STA 141A – Fundamentals of Statistical Data Science

Department of Statistics; University of California, Davis

Instructor: Dr. Akira Horiguchi (ahoriguchi@ucdavis.edu)
A01 TA: Zhentao Li (ztlli@ucdavis.edu)
A02 TA: Zijie Tian (zijtian@ucdavis.edu)
A03 TA: Lingyou Pang (lyopang@ucdavis.edu)

Section 8: Resampling methods

Spring 2025 (Mar 31 – Jun 05), MWF, 01:10 PM – 02:00 PM, Young 198

Overview

Based on Chapter 5 of ISL book James et al. (2021).

- For more R code examples, see R Markdown files in `https://www.statlearning.com/resources-second-edition`

Section 8: Resampling methods

- Cross-validation
- Bootstrap

Resampling methods are an indispensable tool in modern statistics.

- Idea: Repeatedly draw samples from a training set, then refit a model on each sample in order to get additional info about the fitted model.
- For example, in order to estimate the variability of a linear regression fit, we can repeatedly draw different samples from the training data, fit a linear regression to each new sample, and then examine the extent to which the resulting fits differ.
- Might provide information that would not be available from fitting the model only once using the original training sample.

We discuss two resampling methods: *cross-validation* and *bootstrap*

- *Cross-validation* can be used to estimate test error.
- *Bootstrap* can help to provide a measure of accuracy of a parameter estimate or of a given statistical learning method.

How does this help us?

- Evaluating a model's performance is known as *model assessment*.
- Helps to select proper level of flexibility for a model; process known as *model selection*.

Resampling methods

Cross-validation

Recall distinction between *test error rate* and *training error rate* of a predictor.

- Choose predictor that produces smallest test error (better generalization).
- Test error can easily be calculated if a designated test set is available, but usually this is not the case.
- How to estimate test error in such cases?
- Saw that training error rate is often quite smaller than the test error rate.
- Can instead train the predictor on a *subset* of the available data, then assess performance on the unused data.

For now we consider only regression (classification is similar).

Randomly split the available data in two sets of the same size: a *training set* and a *validation set* (or *hold-out set*).

- Procedure of the validation set approach:
  1. Randomly split the available data in two sets of the same size.
  2. Fit the model on the training set.
  3. Use the validation set to assess the performance of the fit (e.g., MSE)

Example: estimate test MSE for linear regression using the validation set approach

We want to do linear regression given the data set

$$(x_1, y_1) = (1, 12), (x_2, y_2) = (2, 14), (x_3, y_3) = (4, 12),$$
$$(x_4, y_4) = (6, 15), (x_5, y_5) = (8, 17), (x_6, y_6) = (9, 22).$$

We split the whole data set into two groups with three elements each.

```
set.seed(37)  # allows these "random" numbers to be reproduced later
n = 6
train_inds = sample(n, n/2)  # 6 2 3
valid_inds = (1:n)[-train_inds]  # 1 4 5
```

$\Rightarrow$ Training set: $(x_6, y_6) = (9, 22), (x_2, y_2) = (2, 14), (x_3, y_3) = (4, 12)$.
$\Rightarrow$ Validation set: $(x_1, y_1) = (1, 12), (x_4, y_4) = (6, 15), (x_5, y_5) = (8, 17)$.

Conceptually simple and easy to implement, but two major drawbacks:

- The validation estimate of the test error rate highly depends on the values in the validation set.

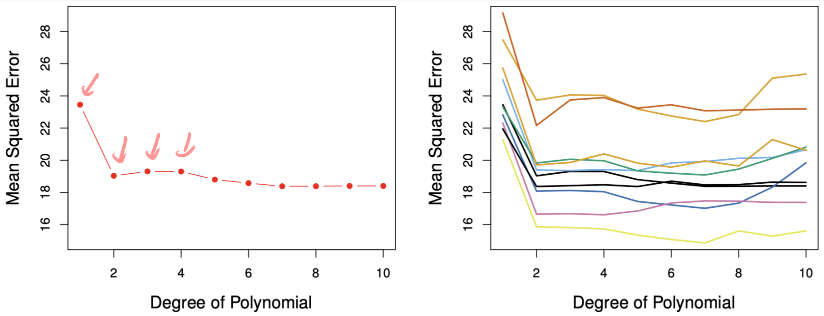


Figure 1: Image by James et al. (2021) using `Auto` data set of validation errors from predicting `mpg` using polynomial functions of `horsepower`. Left: one random split. Right: 10 random splits, illustrating variability in the estimated test MSE.

