

Evaluating Predictive Models (Part 2): Assessing Whether More Complex Models Predict Better

Sigurd W. Hermansen, Westat, Rockville, MD, USA

ABSTRACT

Recent findings outside the mainstream of data mining and Business Intelligence (BI) suggest novel strategies for improving predictive models. In this presentation we initially sweep through a spectrum of methods for automated selection of predictors using datasets with known outcomes, including deprecated step-wise selection; preferred stage-wise parameter estimate shrinkage methods such as LASSO in GLMSELECT; and, CART and CHAID alternatives to parametric models. We then go beyond the ROC AUC c statistic (introduced in Part 1) to explore predictor-conditional (as opposed to outcome-conditional) methods for evaluating predictive models; the role of loss functions that embed costs both of collecting predictor data and of false-positive and false-negative classification errors; and, decisions based on predicted odds of events. Throughout we focus on basic comparisons of predictive accuracy and reliability of models with different degrees of complexity. SAS® System programs illustrate how to compare models used in applications that range from medical screening with diagnostic testing to market segmentation.

KEYWORDS

predictive model specification step-wise stage-wise shrinkage GLMSelect LASSO CART ROC AUC concordance statistic loss function complexity

INTRODUCTION

In Part 1 of Evaluating Predictive Models, we presented the concordance c statistic, a basic measure of discriminatory accuracy of predictions and their related observed outcomes. We also explored bootstrap methods for assessing the reliability of a predictive model. Part 1 emphasized evaluation of one model in one context. In that context we are asking whether a model has sufficient discriminatory accuracy.

In Part 2 we introduce a different context. For a target population a model predicts outcomes using parameters of alternative models that we estimate by observing a sample relation (e.g., Table 1, adapted by the SAS Institute from Lee, 1974).

Table 1. Sample Relation of Predictors and Remiss Outcome

remiss	cell	smear	infil	li	blast	temp	pred1	pred2	x
0	0.2	0.4	0.1	0.8	0.1	0.99	0	1	1
0	0.5	0.4	0.2	0.6	0.1	0.99	0	1	1
0	0.7	0.4	0.3	0.5	0.1	1.01	0	0	0
0	0.7	0.8	0.5	1.2	0.1	0.98	0	0	0
0	0.8	0.5	0.4	0.4	0.4	1.01	1	0	1
0	0.8	0.9	0.7	0.8	0.2	0.98	1	1	0
0	0.9	0.4	0.3	0.7	0.3	0.99	1	0	0

remiss	cell	smear	infil	li	blast	temp	pred1	pred2	x
0	0.9	0.9	0.8	0.6	1.6	1.02	1	1	1
0	1.0	0.3	0.3	0.4	0.2	1.01	0	0	0
0	1.0	0.3	0.3	1.6	0.9	0.99	0	0	0
0	1.0	0.4	0.3	0.5	0.0	1.04	0	0	1
0	1.0	0.5	0.5	0.8	0.3	1.00	1	1	0
0	1.0	0.7	0.7	0.6	0.5	0.98	1	1	0
0	1.0	0.7	0.7	0.7	0.4	0.99	0	0	0
0	1.0	0.8	0.8	1.0	1.3	1.00	1	1	0
0	1.0	0.9	0.8	1.9	1.4	1.02	1	0	1
0	1.0	0.9	0.9	0.7	1.1	0.99	0	0	0
0	1.0	0.9	0.9	1.1	1.0	0.99	1	1	1
1	0.8	0.8	0.7	1.9	1.1	1.00	1	1	0
1	0.9	0.4	0.3	1.4	0.7	0.99	0	1	1
1	0.9	0.8	0.7	1.3	0.5	0.98	0	0	0
1	1.0	0.6	0.6	1.0	0.5	1.00	0	0	0
1	1.0	0.6	0.6	1.1	1.1	0.99	1	0	1
1	1.0	0.6	0.6	1.7	1.0	0.99	1	1	0
1	1.0	0.7	0.7	0.9	0.4	0.99	1	0	1
1	1.0	0.8	0.8	1.9	2.1	1.02	0	1	1
1	1.0	1.0	0.9	1.0	1.2	0.99	1	1	0

A sample relation consists of tuples (rows) of predictor values and a related outcome value. An ideal model would operate on a sample relation among predictors and an outcome. It would define a function f that maps each distinct combination of predictor values to one and only one outcome:

$$outcome = f(pred1, pred2, \dots) \quad (1)$$

In this new context, an ideal predictive model will somehow ignore peculiarities of any one sample relation, and yet will still divine the essential features of the underlying population. Table 2 illustrates one such ideal function of a model.

Table 2: An Ideal Function of a Model

pred1	pred2	remiss	count
0	0	0	9
1	1	0	9
0	1	1	4
1	0	1	5

In the less ideal real world, a mapping of distinct predictor values to outcomes only approximates a function of the model. Consider, for example, the sample relation (excluding pred1, pred2, and x) represented in Table 1. Here we suppose for the moment that content experts see these attributes as possible predictors of remission. . To what extent can one or more of the attributes (cell, smear, infil, li, blast, temp) predict the outcome remiss (remission)?

