An Introduction to the Bootstrap

BRADLEY EFRON
Department of Statistics
Stanford University

and

ROBERT J. TIBSHIRANI
Department of Preventative Medicine and Biostatistics
and Department of Statistics, University of Toronto

CHAPMAN & HALL
New York • London

First published in 1993 by
Chapman & Hall
29 West 35th Street
New York, NY 10001-2299

Published in Great Britain by
Chapman & Hall
2-6 Boundary Row
London SE1 8HN

Printed in the United States of America

Library of Congress Cataloging-in-Publication Data

Efron, Bradley.
An introduction to the bootstrap / Brad Efron, Rob Tibshirani.
p. cm.
Includes bibliographical references.
ISBN 0-412-04231-2
1. Bootstrap (Statistics) I. Tibshirani, Robert. II. Title.
QA276.8.E3745 1993
519.5'44—dc20 93-4489
CIP

British Library Cataloguing in Publication Data also available.

This book was typeset by the authors using a PostScript (Adobe Systems Inc.) based phototypesetter (Linotronic 300P). The figures were generated in PostScript using the S data analysis language (Becker *et. al.* 1988), Aldus Freehand (Aldus Corporation) and Mathematica (Wolfram Research Inc.). They were directly incorporated into the typeset document. The text was formatted using the LATEX language (Lamport, 1986), a version of TEX (Knuth, 1984).

TO

CHERYL, CHARLIE, RYAN AND JULIE

AND TO THE MEMORY OF

RUPERT G. MILLER, JR.

Contents

analysis and graphics. Our language of choice (at present) is "S" (or "S-PLUS"), and a number of S programs appear in the Appendix. Most of these programs could be easily translated into other languages such as Gauss, Lisp-Stat, or Matlab. Details on the availability of S and S-PLUS are given in the Appendix.

1.3 Some of the notation used in the book

Lower case bold letters such as $\mathbf{x}$ refer to vectors, that is, $\mathbf{x} = (x_1, x_2, \ldots x_n)$. Matrices are denoted by upper case bold letters such as $\mathbf{X}$, while a plain uppercase letter like X refers to a random variable. The transpose of a vector is written as $\mathbf{x}^T$. A superscript "*" indicates a bootstrap random variable: for example, $\mathbf{x}^*$ indicates a bootstrap data set generated from a data set $\mathbf{x}$. Parameters are denoted by Greek letters such as θ. A hat on a letter indicates an estimate, such as $\hat{\theta}$. The letters F and G refer to populations. In Chapter 21 the same symbols are used for the cumulative distribution function of a population. I_C is the indicator function equal to 1 if condition C is true and 0 otherwise. For example, $I_{\{x<2\}} = 1$ if $x < 2$ and 0 otherwise. The notation $\mathrm{tr}(A)$ refers to the trace of the matrix A, that is, the sum of the diagonal elements. The derivatives of a function $g(x)$ are denoted by $g'(x), g''(x)$ and so on.

The notation

$$F \rightarrow (x_1, x_2, \ldots x_n)$$

indicates an independent and identically distributed sample drawn from F. Equivalently, we also write $x_i \overset{\text{i.i.d.}}{\sim} F$ for $i = 1, 2, \ldots n$.

Notation such as $\#\{x_i > 3\}$ means the number of x_is greater than 3. $\log x$ refers to the natural logarithm of x.

CHAPTER 17

Cross-validation and other estimates of prediction error

17.1 Introduction

In our discussion so far we have focused on a number of measures of statistical accuracy: standard errors, biases, and confidence intervals. All of these are measures of accuracy for parameters of a model. Prediction error is a different quantity that measures how well a model predicts the response value of a future observation. It is often used for model selection, since it is sensible to choose a model that has the lowest prediction error among a set of candidates.

Cross-validation is a standard tool for estimating prediction error. It is an old idea (predating the bootstrap) that has enjoyed a comeback in recent years with the increase in available computing power and speed. In this chapter we discuss cross-validation, the bootstrap and some other closely related techniques for estimation of prediction error.

In regression models, prediction error refers to the expected squared difference between a future response and its prediction from the model:

$$\text{PE} = \text{E}(y - \hat{y})^2. \tag{17.1}$$

The expectation refers to repeated sampling from the true population. Prediction error also arises in the *classification* problem, where the response falls into one of k unordered classes. For example, the possible responses might be Republican, Democrat, or Independent in a political survey. In classification problems prediction error is commonly defined as the probability of an incorrect classification

$$\text{PE} = \text{Prob}(\hat{y} \neq y), \tag{17.2}$$

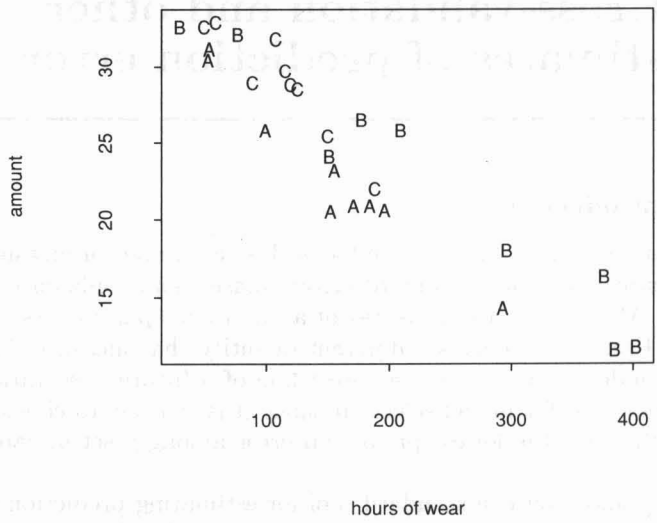

Figure 17.1. *Hormone data. Plot shows the amount of hormone remaining for a device versus the hours of wear. The symbol represents the lot number.*

also called the *misclassification rate*. The methods described in this chapter apply to both definitions of prediction error, and also to others. We begin with a intuitive description of the techniques, and then give a more detailed account in Section 17.6.2.

