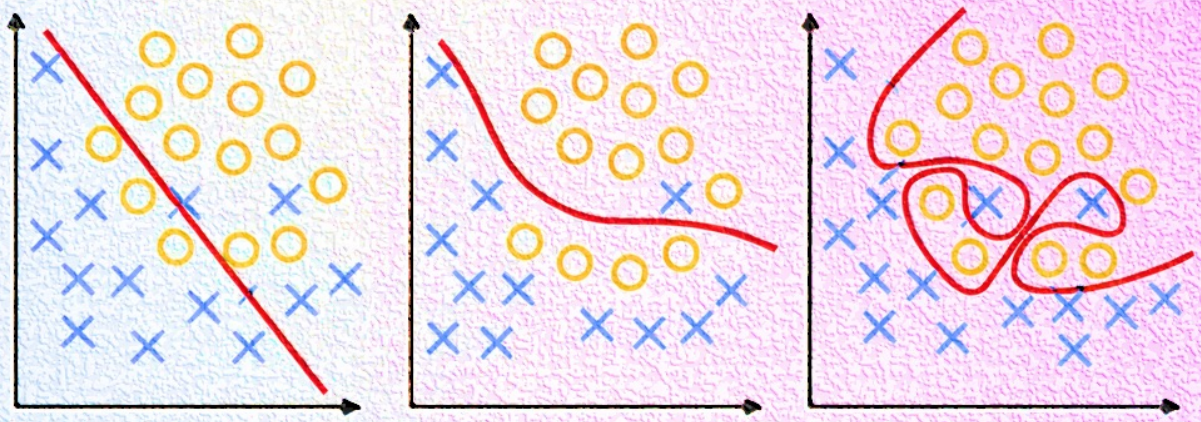
Georgia Tech

Bias–Variance Tradeoff & Cross Validation



Mohsen Moghaddam, Ph.D.

Gary C. Butler Family Associate Professor
H. Milton Stewart School of Industrial and Systems Engineering
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

SAIL
Symbiotic & Augmented Intelligence Laboratory

Spring 2026

Learning Outcomes

- Analyze how model complexity influences bias, variance, numerical stability, and generalization performance, with emphasis on least-squares regression
- Diagnose underfitting and overfitting using polynomial regression, connecting the visual behavior of fitted curves to bias, variance, and instability in the design matrix
- Interpret the bias–variance–noise decomposition of test error and explain the role of the optimal predictor as the best achievable benchmark
- Explain why training error underestimates generalization error and why test data must be reserved for final evaluation
- Apply K-fold cross-validation correctly to estimate test error and to select model complexity and regularization parameters without data leakage
- Explain how cross-validation error curves reflect the bias–variance tradeoff and enable practical model selection

Problem Setup

Fitting Polynomial Function

Polynomial regression: Given noisy data in one dimension, **how flexible** should our model be? (The polynomial degree n is chosen manually before training)

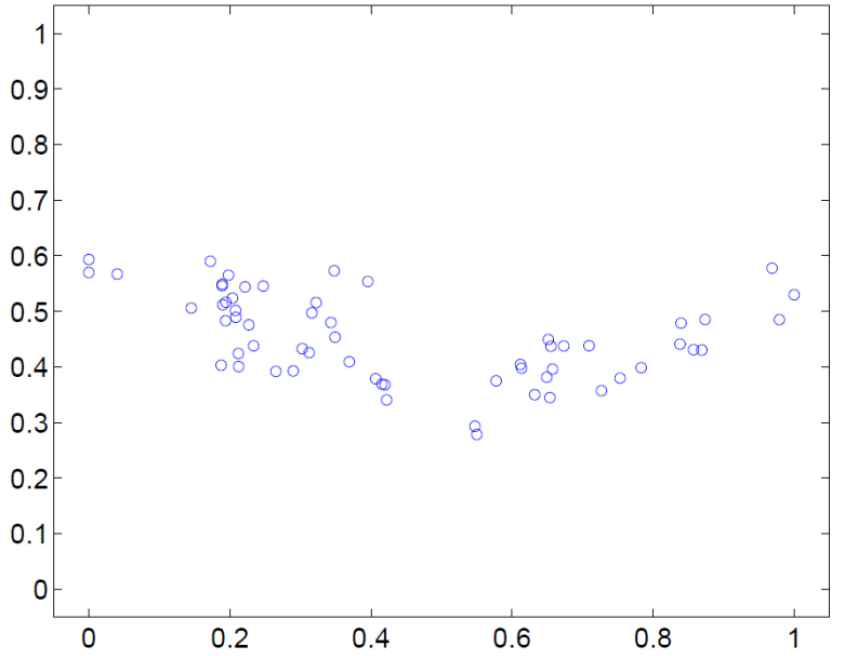
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_n x^n + \epsilon$$

Noise

Let $\tilde{x} = (1, x, x^2, \ldots, x^n)^\top$

$\theta = (\theta_0, \theta_1, \theta_2, \ldots, \theta_n)^\top$

Therefore, $y = \theta^\top \tilde{x} + \epsilon$



If we just crank the degree up to minimize training error, what might happen?

Solving Using Least-Square

Given m data points, find θ that minimizes the MSE:

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top \tilde{x}^i\right)^2 \qquad \tilde{x}^i = \begin{bmatrix} 1 \\ x^i \\ (x^i)^2 \\ \vdots \\ (x^i)^n \end{bmatrix}$$

Set gradient to 0 and find parameter:

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}\sum_{i=1}^{m}\left(y^i \tilde{x}^i - \tilde{x}^i \tilde{x}^{i\top}\theta\right) = 0$$

This procedure minimizes training error—not test error

Matrix–Vector Representation

Given a design matrix $X \in \mathbb{R}^{m\times(n+1)}$, whose rows are the feature vectors $(\tilde{x}^i)^\top$, and the response vector $y \in \mathbb{R}^m$, the gradient becomes

Identical to linear regression in feature space

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{m}(\underbrace{X^\top y}_{\sum_i y^i \tilde{x}^i} - \underbrace{X^\top X\theta}_{\sum_i \tilde{x}^i \tilde{x}^{i\top}}) = 0 \qquad \Rightarrow \qquad \hat{\theta} = (X^\top X)^{-1} X^\top y$$

- A different maximal degree n for the polynomial will change the solution
- As the polynomial degree increases, the dimension of θ increases
- Higher-dimensional models are more flexible but more sensitive to data

