Paper 014-2013

Extension Node to the Rescue of the Curse of Dimensionality via Weight of Evidence (WOE) Recoding

Satish Garla, SAS Institute Inc., Cary, NC
Goutam Chakraborty, Oklahoma State University, Stillwater, OK
Andrew Cathie, SAS Institute (NZ) Ltd, Auckland, New Zealand

ABSTRACT

Predictive models in business data mining applications often involve very large data sets with numerous variables many of which are categorical with large number of levels. Predictive models with categorical variables with large number of levels often suffer from the curse of dimensionality. Enhanced Weight of Evidence (WOE) methods can be used to effectively incorporate high dimensional categorical inputs into a data mining model. Weight of evidence technique converts a nominal input into an interval input by using a function of the distribution of a target variable for each level in the nominal input variable.

While WOE recoding can be done via SAS macros and SAS programming, there is no easy way for business users with little programming background to use this technique. However, SAS® Enterprise Miner™ has a facility to create extension nodes which work in the same way as a usual node which can be easily configured via the properties panel using point-and-click interface. This paper explains creation of an extension node in SAS® Enterprise Miner™ that performs WOE recoding. This node can be used easily even by users who are not proficient in SAS programming during data preparation to convert high-cardinality nominal inputs to lower-dimensional interval inputs that will improve model performance.

INTRODUCTION

Most data analysts are familiar with the curse of dimensionality. With advancements in technology and processes, we are now able to acquire not only more rows of data over time but also more columns (characteristics) of data. Paradoxically adding more data (in terms of numbers of variables) does not necessarily improve the ability to develop a more robust predictive model because of the increases in the dimensions of the input vector. This problem is frequently encountered when dealing with nominal inputs. A nominal input variable typically takes on a finite number of discrete values, such as a product code or a region code. Generally, character variables used as inputs in data mining are nominal variables. Numeric variables may be nominal as well; a classic example being a zip code. There are many techniques used to recode non numeric inputs. In this paper we discuss one such method, weight of evidence (WOE) recoding, and show improvement in model performance using this method.

WEIGHT OF EVIDENCE (WOE)

The typical statistician's approach is dummy-coding the levels of the non-numeric input variables. Beyond increasing the dimensionality, a naïve enumeration of levels such as dummy-coding, is also likely to create overgeneralization due to the possibility of over fitting in the training data. Most models assume a continuous association between input and target: small changes in input should result in small changes in expected output. A naïve enumeration method will violate this continuity assumption. Another simple approach to reduce the complexity of a categorical variable is by using a new level called "other" that includes all low populated levels. In this approach, often labeled thresholding, there is always a possibility of losing valuable information by arbitrarily combining different levels.

Another approach to recoding is a different form of enumeration. Unlike a naïve enumeration this method matches the levels of the nominal inputs over the target. In its simplest form, the levels are enumerated by the average of the target within the level. These levels are ranked based on the average value. The level with the smallest average is ranked 1 and the next smallest level is ranked 2 and so on. However in practice, instead of ranking, analysts typical use the actual target average to represent the levels. In this way levels with identical expected response are effectively merged. This approach is commonly called as Weight of Evidence (WOE).

The Weight of Evidence technique has been modified via Bayesian statistical methods which assume, a priori, distributions for the target average within each level. These prior distributions reflect an analyst's state of knowledge about the expected value of the target with each level of the input variable. Observations from the training data are then combined with the priors to form updated estimates of the target average distribution. The expected value of this a posteriori distribution serves as the estimated target value within each level of the input variable. These posterior estimates generally show substantial reduction in the prediction bias compared to the basic weight of evidence approach, especially in non-numeric inputs with tens of hundreds of levels (6, 7, 8). This technique is called as smoothed weight of evidence.

