

Data Mining Techniques: Predictive Analytics on Big Data

Course Notes

Data Mining Techniques: Predictive Analytics on Big Data Course Notes was developed by Robert Blanchard, Catherine Truxillo, Chip Wells, and Terry Woodfield. Editing and production support was provided by the Curriculum Development and Support Department.

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.

Data Mining Techniques: Predictive Analytics on Big Data Course Notes

Copyright © 2016 SAS Institute Inc. Cary, NC, USA. All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

Book code E70583, course code LWMLDMBD/MLDMBD, prepared date
04Feb2016.

LWMLDMBD_001

ISBN 978-1-62960-221-9

Table of Contents

Course Description	v
Chapter 1 Introduction.....	1-1
1.1 Introduction to Big Data and SAS Big Data Tools.....	1-3
1.2 Chapter Summary.....	1-10
Chapter 2 Data Exploration	2-1
2.1 Data Exploration and Clustering (SAS Visual Statistics).....	2-3
Demonstration: Introduction to the SAS Visual Statistics Environment	2-12
Exercises.....	2-16
Cluster Analysis in SAS Visual Statistics.....	2-17
Demonstration: Building a Cluster Analysis	2-26
Exercises.....	2-29
Segmentation Ideas and SAS Visual Statistics Cluster Details (Self-Study)	2-30
2.2 Data Exploration and Dimension Reduction (SAS Enterprise Miner).....	2-38
Principal Components Analysis	2-44
Demonstration: Principal Components Analysis	2-53
2.3 Solutions	2-63
Solutions to Exercises.....	2-63
Solutions to Student Activities (Polls/Quizzes)	2-65
Chapter 3 Analysis Methods for Categorical Targets	3-1
3.1 Categorical Targets (SAS Visual Statistics).....	3-3
Logistic Regression.....	3-3
Demonstration: Creating a Logistic Regression in SAS Visual Analytics	3-22
Exercises.....	3-26
Interactive Group By Processing	3-28
Demonstration: Adding a Group By Variable to a Logistic Regression	3-34
Decision Trees.....	3-36
Demonstration: Creating and Cultivating a Decision Tree in SAS Visual Analytics	3-46
Exercises.....	3-51

Tree Split Search in SAS Visual Statistics (Self-Study).....	3-53
3.2 Categorical Targets (SAS In-Memory Statistics).....	3-62
Decision Trees in PROC IMSTAT	3-63
Demonstration: Growing a Decision Tree	3-72
Demonstration: Assessing a Decision Tree	3-91
An Introduction to Random Forests (Self-Study)	3-101
Demonstration: Growing a Forest of Trees.....	3-119
Demonstration: Scoring Data with the RANDOMWOODS Score Code	3-123
Logistic Regression in PROC IMSTAT	3-126
Demonstration: Developing and Assessing a Logistic Regression Model.....	3-129
3.3 Solutions	3-145
Solutions to Exercises.....	3-145
Solutions to Student Activities (Polls/Quizzes)	3-151
Chapter 4 Analysis Methods for Interval Targets.....	4-1
4.1 Interval Targets (SAS Visual Statistics).....	4-3
Linear Regression.....	4-3
Demonstration: Building a Linear Regression	4-20
Exercises.....	4-32
Generalized Linear Models.....	4-34
Demonstration: Building a GLM Model.....	4-46
Exercises.....	4-51
4.2 Interval Targets (SAS In-Memory Statistics).....	4-52
Demonstration: Generalized Linear Models for Count Data.....	4-55
Demonstration: Generalized Linear Models for Interval Targets.....	4-72
Demonstration: Fitting a Lognormal Model to an Interval Target	4-81
4.3 Interval Targets (SAS Enterprise Miner).....	4-84
Demonstration: Fitting a Zero-Inflated Poisson Model Using the HP GLM Node in SAS Enterprise Miner	4-86
4.4 Solutions	4-90
Solutions to Exercises.....	4-90
Solutions to Student Activities (Polls/Quizzes)	4-93

Course Description

This course introduces applications and techniques for assaying and modeling large data, and presents basic and advanced modeling strategies, such as group-by processing for linear models, random forests, generalized linear models, and mixture distribution models. Students perform hands-on exploration and analyses using tools such as SAS Enterprise Miner, SAS Visual Statistics, and SAS In-Memory Statistics.

To learn more...



For information about other courses in the curriculum, contact the SAS Education Division at 1-800-333-7660, or send e-mail to training@sas.com. You can also find this information on the web at <http://support.sas.com/training/> as well as in the Training Course Catalog.



For a list of other SAS books that relate to the topics covered in this course notes, USA customers can contact the SAS Publishing Department at 1-800-727-3228 or send e-mail to sasbook@sas.com. Customers outside the USA, please contact your local SAS office.

Also, see the SAS Bookstore on the web at <http://support.sas.com/publishing/> for a complete list of books and a convenient order form.

Chapter 1 Introduction

1.1	Introduction to Big Data and SAS Big Data Tools	1-3
1.2	Chapter Summary.....	1-10

1.1 Introduction to Big Data and SAS Big Data Tools

Objectives

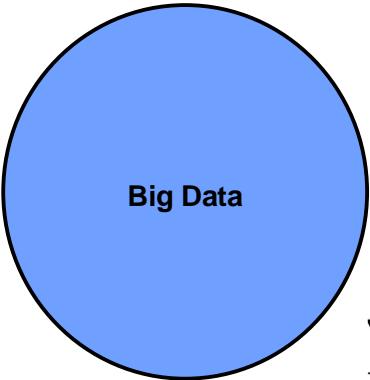
- Define big data.
- Identify the strengths of using SAS Enterprise Miner High-Performance nodes (High-Performance Data Mining), SAS In-Memory Statistics, and SAS Visual Analytics.

2

Big Data

“Big data is data of a very large size, typically to the extent that its manipulation and management present significant logistical challenges.”

» *Oxford English Dictionary*



Big Data



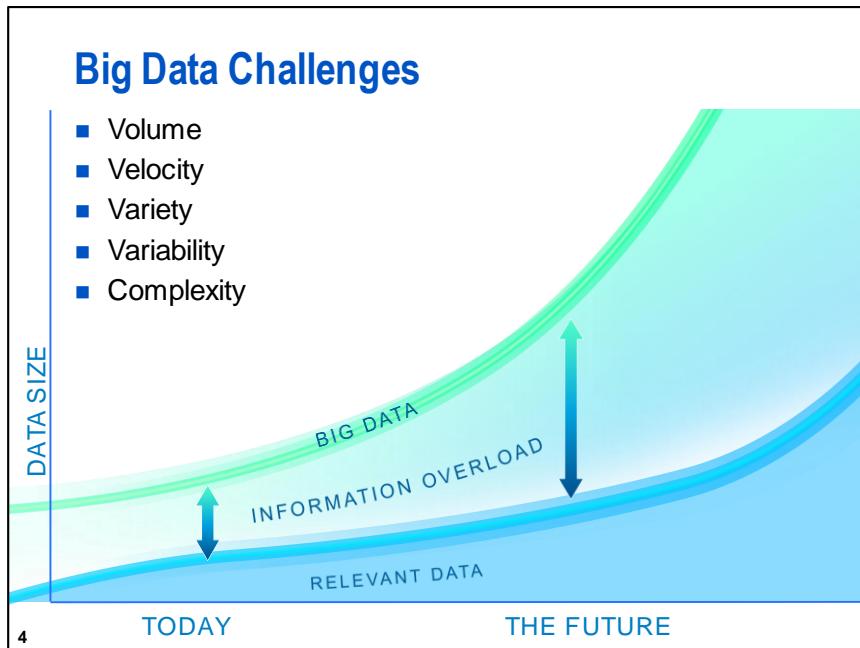
3

According to the Oxford English Dictionary:

“Big data is data of a very large size, typically to the extent that its manipulation and management present significant logistical challenges.”

Big data poses challenges that can make it difficult or even infeasible to gain knowledge from data or productionalize an analytic process. Some difficulties arise from limits on computation. Other complications can occur because the analytic process is reloading data into memory across analytical steps.

To further illustrate the complexities that accompany big data, consider the concept of the “five Vs.”



Volume: Organizations collect data from a variety of sources, including business transactions, social media, and information from sensor or machine-to-machine data. In the past, storing it posed a problem, but new technologies (such as Hadoop) can ease the burden.

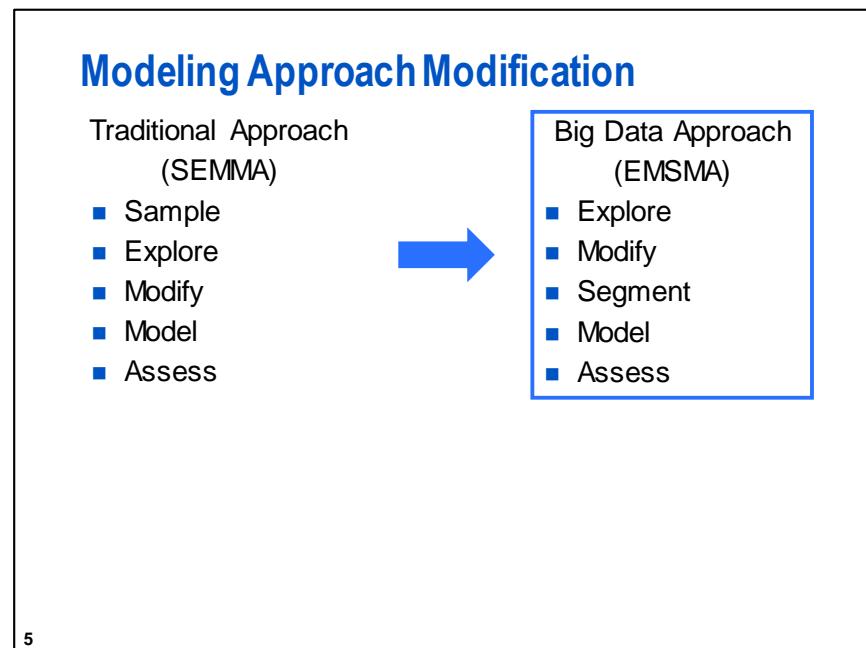
Velocity: Data streams in at an unprecedented speed and must be dealt with in a timely manner. RFID tags, sensors, and smart metering drive the need to deal with torrents of data in near real time.

Variety: Data comes in all types of formats from structured, numeric data in traditional databases to unstructured text documents, email, video, audio, stock ticker data, and financial transactions.

Variability: In addition to the increasing velocities and varieties of data, data flows can be highly inconsistent with periodic peaks. Is something trending in social media? Daily, seasonal, and event-triggered peak data loads can be challenging to manage. This is especially challenging with unstructured data because the formats and technologies vary widely.

Complexity: Data comes from multiple sources, which makes it difficult to link, match, cleanse, and transform data across systems. However, it is necessary to connect and demonstrate relationships, hierarchies, and multiple data linkages. If you do not, your data can become unmanageable.

In this course, you learn to use three tools that data scientists use to navigate and avoid the logistical challenges that accompany data mining big data. The three tools discussed are SAS Visual Statistics, SAS In-Memory Statistics (PROC IMSTAT), and SAS High-Performance Data Mining (HPDM) nodes in SAS Enterprise Miner. You also apply a new modeling methodology that is well suited for big data: EMSMA (Explore, Modify, Segment, Model, and Assess).



5

SAS In-Memory Interfaces for Hadoop

Interface	Purpose	Product
High-Performance Analytics Procedures	Perform complex analytical computations on Hadoop tables within the data nodes of the Hadoop distribution via SAS procedure language. HPDS2 enables manipulation of data structure (column derivation).	SAS High-Performance Analytics Solutions
SAS Visual Analytics, SAS Visual Statistics	A web interface to generate graphical visualizations of data distributions and relationships on Hadoop tables preloaded into memory within the data nodes of the Hadoop distribution.	SAS Visual Analytics
IMSTAT Procedure	A programming interface to perform complex analytical calculations on Hadoop tables preloaded into memory within the data nodes of the Hadoop distribution.	SAS In-Memory Statistics
DS2	A SAS proprietary language for table manipulation that is a base language and executes in the data nodes of a distribution database.	SAS In-Database Code Accelerator
Collectively called SAS In-Memory Analytics		

6

Why might a data scientist choose one SAS tool or another? There is some overlap in capabilities among the tools, but each tool has its strengths. The High-Performance Analytics nodes in SAS Enterprise Miner perform complex analytical computations on Hadoop tables within data nodes of the Hadoop distribution via SAS procedure language.

Data does not persist after an analytic procedure completes, which can increase computation time. Not allowing the persistence of data in-memory for a distributed environment is seen as a disadvantage of high-performance analytic procedures.

Both SAS Visual Statistics and the IMSTAT procedure use the SAS LASR Analytic Server, which enables the persistence of data in-memory for a distributed environment. That is, the SAS LASR Analytic Server provides a multi-user concurrent environment. This means that multiple users can access the same raw data in a shared memory environment while performing different analytical tasks.

SAS Visual Statistics is a web interface that generates graphical visualizations of data distributions and relationships on Hadoop tables that are preloaded into memory within the data nodes of the Hadoop distribution.

The IMSTAT procedure is a programming procedure in SAS that performs complex analytical calculations on preloaded Hadoop tables.

Why Use SAS High-Performance Analytics?

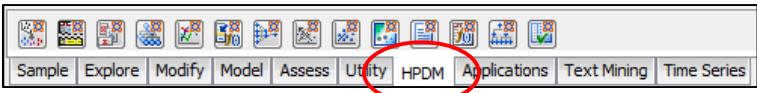
SAS High-Performance Analytics has the following advantages:

- handles big data using multithreaded procedures that execute concurrently
- simplifies many common task associated with applied analysis
- contains a versatile data mining workbench
- easily drives insights in a self-sufficient and automated manner

7

SAS Enterprise Miner has a subset of high-performance nodes that implement high-performance data mining (HPDM) procedures. The HPDM procedures can run multithreaded processing. This means that SAS divides the data between CPUs for quicker processing. This can substantially improve performance through a reduced execution time for large data sets. The rich set of pre-built algorithms in SAS Enterprise Miner provides a versatile set of tools for data mining and other applied analyses. The graphical user interface of SAS Enterprise Miner provides an intuitive approach to building analyses for data mining.

High-Performance Data Mining Nodes



The screenshot shows the SEMMA tools palette with various tabs at the top: Sample, Explore, Modify, Model, Assess, Utility, HPDM (highlighted with a red circle), Applications, Text Mining, and Time Series.

HP Cluster	HP Principal Components
HP Data Partition	HP Regression
HP Explore	HP SVM
HP Forest	HP Text Miner
HP GLM	HP Transform
HP Impute	HP Tree
HP Neural	HP Variable Selection

8

The HP nodes are found on the HPDM (High-Performance Data Mining) tab in the SEMMA tools palette. The HP nodes, along with a brief summary, are listed below.

- **HP Cluster:** Performs k -means clustering analysis in distributed computing environments.
- **HP Data Partition:** Generates an identifier variable that identifies the observations to be used for training and for validation.
- **HP Explore:** Computes summary statistics in the HP environment.
- **HP Forest:** Creates random forest models in the high-performance environment.
- **HP GLM:** Fits a generalized linear model in a distributed computing environment.
- **HP Impute:** Imputes missing values using high-performance procedures.
- **HP Neural:** Creates neural network models in the high-performance environment.
- **HP Principal Components:** Performs principal component analysis.
- **HP Regression:** Uses high-performance linear and logistic regression models.
- **HP SVM:** Uses the support vector machine procedure to construct separating hyper-planes that maximize the margin between two classes.
- **HP Text Miner:** Performs text mining in the high-performance environment.
- **HP Transform:** Creates transformation variables using high-performance procedures.
- **HP Tree:** Creates tree models in the high-performance environment.
- **HP Variable Selection:** Performs supervised and unsupervised variable selection using high-performance procedures.

Why Use SAS In-Memory Statistics (PROC IMSTAT)?

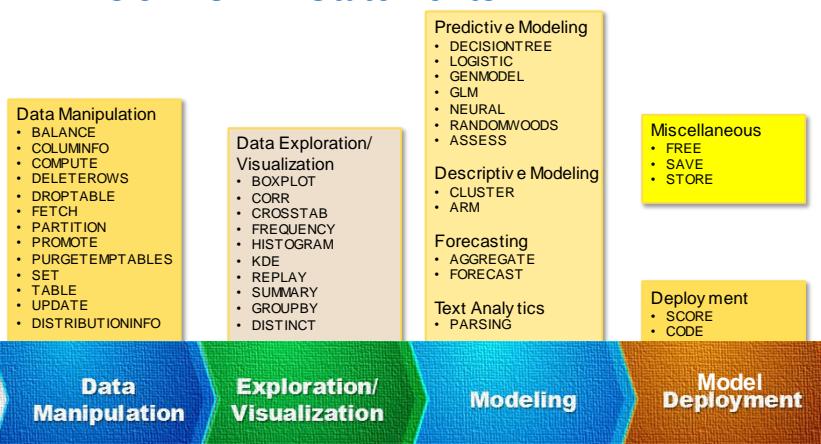
SAS In-Memory Statistics has the following advantages:

- handles big data with an in-memory infrastructure that can eliminate costly data movement and reduce data latency to provide rapid analysis
- enables interactive programming so that multiple users can analyze data extremely quickly
- provides ultimate flexibility and customization that comes with a coding environment

9

SAS In-Memory Statistics (PROC IMSTAT) is an in-memory engine designed for scalable processing. PROC IMSTAT uses in-memory analytics to perform analyses that range from data exploration, visualization, and descriptive statistics to model building with advanced statistical and machine learning algorithms and scoring new data. In addition, in-memory statistics procedures are dedicated to model deployment and implementation.

PROC IMSTAT Statements



10

Another advantage of using PROC IMSTAT is the flexibility that comes with a coding environment. The user can run customized scripts that can automate processes and schedule tasks.

Why Use SAS Visual Statistics?

SAS Visual Statistics has the following advantages:

- handles big data with in-memory analytical processing and can use the Hadoop Distributed File System (HDFS)
- has powerful visual data exploration capabilities
- is a business intelligence tool
- provides interactive model-building tools that can quickly surface visual insights into your data
- requires no coding

11

SAS Visual Analytics and SAS Visual Statistics (an add-on to Visual Analytics) provide a graphical user interface (GUI) that overlays in-memory technologies and can use Hadoop distributed file systems. The full functionality of SAS Visual Analytics can be exploited without any coding required by the user. SAS Visual Analytics functionality includes visual data exploration capabilities, built-in analytic analyses, and business intelligence applications.

SAS Visual Analytics is a business intelligence tool. This means that it can support a wide range of business functions, including data exploration, analytics, dashboard reporting capabilities, scalable deployment options. This tool also has mobile applications.

Which Tool Should Be Used?

Consider the following when you decide which tool should be used:

- depends on the business objective
- relies on a company's technology infrastructure
- depends on the preference and skill strengths of the data scientist

12

The data scientist should consider the following when deciding which tool should be used for an analytic project:

- First, the business objective should be considered. For example, SAS Visual Analytics should be the chosen tool if understanding data relationships and reporting those relationships to management is the primary goal of the project.
- Secondly, the company's technology infrastructure setup should be considered. For example, does the company have the Hadoop Distributed File System Installed? If the answer is "no," then the high-performance nodes in SAS Enterprise Miner might be the method to use.
- Lastly, the preferences and skill strengths of the data scientist should be considered when you choose which tool to use for solving the immediate analytic challenge. Most data scientists have the skills and knowledge to use all three big data tools. However, leveraging the tool that is most familiar to the data scientist can reduce time spent on the analysis. For example, PROC IMSTAT should be the tool of choice if the data scientist prefers the flexibility of coding and finds the working environment easier than an environment on a graphical user interface.

1.2 Chapter Summary

Data scientists need knowledge of the tools that can handle increasingly large and complex data. The SAS Enterprise Miner high-performance nodes, SAS In-Memory Statistics (PROC IMSTAT), and SAS Visual Analytics are three tools that scale well to handle larger amounts of data in an efficient manner.

Each tool presents its own strengths and capabilities. When choosing a tool, the data scientist should consider the current business objective, the company's technology infrastructure setup, and the skill strengths of the data scientist.

Chapter 2 Data Exploration

2.1 Data Exploration and Clustering (SAS Visual Statistics)	2-3
Demonstration: Introduction to the SAS Visual Statistics Environment	2-12
Exercises.....	2-16
Cluster Analysis in SAS Visual Statistics.....	2-17
Demonstration: Building a Cluster Analysis.....	2-26
Exercises.....	2-29
Segmentation Ideas and SAS Visual Statistics Cluster Details (Self-Study).....	2-30
2.2 Data Exploration and Dimension Reduction (SAS Enterprise Miner)	2-38
Principal Components Analysis	2-44
Demonstration: Principal Components Analysis.....	2-53
2.3 Solutions	2-63
Solutions to Exercises	2-63
Solutions to Student Activities (Polls/Quizzes)	2-65

2.1 Data Exploration and Clustering (SAS Visual Statistics)

Objectives

- Describe the benefits and functionality of SAS Visual Statistics.
- Access SAS Visual Statistics functionality within the SAS Visual Analytics environment.
- Examine the **VS_BANK** data set.
- Work with the interface.

3



The SAS Visual Statistics functionality can be accessed from the Data Explorer task in SAS Visual Analytics. SAS Visual Statistics adds interactive modeling functionality to the Data Explorer task. Its functionality does not extend to the Visual Analytics Reporter.

What Is SAS Visual Statistics?

With the SAS Visual Statistics add-on, you can use SAS Visual Analytics to rapidly build and modify predictive models.

SAS Visual Statistics models

- take advantage of the SAS LASR Analytic Server to persist and analyze data in-memory
- enable concurrent access to in-memory data so that multiple users can formulate and refine models

4

continued...

Functional Overview

- use SAS Visual Analytics to prepare and explore data
- access the SAS Visual Statistics functionality for advanced model building
- use predictive and exploratory modeling techniques
 - linear and logistic regression analyses
 - generalized linear models
 - decision trees
 - clustering
- work with detailed data at the observational level
- add or remove variables, and filter interactively
- create interactions spontaneously

5

continued...

Functional Overview

- identify and remove influential outliers interactively
- apply Group By (that is, create separate models by group-by variables or segments)
- perform model comparison via lift charts, ROC charts, concordance statistics, misclassification tables, and other model comparison metrics
- export model score code
- create a secure, multi-user environment for concurrent access to in-memory data via the LASR Analytic Server.

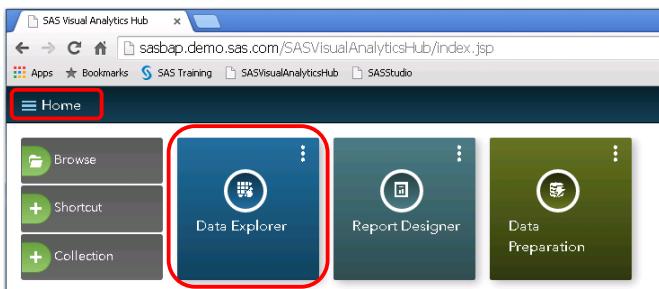
6



In contrast to traditional SAS tools that feature Group By functionality, Visual Statistics can perform analyses on BY groups without first sorting the data.

Accessing SAS Visual Statistics Functionality

From the SAS Visual Analytics Hub (also known as the Home Page), select the **Data Explorer** task.



7

Users need an account to sign in to the SAS Visual Analytics Hub. Access to Visual Analytics and the Visual Statistics add-on functionality is determined by the roles and capabilities assigned to the account with which you sign in. You use SAS Management Console to administer SAS accounts and their respective roles and capabilities.

A URL for accessing SAS Visual Statistics could be similar to the following:

<http://sasbap.demo.sas.com/SASVisualStatistics>

To sign out of SAS Visual Analytics, click **Log Off** in the upper right corner. You are prompted to save your project if changes were made since it was last saved.

 Position your mouse pointer over **Log Off** to display the account that is currently signed in.

By default, you are automatically signed out after 30 minutes of inactivity. Any actions that you took that are not saved are lost. Be sure to save your work at regular intervals. (Select **File** \Rightarrow **Save**, select **File** \Rightarrow **Save As**, or click the **Save Exploration** tool on the toolbar.)

When you access SAS Visual Statistics functionality through the Data Explore task on the Visual Analytics Home Page or via a direct URL, the Open Data Source window appears. In the window, you either select a data source or open an existing project. Make sure that your data are already loaded into memory before attempting to open the data source in Visual Analytics.

SAS Visual Analytics New Exploration Window

Use the New Exploration window to do one of the following:

- click a data source to begin a new exploration
- select an existing exploration and open it

8

Data are loaded into memory by an administrator through the Visual Analytics Administrator tool. Libraries and tables must be registered through SAS Management Console.

Data Explorer Task

9

- The menu bar offers common tasks, such as creating a new exploration.
- The toolbar provides quick access to the most commonly used features.
- The Data pane enables you to manage the data that are used in your projects.
- The Summary table displays detailed statistics about the current visualization.
- The Model pane displays modeling results.
- The right pane's tabs enable you to set data roles, properties, and filters.

SAS Visual Statistics Icons

-  Create a new linear regression
-  Create a new logistic regression
-  Create a new generalized linear model
-  Create a new decision tree model
-  Create a new cluster model
-  Compare two or more models

10



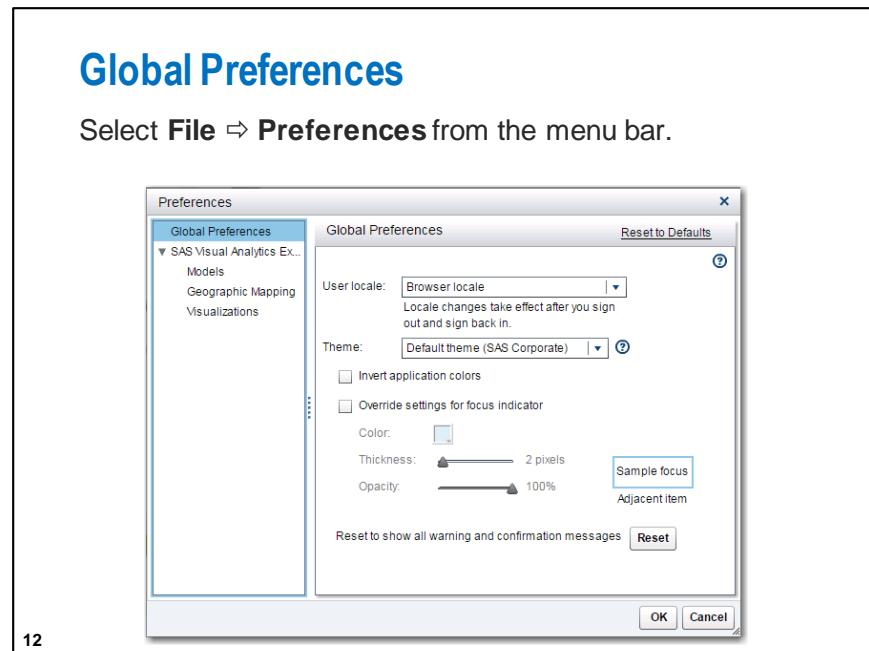
Decision tree models are available in the current version of SAS Visual Analytics without the Visual Statistics add-on. However, the Visual Statistics add-on augments and extends the tree functionality. Further information is provided in the Decision Trees section of the “Analysis Methods for Categorical Targets” chapter.

Specifying Preferences

Two types of preferences are available.

- Global – preferences that are shared between all of the SAS Visual Analytics applications
- Application-specific – preferences that are specific to an exploration
 - p -value precision

11



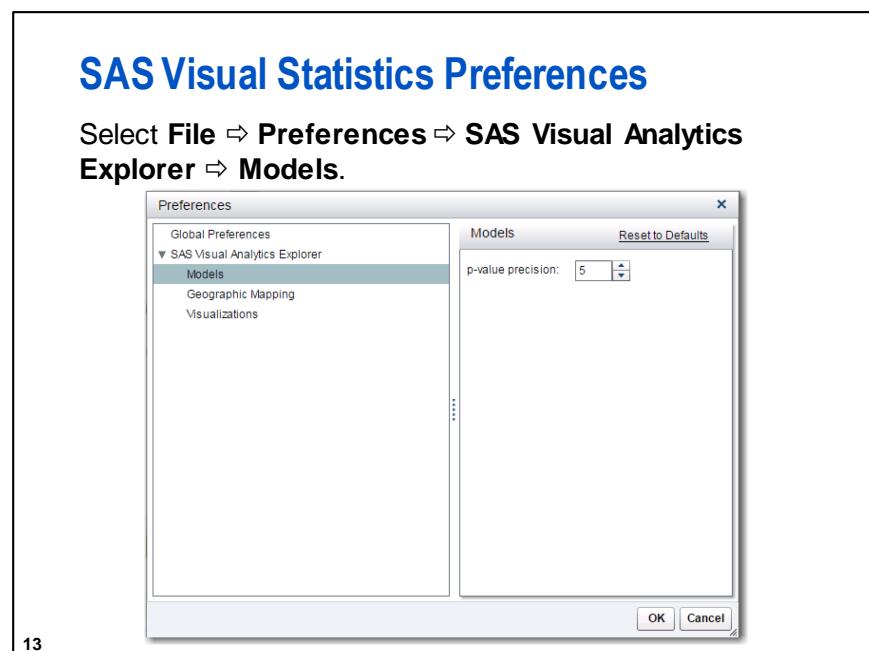
Select a user locale to specify your language and geographic region.

Select a theme to change the color scheme and other visual settings for all of your SAS web applications. The SAS Light theme was chosen for this course.

Select the **Invert application colors** check box to invert all of the colors in your SAS web applications. You can also hold down the Ctrl key and press ~ to invert the colors while you use the application.

Select the **Override settings for focus indicator** check box to change the color, thickness, and opacity of the focus in your SAS web applications.

If the user selected to not show warning and confirmation messages sometime during the Explorer session, clicking the **Reset** button enables all warning and confirmation messages to be displayed again.



The **p-value precision** setting specifies the number of decimal places that are used to show *p*-values.

Idea Exchange

- How many rows does a typical data set (table) that you work with contain?
- How many variables (fields) are in it?

Course Data Introduction

Anonymized and transformed campaign data from the following:

- large financial services firm's accounts
- including home equity lines of credit, loans, and so on
- more than half-a-year time interval
- focusing on direct and indirect promotions
- with three target variables (**b_tgt** is the focus.)
- with an account identifier



15

Data Description

The data set used in this course is a fairly large data set. It has more than one million rows (or observations) and 24 columns (or fields). Three target variables are provided in the course data. The data set **VS_BANK** consists of observations taken from a large financial services firm's accounts. Accounts in the data represent consumers of home equity lines of credit, automobile loans, and other types of short- to medium-term credit instruments.

The data were anonymized and transformed to conform to the following description:

A campaign interval for the bank runs for half of a year. A campaign is used here to denote all marketing efforts that provide information about and motivate the contracting (purchase) of the bank's financial services products. Campaign promotions are categorized into *direct* and *indirect*. Direct promotions consist of sales offers that involve an incentive. These offers are directed to a particular account. Indirect promotions are marketing efforts that do not involve an incentive.

 In addition to the account identifier (**Account ID**), the variables below are in the data set.

Target variables quantify account responses over the current campaign season.

Name	Label	Description
B_TGT	Tgt Binary New Product	A binary target variable. Accounts coded with a 1 contracted for at least one product in the previous campaign season. Accounts coded with a zero did not contract for a product in the previous campaign season.
INT_TGT	Tgt Interval New Sales	The amount of financial service's product (sum of sales) per account in the previous campaign season, denominated in U.S. dollars.
CNT_TGT	Tgt Count Number New Products	The number of financial service's products (count) per account in the previous campaign season.

Categorical-valued inputs summarize account-level attributes related to the propensity to buy products and other characteristics related to profitability and creditworthiness. These variables were transformed to anonymize account-level information and to mitigate quality issues related to excessive cardinality.

Name	Label	Description
CAT_INPUT1	Category 1 Account Activity Level	<p>A three-level categorical variable that codes the activity of each account.</p> <ul style="list-style-type: none"> • X → high activity. The account enters the current campaign period with many products. • Y → average activity. • Z → low activity.
CAT_INPUT2	Category 2 Customer Value Level	A five-level (A-E) categorical variable that codes customer value. For example, the most profitable and creditworthy customers are coded with an A.

Interval-valued inputs provide continuous measures on account-level attributes related to the recency, frequency, and sales amounts (**RFM**). These variables have been transformed to anonymize account-level information. All measures below correspond to activity prior to the current campaign season.

Name	Label	Description
RFM1	RFM1 Average Sales Past 3 Years	Average sales amount attributed to each account over the past three years
RFM2	RFM2 Average Sales Lifetime	Average sales amount attributed to each account over the account's tenure
RFM3	RFM3 Avg Sales Past 3 Years Dir Promo Resp	Average sales amount attributed to each account in the past three years in response to a direct promotion

Name	Label	Description
RFM4	RFM4 Last Product Purchase Amount	Amount of the last product purchased
RFM5	RFM5 Count Purchased Past 3 Years	Number of products purchased in the past three years
RFM6	RFM6 Count Purchased Lifetime	Total number of products purchased in each account's tenure.
RFM7	RFM7 Count Prchsd Past 3 Years Dir Promo Resp	Number of products purchased in the previous three years in response to a direct promotion
RFM8	RFM8 Count Prchsd Lifetime Dir Promo Resp	Total number of products purchased in the account's tenure in response to a direct promotion
RFM9	RFM9 Months Since Last Purchase	Months since the last product purchase
RFM10	RFM10 Count Total Promos Past Year	Number of total promotions received by each account in the past year
RFM11	RFM11 Count Direct Promos Past Year	Number of direct promotions received by each account in the past year
RFM12	RFM12 Customer Tenure	Customer tenure in months.

Demographic variables describe the profile of each account in terms of income, homeownership, and other characteristics.

Name	Label	Description
DEMOG_AGE	Demog Customer Age	Average age in each account's demographic region
DEMOG_GENF	Demog Female Binary	A categorical variable that is 1 if the primary holder of the account is female and 0 otherwise.
DEMOG_GENM	Demog Male Binary	A categorical variable that is 1 if the primary holder of the account is male and 0 otherwise
DEMOG_HO	Demog Homeowner Binary	A categorical variable that is 1 if the primary holder of the account is a homeowner and 0 otherwise.
DEMOG_HOMEVAL	Demog Home Value	Average home value in each account's demographic region
DEMOG_INC	Demog Income	Average income in each account's demographic region
DEMOG_PR	Demog Percentage Retired	The percentage of retired people in each account's demographic region



Introduction to the SAS Visual Statistics Environment

In this demonstration, you access and examine the SAS Visual Statistics interface, customize your preferences, and explore relationships among variables in the course data.

1. From the Windows desktop, launch your web browser. When the browser opens, click **SAS Visual Analytics Hub**.
2. Sign in to SAS. Use the user ID and password supplied by your instructor.

You need to start the SAS LASR Analytic Server and load the **VS_BANK** table into memory. Your instructor can walk you through this process using the Administrator tool.
3. After you sign in to the Hub, click the Side menu (☰ or “Hamburger” button) next to **Home**.
4. Select **Administrator**.
5. On the LASR Servers tab, select **LASR Analytic Server - Server1** and click the **Start the Selected (checked) Servers** button to start the server.
6. Under Folders, select **Shared Data** ⇔ **Source Data**.
7. Right-click **VS_BANK** and select **Load a Table**.
8. Right-click **PVA97NK** and select **Load a Table**. This data set is used in the exercises later.
9. Click the **Home Page** button in the top left corner to return to the Home Page.
10. Click the **Data Explorer** button.
11. The Open a Data Source window appears. Select **VS_BANK** and click **Open**.

The Explorer shows the variables that are listed in the left pane.

The variable **tgt Binary New Product (b_tgt)** is the primary dependent variable for categorical response modeling in this course. It is a binary flag that codes responders with *1* and non-responders with *0*. Because it is numeric, it is treated as an interval value by default.

12. Right-click **tgt Binary New Product** and change its measurement level to **Category**.

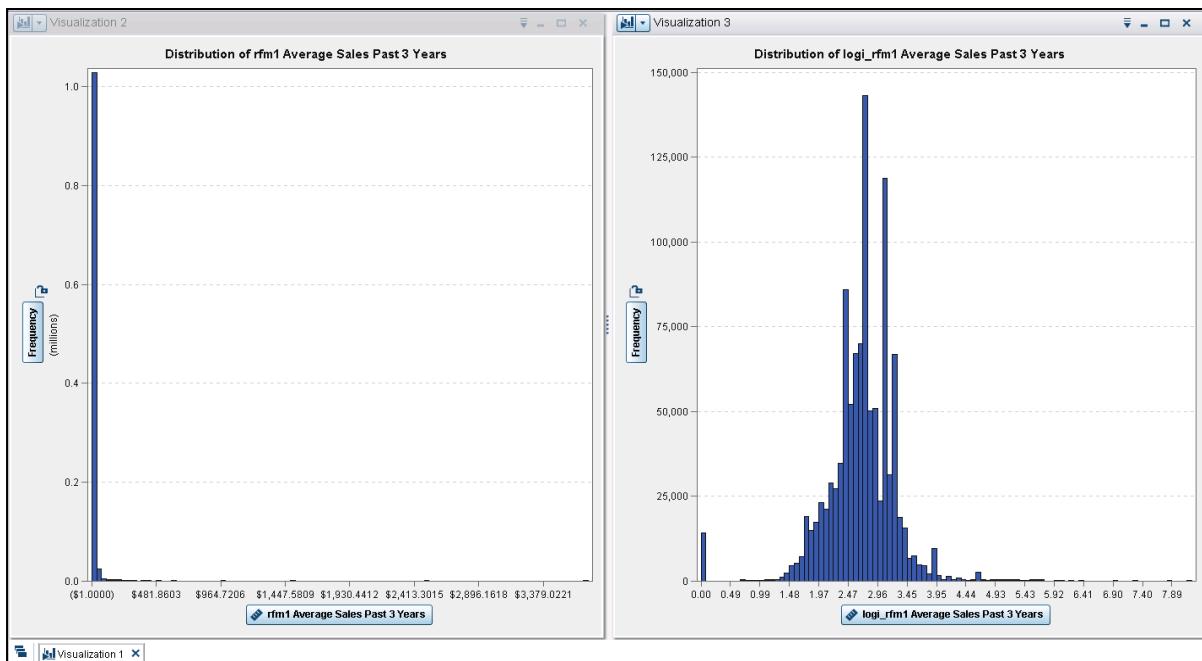
tgt Binary New Product	
	Value
tgt Count Number New Products	
tgt Interval New Sales	
.....	
Property	Value
Name	tgt Binary N
Classification	Measure
Model type	Continuous
Format	Numeric
Aggregation	Sum

Add to Visualization
 Add as Filter on Visualization
 Add as Filter on VS_BANK
 Create >
 New Custom Category...
 Duplicate Data Item...
 Rename...
 Hide
 Category **Category**
 • Measure
 Geography >

13. Drag the **tgt Binary New Product** variable to the workspace. A bar chart is created.

You can see from the chart that approximately 200,000 of the 1,000,000 records represent customers who made a purchase.

14. Minimize the **Frequency of tgt Binary New Product** bar chart.
 15. Select **Visualization** \Rightarrow **New** for a new visualization. Drag **rfm1 Average Sales Past 3 Years** to the workspace.
- The results are skewed to the right. For some models, it is useful to use a regularizing transformation for input variables. The log transformation regularizes this distribution.
16. Select **Visualization** \Rightarrow **New**. Drag **logi_rfm1 Average Sales Past 3 Years** into the new visualization area.
 17. Compare the histograms.