- Statistical methods tend to perform worse if trained on half of the whole data set compared to using the whole data set.

Leave-one-out cross validation (LOOCV): one data point for the validation set, and the remaining $n-1$ data points for the training set.

- Start by leaving (x_1, y_1) out, train our model on $(x_2, y_2), \ldots, (x_n, y_n)$, and predict y_1 by $\hat{y}_1$ based on the trained model, and calculate MSE_1.
- MSE_1 is based on a single observation (x_1, y_1), making it highly variable and hence a poor estimate for the test error. Thus we repeat the LOOCV by leaving out (x_2, y_2), then (x_3, y_3), etc.

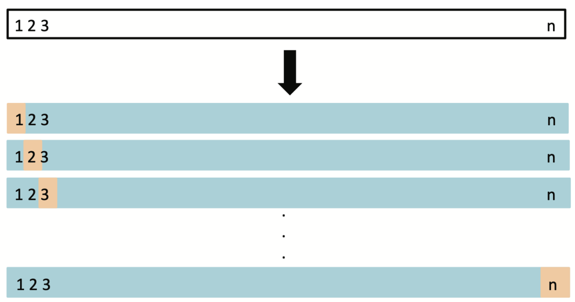


Figure 2: Image by James et al. (2021).

LOOCV (IDEA)

Procedure of the LOOCV, given the data $(x_1, y_1), \ldots, (x_n, y_n)$:

- 1st step:
  - Leave (x_1, y_1) out, and use it as validation set.
  - Derive an estimator $\hat{f}_1$ based on the training set $(x_2, y_2), \ldots, (x_n, y_n)$.
  - Calculate $MSE_1 := (y_1 - \hat{y}_1)^2$, where $\hat{y}_1 = \hat{f}_1(x_1)$.
- $\vdots$

  MSE_2
  MSE_3
- nth step:
  - Leave (x_n, y_n) out, and use it as validation set.
  - Derive an estimator $\hat{f}_n$ based on the training set $(x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$.
  - Calculate $MSE_n := (y_n - \hat{y}_n)^2$ where $\hat{y}_n = \hat{f}_n(x_n)$.
- $(n + 1)$st step: Calculate the LOOCV estimate for the test MSE, namely

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} MSE_i.$$

Example: estimate test MSE for linear regression using LOOCV.

Data set $(x_1, y_1) = (5, 50), (x_2, y_2) = (6, 60), (x_3, y_3) = (4, 20)$, so $n = 3$.

1. Leave out $(x_1, y_1) = (5, 50)$.
   Train $\hat{f}_1$ on $(x_2, y_2) = (6, 60), (x_3, y_3) = (4, 20) \Longrightarrow \hat{f}_1(x) = 20x - 60$.
   As $\hat{f}_1(5) = \hat{y}_1 = 40$, get $MSE_1 = (y_1 - \hat{y}_1)^2 = (50 - 40)^2 = 100$.
2. Leave out $(x_2, y_2) = (6, 60)$.
   Train $\hat{f}_2$ on $(x_1, y_1) = (5, 50), (x_3, y_3) = (4, 20) \Longrightarrow \hat{f}_2(x) = 30x - 100$.
   As $\hat{f}_2(6) = \hat{y}_2 = 80$, get $MSE_2 = (y_2 - \hat{y}_2)^2 = (60 - 80)^2 = 400$.
3. Leave out $(x_3, y_3) = (4, 20)$.
   Train $\hat{f}_3$ on $(x_1, y_1) = (5, 50), (x_2, y_2) = (6, 60) \Longrightarrow \hat{f}_3(x) = 10x$.
   As $\hat{f}_3(4) = \hat{y}_3 = 40$, get $MSE_3 = (y_3 - \hat{y}_3)^2 = (20 - 40)^2 = 400$.