If the Target is Binary,

Smoothed Weight of Evidence (SWOE) for level 'I' is calculated as,

Equation 1:

$$SWOE_i = \log\left(\frac{n_{1i} + rho_1 \,.\, smooth}{n_{0i} + rho_0 \,.\, smooth}\right)$$

Where, n_{1i} is the number of Target=1 for level i, n_{0i} is the number of Target=0 in level i, rho_1 is the proportion of Target=1 in the training data set, rho_0=1- rho_1, and smooth is the value of the smoothing parameter.

If the Target is Interval, a Bayesian-inspired estimate for the target mean is calculated in place of the smoothed weight of evidence. This estimate can be thought of as a weighted average of the overall target mean and the observed target mean in level i and is given as:

Equation 2:

$$\textbf{SMOOTHMEAN} = (\boldsymbol{smooth} \,.\, \hat{Y} + \boldsymbol{ni} \,.\, \hat{Y}\boldsymbol{i}) \,/\, (\boldsymbol{smooth} + \boldsymbol{ni})$$

Where, $\hat{Y}$ is the overall target mean, $\hat{Y}i$ is the level i target mean, ni is the number of cases in level i, and smooth is the value of the smoothing parameter.

Nowadays, Weight-of-Evidence is heavily used in the field of credit scoring to measure the relative risk of an attribute or group level. This technique is available in the Interactive Grouping node which is available only when you have license to Credit Scoring application of SAS Enterprise Miner. We created an extension node of smoothed weight of evidence to make this available to general predictive modeling community. We used SAS macros from the SAS courses Extending SAS® Enterprise Miner™ with User-Written Nodes (DMEX) [2] and Advanced Predictive Modeling using SAS® Enterprise Miner™ 6.1(PMAD61) [8] and modified them to develop this new node.

EXAMPLE

We used the new extension node on the PVA 97 data set. In this data set, there is a variable called DemCluster which is nominal with 54 distinct levels. Figure 1.1 and 1.2 shows the distribution of the DemCluster variable before and after recoding using weight of evidence extension node.

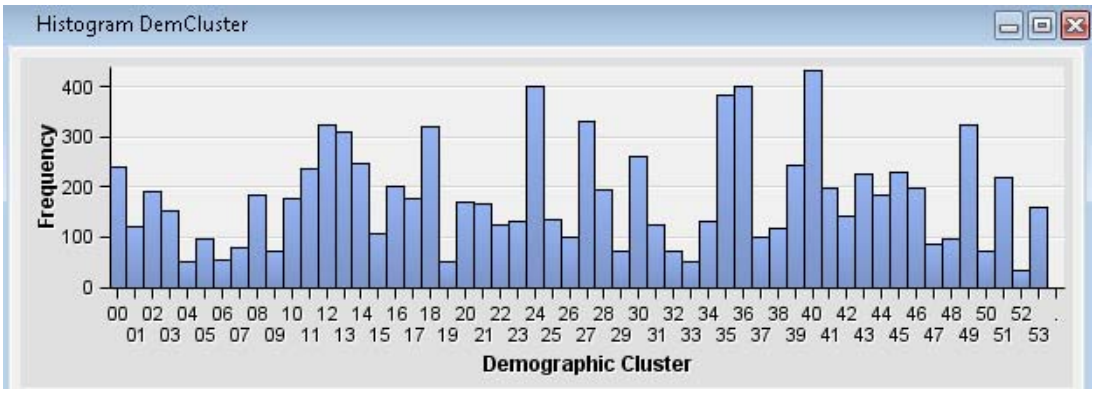


Figure 1.1 Distribution of DemCluster variable: Before WOE recoding

Figure 2.2 Distribution of DemCluster variable: After WOE recoding

DemCluster variable was never identified as important in predicting the target variable until it was transformed to an interval variable via the WOE node. In the transformed case, the variable turned out to be the most significant variable. We built two logistic regression models, one with the actual inputs and the other with recoded inputs. The odds ratio estimate was used to identify the most important variables. Figure 2 shows the process flow from Enterprise Miner.

