SOCIETY OF ACTUARIES

Article from:

CompAct

October 2009 – Issue 33

R Corner – Combining Cluster Analysis and Predictive Modeling

By Steven Craighead

Editor's note: R Corner[i] is a series by Steve Craighead introducing readers to the "R" language used for statistics and modeling of data. The first column was published in the October 2008 issue, and explains how to download and install the package, as well as providing a basic introduction to the language. Refer to each CompAct issue since then for additional articles in the series. The introductory article can be found on p. 24 of the October 2008 issue on the SOA Web site: *http://soa.org/library/newsletters/compact/2008/october/com-2008-iss29.pdf*

Steve Craighead, ASA, MAAA, is an actuarial consultant at TowersPerrin in Atlanta, Ga. He can be reached at steven.craighead@towersperrin.com.

In this issue, I will outline a combination approach that I used within R to estimate the 90 percent Conditional Tail Expectation (CTE90) on an old block of business that I used more than 10 years ago[1]. I have used this approach to dramatically reduce the computer processing time regarding the calculation of Principle-Based Reserves and Capital.

This is done by using representative scenarios to train a separate smaller predictive model to replicate a full business model. After the predictive model is constructed, I then use all the scenarios within this less time-expensive model and not use any probability weights at all as is done with most representative scenario techniques.

In the study I use a single clustering algorithm to select these representative scenarios.

Let's briefly discuss the method used to select representative scenarios.

REPRESENTATIVE SCENARIOS

There are several (actuarially) published, as well as commonly known, methods to determine representative scenarios from a larger collection. However, I will use a cluster analysis technique called CLARA (Clustering LARge Applications). I will not introduce any weighting within our scenario selection process and I will treat the selection process as directly formulated by the sources. However, later I will discuss the use of weighting in the control of bias.

The CLARA Cluster Algorithm[2] can either use a sum of squares or a sum of absolute values metric to measure distance. In our work below I will use a sum of absolute values to indicate distance between separate scenarios. I have found the sum of absolute differences tends to select more extreme scenarios than does the sum of squares approach. This is important because when training a predictive model, the available information obtained from extreme scenarios increases the accuracy of the predictive model.

These are the steps of how CLARA works:

1. Choose the number of clusters (representative scenarios) desired, say M.
2. Choose an arbitrary data point out of the N sampled ones and call it Pivot #1.
3. Calculate the distances from Pivot #1 to the remaining N -1 data points.
4. Name the data point with the largest distance from Pivot #1 as Pivot #2. Randomly decide between ties.
5. Calculate the distances of the N - 2 non-pivot points to Pivot #1 and Pivot #2.
6. Assign each of the N - 2 points to the closest of Pivot #1 or Pivot #2, thus forming two disjoint clusters. Flip a coin if the distances are equal. Each of the N - 2 points now has a unique distance to its pivot point.
7. Rank these N - 2 distances in descending order. The point producing the top distance is called Pivot #3. (Break ties randomly.)
8. Following the above procedure to select the additional M - 3 pivot points.
9. If the number of points associated to a pivot #k is Nk, assign a probability of Nk/N to this pivot point.

For further insight on how CLARA is used within R, use the command:

```
help(clara)
```

Now, let's look at predictive modeling.

CONTINUED ON **PAGE 22**

FOOTNOTES

[1] Craighead, S. (2000), "Insolvency Testing: An Empirical Analysis of the Generalized Beta Type 2 Distribution, Quantile Regression, and a Resampled Extreme Value Technique," ARCH, pp. 13-149, available at http://www.soa.org/library/research/actuarial-research-clearing-house/2000-09/2000/arch-2/arch00v26.pdf.

[2] CLARA algorithm, see: http://rweb.stat.umn.edu/R/library/cluster/html/clara.object.html

PREDICTIVE MODELING

Predictive modeling is a means by which you can design or create a model that can be used to predict an outcome with approximately the same probability that is observed with the actual data. There are many different techniques, but while working independently with the Academy Valuation Basis Table subcommittee I found one type of predictive model that excelled all expectations. This modeling technique is called Projection Pursuit Regression[3] (PPR). Below, is a fairly extensive discussion around what it is and how you use it to create models.

