8 Bootstrapping, jackknifing and cross validation. Reusing your data

Bootstrapping, jackknifing and *cross validation* are three superficially similar statistical techniques that involve reusing or re-sampling your data. In each case a single sample of observations is considered as many samples with the same estimation process being applied to each of them. However, the purposes of this reuse of he samples are quite different for each method. In summary

- Bootstrapping is a method for evaluating the variance of an estimator
- Jackknifing is a method for reducing the bias of an estimator, and evaluating the variance of an estimator.
- Cross validation is a method for evaluating the error involved in making predictions.

Since each of these methods involves repeatedly applying a statistical procedure to a modified or sampled set of data, they are all computationally intensive. In many special cases computational shortcuts can be worked out, however, in most applications it is far easier to simply use existing methods in a loop.

8.1 Bootstrapping

In many of the examples of estimation we have met so far we have been able not only to get estimators but to estimate the variance of the estimators. For example we know that $\bar{X}$, the sample mean, is generally a good estimator of μ the population mean. But we also know that the variance of $\bar{X}$ over repeated sampling is $\frac{\sigma^2}{n}$, when σ^2 is the population variance. Hence we are able to compute the *standard error*

$$\sqrt{\frac{s^2}{n}} \quad \text{where} \quad s^2 = \frac{1}{n-1}\sum_i (X_i - \bar{X})^2$$

Note here that we have used our data twice. Once to get the estimate and once to get the estimate of the variance of the estimate. Bootstrapping takes this idea to an extreme and is useful in cases when the variance of the estimator is difficult to get. The process is like this:

- Observe a sample $X = \{X_1 \dots X_n\}$.
- Compute $\hat{\theta}(X)$ a function of the data which estimates some parameter θ of the model.
- For $i = 1$ up to s where s is the number of bootstrap samples being generated:
  - generate a *bootstrap sample* $X^i = \{X_1^i \dots X_n^i\}$ by **sampling with replacement from the original observed data set**
  - compute $\hat{\theta}^i = \hat{\theta}^i(X^i)$ in the same way that you calculated the original estimate $\hat{\theta}$.

- Compute the sample mean and sample variance of the $\hat{\theta}^i$s

$$\bar{\hat{\theta}} = \frac{1}{s}\sum_i \hat{\theta}^i \quad \text{and} \quad b = \frac{1}{s-1}\sum_i (\hat{\theta}^i - \bar{\hat{\theta}})^2$$

- Give $\hat{\theta}$ as you estimate of θ and $\sqrt{b}$ as your *bootstrap estimate of its standard error.*

Notes

Suppose that the data $X_1 \ldots X_n$ come from a distribution with cumulative distribution function $F(x)$. In an ideal world, we would not need to bootstrap, we would just get more samples from F and evaluate the variance of our estimate with these new samples. What the bootstrap does is to replace sampling from F with sampling from the empirical distribution function which is an estimate of F. This is shown in figure 17: the blue line shows the distribution function for the standard Normal. The red line shows the empirical distribution function of a sample of 100 random normal observations. We would like to take new samples from the distribution shown in blue but in reality we don't know this so we can not. Instead we take new samples from the distribution shown in red and hope that the red line is a good approximation to the blue one. Thus, we see that the bootstrap is only as good as our empirical distribution function is and estimate of the true distribution function.

The number of bootstrap samples, s, needs to be large but since the error in evaluating the variance decreases as $\frac{1}{\sqrt{s}}$, there is no point in making s too big. Some trial and error is called for. Also you can plot the value of the bootstrap estimate of variance against s to see whether it has settled down to some value. See figure 18.

The bootstrap is a robust, non-parametric, method that does well with smaller samples or awkward distributions. There is also the *semi-parametric bootstrap* in which a smoothed version of the empirical distribution function is sampled from. This is done very easily by adding a small random Normal$(0, \sigma^2)$ to each re-sampled observation. When σ^2 is small the amount of smoothing is small, when large the amount of smoothing is large.

