FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
NATURWISSENSCHAFTLICHE
FAKULTÄT

Model Selection

Lecture “Mathematics of Learning” 2022/2023

Wigand Rathmann
Friedrich-Alexander-Universität Erlangen-Nürnberg

WIRTSCHAFTS
MATHEMATIK

Model Assessment and Selection

based on Elements Statistical Learning, ch. 7

After having trained some learning method, how well will it predict on independent test data?
Here, we can only give a very short overview and refer to the literature on statistical learning theory for more details.

Some definitions

target variable Y,
vector of inputs X,
prediction model $\hat{f}(X)$ that has been learned,
training set is T.
We assume to have some randomness in observing targets.
loss function for measuring errors between Y and $\hat{f}(X)$: $L(Y, \hat{f}(X))$.

Model Assessment and Selection

based on Elements Statistical Learning, ch. 7

After having trained some learning method, how well will it predict on independent test data?
Here, we can only give a very short overview and refer to the literature on statistical learning theory for more details.

Some definitions

target variable Y,
vector of inputs X,
prediction model $\hat{f}(X)$ that has been learned,
training set is T.
We assume to have some randomness in observing targets.
loss function for measuring errors between Y and $\hat{f}(X)$: $L(Y, \hat{f}(X))$.

Examples

squared error: $L(Y, \hat{f}(X)) = (Y - \hat{f}(X))^2$
absolute error: $|Y - \hat{f}(X)|$

Model Assessment and Selection

Training Error, Test Error and Expected Prediction Error

training error: average loss over training sample

$$\overline{\text{err}} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{1}$$

test error (generalization error) of model $\hat{f}$:

$$\text{Err}_T = E\left(L(Y, \hat{f}(X))|T\right) \tag{2}$$

where both X, Y drawn randomly from joint distribution (population).
Test error refers to the error for specific training set T.
expected prediction error (or *expected test error*):

$$\text{Err} = E\left[L(Y, \hat{f}(X))\right] = E\left[\text{Err}_T\right]. \tag{3}$$

Expectation averages over everything that is random, including the randomness in the training set that produced $\hat{f}$.

Model Assessment and Selection

Training Error, Test Error and Expected Prediction Error

training error: average loss over training sample

$$\overline{\text{err}} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{1}$$

test error (generalization error) of model $\hat{f}$:

$$\text{Err}_T = E\left(L(Y, \hat{f}(X))|T\right) \tag{2}$$

where both X, Y drawn randomly from joint distribution (population).
Test error refers to the error for specific training set T.
expected prediction error (or *expected test error*):

$$\text{Err} = E\left[L(Y, \hat{f}(X))\right] = E\left[\text{Err}_T\right]. \tag{3}$$

Expectation averages over everything that is random, including the randomness in the training set that produced $\hat{f}$.

Model Assesment and Selection

see picture: prediction error (light red curves) $\text{Err}_{\mathcal{T}}$ for 100 simulated training sets each of size 50. The lasso was used to produce the fits. The solid red curve is the average, and hence an estimate of Err.

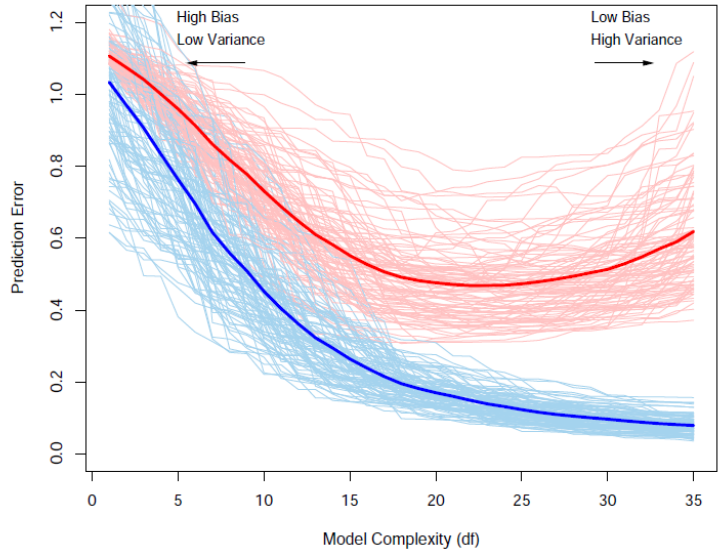


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Model Assesment and Selection

Goal: estimate $\text{Err}_{\mathcal{T}}$, however statistics can usually be done easier for Err.
How to determine expected test error of estimated model $\hat{f}$?

