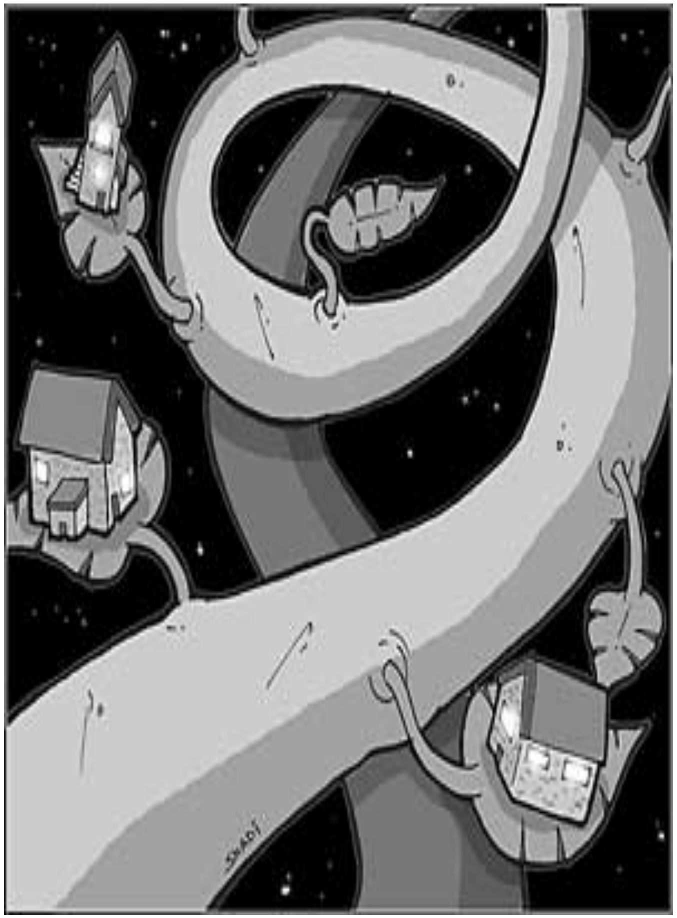
Overfitting and Cross-Validation

Le Song

Machine Learning
CSE/ISYE 6740, Fall 2019

Apartment hunting



- Suppose you are to move to Atlanta
- And you want to find the **most reasonably priced** apartment satisfying your **needs:**

  square-ft., # of bedroom, distance to campus …

Living area (ft^2)	# bedroom	Rent ($)
230	1	600
506	2	1000
433	2	1100
109	1	500
…		
150	1	?
270	1.5	?

Linear Regression Model

- Assume y is a linear function of x (features) plus noise ϵ

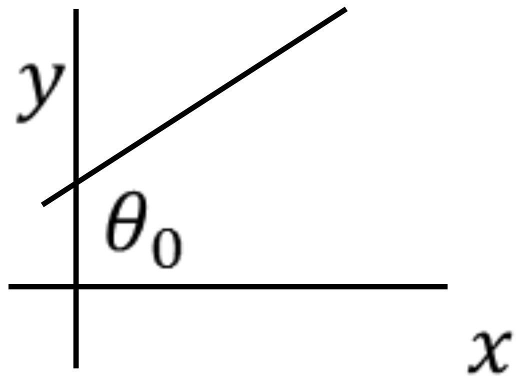
$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n + \epsilon$$

- where ϵ is an error term of unmodeled effects or random noise

- Let $\theta = (\theta_0, \theta_1, \dots, \theta_n)^\top$, and augment data by one dimension

$$x \leftarrow (1, x)^\top$$

- Then $y = \theta^\top x + \epsilon$



Least mean square method

- Given m data points, find θ that minimizes the mean square error

$$\hat{\theta} = argmin_{\theta}\, L(\theta) = \frac{1}{m}\sum_{i}^{m}\left(y^i - \theta^\top x^i\right)^2$$

- Our usual trick: set gradient to 0 and find parameter

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i}^{m}\left(y^i - \theta^\top x^i\right)x^i = 0$$

$$\Leftrightarrow -\frac{2}{m}\sum_{i}^{m} y^i x^i + \frac{2}{m}\sum_{i}^{m} x^i {x^i}^\top \theta = 0$$

Matrix version of the gradient

- $\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_i^m y^i x^i + \frac{2}{m}\sum_i^m x^i {x^i}^\top \theta = 0$

- Equivalent to

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}(x^1 \dots, x^m)(y^1 \dots, y^m)^\top + \frac{2}{m}(x^1, \dots x^m)(x^1, \dots x^m)^\top \theta = 0$$

- Define $X = (x^1, x^2, \dots x^m), y = (y^1, y^2, \dots, y^m)^\top$, gradient becomes

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m} Xy + \frac{2}{m} XX^\top \theta$$

$$\Rightarrow \hat{\theta} = (XX^\top)^{-1} Xy$$

Ridge regression

- Given m data points, find θ that minimizes the regularized mean square error

$$\theta^r = argmin_\theta\, L(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top x^i\right)^2 + \lambda||\theta||^2$$

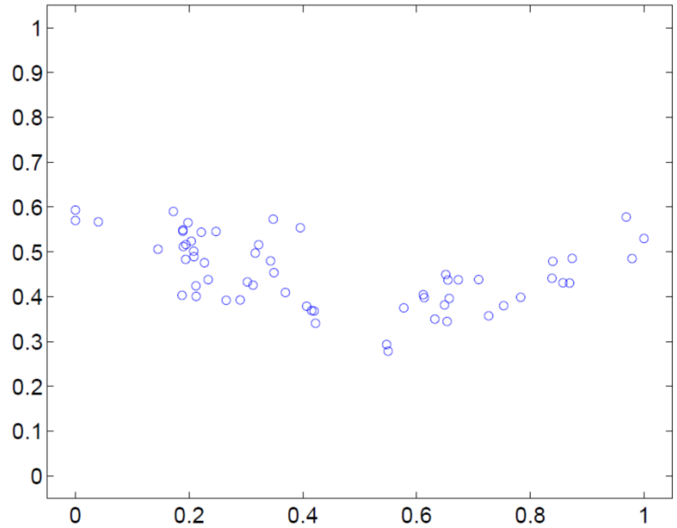
- gradient becomes

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}Xy + \frac{2}{m}XX^\top\theta + \frac{2\lambda}{m}\theta = 0$$

$$\Rightarrow \theta^r = (XX^\top + \lambda I)^{-1}Xy$$

If we choose a different λ, the solution will be different.

Nonlinear regression



- Want to fit a polynomial regression model

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_n x^n + \epsilon$$

- Let $\tilde{x} = (1, x, x^2, \ldots, x^n)^\top$ and $\theta = (\theta_0, \theta_1, \theta_2, \ldots, \theta_n)^\top$

$$\mathrm{y} = \theta^\top \tilde{x}$$

Least mean square method

- Given m data points, find θ that minimizes the mean square error

$$\hat{\theta} = argmin_{\theta}\, L(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top \tilde{x}^i\right)^2$$

- Our usual trick: set gradient to 0 and find parameter

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i}^{m}\left(y^i - \theta^\top \tilde{x}^i\right)\tilde{x}^i = 0$$

$$\Leftrightarrow -\frac{2}{m}\sum_{i}^{m} y^i \tilde{x}^i + \frac{2}{m}\sum_{i}^{m} \tilde{x}^i {\tilde{x}^i}^\top \theta = 0$$

