Resampling Methods: Cross Validation

Rebecca C. Steorts, Duke University

STA 325, Chapter 5 ISL

Agenda

- Re-sampling Methods
- Validation Method
- Leave one out cross validation (LOOCV)
- k-fold cross validation (CV)
- Application

Re-sampling Methods

A re-sampling method involves repeatedly drawing samples from a **training data set** and refitting a model to obtain addition information about that model.

Example: Suppose we want to know the variability associated with a linear regression model.

1. Draw different samples from the training data
2. Fit a linear regression to **each sample**
3. Examine how the fits differ

Re-sampling Methods

In this module, we focus on cross-validation (CV) and the bootstrap.

- CV can be used to estimate the test error associated with a statistical learning method to evaluate its performance or to select a model's level of flexibility
- The boostrap most commonly measures the accuracy of a parameter estimate or of a given statistical learning method.

1. Model assessment: the process of evaluating a model's performance
2. Model selection: the process of selectiing the proper level of flexibility for a model

Test error versus Training error (review, Ch 2)

- The **test error** is the average error that results from using a statistical learning method to predict the response on a new observation.
- The test error can be easily calculated if a designated test set is available. (Sometimes this is not the case).
- The **training error** is the average error that results from using a statistical learning method to predict the response on the data that was used to train the method.

Test Error (via Holding Out Training Data)

- In the absence of a very large designated test set that can be used to directly estimate the test error rate, a number of techniques can be used to estimate this quantity using the available training data.
- (More details in Chapter 6 using mathematical adjustments.)
- Consider a class of methods that estimate the test error rate by holding out a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations.
- For simplicity we assume that we are interested in performing regression with a quantitative response.

The Validation Set Approach

Goal: we want to estimate the test error associated with fitting a particular statistical learning method on a set of observations.

We first consider the **validation set approach**

The Validation Set Approach

1. Randomly divide the available set of observations into two parts, a training set and a validation set or hold-out set.
2. Fit the model on the training set.
3. Use the resulting fitted model to predict the responses for the observations in the validation set.
4. The resulting validation set error rate is typically assessed using the MSE in the case of a quantitative response. This provides an estimate of the test error rate.

The Validation Set Approach

Figure 1: Toy data set with n observations.

The Validation Set Approach

Figure 2: A set of n observations are randomly split into a training set (shown in blue, containing observations 7, 22, and 13, among others) and a validation set (shown in beige, and containing observation 91, among others). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.

Drawbacks to the Validation Approach

1. The validation estimate of the test error rate can be highly variable, which depends on which observations are in the training and validations sets.
2. Only a subset of the observations (training set) are used to fit the model.

The validation set error may tend to over-estimate the test error rate for the model fit on the entire data set.[1]

CV addresses both issues above!

[1] This might happen if the training set is very small.

Leave-One-Out Cross-Validation (LOOCV)

LOOCV does not create two subsets of comparable size. Instead it does the following:

1. A single observation (x_1, y_1) is used for the validation set.
2. The remaining observations $\{(x_2, y_2), \ldots, (x_n, y_n)\}$ make up the training set.
3. The statistical learning method is fit on the $n - 1$ training observations, and a prediction y_1 is made for the excluded observation, using its value x_1.
4. Since (x_1, y_1) was not used to fit the model, $\text{MSE}_1 = (y_1 - \hat{y}_1)^2$ provides an approximately unbiased estimate for the test error.

This isn't enough though, because it's only for one single observation.

Leave-One-Out Cross-Validation (LOOCV)

- We repeat the procedure by selecting (x_2, y_2) on the validation set.
- We train the model on the $n - 1$ observations that are leftover.
- We compute the MSE_2 as we did before, now using (x_2, y_2).

We now repeat this process approach n times to produce n squared errors,

$$\text{MSE}_1 \ldots, \text{MSE}_n.$$

The LOOCV estimate for the test MSE is the average of these n test error estimates:

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

Leave-One-Out Cross-Validation (LOOCV)

Figure 3: Toy data set with n observations. (The same as the validation approach.)

Leave-One-Out Cross-Validation (LOOCV)