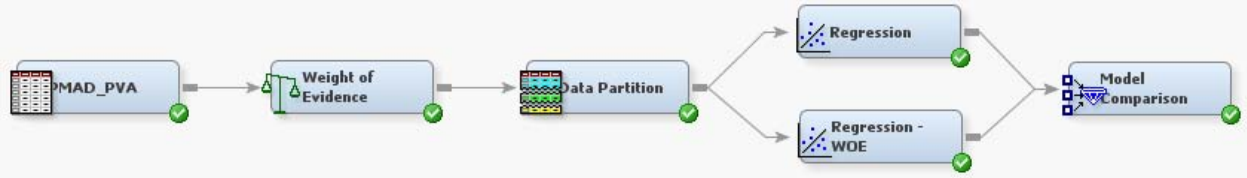


Figure 3 Process Flow from SAS® Enterprise Miner™

We can see a slight improvement in the misclassification rate of the model due to the recoded DemCluster variable. The misclassification rate in the validation data set improved from 0.417 (without recoding) to 0.404 (with WOE recoding). In this dataset we used only one nominal input with 52 levels for recoding. We believe there will be substantial improvement in model performance in data sets with many nominal inputs with tens of hundreds of levels.

Missing values in the input variables are not considered for recoding and therefore the recoded values remains missing. The weight of evidence method has its limitations. The process of applying a WOE transformation to a nominal variable may degrade the predictive performance of the model if interactions are present. For example, two levels of a Zip code may have been transformed to the same interval value based on their univariate relationship to the target. However, there may actually be an interaction effect with another input variable (e.g. Population Density in the zip code), which is now no longer going to be available. In this type of situation the advantages from a reduction on dimensionality might be out-weighed by the loss of the interaction information.

The curse of dimensionality and its solution via the WOE method is relevant beyond the predictive modeling context. For example, in segmentation (clustering) problems, the curse of dimensionality is equally applicable when we have categorical inputs with large number of levels. The dummy variable coding (the default choice by most data miners) is very unappealing in clustering (segmentation) problems because the default choice of Euclidean distance metric to operationalize similarity is difficult to justify with dummy variables. While typical clustering problems often do not have a target variable, in our experience with business data, we have often found that we can use a pseudo-target variable (such as customers' responses to last year's marketing campaigns) to recode such categorical inputs via WOE method.

EXTENSION NODE

Users can develop additional data mining functionalities in SAS® Enterprise Miner™ using extension nodes. These nodes can be seamlessly integrated with the existing nodes and can generate the same outputs as any standard node. There are various macro variables and macro programs available in the tool that a user can use to define node properties, modify metadata and generate a variety of reports.

There are three main building blocks those are needed to create a node:

1. Icon graphics to display in toolbar and diagram workspace

2. An XML file to capture nodes properties and structure

3. SAS server code for the create, train, score and report functions of the node

ICON GRAPHICS

This is relatively straight-forward, with two icons required as .gif files, one 16x16 pixels and the other 32x32 pixels. Both files should have the same name and must reside in separate folders in SAS® Enterprise Miner™ Installation.

Note that there is a cosmetic advantage to having a transparent background in the icons. Tools such as Microsoft Paint do not have the ability to do transparency, however tools such as the open-source GIMP (3) do.

