Paper 272-28

# Known Nonsense

Carl Formoso, Division of Child Support, Olympia, WA

**Abstract**
Model selection in regression analyses can be aided by the addition of known non-information. A randomly generated variable is known to have no explanatory power for any outcome of interest. When the random variable is selected as the 'best' entry for model improvement we can be sure that the model is not improved, even though usual measures such as adjusted R square and Mallow's Cp may show improvement. When the random variable remains in the model, later entered variables can also be considered to not improve the model. We also demonstrate the generation of randomness through disordering an existing variable to create a random variable with the exact values and distribution of a real variable.

**Introduction**
In physical sciences a common approach is to use a known commodity as a reference in determining an unknown commodity. In statistical analysis there is often little frame of reference, and no known relationships between explanatory variables and outcome variables. However it is easily possible to create the bottom of the scale by adding a variable known to have no explanatory power. We provide two examples in this paper. The first uses a variable randomly generated from a uniform distribution in an explanatory model, and the second uses a disordered form of an existing variable in a predictive model.

**Using a Randomly Generated Variable**
The first example is from the development of an explanatory resource allocation model, where we are attempting to find relationships between employee work activities and agency outcome measures. The data comes from an automated system which captures individual work activities and from Federal Incentive Measures for child support collections.

The analytical data set is created by the code below, with explanatory variables a4-a22, outcome variables o1-o4, categories dt (month), fo (field office), emp (employee number), jc (job class), and sup (supervisor), and random variable a3. It is useful to name the random variable so that when it is in the model, it will be listed first.

```
data jc;
infile
   'd:\emodel\empfedincvjc.txt';
input
   a4-a22
   o1-o4
   dt fo emp jc sup;
a3=ranuni(0);
run;
```

Because we are considering models across 10 field offices, 7 job classes, and 4 outcomes it was necessary to automate the procedure as much as possible. For demonstration, the coding example below is a modified excerpt of the actual method, which is a macro procedure.

```
ods listing close;
%let i=2;
%let j=7;

proc reg data=jc;
model o&i=a3-a22/
selection=rsquare
   best=1
   adjrsq cp mse sse;
where jc=&j;
ods output
   SubsetSelSummary= tmpJ&j.O&i;
run;
```

PROC REG® with the model options specified will try all one-variable models, all two-variable models, and so on. The output would usually list all models in order of the best model for each number of variables, but the "best=1" option simply selects the first record, i.e., the best model, for each number of variables.

Appendix Table 1 shows an example of results where the random variable enters the model and stays in while adjusted R square and Cp continue to show improvement.

In this example a3 was the "best" third variable to enter the model even though the minimum value of Cp is for a 5 variable model. Since we know a3 contains no information relating to outcome o2, there must also be very little information relating to o2 in variable a20, a15, or any of the later entered variables because a3 remains in the model – no subsequent variable is able to replace it. The best model appears to be o2= a7  a21.

It would be easy to automate finding a3 by a simple statement such as: *if substr(varsinmodel,1,2)= "a3" then . . . .*

But there are also situations, demonstrated in Appendix Table 2, where the random variable enters the model, then is removed, and finally enters to stay.

Here a3 is part of the 5 variable model and the 6 variable model, but is removed from the 7 variable model as are a8 and a17. The 12 variable model once again enters a3 and it is part of all subsequent models. Multicollinearity or interactive effects may have caused this behavior. There are two possible model choices here, the 4 variable model before a3 enters or the 10 variable model where Cp is minimal.

**Creating Randomness by Disordering**
While creating a random variable from a standard distribution can be useful there are situations where it may be better to create a random variable which matches the values and distribution of an actual variable. We used the techniques described here in selecting input variables for a neural network predictive model for changes in child support arrearage debt. Starting with over one hundred candidate variables we were able to select ten variables with consistent predictive power, where no variable added to the set significantly increased predictive power. The core of the selection procedure measures the information gain of a variable against a scrambled version of the same variable. Because of the complexity of the full procedure, we present a simplified version here for demonstration.

