Lecture 15: Model Selection and Validation, Cross Validation

Hao Helen Zhang

Outline

- Read: ESL 7.4, 7.10
- better understand CV

In general,

- the linear regression method like OLS has low bias (if the true model is linear) but high variance (especially if data dimension is high) – leading to poor predictions.
- Modern methods, such as LASSO and ridge regression, solve

$$\min_{\beta} \|\mathbf{y} - X\beta\|_2^2 + \lambda J(\beta).$$

They introduce some bias to reduce variance, leading to better prediction accuracy.

- J is called a penalty function.
- Regularization can help to achieve sparsity or smoothness in estimation.

Model Performance Measurements

Given the data $(\mathbf{x}_i, y_i), i = 1, \cdots, n$, where $\mathbf{x}_i$ is the input and y_i is the response, we assume

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \quad i = 1, \ldots, n.$$

We fit the model using some method, and obtain $\hat{f}$. If the model is indexed by a parameter, we denote the fitted model by $\hat{f}_\alpha$ or $\hat{f}_\lambda$.

For example,

- the OLS estimator: $\hat{f}(\mathbf{x}) = \mathbf{x}^T \hat{\beta}^{ols}$.
- the LASSO estimator $\hat{f}_\lambda(\mathbf{x}) = \mathbf{x}^T \hat{\beta}_\lambda^{lasso}$.

Bias-Variance Trade-off

As complexity increases, the model tends to

- adapt to more complicated underlying structures (*decrease in bias*)
- have increased estimation error (*increase in variance*)

In between there is an optimal model complexity, giving the minimum test error.

Role of tuning parameter α or λ:

- Tuning parameter varies the model complexity
- Find the best α to produce the minimum of test error

Model Performance Evaluation

How do we evaluate the performance of $\hat{f}$? Consider

- its generalization performance
- its prediction capability on an independent set of data, the so-called "test data"

Model assessment is extremely important, as it

- measures the quality of the final model.
- allows us to compare different methods and make an optimal choice.
- guides the choice of tuning parameters in regularization methods.

Model Selection and Assessment

Two Goals:

- Model Selection: estimate the performance of different models and choose the best model
- Model Assessment: after choosing a final model, estimate its prediction error (generalization performance) on new data.

Common Techniques

The following are commonly used to estimate the prediction error

- Validation set (data-rich situation)
- Resampling methods:
  - cross validation
  - bootstrap
- Analytical estimation criteria

Next, we will introduce their basic ideas.

See Efron (2004, JASA) for more details.

Different Types of Errors

In machine learning, we encounter different types of "errors"

- training error
- test error, prediction error
- tuning error
- cross validation error

Example: If Y is continuous, two commonly-used loss functions are

$$\begin{array}{ll} \text{squared error} & L(Y, \hat{f}(\mathbf{X})) = (Y - \hat{f}(\mathbf{X}))^2 \\ \text{absolute error} & L(Y, \hat{f}(\mathbf{X})) = |Y - \hat{f}(\mathbf{X})| \end{array}$$

Training error and Test error

- **Training error**: the average loss over the training samples

$$\text{TrainErr} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}(\mathbf{x}_i))$$

- **Prediction error**: the expected error over all the input

$$\text{PE} = E_{(\mathbf{x}, Y', \text{data})}[\frac{1}{n}\sum_{i=1}^{n} L(Y_i', \hat{f}(\mathbf{X}_i))],$$

  where $Y_i' = f(\mathbf{X}_i) + \epsilon_i', i = 1, \cdots, n$, are independent of $y_1, \cdots, y_n$. Expectation is taken w.r.t. all random quantities.
- **Test error**: Given a test data set $(\mathbf{x}_i, y_i')$ for $i = 1, \ldots, n'$,

$$\text{TestErr} = \frac{1}{n'}\sum_{i=1}^{n'} L(Y_i', \hat{f}(\mathbf{X}_i)).$$

Classification Problems: Y is discrete

If Y takes values $\{1, ..., K\}$, the common loss functions are

$$\begin{array}{ll} 0-1 \text{ loss} & L(Y, \hat{f}(\mathbf{X}) = I(Y \neq \hat{f}(\mathbf{X})) \\ \text{log-likelihood} & L(Y, \hat{p}(\mathbf{X}) = -2\sum_{k=1}^{K} I(Y = k) \log \hat{p}_k(\mathbf{X}) \\ & = -2 \log \hat{p}_Y(\mathbf{X}) \end{array}$$

