

STA 35C: Statistical Data Science III

Lecture 14: k -fold Cross-Validation and the Bootstrap

Dogyoon Song

Spring 2025, UC Davis

Announcement

Homework 3 is due next Tuesday (May 6, 11:59 pm PT)

- Submit any 2 of Problems 1–3, **plus** Problems 4 and 5
- A new subproblem, 3(e), is added after the first release

Mid-course survey

- Thank you for your constructive feedback and requests
- I'll accommodate some requests, e.g., slow the pace, add more examples, reduce HW workload, etc.
- I cannot accommodate some requests, e.g., video recording
 - The authors' slides and [YouTube lectures](#) may be helpful
- Please follow the syllabus, announcements, and Piazza for supplementary notes
- If you're struggling with this course, let me know how I can help, and/or come see me

Office hours

- Regular: Wednesdays, 4:30–5:30 pm
- Extra, occasional (Thu 2:30–3:00 pm): May 8, May 22, May 29
- TA office hours: Mon/Thu 1-2 pm

Today's topics

- **Brief recap:**
 - Validation set (hold-out) for estimating test MSE
 - Leave-one-out cross-validation (LOOCV)
- **k -fold cross-validation**
- **The bootstrap:** quantifying uncertainty via resampling

Recap: Validation set approach

Holdout for validation:

- Training error \neq test error in general
- **Idea:** Split the training data and hold out part for validation to estimate test error



Figure: Splitting n observations into a **training set** and a **validation set**. The model is fit on the training set and assessed on the validation set [JWHT21, Figure 5.1]

Recap: Leave-one-out cross-validation (LOOCV)

Key ideas:

- For each observation, leave that single point as “validation,” train on the remaining $n - 1$ observations.
- Repeat for all n points, giving n different estimates of validation error.
- Average these n errors to approximate test error.

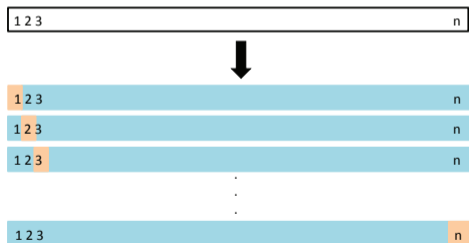


Figure: A set of n data points is repeatedly split into a training set of size $n - 1$ and a validation set of size 1. The test error is estimated by averaging the n partial MSEs [JWHT21, Figure 5.3]

Recap: Single validation vs. LOOCV

Validation (single split):

- Enables estimating test MSE from training data alone & widely applicable
- Highly variable across random splits & training is less efficient due to fewer data points in the training subset

LOOCV:

- Removes randomness of splitting; each model is trained on $n - 1$ samples
- Computationally expensive: requires fitting n models

To mitigate the computational burden:

Use fewer splits to reduce computational cost \implies **k -fold cross-validation**

k-fold cross-validation: 1) Basic ideas

Key ideas:

- Randomly partition the observations into k groups (folds) of roughly equal size
- LOOCV is a special case of k -fold CV with $k = n$
- In practice, common choices are $k = 5$ or $k = 10$

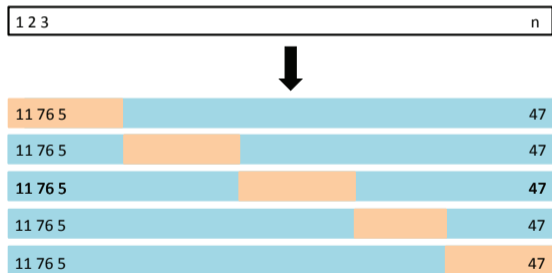


Figure: A schematic of 5-fold CV. The data are split into five non-overlapping groups: one as the **validation set** and the remainder as the **training set** [JWHT21, Figure 5.5].

k-fold cross-validation: 2) Procedure

Pseudocode:

- Randomly partition the data into k folds
- **For** each fold $j = 1, \dots, k$:
 - Take fold j as the validation set
 - Combine the other $k - 1$ folds into a training set and fit the model on it
 - Compute validation error MSE_j on fold j
- Estimate test MSE by averaging:

$$\widehat{\text{MSE}}_{\text{CV}} = \frac{1}{k} \sum_{j=1}^k \text{MSE}_j$$

k-fold cross-validation: 3) Example (2-fold, 4 data points)

Example (2-fold CV)

Let our dataset be $\{(5, 12), (7, 14), (12, 17), (16, 19)\}$ and choose $k = 2$:

- **Fold 1 (Validation):** $(5, 12), (12, 17)$
- **Fold 2 (Validation):** $(7, 14), (16, 19)$

Step 1: Train on Fold 2, validate on Fold 1.

