Week 2, Lecture 4 - Does my model work? Crossvalidation, bootstrap, and friends.

Aaron Meyer

Outline

- Administrative Issues
- Model Evaluation
  - Crossvalidation
  - Bootstrap
- Example: Cancer survival gene signatures

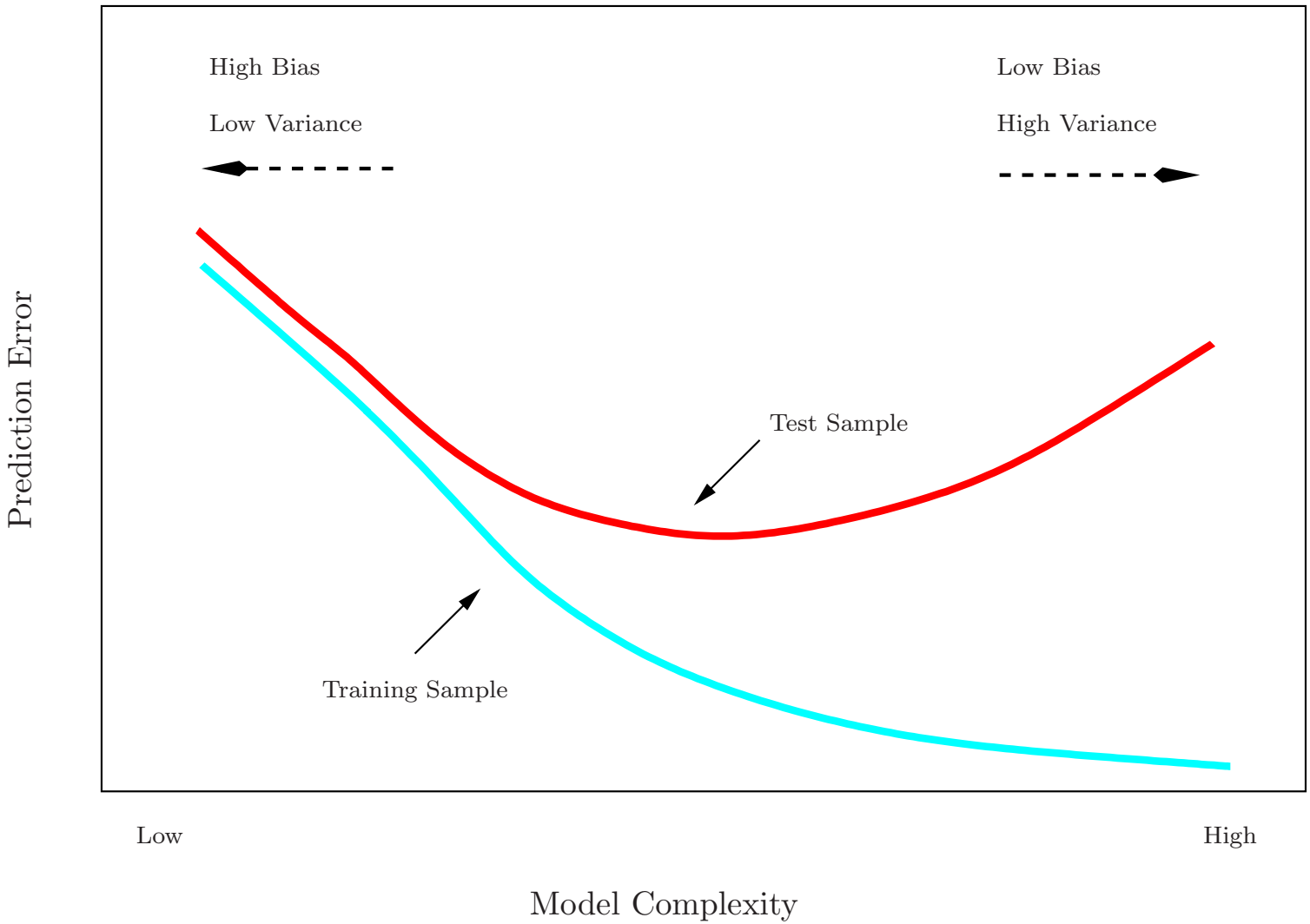
Cross-validation and the Bootstrap

- 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

Training Error vs. 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.

Training- vs. Test-Set Performance



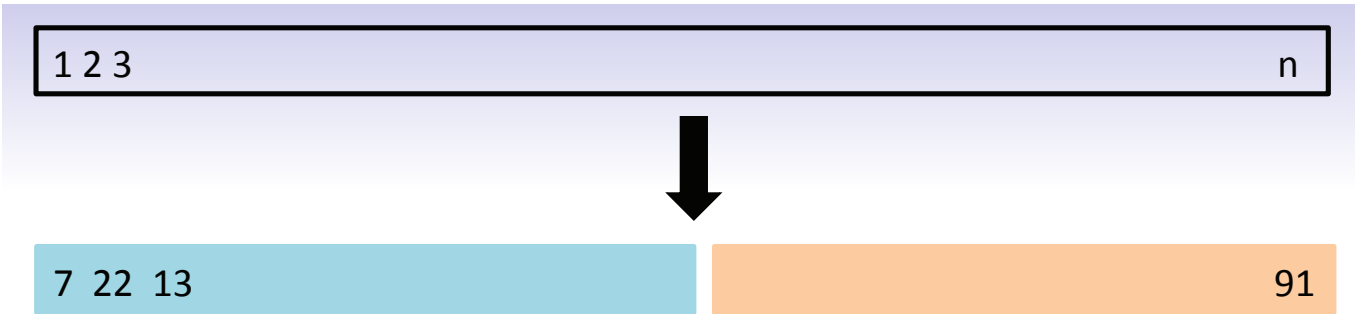
More on prediction-error estimates

- Best solution: an infinitely large designated test set. Often not available.
- 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*.
- 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

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.