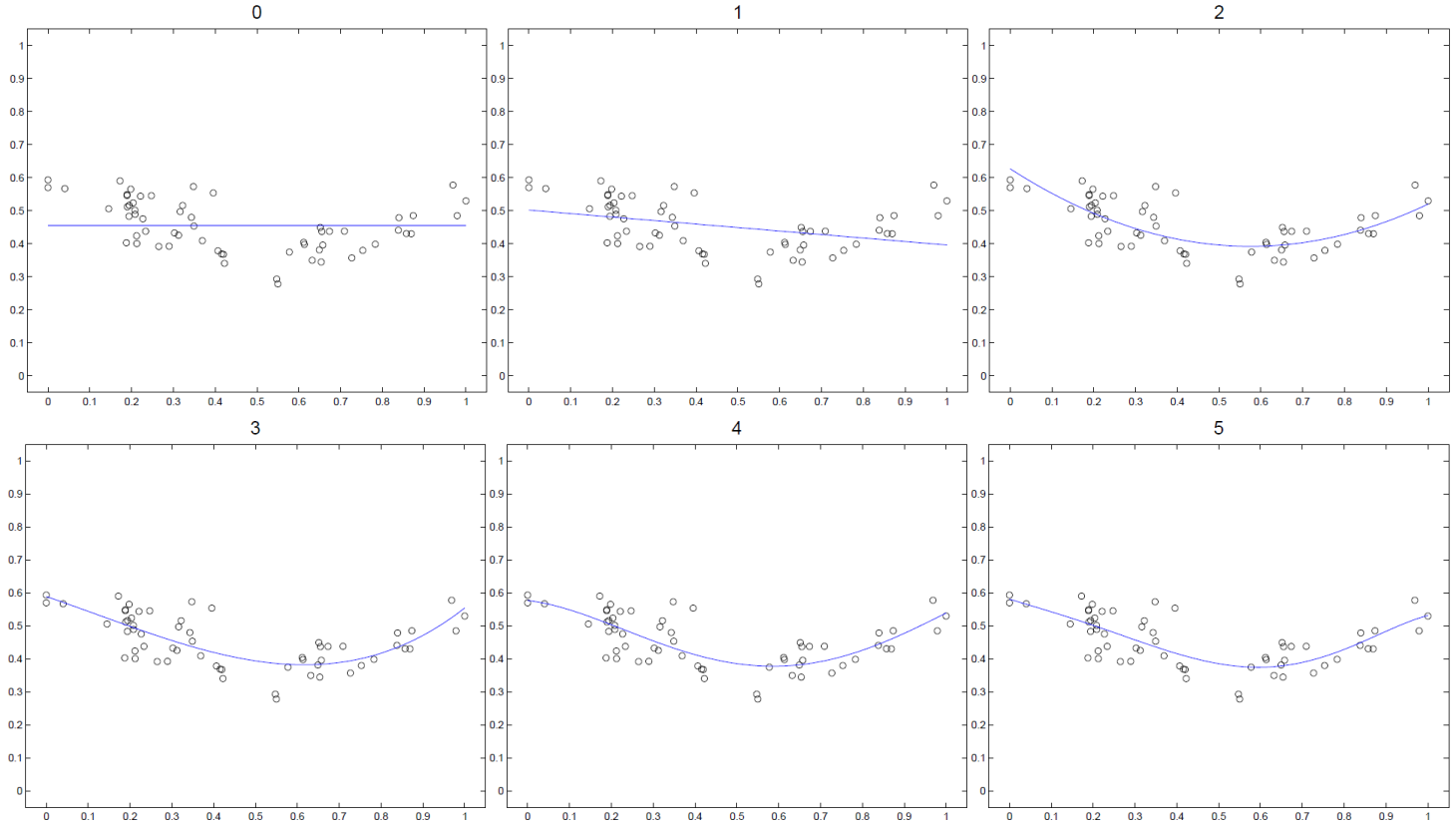
Polynomial degree is fixed before training—optimization only fits parameters

Underfitting, Overfitting, & Numerical Instability

Underfitting → High Bias

Low-degree models are too simple to capture the underlying trend (e.g., curvature in the true function), resulting in high bias

- Bias refers to **systematic error due to model assumptions**
- It measures how far the model's average prediction is from the true function



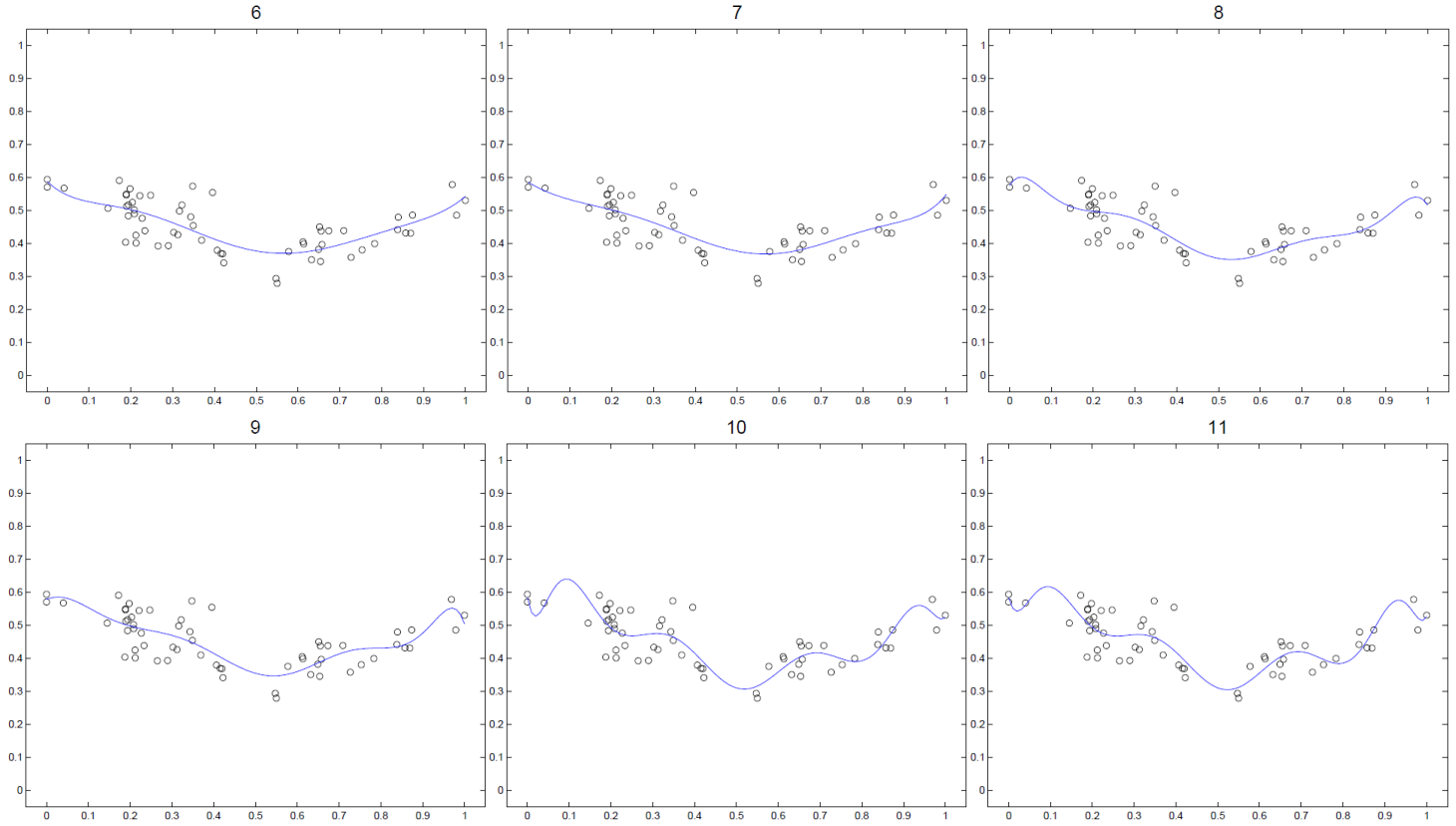
Why doesn't adding more data fix underfitting here?

Overfitting → High Variance

High-degree models fit noise in the training data rather than signal, leading to high variance and poor generalization

- Variance refers to **dependence on the specific training sample**
- It measures how much the model's prediction changes depending on which training dataset you use

Why does the curve look so wiggly near the edges?