Training set: $\{(7, 14), (16, 19)\}$.

$$\hat{\beta}_1 = \frac{19 - 14}{16 - 7} = \frac{5}{9} \approx 0.556, \quad 14 = 0.556 \times 7 + \hat{\beta}_0 \implies \hat{\beta}_0 \approx 10.108.$$

Hence, $\hat{y} = 10.108 + 0.556x$.

$$\hat{y}(5) = 10.108 + 0.556 \times 5 = 10.108 + 2.780 \approx 12.888 \quad (\text{actual} = 12),$$

$$\hat{y}(12) = 10.108 + 0.556 \times 12 = 10.108 + 6.672 \approx 16.780 \quad (\text{actual} = 17).$$

$$\text{MSE}_1 = \frac{(12 - 12.888)^2 + (17 - 16.780)^2}{2} = \frac{(-0.888)^2 + (0.220)^2}{2} = \frac{0.789 + 0.048}{2} = 0.419.$$

k-fold cross-validation: 3) Example (cont'd)

Example (2-fold CV continued)

Step 2: Train on Fold 1, validate on Fold 2.

Training set: $\{(5, 12), (12, 17)\}$.

$$\hat{\beta}_1 = \frac{17 - 12}{12 - 5} = \frac{5}{7} \approx 0.7143, \quad 12 = 0.7143 \times 5 + \hat{\beta}_0 \implies \hat{\beta}_0 \approx 8.4286.$$

Hence, $\hat{y} = 8.4286 + 0.7143x$.

On the validation points $\{(7, 14), (16, 19)\}$:

$$\hat{y}(7) = 8.4286 + 0.7143 \times 7 \approx 13.4286 \quad (\text{actual} = 14),$$

$$\hat{y}(16) = 8.4286 + 0.7143 \times 16 \approx 19.8574 \quad (\text{actual} = 19).$$

$$\text{MSE}_2 = \frac{(14 - 13.4286)^2 + (19 - 19.8574)^2}{2} \approx \frac{(0.5714)^2 + (-0.8574)^2}{2} = \frac{0.3265 + 0.7351}{2} \approx 0.5308.$$

Final 2-fold CV error:

$$\widehat{\text{MSE}}_{\text{CV}} = \frac{\text{MSE}_1 + \text{MSE}_2}{2} \approx \frac{0.4199 + 0.5308}{2} \approx 0.4754.$$

k-fold cross-validation: 4) the **auto** dataset

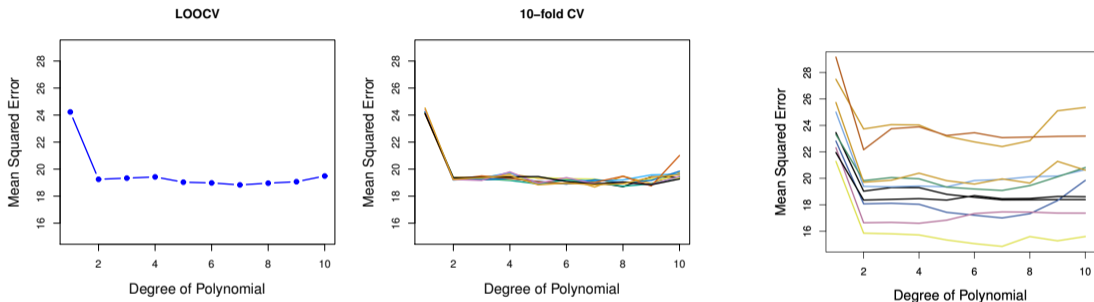


Figure: k -fold CV on the Auto dataset for polynomial fits of mpg on horsepower. **Left:** LOOCV error curve, **Center:** $k = 10$ CV curve, **Right:** single-split validation repeated ten times [JWHT21, Figures 5.2 & 5.4].