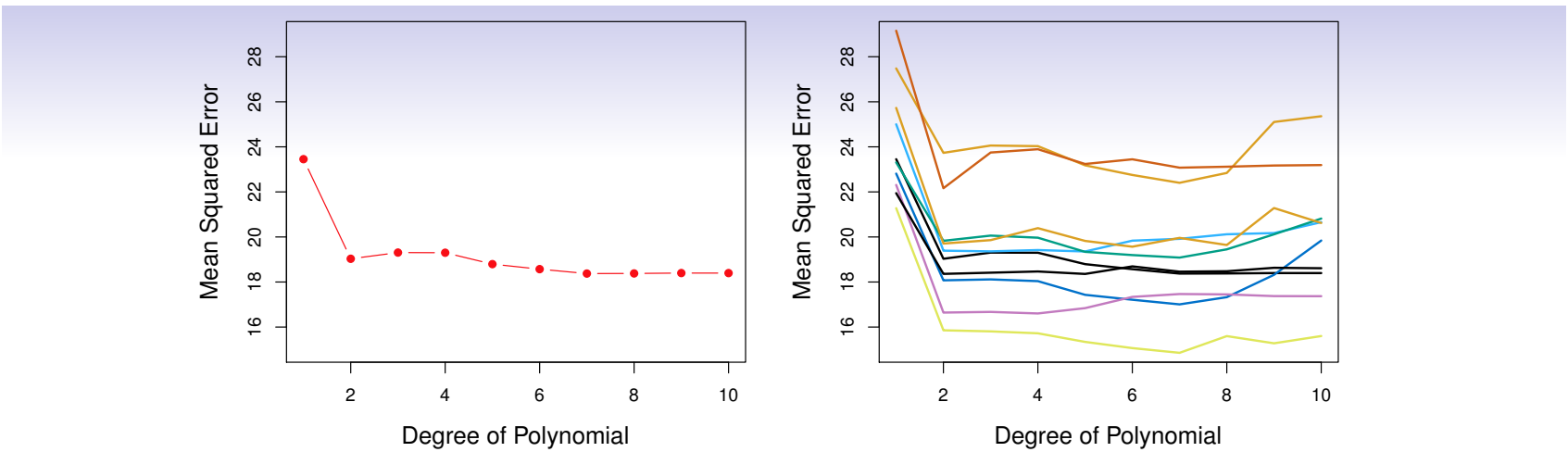
The Validation process



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

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.



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.
- This suggests that the validation set error may tend to *overestimate* the test error for the model fit on the entire data set.
  - *Why?*

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

Table 1: Divide data into K roughly equal-sized parts ($K = 5$ here)

1	2	3	4	5
Validation	Train	Train	Train	Train

The details

- Let the K parts be $C_1, C_2, \ldots C_K$, 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:
  - $$CV_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} MSE_k$$
    - Where $MSE_k = \sum_{i \in C_k} (y_i - \hat{y}_i)^2 / n_k$
    - $\hat{y}_i$ 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).

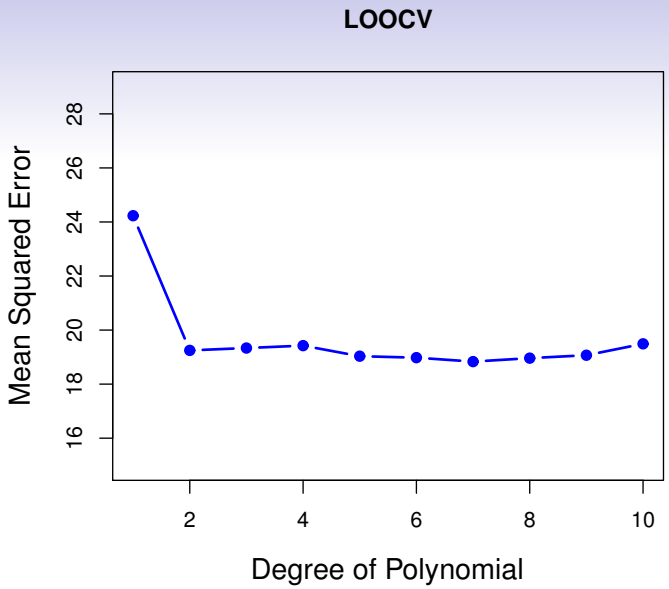
A nice special case!

