STA 35C: Statistical Data Science III

Lecture 13: Cross-Validation

Dogyoon Song

Spring 2025, UC Davis

Today's topics

- Recap: Model assessment & the bias-variance tradeoff
- Motivation for resampling methods
- Key ideas in validation set approach
- Cross-validation techniques
  - Leave-one-out cross-validation (LOOCV)
  - k-fold cross-validation ($\rightarrow$ coming next lecture)

Assessing models: 1) Error metrics

Regression models: Commonly use **MSE** (Mean Squared Error):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

- Lower MSE indicates a better fit

Classification models: Often use **error rate**:

$$\text{Error Rate} = \frac{\#\,\text{Misclassified}}{\text{Total Sample Size}}$$

- Lower error rate indicates a better fit
- **False Positives (FP)** vs. **False Negatives (FN)** may also matter
- A confusion matrix or ROC curve can help visualize these outcomes

Assessing models: 2) Bias-variance tradeoff

Training vs. test performance:

- We fit a model using training data to reduce training MSE (or error rate)
- However, it may not generalize well to new (test) data

Bias-variance tradeoff:

- More flexible models tend to fit training data better, but can fail to generalize

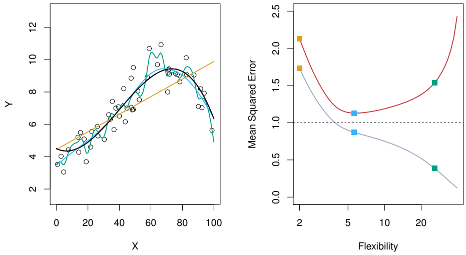


Figure: As model flexibility increases, training MSE typically goes down, while test MSE may go back up [JWHT21, Figure 2.9]

- High flexibility $\Longrightarrow$ low bias but potentially high variance
- Low flexibility $\Longrightarrow$ higher bias but lower variance

Open questions

We only have training data to fit our models, yet we want to:

- Estimate test performance (e.g., test MSE) to compare models
- Quantify uncertainty in the fitted model, akin to $\mathrm{SE}(\hat{\beta}_i)$ in linear regression

Open questions:

- How can we estimate test error *using only training data*?
- How can we perform valid inference (e.g., confidence intervals, significance tests) for *flexible or complex models beyond linear regression*?

Resampling methods

Ideally, if we could draw fresh test data from nature, we would:

- Train on one dataset, then measure performance on a new test dataset
- Re-draw multiple training sets to gauge uncertainty in our estimates

However, this is rarely feasible

Resampling methods in a nutshell:

- **Holdout approach:** Split the existing training data so that one portion acts as a surrogate test set
  $\rightarrow$ Cross-validation (today)
- **Resampling:** Treat our training data as if it were the "population," creating synthetic samples to estimate variability
  $\rightarrow$ The bootstrap (Friday; Lecture 14)

Validation set approach: 1) Basic ideas

Resampling viewpoint:

- In principle, we want to minimize test error, but we only have training data
- Training error $\neq$ test error in general
- **Idea:** Split the training data and hold out part for validation to estimate test error

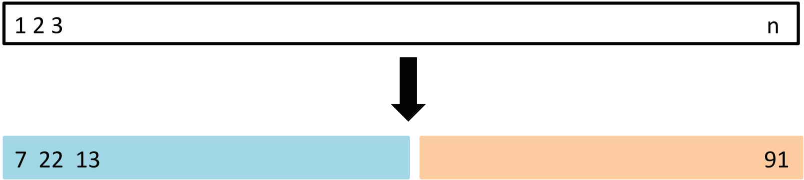


Figure: Splitting n observations into a training set and a validation set. The model is fit on the training set and assessed on the validation set [JWHT21, Figure 5.1]

Validation set approach: 2) Procedure

- **Step 1:** Randomly split the data into "training" and "validation" sets
- **Step 2:** Fit the model on the training set only
- **Step 3:** Evaluate performance on the validation set (estimate validation error)

Example (No split)

Given $\{(5, 12), (7, 14), (12, 17), (16, 19)\}$ for linear regression, we can fit on all points and compute the training MSE.

$$\hat{\beta}_1 \approx 0.6216, \quad \hat{\beta}_0 \approx 9.284 \quad \Longrightarrow \quad \mathrm{MSE}_{\mathrm{train}} \approx 0.101.$$

However, we'd have no insight into test MSE, because we have no held-out data.

What if we split? See next slide.

Validation set approach: 3) Example

Example (Split)

Suppose we have the dataset $\{(5,12),(7,14),(12,17),(16,19)\}$ and want to do linear regression.