Computation

Suppose we have an R function `f()`, say, that computes a statistic from some data set `x` we can generate our bootstrapped estimate of variance `b` as follows:

```
> s = 1000
> t = rep(0,s)
> for (i in 1:s) t[i] = f(sample(x,replace=TRUE))
> b = var(t)
```

Using `hist(t)` we can also get an impression of the sampling distribution of the statistic.

To use the semi-parametric bootstrap we would replace the third line above with

```
> for (i in 1:s) t[i] = f( sample(x,replace=TRUE) + a * rnorm(length(x)) )
```

where `a` is the smoothing parameter.

Figure 17: The empirical distribution function as an estimate of the true distribution function

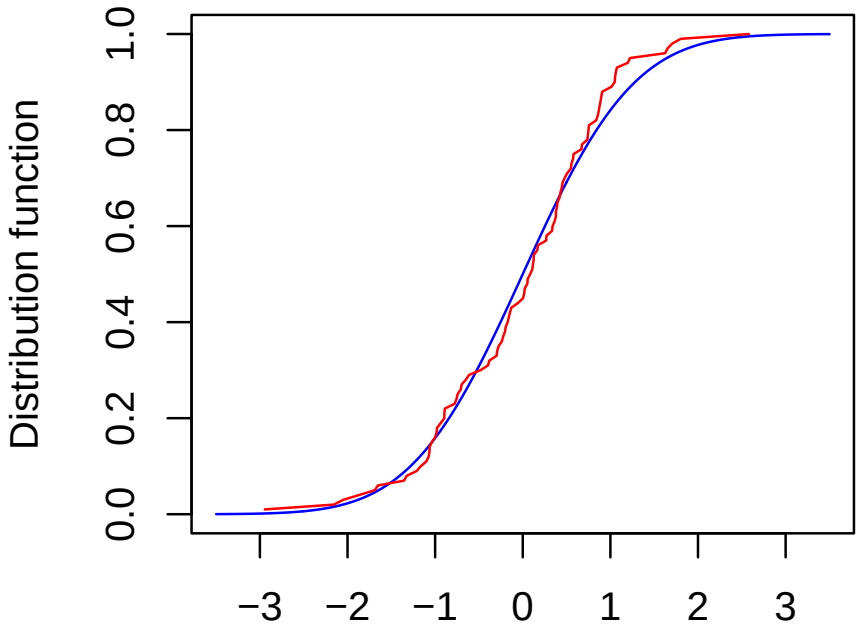
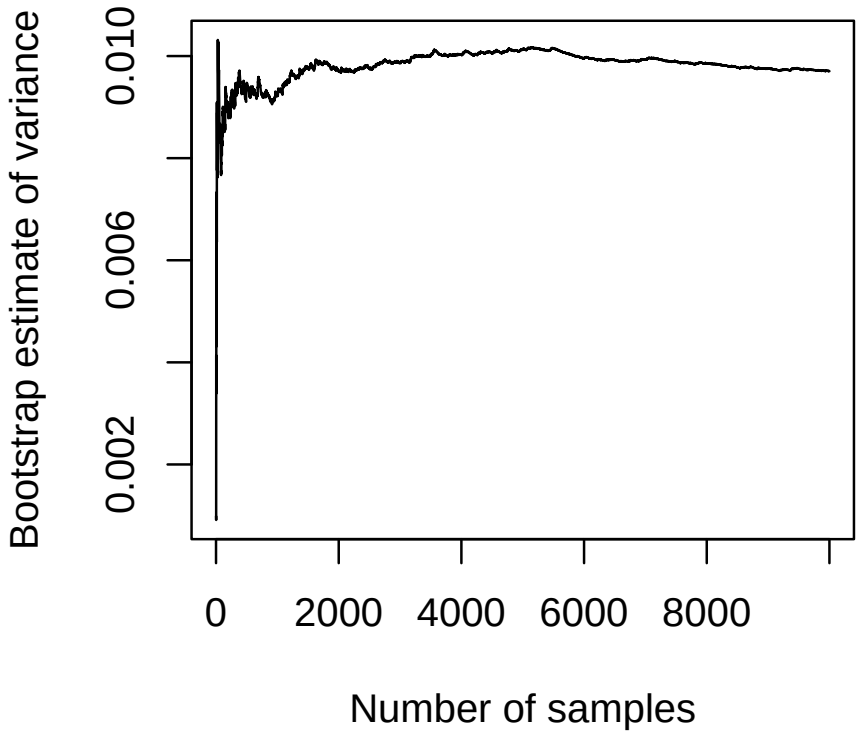


