CS 486/686
Introduction to Machine Learning

Yuntian Deng

Lecture 16

RN 19.1–19.2 · PM 7.1–7.2

Learning goals

- Identify reasons for building an agent that can **learn**.
- Describe different **types of learning** (supervised, unsupervised, RL).
- Define **supervised learning**, **classification**, and **regression**.
- Define **bias**, **variance**, and describe the trade-off.
- Use **cross-validation** to prevent overfitting.

Why an agent that learns?

Medical diagnosis

Spam filtering

Face recognition

Speech

Handwriting

Learning = the ability to improve on future tasks based on experience.

- We can't anticipate every situation.
- We can't anticipate every change over time.
- For many tasks, we have no idea how to *program* a solution — only how to *show examples*.

The learning architecture

Every learning system is defined by four ingredients:

Task

The behaviour we want to improve (classify, predict, decide).

Data

Experience used to improve (examples, demonstrations, rewards).

Background knowledge / bias

What we assume up front (model class, smoothness, priors).

Measure of improvement

How we score progress (accuracy, speed, new abilities).

Three kinds of learning

Supervised

Given input + target features, predict targets for new inputs.

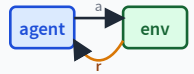
Today's focus.

Unsupervised

No targets — find structure: clustering, dimensionality reduction.

L17.

Reinforcement



Learn from rewards and punishments — what to *do*.

L15.

Supervised or unsupervised?

Q1. We have a user's credit-card transactions and want to flag any that look *different* from the rest. We have no fraud labels.

A. Supervised

B. **Unsupervised**

> **B — Unsupervised.** No target labels to predict.

Q2. We have historical weather labels (sunny/cloudy/rain/snow) for a date, and we want to predict the same date next year.

A. **Supervised**

B. Unsupervised

> **A — Supervised.** Each example has a target (the weather label).

Two flavors of supervised learning

Classification

Target features are **discrete** — e.g. dog vs cat.

Regression

Target features are **continuous** — e.g. tomorrow's temperature.

Quick check: classification or regression?

Q3. Historical weather labels (sunny / cloudy / rain / snow); predict next year's label for a given date.

A. **Classification**

B. Regression

> **A — Classification.** The target takes a discrete value from a fixed set.

Q4. Historical house prices over time; predict next month's price for a particular house.

A. Classification

B. **Regression**

> **B — Regression.** The target is a continuous real number.

Supervised learning = function approximation

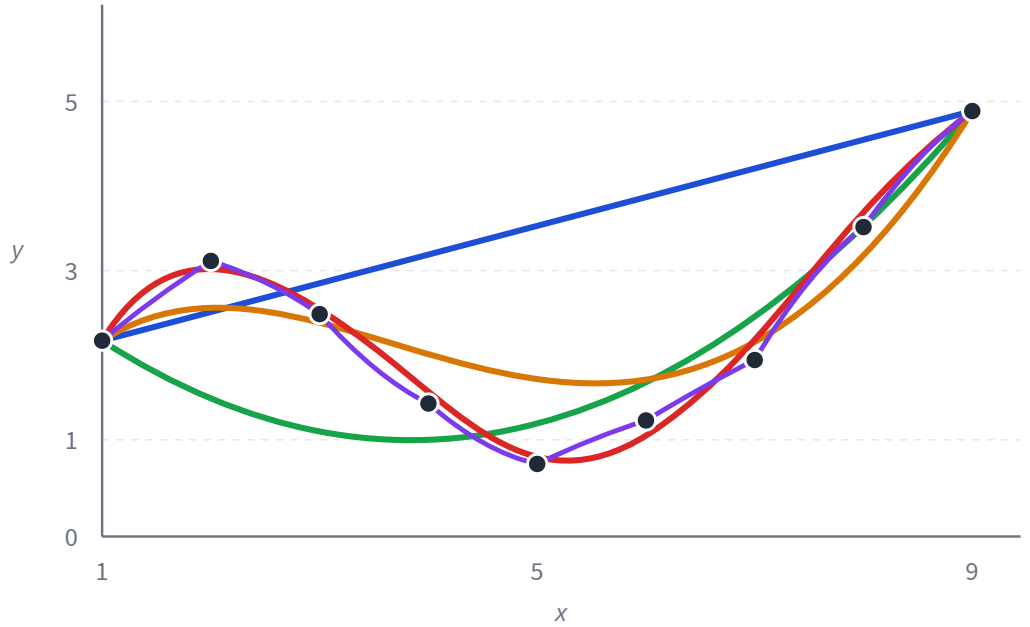
> Given training examples $(x_i, f(x_i))$, find a **hypothesis** h that approximates the unknown true function $f : X \to Y$.