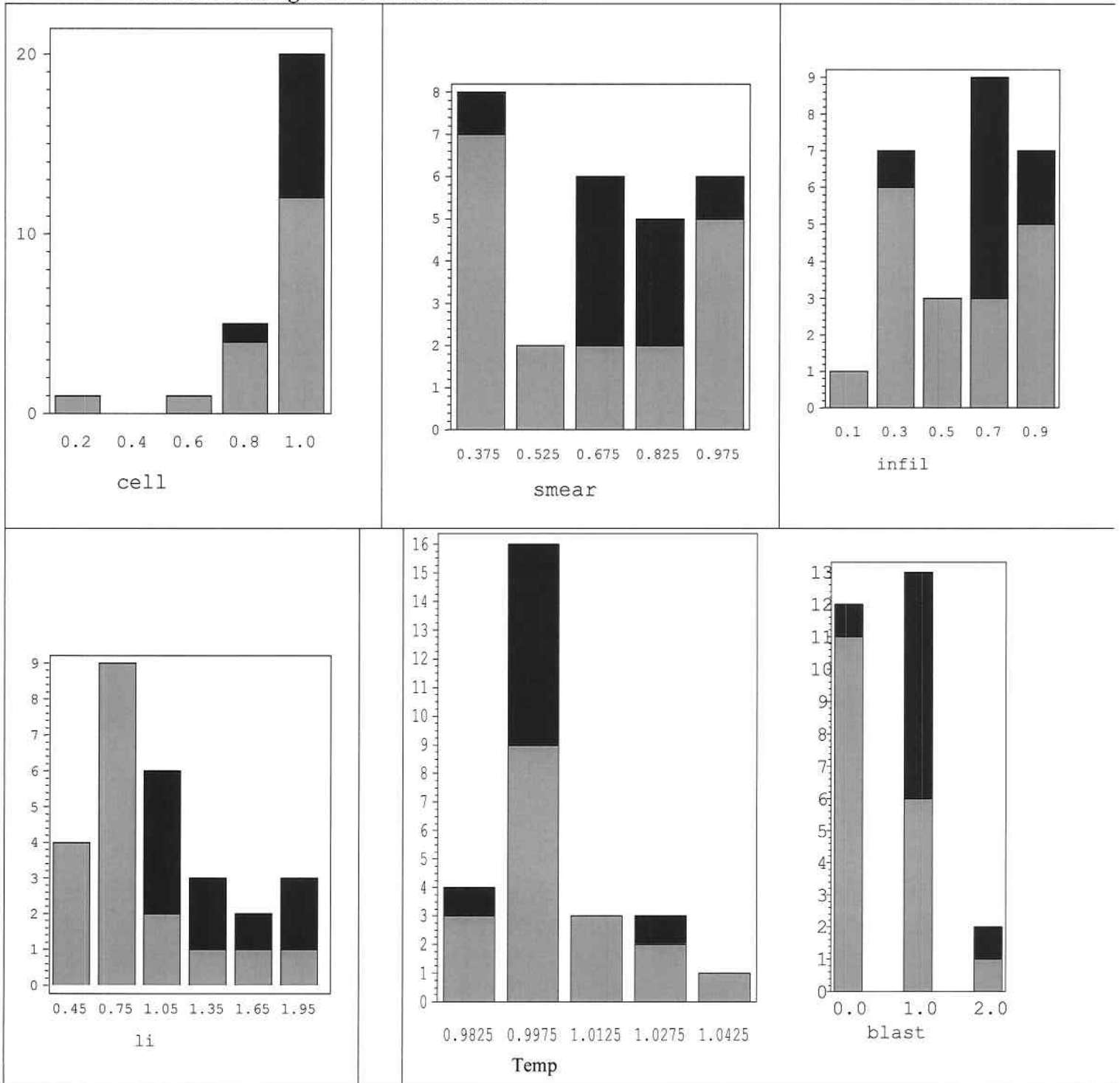
While again recognizing that we are observing a sample relation only, we look for clues in the distribution of the attributes. Beginning with basic histograms of individual attributes, this SAS program computes summaries of attributes from data in Table 1. A display of distributions of summarized attributes' values appears in Exhibit 1.

SAS Summarization and Histogram Program

```
proc sql;
  create view vwrelation as
  select round(cell,.1) as cell, round(smear,.1) as smear, round(infil,.1) as infil, round(li,.1) as li,
    round(blast,1.) as blast, round(temp,.01) as temp, remiss,count(*) as count
  from Remission
  group by calculated cell,calculated smear, calculated infil,calculated li,calculated blast, calculated
temp,remiss
  order by remiss,cell,smear,infil,li,blast,temp;
quit;
ods rtf body="C:\grafs.rtf";
%macro vbars(varlist,dv);
%let __n=1;
%let __var=%scan(&varlist,&__n);
%do %while (&__var ^=);
  proc gchart data=vwrelation;
    vbar &__var/ subgroup=&dv nolegend ;
  run;
  %let __n=%eval(&__n + 1);
  %let __var=%scan(&varlist,&__n);
%end;
%mend vbars;
%vbars(%str(cell smear infil li blast temp),remiss)
ods rtf close;
```

What shows up in the histograms that approximates a function of potential predictors and the remiss outcome? A quick glance doesn't tell us much. (If obvious, we wouldn't have an interesting predictive modeling problem!) In the **Better Predictions** section that follows, we consider how to evaluate alternatives to an ideal model and select useful ones. We consider simple and obvious ways to select those that offer better predictive accuracy when applied to data other than the sample used to estimate the model. Next, the **Controlling Model Complexity** section looks at complexity as an asset and liability, then at model optimism. In a final section on **Better Model Evaluation Measures**, we consider how accuracy and reliability diagnostics fit into medical, business, and policy decisions.

Exhibit 1: Histograms of Attribute Values



Legend: Remiss No =



Yes =



BETTER PREDICTIONS

Developers of predictive models come from a wide variety of academic and professional disciplines, and they bring with them a wide a range of analytic methods and tools. One large camp sees predictive models as extensions of quantitative sciences. Models act as abstract representations of cause and effect relations among real objects and events. The ability to interpret the internal workings of a model has special importance. Another large camp sees predictive models more as observed associations among data values and outcome values. If a so-called “black box” model predicts well, it doesn’t matter whether anyone understands why.

