Sample Statistical Workflow:
Model Building

Adam Davey

March 9, 2015

> You never change things by fighting the existing reality. To change something, build a new model that makes the existing model obsolete.
>
> Buckminster Fuller

> Pluralitas non est ponenda sine necessitate.
>
> John Duns Scotus

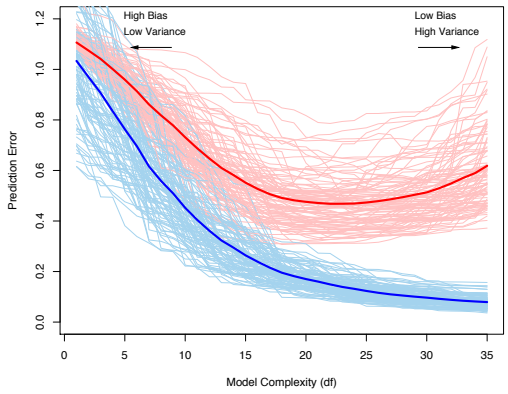
1 Preamble

Based on what we have seen so far, it is clear that there is no single formula or approach that will lead us directly from our observed data to the "correct" model to understand them. The process of model building involves both conceptual and empirical considerations, and represents an area of fairly rapid development. The guidelines presented here represent an overview of some of the most commonly applied approaches to model building in multivariable biostatistics, along with some examples of emerging methods.

2 Bias-Variance Tradeoff

All statistical methods are designed to optimize parameter estimates within a specific observed data set, ideally with some bounds around those estimates to represent the range of our uncertainty about those estimates. We recognize, however, that any observed data set represents just one possible outcome. This introduces a disconnection between the estimation process (made within our specific sample) and the generalization we wish to

make (i.e., out of sample). What this means in practice is that estimates made *within* our sample always favor more complex models. We can always improve prediction within sample by adding more variables. At the same time, models that are overly complex are less likely to replicate and generalize *outside* of our estimation sample.



3 Theoretical Approaches to Model Building

No amount of statistical wizardry can substitute from careful thought and domain knowledge in the model building process. In some disciplines, the empirical approaches to model building described in the next section are widely discouraged, with only theoretically-derived approaches being seen as acceptable.

3.1 Hierarchical Regression

Not to be confused with hierarchical linear models, hierarchical model building in mlutiple linear regression refers to the process of adding variables to a model in conceptually distinct blocks. For example, a baseline model might include the predictor of interest (POI) and a number of background or demographic characteristics. To this, various blocks of potentially "explanatory" variables might be added to the model, such as relating to socioeconomic resources, comorbid conditions, or health care characteristics. The aim of such models is to understand how the coefficient associated with the POI changes as a function of what other variables are included in the model. One common aim is to "explain away" a significant coefficient.

In hierarchical regression, what is tested is the change in R^2 (i.e., ΔR^2). The change in R^2 is evaluated using a partial F-test $\frac{\left(SS_{Reg}(Full) - SS_{Reg}(Reduced)\right)/(\Delta df_{Numerator})}{MS_{Resid}(Full)}$. A significant change in R^2 tells you that addition of that block of variables significantly increases (within-sample) prediction of the response variable. These tests can be obtained in Stata using the `nestreg` command, i.e., `nestreg: regress y (x1 x2 x3) (x4 x5) (x6-x10)`.

4 Empirical Approaches to Model Building

A number of empirical approaches to model building are widely used in biostatistics. These include best subset regression and a number of stepwise methods.

4.1 Best Subset

Best subset regression is a brute-force method for model building and comparison. Starting from a set of k potential predictor variables, the best subset approach to model building estimates all possible models containing between 0 and k predictors, i.e., all variables individually, all pairs of variables, ..., all k variables together. For each number of predictors, the "best" model is recorded. Once all possible sets of variables has been estimated, the overall best performing model is returned.