- With the log-likelihood loss,
  - Training error is the sample log-likelihood

$$\text{TrainErr} = -\frac{2}{n}\sum_{i=1}^{n} \log \hat{p}_{y_i}(\mathbf{x}_i)$$

  - Log-likelihood is referred to as *cross-entropy loss* or *deviance*.
  - General densities: Poisson, gamma, exponential, log-normal

$$L(Y, \theta(\mathbf{X})) = -2 \log \Pr_{\theta(\mathbf{X})}(Y).$$

- With $0-1$ loss, the test error is *expected misclassification rate*

Training Error vs Test Error

- Training error can be easily calculated by applying the statistical learning method to the training set.
- Test error is the average error of a statistical learning method when being used to predict the response for a new observation, one that was not used in training the method.

TrainErr is not a good estimate of TestErr

- The training error rate often is quite different from the test error rate.
- TrainErr can dramatically underestimate TestErr.

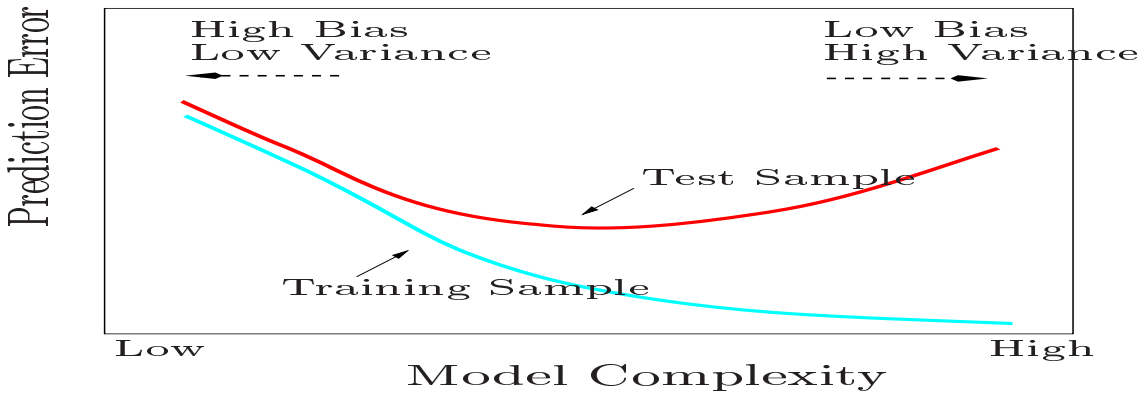

Figure 2.11: *Test and training error as a function of model complexity.*

What is Wrong with Training Error

- TrainErr is not an objective measure on the generalization performance of the method, as we build $\hat{f}$ based on the training data.
- TrainErr decreases with model complexity.
  - Training error drops to zero if the model is complex enough
  - A model with zero training error overfits the training data; over-fitted models typically generalize poorly

Question: How do we estimate TestErr accurately? We consider

- data-rich situation
- data-insufficient situation

Data-Rich Situation

Key Idea: Estimate the test error 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.

Validation-set Approach: without tuning

- Randomly divide the available set of samples into two parts: a training set and a validation or hold-out set.
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set.
- The resulting validation-set error provides an estimate of the test error. This is typically assessed using MSE in the case of a quantitative response and misclassification rate in the case of a qualitative (discrete) response.

Validation-set Approach: with tuning

Randomly divide the dataset into three parts:

- **training** set: to fit the models
- **validation** (tuning) set: to estimate the prediction error for model selection
- **test** set: to assess the generalization error of the final chosen model

The typical proportions are respectively: $50\%, 25\%, 25\%$.

Drawbacks of Validation-set Approach

- The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set.
- In the validation approach, only a subset of the observations - those that are included in the training set rather than in the validation set - are used to fit the model.
- This suggests that the validation set error may tend to overestimate the test error for the model fit on the entire data set.

Data-Insufficient Situation

How much training data is enough?

- depends on signal-to-noise ratio of the underlying function
- depends on complexity of the models
- too difficulty to give a general rule

Resampling Methods

These methods refit a model of interest to samples formed from the training set, and obtain additional information about the fitted model.

