CS229 Notes - June 26, 2025

Yuren Hao

June 26, 2025

1 Maximum Likelihood Estimation (MLE)

Given i.i.d. observations $x_1, x_2, \ldots, x_n$ from $f(x;\theta)$.
Likelihood: $L(\theta) = \prod_{i=1}^{n} f(x_i;\theta)$
Log-likelihood: $\ell(\theta) = \sum_{i=1}^{n} \log f(x_i;\theta)$
MLE: $\hat{\theta}_{MLE} = \arg\max_\theta \ell(\theta)$
Solve: $\frac{\partial \ell(\theta)}{\partial \theta} = 0$

2 Linear Regression - Normal Equation

Given training data $\{(x_i, y_i)\}_{i=1}^{n}$ with $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$.
RSS: $RSS(\theta) = \sum_{i=1}^{n}(y_i - \theta^T x_i)^2 = \|y - X\theta\|_2^2$
Derivative: $\frac{\partial RSS(\theta)}{\partial \theta} = -2X^T(y - X\theta)$
Normal Equation: $X^T X\theta = X^T y$
Closed-form solution: $\hat{\theta} = (X^T X)^{-1} X^T y$

3 Gradient Descent for Linear Regression

Cost function: $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$
Gradient: $\frac{\partial J}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
Algorithm:

1. Initialize θ randomly
2. While not converged:
   - $\theta_j := \theta_j - \alpha\frac{\partial J}{\partial \theta_j}$ for all j
   - Check convergence: $|J(\theta^{(t+1)}) - J(\theta^{(t)})| < \epsilon$

Vectorized update: $\theta := \theta - \alpha\frac{1}{m}X^T(X\theta - y)$

3.1 Example: 3 Data Points

Data: $(1,1)$, $(2,2)$, $(3,4)$. Model: $h_\theta(x) = \theta_0 + \theta_1 x$
Initial: $\theta_0 = 0$, $\theta_1 = 0$, $\alpha = 0.1$
Iteration 1:

- $J(\theta) = \frac{1}{6}[(0-1)^2 + (0-2)^2 + (0-4)^2] = 3.5$
- $\frac{\partial J}{\partial \theta_0} = \frac{1}{3}[(-1) + (-2) + (-4)] = -2.33$
- $\frac{\partial J}{\partial \theta_1} = \frac{1}{3}[(-1)(1) + (-2)(2) + (-4)(3)] = -5.67$
- $\theta_0 := 0 - 0.1(-2.33) = 0.233$
- $\theta_1 := 0 - 0.1(-5.67) = 0.567$

Iteration 2: $J(\theta) = 1.26$ (continues until convergence...)

4 ML Framework & Loss Functions

4.1 General ML Pipeline

Model → **Algorithm** → **Estimated Parameters**
Predictions → **Decisions** → **Outcomes**
Example: Linear model → Gradient descent → $\hat{\theta}$
House price prediction → Buy/sell decision → Profit/loss

4.2 Loss Functions

Regression:

- **MSE:** $L(y, \hat{y}) = (y - \hat{y})^2$
- **MAE:** $L(y, \hat{y}) = |y - \hat{y}|$
- **Huber:** $L(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{if } |y - \hat{y}| \leq \delta \\ \delta|y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$

Classification:

- **0-1 Loss:** $L(y, \hat{y}) = \mathbf{1}[y \neq \hat{y}]$
- **Logistic:** $L(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$
- **Hinge:** $L(y, \hat{y}) = \max(0, 1 - y\hat{y})$ (SVM)

5 Training Loss vs Model Complexity

5.1 Bias-Variance Tradeoff

Low Complexity: High bias, low variance → **Underfitting**
High Complexity: Low bias, high variance → **Overfitting**

5.2 Typical Curves

- **Training Error:** Decreases as complexity increases
- **Validation Error:** U-shaped curve
- **Optimal Complexity:** Minimum validation error

Total Error = Bias^2 + Variance + Irreducible Error
Regularization: Controls complexity via penalty terms

- **L1 (Lasso):** $\lambda \sum_j |\theta_j|$ (sparse solutions)
- **L2 (Ridge):** $\lambda \sum_j \theta_j^2$ (smooth solutions)

6 Generalization Error

6.1 Definition

Generalization Error: Expected error on unseen data from same distribution
Continuous Case: For regression with squared loss

$$\text{Gen Error} = \mathbb{E}_{(x,y)\sim D}[(h(x) - y)^2] = \int_{x,y} (h(x) - y)^2 p(x, y)\, dx\, dy$$