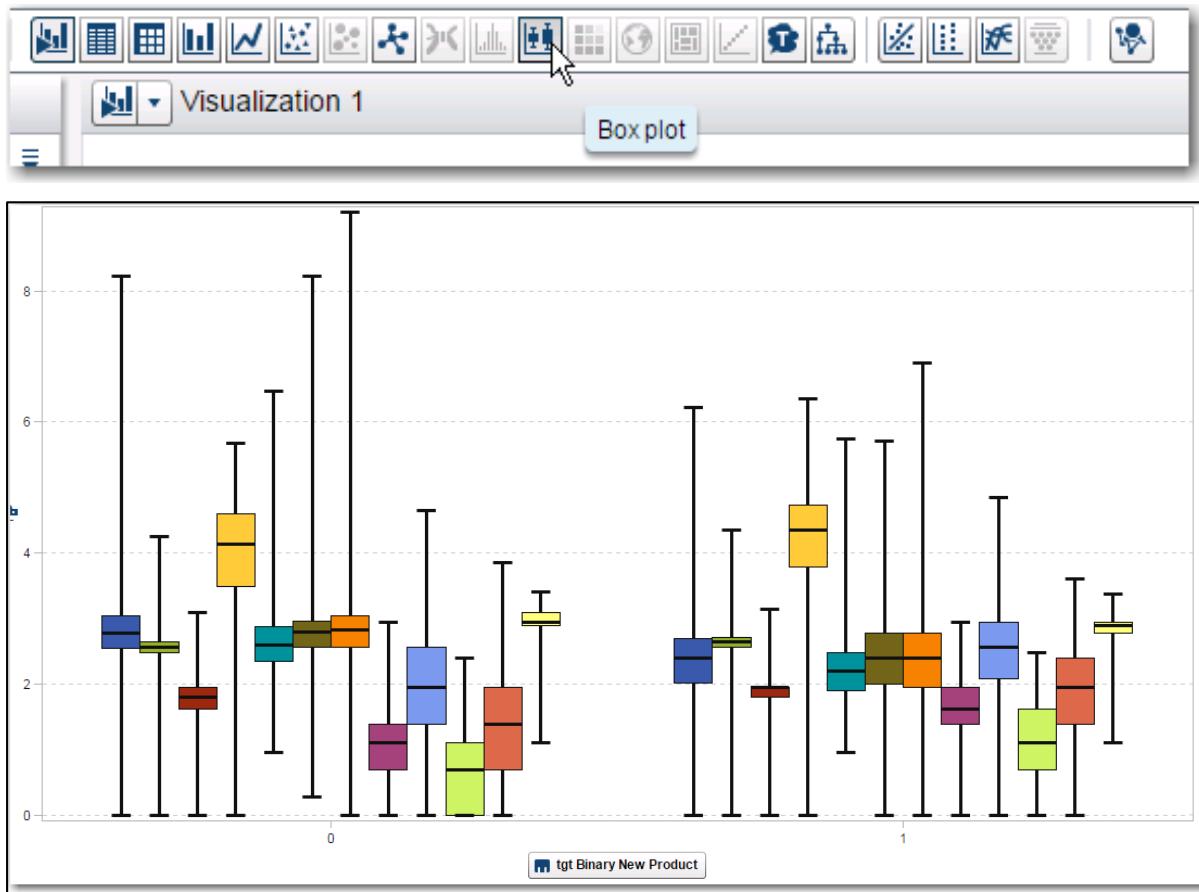


Imputed and log-transformed RFM variables were added to the data for convenience.

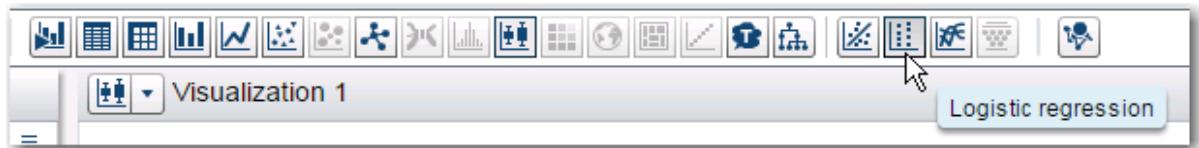
Fitting an Automatic Model

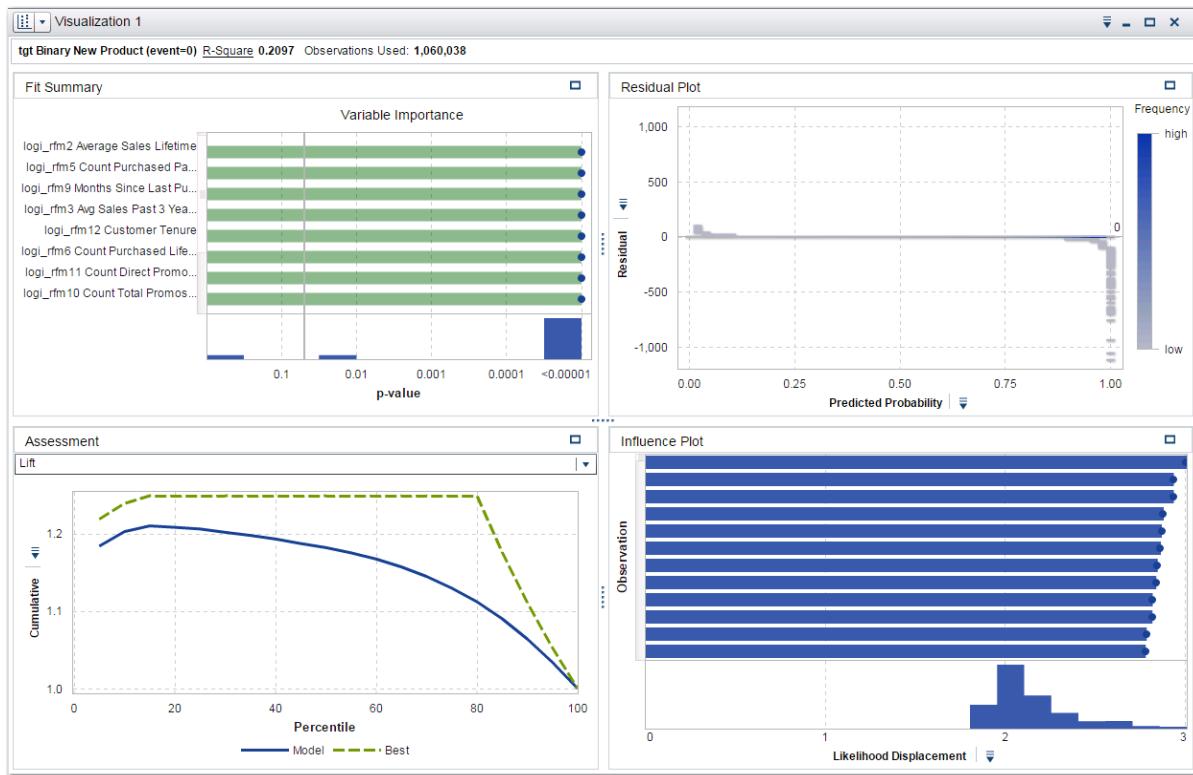
The inputs for modeling are the RFM variables. These variables had missing values and are heavily skewed. Therefore, a transformed and imputed version of each is in the data set, with the prefix **logi_**.

- You can explore the distribution of any input variable by creating a new visualization for it and dragging the variable to the workspace. Spend a few moments exploring the RFM input variables.
1. Minimize the two histograms for the **rfm1** variables, and click **Visualization 1** to maximize the bar chart for **tgt Binary New Product**.
 2. Hold down the Shift key and select all of the **logi_rfm** variables and drag them to the **Measures** role on the **Frequency of tgt Binary New Product** Roles tab.
- A grouped bar chart is created. This might be better viewed as a box plot.
3. Click the **Box plot** icon to change the plot type to a side-by-side box plot.



4. You can explore relationships between the interval-valued measure variables and the binary target by clicking the **Logistic regression** button.





5. Maximize the Fit Summary window to see the Variable Importance plot for the effects in the model. The Variable Importance window enables you to change the criterion of significance for the bar colors.
6. Restore down the Fit Summary window.
7. On the Roles tab, click **Advanced**. Change the event level from **0** to **1**. Click **OK**.
8. Maximize the Assessment window. The cumulative lift shows the fitted model versus a hypothetical model in which every case was correctly classified (an unlikely scenario, but a true upper bound for comparison).
9. Restore down the Assessment window.
10. Select **File** \Rightarrow **Preferences**.
11. Select **File** \Rightarrow **Preferences** again. Select **Invert application colors**. Click **OK**.
12. To reset the default theme, select **File** \Rightarrow **Preferences**. Clear the **Invert application colors** check box. Set the theme to **Default theme (SAS Corporate)**.
13. Click **OK**.
14. Explore general preferences for the analyses. Select **File** \Rightarrow **Preferences**. Select **SAS Visual Analytics Explorer** \Rightarrow **Models**.
The **p-value precision** field changes the number of decimal places to display *p*-values.
15. Save the Exploration as **VS_Exploration**.

End of Demonstration



Exercises

In the exercises, you work with a data set named **PVA97NK**. The data set contains data that represents charitable donations made to an American veterans association. The data represent the results of a mail campaign to solicit donations. The data set contains the following:

- a flag to indicate respondents to the appeal and the dollar amount of their donations (**Target Gift Flag** and **Target Gift Amount**)
- PVA promotion and giving history
- demographic data

In the first exercise, you use SAS Visual Analytics Data Explorer to familiarize yourself with the data.

1. Using SAS Visual Analytics Data Explorer

- a. Sign in to Visual Analytics.
- b. Select **Data Explorer** to open SAS Visual Analytics Explorer.
- c. Select the **PVA97NK** data source.
- d. Select **Data Source Details** from the Data pane drop-down list.
 - 1) How many rows of data does **PVA97NK** contain? _____
 - 2) How many columns of data does **PVA97NK** contain? _____
 - 3) How many category variables does **PVA97NK** contain? _____
 - 4) How many measure variables does **PVA97NK** contain? _____
- e. Select **Measure Details** from the Data pane drop-down list to view the properties of all the measures in the data.
 - 1) Which measure variables have missing values? _____
 - 2) How many missing values are there for **Target Gift Amount**? _____
 - 3) How many distinct demographic clusters are represented in the data? _____
 - 4) How many observations represent those with **Status Category 96NK = F**? _____
- f. Change **Target Gift Flag** from a measure to a category. It is a binary indicator that represents a response to a mailing, where *1* indicates that customers did respond.
Use a bar chart. How many females responded to the campaign? _____
- g. Create a histogram of **Target Gift Amount**.
 - 1) Is the variable skewed? _____
 - 2) If so, in which direction? _____
- h. Save the exploration. Click the **Save** icon on the toolbar and save the exploration in My Folder with the name **Exercise 1**.

End of Exercises

Cluster Analysis in SAS Visual Statistics

Objectives

- View and edit cluster properties.
- Build a cluster analysis in SAS Visual Statistics.

20

Cluster

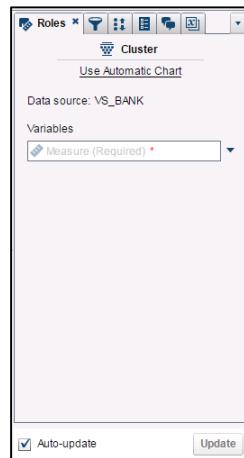
- *Cluster analysis* divides observations into mutually exclusive groups so that observations within a group are as similar as possible, and different groups are as dissimilar as possible.
- Visual Statistics uses the *k*-means method to perform cluster analysis.
- A cluster analysis in Visual Statistics requires two or more measures.
- You can generate cluster IDs, which can be used in other models in Visual Statistics.

21

Clustering is a method of data segmentation that puts observations into groups that are suggested by the data. The observations in each cluster tend to be similar in some measurable way, and observations in different clusters tend to be dissimilar. Observations are assigned to, at most, one cluster. From the clustering analysis, you can generate a cluster ID variable to use in other modeling tasks. You can build stratified models by using the cluster ID as input to another model (linear, GLM, or logistic).

Cluster Roles

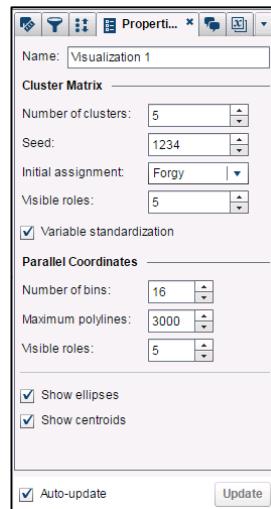
- Variables – at least two or more measures; no categories or interaction terms allowed



22

Cluster Properties

- Name (of model)
- Cluster Matrix
 - Number of clusters
 - Seed
 - Initial assignment
 - Visible roles
 - Variable standardization
- Parallel Coordinates
 - Number of bins
 - Maximum polylines
 - Visible roles
- Show ellipses
- Show centroids



23

Name – enables you to specify the name for this model.

Number of clusters – specifies the number of clusters that are generated. The default is 5, and the maximum is 100.

Seed – specifies the seed value of the random number generator that is used during initial cluster assignments. The default is 1234.

Initial assignment – specifies the method that is used to create the initial cluster assignments. The available methods are as follows:

- **Forgy** – specifies that k data points are selected at random to use as the centroids of the k clusters. This is the default.
- **Random** – assigns observations to a cluster at random.

Visible roles – determines how many effects are shown in the cluster matrix. Valid values are integers between 2 and 6, inclusive. When you specify a value n , the first n effects listed in the Variables table on the Roles tab are displayed. To change the effect pairs that are plotted in the cluster matrix, you can remove an effect from the analysis and then immediately add it back. The clustering results remain unchanged because you used the same input data. However, the Variables table adds new effects to the bottom of the list. The default is 5.

Variable standardization – transforms the effect variables so that they have a mean of zero and a standard deviation of 1. This property is enabled by default and affects the results that are displayed in the summary table. The Cluster Matrix window and the Parallel Coordinates window display the original variables.

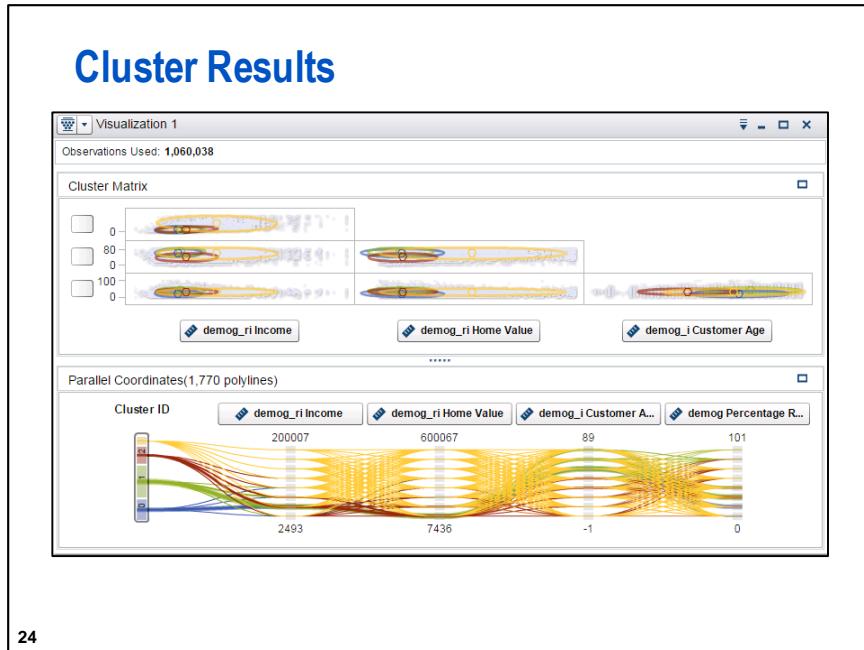
-  Variable standardization improves cluster formation by minimizing the impact of different scales among the involved variables.

Parallel Coordinates

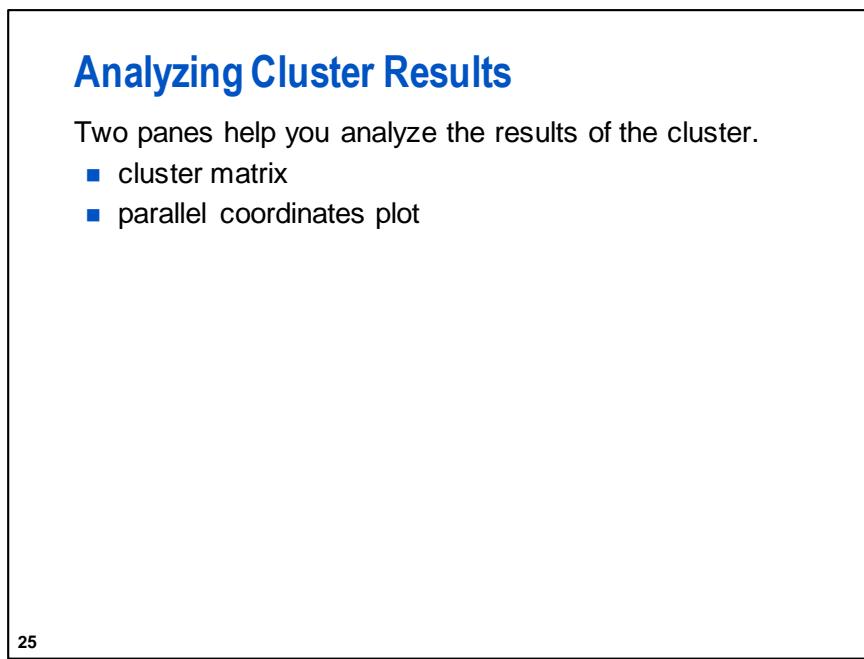
- **Number of bins** – specifies the number of bins that are used when the parallel coordinate polyline plots are generated. The default is 6, and the maximum is also 6.
- **Maximum polylines** – specifies the maximum number of polylines that are generated by the parallel coordinate algorithm. The default is 3000, and the maximum is 5000.
- **Visible roles** – determines how many effects are shown in a parallel coordinates plot. Valid values are integers between 2 and 10, inclusive. When you specify a value n , the first n effects listed in the Variables table on the Roles tab are displayed. To change the effect pairs that are plotted in a parallel coordinates plot, you can remove an effect from the analysis and then immediately add it back. The clustering results remain unchanged because you used the same input data. However, the Variables table adds new effects to the bottom of the list. The default is 5.

Show ellipses – enables you to display the cluster projection ellipses in the cluster matrix.

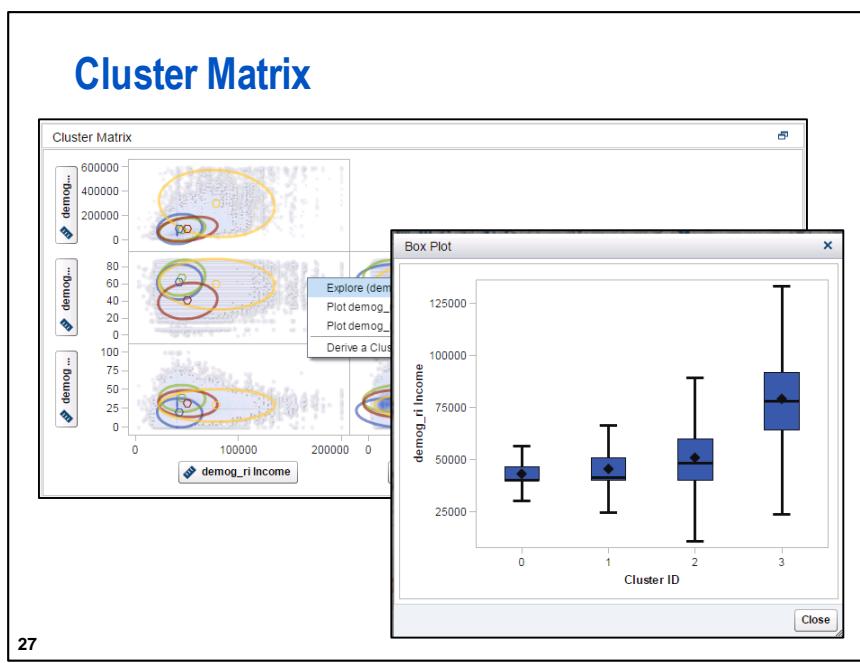
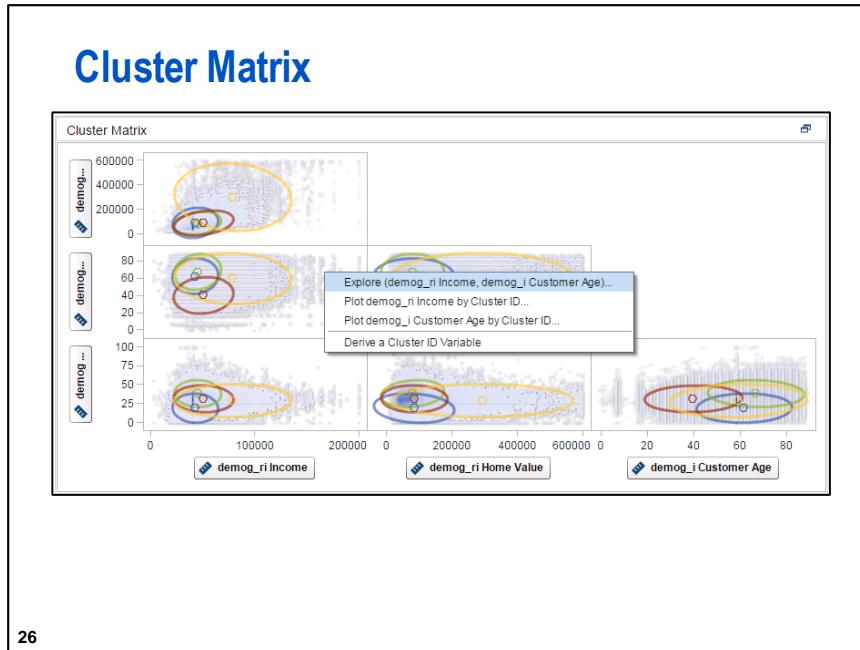
Show centroids – enables you to display the centroids in the cluster matrix.



24



25



The *cluster matrix* displays a two-dimensional projection of each cluster onto a specified number of effect pairs. These projections are useful for spotting cluster similarities and differences within the plotted effect pairs. Maximizing the window can make it easier to discern the clusters.

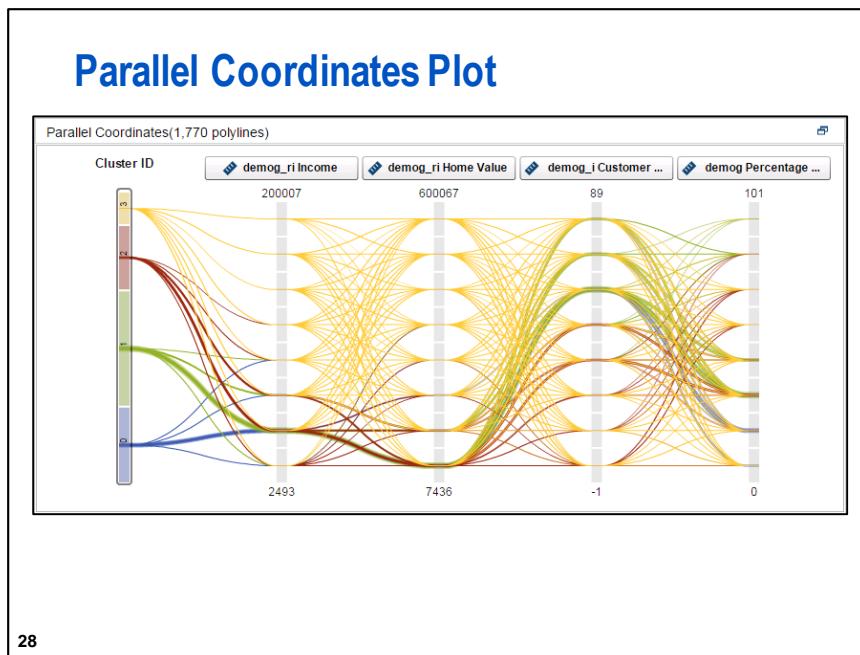
Each cluster is assigned a unique color. Although each cluster is unique in n space, the two-dimensional projections overlap. If the individual plots are shown as scatter plots, the observations are color coded as well. If the individual plots are displayed as heat maps due to large data, only the clusters are color coded, not the observations or bins.

Right-click an individual plot to display a pop-up menu from which you can select the following:

- **Explore** (the plot in a larger view).
- **Plot <the X axis variable> by Cluster ID** (as a box plot) – The box plot is used to determine how similar the clusters are for a variable.
- **Plot <the Y axis variable> by Cluster ID** (as a box plot) – The box plot is used to determine how similar the clusters are for a variable.
- **Derive a Cluster ID variable** – When you select this item, SAS Visual Statistics creates a category variable that contains the cluster ID for each observation. You can use this variable as an effect in other models.

Exploring an individual plot makes it easier to view and select observations. When you select an observation, the clusters that overlap the selected observation are also selected.

It is important to notice that every observation can belong to at most one cluster. However, because the cluster matrix displays a projection in only two dimensions, multiple clusters can overlap an observation.

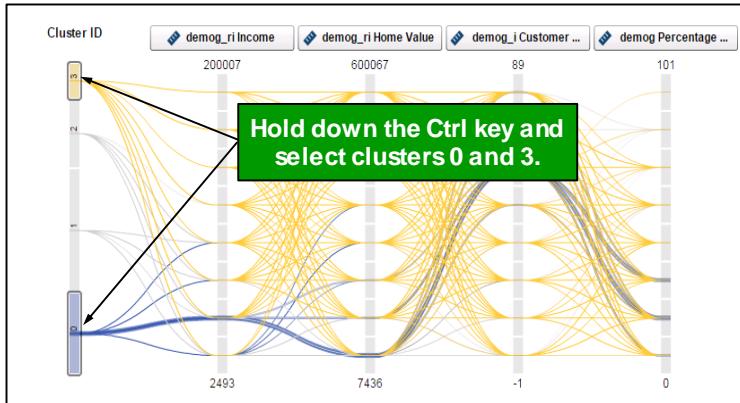


The *parallel coordinates plot* shows patterns in the data and clusters. In this plot, the cluster ID is on the far left, and each variable is a column and its binned range of values are displayed vertically. Color-coded polylines are drawn from each cluster. They show which range of values the cluster contains for every variable that is displayed along the top.

You control the number of variables displayed at the top using the **Visible Roles** property under the **Parallel Coordinates** section on the **Properties** tab.

You can use the parallel coordinates plot to make several inferences about the data. You can adjust the plot to explore the data based on cluster membership, a specified range for one or more variables, or both.

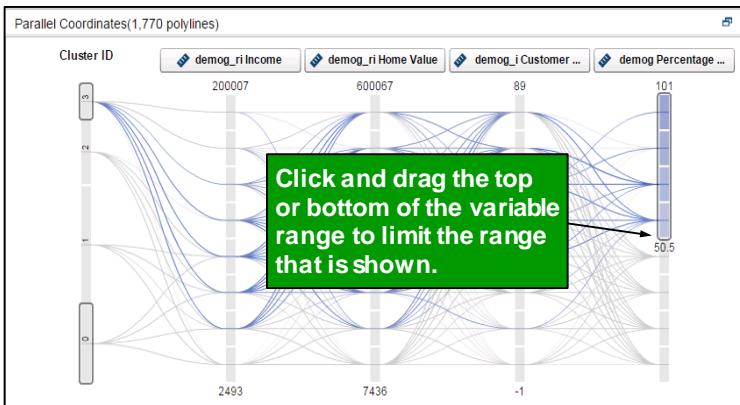
Parallel Coordinates Plot with Selected Clusters



29

When there are several clusters, it can be confusing to determine how each cluster classifies the data. To view the polylines for a single cluster only, select that cluster ID on the far left. Notice that the polylines for all other clusters are dimmed. This enables you to focus on one cluster. Hold down the Ctrl key and click multiple clusters to show only those clusters.

Parallel Coordinates Plot with Selected Variable Range



30

Click a variable name at the top of the plot to select that variable. This action changes the color gradient of the polylines so that larger values are darker than smaller values. You can click and drag from the top or bottom of a variable range to adjust the range of values that is shown. You can repeat this step for multiple variables.

When you combine the selection of clusters and variable ranges, you can restrict the display to the data that interests you.

Cluster Summary Tab

The *Cluster Summary* tab is displayed when you click the **table** button (grid icon), and it provides summary statistics for each cluster.

- Cluster ID
- Observations
- RMS of STD
- Within-Cluster SS
- Min centroid-to-observation
- Max centroid-to-observation
- Nearest Cluster
- Centroid Distance

Cluster Summary							
Cluster ID	Observations	RMS of STD	Within-Cluster SS	Min centroid-to-observation	Max centroid-to-observation	Nearest Cluster	Centroid Distance
0	114	0.583775	154.0384	0.292318	3.095469	2	1.757319
1	39	0.773925	91.04183	0.292317	4.452674	0	2.143922
2	184	0.416088	126.7303	0.217016	1.67934	0	1.757319
3	91	0.549878	108.8518	0.119344	5.665281	2	2.055082

31

-  Column widths were expanded to show the column title in this display.
- **RMS of STD** – the root mean square across variables of the cluster standard deviations, which is equal to the root mean square distance between observations in the cluster.
 - **Within-Cluster SS** – within-cluster sum of squares. *K*-means clustering tries to minimize this.
 - **Min centroid-to-observation** – the minimum distance from the centroid to an observation in the cluster.
 - **Max centroid-to-observation** – the maximum distance from the centroid to an observation in the cluster.
 - **Nearest Cluster** – the number of the cluster that has a mean closest to the mean of the current cluster.
 - **Centroid Distance** – distance between cluster centroids.

2.01 Multiple Choice Poll

In SAS Visual Statistics, the default number of clusters is which of the following?

- a. determined by the CCC statistics
- b. the optimal number of clusters, given the distribution of the data
- c. five, but you can change this to another number
- d. There is no default; the user must specify the number.



Building a Cluster Analysis

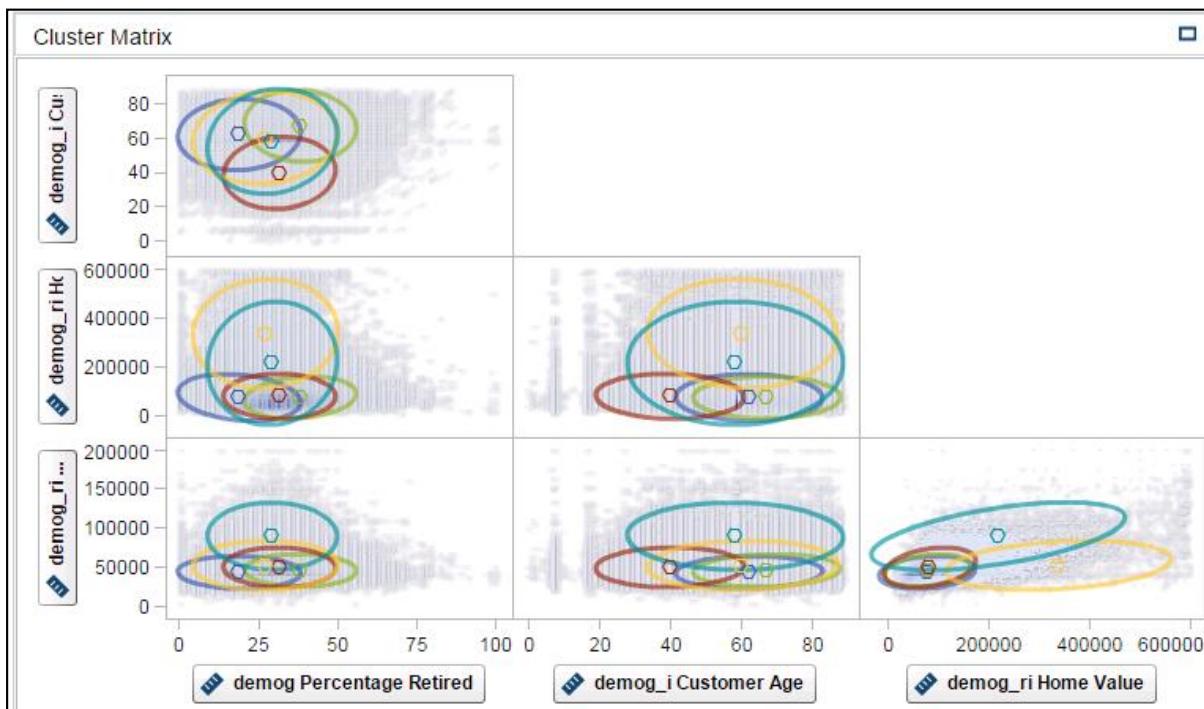
In this demonstration, you extend the data to Visual Statistics and perform a cluster analysis on the demographics variables.

This demonstration illustrates using the **VS_BANK** data set to create segments based on demographic input variables. This is a data set that contains the modifications that you used in the previous chapter.

1. From the SAS Visual Analytics Home Page, select **Data Explorer**. Alternatively, click the link for SAS Visual Analytics in the Google Chrome Favorites.
2. Click **Select a Data Source**.
3. Click **VS_BANK** \Rightarrow **Open**.

In the exploration window, the Data pane shows the three category variables and the 51 measure variables, including the imputed and transformed input variables. Imputed RFM variables have the prefix **i_**, and log-transformed inputs have the prefix **logi_**. Demographic variables were recoded (**_r**) and imputed (**_i**).

4. From the toolbar, click (the **Cluster** button).
5. Select the following demographic variables by holding down the Ctrl key and selecting them in the Data pane: **demog Percentage Retired**, **demog_i Customer Age**, **demog_ri Home Value**, and **demog_ri Income**.
6. Drag the selected variables into the work area.



The results show the Cluster Matrix window and the Parallel Coordinates window. The Parallel Coordinates window does not show a graph. You remedy this soon.

- Click the **Properties** tab in the right pane and investigate the settings.

The default number of clusters is five. This is arbitrary. For the purposes of modeling the banking data, four clusters are convenient.

- Change the Number of clusters property to **4**. The results are updated to show a solution that has four clusters.

The initialization of the cluster centers is determined jointly by the Seed and Initial Assignment settings. By default, *Forgy* is used to select the initial centers, with a random number seed of 1234. You can change the seed to use a different randomization.

 The Forgy method randomly chooses k observations from the data set and uses these as the initial means. The Random Partition method first randomly assigns a cluster to each observation and then proceeds to the Update step. This computes the initial mean to be the centroid of the cluster's randomly assigned points. The Forgy method tends to spread the initial means out, whereas the Random Partition method places all of them close to the center of the data set.

Variable standardization is performed by default.

The parallel coordinates graph is useful for profiling your clusters. You can modify the properties to see the graph by increasing the maximum number of polylines, the number of bins, or both.

The number of visible roles controls how many variables are used for profiling.

The Cluster Matrix window shows prediction ellipses and cluster centroids on each scatter plot.

- Change the Number of bins property from **16** to **8** and press Enter. This enables the parallel coordinates graph to be displayed.

The parallel coordinates graph shows a mosaic plot of the clusters on the left. The mosaic plot tiles are proportional to the number of polylines. The number is a measure of the complexity (variability) of the clusters, in terms of the variables in the analysis.

The parallel coordinates graph also bins each input into k equal-interval bins, where k is the value that is specified in the Number of bins property.

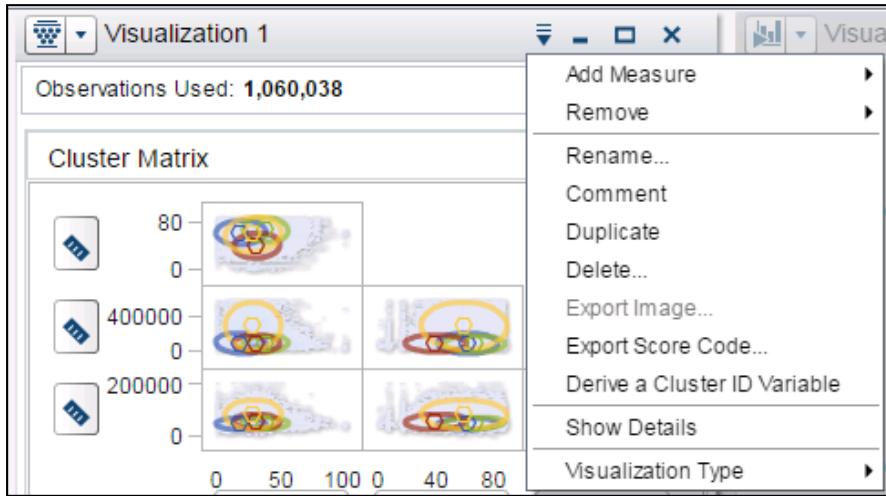
Because these bins are equal interval rather than equal proportion, some bins might have no data.

Each input is labeled with the minimum and maximum data values, and polylines extend from each cluster through the bins on the inputs. The thickness of the polyline is proportional to the number of cases in that polyline.

You can use interactive graph actions to filter clusters, ranges of values, and observations for profiling. **Your instructor can walk you through examples of profiling these clusters.**

- Place the mouse pointer on the title bar, and on the right side, click the icon for **Show Details** to see summary statistics for the clusters. Notice that the clusters are numbered 0 through 3.
- Right-click the cluster matrix and select **Derive a Cluster ID Variable**. Accept the default name and click **OK**.

 You can also save the score code to an external file. This enables you to assign cluster membership to new data. To do so, select the menu arrow on the cluster heading and select **Export Score Code**.

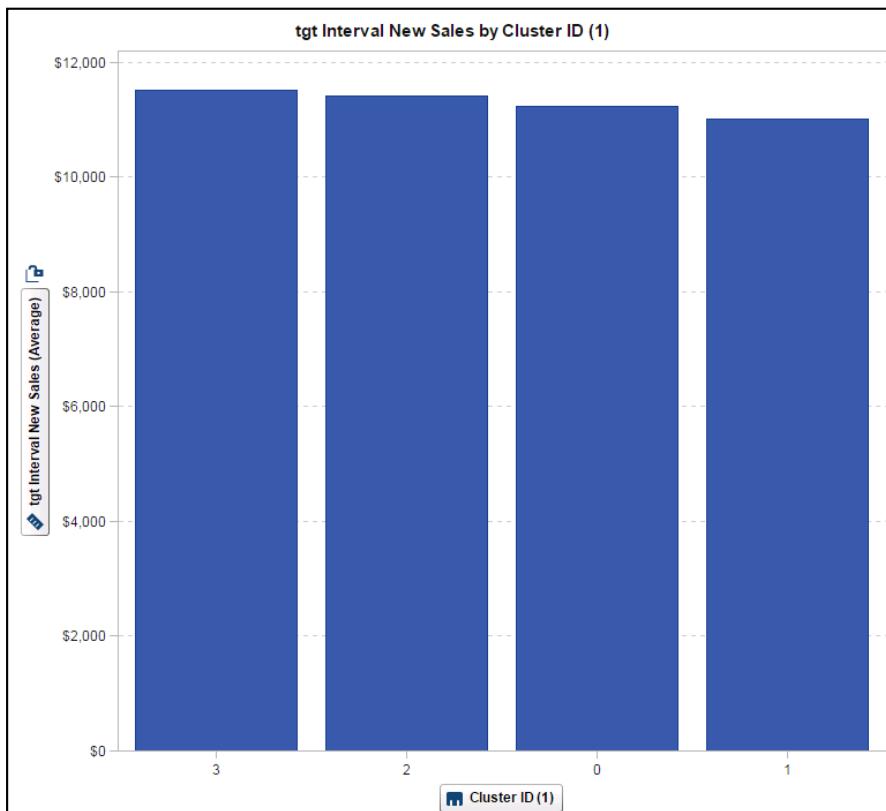


To evaluate the segments further, use the derived Cluster ID variable in a new visualization.

12. Select **Visualization** \Rightarrow **New**. Drag **Cluster ID (1)** onto the workspace. Drag **tgt Interval New Sales** onto the workspace.

The default aggregation is **Sum**. Change this to **Average**.

13. Right-click the vertical axis label for **tgt Interval New Sales (Sum)**. Select **Aggregation** \Rightarrow **Average**.



New sales are highest for cluster 3, and lowest for cluster 1.

14. Save the project as **VS_Bank Cluster** in the My Folder location.

End of Demonstration



Exercises

2. Performing a Cluster Analysis

This exercise uses the **PVA97NK** data set that you explored earlier.

- a. Perform a cluster analysis for the **Gift...** variables. Experiment with three-, four-, and five-cluster solutions. Which cluster gave the most money per donation? Which cluster donated most frequently?
- b. Save the analysis as **Exercise 2**.

End of Exercises

Segmentation Ideas and SAS Visual Statistics Cluster Details (Self-Study)

Objectives

- Illustrate segmentation concepts in the context of big data.
- Describe the clustering method implemented in SAS Visual Statistics.

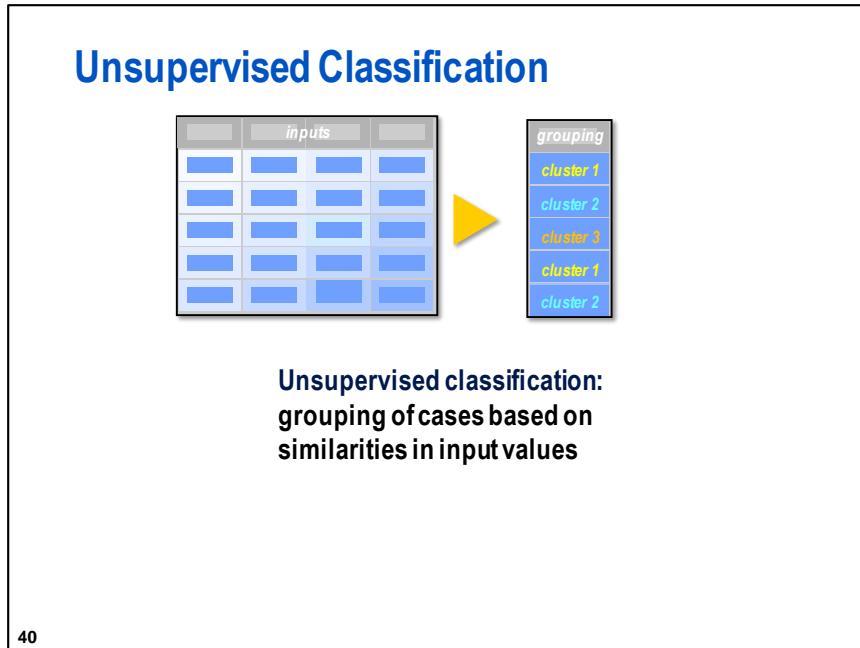
38

Segmentation Concepts

Why Segmentation?

