5.1 Cross-validation

https://youtu.be/_2ij6eaaSI0

Cross-validation and the Bootstrap

we are going to tell you about cross-validation, which is a very clever device for using the same training data to tell you how well your prediction method works. The main thing we use cross-validation for is to get an idea of the test set error of our model.

- In the section we discuss two *resampling* methods: cross-validation and the bootstrap.
- These methods refit a model of interest to samples formed from the training set, in order to obtain additional information about the fitted model.
- For example, they provide estimates of test-set prediction error, and the standard deviation and bias of our parameter estimates

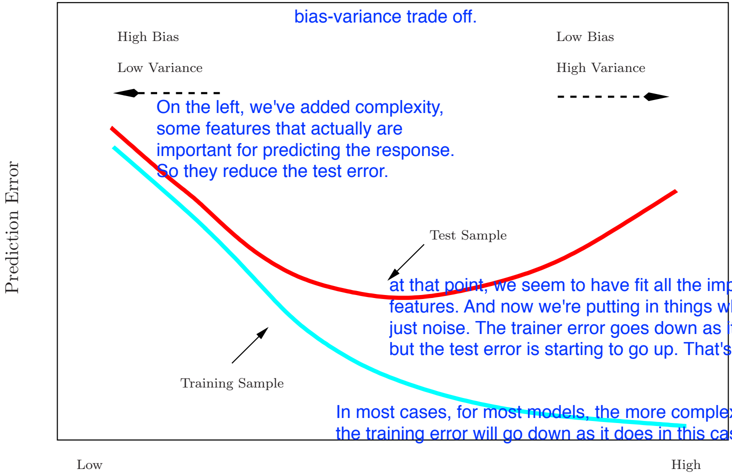
Bootstrap is most useful to get an idea of the variability or standard deviation of an estimate and its bias.

Training Error versus Test error

- Recall the distinction between the *test error* and the *training error:*
- The *test error* is the average error that results from using a statistical learning method to predict the response on a new observation, one that was not used in training the method.
- In contrast, the *training error* can be easily calculated by applying the statistical learning method to the observations used in its training.
- But the training error rate often is quite different from the test error rate, and in particular the former can *dramatically underestimate* the latter.

bias: how far off on the average the model is from the truth.
variance: is how much that the estimate varies around its average.
So bias and variance together give us test error.We want to find the model complex given the smallest test error,

Training- versus Test-Set Performance



bias-variance trade off.

On the left, we've added complexity, some features that actually are important for predicting the response. So they reduce the test error.

at that point, we seem to have fit all the important features. And now we're putting in things which are just noise. The trainer error goes down as it has to, but the test error is starting to go up. That's overfitting.

In most cases, for most models, the more complex the model, the training error will go down as it does in this case.

low means a few number of features or coefficients or predictors.

Model Complexity

Think about fitting a polynomial with the higher degree.

More on prediction-error estimates

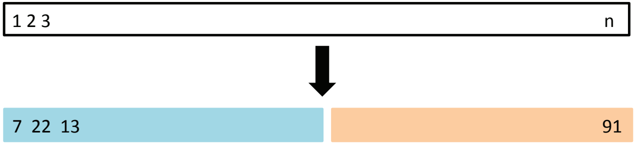
- Best solution: a large designated test set. Often not available   we don't have a large test set.
- Some methods make a *mathematical adjustment* to the training error rate in order to estimate the test error rate. These include the *Cp statistic*, *AIC* and *BIC*. They are discussed elsewhere in this course
- Here we instead consider a class of methods that 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

called “validation”, or “cross-validation”

Validation-set approach

- Here we 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.

The Validation process



A random splitting into two halves: left part is training set, right part is validation set

It seems a little wasteful if you've got a very small data set, cross-validation will remove that waste and be more efficient.

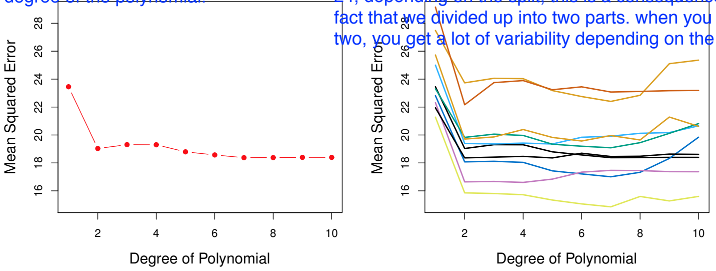
2 things you want to use cross-validation for: (1) to pick the best size of the model-- and in this case, the degree of polynomial (2) to give us an idea of how good the test error is

Example: automobile data

- Want to compare linear vs higher-order polynomial terms in a linear regression
- We randomly split the 392 observations into two sets, a training set containing 196 of the data points, and a validation set containing the remaining 196 observations.