Each camp has its own merits. It makes sense, for example, to add important known and theoretically relevant predictors to a model.

Adding Relevant Predictors

Suppose a predictor x distinguishes males ($=0$) from females ($=1$), and that we know that males tend to have higher values of li than do females. Including x along with li in a predictive model means that in the sample relation in Table 1, the model $remiss=f(li,x)$ better approximates a function. In Table 2, the sex of the patient differentiates $li=1$ observations for males from females. Remission predominates in females with $li=1$. For males, only a small proportion with $li=1$ achieve remission.

Table 3: Approximate Function of Model $remiss=f(li,x)$

Obs	x	li	remiss	count
1	0	0	0	2
2	0	1	0	12
3	0	1	1	1
4	0	2	0	1
5	1	1	0	2
6	1	1	1	5
7	1	2	0	1
8	1	2	1	3

We can see that including x in the model increases its predictive accuracy substantially. (In fact, omitting an important predictor would likely bias predictions since the binary sex class properly separates $remiss=f(li)$ into two functions that better fit observed values in the sample relation.) As computed by SAS PROC LOGISTIC,

```
proc logistic data=Remission descending;
model remiss= li;
model remiss= li x;
run;
```

the concordance c statistic for the reduced model increases from a modest 0.855,

Percent Concordant	84.0	Somers' D	0.710
Percent Discordant	13.0	Gamma	0.732
Percent Tied	3.1	Tau-a	0.328
Pairs	162	c	0.855

to a stronger 0.910,

Percent Concordant	90.1	Somers' D	0.821
Percent Discordant	8.0	Gamma	0.836
Percent Tied	1.9	Tau-a	0.379
Pairs	162	c	0.910

Improving Fit of Model to Observed Data

A better “fit” of model parameters to a sample relation may also improve predictive accuracy of a model. For example, transformations of predictors may also have smaller but still substantial impact on the accuracy of a predictive model. We begin with square root and square transformations:

```
proc sql;
  create view vwRemission as
  select *,li**(1/2) as lisqrt,li**2 as lisq
  from Remission ;
quit;
```

Using the view vwRemission, we compute parameter estimates for models featuring predictor x, different transformations of li, and expansions of the basis li.

```
proc genmod data=vwRemission descending;
  model remiss=x lisqrt/link=logit dist=binomial;
  output out=preds p=pred STDRESDEV=STDRESDEV u=u l=l;
run;
proc sql;
  create table predictions as select remiss as outcome,pred
  from preds ;
quit;
%cStatistic(pred)
```

Substituting the square root of li, lisqrt, for its square, li, doesn't change the c statistic value of 0.855, but expanding the basis of x, lisqrt to x, lisqrt, li increases the c statistic value from 0.910 to 0.941. Finding a better fit for a linear model through transformations of predictors may compensate for misfits of measurement scales of outcome and predictors. Treating a predictor as categorical as opposed to continuous or ordinal numeric would also affect the fit of the model.

Bootstrapping

But does the better fit due to the expanded basis of li hold up in estimates from a thousand bootstrapped replicates of a sample with an obs omitted from each? Let's use Cassell's SAS bootstrap method (as specified in Part 1) to see. To save space, we'll simply refer to the method using a SAS Macro call:

```
%CassellBootStrap() (For details, see Part 1.)
```

Yes, in fact, aggregations of predictions generated by the expanded model from a thousand bootstrapped $n-1$ samples from the Remission sample relation do have a higher concordance c of 0.973 as compared to 0.941 for estimates from the same model applied to the full sample relation. That means fitting a more complex model to many bootstrapped samples improves accuracy of predictions of outcomes despite the fact that we only observe that sample relation. The bootstrap method reduces the impact of outliers and coincidence on estimates derived from small samples.

Of course in typical data mining applications we may have such large samples that a few outliers or coincidences won't have much impact on estimates of model parameters. I say "may" because a huge volume of data may have very few instances of an outcome of interest. Volunteer blood donors with HIV infections come to mind. And even with much larger proportions of outcomes of interests, we may still have to contend with missing, incomplete, and inaccurate observations, with interactions among predictors, with repeated measures and other sources of covariances, and with time series that require extrapolations. Observations available in large numbers often include many collinear predictors that, if one predicts well, adding another to a model doesn't do much to improve accuracy of prediction. In a sense each is telling a slightly different version of the same story. Even so, estimating model parameters from large volumes of data tend to be more accurate as predictors of outcomes than "explainers" of how outcomes in different subsets of a population compare to one another. For that reason the trade-offs among sensitivity, specificity, and pure chance, as measured by the c statistic, has a primary role in predictive modeling.

So we have quickly touched on three established paths to computing better predictions:

1. adding relevant and important predictors to a predictive model to reduce bias;
2. improving model fit by transforming predictors or taking interactions, covariances and time-dependent effects into account (thereby decreasing variances of predictions);
3. obtaining improved variance of estimates by bootstrapping.

Each of these paths has its own pitfalls. Since the object of the game is to predict outcomes in other sample relations drawn from the same population, we have to avoid being too optimistic about how well a predictive model will extrapolate to new sets of observations. The next section addresses some of the more dangerous pitfalls.

CONTROLLING MODEL COMPLEXITY

Let's begin by looking at an example that tells a decidedly cautionary tale. We can add really random variables to set of predictors:

```
data Remission;  
  array v v1-v10 _temp_;  
  set test.Remission;  
  do j=1 to 10;  
    v[j]=ranuni(0);  
  end;  
  output;  
run;
```

Now we should be able to distinguish really random predictors from real predictors, right?