The code below will create a disordered version of an input variable. The data set pickS contains the ordered variable T95Q3 and the scrambled version T95Q3S, both with exactly the same values and distribution.

```
data t95;
   set pick;
   keep T95q3S ro;
   T95Q3S=T95q3;
   ro=rannor(0);
proc sort;
   by ro;
data pickS;
   set pick;
   set t95;
   drop ro;
run;
```

Information content is related to the number of binary questions required to obtain the desired answer. When outcomes are not equally likely the following equation applies:

$$I = -\sum N_i \log_2 f_i$$

where **I** is information content, $-\log_2 f_i$ measures the 'bits' of information for each correct prediction of outcome *i*, with $N_i$ the number of correct predictions for outcome *i*.

The outcome frequencies, $f_i$, are obtained from the known outcomes, for example using the code below.

```
proc sql;
   select
    -log2(sum(miss)/count(ssn))
         into: mbit from pick;
   select
    -log2(sum(up)/count(ssn))
        into: ubit from pick;
   select
    -log2(sum(down)/count(ssn))
          into: dbit from pick;
   select
    -log2(sum(same)/count(ssn))
          into: sbit from pick;
```

In the code below we use a multinomial linear model for demonstration – essentially a neural network with no hidden layer.

```
proc glm data=pick ;
   model miss up down same =
               durp durn dur0;
   output out=prd
        p=pm pu pd ps;
run;
```

The predictions are converted to dichotomous outputs in the code shown below.

```
data compare;
   set prd;
   keep miss up down same
            m0 u0 d0 s0;
   mx=max(pm,pu,pd,ps);
   m0=mx=pm;
   u0=mx=pu;
   d0=mx=pd;
   s0=mx=ps;
run;
```

Finally the code below allows an estimation of the information content extracted from the input variables.

```
proc sql;
   select mi+ui+di+si as info
    from (select
sum( m0=1 and miss=1)*&mbit as mi,
sum(u0=1 and up=1)*&ubit as ui,
sum( d0=1 and down=1)*&dbit as di,
sum( s0=1 and same=1)*&sbit as si
from compare);
```

We next use the above methods to look at the information contained in the three most powerful predictors, and ask about the information contained in three examples of a fourth predictor.

Our basic predictors are durations representing patterns of past debt behavior (see PROC GLM® code above) and in this simple model 199,769 bits of information for predicting debt are extracted from these three variables (see Appendix Table 3).

Using the debt level in the "current" quarter as a fourth predictor gains additional information, but the scrambled version adds no information. This demonstrates both that T95Q3 has predictive power and that scrambling removes all information from T95Q3 (see Appendix Table 3).

Scrambling works best for continuous variables such as T95Q3, but can also work for dichotomous variables. Using the gender of the debtor (NCPgen) as a fourth predictor gains additional information, with the scrambled version once again adding no information (see Appendix Table 3).

However, with another dichotomous variable (Typem, an indicator for a rare case type) there is not much information gained and the scrambled version appears to retain some information (see Appendix Table 3).

Typem is strongly skewed with only 0.6% valued 1. This already tells us that it's unlikely to contain very much information. But scrambling will not have much effect on Typem because a zero-valued observation has a very high probability of remaining zero-valued after the scrambling. NCPgen has 12.3% valued 1 which is apparently enough so that scrambling completely removes the information contained in NCPgen.

This approach also helps resolve the use of correlated variables. While NCPgen is shown to contain predictive information in the example presented above, it is not part of the final predictive model. NCPgen shows little information gain when other more powerful predictors, not discussed here, are in the model. The other variables are correlated with NCPgen and NCPgen adds no new information. However the final model does contain two strongly correlated variables. There is an overlap of information contained in the two variables, but each variable contributes predictive information not contained in the other.