- Let's say we randomly choose $(5,12)$ and $(12,17)$ for training, and keep $(7,14)$ and $(16,19)$ for validation.
- Fitting a simple linear model on the training set:

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

- Then predict on validation points:

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

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

Compute the validation MSE by averaging the squared errors:

$$\text{MSE}_{\text{val}} = \frac{(14-13.4286)^2 + (19-19.8574)^2}{2} \approx 0.53.$$

Validation set approach: 4) The `auto` dataset

Recall the `auto` dataset from Lecture 5, relating `mpg` (Y) to `horsepower` (X):

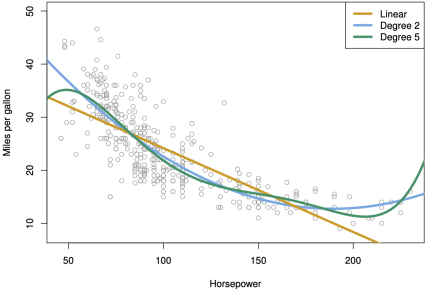


Figure: A scatter plot of the `auto` dataset suggests a noticeable non-linear relationship between `mpg` and `horsepower` [JWHT21, Figure 3.8].

We may consider a polynomial regression:

$$\texttt{mpg} \approx \beta_0 + \beta_1 \texttt{horsepower} + \cdots + \beta_p \texttt{horsepower}^p$$

Question: Should we add $\texttt{horsepower}^2$, $\texttt{horsepower}^3$, ...? Up to what degree?

Validation set approach: 4) The auto dataset (cont'd)

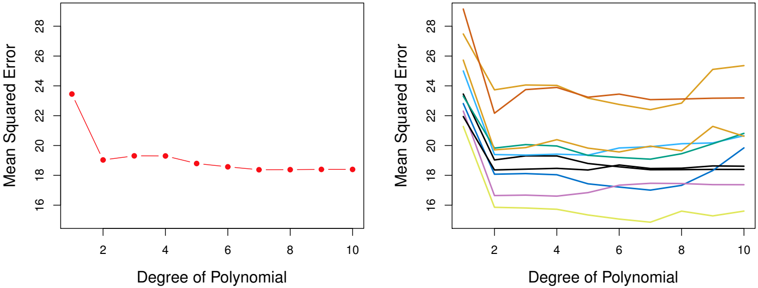


Figure: Using the validation set approach on the `auto` dataset to estimate test error for polynomial fits of `mpg` on `horsepower`. **Left:** Validation error for a single random split. **Right:** The same procedure repeated ten times with different random splits [JWHT21, Figure 5.2].

- **Left:** $\mathrm{MSE}_{\mathrm{val}}$ drops markedly (p: $1 \to 2$), indicating a simple linear model is suboptimal
- **Right:** We observe a large variability in $\mathrm{MSE}_{\mathrm{val}}$ due to different random splits

Validation set approach: 5) Benefits and drawbacks

Benefits:

- Allows estimating test MSE from training data alone
- Applies to any learning method (no special assumptions needed)

Drawbacks:

- High variability: a single random split may not be representative
- Reduced training data size (some portion is "held out") can lead to less efficient model fitting

Question: How can we refine the validation set approach to address the two issues?
$\Rightarrow$ **Cross-validation!** (Split multiple times and aggregate results)

Leave-one-out cross-validation: 1) Basic ideas

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

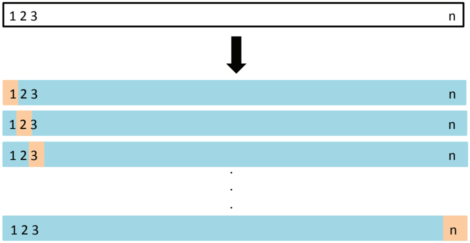


Figure: Splitting a set of n data points into a training set of size $n-1$ and a validation set of size 1, done n times [JWHT21, Figure 5.3]

Leave-one-out cross-validation: 2) Procedure

Pseudocode:

- **For** $i = 1$ to n:
  - Remove observation i to form

$$\mathcal{D}_i = \{(x_1, y_1), \ldots, (x_{i-1}, y_{i-1}), (x_{i+1}, y_{i+1}), \ldots, (x_n, y_n)\}$$

  - Fit the model (e.g., linear regression) on these $n-1$ points to get $\hat{f}_i : X \to Y$
  - Compute the squared prediction error for the held-out observation i:

$$\mathrm{MSE}_i = \left(y_i - \hat{f}_i(x_i)\right)^2$$

- Average the n errors to obtain the **LOOCV error**:

$$\widehat{\mathrm{MSE}}_{\mathrm{LOOCV}} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{MSE}_i$$

Leave-one-out cross-validation: 3) Example

Example (3 data points)

Let our dataset be $\{(x_1, y_1) = (5, 12), (x_2, y_2) = (7, 14), (x_3, y_3) = (12, 17)\}$.