Best subsets regression requires that the statistic used for evaluation includes some kind of penalty or adjustment for model parsimony (i.e., $AdjR^2$ instead of R^2. Also, k must be a small set of variables or else the set of all potential models quickly becomes too large to estimate even on extremely fast computers. Best subset regression with $k = 30$ requires estimation of more than 1 billion models, for example, with every 10 variables adding roughly 3 orders of magnitude to the number of models that must be estimated.

In Stata, the easiest way to estimate best subsets regression models is via the `tryem` command that can be installed via `findit tryem`.

4.2 Stepwise Methods

Stepwise methods address the process of selectively adding and removing variables from a model according to some criterion. *Forward Selection*, for example, begins with a baseline (e.g., intercept only) model and adds variables, one at a time, as long as they enter the model with a p-value at or below some specified threshold (e.g., $p < .05$). While this guarantees that all variables will have a significant p-value when they are added, it does not provide any guarantee that the variable will remain significant as more covariates are added to the model. In Stata, forward stepwise is indicated by setting a probability for a variable to *enter* the model (e.g., `sw, pe(.05): reg y x1-x10`).

A second stepwise approach is *Backward Elimination*. This approach begins with all variables entered into the model and systematically removes them, one at a time, as long as their p-value is above some threshold (e.g., $p > .05$). As with forward selection, this in no way guarantees that the predictor removed would not have been statistically significant in a model with a different subset of covariates; only that all variables remaining in the model are significant at some criterion. Because this model automatically adjusts for all other covariates included in the model, it is often preferred in biostatistics. In Stata, backward elimination is indicated by setting a probability for a variable to be *removed* the model (e.g., `sw, pr(.05): reg y x1-x10`).

Bridging these two methods, a *Hybrid* model can, at each step, decide whether is is better to remove an existing variable from a model or to add in a variable that is currently excluded from the model. In this way, variables can enter or exit the model at various points during the estimation process. There is still no guarantee, however, that the final selected model will be best overall. In Stata, this hybrid approach can be obtained by specifying both a probability to enter and a (larger) probability to remove a variable (e.g., `sw, pe(.05) pr(.1): reg y x1-x10`).

One problem with each of these approaches is that p-values for model coefficients are no longer statistically valid, since the same parameters are often estimated multiple times. There is no current consensus on how to adjust for this fact, however, and so the issue is most often ignored.

When each of these approaches is applied to a real data situation, a subset of covariates is typically "forced" to be included. In this way, all models can adjust for a specific set of variables, regardless of their level of statistical significance. To force a set of variables to be included using Stata, the `lockterm1` syntax is used (e.g., `sw, lockterm1 pr(.10): reg y (x1 x2) x3-x10` would require variables x1 and x2 to be included in all models). This is useful when estimates should always adjust for certain key covariates and represents a link between theoretical and empirical approaches to model building.

4.3 Shrinkage Methods

In recent years, there has been growing interest in so-called "shrinkage" methods that shrink model coefficients toward 0 by imposing an additional penalty on their magnitude. *Ridge regression* shrinks coefficients toward 0, and *lasso regression* shrinks some coefficients to exactly 0 (i.e., they are excluded from the model). In both cases, the extent of shrinkage is controlled through an additional parameter, λ, the shrinkage parameter.

For ridge regression, $\hat{beta}$ is defined as the value of β minimizing $\sum_{i}^{N}(y_i - x_i^T\beta)^2 + \lambda \sum_{j=1}^{p} \beta^2$. Ridge regression can be estimated in Stata using the `ridgereg` command.

Lasso regression uses a slightly different penalty. Specifically, $\hat{beta}$ is defined as the value of β minimizing $\sum_{i}^{N}(y_i - x_i^T\beta)^2 + \lambda \sum_{j=1}^{p} |\beta|$. Lasso regression can be estimated in Stata using the `lars` command.