In linear regression, you fit a response variable Y to a collection of n predictor variables X_i in the familiar form:

$$Y = \alpha + \sum_{i=1}^{n} \beta_i X_i + \varepsilon$$

Where in additive models, the $\beta_i X_i$ are replaced with various functions $f_i(X_i)$, with this form:

$$Y = \alpha + \sum_{i=1}^{n} f_i(X_i) + \varepsilon$$

Projection Pursuit Regression (PPR) was introduced by Friedman and Stuetzle[4] and it is a modification of the additive model in that there are:

- M different f_i.
- Each f_i acts on a different linear combination of all n of the X_k.
- A specific coefficient of these linear combinations is denoted by α_{ik}.
- Each f_i is multiplied by a β_i.
- The constant term is the average of the response variable.

So PPR takes on the following form:

$$Y = \bar{y} + \sum_{i=1}^{M} \beta_i f_i\left(\sum_{k=1}^{n} \alpha_k X_k\right) + \varepsilon$$

or in vector format:

$$Y = \bar{y} + \sum_{i=1}^{M} \beta_i f_i(\alpha_i \cdot X) + \varepsilon$$

where $X = (X_1, X_2, ..., X_n)$ is a vector of predictor values, and $\alpha_i = (\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{in})$ is a vector of directions.

The term Projection in PPR comes from the fact that X is projected onto the directional vector α_i for each i.

The Pursuit arises from the algorithm that is used to determine optimal direction vectors $\alpha_1, \alpha_2, \ldots, \alpha_M$.

Each f_i is called a ridge function. This is because they only have values in the α_i direction and are considered constant elsewhere. Effectively, what occurs is that the overall PPR model is a linear combination (β_i are the coefficients) of the ridge functions. These functions only take on values that arise from the projection of the predictors against the direction vectors, and the functions as assumed to take on a constant value in any other direction. So, each ridge function is like the profile of a mountain range, and you linearly combine these functions along all different ridges (as pointed out by the α_i).

On a formal basis, Y and X are assumed to satisfy the following conditional expectation:

$$E[Y \mid X_1, X_2, \ldots, X_n] = \mu_y + \sum_{i=1}^{M} \beta_i f_i(\alpha_i \cdot X)$$

with $\mu_y = E[Y]$ and the f_i having been standardized to have zero mean and a unit variance. That is: $E[f_i(a_i \lozenge X)] = 0$ and $E[f_i^2(a_i \lozenge X)] = 1$, where i takes on values from 1 to M. I assume that the realized sample values for the random variables Y and $X = (X_1, X_2, \cdots, X_n)$ are independent and identically distributed to the distributions of Y and X, respectively.

FOOTNOTES

[3] Projection Pursuit Regression, see: http://www.slac.stanford.edu/pubs/slacpubs/2250/slac-pub-2466.pdf

[4] Friedman, J. H. and Stuetzle, W. (1981) "Projection pursuit regression," Journal of the American Statistical Association, vol. 76, 817-823.

[5] R Development Core Team (2006). "R: A language and environment for statistical computing," R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

The PPR algorithm in the R stats library[5] estimates the best $\beta_i, f_i,$ and the α_i by minimizing the following target function for the mean square error

$$E\left[Y - \mu_y - \sum_{i=1}^{M} \beta_i f_i(\alpha_i \cdot X)\right]^2$$

across all the data samples for Y and X. Note: This expectation can be based on a weighted average as well.

A powerful trait of PPR models, since the predictor vector X is projected, is that interactions beween different X_j and X_k are included within the model, whereas other model algorithms cannot do this without user intervention.