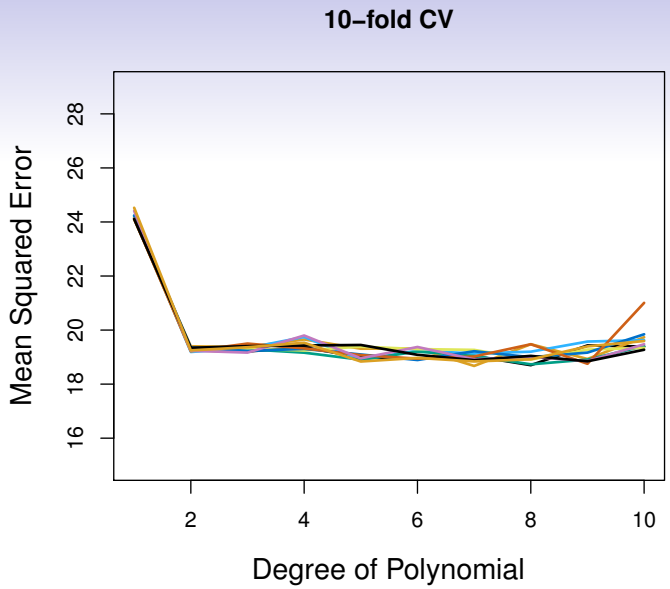
- With least-squares linear or polynomial regression, an amazing shortcut makes the cost of LOOCV the same as that of a single model fit! The following formula holds:

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

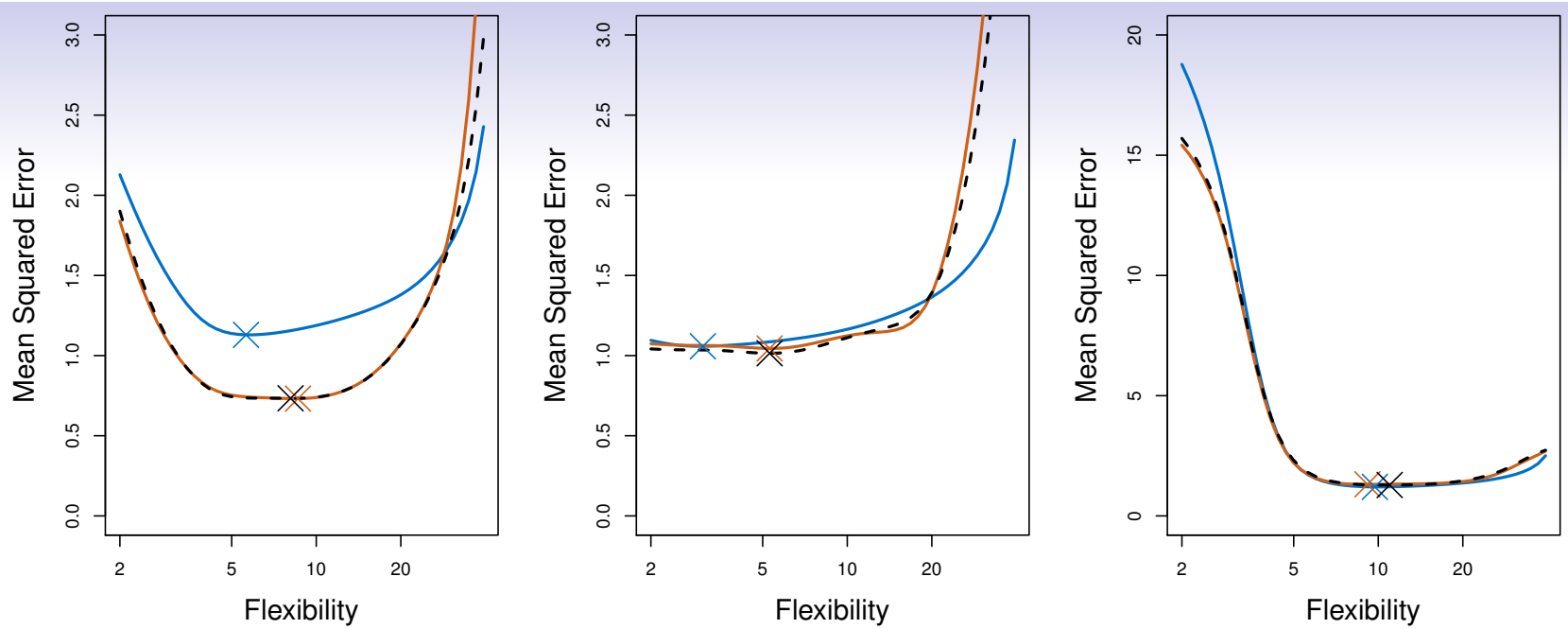
  where $\hat{y}_i$ is the ith fitted value from the original least squares fit, and h_i is the leverage (diagonal of the "hat" matrix; see book for details.) This is like the ordinary MSE, except the ith residual is divided by $1 - h_i$.
- LOOCV sometimes useful, but typically doesn't *shake up* the data 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.

Auto data revisited





True and estimated test MSE for the simulated data



Other issues with Cross-validation

- 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.
- $K = 5$ 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 $n_k = \frac{n}{K}$.
- Compute:
  - $$CV_K = \sum_{k=1}^{K} \frac{n_k}{n} Err_k$$
  - where $Err_k = \sum_{i \in C_k} I(y_i \neq \hat{y}_i)/n_k$.
- The estimated standard deviation of CV_K is
  - $$\hat{SE}(CV_K) = \sqrt{\sum_{k=1}^{K} (Err_k - \bar{Err}_k)^2/(K-1)}$$
- This is a useful estimate, but strictly speaking, not quite valid. *Why not?*

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?