There are many reasons for performing segmentation. In the class scenario, you cluster the customers on demographic variables to create segments, which can be used for stratified modeling later.

39



Cluster analysis is a form of unsupervised classification that attempts to group cases in the data based on similarities in **input** variables. It is a data-reduction method because an entire training data set can be represented by a small number of clusters. The groupings are known as *clusters* or *segments*, and they can be applied to other data sets to classify new cases. Unsupervised classification is distinguished from *supervised classification* (in which there is a known criterion).

The purpose of clustering is often description. For example, segmenting existing customers into groups and associating a distinct profile with each group might help future marketing strategies. However, there is no guarantee that the resulting clusters are meaningful or useful.

Clustering can be useful as a preliminary step in predictive modeling. For example, customers can be clustered into homogeneous groups based on sales of different items. Then, the clusters can be used as stratification variables for building predictive models separately on each group.

Segmentation for Customer Types

You want to identify segments. You have many customers, but there are really only a handful of major types into which most of your customers can be grouped.

- bargain hunter
- man or woman on a mission
- impulse shopper
- weary parent
- DINK (dual income, no kids)



40

Segmentation for Store Location

You want to open new grocery stores in the U.S. based on demographics. Where should you locate the following types of new stores?

- low-end budget grocery stores
- small boutique grocery stores
- large full-service supermarkets



41

Classifying Fashion Trends

Based on the four styles of pants that your customers can purchase, can you identify stores as serving similar fashion types?

- country-club dresser
- fashion trendsetter
- comfort kick-back dresser

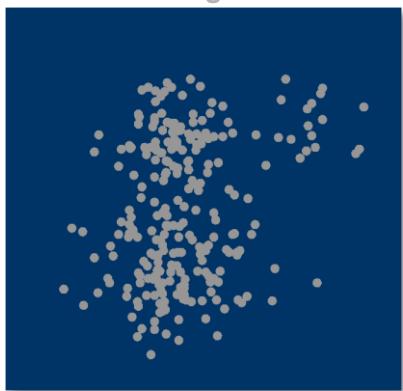


42

SAS Visual Statistics: Clustering Method Details

k-Means Clustering Algorithm

Training Data



1. Select inputs.
2. Select k cluster centers.
3. Assign cases to closest center.
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

...

43

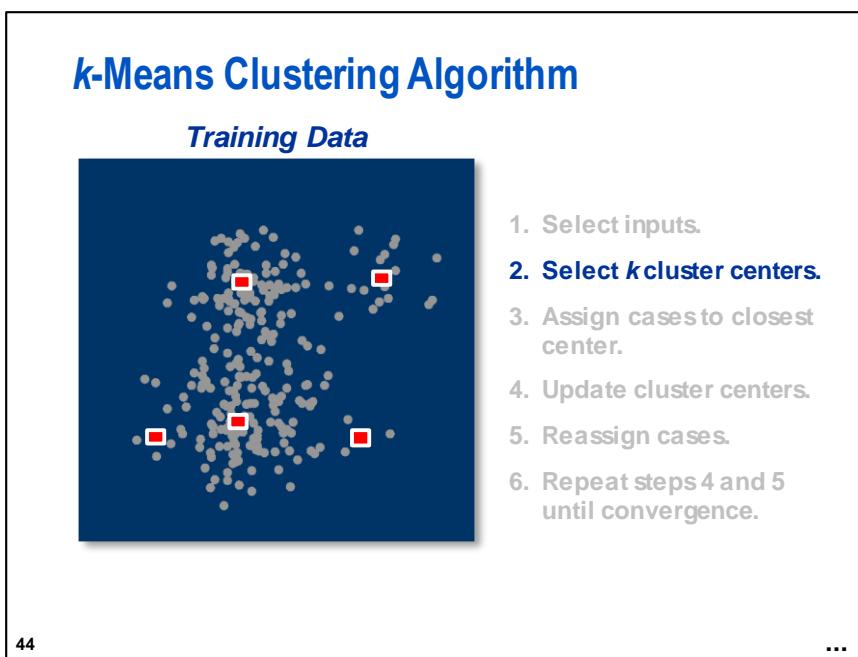
One of the most commonly used methods for clustering is the *k-means algorithm*. It is a straightforward algorithm that scales well to large data sets.

Although it is often overlooked as an important part of a clustering process, the first step in using the *k-means* algorithm is to choose a set of inputs. In general, you should seek inputs that have these attributes:

- are meaningful to the analysis objective
- are relatively independent
- are limited in number

- have a measurement level of *Interval*
 - have low kurtosis and skewness (at least in the training data)
-  Skewness is a measure of the symmetry of a distribution. Skewness far from 0 indicates asymmetric data. Kurtosis is a measure of the density of a distribution in the peak, flanks, and tails. A symmetric distribution with kurtosis near 0 (or 3 in some software) has the characteristic bell-shaped curve of the normal distribution. Kurtosis can easily be interpreted if there is no symmetry. See DeCarlo (1997)¹ for more information about the impact and meaning of kurtosis in data analysis.

Choosing meaningful inputs is clearly important for the interpretation and explanation of the generated clusters. Independence and limited input count make the resulting clusters more stable. An interval measurement level is recommended for k -means to produce nontrivial clusters. Low kurtosis and skewness statistics on the inputs avoid creating single-case outlier clusters.

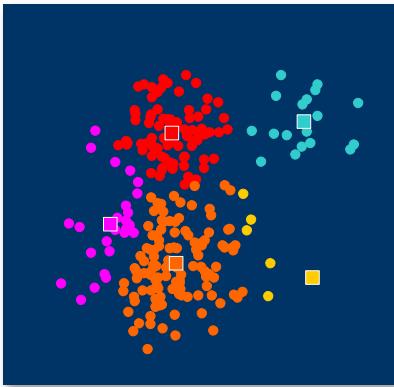


The next step in the k -means algorithm is to choose a value for k , the number of cluster centers. SAS Visual Statistics uses five clusters by default, although you can change this easily. You should choose k to be consistent with the natural concentrations of cases, or with your analytic objectives. For example, if you are interested in promoting three offerings, then three (or more) clusters would be most useful.

¹ DeCarlo, L. T. 1997. "On the meaning and use of kurtosis." *Psychological Methods* 2:292-307.

k-Means Clustering Algorithm

Training Data



1. Select inputs.
2. Select k cluster centers.
- 3. Assign cases to closest center.**
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

45

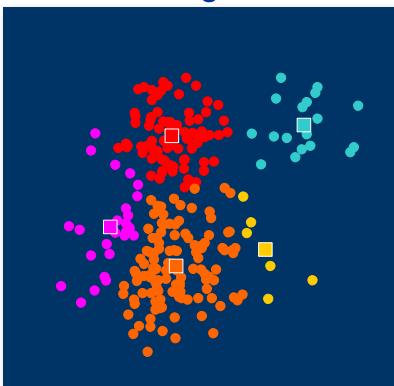
...

The Euclidean distance from each case in the training data to each cluster center is calculated. Cases are assigned to the closest cluster center.

- ✍ The *Euclidean distance* is the distance between two points measured as a straight line. It is computed based on the Pythagorean Theorem and does not directly account for differences in scale of measurement. Because the distance metric is Euclidean, it is important for the inputs to have compatible measurement scales. Unexpected results can occur if one input's measurement scale differs greatly from the others.

k-Means Clustering Algorithm

Training Data



1. Select inputs.
2. Select k cluster centers.
3. Assign cases to closest center.
- 4. Update cluster centers.**
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

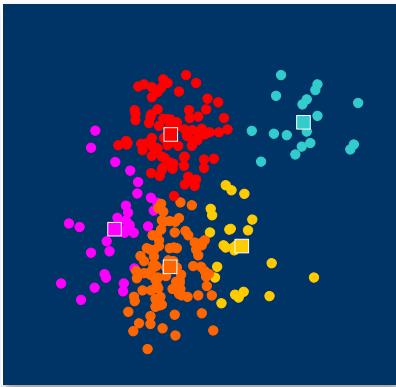
46

...

The cluster centers are updated to equal the average of the cases assigned to the cluster in the previous step.

k-Means Clustering Algorithm

Training Data



1. Select inputs.
2. Select k cluster centers.
3. Assign cases to closest center.
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

49

...

Cases are reassigned to the closest cluster center.

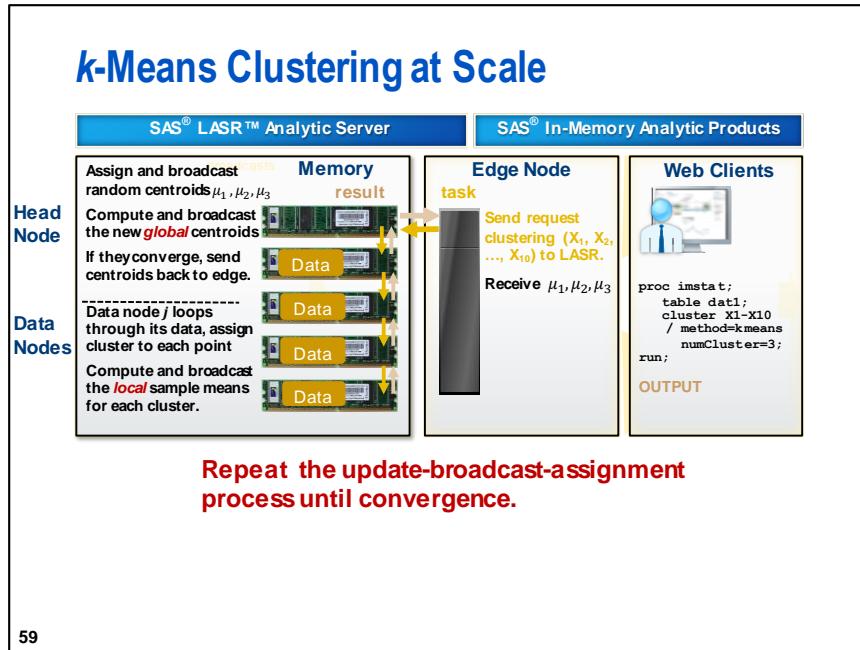
The update and reassign steps are repeated until the process converges. On convergence, final cluster assignments are made. Each case is assigned to a unique segment. The segment definitions can be stored and applied to new cases outside of the data used to develop clusters.

How to Select a Value of k

- subject-matter knowledge
- statistical techniques (CCC, PSF, PST2, Ward's, and so on)
- convenience
- arbitrary
- profiling trial and error

58

There are many techniques, from the scientifically rigorous to the arbitrary, for determining the number of clusters, or k . In SAS Visual Statistics, you can specify the number of clusters, profile that selection, and try other candidate solutions. The default initial number of clusters is 5.



If you use SAS Visual Statistics to perform clustering in a distributed environment, then the k -means algorithm is modified somewhat to take advantage of multiple compute and data nodes. For example, suppose you are running in a distributed environment with data in Hadoop.

1. Submit code that asks for k -means clustering of 10 variables on a data set named **dat1**. The number of clusters is 3.
2. The Edge node sends the request to the server and waits for the results.
3. The Head node randomly initializes three centroids and broadcasts them to the data nodes. In this case, the data structure being broadcast is actually a 3-by-10 array.
4. Each of the data nodes goes through its own local data, and assigns each observation to the closest centroid.
5. The local cluster sample means are then calculated on each data node, and this information is sent back to the Head node. Again, the data structure being broadcast is only the 3-by-10 array.
6. The cluster sample means for the entire data, referred to as **global centroids** here, are aggregated to the Head node.
7. Then, the new clusters are broadcast back to the data nodes for the next round of cluster assignment.

Repeat this update-broadcast-assignment process until convergence. After the algorithm converges, the results are sent back to the Edge node and then to the web client for final output.

- What is described here is one of many possible *reduce* strategies that can be implemented with the SAS LASR Analytic Server. There are others that are more efficient, although they are perhaps less convenient, for teaching purposes, than returning the results to the Head node repeatedly. The reduce method used by the LASR Analytic Server depends on the specific problem that the LASR Analytic Server is solving.

2.2 Data Exploration and Dimension Reduction (SAS Enterprise Miner)

Objectives

- Motivate the need for High-Performance Analytics.
- Examine the SAS approach to high-performance computing.

61

Dramatic Performance Improvement: Airlines Industry Example

- Analytic Goal: Predicting whether flights are delayed
- The Data:
 - arrival and departure details for all U.S. commercial flights, October 1987 to April 2008
 - more than 120 million records
- The Analysis: Logistic Regression
 - The traditional Regression node required more than seven hours to fit the model.
 - The HP Regression node fits the model in less than a minute!

62

Performance Challenges

- Very computationally intensive model fitting or simulations can cause slow execution.
- Massive amounts of data are stored in databases. Extraction into traditional computing environments requires massive data movement, which results in very poor performance.
- Many data mining algorithms require multiple passes of the data for training models. This exacerbates the performance problem.

63

Performance issues can arise for more than one reason.

In some situations, the amount of data is not large, but many calculations are required, such as simulations that are typically done for Value-at-Risk (VaR) calculations or stress-testing credit-scoring models.

Other situations might involve very large data sets with millions of rows. Neural networks, for example, require multiple passes of the data to estimate the weights for the model. Passing a very large data set multiple times can lead to severe performance issues.

Some Terminology

Appliance	Is a grid of computers that runs database software.
Access Engine	Enables communication between the SAS server and the appliance. It is the mechanism by which data is read from and written to databases.
Threaded Kernel (TK) Extensions	Are the mechanisms by which distributed computing is done.
HPDM nodes	Are high-performance data mining nodes; the High-Performance nodes in Enterprise Miner.

64

A computing *appliance* is a dedicated hardware and software environment that provides computing resources in a client or server model.

An *access engine* is a network connection between the client machine and the computing appliance. They can be ***multithreaded*** to improve performance.

- ✍ It is possible to have an access engine without an appliance. For example, you can issue a LIBNAME statement to a Teradata library without having an appliance. The engine enables SAS to access files in that particular format.

Threaded kernel (TK) extensions are a mechanism by which distributed computing is done. They are necessary to perform massively parallel processing, which is defined shortly.

HPDM is an abbreviation for High-Performance Data Mining. HPDM is a tab that appears in the SEMMA tools palette in Enterprise Miner. **HPDM nodes** refer to the High-Performance Data Mining nodes that are contained on the HPDM tab.

When you use Enterprise Miner HPDM nodes, data and complex analytical tasks are divided among processing nodes to perform intermediate calculations and return the results to a controller, which assembles the results into a single, final result. For example, to compute variance, the controlling node can collect the sums of squares (SS) from all the nodes. Each of the nodes has a subset of the data. The controlling node combines the nodes, using the sum of the number of rows from each node. Similarly, to compute a mean, the controlling node adds the sums of data values from each node and the sample size from each node, and takes the ratio. In regression, the controlling node can collect sums of squares and cross-products (SSCP) from all nodes and add them together.

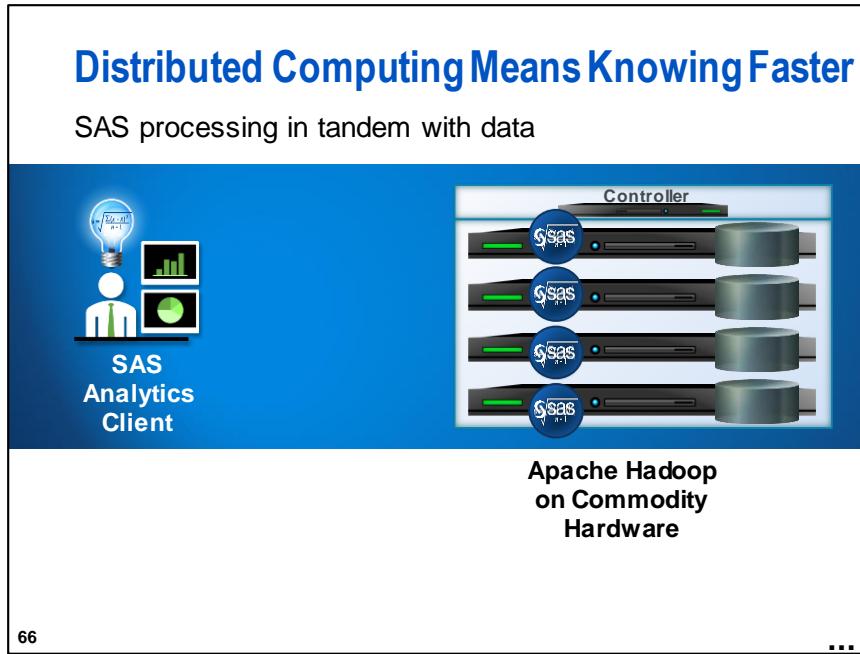
Distributed Computing and Parallel Processing

The High-Performance Data Mining (HPDM) nodes use SAS High-Performance Analytics procedures that have the following characteristics:

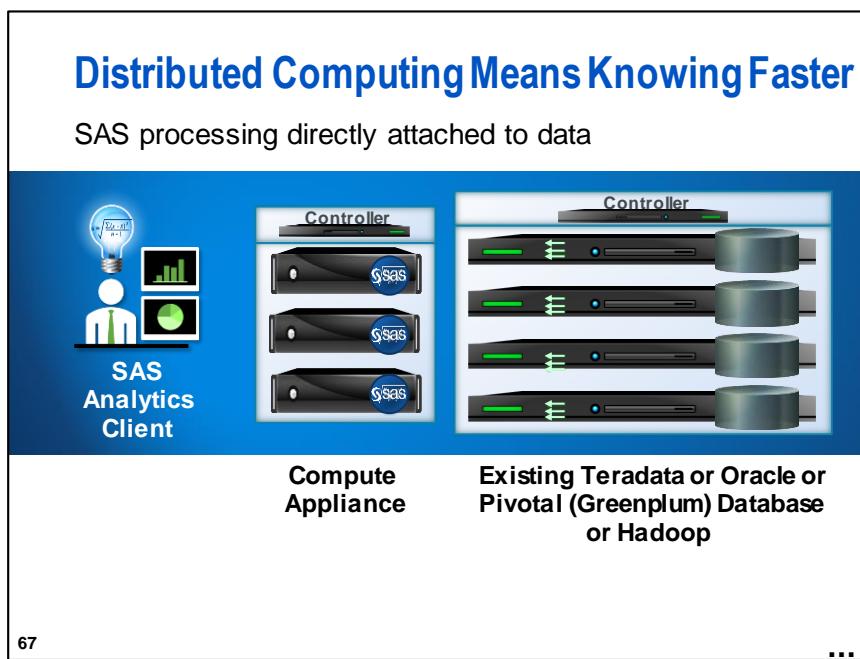
- are written using threaded kernel (TK) extensions
- are designed to run in a distributed environment
- are multi-threaded, to take advantage of multiple cores in single-machine mode
- use algorithm choices that consider data movement and replication
- use in-memory analytics to minimize data movement, which substantially reduces the required computing time, especially for algorithms that require multiple passes

65

SAS High-Performance Analytics procedures run behind the scenes when Enterprise Miner HPDM nodes are used. (Additional training, documentation, and videos about SAS High-Performance Analytics procedures are available on support.sas.com.)



When you use SAS High-Performance Analytics, the data are distributed across a collection of compute nodes, or processors, and instructions for computations are executed on each node simultaneously. This can occur with the SAS computing occurring in-tandem with the data, as shown above.



Distributed processing can also occur when SAS executes instructions on an appliance that is separate from the one on which the data reside, as shown above.

SAS Solution to the Performance Challenge

- Single-machine mode
 - Also called *symmetric multiprocessing* (SMP), this mode uses multiple cores on a single computer.
- Distributed mode
 - Also known as *massively parallel processing* (MPP), this mode can exploit more than one computer.
 - Additional licensing and resources are needed for distributed mode computing.

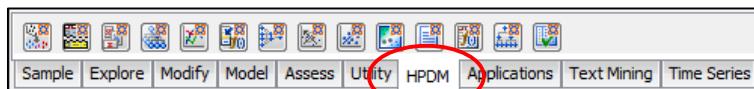
68

Single-machine mode (SMP) enables SAS to use all of the processors and cores that are on a given computer or server. More simply, single-machine mode for High-Performance Analytics applications means multithreading on the client machine.

Distributed mode (MPP) is commonly referred to as *grid* computing. It can exploit more than one computer or node to increase the number of processors and cores that are available for the calculations.

 This course uses Enterprise Miner in single-machine mode.

High-Performance Data Mining Nodes



- | | |
|--|---|
| <ul style="list-style-type: none"> ■ HP Cluster ■ HP Data Partition ■ HP Explore ■ HP Forest ■ HP GLM ■ HP Impute ■ HP Neural | <ul style="list-style-type: none"> ■ HP Principal Components ■ HP Regression ■ HP SVM ■ HP Text Miner ■ HP Transform ■ HP Tree ■ HP Variable Selection |
|--|---|

69

The HP nodes are found on the HPDM (High-Performance Data Mining) tab in the SEMMA tools palette. The nodes and a brief summary are presented below.

- **HP Cluster** performs k-means clustering analysis in distributed computing environments.
- **HP Data Partition** generates an identifier variable that identifies the observations to be used for training and for validation.
- **HP Explore** computes summary statistics in the HP environment.
- **HP Forest** creates random forest models in the high-performance environment.
- **HP GLM** fits a generalized linear model in a distributed computing environment.
- **HP Impute** imputes missing values using high-performance procedures.
- **HP Neural** creates neural network models in the high-performance environment.
- **HP Principal Components** performs principal component analysis.
- **HP Regression** uses high performance linear and logistic regression models.
- **HP SVM** uses the support vector machine procedure to construct separating hyper-planes that maximize the margin between two classes.
- **HP Text Miner** performs text mining in the high-performance environment.
- **HP Transform** creates transformation variables using high-performance procedures.
- **HP Tree** creates tree models in the high-performance environment.
- **HP Variable Selection** performs supervised and unsupervised variable selection using high-performance procedures.

General Comments about HPDM Nodes

- HPDM nodes are not meant to be a replacement for their traditional counterpart nodes.
- The functionality of HPDM nodes can grow over time.

2.02 Multiple Choice Poll

Which of the following is true regarding the HPDM nodes in Enterprise Miner?

- a. HPDM node functionality is not available for data sources with less than one million rows.
- b. Data exploration and predictive modeling can be performed in Enterprise Miner using only HPDM nodes.
- c. Properties and functionality associated with HPDM nodes probably never change.

71

Principal Components Analysis

Objectives

- Describe principal components analysis.
- Explain how to use high-performance principal components analysis in SAS Enterprise Miner.
- Discuss methods for selecting an optimal number of principal components.

74

Principal Component Analysis

- Target used or not?
 - not used
- Original or constructed variables as output?
 - constructed variables

75

Principal Component Analysis: Main Features

- Principal components are constructed as mathematical transformations of the input variables.
- The first principal component is constructed in such a way that it captures as much of the variation in the input variables (the X-space) set as possible.
- The second principal component is orthogonal to the first principal component.
- The second principal component captures as much as possible of the input data's variation that is not captured by the first principal component.
- and so on ...

76

Analysts deploy a variety of techniques to correct for (or perhaps more accurately, take advantage of) the distribution flattening. One of the most common statistical approaches is that of *principal component analysis* (PCA). PCA attempts to find a series of orthogonal vectors that better describe the directions of variation in the data than the original inputs do. (A geometric interpretation of orthogonal is that the vectors are perpendicular; a statistical interpretation is that the vectors are uncorrelated.) The goal is to be able to characterize most of the variation in the data with as few vectors as possible.

Details:

PCA starts by searching the data's standardized joint distribution for the direction of maximum variation. When found, this direction is labeled the *first principal component* or *first eigenvector*.

The effect of the first principal component can be removed by projecting the data to a lower dimensional subspace perpendicular to the first principal component.

The difference between the dimension of the original distribution (that is, the number of inputs) and the effective dimension of the projected points is called the *first eigenvalue*.

The data, projected to remove variation in the direction of the first principal component, are again searched for the direction of maximum variation. When identified, this direction is labeled the *second principal component*. The corresponding second eigenvalue can be calculated by again projecting (the data already projected in the first step) along the direction of the second principal component and determining the difference in dimension between the once and twice projected data.

The process of identifying directions of variability, projecting, and calculating eigenvalues continues until the sum of the eigenvalues calculated at each step is close to the dimension of the original input space. This is a common stopping rule: **How many unique components exist in the data?** There are as many components as it takes to ensure that the sum of the eigenvalues is greater than 80% or 90% of the input count, for example.

In the presence of redundant inputs, most of the data variability can be described by a few independent principal component vectors. Because it can be difficult to interpret the resulting components, and that makes it difficult to interpret the resulting models, in some cases it might be necessary to replace these techniques with a similar tactic that yields more interpretable inputs. This is addressed in the next section.

Input and Output Variables

- Input variables:

$$x_1, x_2, x_3$$

- Principal component 1:

$$pc_1 = a_1x_1 + b_1x_2 + c_1x_3$$

- Principal component 2:

$$pc_2 = a_2x_1 + b_2x_2 + c_2x_3$$

- Principal component 3:

$$pc_3 = a_3x_1 + b_3x_2 + c_3x_3$$

Background

- With k input variables, you can compute k principal components.
- Each of the principal components is an uncorrelated, linear combination of all original input variables.
- The coefficients of such a linear combination are the eigenvectors of the correlation or covariance matrix.
- The principal components are sorted by descending order of the eigenvalues.
- The eigenvalues represent the variances of the principal components.

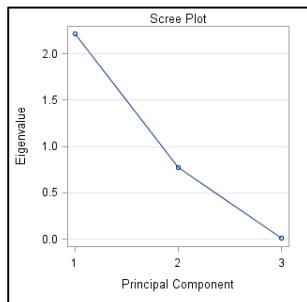
78

Although the eigenvalues and eigenvectors can be constructed from the correlation, covariance, or uncorrected covariance matrix, this course recommends using the correlation matrix. This is the default in the Principal Components node.

Selection of the Number of Principal Components

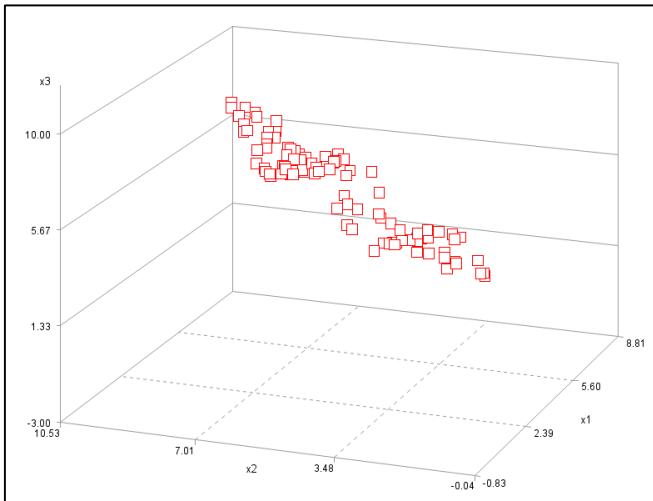
The number of principal components used as input variables for the successor modeling nodes can be selected using one of the following:

- Proportion of variance explained
- Scree plot
- Eigenvalue > 1



79

Example 1



80

The original dimension of the data is 3. That is, there are three input variables. The dimension after projection appears to be much smaller. Visually, the dimension after projection seems to be approximately 1.

Example 1

Eigenvalues of the Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	2.21358154	1.44403082	0.7379	0.7379
2	0.76955072	0.75268297	0.2565	0.9944
3	0.01686775		0.0056	1.0000

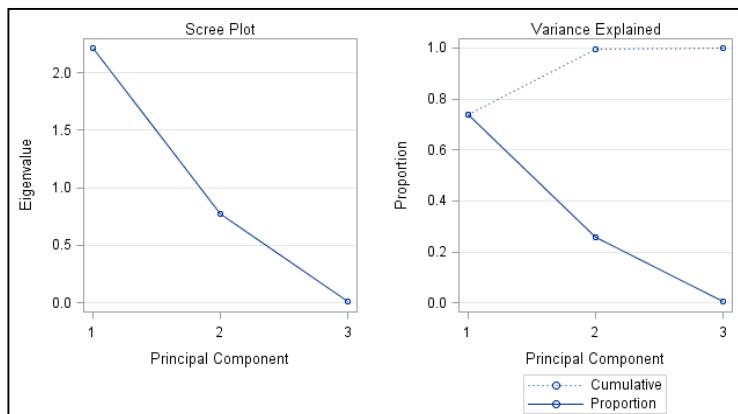
Eigenvectors			
	Prin1	Prin2	Prin3
x1	0.650940	0.263685	0.711862
x2	0.645235	0.301851	-0.701825
x3	-0.399937	0.916164	0.026348

81

Because the initial dimension of the data is 3 and the dimension after projection appears to be approximately 1, the first eigenvalue is the difference of these dimensions, which is approximately 2 (precisely 2.213 from the table). Intuitively, the ratio of the first eigenvalue to the original number of inputs can be thought of as the proportion of total variation in the data in the direction of the first principal component. In this example, the first principal component explains more than 2/3 of the overall variation in the data (precisely 0.7379 from the table).

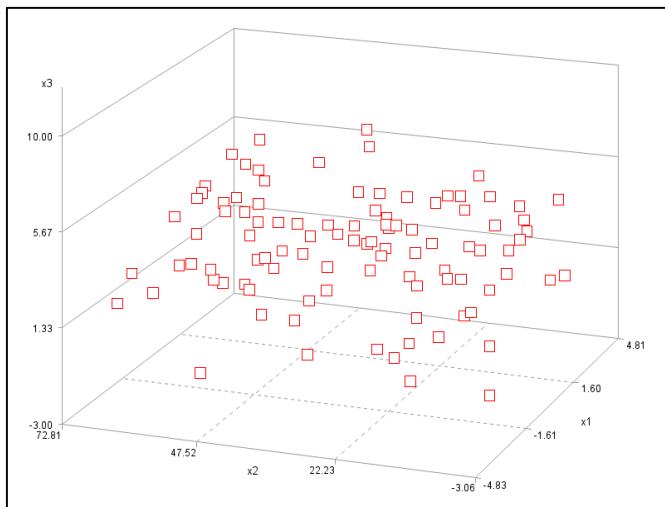
The once-projected data have a dimension approximately equal to 1, whereas the twice-projected data are essentially a constant and thus have a dimension approximately equal to 0. The difference in these dimensions is approximately 1, and this is the approximate value of the second eigenvalue (precisely 0.77 from the table). The second eigenvalue therefore explains approximately one third of the overall variation in the data (precisely 0.2565 from the table).

Example 1



82

Example 2



83

Again, the dimension of the data is 3. The dimension after projection is larger than for the previous example, perhaps approximately 2.

Example 2

Eigenvalues of the Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	1.24205697	0.23274599	0.4140	0.4140
2	1.00931098	0.26067894	0.3364	0.7505
3	0.74863205		0.2495	1.0000

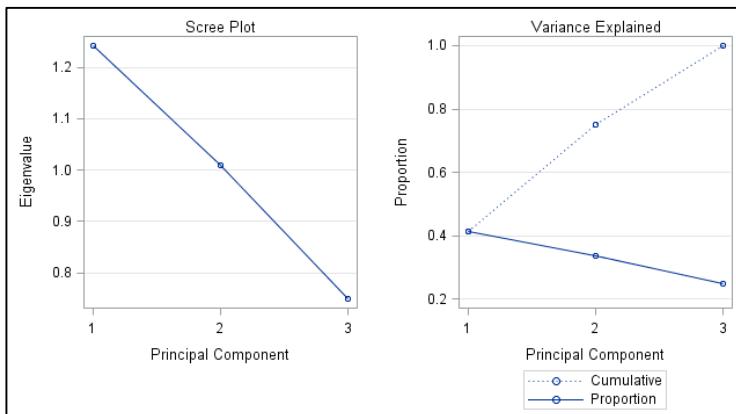
Eigenvectors			
	Prin1	Prin2	Prin3
x1	0.704619	-.156546	0.692102
x2	0.708942	0.113759	-.696032
x3	0.030228	0.981097	0.191139

84

The first eigenvalue is the difference in dimensions discussed under the previous slide, which is approximately 1 (precisely 1.242 from the table). The first eigenvalue accounts for less than half of the total original variation (precisely 0.414 from the table).

The second eigenvalue accounts for nearly one third of the total variation in the original data, and even the third eigenvalue still accounts for about one fourth of the total variation.

Example 2



85



Discussion

Do you have any experience with principal component analysis? Did you ever use PCA in an analysis?
What do you think are the benefits?
What are the limitations?

86

Principal Component Analysis: Pros

- Constructed output variables are definitely uncorrelated.
- The selection order of the principal components is automatically determined.
- The principal components are constructed in such a way that the first principal component represents more of the variation in the data cloud than the second one, and so on.
- Often, a very small number of principal components must be retained so that a lot of the variation in the data cloud can be explained.

87

Categorical inputs can be used, but a set of dummy variables is created for each level of the categorical variable.

Principal Component Analysis: Cons

- It is difficult or impossible to interpret the constructed principal components.
- It is difficult to know how many principal components should be selected as new input variables.
- All original input variables are still used because they build the principal components.
- Misinterpretation of the coefficients of the linear combinations is common.

88

High-cardinality categorical inputs require many dummy variables.



Principal Components Analysis

The **VS_BANK** data, described earlier in the chapter, is used in this demonstration. This demonstration illustrates how a principal component analysis can be used for dimension exploration of the input variable space. That is, the analysis is used to investigate collinearity issues among the group of candidate input variables. Property settings are discussed, and the results of the node are interpreted. Opening a SAS project, defining a data source, and creating a diagram are briefly discussed.

Opening an Analysis Project

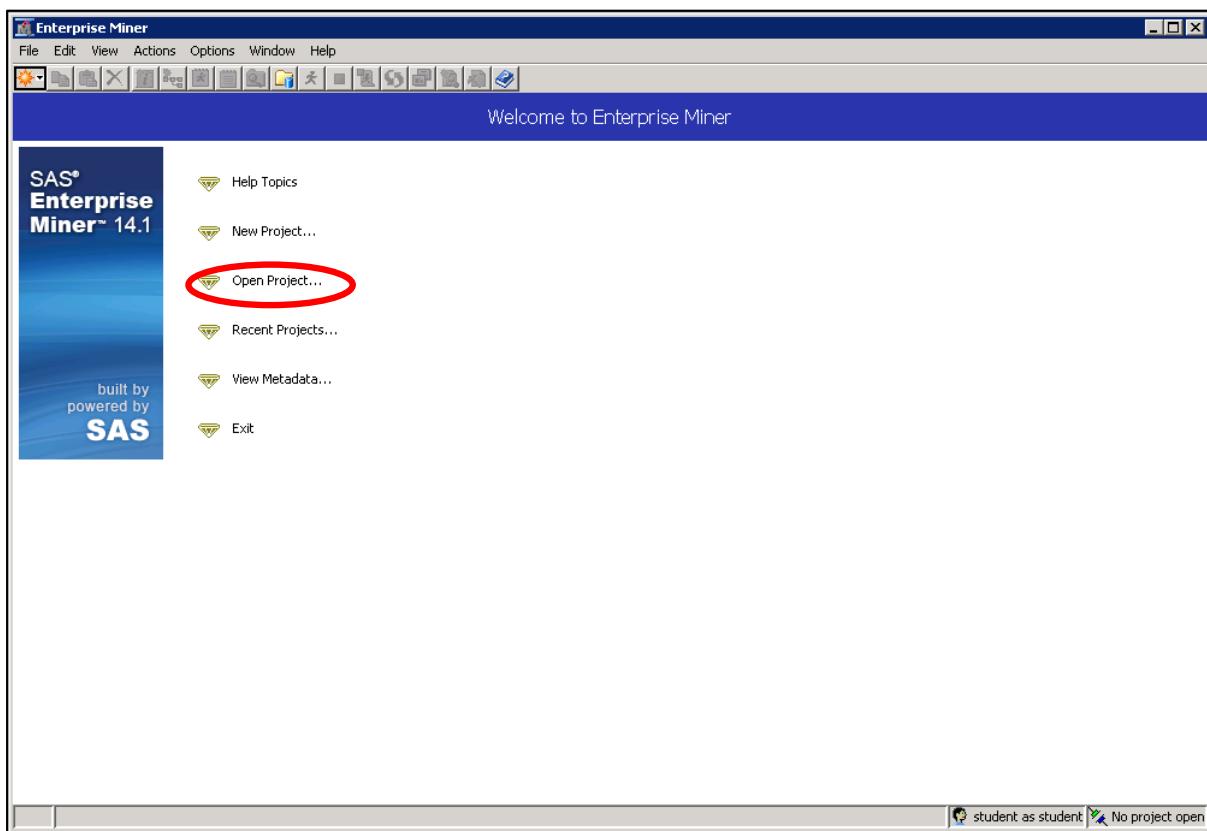
A SAS Enterprise Miner project contains materials related to a particular analysis task. These materials include analysis process flows, intermediate analysis data sets, and analysis results.

To define a new project, you must specify a project name and the location of the project on the SAS Foundation Server. In this case, a project was already created for you.

1. Open SAS Enterprise Miner and log on. In the classroom, the user name and password should be prefilled for you. Below, the password is **Metadata0** (with a capital M and the numeral zero).

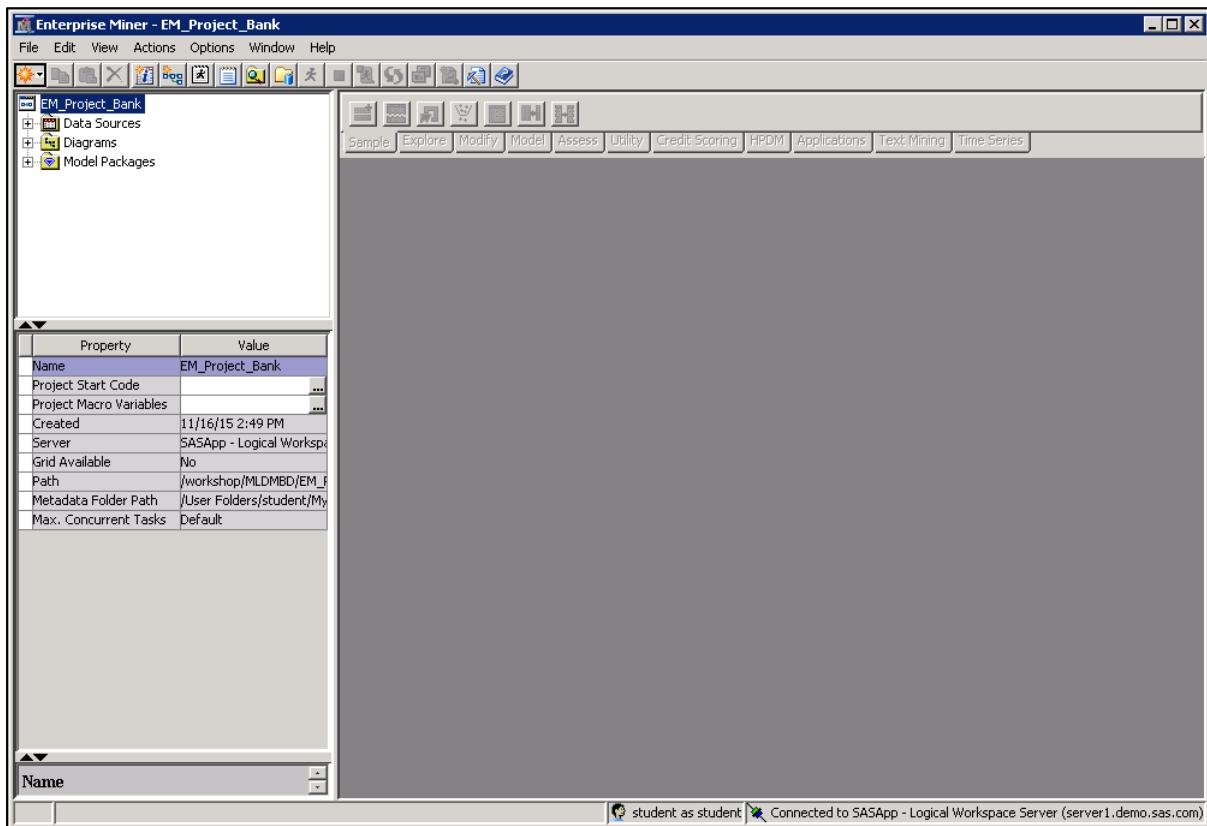


2. Select Open Project.

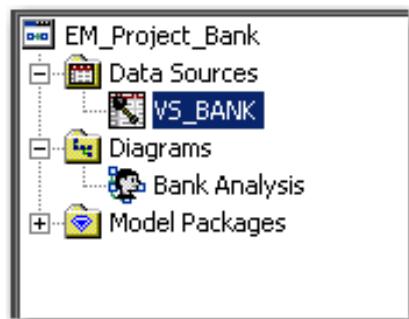


3. Navigate to the following location: \workshop\MLDMBD\Enterprise Miner Projects.
 - ✍ The project location for your class might be different. Your instructor can provide the correct project directory.
4. Select the **EM_Project_Bank** folder and click **OK**.

The SAS Enterprise Miner client application opens for the project that you selected.