16 X 16

32 X 32

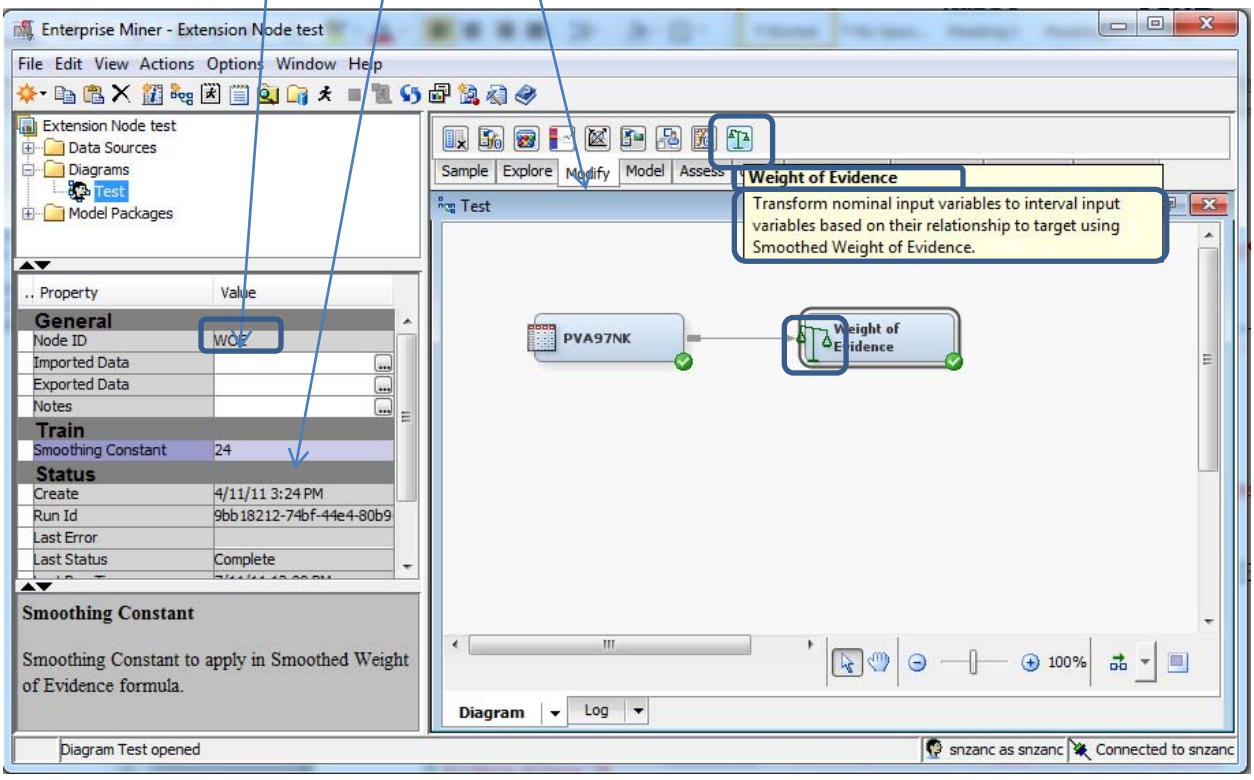
Figure 4: Icon graphics, 16x16 and 32x32 pixels

XML FILE AND NODE PROPERTIES

The XML file provides the link for the Enterprise Miner Java client to know where to find the required icons, where to find the SAS code to operate the node, and what properties the node will have. You will also have to specify in what sub-menu tab this node should appear. Since this node deals with transforming the data, we placed it in the MODIFY menu. The other key parameter is the prefix. All objects created by this node in the process flow diagram will have this prefix. You should keep it reasonably short because of the SAS name restrictions. In addition to these basic settings you should include all the properties that you want to control through node's properties panel. In our case we have defined only one property which defines the smoothing parameter used in the Smoothed Weight of Evidence calculation.

```
<?xml version="1.0"?>
<!DOCTYPE Component PUBLIC
"-//SAS//EnterpriseMiner DTD Components 1.3//EN"
"Components.dtd">
<Component type        = "AF"
           resource    = "com.sas.analytics.eminer.visuals.PropertyBundle"
           serverclass = "EM6"
           name        = "WOE"
           displayName = "Weight of Evidence"
           description = "Transform nominal input variables to interval input variables based on
                          their relationship to target using Smoothed Weight of Evidence."

           group       = "MODIFY"
           icon        = "WOE.gif"
           prefix      = "WOE" >
       <PropertyDescriptors>
              <Property type        = "String"
                        name        = "Location"
                        initial     = "CATALOG" />
              <Property type        = "String"
                        name        = "Catalog"
                        initial     = "SASHELP_EXTN_WOE_WOE_SOURCE" />
              <Property type        = "double"
                        name        = "Smooth"
                        displayName = "Smoothing Constant"
                        description = "Smoothing Constant to apply in Smoothed Weight of
                                       Evidence formula."
                        edit        = "Y"
                        initial     = "24" />
       </PropertyDescriptors>
       <Views>
              <View name="Train">
                     <PropertyRef nameref="Smooth"/>
              </View>
       </Views>
</Component>
```



