Regression and generalization

CE-717: Machine Learning
Sharif University of Technology

M. Soleymani
Fall 2016

Topics

- Beyond linear regression models
- Evaluation & model selection
- Regularization
- Probabilistic perspective for the regression problem

Recall: Linear regression (squared loss)

- Linear regression functions

$$f : \mathbb{R} \rightarrow \mathbb{R} \quad f(x; \boldsymbol{w}) = w_0 + w_1 x$$

$$f : \mathbb{R}^{\mathrm{d}} \rightarrow \mathbb{R} \quad f(\boldsymbol{x}; \boldsymbol{w}) = w_0 + w_1 x_1 + \ldots w_d x_d$$

$\boldsymbol{w} = [w_0, w_1, \ldots, w_d]^T$ are the parameters we need to set.

- Minimizing the squared loss for linear regression

$$J(\boldsymbol{w}) = ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}||_2^2$$

- We obtain $\hat{\boldsymbol{w}} = (\boldsymbol{X}^T \boldsymbol{X})^{-\mathbf{1}} \boldsymbol{X}^T \boldsymbol{y}$

Beyond linear regression

- How to extend the linear regression to non-linear functions?
  - Transform the data using basis functions
  - Learn a linear regression on the new feature vectors (obtained by basis functions)

Beyond linear regression

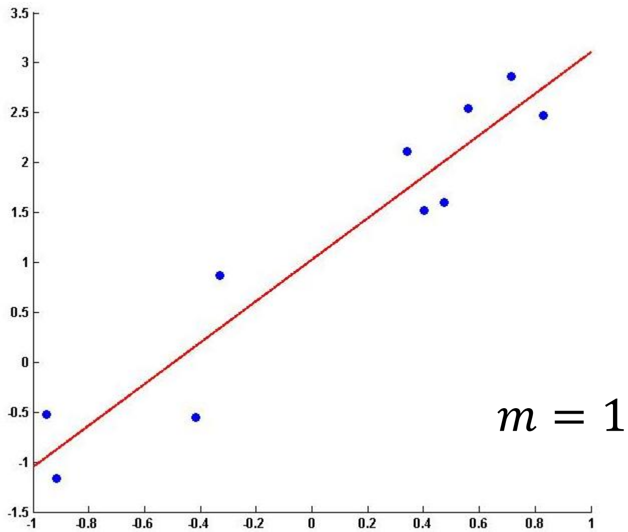
- m^{th} order polynomial regression (univariate $f : \mathbb{R} \longrightarrow \mathbb{R}$)

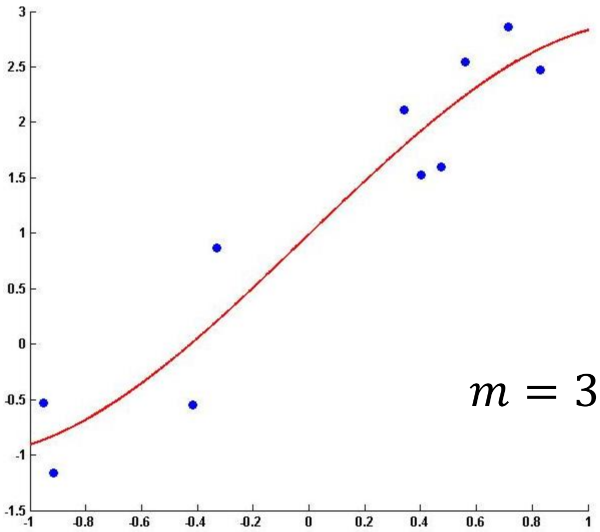
$$f(x;\boldsymbol{w}) = w_0 + w_1 x + \ldots + w_{m-1} x^{m-1} + w_m \, x^m$$

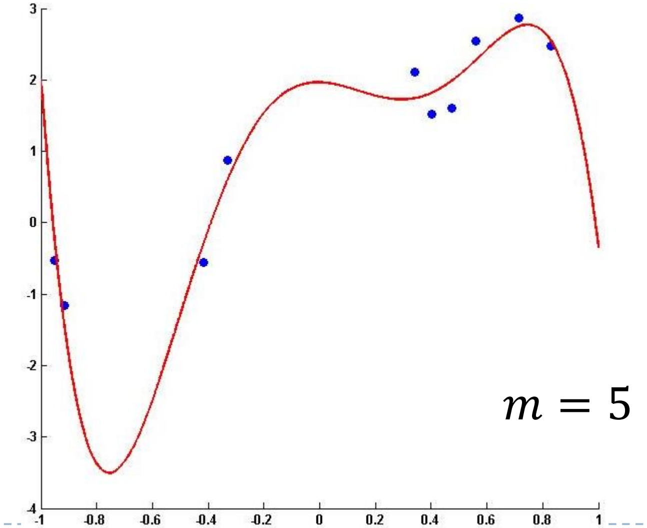
- Solution: $\widehat{\boldsymbol{w}} = \left(\boldsymbol{X}'^T \boldsymbol{X}'\right)^{-1} \boldsymbol{X}'^T \boldsymbol{y}$

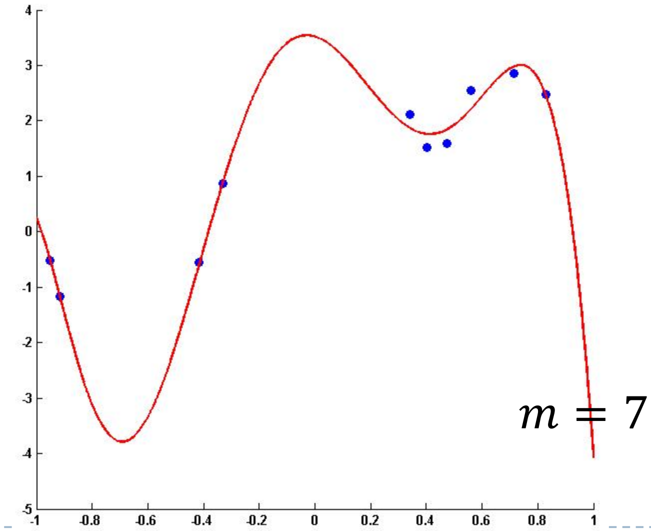
$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \boldsymbol{X}' = \begin{bmatrix} 1 & x^{(1)^1} & x^{(1)^2} & \cdots & x^{(1)^m} \\ 1 & x^{(2)^1} & x^{(2)^2} & \cdots & x^{(2)^m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x^{(n)^1} & x^{(n)^2} & \cdots & x^{(n)^1} \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \widehat{\boldsymbol{w}}_0 \\ \widehat{\boldsymbol{w}}_1 \\ \vdots \\ \widehat{\boldsymbol{w}}_m \end{bmatrix}$$

Polynomial regression: example









Generalized linear

- Linear combination of fixed non-linear function of the input vector

$$f(\boldsymbol{x};\boldsymbol{w}) = w_0 + w_1\phi_1(\boldsymbol{x}) + \ldots w_m\phi_m(\boldsymbol{x})$$

$\{\phi_1(\boldsymbol{x}),\ldots,\phi_m(\boldsymbol{x})\}$: set of basis functions (or features)

$$\phi_i(\boldsymbol{x}): \mathbb{R}^d \to \mathbb{R}$$

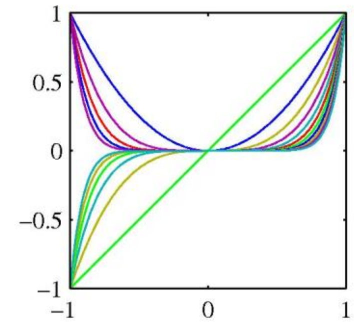
Basis functions: examples

- Linear

If $m = d$, $\phi_i(\mathbf{x}) = x_i$, $i = 1, \ldots, d$, then

$$f(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_d x_d$$

- Polynomial (univariate)

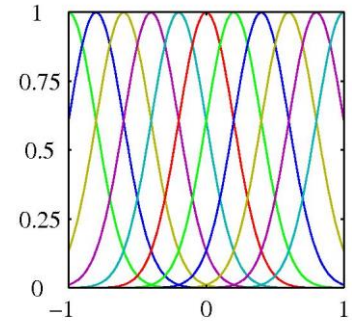


If $\phi_i(x) = x^i$, $i = 1, \ldots, m$, then

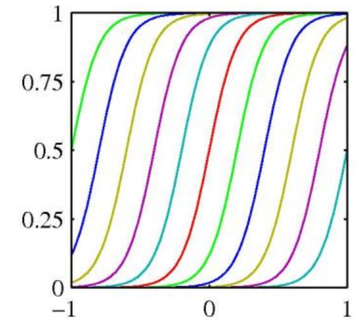
$$f(x; \mathbf{w}) = w_0 + w_1 x + \ldots + w_{m-1} x^{m-1} + w_m x^m$$

Basis functions: examples

- Gaussian: $\phi_j(\boldsymbol{x}) = exp\left\{-\frac{(\boldsymbol{x}-\boldsymbol{c}_j)^2}{2\sigma_j^2}\right\}$



- Sigmoid: $\phi_j(\boldsymbol{x}) = \sigma\left(\frac{\|\boldsymbol{x}-\boldsymbol{c}_j\|}{\sigma_j}\right)$ $\qquad \sigma(a) = \frac{1}{1+\exp(-a)}$



Radial Basis Functions: prototypes

- Predictions based on similarity to "prototypes":

$$\phi_j(\boldsymbol{x}) = exp\left\{-\frac{1}{2\sigma_j^2}\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|^2\right\}$$

- Measuring the similarity to the prototypes $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m$
  - σ^2 controls how quickly it vanishes as a function of the distance to the prototype.
  - Training examples themselves could serve as prototypes

Generalized linear: optimization

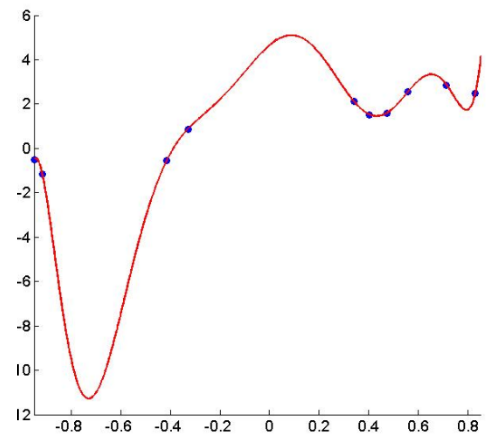
$$J(\boldsymbol{w}) = \sum_{i=1}^{n} \left(y^{(i)} - f\left(\boldsymbol{x}^{(i)}; \boldsymbol{w}\right)\right)^2$$
$$= \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w}^T \boldsymbol{\phi}\left(\boldsymbol{x}^{(i)}\right)\right)^2$$

$$\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix} \quad \boldsymbol{\Phi} = \begin{bmatrix} 1 & \phi_1(\boldsymbol{x}^{(1)}) & \cdots & \phi_m(\boldsymbol{x}^{(1)}) \\ 1 & \phi_1(\boldsymbol{x}^{(2)}) & \cdots & \phi_m(\boldsymbol{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \phi_1(\boldsymbol{x}^{(n)}) & \cdots & \phi_m(\boldsymbol{x}^{(n)}) \end{bmatrix} \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_m \end{bmatrix}$$

$$\widehat{\boldsymbol{w}} = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-\mathbf{1}} \boldsymbol{\Phi}^T \boldsymbol{y}$$

Model complexity and overfitting

- With limited training data, models may achieve zero training error but a large test error.



Training (empirical) loss

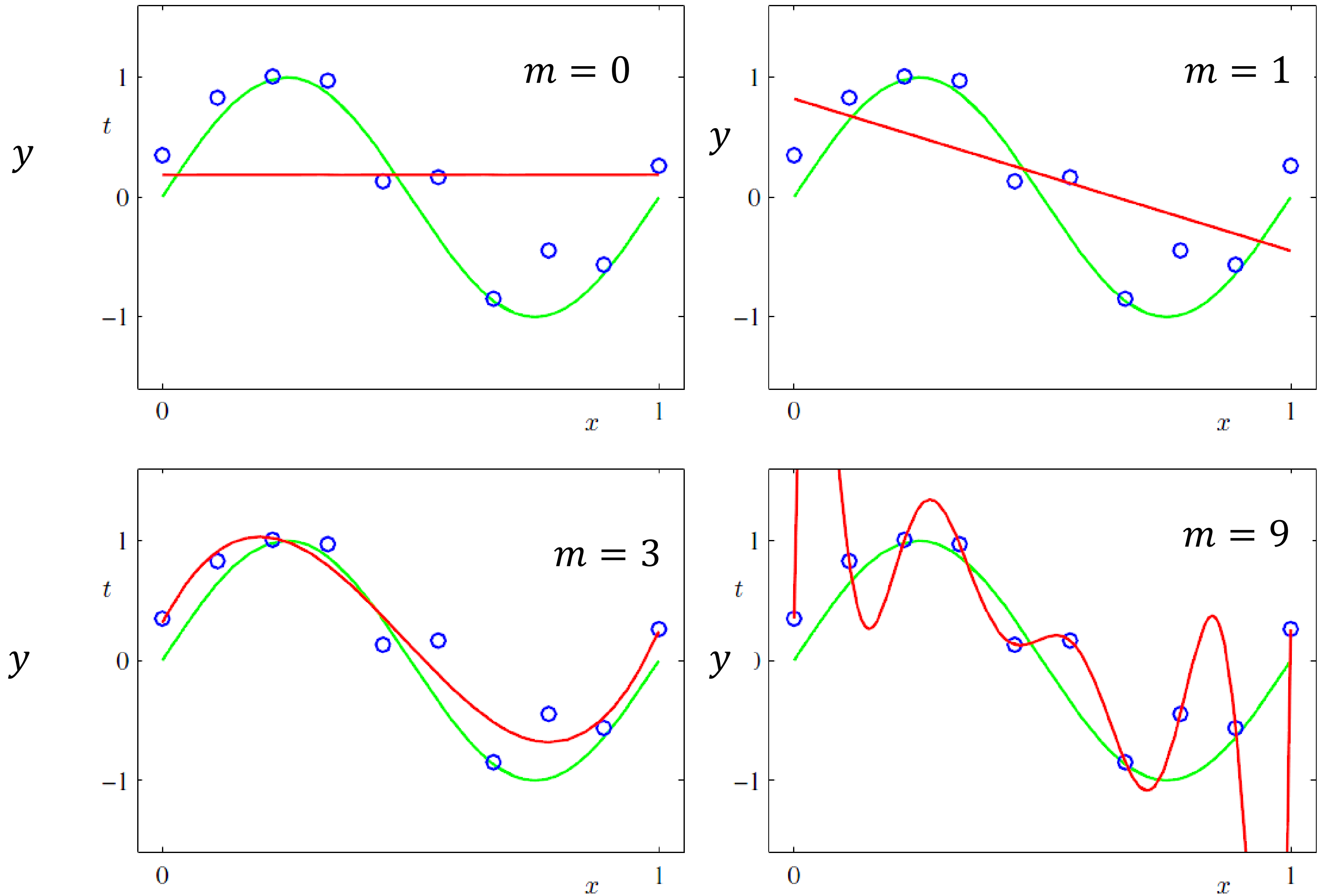
$$\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)\right)^2 \approx 0$$

Expected (test) loss

$$\mathrm{E}_{\mathbf{x},\mathrm{y}}\left\{\left(y - f(\boldsymbol{x};\boldsymbol{\theta})\right)^2\right\} \gg 0$$

- <u>Over-fitting</u>: when the training loss no longer bears any relation to the test (generalization) loss.
  - Fails to generalize to unseen examples.