Matrix version of the gradient

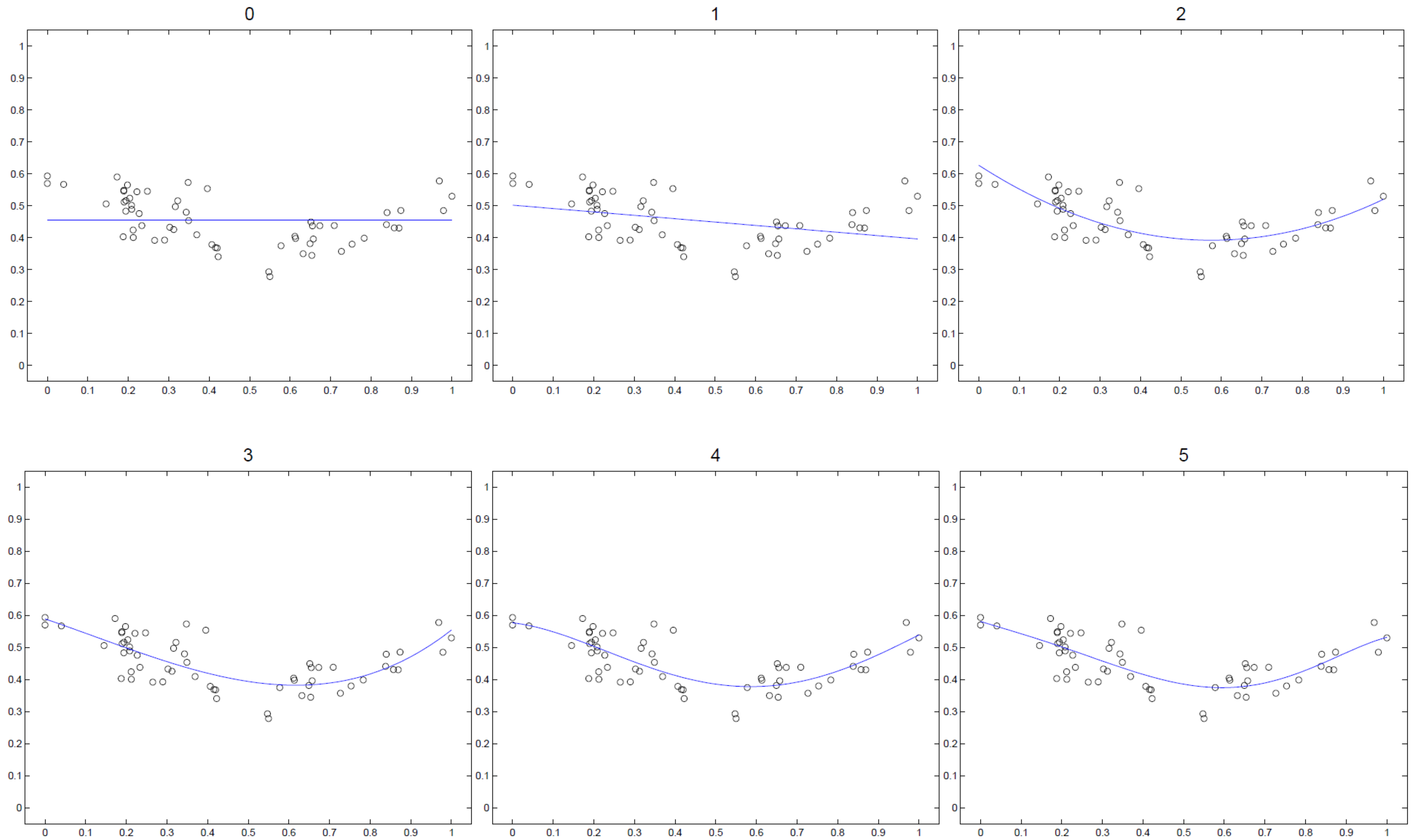
- Define $\tilde{X} = \left(\tilde{x}^{(1)}, \tilde{x}^{(2)}, \dots \tilde{x}^{(m)}\right), y = \left(y^{(1)}, y^{(2)}, \dots, y^{(m)}\right)^{\top}$, gradient becomes

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\tilde{X}y + \frac{2}{m}\tilde{X}\tilde{X}^{\top}\theta = 0$$
$$\Rightarrow \hat{\theta} = \left(\tilde{X}\tilde{X}^{\top}\right)^{-1}\tilde{X}y$$

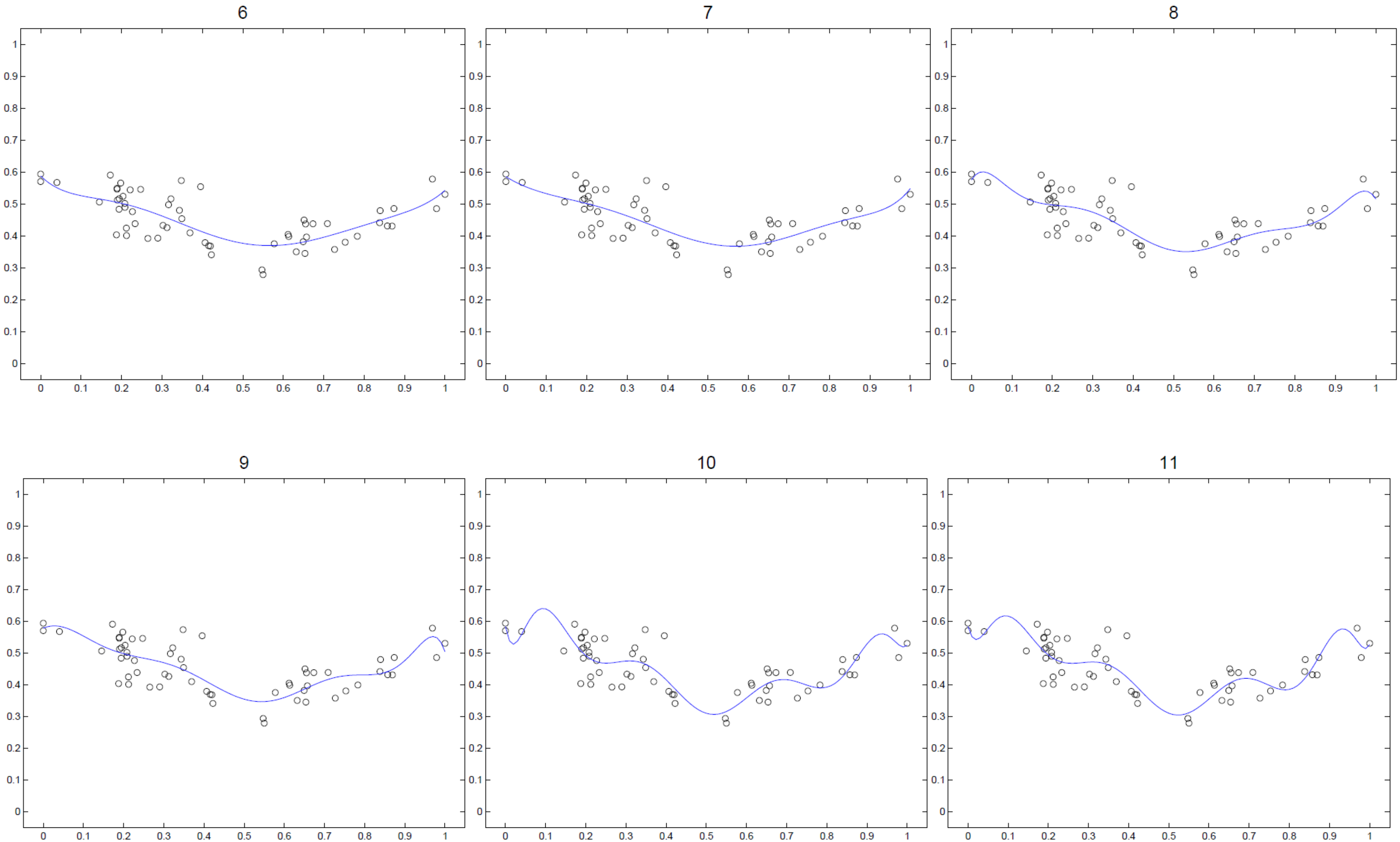
- Note that $\tilde{x} = (1, x, x^2, \dots, x^n)^{\top}$

- If we choose a different maximal degree n for the polynomial, the solution will be different.