17.2 Example: hormone data

Let's look again at the hormone data example of chapter 9. Figure 17.1 redisplays the data for convenience. Recall that the response variable y_i is the amount of anti- inflammatory hormone remaining after z_i hours of wear, in 3 lots A, B, and C indicated by the plotting symbol in the figure. In Chapter 9 we fit regres-

sion lines to the data in each lot, with different intercepts but a common slope. The estimates are given in Table 9.3 on page 110.

Here we consider two questions: 1) "How well will the model predict the amount of hormone remaining for a new device?", and 2) "Does this model predict better (or worse) than a single regression line?" To answer the first question, we could look at the average residual squared error for all $n = 27$ responses,

$$\text{RSE}/n = \sum_{1}^{n}(y_i - \hat{y}_i)^2/n = 2.20, \tag{17.3}$$

but this will tend to be too "optimistic"; that is to say, it will probably underestimate the true prediction error. The reason is that we are using the same data to assess the model as were used to fit it, using parameter estimates that are fine-tuned to our particular data set. In other words the *test sample* is the same as the original sample, sometimes called the *training sample*. Estimates of prediction error obtained in this way are aptly called "apparent error" estimates.

A familiar method for improving on (17.3) is to divide by $n - p$ instead of n, where p is the number of predictor variables. This gives the usual unbiased estimate of residual variance $\hat{\sigma}^2 = \sum(y_i - \hat{y}_i)^2/(n-p)$. We will see that bigger corrections are necessary for the prediction problem.

17.3 Cross-validation

In order to get a more realistic estimate of prediction error, we would like to have a test sample that is separate from our training sample. Ideally this would come in the form of some new data from the same population that produced our original sample. In our example this would be hours of wear and hormone amount for some additional devices, say m of them. If we had these new data, say $(z_1^0, y_1^0), \ldots (z_m^0, y_m^0)$, we would work out the predicted values $\hat{y}_i^0$ from (9.3)

$$\hat{y}_i^0 = \hat{\beta}_j + \hat{\beta}_1 z_i^0 \tag{17.4}$$

(where $j = A, B$, or C depending on the lot), and compute the average prediction sum of squares

$$\sum_{1}^{m}(y_i^0 - \hat{y}_i^0)^2/m. \tag{17.5}$$

Algorithm 17.1

K-fold cross-validation

1. Split the data into K roughly equal-sized parts.
2. For the kth part, fit the model to the other $K-1$ parts of the data, and calculate the prediction error of the fitted model when predicting the kth part of the data.
3. Do the above for $k = 1, 2, \ldots K$ and combine the K estimates of prediction error.

This quantity estimates how far, on the average, our prediction $\hat{y}_i^0$ differs from the actual value y_i^0.

Usually, additional data are not often available, for reasons of logistics or cost. To get around this, cross-validation uses part of the available data to fit the model, and a different part to test it. With large amounts of data, a common practice is to split the data into two equal parts. With smaller data sets like the hormone data, "K-fold" cross-validation makes more efficient use of the available information. The procedure is shown in Algorithm 17.1.

Here is K-fold cross-validation in more detail. Suppose we split the data into K parts. Let $k(i)$ be the part containing observation i. Denote by $\hat{y}_i^{-k(i)}$ the fitted value for observation i, computed with the $k(i)$th part of the data removed. Then the cross-validation estimate of prediction error is

$$\text{CV} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i^{-k(i)})^2. \tag{17.6}$$

Often we choose $k = n$, resulting in "leave-one-out" cross-validation. For each observation i, we refit the model leaving that observation out of the data, and then compute the predicted value for the ith observation, denoted by $\hat{y}_i^{-i}$. We do this for each observation and then compute the average cross-validation sum of squares $\text{CV} = \sum(y_i - \hat{y}_i^{-i})^2/n$.

We applied leave-one-out cross-validation to the hormone data: the value of CV turned out to be 3.09. By comparison, the average residual squared error (17.3) is 2.20 and so it underestimates the prediction error by about 29%. Figure 17.2 shows the usual residual

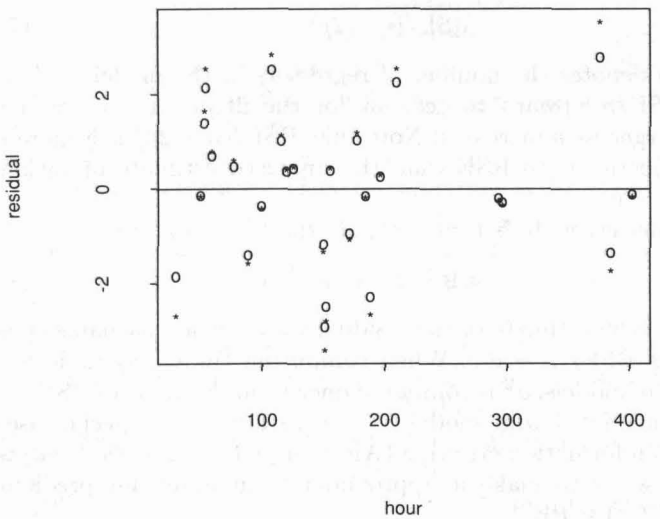

Figure 17.2. *Plot of residuals (circles) and cross-validated residuals (stars) for hormone data.*

$y_i - \hat{y}_i$ (circles) and the cross-validated residual $y_i - \hat{y}_i^{-i}$ (stars). Notice how the cross-validated residual is equal to or larger (in absolute value) than the usual residual for every case. (This turns out to be true in some generality: see Problems 17.1 and 18.1.)

We can look further at the breakdown of the CV by lot: the average values are 2.09, 4.76 and 2.43 for lots A, B and C, respectively. Hence the amounts for devices in lot B are more difficult to predict than those in lots A and C.

Cross-validation, as just described, requires refitting the complete model n times. In general this is unavoidable, but for least-squares fitting a handy shortcut is available (Problem 17.1).

17.4 C_p and other estimates of prediction error

There are other ways to estimate prediction error, and all are based on adjustments to the residual squared error RSE. The last part of this chapter describes a bootstrap approach. A simple analytic measure is the adjusted residual squared error

$$\text{RSE}/(n-2p) \tag{17.7}$$

where p denotes the number of regressors in the model. This adjusts RSE/n upward to account for the fitting, the adjustment being larger as p increases. Note that RSE/$(n-2p)$ is a more severe adjustment to RSE than the unbiased estimate of variance RSE/$(n-p)$.