If we do this once, do a single split, and we record the mean squared error, we get this red curve as a function of the degree of the polynomial.

when we repeat this process with more splits at random into two parts. We get a lot of variability. The minimum does tend to occur around 2. But the error is varying from about 16 up to 24, depending on the split, this is a consequence part of the fact that we divided up into two parts. when you divide data in two, you get a lot of variability depending on the split.



Left panel shows single split; right panel shows multiple splits

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. And because we're splitting in two, we're losing a lot of the power of the training set. We're throwing away half the data each time in training.
- This suggests that the validation set error may tend to *overestimate* the test error for the model fit on the entire data set. *Why?*

5.2 K-fold Cross-Validation

https://youtu.be/nZAM5OXrktY

K-fold Cross-validation

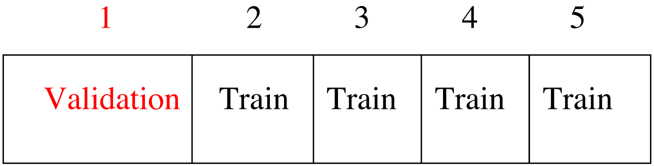
- *Widely used approach* for estimating test error.
- Estimates can be used to select best model, and to give an idea of the test error of the final chosen model.
- Idea is to randomly divide the data into K equal-sized parts. We leave out part k, fit the model to the other $K - 1$ parts (combined), and then obtain predictions for the left-out kth part.
- This is done in turn for each part $k = 1, 2, \ldots K$, and then the results are combined.

K-fold Cross-validation in detail

the best choices for K, the number of folds, is usually about 5 or 10.

randomly

Divide data into K roughly equal-sized parts ($K = 5$ here)



The details

- Let the K parts be $C_1, C_2, \dots C_K$, where C_k denotes the indices of the observations in part k. There are n_k observations in part k: if N is a multiple of K, then $n_k = n/K$.
- Compute

$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{MSE}_k$$

where $\mathrm{MSE}_k = \sum_{i \in C_k} (y_i - \hat{y}_i)^2 / n_k$, and $\hat{y}_i$ is the fit for observation i, obtained from the data with part k removed.
- Setting $K = n$ yields n-fold or *leave-one out cross-validation* (LOOCV).

if 5-fold,
CV_K ~ (MSE_1 + … MSE_5)/5

if K=n=> n-fold,
CV_n ~ (MSE_1 + … MSE_n)/n

A nice special case!

- With least-squares linear or polynomial regression, an amazing shortcut makes the cost of LOOCV the same as that of a single model fit! The following formula holds:

the overall point of this is that to do a leave one out cross-validation for these particular models, you don't actually have to leave anything out. You can do the fit on the overall data set, and then extract the information you need to get the cross-validation sum of squares.

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1-h_i}\right)^2,$$

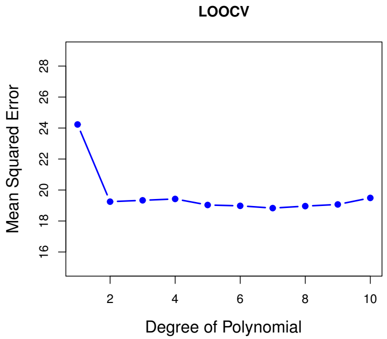
the hi tells you how much influence an observation has on its own fact. It's a number between 0 and 1.And so as an observation, it's very influential on its own fit. You can see it punishes the residual, because it divides by a number that's small, and it inflates the residual.

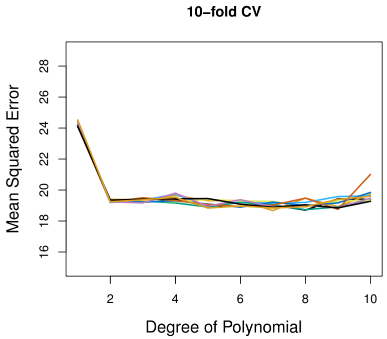
where $\hat{y}_i$ is the ith fitted value from the original least squares fit, and h_i is the leverage (diagonal of the "hat" matrix; see book for details.) This is like the ordinary MSE, except the ith residual is divided by $1 - h_i$.