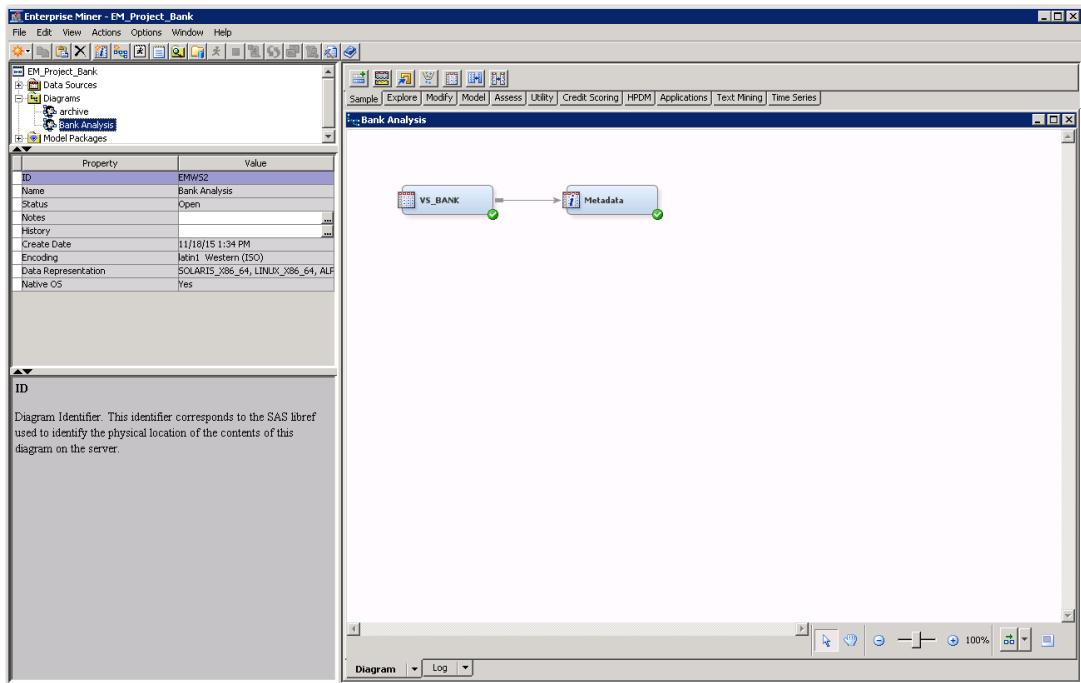


5. In the Project panel (upper left corner of the project window), click the plus signs (+) next to the **Data Sources** folder and the **Diagrams** folder. Notice that the **VS_BANK** data source and a **Bank Analysis** diagram are predefined.



6. Double-click the **Bank Analysis** diagram to open it.

Notice that the **VS_BANK** data source and a metadata node are already present in the diagram. The metadata node is used to define the metadata for **VS_BANK**.



Data Source Configuration

1. Scroll down to the Variables section in the Metadata node properties. Select the ellipsis for the Train property. The Variables window appears.
2. You can expand the heading column for Label. It provides a rudimentary data dictionary of the variables in the VS_BANK data set.

The screenshot shows the 'Variables - Meta2' window, which is a data dictionary for the VS_BANK data set. The 'Label' column is highlighted with a red circle. The table includes columns for Name, Label, New Role, and New Level. The 'Label' column lists variable names like account, logi_rfm4, logi_rfm2, logi_rfm12, logi_rfm3, logi_rfm10, logi_rfm9, logi_rfm1, logi_rfm11, logi_rfm8, ri_demog_inc, logi_rfm5, ri_demog_home, logi_rfm6, logi_rfm7, demog_ho, i_demog_age, demog_genm, demog_pr, cat_input1, and cat_input2, along with their corresponding labels and roles.

Name	Label	New Role	New Level
account	Account ID	ID	Nominal
logi_rfm4	logi_rfm4 Last Product Purchase Amount	Input	Interval
logi_rfm2	logi_rfm2 Average Sales Lifetime	Input	Interval
logi_rfm12	logi_rfm12 Customer Tenure	Input	Interval
logi_rfm3	logi_rfm3 Avg Sales Past 3 Years Dir Promo	Input	Interval
logi_rfm10	logi_rfm10 Count Total Promos Past Year	Input	Interval
logi_rfm9	logi_rfm9 Months Since Last Purchase	Input	Interval
logi_rfm1	logi_rfm1 Average Sales Past 3 Years	Input	Interval
logi_rfm11	logi_rfm11 Count Direct Promos Past Year	Input	Interval
logi_rfm8	logi_rfm8 Count Prchsd Lifetime Dir Promo R	Input	Interval
ri_demog_inc	demog_ri Income	Input	Interval
logi_rfm5	logi_rfm5 Count Purchased Past 3 Years	Input	Interval
ri_demog_home	demog_ri Home Value	Input	Interval
logi_rfm6	logi_rfm6 Count Purchased Lifetime	Input	Interval
logi_rfm7	logi_rfm7 Count Prchsd Past 3 Years Dir Prom	Input	Interval
demog_ho	demog_Homeowner Binary	Input	Binary
i_demog_age	demog_i Customer Age	Input	Interval
demog_genm	demog_Male Binary	Input	Binary
demog_pr	demog_Percentage Retired	Input	Interval
cat_input1	category 1 Account Activity Level	Input	Nominal
cat_input2	category 2 Customer Value Level	Input	Nominal

3. Explore the variables as needed. You can highlight sections of variables by holding down the Ctrl key and selecting the desired variables to highlight them. Then click the **Explore** button in the lower part of the window.
4. No changes to the variable metadata are performed. Click **OK** to close the Variables window.
5. Right-click the **Metadata** node and click **Run**.
6. Select **OK** in the Run Status window.

Running the Principal Components Node

1. Drag an **HP Principal Components** node from the HPDM tab and connect the **Metadata** node to the HP Principal Components node, as shown below.



2. Click the **HP Principal Components** node in the diagram to select the node.
3. Find the ellipsis for the **Variables** property in the **HP Principal Components** node. Click the ellipsis to open the Variables window.

It is not uncommon to avoid using principal components analysis (PCA) on categorical data. Discrete variables that are binary in nature tend to be less problematic (Gower, J.C. 1966). However, in this example, you use only continuous or interval-valued variables in the PCA analysis.

4. Change the property value in the **Use** column from Default to **No** for the four nominal and binary scaled inputs.

Name	Use	Role	Level /
demog_genm	No	Input	Binary
demog_ho	No	Input	Binary
b_tgt	Default	Target	Binary
logi_rfm5	Default	Input	Interval
logi_rfm6	Default	Input	Interval
logi_rfm3	Default	Input	Interval
logi_rfm2	Default	Input	Interval
logi_rfm4	Default	Input	Interval
logi_rfm9	Default	Input	Interval
logi_rfm8	Default	Input	Interval
ri_demog_home	Default	Input	Interval
logi_rfm7	Default	Input	Interval
ri_demog_inc	Default	Input	Interval
i_demog_age	Default	Input	Interval
logi_rfm1	Default	Input	Interval
cnt_tgt	Default	Target	Interval
demog_pr	Default	Input	Interval
logi_rfm11	Default	Input	Interval
logi_rfm12	Default	Input	Interval
logi_rfm10	Default	Input	Interval
cat_input1	No	Input	Nominal
account	Default	ID	Nominal
cat_input2	No	Input	Nominal

5. Click **OK** to close the Variables window.

6. Examine the settings of the properties in the Select Principal Components section in the Properties panel.

Apply Maximum Number	Yes
Maximum Number	20

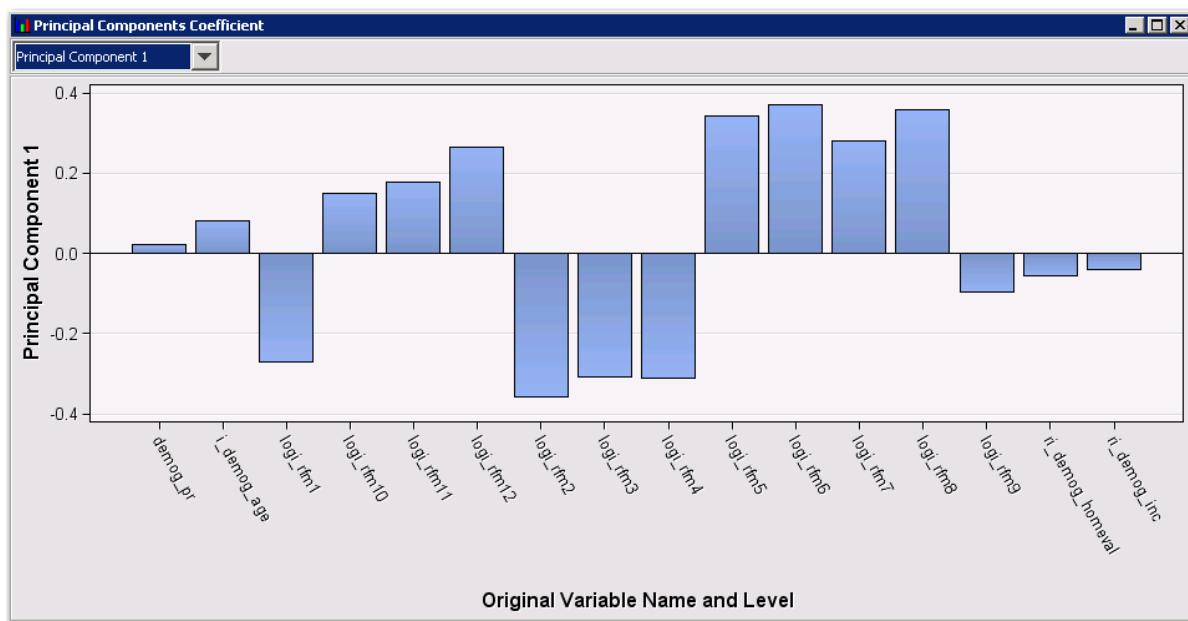
The **Yes** setting of the Apply Maximum Number property indicates that the number in the Maximum Number property on the line below is the absolutely highest number of principal components that could be selected as output variables when the node is run. Notice that the default value is a maximum of 20 principal components.

Cumulative	0.99
Increment	0.001

The Cumulative property specifies the cutoff value for the cumulative proportion of the total variance explained by the principal components. Principal components with a value higher than this cutoff value are not passed as input variables to successor nodes. The default value is 0.99.

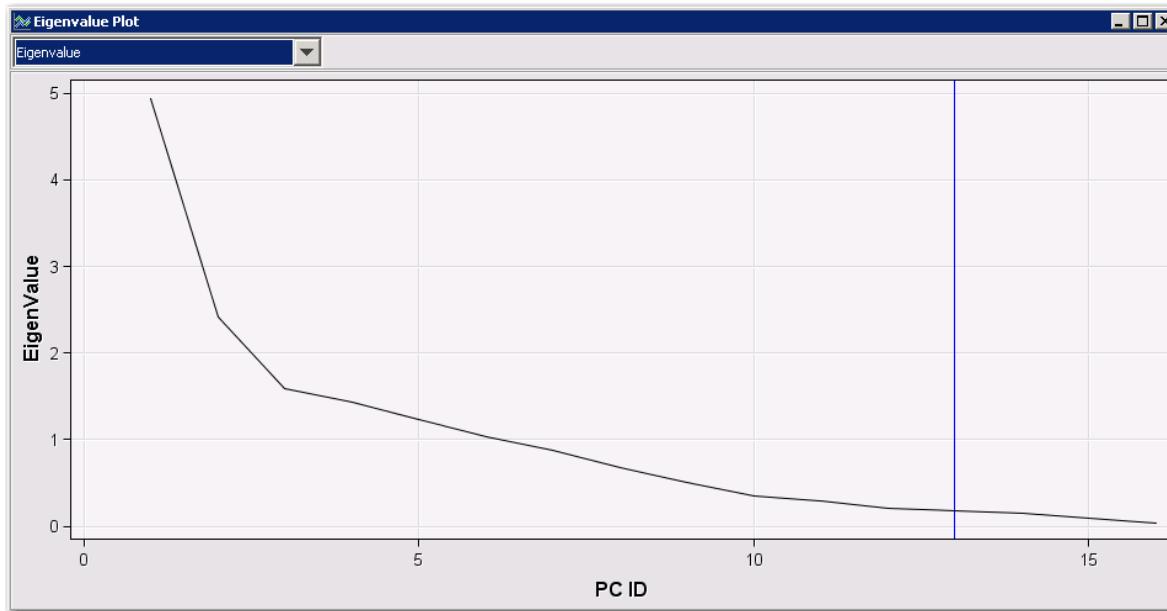
The Increment property specifies the cutoff value for the increment of the cumulative proportion of the total variance explained by the principal components, when a new principal component is considered. If the cumulative proportion of the total variance explained by the principal components reached 0.90, principal components with a proportional increment less than the value specified in the Increment property are not passed as input variables to successor nodes. The default value is 0.001.

7. Run the HP Principal Components node with default settings. Right-click the **HP Principal Components** node in the diagram and select **Run**.
8. Select **Results** in the Run Status dialog box. The Results - Node: Principal Components Variable Selection diagram appears.
9. Maximize the Principal Components Coefficient window.



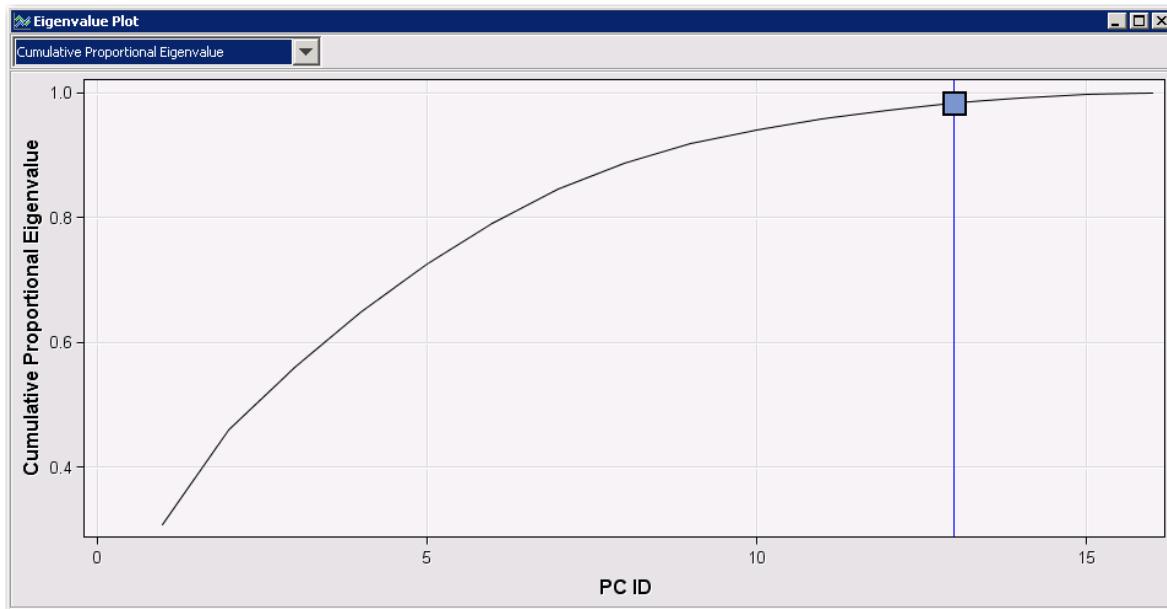
This plot illustrates the estimated coefficients for the first linear combination of the input variables (that is, for the first principal component). To display the estimated coefficients for another principal component, use the drop-down menu in the upper left corner.

10. Tile the **Principal Components Coefficient** plot and examine the Eigenvalue Plot window.



By default, the scree plot is shown. The eigenvalues are plotted on the Y axis and the number of the principal components on the X axis. The vertical blue line at principal component 13 means that 13 principal components are passed forward as input variables to successor nodes in the process flow. From the results shown in the scree plot, you might choose to use two or three principal components because the most drastic change in slope (that is, the elbow) occurs for PC ID 2 and PC ID 3. This approach to selecting the appropriate number of principal components is known as the *scree test* (Bryant and Yarnold, 1995). The scree test suggests that you examine the scree plot and identify where the slope of the line stops descending precipitously. All components that follow and *include* the leveling-off or transition point should be ignored.

11. Use the drop-down menu in the upper left corner and select **Cumulative Proportional Eigenvalue**.



Why are 13 principal components selected? Why are 14 principal components not selected?

The default value for the Maximum Number property in the Properties panel is 20. Thus, another criterion was used to select 13 principal components.

Move the mouse pointer on the curve for principal component 14. A tooltip tells that the cumulative proportional eigenvalue of principal component 14 is 0.9916. The default setting for the Cumulative property in the Properties panel is 0.99. That is, this criterion appears to be exceeded at principal component 14. Thus, you can conclude that the cumulative proportional eigenvalue criterion was used to select 13 principal components.

- Maximize the **Output** window and scroll down to line 33.

Simple Statistics			
		Mean	Standard Deviation
30			
31			
32			
33	Variable	Label	
34			
35	demog_pr	demog Percentage Retired	30.56937 11.52601
36	i_demog_age	demog_i Customer Age	58.71637 14.57619
37	logi_rfml	logi_rfml Average Sales Past 3 Years	2.68603 0.57077
38	logi_rfml0	logi_rfml0 Count Total Promos Past Year	2.59002 0.27435
39	logi_rfml1	logi_rfml1 Count Direct Promos Past Year	1.82454 0.23487
40	logi_rfml2	logi_rfml2 Customer Tenure	4.06089 0.62602
41	logi_rfml2	logi_rfml2 Average Sales Lifetime	2.54387 0.46837
42	logi_rfml3	logi_rfml3 Avg Sales Past 3 Years Dir Promo Resp	2.68001 0.45573
43	logi_rfml4	logi_rfml4 Last Product Purchase Amount	2.76024 0.54452
44	logi_rfml5	logi_rfml5 Count Purchased Past 3 Years	1.23313 0.51981
45	logi_rfml6	logi_rfml6 Count Purchased Lifetime	2.04993 0.81291
46	logi_rfml7	logi_rfml7 Count Prchsd Past 3 Years Dir Promo Resp	0.82397 0.56702
47	logi_rfml8	logi_rfml8 Count Prchsd Lifetime Dir Promo Resp	1.52294 0.76329
48	logi_rfml9	logi_rfml9 Months Since Last Purchase	2.93453 0.26104
49	ri_demog_homeval	demog_ri Home Value	107318 92589
50	ri_demog_inc	demog_ri Income	50013 17415

You use 16 inputs, or said differently, you apply principal components analysis to 16 degrees of freedom.

- Scroll down to line 55 in the Output window.

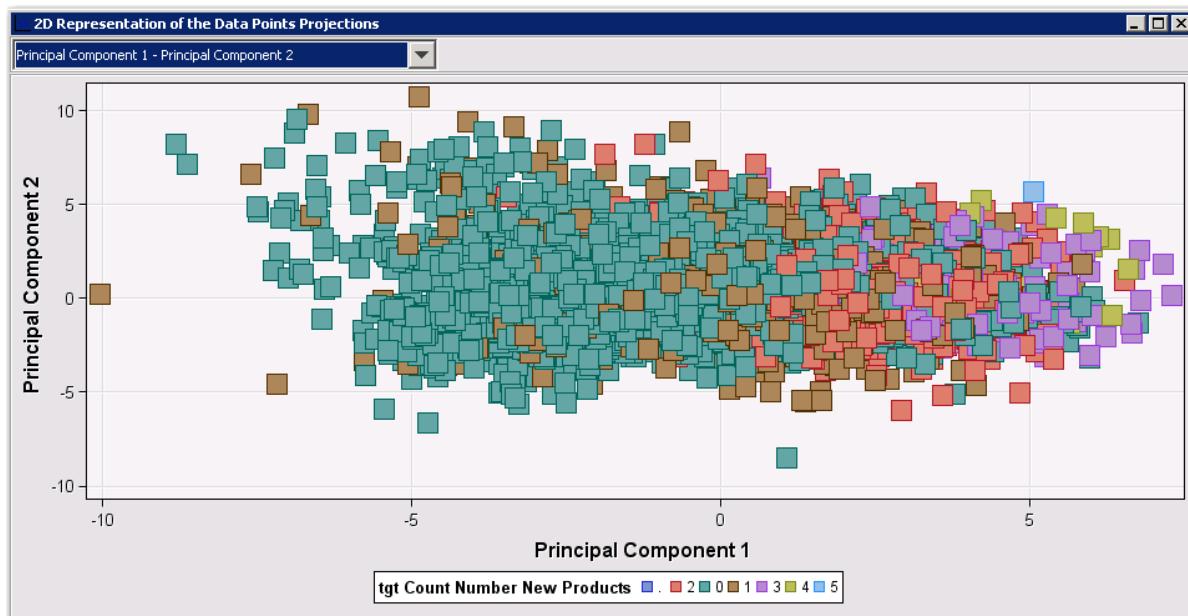
Eigenvalues of the Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
52				
53				
54				
55	1	4.939156	2.520097	0.3087 0.3087
56	2	2.419059	0.835677	0.1512 0.4599
57	3	1.583383	0.151544	0.0990 0.5588
58	4	1.431839	0.198857	0.0895 0.6483
59	5	1.232982	0.200777	0.0771 0.7254
60	6	1.032205	0.151478	0.0645 0.7899
61	7	0.880727	0.206262	0.0550 0.8450
62	8	0.674465	0.166501	0.0422 0.8871
63	9	0.507964	0.162395	0.0317 0.9189
64	10	0.345569	0.058092	0.0216 0.9405
65	11	0.287477	0.086270	0.0180 0.9584
66	12	0.201207	0.016219	0.0126 0.9710
67	13	0.184988	0.039821	0.0116 0.9826
68	14	0.145167	0.053048	0.0091 0.9916
69	15	0.092120	0.050426	0.0058 0.9974
70	16	0.041694		0.0026 1.0000

At 13 principal components, you reduce the degrees of freedom by 3 and still explain .9826 percent of the total variation of the original inputs.

You might remember that the number of principal components that were chosen by the default properties of the HP Principal Components node is 13. The 13 chosen principal components were determined by the cumulative proportional eigenvalue criterion. Was the cumulative proportional eigenvalue criterion the correct criterion to use for selecting the optimal number of principal components? If "yes," then was .99 percent the correct cutoff for cumulative proportional variation explained?

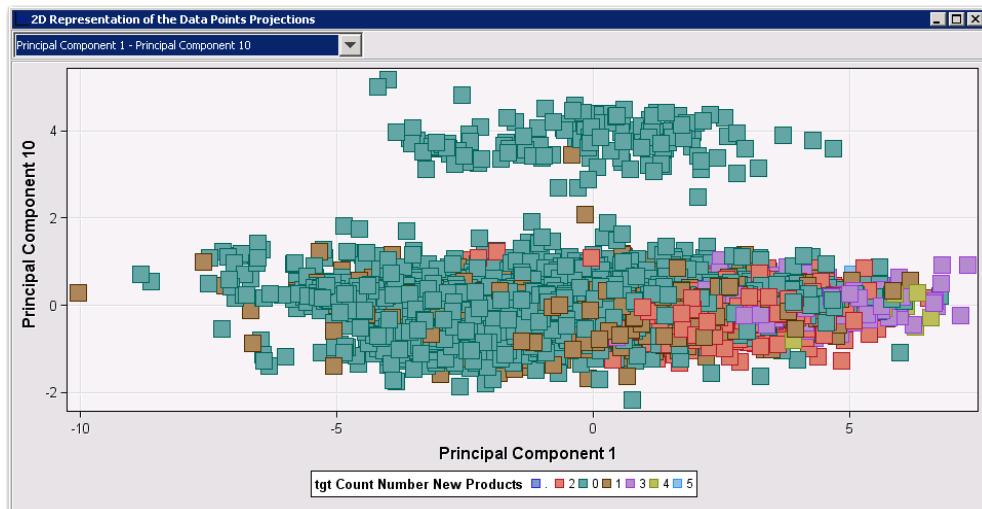
The appropriate cutoff for cumulative proportional variation is undeterminable by cumulative proportional variation alone. The scree test mentioned before is a strategy that is commonly used for selecting the optimal number of components. Another rule for selecting the number of principal components is known as the Kaiser-Guttman method. The Kaiser-Guttman method states that only the number of principal components with eigenvalues greater than 1.00 should be considered in the analysis. Using the Kaiser-Guttman method guides you to use only the first six principal components.

14. Tile the Output window and now examine the 2D Representation of the Data Points Projections.



The 2D Representation of the Data Points Projections plot shows the interaction between each pair of principal components. The color of each point represents the value of the **cnt_tgt** variable.

15. Change the drop-down menu to view the interaction between **Principal Component 1** and **Principal Component 10**.



It appears that the combination of Principal Component 10 values less than 2 and Principal Component 1 values greater than 0 generate separation in target level outcomes. This indicates that an interaction term comprised of Principal Component 1 and Principal Component 10 might be a useful input in subsequent modeling efforts.

End of Demonstration

2.3 Solutions

Solutions to Exercises

1. Using SAS Visual Analytics Data Explorer

- a. Sign in to Visual Analytics.
- b. Select **Data Explorer**.
- c. Select the **PVA97NK** data source.
- d. Select **Data Source Details** from the Data pane drop-down list.
 - 1) How many rows of data does **PVA97NK** contain? **106,546**
 - 2) How many columns of data does **PVA97NK** contain? **29**
 - 3) How many category variables does **PVA97NK** contain? **5**
 - 4) How many measure variables does **PVA97NK** contain? **24**
- e. Select **Measure Details** from the Data pane drop-down list to view the properties of all the measures in the data.
 - 1) Which measure variables have missing values? **Age, Gift Amount Average Card 36 Months, Target Gift Amount**
 - 2) How many missing values are there for **Target Gift Amount**? **53,273**
 - 3) How many distinct demographic clusters are represented in the data? **54**
 - 4) How many observations represent those with **Status Category 96NK=F**? **7260**
- f. Change **Target Gift Flag** from a measure to a category. It is a binary indicator that represents a response to a mailing, where 1 indicates that customers did respond.
Use a bar chart. How many females responded to the campaign? **28,699**
- g. Create a histogram of **Target Gift Amount**.
 - 1) Is the variable skewed? **yes**
 - 2) If so, in which direction? **to the right**
- h. Save the exploration. Click the **Save** icon on the toolbar and save the exploration in My Folder with the name **Exercise 1**.

2. Performing a Cluster Analysis

- a. Perform a cluster analysis for the **Gift...** variables. Experiment with three-, four-, and five-cluster solutions.
 - 1) From the Visual Analytics Home Page, select **Data Exploration**. Select the **PVA97NK** data source.
 - 2) Click the **Cluster** icon.
 - 3) Highlight and drag all of the **Gift...** variables onto the workspace.

Five clusters are created by default. Some of the variables are quite skewed, showing outliers on the bivariate scatterplot matrix. Explore the three- and four-cluster solutions.

Which cluster gave the most money per donation? **The parallel coordinates plot indicates that the largest gift amounts per donation were generally in cluster 1.**

Which cluster donated most frequently? **The most frequent gifts were among cluster 2.**

- b. Save the analysis as **Exercise 2**.

End of Solutions

Solutions to Student Activities (Polls/Quizzes)

2.01 Multiple Choice Poll – Correct Answer

In SAS Visual Statistics, the default number of clusters is which of the following?

- a. determined by the CCC statistics
- b. the optimal number of clusters, given the distribution of the data
- c. five, but you can change this to another number
- d. There is no default; the user must specify the number.

33

2.02 Multiple Choice Poll – Correct Answer

Which of the following is true regarding the HPDM nodes in Enterprise Miner?

- a. HPDM node functionality is not available for data sources with less than one million rows.
- b. Data exploration and predictive modeling can be performed in Enterprise Miner using only HPDM nodes.
- c. Properties and functionality associated with HPDM nodes probably never change.

72

Chapter 3 Analysis Methods for Categorical Targets

3.1 Categorical Targets (SAS Visual Statistics).....	3-3
Logistic Regression.....	3-3
Demonstration: Creating a Logistic Regression in SAS Visual Analytics	3-22
Exercises.....	3-26
Interactive Group By Processing	3-28
Demonstration: Adding a Group By Variable to a Logistic Regression.....	3-34
Decision Trees	3-36
Demonstration: Creating and Cultivating a Decision Tree in SAS Visual Analytics.....	3-46
Exercises.....	3-51
Tree Split Search in SAS Visual Statistics (Self-Study).....	3-53
3.2 Categorical Targets (SAS In-Memory Statistics)	3-62
Decision Trees in PROC IMSTAT	3-63
Demonstration: Growing a Decision Tree	3-72
Demonstration: Assessing a Decision Tree.....	3-91
An Introduction to Random Forests (Self-Study).....	3-101
Demonstration: Growing a Forest of Trees	3-119
Demonstration: Scoring Data with the RANDOMWOODS Score Code.....	3-123
Logistic Regression in PROC IMSTAT.....	3-126
Demonstration: Developing and Assessing a Logistic Regression Model.....	3-129
3.3 Solutions	3-145
Solutions to Exercises	3-145
Solutions to Student Activities (Polls/Quizzes)	3-151

3.1 Categorical Targets (SAS Visual Statistics)

Logistic Regression

Objectives

- Discuss options related to the SAS Visual Statistics functionality.
- List and explain the logistic regression results.
- Create a binary logistic regression in SAS Visual Analytics.

3

Logistic Regression in SAS Visual Analytics

- Logistic regression enables you to investigate the relationship between a discrete response variable and one or more effects variables.
- There is only one discrete response variable (**Response**).
- This chapter focuses on binary logistic regression and decision trees.
- There can be multiple effects variables that can be
 - continuous (**Continuous Effects**)
 - categorical (**Classification Effects**)
 - interaction terms (**Interactions**).

4

Classification models predict the probability of class membership. You try to classify whether someone is likely to leave, whether he or she might respond to a solicitation, whether he or she is a good or bad credit risk, and so on. The response variable might be represented as a 0 or 1, with 1 as the event that you are targeting. You can build a classification model where the response variable has several possible values. You are trying to predict which group observations belong to (such as low, medium, or high) or predict which product someone might buy.

Logistic Regression in SAS Visual Analytics

- If the response variable is binary, those values are used by SAS Visual Statistics as a binary {0, 1} outcome.
 - You can specify which level is the event to model.
- If the response variable is multinomial, you can choose one response level as the *event*, and the others are grouped into one *non-event*.
- If the response variable is continuous, you can create a calculation rule to convert it into a dichotomized variable for simple binary logistic regression. (For example, Weights > 200 is the event.) This can be performed in SAS Visual Analytics.

5

-  For response variables that are continuous and integer (such as count data), you can simply convert that variable from a measure to a category by right-clicking in the Data pane before you begin assigning roles in SAS Visual Statistics.
- Of course, there are other ways to model categorical responses with logistic regression, including ordinal and multinomial logistic regression. These models are available for in-memory processing using the HPLOGISTIC or HPGENSELECT procedures, and also using other SAS/STAT and SAS/ETS procedures.

Logistic Regression: Roles

- Response – assign only one discrete response variable (category)
- Advanced – choose the event level to be modeled
- Continuous Effects – assign one or more measures
- Classification Effects – assign one or more categories
- Interactions – assign one or more interaction terms
- Group By – can assign one or more categories as group by variables
- Frequency – only one measure
- Offset – only one measure
- Weight – only one measure

6

Logistic regression requires that a discrete or categorical variable is assigned as the response. If you assign a categorical variable that has more than two levels, a dialog box is displayed. The dialog box enables you to select the event level that you want to model. Click **Choose**, and the Response Event Settings window appears. In this window, you can choose the event level.

You can also access the Response Event Settings window by clicking **Advanced** to the right of **Response**. You can now select the event level to model. By default, the event levels are sorted alphabetically. Make sure that you click **Advanced** and that you are modeling for the event of interest.

You need to create the interaction terms before you can assign them to the Interactions role.

If you specify more than one value for the Group By role, the groups are combined to form compound groups. (See the slides regarding Group By in the upcoming section for more information.)

Frequency specifies the variable or variables that are used to perform the frequency analysis for each effect. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used.

Offset is often used in Poisson regression with the log link function to account for exposure. An offset variable is treated like a regression covariate whose parameter is fixed to be 1.0.

Weight specifies the numeric variable to use as a weight variable when you solve the linear model.

Only observations with a weight value that is not missing and greater than zero are used in the analysis.

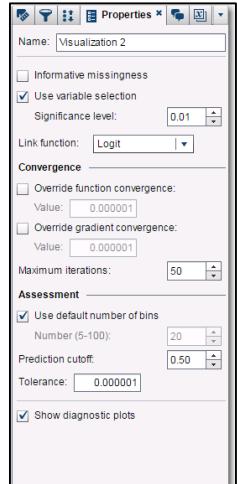
Adding roles to the model automatically updates the model. If you do not want the model to automatically update as you add roles, clear the **Auto-update model** check box on the Roles tab. When all of the roles are defined, you can update the model by clicking **Update** at the bottom of the Roles tab.

Idea Exchange

- Describe some categorical target variables that you encounter in your work.
- What modeling algorithm do you use most often to classify them?

Logistic Regression: Properties

- Name of model
- Informative missingness
- Use variable selection
- Link function
- Convergence
 - Override function convergence
 - Override gradient convergence
 - Maximum iterations
- Assessment
 - Use default number of bins
 - Number
 - Prediction cutoff
 - Tolerance
- Show diagnostic plots
(residual plot, assessment, influence plot)



8

By default (except for decision trees), the SAS Visual Statistics functionality handles missing values by dropping all observations that contain a missing value in any assigned role variable. However, the linear regression, logistic regression, and GLM models provide the informative missingness property.

Informative missingness requests that missing values be handled by modeling them through extra model effects. These effects consist of dummy variables that take on the value *1* when the value of a continuous model variable that is involved in the effect is missing. Otherwise, they are assigned the value *0*. The missing value in the original model effect is replaced with the average value for the effect from the nonmissing values. For category variables, missing values are considered a distinct level. (The same is true for a missing level of Group By.)

If you select the **Use variable selection** check box, the fast backward selection method is used to determine whether effects should stay in the model during variable selection. You can then also set the significance level at which the effects should remain in the model. Variable selection attempts to reduce the number of input variables to include only the most important variables.

Link function specifies the link function to use for the model-fitting process. Use the drop-down list to make a selection. These are your choices:

- **Logit** (default) – the inverse of the cumulative logistic distribution function
- **Probit** – the inverse of the cumulative standard normal distribution function

The Convergence properties enable you to enter values that control how quickly the model converges or that find a satisfactory solution.

- **Override function convergence** – Entering a large value reduces the time to train the model but can create a suboptimal model.
- **Override gradient convergence** – Entering a large value reduces the time to train the model but can create a suboptimal model.
- **Maximum iterations** – Entering a small value can reduce the time to train the model but can create a suboptimal model.

Assessment

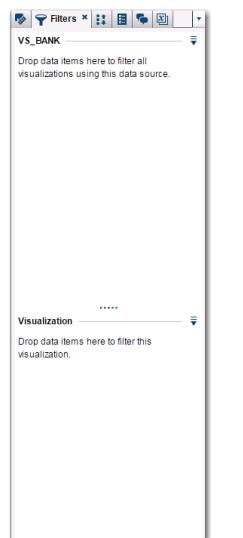
- **Use default number of bins** specifies the number of bins to use in the lift calculations in the assessment plot. This option is enabled by default and set at 20. You can clear the check box and enter your own number of bins. Increasing the number of bins enables the user to look at a smaller percentile of data at the expense of computing time.
- **Prediction Cutoff** specifies the value above which a computed probability is considered an event.
- **Tolerance** specifies the tolerance value that is used to determine the convergence of the iterative algorithm that estimates the percentiles. Specify a smaller value to increase the algorithmic precision.

Show diagnostic plots displays the residual plot, the assessment, and the influence plot. This option is enabled by default. Clear the check box if you do not want the diagnostic plots to be displayed.

Filters

Logistic Regression: Filters

- There are two types of filters.
- Filter variables applied in the top part of the Filters tab are applied to all visualizations in the exploration using the listed data source.
- Filter variables applied in the bottom part are applied only to the active visualization.



9

Filters can be created by either dragging and dropping variables into the desired parts of the Filters tab, or by right-clicking variables under the Data section and assigning them as filter (visualization or data source) variables.

Outlier observations can be excluded to assess their impact on the results. To exclude points in a residual plot, place the mouse pointer in the Residual Plot panel and hold down the right mouse button. Drag the cursor over the points that you want to exclude. A blue box is displayed around the plots that you select.

 Both the residual plot and influence plot can be viewed as histograms from which you can also exclude data.

You can view how many observations were excluded by looking at the observations number above the Fit Summary panel. After you exclude some observations, the observations number is updated to reflect how many observations remain.

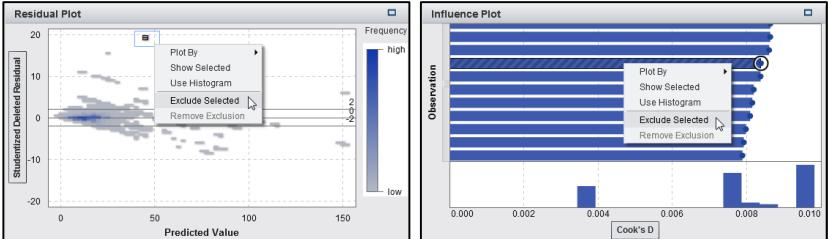
When data are excluded, the Filters tab displays an Excluded Data section that contains the message “Data has been excluded from results.” To re-include the data, you can select **Remove All** from the drop-down list on the Filters tab, or you can right-click in either the residual or influence plot and select **Remove Exclusion**.

If a group by variable was assigned, you can exclude observations from a specific group or segment. In the Goodness of Fit panel, select a horizontal bar that represents the group of interest. Both the residual plot and the influence plot are updated to show data only for that group. Then, make your selections from either one.

Filtering Data by Excluding Selections

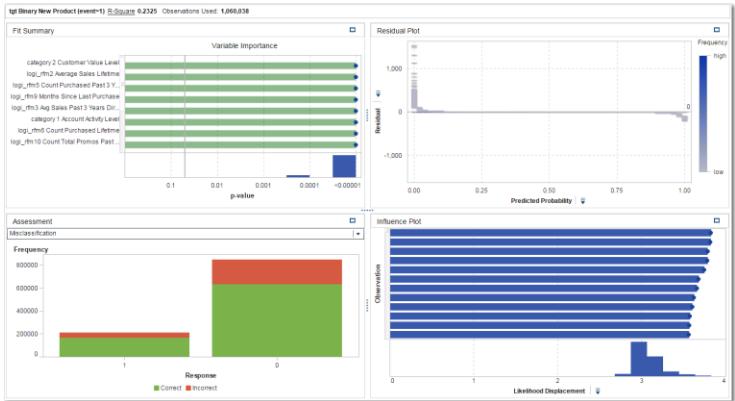
You can also filter data by excluding selections from either a residual plot or influence plot.

- Select one or more points in a residual plot. Then, right-click and select **Exclude selected**.
- Select one or more bars in an influence plot. Then, right-click and select **Exclude selected**.



10

Logistic Regression Results



11

After a logistic model is created, a summary bar along the top of the four panels displays the results. Although the Logistic Regression results panels look essentially the same as for linear regression, there are some differences in the statistics and criteria for a logistic model that are detailed below.

Notice the different model criteria selections that are available on the summary bar.

These are the selections for the Y axis on the **residual plot**:

- Residual
- Pearson Residual
- Deviance Residual
- Standardized Pearson Residual

 You can also switch to **Linear Predictor** (versus **Predicted Probability**) on the X axis of the residual plot to display the predicted values rather than the probabilities.

The Assessment panel enables you to choose to display a lift chart, an ROC chart, or a misclassification chart. These charts are discussed on the slides that follow.

These are the selections for the X axis on the **influence plot**:

- Likelihood Displacement
- CBAR
- Deviance Change
- Pearson Change

Summary Bar

For all models except a cluster model, general model information is displayed at the top of the Model pane.

- name of response variable
- model evaluation criteria (selectable from a drop-down list)
- number of observations used to build the model
- link to show or hide the summary table

tgt Binary New Product (event=1) R-Square 0.2325 Observations Used: 1,060,038		
Fit Summary	-2 Log Likelihood	
category 2 Customer Value Le	AIC	
logi_rf2 Average Sales Lifeti	AICC	
	BIC	
	Max-rescaled R-Square	Variable Impor
	R-Square	

12

You can click the model evaluation criteria on the summary bar to select different criteria from the pop-up menu. The choices depend on the model type that is used. For a logistic regression, these are the evaluation criteria choices:

- -2 Log Likelihood
- AIC
- AICC

- BIC
- Max-rescaled R-Square
- R-Square

If a group by variable was assigned to the model, the number of observations that are displayed at the top represents the observations for the group selected in the Variable Importance panel.

Analyzing Logistic Regression Results

These four panels are displayed to help you analyze the results of the logistic regression model:

- Fit Summary – displays how significant the effects variables are to the response variable.
- Residual Plot – displays the difference between the predicted data and the actual data.
- Assessment
 - Lift measures model effectiveness.
 - ROC (receiver operating characteristic) measures classification accuracy.
 - Misclassification measures predictive accuracy.
- Influence Plot – displays the observations that might influence the overall analysis.