Other advantanges of PPR are:

- The PPR model is a continuous function. According to Venables and Ripley,[6] they cite Diaconis and Shahshahani[7] and say that given a large enough number of ridge functions, PPR can approximate any arbitrary continuous function.

- It is the best possible fit since every component is solved for the minimization of the weighted least squares.

- Each ridge function does not extrapolate outside of its specific domain. If the specific $\alpha_i \cdot X$ is outside the domain the relevant domain endpoint is used.

- The model handles the interactions between the different predictors as I saw in the last section.

- PPR models categorical predictors as easily as continuous predictors.

- PPR models can take extremely large amounts of data and create a very good model of the underlying data. You can also adjust the model to distinguish between model fit and model smoothness.

- PPR does not suffer from the curse of dimenionality (COD). COD arises from the increased complexity of a multi-dimensional surface. Since PPR optimally is solved one ridge function at a time, the difficulty of trying to locate global optimal values for model calibration is eliminated.

But PPR is not perfect and there are a few disadvantages:

- The range of a PPR model may be outside of the range of acceptable values. For instance, if you are using PPR to model mortality, model results could fall below zero or above one. However, PPR will not extrapolate outside the existing ridge functions, so if any predictor projects on a specific α with a value outside the domain of a specific ridge function, the ridge function takes on the value either at the furthest point on the right hand side or left hand side. This no extrapolation rule can lead to biased results.

- All of the parameters are point estimates and there is no distributional consideration given to the significance of a specific parameter. Because you are not able to create a confidence interval using the R ppr function around each of the α_k or the β_i, you will not be able to determine if a specific parameter is significant to the model. In fact you are unable to actually test to see if the actual model is significant, other than the use of the goodness of fit statistic. There are complex methods that have been developed using spherical statistics to overcome this, but these require an understanding of advanced Banach Algebra in functional analysis and have not been included within the R ppr function.

- You can easily overfit or over explain the data. See Venables and Ripley for a further discussion.

R TECHNIQUES AND DIAGNOSTICS FOR PPR

The procedure when you use the R ppr algorithm is that you must first, specify that M should range between $M_{MIN} = 1$ and some positive integer M_{MAX}. The ppr algorithm then creates a PPR model for each M from M_{MAX} to M_{MIN} in a descending fashion, and

CONTINUED ON **PAGE 24**

FOOTNOTES

[6] Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S, Fourth Edition. Springer.

[7] Diaconis, P. and Shahshahani, M (1984) "On non-linear functions of linear combinations,", SIAM Journal of Scientific and Statistical Computing, vol 5, pp. 175-191.

at the same time produces a goodness of fit statistic for each value of . You then scan this list of goodness of fit values looking for a local minimum or zero. If this local minimum is at M_{MAX} you should reprocess the experiment with a larger M_{MAX}. Once you determine the local minimum, say s, reset $M_{MIN} = s$ and reprocess the ppr algorithm with the same M_{MAX} as before. The resultant model arising from the backward iteration from M_{MAX} to M_{MIN} will then be the best PPR model.

Two other components that are implemented in ppr is the concept of Bass and Optlevel. The option "bass" is to allow the calibration algorithm access to Friedman's super smoother bass tone control[8] that is used with automatic span selection. It is used in ppr to smooth the results. The range of values allowed with this component is from 0 to 10. To increase smoothing within the data, increase this value. The default is 0 and this setting gives the best fit to the underlying data. Bass is similar to the h smoothness parameter used within the Whitaker-Henderson graduation formula.

The option Optlevel is an integer from 0 to 3, which determines the optimization thoroughness. The best models usually are obtained if this is set to three. Level 0, the ridge functions, are not refitted. Level 1, the projection directions, are not refitted, but the ridge functions and the regression coefficients are. Levels 2 and 3 refit everything, but level 3 takes pains to rebalance each regressors' contribution at each step and so reduces the chance of converging to a saddle point in the sum of squares.