- LOOCV sometimes useful, but typically doesn't *shake up* the data enough. The estimates from each fold are highly correlated and hence their average can have high variance.
- a better choice is $K = 5$ or 10.

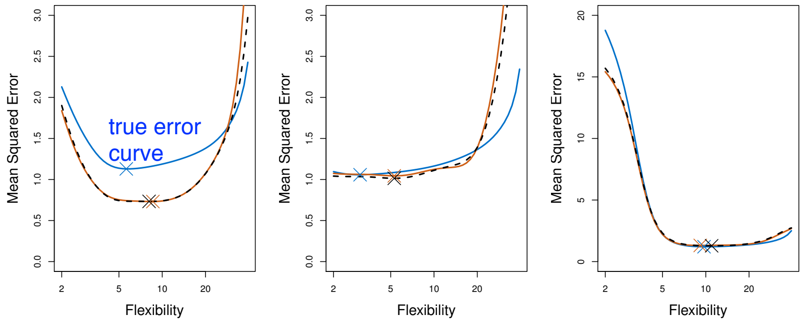
average of errors over the n folds, the n folds look very similar to each other, So as a result, that average has a high variance, because the ingredients are highly correlated.On the other hand, the LOOCV is actually trying to estimate the error rate for the training sample of almost the same size as what you have.So it's got low bias (high variance.)

Auto data revisited





True and estimated test MSE for the simulated data



LOOCV: black broken line
10 fold cross validation: orange

Other issues with Cross-validation

k fold: the training set is not as big as the original training set, the essence of prediction will be biased up a little bit, because you have less data that you're working with.

- Since each training set is only $(K-1)/K$ as big as the original training set, the estimates of prediction error will typically be biased upward. *Why?*
- This bias is minimized when $K = n$ (LOOCV), but this estimate has high variance, as noted earlier.

  when K = n, k-fold becomes LOOCV,it has minimized bias but large variance
- $K = 5$ or 10 provides a good compromise for this bias-variance tradeoff.

Cross-Validation for Classification Problems

- We divide the data into K roughly equal-sized parts $C_1, C_2, \ldots C_K$. C_k denotes the indices of the observations in part k. There are n_k observations in part k: if n is a multiple of K, then $n_k = n/K$.
- Compute

$$\mathrm{CV}_K = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{Err}_k$$

=(Err_1 + … Err_K)/K
= Err_mean

where $\mathrm{Err}_k = \sum_{i \in C_k} I(y_i \neq \hat{y}_i)/n_k$.

- The estimated standard deviation of CV_K is

When we would draw a CV curve, we should always put a standard air band around the curve

$$\widehat{\mathrm{SE}}(\mathrm{CV}_K) = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\frac{(\mathrm{Err}_k - \overline{\mathrm{Err}_k})^2}{K-1}}$$

why /k/(K-1) ????????

http://www.math.canterbury.ac.nz/~r.vale/Crossvalidation.pdf
Err_k = test error on k'th fold
Err_mean = (Err_1 + ..Err_K)/K
deviation = sqrt(1/K/(K-1) * sigma[Err_k-Err_mean]^2)

- This is a useful estimate, but strictly speaking, not quite valid. *Why not?*

we're computing the standard error if these were independent observations. But they're not strictly independent. Error_k overlaps with Error_j because they share some training samples. So there's some correlation between them. But we use this anyway.

5.3 Cross-Validation: the wrong and right way

https://youtu.be/S06JpVoNaA0

Cross-validation: right and wrong

- Consider a simple classifier applied to some two-class data:
  1. Starting with 5000 predictors and 50 samples, find the 100 predictors having the largest correlation with the class labels.
  2. We then apply a classifier such as logistic regression, using only these 100 predictors.

  How do we estimate the test set performance of this classifier?

  Can we apply cross-validation in step 2, forgetting about step 1?

let's just forget the fact that we filtered the predictors in step one. That we chose the 100 best among the 5,000. Let's just pretend like we started in step two. That we started with the 100 predictors, and that was our data. Can we apply cross-validation in step two, forgetting about step one?

NO!

You can simulate data with class labels having no correlation with the outcome so the true test error is 50%. But cross-validation, if we ignore step one, will give you an error rate of 0. So it's a serious bias. Cross-validation is telling you your classifier is perfect, when in fact your classifier is the same as flipping a coin.

- This would ignore the fact that in Step 1, the procedure *has already seen the labels of the training data*, and made use of them. This is a form of training and must be included in the validation process.
- It is easy to simulate realistic data with the class labels independent of the outcome, so that true test error =50%, but the CV error estimate that ignores Step 1 is zero! *Try to do this yourself*
- We have seen this error made in many high profile genomics papers.