A **hypothesis space** is the set of functions we'll consider (e.g. all polynomials of degree $\leq k$).

Learning = a search problem in the hypothesis space — usually some form of local search (gradient descent, etc.).

Example: fitting a polynomial

Same 9 data points, five candidate polynomial models. Watch how the fit changes with complexity.



degree 1 (line) degree 2 degree 3 degree 5 degree 9 (overfits)

Which curve is "right"?

- If the data has **outliers** or is noisy, simpler curves are more trustworthy.
- If the data really does have wild swings (stock prices, etc.), the complex curve is closer to truth.
- **There is no perfect answer** — each curve is justified *by some assumption*.

No-free-lunch

To learn something useful, we *must* make assumptions. Those assumptions are the **inductive bias** of the learner.

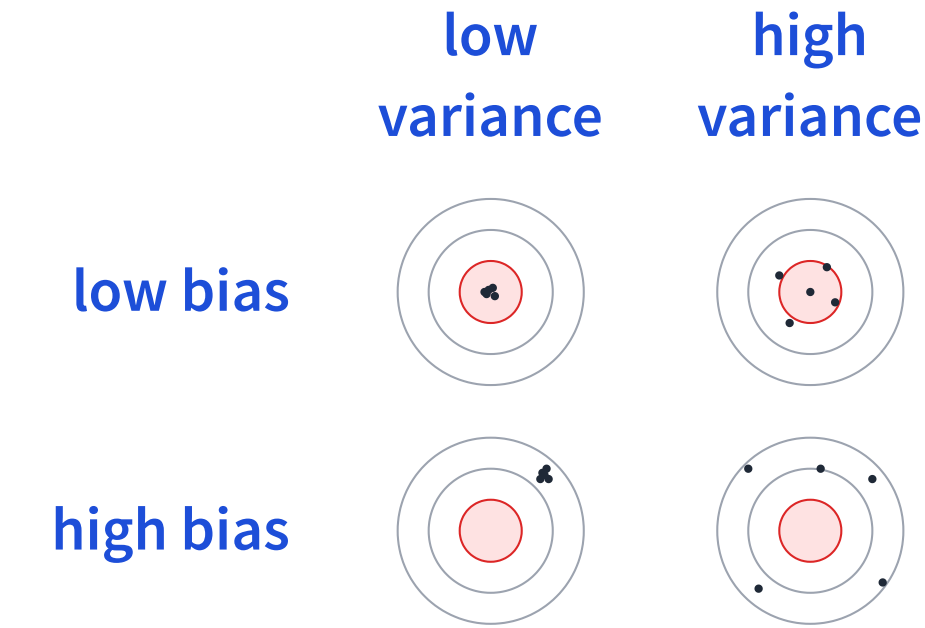
Bias and variance

Bias

With infinite data, how well can the family fit f?
High bias = too simple, **underfits**.

Variance

How much does the learned hypothesis change with different training sets? High variance = too flexible, **overfits**.



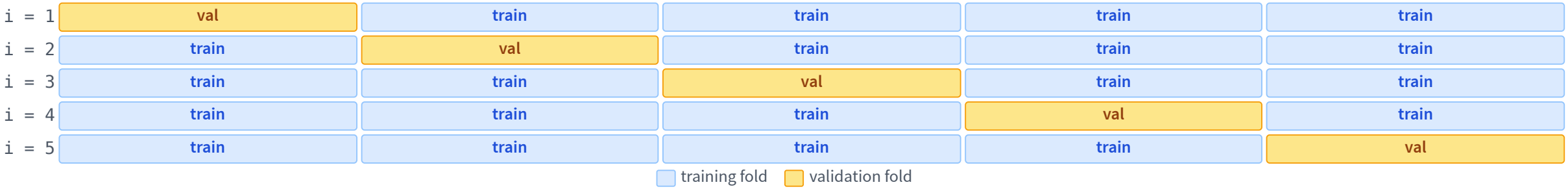
Dots = predictions from re-trained models.
Bullseye = truth.