Model Assesment and Selection

Goal: estimate Err_T, however statistics can usually be done easier for Err.
How to determine expected test error of estimated model $\hat{f}$?

Some Observations on Training and Test Error

- 'some' intermediate model complexity gives minimum expected test error
- training error is not a good estimate of the test error, see figure.
- training error decreases with model complexity, typically dropping to zero if we increase the model complexity enough.
- However, a model with zero training error is overfit to the training data and will typically generalize poorly.

Bias and Complexity of a Model

- bias: quantification of systematic difference between expected (learned) result and actually observed result
- increasing model complexity
  $\Rightarrow$ able to adapt to more complicated underlying structures
  $\Rightarrow$ decrease in bias, but increase in variance.

Estimating Expected Error of a Model

Typically our model will have tuning parameter(s) α, thus we can write our predictions as $\hat{f}_\alpha(x)$, that varies the complexity of our model
How to determine the value of α that minimizes (average test) error?
two separate goals:

- **model selection**: estimating the performance of different models in order to choose the best one.
- **model assessment**: having chosen a final model, estimating its prediction error (generalization error) on new data.

Train - Validate - Test

data-rich situation: randomly divide dataset into a training set for model fitting, a validation set for prediction error estimate, and a test set for generalization error.

Test set: brought out only at the end of the data analysis. (if otherwise the test-set is used repeatedly, choosing the model with smallest test-set error will underestimate true test error.)

split typically 50% training, 25% validation, 25% testing:

however: one is often *not* in data-rich situation.

Bias-Variance Decomposition

- Assume $Y = f(X) + \epsilon$ with mean $E(\epsilon) = 0$, variance $\text{Var}(\epsilon) = \sigma_\epsilon^2$
- It is $\sigma_\epsilon^2 = E(Y^2) - E(Y)^2$ and thus $E(Y^2) = \sigma_\epsilon^2 + E(Y)^2$
- f is deterministic, thus $E(f) = f$
- $Y = f(X) + \epsilon$ which means $E(Y) = E(f + \epsilon) = E(f) = f$

We compute the expected prediction error of a regression fit $\hat{f}(X)$ at an input $X = x_0$.

Bias-Variance Decomposition

- Assume $Y = f(X) + \epsilon$ with mean $E(\epsilon) = 0$, variance $\text{Var}(\epsilon) = \sigma_\epsilon^2$
- It is $\sigma_\epsilon^2 = E(Y^2) - E(Y)^2$ and thus $E(Y^2) = \sigma_\epsilon^2 + E(Y)^2$
- f is deterministic, thus $E(f) = f$
- $Y = f(X) + \epsilon$ which means $E(Y) = E(f + \epsilon) = E(f) = f$

We compute the expected prediction error of a regression fit $\hat{f}(X)$ at an input $X = x_0$.

Bias-Variance Decomposition

We use the squared-error loss function and write it out step by step:

$$
\begin{aligned}
\text{Err}(x_0) &= E\left[(Y - \hat{f}(x_0))^2 | X = x_0\right] = E(Y^2 + \hat{f}^2(x_0) - 2y\hat{f}(x_0)) \\
&= E(Y^2) + E(\hat{f}^2(x_0)) - E(2y\hat{f}(x_0)) \\
&= \sigma_\epsilon^2 + E(Y)^2 + \text{Var}(\hat{f}(x_0)) + E(\hat{f})^2 - E(2y\hat{f}(x_0)) \\
&= \sigma_\epsilon^2 + E(Y)^2 + \text{Var}(\hat{f}(x_0)) + E(\hat{f})^2 - 2fE(\hat{f}(x_0)) \\
&= \sigma_\epsilon^2 + \text{Var}(\hat{f}(x_0)) + (f - E(\hat{f}))^2 \\
&= \sigma_\epsilon^2 + \text{Var}(\hat{f}(x_0)) + E(f - \hat{f})^2 \\
&= \sigma_\epsilon^2 + \text{Var}(\hat{f}(x_0)) + \text{Bias}^2(\hat{f}(x_0)) \\
&= \text{Irreducible Error} + \text{Variance} + \text{Bias}^2.
\end{aligned}
$$

Bias-Variance Decomposition

(could also use translation theorem $E((X-a)^2) = \text{Var}(X) + (E(X)-a)^2$)

Bias-Variance Decomposition

$$\text{Err}(x_0) = \text{Irreducible Error} + \text{Variance} + \text{Bias}^2. \tag{4}$$

- first term σ_ϵ^2: variance of the target around its true mean $f(x_0)$, cannot be avoided no matter how well we estimate $f(x_0)$, unless $\sigma_\epsilon^2 = 0$.
- second term $\text{Var}(\hat{f}(x_0))$: variance, expected squared deviation of $\hat{f}(x_0)$ around its mean.
- third term $E(f - \hat{f})^2$: squared bias, the amount by which the average of our estimate differs from the true mean

Typically the more complex we make the model $\hat{f}$, the lower the (squared) bias but the higher the variance.

Example

For k-nearest-neighbor regression, we have the following formula

$$\text{Err}(x_0) = E[(Y - \hat{f}_k(x_0)|X = x_0] \tag{5}$$

$$= \sigma_\epsilon^2 + \frac{\sigma_\epsilon^2}{k} + \left(f(x_0) - \frac{1}{k} \sum_{l=1}^{k} f(x_l) \right)^2 \tag{6}$$

- We have assumed that training inputs x_i are fixed, randomness arises from the y_i (i.e., if we draw again the observations, results y_i may be different, because there is some randomness). We will need this later.
- small k: estimate $\bar{f}_k(x)$ can potentially adapt itself better to the underlying $f(x)$
- increasing k: the bias (squared difference between $f(x_0)$ and average of $f(x)$ at the k-nearest neighbors will typically increase, while variance decreases.

Optimism of Training Error Rate

We will rewrite now in more detail definitions that we have already seen.
given (fixed) training set: $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$

Extra-Sample Error

test error (generalization error) of model $\hat{f}$:

$$\text{Err}_T = E_{X^0, Y^0}(L(Y^0, \hat{f}(X^0))|T) \tag{7}$$

point (X^0, Y^0) is new test data point drawn from joint distribution F of data.

expected prediction error

expected prediction error (or expected test error):

$$\text{Err} = E_T E_{X^0, Y^0}[L(Y^0, \hat{f}(X^0))|T] \tag{8}$$

most methods effectively estimate expected error rather than E_T.

Optimism of Training Error Rate

Training error

$$\overline{\text{err}} = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{9}$$

is typically smaller than true error Err_T, because *same* data is used to fit the model ($\Rightarrow$ small error) *and* to assess its error.
reason: Err_T is *extra-sample error* using (also) new data points, whereas $\overline{\text{err}}$ is an *in-sample error*.

in-sample error

$$\overline{\text{Err}}_{in} = \frac{1}{N} \sum_{i=1}^{N} E_{Y^0} \left[L(Y_i^0, \hat{f}(x_i) | T) \right] \tag{10}$$

Y^0 indicates that we observe N new responses from training set for the x_i (recall we have assumed there is some randomness in drawing observations...).

optimism

We define the *optimism* as difference between in-sample and training error:

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

This is typically positive, see arguments above.
average optimism: expectation of the optimism over training sets

$$\omega = E_y(\text{op}).$$

We can usually estimate only the expected error ω rather than op, in the same way that we can estimate the expected error Err rather than the conditional error Err_T.
For squared error, 0–1, and other loss functions, one can show quite generally that

$$\omega = \frac{2}{N}\sum_{i=1}^{N}\text{Cov}(\hat{y}_i, y_i)$$

where Cov indicates covariance.