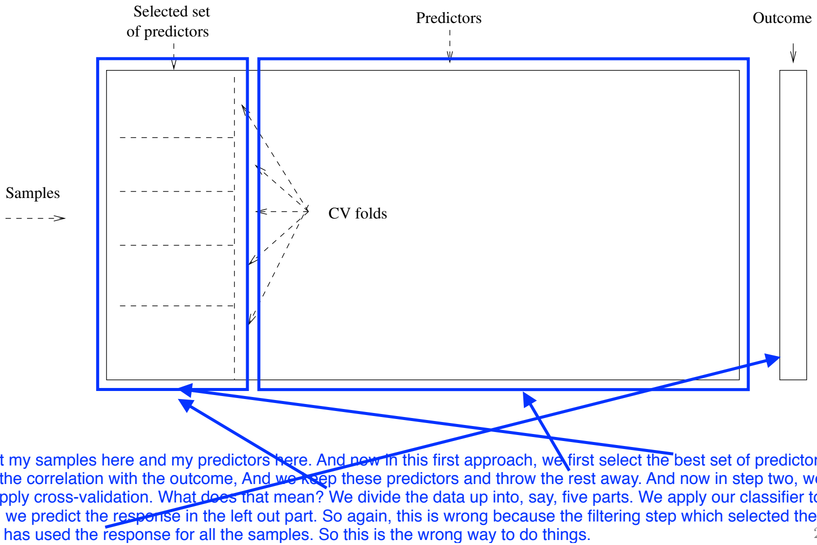
It will make the point more clearly if we increase the 5,000 to 5 million predictors. Suppose we have 5 million predictors and 50 samples. And again, there's no correlation in the population between the predictors and the class labels. We go ahead and we pick the best 100 among those 5 million predictors. We're going to find some very good-looking predictors. Despite the fact in the population no predictor has correlation with the outcome. In the data, if we look at the best among 5 million, we're going to find some very good predictors that look in the data like they have a lot of power for classifying. If we then pretend like those 100 cherry picked out of 5 million were the predictors we started with, they're going to look very good to cross-validation. So we fooled cross-validation by leaving out the first filtering step and giving it a very cherry-picked set of predictors in the second step.

The Wrong and Right Way

in some genomic studies researchers are faced with tens of thousands of genes, maybe. And it's just hard to handle them. So they do some kind of screening in the beginning just to reduce the number of variables down to a manageable set. And then forget about it afterwards, but that leads to this kind of bias. You can simulate a situation just like this where the true test error is 50%. And simulate a large number of predictors. And apply cross-validation in step two. And you'll see the error is actually very low.

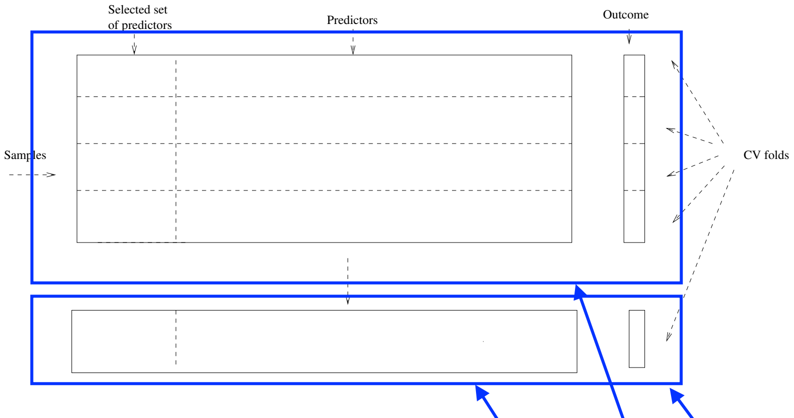
- *Wrong:* Apply cross-validation in step 2.
- *Right:* Apply cross-validation to steps 1 and 2.

Wrong Way



So I've got my samples here and my predictors here. And now in this first approach, we first select the best set of predictors based on the correlation with the outcome, And we keep these predictors and throw the rest away. And now in step two, we're going to apply cross-validation. What does that mean? We divide the data up into, say, five parts. We apply our classifier to four parts. And we predict the response in the left out part. So again, this is wrong because the filtering step which selected these predictors has used the response for all the samples. So this is the wrong way to do things.