So to see we take an example directly, unaltered, and unabridged from documentation of PROC LOGISTIC and its stepwise option (p.2391). The SAS/Stat manual uses this program and data in Table 1 to illustrate step-wise automated variable selection:

```
proc logistic data=Remission outest=betas covout descending;
model remiss=cell smear infil li blast temp v1-v10
/ selection=stepwise slentry=0.3 slstay=0.35 details lackfit;
output out=pred p=phat lower=lcl upper=ucl
predprob=(individual crossvalidate);
run;
```

For the purpose of comparison of stepwise and a new SAS 9.2 alternative (LASSO stage-wise selection) we add this procedure call:

```
proc glmselect data=Remission;
model remiss=cell smear infil li blast temp v1-v10/selection=lasso;
run;
```

The RANUNI() function seed argument of 0 in the program that creates data for the stepwise and GLMSelect LASSO procedures adds an interesting twist: the really random variables differ each time the program runs. Imagine what statisticians Pratt and Schlaifer have called a “prankster” in nature who lets us listen for a melodic structure amidst a set of distracting noises, then introduces a new set of noises. In any one sample the prankster arranges noises to sound like fragments of melodies. He also arranges several mixtures of fragments of the same melody with different fragments of noises. Only in different or larger samples does a careful listener have a chance of distinguishing the melody from the cacophony surrounding it. Adding really random predictors doesn’t come close to what a prankster might do, but it brings us as close as possible to the ultimate “black box” model. The random predictors by design have no relation to the outcomes.

Step-wise Selection

The step-wise method (forward selection) selects a predictor for inclusion in the model if its p-value falls below the slentry setting and keeps it in the model after adding other predictors so long as its p-value remains below the slstay setting. Step-wise selection in this case reduces the AIC diagnostic statistic (see Table 4) from 36.372 for intercept only to a better value of 32.008 for a fuller model that adds six covariates (predictors).

Criterion	Intercept Only	Intercept & Covariates
AIC	36.372	32.008
SC	37.668	41.078
-2 Log L	34.372	18.008

Table 4. Key Model Likelihood and Fit Statistics

L	$L(B) = \prod f_i(Y_i; B)$	Likelihood function of outcomes and parameter vector B
AIC	$-2 \log L - 2p$	Akaike's Information Criteria for model with p parameters.
LR	$L_{\text{Reduced}}/L_{\text{Full}}$	Likelihood Ratio of reduced to full model
SBC	$-2 \log L + p \log(n)$	Schwartz Bayesian Criteria

Bloated estimates of the intercept and parameters call for caution. Even without the random predictors it would prove difficult to explain this set of estimates:

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	96.4204	68.6260	1.9741	0.1600
li	1	1.5532	1.6314	0.9064	0.3411
blast	1	3.5947	2.3773	2.2865	0.1305
temp	1	-98.5142	69.1185	2.0315	0.1541
v1	1	-5.8348	3.6259	2.5896	0.1076
v6	1	3.3845	3.2130	1.1096	0.2922
v10	1	-4.3555	2.9211	2.2232	0.1359

Even so, the predictive accuracy of the more complex model exceeds that of a reduced model including only an intercept and the li predictor (0.907 vs. 0.855):

Association of Predicted Probabilities and Observed Responses

Percent Concordant	90.7	Somers' D	0.815
Percent Discordant	9.3	Gamma	0.815
Percent Tied	0.0	Tau-a	0.376
Pairs	162	c	0.907

GLMSelect LASSO

In contrast the GLMSelect Least Absolute Shrinkage And Selection Operator (LASSO) method selects only one predictor, li, the same result as step-wise selection without the really random predictors:

The GLMSELECT Procedure

Data Set	WORK.REMISSION
Dependent Variable	remiss
Selection Method	LASSO
Stop Criterion	SBC
Effect Hierarchy Enforced	None

LASSO Selection Summary

Step	Effect	Effect	SBC
	Entered	Removed	
0	Intercept	1	-37.3143
1	li	2	-41.5404*

Further, the model selected by LASSO minimizes the Schwartz Bayesian Criterion (and usually the AIC as well) instead of maximizing predictor p-values and, typically, the accuracy of prediction as represented by the c statistic.

* Optimal Value Of Criterion

Selection stopped at a local minimum of the SBC criterion.

AIC	-17.00009
AICC	-15.95662
SBC	-43.40842

Parameter Estimates		
Parameter	DF	Estimate
Intercept	1	0.006542
li	1	0.325586

The LASSO method by intent shrinks confidence intervals of parameter estimates at the expense of some bias in maximum likelihood of a model. It trades off bias for more reliable estimates. The greater value of the c statistic for the more complex model selected step-wise (0.907) exceeds the c statistic we find for the LASSO model. For predictions computed as $0.006542 + 0.325586 \times li$, the c statistic has the same value of 0.855 that PROC LOGISTIC computes for the reduced model with li as the only predictor (but with larger absolute values of parameter estimates). The LASSO controls model optimism due to overfitting a model to a sample relation. Not a perfect process, one has to concede when the LASSO on occasion selects a really random variable from a set of 100, but a better one.