Thus the amount by which $\overline{\text{err}}$ underestimates the true error depends on how strongly y_i affects its own prediction. The more difficult it is to fit the data, the greater $\text{Cov}(\hat{y}_i, y_i)$ will be, thereby increasing the optimism.
In summary, we have the important relation

$$E_y(\text{Err}_{in}) = E_y(\overline{\text{err}}) + \frac{2}{N}\sum_{i=1}^{N}\text{Cov}(\hat{y}_i, y_i).$$

This simplifies if $\hat{y}_i$ is obtained by a linear fit with d inputs or basis functions.

Example

$$\sum_{i=1}^{N}\text{Cov}(\hat{y}_i, y_i) = d\sigma_\epsilon^2$$

for the additive error model $Y = f(X) + \epsilon$, and so

$$E_y(\text{Err}_{in}) = E_y(\overline{\text{err}}) + 2\frac{d}{N}\sigma_\epsilon^2$$

The optimism (in this example $= 2\frac{d}{N}\sigma_\epsilon^2$) increases linearly with the number d of inputs or basis functions we use, but decreases as the training sample size increases.

Some Estimate of In-Sample Prediction Error

Basics on Akaike information criterion

uses log-likelihood loss function, estimates error relative to other models.
Trades off model complexity and model quality (error)
The Akaike information criterion is an estimate of Err_{in} when a log-likelihood loss function (i.e., sums of ln of probabilities) is used. It holds asymptotically as $N \to \infty$. We skip its exact derivation here.
To use it for model selection, we choose the model giving smallest AIC over the set of models considered.

test error estimates via AIC

Given a set of models $f_\alpha(x)$:

- tuning parameter α
- $\text{err}(\alpha)$ training error
- $d(\alpha)$ number of parameters

We define

$$AIC(\alpha) = \text{err}(\alpha) + 2\frac{d(\alpha)}{N}\hat{\sigma}_\epsilon.$$

Best value

AIC(α) estimates test error curve. We determine the tuning parameter $\hat{\alpha}$ that minimizes it. Final chosen model is $f_{\hat{\alpha}}(x)$.