**Author Contact Information**

Carl Formoso, Ph.D.
Research and Development Manager
Division of Child Support
PO Box 9162
Olympia WA 98507
(360)664-5090
cformoso@dshs.wa.gov

**Appendix**

**Table 1: Random Variable which Remains in Model**

| j | Dependent | NumInModel | RSquare | Adjrsq | Cp | VarsInModel |
|---|---|---|---|---|---|---|
| 7 | o2 | 1 | 0.320 | 0.312 | 20.040 | a18 |
| 7 | o2 | 2 | 0.410 | 0.396 | 7.739 | a7 a21 |
| 7 | o2 | 3 | 0.449 | 0.430 | 3.462 | a3 a7 a21 |
| 7 | o2 | 4 | 0.477 | 0.453 | 1.091 | a3 a7 a20 a21 |
| 7 | o2 | 5 | 0.496 | 0.466 | 0.065 | a3 a7 a15 a20 a21 |
| 7 | o2 | 6 | 0.506 | 0.472 | 0.363 | a3 a6 a7 a18 a20 a21 |
| 7 | o2 | 7 | 0.515 | 0.474 | 1.077 | a3 a6 a7 a18 a20 a21 a22 |
| 7 | o2 | 8 | 0.526 | 0.481 | 1.212 | a3 a5 a6 a7 a18 a20 a21 a22 |

*List truncated at 8 variable model*

**Table 2: Random Variable which Enters Model, Leaves, and Re-enters**

| j | Dependent | NumInModel | RSquare | Adjrsq | Cp | VarsInModel |
|---|---|---|---|---|---|---|
| 3 | o2 | 1 | 0.852101 | 0.849257 | 59.30426 | a22 |
| 3 | o2 | 2 | 0.897091 | 0.893055 | 28.05485 | a21 a22 |
| 3 | o2 | 3 | 0.911818 | 0.906527 | 19.17092 | a8 a21 a22 |
| 3 | o2 | 4 | 0.920446 | 0.913952 | 14.79398 | a8 a17 a21 a22 |
| 3 | o2 | 5 | 0.924756 | 0.916918 | 13.60888 | a3 a8 a17 a21 a22 |
| 3 | o2 | 6 | 0.928083 | 0.918902 | 13.15043 | a3 a8 a9 a17 a21 a22 |
| 3 | o2 | 7 | 0.934423 | 0.924443 | 10.46483 | a4 a6 a7 a9 a12 a21 a22 |
| 3 | o2 | 8 | 0.939837 | 0.929141 | 8.463352 | a4 a6 a7 a9 a12 a18 a21 a22 |
| 3 | o2 | 9 | 0.94378 | 0.932281 | 7.549034 | a4 a6 a9 a10 a12 a16 a17 a21 a22 |
| 3 | o2 | 10 | 0.947564 | 0.935369 | 6.753033 | a4 a6 a7 a9 a10 a12 a16 a17 a21 a22 |
| 3 | o2 | 11 | 0.948936 | 0.935562 | 7.738769 | a4 a6 a7 a9 a10 a12 a14 a16 a17 a21 a22 |
| 3 | o2 | 12 | 0.950794 | 0.936392 | 8.3658 | a3 a4 a6 a9 a10 a12 a14 a16 a17 a20 a21 a22 |

*List truncated at 12 variable model*

**Table 3: Information Gain of Ordered and Disordered Variables**

| X1 | X2 | X3 | X4 | Information, bits |
|---|---|---|---|---|
| durp | durn | duro | - | 199,769 |
| durp | durn | duro | T95Q3 | 201,456 |
| durp | durn | duro | T95Q3S | 199,769 |
| durp | durn | duro | NCPgen | 202,752 |
| durp | durn | duro | NCPgenS | 199,769 |
| durp | durn | duro | Typem | 199,922 |
| durp | durn | duro | TypemS | 199,781 |