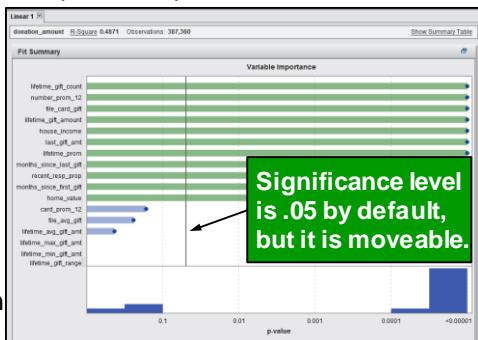
13

The assessment for a logistic regression model consists of three charts that are selectable from the drop-down list.

- A *lift chart* is a graphical representation of the advantage (or lift) of using a predictive model to capture target responders versus the naïve (or “coin flip”) model. The idea is that if the model is good, the data, when rank-ordered by the model’s assigned probabilities, have a high proportion of responders in the top deciles. The lift associated with the naïve model is represented by a horizontal line with an intercept of 1. When the lift is higher, the model is proportionally better.
- The *ROC (receiver operating characteristic) chart* is a graphical display that gives a measure of the predictive accuracy of a logistic model. The classification accuracy of a model is demonstrated by the degree that the ROC curve pushes upward and to the left. This degree can be quantified by the area under the curve. The area ranges from 50, for a worthless model, to 100, for a perfect classifier.
- A *misclassification plot* displays how many observations were correctly and incorrectly classified for each value of the response variable. When the response variable is not binary, the logistic regression model considers all levels that are not events as equal.

Fit Summary

- The Fit Summary panel displays how significant the effects (predictor) variables are to the response variable by displaying a Variable Importance panel.
- The vertical significance level line plotted is $-\log(.05)$.
- Notice that $-\log(.1) = +1$.
- This panel is displayed for
 - linear regression
 - logistic regression
 - generalized linear model.



14

The Fit Summary panel is used to determine the most significant predictor variables that affect the response variable.

The variable importance plot displays the effects on the Y axis and the p -values on the X axis. The variable importance is based on the negative log of the p -value ($-\log(p\text{-value})$). As the $-\log(p\text{-value})$ increases, the variable becomes more important.

Significance of the effect is shown by the color and length of the horizontal bars. A green horizontal bar shows that the variable importance is above the significance level. The length of the green bar indicates how meaningful the variable is. A blue horizontal bar shows that the variable importance is below the significance level.

The significance level (or alpha value, which is plotted as $-\log(\alpha)$) is set to 0.05 and is displayed as a black vertical line in the panel. Position the mouse pointer on the line to see the alpha and $-\log(\alpha)$ values. Click the line and you move this line to the left or right to change the significance level.

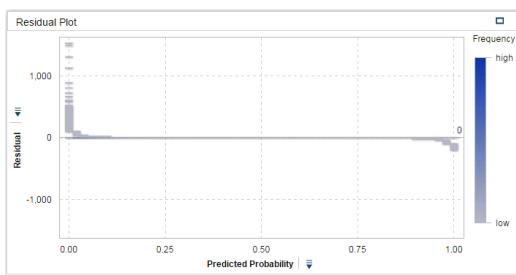
The significance value line is purely for reference in the plot. Changing this line does not impact model estimation or variable selection algorithms.

Position the mouse pointer on the horizontal bars to see the p -value and the $-\log(p\text{-value})$. Positioning the pointer on the histogram bars at the bottom of the graph displays the percent of data that falls within the range.

To remove an effect from the model, click a horizontal bar to select it. Then right-click and select **Remove**.

Residual Plot

- The Residual Plot panel displays the difference between the predicted data and the actual data. It uses a scatter plot or heat map.
- This panel is displayed for
 - linear regression
 - logistic regression
 - generalized linear model.



15

The residual plot is used to assess the quality of the model and identify outlier observations. The plot is displayed as either a scatter plot for smaller data or a heat map when used with larger data. If it is displayed as a heat map, a color-coded frequency bar is displayed on the right side of the graph, which represents the frequency of each intersection of values. As the values become darker, this indicates that they are more frequent.

The residual plot displays the predicted probability of the response variable on the X axis and the residual statistic on the Y axis. You can choose a different statistic for the Y axis by clicking **Residual** and selecting another statistic from the pop-up menu. The choices are listed below.

- Residual
- Pearson Residual
- Deviance Residual
- Standardized Pearson Residual

You can also switch to a linear predictor (instead of predicted probability) on the X axis of the residual plot to display the predicted values rather than the probabilities.

Selecting one or more observations in the residual plot highlights the corresponding bars in the influence plot if the observations have a significant influence on the model.

Select one or more observations, right-click, and select **Exclude Selected** to remove them from the model. The model is refit automatically, unless you cleared the **Auto-update** check box.

Residual plots have several uses when you examine your model. Similar to linear regression, with a large sample size and assuming the fitted logistic regression model being true, you can use the Pearson residuals and deviance residuals for model diagnostics. First, obvious patterns in the residual plot indicate that the model might not fit the data. Second, residual plots can detect non-constant variance in the input data when you plot the residuals against the predicted value. Non-constant variance is evident when the relative spread of the residual values changes as the predicted values change. Last, under the stated conditions, you would expect most residuals to fall within ± 3 . The residual plot can help you identify possible outliers in your data.

Assessment

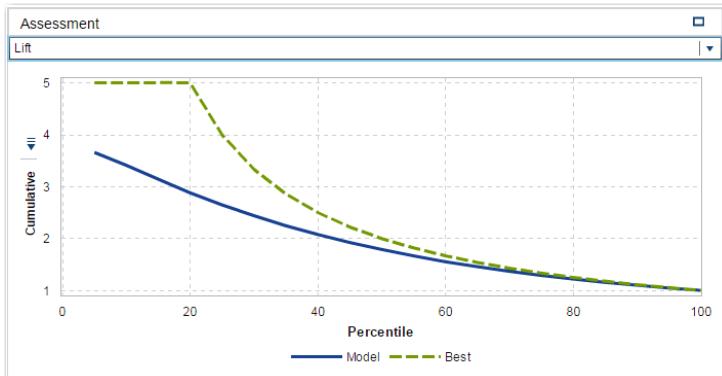
- The Assessment panel defaults to a cumulative lift chart. A ROC chart and misclassification plot are available.
- This panel is displayed for
 - linear regression
 - logistic regression
 - generalized linear model
 - decision tree.

16

The Assessment panel enables you to choose to display a lift chart, ROC chart, or misclassification chart.

If a group by variable was assigned, the assessment plot displays the data from the group selected in the Fit Summary panel.

Logistic Regression Results: Lift



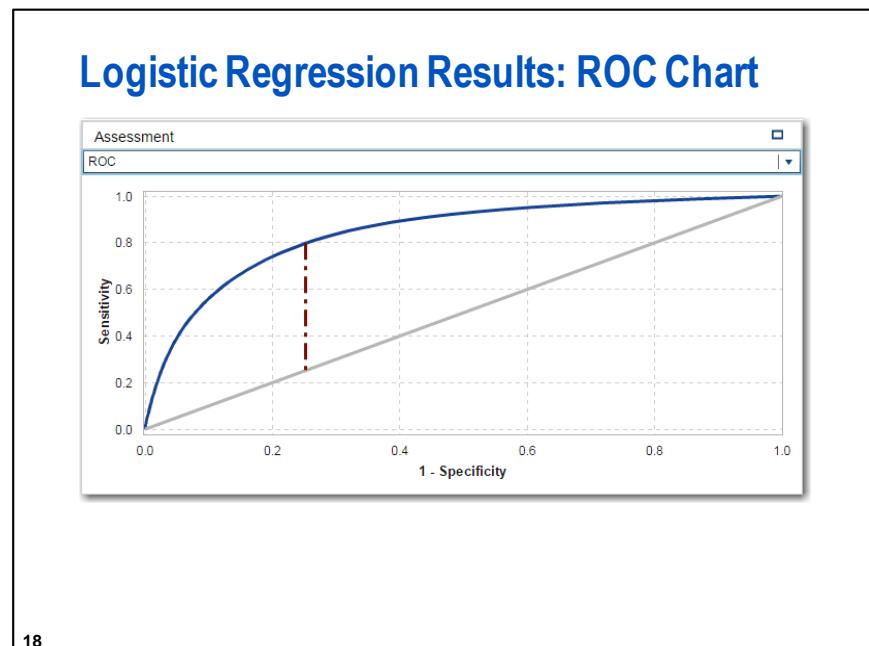
17

Simply, a lift chart is a graphical representation of the advantage (or lift) of using a predictive model to improve on the target response versus not using a model.

Technically, *lift* is the ratio of the percent of captured responses within each percentile bin to the average percent of responses for the model. Similarly, *cumulative lift* is calculated by using all of the data up to and including the current percentile bin. The default lift chart displays the cumulative lift of the model.

The chart shows two lines. One line represents the model that you built, and the other line represents the best achievable model (or a *perfect classifier*). As the Model line moves closer to the Best line, especially in the lower percentiles, the model improves.

Click **Cumulative** along the Y axis to switch between a cumulative chart and lift chart (noncumulative).



18

A ROC chart displays the ability of a model to avoid false positive and false negative classifications. A false positive classification means that an observation was identified as an event when it is actually a nonevent (also referred to as a *Type I error*). A false negative classification means that an observation was identified as a nonevent when it is actually an event (also referred to as a *Type II error*).

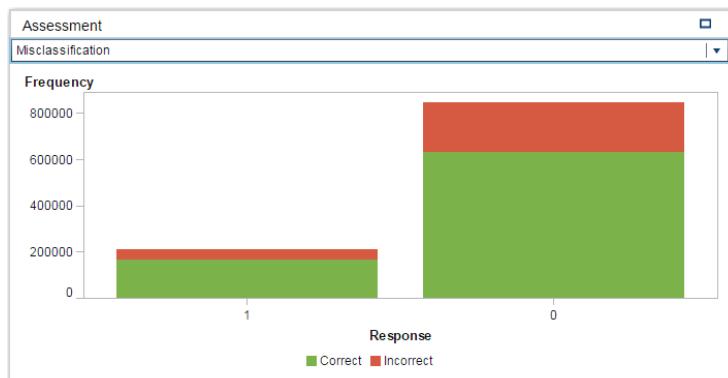
The specificity of a model is the true negative rate. To derive the false positive rate, subtract the specificity from 1. The false positive rate, labeled **1 – Specificity**, is the X axis of the ROC chart. The *sensitivity* of a model is the true positive rate. This is the Y axis of the ROC chart. Therefore, the ROC chart plots how the true positive rate changes as the false positive rate changes.

The classification accuracy of a model is demonstrated by the degree that the ROC curve pushes upward and to the left. This degree can be quantified by the area under the curve. The area ranges from 50, for a worthless model, to 100, for a perfect classifier. For a perfect model, one with no false positives and no false negatives, the ROC chart starts at (0,0), continues vertically to (0,1), and then horizontally to (1,1). In this instance, the model correctly classifies every observation before a single misclassification can occur. The display shown here indicates that the model does not do a good job in classifying the data.

The red vertical line indicates the Kolmogorov-Smirnov (or K-S) statistic. This is a goodness-of-fit statistic that represents the maximum separation between the model ROC curve and the baseline ROC curve. To see the actual number, position the mouse pointer on the top or bottom of the red line.

Logistic Regression Results: Misclassification

Update the cutoff probability for a modeled event and the number of bins used in the model assessment plots on the Properties tab.



19

The misclassification plot displays how many observations were correctly and incorrectly classified for each value of the response variable. When the response variable is not binary, the logistic regression model considers all levels that are not events as equal.

A significant number of misclassifications might indicate that the model does not fit the data. The high number of false negatives and false positives in the display in the slide above suggests that.

You can specify the number of bins to use in the lift calculations in the assessment plot. This is enabled by default and set at 20. Clear the check box and enter your own number of bins. Increasing the number of bins increases the accuracy of the assessment at the expense of computing time.

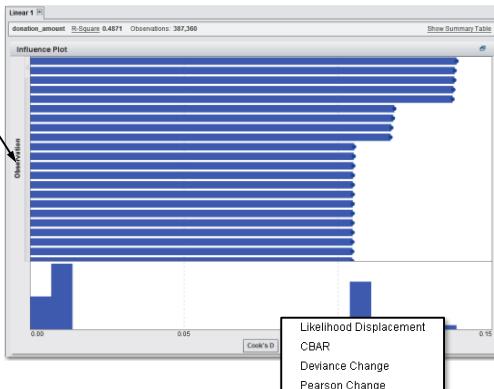
The default prediction cutoff is .5. You can change the value at which a computed probability is considered an event. The misclassification plot is automatically updated.

Influence Plot

- The influence plot displays the observations that might influence the overall analysis.

Only the top 2000 observations are shown.

- This plot is displayed for
 - linear regression
 - logistic regression.



20

The influence plot is used to determine which observations yield a high level of influence on the model. Observations are influential if they are outliers and exhibit leverage. *Leverage* refers to an observation with an extreme value for a predictor variable. An observation is said to be *influential* if removing it from the model substantially changes the regression parameter estimates. An influence plot can help determine whether outliers should be removed from the model or at least deserve more scrutiny.

Only the top 2000 observations are included in the influence plot by default. If you exclude any, the display is refreshed to keep the number of observations being displayed at 2000.

The influence plot displays the individual observations on the Y axis and the estimate of the influence criteria on the X axis, which is **Likelihood Displacement** by default. You can choose a different influence statistic for the X axis by left-clicking **Likelihood Displacement** and selecting another statistic from the pop-up menu. These are the choices:

- Likelihood Displacement
- CBAR
- Deviance Change
- Pearson Change

The horizontal blue bars can represent either individual observations or multiple observations that have the same values for all effects variables. The length of the bar shows how far the observation or observations are from the normal. Selecting one or more bars displays the corresponding points (or bins) in the Residual Plot panel. Positioning the mouse pointer on a bar displays the values for all of the variables as well as the value for the estimate of influence criteria.

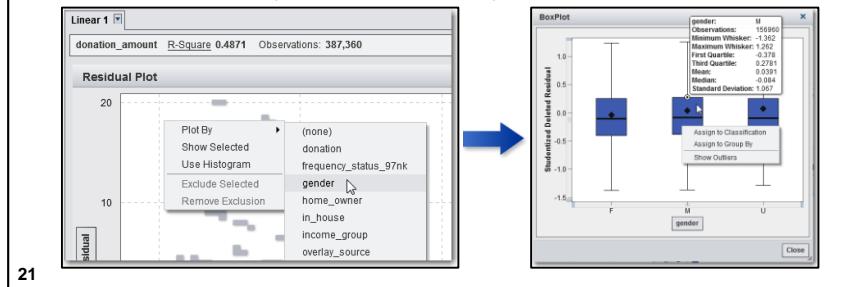
Select one or more observations, right-click, and select **Exclude Selected** to remove the observations from the model. The model is refit automatically (unless you cleared the **Auto-update** check box).

Clicking any of the histogram bars at the bottom of the Influence Plot panel displays the observations represented by the bar.

Residual Plot and Influence Plot Viewing Options

You can do the following tasks:

- display a box plot of all observations against a categorical variable (**Plot By** \Rightarrow **categorical-variable**)
- select data points and display additional data (**Show Selected**)
- display a histogram of all observations (**Use Histogram**)
- exclude selected data points (**Exclude Selected**) or remove the exclusion (**Remove Exclusion**)



21

Right-clicking within either a residual plot or an influence plot and selecting **Plot by** accesses a submenu from which you select a categorical variable. After you make this selection, a box plot is displayed with the model evaluation criteria on the Y axis and the categorical variable on the X axis. The box plot shows the distribution of values using a rectangular box and lines called *whiskers*. Position the mouse pointer on any part of the individual box plots to display the minimum, maximum, first and third quartiles, mean, median, and standard deviation. Right-click anywhere on the box plot, and a pop-up menu appears. The menu lists options to assign the categorical variable as either a classification effect (**Assign to Classification**) or a group by variable (**Assign to Group By**). You can also choose to show outliers on the box plot. The categorical variable on the X axis can be changed by clicking the category at the bottom of the box plot and selecting another value from the pop-up menu. To close the box plot, select **Close**.

Selecting **Show selected** from the pop-up menu displays additional data for the selected observations in a table view.

Right-clicking and selecting **Use histogram** displays the distribution of values for the model evaluation criteria along the X axis and the count of observations along the Y axis. Position the mouse pointer on any of the histogram bars to view the model evaluation criteria value, count, and percent represented by the bar. To change the model evaluation criteria used on the X axis, click the current label at the bottom of the histogram and select another value from the pop-up menu. To close the histogram view and return to the residual or influence plot, right-click anywhere on the histogram and clear the **Use histogram** check box on the pop-up menu.

You can select one or more observations and remove them from the model. Select them, right-click, and select **Exclude selected** from the pop-up list. The observations number at the top of the model is updated to reflect the data still included in the model. The Filters tab also indicates that data are excluded from the results. To re-include the removed observations, right-click the residual plot or influence plot and select **Remove exclusion**. Otherwise, you can click the **Filters** tab and then select **Remove All** from the drop-down list for **Excluded Data**.

Summary Table: Logistic Regression

- The summary table provides detailed statistics about the model via the different tabs, which are model dependent.
 - Dimensions
 - Iteration History
 - Convergence
 - Fit Statistics
 - Type III Test
 - Parameter Estimates
 - Response Profile
- To display the summary table, click **Show Summary Table** in the upper right of the Model pane.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile
Description		Value				
Number of Classification Effects		2				
Number of Columns in X		32				
Rank of Cross-product Matrix		29				
Number of Observations Read		1,549,520				
Number of Observations Used		1,165,920				

22

To hide the summary table, click **Hide Summary Table** in the upper right of the Model pane.

If the summary table is enabled and you maximize any of the individual panels, the summary table remains open at the bottom of the Model pane.

Logistic Regression: Iteration History

The Iteration History tab displays the progress of the estimation results.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile
Iteration	Evaluations	Objective	Change		Max Gradient	
0	4	0.367636	.		0.027438	
1	2	0.367441	0.000195		0.000386	
2	2	0.367441	7.84E-8		2.28E-7	
3	2	0.367441	3.43E-14		1.18E-13	

23

The Iteration History tab shows the progress of the iterative optimization process. The tab also shows the value of the objective function, its change in value, and its maximum gradient.

Logistic Regression: Convergence

The Convergence tab displays the reason and convergence status of the model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile	
Reason	Convergence criterion (GCONV=1E-8) satisfied.						Status
							0

24

Models converge when there is no longer a significant improvement between iterations. On the Properties tab of the logistic model, you have the option to override the default function convergence (FCONV) and gradient convergence (GCONV) numbers. Users should be aware that even though the user specifies convergence criteria on the Properties tab, it is possible that some of the internal controls are met first (such as XCONV, the relative parameter convergence criterion). This might be confusing to the user.

Logistic Regression: Fit Statistics

The Fit Statistics tab displays statistics about the estimated model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile	
Statistic							Value
-2 Log Likelihood							1281652
AIC							1281710
AICC							1281710
BIC							1282057
R-Square							0.030117
Max-rescaled R-Square							0.044491

25

Logistic Regression: Type III Test

The Type III Test tab displays the significance of each effect, considering all other effects in the model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile
Effect		DF	Chi-Square			Pr > ChiSq
age		1	494.3078			<0.0001
file_avg_gift		1	2.299938			0.1294
file_card_gift		1	218.2746			<0.0001
house_income		1	1.164941			0.2804
home_value		1	1550.922			<0.0001
last_gift_amt		1	729.6564			<0.0001
lifetime_avg_gif...		1	2.467593			0.1162
lifetime_gift_a...		1	17.78363			<0.0001
lifetime_gift_co...		1	235.1321			<0.0001
lifetime_gift_ra...		0	.			.
lifetime_max_gi...		0	.			.
lifetime_min_gi...		0	.			.
lifetime_prom		1	16.25604			<0.0001
months_since_...		1	255.6429			<0.0001
months_since_...		1	2710.109			<0.0001
number_prom_...		1	110.4213			<0.0001
recent_resp_pr...		1	4128.207			<0.0001
income_group		7	2937.623			<0.0001
recency_status...		5	1099.794			<0.0001

26

The Type III Test tab examines the significance of each partial effect with all other effects in the model. For more information, see the chapter “The Four Types of Estimable Functions” in *SAS/STAT® 13.2 User’s Guide*.

Logistic Regression: Parameter Estimates

The Parameter Estimates tab displays the parameter estimates (coefficients) of each model effect and the associated statistics.

Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Parameter	Estimate	Standard Error	t Value		Pr > t
Intercept	11.32107	0.142154	79.63963		<0.0001
card_prom_12	0.01774	0.012788	1.387177		0.1654
file_avg_gift	0.348892	0.297737	1.171811		0.2413
file_card_gift	0.103976	0.008109	12.8225		<0.0001
home_value	0.000081	0.000002	4.123035		<0.0001
house_income	0.002275	0.000117	19.38853		<0.0001
last_gift_amt	0.527597	0.002207	239.033		<0.0001
lifetime_avg_gift_amt	-0.23445	0.297723	-0.78749		0.431
lifetime_gift_amount	0.028689	0.000269	106.4795		<0.0001
lifetime_gift_count	-0.26844	0.004924	-54.5134		<0.0001
lifetime_gift_range	0.332109	0.004325	76.78409		<0.0001
lifetime_max_gift_amt	-0.351113	0.004526	-77.5837		<0.0001
lifetime_min_gift_amt	0	.	.		.
lifetime_prom	-0.05038	0.00231	-21.8071		<0.0001
months_since_first_gift	-0.02383	0.001032	-23.0945		<0.0001
months_since_last_gift	0.089265	0.004405	20.26551		<0.0001
number_prom_12	0.059703	0.005615	10.63353		<0.0001
recent_resp_prop	-10.8921	0.162732	-66.9325		<0.0001

27

This display shows that the Parameter column width was expanded to show the full name of the parameter.

Logistic Regression: Response Profile

The Response Profile tab displays information about the values of the binary response variable, such as the level order and frequency.

Ordered Value	donation	Count
1	Donated	294240
2	No	871680

28

The Response Profile tab displays the event and non-event counts.

-  Model generated predictions for the target variable can be retained as temporary variables, and used in post-modeling analyses in Visual Statistics.



Creating a Logistic Regression in SAS Visual Analytics

This demonstration illustrates how to build a logistic regression model in SAS Visual Analytics. The demonstration uses the **VS_BANK** data to model whether a customer contracted for at least one product in the previous campaign season. You create a binary logistic regression with both categorical and continuous explanatory variables.

1. Open the **VS_Bank Cluster** exploration (from a previous chapter) from the My Folder location. (You need the cluster ID variable later in this section.)
2. From the toolbar, click to begin modeling a logistic regression.
3. If you did not do so already, in the Measure column, right-click **tgt Binary New Product** and select **Category**.
4. Click the **Properties** tab if it is not selected. Select the **Use Variable selection** check box and set the significance level to **.01**.
5. Clear the **Auto-update model** check box at the bottom of the Properties tab.



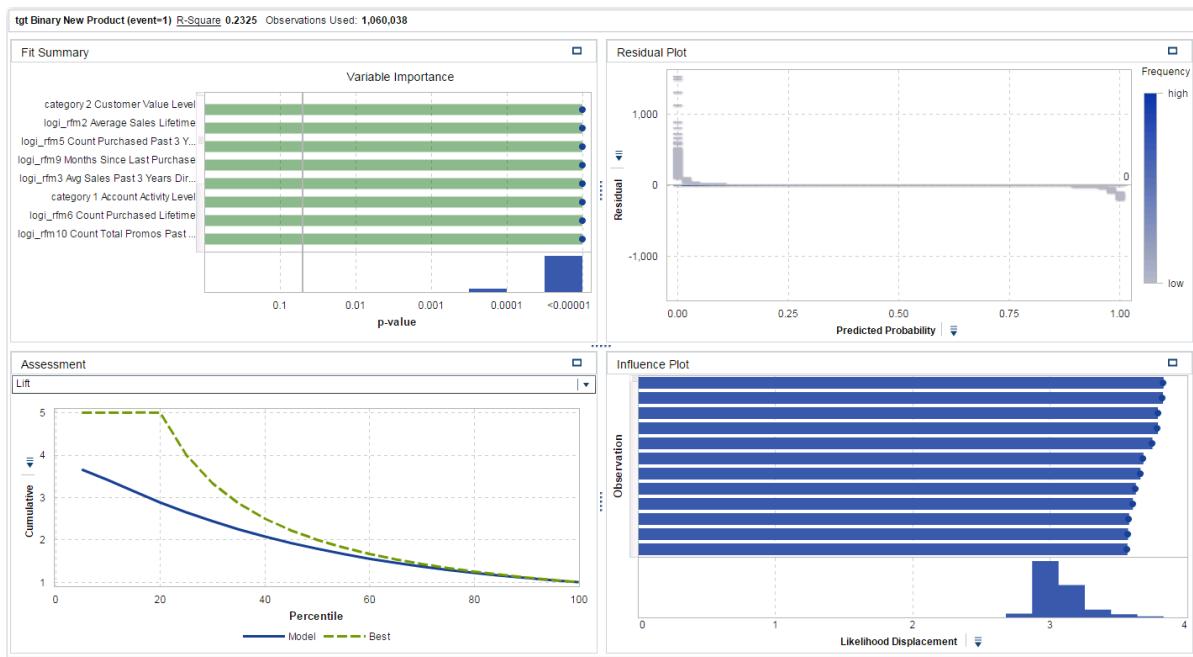
Turning off auto-update enables you to set up several roles in the model before it is created. Otherwise, the model is updated anytime a change is made.

6. Click the **Roles** tab to begin assigning variables to specific roles in the model.
7. From the Category column, drag **tgt Binary New Product** into the work area. Select **Advanced** on the Roles tab. In the Response Event Settings window, select **1** as the event level and click **OK**.

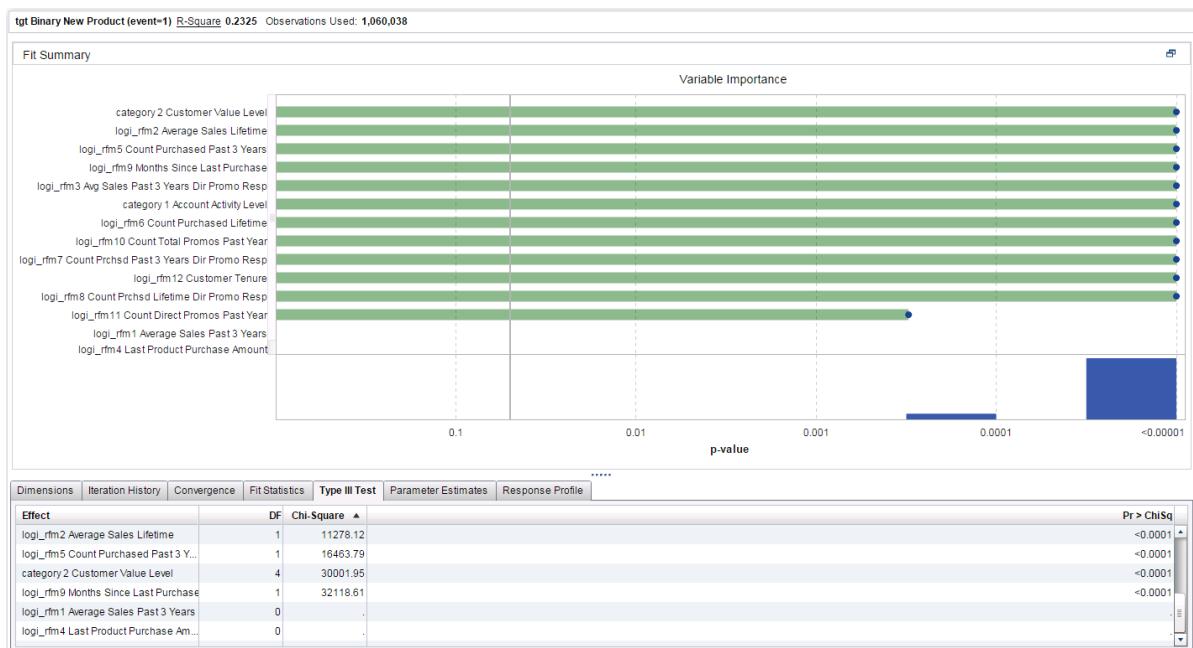
The first categorical variable dropped into the work area defaults to the Response role unless otherwise specified. Under the Advanced options, you can change the response variable to any other categorical variable.

8. From the Category column, click **category 1 Account Activity Level** and **category 2 Customer Value Level** while holding down the Ctrl key. Assign both of these variables to Classification Effects by dragging them into the work area.
9. From the Measure column, hold down the Shift key to select **logi_rfm1 Average Sales Past 3 Years** through **logi_rfm12 Customer Tenure**. Assign all 12 of these variables to Continuous Effects by dragging them into the work area.

10. Create the logistic regression model by selecting either of the **Auto-Update model** check boxes.



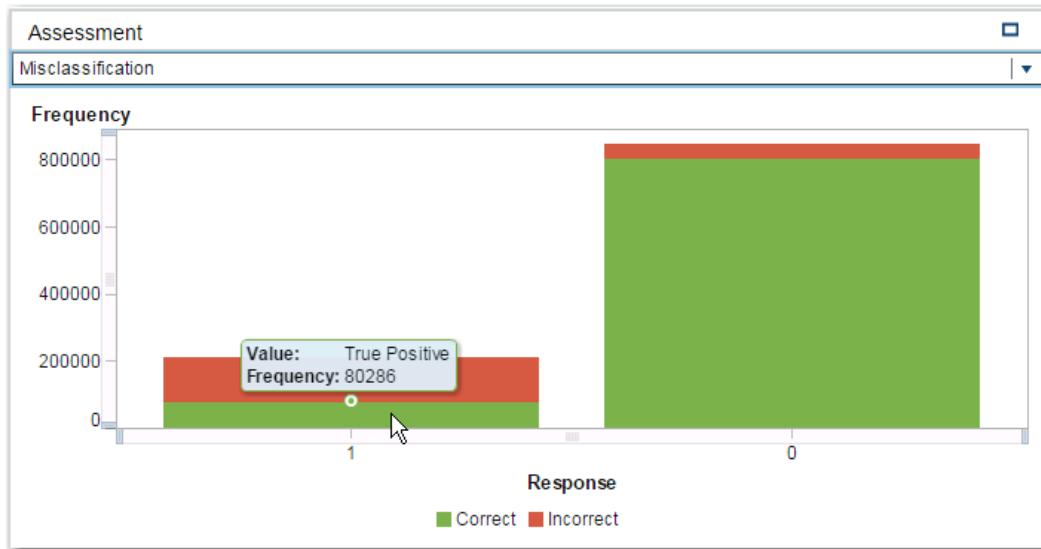
11. Maximize the Fit Summary panel and scroll down until you see the removed variables.
 12. Using the arrow in the top right of the display, open the drop-down menu and select **Show Details**. Click the **Type III Test** tab to verify that both the **logi_rfml Average Sales Past 3 Years** and **logi_rfml4 Last Product Purchase Amount** variables were eliminated during the backward selection.



13. Click the **Response Profile** tab to review the original distribution of the target variable.

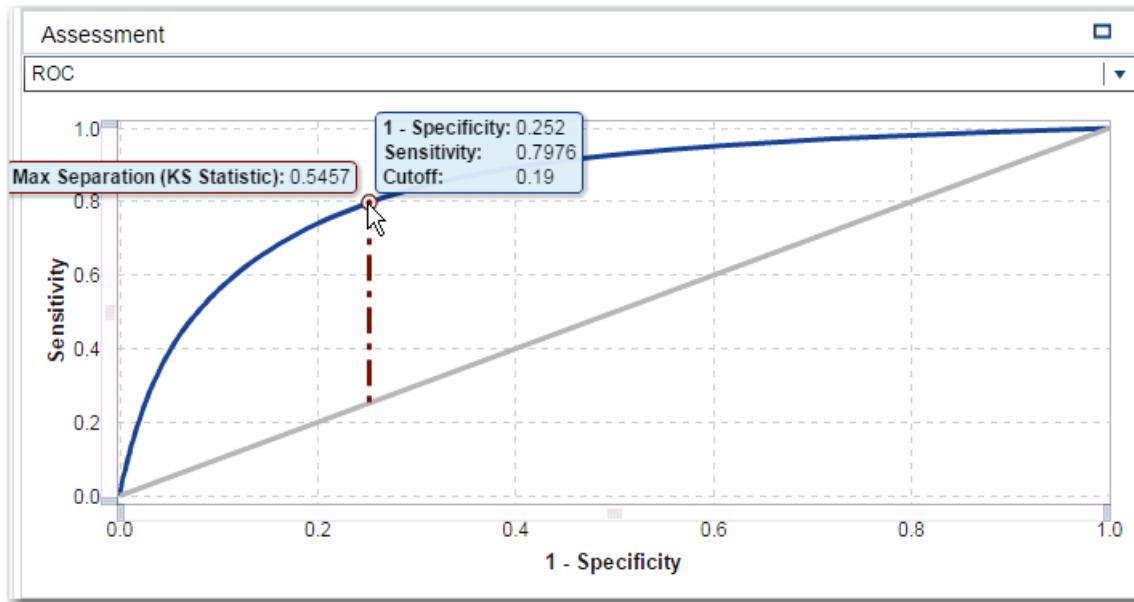
Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile	
Ordered Value	Binary Target					Count	
1	0					848529	
2	1					211509	

14. Clear **Show details** to close the logistic regression details table, and then restore the Fit Summary panel to its original size. Now maximize the Assessment panel and select the misclassification plot. This model does a good job of classifying the nonevent, but it does not do as well on the event itself.



The true positive frequency is 80286 at the default prediction cutoff value of .50.

15. In the Assessment panel, select the ROC chart. Find the tooltip at the very top of the Max Separation line where it intersects the ROC curve. It reveals an optimal cutoff value of 19%.



The threshold indicated above is where the sum of sensitivity and specificity are maximized.

16. Click the **Properties** tab. Change the prediction cutoff to **.19** and press Enter. Check the misclassifications plot to see that the true positive slightly more than doubles to 168691.
 17. Save the exploration as **VS_BankLogistic** in the My Folder location.

End of Demonstration



Exercises

In this exercise, you continue to use the **PVA97NK** data to build a logistic regression model to classify those customers who made a donation.

1. Building a Logistic Regression in SAS Visual Analytics

- a. Return to your remote desktop client machine. If your session timed out, sign in and use the information provided by your instructor.
- b. Start a new Data Explore task by selecting **Data Explore** from the Visual Analytics Hub. Then, select the **PVA97NK** data source.
- c. Click the **Logistic Regression** tool on the toolbar.
- d. If you did not already do so, in the Measure column, right-click **Target Gift Flag** and select **Category** to create a binary target variable for donations.
- e. Clear the **Auto-Update model** check box at the bottom of the Properties tab.
- f. Select **Target Gift Flag** as the response. Because you want to model customers who make donations, select **1** as the event level under **Advanced** on the Roles tab.
- g. From the Category column, add **Gender**, **Home Owner**, and **Status Category 96NK** as classification effects.
- h. From the Measure column, add all 22 variables *except* **Target Gift Amount** as continuous effects. (You add 21 columns.)
- i. On the Properties tab, click the **Use Variable selection** check box and set the significance level to **.05**.
- j. Create the logistic model by selecting either of the **Auto-update model** check boxes.
 - Maximize the **Fit Summary** panel and scroll to the bottom of the variable importance list. How many variables are not included in this model?
 - Are any of the insignificant variables that are not included in this model classification effects?

2. Examining Additional Logistic Regression Results

- a. Open the summary table and click the **Parameter Estimates** tab.
 - 1) Select the **Estimate** column heading to sort the parameter estimates and then determine which parameter had the largest estimate. What was the value?
 - 2) Click the **Response Profile** tab. How many of the customers made donations?
- b. Hide the summary table.

- c. Maximize the **Assessment** panel to gain access to the assessment charts.
 - 1) Examine the lift chart to determine the advantages of using this model for prediction. How does this model compare to the Best model?
 - 2) Select the ROC chart. What are the KS statistic and the cutoff value?
 - 3) What is the prediction cutoff value that is used in the current logistic regression model?
 - 4) Select the misclassification chart to determine whether this model predicts more true positives or more true negatives.
- d. Save your project as **VS Regression Exercise**.

End of Exercises

Interactive Group By Processing

Objectives

- Explain group by processing.
- Describe interface differences with group by processing.
- Discuss advanced group by features.
- Add a group by variable to a logistic regression.

33

Binary Logistic Models: Group By

Credit Scoring: Can credit score and home ownership predict loan default?

Predictor Variables:

Credit Score: 300-850

Home Ownership: Yes/No/Rent



Response Variable:

Loan Default: Yes/No



Group By: Region: N/S/E/W

34

Consider the binary logistic regression model where a bank is attempting to determine whether a customer might default on a loan by examining that customer's credit score and home ownership. By grouping this analysis by demographic region, is it possible that in some regions credit score plays a stronger role in the model than home ownership?

Binary Logistic Models: Group By

Biostatistics: Are alcohol and smoking related to heart disease?

Predictor Variables:

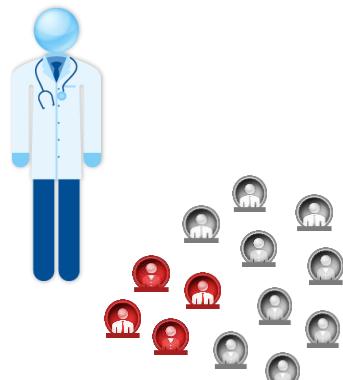
Alcohol: ounces per day

Smoking: cigarettes per day



Response Variable:

Heart Disease: 1/0



35

Interactive Group By Analysis

- Assigning a variable to the Group By role results in a separate analysis for each level of the categorical variable.
- Interactive examination of the fit summary can reveal whether the variable importance for each variable in the model changes significantly from one level to the next.
- Interactive examination of the fit statistics yields model performance information at each level of the categorical variable.



36

Group By Role

- The Group By role fits a model for each data segment defined by one or more categorical variables.
- The Fit Summary panel displays goodness-of-fit statistics and variable importance for each value of the group by variable.
- You can assign a group by variable in one of two ways.
 - Drag and drop one or more category data items to the **Group By** role on the Roles tab.
 - Right-click a category. Select **Assign** ⇒ **Group By**.
- Models are automatically updated when a group by variable is added (unless you clear the Auto-update check box.)
- This role is available for linear, logistic, and GLM models.

37

If you add two or more group by variables, the results are grouped in the order in which the variables appear in the Group By box.

In the Fit Summary panel, selecting a bar in either the goodness-of-fit or variable importance plot updates the other diagnostic plots to display results for that particular Group By model.

Positioning the mouse pointer on any of the plots in the graphs displays additional information about that plot.

The maximum number of BY groups allowed is 1024. Empty data segments count against the maximum number of BY groups allowed in a model.

 The default maximum number of BY groups can be modified in SAS Management Console.

Fit Summary with Group By

The screenshot shows the 'Fit Summary' panel with two main plots. On the left is the 'Goodness of Fit' plot, which displays horizontal bars for different model evaluation criteria (R-Square, -2 Log Likelihood, AIC, AICC, BIC, Max-rescaled R-Square) across four categories (Cluster ID 0, 1, 2, 3). The X-axis ranges from 0.75 to 1.00. On the right is the 'Variable Importance' plot, which shows a scatter plot of p-values for 'category 1 Account Activity Level' across four categories. A green box highlights the text 'Goodness of Fit – Select from the drop-down list.' pointing to the top-left dropdown, and another green box highlights 'Select the model effect from the drop-down list.' pointing to the top-right dropdown.

38

When you use a group by variable in your model, the fit summary shows both a goodness-of-fit plot and a variable importance plot.

Goodness of fit measures the model fit for each variable. The X axis displays the values for the model evaluation criteria that is selected, and the Y axis displays the values of the group by variable. You can change the model evaluation criteria by selecting one from the drop-down list at the top of the Goodness of Fit panel. These are the choices:

- R-Square
- -2 Log Likelihood
- AIC
- AICC
- BIC
- Max-rescaled R-Square

Position the mouse pointer on the horizontal bars in the Goodness of Fit panel to display the group by value, the model evaluation criteria, and the number of observations. Position the mouse pointer on the histogram bars at the bottom of the Goodness of Fit panel to display the range of values represented by the bar for the selected model evaluation criteria and the percentage of observations included in the bar.

In the Goodness of Fit panel, selecting and highlighting a group by value (clicking on a horizontal bar) updates the Residual Plot panel and the Influence Plot panel to display only the observations in that segment.

When you assign a group by variable, the Variable Importance panel displays the p -values for a selected model effect for each value of the group by variable. You see what equates to a linear scatter plot of the p -values. You can position the mouse pointer on the points to display the BY-group value, the effect name, the p -value, and the $-\log(p\text{-value})$.

Position the mouse pointer on the histogram bars at the bottom of the Variable Importance panel to display the range of all p -values represented by the bar for the selected model effect and the percentage of observations included in the bar.

You can sort the order of the variables by either goodness of fit or variable importance by clicking the icon in the top right of the panel.

Advanced Group By Features

Advanced features enable you to fine-tune the group by variables.