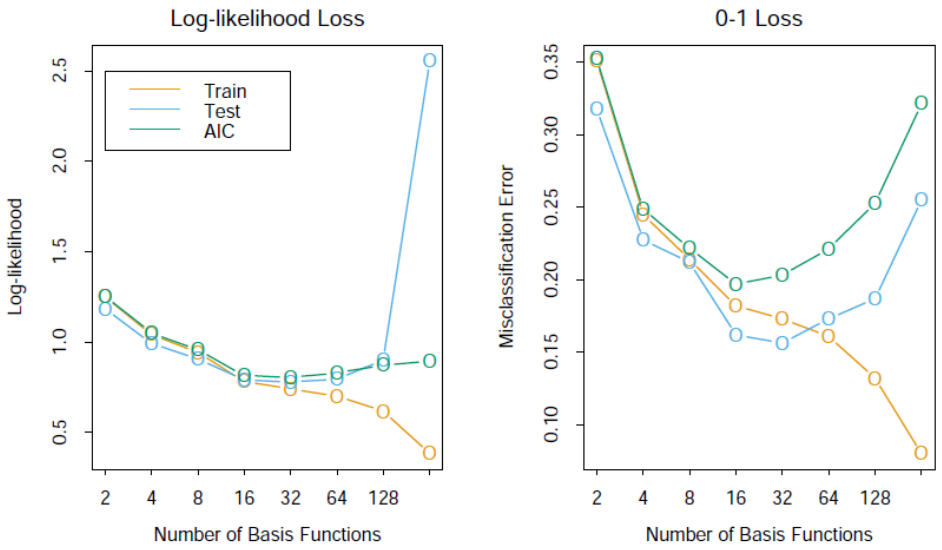


FIGURE 7.4. *AIC used for model selection for the phoneme recognition example of Section 5.2.3. The logistic regression coefficient function* $\beta(f) = \sum_{m=1}^{M} h_m(f)\theta_m$ *is modeled as an expansion in M spline basis functions. In the left panel we see the AIC statistic used to estimate* $\mathrm{Err}_{\mathrm{in}}$ *using log-likelihood loss. Included is an estimate of* Err *based on an independent test sample. It does well except for the extremely over-parametrized case (*$M = 256$ *parameters for* $N = 1000$ *observations). In the right panel the same is done for 0–1 loss. Although the AIC formula does not strictly apply here, it does a reasonable job in this case.*

Vapnik–Chervonenkis Dimension

How to define complexity of a model?

Suppose we have a class of functions $\{f_\alpha(x)\}, x \in \mathbb{R}^p$.
Let first consider indicator function, i.e., takes values 0 or 1, $I(\alpha_0 + \alpha_1^\top x > 0)$. We say here: complexity of f is the number of parameters $p + 1$.
How about $f_\alpha(x) = I(\sin(\alpha x))$ where α is some real number?
This is a very wiggly function that gets rougher as frequency α increases, but has only one parameter. It should not have less 'complexity' than indicator function. How to define its complexity?

Vapnik–Chervonenkis Dimension

Vapnik–Chervonenkis Dimension

The VC dimension of a class $f_\alpha(x)$ is defined as largest number of points (in some configuration) that can be 'shattered' by members of $f_\alpha(x)$.
A set of points is said to be shattered by a class of functions if, no matter how we assign a binary label to each point, a member of the class can perfectly separate them.
VC dimension of linear indicator functions in the plane is 3 but not 4, since no four points can be shattered by a set of lines.
A linear indicator function in p dimensions has VC dimension $p + 1$, which is also the number of free parameters.

Vapnik–Chervonenkis Dimension

VC Dimension

It can be shown that the family $\sin(\alpha x)$ has infinite VC dimension. By appropriate choice of α, any set of points can be shattered by this class.
The VC dimension of a class of real-valued functions $\{g(x, \alpha)\}$ is defined to be the VC dimension of the indicator class $\{I(g(x, \alpha) - \beta > 0)\}$, where β takes values over the range of g.
Use VC dimension for estimating (extra-sample) prediction error.
show results about the optimism of the training.

Example Result (Vapnik–Chervonenkis Dimension)

If we fit N training points using a class of functions with VC dimension h, then with probability at least $1 - \eta$ over training sets it is:

$$\mathrm{Err}_T \leq \overline{\mathrm{err}} + \frac{\epsilon}{2}\left(1 + \sqrt{1 + \frac{4\mathit{err}}{\epsilon}}\right) \text{ (binary classification)}$$

$$\mathrm{Err}_T \leq \frac{\overline{\mathrm{err}}}{1 - c\sqrt{\epsilon}_+} \text{ (regression)}$$

where $\epsilon = a_1 \frac{h\log(a_2 N/h)+1-\log(\eta/4)}{N}$
and $0 < a_1 \leq 4, 0 < a_2 \leq 2$
These bounds hold simultaneously for all members $f(x, \alpha)$. Good choices $c = 1$ and (for regression): $a_1 = a_2 = 1$.
The bounds suggest that the optimism increases with h and decreases with N in qualitative agreement with the AIC correction $\frac{d}{N}$.
However, these results here are stronger: rather than giving the expected optimism for each fixed function $f(x, \alpha)$, they give probabilistic upper bounds for all functions $f(x, \alpha)$, and hence allow for searching over the class.