SAS CODE FOR THE EXTENSION NODE

We like to make extensive use of macro language %PUT statements to add diagnostic information in to the SAS log. There's little overhead from doing this, and it does make any debugging substantially easier.

Error messages: Enterprise Miner makes use of a macro variable EMEXCEPTIONSTRING to surface errors to the user. This can be used in a couple of ways:

1. As a holder for free text:
   %Let EMEXCEPTIONSTRING = Nominal Target variable has been defined. This node can only use binary/interval target variable;
2. As a pointer to a SAS system-supplied message:
   %Let EMEXCEPTIONSTRING = exception.server.METADATA.USE1TARGET ;

The second approach is preferred as this supports internationalization.

CREATE CODE

CREATE code is run when the SAS® Enterprise Miner™ system variable &EM_ACTION has the value CREATE; which happens when a node is placed on to an Enterprise Miner process flow diagram.

In our case the CREATE process does two things:

1. Sets up properties used by the node, in this case the smoothing constant used by the Weight-of-evidence algorithm (ref.
2. Equation **1**). This is subsequently referenced in code as

```
 &EM_PROPERTY_Smooth.
%EM_PROPERTY(name=Smooth,    value=24);
```

3. Registers data sets used by the nodes at runtime.

```
%EM_REGISTER(key=SUM,              type=DATA)
%EM_REGISTER(key=RECODE,           type=DATA)
```

   These are subsequently referenced as `&EM_USER_SUM` and `&EM_USER_RECODE.`

Bearing in mind that an Enterprise Miner process flow can have multiple instances of the same node, and these can run simultaneously, use of these utility macros (`%EM_PROPERTY, %EM_REGISTER`) ensures proper management of the macro variables and data sets, and avoids issues such as name collisions where two or more nodes might attempt to access the same data.

TRAIN CODE

TRAIN code runs when the Enterprise Miner user executes a Run on a process flow, and is detected by &EM_ACTION having the value TRAIN.

In the WOE node, TRAIN code performs these functions:

1. Error checking on the input dataset, such as:
   a. Input (Training) data set exists,
   b. Training data set has a target variable,
   c. There is only one target variable,
   d. Target variable level is Binary or Interval
   e. Input variables exist
   f. Input variables are Nominal
2. Build and execute scoring code that performs the Weight-of-Evidence calculation (Ref. Equation 1)

```
%If %EM_TARGET_LEVEL = BINARY %Then
   %Do ;

LINKTARGET = log((BIN_T + &targetavg * &EM_PROPERTY_smooth) /
                 (BIN_N - BIN_T + (1 - &targetavg) * &EM_PROPERTY_smooth)) ;

   %End ;
%Else
   %Do ;

LINKTARGET = (&targetavg + BIN_T / &EM_PROPERTY_smooth) /
             (BIN_N              / &EM_PROPERTY_smooth + 1) ;

   %End ;
```

3. Adjust Metadata of variables:
   a. Nominal Inputs used in the WOE calculation are set to the role of REJECTED
   b. New Interval variables created by the WOE calculation are set to the role of INPUT

SCORE CODE

SCORE code runs when the &EM_ACTION variable has the value SCORE. In this example there is no specific scoring code to run – it has all been done already in the TRAIN section.