Figure 18: Convergence of the bootstrap estimate of the variance of an estimate of the mean of 100 observations from an Exponential(1) distribution



8.2 Jackknifing

The *jackknife* is a more orderly version of the bootstrap. Instead of generating a set of random samples from $X_1 \dots X_n$ we generate n samples of size $n-1$ by leaving out one observation at a time.

- Observe a sample $X = \{X_1 \dots X_n\}$.
- Compute $\hat{\theta}(X)$ a function of the data which estimates some parameter θ of the model.
- For $i = 1$ up to n
  - generate a *jackknife sample* $X^{-i} = \{X_1, \dots X_{i-1}, X_{i+1} \dots X_n\}$ by leaving out the ith observation
  - calculate $\hat{\theta}_{-i}$ by applying the estimation process to the jackknife sample
- Calculate the *jackknifed estimate*

$$\hat{\theta}_* = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_{-i}$$

- and the *jackknife estimate of variance*

$$\frac{n-1}{n} \sum_{i=1}^{n} (\hat{\theta}_{-i} - \hat{\theta}_*)^2$$

Notes

One of the criteria for choosing an estimate is whether it is *biased* or *unbiased*. If an estimator is unbiased this means that its mean is equal to the parameter that we want to estimate. For example, we know that the mean of the sample variance $s^2 = \frac{1}{n-1} \sum_i (X_i - \bar{X})^2$ is the population variance σ^2. Hence, s^2 is an unbiased estimator for σ^2. In fact this is one reason why we choose to divide by $n-1$ rather than n. The sample mean is also an unbiased estimator of the population mean. The difference between the mean of an estimator and the parameter we want to estimate is called the *bias* of the estimator. If $\hat{\theta}$ is an estimator of θ the bias is

$$\theta - E\hat{\theta}$$

The jackknife can be used to estimate the variance of an estimate and also the bias of an estimate. The variance is given above. The estimate of the bias is

$$(n-1)(\hat{\theta}_* - \hat{\theta})$$

Hence, we get the *bias corrected jackknife estimate*

$$\begin{aligned} \tilde{\theta} &= \hat{\theta} - (n-1)\,(\hat{\theta}_* - \hat{\theta}) \\ &= n\,\hat{\theta} - (n-1)\,\hat{\theta}_* \end{aligned}$$

Computation

In R, again assuming that we have a data set `x` and some function `f()` that evaluates an estimator we can jackknife as follows making use of the `[-i]` construct:

```
> t = rep(0,length(x))
> for (i in 1:length(x)) t[i] = f(x[-i])
> jke = mean(t)
> jkv = (n-1)/n * sum( (t-jke)^2 )
> jkbce = n * f(x) - (n-1) * jke
```

so that `jke` is the jackknifed estimate, `jkv` is the jackknife estimate of variance and `jkbce` is the jackknife bias corrected estimate.

8.3 Cross validation

Cross validation is a method for evaluating the predictive error when we fit a function relating two, or more, variables. Although, the purpose and setting are quite different to the jackknife it also involves leaving out one observation at a time.