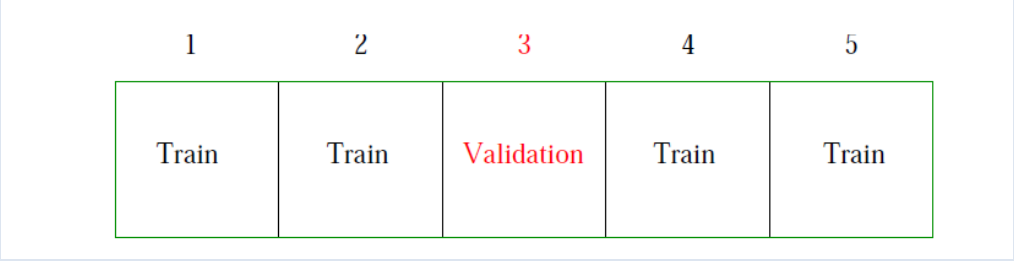
Some Discussion

We note that these upper bounds are often loose, but that doesn't rule them out as good criteria for model selection, where the relative (not absolute) size of the test error is important. The main drawback of this approach is the difficulty in calculating the VC dimension. Often only a crude upper bound for VC dimension is obtainable.
This is a research field on its own.

Cross-Validation

Task: estimate the expected extra-sample error $\text{Err} = E[L(Y, \hat{f}(X))]$ (i.e., average generalization error) when learned function $f(\hat{X})$ is used for an independent test sample from the joint distribution of X and Y.

K-Fold Cross Validation



For k-th part: fit model to the other $K - 1$ parts of the data, calculate prediction error of fitted model when predicting the k-th part of the data.
For $k = 1, 2, \ldots, K$, combine the K estimates of prediction error.
Let $\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\}$ indicate partition to which observation i is allocated. $\hat{f}^{-k}(x)$ is fitted function, computed with the k-th part of the data removed.

Cross-Validation

Cross-validation Estimate

for prediction error:

$$CV(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i)).$$

Typical choices: $K = 5$ or 10.
$K = N$: leave-one-out cross-validation.
$\kappa(i) = i$. For the i-th observation, fit is computed using all data except the i-th.
Given a set of models $f(x, \alpha)$ indexed by a tuning parameter α, denote by $\hat{f}^{-k}(x, \alpha)$ the α-th function fit where k-th data part removed. Define

$$CV(\hat{f}, \alpha) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha)).$$

$CV(\hat{f}, \alpha)$ is estimate of the test error curve. Find tuning parameter $\hat{\alpha}$ that minimizes it, then fit $f(x, \hat{\alpha})$, to all data.

Practical Discussion: Good Values for K?

Cross-validation estimates expected error Err.

$K = N$:

- cross-validation estimator approximately unbiased for true (expected) prediction error
- can have high variance as N 'training sets' are similar
- high computational burden (N times learning method)

With $K = 5$ say, cross-validation has lower variance, but bias could be a problem hypothetical 'learning curve' see picture.

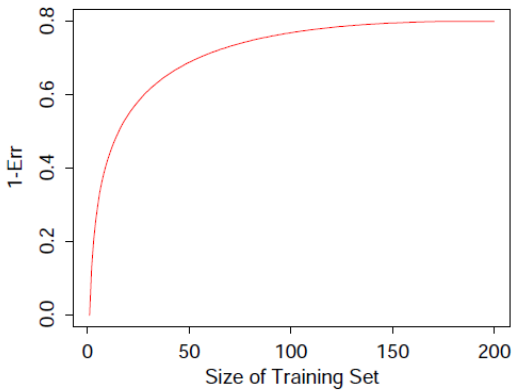


FIGURE 7.8. *Hypothetical learning curve for a classifier on a given task: a plot of* $1 - \text{Err}$ *versus the size of the training set* N. *With a dataset of* 200 *observations,* 5*-fold cross-validation would use training sets of size* 160, *which would behave much like the full set. However, with a dataset of* 50 *observations fivefold cross-validation would use training sets of size* 40, *and this would result in a considerable overestimate of prediction error.*

Recommendation

If learning curve has considerable slope at training set size, five- or tenfold cross-validation will overestimate true prediction error. Leave-one-out cross-validation has low bias but can have high variance.
Overall, five- or tenfold cross-validation are recommended as a good compromise.