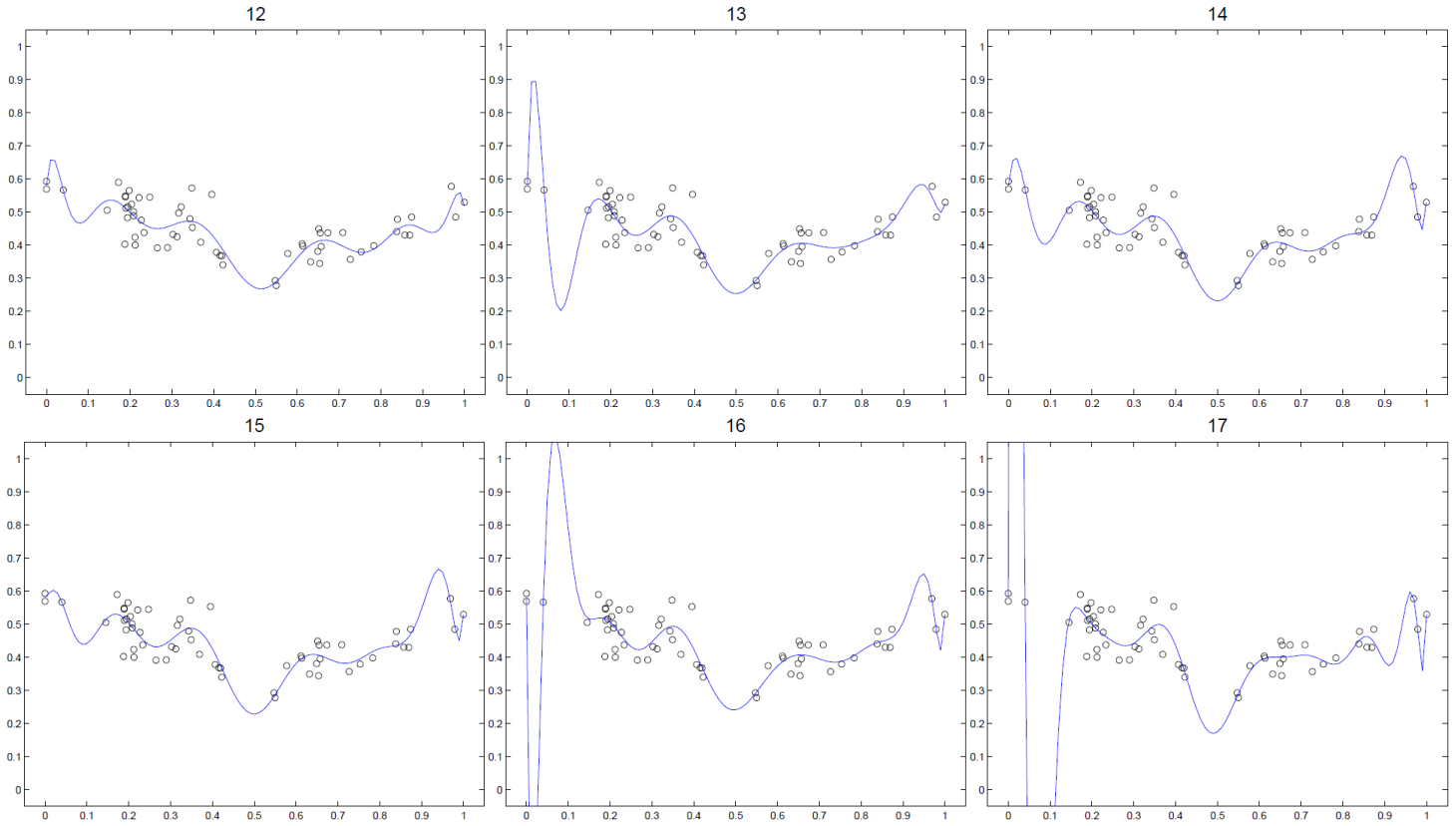


Numerical Instability

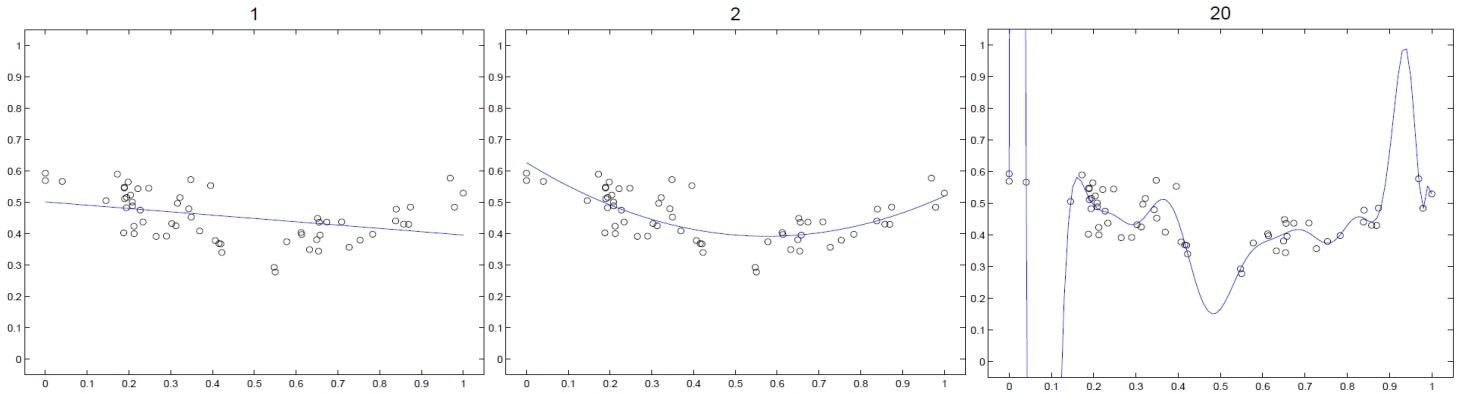
As model complexity increases, the design matrix becomes ill-conditioned, making the least-squares solution numerically unstable

- Numerical instability is a structural consequence of high complexity
- Polynomial features become highly correlated, adding little new information

Models might give different solutions for the same data



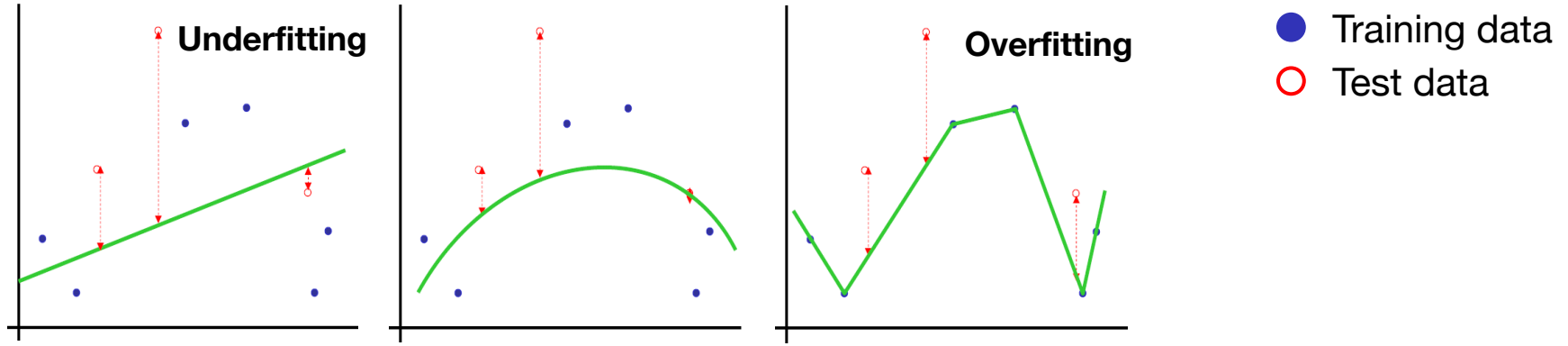
How to Choose the Degree?



