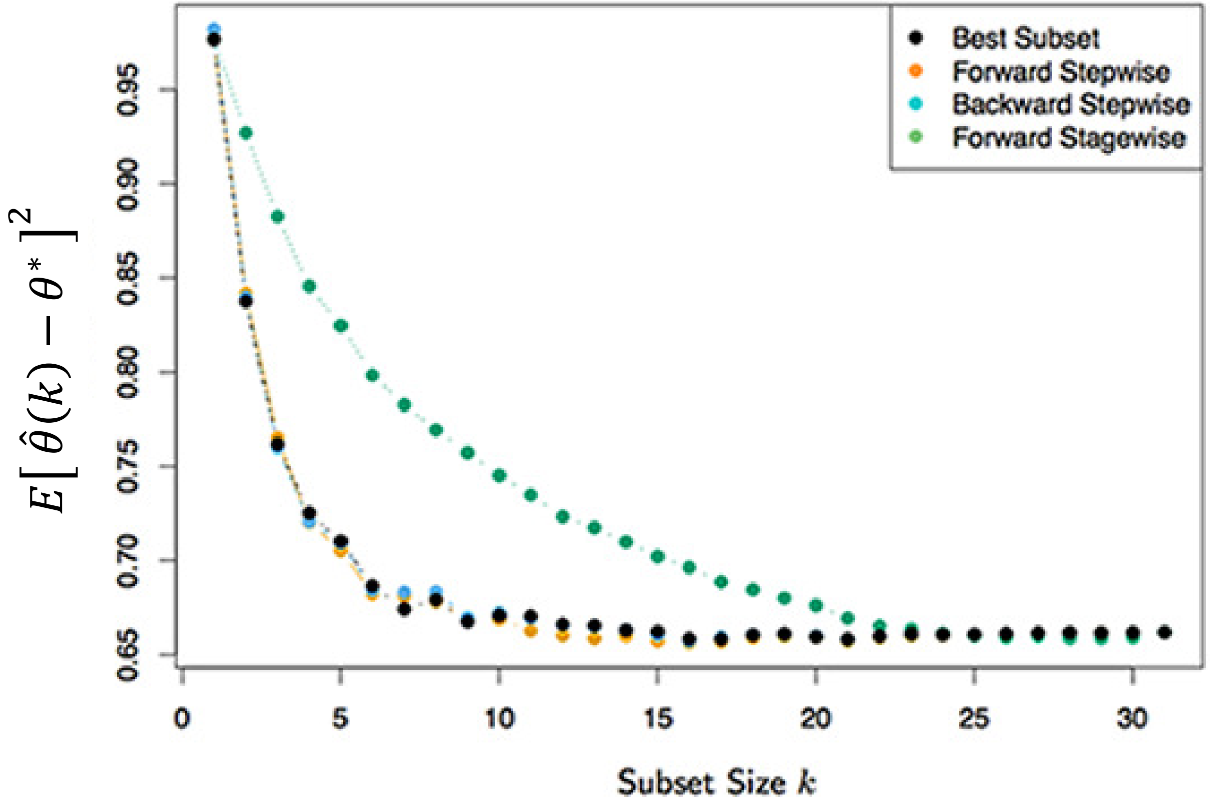
Backward Stepwise selection

- Method:
  - Start with *full* model $R = \{1, \ldots, |\mathrm{F}|\}$
  - Sequentially *delete* feature f that produces the *smallest* value ...
    - ... of correlation
    - ... of F statistic
  - Stop when no significant gain

- ∃ hybrids between
  forward and backward stepwise selection

Comparison

- Which is best… depends …
- One example…



Feature selection:

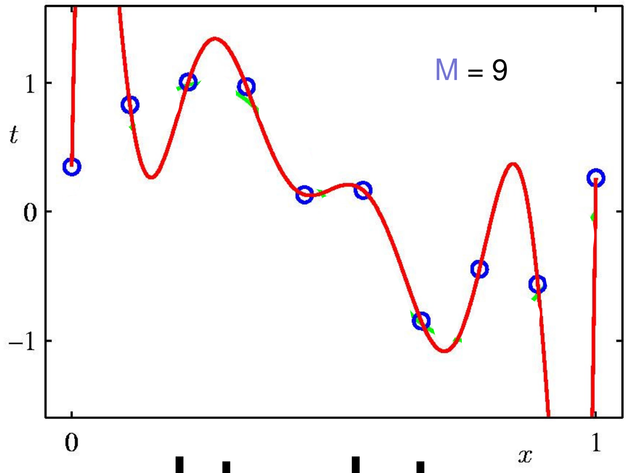
- Feature Selection can help to...
  1. ... improve predictive accuracy
  2. ... help explain the classifier (?)
  3. … reduce cost of features (+ time) at performance time
  4. ... reduce training time
- Here… just #1:
  Can consider other methods

Shrinkage methods

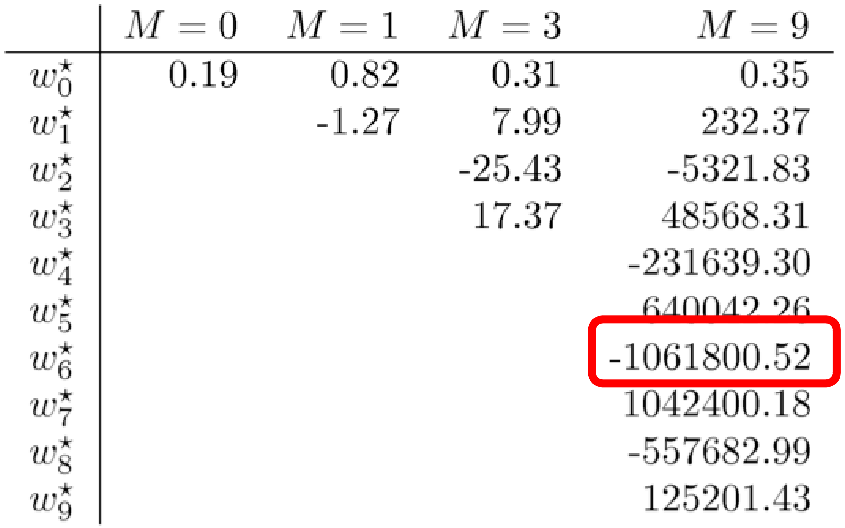
- Use additional penalties/constraints to reduce (range of) parameters
  - Weights
- Shrinkage methods are “more continous” than stepwise selection

  ⇒ do not suffer as much from variability

Coefficient Size…



- At M=9: coefficients are finely tuned to data
  - … too tuned! … matches NOISE in target values
- Why bad?
  - … between data points, function goes crazy
- As M increases, the magnitude of coefficients gets larger



	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Why should regularization help?

- Extreme curves typically require extreme parameter values
  - … eg coefficients for polynomials
- So to avoid *extreme curves*, avoid *extreme values*, θ_i
- Approach:
  - $\hat{\boldsymbol{\Theta}}^{ridge} = \arg\min_{\boldsymbol{\theta}} \sum_i \left(y\left(\mathbf{x}^{(\mathrm{i})}; \boldsymbol{\theta} \right) - \mathrm{t_i} \right)^2$
  - Subject to $|\boldsymbol{\theta}|_2{}^2 \leq \boldsymbol{s}$
- Lagrange: $\arg\min_{\boldsymbol{\theta}} \sum_i \left(y\left(\mathbf{x}^{(\mathrm{i})}; \boldsymbol{\theta} \right) - \mathrm{t_i} \right)^2 + \boldsymbol{\lambda} \, |\boldsymbol{\theta}|_2{}^2$

One-to-one correspondence between $\boldsymbol{s}$ and λ

Regularization

- **Idea**: Penalize overly-complicated answers
- Regular regression minimizes:

$$\sum_i \left(y(\mathbf{x}^{(i)}; \theta) - t_i \right)^2$$

- Regularized regression minimizes:

$$\sum_i \left(y(\mathbf{x}^{(i)}; \theta) - t_i \right)^2 + \boxed{\lambda \, |\theta|_2^2}$$

Solving Regularized Form

$$Solving\quad \theta^{OLS} = \arg\min_{\theta}\left[\sum_{j}\left[y^{j} - \sum_i \theta_i x_i^{j}\right]^2\right]$$

$$\boxed{\theta^{OLS} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\,\mathrm{y}}$$

$$Solving\quad \theta^{Ridge} = \arg\min_{\theta}\left[\sum_{j}\left[y^{j} - \sum_i \theta_i x_i^{j}\right]^2 + \boxed{\lambda\sum_{i>0}\theta_i^{\,2}}\right]$$

$$\boxed{\theta^{Ridge} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \boxed{\lambda \mathrm{I}}\right)^{-1}\mathbf{X}^{\mathrm{T}}\,\mathrm{y}}$$

Note i>0 … not $i \geq 0$ ...