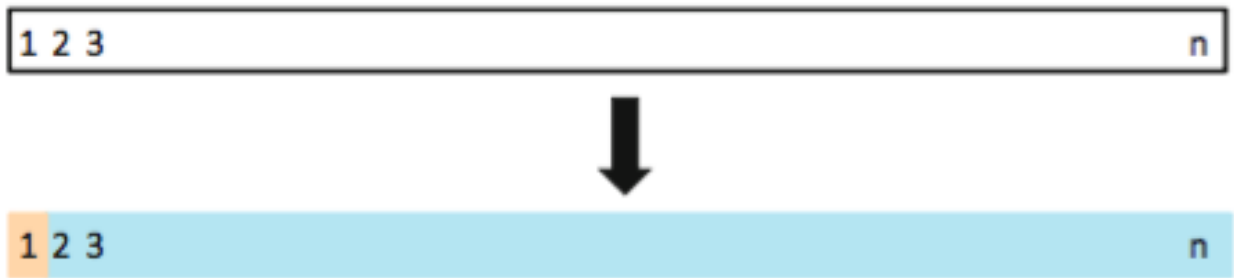

Figure 4: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

Figure 5: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

Figure 6: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

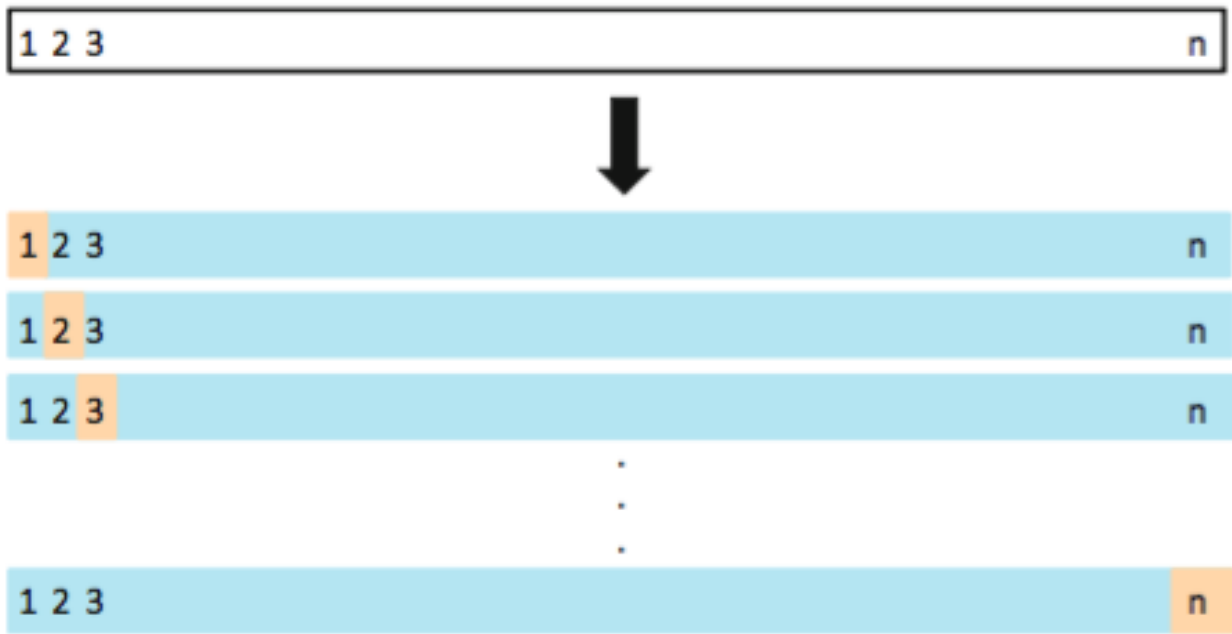

Figure 7: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.

LOOCV versus the Validation Method

LOOCV has a couple of major advantages over the validation set approach.

1. It has far less bias.

   - LOOCV: Repeatedly fit the statistical learning method using training sets that contain $n - 1$ observations, there are almost as many as are in the entire data set.
   - This contrasts the validation method, where the training set is about half the size of the original data set.
   - LOOCV approach tends not to overestimate the test error rate as much as the validation set approach does.

2. Performing LOOCV multiple times produces similar results. This is not typically true for the validation method.

Remark: LOOCV has the potential to be expensive to implement, since the model has to be fit n times. This can be very time consuming if n is large, and if each individual model is slow to fit.

k-fold CV

This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size.

1. The first fold is treated as a validation set, and the method is fit on the remaining $k - 1$ folds.
2. The mean squared error, MSE_1, is then computed on the observations in the held-out fold.
3. This procedure is repeated k times;

- Each time, a different group of observations is treated as a validation set.
- This process results in k estimates of the test error, $\text{MSE}_1, \ldots, \text{MSE}_k$.

4. The k-fold CV estimate is computed by averaging these values:

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} \text{MSE}_i \tag{2}$$

k-fold CV

- Notice that LOOCV is a special case of k-fold CV when $k = n$.
  - In practice, one typically performs k-fold CV using $k = 5$ or $k = 10$. Why?
  - What is the advantage of using $k = 5$ or $k = 10$ rather than $k = n$?
  - The most obvious advantage is computational since LOOCV requires fitting the statistical learning method n times.

5-fold CV

Figure 8: Toy data set with n observations. (The same as the validation approach.)