Another estimate is (one form of) the "C_p" statistic

$$C_p = \text{RSE}/n + 2p\hat{\sigma}^2/n. \tag{17.8}$$

Here $\hat{\sigma}^2$ is an estimate of the residual variance; a reasonable choice for $\hat{\sigma}^2$ is RSE/$(n-p)$. (When computing the C_p statistic for a number of models, $\hat{\sigma}^2$ is computed once from the value of RSE/$(n-p)$ for some fixed large model.) The C_p statistic is a special case of Akaike's information criterion (AIC) for general models. It adjusts RSE/n so as to make it approximately unbiased for prediction error: $\text{E}(C_p) \approx \text{PE}$.

Implicitly these corrections account for the fact that the same data is being used to fit the model and to assess it through the residual squared error. The "p" in the denominator of the adjusted RSE and the second term of C_p are penalties to account for the amount of fitting. A simple argument shows that the adjusted residual squared error and C_p statistic are equivalent to a first order of approximation (Problem 17.4.)

Similar to C_p is Schwartz's criterion, or the BIC (Bayesian Information Criterion)

$$\text{BIC} = \text{RSE}/n + \log n \cdot p\hat{\sigma}^2/n \tag{17.9}$$

BIC replaces the "2" in C_p with $\log n$ and hence applies a more severe penalty than C_p, as long as $n > e^2$. As a result, when used for model comparisons, BIC will tend to favor more parsimonious models than C_p. One can show that BIC is a consistent criterion in the sense that it chooses the correct model as $n \to \infty$. This is not the case for the adjusted RSE or C_p.

In the hormone example, RSE $= 59.27$, $\hat{\sigma}^2 = 2.58$ and $p = 4$ and

hence $\text{RSE}/(n-2p) = 3.12$, $C_p = 2.96$, BIC $= 3.45$, as compared to the value of 3.09 for CV.

Why bother with cross-validation when simpler alternatives are available? The main reason is that for fitting problems more complicated than least squares, the number of parameters "p" is not known. The adjusted residual squared error, C_p and BIC statistics require knowledge of p, while cross-validation does not. Just like the bootstrap, cross-validation tends to give similar answers as standard methods in simple problems and its real power stems from its applicability in more complex situations. An example involving a classification tree is given below.

A second advantage of cross-validation is its robustness. The C_p and BIC statistics require a roughly correct working model to obtain the estimate $\hat{\sigma}^2$. Cross-validation does not require this and will work well even if the models being assessed are far from correct.

Finally, let's answer the second question raised above, regarding a comparison of the common slope, separate intercept model to a simpler model that specifies one common regression line for all lots. In the same manner as described above, we can compute the cross-validation sum of squares for the single regression line model. This value is 5.89 which is quite a bit larger than the value 3.27 for the model that allows a different intercept for each lot. This is not surprising given the statistically significant differences among the intercepts in Table 9.3. But cross-validation is useful because it gives a quantitative measure of the price the investigator would pay if he does not adjust for the lot number of a device.

17.5 Example: classification trees

For an example that illustrates the real power of cross-validation, let's switch gears and discuss a modern statistical procedure called "classification trees." In an experiment designed to provide information about the causes of duodenal ulcers (Giampaolo *et al.* 1988), a sample of 745 rats were each administered one of 56 model alkyl nucleophiles. Each rat was later autopsied for the development of duodenal ulcer and the outcome was classified as 1, 2 or 3 in increasing order of severity. There were 535 class 1, 90 class 2 and 120 class 3 outcomes. Sixty-seven characteristics of these compounds were measured, and the objective of the analysis was to ascertain which of the characteristics were associated with the development of duodenal ulcers.

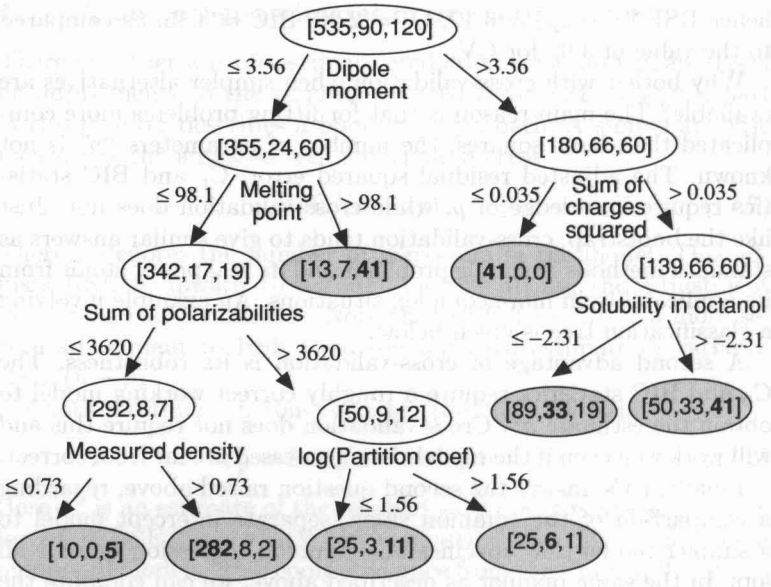

Figure 17.3. *CART tree. Classification tree from the CART analysis of data on duodenal ulcers. At each node of the tree a question is asked, and data points for which the answer is "yes" are assigned to the left branch and the others to the right branch. The shaded regions are the terminal nodes, or leaves, of the tree. The numbers in square brackets are the number of observations in each of the three classes present at each node. The bold number indicates the predicted class for the node. In this particular example, five penalty points are charged for misclassifying observations in true class 2 or 3, and one penalty point is charged for misclassifying observations in class 1. The predicted class is the one resulting in the fewest number of penalty points.*

The CART method (for Classification and Regression Trees) of Breiman, Friedman, Olshen and Stone (1984) is a computer-intensive approach to this problem that has become popular in scientific circles. When applied to these data, CART produced the classification tree shown in Figure 17.3.