- k -fold CV offers a balance between single-split validation and LOOCV

k-fold cross-validation: 5) Pros and cons

Pros:

- Reduced variability compared to a single-split validation
- Fewer models to fit than LOOCV (esp. when $k \ll n$), lowering computational cost
- Each fold uses $\frac{k-1}{k} \cdot n$ points for training

Cons:

- Still computationally more expensive than a single-split approach
- Some randomness persists when compared to LOOCV

Recall: Sampling distribution

Coin flip example: Suppose we want to estimate $p = \Pr(X = 1)$, the probability of Head

We can flip a coin 10 times and compute $\hat{p} = \bar{X} = \frac{1}{10} \sum_{i=1}^{10} X_i$ to estimate p

Trial 1: 6 Heads, 4 Tails $\Rightarrow \hat{p}_1 = 0.6$



Trial 2: 4 Heads, 6 Tails $\Rightarrow \hat{p}_2 = 0.4$



⋮

Recall: Sampling distribution

As we repeat many trials, we obtain a distribution of \hat{p} (or equivalently \bar{X}):

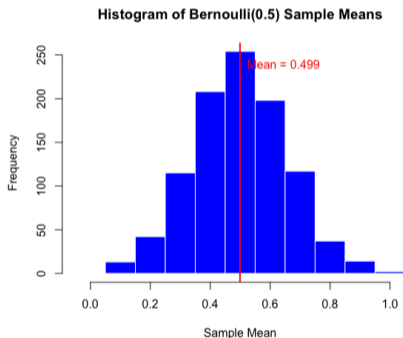


Figure: Histogram of 1000 sample means from 10 coin flips ($p = 0.5$).

This is called the **sampling distribution** of \hat{p}

- The estimate \hat{p} from a random sample is itself a random variable!
- Its variance reflects the uncertainty in \hat{p}

$$SE(\hat{p}) = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{p}_b - \bar{p})^2}$$

- In this example (see left), $B = 1000$,
 $\bar{p} = \frac{1}{1000} \sum \hat{p}_i = 0.499$
- This calculation requires 1000 *fresh* samples! (which we usually do not have)

Challenge: We have only “one” sample in practice

In most practical scenarios, we only have a single sample (=dataset)

Trial 1: 6 Heads, 4 Tails $\Rightarrow \hat{p}_1 = 0.6$



Question: How can we approximate the sampling distribution and estimate $SE(\hat{p})$, using *only* this single sample?

Answer: The bootstrap

We can “*sample*” from this given dataset to generate fresh, synthetic samples

The Bootstrap: 1) Motivation

Reasoning behind the bootstrap:

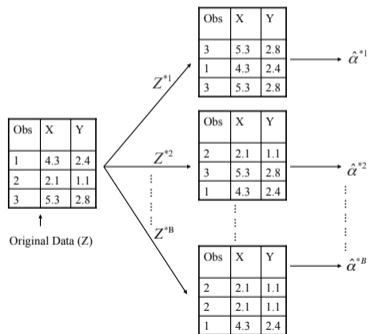
- We often want to measure the uncertainty of an estimate (mean, regression coefficients, etc.)
- Some estimates have known standard error formulas (like the sample mean in a simple scenario), but these rely on assumptions that may fail
- In more complex scenarios (e.g., optimal portfolios, complex regression), we may not have closed-form SE formulas for SE

Goal: Estimate uncertainty *without* strong parametric assumptions, by reusing our one dataset as if it were the population

Question: How do we resample to create synthetic samples from our data?

The Bootstrap: 2) (Re-)sampling with replacement

Core idea: Treat our dataset as an empirical approximation of the population



- Draw *resamples* of size n with *replacement* from the dataset
- Compute the statistic (mean, regression coefficient, etc.) each time
- The spread of these “bootstrap statistics” approximates the sampling distribution (and hence the SE)

Figure: Illustration of the bootstrap for $n = 3$ observations. Each bootstrap dataset yields an estimate of α [JWHT21, Figure 5.11].

The Bootstrap: 3) Procedure

Pseudocode: To estimate parameter α (e.g., sample mean, slope),

- Let $Z = \{Z_1, Z_2, \dots, Z_n\}$ be our observed data (of size n)
- **For** $b = 1$ to B :
 - Sample n observations *with replacement* from Z , call that Z_b^*
 - Compute $\hat{\alpha}_b^* = f(Z_b^*)$

Bootstrap SE formula:

$$\text{SE}_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{\alpha}_b^* - \bar{\hat{\alpha}}^*)^2} \quad \text{where} \quad \bar{\hat{\alpha}}^* = \frac{1}{B} \sum_{b=1}^B \hat{\alpha}_b^*$$

- Note that this is simply the sample standard deviation of $\{\hat{\alpha}_1^*, \dots, \hat{\alpha}_B^*\}$

Bootstrap: 4) An example of sample mean

Observed dataset ($n=5$):

$$x = \{2.1, 3.5, 1.8, 2.7, 3.2\}.$$

We want to estimate $\mu = \mathbb{E}[X]$ and its uncertainty, namely, $\text{SE}(\hat{\mu})$.