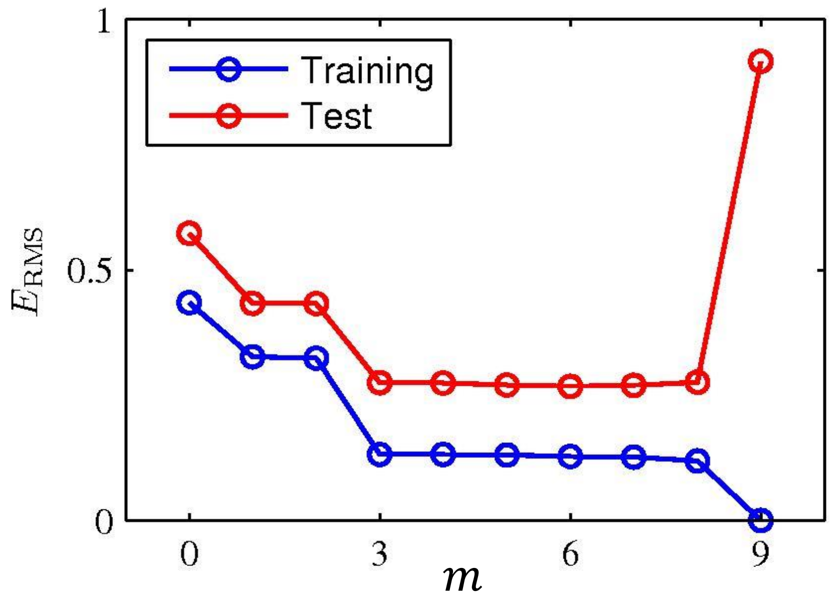
Polynomial regression



[Bishop]

Polynomial regression: training and test error

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left(y^{(i)} - f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)\right)^2}{n}}$$



[Bishop]

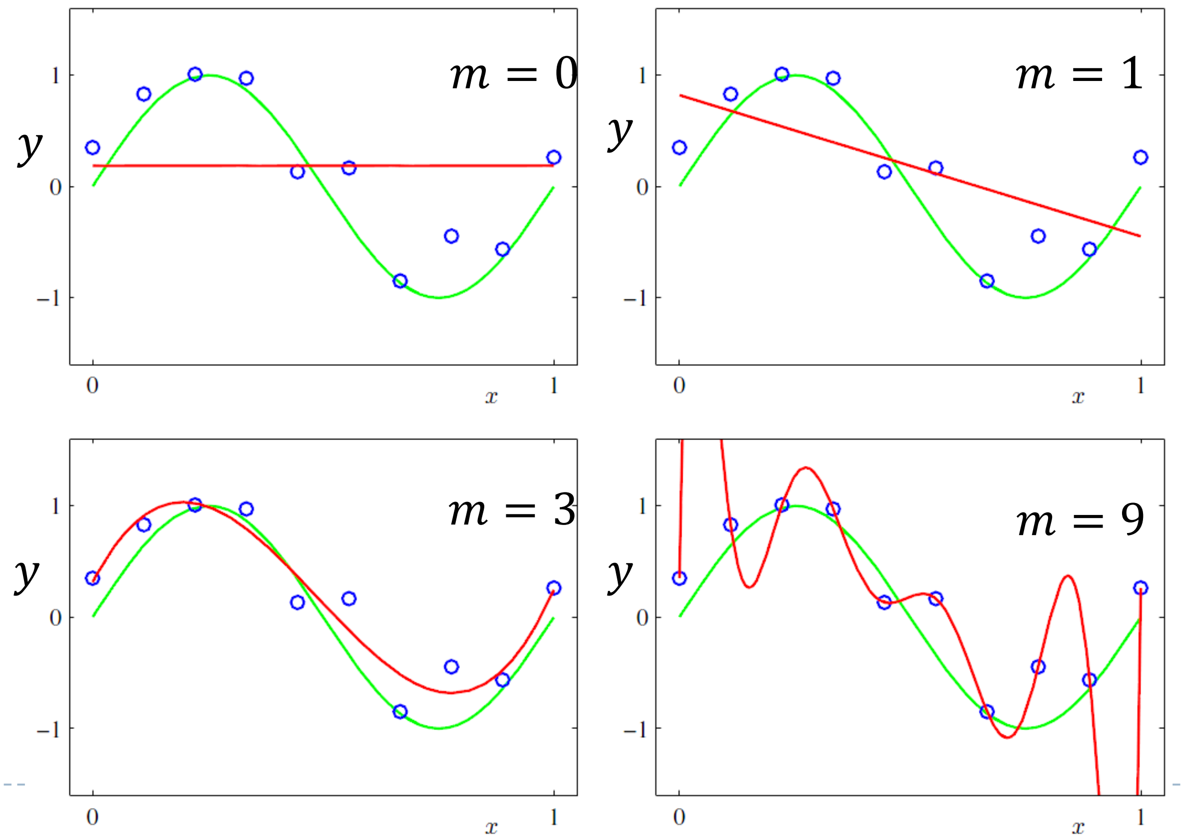
Over-fitting causes

- Model complexity
  - E.g., Model with a large number of parameters (degrees of freedom)

- Low number of training data
  - Small data size compared to the complexity of the model

Model complexity

- Example:
  - Polynomials with larger m are becoming increasingly tuned to the random noise on the target values.

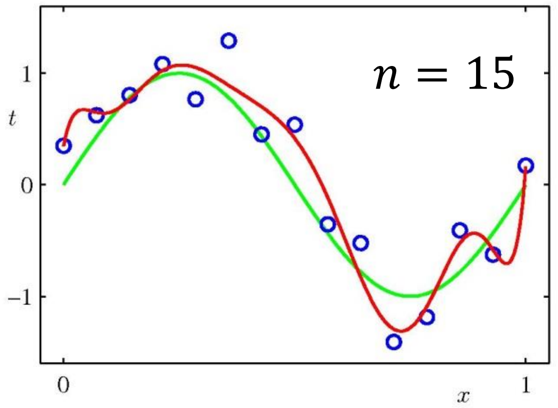


[Bishop]

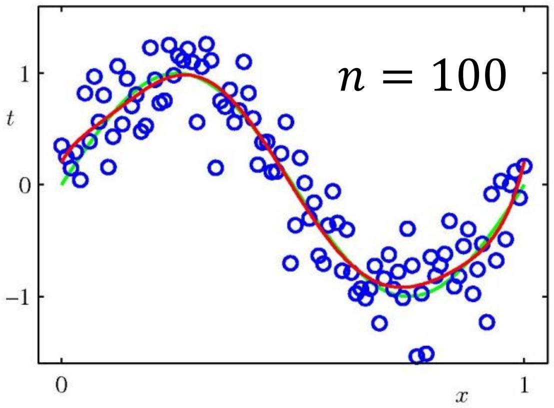
Number of training data & overfitting

- **Over-fitting problem becomes less severe as the size of training data increases.**

$m = 9$



$m = 9$



[Bishop]

How to evaluate the learner's performance?

- Generalization error: true (or expected) error that we would like to optimize

- Two ways to assess the generalization error is:
  - Practical: Use a separate data set to test the model
  - Theoretical: Law of Large numbers
    - statistical bounds on the difference between training and expected errors

Evaluation and model selection

- **Evaluation**:
  - We need to measure how well the learned function can predicts the target for unseen examples

- **Model selection:**
  - Most of the time we need to select among a set of models
    - Example: polynomials with different degree m
  - and thus we need to evaluate these models first

Avoiding over-fitting

- Determine a suitable value for model complexity
  - **Simple hold-out method**
  - **Cross-validation**
- Regularization (Occam's Razor)
  - Explicit preference towards simple models
  - Penalize for the model complexity in the objective function
- Bayesian approach

Simple hold-out: model selection

- Steps:
  - Divide training data into training and validation set v_set
  - Use only the training set to train a set of models
  - Evaluate each learned model on the validation set
    - $J_v(\boldsymbol{w}) = \frac{1}{|v\_set|} \sum_{i \in v\_set} \left(y^{(i)} - f\left(\boldsymbol{x}^{(i)}; \boldsymbol{w}\right) \right)^2$
  - Choose the best model based on the validation set error

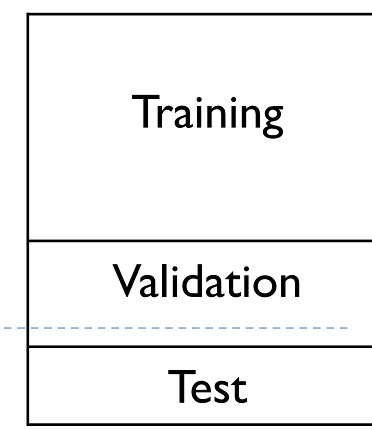
- Usually, too wasteful of valuable training data
  - Training data may be limited.
  - On the other hand, small validation set give a relatively noisy estimate of performance.

Simple hold out: training, validation, and test sets

- Simple hold-out chooses the model that minimizes error on validation set.

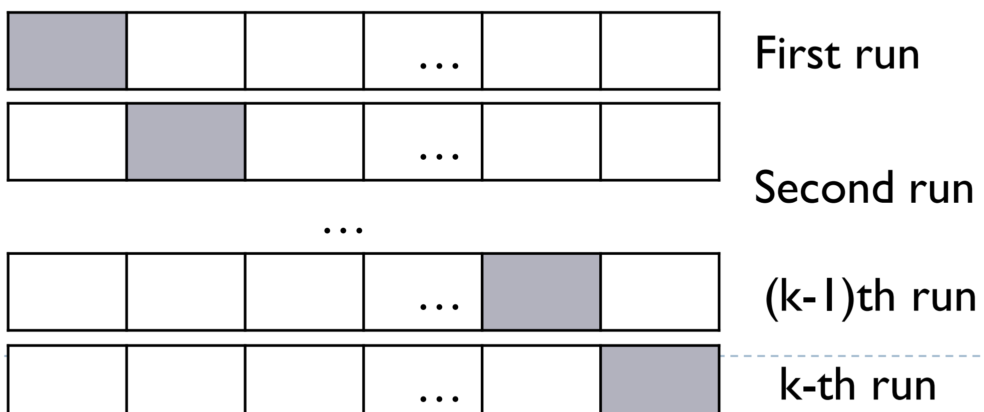
- $J_v(\widehat{\boldsymbol{w}})$ is likely to be an optimistic estimate of generalization error.
  - extra parameter (e.g., degree of polynomial) is fit to this set.

- Estimate generalization error for the test set
  - performance of the selected model is finally evaluated on the test set



Cross-Validation (CV): Evaluation

- k-fold cross-validation steps:
  - Shuffle the dataset and randomly partition training data into k groups of approximately equal size
  - for $i = 1$ to k
    - Choose the i-th group as the held-out validation group
    - Train the model on all but the i-th group of data
    - Evaluate the model on the held-out group
  - Performance scores of the model from k runs are **averaged**.
    - The average error rate can be considered as an estimation of the true performance.



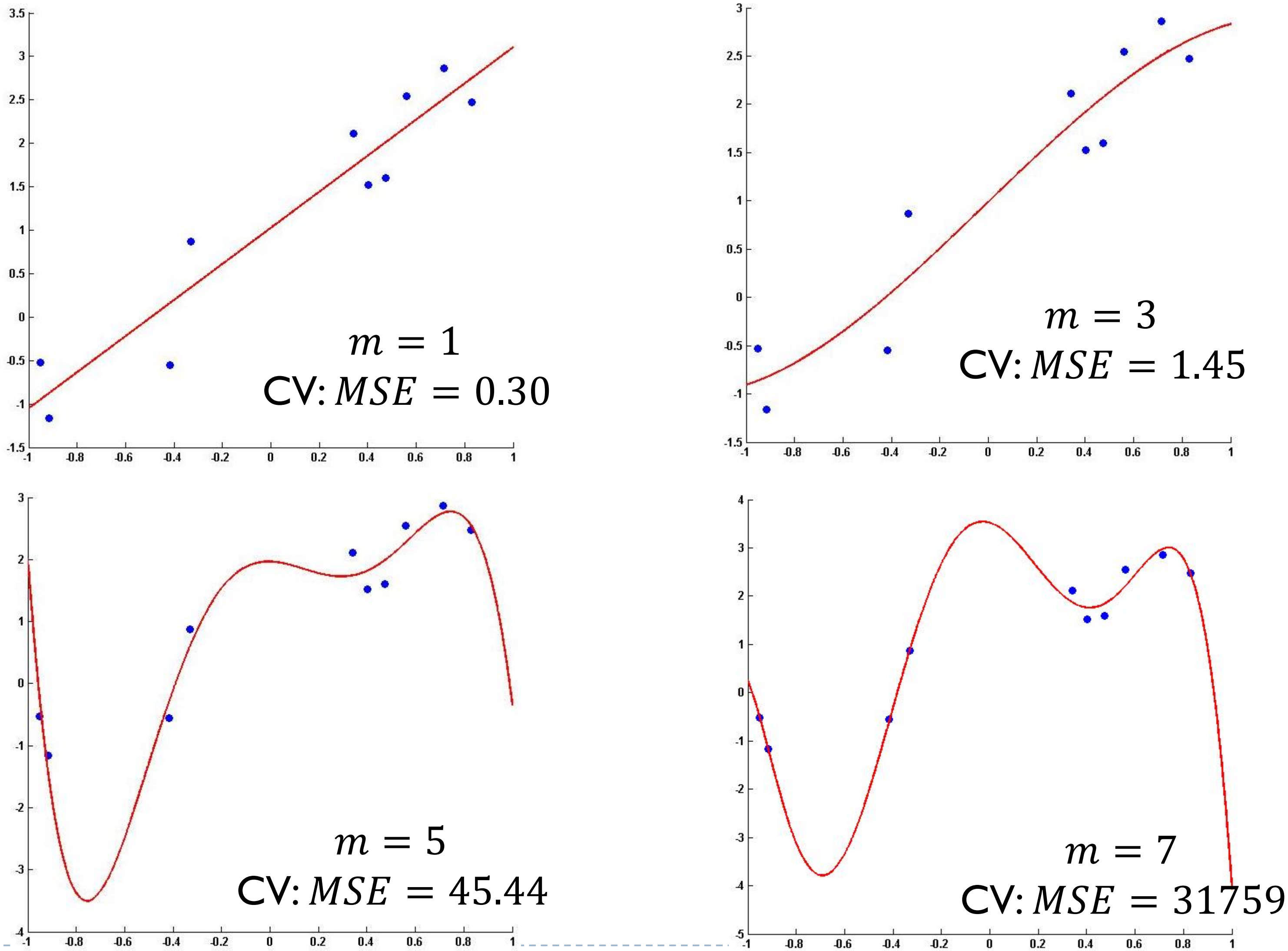
Cross-Validation (CV): Model Selection

- For each model we first find the average error find by CV.