So now we have a clear choice between a black box that predicts better in a sample relation and a simpler and more plausible model that predicts not quite as well. Except in this case we know for certain that predictors v1, v6, and v10 cannot have the same relation to the outcome in any other sample drawn from the population that they have to the outcome in the model selected step-wise. We know that because we invented them out of thin air. In this case at least, the nod has to go to the method that selects a more plausible models.¹

CART

A choice of model selection method that lies somewhere between black box and explanatory or causal model selection, Classification and Regression Trees (CART), offers several features that complement LASSO and other stage-wise methods. CART begins with all data in one node of a tree (the root) and successively finds binary splits of nodes that come closest to complete separation of outcomes (node purity). A classic CART problem defines a set of binary predictors of a binary outcome, e.g.,

Table 5. CART Data Example

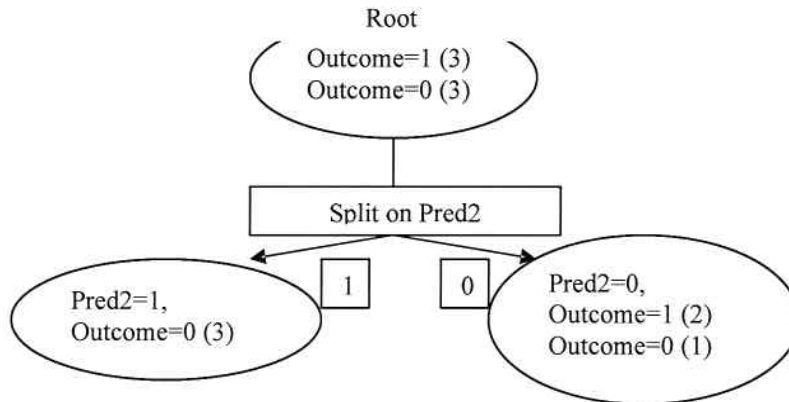
Outcome	Pred1	Pred2	Pred3	Pred4
1	1	0	0	1
0	1	1	0	0
1	1	1	1	0
0	0	1	1	1
0	0	1	0	0
1	0	0	1	1

For these data CART would first split on Pred2, leaving Outcome=0 completely determined by Pred2=1 and Outcome=1 in this state

Table 6. CART Data Subset

Outcome	Pred1	Pred3	Pred4
1	1	0	1
1	1	0	0
1	0	1	0

After splitting on Pred2, the resulting classification tree has two levels (including root):



At this second level none of the predictors separate Outcome=1 better than any of the others. A split on any of the predictors, including Pred2, will then completely determine Outcome.

This simple example illustrates the main strengths of CART: divide-and-conquer computing and ease of interpretation. It also hints at its main weaknesses: its tendency to make somewhat arbitrary selections of splits and to take splitting too far. CART divides a complex decision problem into many simpler problems, and if, say, Pred2 and Pred4 refer to different vaccines against a disease and Outcome a case of that disease, a two-level decision tree makes intuitive sense of these data: the Pred2 vaccine protects against the disease, while subjecting a patient to another vaccination either has no observed benefit or could do harm. Yet we can also see that the first split predetermines all subsequent splits and that a different sample relation could lead to a very different classification tree.

Though not necessarily superior, in a wide variety of applications CART often complements automated model specification methods such as LASSO. Models with many collinear predictors, a bane of linear regression, do not cause the same complications in CART. To the contrary, CART takes advantage of collinear predictors when confronted with missing values, and uses observed values in one set of collinear predictors as surrogates for missing values. Logistic regression procedures, for example, exclude observations with missing values. Missing values scattered among many predictors may reduce substantially the predictive power of a model and could, where something systematic is causing missing values, bias predictions.

To illustrate CART model selection (and by analogy similar features of Chi-squared Automatic Interaction Detector [CHAID] decision tree methods in SAS/BI), we turn to ordinally-coded customer profile data used to illustrate market segmentation methods in Hastie et. al. (2001) and available at

<<<http://www-stat.stanford.edu/~tibs/ElemStatLearn/>>>

The MarketBasket dataset has a substantial number of missing values distributed across a dozen predictors. In this exercise we are regressing a binary variable ($Ilt40 \equiv \text{"Annual Income} < \$40,000 \text{ (level 7)} = 0; \text{ else } 1$) on a variety of less than ideal predictors.

An Extended Example

We again add really random predictors (in this case 100) to the 8,993 observations in the dataset. We then repeat step-wise PROC Logistic and stage-wise GLMSelect as before, but with data from the augmented MarketBasket and step-wise selection criteria lowered to defaults ($slentry=0.05$ $slstay=0.05$).

PROC LOGISTIC reports under "Number of Observations Used" that at least one missing value occurs among model predictors in 2,117 (~25%) of the 8,993 observations in the MarketBasket dataset. Regression procedures omit these and estimate parameters using only 6,876 observations.

The LOGISTIC Procedure

Model Information

Data Set TEST.MARKETBASKETRANDOMMISSING

Response Variable *Ilt40 (Income level $\geq 7 = 0$; else 1)*

Number of Response Levels 2

Probability modeled is $Ilt40=1$.

Model binary logit

Optimization Technique Fisher's scoring

Number of Observations Read 8993

Number of Observations Used 6876

Response Profile

Ordered Value	Ilt40	Total Frequency
1	1	2596
2	0	4280

Probability modeled is $Ilt40=1$.

NOTE: 2117 observations were deleted due to missing values for the response or explanatory variables.

Model Fit Statistics

Criterion	Intercept Only	Intercept & Covariates
AIC	9117.507	6887.820
SC	9124.342	6997.193
-2 Log L	9115.507	6855.820

Testing Global Null Hypothesis: $BETA=0$

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	2259.6864	15	<.0001
Score	1962.7151	15	<.0001
Wald	1482.3051	15	<.0001

Analysis of Maximum Likelihood Estimates

Standard Wald

Parameter	DF	Estimate	Error	Chi-Square	Pr > ChiSq
Intercept	1	-0.7620	0.3037	6.2941	0.0121
sex	1	-0.2489	0.0608	16.7844	<.0001
married	1	-0.2901	0.0199	212.3660	<.0001
education	1	0.4948	0.0301	269.7877	<.0001
occupation	1	-0.1486	0.0127	136.0671	<.0001
years_Bay_area	1	0.1134	0.0258	19.3254	<.0001
NInHouse	1	0.2297	0.0300	58.6373	<.0001
NChildren	1	-0.1892	0.0415	20.7461	<.0001
Status*	1	-0.5205	0.0544	91.4149	<.0001
typeHome	1	-0.3544	0.0312	129.2572	<.0001
ethnicity	1	0.0629	0.0180	12.2642	0.0005
v22	1	0.2193	0.1031	4.5232	0.0334
v31	1	0.3330	0.1029	10.4721	0.0012
v64	1	-0.2866	0.1030	7.7455	0.0054
v78	1	0.2082	0.1030	4.0908	0.0431
v92	1	-0.2264	0.1032	4.8092	0.0283