Right Way



We first define our folds, five folds cross-validation. Before we do any fitting, we remove one of the folds. All the data for that fold, the predictors and the response variable. And now we can do whatever we want on the other four parts. We can filter and fit however we want. When we've finished our fitting, we then take the model and we predict the response for the left out part. Key point being, though, that we form the folds before we filter or fit to the data. So that we're applying cross-validation to the entire process, not just the second step. So that's the right way to do it. So in each of the 4/5ths folds, we might screen off a different set of predictors each time. And we probably will. And so that variability is going to get taken into account here.

5.4 The Bootstrap

https://youtu.be/p4BYWX7PTBM

The Bootstrap

- The *bootstrap* is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method.
- For example, it can provide an estimate of the standard error of a coefficient, or a confidence interval for that coefficient.

Where does the name came from?

- The use of the term bootstrap derives from the phrase *to pull oneself up by one's bootstraps*, widely thought to be based on one of the eighteenth century "The Surprising Adventures of Baron Munchausen" by Rudolph Erich Raspe:

  The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.

- It is not the same as the term "bootstrap" used in computer science meaning to "boot" a computer from a set of core instructions, though the derivation is similar.

A simple example

- Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of X and Y, respectively, where X and Y are random quantities.
- We will invest a fraction α of our money in X, and will invest the remaining $1 - \alpha$ in Y.
- We wish to choose α to minimize the total risk, or variance, of our investment. In other words, we want to minimize $\mathrm{Var}(\alpha X + (1 - \alpha)Y)$.
- One can show that the value that minimizes the risk is given by

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}},$$

where $\sigma_X^2 = \mathrm{Var}(X), \sigma_Y^2 = \mathrm{Var}(Y)$, and $\sigma_{XY} = \mathrm{Cov}(X, Y)$.

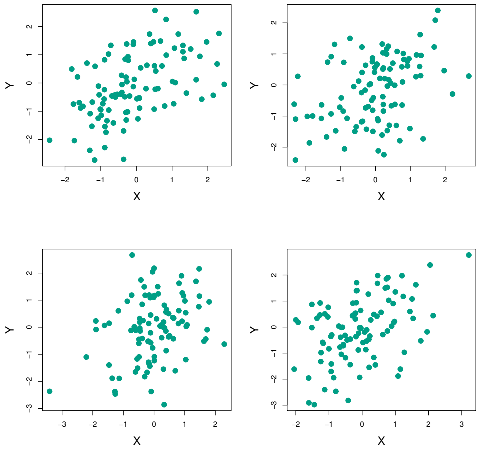
Example continued

- But the values of σ_X^2, σ_Y^2, and σ_{XY} are unknown.
- We can compute estimates for these quantities, $\hat{\sigma}_X^2$, $\hat{\sigma}_Y^2$, and $\hat{\sigma}_{XY}$, using a data set that contains measurements for X and Y.
- We can then estimate the value of α that minimizes the variance of our investment using

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}.$$

if we have a sample of x and y, we can get the empirical estimates of the variances and covariances, plug them in, and get an estimate of alpha.

Example continued



Each panel displays 100 *simulated returns for investments* X *and* Y*. From left to right and top to bottom, the resulting estimates for* α *are* 0.576, 0.532, 0.657, *and* 0.651.

Example continued

- To estimate the standard deviation of $\hat{\alpha}$, we repeated the process of simulating 100 paired observations of X and Y, and estimating α 1,000 times.
- We thereby obtained 1,000 estimates for α, which we can call $\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{1000}$.
- The left-hand panel of the Figure on slide 29 displays a histogram of the resulting estimates.
- For these simulations the parameters were set to $\sigma_X^2 = 1, \sigma_Y^2 = 1.25$, and $\sigma_{XY} = 0.5$, and so we know that the true value of α is 0.6 (indicated by the red line).

Example continued

- The mean over all 1,000 estimates for α is

$$\bar{\alpha} = \frac{1}{1000} \sum_{r=1}^{1000} \hat{\alpha}_r = 0.5996,$$

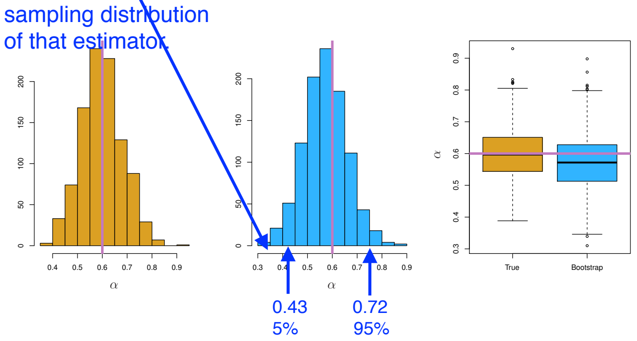
  very close to $\alpha = 0.6$, and the standard deviation of the estimates is

$$\sqrt{\frac{1}{1000-1} \sum_{r=1}^{1000} (\hat{\alpha}_r - \bar{\alpha})^2} = 0.083.$$