At each node of the tree a yes-no question is asked, and data points for which the answer is "yes" are assigned to the left branch and the others to the right branch. The leaves of the tree shown in Figure 17.3 are called "terminal nodes." Each observation is

assigned to one of the terminal nodes based on the answers to the questions. For example a rat that received a compound with Dipole moment ≤ 3.56 and melting point > 98.1 would go left then right and end up in the terminal node marked "[13, 7, 41]." Triplets of numbers such as "[13, 7, 41]" below each terminal node number indicate the membership at that node, that is, there are 13 class 1, 7 class 2 and 41 class 3 observations at this terminal node.

Before discussing how the CART procedure built this tree, let's look at how it is used for classification. Each terminal node is assigned a class (1,2 or 3). The most obvious way to assign classes to the terminal nodes would be to use a majority rule and assign the class that is most numerous in the node. Using a majority rule, the node marked "[13,7,41]" would be assigned to class 3 and all of the other terminal nodes would be assigned to class 1. In this study, however, the investigators decided that it would be five times worse to misclassify an animal that actually had a severe ulcer or moderate ulcer than one with a milder ulcer. Hence, five penalty points were charged for misclassifying observations in true class 2 or 3, and one penalty point was charged for misclassifying observations in class 1. The predicted class is the one resulting in the fewest number of penalty points. In Figure 17.3 the predicted class is in boldface at each terminal node; for example, the node at the bottom left marked "[10,0,**5**]" has the "5" in boldface and hence is a class 3 node.

We can summarize the tree as follows. The top ("root") node was split on dipole moment. A high dipole moment indicates the presence of electronegative groups. This split separates the class 1 and 2 compounds: the ratio of class 2 to class 1 in the right split, 66/190, is more than 5 times as large as the ratio 24/355 in the left split. However, the class 3 compounds are divided equally, 60 on each side of the split. If in addition the sum of squared atomic charges is low, then CART finds that all compounds are class 1. Hence, ionization is a major determinant of biologic action in compounds with high dipole moments. Moving further down the right side of the tree, the solubility in octanol then (partially) separates class 3 from class 2 compounds. High octanol solubility probably reflects the ability to cross membranes and to enter the central nervous system.

On the left side of the root node, compounds with low dipole moment and high melting point were found to be class 3 severe. Compounds at this terminal node are related to cysteamine. Com-

pounds with low melting points and high polarizability, all thiols in this study, were classified as class 2 or 3 with the partition coefficient separating these two classes. Of those chemicals with low polarizability, those of high density are class 1. These chemicals have high molecular weight and volume, and this terminal node contains the highest number of observations. The low density side of the split are all short chain amines.

In the terminology mentioned earlier, the data set of 745 observations is called the training sample. It is easy to work out the misclassification rate for each class when the tree of Figure 17.3 is applied to the training sample. Looking at the terminal nodes that predict classes 2 or 3, the number of errors for class 1 is $13 + 89 + 50 + 10 + 25 + 25 = 212$, so the apparent misclassification rate for class 1 is 212/535=39.6%. Similarly, the apparent misclassification rates for classes 2 and 3 are 56.7% and 18.3%. "Apparent" is an important qualifier here, since misclassification rates in the training sample can be badly biased downward, for the same reason that the residual squared error is overly optimistic in regression.

How does CART build a tree like that in Figure 17.3? CART is a fully automatic procedure that chooses the splitting variables and splitting points that best discriminate between the outcome classes. For example, "Dipole moment$\leq$ 3.56" is the split that was determined to best separate the data with respect to the outcome classes. CART chose both the splitting variable "Dipole moment" and the splitting value 3.56. Having found the first splitting rule, new splitting rules are selected for each of the two resulting groups, and this process is repeated.

Instead of stopping when the tree is some reasonable size, CART uses a more effective approach: a large tree is constructed and then pruned from the bottom. This latter approach is more effective in discovering interactions that involve several variables.

This brings up an important question: how large should the tree be? If we were to build a very large tree with only one observation in each terminal node, then the apparent misclassification rate would be 0%. However, this tree would probably do a poor job predicting the outcomes for a new sample. The reason is that the tree would be geared to the training sample; statistically speaking it is "overfit."

The best-sized tree would be the one that had the lowest misclassification rate for some new data. Thus if we had a second data set available (a test sample), we could apply trees of various sizes

to it and then choose the one with lowest misclassification rate.

Of course in most situations we do not have extra data to work with, and this is where cross-validation comes in handy. Leave-one-out cross-validation doesn't work well here, because the resulting trees are not different enough from the original tree. Experience shows that it is much better to divide the data up into 10 groups of equal size, building a tree on 90% of the data, and then assessing its misclassification rate on the remaining 10% of the data. This is done for each of the 10 groups in turn, and the total misclassification rate is computed over the 10 runs. The best tree size is determined to be that tree size giving lowest misclassification rate. This is the size used in constructing the final tree from all of the data. The crucial feature of cross-validation is the separation of data for building and assessing the trees: each one-tenth of the data is acting as a test sample for the other 9 tenths. The precise details of the tree selection process are given in Problem 17.9.

The process of cross-validation not only provides an estimate of the best tree size, it also gives a realistic estimate of the misclassification rate of the final tree. The apparent error rates computed above are often unrealistically low because the training sample is used both for building and assessing the tree. For the tree of Figure 17.3, the cross-validated misclassification rates were about 10% higher than the apparent error rates. It is the cross-validated rates that provide an accurate assessment of how effective the tree will be in classifying a new sample.

17.6 Bootstrap estimates of prediction error

17.6.1 Overview

In the next two sections we investigate how the bootstrap can be used to estimate prediction error. A precise formulation will require some notation. Before jumping into that, we will convey the main ideas. The simplest bootstrap approach generates B bootstrap samples, estimates the model on each, and then applies each fitted model to the *original sample* to give B estimates of prediction error. The overall estimate of prediction error is the average of these B estimates. As an example, the left hand column of Table 17.1 shows 10 estimates of prediction error ("err") from 10 bootstrap samples, for the hormone data example described in Section 17.2. Their average is 2.52, as compared to the value of 2.20 for RSE/n.