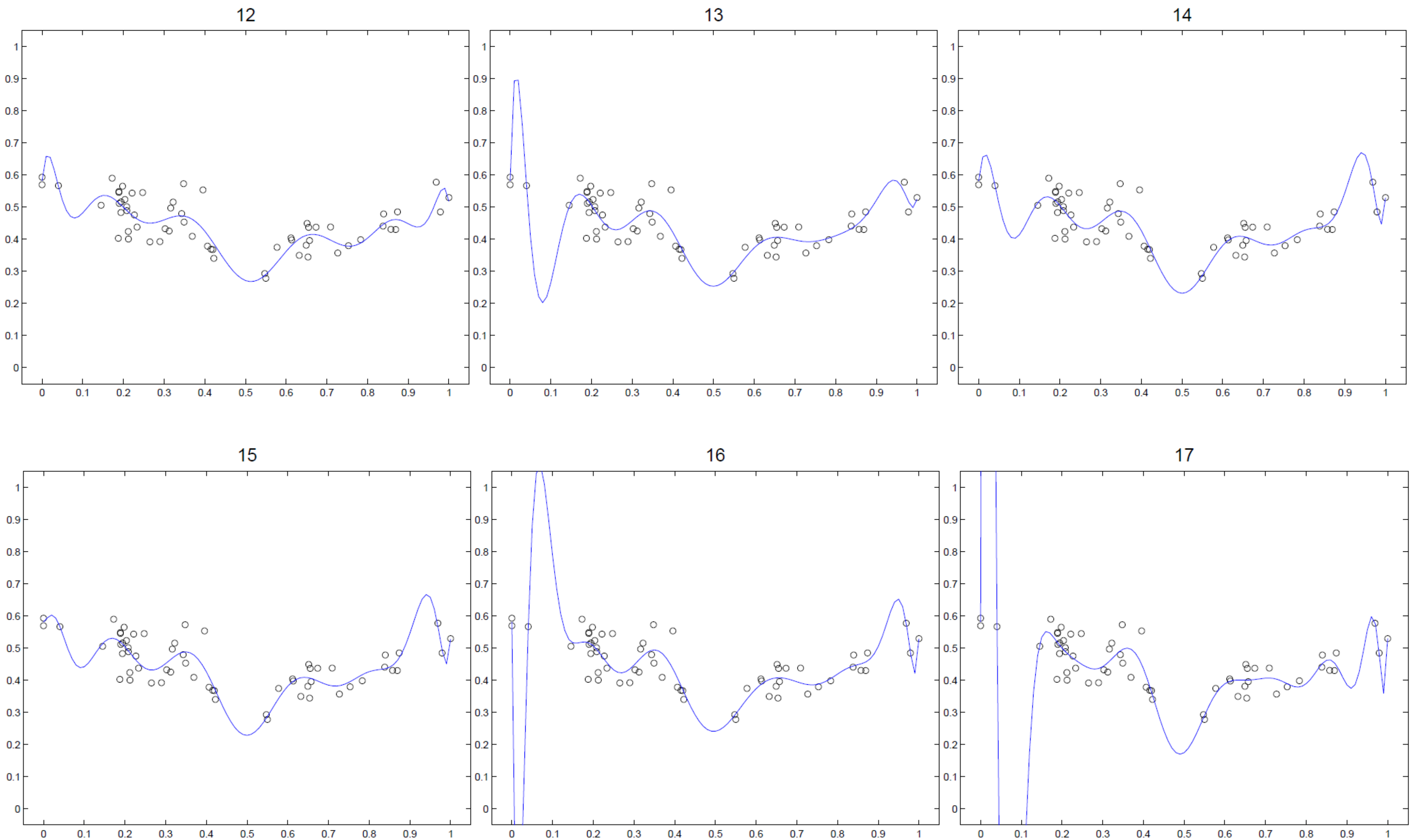
Increasing the maximal degree



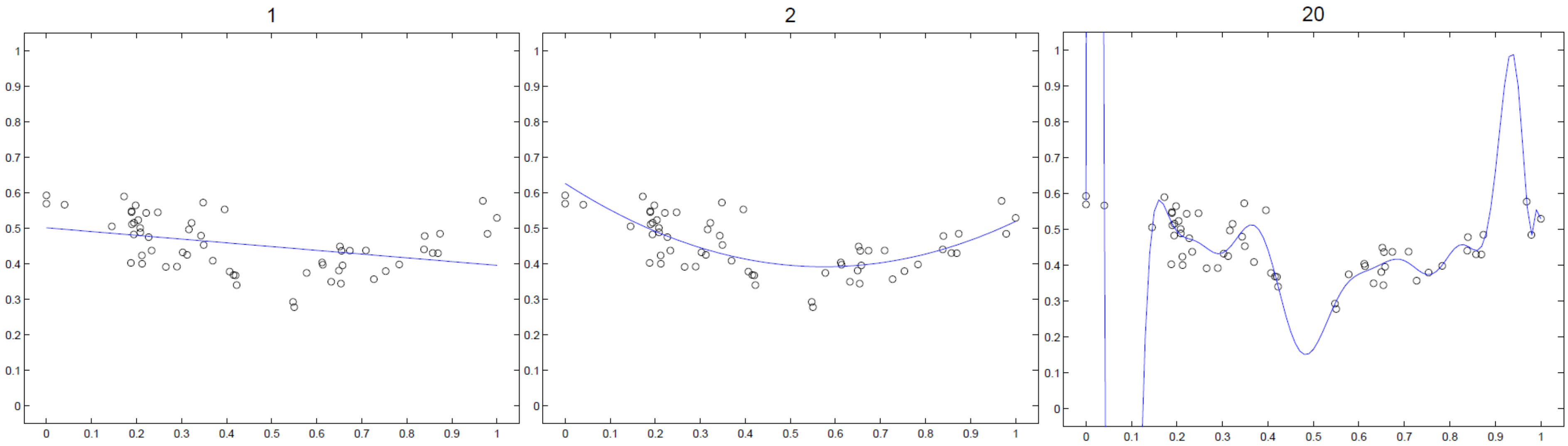
Increasing the maximal degree



Increasing the maximal degree



Which one is better?



- Can we increase the maximal polynomial degree to very large, such that the curve passes through all training points?
- The optimization does not prevent us from doing that

When maximal degree is very large

- Define $\tilde{X} = \left(\tilde{x}^{(1)}, \tilde{x}^{(2)}, \dots \tilde{x}^{(m)}\right), y = \left(y^{(1)}, y^{(2)}, \dots, y^{(m)}\right)^{\top}$, set gradient to zero, $\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\tilde{X}y + \frac{2}{m}\tilde{X}\tilde{X}^{\top}\theta = 0$

$$\Rightarrow \tilde{X}\tilde{X}^{\top}\theta = \tilde{X}y$$

- Each $\tilde{x} = (1, x, x^2, \dots, x^n)^{\top}$ is a vector of polynomial features, the size of $\tilde{X}$ is $n \times m$, and $\tilde{X}\tilde{X}^{\top}$ is $n \times n$

- When $n > m$,

$\tilde{X}\tilde{X}^{\top}$ is not invertible; there are multiple solutions θ which give zero objective

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^{\top}\tilde{x}^i\right)^2$$

Geometric Interpretation of LMS

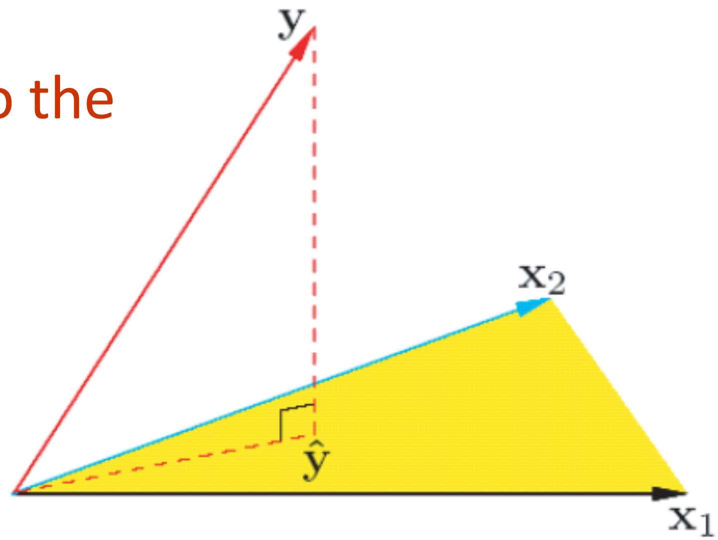
- The predictions on the training data are:
$$\hat{y} = X^\top \theta = X^\top (XX^\top)^{-1} X y$$