Properties of Ridge Regression

- Solution by matrix notation:

$$\boxed{\theta^{Ridge} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda \mathrm{I}\right)^{-1} \mathbf{X}^{\mathrm{T}}\, \mathrm{y}}$$

- Adding $\lambda>0$ to the diagonal of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ makes the problem nonsingular...

  (easier to invert)

  ... even if $\mathbf{X}$ is not of full rank

- Quadratic penalty makes the ridge solution a linear function of y

Comparing Ridge Regression

Advantages w.r.t. Least Squares

- $(\mathbf{X}^T\mathbf{X} + \lambda I)$ is always invertible ⇒ closed form solution always exists
- Ridge regression controls the complexity with regularization term via λ , which is less prone to overfitting (vs OLS)
  - e.g. sometimes a wildly large θ_i on one variable can be cancelled by another wildly large θ_k of a correlated variable
- Often higher prediction accuracy,
  as the estimates of ridge regression trade a little bias for less variance

Advantages w.r.t. Subset Selection Methods

- Ridge regression is a *continuous* shrinkage method,
  which has less variance than subset selection methods

Disadvantages w.r.t. Subset Selection Methods

- Interpretability and compactness:
  Though coefficients are shrunk, but not to 0
  So do NOT help "interpretability"
- Unlike methods that select subset of the features,
  ridge regression may be inefficient interpretations in high dimensional problems.

Properties of Ridge Regression, cont.

- Later: motivate via Bayesian statistics, using appropriate prior for $\boldsymbol{\theta}$
- Does not automatically select variables
  - See L1-regularization (LASSO)
- Ridge existence theorem: ... $\exists\ \lambda>0$ such that

$$\boxed{MSE\left(\hat{\boldsymbol{\theta}}^{ridge}\right) \le MSE\left(\hat{\boldsymbol{\theta}}^{\mathrm{OLS}}\right)}$$

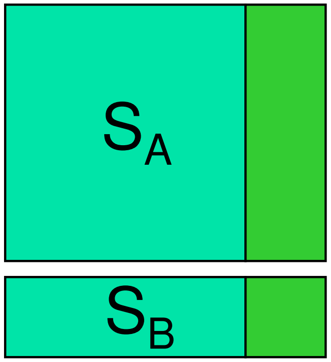
- Must estimate effective complexity parameter λ

Regularization: Empirical Approach

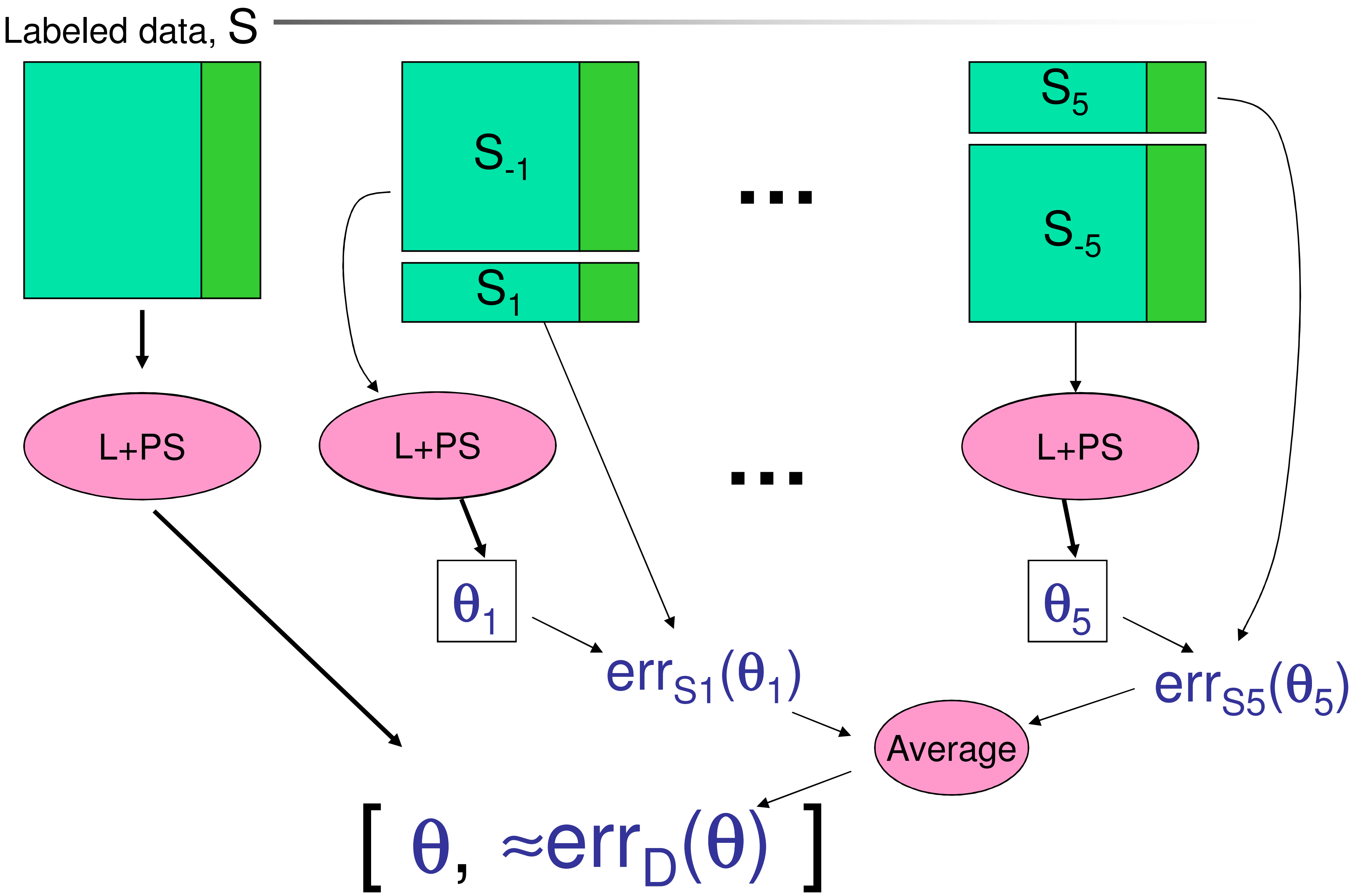
- Problem:
  magic constant λ trading-off complexity vs fit
- Solution 1:
  - Generate multiple models
  - Use lots of test data to discover and discard bad models
- Solution 2: k-fold INTERNAL cross validation:
  - Divide data S into *k* subsets { S_1, …, S_k }
  - Create training subset $S_{-i} = S - S_i$
    - Produces k groups, each of size (k - 1)/k
  - For each value of $\lambda \in \{\ldots, 0.01,\ 0.1,\ 1,\ 10,\ 100, \ldots\}$
    - For i=1..k: Train using this λ on S_{-i}, Eval on S_i : val(λ, i)
    - Compute mean value: val(λ) = average{ val(λ, i) }
  - Set $\lambda^* = \arg\min_{\lambda} \mathrm{val}(\ \lambda\ )$