Table 17.1. *Bootstrap estimates of prediction error for hormone data of Chapter 9. In each row of the table a bootstrap sample was generated by sampling with replacement from the hormone data, and the model specified in equation (9.21) was fit. The left column shows the resulting prediction error when this model is applied to the original data. The average of the left column (=2.52) is the simple bootstrap estimate of prediction error. The center column is the prediction error that results when the model is applied to the bootstrap sample, the so-called "apparent error." It is unrealistically low. The difference between the first and second columns is the "optimism" in the apparent error, given in the third column. The more refined bootstrap estimate adds the average optimism (=0.82) to the average residual squared error (=2.20), giving an estimate of 3.02.*

	$\text{err}(\mathbf{x}^*, \widehat{F})$	$\text{err}(\mathbf{x}^*, \widehat{F}^*)$	$\text{err}(\mathbf{x}^*, \widehat{F}) - \text{err}(\mathbf{x}^*, \widehat{F}^*)$
sample 1:	2.30	1.47	0.83
sample 2:	2.56	3.03	-0.47
sample 3:	2.30	1.65	0.65
sample 4:	2.43	1.76	0.67
sample 5:	2.44	2.00	0.44
sample 6:	2.67	1.17	1.50
sample 7:	2.68	1.23	1.45
sample 8:	2.39	1.55	0.84
sample 9:	2.86	1.76	1.10
sample 10:	2.54	1.37	1.17
AVERAGE:	2.52	1.70	0.82

This simple bootstrap approach turns out not to work very well, but fortunately, it is easy to improve upon. Take a look at the second column of Table 17.1: it shows the prediction error when the model estimated from the bootstrap sample is applied to the *bootstrap sample itself.* Not surprisingly, the values in the second column are lower on the average than those in the first column. The improved bootstrap estimate focuses on the difference between the first and second columns, called appropriately the "optimism"; it is the amount by which the average residual squared error (or "apparent error rate") underestimates the true prediction error. The overall estimate of optimism is the average of the B differences between the first and second columns, a value of 0.82 in this example.

Once an estimate of optimism is obtained, it is added to the apparent error rate to obtain an improved estimate of prediction error. Here we obtain 2.20+0.82=3.02. Of course 10 bootstrap samples are too few; repeating with 200 samples gave a value of 2.77 for the simple bootstrap estimate, and an estimate of .80 for the optimism leading to the value 2.20+0.80=3.00 for the improved estimate of prediction error. Essentially, we have added a bias-correction to the apparent error rate, in the same spirit as in Chapter 10.

17.6.2 Some details

The more refined bootstrap approach improves on the simpler approach by effectively removing the variability between the rows of Table 17.1, much like removing block effects in a two way analysis of variance. To understand further the justification for the bootstrap procedures, we need to think in terms of probability models for the data.

In Chapters 7 and 9, we describe two methods for bootstrapping regression models. The second method, which will be our focus here, treats the data $\mathbf{x}_i = (\mathbf{c}_i, y_i)$, $i = 1, 2, \ldots n$ as an i.i.d sample from the multi-dimensional distribution F. Recall that $\mathbf{c}_i$ might be a vector: in the hormone data, $\mathbf{c}_i$ would be the lot number and hours worn for the *ith* device. Call the entire sample $\mathbf{x}$. A classification problem can be expressed in the same way, with y_i indicating the class membership of the ith observation. Our discussion below is quite general, covering both the regression and classification problems.

Suppose we estimate a model from our data, producing a predicted value of y at $\mathbf{c} = \mathbf{c}_0$ denoted by

$$\eta_{\mathbf{x}}(\mathbf{c}_0). \tag{17.10}$$

We assume that $\eta_{\mathbf{x}}(c_0)$ can be expressed as a plug-in statistic, that is $\eta_{\mathbf{x}}(\mathbf{c}_0) = \eta(\mathbf{c}_0, \hat{F})$ for some function η, where $\hat{F}$ is the empirical distribution function of the data. If our problem is a regression problem as in the hormone example, then $\eta_{\mathbf{x}}(\mathbf{c}_0) = \mathbf{c}_0\hat{\beta}$ where $\hat{\beta}$ is the least squares estimate of the regression parameter. In a classification problem, $\eta_{\mathbf{x}}(\mathbf{c}_0)$ is the predicted class for an observation with $\mathbf{c} = \mathbf{c}_0$.

Let $Q[y, \eta]$ denote a measure of error between the response y and the prediction η. In regression we often choose $Q[y, \eta] = (y - \eta)^2$;

in classification typically $Q[y, \eta] = I_{\{y \neq \eta\}}$, that is $Q[y, \eta] = 1$ if $y \neq \eta$ and 0 otherwise.

The prediction error for $\eta_{\mathbf{X}}(\mathbf{c}_0)$ is defined by

$$\text{err}(\mathbf{x}, F) \equiv \text{E}_{0F}\{Q[Y_0, \eta_{\mathbf{X}}(\mathbf{C}_0)]\}. \tag{17.11}$$

The notation E_{0F} indicates expectation over a new observation $(\mathbf{C}_0, Y_0)$ from the population F. Note that E_{0F} does not average over the data set $\mathbf{x}$, which is considered fixed. The apparent error rate is

$$\text{err}(\mathbf{x}, \hat{F}) = \text{E}_{0\hat{F}}\{Q[Y_0, \eta_{\mathbf{X}}(\mathbf{c}_i)]\} = \frac{1}{n}\sum_{1}^{n} Q[y_i, \eta_{\mathbf{X}}(\mathbf{c}_i)] \tag{17.12}$$

because "$\text{E}_{0\hat{F}}$" simply averages over the n observed cases $(\mathbf{c}_i, y_i)$. In regression with $Q[y, \eta] = (y - \eta)^2$, we have $\text{err}(\mathbf{x}, \hat{F}) = \sum_1^n [y_i - \eta_{\mathbf{X}}(\mathbf{c}_i)]^2/n$, while in classification with $Q[y, \eta] = I_{\{y \neq \eta\}}$, it equals $\#\{\eta_{\mathbf{X}}(\mathbf{c}_i) \neq y_i\}/n$ the misclassification rate over the original data set.