Select **Advanced** next to the Group By role in the right pane.

- **Group by** – Choose one or more variables.
- Select **Use advanced features**.
- **Measure** – Specify the variable to use in the aggregation.
- **Aggregation** – Average or Sum
- **Count** – Top or Bottom (100 is the default.)

The results are displayed at the bottom.



39

The Advanced Group By window enables you to select one or more group by variables and then select an aggregated measure on which to rank the results. After you select one or more group by variables, select a measure, aggregation type (Average or Sum), and a count (either Top or Bottom) as well as a count value. (The default is 100.) The model results are then limited to show only the ranked (Top or Bottom) *n* segments.

For example, suppose the chosen group by variable has 30 possible values. If you also select a measure using the advanced features and set that measure to be **Average** and **Top 10**, then the model results show only the top 10 group by values based on the chosen average aggregated measure.

Advanced Group By Example

Name	Value
3	19.855496269
2	17.557108737
0	17.334174245
1	16.831548103

40

In the display above, **Target Gift Flag** is the response variable, and several variables are assigned as continuous effects. An advanced group by was entered using **Demographic Cluster** as the group by variable, and **Median Income Region** was used as the measure. **Average** and **Top 10** were entered as the parameters for **Median Income Region**. In the resulting Fit Summary panel, only the top 10 demographic clusters based on the average of **Median Income Region** are shown.

3.01 Multiple Choice Poll

Which of the following are true regarding measure variables?

- A measure variable must be a cluster or group by variable.
- Measure variables can be used to rank-order the clusters.
- All clusters must be displayed in an advanced group by.

41

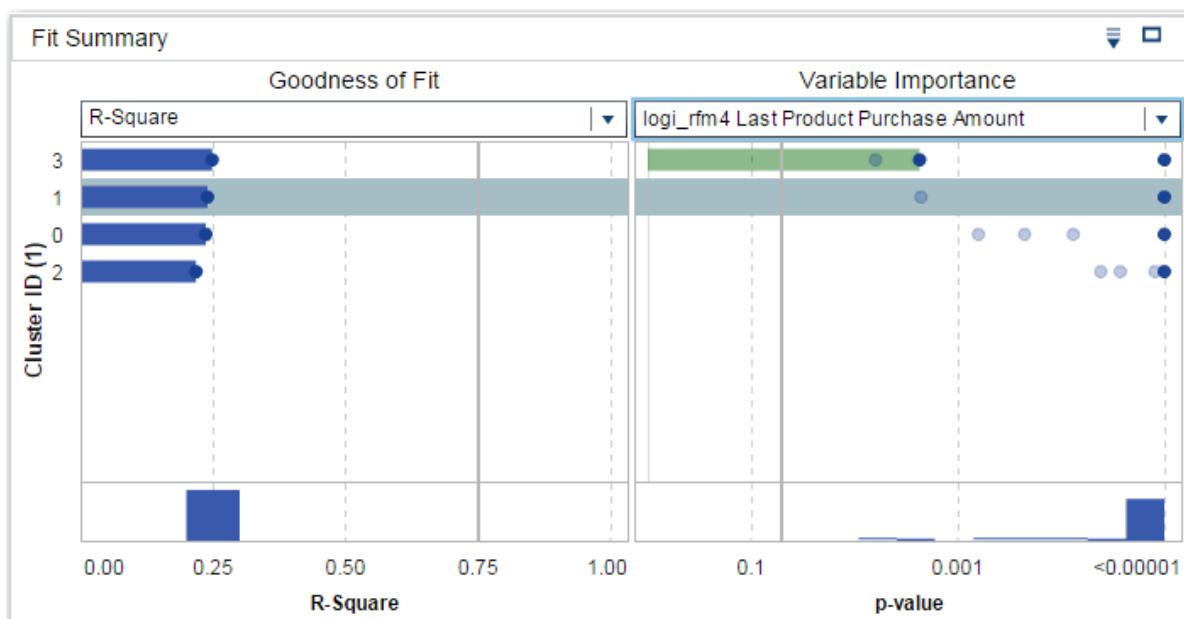


Adding a Group By Variable to a Logistic Regression

This demonstration illustrates how to interactively group a logistic regression model by a variable.

1. Open the **VS_BankLogistic** exploration from the My Folder location, if it is not already open.
2. Click the **Roles** tab if it is not selected and assign the **Cluster ID (1)** variable to the Group By role.
3. Maximize the Fit Summary panel. Select **logi_rfm4 Last Product Purchase Amount** under Variable Importance. Notice that it is significant only to the cluster 3 BY group.

This might suggest that the amount spent on the last product that a customer purchases can be useful in only those models with a particular demographic. Examine the cluster created previously to obtain the attributes of this group of customers.

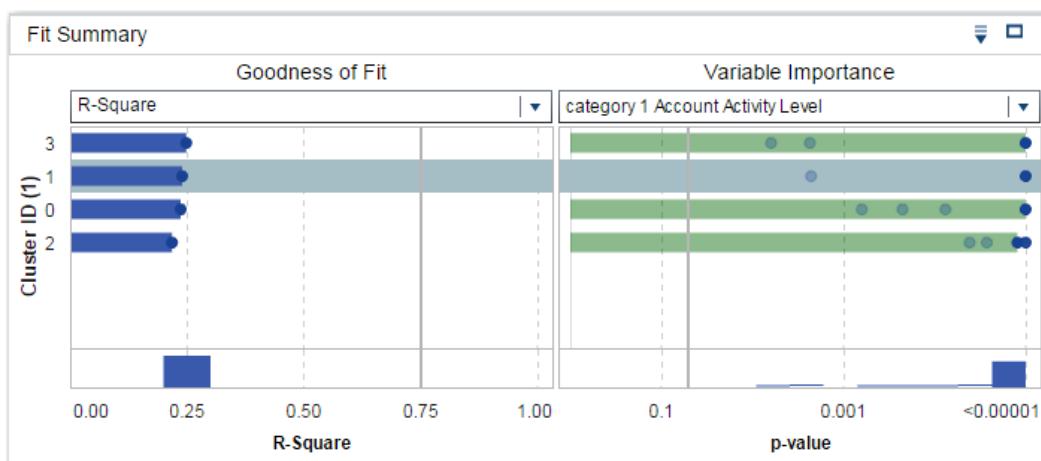


With group by processing, the Fit Summary panel enables you to visually assess quickly which variables are important to a BY group. Select the BY group first on the left side of the panel. Then select a variable of interest on the right side of the panel. The green bars indicate which BY groups are of significance to the selected variable.

4. Select cluster **3** in the Fit Summary window. Then click the arrow at the top right of the Visualization toolbar and select **Show Details**. Click the **Parameter Estimates** tab. Select the **Estimate** column heading to sort the column. Examine the estimates to verify that there were four terms dropped for the logistic regression model built for this BY group: **category 2 Customer Value Level E, logi_rfm8 Count Prchsd Lifetime Dir Promo Resp, logi_rfm10 Count Total Promos Past Year, and category 1 Account Activity Level Z**.

Parameter	Estimate ▲	Standard Error	z Value	Pr > z
logi_rf9 Months Since Last Purchase	-1.48739	0.027367	-54.3505	<0.0001
logi_rf2 Average Sales Lifetime	-1.25219	0.036046	-34.7383	<0.0001
logi_rf11 Count Direct Promos Past Year	-0.46559	0.039365	-11.8277	<0.0001
category 2 Customer Value Level D	-0.21998	0.059371	-3.70517	0.0002
logi_rf3 Avg Sales Past 3 Years Dir Promo Resp	-0.18005	0.030522	-5.89888	<0.0001
logi_rf7 Count Purchsd Past 3 Years Dir Promo Resp	-0.13256	0.020221	-6.55583	<0.0001
logi_rf12 Customer Tenure	-0.0822	0.03005	-2.73538	0.0062
logi_rf8 Count Purchsd Lifetime Dir Promo Resp	0	.	.	.
category 2 Customer Value Level E	0	.	.	.
logi_rf10 Count Total Promos Past Year	0	.	.	.
category 1 Account Activity Level Z	0	.	.	.
category 2 Customer Value Level C	0.008442	0.054174	0.155827	0.8762
logi_rf4 Last Product Purchase Amount	0.071308	0.023428	3.043738	0.0023
category 2 Customer Value Level B	0.073868	0.048461	1.524292	0.1274
logi_rf6 Count Purchased Lifetime	0.177374	0.025634	6.91952	<0.0001
logi_rf1 Average Sales Past 3 Years	0.261168	0.033776	7.73236	<0.0001
category 1 Account Activity Level X	0.274569	0.030991	8.859597	<0.0001
category 1 Account Activity Level Y	0.455751	0.039288	11.60028	<0.0001
category 2 Customer Value Level A	0.474	0.047515	9.975772	<0.0001
logi_rf5 Count Purchased Past 3 Years	1.883665	0.031078	60.61181	<0.0001
Intercept	4.627403	0.145083	31.89496	<0.0001

- Select cluster **1** in the Fit Summary panel and notice that a new logistic regression model is created for this BY group. Examine the Parameter Estimates tab in the summary table to verify now that **logi_rf4** is no longer part of the model, but **logi_rf10** is.
- Select **category 1 Account Activity Level** in the Fit Summary panel under Variable Importance. Verify that the **Account Activity Level** is not important to this cluster, but it is to all others.



This lack of significance for **Account Activity Level** can be corroborated by examining the 0 estimates on the Parameter Estimates tab of the summary table.

- Close the current exploration and do **not** save it.

End of Demonstration

Decision Trees

Objectives

- Describe decision tree variable roles in SAS Visual Analytics.
- Describe decision tree properties in SAS Visual Analytics.
- Cultivate a decision tree.
- Assess decision tree performance.

46

Decision Trees in SAS Visual Analytics

- There is only one **response** variable. It can be either a category or a measure.
- There can be multiple **predictor** variables.
- Both categorical and interval valued predictors are accommodated. (No interaction terms are allowed.)
- You can manually train and prune a decision tree using interactive mode.
- You can derive a leaf ID. This ID can be used in other models featured in the SAS Visual Statistics functionality.

47

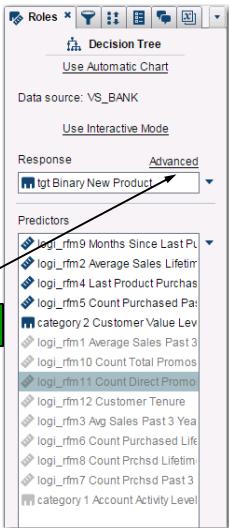
The decision tree in SAS Visual Analytics uses a modified version of the C4.5 algorithm.



One difference between trees and other modeling algorithms presented in this course is that decision trees are available in SAS Visual Analytics without the SAS Visual Statistics add-on. However, the addition of SAS Visual Statistics does augment the decision tree functionality. Further, some decision tree default settings are modified with the Visual Statistics add-on. An excellent summary of additional and modified tree functionality associated with the addition of Visual Statistics is found in “Decision Trees in VA/VS 7.2 – What’s the Deal?” (<http://www.sas.com/blogs/wp/gate/4419/decision-trees-in-vavs-7-2-whats-the-deal/sasmrs/2015/03/20/>).

Decision Tree Roles

- Response – assign only one response or target variable
- Predictors – assign any number of categorical and interval variables



Choose event level.

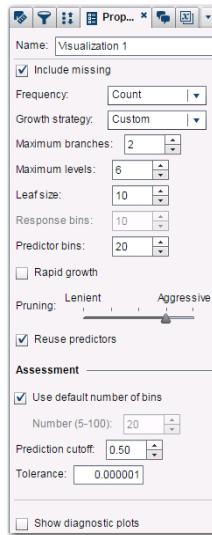
48

Categorical- and interval-valued response or target variables are accommodated in the SAS Visual Statistics decision tree model. Although multilevel categorical target variables are allowed, one level is chosen as the event level, and other levels are combined into the non-event category.

For binary target variables, changing the event level does not affect the hierarchical structure of the decision tree. However, it does change the assessment plots (lift, ROC, and misclassification) that are generated for each event level. In order to do model comparisons (for example, between a logistic regression and a decision tree), you need to make sure that your models are targeting the same outcome.

Decision Tree Properties

- Name of model
- Maximum branches
- Maximum levels
- Leaf size
- Response bins
- Predictor bins
- Pruning
- Rapid growth
- Include missing
- Reuse predictors
- Leaf statistics
- Assessment
 - Use default number of bins
 - Number of bins
 - Prediction cutoff
 - Tolerance
- Show diagnostic plots



49

Name enables you to specify the name for this model.

Maximum branches specifies the maximum number of branches that are allowed when you split a node. The default is 2, and the maximum is 10.

Maximum levels specifies the maximum depth of the decision tree. The default is 6, and the maximum is 20.

Leaf size specifies the minimum number of observations that are allowed in a leaf node. The default is 10.

Response bins specifies the number of bins that are used to categorize a continuous response variable. The default is 10, and the maximum is 100.

Predictor bins specifies the number of bins that are used to categorize a predictor that is a continuous variable. The default is 20.

Rapid growth enables you to use the information gain ratio and k -means fast search methods for decision tree growth. When **Rapid growth** is disabled, the information gain and greedy search methods are used. They generally produce a larger tree and require more time to create.

 Selecting **Rapid growth** creates a decision tree most similar to a decision tree created in SAS Visual Analytics Explorer. This assumes that all other properties are set the same way as when in Expert mode in the Explorer.

Pruning specifies the aggressiveness of the tree-pruning algorithm. A more aggressive setting creates a smaller decision tree. The default setting is on the fourth tick mark.

 In SAS Visual Statistics, pruning is based on the C4.5 algorithm and the assumption that the error rate follows the BETA distribution. This method uses a confidence bound constructed on the estimated error rate of the tree over a given subset of training cases to prune. The upper confidence bound of the estimated error rate is used to approximate the true error rate. The “aggressiveness” of the algorithm corresponds to the alpha parameter chosen to construct the confidence interval. See *SAS® LASR Analytic Server Reference Guide* for additional details.

Include missing enables you to include observations with missing values. For category variables, a missing value is assigned to its own level. For continuous variables, a missing value is assigned to the smallest available machine value (negative infinity).

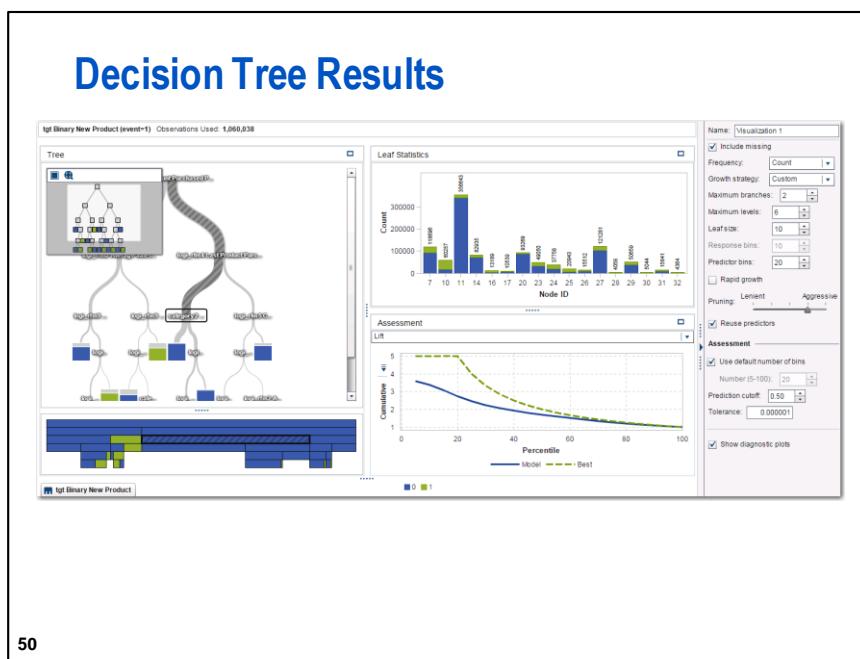
Reuse predictors allows more than one split in the same branch based on a predictor.

Assessment

- **Use default number of bins** specifies whether you want to use the default number of bins or to set your own value.
- **Number** specifies the number of bins to use when the **Use default number of bins** property is not selected. You must specify an integer value between 5 and 100.
- **Prediction cutoff** specifies the value at which a computed probability is considered an event.
- **Tolerance** specifies the tolerance value that is used to determine the convergence of the iterative algorithm that estimates the percentiles. Specify a smaller value to increase the algorithmic precision.

Show diagnostic plots specifies whether the Leaf Statistics and Assessment windows appear in the Model pane.

The **Show tree overview** button is located at the top right of the tree diagram window. It displays the tree overview. The tree overview enables quick navigation of large decision trees. When you zoom in to view a specific area of the decision tree, the tree overview shows the entire decision tree and highlights the area that you are viewing. You can click and drag the highlighted area to change the display of the decision tree. Click the **Zoom to Fit** icon in the upper left corner of the tree overview to view the entire decision tree. You can alternately click the **Minimize Overview/Maximize Overview** icon in the upper left corner of the tree overview to minimize or maximize the tree overview.



This decision tree is trying to predict what factors cause people to respond or purchase a new product.

3.02 Multiple Choice Poll

Which of the following is false regarding decision trees in Visual Statistics?

- a. An input variable can appear in only one split of the tree, by default.
- b. Probabilities can be derived from the proportion of responders in each (terminal) leaf.
- c. Trees are relatively easy to interpret and explain.

51

Analyzing Decision Tree Results

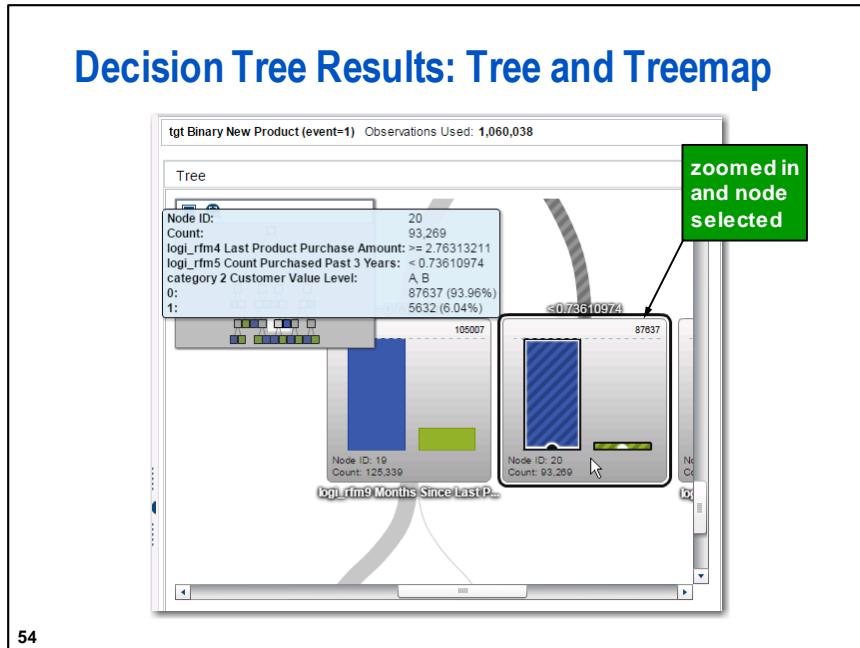
Three panels are displayed to help you analyze the results of the decision tree model.

- Tree with Treemap
- Leaf Statistics
- Assessment
 - Lift measures model effectiveness.
 - ROC (receiver operating characteristic) measures classification accuracy.
 - Misclassification measures predictive accuracy.

53

Tree with Treemap is the main panel that is displayed by default when you build a decision tree model. The Leaf Statistics and Assessment panels are not enabled by default. To display them, click the **Properties** tab and select the **Show diagnostic plots** check box.

When it is enabled, the Assessment panel enables you to choose between a lift chart, ROC chart, and misclassification chart. These work the same way they do when you work with a logistic regression model. The only exception to that is when you use a continuous response variable and set the response bins to more than 10. (This particular situation is addressed later in this chapter.)



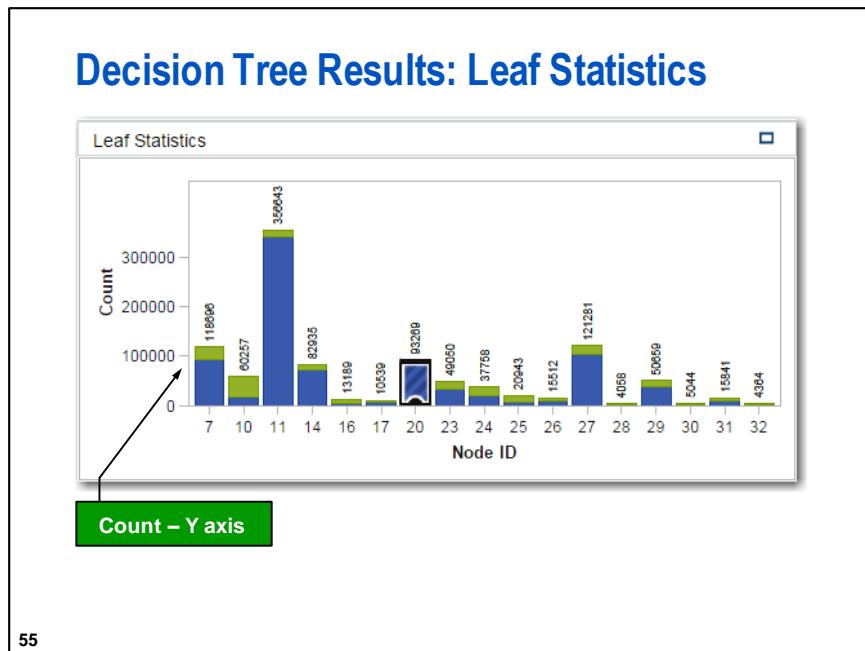
To navigate the decision tree, you can use the mouse and keyboard. Hold down the Shift key and click anywhere in the Tree window to move the decision tree within the window. Use your mouse's scroll wheel to zoom in and out of the decision tree. Scroll up to zoom in, and scroll down to zoom out. The zoom is centered on the position of the mouse pointer.

The color of the node in the treemap indicates the predicted level for that node. It is indicative of the event level that has the most observations in the node.

When you select a node in either the decision tree or the treemap, the corresponding node is selected in the other location. When you select a leaf node, that node is selected in the Leaf Statistics window. A legend is available at the bottom of the Model pane. When the response variable is a continuous variable, a gradient is used to denote the predicted bin. Darker colors represent larger values.

Right-click outside of a node in the Tree window to display a pop-up menu. The first item in this menu is **Derive a Leaf ID Variable**. When you click this item, SAS Visual Statistics creates a category variable that contains the leaf ID for each observation. You can use this variable as an effect in other models. These are the other options:

- **Show branch coloring**
- **Show diagnostic plots**
- **Show tree overview**
- **Leaf Statistics** \Rightarrow **Count or Percent**



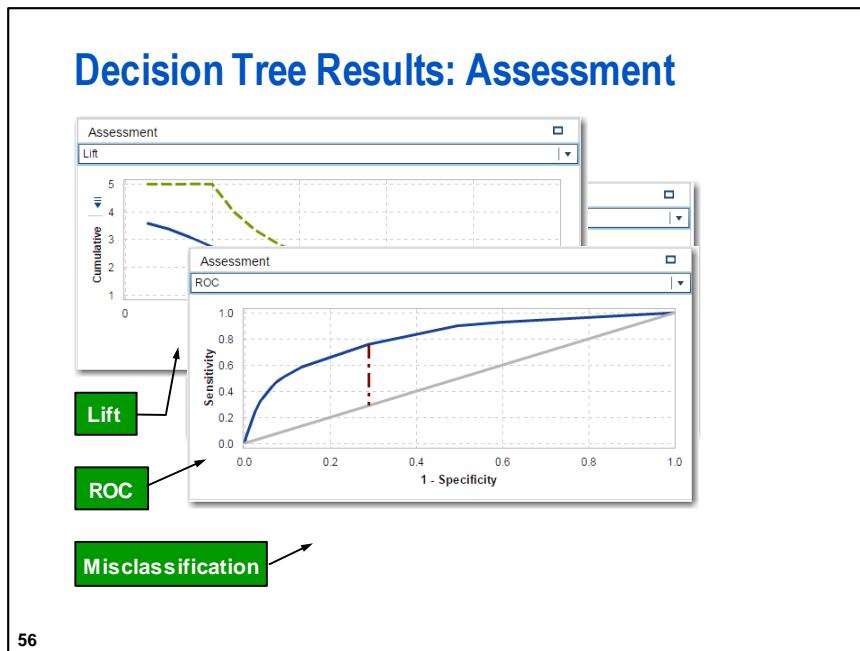
The Leaf Statistics window plots the count of observations on the Yaxis and the node ID of each leaf node on the X axis.

Click the **Properties** tab and change **Count** to **Percent** for the **Leaf statistics** property if you want the percent to appear on the Y axis.

The most common level in a node is the predicted value assigned to that node. Leaf nodes that contain approximately equal amounts of more than one level might benefit from additional training.

When you select a column in the Leaf Statistics window, the corresponding leaf is selected in the Tree window.

When you evaluate the results of a decision tree with a lift chart, where the lift chart begins to decrease indicates that the next leaf or group of leaves is significantly less powerful in predicting the response.



Decision Tree Summary Table: Node Statistics

The Node Statistics tab provides summary statistics for each node in the decision tree.

Node Statistics		Node Rules		Type	Observations	% Observations	Gain	Predicted Value	Cancer	Cerebral Vas...
Node ID	Depth	Parent ID	N Children							
0	0	-1	2	Class	1918	100.00%	0.0230933649...	Coronary Heart...	521 (27.16%)	363 (18.93%)
1	1	0	2	Class	786	40.98%	0.0145175478...	Coronary Heart...	155 (19.72%)	192 (24.43%)
2	1	0	2	Class	1132	59.02%	0.0201939254...	Cancer	368 (32.33%)	171 (15.11%)
3	2	1	2	Class	36	1.88%	0.2380331436...	Cerebral Vascul...	3 (8.33%)	18 (50.00%)
4	2	1	2	Class	750	39.10%	0.0151500536...	Coronary Heart...	152 (20.27%)	174 (23.20%)
5	2	2	2	Class	728	37.96%	0.0187648780...	Coronary Heart...	209 (28.71%)	112 (15.38%)
6	2	2	2	Class	404	21.06%	0.0309955317...	Cancer	157 (38.86%)	59 (14.60%)
7	3	3	0	Leaf	13	0.68%	0	Coronary Heart...	3 (23.08%)	

57

This table displays a column for each value of the response that contains both the number and percentage of observations in each node. (This is shown in the display.)

Decision Tree Summary Table: Node Rules

The Node Rules tab provides the sorting rule used for each node in the decision tree.

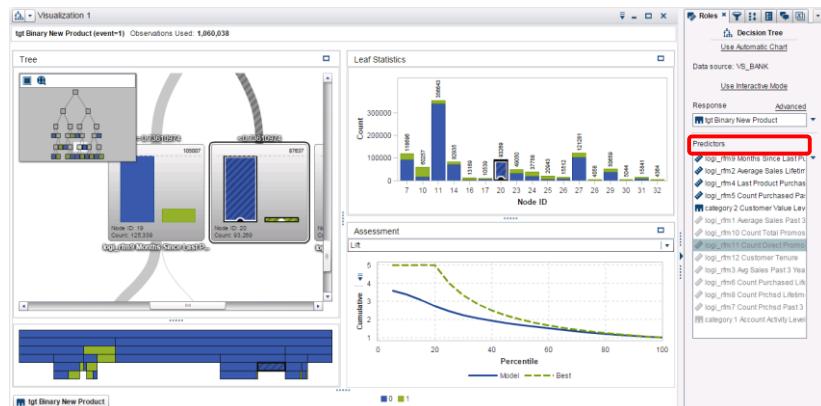
- Node ID
- Parent ID
- Type (Class or Leaf)
- Column for each predictor and rule that is applied

Node Statistics		Node Rules					
Node ID	Parent ID	Type	Cholesterol	Diastolic	Weight	Systolic	Smoking
0	-1	Class				≥ 147.4	
1	0	Class				< 147.4	
2	0	Class				≥ 223.7	
3	1	Class				$147.4 - 223.7$	
4	1	Class					
5	2	Class	≥ 214			< 147.4	
6	2	Class	< 214			≤ 147.4	
7	3	Leaf			≥ 183.5	≥ 223.7	
8	3	Leaf			< 183.5	≥ 223.7	

58

If a rule was applied for a predictor variable in a node or any of its parent nodes, then it is listed in the table. Otherwise, the entry is blank.

Decision Tree Interactive Mode



59

The decision tree enables you to manually train and prune a decision tree by entering interactive mode. In interactive mode, you are unable to modify the response variable or predictors that are being used.

 You cannot export model score code.

To enter interactive mode, you can start making changes to the decision tree in the Tree window (by right-clicking the node) or you can click **Use Interactive Mode** on the Roles tab in the right pane. To leave interactive mode, click **Use Non-Interactive Mode** on the Roles tab. A warning message is displayed. It indicates that your changes are lost if you exit interactive mode. Click **Yes** or **No** to continue.



When you leave interactive mode, you lose all of your changes.

Right-click a node to display the pop-up menu that enables you to interactively make changes to the tree. Your choice of options in the pop-up menu depends on whether the node is a leaf node.

For leaf nodes, you can select from the following menu options:

- **Split** displays the Split Decision Tree window. Use this window to select the variable that is used to split the node. Click **OK** to split the node based on the selected variable. Click **Cancel** to not split the node. Variables are sorted in descending order by their log worth.
- **Split Best** splits the node based on the variable with the best information gain ratio.
- **Train** displays the Train Decision Tree window. Use this window to train more than one level beyond the leaf node. First, select every variable that you want to be available for training. Only those variables selected in the Train Decision Tree window are available for training. Specify the maximum depth of training in the **Maximum depth of subtree** property. Click **OK** to train the decision tree.

For non-leaf nodes, select **Prune** to remove all nodes that follow the selected node. This changes the selected node into a leaf node. After pruning a node, you can select **Restore** to undo the prune.

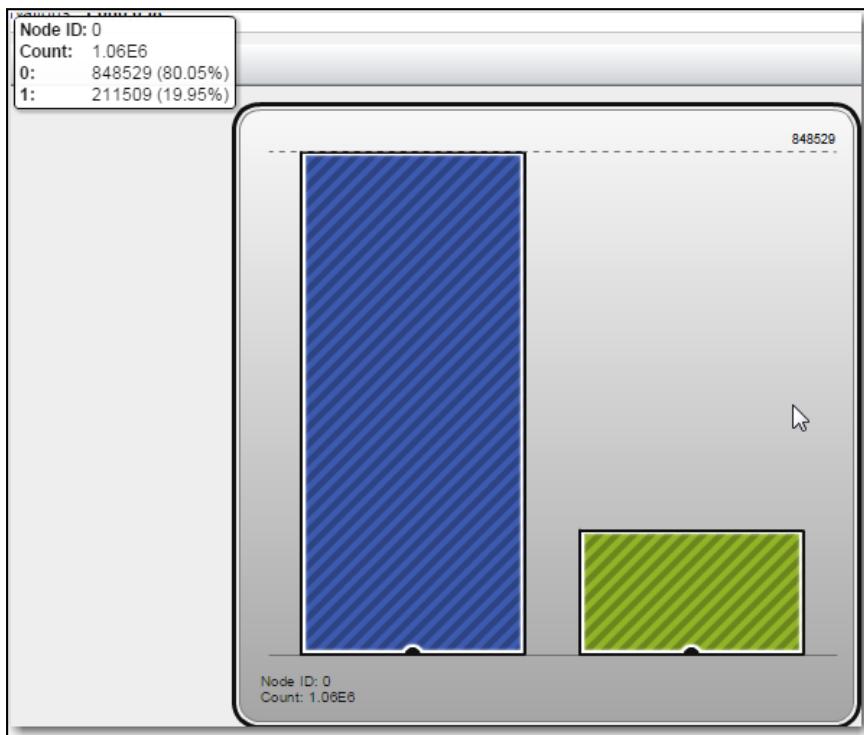


Creating and Cultivating a Decision Tree in SAS Visual Analytics

This demonstration illustrates how to create a decision tree in SAS Visual Analytics and cultivate the tree autonomously.

Creating a Decision Tree Analysis in SAS Visual Analytics

1. Open the **VS_BankLogistic exploration** from the My Folder location.
2. Select **New Visualization**, and click the **Decision Tree** button. The dependent variable for analysis is the **tgt Binary New Product** field in the **VS_BANK** data set.
3. Recall that the binary target is a numeric (1/0) flag that indicates response or non-response. Make sure that its measurement level is **Category**.
4. Click the **Roles** tab on the right side of the Decision Tree window. Select **tgt Binary New Target** as the response variable.

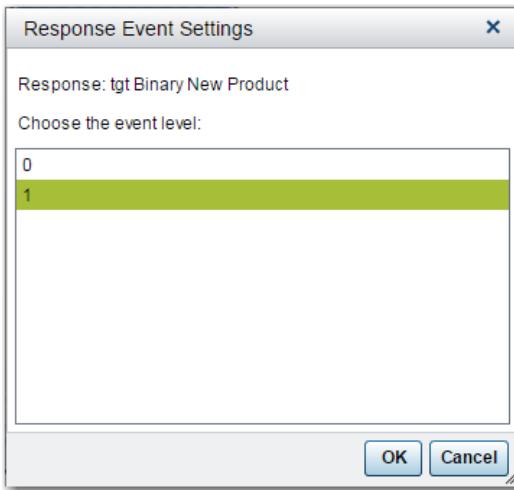


The distribution of the values in **Binary Target** is shown in the middle window. About 20% of the data consists of 1s (responders). This is the root node of the decision tree.



For **Binary Target**, the number of observations is more than a million and the event target level (event of interest) to be modeled is 0. The event of interest is changed to 1 to be consistent with subsequent models.

5. Select the **Advanced** property next to **Response**. Change the event level from **0** to **1**, and click **OK**.



Cultivating a Decision Tree Autonomously

1. Clear the **Auto-update model** check box at the bottom of the Decision Tree Properties portion of the window.



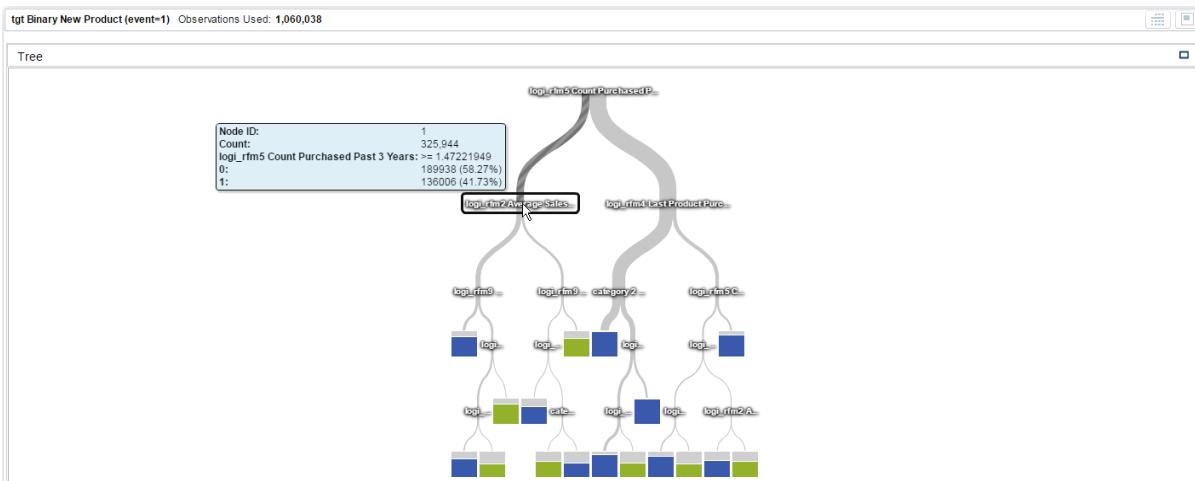
2. Select the input variables.

Select **category 1 Account Activity Level** in the data portion of the Decision Tree window. Press and hold the Ctrl key, and then select **category2 Customer Value Level** and all of the **logi_RFIM** variables. Drag and drop the selected variables onto the **Predictors** role.

3. Click **Update**.

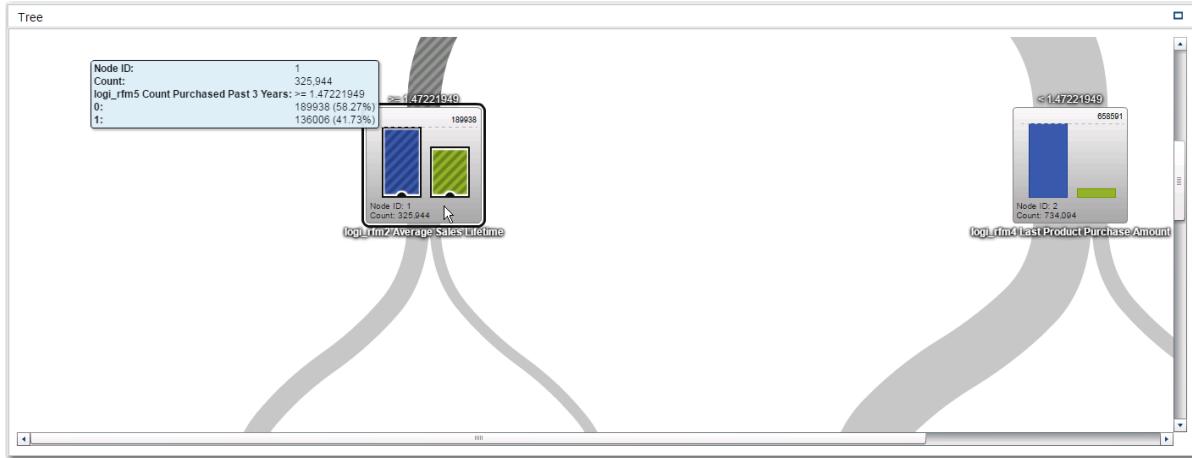


4. Click the left node after the first split of the top (root) node of the decision tree.



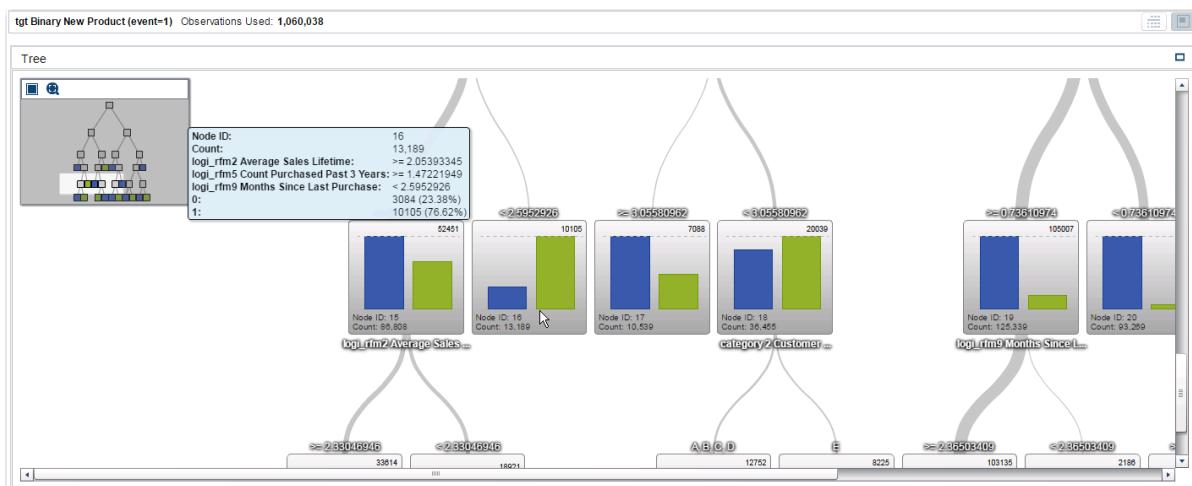
5. Use the wheel on your mouse to zoom in and view the characteristics of observations in this partition of the data.

 The Zoom functionality centers the diagram on your cursor. A good tip is to position the mouse pointer on the part of the tree that you are interested in investigating before zooming.



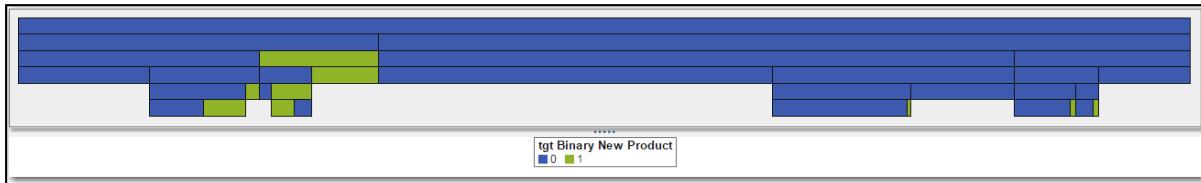
The split point is the number on top of the node. Cases in this partition of the data have a value of **logi_rf5 Count Purchased Past 3 Years** that is greater or equal to 1.472. There are 325,944 cases in this node, and 136,006 of them are responders. The proportion of responders is approximately 42%. Recall that the proportion of responders in the training data is approximately 20%.