5-fold CV

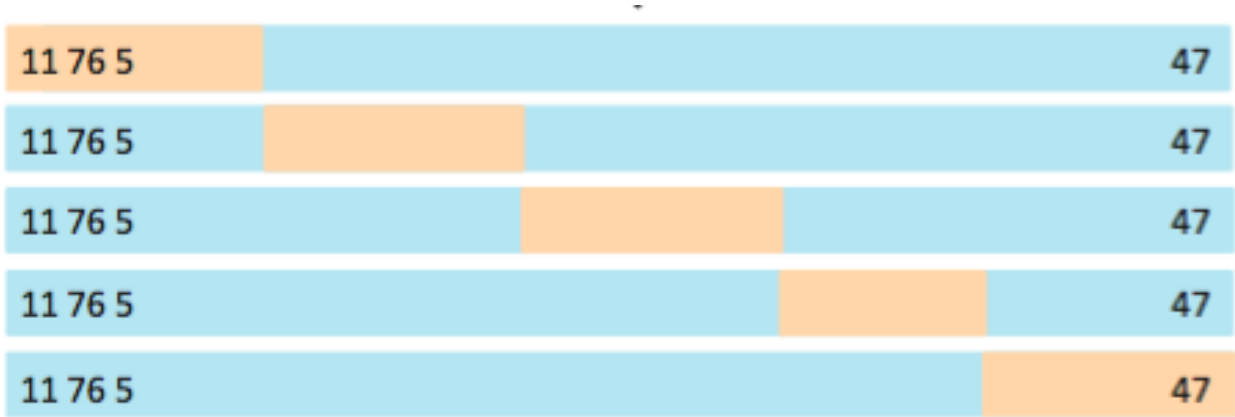

Figure 9: A set of n observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.

Bias-Variance Trade-Off for k-Fold Cross-Validation

- Recall that k-fold CV with $k < n$ has a computational advantage to LOOCV.
- But putting computational issues aside, a less obvious but potentially more important advantage of k-fold CV is that it often gives **more accurate estimates** of the test error rate than does LOOCV.
- This has to do with a bias-variance trade-off.

Bias-Variance Trade-Off for k-Fold Cross-Validation

- Recall that the validation set approach can lead to overestimates of the test error rate, since in this approach the training set used to fit the statistical learning method contains only half the observations of the entire data set.
- Using this logic, it is not hard to see that LOOCV will give approximately unbiased estimates of the test error, since each training set contains $n - 1$ observations, which is almost as many as the number of observations in the full data set.
- Performing k-fold CV for, say, k $= 5$ or k $= 10$ will lead to an intermediate level of bias, since each training set contains $\frac{(k-1)n}{k}$ observations — fewer than in the LOOCV approach, but substantially more than in the validation set approach.

Therefore, from the perspective of bias reduction, LOOCV is preferred to k-fold CV.

What about the variance?

Bias is not the only source for concern in an estimating procedure; we must also consider the procedure's variance.

It turns out that LOOCV has higher variance than does k-fold CV with $k < n$.

What about the variance?

- When we perform LOOCV, we are in effect averaging the outputs of n fitted models, each of which is trained on an almost identical set of observations.
- These outputs are highly (positively) correlated with each other.
- In contrast, when we perform k-fold CV with $k < n$, we are averaging the outputs of k fitted models that are somewhat less correlated with each other, since the overlap between the training sets in each model is smaller.
- Since the mean of many highly correlated quantities has higher variance than does the mean of many quantities that are not as highly correlated, the test error estimate resulting from LOOCV tends to have higher variance than does the test error estimate resulting from k-fold CV.

Bias-Variance Tradeoff for k-fold CV

There is a bias-variance trade-off associated with the choice of k in k-fold cross-validation.

Typically, given these considerations, one performs k-fold cross-validation using $k = 5$ or $k = 10$, as these values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance.

Cross-Validation on Classification Problems

- We have illustrated the use of CV in the regression setting where the outcome Y is quantitative, and so have used MSE to quantify test error.
- CV can also be a very useful approach in the classification setting when Y is qualitative.
- CV works the same as above except that rather than using MSE to quantify test error, we instead use it to quantify the number of misclassified observations.

Cross-Validation on Classification Problems

For instance, in the classification setting, the LOOCV error rate takes the form

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \text{Err}_i, \tag{3}$$

where $\text{Err}_i = I(y_i \neq \hat{y}_i)$.

- The k-fold CV error rate and validation set error rates are defined analogously.

See Section 5.1.5 for more details and further reading.

An Application to the Auto data set

We explore the use of the validation set approach in order to estimate the test error rates that result from fitting various linear models on the Auto data set.

We first do some data set up and pre-processing before we begin.