- Which model is better?
- If we choose a very large polynomial degree, we can fit all training points—but is that a reliable criterion?
- Perfect training fit often leads to poor generalization on unseen data

Need a data-driven approach to select model complexity without the test data

Balancing Bias and Variance



$$\hat{\theta} = (X^\top X)^{-1} X^\top y, \qquad X = (\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m) \in \mathbb{R}^{m \times (n+1)}$$

$$X^\top X \in \mathbb{R}^{(n+1)\times(n+1)}, \qquad \operatorname{rank}(X^\top X) \leq \min\{m, n+1\}$$

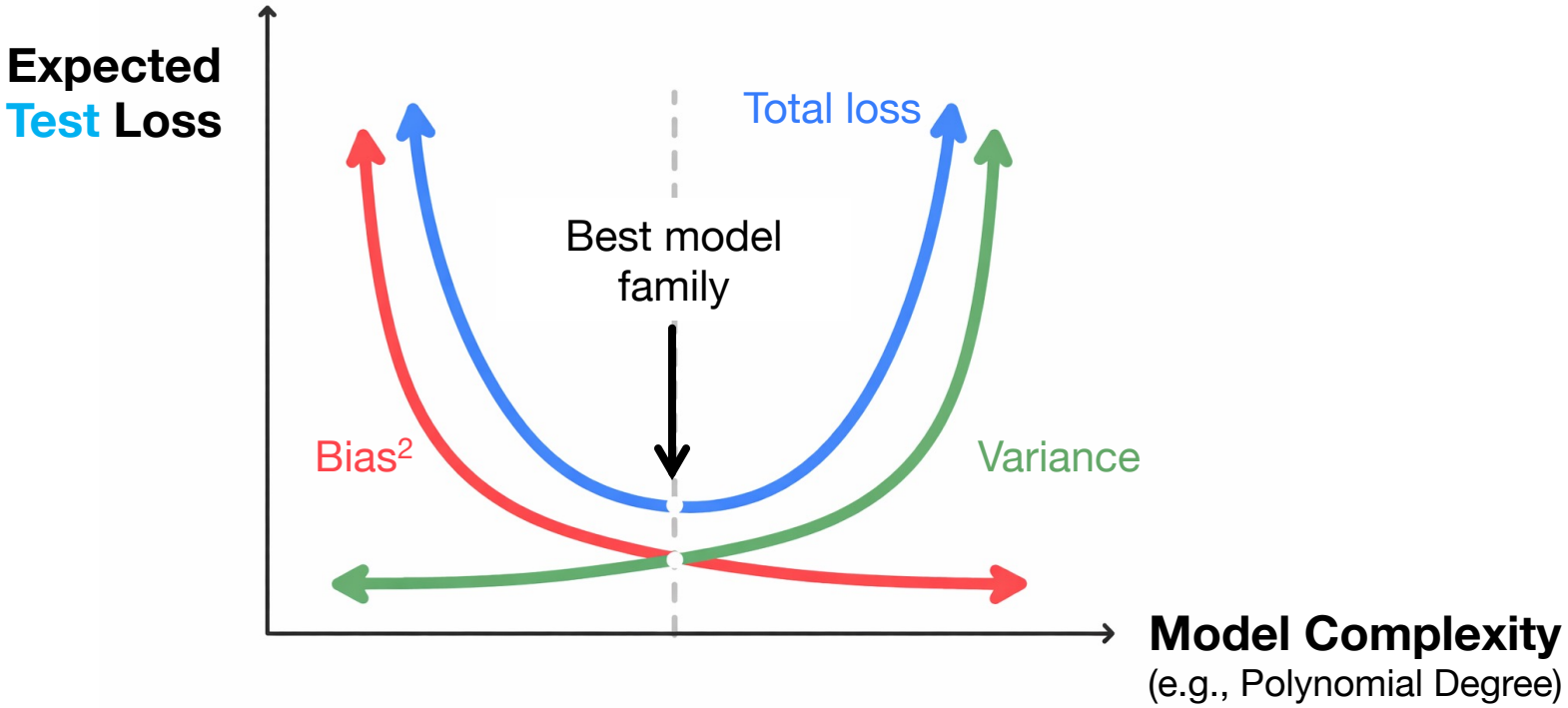
When $n + 1 > m$, $\operatorname{rank}(X^\top X) \leq m < n + 1$; therefore $X^\top X$ is **not invertible**

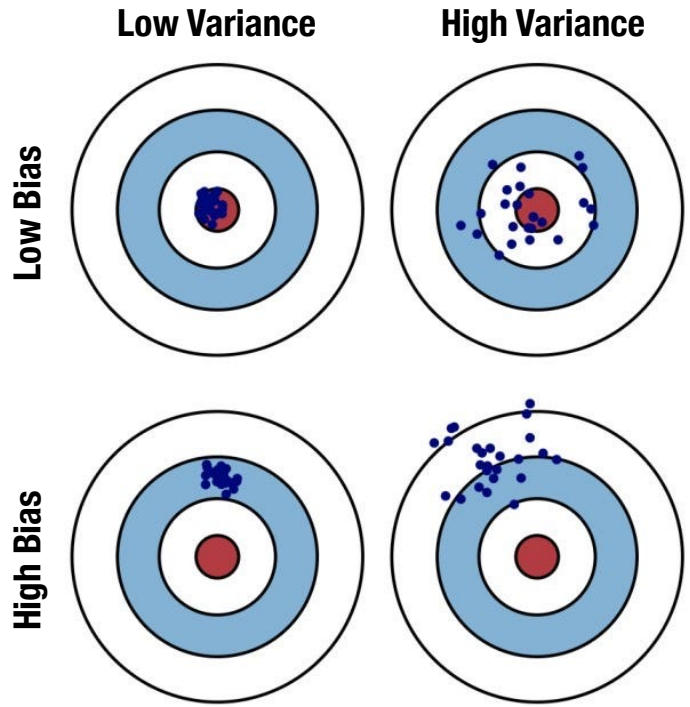
Rank indicates when the model is too complex for the available data

Bias–Variance Tradeoff

Intuition of Bias–Variance Tradeoff

Find the right model family that minimizes expected loss (test error) by balancing bias and variance





Note: This curve is conceptual and cannot be directly observed

Bias–Variance Tradeoff in Linear Regression

Given m **training data points** $\mathcal{D} = (x^i, y^i)_{i=1}^m$, find θ that minimizes the MSE:

$$\hat{\theta} = \arg\min_\theta \hat{L}(\theta) \coloneqq \frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top x^i\right)^2$$ ← Empirical (training) risk minimization

But we really want to minimize the error for **unseen "test" data** points (x, y) with respect to the entire distribution of data:

$$\theta^* = \arg\min_\theta L(\theta) \coloneqq \mathbb{E}_{(x,y)}[(y - \theta^\top x)^2]$$ ← True (population) risk minimization

Empirical risk minimization merely approximates true risk minimization

What is True Risk Minimization?