- The model with **the best average performance** is selected.

Cross-validation: polynomial regression example

- 5-fold CV
- 100 runs
  - average



Leave-One-Out Cross Validation (LOOCV)

- When data is particularly scarce, cross-validation with $k = N$
  - Leave-one-out treats each training sample in turn as a test example and all other samples as the training set.

- Use for small datasets
  - When training data is valuable
  - LOOCV can be time expensive as N training steps are required.

Regularization

- Adding a penalty term in the cost function to discourage the coefficients from reaching large values.

- Ridge regression (weight decay):

$$J(\boldsymbol{w}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w}^T \boldsymbol{\phi}\left(\boldsymbol{x}^{(i)}\right)\right)^2 + \lambda \boldsymbol{w}^T \boldsymbol{w}$$

$$\widehat{\boldsymbol{w}} = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \boldsymbol{I}\right)^{-\mathbf{1}} \boldsymbol{\Phi}^T \boldsymbol{y}$$

Polynomial order

- Polynomials with larger m are becoming increasingly tuned to the random noise on the target values.
  - magnitude of the coefficients typically gets larger by increasing m.

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	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

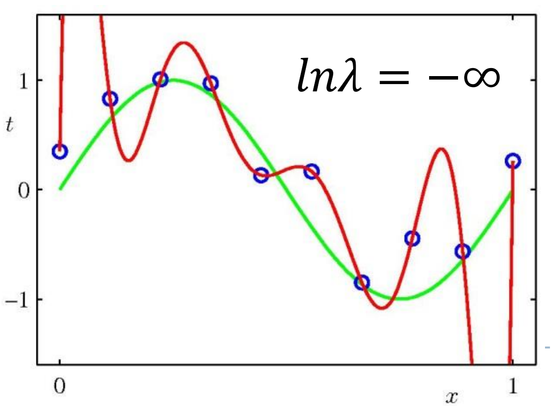
[Bishop]

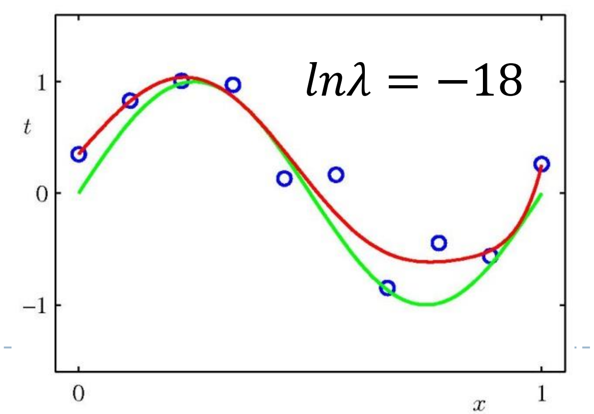
Regularization parameter

$m = 9$

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$\widehat{w}_0$	0.35	0.35	0.13
$\widehat{w}_1$	232.37	4.74	-0.05
$\widehat{w}_2$	-5321.83	-0.77	-0.06
$\widehat{w}_3$	48568.31	-31.97	-0.05
$\widehat{w}_4$	-231639.30	-3.89	-0.03
$\widehat{w}_5$	640042.26	55.28	-0.02
$\widehat{w}_6$	-1061800.52	41.32	-0.01
$\widehat{w}_7$	1042400.18	-45.95	-0.00
$\widehat{w}_8$	-557682.99	-91.53	0.00
$\widehat{w}_9$	125201.43	72.68	0.01

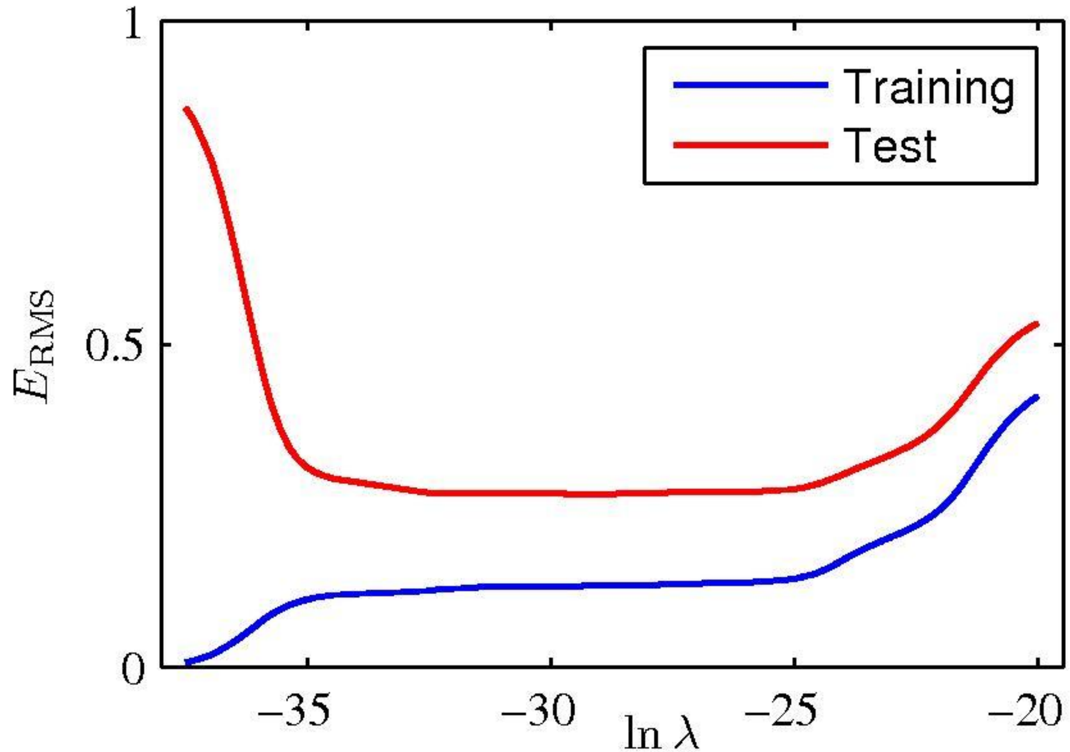
[Bishop]





Regularization parameter

- Generalization
  - λ now controls the effective complexity of the model and hence determines the degree of over-fitting



[Bishop]

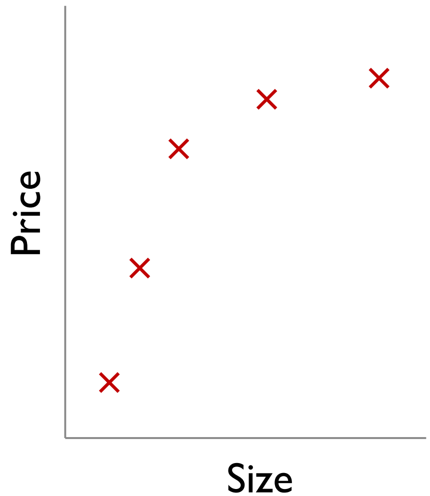
Choosing the regularization parameter

- A set of models with different values of λ.
- Find $\widehat{\boldsymbol{w}}$ for each model based on training data
- Find $J_v(\widehat{\boldsymbol{w}})$ (or $J_{cv}(\widehat{\boldsymbol{w}})$) for each model
  - $J_v(\boldsymbol{w}) = \frac{1}{n\_v} \sum_{i \in v\_set} \left(y^{(i)} - f\left(x^{(i)}; \boldsymbol{w}\right)\right)^2$
- Select the model with the best $J_v(\widehat{\boldsymbol{w}})$ (or $J_{cv}(\widehat{\boldsymbol{w}})$)

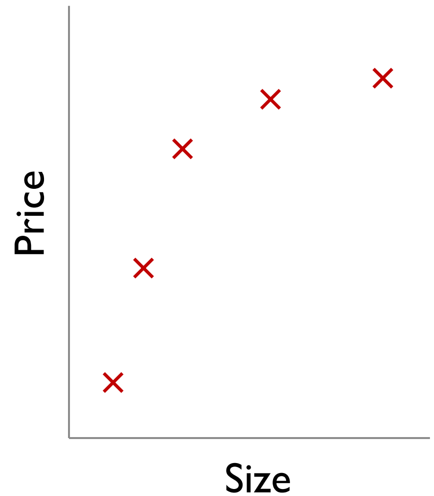
The approximation-generailization trade-off

- Small true error shows good approximation of f out of sample
- More complex $\mathcal{H} \Rightarrow$ better chance of approximating f
- Less complex $\mathcal{H} \Rightarrow$ better chance of generalization out of f

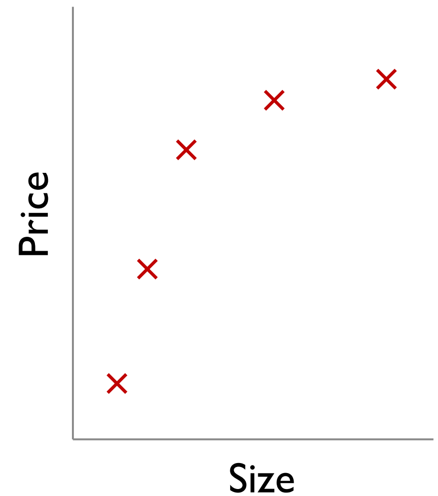
Complexity of Hypothesis Space: Example



$w_0 + w_1 x$



$w_0 + w_1 x + w_2 x^2$



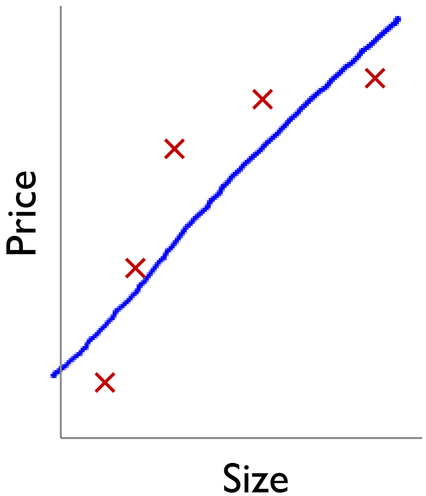
$w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$