Learn + Parameter Setting

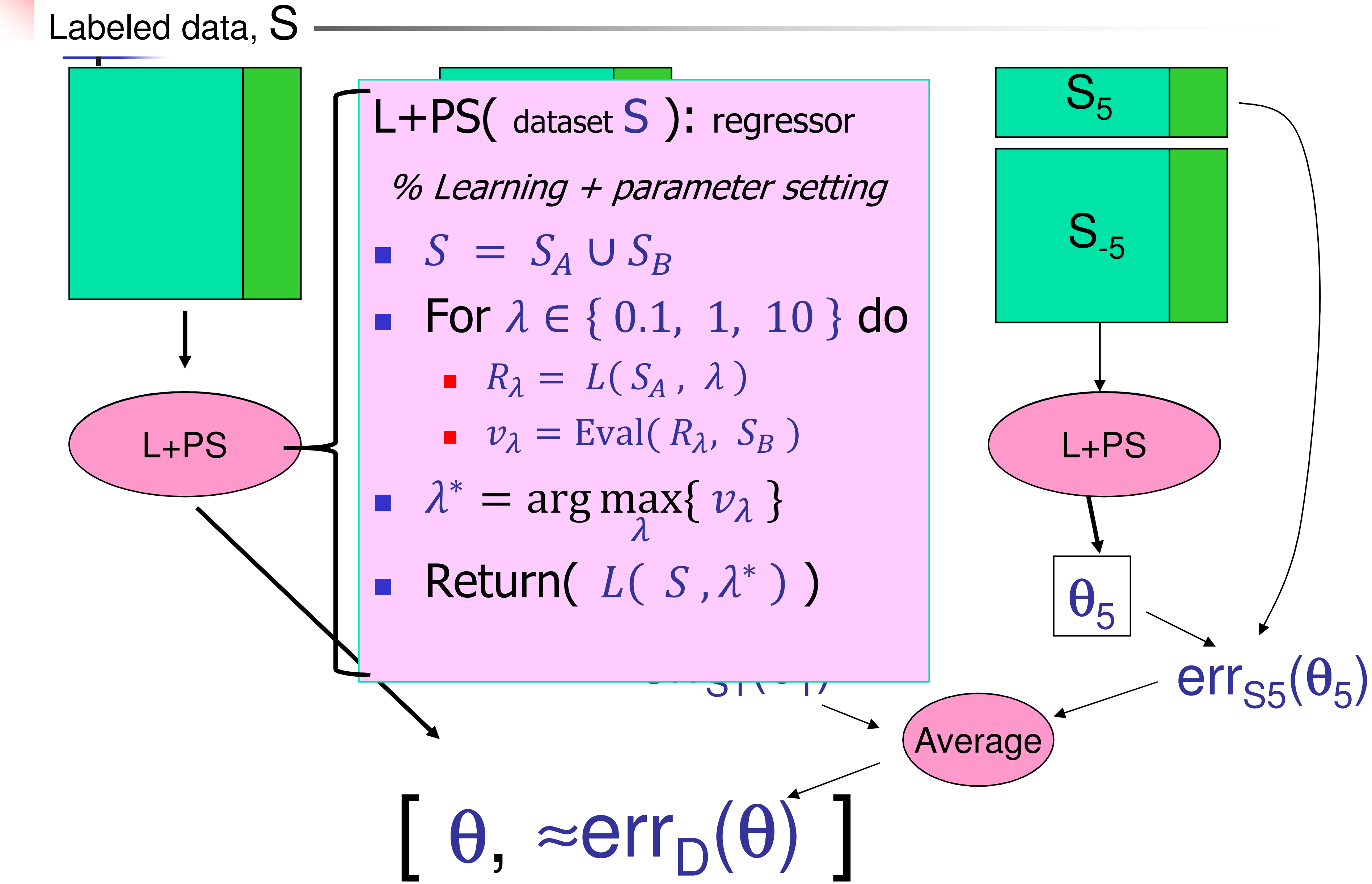
```
L+PS( dataset S ): regressor
        % Learning + parameter setting
- $S = S_A \cup S_B$
- For $\lambda \in \{ 0.1,\ 1,\ 10 \}$ do
    - $R_\lambda = L( S_A ,\ \lambda )$
    - $v_\lambda = \mathrm{Eval}( R_\lambda,\ S_B )$
- $\lambda^* = \arg\max_\lambda \{ v_\lambda \}$
- Return( $L( S , \lambda^* )$ )
```



Return: [Predictor + Est Quality]



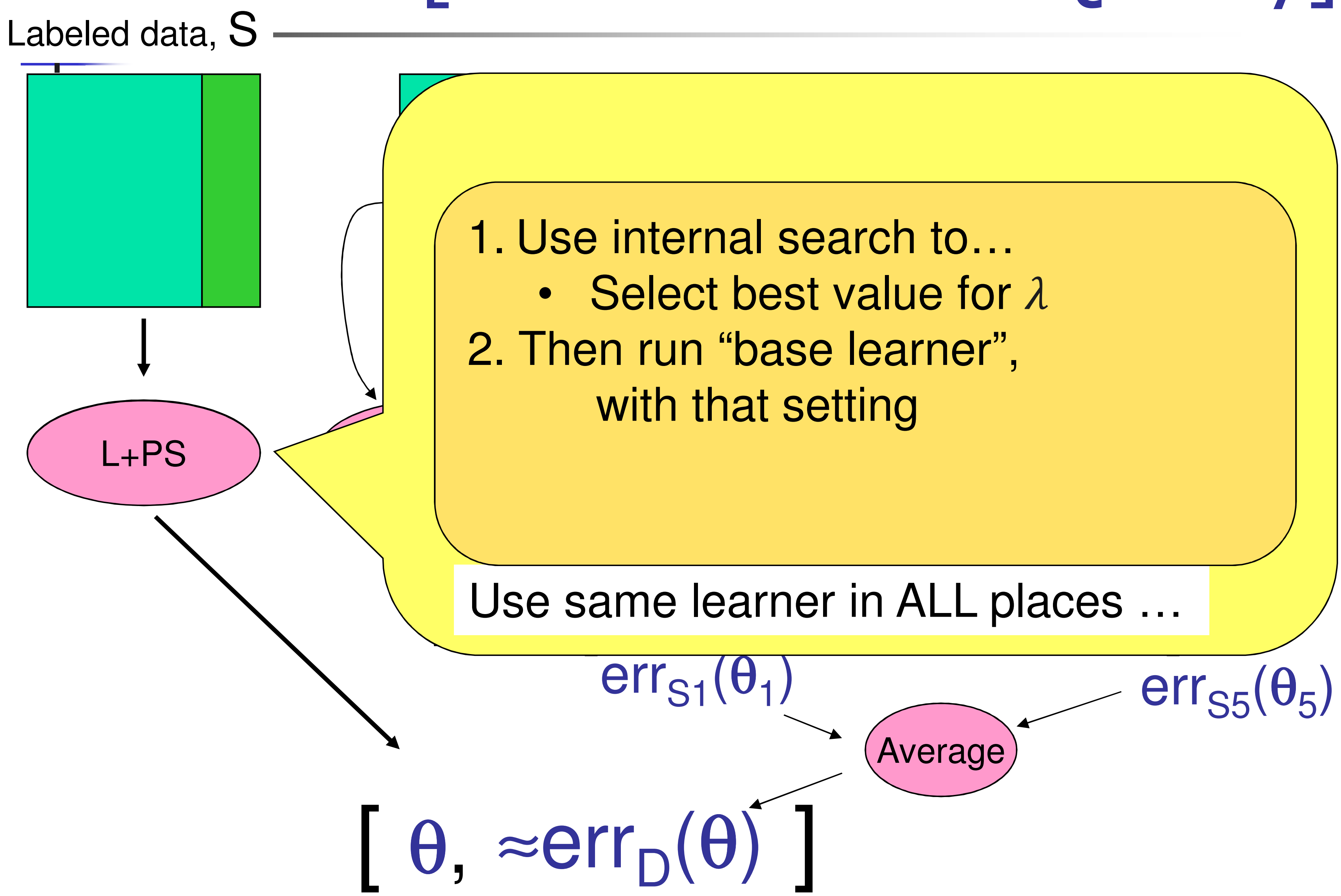
Return: [Predictor + Est Quality]



```
L+PS( dataset S ): regressor
  % Learning + parameter setting
  $S = S_A \cup S_B$
  For $\lambda \in \{ 0.1,\ 1,\ 10 \}$ do
      $R_\lambda = L( S_A ,\ \lambda )$
      $v_\lambda = \text{Eval}( R_\lambda,\ S_B )$
  $\lambda^* = \arg\max_\lambda \{ v_\lambda \}$
  Return( $L( S , \lambda^* )$ )
```

$$[\ \theta,\ \approx \text{err}_D(\theta)\ ]$$

Return: [Predictor + Est Quality]