- Look at residule $\hat{y} - y$

$$\hat{y} - y = (X^\top (XX^\top)^{-1} X^\top - I) y$$

$$X(\hat{y} - y) = X(X^\top (XX^\top)^{-1} X^\top - I) y = 0$$

- $\hat{y}$ is the orthogonal projection of y into the space spanned by the columns of X

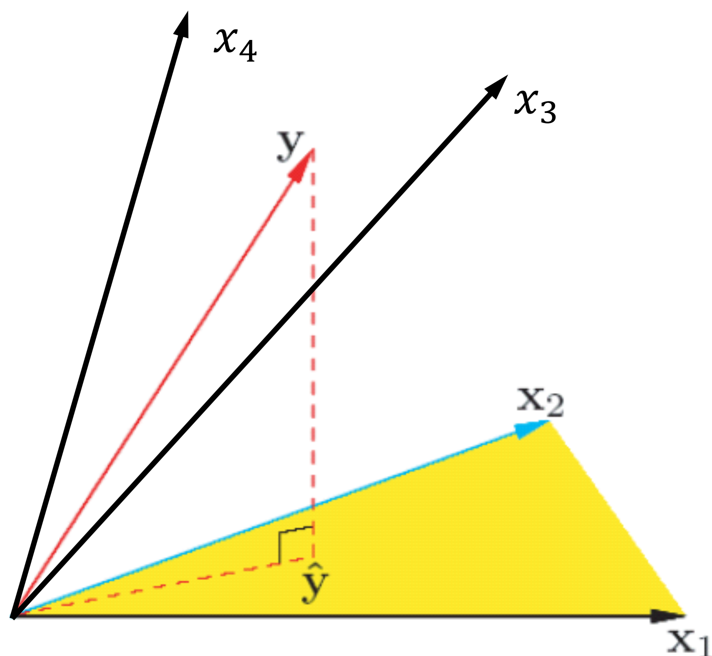


Geometric interpretation

- $\tilde{X} = \left(\tilde{x}^{(1)}, \tilde{x}^{(2)}, \dots \tilde{x}^{(m)}\right), y = \left(y^{(1)}, y^{(2)}, \dots, y^{(m)}\right)^\top$, Each $\tilde{x} = (1, x, x^2, \dots, x^n)^\top$

- Suppose $m = 3, n = 3$

- View the rows of $\tilde{X}$ as vectors
  - $x_1 = (1,1,1)^\top$
  - $x_2 = \left(x^{(1)}, x^{(2)}, x^{(3)}\right)^\top$
  - $x_3 = \left(x^{(1)2}, x^{(2)2}, x^{(3)2}\right)^\top$
  - $x_4 = \left(x^{(1)3}, x^{(2)3}, x^{(3)3}\right)^\top$