Less complex $\mathcal{H}$

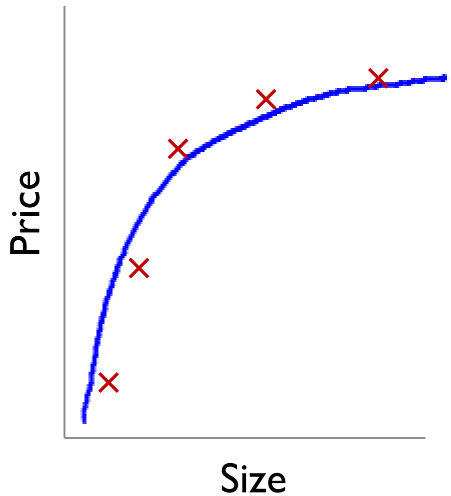
More complex $\mathcal{H}$

This example has been adapted from: Prof. Andrew Ng's slides

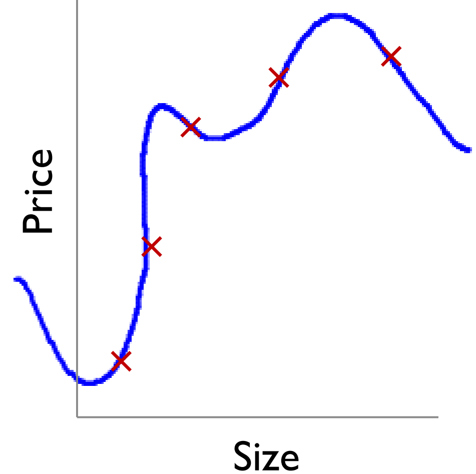
Complexity of Hypothesis Space: Example



$w_0 + w_1 x$



$w_0 + w_1 x + w_2 x^2$



$w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$

Less complex $\mathcal{H}$

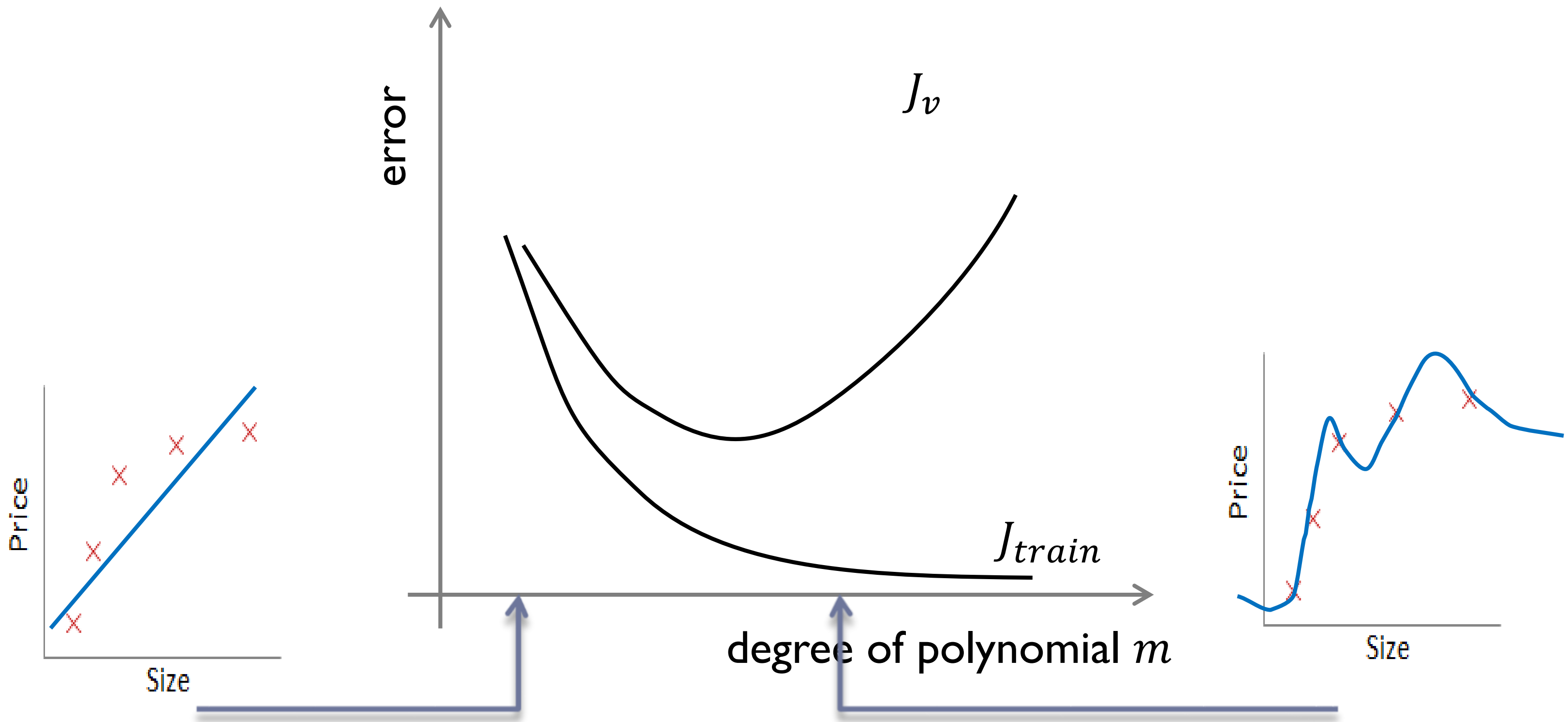
More complex $\mathcal{H}$

This example has been adapted from: Prof. Andrew Ng's slides

Complexity of Hypothesis Space: Example

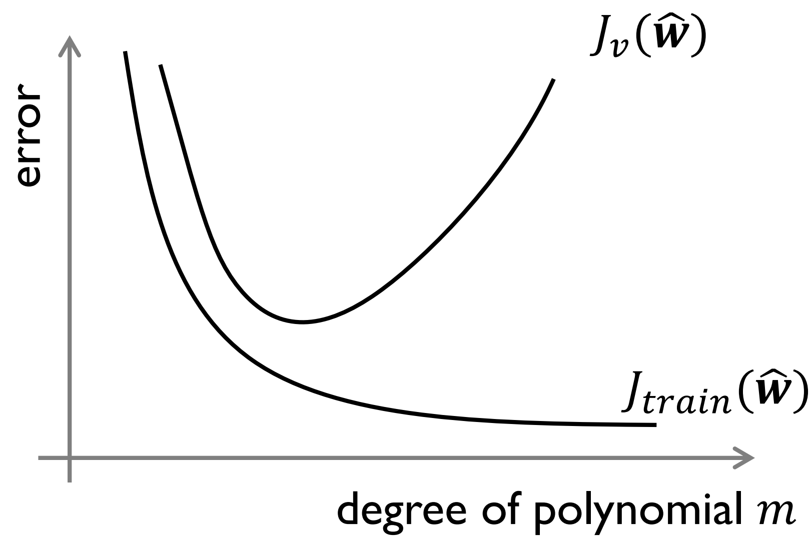
$$J_v(\boldsymbol{w}) = \frac{1}{n\_v}\sum_{i\in val\_set}\left(y^{(i)} - f\left(\boldsymbol{x}^{(i)};\boldsymbol{w}\right)\right)^2$$

$$J_{train}(\boldsymbol{w}) = \frac{1}{n\_train}\sum_{i\in train\_set}\left(y^{(i)} - f\left(\boldsymbol{x}^{(i)};\boldsymbol{w}\right)\right)^2$$



Complexity of Hypothesis Space

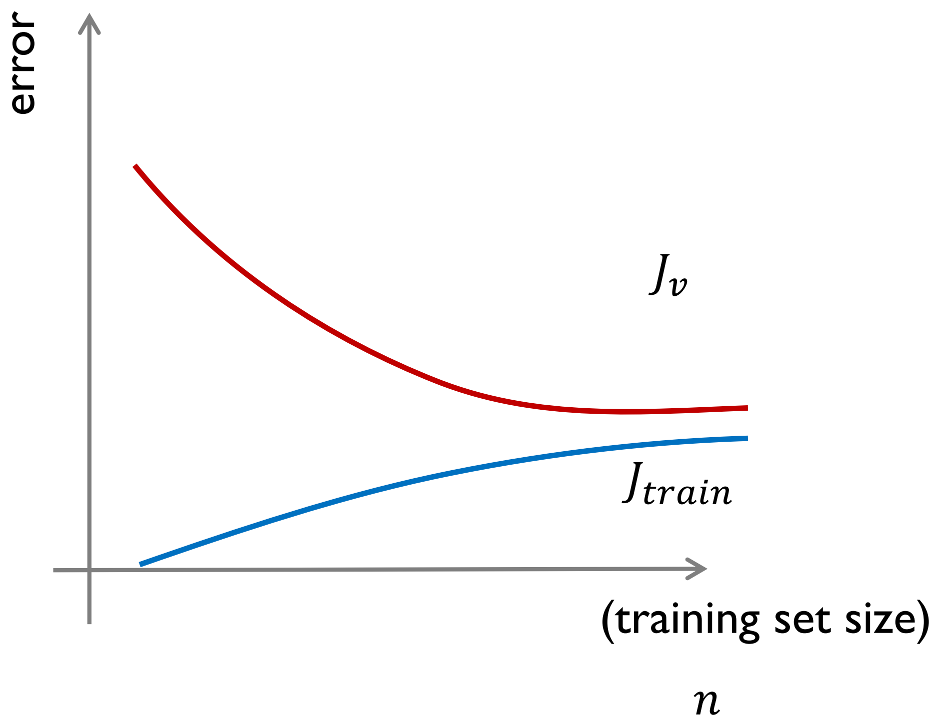
- Less complex $\mathcal{H}$:
  - $J_{train}(\widehat{\boldsymbol{w}}) \approx J_v(\widehat{\boldsymbol{w}})$ and $J_{train}(\widehat{\boldsymbol{w}})$ is very high
- More complex $\mathcal{H}$:
  - $J_{train}(\widehat{\boldsymbol{w}}) \ll J_v(\widehat{\boldsymbol{w}})$ and $J_{train}(\widehat{\boldsymbol{w}})$ is low



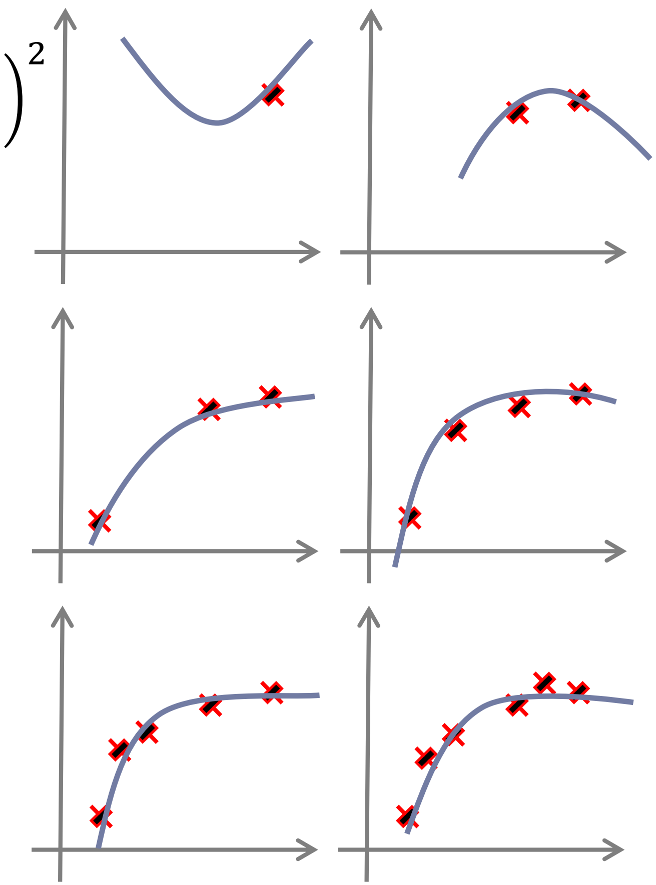
Size of training set

$$J_v(\boldsymbol{w}) = \frac{1}{n\_v} \sum_{i \in val\_set} \left(y^{(i)} - f\left(x^{(i)}; \boldsymbol{w}\right) \right)^2$$

$$J_{train}(\boldsymbol{w}) = \frac{1}{n\_train} \sum_{i \in train\_set} \left(y^{(i)} - f\left(x^{(i)}; \boldsymbol{w}\right) \right)^2$$

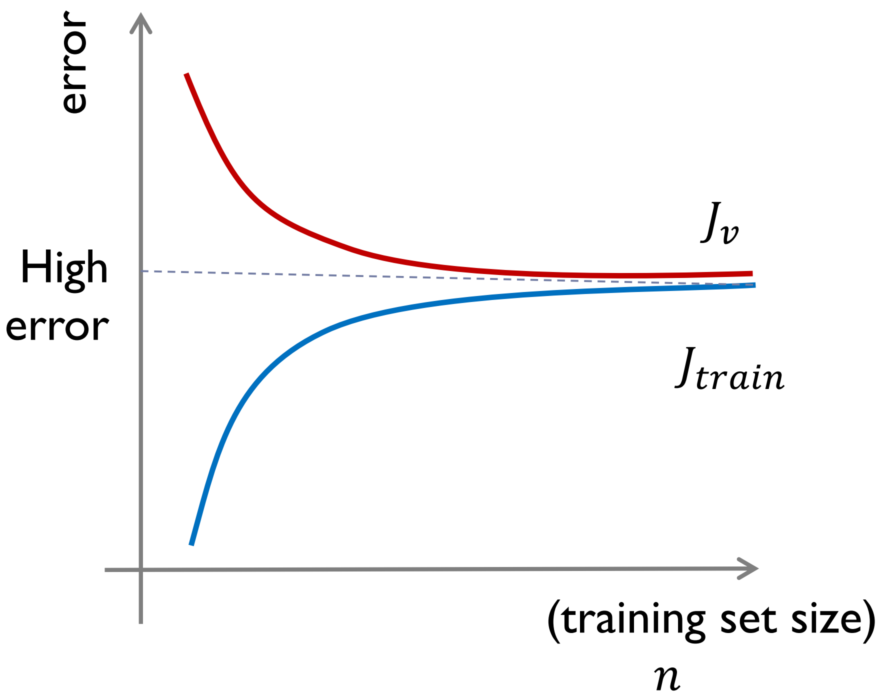


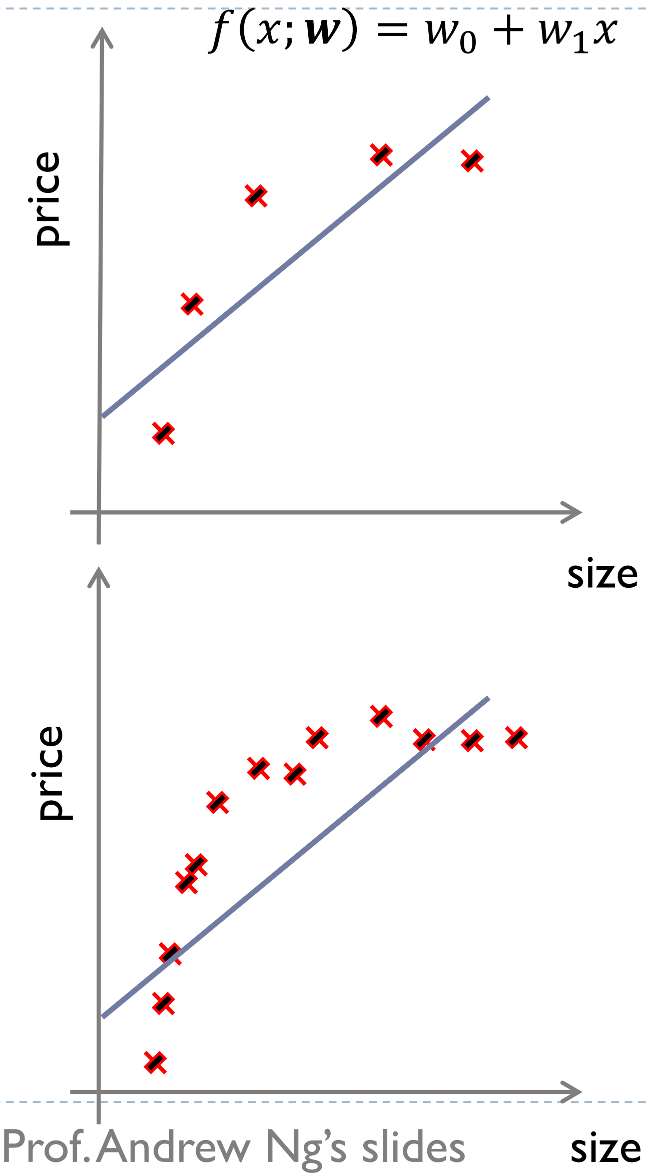
$$f(x; \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2$$