- This gives us a very good idea of the accuracy of $\hat{\alpha}$: $\mathrm{SE}(\hat{\alpha}) \approx 0.083$.
- So roughly speaking, for a random sample from the population, we would expect $\hat{\alpha}$ to differ from α by approximately 0.08, on average.

Suppose I want a 0.9 confidence interval so that it contains a true value with probability 90%. => to grab 5 and 95% of quantiles, or 5 and 95% points of this histogram. There's 1,000 points here. So 5% point would be the 50th smallest and 950 biggest value of alpha. => It's 0.43 and 0.72 (see slide 35).

Results

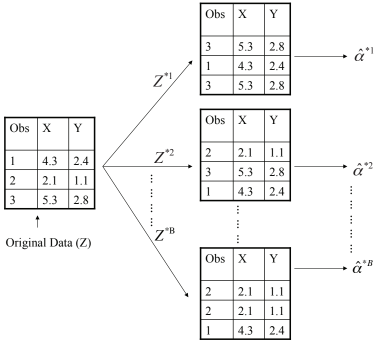


Left: A histogram of the estimates of α obtained by generating 1,000 simulated data sets from the true population. *Center:* A histogram of the estimates of α obtained from 1,000 bootstrap samples from a single data set. *Right:* The estimates of α displayed in the left and center panels are shown as boxplots. In each panel, the pink line indicates the true value of α.

Now back to the real world

- The procedure outlined above cannot be applied, because for real data we cannot generate new samples from the original population.
- However, the bootstrap approach allows us to use a computer to mimic the process of obtaining new data sets, so that we can estimate the variability of our estimate without generating additional samples.
- Rather than repeatedly obtaining independent data sets from the population, we instead obtain distinct data sets by repeatedly sampling observations from the original data set *with replacement*.
- Each of these "bootstrap data sets" is created by sampling *with replacement*, and is the *same size* as our original dataset. As a result some observations may appear more than once in a given bootstrap data set and some not at all.

Example with just 3 observations



A graphical illustration of the bootstrap approach on a small sample containing $n = 3$ observations. Each bootstrap data set contains n observations, sampled with replacement from the original data set. Each bootstrap data set is used to obtain an estimate of α

ref: 5.4_Review Questions.pdf

- Denoting the first bootstrap data set by Z^{*1}, we use Z^{*1} to produce a new bootstrap estimate for α, which we call $\hat{\alpha}^{*1}$
- This procedure is repeated B times for some large value of B (say 100 or 1000), in order to produce B different bootstrap data sets, $Z^{*1}, Z^{*2}, \ldots, Z^{*B}$, and B corresponding α estimates, $\hat{\alpha}^{*1}, \hat{\alpha}^{*2}, \ldots, \hat{\alpha}^{*B}$.
- We estimate the standard error of these bootstrap estimates using the formula

$$\mathrm{SE}_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1} \sum_{r=1}^{B} \left(\hat{\alpha}^{*r} - \bar{\hat{\alpha}}^{*}\right)^2}.$$

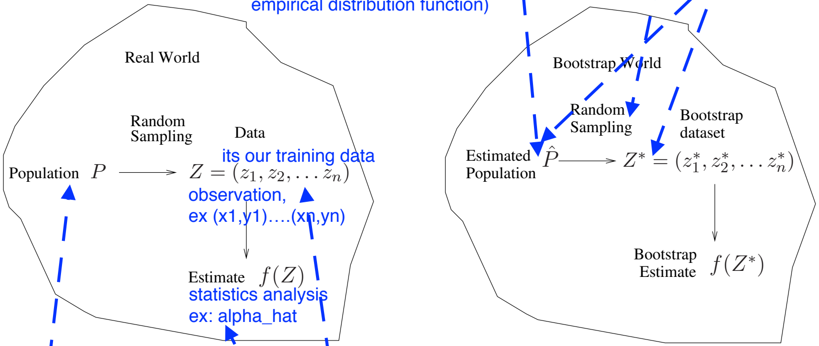
- This serves as an estimate of the standard error of $\hat{\alpha}$ estimated from the original data set. See center and right panels of Figure on slide 29. Bootstrap results are in blue. For this example $\mathrm{SE}_B(\hat{\alpha}) = 0.087$.