One diagnostic aid in PPR model building is to plot the ridge functions. If these ridge functions are very noisy or discontinuous, you should expect that the resultant PPR model will behave oddly.

Another effective diagnostic aid is to both plot the fitted $\hat{Y}$ against the actual Y and do a simple linear regression of Y against $\hat{Y}$, assuming no intercept. The scatterplot should display symmettry around the 45-degree line and the coefficient of the regression should be approximately 1. These two diagnostics will indicate how well the PPR model will perform as a predictive model.

Note: Since a PPR model does not extrapolate outside of the sample data, the resultant fitted values from PPR model will hit a maximum value and will not grow any larger no matter how you manipulate the predictors. This is not the case for linear regression models, where there are no natural limits placed on how you set any respective X_i. However, you may revise the prediction object to conduct extrapolations. However, you must first feel comfortable with the continuity of the separate ridge functions. If these functions are very noisey or appear not to be differentiable, you might want to avoid all forms of extrapolation.

COMBINATION OF METHODS

In the past, most actuarial research concentrated only on the use of representative scenarios and weighting the results based on the probabilities associated with each representative. I have found that this approach alone does not adequately represent the overall behavior that you obtain when using all of the scenarios. However, the goal of model efficiency is to reduce the entire processing time of the various reserve or capital models. This has been mostly done in the past by either reducing either the number of model points used with the liabilities or assets or by reducing the number of scenarios processed through the model.

Independently, I have observed that PPR models are very effective, not overly sensitive to outliers within the calibration data, and replicate the overall behavior of high dimensional models well. Another nice feature of PPR is that is very quick when asked to evaluate additional input besides that of its training data.

In past experience, I have also been observed that the CLARA algorithm is very effective in selecting representative scenarios. This is due to the fact that the process discovers a majority of the extreme scenarios, which con-

FOOTNOTES

[8] Friedman, J.H. (1984). "A variable span scatterplot smoother." Stanford University Technical Report No. 5 Laboratory for Computational Statistics.

tribute to the tail of the reserve or the capital distribution. When you use representative scenarios and then apply the associated probability weighting to the results, the final results will be very dependent upon how those weights are obtained or used. However, if you do not use these weights at all, but only use the representatives as a training dataset for a predictive model, you can then process all scenarios through the resulting predictive model. For this to work well, the number of representative scenarios needs to be rich enough to adequately span the high dimensional business model. Also, you hope that the predictive model will also adequately model the business model as well.

I will now test the hybrid approach of using representative scenarios as training data and then processing all the scenarios through the predictive model.

Next, I briefly discuss what data is being used.

DATA SOURCES

I[9] described and modeled from over 100 insurance related datasets. In the work below I will concentrate on the 1993 dataset associated with business model 4 and the associated 10,000 interest rate scenarios used in the generation of those values. I also discussed the generation process of these scenarios as well in that paper. I have restricted myself to this specific dataset because I determined that this specific data set had such complex behavior, that if you are able to adequately model the underlying data, the remaining datasets are very easily modeled.

Now let's look at our experiment.

PROCESS OUTLINE

Using the information of the 10,000 scenarios and the 10,000 associated capital values mentioned above, I conducted 100 separate experiments using random samples of 5,000 scenarios for each representative set size. On each of these scenario sets, I apply the CLARA Algorithm and choose separate representative subsets. Once a specific representative set is selected, the representative scenarios, in addition to their associated capital values, are used as the training data for a PPR model. Once the PPR model is trained (or calibrated), the entire sample of 5,000 scenarios are then projected using the resultant PPR model. Using the PPR model, I calculate the CTE90 (which I will refer to as Model). Also, based on the specific sample of 5,000 capital values, I also calculate the CTE90, (this is referred as Actual). I then calculate the relative error associated between Actual and Model, by the formula: RE = (Actual – Model)/Actual.