REPORT CODE

REPORT code runs when the &EM_ACTION variable has the value REPORT. This typically is used to generate the diagnostic information that the user sees in the Results window of a node. In this node we generate a pair of histograms for each input variable; one showing the distribution of the input nominal values, and one showing the distribution of the recoded interval values.

```
%EM_Report(
       key=WOEExport,
       viewtype=HISTOGRAM,
       x=%scan(&inputs,&i),
       block=Plots,
       description=Histogram %scan(&inputs,&i),autodisplay=Y
     );
```

Figure 5.1 and 4.2 show the histograms generated in the results window of the WOE node by using the %EM_REPORT macro.

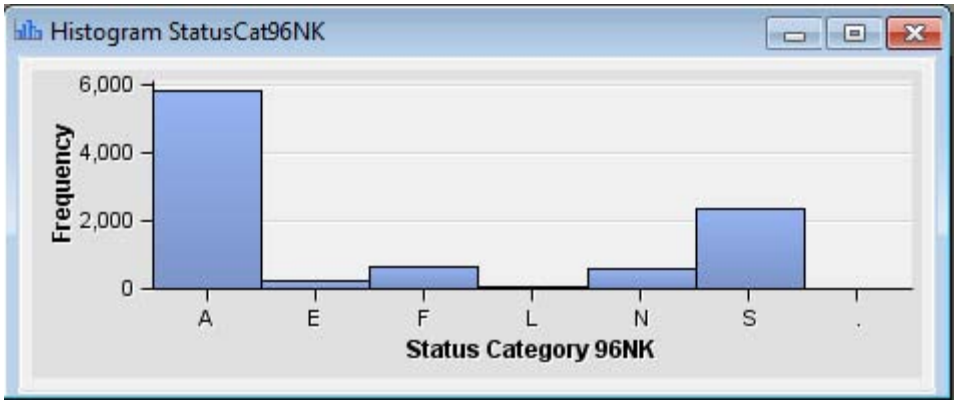


Figure 5.1 Example of Histogram generated in the WOE results window

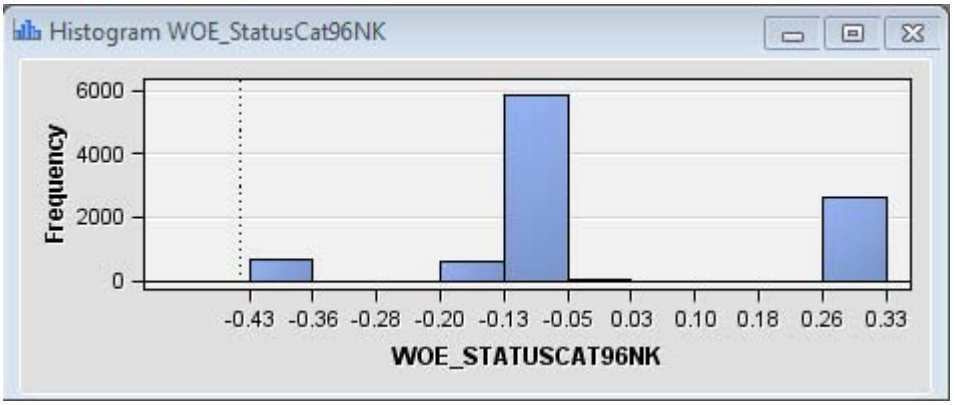


Figure 6.2 Example of Histogram generated in the WOE results window

NOTE: There are slight differences between developing extension nodes for Enterprise Miner 6x installations and Enterprise Miner 7x, 12x installations. For example, in 7.1 and 12.1 installations on Windows 7 for any configurations other than workstation configuration you would need to define a system environment variable named `dminemid.components.extra.dir` that contains the path for the directory that contains the extension xml file. It can be any directory accessible from the machine. For more details on how to define a system environment variable, refer to the below mentioned document.