5.5 More on the Bootstrap

https://youtu.be/BzHz0J9a6k0

A general picture for the bootstrap

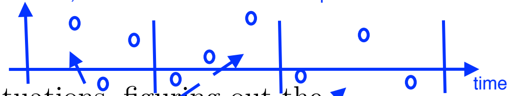
But not often can you actually do that in practice. Because you don't have access to the population. You can't get more training data, typically. What you have to work with is just your given training example (p hat) (= empirical distribution function)



And what we wanted to get was an idea of the standard error of alpha hat. Now, we made the point earlier that ideally, if we had access to the population, we could get more training data. We could simply grab more samples of training data from the population, more investments from the possible population investments. (a new sample, the same size) and from the new sample, do the same analysis, get an estimate alpha hat, and do that many times.

The bootstrap in general

Because in the schematic, we were sampling here where we assumed that the sampling was IID (=independent, identically distributed.) from the population. But in this situation, that's not a reasonable assumption. Because the observations are not independent.i.i.d.



- In more complex data situations, figuring out the appropriate way to generate bootstrap samples can require some thought.
- For example, if the data is a time series, we can't simply sample the observations with replacement (*why not?*).
- We can instead create blocks of consecutive observations, and sample those with replacements. Then we paste together sampled blocks to obtain a bootstrap dataset.

So one thing people do in a time series is to use what's called the "block bootstrap". The block bootstrap divides the data up into blocks. And between blocks, one assumes that things are independent. so for example, if we use a block size of three (3 observations in a block), these three observations would be the first block, these three would be the second block. These would be the next block. And now our sampling units are not individual observations. But they're entire blocks. So we would sample with replacement from all the blocks and then paste them together into a new time series.And that becomes the bootstrap sample to which we apply the estimate-- in this case, a regression model. So the point is you have to arrange for find the parts of the data that are uncorrelated and use that as a basis for bootstrap sampling. In this case, we use a block of size three under the assumption that beyond a time lag of three observations, things are somewhat uncorrelated. But within a block, we expect correlation. So we keep the blocks intact and sample them as units. That's called the block bootstrap.

Other uses of the bootstrap

Suppose I want a 0.9 confidence interval so that it contains a true value with probability 90%.

- Primarily used to obtain standard errors of an estimate.
- Also provides approximate confidence intervals for a population parameter. For example, looking at the histogram in the middle panel of the Figure on slide 29, the 5% and 95% quantiles of the 1000 values is $(.43, .72)$.
- This represents an approximate 90% confidence interval for the true α. *How do we interpret this confidence interval?*
- The above interval is called a *Bootstrap Percentile* confidence interval. It is the simplest method (among many approaches) for obtaining a confidence interval from the bootstrap.

What this means is that if we were to repeat this experiment from the population many times, it would be the case that the confidence interval that we obtain would contain the true value of alpha 90% of the time.

because we're trying to get an idea of test set error on new data. So cross-validation is taking a piece of data, separating validation set out from the training set so that when I fit on training data, predicted on validation data, this validation data is, in a sense, new. It hasn't been seen by the model train on the training set. So there's a separation, a clean separation, between training and validation

Can the bootstrap estimate prediction error?

no overlap between training set (K-1 folds) and validation set (k'th)

- In cross-validation, each of the K validation folds is distinct from the other $K - 1$ folds used for training: *there is no overlap*. This is crucial for its success. *Why?*
- To estimate prediction error using the bootstrap, we could think about using each bootstrap dataset as our training sample, and the original sample as our validation sample.
- But each bootstrap sample has significant overlap with the original data. About two-thirds of the original data points appear in each bootstrap sample. *Can you prove this?* ref:5.5_Review Questions.pdf
- This will cause the bootstrap to seriously $\boxed{\text{underestimate}}$ the true prediction error. *Why?*
- The other way around— with original sample = training sample, bootstrap dataset = validation sample— is worse!

Removing the overlap

- Can partly fix this problem by only using predictions for those observations that did not (by chance) occur in the current bootstrap sample.
- But the method gets complicated, and in the end, cross-validation provides a simpler, more attractive approach for estimating prediction error.

And so our general philosophy is if you can get the job done with a simple method (cross-validation), it's far better than using a more complicated method (bootstrap) just because it maybe looks sexy or whatever. So keep it simple-- that's the idea.