The CLARA algorithm uses the following distance formula:

$$\sum_{t=0}^{20} (\mid i_{90:t} - i_{90:t}^{P} \mid + \mid i_{10:t} - i_{10:t}^{P} \mid)$$

The 90:t or 10:k notation is used to indicate the 90-day or the 10-year Treasury rates in year t.

RESULTS

Regarding, the CLARA Algorithm, I use representative set sizes of 50, 75, 125, 175, 200, 250, 300, 400, and 500.

As mentioned before, for each of these representative set sizes, I repeat the random sampling of 5,000 scenarios 100 separate times. By conducting this repeated sampling, you can observe the effectiveness of the overall process and approximate the sample error associated in the tests.

Now let's examine the corresponding CLARA Algorithm results for the CTE90 capital estimation. The graph at the end of this article displays the collection of box-and-whisker plots[10] where the CLARA Algorithm is used to select the separate representative scenarios.

For the CTE90 results (shown in the table at the end of the article), notice that these results are liberally biased, but the median results as well as the minimum and maximum relative errors are very tight. I have observed

CONTINUED ON **PAGE 26**

FOOTNOTES

[9] Craighead, S. (2000), "Insolvency Testing: An Empirical Analysis of the Generalized Beta Type 2 Distribution, Quantile Regression, and a Resampled Extreme Value Technique," ARCH, pp. 13-149, available at http://www.soa.org/library/research/actuarial-research-clearing-house/2000-09/2000/arch-2/arch00v26.pdf.

from prior experience using the CLARA algorithm (set to use the sum of absolute value of the differences metric), that the algorithm chooses more tail scenarios than any other published technique that I have used.

Below is an R function that actually takes the scenarios to be clustered in the dataframe clustdata. It then uses the actual interest rate scenarios used, which for this example, is the same dataframe associated with clustdata. If you want to cluster on some transformation of the scenario data, you need to use separate dataframes. Next the actual surplus values are provided in the dataframe observed. Finally, you specify with the ssize variable how many representative scenarios you want to model.

```
randomtest<- function(clustdata, intdata, observed, ssize)
{
clus<- clara(clustdata,ssize,metric="manhattan",samples=100)
sam <- clus$i.med
sam_clusinfo<-clus$clusinfo[,1]
pprsam<- ppr(intdata[sam,],observed[sam],nterms=1,max.terms=125,optlevel=3)
best <- order(pprsam$gofn)[1]
pprsam<- ppr(intdata[sam,],observed[sam],nterms=best,max.terms=best+10,optlevel=3)
yhat<-predict(pprsam,newdata=intdata)
}
```

Below are the commands that I used to generate one example from the experiment regarding 200 representative scenarios. If you want to produce 100 experiment, you will need to insert the code below in a for loop like for(q in 1:100) { }

```
ssize <- 200
bigsampsize<-5000 #let's just choose 5000 samples from the 10000 available
bigsamp<-sample(1:10000,bigsampsize)
dep<-x[bigsamp,] #x is the scenario dataframe
obs<-y[bigsamp,4] #y is the capital values that correspond to each scenario
clu<-x[bigsamp,] #this is the scenario dataframe

yhats <- randomtest(clu, dep,obs,ssize)
bot90<-floor(.90*bigsampsize) +1 #this sets up the logical location of where the tail is
top <- bigsampsize #last scenario
list90 <- seq(bot90,top) #
cte90<- (mean(sort(obs)[list90])- mean(sort(yhats)[list90]))/mean(sort(obs)[list90])
cat( ssize, bigsampsize, i,cte90,sep=" ","\n")
```

ISSUES REGARDING BIAS

Though the results are positively biased and so understate the capital, you can see that the average error is reasonable given the speed enhancement. Of course you may increase the size of the representative set and this will also reduce the bias. Donald Krouse at AEGON has given some insight into what a practitioner may take to reduce this bias. Based on his suggestions, you could introduce weights to the scenario selection process or by experimenting with other various metrics. This may help, because the bias may arise from the fact that the training scenarios may over- or underemphasize certain attributes within the scenarios. Also, the PPR model itself can lead to biased results just due to the fact of how it calibrates as discussed before. You may want to introduce weighting to the calibration process or manipulate other settings to see if the bias can be eliminated or reduced. Currently, I have used other predictive models such as neural networks and other types of machine learning, to eliminate the bias, but I have found that PPR is still superior because it does not suffer from the curse of dimensionality. Furthermore, it simulates the underlying structure of the complex capital model quite well, where these other techniques poorly calibrate to the representative sets.