An Application to the Auto data set

- We set a seed to make our methods reproducible
- Use the sample() function to split the set of observations into two equal parts.
- We randomly select a random subset of 196 observations out of the original data set of 392 observations. We refer to this new sample as the **training set**

Data set up

```
library(ISLR)
set.seed (1)
# randomly select 196 units
# from the original data set.
train <- sample(392,196)
```

Linear regression

We now run a linear regression using the training data.

```
lm.fit <- lm(mpg~horsepower , data = Auto,
             subset=train)
```

Linear regression

We now use predict() to estimate the response for all 392 observations, and we use the mean() function to calculate the MSE of the 196 observations in the validation set.

Note that the -train index below selects only the observations that are not in the training set.

```
attach(Auto)
mean((mpg-predict(lm.fit,Auto))[-train]^2)
```

```
## [1] 26.14142
```

Therefore, the estimated test MSE for the linear regression fit is 26.14.

Linear regression

We can use the poly() function to estimate the test error for the polynomial and cubic regressions.

```
lm.fit2=lm(mpg~poly(horsepower ,2),data=Auto,
           subset=train)
mean((mpg-predict(lm.fit2,Auto))[-train]^2)
```

```
## [1] 19.82259
```

```
lm.fit3=lm(mpg~poly(horsepower ,3),data=Auto,
           subset=train)
mean((mpg-predict(lm.fit3,Auto))[-train]^2)
```

```
## [1] 19.78252
```

These error rates are 19.82 and 19.78, respectively. If we choose a different training set instead, then we will obtain somewhat different errors on the validation set. (Verify this on your own).

Summary of validation method

Using this split of the observations into a training set and a validation set, we find that the validation set error rates for the models with linear, quadratic, and cubic terms are 23.30, 18.90, and 19.26, respectively.

These results suggest: a model that predicts mpg using a quadratic function of horsepower performs better than a model that involves only a linear function of horsepower, and there is little evidence in favor of a model that uses a cubic function of horsepower.

LOOCV for Auto data set

The LOOCV estimate can be automatically computed for any generalized linear model using the glm() and cv.glm() functions.

Recall if we do not give the glm() a family, then it will perform linear regression.

The cv.glm() function is part of the boot library.

LOOCV for Auto data set

```
# run a linear model using glm package
library(boot)
glm.fit=glm(mpg~horsepower ,data=Auto)
cv.err=cv.glm(Auto,glm.fit)
cv.err$delta
```

```
## [1] 24.23151 24.23114
```

The cv.glm() function produces a list with several components.

The two numbers in the delta vector contain the cross-validation results. (These coorrespond to the LOOCV statistic, equation 5.1).

Here, they are both the same. (They can differ).

Our cross-validation estimate for the test error is approximately 24.23.

LOOCV for Auto data set

We can repeat this procedure for increasingly complex polynomial fits.

```
#look at polynomial fits
#up to order 5
cv.error=rep(0,5)
for (i in 1:5){
  glm.fit=glm(mpg~poly(horsepower ,i),data=Auto)
  cv.error[i]=cv.glm(Auto,glm.fit)$delta[1]
}
cv.error
```

```
## [1] 24.23151 19.24821 19.33498 19.42443 19.03321
```

We see a sharp drop in the estimated test MSE between the linear and quadratic fits, but then no clear improvement from using higher-order polynomials.

k-fold for Auto data set

- The cv.glm() function can also be used to implement k-fold CV.
- Below we use $k = 10$, a common choice for k, on the Auto data set.
- We once again set a random seed and initialize a vector in which we will store the CV errors corresponding to the polynomial fits of orders one to ten.

k-fold for Auto data set

```
set.seed(17)
# look at polynomials of order
# up to 10
cv.error.10=rep(0,10)
for (i in 1:10){
  glm.fit=glm(mpg~poly(horsepower ,i),data=Auto)
  cv.error.10[i]=cv.glm(Auto,glm.fit,K=10)$delta[1]
}
cv.error.10
```

```
##  [1] 24.20520 19.18924 19.30662 19.33799 18.87911 19.021
##  [8] 19.71201 18.95140 19.50196
```

k-fold for Auto data set

- Notice that the computation time is much shorter than that of LOOCV.
- (In principle, the computation time for LOOCV for a least squares linear model should be faster than for k-fold CV, due to the availability of the formula (5.2) for LOOCV; however, unfortunately the cv.glm() function does not make use of this formula.)
- We still see little evidence that using cubic or higher-order polynomial terms leads to lower test error than simply using a quadratic fit.

delta for CV

- We saw that the two numbers associated with delta are essentially the same when LOOCV is performed.
- When we instead perform k-fold CV, then the two numbers associated with delta differ slightly.
- The first is the standard k-fold CV estimate (equation 5.3).
- The second is a bias-corrected version.
- On this data set, the two estimates are very similar to each other.