Labeled data, S

L+PS

1. Use internal search to…
   - Select best value for λ
2. Then run “base learner”, with that setting

Use same learner in ALL places …

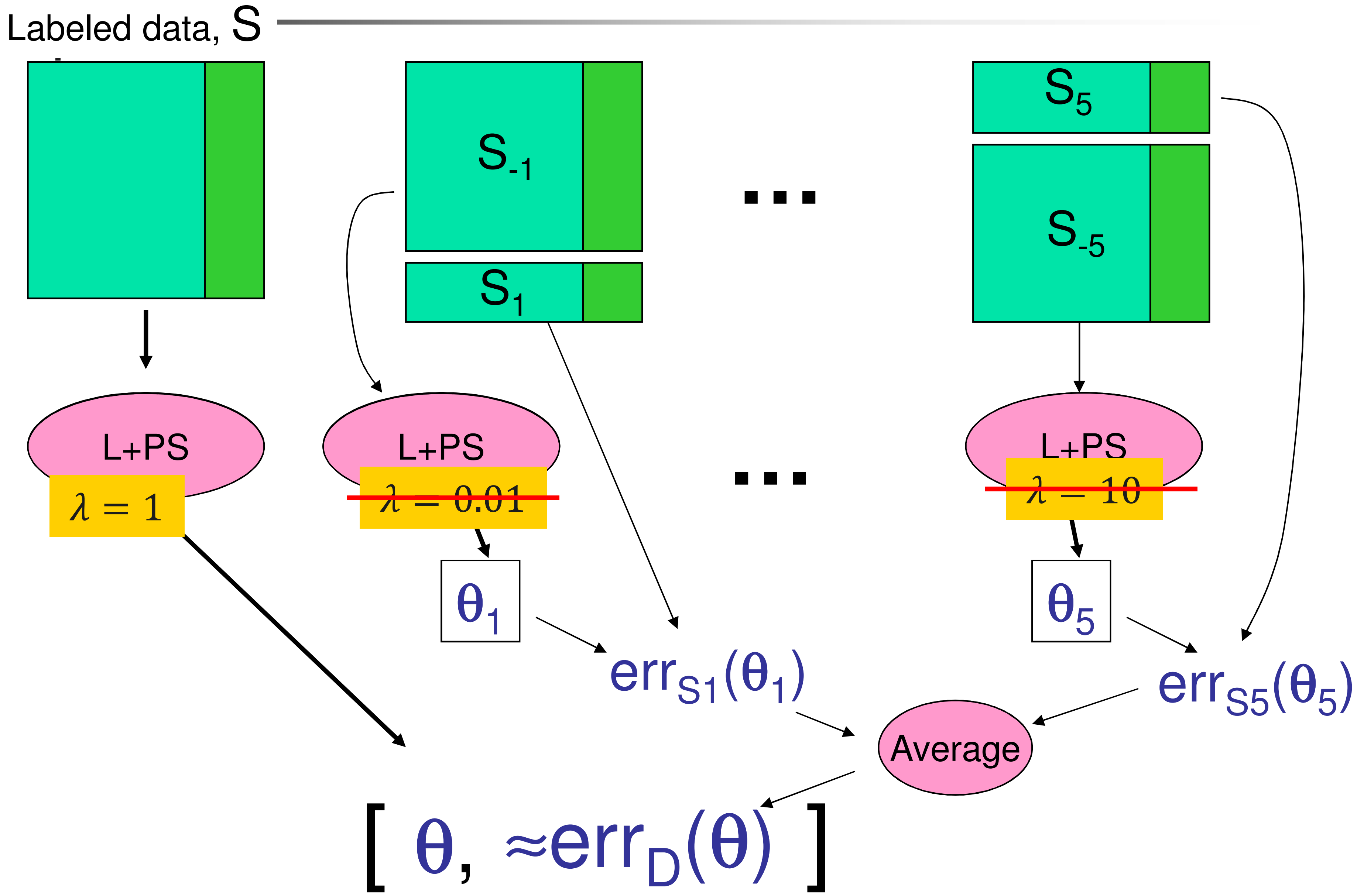
$err_{S1}(\theta_1)$

$err_{S5}(\theta_5)$

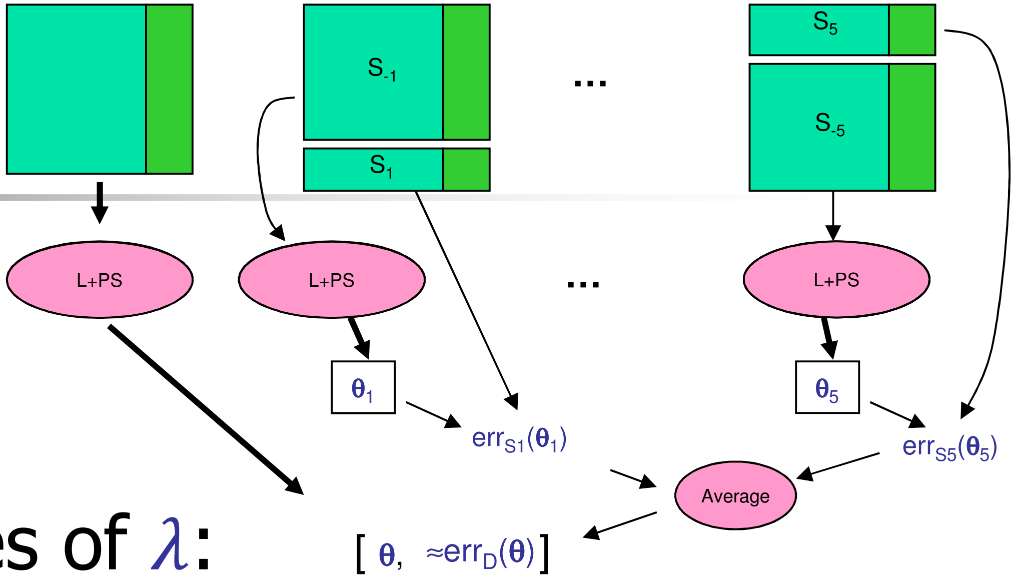
Average

$[\ \theta,\ \approx err_D(\theta)\ ]$

Return: [Predictor + Est Quality]

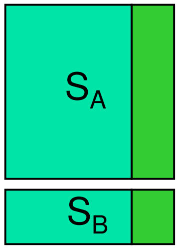


Complexity



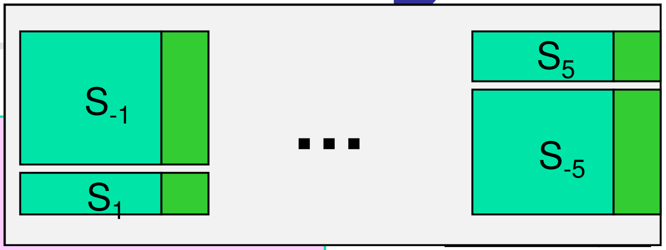
Here, considering 3 values of λ:

- Run *base classifier* $L(S_A, \lambda)$ 3 times (one for each value of λ)
  - + 1 to produce final predictor, on all data $L(S, \lambda^*)$
  - Total: $3 + 1 = 4$



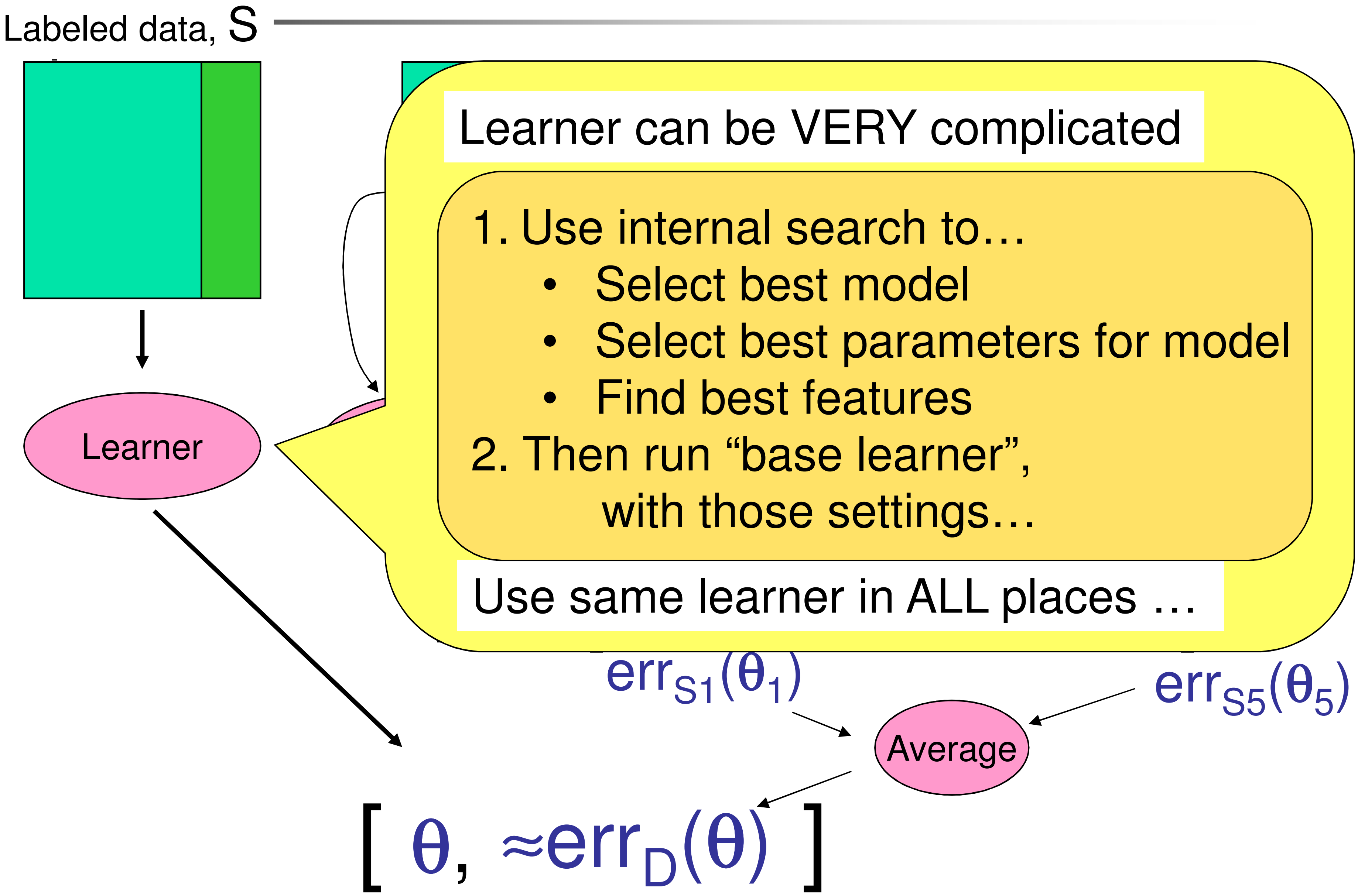
- Do this *entire process* 6 times
  - 1 to produce best regressor
  - One for each of 5 EXTERNAL folds
- Total: $6 \times 4 = 24$