The K-fold cross-validation estimate of Section 17.3 can also be expressed in this framework. Let $k(i)$ denote the part containing observation i, and $\eta_{\mathbf{X}}^{-k(i)}(\mathbf{c})$ be the predicted value at $\mathbf{c}$, computed with the $k(i)$th part of the data removed. Then the cross-validation estimate of the true error rate is

$$\frac{1}{n}\sum_{i=1}^{n} Q[y_i, \eta_{\mathbf{X}}^{-k(i)}(\mathbf{c}_i)]. \tag{17.13}$$

To construct a bootstrap estimate of prediction error we apply the plug-in principle to equation (17.11). Let $\mathbf{x}^* = \{(\mathbf{c}_1^*, y_1^*), (\mathbf{c}_2^*, y_2^*), \ldots (\mathbf{c}_n^*, y_n^*)\}$ be a bootstrap sample. Then the plug-in estimate of $\text{err}(\mathbf{x}, F)$ is

$$\text{err}(\mathbf{x}^*, \hat{F}) = \frac{1}{n}\sum_{1}^{n} Q[y_i, \eta_{\mathbf{X}^*}(\mathbf{c}_i)] \tag{17.14}$$

In this expression $\eta_{\mathbf{X}^*}(\mathbf{c}_i)$ is the predicted value at $\mathbf{c} = \mathbf{c}_i$, based on the model estimated from the bootstrap data set $\mathbf{x}^*$.

We could use $\text{err}(\mathbf{x}^*, \hat{F})$ as our estimate, but it involves only a single bootstrap sample and hence is too variable. Instead, we must focus on the *average* prediction error

$$\text{E}_F[\text{err}(\mathbf{x}, F)], \tag{17.15}$$

with E_F indicating the expectation over data sets $\mathbf{x}$ with observations $\mathbf{x}_i \sim F$. The bootstrap estimate is

$$E_{\hat{F}}[\text{err}(\mathbf{x}^*, \hat{F})] = E_{\hat{F}} \sum_1^n Q[y_i, \eta_{\mathbf{x}^*}(\mathbf{c}_i)]/n. \tag{17.16}$$

Intuitively, the underlying idea is much the same as in Figure 8.3: in the "bootstrap world", the bootstrap sample is playing the role of the original sample, while the original sample is playing the role of the underlying population F.

Expression (17.16) is an ideal bootstrap estimate, corresponding to an infinite number of bootstrap samples. With a finite number B of bootstrap samples, we approximate this as follows. Let $\eta_{\mathbf{x}^{*b}}(\mathbf{c}_i)$ be the predicted value at $\mathbf{c}_i$, from the model estimated on bth bootstrap sample, $b = 1, 2, \ldots B$. Then our approximation to $E_{\hat{F}}[\text{err}(\mathbf{x}^*, \hat{F})]$ is

$$\widehat{E}_{\hat{F}}[\text{err}(\mathbf{x}^*, \hat{F})] = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{n} Q[y_i, \eta_{\mathbf{x}^{*b}}(\mathbf{c}_i)]/n. \tag{17.17}$$

In regression $\sum_1^n Q[y_i, \eta_{\mathbf{x}^{*b}}(\mathbf{c}_i)]/n = \sum_{i=1}^n [y_i - \eta_{\mathbf{x}^{*b}}(\mathbf{c}_i)]^2/n$; these are the values in the left hand column of Table 17.1, and their average (2.52) corresponds to the formula in equation (17.17).

The more refined bootstrap approach estimates the bias in $\text{err}(\mathbf{x}, \hat{F})$ as an estimator of $\text{err}(\mathbf{x}, F)$, and then corrects $\text{err}(\mathbf{x}, \hat{F})$ by subtracting its estimated bias. We define the average optimism by

$$\omega(F) \equiv E_F[\text{err}(\mathbf{x}, F) - \text{err}(\mathbf{x}, \hat{F})]. \tag{17.18}$$

This is the average difference between the true prediction error and the apparent error, over data sets $\mathbf{x}$ with observations $\mathbf{x}_i \sim F$. Note that $\omega(F)$ will tend to be positive because the apparent error rate tends to underestimate the prediction error. The bootstrap estimate of $\omega(F)$ is obtained through the plug-in principle:

$$\omega(\hat{F}) = E_{\hat{F}}[\text{err}(\mathbf{x}^*, \hat{F}) - \text{err}(\mathbf{x}^*, \hat{F}^*)]. \tag{17.19}$$

Here $\hat{F}^*$ is the empirical distribution function of the bootstrap

sample $\mathbf{x}^*$. The approximation to this ideal bootstrap quantity is

$$\widehat{\omega}(\hat{F}) = \frac{1}{B \cdot n}\Big\{\sum_{b=1}^{B}\sum_{i=1}^{n} Q[y_i, \eta_{\mathbf{X}^{*b}}(\mathbf{c}_i)] - \sum_{b=1}^{B}\sum_{i=1}^{n} Q[y^*_{ib}, \eta_{\mathbf{X}^{*b}}(\mathbf{c}^*_i)]\Big\}. \tag{17.20}$$

In the above equation, $\eta_{\mathbf{X}^{*b}}(\mathbf{c}^*_i)$ is the predicted value at $\mathbf{c}^*_i$ from the model estimated on the bth bootstrap sample, $b = 1, 2, \ldots B$, and y^*_{ib} is the response value of the ith observation for the bth bootstrap sample. In Table 17.1, this is estimated by the average difference between the second and third columns, namely 0.82. The final estimate of prediction error is the apparent error plus the downward bias in the apparent error given by (17.20),

$$\text{err}(\mathbf{x}, \hat{F}) + \omega(\hat{F}) \tag{17.21}$$

which is approximated by $\frac{1}{n}\sum_1^n Q[y_i, \eta_{\mathbf{X}}(\mathbf{c}_i)] + \widehat{\omega}(\hat{F})$. This equals 2.20+0.82=3.02 in our example.