*Homeowner Status: 1=Own; 2=Rent; 3=Live with Parents

Odds Ratio Estimates

Effect	Point	95% Wald	
	Estimate	Confidence Limits	
sex	0.780	0.692	0.878
married	0.748	0.720	0.778
education	1.640	1.546	1.740
occupation	0.862	0.841	0.884
years_Bay_area	1.120	1.065	1.178
NInHouse	1.258	1.186	1.334
NChildren	0.828	0.763	0.898
status	0.594	0.534	0.661
typeHome	0.702	0.660	0.746
ethnicity	1.065	1.028	1.103
v22	1.245	1.017	1.524
v31	1.395	1.140	1.707
v64	0.751	0.614	0.919
v78	1.232	1.006	1.507
v92	0.797	0.651	0.976

Association of Predicted Probabilities and Observed Responses

Percent Concordant	82.0	Somers' D	0.641
Percent Discordant	17.9	Gamma	0.642
Percent Tied	0.2	Tau-a	0.301
Pairs	11110880	c	0.821

.....

GLMSELECT handles missing values the same way. Loss of statistical power would not typically be an issue in this case due to the large number of observations remaining. (An exception could be a model of risk factors for a very rare condition that has only 100 positive instances amidst 8,000 negative instances.) Of more concern would be whether values are Missing at Random (MAR). If not, we may lose important information about the relation of income to the predictors.

The GLMSELECT Procedure
Number of Observations Read 8993
Number of Observations Used 6876

LASSO Selection Summary				
Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	-10337.708
1	status		2	-10879.113
2	married		3	-11021.699
3	education		4	-11618.251
4	occupation		5	-12179.781
5	age		6	-12990.843
6	typeHome		7	-13042.168
7	dual_incomes		8	-13084.492
8	sex		9	-13148.490
9	years_Bay_area		10	-13157.890
10	Language		11	-13160.835
11	v90		12	-13165.302*

Selected Model

The selected model is the model at the last step (Step 11).
Effects: Intercept sex married age education occupation years_Bay_area dual_incomes
status typeHome Language v90

Analysis of Variance				
Source	DF	Sum of Squares	Mean Square	F Value
Model	11	534.11573	48.55598	335.68
Error	6864	992.88297	0.14465	
Corrected Total	6875	1526.99869		
Root MSE		0.38033		
Dependent Mean		0.66710		
R-Square		0.3498		
Adj R-Sq		0.3487		
AIC		-6404.29472		
AICC		-6404.24167		
SBC		-13200		

Parameter Estimates		
Parameter	DF	Estimate
Intercept	1	0.790027
sex	1	-0.030425
married	1	-0.041647
age	1	0.033992
education	1	0.057932
occupation	1	-0.037481
years_Bay_area	1	0.004098
dual_incomes	1	0.020887
status	1	-0.088818
typeHome	1	-0.020783
Language	1	-0.007440
v90	1	0.006414

LOGISTIC step-wise and GLMSELECT identify many of the same predictors. From this particular set of data GLMSELECT somewhat atypically selects one of the really random variables as its last choice; nonetheless, it shrinks its parameter estimate for V90 to 0.006, a small fraction of the $\{0,1\}$ range of V90, so that this “nuisance parameter” adds little to predictions of the dependent variable. In contrast, the LOGISTIC step-wise model selects five really random predictors with significant odds ratio confidence intervals. Also, some of the LOGISTIC parameter estimates appear to be less plausible than the GLMSELECT parameter estimates.

Salford Systems CART® V. 6 Professional Ed.’s supervised classification (observing outcomes as well as predictors) selects a set of predictors that partially overlaps the LOGISTIC as well as the GLMSELECT selections. Some of the differences may be due to LOGISTIC and GLMSELECT exclusion of observations with missing values

Table 6: Variable Importance (Classification Trees, Supervised)

Variable	Score	
STATUS	100.00	
MARRIED	97.60	
DUAL_INCOMES	94.09	
OCCUPATION	81.43	
AGE	56.95	
TYPEHOME	39.46	
EDUCATION	25.18	
NINHOUSE	8.90	
ETHNICITY	2.25	
V31	0.92	
LANGUAGE	0.79	
NCHILDREN	0.55	
V63	0.40	
etc.	.	

CART builds a decision tree hierarchy in a way that progressively improves node purity, while parametric modeling methods such as logistic and generalized linear models fit a function of a model to data. For that reason we do not find it surprising that CART rank predictors differently than GLMSELECT. Ranking of predictors seems less critical than the subset of predictors that both methods select.

CART as a rule grows a classification tree to a level at which splits barely separate outcomes, then “prunes” the tree back to a level that classifies outcomes almost as well. Something along those lines seems appropriate for control of models selected by GLMSELECT as well. Parameter estimates shrunk to very small values become good candidates for exclusion. Predictions of a model will not change much whether or not a model includes predictors of normal scale with very small parameter estimates.