Learn + Parameter Setting



```
L+PS( dataset S ): regressor
        % Learning + parameter setting
- $S = S_1 \cup \cdots \cup S_5$
- For $\lambda \in \{ \ldots, 0.1, 1, 10, \ldots \}$ do
    - For $i = 1..5$
        - $R_{\lambda,i} = L( S_{-i}, \lambda )$
        - $v_{\lambda,i} = Eval( R_{\lambda,i}, S_i )$
    - $v_\lambda = Average( v_{\lambda,1}, \ldots, v_{\lambda,5} )$
- $\lambda^* = \arg\max_\lambda \{ v_\lambda \}$
- Return( $L( S, \lambda^* )$ )
```

Return: [Predictor + Est Quality]



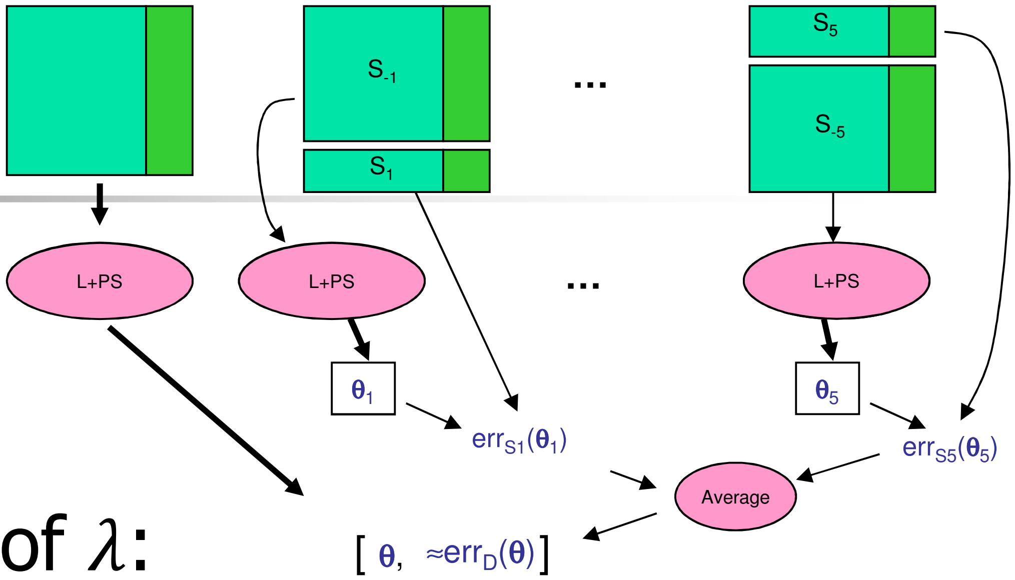
Learner can be VERY complicated

1. Use internal search to…
   - Select best model
   - Select best parameters for model
   - Find best features
2. Then run "base learner",
   with those settings…

Use same learner in ALL places …

$$[\ \theta,\ \approx err_D(\theta)\ ]$$

Complexity



If considering 10 values of λ:

- Run *base classifier* 10 times for each of 5 internal folds (in L+PS)
  - + 1 to produce final predictor, on all data
  - Total: $10 \times 5 + 1 = 51$
- Do this *entire process* 6 times
  - 1 to produce best regressor
  - One for each of 5 EXTERNAL folds
- Total: $6 \times 51 = 306$

Finding Best Model/Parameters

- Want to learn which “parameter values” **v** work best?
  - **v** = Any “setting” for learner:
    λ, degree of polynomial? Number k of features?
    Best set of k features?

    … any other parameters?
    Model class: SVM or Decision tree or Logistic Regression or …
    RBF vs linear? …
  - $\text{argmin}_v \{ \ \text{err}_D(\ L(S, v)) \ \}$
- Learner LV(S) == { Find v*; return $L_{v*}(S)$ }
  involves BOTH

  finding best v, then best predictor, wrt v: $L_v(S)$
  - Finding v* may involve internal C-V steps
    - $v^* = \text{argmin}_v \{ \sum_i E[\, err_{S_i}(L(S_{-i}, v))] \}$
  - To estimate quality of LV(S) , C-V of this COMPLEX LV(.)
    - CVerr(LV, S) = $\frac{1}{k}\sum_i E[\, err_{S_i}(LV(S_{-i})]$

Ways to Cheat
… ie, what NOT to do!

- Select k«p features
  based on all n labeled instances
- Then run learner on just these features
- Run 5-fold CV to estimate accuracy of that classifier
  - each built using 4/5 of the data, eval’ed on 1/5
  - Report average value

Jump

A Lesson

- Van't Veer et al. (2002). Gene expression profiling predicts clinical outcome of breast cancer. Nature, 415(6871), 530-536.

- Goal: Predict Good vs Bad Prognosis
  - Good == “no metastasis for 5 years after initial diagnosis”
- Alg:
  - Select 231 features, ensemble size *based on **entire** data set*
  - Then run 5-fold Cross-Validation (w/those features, …)
- Apparent accuracy: 83%

- If done correctly: (select features *within fold*):
  Accuracy: 73%

- Note: same final predictor;
  only difference is “certificate” (accuracy)…

622/623 of the data

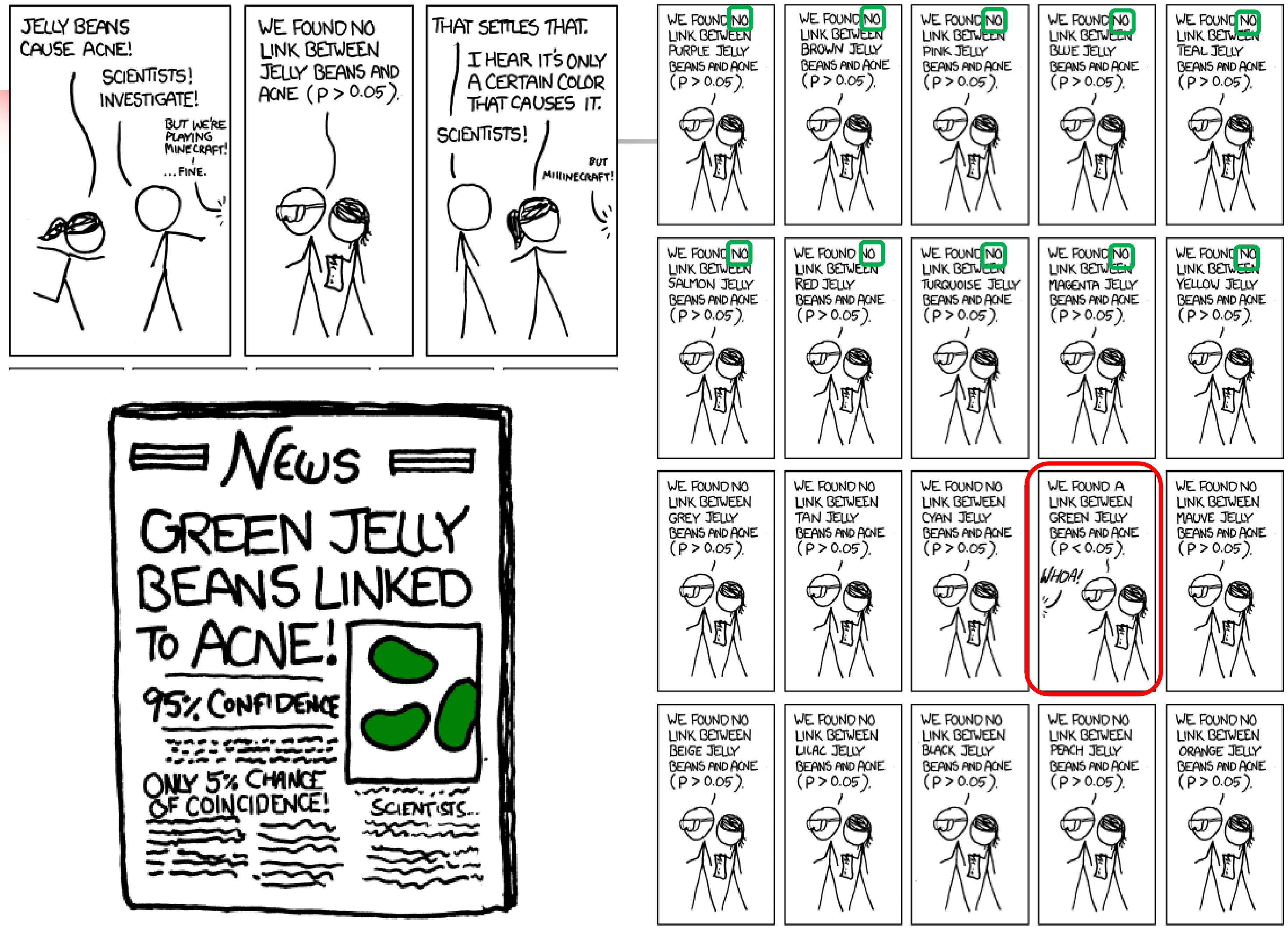
- SNP study
  - 623 instances (302 case and 321 control)
  - > 500K features