This slide has been adapted from: Prof. Andrew Ng's slides

Less complex $\mathcal{H}$



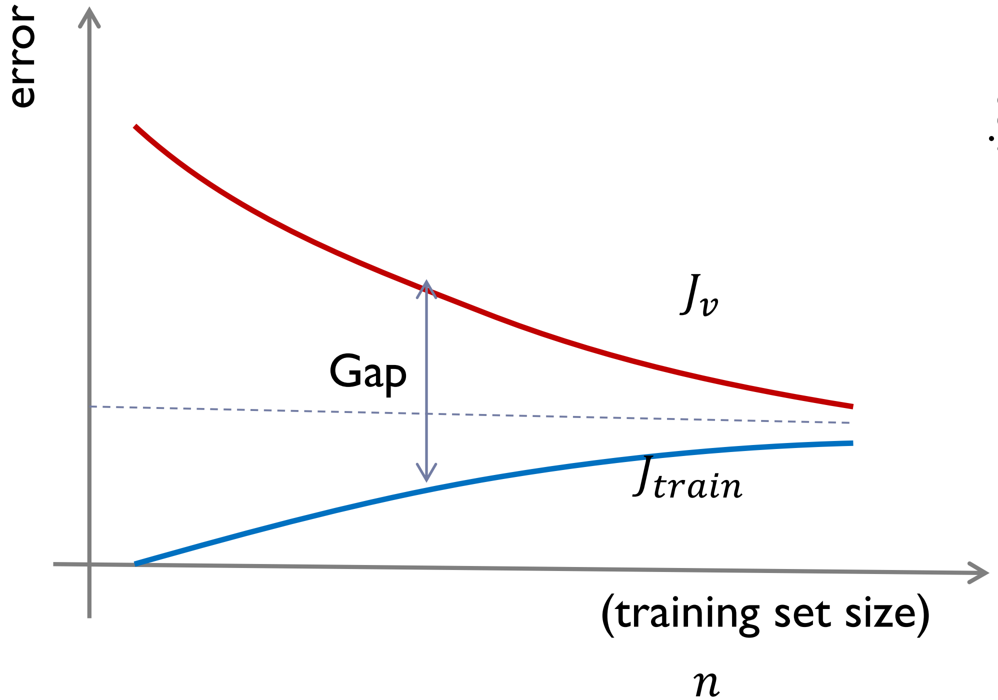


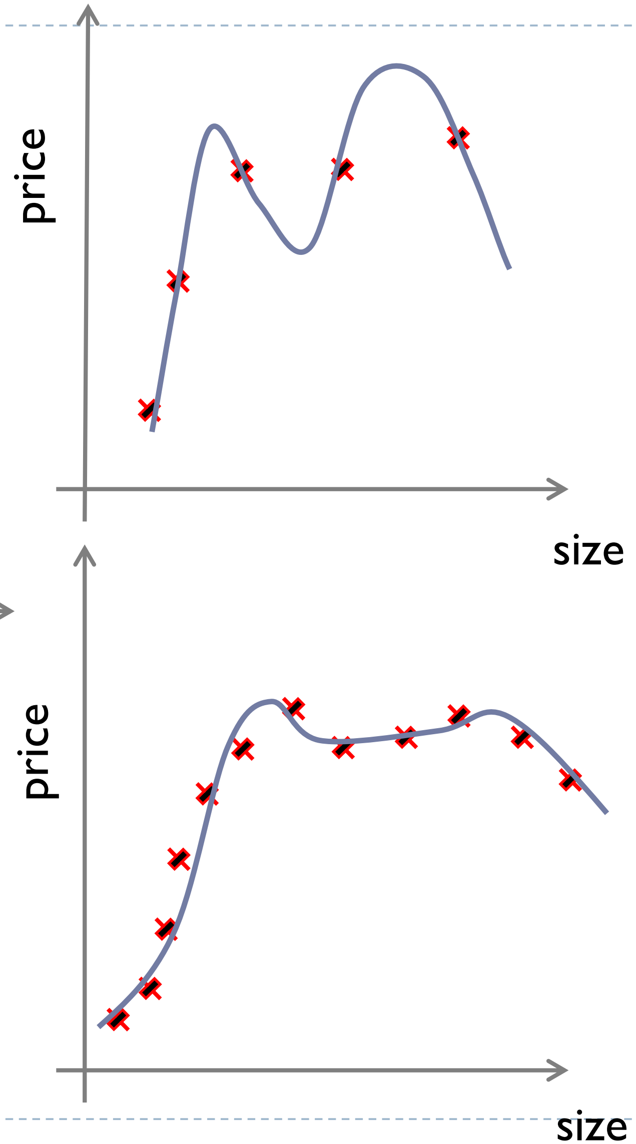
If model is very simple, getting more training data will not (by itself) help much.

This slide has been adapted from: Prof. Andrew Ng's slides

More complex $\mathcal{H}$

$$f(x; \boldsymbol{w}) = w_0 + w_1 x + \cdots w_{10} x^{10}$$





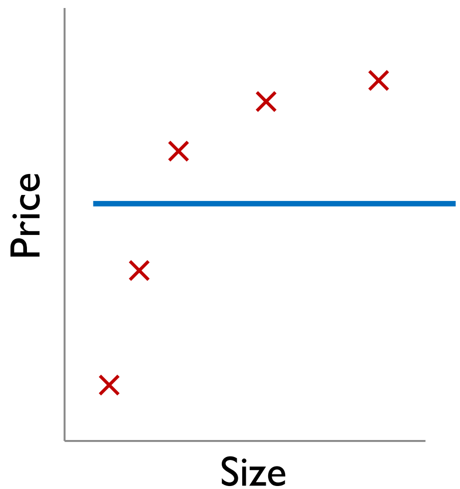
For more complex models, getting more training data is usually helps.

This slide has been adapted from: Prof. Andrew Ng's slides

Regularization: Example

$$f(x; \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$$

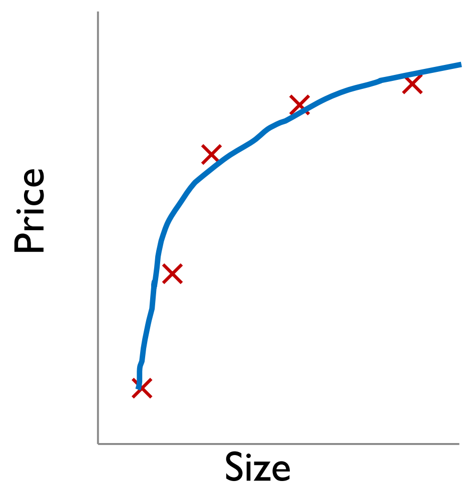
$$J(\boldsymbol{w}) = \frac{1}{n}\left(\sum_{i=1}^{n}\left(y^{(i)} - f\left(x^{(i)}; \boldsymbol{w}\right)\right)^2 + \lambda \boldsymbol{w}^T \boldsymbol{w}\right)$$



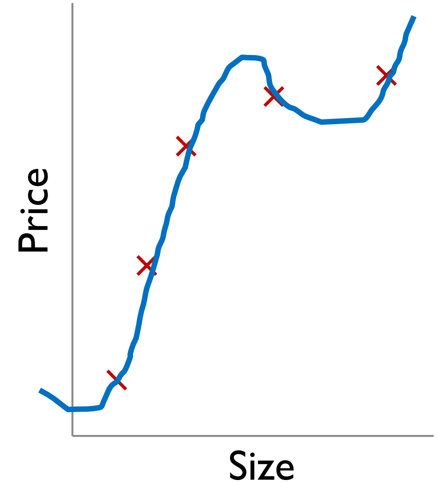
Large λ

(Prefer to more simple models)

$$w_1 = w_2 \approx 0$$



Intermediate λ



Small λ

(Prefer to more complex models)

$$\lambda = 0$$

This example has been adapted from: Prof. Andrew Ng's slides

Model complexity: Bias-variance trade-off

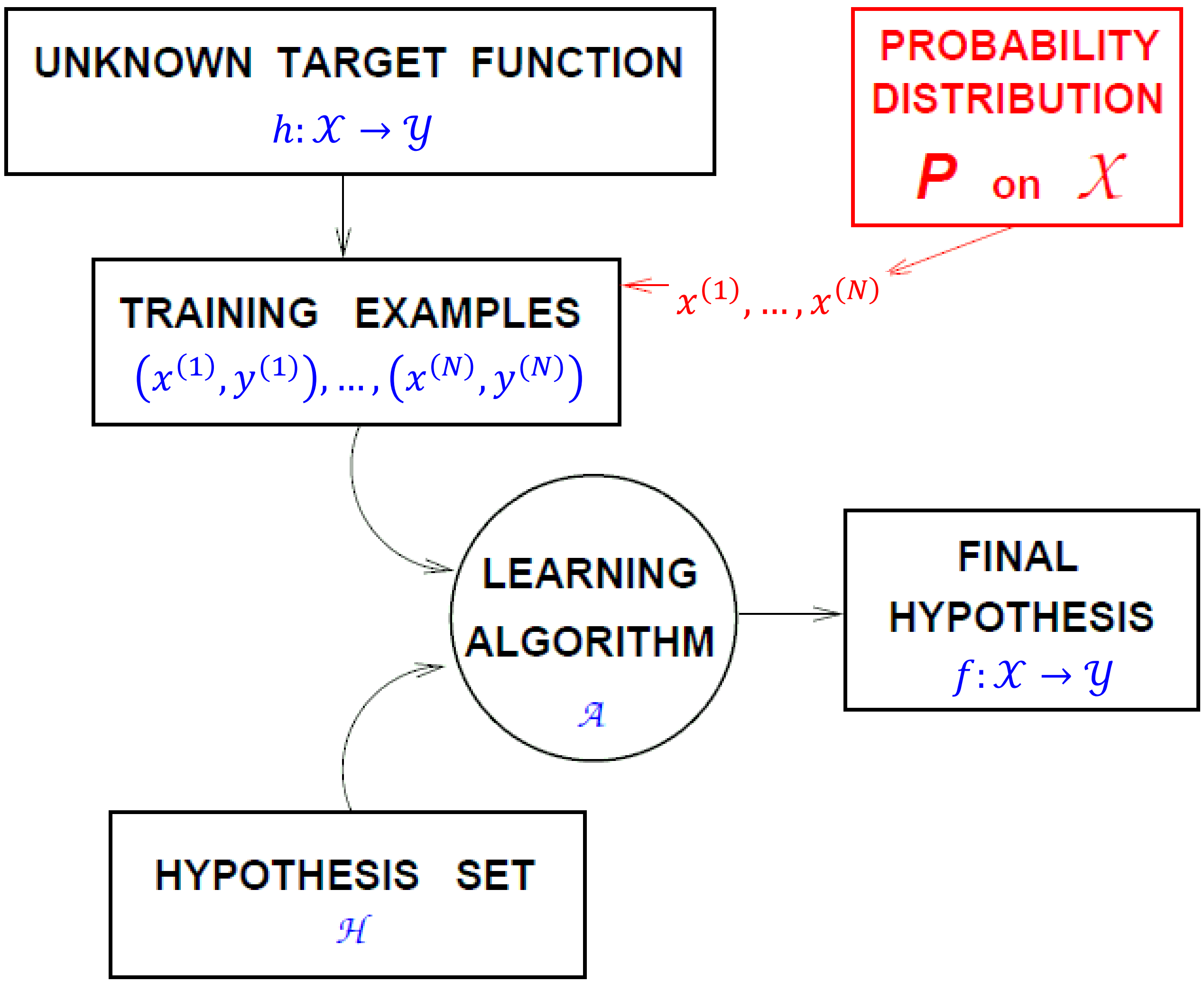
- Least squares, can lead to severe over-fitting if complex models are trained using data sets of limited size.

- A frequentist viewpoint of the model complexity issue, known as the *bias-variance* trade-off.

Formal discussion on bias, variance, and noise

- Best unrestricted regression function
- Noise
- Bias and variance

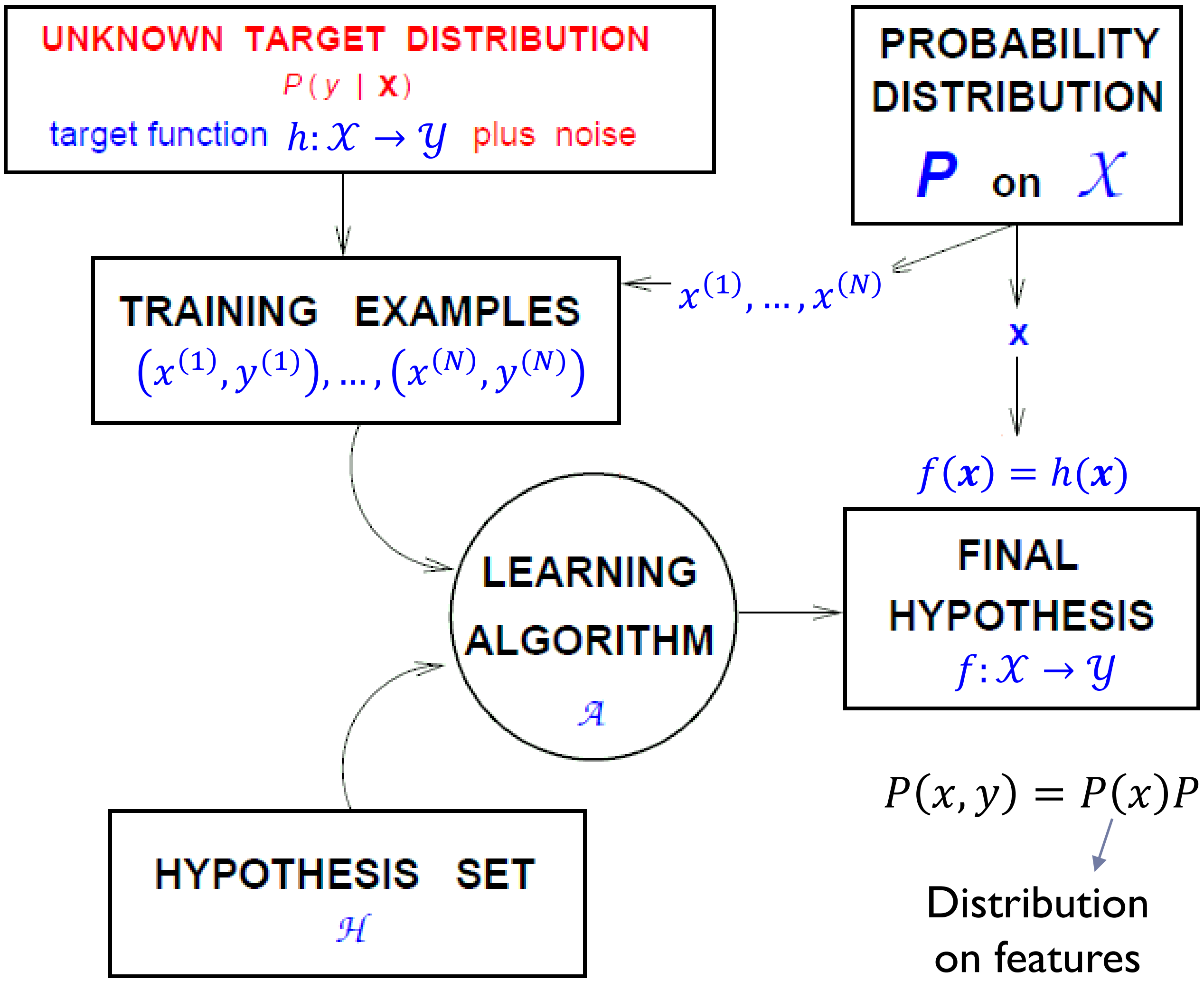
The learning diagram: deterministic target