6. Select the **Tree Overview** button at the top right of the tree diagram. Then navigate to and select the terminal leaf shown below.



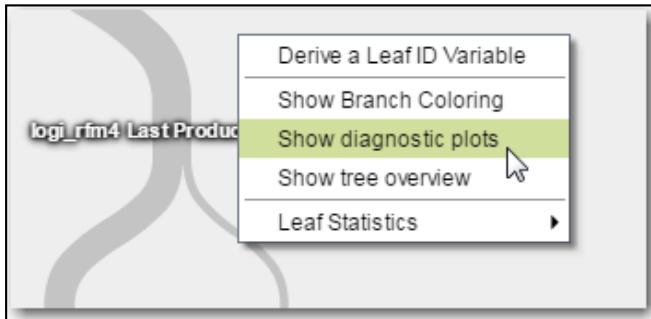
The listed predictors and corresponding split points summarize the characteristics of the 13,189 cases in the node. The node contains approximately 77% responders. That is, according to this tree, cases with predictors in the ranges listed by the input split points have a probability of response equal to 77%.

7. Scroll down to the icicle plot.

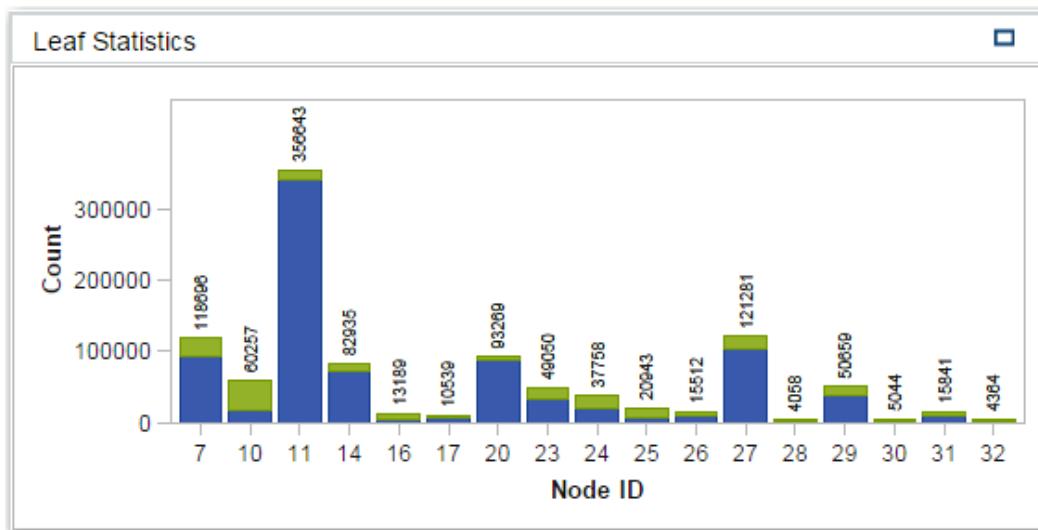


This plot summarizes the leaves of the decision tree. The area of each rectangle corresponds to the count of cases in each leaf. The color of each rectangle indicates the majority level of the binary target in each leaf.

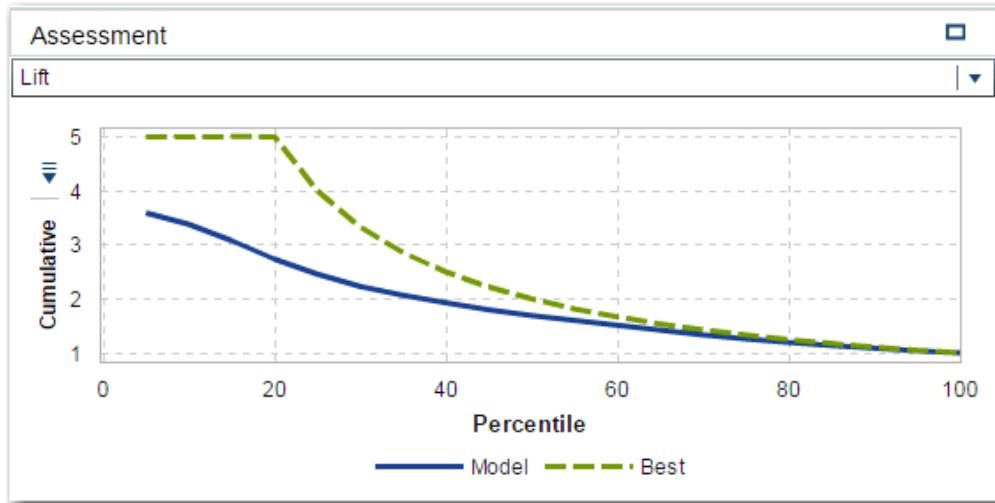
- Right-click in the (non-Tree) field of the Tree diagram and select **Show diagnostic plots**.



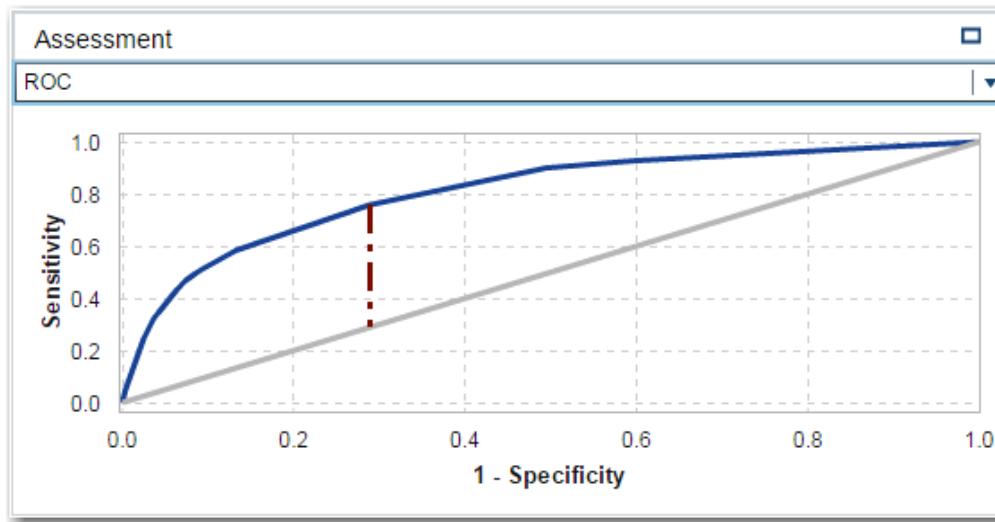
- The Leaf Statistics panel displays a bar chart of simple statistics for each leaf. This enables you to do a quick visual comparison of leaf size. Also, selecting a leaf in this panel highlights it in the other panels and tables. The majority of customers are in Node 11, and they are non-responders.



10. The lift plot summarizes the tree model's ability to rank-order cases (blue line) relative to a naïve model (a horizontal line with an intercept of 1) and a best model (green line). For example, when rank-ordered by the model, the top 10% (percentile) of the data has about 3.4 times as many responders as a random ordering of the data, and 1.6 times fewer responders than a perfect ordering of the training data.



11. Change the assessment plot to **ROC**. The ROC chart summarizes the true positive rate (Sensitivity) and false negative rate ($1 - \text{Specificity}$) across thresholds or cutoffs in the data. The 45-degree line represents the performance of the naïve model, and the vertical red line corresponds to an optimal threshold in the data.



12. Save this project as **VS_BankDecision** in the My Folder location.

The Export Image and Export Data (for example, CSV) options are useful for disseminating work done in your explorations.

End of Demonstration



Exercises

In this exercise, you continue to use the **PVA97NK** data to build a decision tree to classify those customers who made a donation.

3. Building a Decision Tree in SAS Visual Analytics

- a. Return to your remote desktop client machine. If your session timed out, use the information provided by your instructor to sign in.
- b. From the SAS Visual Analytics Hub, start a new data exploration, and select the **PVA97NK** data source.
- c. Click the **Decision Tree** tool on the toolbar.
- d. If you did not already do so, in the Measure column, right-click **Target Gift Flag** and select **Category** to create a binary target variable for donations.
- e. Select **Target Gift Flag** as the response. Because you want to model customers who make donations, select **1** as the event level under **Advanced** on the Roles tab.
 - How many customers made donations?
 - What proportion of individuals does that represent in the target node?
- f. Clear the **Auto-Update model** check box at the bottom of the Properties tab.
- g. From the Category column, add **Gender**, **Home Owner**, and **Status Category 96NK** as predictors.
- h. From the Measure column, add all 22 variables, *except* **Target Gift Amount**, as predictors. (You add 21 columns.)
- i. Create the decision tree by selecting either of the **Auto-update model** check boxes.

Zoom in toward the root node. (You can use either the wheel on your mouse or right-click in the tree background and select **Show tree overview**.)

- On what column does the top split occur?
- What is the split point that determines to which branch a customer belongs?
- In which branch do the majority of customers fall at this split point?
- How many customers were less than this value and belong to Node 2?
- What proportion of the customers in Node 2 made donations?

4. Examining Additional Decision Tree Results

- a. Open the summary table to examine the node statistics.
 - 1) Examine the last column to see whether there are any 100% donator nodes. If so, which node or nodes?
 - 2) Click the **Node Rules** tab. Is Node 29 a class node or a leaf node?
 - 3) Use the node rules to describe the customers in Node 29 in laymen's terms.
- b. Hide the summary table.
- c. Right-click in the tree background to open the diagnostic plots.
 - 1) Maximize the **Leaf Statistics** panel. Which node has the most customers?

- 2) Select the node with the most customers and examine either the tree or the summary table. What is the proportion of donors?
 - 3) Maximize the **Assessment** panel and examine the lift chart. What can you determine about the top 10% (percentile) of the data?
- d. Save your project as **VS Tree Exercise**.

End of Exercises

Tree Split Search in SAS Visual Statistics (Self-Study)

Objectives

- Describe how decision trees are split in SAS Visual Statistics.
- Describe how predictions are formulated for a decision tree.
- Explain variable selection methods for decision trees.
- Describe the tree variable selection methods that are available in SAS Visual Statistics.

63

As seen in the previous section, regressions, as parametric models, assume a specific association structure between inputs and target. By contrast, decision trees, as predictive algorithms, do not assume any association structure. They simply seek to isolate concentrations of cases with like-valued target measurements.

Decision trees are similar to other modeling methods described in this course. Cases are scored using *prediction rules*. A *split-search* algorithm facilitates predictor variable selection.

Useful predictions depend, in part, on a well-formulated model. Good formulation primarily consists of preventing the inclusion of redundant and irrelevant predictors (input variables) in the model. The predictor variable selection function is complicated with large data. There are usually many predictors to consider and several pieces of information (rows) about these columns to process. This complication adds to the requirements of the input search method for any given model. The method must eradicate redundancies and irrelevancies, and also be extremely efficient. The input search methods that are available in the decision tree algorithm in Visual Statistics are described in this section.

The simple prediction problem shown below illustrates each of these model essentials.

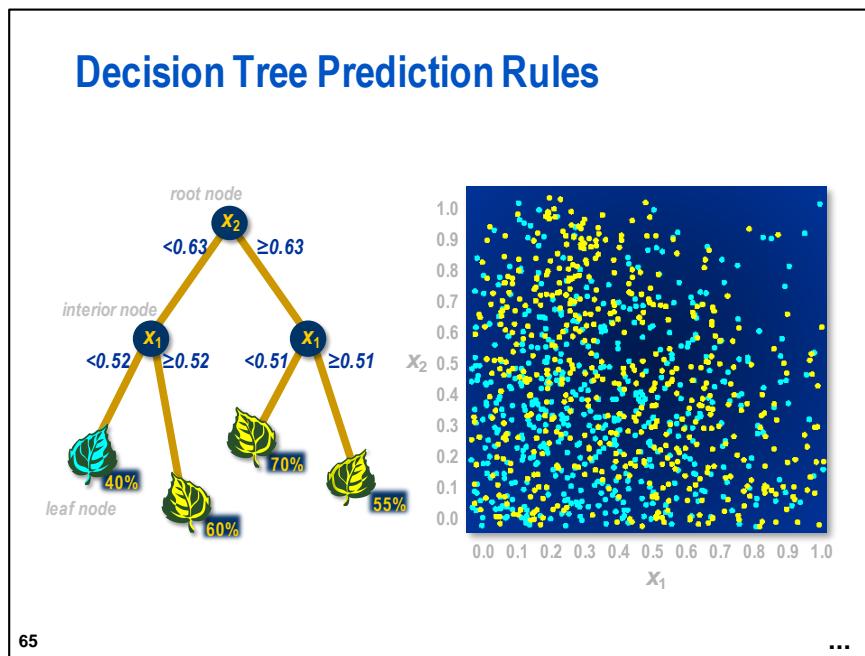
Model Essentials: Decision Trees

- ▶ Predict cases.
- ▶ Select useful predictors.
- ▶ Split search

Prediction rules

64

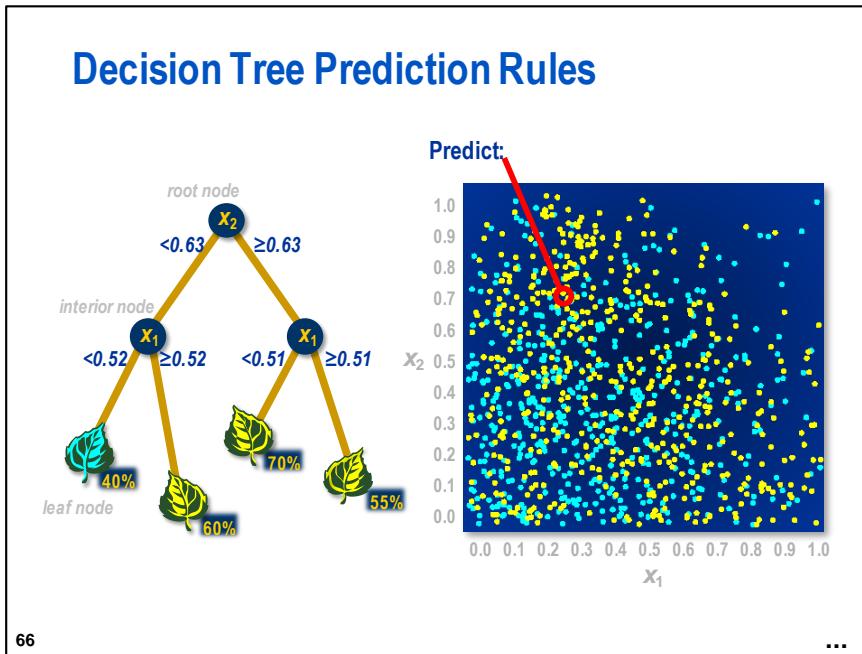
Consider again a data set with two predictors and a binary target. The predictors, x_1 and x_2 , locate each case in the unit square. The target outcome is represented by a color. Yellow is primary and blue is secondary. The analysis goal is to predict the outcome based on the location in the unit square.



To predict cases, decision trees use rules that involve the values of the predictor variables.

The rules are arranged hierarchically in a tree-like structure with nodes connected by lines. The nodes represent decision rules, and the lines order the rules. The first rule, at the base (top) of the tree, is named the *root node*. Subsequent rules are named *interior nodes*. Nodes with only one connection are *leaf nodes*.

To score a new case, examine the associated input variable values and then apply the rules defined by the decision tree.



The input variables' values of a new case eventually lead to a single leaf in the tree. A tree leaf provides a decision (for example, classify as yellow) and an estimate (for example, the primary-target proportion).

Model Essentials: Decision Trees

- Predict cases.
 - Prediction rules
 - Select useful predictors.
 - Split search

To select useful predictors, trees use a *split-search* algorithm. Decision trees confront the curse of dimensionality by ignoring irrelevant predictors.

Decision Tree Split Search

Calculate information gain on partitions on input x_1 .

69 ...

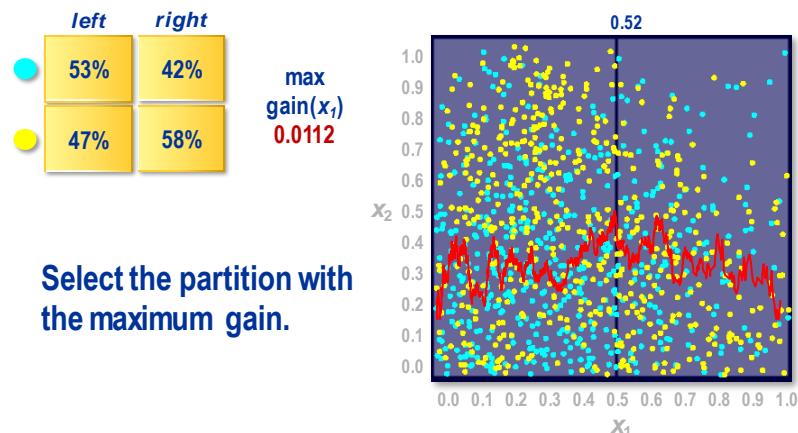
Understanding a split search algorithm for building trees enables you to better use the Tree tool and interpret your results. The description presented here assumes a binary target, but the algorithm for interval targets is similar.

The first part of the algorithm is called the *split search*. The split search starts by selecting an input for partitioning the available training data. If the measurement scale of the selected input is *interval*, unique values serve as a potential split point for the data. If the input is *categorical*, the average value of the target is taken within each categorical input level. The averages serve the same role as the unique interval input values.

For a selected input and fixed split point, two groups are generated. Cases with input values less than the split point are said to *branch left*. Cases with input values greater than the split point are said to *branch right*. The groups, combined with the target outcomes, form a 2x2 contingency table with columns specifying branch direction (left or right) and rows specifying target value (0 or 1). An information gain statistic that is based on the entropy of the root node and the entropy of the data in each partition of the split can be used to quantify the separation of counts in the table's columns. Large values for the gain statistic suggest that the proportion of zeros and ones in the left branch is different from the proportion in the right branch. A large difference in outcome proportions indicates a good split. An example of this calculation is given below.

- ✍ The split search diagnostic used in Visual Statistics depends on the approach used to grow or train the tree. Under Interactive training, a chi-squared log-likelihood-based statistic is used to evaluate splits. The default split search method under autonomous tree growth is based on an information gain statistic. An example of calculating the gain under the default method is given below. The Rapid Growth functionality combines k-means clustering with the gain statistic to grow the tree.
- ✍ A gain ratio statistic is used for split evaluation when the Split Best option is used in combination with the Rapid Growth property.

Decision Tree Split Search



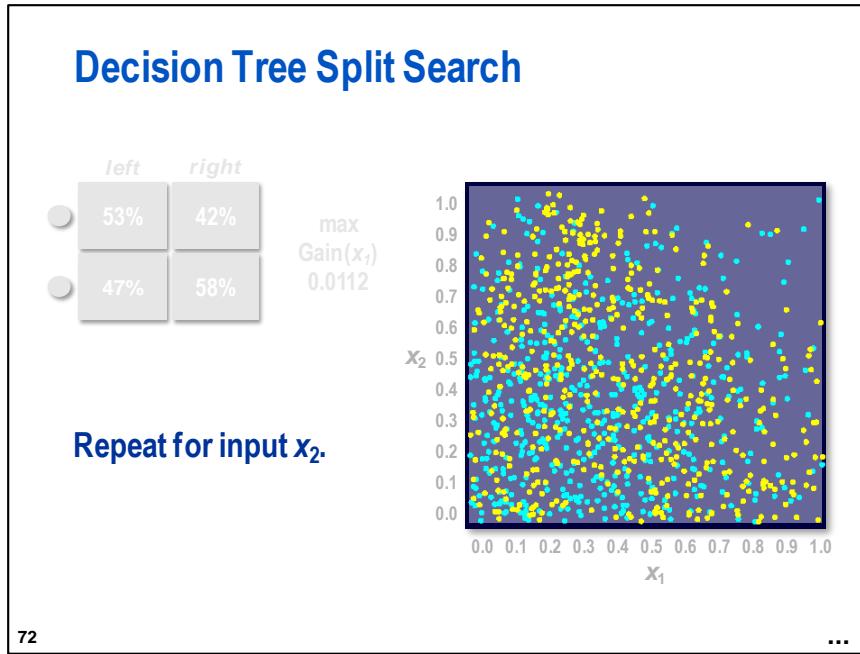
The best split for a predictor is the split that yields the highest information gain. For the gain calculation example, assume that there are 100 total observations and a 50/50 split of yellow and blue in the training data. Also, there are 52 observations to the left of the 0.52 split point and 48 observations to the right of the split in the diagram shown above. Based on this and the numbers given in the table, gain can be formulated as shown below.

$$EntropyTotal = .5 \cdot \log_2(.5) + .5 \cdot \log_2(.5) = 1$$

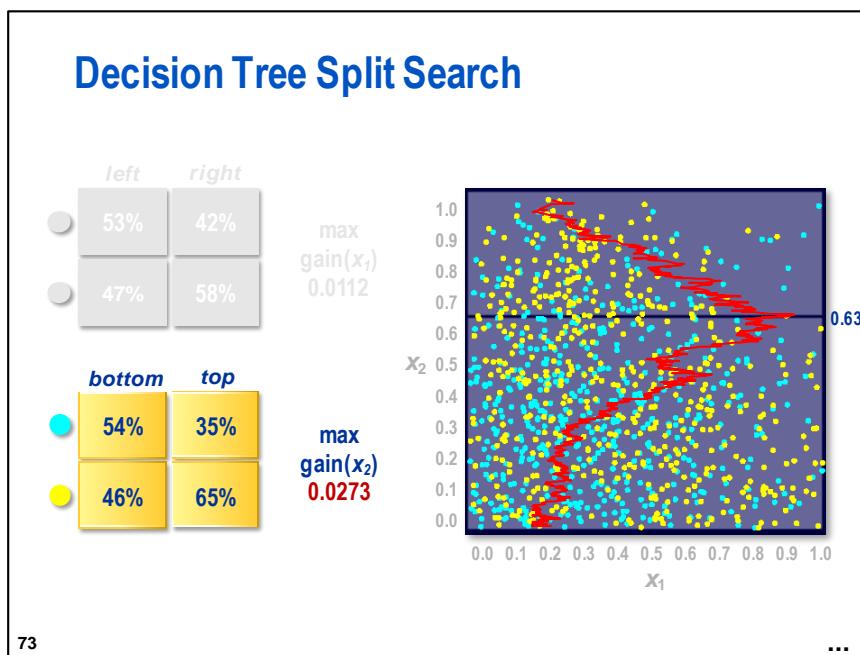
$$Entropy\ Left = .53 \cdot \log_2(.53) + .47 \cdot \log_2(.47) = 0.997$$

$$Entropy_{Right} = .42 \cdot \log_2 (.42) + .58 \cdot \log_2 (.58) = 0.98$$

$$Gain = 1 - \frac{52}{100} \cdot 0.997 - \frac{48}{100} \cdot 0.98 = 0.0112$$



The partitioning process is repeated for every input in the training data.



Again, the optimal split for the input is the one that maximizes the gain function.

Decision Tree Split Search

left right
53% 42%
47% 58%

bottom top
54% 35%
46% 65%

max gain(x_1) 0.0112
max gain(x_2) 0.0273

Compare partition gain ratings.

74 ...

After you determine the best split for every input, the tree algorithm compares each best split's corresponding gain. The split with the highest gain is deemed best.

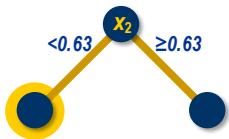
Decision Tree Split Search

Create a partition rule from the best partition across all inputs.

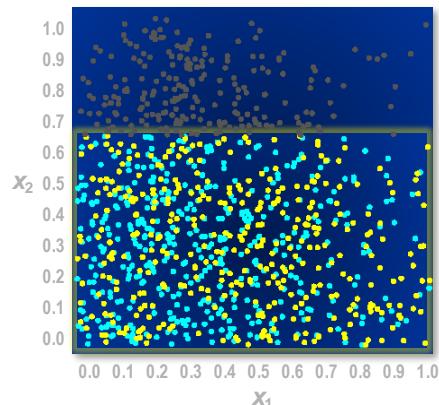
75 ...

The training data are partitioned using the best split rule.

Decision Tree Split Search



Repeat the process
in each subset.



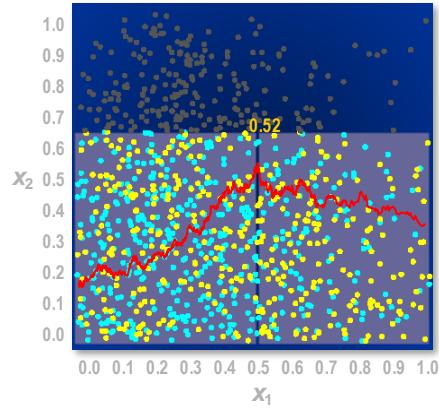
76

...

Decision Tree Split Search

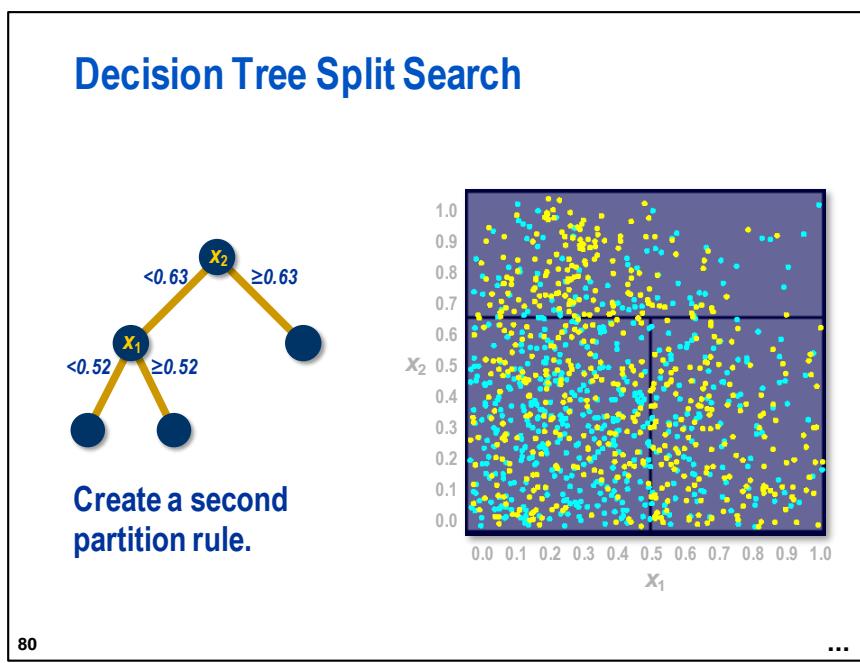
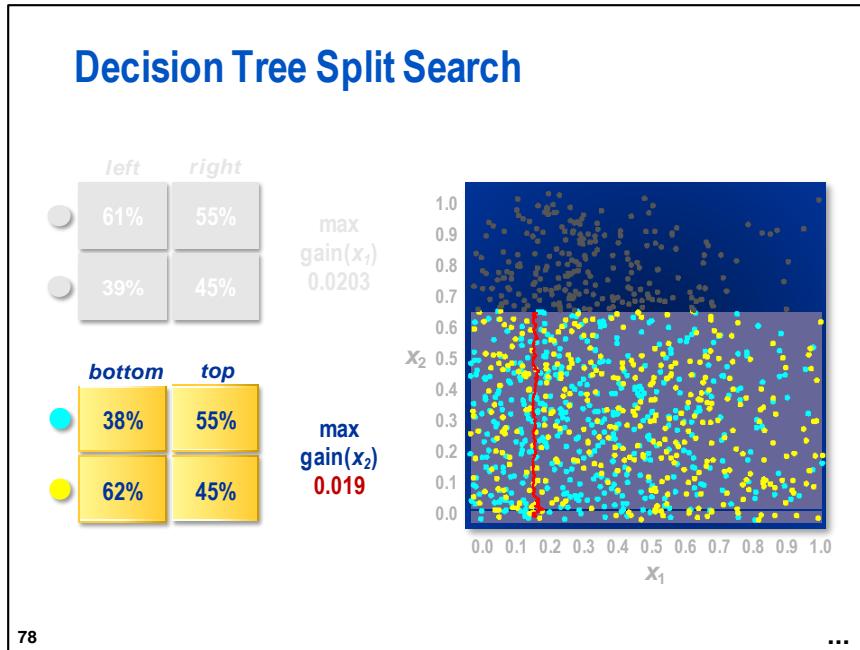
	<i>left</i>	<i>right</i>
●	61%	55%
●	39%	45%

max
 $gain(x_1)$
0.0203



77

...



The split search continues within each leaf. Gain statistics are compared as before.

Decision Tree Split Search

Repeat to form a maximal tree.

81

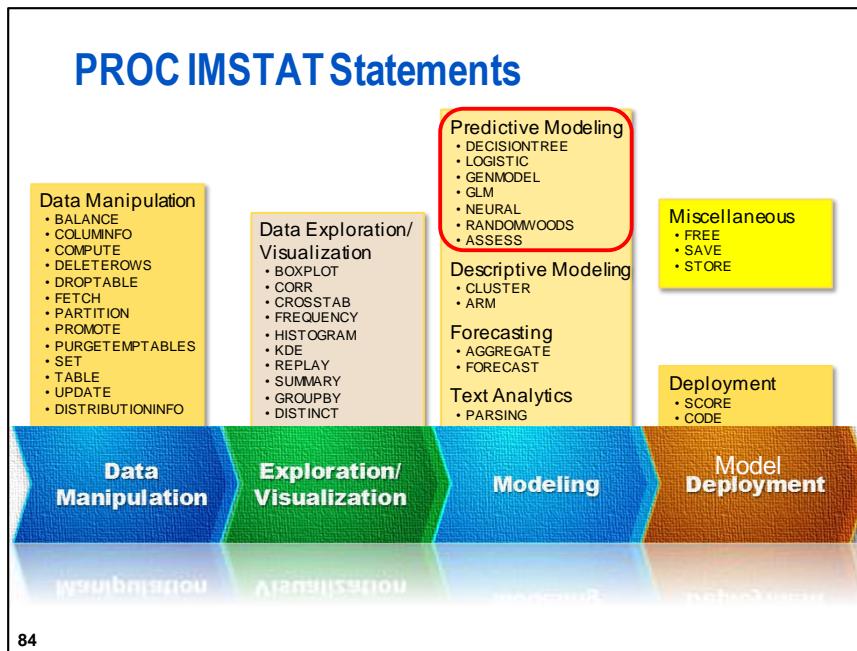
The resulting partition of the predictor variable space is known as the *maximal tree*. Under the default settings, development of the maximal tree is based exclusively on statistical measures of gain on the training data.

3.2 Categorical Targets (SAS In-Memory Statistics)

In this section and the next few sections, you learn to use supervised classification techniques in the IMSTAT procedure. In these examples, the goal is to train a model to correctly classify observations based on inputs for which the target is known. Recall that PROC IMSTAT is one of the programming components of the LASR Analytic Server.

The standard approach to predictive modeling is used. To review, this approach divides the modeling data into three partitions: training, validation, and test. The training data set is used to train the model or to derive the predictive algorithm. The validation data set is used to adjust any tuning properties, such as number of splits (pruning), number of inputs (input selection), or number and nature (or both) of parameters (complexity and flexibility). The test data set is used to get an unbiased assessment of prediction accuracy.

- ✎ For educational demonstrations, the data are partitioned into training and validation data sets. The test data set is not used in the demonstrations. If you want an unbiased estimate of accuracy for model assessment and quality control assessment of predictions in the live environment, you need a test data set.



The discussion of predictive modeling using PROC IMSTAT begins with decision trees.

Decision Trees in PROC IMSTAT

Objectives

- Describe decision tree variable roles in PROC IMSTAT.
- Describe decision tree properties in PROC IMSTAT.
- Cultivate a decision tree.
- Assess decision tree performance.

86

Decision Trees in PROC IMSTAT

- There is only one **target** variable, which can be either categorical or numeric.
- There can be multiple **input** variables.
- Both categorical and numeric valued inputs are accommodated.
- You can derive a leaf ID, which can be used in other models in PROC IMSTAT.

87

The decision tree in PROC IMSTAT uses a modified version of Quinlan's C4.5 algorithm. (This was briefly described in the Visual Statistics section about trees.)

DECISIONTREE Statement

DECISIONTREE *target-variable-name* / <options>;

- The target is assumed to be binary unless the REG option is specified.

Selected Options

- **NOMINAL=(/list)** is used to specify nominal inputs.
- **INPUTS=(/list)** is used to specify input variables including nominal.
- **SAVE=***data-set-name* saves the tree details for later scoring and assessment.
- **TREETAB=***data-set-name* specifies an existing tree that is produced by the SAVE statement and used for scoring input data.
- **ASSESS** adds predictions to the scored table that can be used as input to an ASSESS statement for model assessment.

88

Categorical- and interval-valued target variables are accommodated in the PROC IMSTAT decision tree algorithm. Although multilevel categorical target variables are allowed, one level is chosen as the event level, and other levels are combined into the non-event category.

For binary target variables, changing the event level does not affect the hierarchical structure of the decision tree. It does change the assessment plots (lift, ROC, and misclassification) that are generated for each event level. In order to do model comparisons (for example, between a logistic regression and a decision tree), you need to make sure that your models are targeting the same outcome.

The following DECISIONTREE syntax specification is extracted from *SAS® LASR Analytic Server 2.5: Reference Guide*.

DECISIONTREE Statement

The DECISIONTREE statement provides an implementation of a C4.5 decision tree method for classification. You specify a single column as the target variable when you generate the decision tree. You can also score against the generated tree.

Syntax

DECISIONTREE *target-variable* </ *option* >;

DECISIONTREE Statement Options

ASSESS

specifies that predicted probabilities are added to the temporary result table for the event levels. You can use these predicted probabilities in an ASSESS statement.

CODE <(code-generation-options)>

requests that the server produce SAS scoring code based on the actions that it performed during the analysis. The server generates DATA step code. By default, the code is replayed as an ODS table by the procedure as part of the output of the statement. More frequently, you might want to write the scoring code to an external file by specifying options.

The scoring code computes the predicted value of the response variable and prefixes the name with DT_. For example, if the response variable is Y, the generated code stores the predicted value as DT_Y. The name of the variable is truncated to fit within the SAS name length requirements.

COMMENT

specifies to add comments to the code in addition to the header block. The header block is added by default.

FILENAME='path'

specifies the name of the external file to which the scoring code is written. This suboption applies only to the scoring code itself.

Alias

FILE=

FORMATWIDTH=k

specifies the width to use when you format derived numbers such as parameter estimates in the scoring code. The server applies the BEST format, and the default format for code generation is BEST20.

Alias

FMTW=

Range

4 to 32

LABELID=id

specifies a group identifier for group processing. The identifier is an integer and is used to create array names and statement labels in the generated code.

LINESIZE=n

specifies the line size for the generated code.

Alias

LS=

Default

72

Range

64 to 256

NOTRIM

specifies to format the variables using the full format width with padding. By default, leading and trailing blanks are removed from the formatted values.

REPLACE

specifies to overwrite the external file if a file with the specified name already exists.

The option has no effect unless you specify the FILENAME= option.

CFLEV=number

specifies a value between 0 and 1 that controls the aggressiveness of tree pruning according to the C4.5 algorithm. Smaller numbers indicate to more aggressive pruning.

DETAIL

requests detailed information about the classification results when you score a table against a previously calculated tree.

FORMATS=(“format-specification”,…)

specifies the formats for the input variables. If you do not specify the FORMATS= option, the default format is applied for that variable.

Enclose each format specification in quotation marks and separate each format specification with a comma.

Example

```
proc imstat data=lasr1.table1;
  decisiontree x / input=(a b) formats=("8.3", "$10");
quit;
```

GAIN

specifies that the splitting criterion is changed to information gain. Typically, this criterion intends to generate trees with more nodes than information gain ratio.

GREEDY

specifies how to perform splitting under specific circumstances.

Assuming that one variable has q levels, when binary splitting is performed and q is less than 15, or option MAXBRANCH > 2 and $q < 12$, all possible binary splits are enumerated. The split with the largest gain or gain ratio is chosen for the variable.

When q is less than 1024 and splitting is not binary only, local greedy searches are applied to determine the optimum local split. Specifically, when the variable is numeric, q levels (similar to q bins) are sorted by value.

When the variable is nominal, the q levels are ordered by random weights. The best binary splitting is applied until the desired number of branches is reached. Only a local optimum can be found with this technique.

For values of $q \geq 1024$, the default k -means clustering algorithm is applied to determine the splits.

IMPUTE

specifies how to treat observations with nonmissing values for the target variable during scoring. When this option is specified, the observed values are used as the predicted values. That is, the observed value is assumed to be known without error. Only the observations with missing values for the target variable are then scored against the decision tree based on their values for the input variables.

The IMPUTE option is useful if you want to replace missing values of a target variable with classified values based on the decision tree.

INPUT=variable-name

INPUT=(variable-name1 <variable-name2, ...>)

specifies the variables to use for building the tree. You can add the target variable to the input list if you want to assign a format to the target variable by using the FORMATS= option. Any numeric variable that is not specified in the NOMINAL= option is binned according to the NBINS= specification.

LEAFSIZE=m

specifies the minimal number of observations on each node. When the number of observations on a tree node is less than m, the node is changed to a leaf during the building of the decision tree.

Interaction

Specifying the LEAFSIZE option affects the pruning of the tree.

MAXBRANCH=n

specifies the maximum number of children (branches) to allow for each level of the tree.

Default

2

MAXLEVEL=n

specifies the maximum number of the tree level.

Default

6

MULTVAR

specifies to allow a variable to appear multiple times when traversing the tree from top to bottom.

NBINS=k

specifies the number of bins to use in the calculation of the decision tree. The number of bins affects the accuracy of the tree. The accuracy increases as values of k increase. However, computing time and memory consumption also increase as values of k increase.

Default

2

NBINSTARGET=k

specifies the number of bins to use for a numeric target variable. The number of bins affects the accuracy of the tree. The accuracy increases as values of k increase. However, computing time and memory consumption also increase as values of k increase. When k is greater than zero, the numeric target variable is binned into equally sized bins first and then the bins are used to perform the classification.

Default

0

NOMINAL=variable-name**NOMINAL=(variable-list)**

specifies the numeric variables to use as nominal variables. Binning is not applied to the specified variables. The target variable is always treated as a nominal variable and does not need to be listed.

NOMISSOBS

specifies to ignore observations that have missing values in the analysis variables when building a decision tree. When scoring a data set, any observations with missing values in the analysis variables for the decision tree are ignored when this option is specified.

When this option is not specified, the RANDOMWOODS statement builds a tree by applying the following policy for missing values:

- for an interval variable, the smallest machine value is assigned
- for a nominal variable, missing values are represented by a separate level

NOPREPARSE

prevents the procedure from preparsing and pregenerating code for temporary expressions, scoring programs, and other user-written SAS statements.

When this option is specified, the user-written statements are sent to the server "as is" and then the server attempts to generate code from it. If the server detects problems with the code, the error messages might not be as detailed as the messages that are generated by SAS client. If you are debugging your user-written program, then you might want to preparse and pregenerate code in the procedure. However, if your SAS statements compile and run as you want them to, then you can specify this option to avoid the work of parsing and generating code on the SAS client.

When you specify this option in the PROC IMSTAT statement, the option applies to all statements that can generate code. You can also exclude specific statements from preparsing by using the NOPREPARE option in statements that allow temporary columns or the SCORE statement.

Alias

NOPREP**NOPRUNEOBS**

specifies not to prune any observations when building a decision tree.

NOSCORE

suppresses the generation of the scoring temporary table when the TEMPTABLE option is specified. In this case, the server generates only one temporary table and the table contains the decision tree.

PRUNE

requests to prune the tree according to the C4.5 algorithm. Pruning can increase the error of misclassification. You can control the aggressiveness of pruning with the CFLEV= option. Smaller values for the CFLEV= option result in more aggressive pruning.

PRUNEGROW

specifies to enable C4.5 pruning when building a classification decision tree. The tree could have a large misclassification rate but the building process is performed quickly.

REG

specifies to build a regression tree. Minimal cost-complexity pruning is applied to prune the tree.

SAVE=*table-name*

saves the result table so that you can use it in other IMSTAT procedure statements like STORE, REPLAY, and FREE. The value for *table-name* must be unique within the scope of the procedure execution. The name of a table that has been freed with the FREE statement can be used again in subsequent SAVE= options.

SCOREDATA=*table-name*

specifies the in-memory table that contains the scoring data. The table must exist in memory on the server. The DECISIONTREE statement in the IMSTAT procedure does not transfer a local data set to the server.

SETSIZE

requests that the server estimate the size of the result set. The procedure does not create a result table if the SETSIZE option is specified. Instead, the procedure reports the number of rows that are returned by the request and the expected memory consumption for the result set (in KB). If you specify the SETSIZE option, the SAS log includes the number of observations and the estimated result set size. See the following log sample:

- ☞ The LASR Analytic Server action request for the SETSIZE statement would return 17 rows and approximately 3.641 kBytes of data.

The typical use of the SETSIZE option is to get an estimate of the size of the result set in situations where you are unsure whether the SAS session can handle a large result set. Be aware that in order to determine the size of the result set, the server has to perform the work as if you were receiving the actual result set. Requesting the estimated size of the result set does consume resources on the server. The estimated number of KB is very close to the actual memory consumption of the result set. It might not be immediately obvious how this size relates to the displayed table, because many tables contain hidden columns. In addition, some elements of the result set might not be converted to tabular output by the procedure.