Find the θ^* that minimizes the average prediction error over all possible data points

$$\theta^* = \arg\min_{\theta} L(\theta) \coloneqq \mathbb{E}_{(x,y)}[(y - \theta^\top x)^2]$$

Example: Predict exam scores from study hours

Model A: $\hat{y} = 10x$

$$\mathbb{E}_{(x,y)}[(y - 10x)^2]$$
$$= \frac{1}{3}\big((10-10)^2 + (20-20)^2 + (30-30)^2\big) = 0$$

Model B: $\hat{y} = 8x$

$$\mathbb{E}_{(x,y)}[(y - 8x)^2]$$
$$= \frac{1}{3}\big((10-8)^2 + (20-16)^2 + (30-24)^2\big) = 18.67$$

True relationship: $y = 10x$

Hours (x)	Score (y)
1	10
2	20
3	30

$\mathbb{E}_{(x,y)}$**= average error over all possible future data, not just training data**

General Bias–Variance Tradeoff

Estimate your function from a finite training data set $\mathcal{D}$, where $\hat{f}$ is a **random function** depending on the distribution of the training data:

$$\hat{f} = \arg\min_f \hat{L}(f) \coloneqq \frac{1}{m}\sum_{i=1}^{m}\left(y^i - f\left(x^i\right)\right)^2 \quad \Longrightarrow \quad \mathbb{E}_{\mathcal{D}}\left[\hat{f}(x)\right] \approx \frac{1}{N}\sum_{i=1}^{N}\hat{f}^{(i)}(x)$$

The expected loss of $\hat{f}$ (with respect to training data $\mathcal{D}$ and test data distributions):

$$L\left(\hat{f}\right) \coloneqq \mathbb{E}_{\mathcal{D}}\mathbb{E}_{(x,y)}\left[\left(y - \hat{f}(x)\right)^2\right]$$

Average prediction at x over all possible training datasets

Average prediction at x over test points drawn from the population

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

What is the Optimal Predictor?

$$L(\hat{f}) \coloneqq \mathbb{E}_{\mathcal{D}}\mathbb{E}_{(x,y)}\left[\left(y-\hat{f}(x)\right)^2\right]$$

$$= \mathbb{E}_{\mathcal{D}}\underbrace{\left[\iint \left(y-\hat{f}(x)\right)^2 p(x,y)dxdy\right]}_{A(\hat{f})}$$

$p(x,y)$: probability density that input x and output y occur together

The expected loss is:

$$\frac{\partial A(f)}{\partial f(x)} = \frac{\partial}{\partial f(x)}\iint \big(y-f(x)\big)^2 p(x,y)dxdy = -2\int \big(y-f(x)\big)p(x,y)dy = 0$$

This derivation identifies the best possible predictor, even with infinite data

What is the Optimal Predictor? (cont.)

$$\frac{\partial A(f)}{\partial f(x)} = \frac{\partial}{\partial f(x)} \iint \big(y - f(x)\big)^2 p(x,y)dxdy = -2\int \big(y - f(x)\big)p(x,y)dy = 0$$

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

The left-hand-side is $\int f(x)p(x,y)dy = f(x)p(x)$, hence:

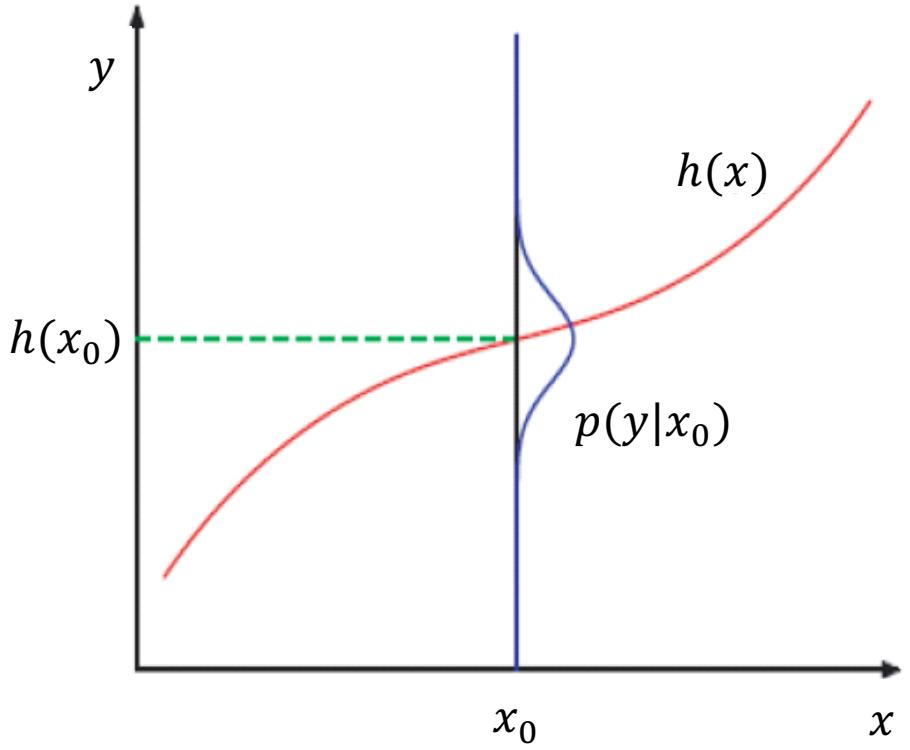
Following Baye's rule:
$p(x,y) = p(y|x)p(x)$

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

The conditional expectation minimizes mean squared error

The Best Predictor

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



Example

Suppose for $x_0 = 2$, the outcomes you have seen are $y = 8, 10, 12$

The average is

$$\frac{8 + 10 + 12}{3} = 10$$

So, the best predictor is

$$h(2) = 10$$

The "best predictor" is just: the average of all outcomes when x occurs

Expected Loss for Optimal Predictor

Given $h(x) = \mathbb{E}(y|x)$ as the **optimal predictor** and $\hat{f}(x)$ as our actual predictor

$$\mathbb{E}_{\mathcal{D}}\mathbb{E}_{(x,y)}\left[\left(y - \hat{f}(x)\right)^2\right] = \mathbb{E}_{\mathcal{D}}\left[\iint \left(\hat{f}(x) - h(x) + h(x) - y\right)^2 p(x,y)dxdy\right]$$

$$= \mathbb{E}_{\mathcal{D}}\left[\iint \left(\left(\hat{f}(x) - h(x)\right)^2 + 2\left(\hat{f}(x) - h(x)\right)(h(x) - y) + (h(x) - y)^2\right) p(x,y)dxdy\right]$$

($\hat{f}(x) - h(x)$ in the cross term: $= 0$)

$$= \mathbb{E}_{\mathcal{D}}\left[\int \left(\hat{f}(x) - h(x)\right)^2 p(x)dx\right] + \iint (h(x) - y)^2 p(x,y)dxdy$$

Will decompose further

Noise term (can't do better than this)

The noise term doesn't depend on the dataset, so $\mathbb{E}_{\mathcal{D}}$ has no effect (removed)

Why Does the Cross Term Vanish?

Since by definition $h(x) = \mathbb{E}(y|x)$, the equation is 0:

$$\iint \left(\hat{f}(x) - h(x)\right)(h(x) - y)p(x,y)dxdy$$

$$= \int \left(\hat{f}(x) - h(x)\right)\left(h(x) - \underbrace{\int y\, p(y|x)dy}_{\mathbb{E}(y|x) = h(x)}\right)p(x)dx$$

- $h(x) = \mathbb{E}(y|x)$ is the best predictor in MSE sense
- Deviations from $h(x)$ are uncorrelated with the noise