Bootstrap Method

as another tool to estimate expected prediction error Err

Let training set $Z = (z_1, z_2, \ldots, z_N)$ where $z_i = (x_i, y_i)$.

- B times randomly draw datasets with replacement from the training data, each of same size as original set,
- refit model to each set

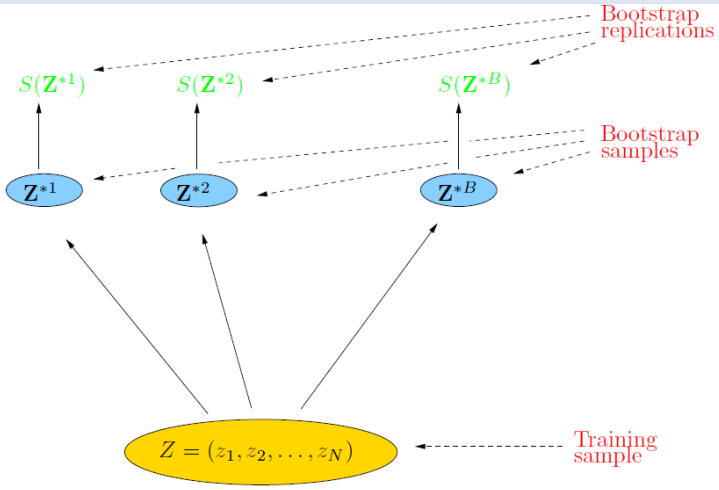


FIGURE 7.12. *Schematic of the bootstrap process. We wish to assess the statistical accuracy of a quantity $S(\mathbf{Z})$ computed from our dataset. B training sets $\mathbf{Z}^{*b}$, $b = 1, \ldots, B$ each of size N are drawn with replacement from the original dataset. The quantity of interest $S(\mathbf{Z})$ is computed from each bootstrap training set, and the values $S(\mathbf{Z}^{*1}), \ldots, S(\mathbf{Z}^{*B})$ are used to assess the statistical accuracy of $S(\mathbf{Z})$.*

Details on Bootstrap

$S(Z)$ is any quantity computed from the data Z. From the bootstrap sampling we can estimate any aspect of the distribution of $S(Z)$, for example, its variance,

$$\widehat{\text{Var}}[S(Z)] = \frac{1}{B-1}\sum_{b=1}^{B}(S(Z^{*b}) - \bar{S}^{*})^2,$$

where $\bar{S}^{*} = \sum_b \frac{S(Z^{*b})}{B}$.
$\widehat{\text{Var}}[S(Z)]$ can be thought of as a Monte-Carlo estimate of the variance of $S(Z)$ under sampling from the empirical distribution function for $(z_1, z_2, \ldots, z_N)$.

bootstrap for estimating prediction error

fit model on a set of bootstrap samples. If $\hat{f}^{*b}(x_i)$ is predicted value at x_i, from the model fitted to the b-th dataset, estimate would be

$$\text{Err}_{\text{boot}} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L(y_i, \hat{f}^{*b}(x_i)).$$

However, this does not provide a good estimate in general, because bootstrap datasets act as the training samples, while the original training set acts as test sample. This overlap can make overfit predictions look unrealistically good (recall cross-validation uses non-overlapping data for training and test) samples.

Enhancements of Bootstrap

A better bootstrap estimate can be obtained similar as cross-validation. For each observation, we only keep track of predictions from bootstrap samples not containing that observation. The leave-one-out bootstrap estimate of prediction error is defined by

$$\widehat{\text{Err}}^{1} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(y_i, \hat{f}^{*b}(x_i))$$

C^{-i} is the set of indices of the bootstrap samples b that do not contain i, and $|C^{-i}|$ is number of such samples.
The leave-one out bootstrap solves the overfitting problem, but has the training-set-size bias mentioned in the discussion of cross-validation.
Enhancements are possible, that we however will not discuss here further.