[Y.S. Abou Mostafa, et. al]

The learning diagram including noisy target

- Type



$$P(x, y) = P(x)P(y|x)$$

Distribution on features

Target distribution

[Y.S. Abou Mostafa, et. al]

Best unrestricted regression function

- If we know the joint distribution $P(\boldsymbol{x}, y)$ and no constraints on the regression function?
  - cost function: mean squared error

$$h^* = \underset{h:\mathbb{R}^d \to \mathbb{R}}{\operatorname{argmin}} \mathbb{E}_{\boldsymbol{x},y}\left[\left(y - h(\boldsymbol{x})\right)^2\right]$$

$$h^*(\boldsymbol{x}) = \mathbb{E}_{y|\boldsymbol{x}}[y]$$

Best unrestricted regression function: Proof

$$\mathbb{E}_{\boldsymbol{x},y}\left[\left(y-h(\boldsymbol{x})\right)^2\right]=\iint\left(y-h(\boldsymbol{x})\right)^2 p(\boldsymbol{x},y)d\boldsymbol{x}dy$$

- For each $\boldsymbol{x}$ separately minimize loss since $h(\boldsymbol{x})$ can be chosen independently for each different $\boldsymbol{x}$:

$$\frac{\delta\mathbb{E}_{\boldsymbol{x},y}\left[\left(y-h(\boldsymbol{x})\right)^2\right]}{\delta h(\boldsymbol{x})}=\int 2\left(y-h(\boldsymbol{x})\right)p(\boldsymbol{x},y)dy=0$$

$$\Rightarrow h(\boldsymbol{x})=\frac{\int yp(\boldsymbol{x},y)dy}{\int p(\boldsymbol{x},y)dy}=\frac{\int yp(\boldsymbol{x},y)dy}{p(\boldsymbol{x})}=\int yp(y|\boldsymbol{x})dy=\mathbb{E}_{y|\boldsymbol{x}}[y]$$

$$\Longrightarrow h^*(\boldsymbol{x})=\mathbb{E}_{y|\boldsymbol{x}}[y]$$

Error decomposition

$(\boldsymbol{x}, y) \sim P$

$h(\boldsymbol{x})$: minimizes the expected loss

$$E_{true}\left(f_{\mathcal{D}}(\boldsymbol{x})\right) = \mathbb{E}_{\boldsymbol{x},y}[(f_{\mathcal{D}}(\boldsymbol{x}) - y)^2]$$

Expected loss

$$= \mathbb{E}_{\boldsymbol{x},y}\,[(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x}) + h(\boldsymbol{x}) - y)^2]$$

$$= \mathbb{E}_{\boldsymbol{x}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right] + \mathbb{E}_{\boldsymbol{x},y}[(h(\boldsymbol{x}) - \boldsymbol{y})^2]$$

$$+2\underbrace{\mathbb{E}_{\boldsymbol{x},y}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)(h(\boldsymbol{x}) - y)\right]}_{\mathbb{E}_{\boldsymbol{x}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)\underbrace{\mathbb{E}_{y|\boldsymbol{x}}[(h(\boldsymbol{x}) - y)]}_{0}\right]}$$

Error decomposition

$(\boldsymbol{x}, y) \sim P$

$h(\boldsymbol{x})$: minimizes the expected loss

$$E_{true}\left(f_{\mathcal{D}}(\boldsymbol{x})\right) = \mathbb{E}_{\boldsymbol{x},y}[(f_{\mathcal{D}}(\boldsymbol{x}) - y)^2]$$

$$= \mathbb{E}_{\boldsymbol{x},y}\,[(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x}) + h(\boldsymbol{x}) - y)^2]$$

$$= \mathbb{E}_{\boldsymbol{x}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right] + \underbrace{\mathbb{E}_{\boldsymbol{x},y}[(h(\boldsymbol{x}) - \boldsymbol{y})^2]}_{\text{noise}}$$

$$+\,0$$

- Noise shows the irreducible minimum value of the loss function

Expectation of true error

$$E_{true}\big(f_{\mathcal{D}}(\boldsymbol{x})\big) = \mathbb{E}_{\boldsymbol{x},y}[(f_{\mathcal{D}}(\boldsymbol{x}) - y)^2]$$
$$= \mathbb{E}_{\boldsymbol{x}}\left[\big(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\big)^2\right] + noise$$

$$\mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{x}}\left[\big(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\big)^2\right]\right]$$
$$= \mathbb{E}_{\boldsymbol{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\big(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\big)^2\right]\right]$$

We now want to focus on $\mathbb{E}_{\mathcal{D}}\left[\big(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\big)^2\right]$.

The average hypothesis

$$\bar{f}(\boldsymbol{x}) \equiv E_{\mathcal{D}}[f_{\mathcal{D}}(\boldsymbol{x})]$$

$$\bar{f}(\boldsymbol{x}) \approx \frac{1}{K}\sum_{k=1}^{K} f_{\mathcal{D}^{(k)}}(\boldsymbol{x})$$

K training sets (of size N) sampled from $P(\boldsymbol{x}, y)$:
$\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, \dots, \mathcal{D}^{(K)}$

Using the average hypothesis

$$\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right]$$

$$= \mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}) + \bar{f}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right]$$

$$= \mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2 + \left(\bar{f}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right.$$

Bias and variance

$$\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2\right]}_{\mathrm{var}(\boldsymbol{x})} + \underbrace{\left(\bar{f}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2}_{\mathrm{bias}(\boldsymbol{x})}$$

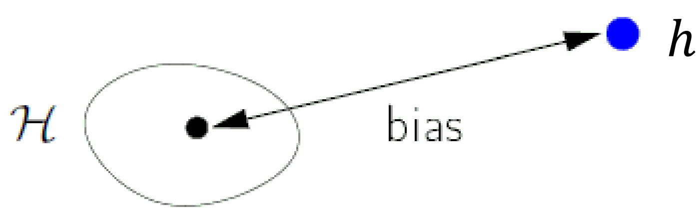
$$\mathbb{E}_{\boldsymbol{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - h(\boldsymbol{x})\right)^2\right]\right] = \mathbb{E}_{\boldsymbol{x}}[\mathrm{var}(\boldsymbol{x}) + \mathrm{bias}(\boldsymbol{x})]$$

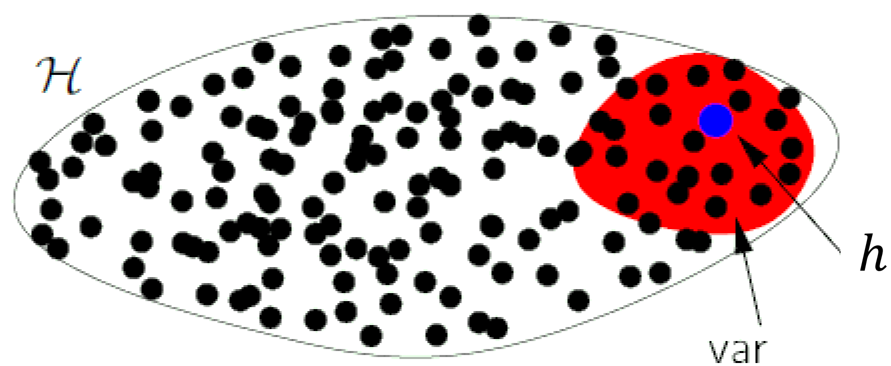
$$= \mathrm{var} + \mathrm{bias}$$

Bias-variance trade-off

$$\text{var} = \mathbb{E}_{\boldsymbol{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\boldsymbol{x}) - \bar{f}(\boldsymbol{x})\right)^2\right]\right]$$

$$\text{bias} = \mathbb{E}_{\boldsymbol{x}}\left[\bar{f}(\boldsymbol{x}) - h(\boldsymbol{x})\right]$$



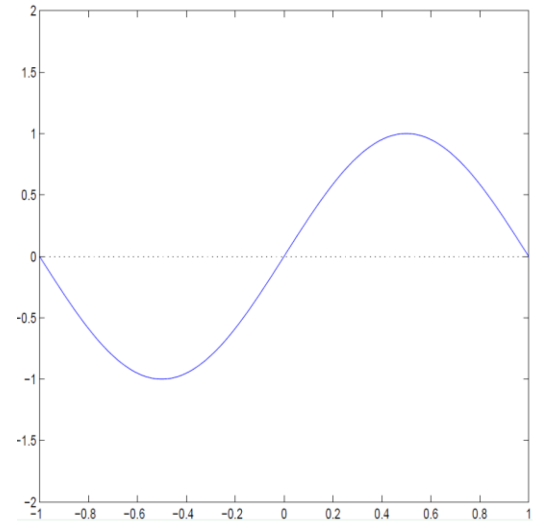


More complex $\mathcal{H}$ $\Rightarrow$ lower bias but higher variance

[Y.S. Abou Mostafa, et. al]

Example: sin target

- Only two training example $N = 2$

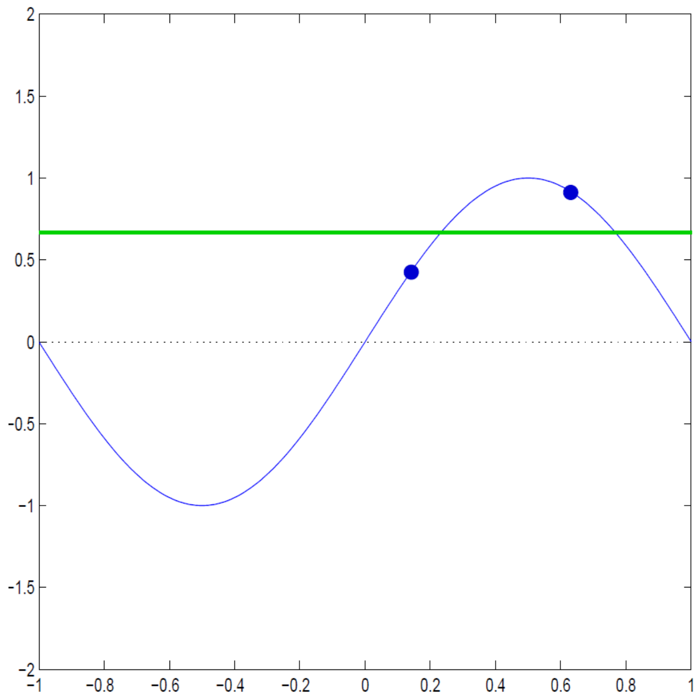


- Two models used for learning:
  - $\mathcal{H}_0 : f(x) = b$
  - $\mathcal{H}_1 : f(x) = ax + b$

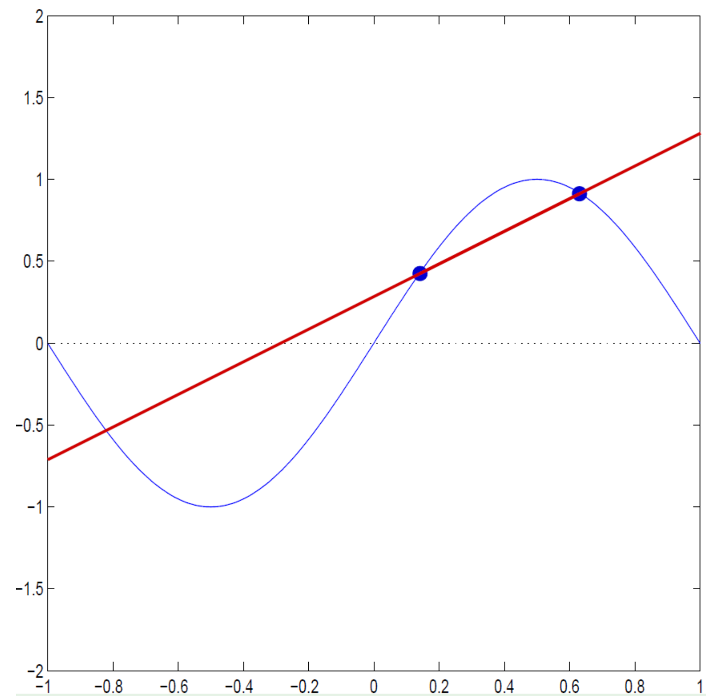
- Which is better $\mathcal{H}_0$ or $\mathcal{H}_1$?