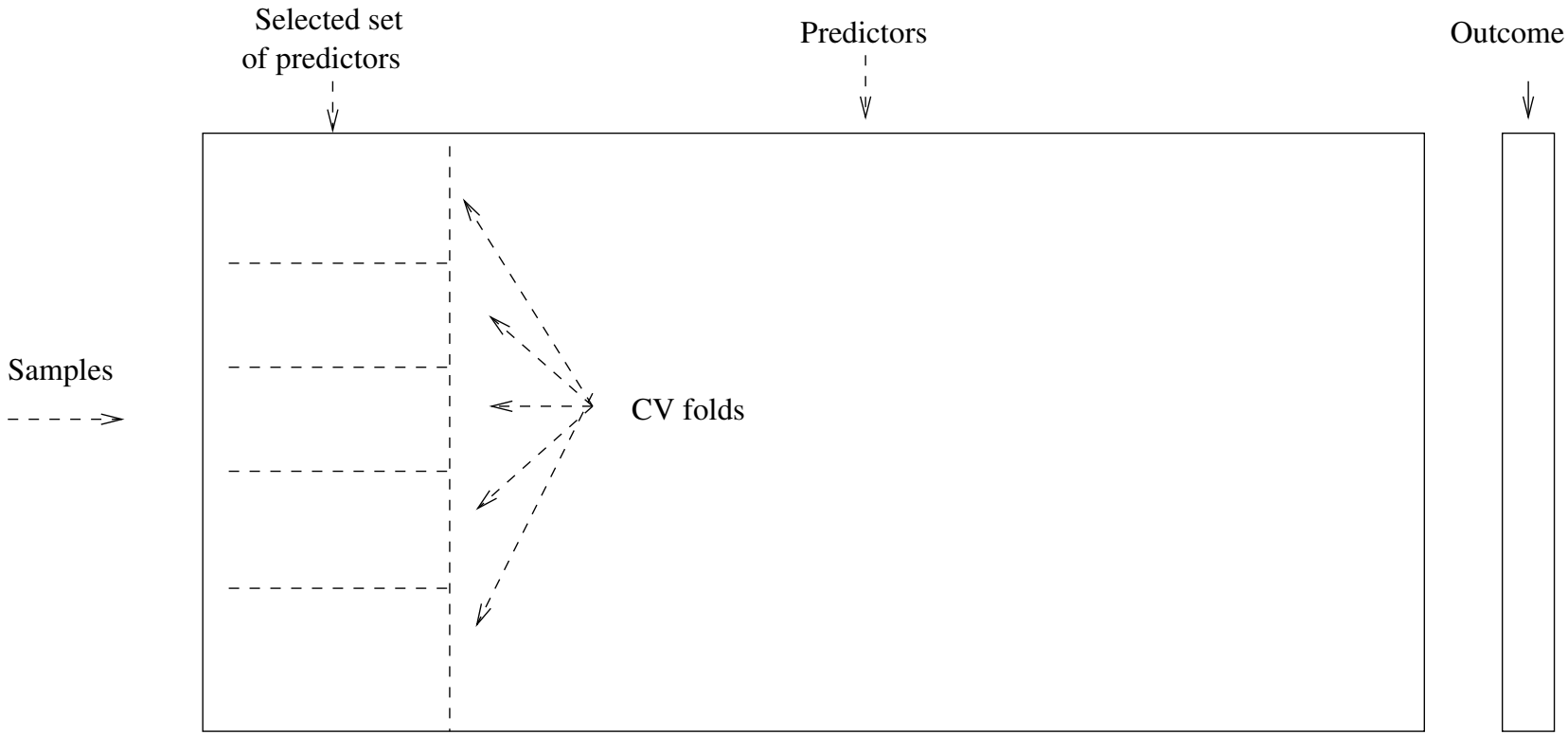
NO!

- 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!
- This error is made in *many* genomics papers.