Thus the test-MSE estimate for linear regression is

$$CV_{(3)} = (100 + 400 + 400)/3 = 300.$$

We could also compute $CV_{(3)}$ for a quadratic fit, and then choose the model — linear fit vs quadratic fit — that produces the smaller $CV_{(3)}$ value.

Pros:

- Compared to the validation set approach, we have a larger sample size $n - 1$ for the training data instead of only approximately half, thus LOOCV tends not to overestimate the test error rate.
- In LOOCV every data point is left out once, so data splits are not random (unlike in validation set approach).
- LOOCV is a very general method and can be used for many statistical learning methods (also logistic regression and LDA etc.).

Cons: LOOCV can computationally be very expensive since n predictors are fit.

- Exception: with least squares linear or polynomial regression, the cost of LOOCV is (amazingly!) the same as that of a single model fit:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - h_i} \right)^2$$

  where the leverage h_i is defined in the textbook (don't need to remember this for HW/exams).

k-fold CV randomly splits the given data with n elements in k groups (*folds*) of approximately equal size, by leaving the first fold out as a validation set, using the remaining $k-1$ folds as a training set, and repeating the procedure k times.

- Could do: permute indices $1, 2, \ldots, n$, then partition into k folds.

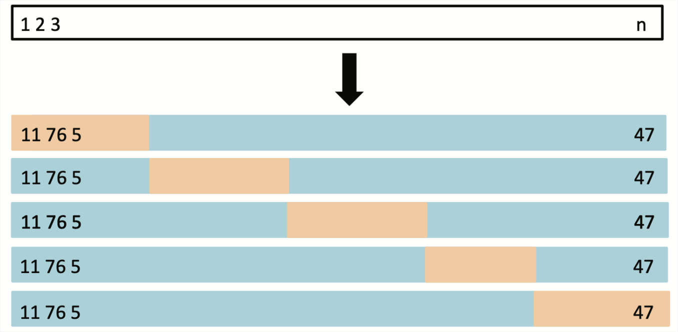


Figure 3: Image by James et al. (2021). Here we chose to use $k = 5$.

k-FOLD CV (PROCEDURE)

Procedure of the k-fold CV, given the data $(x_1, y_1), \ldots, (x_n, y_n)$:

- 0st step: Randomly split the given data in k folds (k is predefined).
- 1st step:
  - Leave the 1st fold out, and use it as validation set.
  - Derive an estimator $\hat{f}$ based on the remaining $k-1$ folds.
  - Calculate MSE_1 based on the 1st left out fold (if $n = 100$ and $k = 5$, so we have $k = 5$ folds with $n/k = 20$ elements each, then with I_1 denoting the set of the indices of all elements in the first fold (e.g. $I_1 = \{1, 3, 5, 10, 11, 86, \ldots, 100\}$), we have $MSE_1 = \frac{1}{n/k} \sum_{i \in I_1} (y_i - \hat{y}_i)^2$).
- $\vdots$
- kth step:
  - Leave the kth fold out, and use it as validation set.
  - Derive an estimator $\hat{f}$ based on the remaining $k-1$ folds.
  - Calculate MSE_k based on the kth left out fold.
- $(k+1)$st step: Calculate the k-fold CV estimate for the test MSE, namely

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i \,. \tag{1}$$

k-FOLD CV (COMMENTS)

k-fold CV generalizes LOOCV ($k = n$), but often use $k = 5$ or $k = 10$ in practice.

- If $k < n$, then k-fold CV is less computationally expensive than LOOCV.
- Another advantage of k-fold CV involves bias-variance trade-off.
  - Two sources of variability: (1) random data split and (2) data from unk. distr.

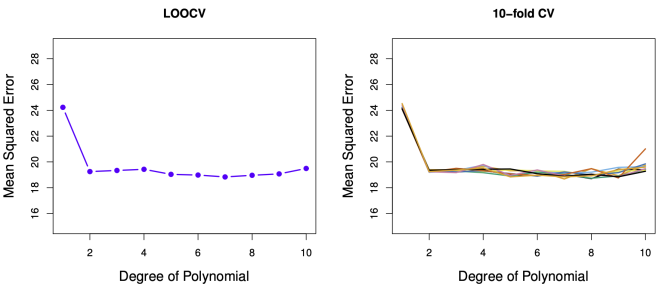


Figure 4: Image by James et al. (2021) using **single** `Auto` data set of validation errors from predicting `mpg` using polynomial functions of `horsepower`.