Learning from a training set

$\mathcal{H}_0$

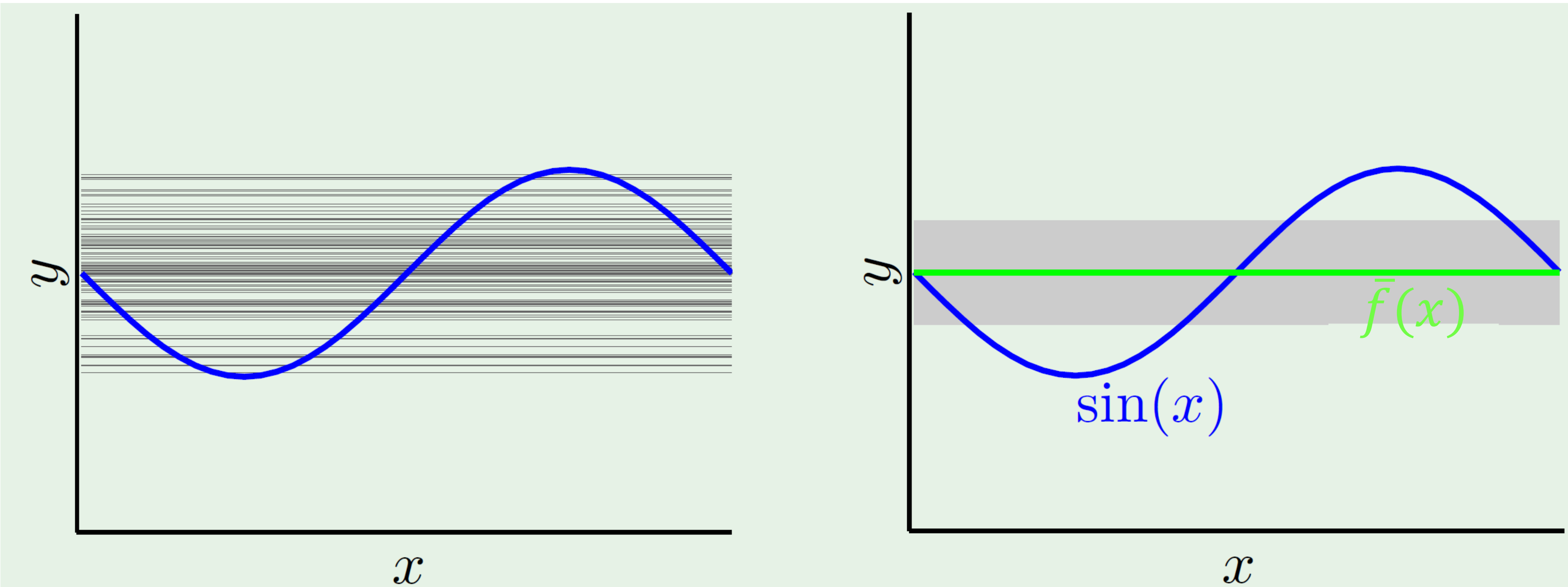


$\mathcal{H}_1$



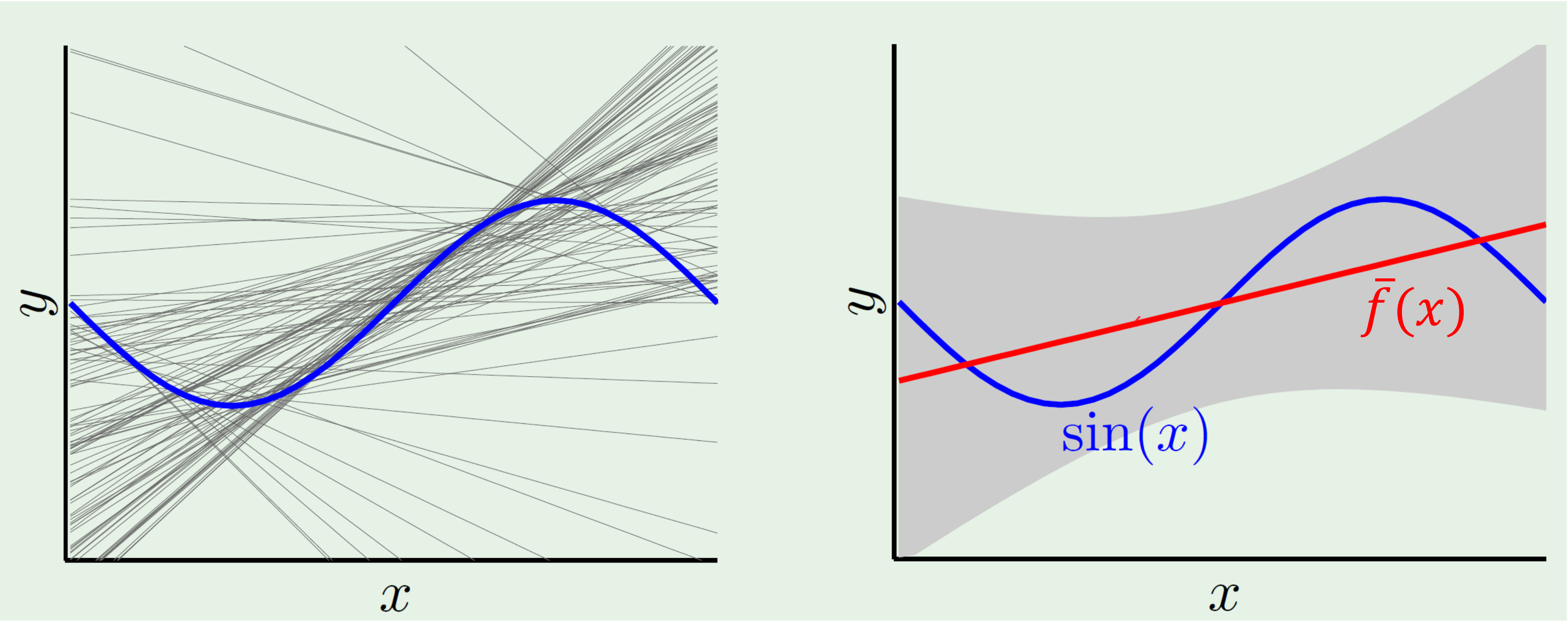
[Y.S. Abou Mostafa, et. al]

Variance $\mathcal{H}_0$



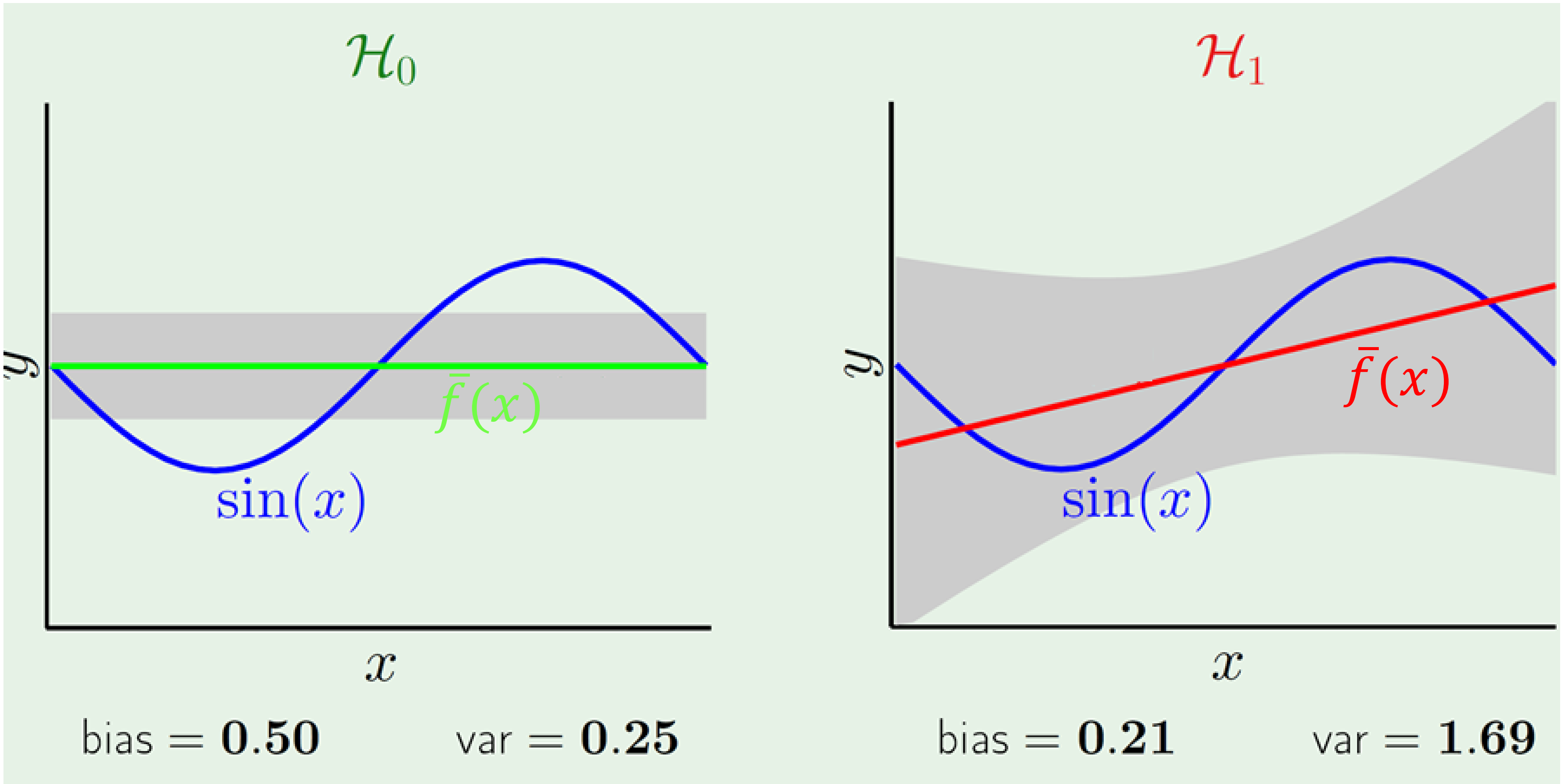
[Y.S. Abou Mostafa, et. al]

Variance $\mathcal{H}_1$



[Y.S. Abou Mostafa, et. al]

Which is better?



[Y.S. Abou Mostafa, et. al]

Lesson

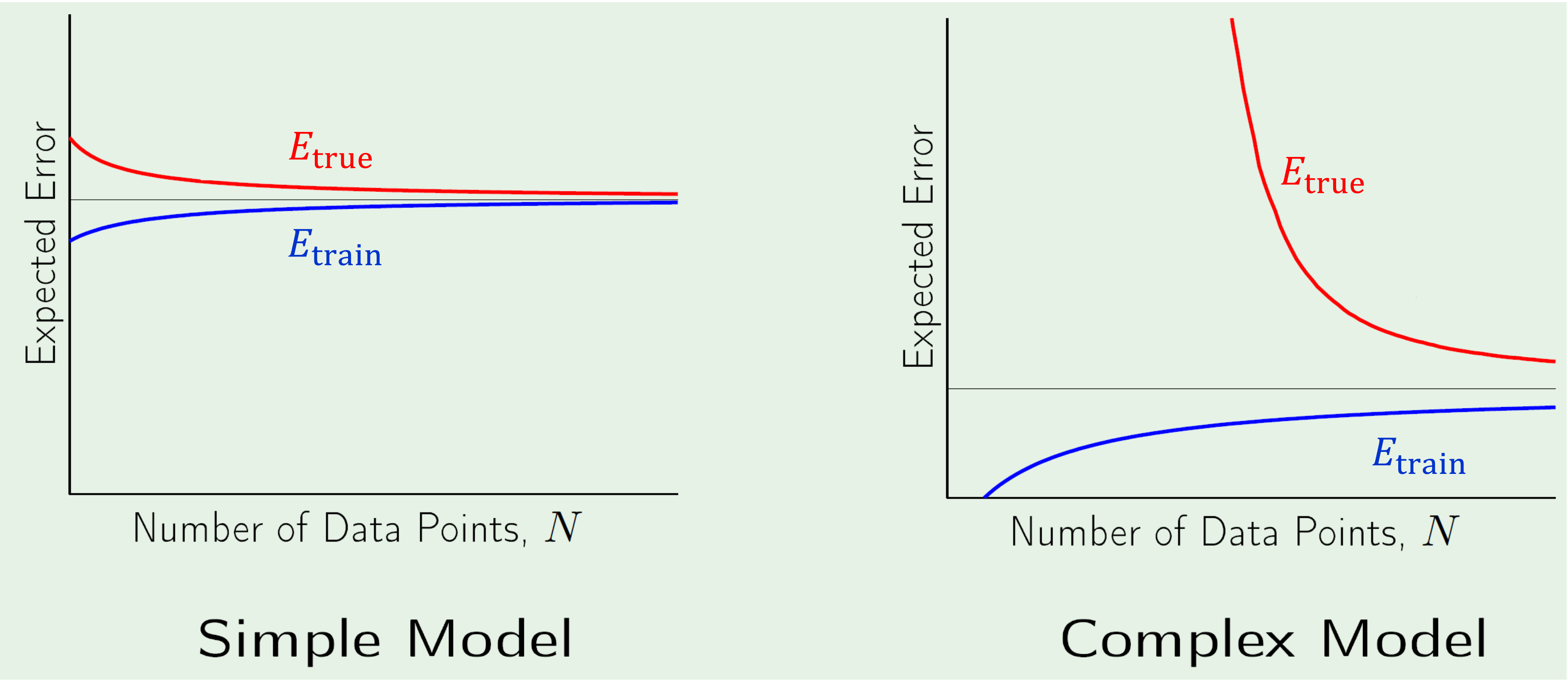
Match the **model complexity**

to the data sources

not to the complexity of the target function.

Expected training and true error curves

- Errors vary with the number of training samples

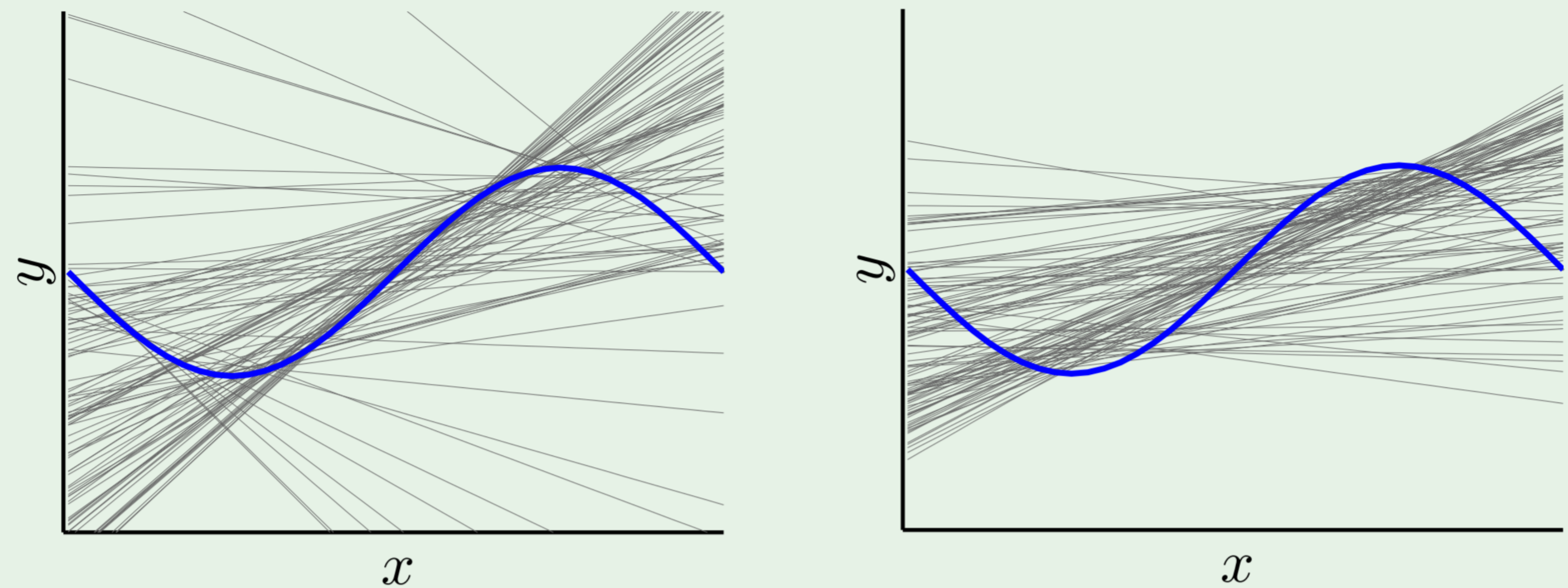


expected true error: $\mathbb{E}_{\mathcal{D}}\left[E_{true}\left(f_{\mathcal{D}}(\boldsymbol{x})\right)\right]$

expected training error: $\mathbb{E}_{\mathcal{D}}\left[E_{train}\left(f_{\mathcal{D}}(\boldsymbol{x})\right)\right]$