Both $\omega(\hat{F})$ and $\widehat{\text{E}}[\text{err}(\mathbf{x}^*, \hat{F})]$ do not fix $\mathbf{x}$ (as specified in definition 17.11), but instead measure averages over data sets drawn from $\hat{F}$. The refined estimate in (17.21) is superior to the simple estimate (17.17) because it uses the observed $\mathbf{x}$ in the first term $\text{err}(\mathbf{x}, \hat{F})$; averaging only enters into the correction term $\omega(\hat{F})$.

17.7 The .632 bootstrap estimator

The simple bootstrap estimate in (17.17) can be written slightly differently

$$\widehat{\text{E}}_{\hat{F}}[\text{err}(\mathbf{x}^*, \hat{F})] = \frac{1}{n}\sum_{i=1}^{n}\sum_{b=1}^{B} Q[y_i, \eta_{\mathbf{X}^{*b}}(\mathbf{c}_i)]/B. \tag{17.22}$$

We can view equation (17.22) as estimating the prediction error for each data point $(\mathbf{c}_i, y_i)$ and then averaging the error over $i = 1, 2, \ldots n$. Now for each data point $(\mathbf{c}_i, y_i)$, we can divide the bootstrap samples into those that contain $(\mathbf{c}_i, y_i)$ and those that do not. The prediction error for the data point $(\mathbf{c}_i, y_i)$ will likely be larger for a bootstrap sample *not* containing it, since such a bootstrap sample is "farther away" from $(\mathbf{c}_i, y_i)$ in some sense. The idea behind the .632 bootstrap estimator is to use the prediction error from just these cases to adjust the optimism in the apparent error rate.

Let ϵ_0 be the average error rate obtained from bootstrap data sets not containing the point being predicted (below we give details on the estimation of ϵ_0). As before, $\text{err}(\mathbf{x}, \hat{F})$ is the apparent error rate. It seems reasonable to use some multiple of $\epsilon_0 - \text{err}(\mathbf{x}, \hat{F})$ as an estimate of the optimism of $\text{err}(\mathbf{x}, \hat{F})$. The .632 bootstrap estimate of optimism is defined as

$$\hat{\omega}^{.632} = .632[\epsilon_0 - \text{err}(\mathbf{x}, \hat{F})]. \tag{17.23}$$

Adding this estimate to $\text{err}(\mathbf{x}, \hat{F})$ gives the .632 estimate of prediction error

$$\begin{aligned} \widehat{\text{err}}^{.632} &= \text{err}(\mathbf{x}, \hat{F}) + .632[\epsilon_0 - \text{err}(\mathbf{x}, \hat{F})] \\ &= .368 \cdot \text{err}(\mathbf{x}, \hat{F}) + .632 \cdot \epsilon_0. \end{aligned} \tag{17.24}$$

The factor ".632" comes from a theoretical argument showing that the bootstrap samples used in computing ϵ_0 are farther away on the average than a typical test sample, by roughly a factor of $1/.632$. The adjustment in (17.23) corrects for this, and makes $\widehat{\text{err}}^{.632}$ roughly unbiased for the true error rate. We will not give the theoretical argument here, but note that the value .632 arises because it is approximately the probability that a given observation appears in bootstrap sample of size n (Problem 17.7).

Given a set of B bootstrap samples, we estimate ϵ_0 by

$$\hat{\epsilon}_0 = \frac{1}{n} \sum_{i=1}^{n} \sum_{b \in C_i} Q[y_i, \eta_{\mathbf{X}^{*b}}(\mathbf{c}_i)]/B_i \tag{17.25}$$

where C_i is the set of indices of the bootstrap samples not containing the ith data point, and B_i is the number of such bootstrap samples. Table 17.2 shows the observation numbers appearing in each of the 10 bootstrap samples of Table 17.1. Observation #5, for example, does not appear in bootstrap samples 3,4,8, and 9. In the notation of equation (17.25), $C_i = (3, 4, 8, 9)$. So we would use only these four bootstrap samples in estimating the prediction error for observation $i = 5$ in equation (17.25).

In our example, $\hat{\epsilon}_0$ equals 3.63. Not surprisingly, this is larger than the apparent error 2.20, since it is the average prediction error for data points *not appearing* in the bootstrap sample used for their prediction. The .632 estimate of prediction error is therefore $.368 \cdot 2.20 + .632 \cdot 3.63 = 3.10$, close to the value of 3.00 obtained from the more refined bootstrap approach earlier.

Table 17.2. *The observation numbers appearing in each of the 10 bootstrap samples of Table 17.1.*

Bootstrap sample									
1	2	3	4	5	6	7	8	9	10
1	16	25	1	14	15	14	23	6	5
5	5	4	7	10	24	7	17	26	9
23	16	12	12	2	12	1	15	10	3
11	24	16	7	8	18	6	9	9	3
11	11	14	14	13	15	11	6	27	26
24	14	27	25	5	23	21	22	10	4
15	17	24	1	1	9	22	9	23	25
10	26	7	22	7	8	5	22	7	21
27	11	23	26	1	7	27	3	3	20
26	27	18	4	6	9	25	8	7	15
4	20	14	26	25	25	25	7	9	14
2	10	13	15	25	9	23	26	4	5
5	26	2	9	19	6	22	2	18	7
24	26	27	6	20	22	8	17	11	25
1	22	14	26	5	18	6	17	19	20
27	22	8	7	20	25	23	22	20	16
8	21	3	21	17	2	11	27	21	17
17	21	6	10	25	26	4	22	17	23
9	26	17	17	4	7	22	8	3	12
4	16	27	14	11	21	17	15	11	8
14	14	11	13	21	14	25	24	2	26
14	20	25	18	12	15	7	16	12	19
13	14	8	22	16	24	16	3	8	15
22	23	25	25	24	4	3	19	22	3
8	13	19	24	9	14	27	27	8	9
2	13	26	7	9	27	18	23	1	15
3	16	25	1	18	5	8	3	14	23

As a matter of interest, the average prediction error for data points that *did* appear in the bootstrap sample used for their prediction was 3.08; this value, however, is not used in the construction of the .632 estimator.