Pre-validation

- In microarray and other genomic studies, an important problem is to compare a predictor of disease outcome derived from a large number of “biomarkers” to standard clinical predictors.
- Comparing them on the same dataset that was used to derive the biomarker predictor can lead to results strongly biased in favor of the biomarker predictor.
- *Pre-validation* can be used to make a fairer comparison between the two sets of predictors.

Motivating example

An example of this problem arose in the paper of van't Veer *et al. Nature* (2002). Their microarray data has 4918 genes measured over 78 cases, taken from a study of breast cancer. There are 44 cases in the good prognosis group and 34 in the poor prognosis group. A "microarray" predictor was constructed as follows:

1. 70 genes were selected, having largest absolute correlation with the 78 class labels.
2. Using these 70 genes, a nearest-centroid classifier $C(x)$ was constructed.
3. Applying the classifier to the 78 microarrays gave a dichotomous predictor $z_i = C(x_i)$ for each case i.

Results

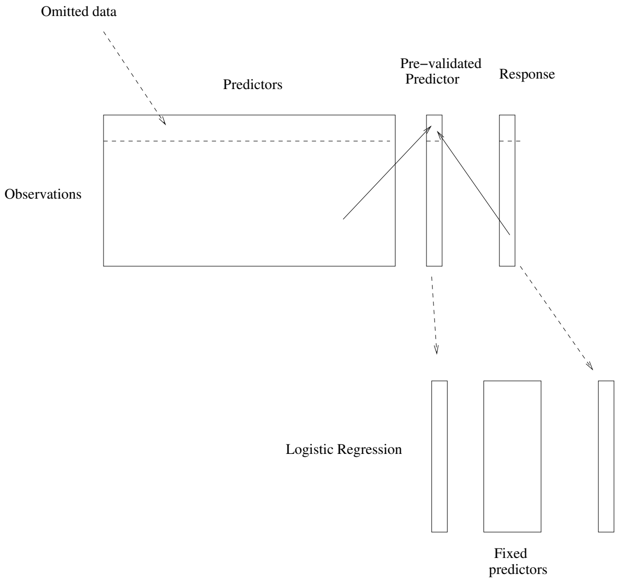
Comparison of the microarray predictor with some clinical predictors, using logistic regression with outcome `prognosis`:

Model	Coef	Stand. Err.	Z score	p-value
Re-use				
microarray	4.096	1.092	3.753	0.000
angio	1.208	0.816	1.482	0.069
er	-0.554	1.044	-0.530	0.298
grade	-0.697	1.003	-0.695	0.243
pr	1.214	1.057	1.149	0.125
age	-1.593	0.911	-1.748	0.040
size	1.483	0.732	2.026	0.021
Pre-validated				
microarray	1.549	0.675	2.296	0.011
angio	1.589	0.682	2.329	0.010
er	-0.617	0.894	-0.690	0.245
grade	0.719	0.720	0.999	0.159
pr	0.537	0.863	0.622	0.267
age	-1.471	0.701	-2.099	0.018
size	0.998	0.594	1.681	0.046

Idea behind Pre-validation

- Designed for comparison of adaptively derived predictors to fixed, pre-defined predictors.
- The idea is to form a “pre-validated” version of the adaptive predictor: specifically, a “fairer” version that hasn’t “seen” the response y.

Pre-validation process



Pre-validation in detail for this example

1. Divide the cases up into $K = 13$ equal-sized parts of 6 cases each.
2. Set aside one of parts. Using only the data from the other 12 parts, select the features having absolute correlation at least .3 with the class labels, and form a nearest centroid classification rule.
3. Use the rule to predict the class labels for the 13th part
4. Do steps 2 and 3 for each of the 13 parts, yielding a "pre-validated" microarray predictor $\tilde{z}_i$ for each of the 78 cases.
5. Fit a logistic regression model to the pre-validated microarray predictor and the 6 clinical predictors.

The Bootstrap versus Permutation tests

- The bootstrap samples from the estimated population, and uses the results to estimate standard errors and confidence intervals.
- Permutation methods sample from an estimated *null* distribution for the data, and use this to estimate p-values and False Discovery Rates for hypothesis tests.
- The bootstrap can be used to test a null hypothesis in simple situations. Eg if $\theta = 0$ is the null hypothesis, we check whether the confidence interval for θ contains zero.
- Can also adapt the bootstrap to sample from a null distribution (See Efron and Tibshirani book "An Introduction to the Bootstrap" (1993), chapter 16) but there's no real advantage over permutations.