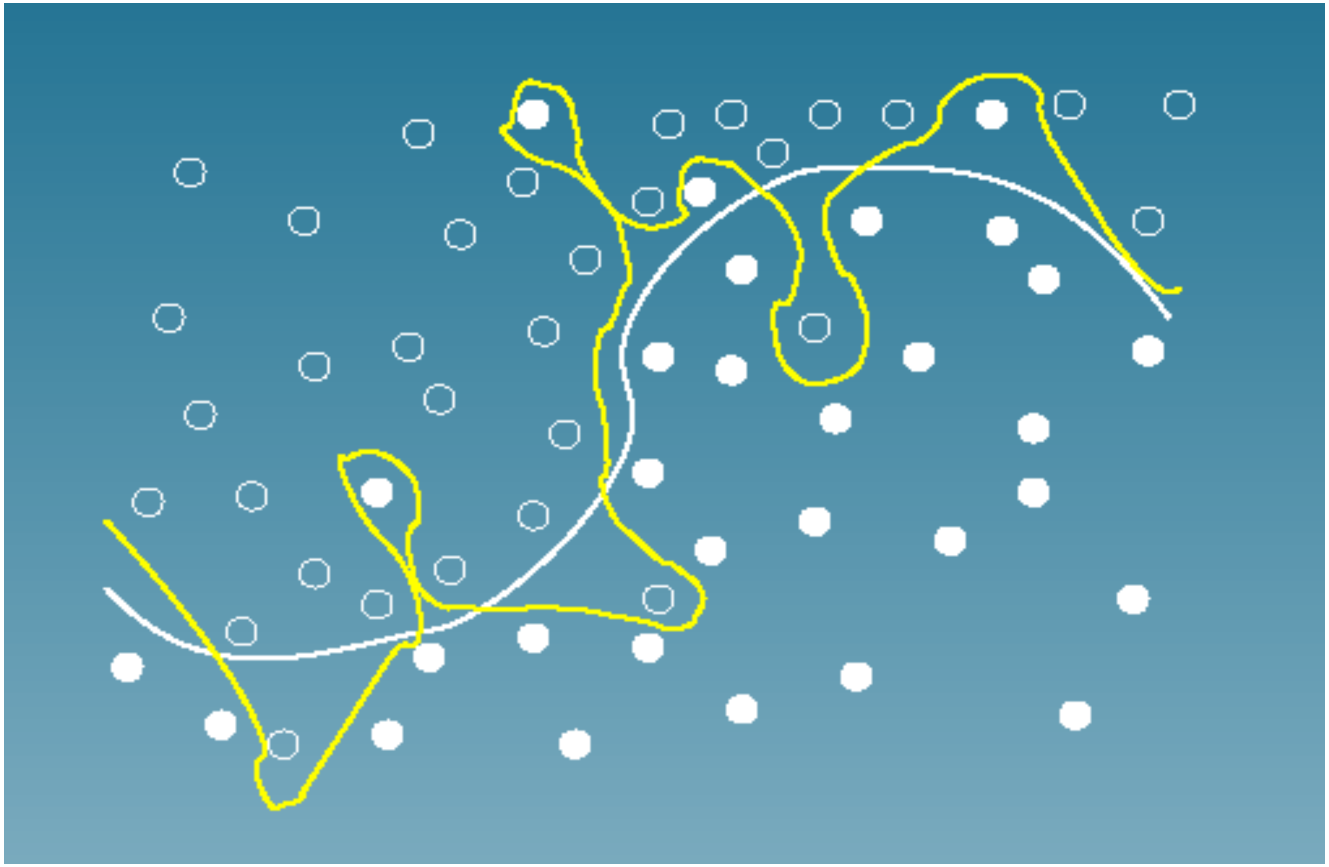


- Multiple θ with $\tilde{X}\tilde{X}^\top\theta = \tilde{X}y$

Classification with polynomials

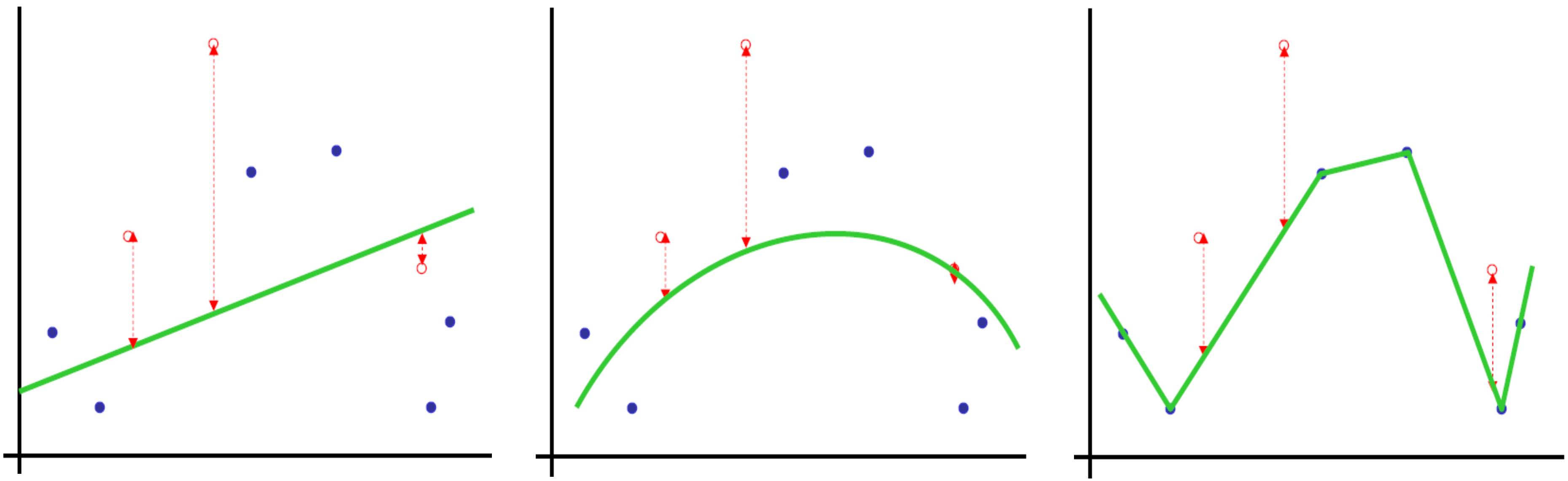
- Eg. Logistic regression with polynomial features

$$p(y=1|x,\theta)=\frac{1}{1+\exp(-\theta^\top \tilde{x})}$$



Overfitting/Underfitting

- Blue points: training data points, Red points: test data points
- The fit in the middle panel achieves a balance of small error in both training and test points.



Underfitting

Overfitting

What is the problem?

- Given m data points $D = \{(\tilde{x}^i, y^i)\}$, find θ that minimizes the mean square error

$$\hat{\theta} = argmin_\theta \hat{L}(\theta) := \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top \tilde{x}^i\right)^2$$

- But we really want to minimize the error for unseen data points, or with respect to the entire distribution of data

$$\theta^* = argmin_\theta L(\theta) \coloneqq \mathbb{E}_{(\tilde{x},y)\sim P(\tilde{x},y)}[(y - \theta^\top \tilde{x})^2]$$

- It is the finite number training point that creates the problem

Decomposition of expected loss

- Estimate your function from a finite data set D

$$\hat{f} = argmin_f \hat{L}(f) := \frac{1}{m}\sum_{i=1}^{m}\left(y^i - f\left(x^i\right)\right)^2$$

$\hat{f}$ is a random function, generally different for different data set

- Expected loss of $\hat{f}$

$$L(\hat{f}) := \mathbb{E}_D \mathbb{E}_{(x,y)} \left[\left(y - \hat{f}(x)\right)^2\right]$$

- Bias-variance decomposition

Expected loss = (bias)^2 + variance + noise

What is the best we can do?

- The expected squared loss is

$$L(\hat{f}) := \mathbb{E}_D \mathbb{E}_{(x,y)} \left[\left(y - \hat{f}(x) \right)^2 \right]$$

$$= \mathbb{E}_D \left[\underbrace{\int \int \left(y - \hat{f}(x) \right)^2 p(x,y) dx dy}_{A} \right]$$

- Our goal is to choose $\hat{f}(x)$ that minimize $L(\hat{f})$. Calculus of variations

$$\frac{\partial A}{\partial f(x)} = 2 \int \left(y - f(x) \right) p(x,y) dy = 0$$

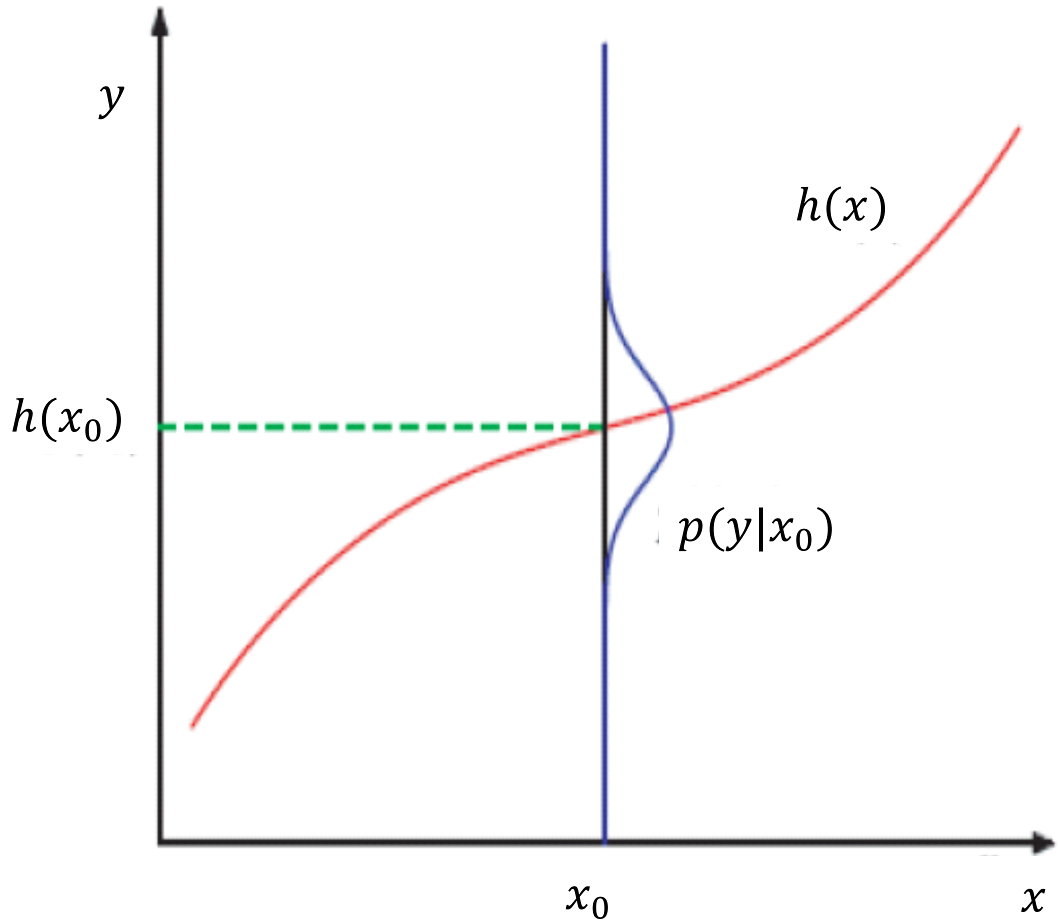
$$\Leftrightarrow \int f(x) p(x,y) dy = \int y p(x,y) dy$$

f(x)

$$\Leftrightarrow h(x) := \int \frac{y p(x,y)}{p(x)} dy = \int y p(y|x) dt = \mathbb{E}_{y|x}[y] = \mathbb{E}[y|x]$$

The best predictor is the expected value

- The best you can do is $h(x) = \mathbb{E}_{y|x}[y]$: the expected value of y given a particular x