- LOOCV has smallest bias compared to k-fold CV for any other k; gives approx. unbiased estimates of the test error since each training set has $(n-1)$ obs.
- LOOCV also has the largest variance; because the n fitted models are trained on almost identical data sets, their outputs are highly positively correlated, so the variance does not lessen much when averaging over the n fitted models.

Model assessment vs model selection

When examining data, we usually do not know true test MSE, making it difficult to determine accuracy of the cross-validation estimate.

- If we examine simulated data, then we can compute the true test MSE.
- Select flexibility level that produces smallest estimated test error.

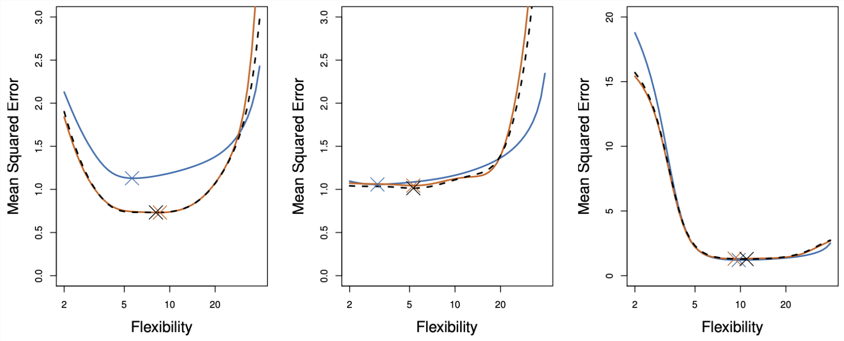


Figure 5: Image by James et al. (2021). For three simulated data sets, shows true test MSE (blue), LOOCV estimate (black dashed), and 10-fold CV estimate (orange). Cross indicates minimum of MSE curve.

End of May 12 lecture

Cross-validation can also be used for qualitative responses (in classification).

- The LOOCV error rate in the classification setting takes the form

Same as squared error

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} Err_i \,, \qquad (2)$$

  where $Err_i := 1_{\{y_i \neq \hat{y}_i\}}$ is 1 if $y_i \neq \hat{y}_i$ (obs i is misclassified), and 0 if $y_i = \hat{y}_i$ (obs i is assigned to correct class).
- Bias-variance tradeoff again in Figures 5.7 and 5.8 of James et al. (2021).

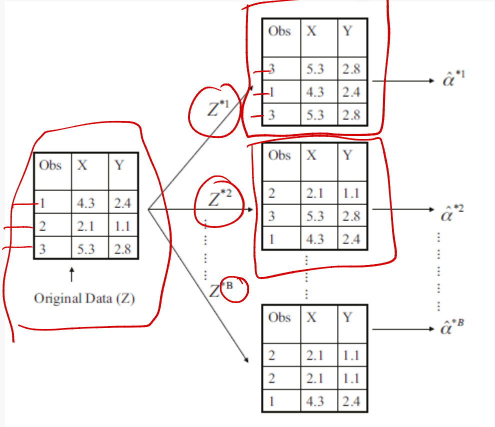
Resampling methods

Bootstrap

IDEA

The *bootstrap* is a widely applicable and extremely powerful statistical tool.

- Cross-validation randomly samples full data set without replacement; bootstrap randomly samples full data set **with replacement**. (Picture)



- Useful for many purposes, including for quantifying the uncertainty associated with a given estimator or statistical learning method.
- Easier to illustrate through an example.

Example

Suppose we wish to invest a fixed sum of money in two financial assets that yield (random) returns of X and Y.