CONCLUSIONS

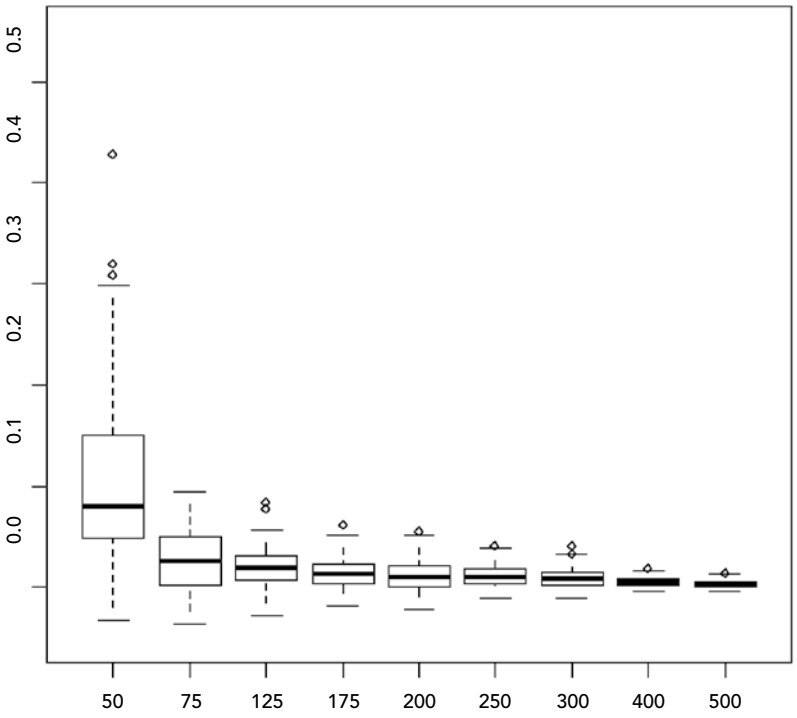
In this issue, I just discussed the results of the combination of CLARA and PPR. In the next issue, I will go into greater detail regarding the use of CLARA and the multiple applications to which I have applied that algorithm. In a future issue, I will also discuss further implementations of PPR as well.

If you want to examine this process further, the following two documents on the SOA Web site discuss this combination approach in more detail:

http://www.soa.org/files/pdf/2008-qc-craighead-dardis-51.pdf

http://www.soa.org/library/newsletters/financial-reporter/2008/june/frn-2008-iss73.pdf ■

CLARA Algorithm with PPR- CTE90



	Min	1st Qu.	Median	Mean	3rd Qu.	Max.
50	-0.03344	0.04874	0.08013	0.1027	0.1484	0.4273
75	-0.03633	0.002239	0.02523	0.02672	0.0496	0.09426
125	-0.02789	0.006763	0.02003	0.01907	0.03103	0.08259
175	-0.01918	0.003258	0.01248	0.01382	0.02327	0.06151
200	-0.02165	0.001014	0.01065	0.01294	0.0219	0.05468
250	-0.01053	0.003376	0.009998	0.01089	0.01766	0.03989
300	-0.01077	0.002906	0.007679	0.008806	0.01451	0.04095
400	-0.00422	0.001636	0.005093	0.005152	0.008281	0.01846
500	-0.00379	0.000105	0.002869	0.003092	0.005355	0.01331

FOOTNOTES

[10] Box-Whisker Plot, see http://en.wikipedia.org/wiki/Box_plot.