Noise term in the decomposition

- $h(x) = \mathbb{E}(y|x)$ is the **optimal** predictor, and $\hat{f}(x)$ our actual predictor, decompose the error a bit

$$
\begin{aligned}
\mathbb{E}_D \mathbb{E}_{(x,y)} \left[\left(y - \hat{f}(x) \right)^2 \right] &= \mathbb{E}_D \left[\int \int \left(y - h(x) + h(x) - \hat{f}(x) \right)^2 p(x,y) dx dy \right] \\
&= \mathbb{E}_D \left[\int \int \left(\left(\hat{f}(x) - h(x) \right)^2 + 2 \left(\hat{f}(x) - h(x) \right) (h(x) - y) \right. \right. \\
&\qquad \left. \left. + (h(x) - y)^2 \right) p(x,y) dx dy \right] \\
&= \underbrace{\mathbb{E}_D \left[\int \left(\hat{f}(x) - h(x) \right)^2 p(x) dx \right]}_{\text{Will decompose further}} + \underbrace{\int \int (h(x) - y)^2 p(x,y) dx dy}_{\text{Noise term. can not do better than this. a lower bound of the expected loss}}
\end{aligned}
$$

Bias-variance decomposition

- $\hat{f}(x)$ is a random function, generally different for different dataset D
- $\mathbb{E}_D\left[\hat{f}(x)\right]$: expected value of $\hat{f}(x)$ with respected to random dataset

$$\mathbb{E}_D\left[\int\left(\hat{f}(x)-h(x)\right)^2 p(x)dx\right]=\mathbb{E}_D\mathbb{E}_x\left[\left(\hat{f}(x)-h(x)\right)^2\right]$$

$$=\mathbb{E}_x\mathbb{E}_D\left[\left(\hat{f}(x)-\mathbb{E}_D[\hat{f}(x)]+\mathbb{E}_D[\hat{f}(x)]-h(x)\right)^2\right]$$

$$=\mathbb{E}_x\mathbb{E}_D\left[\left(\hat{f}(x)-\mathbb{E}_D[\hat{f}(x)]\right)^2\right]+\mathbb{E}_x\mathbb{E}_D\left[\left(\mathbb{E}_D[\hat{f}(x)]-h(x)\right)^2\right]$$
$$-2\mathbb{E}_x\mathbb{E}_D\left[\left(\hat{f}(x)-\mathbb{E}_D[\hat{f}(x)]\right)\left(\mathbb{E}_D[\hat{f}(x)]-h(x)\right)\right]$$

$$=\underbrace{\mathbb{E}_x\mathbb{E}_D\left[\left(\hat{f}(x)-\mathbb{E}_D[\hat{f}(x)]\right)^2\right]}_{\text{Bias}^2}+\underbrace{\mathbb{E}_x\left[\left(\mathbb{E}_D[\hat{f}(x)]-h(x)\right)^2\right]}_{\text{Variance}}$$

Overall decomposition of expected loss

- Putting things together

  Expected loss = $(\text{bias})^2$ + variance + noise

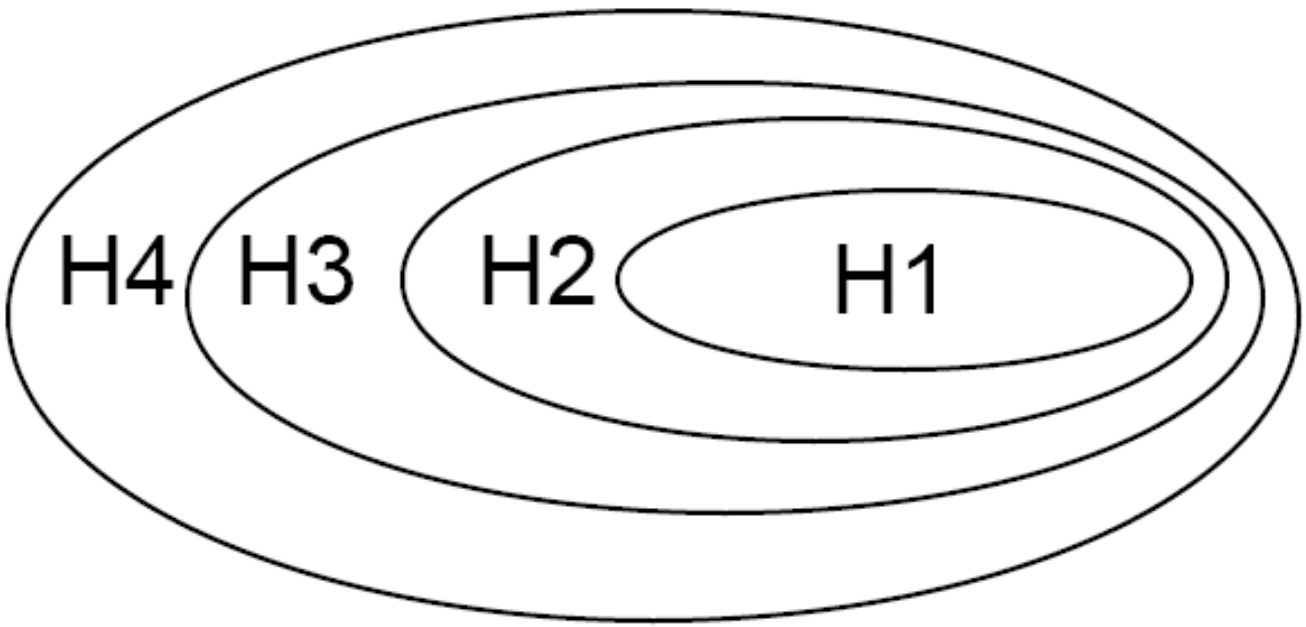
- In formula

$$\mathbb{E}_D \mathbb{E}_{(x,y)} \left[\left(y - \hat{f}(x) \right)^2 \right]$$
$$= \mathbb{E}_x \left[\left(\mathbb{E}_D[\hat{f}(x)] - h(x) \right)^2 \right] \; (bias^2)$$
$$+ \mathbb{E}_x \mathbb{E}_D \left[\left(\hat{f}(x) - \mathbb{E}_D[\hat{f}(x)] \right)^2 \right] \; (variance)$$
$$+ \mathbb{E}_{(x,y)}[(h(x) - y)^2] \; (noise)$$

- Key quantities
  - $\hat{f}(x)$: actual predictor
  - $\mathbb{E}_D[\hat{f}(x)]$: expected predictor
  - $h(x) = \mathbb{E}(y|x)$: **optimal** predictor