CART unsupervised classification looks for novel and hidden associations among members of clusters of predictors. Market segmentation studies use unsupervised classification and related association rules mining methods to find common attributes of groups of prospective customers. Extending this example to CART unsupervised classification reveals little more than chance associations among random and real

predictors:

Table 7: Variable Importance (Classification Trees, Unsupervised)

Variable	Score	
V72	100.00	
V93	23.05	
V75	18.94	
V70	16.85	
V21	15.10	
EDUCATION	11.66	
DUAL_INCOMES	10.59	
V39	9.04	
MARRIED	7.93	
V74	6.71	
STATUS	6.34	
V33	5.03	
NINHOUSE	4.97	
TYPEHOME	4.67	
V73	4.39	
V95	4.28	
Etc.	.	
AGE	3.77	
V2	3.75	
Etc.	.	
NCHILDREN	2.52	
V98	2.18	
V66	2.13	
OCCUPATION	1.73	
V55	1.52	
ETHNICITY	0.31	
Etc.	.	
YEARS_BAY_AREA	0.23	
V53	0.23	

This extended example demonstrates some of the pitfalls inherent in predictive models. While more complex models often predict outcomes in a sample strikingly well, some really random variables improve predictions of sample outcomes but would have no value as predictors of outcomes in different samples.

In this example we can distinguish randomly generated predictors from real ones by variable name. In other contexts we cannot. We have to rely on two basic methods for controlling model complexity: first, limit predictors in a model to those that theory and knowledge ascribe roles in determining an outcome; and, second, use multiple and complementary methods, such as GLMSELECT and CART, to distinguish important predictors from nuisance parameters.

BETTER MODEL EVALUATION MEASURES

Though convenient measures of predictive accuracy, the concordance c statistic and related ROC AUC have theoretical and practical drawbacks. The c statistic indicates how well a predictive model discriminates one population from another in retrospect, once we know outcomes, but does not take into account uncertainty inherent in select a classification rule (say, a score cut point) prior to knowing outcomes. This situation calls

for a predictor-conditional measure as opposed to an outcome-conditional measure. We also have to consider what we lose by selecting one classification rule instead of another. It would seem a rarity to find in any one situation that the cost, however measured, of a “false positive” prediction would be even approximately the same as a “false negative” prediction, yet that assumption lies behind typical statistical decision rules. A simple loss function evaluates the trade-off between misclassification costs.

Predictor-Conditional Concordance

Suppose that we use predicted values generated by PROC LOGISTIC to separate observations into outcome groups. For example, we observe predicted predictions of an outcome in a sample relation and determine a cutpoint that approximately separates outcome=1 from outcome=0. Then by adjusting the focus to a narrow range of cut-off values, we can improve the predictive accuracy of a model.

Improving outcome-conditional concordance based on sensitivity and specificity introduces a “prosecutor’s fallacy” in that we have to separate $Y=1$ from $Y=0$ observations before we predict outcome Y . It tempts us to overfit a model to a sample relation. Greenland () commented on this issue recently and suggested predictor-conditional concordance based in part on positive and negative predictive values as more appropriate: “Statistically, focusing on predictive values is simpler than focusing on ROC parameters, since those values are nothing more than classification probabilities derived from dichotomized risk scores” given a cutpoint.

Conditioning on predicted probabilities of an outcome initially presents an operational problem: how to dichotomize predictions into predicted outcome=1 and predicted outcome=2. Any selection of a cutpoint seems somewhat arbitrary. Perhaps, as Greenland also suggests, plausible cutpoints will fall into a relatively narrow range, but what if are looking for a measure that they can use to evaluate a predictive model?

Assessing $\Pr(y=1|x=1)$ introduces both conceptual and computing difficulties. Using the MarketBasket sample, this experimental SAS Macro computes a predictor-conditional statistic cx over a range (0.68 to 0.72) of probabilities of income >\$40,000 (level 7). It generates Table 8:

```
%Macro cxStatistic(__dsn,__Outcome,__pred);
proc sql noprint;
create table cx as
select 0.&__cutpoint as cutpoint,sum(cVal)/sum(count) as cx,
       sum(count) as count,mean(obsPPV) as MeanObsPPV,mean(obsNPV) as meanObsNPV
from
  (select (t1.n * t2.n) as count,
         sum((outcome1=1)*t1.n)/sum(t1.n) as obsPPV,
         sum((outcome2=0)*t2.n)/sum(t2.n) as obsNPV,
         case when t1.LL>t2.LL and outcome1=1 and outcome2=0 then 1*calculated count
              when t1.LL=t2.LL and outcome1=1 then 0.5*calculated count
              else 0
         end as cVal
  from
    (select &__Outcome as outcome1,&__pred as LL,n
```

```

from &__dsn where &__pred >= 0.&__cutpoint
) as t1,
(select &__Outcome as outcome2,&__pred as LL,n
from &__dsn where &__pred < 0.&__cutpoint
) as t2 ); quit;
%Mend cxStatistic;

%macro cxDist;
%do __cutpoint=68 %to 72;
  %cxStatistic(%str(test.Prediction),llt40,pred)
  proc append base=cxs data=cx; run;
%end;
%mend cxDist;
%cxDist

```

TABLE 8: PREDICTOR-CONDITIONAL CONCORDANCE

cutpoint	cx	MeanObsPPV	meanObsNPV
0.68	0.597521	0.880658	0.678494
0.69	0.593802	0.883436	0.672151
0.70	0.598997	0.894737	0.669467
0.71	0.598201	0.896552	0.667224
0.72	0.599592	0.901639	0.665002

The cx statistic increases with a cutpoint that dichotomizes predictions of Outcome=0 and Outcome=1, while observed NPV decreases. One can choose either fewer false positives or fewer false negatives. PPV changes in parallel with cx.