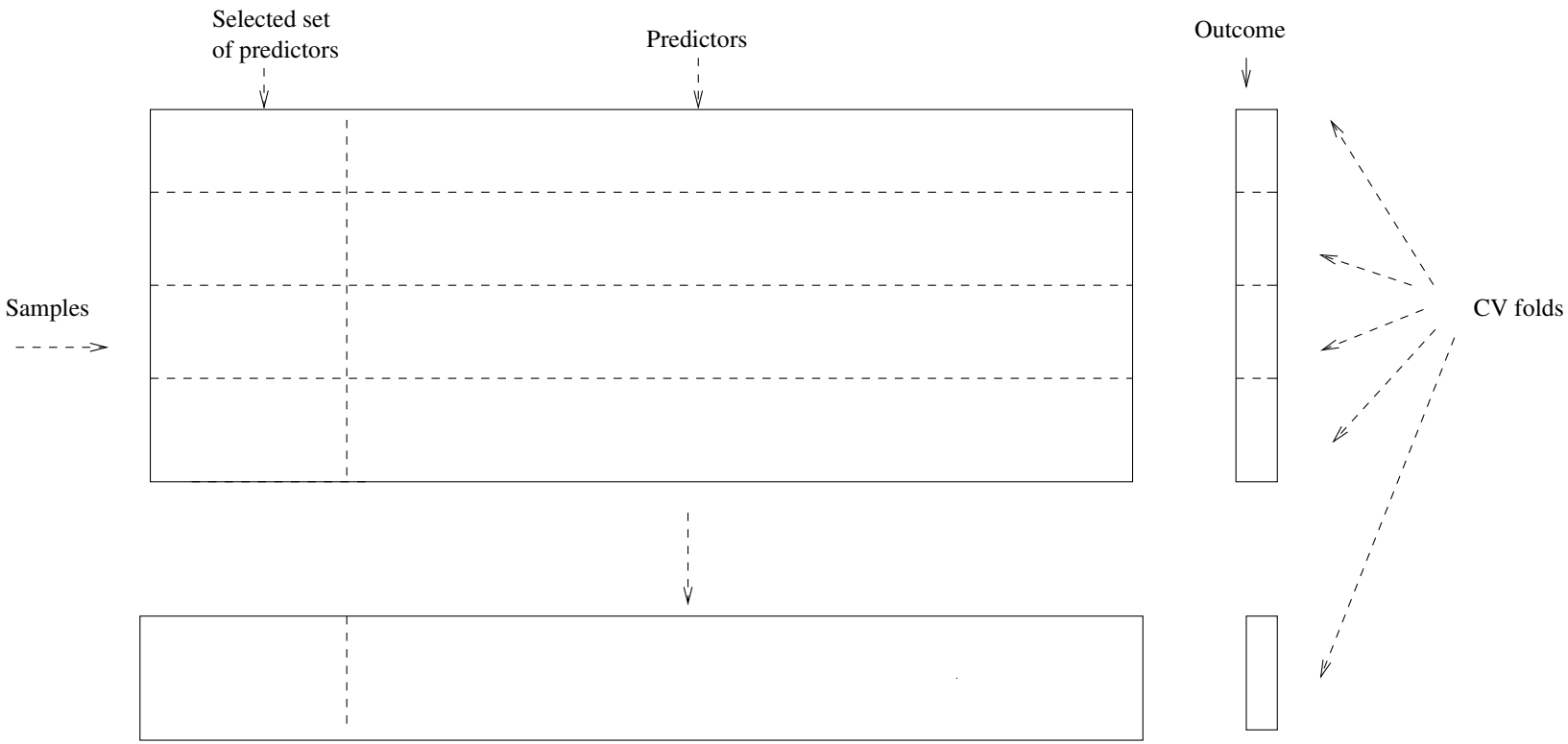
The Wrong and Right Way

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

The Wrong Way



The Right Way



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:

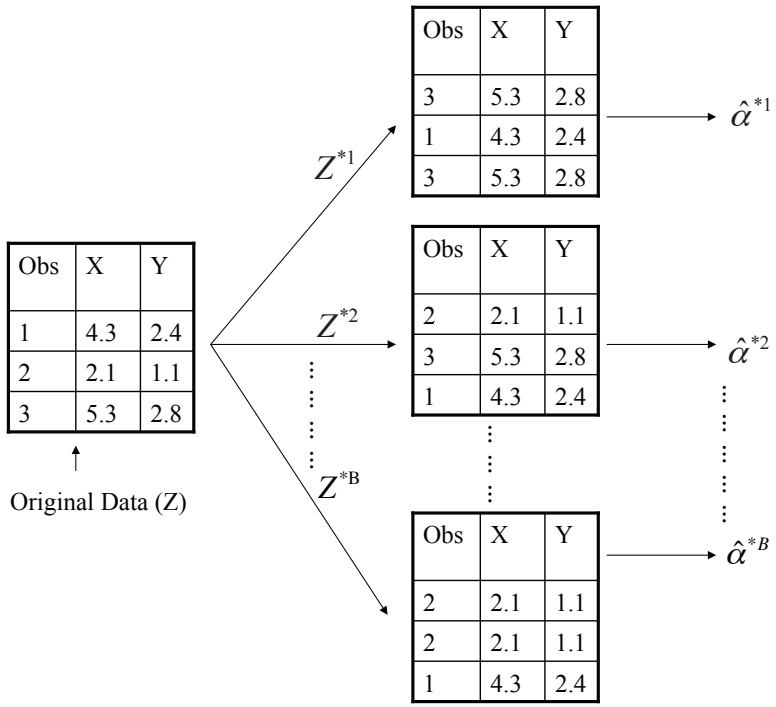
The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.

- 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.