The cross term is zero because conditional expectation is the MSE minimizer

Bias–Variance Decomposition

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

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

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

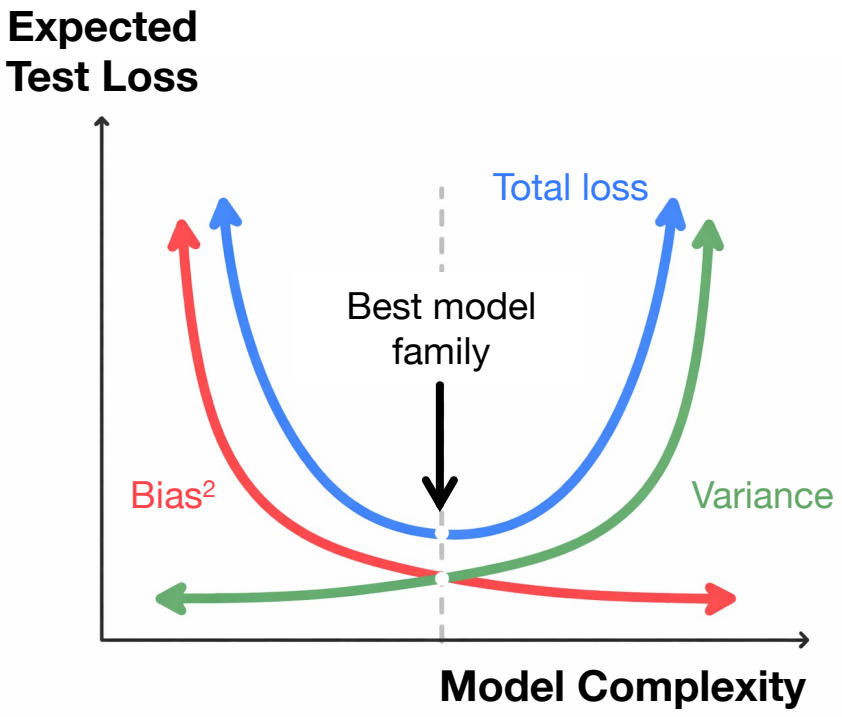
$$= \mathbb{E}_x\underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(\hat{f}(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}(x)]\right)^2\right]}_{\text{Variance at } x} + \mathbb{E}_x\underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(\mathbb{E}_{\mathcal{D}}[\hat{f}(x)] - h(x)\right)^2\right]}_{\text{Bias}^2 \text{ at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

The error splits into randomness (variance) and systematic error (bias)

Putting It All Together

Expected loss = (Bias)^2 + Variance + noise

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



$\hat{f}(x)$: actual predictor

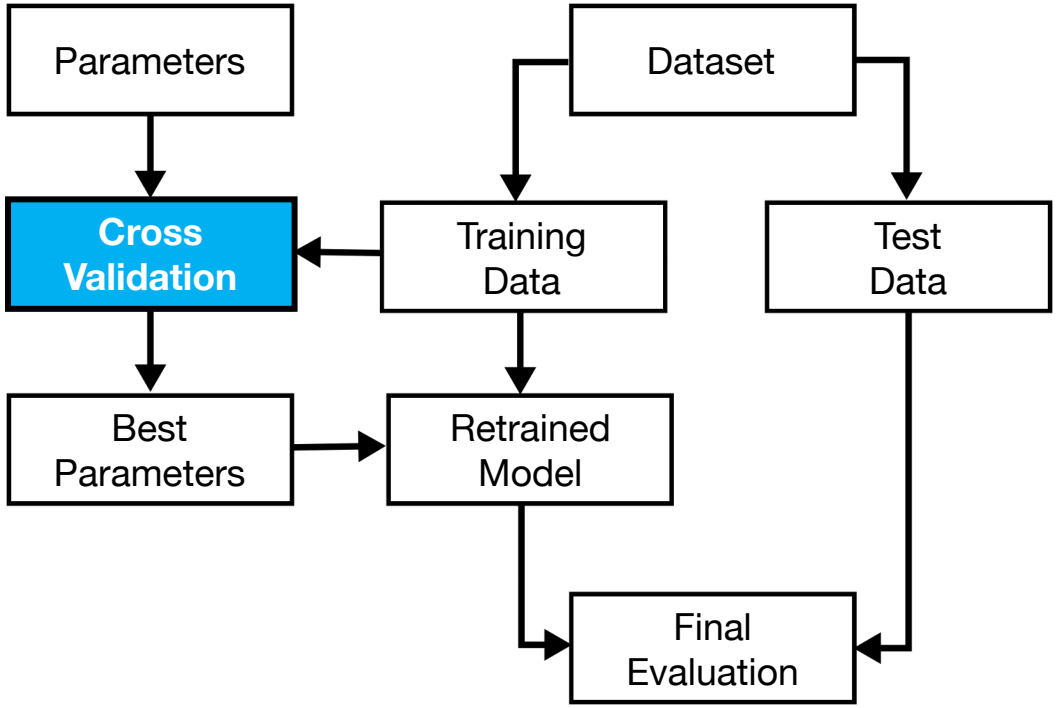
$\mathbb{E}_{\mathcal{D}}[\hat{f}(x)]$: expected predictor

$h(x) = \mathbb{E}(y|x)$: optimal predictor

Cross Validation

Cross Validation Workflow

Cross-validation is done only on training data—test data is used once at the end



What would go wrong if the test data leaked into cross validation?

Motivation

- **Training error is optimistically biased**, especially for complex models that can overfit the training data
- **To obtain an unbiased estimate of generalization performance**, we must evaluate the model on data not used for training
- A **simple train–test split** reserves part of the data for testing, but this can be inefficient when data are limited

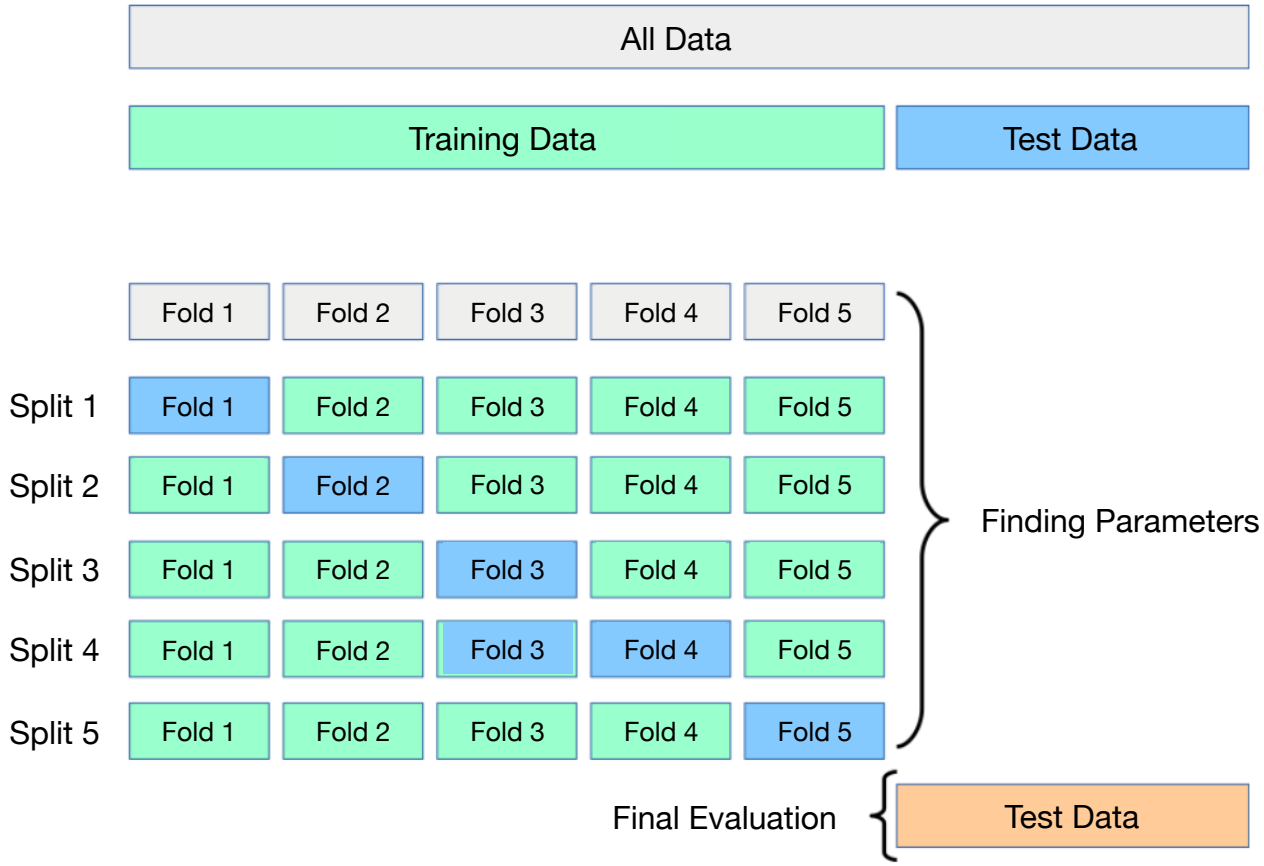
Cross validation helps select hyperparameters without using the test data