- Nonparametric methods: efficient sample reuse
- Cross-validation, bootstrap techniques
- They provide estimates of test-set prediction error, and the standard deviation and bias of our parameter estimates.

K-fold Cross Validation

The simplest and most popular way to estimate the test error.

1. Randomly split the data into K roughly equal parts
2. For each $k = 1, ..., K$
   - leave the kth portion out, and fit the model using the other $K - 1$ parts. Denote the solution by $\hat{f}^{-k}(\mathbf{x})$
   - calculate the prediction error of the $\hat{f}^{-k}(\mathbf{x})$ on the kth (left-out) portion
3. Average the errors

Define the Index function (allocating memberships)

$$\kappa : \{1, ..., n\} \to \{1, ..., K\}.$$

Then the K-fold cross-validation estimate of prediction error is

$$CV = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(\mathbf{x}_i))$$

Leave-Out-One (LOO) Cross Validation

If K=n, we have leave-out-one cross validation.

- $\kappa(i) = i$
- LOO-CV is approximately unbiased for the true prediction error
- LOO-CV can have high variance because the n "training sets" are so similar to one another

Computation:

- Computational burden is generally considerable, requiring n model fitting.
- For special models like linear smoothers, computation can be done quickly. For example: Generalized Cross Validation (GCV)

Statistical Properties of Cross Validation Error Estimates

Cross-validation error is very nearly unbiased for test error.

- The slight bias is due to that the training set in cross-validation is slightly smaller than the actual data set (e.g. for LOOCV the training set size is n 1 when there are n observed cases).
- In nearly all situations, the effect of this bias is conservative, i.e., the estimated fit is slightly biased in the direction suggesting a poorer fit. In practice, this bias is rarely a concern.

The variance of CV-error can be large.

- In practice, to reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

Choose α

Given tuning parameter α, the model fit $\hat{f}^{-k}(\mathbf{x}, \alpha)$.
The $CV(\alpha)$ curve is

$$CV(\alpha) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-\kappa(i)}(\mathbf{x}_i, \alpha))$$

- $CV(\alpha)$ provides an estimate of the test error curve
- We find the best parameter $\hat{\alpha}$ by minimizing $CV(\alpha)$.
- Final model is $f(\mathbf{x}, \hat{\alpha})$.

Performance of CV

Example: Two-class classification problem

- Linear model with best subsets regression of subset size p

In Figure 7.9, (p is tuning parameter)

- both curves picks best $\hat{p} = 10$. (The true model)
- *CV* curve is flat beyond 10.
- Standard error bars are the standard errors of the individual misclassification error rates

Standard Errors for K-fold CV

Let $CV_k(\hat{f}_\alpha)$ denote the validation errors of $\hat{f}_\alpha$ in each fold for α, or simply $CV_k(\alpha)$. Here $\alpha \in \{\alpha_1, \ldots, \alpha_m\}$, within a set of candidate values for α. Then we have

- The cross validation error

$$CV(\alpha) = \frac{1}{K} \sum_{k=1}^{K} CV_k(\alpha).$$

- The standard deviation of $CV(\alpha)$ is calculated as

$$SE(\alpha) = \sqrt{\text{Var}(CV(\alpha))}.$$

One-standard Error for K-fold CV

Let

$$\hat{\alpha} = \arg \min_{\alpha \in \{\alpha_1, \ldots, \alpha_m\}} CV(\alpha).$$

First we find $\hat{\alpha}$; then we move α in the direction of **increasing regularization** as much as can as long as

$$CV(\alpha) \leq CV(\hat{\alpha}) + SE(\hat{\alpha}).$$

- choose the most parsimonious model with error no more than one standard error above the best error.
- Idea: "All equal (up to one standard error), go for the simplest model".

In this example, one-SE rule CV would choose $p = 9$.