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

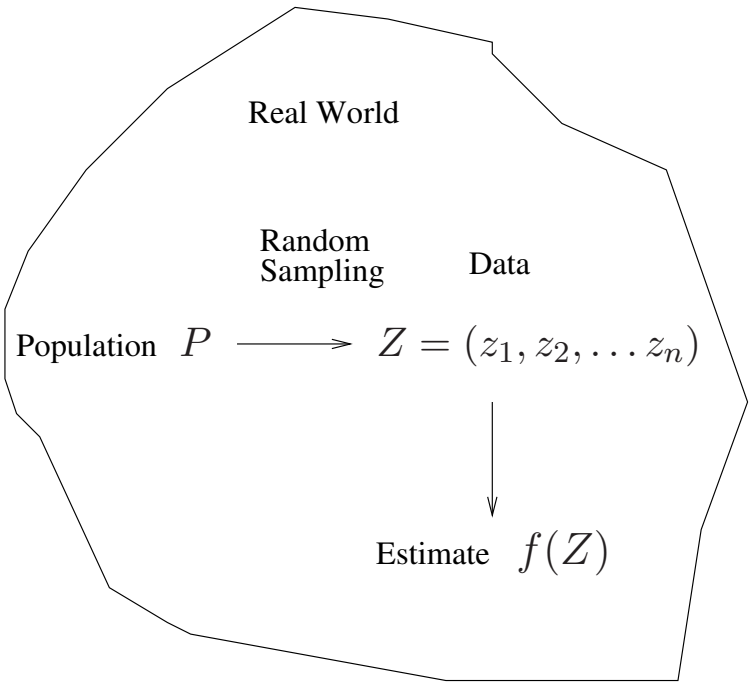
Example With Just 3 Observations

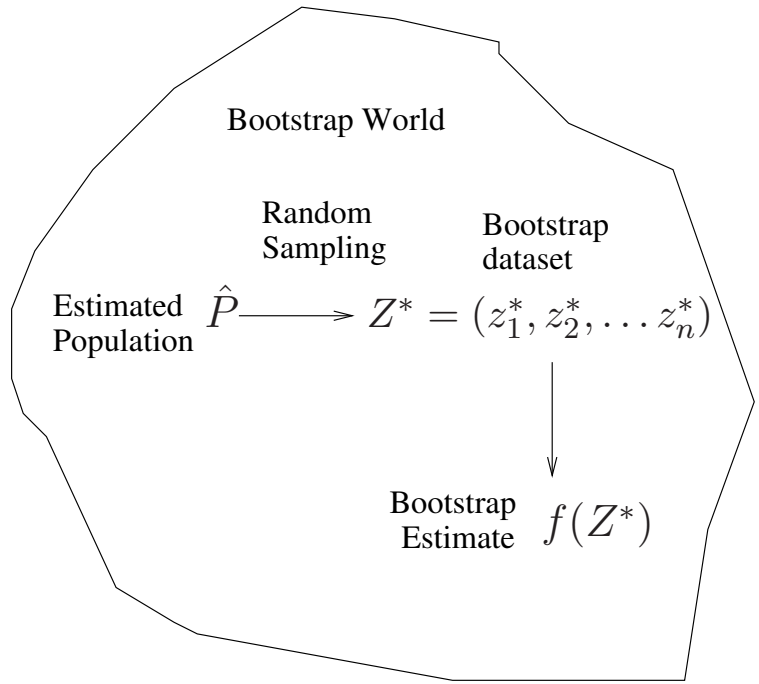
- Denoting the first bootstrap data set by Z^{*1}, we use Z^{*1} to produce a new bootstrap estimate for α, which we call $\hat{\alpha}^{*1}$
- This procedure is repeated B times for some large value of B (say 100 or 1000), in order to produce B different bootstrap data sets, $Z^{*1}, Z^{*2}, \ldots,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

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

- This serves as an estimate of the standard error of $\hat{\alpha}$ estimated from the original data set.

A general picture for the bootstrap





The bootstrap in general

- 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.

Other uses of the bootstrap

- Primarily used to obtain standard errors of an estimate.
- Also provides approximate confidence intervals for a population parameter.
- 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.

Can The Bootstrap Estimate Prediction Error?

- 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?*
- This will cause the bootstrap to seriously 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.

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.

Example - Gene Expression Signatures

OPEN ACCESS Freely available online

PLoS COMPUTATIONAL BIOLOGY

Most Random Gene Expression Signatures Are Significantly Associated with Breast Cancer Outcome

David Venet[1], Jacques E. Dumont[2], Vincent Detours[2,3]*

1 IRIDIA-CoDE, Université Libre de Bruxelles (U.L.B.), Brussels, Belgium, **2** IRIBHM, Université Libre de Bruxelles (U.L.B.), Campus Erasme, Brussels, Belgium, **3** WELBIO, Université Libre de Bruxelles (U.L.B.), Campus Erasme, Brussels, Belgium

Abstract

Bridging the gap between animal or *in vitro* models and human disease is essential in medical research. Researchers often suggest that a biological mechanism is relevant to human cancer from the statistical association of a gene expression marker (a signature) of this mechanism, that was discovered in an experimental system, with disease outcome in humans. We examined this argument for breast cancer. Surprisingly, we found that gene expression signatures—unrelated to cancer—of the effect of postprandial laughter, of mice social defeat and of skin fibroblast localization were all significantly associated with breast cancer outcome. We next compared 47 published breast cancer outcome signatures to signatures made of random genes. Twenty-eight of them (60%) were not significantly better outcome predictors than random signatures of identical size and 11 (23%) were worst predictors than the median random signature. More than 90% of random signatures >100 genes were significant outcome predictors. We next derived a metagene, called meta-PCNA, by selecting the 1% genes most positively correlated with proliferation marker PCNA in a compendium of normal tissues expression. Adjusting breast cancer expression data for meta-PCNA abrogated almost entirely the outcome association of published and random signatures. We also found that, in the absence of adjustment, the hazard ratio of outcome association of a signature strongly correlated with meta-PCNA ($R^2 = 0.9$). This relation also applied to single-gene expression markers. Moreover, $>50\%$ of the breast cancer transcriptome was correlated with meta-PCNA. A corollary was that purging cell cycle genes out of a signature failed to rule out the confounding effect of proliferation. Hence, it is questionable to suggest that a mechanism is relevant to human breast cancer from the finding that a gene expression marker for this mechanism predicts human breast cancer outcome, because most markers do. The methods we present help to overcome this problem.