- **Original sample mean:**

$$\bar{x} = \frac{2.1 + 3.5 + 1.8 + 2.7 + 3.2}{5} \approx 2.66$$

- **Bootstrap replicates ($B = 1000$):**

- Draw 5 points *with replacement* from x to form each x_b^* .
- Compute \bar{x}_b^* for each.
- For example,
 - Sample 1: $\{2.1, 2.1, 3.5, 2.7, 3.2\} \implies \bar{x}_1^* = 2.72$
 - Sample 2: $\{3.5, 1.8, 1.8, 3.2, 3.2\} \implies \bar{x}_2^* = 2.70$

- **Distribution of \bar{x}_b^* :** Yields an approximate sampling distribution for \bar{x} .

Bootstrap: 4) An example of sample mean (R script)

```
# Define the observed dataset
x <- c(2.1, 3.5, 1.8, 2.7, 3.2)
n <- length(x)

# Number of bootstrap reps
B <- 1000

# Compute the original sample mean
original_mean <- mean(x)

# Initialize a vector
boot_means <- numeric(B)

# Perform the bootstrap
for (b in 1:B) {
  x_star <- sample(x, n, replace=TRUE)
  boot_means[b] <- mean(x_star)
}
```

```
# Plot histogram of bootstrap means
hist(boot_means,
     main="Bootstrap Sample Means",
     xlab="Bootstrapped Mean",
     col="skyblue",
     border="white")

# Add vertical line for original mean
abline(v=original_mean, col="red", lwd=2)

# Label original mean
text(x=original_mean,
     y=par("usr")[4]*0.9,
     labels=paste("Original mean:",
                  round(original_mean,3)),
     pos=4,
     col="red")
```

- Alternatively, use `boot()` in the “boot” package for built-in methods; see [JWHT21, Ch 5.3.4].

Bootstrap: 4) An example of sample mean (histogram)

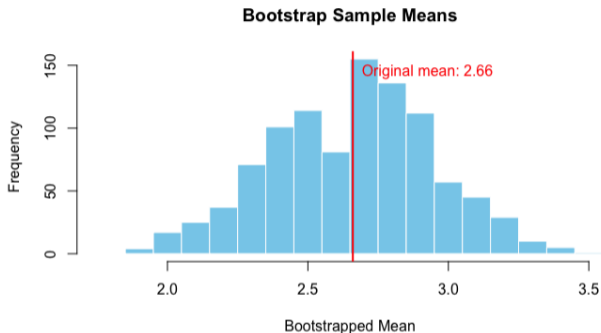


Figure: Histogram of 1000 bootstrap sample means for the example dataset.

Interpretation:

- The center is near $\bar{x} \approx 2.66$
- The spread shows how \bar{x} might vary if we repeatedly sampled from the (unknown) population

Bootstrap: 6) Pros and Cons

Pros:

- Requires minimal assumptions about the population distribution
- Straightforward to implement for many statistics (means, regression coefficients, etc.)
- Flexible for constructing confidence intervals via percentile methods, etc.

Cons:

- Potentially expensive for large n or complex models (because B can be large)
- Relies on the observed sample being representative of the true population (garbage in, garbage out)
- Less straightforward if data are highly dependent or from complex sampling designs

Wrap-up & Takeaways

- **Validation & Cross-Validation:**

- Single-split validation is simple but can vary a lot with random splits
- LOOCV removes randomness but is expensive
- k -fold CV strikes a balance between variance and computation

- **Bootstrap:**

- Resamples from dataset to approximate the sampling distribution of an estimate
- Widely used to get SEs and confidence intervals with minimal assumptions
- Particularly helpful for complex or unknown distributions
- Relies on the observed sample being representative of the true population

References



Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani.

An Introduction to Statistical Learning: with Applications in R, volume 112 of *Springer Texts in Statistics*.

Springer, New York, NY, 2nd edition, 2021.