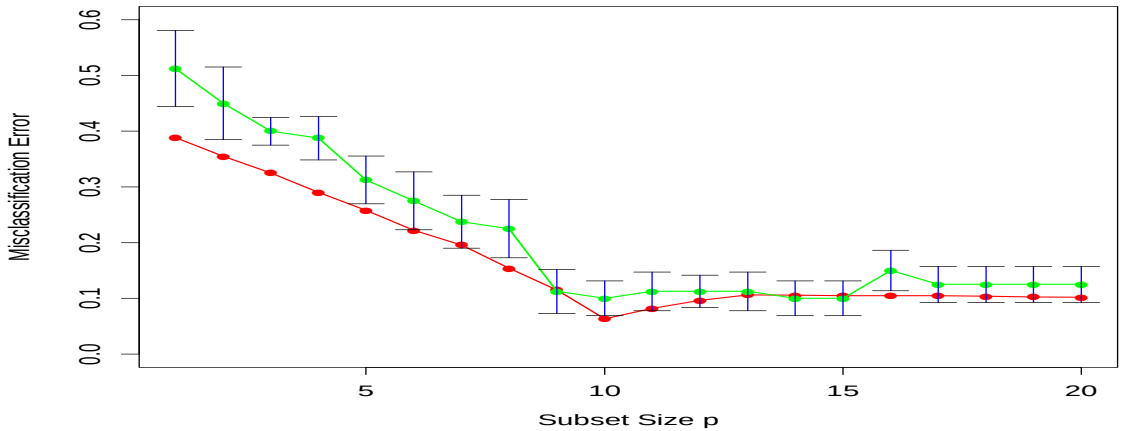

Figure 7.9: *Prediction error (red) and tenfold cross-validation curve (green) estimated from a single training set, from the scenario in the bottom right panel of Figure 7.3.*

The Wrong and Right Way

Consider a simpler classifier for microarrays:

1. Starting with 5,000 genes, find the top 200 genes having the largest correlation with the class label
2. Carry about the nearest-centroid classification using top 200 genes

How do we select the tuning parameter in the classifier?

- Way 1: apply cross-validation to step 2
- Way 2: apply cross-validation to steps 1 and 2

Which is right? – Cross validating the whole procedure.

Generalized Cross Validation

If linear fitting methods $\hat{\mathbf{y}} = S\mathbf{y}$ satisfy

$$y_i - \hat{f}^{-i}(\mathbf{x}_i) = \frac{y_i - \hat{f}(\mathbf{x}_i)}{1 - S_{ii}},$$

then we do **NOT** need to train the model n times.
Using the estimation $S_{ii} = \text{trace}(S)/n$, we have the approximate score

$$GCV = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{y_i - \hat{f}(\mathbf{x}_i)}{1 - \text{trace}(S)/n} \right]^2$$

- computational advantage
- $GCV(\alpha)$ is always smoother than $CV(\alpha)$

Application Examples

For regression problems, apply CV to

- shrinkage methods: select λ
- best subset selection: select the best model
- nonparametric methods: select the smoothing parameter

For classification methods, apply CV to

- kNN: select k
- SVM: select λ

Model-Based Methods: Information Criteria

Covariance penalties

- C_p
- AIC
- BIC
- Stein's UBR

They can be used for both *model selection* and *test error estimation*.

Generalized Information Criteria (GIC)

Information criteria:

$$\text{GIC(model)} = -2 \cdot \text{loglik} + \alpha \cdot \text{ df},$$

the degree of freedom (df) of $\hat{f}$ as the number of effective parameters of the model.

Examples: In linear regression settings, we use

$$\begin{aligned} AIC &= n \log(\text{RSS}/n) + 2 \cdot \text{df}, \\ BIC &= n \log(\text{RSS}/n) + \log(n) \cdot \text{df} \end{aligned}$$

where $\text{RSS} = \sum_{i=1}^{n} [y_i - \hat{f}(\mathbf{x}_i)]^2$ (the residual sum of squares)

Optimism of Training Error Rate

- training error:

$$\bar{\text{err}} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}(\mathbf{x}_i))$$

- test error:

$$\text{Err} = E_{(\mathbf{x}, Y, \text{data})}[L(Y, \hat{f}(\mathbf{X}))].$$

- in-sample error rate (an estimate of prediction error)

$$\text{Err}_{\text{in}} = \frac{1}{n}\sum_{i=1}^{n} E_{y^{new}} E_y L(y_i^{new}, \hat{f}(\mathbf{x}_i))$$

- Optimism

$$\text{op} = \text{Err}_{\text{in}} - E_y \bar{\text{err}}$$

Optimism

op is typically positive (Why?)

For the squared error, 0-1, and other loss function,

$$op = \frac{2}{n} \sum_{i=1}^{n} Cov(\hat{y}_i, y_i).$$

- The amount by which $\bar{err}$ under-estimates the true error depends on the strength of correlation y_i and its production.