To see the value of cross validation we must first appreciate the difference between the fitted error and the predictive error. For example, when we perform a least squares linear regression of Y on X using a bivariate sample of observations $(X_1, Y_1), \ldots (X_n, Y_n)$ we minimize the fitted error

$$\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 = \sum_{i=1}^{n} (Y_i - (\hat{a} + \hat{b} X_i))^2$$

However, if we are primarily interested in predicting Y_{n+1} the next value of Y given only X_{n+1} the next value of X the function we are interested in minimizing

$$(Y_{n+1} - (\hat{a} + \hat{b} X_{n+1}))^2$$

where $\hat{a}$ and $\hat{b}$ are chosen **without using the values X_{i+1} and Y_{i+1}**.

A commonly use technique to evaluate predictive error was to split a sample in half. The first half **only** would be used to choose the fit and the second half would be used to evaluate the predictive error. Thus, we would have a sample for fitting and a sample for validation, also known as a *training set* and a *testing set*. However, this is clearly very wasteful. We should be able to fit a better model using all the data, but can we still get an estimate of the predictive error? Cross validation is devised for exactly this purpose.

- Observe a bivariate sample $(X_1, Y_1), \ldots (X_n, Y_n)$.
- For $i = 1$ up to n
  - leave the point (X_i, Y_i) out of the sample
  - fit the model $Y = f_{-i}(X)$ using the remaining points.

– calculate the predicted value of the excluded point

$$\hat{Y}_{-i} = f_{-i}(X_i)$$

- Estimate the predictive error using

$$\sum_{i=1}^{n}(Y_i - \hat{Y}_{-i})^2$$

Notes

We can see that the fitted and predictive errors are very similar. The value $\hat{Y}$ in the former is simply replaced by $\hat{Y}_{-i}$ in the latter.

Cross validation is primarily used not to estimate the parameters of a fit, but to choose between different models. For example, suppose we have two models that we want to consider, a linear and a quadratic fit:

$$\begin{aligned} Y &= a + bX \\ Y &= a + bX + cX^2 \end{aligned}$$

we know that the fitted error for the quadratic model will always be smaller than the fitted error for the linear model. However, the predictive error may not be smaller and can be used as a criterion for choosing the model.

Another common use for cross validation is to choose parameters such at the smoothing parameter used in the semi-parametric bootstrap above. In that case we would leave out each observation in turn and choose the smoothing parameter that best predicted the missing values on average.

Computation

As an example, suppose we want to use R to find find the predictive error for a fit of Y as a quadratic polynomial in X, we would do something like this:

```
> perrs = rep(0,length(y))
> for (i in 1:length(perrs))
+ {
+       cf = lsfit( cbind( x[-i], x[-i]^2 ) , y[-i] )$coef
+       yhat = cf[1] + cf[2]*x[i] + cf[3]*x[i]^2
+       perrs[i] = y[i] - yhat
+ }
> perr = sum( perrs^2 )
```

8.4 Worksheet

1. Generate a single sample of 100 observations from some Normal distribution. Compute $\bar{X}$, the usual estimate of the mean, and compute the standard error in the usual way. Now use a large number of bootstrap samples to estimate the standard error. Compare your two estimates of the variance of $\bar{X}$, and compare these with the theoretical standard error (that is the one you would get if you actually know the value of σ^2).

2. Use the parametric bootstrap to estimate the variance for the above data. Try different amounts of smoothing.

3. Use the jackknife instead of the bootstrap for question 1.

4. * Download the data set called `income`. This contains two columns the first is the number of adults in a household and the second is the combined income for the household. If we want to estimate the mean income per person we can do this in one of two ways:

   ```
   > mean(income/size)
   > mean(income)/mean(size)
   ```

   Use the bootstrap to evaluate the variances of these two estimators. Which one is the better estimator?

5. * Use the jackknife to evaluate the variance and bias of the two estimators in question 4.

6. * Consider fitting a polynomial relationship between income, Y, and household size, X:

   $$Y = a + bX + cX^2 + dX^3 + \ldots$$

   Use cross validation to choose the polynomial with the smallest predictive error.