5 Cross-Validation

In an ideal world, we would have enough data to independently develop, test, and validate our models. In reality, this is almost never the case. Instead, it is becoming more common to divide a data set into distinct parts in order to approximate this process under practical conditions. k-fold cross-validation involves dividing data into k approximately equal parts.

One fold is excluded at a time, and the model is estimated using the remaining $k-1$ folds. The model is evaluated using the excluded fold. The next fold is excluded, and the process is repeated until each fold has been used for independent evaluation of a model. Since the portion of the data used to estimate the model is independent of the portion of the data used to evaluate it, this reduces some of the capitalization on chance that would affect model results if a model were estimated and evaluated on the entire data set. However, each successful estimate overlaps on $k-2$ folds, so the modeling stages are not truly independent. Many times, this process is repeated for a large number of random folds (i.e., 5-fold cross-validation might be repeated using 100 different random partitions of the data into 5 folds). Ideally, a portion of the data set is also excluded from all of these considerations. It is used only at the end in order to validate the model using a truly independent data set (which is used only once with the final model). In Stata, cross-validation can be estimated using the `crossfold` command. `Jackknife Replication` is equivalent to k-fold cross validation where $k = N - 1$. That is, each observation is excluded one at a time.

Stata Syntax: Model Building and Variable Selection

```
/*****************************************************************
* Model Building and Variable Selection
* PH 8012, Spring 2015
* Adam Davey
* Requires tryem
* Type: findit tryem to install
*****************************************************************/

#delimit;
clear all;
capture log close;
log using "mymodel.log", replace;
*log using "mymodel.smcl", replace;

* Below simulates some data for the workflow;
set seed 31159;
set obs 1000;
local numx = 10;
local numxr = 5;
local evar = 5;
local maxk = 5;

forvalues i=1/‘numx’ {;
 generate x_‘i’ = rnormal();
};

gen y = 0;
forvalues i= 1/‘numxr’ {;
replace y = y + x_‘i’;
};

replace y = y + rnormal(0,‘evar’);

forvalues k=1/‘maxk’ {;
*tryem y x*, k(‘k’) cmd(reg) best(max) stat(r2_a);
tryem y x*, k(‘k’) cmd(reg) best(min) stat(rmse);
};

sw, pe(.1): reg y x*;
sw, pr(.1): reg y x*;
sw, pe(.10) pr(.100001): reg y x*;

gen order = .;
```

```
gen yhat = .;
gen fold = .;

local reps = 2;
local nfolds = 5;
quietly {;
timer on 1;
forvalues r = 1/‘reps’ {;
replace order = runiform();
sort order;
replace fold = mod(_n,‘nfolds’) + 1;
forvalues i=1/‘numx’ {;
 forvalues k = 1/‘nfolds’ {;
  tryem y x* if fold!=‘k’, k(‘i’) best(min) stat(rmse);
  predict e, resid;
  replace e = sqrt(e^2);
  summ e if fold!=‘k’, meanonly;
  local insamp = r(mean);
  summ e if fold==‘k’, meanonly;
  local outsamp = r(mean);
  *noisily: di "i = " %2.0f ‘i’ ", fold = " %2.0f ‘k’ ", RMSE = " %6.4f r(mean);
  mat results = (nullmat(results) \ ‘i’, ‘k’, ‘insamp’, ‘outsamp’);
  drop e;
 };
};
};
timer off 1;
svmat results;
ren results1 i;
ren results2 k;
ren results3 insamp;
ren results4 outsamp;
noisily: di "Results based on ‘reps’ replcations";
noisily: tabstat insamp outsamp, by(i) statistics(mean) longstub nototal;
noisily: timer list 1;
by i, sort: egen insample = mean(insamp);
by i, sort: egen outsample = mean(outsamp);
twoway (line insample i, sort lwidth(thick)) (line outsample i, sort lwidth(thick)),
 ytitle(RMSE) xtitle(Number of Predictors) legend(on position(1) ring(0));
};

crossfold reg y x*, k(10) mae;

lars y x*, algorithm(lasso) graph;
```