For workstation configurations, if the EMEXT folder already exists in 'User Home' on Windows 7 (for example, C:\Users\<userid>\EMEXT), it will be found automatically. For any other location, define a system environment variable named `dminemid.components.extra.dir` that contains the path for the directory that contains the extension xml file. It can be any directory accessible from the machine. Otherwise an easy method is to create a folder EMEXT in the directory C:\Users\<userid> and save your XML and .gif files. You will have to create two different folders, gif32 and gif16, in the EMEXT folder to save the gif files.

For more details on extension node deployment refer to the document:

http://support.sas.com/documentation/cdl/en/emxndg/64759/PDF/default/emxndg.pdf

CONCLUSION

There are various approaches followed by data miners for handling nominal inputs in predictive models. Weight of Evidence recoding is an excellent technique to handle high dimensional nominal inputs using the distribution of target variable for each level in the nominal input. The improvement on the model performance will be significant when you have a lot of inputs with hundreds of levels. Analysts should include this technique with other available techniques and evaluate the improvement in the model.

SAS enables users to extend Enterprise Miner functionality by creating extension nodes. Extension nodes can be developed to perform almost any essential data mining activity. To create your own extension node effectively you will need:

- Macro and SAS programming skills
- A development environment for SAS® Enterprise Miner™
- The Extension Nodes guide from SAS

REFERENCES

1. SAS Enterprise Miner 6.1 Extension Nodes Developer's Guide
2. Extending SAS® Enterprise Miner™ with User-Written Nodes (DMEX), SAS Education Services course
3. GIMP: GNU Image Manipulation Program, www.gimp.org
4. Cathie, A., (2011). The Anti-Curse: Creating an Extension to SAS® Enterprise Miner™ Using PROC ARBORETUM. SAS Global Forum
5. Cerrito, P.B., (2007). Text Mining and PROC KDE to Rank Nominal Data. SAS Global Forum
6. Georges, J. E. 2003. "Beyond Expectations: Quantifying Variability in Predictive Models," Proceedings of the M2003 SAS Data Mining Conference. Cary, NC: SAS Institute Inc.

7. Georges, J. E. 2004. “Qualities to Quantities: Using Non-numeric Data in Parametric Prediction” Proceedings of the M2004 SAS Data Mining Conference. Cary, NC: SAS Institute Inc.
8. Advanced Predictive Modeling Using SAS® Enterprise Miner™ 6.1 Course Notes developed by Jim Georges, and revised by Dan Kelly and Bob Lucas.

CONTACT INFORMATION

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

Satish Garla, SAS Institute Inc., Cary NC, Email: satish.garla@sas.edu

Satish Garla is a Risk Analytics consultant at SAS, Cary, NC. He has three years of experience in the field of data mining and text mining. He is a certified Predictive Modeler using SAS® Enterprise Miner anda certified Advanced SAS programmer. As a student, he won several awards in events such as student poster competitions and annual data mining shootouts and SAS® student Ambassador award. Satish holds a master's degree in Management Information Systems and SAS® and OSU Data Mining Certificate from Oklahoma State University, Stillwater.

Goutam Chakraborty, Oklahoma State University, Stillwater OK, Email: goutam.chakraborty@okstate.edu

Dr. Goutam Chakraborty is a professor of marketing and founder of SAS® and OSU data mining certificate and SAS® and OSU business analytics certificate at Oklahoma State University. He has published in many journals such as Journal of Interactive Marketing, Journal of Advertising Research, Journal of Advertising, Journal of Business Research, etc. He has chaired the national conference for direct marketing educators for 2004 and 2005 and co-chaired M2007 data mining conference. He has over 25 years of experience in using SAS® for data analysis. He is also a Business Knowledge Series instructor for SAS®.

Andrew Cathie, SAS Institute (NZ) Ltd, Auckland, New Zealand, Email: Andrew.Cathie@sas.com

Andrew Cathie is a senior consultant at SAS, New Zealand. He has 20+ years of experience in the areas of data mining, predictive modeling, data warehousing, business intelligence and CRM systems.

TRADEMARKS