K-fold cross-validation

How to pick a hypothesis that generalizes? Use a slice of training data as a **surrogate test set**.

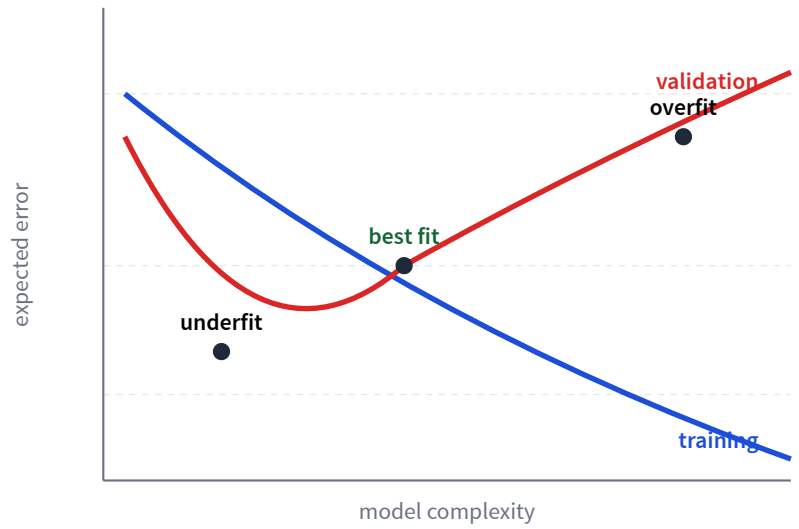
k-fold-CV(model, dataset, K)

1. Split training data into K equally sized folds.
2. For $i = 1, \ldots, K$: train on the other $K - 1$ folds; evaluate on fold i (held-out validation).
3. Average error over the K runs.



After CV: pick the best hyperparameters and (optionally) retrain on all data.

Overfitting: training vs validation error



Q5. Which is more likely to overfit?

A. Straight line (simple)

B. **4th-degree polynomial (complex)**

C. I don't know

B — complex. More degrees of freedom = more wiggle = it can chase the noise.

True error vs empirical error

Setup: domain X, labels Y, unknown true $f : X \to Y$; examples drawn from distribution D over X.

True error (what we want)

$$L_{D,f}(h) = \Pr_{x \sim D}\big[h(x) \neq f(x)\big]$$

Chance of being wrong on a fresh example. **Can't** compute it — D, f unknown.

Empirical error (what we measure)

$$L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$$

Fraction misclassified on the training set S (size m).

Hope: minimizing L_S also keeps $L_{D,f}$ small — *generalization*.

Empirical risk minimization (ERM)

Given a hypothesis class $\mathcal{H}$ and a loss ℓ, pick the hypothesis with smallest training loss:

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \ell\big(h(x_i),\, y_i\big)$$

Three factors determine how well ERM generalizes:

Data size m

Larger is better: more examples $\Rightarrow$ closer empirical $\rightarrow$ true.

Hypothesis space $\mathcal{H}$

Trade-off: small $\mathcal{H}$ = high bias; large $\mathcal{H}$ = high variance.

Loss function ℓ

Encodes what kind of error you care about (0/1, squared, hinge, ...).

Learning goals (recap)

✓ Identify reasons to build a learning agent.

✓ Describe different **types of learning**.

✓ Define **supervised learning**, **classification**, **regression**.

✓ Define **bias**, **variance**, and their trade-off.

✓ Use **cross-validation** to prevent overfitting.

Next: unsupervised learning

Today: labels guided every hypothesis we picked. Without labels, what *structure* can we still find in data?

L17: clustering, k-means, autoencoders.