K-Fold Cross Validation

Each data point is used for validation exactly once and for training $K - 1$ times

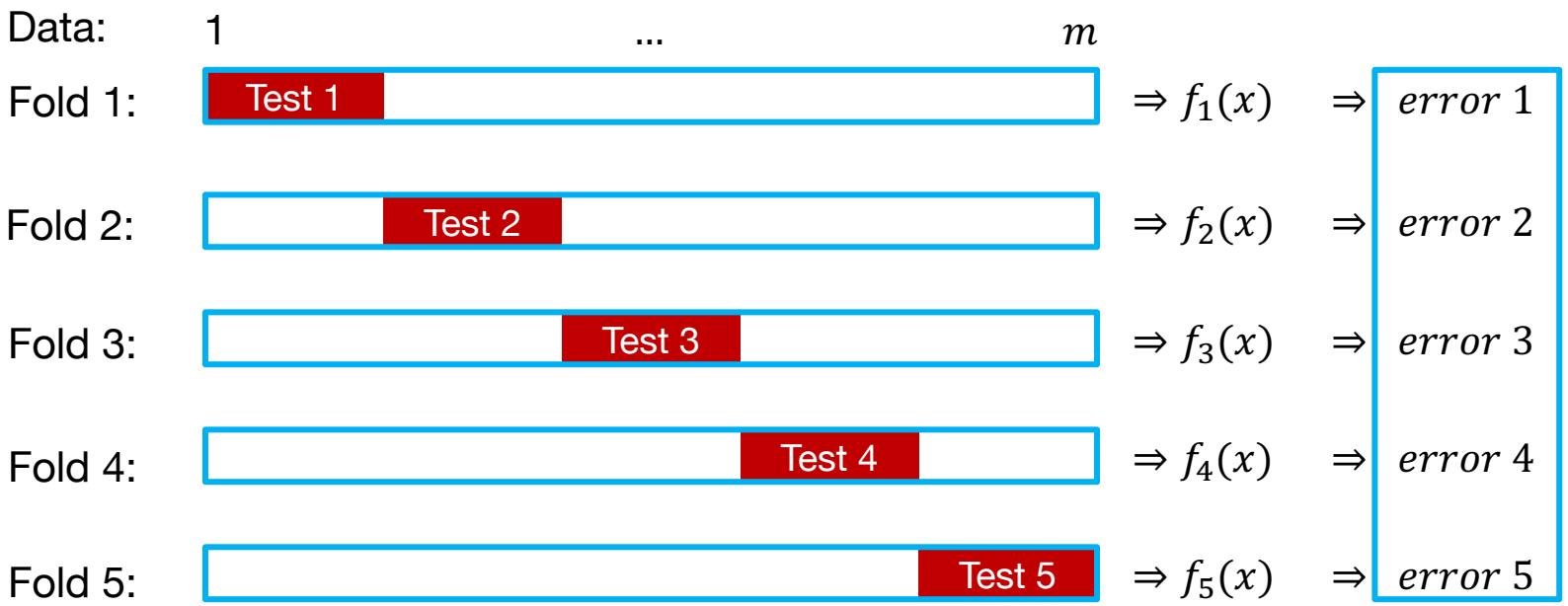
Process

- Train the model on $K - 1$ folds
- Validate it on the remaining fold
- Iterate this process K times (each fold serves as validation once)
- Average the validation errors



K-Fold Cross Validation (cont.)

Example: 5-fold cross-validation (blank: training; red: test)



Note: Validation data in block k is not used to fit model $f_k(x)$

Average **= Cross Validation Error**

The average validation error estimates the expected test error

K-Fold Cross Validation: Formal Definition

For a given model $M(\Theta)$ indexed by Θ, divide training data into K blocks—for each fold k, $k = 1, \ldots, K$

- Use $K - 1$ blocks as training data to fit the model $f_k(x)$ (trained without fold k)
- Use the remaining 1 block of data (X^k, Y^k) for validation to evaluate performance (choosing a **different** validation block at each time)
- Calculate cross validation error by choosing the Θ that minimizes:

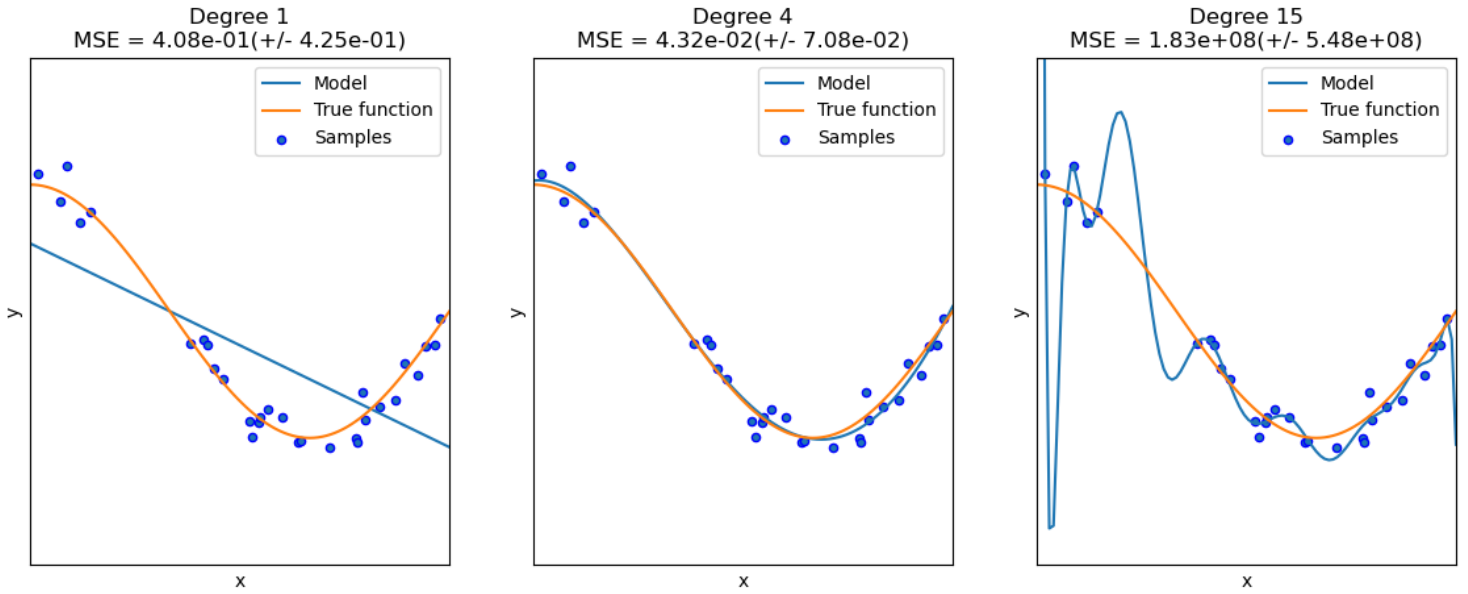
$$CV(\Theta) = \frac{1}{K} \sum_{k=1}^{K} L\left(Y^k, f_k\left(X^k\right)\right)$$

The minimizer of $CV(\Theta)$ determines the selected hyperparameters

Demo: Polynomial Fitting

Selects polynomial degree by balancing bias and variance:

Error for each fold (L samples) → $$MSE = \frac{1}{L}\sum_{l=1}^{L}\left(\tilde{y}^{l} - \hat{\theta}^{\top}\tilde{x}^{l}\right)^{2}$$



https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html#sphx-glr-auto-examples-model-selection-plot-underfitting-overfitting-py

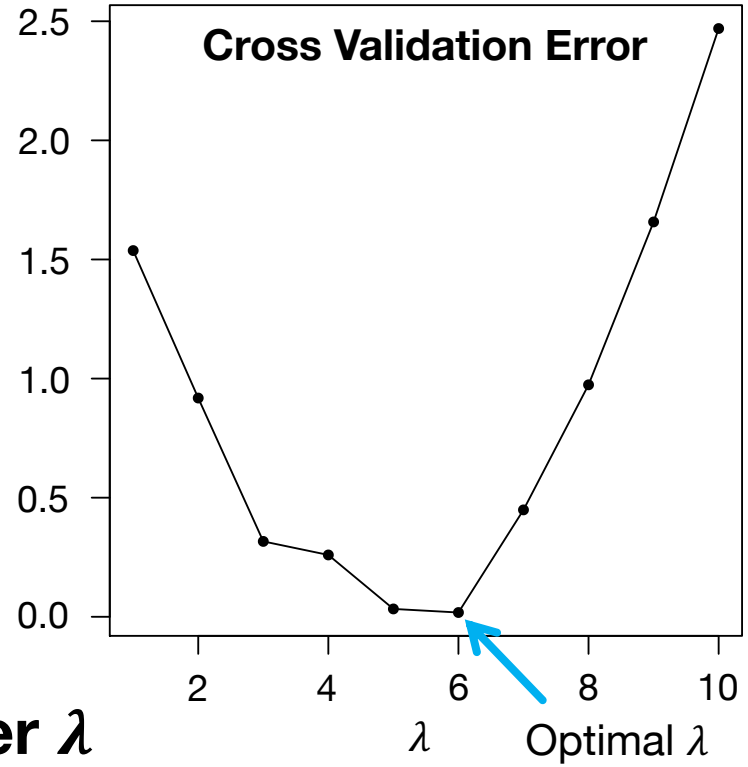
Demo: Ridge Regression

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

$$\theta = \left(\frac{1}{m}XX^\top + \lambda I\right)^{-1}\left(\frac{1}{m}\,Xy\right)$$

Error for each fold (L samples):

$$\text{MSE} = \frac{1}{L}\sum_{l=1}^{L}\left(\tilde{y}^l - \hat{\theta}^\top \tilde{x}^l\right)^2$$



Cross validation helps choose regularization parameter λ

Demo: Diabetes Dataset

Data:

- 442 diabetes patients
- 10 baseline predictors
- Age, sex, body mass index
- Average blood pressure
- Six blood serum measurements

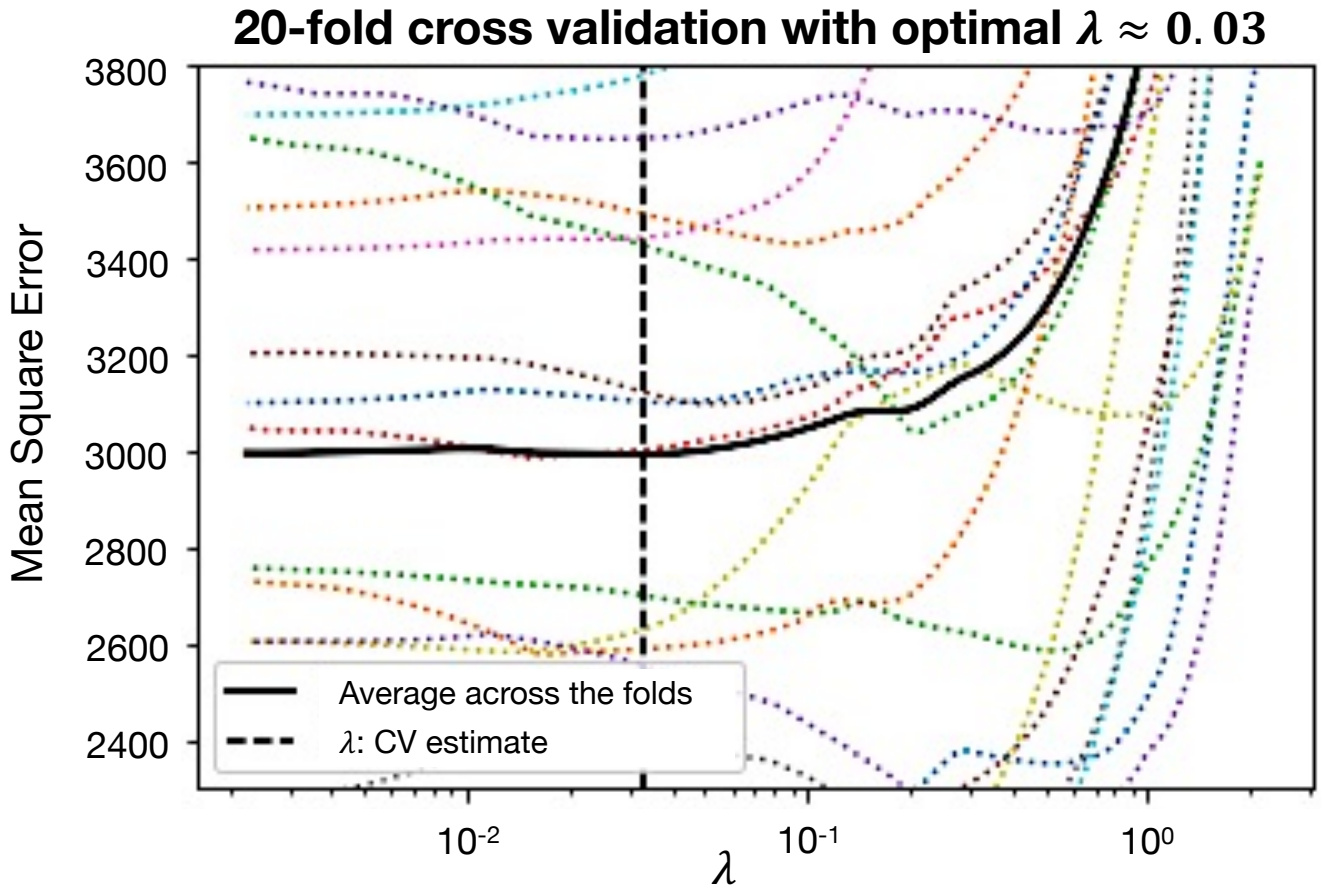
Response variable: Quantitative measure of disease progression one year after baseline

Model selection:

- Cross validation to select regularization parameter λ
- Controls model sparsity and prediction performance

Demo: Diabetes Dataset CV Using LASSO

Combining regularization, feature selection, and cross validation for choosing λ



$$\frac{1}{m}\sum_{i=1}^{m}\left(y^i - \theta^\top x^i\right)^2 + \lambda\|\theta\|_1$$

Key Takeaways

What We Learned This Week

- Model complexity governs the bias–variance tradeoff, with underfitting at low complexity and overfitting at high complexity
- Generalization error is the true objective, while training error is often misleading
- K-fold cross-validation estimates test error using only training data and avoids data leakage
- Cross-validation enables principled selection of models and hyperparameters
- Regularization methods such as Ridge and LASSO control complexity and improve stability
- Expected test error decomposes into bias, variance, and noise, clarifying the sources of prediction error
- Noise imposes an irreducible lower bound on prediction error