[Y.S. Abou Mostafa, et. al]

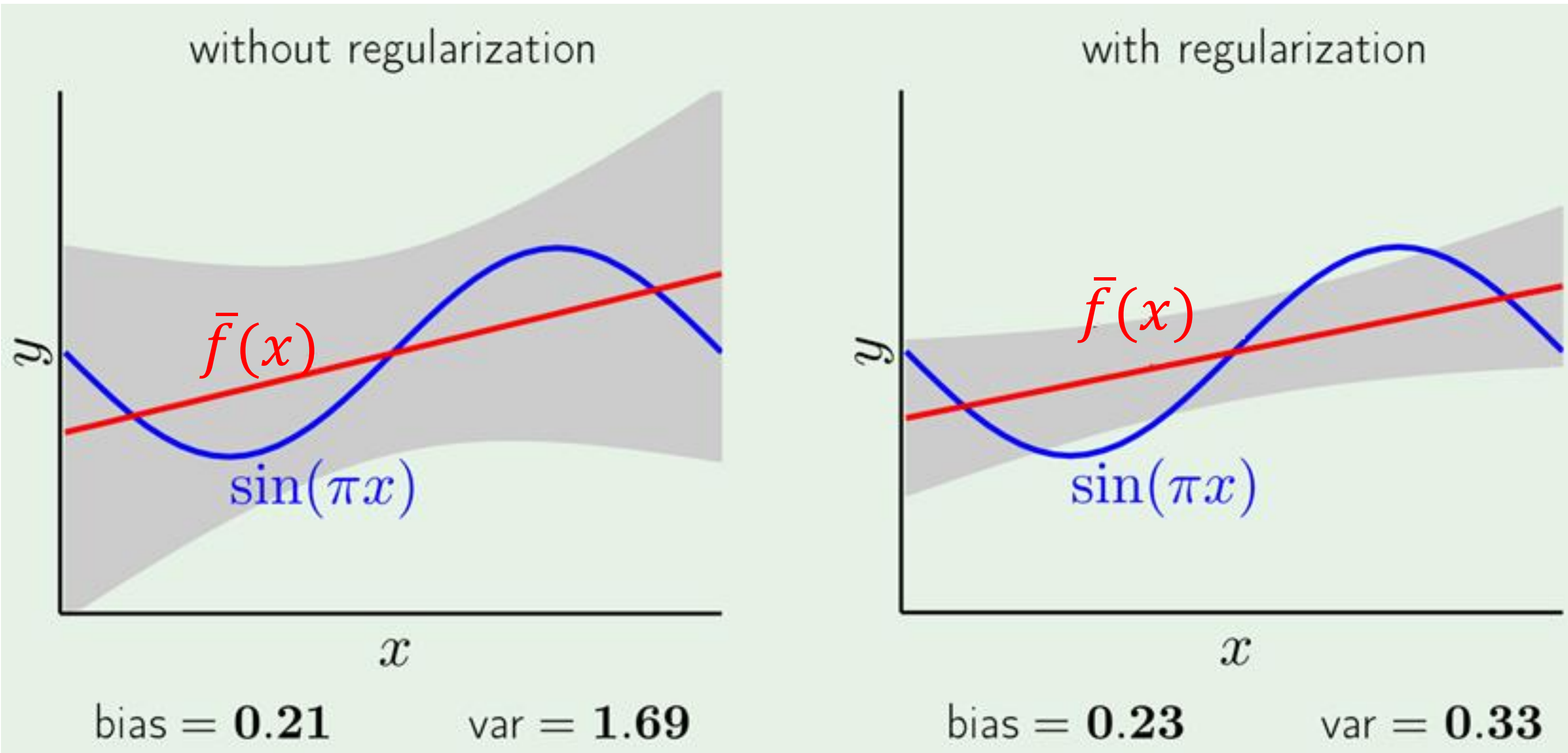
Regularization



without regularization   with regularization

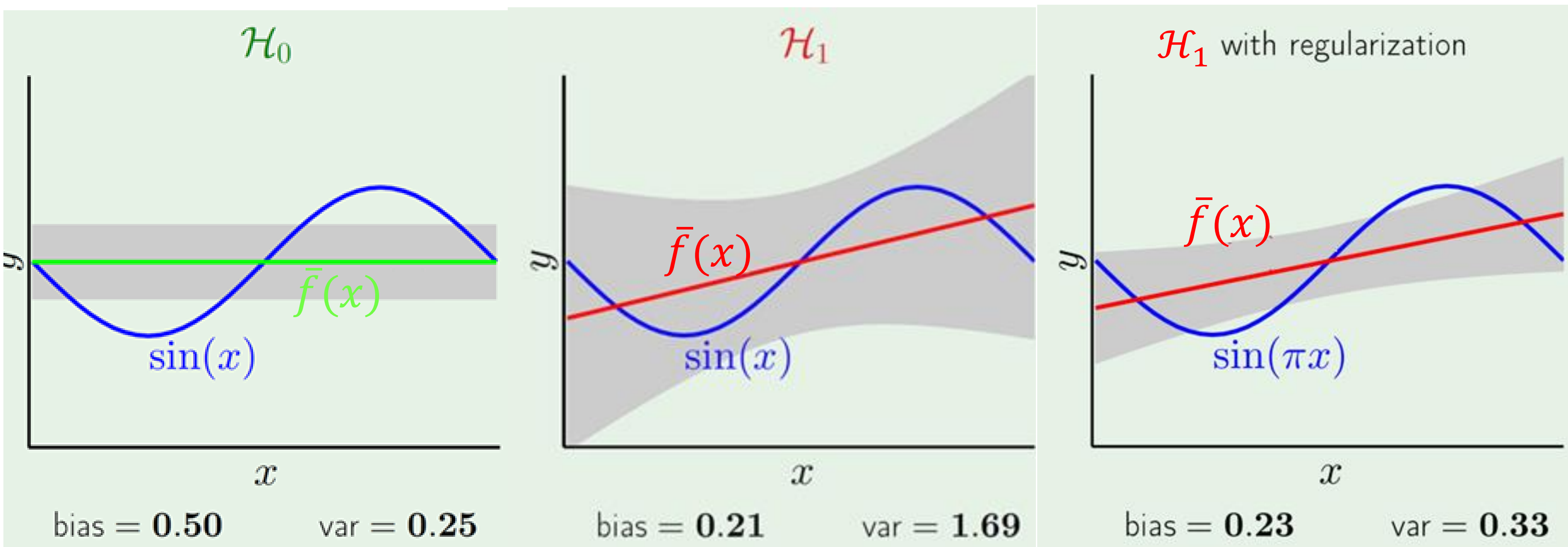
[Y.S. Abou Mostafa, et. al]

Regularization: bias and variance



[Y.S. Abou Mostafa, et. al]

Winner of $\mathcal{H}_0$, $\mathcal{H}_1$, and $\mathcal{H}_1$ with regularization



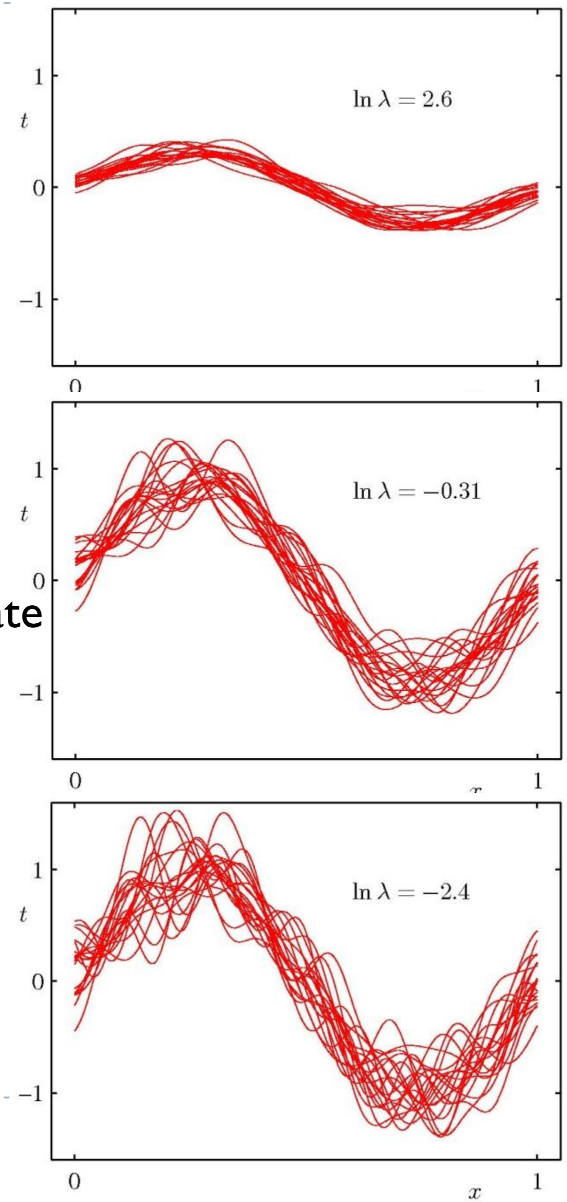
[Y.S. Abou Mostafa, et. al]

Regularization and bias/variance

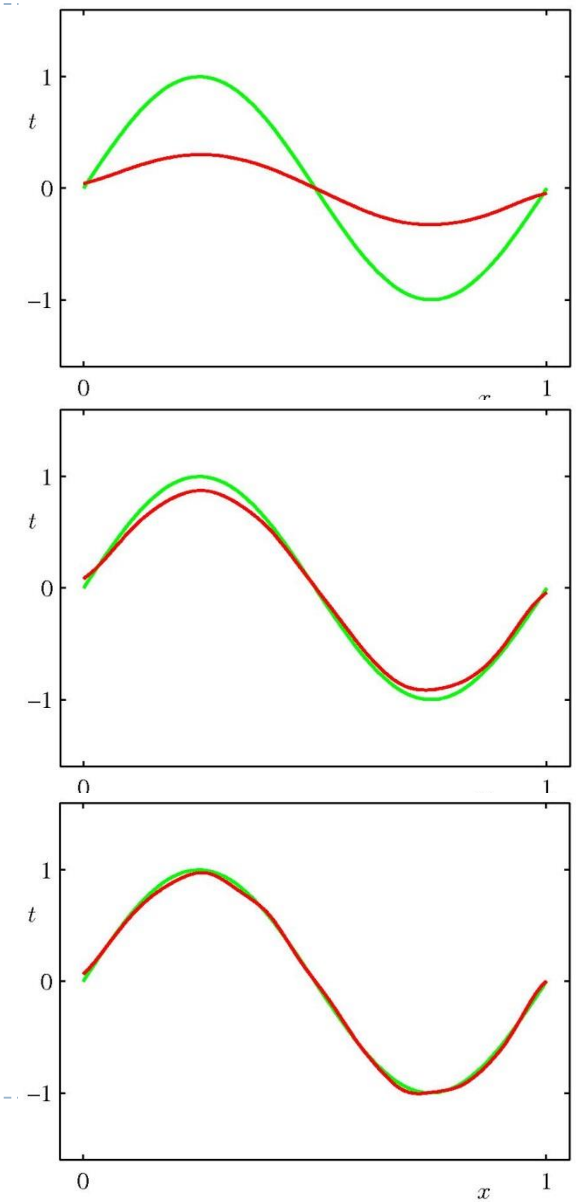
λ is large

λ is intermediate

λ is small

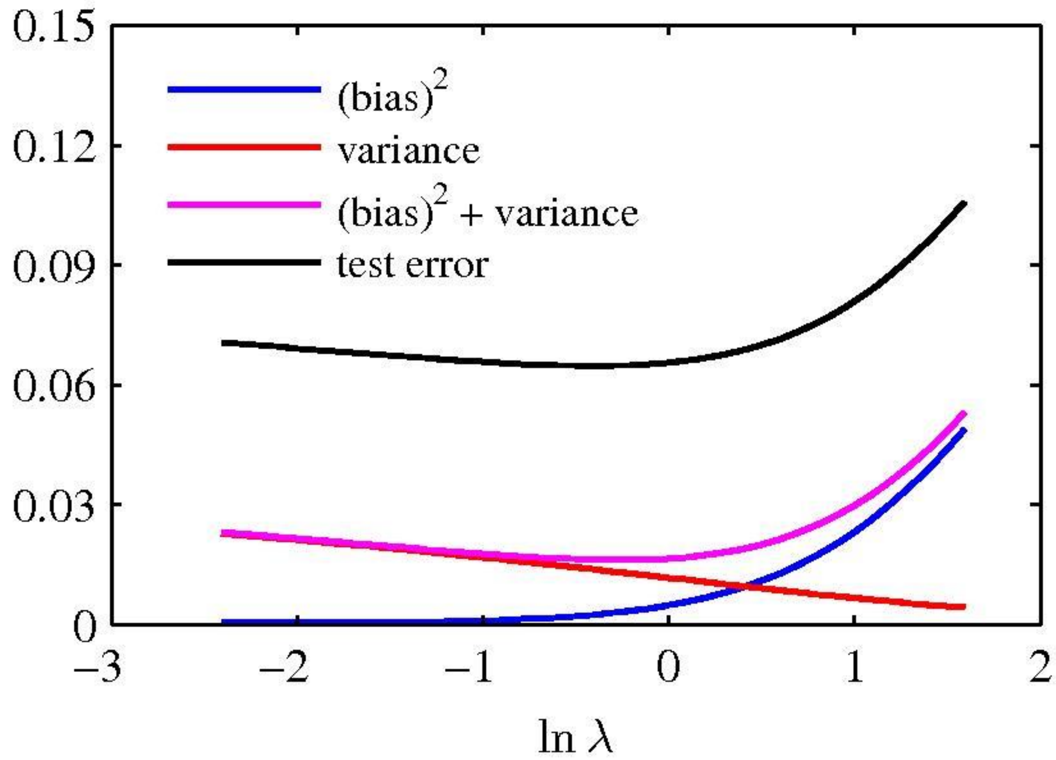


$L = 100$ data sets
$n = 25$
$m = 25$



[Bishop]

Learning curves of bias, variance, and noise



[Bishop]

Bias-variance decomposition: summary

- The noise term is unavoidable.
- The terms we are interested in are bias and variance.
- The approximation-generalization trade-off is seen in the bias-variance decomposition.

Resources

- C. Bishop, "Pattern Recognition and Machine Learning", Chapter 1.1, 1.3, 3.1, 3.2.
- Yaser S. Abu-Mostafa, Malik Maghdon-Ismail, and Hsuan Tien Lin, "**Learning from Data**", Chapter 2.3, 3.2, 3.4.