- We will invest a fraction β of our money to asset 1; fraction $1 - \beta$ to asset 2.
- Returns are random; want β that minimizes total risk of our investment, i.e., that minimizes $Var(\beta X + (1 - \beta)Y)$.
- Letting $\sigma_X^2 = Var(X)$, $\sigma_Y^2 = Var(Y)$, $\sigma_{XY} = Cov(X, Y)$, can show minimizer is

$$\frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}} =: \alpha. \tag{3}$$

- σ_X^2, σ_Y^2, σ_{XY} usually unknown; can estimate (3) by estimating σ_X^2, σ_Y^2, σ_{XY} by e.g., sampling 100 pairs of returns to get an estimate $\hat{\alpha}$ for (3).
- This provides one value of $\hat{\alpha}$; how good is this estimate?
- Get $B = 1000$ new data sets by sampling 100 pairs of returns **from true population** B times. Then compute B ests $\hat{\alpha}^1, \ldots, \hat{\alpha}^B$ and their std error:

$$SE_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{i=1}^{B}\left(\hat{\alpha}^i - \frac{1}{B}\sum_{j=1}^{B}\hat{\alpha}^j\right)^2}.$$

- If we cannot sample from true population, get B "new" data sets by instead repeatedly sampling **with replacement** from original 100 pairs. Then compute standard error of the B **bootstrap** estimates $\hat{\alpha}^{*1}, \ldots, \hat{\alpha}^{*B}$.

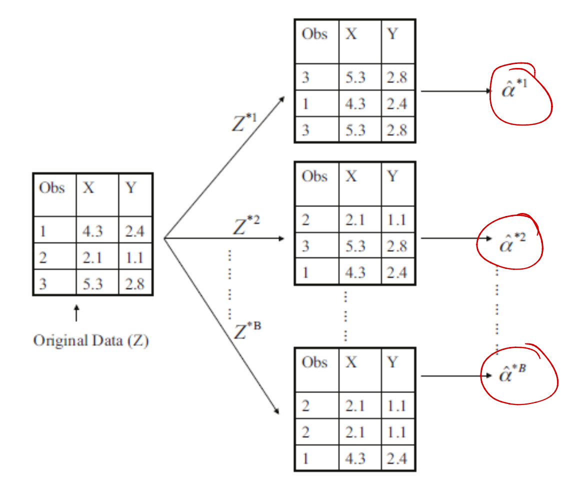


Figure 6: Table by James et al. (2021). We gathered $n = 3$ measurements of a certain species, sampled B times by randomly selecting values from the n observations (with replacement) and obtained the B bootstrap data sets $Z^{*1}, \ldots, Z^{*B}$ for a large number B. Based on the bootstrap data sets, we can derive estimators $\hat{\alpha}^{*1}, \ldots, \hat{\alpha}^{*B}$.

How similar is the distribution of the bootstrap estimates $\hat{\alpha}^{*1}, \ldots, \hat{\alpha}^{*B}$ to the distribution of the estimates $\hat{\alpha}^{1}, \ldots, \hat{\alpha}^{B}$ from the true population?

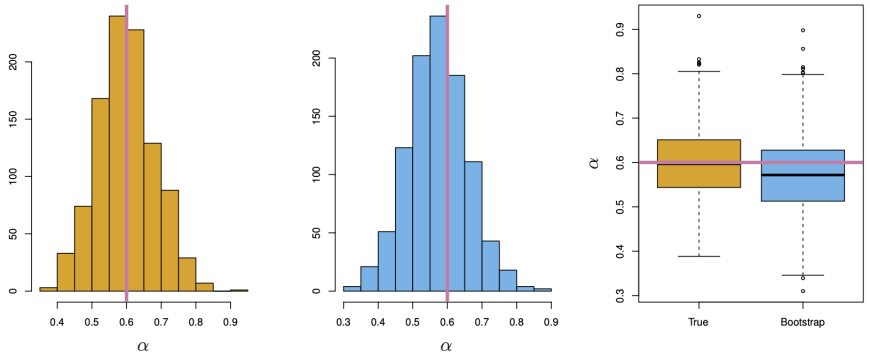


Figure 7: Image by James et al. (2021). Left: histogram of estimates of α obtained by generating 1000 simulated data sets from true population. Center: histogram of estimates of α obtained from 1000 bootstrap samples from a single data set. Right: boxplots of estimates in Left and Center. Pink lines indicate true value of α.