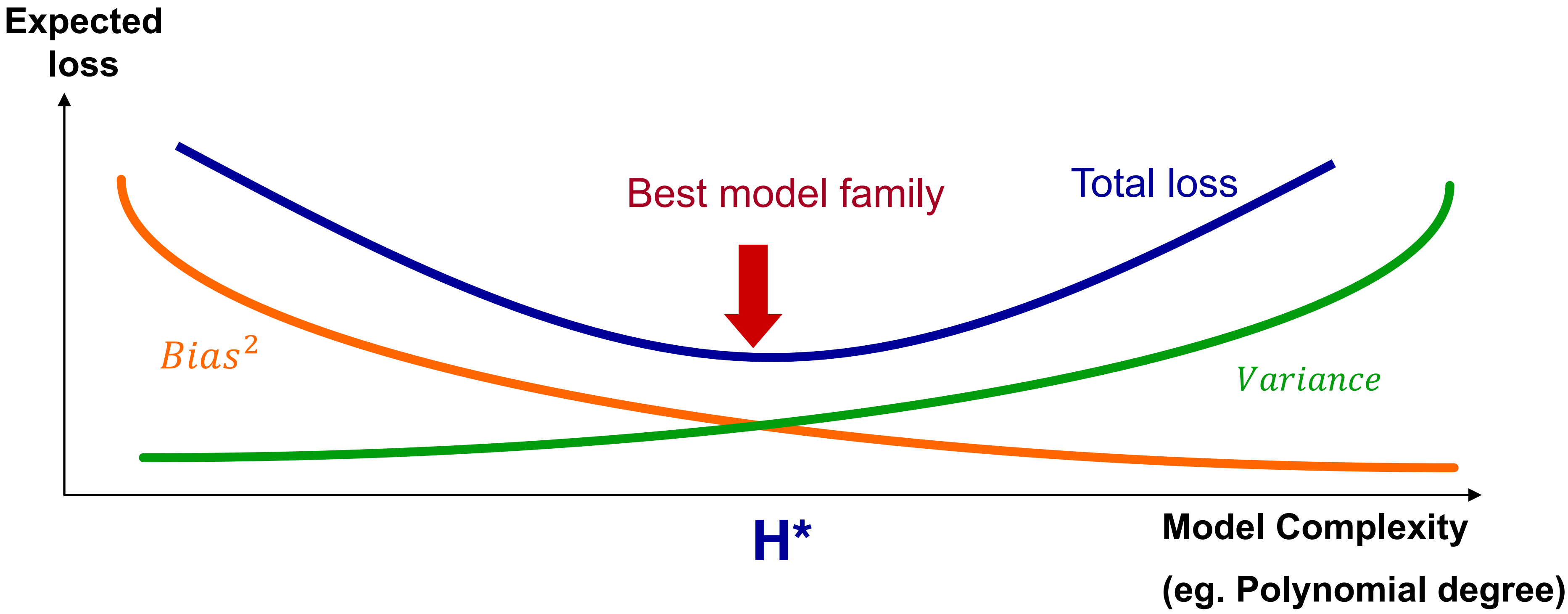
Model space

- Which model space should we choose?
- The more complex the model, the large the model space
- Eg. Polynomial function of degree 1, 2, … corresponds to space H1, H2 …



Intuition of model selection

Find the right model family s.t. expected loss becomes minimum



Other things that control model complexity

- Eg. In the case of linear models $y = \langle w, x \rangle + b$, one wants to make $\|w\|$ a controlled parameter

$$\|w\| < C$$

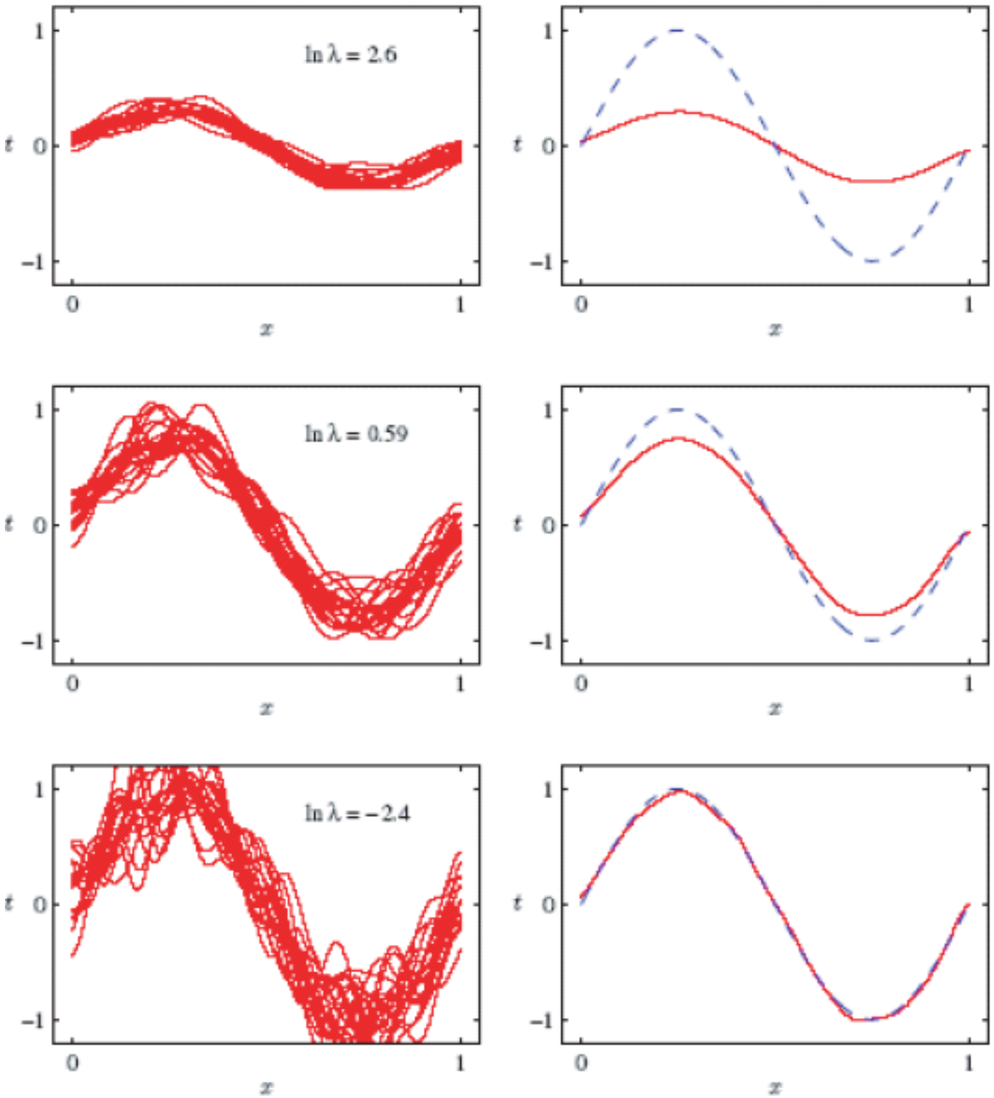
  - H_C the linear model function family satisfying the constraint
  - The large the C, the large the model family

- Eg. the larger the regularization parameter λ, the small the model family

  - $J(w) = \sum_i (w^\top x_i - y_i)^2 + \lambda \|w\|_2^2$
  - $J(w) = \sum_i (w^\top x_i - y_i)^2 + \lambda \|w\|_1$

- Eg. Early stopping in boosting

Experiment with bias-variance tradeoff



- λ is a "regularization" terms in LR, the smaller the λ, is more complex the model
  - Simple (highly regularized) models have low variance but high bias.
  - Complex models have low bias but high variance.
- The actual $\mathbb{E}_D$ can not be computed
- You are inspecting an empirical average over 100 training set.

How to do model selection in practice?

Georgia Tech

- Suppose we are trying select among several different models for a learning problem.
- Examples:

  1. polynomial regression

  $$h(x;\theta) = g(\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_k x^k)$$

  - Model selection: we wish to **automatically** and **objectively** decide if k should be, say, 0, 1, . . . , or 10.

  2. locally weighted regression,

  - Model selection: we want to automatically choose the bandwidth parameter τ.

  3. Mixture models and hidden Markov model,

  - Model selection: we want to decide the number of hidden states

- The Problem:
  - Given model family $F = \{M_1, M_2, \ldots, M_I\}$, find $M_i \in F$ s.t.

  $$M_i = \arg\max_{M \in F} J(D, M)$$

Cross-Validation

- K-fold cross-validation (CV)