Step 1: Leave out $(x_1, y_1) = (5, 12)$.

- Train on $\{(7, 14), (12, 17)\}$.

$$\hat{\beta}_1 = \frac{17 - 14}{12 - 7} = \frac{3}{5} = 0.6, \quad 14 = 0.6 \times 7 + \hat{\beta}_0 \implies \hat{\beta}_0 = 14 - 4.2 = 9.8.$$

So model: $\hat{y} = 9.8 + 0.6\,x$.

$$\mathrm{MSE}_1 = \left(12 - \hat{y}(5)\right)^2 = (12 - (9.8 + 0.6 \cdot 5))^2 = (12 - 12.8)^2 = 0.8^2 = 0.64.$$

(continues to the next slide)

Leave-one-out cross-validation: 3) Example (cont'd)

Example (3 data points)

(continued from the previous slide)

Step 2: Leave out $(x_2, y_2) = (7, 14)$. Similarly, we get

$$\hat{\beta}_1 \approx 0.7143, \quad \hat{\beta}_0 \approx 8.4286 \quad \Longrightarrow \quad \mathrm{MSE}_2 = (14 - \hat{y}(7))^2 \approx 0.3265.$$

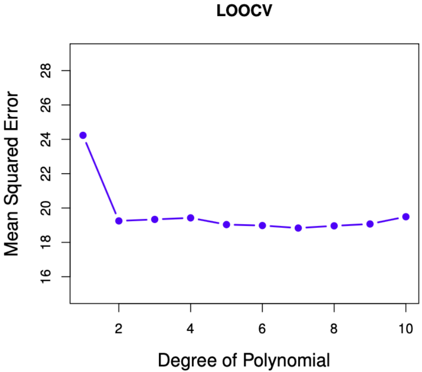
Step 3: Leave out $(x_3, y_3) = (12, 17)$. Similarly, we get

$$\hat{\beta}_1 = 1, \quad \hat{\beta}_0 = 7 \quad \Longrightarrow \quad \mathrm{MSE}_3 = (17 - \hat{y}(12))^2 = 4.$$

Final:

$$\widehat{\mathrm{MSE}}_{\mathrm{LOOCV}} = \frac{\mathrm{MSE}_1 + \mathrm{MSE}_2 + \mathrm{MSE}_3}{3} = \frac{0.64 + 0.3265 + 4}{3} \approx \frac{4.9665}{3} \approx 1.6555.$$

Leave-one-out cross-validation: 4) the `auto` dataset



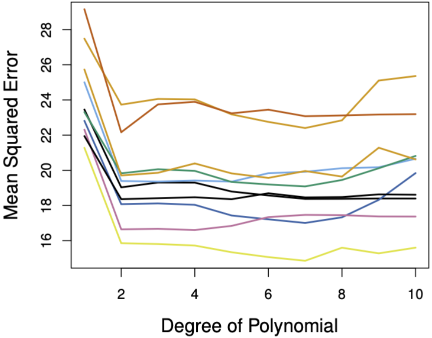


Figure: LOOCV applied to the `Auto` dataset for polynomial fits of `mpg` on `horsepower`. **Left:** LOOCV error curve. **Right:** Single-split validation repeated ten times [JWHT21, Figures 5.2 & 5.4].

- LOOCV yields a single test error estimate with no randomness in splitting

Leave-one-out cross-validation: 5) Pros and cons

Pros:

- Uses nearly all data for training ($n - 1$ points each time)
  - Better than the previous approach, where only $\sim \frac{n}{2}$ points were used
- No randomness from different splits; yields a single stable estimate

Cons:

- Requires fitting n separate models, which can be computationally expensive[1]

Question: How can we enjoy the benefits of LOOCV, while reducing computational burden?
$\Rightarrow$ k-**fold cross-validation** (Use fewer splits to reduce computational cost)

[1] *Note:* Least squares linear regression has a closed-form shortcut for LOOCV, reducing computation

Wrap-up

Key Takeaways:

- **Model assessment** relies on measuring performance beyond training data (e.g., test MSE, error rate)
- The **bias-variance tradeoff** explains why models that fit the training set closely may not generalize well to test data
- **Resampling methods** help us estimate test performance using only training data
  - **Validation set approach:** Simple but variable due to random splitting
  - **LOOCV:** Removes randomness and uses almost all data for training but is computationally expensive
  - ***k*-fold CV (next lecture):** A practical compromise between single-split validation and LOOCV

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.