Citation: Venet D, Dumont JE, Detours V (2011) Most Random Gene Expression Signatures Are Significantly Associated with Breast Cancer Outcome. PLoS Comput Biol 7(10): e1002240. doi:10.1371/journal.pcbi.1002240

Editor: Isidore Rigoutsos, Jefferson Medical College/Thomas Jefferson University, United States of America

Received April 27, 2011; **Accepted** September 7, 2011; **Published** October 20, 2011

Funding: DV was funded by the IRSIB Brussels Region-Capitale ICT-Impulse 2006 program 'InSilico wet lab'. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: vdetours@ulb.ac.be

Example - Gene Expression Signatures

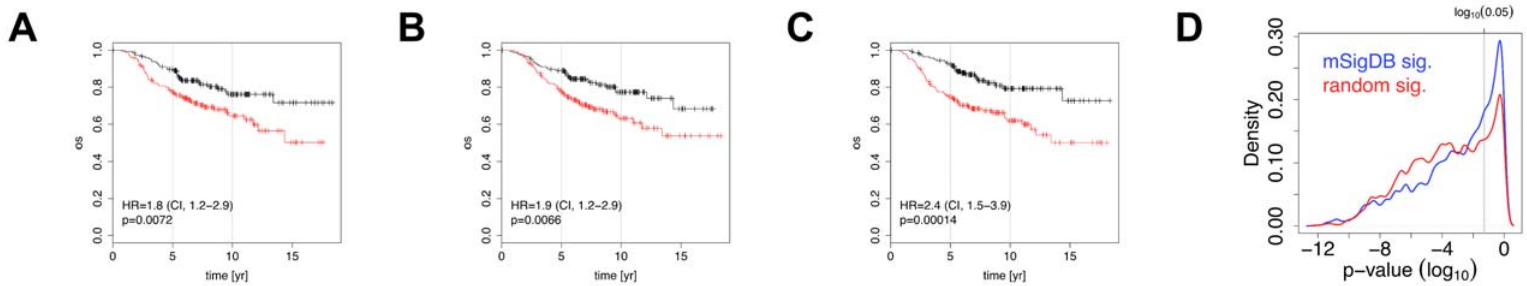


Figure 1. Association of negative control signatures with overall survival. In plots A–C the NKI cohort was split into two groups using a signature of post-prandial laughter (panel A), localization of skin fibroblasts (panel B), social defeat in mice (panel C). In panels A–C, the fraction of patients alive (overall survival, OS) is shown as a function of time for both groups. Hazard ratios (HR) between groups and their associated p-values are given in bottom-left corners. Panel D depicts p-values for association with outcome for all MSigDB c2 signatures and random signatures of identical size as MSigDB c2 signatures.
doi:10.1371/journal.pcbi.1002240.g001

Example - Gene Expression Signatures

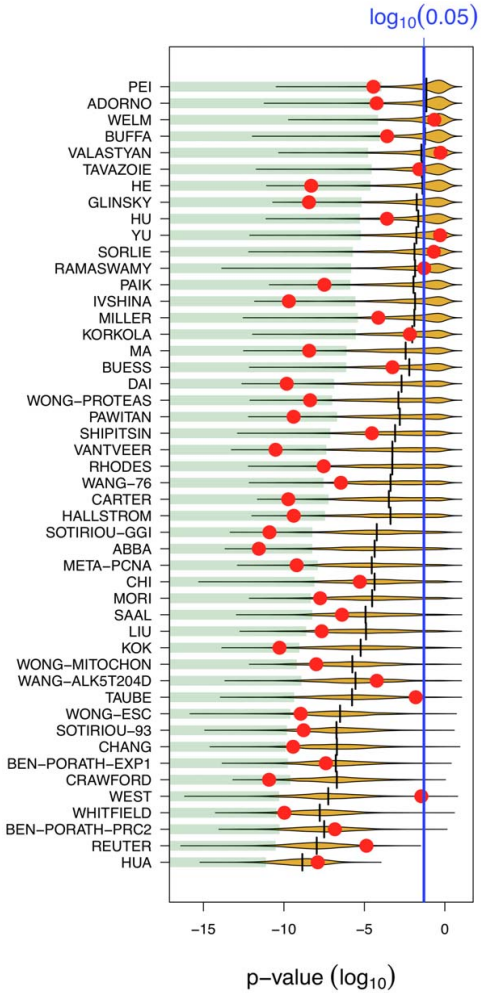


Figure 2. Most published signatures are not significantly better outcome predictors than random signatures of identical size. The x-axis denotes the p-value of association with overall survival. Red dots stand for published signatures, yellow shapes depict the distribution of p-values for 1000 random signatures of identical size, with the lower 5% quantiles shaded in green and the median shown as black line. Signatures are ordered by increasing sizes.
doi:10.1371/journal.pcbi.1002240.g002

Example - Gene Expression Signatures

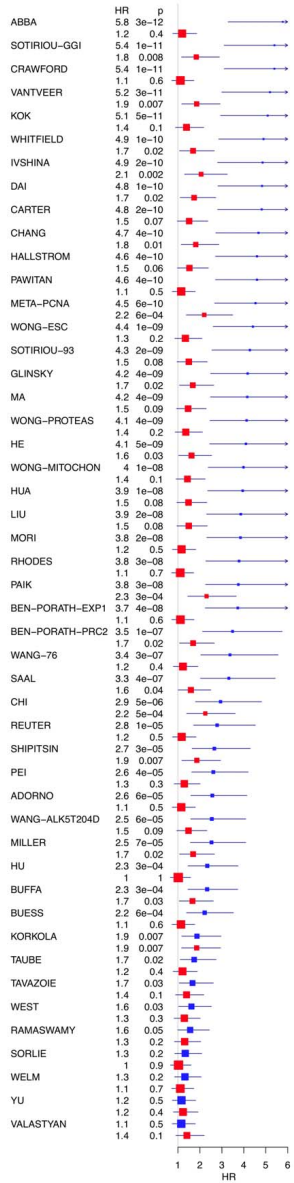


Figure: **Meta-PCNA adjustment decreases the prognostic abilities of published signatures.** Hazard ratios for overall survival association of