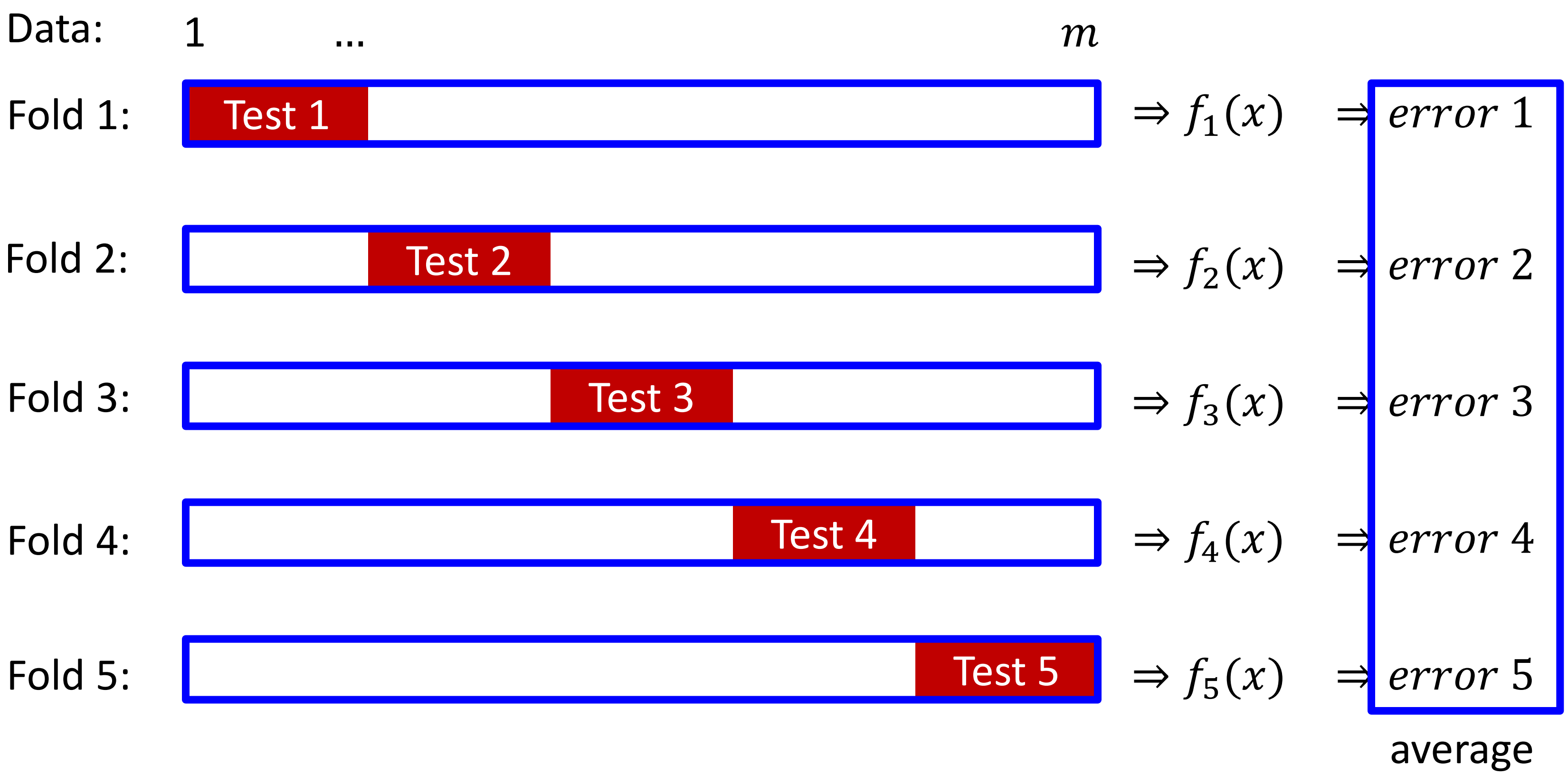
- For each fold i:
  - Set aside $\alpha \cdot m$ samples of D (where $m = |D|$) as the held-out data. They will be used to evaluate the error
  - Fit a model $f_i(x)$ to the remaining $(1-\alpha) \cdot m$ samples in D
  - Calculate the error of the model $f_i(x)$ on the held-out data.

- Repeat the above ***K* times**, choosing a **different** held-out data set each time, and the errors are averaged over the folds.

- For the polynomial degree with the lowest score, we use all of D to find the parameter values for $f(x)$.

Cross-validation

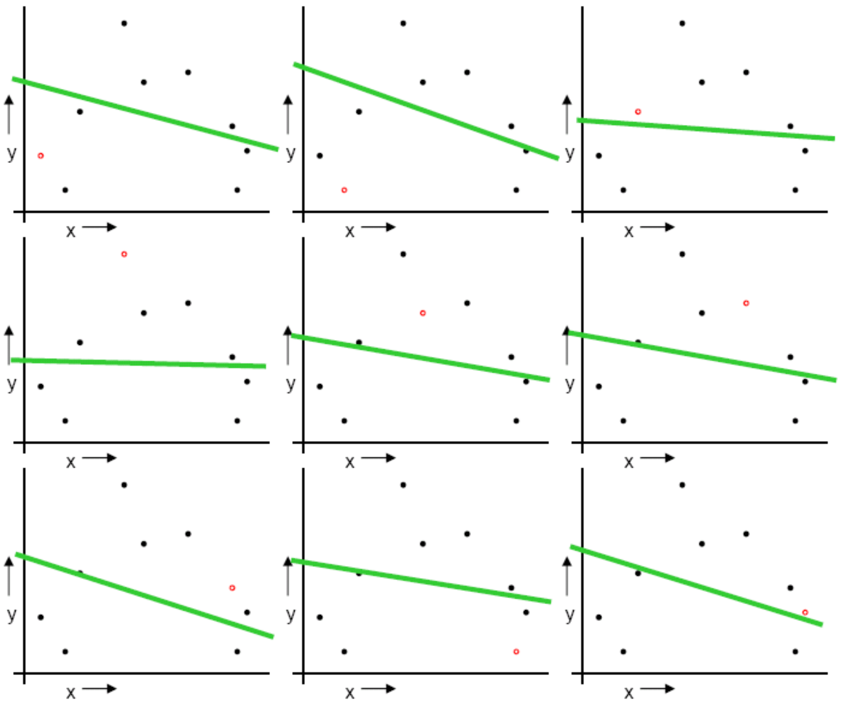
- Eg. Want to select the maximal degree of polynomial
- 5-fold cross-validation (blank: training; red: test)



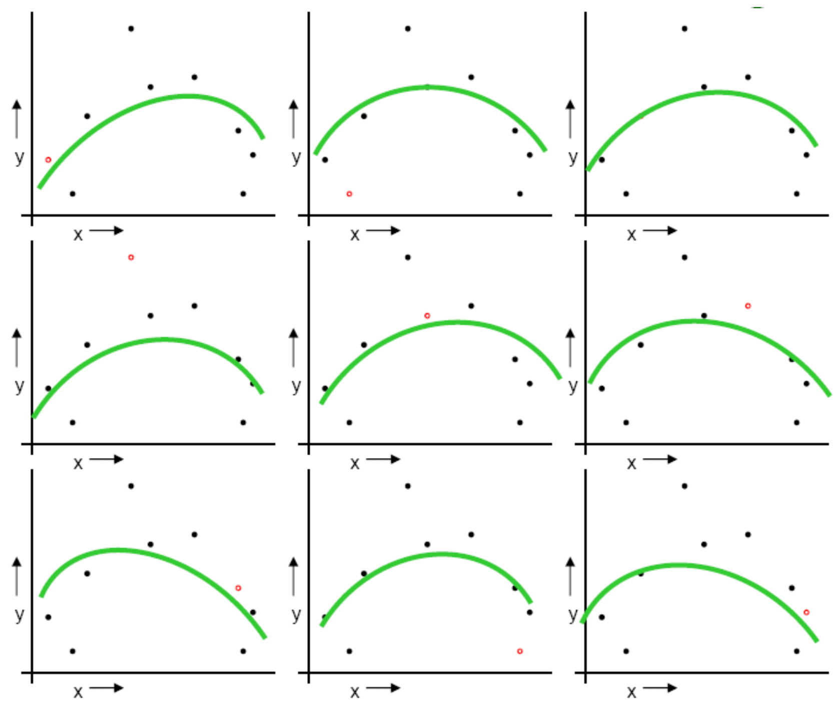
- Important: test data i is not used to fit model $f_i(x)$

Example:

- When $\alpha = 1/N$, the algorithm is known as Leave-One-Out-Cross-Validation (LOOCV)



MSELOOCV(M_1)=2.12



MSELOOCV(M_2)=0.962

Practical issues for K-fold CV

- How to decide the values for K (or α)
  - Commonly used $K = 10$ or ($\alpha = 0.1$).
  - Large K makes it time-consuming.
  - Bias-variance trade-off
    - Large K usually leads to low bias. In principle, *LOOCV* provides an almost unbiased estimate of the generalization ability of a classifier, but it can also have high variance.
    - Small K can reduce variance, but will lead to under-use of data, and causing high-bias.

- One important point is that the test data D_{test} is never used in CV, because doing so would result in overly (indeed dishonest) optimistic accuracy rates during the testing phase.