Loss functions

Loss functions multiply counts of false positive and false negative misclassifications by costs. For example, we might assign a cost of 0.5 to 30 predictions that cause us to misclassify a person without heart disease, and a cost of 2.0 to 10 predictions that causes us to misclassify a person with heart disease. The total loss due to the 40 prediction errors would be $0.5 \cdot 30 + 2.0 \cdot 10 = 35$. We can then compare this total loss with a total from a different trade-off of 20 false positives and 15 false negative. Such costs tend to be difficult to estimate with precision, but a useful loss function only has to have an approximately correct ratio of costs. If we represent a false match rate as loss of concordance ($1 - \text{PPV}$) and a false negative rate as $(1 - \text{NPV})$, the %cxStatisticLF version of the %cxStatistic Macro computes a misclassification loss, costMSC:

```

/* Loss Function ... */
%Macro cxStatisticLF(__dsn,__Outcome,__pred,__LFFN=1,__LFFP=1);
proc sql noprint;
create table cx as select 0.&__cutpoint as cutpoint,sum(cVal)/sum(count) as cx,
                        sum(count) as count,mean(obsPPV) as MeanObsPPV,
                        mean(obsNPV) as meanObsNPV,mean(costMSC) as meanCostMSC
from
(select (t1.n * t2.n) as count,
      sum((outcome1=1)*t1.n)/sum(t1.n) as obsPPV, sum((outcome2=0)*t2.n)/sum(t2.n) as obsNPV,
      case when t1.LL>t2.LL and outcome1=1 and outcome2=0 then 1*calculated count
            when t1.LL=t2.LL and outcome1=1                        then 0.5*calculated count

```

```

else 0
end as cVal,
(1-calculated obsPPV)*t1.n*&__LFFP + (1-calculated obsNPV)*t2.n*&__LFFN as costMSC
from
(select &__Outcome as outcome1,&__pred as LL,n
from &__dsn where &__pred >= 0.&__cutpoint
) as t1,
(select &__Outcome as outcome2,&__pred as LL,n
from &__dsn where &__pred < 0.&__cutpoint
) as t2 ); quit;
%Mend cxStatisticLF;
%macro cxDist;
%do __cutpoint=1 %to 90;
%cxStatisticLF(%str(test.Prediction),llt40,pred,__LFFN=%str(0.5),__LFFP=%str(2))
proc append base=cxLFs data=cx; run;
%end;
%mend cxDist;
%cxDist
proc sql;
create table test.cxLFs as
select cutpoint,cx,meanObsPPV,meanObsNPV,meanCostMSC from cxLFs
order by cutpoint
; quit;

```

In Table 9 the cx statistic increases while the total of misclassification costs (cost of predicting income $\leq$ \$40,000 (level 7) in error = 0.5; cost of predicting income $>$ \$40,000. in error = 2.0) levels out as cutpoints increase from 0.70 to 0.72.

TABLE 9: CONCORDANCE WITH YIELD OF LOSS FUNCTION

cutpoint	cx	MeanObsPPV	meanObsNPV	meanCostMSC
0.68	0.597521	0.880658	0.678494	11.19412
0.69	0.593802	0.883436	0.672151	10.94767
0.70	0.598997	0.894737	0.669467	10.48506
0.71	0.598201	0.896552	0.667224	10.50162
0.72	0.599592	0.901639	0.665002	10.37014

A loss function doesn't add to the complexity of a model to make predictions more accurate. It simply makes them more meaningful.

CONCLUSIONS (DECISIONS, DECISIONS)

We began with the goal of assessing whether more complex predictive models actually predict better. The concordance c statistic that we used in Part 1 to evaluate predictive models, though a good indicator of accuracy of a predictive model derived from and applied to the same sample data, does not always extrapolate well when applied to different sample relations. A high level of concordance proves necessary but not sufficient. To assess whether a model approximates a function of a population of interest, we must control model complexity and thereby prevent overfitting a model. Introducing really random variables into a set of predictors sets up an acid test that step-wise methods fail consistently and methods such as the LASSO and CART fail occasionally. Any predictor that has no known or theoretic role in determining an outcome has to be as

suspect as a really random variable.

We have illustrated how different robust model selection methods such as GLMSelect and CART produce different results. That alone should encourage developers of predictive models to combine results of different methods before settling on a model. Time spent exploring data prior to a first attempt to fit a model to a sample not only increases confidence in a model, it often saves time required to produce a final model.

New measures of model accuracy such as predictor-conditional concordance (the experimental cx statistic) have a potential role in decisions about cutpoints needed to dichotomize predictions of outcomes. Misclassification costs and loss functions help us make better use of good predictions.

The SAS System offers us a fast and sturdy vehicle that we can use to explore data and test a range of hypotheses and predictions. We have not had time during this little test drive to explore covariance structures, survival analysis, time-series and other interesting domains. We don't run out of new things to learn. Thanks to SAS and other developers of data mining and business intelligence tools, mining the past and seeing the future has never been more challenging or rewarding.

REFERENCES

Cassell, D., "Don't Be Loopy: Re-Sampling and Simulation the SAS® Way", Paper 183-2007, **SAS Global Forum 2007**.
Dorfman P & Snell G, "Hashing Rehashed" <<http://www2.sas.com/proceedings/sugi27/p012-27.pdf>>
Harrell, Jr., F., **Regression Modeling Strategies**, Springer (2001) NY.
Hastie, T, Tibshirani, R and Friedman, J, **Elements of Statistical Learning, Data Mining, Inference, and Prediction**, Springer (2001) NY

ACKNOWLEDGMENTS

David Wright of Westat reviewed the content of this paper. His valuable suggestions and comments led to substantial improvements in content and presentation. George Fernandez and Duke Owen provided valuable suggestions for and comments about an earlier version of this paper. The author has sole responsibility for any errors or oversights.

DISCLAIMERS

The contents of this paper are the work of the author and do not necessarily represent the opinions, recommendations, or practices of Westat. SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Sigurd W. Hermansen
Westat
1650 Research Blvd, Rockville, MD 20850
Work Phone: 301.251.4268
E-mail: hermans1@westat.com