If $p(x, y) = p(y|x)p(x)$, then:

$$= \int_x \left[\int_y (h(x) - y)^2 p(y|x)\, dy \right] p(x)\, dx$$

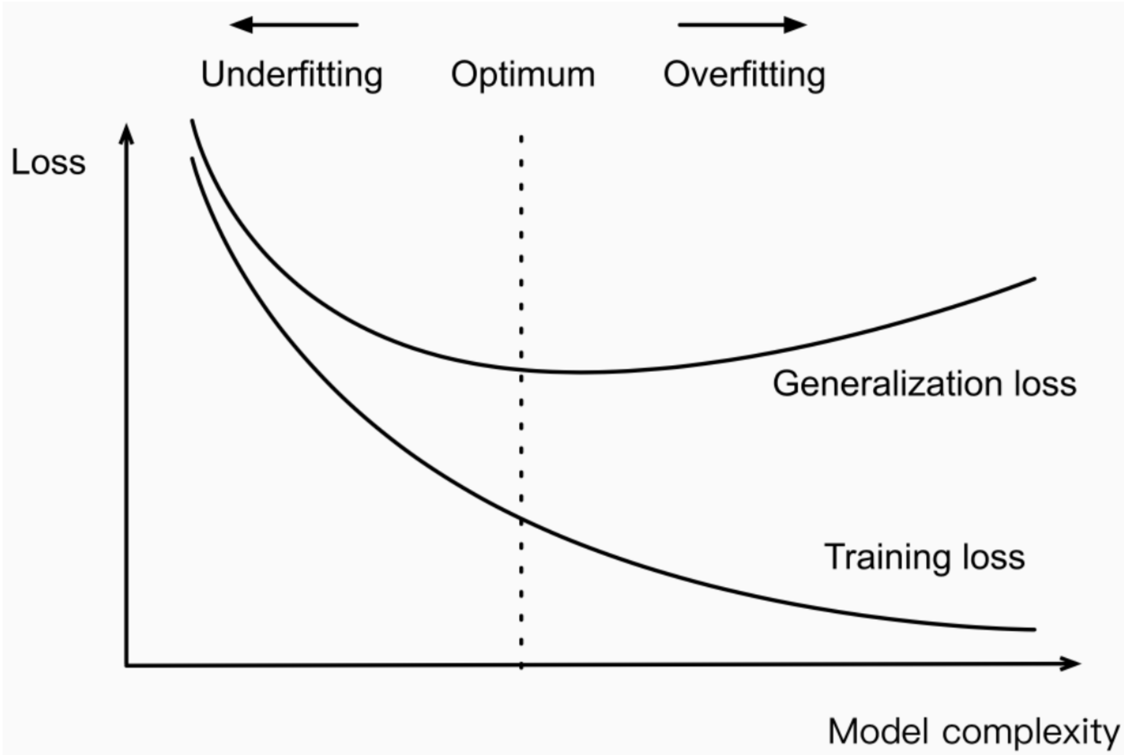

Figure 1: Model Complexity vs Training/Validation Error

Discrete Case: For classification with 0-1 loss

$$\text{Gen Error} = \mathbb{E}_{(x,y)\sim D}[\mathbf{1}[h(x) \neq y]] = \sum_{x,y} \mathbf{1}[h(x) \neq y] \cdot p(x,y)$$

Expanding the sum:

$$= \sum_{x} \sum_{y:y\neq h(x)} p(x,y) = \sum_{x} p(x) \sum_{y:y\neq h(x)} p(y|x)$$

Where D is the data distribution, h is the hypothesis
Empirical Risk: Approximation using training data
Continuous Case:

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} (h(x_i) - y_i)^2$$

Discrete Case:

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}[h(x_i) \neq y_i]$$

General Form:

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} L(h(x_i), y_i) \approx \mathbb{E}_{(x,y)\sim D}[L(h(x), y)]$$

Generalization Gap: $R(h) - \hat{R}(h)$ where $R(h)$ is true risk

6.2 Sources of Error

The expected prediction error can be decomposed into three fundamental components:
Expected Prediction Error:

$$\mathbb{E}[\text{Error}(x)] = \text{Noise} + \text{Bias}^2 + \text{Variance}$$

Detailed Breakdown:

- **Noise:** $\sigma^2 = \mathbb{E}[(y - f(x))^2]$ - Irreducible error from data
- **Bias:** $\text{Bias}[\hat{f}(x)] = \mathbb{E}[\hat{f}(x)] - f(x)$ - Model's systematic error
- **Variance:** $\text{Var}[\hat{f}(x)] = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$ - Model's sensitivity to training data

MSE Decomposition:

$$\text{MSE}(x) = \mathbb{E}[(\hat{f}(x) - y)^2] = \text{Bias}^2[\hat{f}(x)] + \text{Var}[\hat{f}(x)] + \sigma^2$$

Expanding the MSE:

$$= (\mathbb{E}[\hat{f}(x)] - f(x))^2 + \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2] + \mathbb{E}[(y - f(x))^2]$$

Experimental Setup for Bias-Variance Analysis:

We create N different training sets by sampling from the data distribution. Each training set produces estimated model parameters, giving us N different models $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_N$.

Use the average predictions of all the N estimated models, denoted by $\bar{f}_w(x)$:

$$\bar{f}_w(x) = \frac{1}{N} \sum_{i=1}^{N} \hat{f}_i(x)$$

The average fit is akin to the expected prediction $\mathbb{E}[\hat{f}(x)]$ over all possible training sets.

Bias Definition:

$$\text{Bias}(x) = f_{\text{true}}(x) - \bar{f}_w(x)$$

Key Question: Is our approach flexible enough to capture $f_{\text{true}}(x)$?