- Alg1 ("cheating"):
  - Select 500 features $R \subset F$ features using all **623** instances
  - LOO-CV: run SVM on 622 (w/ features R)

  (Apparent) Accuracy: ≈**90%**

- Alg2 ("correct"):
  - LOO-CV…
    select 500 features R_i in each fold (using only **622** instances)
    run SVM on 622 (w/ features R_i)

  Actual Accuracy: ≈**59%**

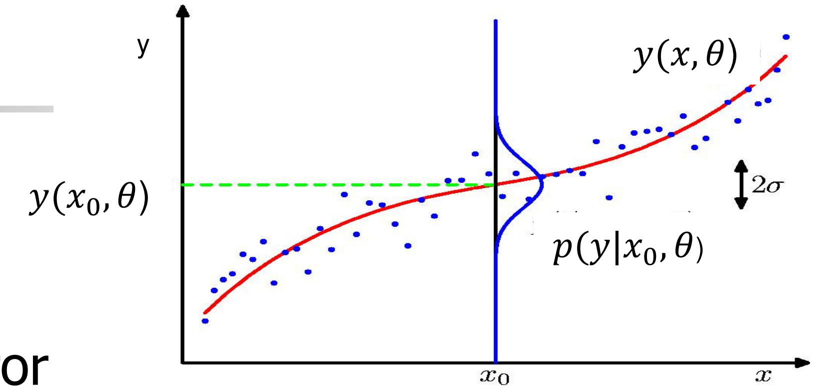

JELLY BEANS CAUSE ACNE!
SCIENTISTS! INVESTIGATE!
BUT WE'RE PLAYING MINECRAFT!
... FINE.
WE FOUND NO LINK BETWEEN JELLY BEANS AND ACNE (P > 0.05).
THAT SETTLES THAT.
I HEAR IT'S ONLY A CERTAIN COLOR THAT CAUSES IT.
SCIENTISTS!
BUT MIIIINECRAFT!
WE FOUND NO LINK BETWEEN PURPLE JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN BROWN JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN PINK JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN BLUE JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN TEAL JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN SALMON JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN RED JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN TURQUOISE JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN MAGENTA JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN YELLOW JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN GREY JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN TAN JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN CYAN JELLY BEANS AND ACNE (P > 0.05).
WE FOUND A LINK BETWEEN GREEN JELLY BEANS AND ACNE (P < 0.05).
WHOA!
WE FOUND NO LINK BETWEEN MAUVE JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN BEIGE JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN LILAC JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN BLACK JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN PEACH JELLY BEANS AND ACNE (P > 0.05).
WE FOUND NO LINK BETWEEN ORANGE JELLY BEANS AND ACNE (P > 0.05).
NEWS
GREEN JELLY BEANS LINKED TO ACNE!
95% CONFIDENCE
ONLY 5% CHANCE OF COINCIDENCE!
SCIENTISTS...

http://xkcd.com/882/

2 reasons for Cross-Validation

- External: to evaluate quality of predictor
- Internal: to set parameters
  (feature selection)
- wrt INTERNAL cross-validation:
  - feel free to “cheat” – doesn’t matter
  - If it is wrong,
    the EXTERNAL cross-validation will catch this!

Outline



- Linear Regression
- Evaluating Predictors
  - Training set error vs Test set error
  - Cross Validation
- Overfitting
  - Bias-Variance analysis
  - Feature Selection
  - L_2 Regularization
  - Setting parameters … internal C-V
  - Bayesian Model
  - L_1 Regularization (Lasso)
- Linear Classification

Bayesian Approach

- Formulate our knowledge about the world probabilistically:
  - define the model that expresses our knowledge qualitatively
    - eg: independence assumptions, forms of distributions
  - Model has some unknown parameters
    - before seeing data
    - capture assumptions (== prior beliefs) about unknown parameters
      (eg, range of plausible values)
      by specifying the prior distribution over those parameters,
- Observe the data
- Compute the posterior probability for the parameters, given observed data
- Use this posterior to…
  - Make predictions by averaging over the posterior distribution
  - Examine/Account for uncertainly in the parameter values
  - Make decisions by minimizing expected posterior loss

A Bayesian Perspective

- Given space of possible hypotheses $H = \{ h_j \}$ find hypothesis with the highest posterior:

$$P(h \mid D) = \frac{P(D \mid h)\ P(h)}{P(D)}$$

- As $P(D)$ does not depend on h:

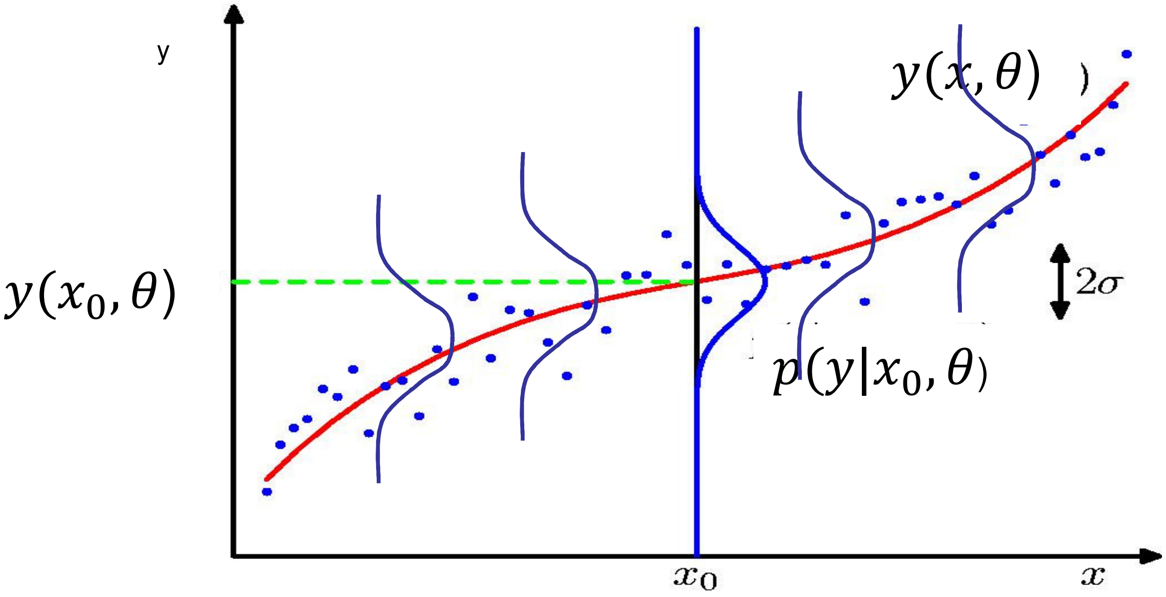
  $\text{argmax}_h\ P(h \mid D) = \text{argmax}_h\ P(D \mid h)\ P(h)$

- "Uniform $P(h)$" ⇒ Maximum Likelihood Estimate
  - (model for which data has highest probability)

- … can use $P(h)$ for ≈regularization …

Bayesian Regression

- Assume that, given **x**, noise is iid Gaussian
- Homoscedastic noise model
  (same σ for each position)



Maximum Likelihood Solution

$$P(D \mid \mathrm{h}_{\boldsymbol{\theta}}) = \mathrm{P}\left(t^{(1)}, \dots, t^{(m)} \,\middle|\, y(\mathbf{x}; \boldsymbol{\theta}), \sigma\right) = \prod_{i} \frac{\exp\left(- \frac{\left(t^{(i)} - y\left(\mathbf{x}^{(\mathbf{i})}; \boldsymbol{\theta}\right)\right)^{\mathbf{2}}}{2\sigma^2} \right)}{\sqrt{2\pi\sigma^2}}$$