Example - Gene Expression Signatures

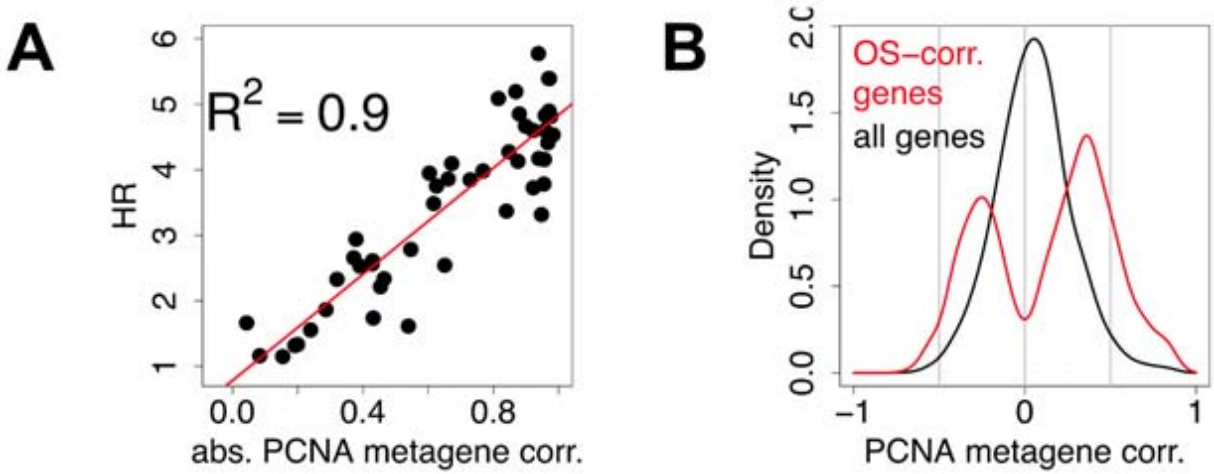


Figure 4. Most prognostic transcriptional signals are correlated with meta-PCNA. A) Each point denotes a signature. The x-axis depicts the absolute value of the correlation of the first principal component of the signatures with meta-PCNA, the y-axis depicts the hazard ratio for outcome association. Details of the analysis for each data point are available in the Supporting Information (Text S1). B) Distribution of the correlations of individual genes with meta-PCNA, for genes significantly associated with overall survival (red) and for all the genes spotted on the microarrays (black).
doi:10.1371/journal.pcbi.1002240.g004

Example - Gene Expression Signatures

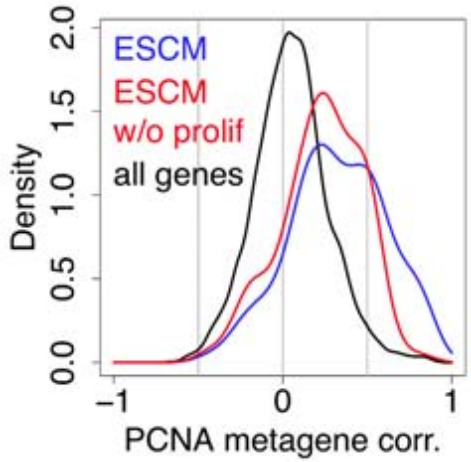


Figure 5. Purging cell cycle genes from a signature does not rule out proliferation signals. Distribution of the correlations with meta-PCNA of genes in the Embryonic Stem Cell Module (blue, ref. [15]), of the correlations of the same module with its cell cycle genes removed (red) and of all of the genes spotted on the microarray (black). doi:10.1371/journal.pcbi.1002240.g005

Implementation - Easiest

`sklearn.model_selection.cross_val_score`

- `estimator`: estimator object implementing 'fit'
- `X`: array-like
- `y`: array-like, optional, default: None
- `groups`: array-like, with shape (n_samples,), optional
- `scoring`: string, callable or None, optional, default: None
- `cv`: int, cross-validation generator or an iterable, optional
- `n_jobs`: integer, optional

Implementation - Iterators

- `sklearn.model_selection.KFold(n_splits=3, shuffle=False, random_state=None)`
- `sklearn.model_selection.LeaveOneOut()`

Both use loop `for train_index, test_index in kf.split(X):`.

`get_n_splits` provides number of iterations that will occur.

Implementation - Bootstrap

```
for bootstrapi in range(num_bootstraps):
    X_index = range(X.shape[0])
    resamp = resample(X_index, random_state=9889)

    ycurr = y[resamp]
    Xcurr = X[resamp]
    # ...
```

Summary

- Randomization and hiding things are the key to success!
  - Crossvalidation hides parts of the data at each step, to see how the model can predict it.
  - Bootstrap generates "new" data by resampling, to get a distribution of models.
- With all model evaluation, think about what your model should be "learning", and mess with that.