Bias is high when the hypothesis class is unable to capture $f_{\text{true}}(x)$. This happens when the model class is too simple or restrictive to represent the true underlying function.

Model Complexity vs Bias-Variance:

Low Complexity Model:

- **High Bias** - Cannot capture complex patterns in $f_{\text{true}}(x)$
- **Low Variance** - Predictions consistent across different training sets
- Example: Linear model for non-linear data

High Complexity Model:

- **Low Bias** - Can approximate $f_{\text{true}}(x)$ well
- **High Variance** - Predictions vary significantly with training data
- Example: High-degree polynomial or deep neural network

Key Insight: There's a fundamental tradeoff - reducing bias often increases variance, and vice versa.

6.3 Approximating Generalization Error

Since true distribution D is unknown, we approximate generalization error using: **Validation Set:** Hold-out data to estimate $R(h) \approx \hat{R}_{val}(h)$ **Cross-Validation:** Average over multiple train/validation splits to reduce variance. The key insight: empirical risk on unseen validation data provides unbiased estimate of generalization error.

6.4 Test Error Definition

Test Error: Performance on final held-out test set, used only once for final evaluation.

Continuous Case (Regression):

$$\text{Test Error} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (h(x_i^{test}) - y_i^{test})^2$$

Expanding the sum:

$$= \frac{1}{n_{test}} \left[(h(x_1^{test}) - y_1^{test})^2 + (h(x_2^{test}) - y_2^{test})^2 + \ldots + (h(x_{n_{test}}^{test}) - y_{n_{test}}^{test})^2 \right]$$

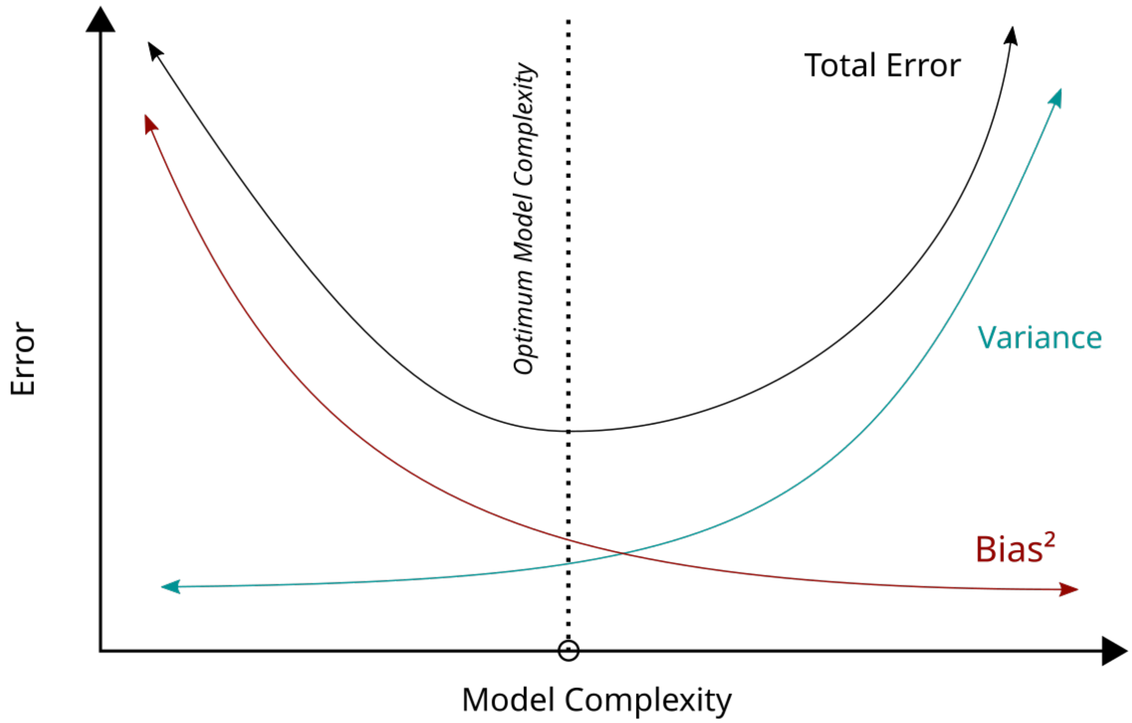

Figure 2: Bias and Variance Contributing to Total Error (Source: Wikimedia Commons)

Discrete Case (Classification):

$$\text{Test Error} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \mathbf{1}[h(x_i^{test}) \neq y_i^{test}]$$

Expanding the indicator sum:

$$= \frac{1}{n_{test}} \left[\mathbf{1}[h(x_1^{test}) \neq y_1^{test}] + \mathbf{1}[h(x_2^{test}) \neq y_2^{test}] + \ldots + \mathbf{1}[h(x_{n_{test}}^{test}) \neq y_{n_{test}}^{test}]\right]$$

Key Point: Test set should only be used once to avoid overfitting to test data.

6.5 Key Factors

- **Model Complexity:** More parameters → higher capacity to overfit
- **Training Set Size:** More data → better generalization
- **Regularization:** Penalty terms reduce overfitting
- **Early Stopping:** Stop training before overfitting occurs