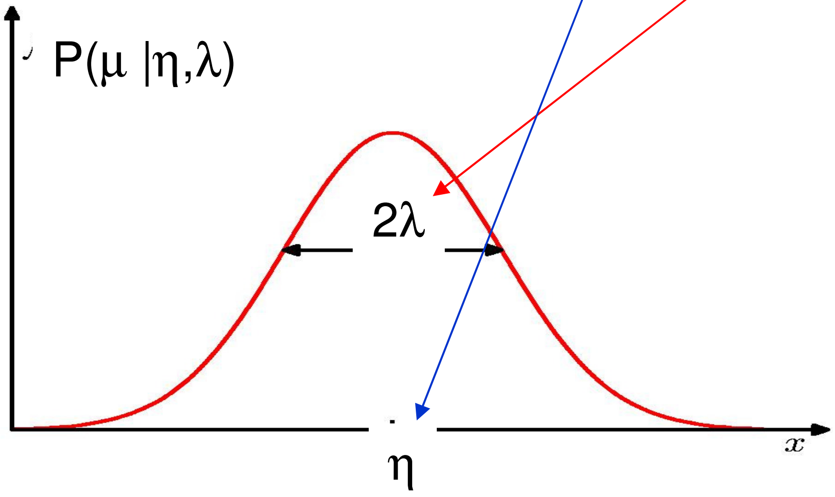
MLE fit for $\boldsymbol{\theta}$ is …

- just linear regression fit
- does not depend upon σ^2

Bayesian learning of Gaussian parameters

- Conjugate priors
  - Mean: Gaussian prior
  - Variance: Wishart Distribution
- Prior for mean:

$$P(\mu \mid \eta, \lambda) = \frac{1}{\lambda\sqrt{2\pi}} e^{\frac{-(\mu-\eta)^2}{2\lambda^2}}$$



Bayesian Solution

h = hypothesis … corresponding to weights $\boldsymbol{\theta}$

- Introduce prior distribution over weights

$$P(\,h_{\theta}\,) = P(\,\boldsymbol{\theta}\,|\rho\,) = N(\boldsymbol{\theta}|\,\mathbf{0},\rho^{2}\mathbf{I}\,)$$

- Posterior is…

$$P(D\,|\,h_{\theta}\,)\,P(h_{\theta}) = P\left(\,t^{(1)},\ldots,t^{(m)}\,\middle|\,y(\mathbf{x};\boldsymbol{\theta}),\boldsymbol{\sigma}\right)\quad P(\boldsymbol{\theta})$$

$$= \quad \prod_{i}\frac{\exp\left(-\,\frac{\left(t^{(i)}\;-y(\,\mathbf{x}^{(\mathrm{i})};\boldsymbol{\theta})\right)^{2}}{2\sigma^{2}}\right)}{\sqrt{2\pi\sigma^{2}}}\quad\frac{\exp\left(\frac{-\boldsymbol{\theta}^{T}\boldsymbol{\theta}}{2\rho^{2}}\right)}{\sqrt{2\pi\rho^{2}}^{\mathrm{k}}}$$

- Bayesian Regression = MAP($\boldsymbol{\theta}$) is…

$$\operatorname{argmax}_{\theta}\quad\prod_{i}\frac{\exp\left(-\,\frac{\left(t^{(i)}\;-y(\,\mathbf{x}^{(i)};\boldsymbol{\theta})\right)^{2}}{2\sigma^{2}}\right)}{\sqrt{2\pi\sigma^{2}}}\quad\frac{\exp\left(\frac{-\theta^{T}\theta}{2\rho^{2}}\right)}{\sqrt{2\pi\rho^{2}}^{k}}$$

Bayesian Solution

- Bayesian Regression = MAP($\boldsymbol{\theta}$) is…

$$\operatorname{argmax}_{\boldsymbol{\theta}} \prod_i \frac{\exp\left(-\frac{\left(t^{(i)} - y\left(\mathbf{x}^{(\mathrm{i})};\boldsymbol{\theta}\right)\right)^{2}}{2\sigma^2}\right)}{\cancel{\sqrt{2\pi\sigma^2}}} \quad \frac{\exp\left(\frac{-\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{\theta}}{2\rho^2}\right)}{\cancel{\sqrt{2\pi\rho^2}^{\mathrm{k}}}}$$

$$= \operatorname{argmax}_{\boldsymbol{\theta}} \sum_i \left(-\frac{\left(t^{(i)} - y\left(\mathbf{x}^{(\mathrm{i})};\boldsymbol{\theta}\right)\right)^{2}}{2\sigma^2}\right) + \frac{-\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{\theta}}{2\rho^2}$$

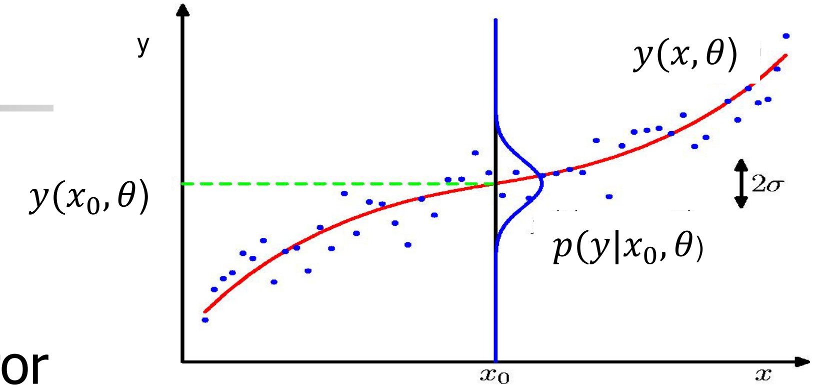
$$= \boxed{\operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - y\left(\mathbf{x}^{(\mathrm{i})};\boldsymbol{\theta}\right)\right)^{2} \quad + \quad \frac{2\sigma^2}{2\rho^2}\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{\theta}}$$

- Compare to Regularized Regression:

$$\boxed{\operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - y\left(\mathbf{x}^{(\mathrm{i})};\boldsymbol{\theta}\right)\right)^{2} \quad + \quad \lambda\, |\boldsymbol{\theta}|_2^{\,2}}$$

Identical,
up to constants!!!

Outline



- Linear Regression
- Evaluating Predictors
  - Training set error vs Test set error
  - Cross Validation
- Overfitting
  - Bias-Variance analysis
  - Feature Selection
  - L_2 Regularization
  - Setting parameters … internal C-V
  - Bayesian Model
  - L_1 Regularization (Lasso)
- Linear Classification

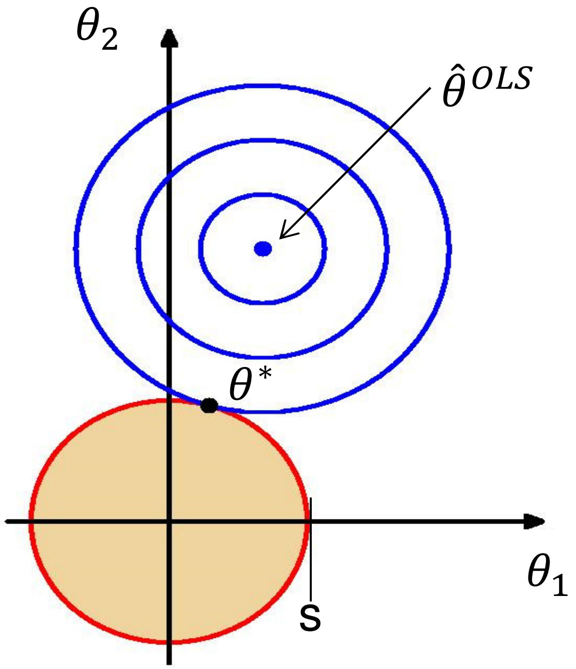
Viewing L_2 Regularization

$$\theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j\, x_j^{(i)} \right)^{\mathbf{2}} + \lambda \sum_j {\theta_j}^2$$