17.8 Discussion

All of the estimates of prediction error described in this chapter are significant improvements over the apparent error rate. Which

is best among these competing methods is not clear. The methods are asymptotically the same, but can behave quite differently in small samples. Simulation experiments show that cross-validation is roughly unbiased but can show large variability. The simple bootstrap method has lower variability but can be severely biased downward; the more refined bootstrap approach is an improvement but still suffers from downward bias. In the few studies to date, the .632 estimator performed the best among all methods, but we need more evidence before making any solid recommendations.

S language functions for calculating cross-validation and bootstrap estimates of prediction error are described in the Appendix.

17.9 Bibliographic notes

Key references for cross-validation are Stone (1974, 1977) and Allen (1974). The AIC is proposed by Akaike (1973), while the BIC is introduced by Schwarz (1978). Stone (1977) shows that the AIC and leave one out cross-validation are asymptotically equivalent. The C_p statistic is proposed in Mallows (1973). Generalized cross-validation is described by Golub, Heath and Wahba (1979) and Wahba (1980); a further discussion of the topic may be found in the monograph by Wahba (1990). See also Hastie and Tibshirani (1990, chapter 3). Efron (1983) proposes a number of bootstrap estimates of prediction error, including the optimism and .632 estimates. Efron (1986) compares C_p, CV, GCV and bootstrap estimates of error rates, and argues that GCV is closer to C_p than CV. Linhart and Zucchini (1986) provide a survey of model selection techniques. The use of cross-validation and the bootstrap for model selection is studied by Breiman (1992), Breiman and Spector (1992), Shao (1993) and Zhang (1992). The CART (Classification and Regression Tree) methodology is due to Breiman *et al.* (1984). A study of cross-validation and bootstrap methods for these models is carried out by Crawford (1989). The CART tree example is taken from Giampaolo *et.al.* (1988).

17.10 Problems

17.1 (a) Let $\mathbf{C}$ be a regression design matrix as described on page 106 of Chapter 9. The projection or "hat" matrix that produces the fit is $\mathbf{H} = \mathbf{C}(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T$. If h_{ii} denotes the iith element of $\mathbf{H}$, show that the cross-validated resid-

ual can be written as

$$y_i - \hat{y}_i^{-i} = \frac{y_i - \hat{y}_i}{1 - h_{ii}}. \tag{17.26}$$

(Hint: see the Sherman-Morrison-Woodbury formula in chapter 1 of Golub and Van Loan, 1983).

(b) Use this result to show that $y_i - \hat{y}_i^{-i} \geq y_i - \hat{y}_i$.

17.2 Find the explicit form of h_{ii} for the hormone data example.

17.3 Using the result of Problem 17.1 we can derive a simplified version of cross-validation, by replacing each h_{ii} by its average value $\bar{h} = \sum_1^n h_{ii}/n$. The resulting estimate is called "generalized cross-validation":

$$\text{GCV} = \frac{1}{n}\sum_1^n \left(\frac{y_i - \hat{y}_i}{1 - \bar{h}}\right)^2. \tag{17.27}$$

Use a Taylor series approximation to show the close relationship between GCV and the C_p statistic.

17.4 Use a Taylor series approximation to show that the adjusted residual squared error (17.7) and the C_p statistic (17.8) are equal to first order, if RSE/n is used as an estimate of σ^2 in C_p.

17.5 Carry out a linear discriminant analysis of some classification data and use cross-validation to estimate the misclassification rate of the fitted model. Analyze the same data using the CART procedure and cross-validation, and compare the results.

17.6 Make explicit the quantities $\text{err}(\mathbf{x}, F)$, $\text{err}(\mathbf{x}, \hat{F})$ and their bootstrap counterparts, in a classification problem with prediction error equal to misclassification rate.

17.7 Given a data set of n distinct observations, show that the probability that an observation appears in a bootstrap sample of size n is $\to (1 - e^{-1}) \approx .632$ as $n \to \infty$.

17.8 (a) Carry out a bootstrap analysis for the hormone data, like the one in Table 17.1, using $B = 100$ bootstrap samples. In addition, calculate the average prediction error $\hat{\epsilon}_0$ for observations that do not appear in the bootstrap sample used for their prediction. Hence compute the .632 estimator for these data.

(b) Calculate the average prediction error $\hat{\epsilon}_j$ for observations that appear exactly j times in the bootstrap sample used for their prediction, for $j = 0, 1, 2, \ldots$. Graph $\hat{\epsilon}_j$ against j and give an explanation for the results.

17.9 *Tree selection in CART.* Let T be a classification tree and define the cost of a tree by

$$\text{cost}(T) = \text{mr}(T) + \lambda|T|, \tag{17.28}$$

where $\text{mr}(T)$ denotes the (apparent) misclassification rate of T and $|T|$ is the number of terminal nodes in T. The parameter $\lambda \geq 0$ trades off the classification performance of the tree with its complexity. Denote by T_0 a fixed (large) tree, and consider all subtrees T of T_0, that is, all trees which can be obtained by pruning branches of T_0.

Let T_α be the subtree of T_0 with smallest cost. One can show that for each value $\alpha \geq 0$, a unique T_α exists (when more than one tree exists with the same cost, there is one tree that is a subtree of the others, and we choose that tree). Furthermore, if $\alpha_1 > \alpha_2$, then T_{α_1} is a subtree of T_{α_2}. The CART procedure derives an estimate $\hat{\alpha}$ of α by 10-fold cross-validation, and then the final tree chosen is $T_{\hat{\alpha}}$.

Here is how cross-validation is used. Let T_α^{-k} be the cost-minimizing tree for cost parameter α, when the *kth* part of the data is withheld ($k = 1, 2, \ldots 10$). Let $\text{mr}_k(T_\alpha^{-k})$ be the misclassification rate when T_α^{-k} is used to predict the *kth* part of the data.

For each fixed α, the misclassification rate is estimated by

$$\frac{1}{10}\sum_{k=1}^{10} \text{mr}_k(T_\alpha^{-k}). \tag{17.29}$$

Finally, the value $\hat{\alpha}$ is chosen to minimize (17.29).

This procedure is an example of *adaptive estimation*, discussed in the next chapter. More details may be found in Breiman *et al.* (1984).

Write a computer program that grows and prunes classification trees. You may assume that the predictor variables are binary, to simplify the splitting process. Build in 10-fold cross-validation and try your program on a set of real data.