- Using Lagrange Multiplier…

$$\Rightarrow \quad \theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j\, x_j^{(i)} \right)^{\mathbf{2}}$$

s.t. $\sum_j {\theta_j}^2 \le s$

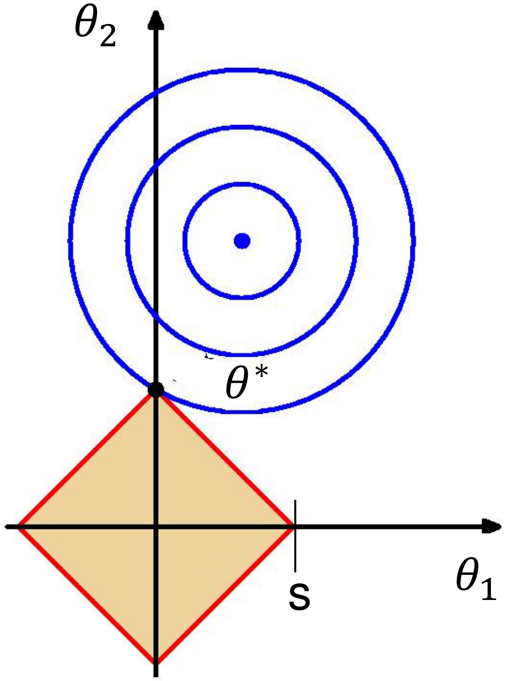


L_2 vs L_1 Regularization

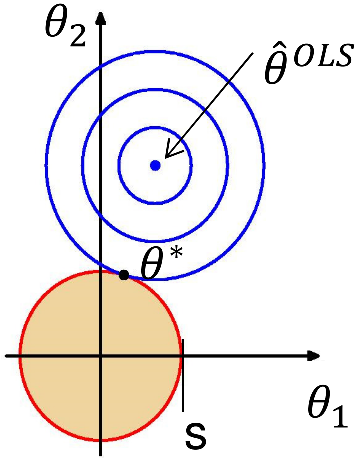
1st power, not 2nd

$$\theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j \, x_j^{(i)} \right)^{\mathbf{2}} + \lambda \sum_j | \theta_j |$$

$$\Rightarrow \quad \theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j \, x_j^{(i)} \right)^{\mathbf{2}} \text{ s.t. } \sum_j | \theta_j | \le s$$



Intersections often on axis!
… here, $\theta_j = 0$!!



LASSO!

The Lasso

- Lasso is a shrinkage method, like Ridge, but 1st

$$\theta^{Lasso} = \text{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j \, x_j^{(i)} \right)^{\mathbf{2}} \text{s.t.} \sum_j | \theta_j | \le s$$

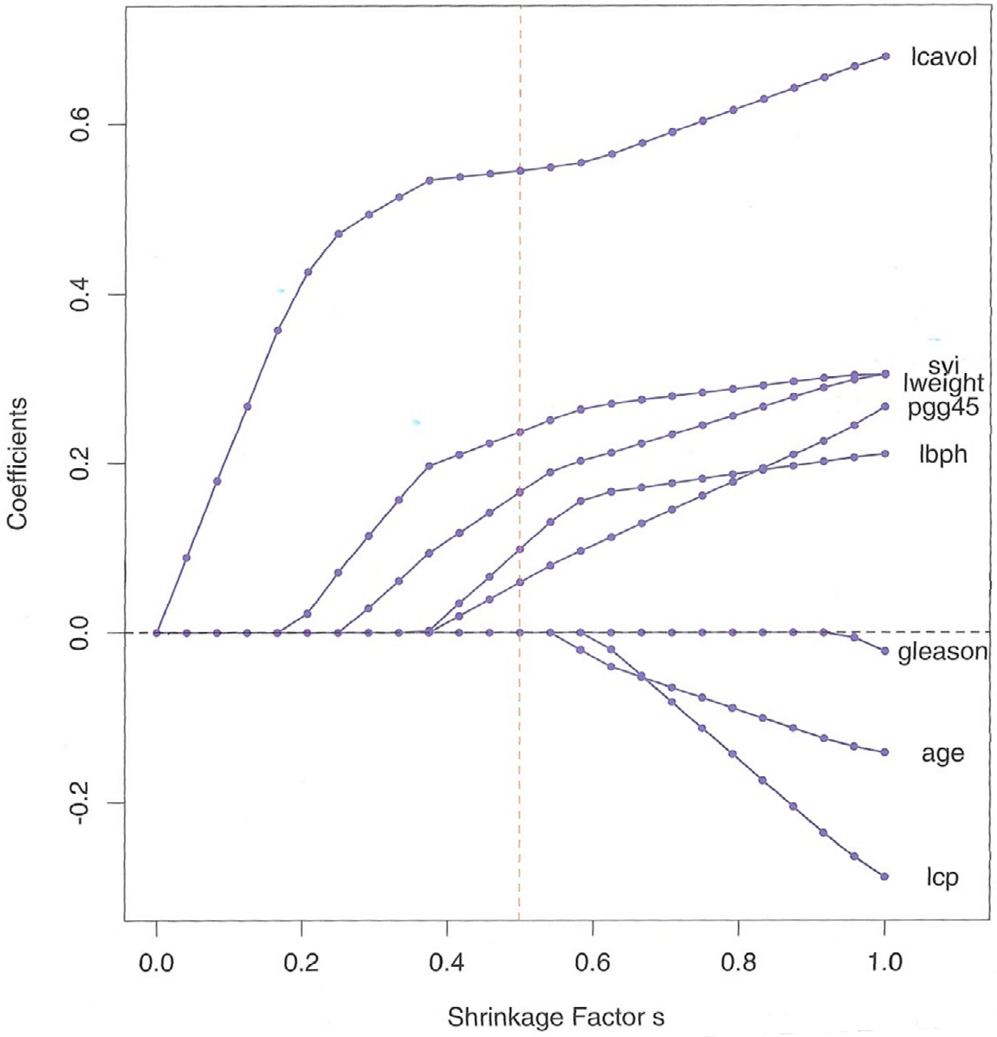
- L_1 penalty ⇒ solution *nonlinear* in arg's (**t**, **x**)
  - Use quadratic programming to compute the solutions
- Sufficient shrinkage will cause some coefficients to be **exactly 0**
  - so it acts like a subset selection method !
- Bayesian: Prior is Laplacian, not Gaussian ...
  - $P(\theta \mid \tau) = \frac{1}{\tau} \exp(\frac{-|\theta|}{\tau})$

Example

Coefficients plotted against

$$s = \frac{|t|}{\sum_{j=1..p} |\hat{\theta}|}$$

Some LASSO coefficients hit 0, while those for Ridge do not.



Example: Prostate cancer

Term	OLS	Best subset	Ridge	Lasso
Intercept	2.480	2.495	2.467	2.477
lcalvol	0.680	0.740	0.389	0.545
lweight	0.305	0.367	0.238	0.237
age	-0.141		-0.029	
lbph	0.210		0.159	0.098
svi	0.305		0.217	0.165
lcp	-0.288		0.026	
Gleason	-0.021		0.042	
Pgg45	0.267		0.123	0.059
Test err.	0.586	0.574	0.540	0.491
Std.err.	0.184	0.156	0.168	0.152

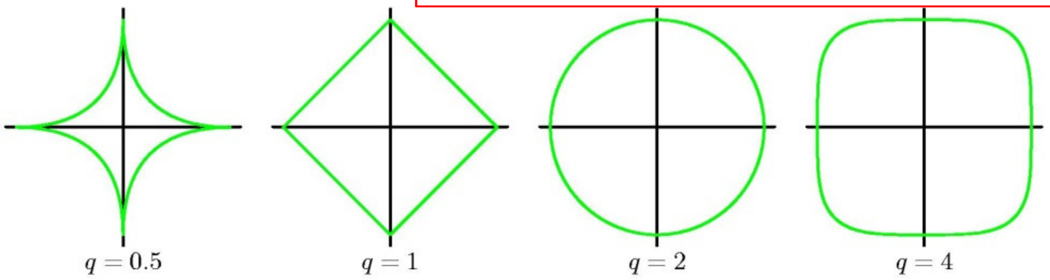
A unifying view

$$\theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j \, x_j^{(i)} \right)^{\mathbf{2}} + \lambda \sum_j | \, \theta_j \, |^q$$

- λ is “bias”, q indicates a prior distribution on $\boldsymbol{\theta}$
  - λ=0: ordinary least squares
  - λ>0, q=2: Ridge regression
  - λ>0, q=1: the LASSO

Other Possible q-norms

$$\theta^* = \operatorname{argmin}_{\boldsymbol{\theta}} \sum_i \left(t^{(i)} - \sum_j \theta_j \, x_j^{(i)} \right)^2 + \lambda \sum_j | \theta_j |^q$$



	Convex	Smooth	Sparse
q<1	No	No	Yes
q=1	**Yes**	No	**Yes**
q>1	Yes	Yes	No

- Want Yes,Yes,Yes… but Yes,No,Yes isn’t bad…
- q=0: pure variable selection
  - just counting #**non-zero coefficients**
  - NP-hard

What you need to know

- Regression
  - Optimizing sum squared error == MLE !
  - Basis functions = features
  - Relationship between regression and Gaussians
- Evaluating Predictor
  - TestSetError ≠ Prediction Error
  - Cross Validation
- Bias-Variance trade-off
  - Model complexity …
- Regularization ≈ Bayesian modeling
- L_1 regularization – finds 0 weights!

Play with Applet

Overfitting Thriller!

https://www.youtube.com/watch?v=DQWI1kvmwRg