STAT

specifies to generate two additional tables that contain statistical information about the variables that are used in the decision tree. One table contains the variable importance information, which is determined by the total Gini reduction. The second table contains the variable splitting information for each node in the decision tree.

TEMPEXPRESS="*SAS expressions***"****TEMPEXPRESS=***file-reference*

specifies either a quoted string that contains the SAS expression that defines the temporary variables or a file reference to an external file with the SAS statements.

Alias

TE=

TEMPNAMES=*variable-name*

TEMPNAMES=(*variable-list*)

specifies the list of temporary variables for the request. Each temporary variable must be defined through SAS statements that you supply with the TEMPEXPRESS= option.

Alias

TN=

TEMPTABLE

specifies to store the results in a temporary table. The type of information that is stored depends on whether you are building a decision tree or scoring a table with a decision tree.

When you are building a decision tree, the generated decision tree is stored in the server and the input table is automatically scored using this tree. The scoring details are saved in a temporary table. The **_TEMPTREE_** macro variable stores the name of the temporary table for the tree.

The **_TEMPSCORE_** macro variable stores the name of the temporary table that has the scoring results of traversing the decision tree. You can suppress the generation of the scoring temporary table (**_TEMPSCORE_**) during the tree building phase by specifying the NOSCORE option.

When you are scoring a table using a decision tree, the TEMPTABLE option requests to store the scoring details in a temporary table in the server. The IMSTAT procedure displays the name of the table and stores it in the **_TEMPSCORE_** macro variable. Be aware that the DETAIL option can generate a very large amount of scoring results when the in-memory table that is specified in the SCOREDATA= option is large. Observations from the scored data set can be transferred to the temporary table using the VARS= option.

TIMEOUT=*s*

specifies the maximum number of seconds that the server should run the statement. If the time-out is reached, the server terminates the request and generates an error and error message. By default, there is no time-out.

TREEDATA=*libref.member-name*

TREETAB=*saved-table*

TREELASRTAB=*table-name*

specifies the saved table that contains the generated tree. In order to score a (validation) data set against the generated tree, you need the validation data and a representation of the tree. Specify these options as follows:

- The TREEDATA= option is used to specify the name of a SAS data set that stores the generated tree. The data set is local to the SAS client.
- The TREETAB= option is used to specify a table on the SAS client that stores the generated tree.
- The TREELASRTAB= option is used to specify a valid decision tree that is stored in an in-memory table.
- The data set with the observations to score is specified in the SCOREDATA= option.

Alias

SCORETAB=

VARS=variable-name

VARS=(variable-name1 <variable-name2, ...>)

specifies the variables to transfer from the input table to the temporary table in the server that contains the results of scoring a decision tree. This option has no effect unless you specify the TEMPTABLE option and you score a decision tree.

Additional Information about Pruning

In PROC IMSTAT, pruning is based on the C4.5 algorithm. This method uses a confidence bound constructed on the estimated error rate of the tree over a given subset of training cases to prune. The upper confidence bound of the estimated error rate is used to approximate the true error rate. The “aggressiveness” of the algorithm corresponds to the alpha parameter chosen to construct the confidence interval. Notice the similarity to the decision tree in SAS Visual Statistics. See *SAS® LASR Analytic Server: Reference Guide* for additional details.

ODS Tables

Decision Tree Results: ODS Table Names		
ODS Table Name	Description	Option
DTREE	Classification decision tree	Default
DTreeVarImplInfo	Variable importance in a decision tree	STAT
DTreeVarStatInfo	Variable information for a decision tree	STAT
DTREESCORE	Classification decision tree scoring summary	SCOREDATA=
GeneratedCode	Generated SAS code from a modeling task	CODE
TempTable	Information about a temporary table	TEMPTABLE



Growing a Decision Tree

This demonstration illustrates basic exploratory analysis with PROC IMSTAT and concludes by growing a decision tree for the binary target.

In this demonstration, a LASR Analytic Server session is started and the SASHDAT table **p_model_bank13** is loaded from HDFS into memory. Three program files are used: **mldmbd_macros.sas**, **mldmbd03d01_GettingStarted.sas**, and **mldmbd03d02_DecisionTree.sas**.



If you are using SAS Studio, make sure that it is set to interactive mode each time you open a program.

- *Interactive mode* is required to run the IMSTAT procedure interactively using RUN statements.
- *Non-interactive mode* submits an implicit QUIT statement after each submission. However, non-interactive mode retains library definitions and macro variable definitions, whereas interactive mode requires redefining macro variables and libraries for each interactive session. In this course, you run PROC IMSTAT code separately for each RUN group. Relevant macros are stored in a macro definition file and included in each interactive session.

To use the IMSTAT procedure, you must define the high-performance environment. Some environment variables are defined at start-up when the server is started.

Two libraries are defined: the first to access SASHDAT files, and the second to access SAS Foundation files.

The first demonstration program starts a LASR Analytic Server on the specified port. The program contains code that defines a **SASIOLA** library to communicate with in-memory tables that are loaded on the LASR Analytic Server. A LASR Analytic Server must be started before the **SASIOLA** library is defined.

Because you are running SAS in interactive mode, you need to first establish the macro variables for the interactive session. The **MLDMBD_Macros.sas** file contains macro definitions that must be invoked for each new interactive session. Your instructor might ask you to edit some of the lines in this file before the first demonstration. The macro file contains the following code to set system options and to establish the SASHDAT library.

```
options set=GRIDHOST="&host"
       set=GRIDINSTALLLOC="&GridInstall" ;

libname HDFSlib
      SASHDAT
      PATH="&HDFSFolder" ;
```

The following code is at the beginning of every SAS program that is run interactively:

```
%let UserID=student;
%include "/home/&UserID/MLDMBD/MLDMBD_Macros.sas" ;
```

You might need to edit this code before running an interactive program.

Open the **mldmbd03d01_Getting_Started.sas** program.

The LIBNAME statement for **HDFSlib** provides the connection between SAS and Hadoop. The code below copies the SAS Foundation data set **p_model_bank13** into the Hadoop Distributed File System using the SASHDAT format. Using SASHDAT permits data to be copied directly from disk to memory without any additional conversions.

```
libname SASlib "&SASdataFolder";
proc copy in=SASlib out=HDFSlib;
  select p_model_bank13;
run;
```

 If there is one way to do something in SAS, there might be many other ways to do the same thing. The following code can also be used to copy data from a SAS Foundation file into HDFS:

```
data HDFSlib.p_model_bank13;
  set SASlib.p_model_bank13;
run;
```

PROC LASR is used to start the LASR Analytic Server. In the code below, the server is configured to shut down after 86,400 seconds, which is one day, if there is no activity for 7,200 seconds, which is two hours. The HOST= and INSTALL= options are controlled by the configuration of the system on which you are working.

```
proc lasr
  port=&SessionPort
  lifetime=86400 (7200)
  path="&LASRpath";
  performance nodes=all;
run;
```

 OPTION statements depend on the environment and are likely changed for the specific virtual lab environments that are used for this course. Also, port numbers depend on the availability of ports for multi-user configurations.

The SAS log confirms that the libraries are defined correctly. The results confirm that the LASR Analytic Server started in distributed mode.

The LASR Procedure	
Performance Information	
Host Node	eduhad01
Install Location	/opt/sas/TKGGrid
Execution Mode	Distributed
Number of Compute Nodes	4

Your virtual lab environment might use a different number of compute nodes.

Because you might need to stop the LASR Analytic Server, the following code is presented in comment indicators so that it does not execute automatically:

```
/* Stop LASR Analytic Server if necessary

proc lasr stop port=&SessionPort;
    performance host    = "&host";
run;

*/
```

For convenience, this code is placed in a permanent program file called **mldmbd00d01_StopLASR.sas**.

A LIBNAME statement uses the SASIOLAengine to establish a connection to the LASR Analytic Server.

```
libname LASRlib sasiola port=&SessionPort tag=&TagString;
```

The macro variables **SessionPort** and **TagString** are defined in the **MLDMBD_Macros.sas** program.
In PROC IMSTAT, data in memory are referenced using the **LASRlib** library name.

The code below lists the contents of **p_model_bank13**. The PROC LASR statement with the ADD option loads the date into parallel memory.

```
***** Get information on project data *****/
proc contents data=HDFSlib.p_model_bank13;
run;

***** Load data from Hadoop (SASHDAT format) into LASR ****/
proc lasr port=&SessionPort
    add data=HDFSlib.p_model_bank13 noclass;
run;
```

The partial results of the PROC CONTENTS step are shown below. The **PartInd** variable is used to distinguish between the training and validation partitions of the data for honest assessment.

The CONTENTS Procedure					
Data Set Name	/user/student2/HPAdata/p_model_bank13.sashdat			Observations	.
Member Type	DATA			Variables	56
Engine	SASHDAT			Indexes	0
Created	01/27/2015 16:01:20			Observation Length	456
Last Modified	01/26/2015 15:49:04			Deleted Observations	0
Protection				Compressed	NO
Data Set Type				Sorted	NO
Label					
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64				
Encoding	latin1 Western (ISO)				

Alphabetic List of Variables and Attributes					
#	Variable	Type	Len	Format	Label
56	_PartInd_	Num	8	BEST12.	Binary Training Indicator
54	account	Char	9	\$9.	Account ID
1	b_tgt	Num	8	BEST12.	tgt Binary New Product
55	bin_int_tgt	Num	8	BEST12.	Rank for tgt interval New Sales
2	cat_input1	Char	5	\$5.	category 1 Account Activity Level
3	cat_input2	Char	1	\$1.	category 2 Customer Value Level
4	cnt_tgt	Num	8	BEST12.	tgt Count Number New Products
5	demog_age	Num	8	BEST12.	demog Customer Age
52	demog_genf	Num	8	BEST12.	demog Female Binary
53	demog_genm	Num	8	BEST12.	demog Male Binary
6	demog_ho	Num	8	BEST12.	demog Homeowner Binary

The results confirm that the data were loaded onto the LASR Analytic Server.

The LASR Procedure					
Performance Information					
Host Node		eduhad01			
Execution Mode		Distributed			
Number of Compute Nodes		4			

Data Access Information					
Data	Engine	Role	Path		
HDFSLIB.P_MODEL_BANK13	SASHDAT	Input	Parallel, Symmetric		

Continue with the program **6_1_GettingStarted.sas**.

```

proc imstat;
  /*---- List the tables in the LASR Analytic Server ----*/
  tableinfo / port=&SessionPort;
run;
/*---- Make model_bank13 the active table ----*/
table LASRlib.p_model_bank13(tag)="&TagString";

/*---- List the columns in the project table --*/
columninfo;
run;
/*---- Print 10 rows ----*/
fetch/ from=1 to=10;
run;

```

The TABLEINFO statement and the PORT= option list the tables on the LASR Analytic Server.

The IMSTAT Procedure						
Table Information						
Table Name	NLS encoding	Number of Rows	Number of Columns	Owner	Created	
USER.STUDENT2.HPADATA.P_MODEL_BANK13	latin1	1060038	56	student2	Tue Jan 27 16:01:32 2015	

The TABLE statement makes **p_model_bank13** the active table. The COLUMNINFO statement lists the columns in the table, and the FETCH statement list all columns for the first five rows.

Partial results are shown below.

Column Information for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13					
Id	Column	Type	Length	Format	Label
1	b_tgt	Num	8	BEST12.	tgt Binary New Product
2	cat_input1	Char	5	\$5.	category 1 Account Activity Level
3	cat_input2	Char	1	\$1.	category 2 Customer Value Level
4	cnt_tgt	Num	8	BEST12.	tgt Count Number New Products
5	demog_age	Num	8	BEST12.	demog Customer Age
6	demog_ho	Num	8	BEST12.	demog Homeowner Binary
7	demog_homeval	Num	8	DOLLAR11.	demog Home Value
8	demog_inc	Num	8	DOLLAR11.	demog Income
9	demog_pr	Num	8	BEST12.	demog Percentage Retired
10	i_demog_age	Num	8	BEST12.	demog_i Customer Age
11	i_rfm1	Num	8	BEST12.	i_rfm1 Average Sales Past 3 Years
12	i_rfm2	Num	8	BEST12.	i_rfm2 Average Sales Lifetime

b_tgt	cat_input1	cat_input2	cnt_tgt	demog_age	demog_ho	demog_homeval	demog_inc	demog_pr
1.000000	X	A	.	.	0	57600	52106	24.000000
1.000000	X	A	2.000000	.	0	57587	52106	24.000000
1.000000	X	A	2.000000	.	0	44167	42422	0
0	X	A	0	68.000000	0	90587	59785	32.000000
0	X	A	0	.	0	100313	0	0
0	X	A	0	26.000000	0	26622	34444	0
0	X	A	0	74.000000	1.000000	95496	0	0
0	X	A	0	83.000000	0	65814	32597	34.000000
0	X	A	0	70.000000	0	26007	33748	38.000000
0	X	A	0	77.000000	1.000000	0	0	0



In a distributed processing environment, row numbers might not be uniquely determined, so the actual results of a FETCH statement might not be the same for different environments.

For tables with many columns, it is convenient to be able to refer to groups of columns that are related in some way. For example, the columns that begin with **rfm**, **i_rfms**, and **logi_rfms** are all interval variables related to each other. They reflect the historical behavior of the cases. Another group of variables is associated with case demographics, and a third group of variables is nominal. SAS macro variables can be created to conveniently reference these groups of columns.

The COLUMNINFO / SAVE=COLUMNS statement saves a temporary table named **columns**. This table can be accessed to create macro variables using a STORE statement. The WHERE clause creates the macro variable **&rfms**. The %PUT statement prints the value of the macro variable in the log so that you can verify whether it was created as intended.

```
columninfo / save=columns;
run;
store columns[1] (where="substr(column,1,3)='rfm'", cols=Column)=rfms;
run;
%put &rfms;
```

Summary statistics for the **&rfms** variables are requested by the code below.

```
summary &rfms;
run;
```

The results of the SUMMARY statement are shown below.

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13									
Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing
rfm1	-1.0000	3713.31	1080038	17059167.6	16.0930	19.3025	0.01875	119.94	0
rfm2	1.5800	650.00	1080038	14153138.2	13.3515	9.4709	0.009199	70.9348	0
rfm3	0	3713.31	834252	12772302.1	15.3099	18.9667	0.02077	123.89	225788
rfm4	-1.0000	10000	1080038	18520644.2	17.4717	37.5505	0.03647	214.92	0
rfm5	0	18.0000	1080038	3082375	2.9078	2.0286	0.001970	69.7633	0
rfm6	0	127.00	1080038	101111707	9.5390	8.4721	0.008229	88.8156	0
rfm7	0	11.0000	1080038	1766746	1.6667	1.5257	0.001482	91.5402	0
rfm8	0	48.0000	1080038	5327671	5.0259	4.5101	0.004381	89.7367	0
rfm9	2.0000	29.0000	1080038	19453344	18.3516	4.0216	0.003906	21.9143	0
rfm10	0	77.0000	1080038	13663586	12.8897	4.6074	0.004475	35.7445	0
rfm11	0	22.0000	1080038	5680291	5.3586	1.3604	0.001321	25.3868	0
rfm12	0	571.00	1080038	72225458	68.1348	37.3493	0.03628	54.8168	0

Notice that **rfm3** has many missing values.

The code below performs the same actions as above for the other versions of the **rfm** variables.

```

store columns[1]
  (where="substr(column,1,5)='i_rfms'",cols=Column)=i_rfms;
run;
%put &i_rfms;
summary &i_rfms;
run;
store columns[1]
  (where="substr(column,1,8)='logi_rfms'",cols=Column)=logi_rfms;
run;
%put &logi_rfms;
summary &logi_rfms;
run;

```

The results of the SUMMARY statements are shown below.

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13										
Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing	
i_rfm1	0	3713.31	1060038	17060671.7	16.0944	19.3019	0.01875	119.93	0	
i_rfm2	1.5800	650.00	1060038	14153138.2	13.3515	9.4709	0.009199	70.9348	0	
i_rfm3	0	3713.31	1060038	18229059.7	15.3099	16.8260	0.01634	109.90	0	
i_rfm4	0	10000	1060038	18521198.3	17.4722	37.5504	0.03647	214.92	0	
i_rfm5	0	18.0000	1060038	3082375	2.9078	2.0286	0.001970	69.7633	0	
i_rfm6	0	127.00	1060038	10111707	9.5390	8.4721	0.008229	88.8156	0	
i_rfm7	0	11.0000	1060038	1766746	1.6667	1.5257	0.001482	91.5402	0	
i_rfm8	0	46.0000	1060038	5327671	5.0259	4.5101	0.004381	89.7367	0	
i_rfm9	2.0000	29.0000	1060038	19453344	18.3516	4.0216	0.003906	21.9143	0	
i_rfm10	0	77.0000	1060038	13663586	12.8897	4.6074	0.004475	35.7445	0	
i_rfm11	0	22.0000	1060038	5680291	5.3586	1.3604	0.001321	25.3868	0	
i_rfm12	0	571.00	1060038	72225458	68.1348	37.3493	0.03628	54.8168	0	

Notice that the missing values for **rfm3** were imputed. That is, **i_rfm3** is **rfm3** with imputed missing values. Imputed values for interval variables are derived using a means method.

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13										
Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing	
logi_rfm1	0	8.2199	1060038	2847295.05	2.6860	0.5708	0.000554	21.2497	0	
logi_rfm2	0.9478	8.4785	1060038	2696596.68	2.5439	0.4684	0.000455	18.4116	0	
logi_rfm3	0	8.2199	1060038	2840912	2.6800	0.4557	0.000443	17.0049	0	
logi_rfm4	0	9.2104	1060038	2925957.38	2.7802	0.5445	0.000529	19.7272	0	
logi_rfm5	0	2.9444	1060038	1307169.47	1.2331	0.5198	0.000505	42.1536	0	
logi_rfm6	0	4.8520	1060038	2173006.68	2.0499	0.8129	0.000790	39.6555	0	
logi_rfm7	0	2.4849	1060038	873439.97	0.8240	0.5670	0.000551	68.8156	0	
logi_rfm8	0	3.8601	1060038	1614374.84	1.5229	0.7633	0.000741	50.1197	0	
logi_rfm9	1.0986	3.4012	1060038	3110714.02	2.9345	0.2610	0.000254	8.8955	0	
logi_rfm10	0	4.3567	1060038	2745514.83	2.5900	0.2743	0.000266	10.5925	0	
logi_rfm11	0	3.1355	1060038	1934083.36	1.8245	0.2349	0.000228	12.8730	0	
logi_rfm12	0	6.3491	1060038	4304694.12	4.0609	0.6260	0.000608	15.4158	0	

A small amount was added to the **i_rfm** variables so that no missing values were introduced using the log transformation.

The code below creates macro variables for the demographic and categorical variables.

```
store columns[1]
  (where="substr(column,1,5)='demog'",cols=Column)=demogs;
run;
%put &demogs;
store columns[1] (where="column in ('cat_input1' 'cat_input2')",
  cols=Column)=catvars;
run;
%put &catvars;
```

The count of distinct values in a column can be obtained using the DISTINCT statement as shown below.

```
distinct account &demogs &catvars b_tgt cnt_tgt _partInd_;
run;
```

The results are shown below.

Number of Distinct Values in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13		
Column	Number of Distinct Values	Number of Missing Values
account	1060038	0
demog_age	92	266861
demog_ho	2	0
demog_homeval	226725	0
demog_inc	48879	0
demog_pr	99	0
demog_genf	2	0
demog_genm	2	0
cat_input1	3	0
cat_input2	5	0
b_tgt	2	0
cnt_tgt	8	1
PartInd	2	0

The two demographic variables, **demog homeval** and **demog inv**, exhibit a high percentage of values equal to zero, which are omitted from the displayed results. These zero values should be coded as unknown. To do so, first code the zeros as missing and then impute the missing values.

The results of these steps are shown below using the following code:

```
summary r_demog: ri_demog: int_tgt;
run;
```

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13										
Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing	
r_demog_homeval	7436.00	600067	1047905	1.12474E11	107332	93123	90.9697	86.7618	12133	
r_demog_inc	2493.00	200007	806785	4.27923E10	53041	18977	21.1271	35.7775	253253	
ri_demog_homeval	7436.00	600067	1060038	1.13761E11	107318	92589	89.9286	86.2752	0	
ri_demog_inc	2493.00	200007	1060038	5.30158E10	50013	17415	16.9145	34.8205	0	
int_tgt	0	500000	211509	2376487040	11236	8491.80	18.4644	75.5776	848529	

Notice that the interval target also has many missing values, which are explained in a subsequent analysis.

Frequencies and two-way cross tabulations are also useful for exploratory analysis. One-way frequencies can be calculated using the FREQUENCY statement.

```
frequency partind b tgt cnt tgt &catvars
           demog_ho demog_genf demog_genm;
run;
```

Frequencies for Column _PartInd_ in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Level	Formatted	Value	Frequency	
1	1	1	531423	
2	2	2	528815	

Frequencies for Column b_tgt in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Level	Formatted	Value	Frequency	
1	0	0	848529	
2	1	1	211509	

Frequencies for Column cnt_tgt in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Level	Formatted	Value	Frequency	
1	-		1	
2	0	0	848529	
3	1	1	115819	
4	2	2	75592	
5	3	3	17237	
6	4	4	2552	
7	5	5	297	
8	6	6	11	

Frequencies for Column cat_input1 in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13

Level	Formatted	Value	Frequency
1	X	X	831371
2	Y	Y	77847
3	Z	Z	150820

Frequencies for Column cat_input2 in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13

Level	Formatted	Value	Frequency
1	A	A	188398
2	B	B	192382
3	C	C	169550
4	D	D	122282
5	E	E	387426

Frequencies for Column demog_ho in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13

Level	Formatted	Value	Frequency
1	0	0	476741
2	1	1	583297

Frequencies for Column demog_genf in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13

Level	Formatted	Value	Frequency
1	0	0	464288
2	1	1	595750

Frequencies for Column demog_genm in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13

Level	Formatted	Value	Frequency
1	0	0	595750
2	1	1	464288

Two-way cross tabulations can be calculated using the CROSSTAB statement as shown below.

```
crosstab partind *b tgt;
crosstab b_tgt*cnt_tgt;
crosstab cat_input1*cat_input2;
run;
```

Notice that the partition is very close to 50% training and 50% validation.

Cross-tabulation of _PartInd_ by b_tgt for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13 with Aggregator N		
PartInd	0	1
1	424385	107038
2	424144	104471

Cross-tabulation of _PartInd_ by b_tgt for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13 with Aggregator N		
PartInd	0	1
1	424385	107038
2	424144	104471

Cross-tabulation of b_tgt by cnt_tgt for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13 with Aggregator N								
b_tgt	.	0	1	2	3	4	5	6
0	0	848529	0	0	0	0	0	0
1	1	0	115819	75592	17237	2552	297	11

Cross-tabulation of cat_input1 by cat_input2 for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13 with Aggregator N						
cat_input1	A	B	C	D	E	F
X	153689	146828	133376	94371	303107	1000
Y	5478	17757	13234	10106	31272	1000
Z	29231	27797	22940	17805	53047	1000

You can compare the distribution of the interval targets by the partition and by the binary target with the GROUPBY= option as shown below.

```
summary cnt tgt int tgt/ groupby=b tgt;
summary cnt_tgt int_tgt/ groupby=_partind_;
run;
```

 The use of a GROUPBY statement does not require sorting the data before using the SUMMARY statement. Furthermore, you can combine the two SUMMARY statements into a single MDSUMMARY statement, which enables multiple group by variables.

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13										
b_tgt	Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing
0	cnt_tgt	0	0	848529	0	0	0	0	-	0
0	int_tgt	.	.	0	-	.	.	.	-	848529
1	cnt_tgt	1.0000	6.0000	211508	330473	1.5625	0.7087	0.001637	45.2278	1
1	int_tgt	0	500000	211509	2376487040	11236	8491.80	18.4844	75.5776	0

Summary Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13										
PartInd	Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing
1	cnt_tgt	0	6.0000	531423	167260	0.3147	0.7024	0.000964	223.17	0
1	int_tgt	0	500000	107038	1203146420	11240	8489.09	25.8862	75.3453	424385
2	cnt_tgt	0	6.0000	528614	163213	0.3088	0.6989	0.000959	225.71	1
2	int_tgt	0	500000	104471	1173340620	11231	8515.05	28.3445	75.8156	424144

The first table shows why there are so many missing values for **int_tgt**. If **b_int=0**, then **int_tgt** is missing. The second table shows that the distribution of the count and interval targets is very similar within each partition. The partition was constructed to ensure that the distributions are similar to improve the accuracy of honest assessment.

Neither the IMSTAT procedure nor the LASR Analytic Server produces graphics. In the GUI environment, the LASR Analytic Server calculates summary statistics and the GUI renders the graphs. In the programming environment, you can calculate summary statistics, save the summary statistics to the SAS server, and produce graphs using Base SAS graphics procedures.

The code below uses the HISTOGRAM statement to calculate the count target summary statistics and saves the results in the SAS data file **work.cnt_histogramdata** using the ODS OUTPUT statement.

```
ods output histogram=work.cnt_histogramdata;
histogram cnt_tgt/nbins=8;
run;
```

The results are below.

Histogram for Column cnt_tgt in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13					
Bin Number	Min	Mid	Max	Frequency	Percent
1	-0.5	0	0.5	848529	80.0471
2	0.5	1	1.5	115819	10.9259
3	1.5	2	2.5	75592	7.1311
4	2.5	3	3.5	17237	1.6261
5	3.5	4	4.5	2552	0.2407
6	4.5	5	5.5	297	0.02802
7	5.5	6	6.5	11	0.001038
8	6.5	7	7.5	0	0

The code below calculates the histogram data for the interval target.

```
ods output histogram=work.int histogramdata;
histogram int_tgt/nbins=200 noemptybin;
run;
quit;
```

The QUIT statement terminates the IMSTAT procedure.

A listing of the **int_tgt** histogram data is shown below.

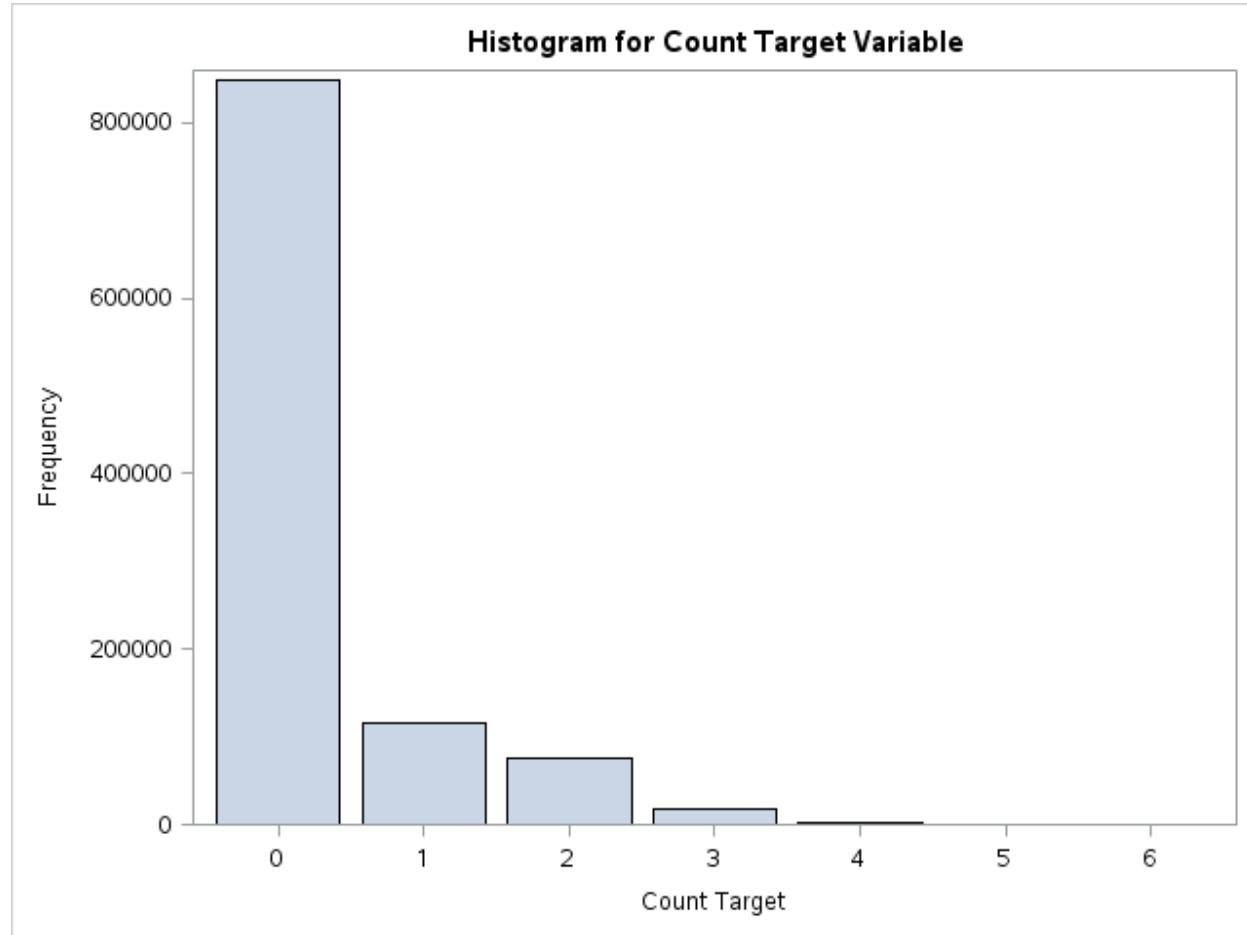
Histogram for Column int_tgt in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13					
Bin Number	Min	Mid	Max	Frequency	Percent
1	-2500	0	2500	20262	9.5797
2	2500	5000	7500	49008	23.1697
3	7500	10000	12500	56375	26.6537
4	12500	15000	17500	49896	23.5905
5	17500	20000	22500	28941	13.6831
6	22500	25000	27500	4279	2.0231
7	27500	30000	32500	902	0.4265
8	32500	35000	37500	462	0.2184
9	37500	40000	42500	341	0.1612
10	42500	45000	47500	99	0.04681
11	47500	50000	52500	572	0.2704
12	52500	55000	57500	44	0.02080
13	57500	60000	62500	22	0.01040
14	62500	70000	72500	22	0.01040
15	72500	75000	77500	33	0.01560
16	77500	80000	82500	11	0.005201
17	82500	90000	92500	11	0.005201
18	92500	100000	102500	176	0.08321
19	102500	110000	112500	11	0.005201
20	112500	200000	202500	33	0.01560
21	202500	500000	502500	11	0.005201

Histograms for the two targets are produced by the SG PLOT procedure using the code below.

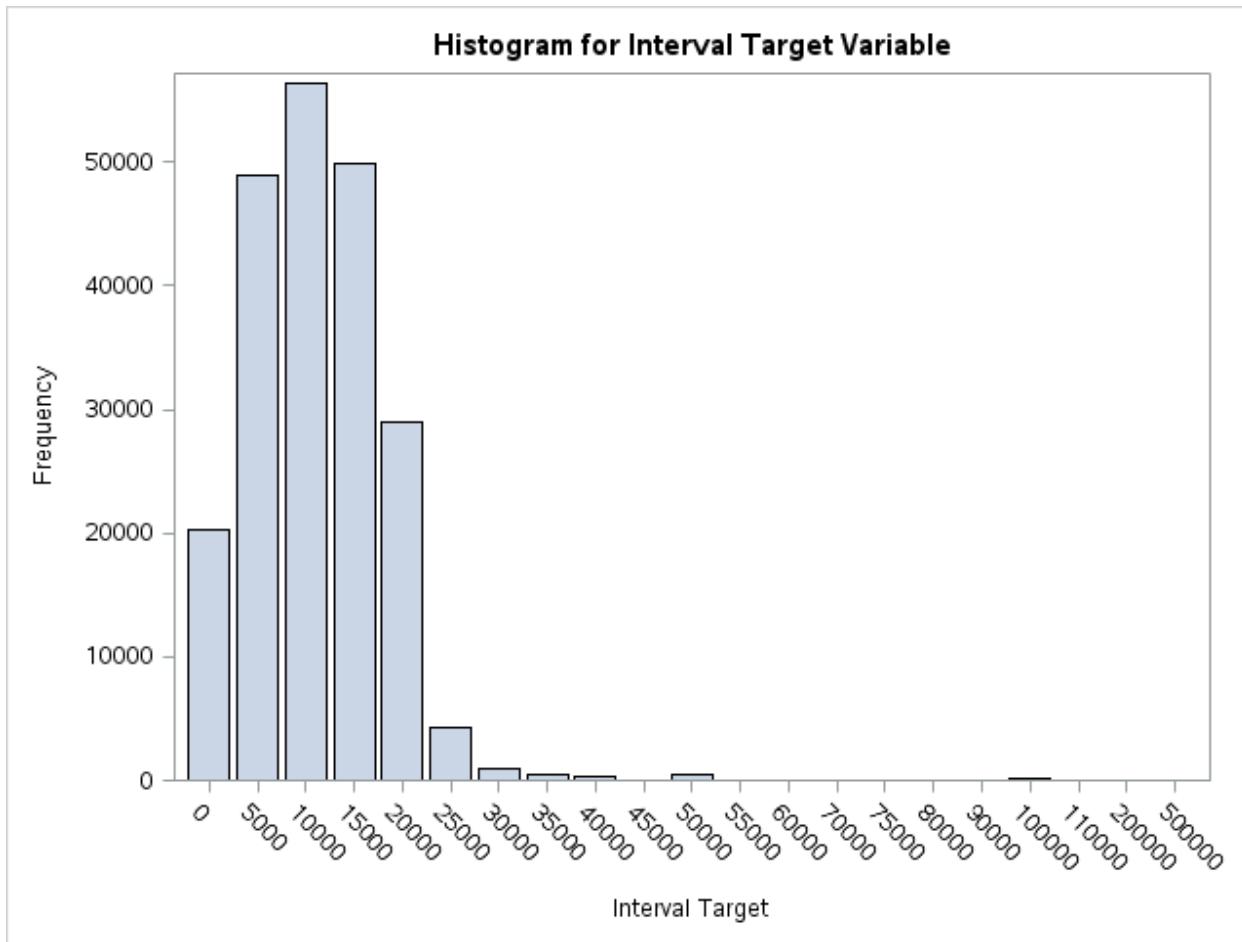
```
proc sgplot data=work.cnt histogramdata;
  vbar binmid / freq=frequency;
  label binmid="Count Target";
run;

proc sgplot data=work.int histogramdata;
  vbar binmid / freq=frequency;
  label binmid="Interval Target";
run;
```

The histograms are below.



The count target is modeled in a later section.



The interval target is skewed to the right. In a later section, several candidate models for skewed interval targets are created and assessed. (The next section focuses on modeling the binary target.)

You are now ready to derive a decision tree. Open the program **mldmbd03d02_DecisionTree.sas**.



Change to interactive mode.

The WHERE clause is used to restrict the tree growth to the training data.

All PROC IMSTAT programs are run in interactive mode, which necessitates that macro variables be defined in the current session. Code at the beginning of each program includes the macro definition file **MLDMBD_Macros.sas**. You must always run this portion of the code before running anything else in the code file. Notice that the macro variables defining groups of input variables are hardcoded for convenience, whereas they were initially derived using the program **mldmbd03d01_GettingStarted.sas**.

The DECISIONTREE statement requests that a decision tree be grown on the active table, **p_model_bank13**. The target variable is listed first. After the slash, options are specified, including the inputs and their measurement levels. Nominal variables must be listed in both the NOMINAL and INPUT options. The MULTVAR option permits an input to be used for splitting multiple times. The LEAFSIZE= option specifies the minimal number of cases in a terminal leaf. The MAXBRANCH= option specifies the maximum number of child nodes for each split. The PRUNE option requests pruning of the tree using the C4.5 algorithm. The CFLEV= option controls the aggressiveness of pruning. Values between 0 and 1 are allowed. Small values indicate more aggressive pruning. The SAVE= option saves the tree rules for later use for scoring the in-memory table **Tree1**.

```

proc imstat;
  table LASRlib.p model_bank13(tag="&TagString");
  where _PartInd_=1;
  decisiontree b tgt /
    nominal=(&catvars)
    input=(&rfms &r_demogs &catvars)
    multvar
    leafsize=100
    maxbranch=2
    nbins=2      /*---- Default is 2 ---- */
    prune
    cflev=0.9   /*---- Affects pruning --*/
    save=Tree1
    ;
run;

```

A partial result summary for the decision tree is produced.

The IMSTAT Procedure									
Tree Node Information for 51 nodes in Table USER.STUDENT2.HPADATA.P_MODEL_BANK13									
Node Number	Tree Level	Parent Node	Parent	Node Type	Node Name	Gain Ratio	Number of Observations	Target Value	LogWorth
0	0	-1		NUM	rfm11	0.0568163814	531423	0	307.65265557
1	1	0	rfm11	NUM	rfm7	0.0535166214	528046	0	307.65265557
2	1	0	rfm11	NUM	rfm9	0.1097698856	3377	1	110.69431139
3	2	1	rfm7	NUM	rfm9	0.0439600996	499604	0	307.65265557
4	2	1	rfm7	NUM	rfm9	0.0824652506	28442	0	242.31802115
5	2	2	rfm9	NUM	rfm5	0.0246708361	1976	1	10.890504084
6	2	2	rfm9	CLASS	cat_input1	0.0222490701	1401	0	6.56072168
7	3	3	rfm9	NUM	r_demog_homeval	0.0464859507	469204	0	307.65265557
8	3	3	rfm9	NUM	rfm5	0.063172897	30400	0	18.648018986
9	3	4	rfm9	NUM	r_demog_inc	0.0645700416	26638	0	54.28835422
10	3	4	rfm9	CLASS	cat_input2	0.0087450264	1804	1	5.5345851732
11	3	5	rfm5	NUM	rfm10	0.0118126349	1622	1	4.5878153206
12	3	5	rfm5	CLASS	cat_input2	0.028181442	354	1	3.591507541
13	3	6	cat_input1	NUM	rfm5	0.0120963445	1190	0	3.0940720811
14	3	6	cat_input1	LEAF	b_tgt	0	211	0	0
15	4	7	r_demog_homeval	NUM	rfm8	0.0279748856	446329	0	32.155794441
16	4	7	r_demog_homeval	CLASS	cat_input1	0.0870488405	22875	0	307.65265557
17	4	8	rfm5	CLASS	cat_input1	0.0226877472	30284	0	150.40871734
18	4	8	rfm5	LEAF	b_tgt	0	116	1	0
19	4	9	r_demog_inc	NUM	r_demog_homeval	0.0486040273	26282	0	83.948786334
20	4	9	r_demog_inc	NUM	demog_genf	0.0155935811	356	1	2.223658274
21	4	10	cat_input2	NUM	rfm5	0.0011338778	892	1	0.5945775982
22	4	10	cat_input2	CLASS	cat_input2	0.0018334015	912	1	0.8879877258
23	4	11	rfm10	CLASS	cat_input1	0.0131389495	1340	1	2.7131266388
24	4	11	rfm10	CLASS	cat_input2	0.0516433597	282	1	4.9710323702
25	4	12	cat_input2	NUM	demog_age	0.0246333176	215	1	2.1715682773

In the following code, the WHERE clause restricts the active table to the training subset. The DECISIONTREE statement with the TREETAB=TREE1 option uses the saved tree to score the active table. The ASSESS option adds the predicted probabilities to the scored data. The TEMPTABLE option requests that the scored table be added to the LASR Analytic Server.

```
table LASRlib.p_model_bank13(tag="&TagString");
where PartInd =1;
decisiontree b_tgt/
  treetab=Tree1
  assess
  temptable
  vars=(b_tgt);
run;
```

The temporary table created above is printed below via the `&_tempscore_` macro variable.

```
table LASRlib.&_tempscore_;
fetch;
run;
```

Temporary Table Information for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13														
Statement		DECISIONTREE												
Temporary Table		_T_A8815008_7F4710120B78												
Table Type		DECISIONTREE												
Selected Records from Table _T_A8815008_7F4710120B78														
b_tgt	_TargetName_	_TargetLevel_	_Missit_	_NumNodes_	_NodeList0_	_NodeList1_	_NodeList2_	_NodeList3_	_NodeList4_	_NodeList5_	_DT_Leafid_	_DT_Level_	_DT_P_	
1.000000	0	0	1.000000	6.000000	0	1.000000	4.000000	9.000000	19.000000	35.000000	35.000000	0	0.563094	
1.000000	0	0	1.000000	6.000000	0	1.000000	4.000000	9.000000	19.000000	35.000000	35.000000	1	0.436008	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	15.000000	31.000000	31.000000	0	0.482400	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	0	0.845384	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	1	0.154616	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	0	0.845384	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	1	0.154616	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	0	0.845384	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	1	0.154616	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
1.000000	0	0	1.000000	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	0	0.517800	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	16.000000	31.000000	31.000000	1	0.482400	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	0	0.845384	
0 0	0	0	0	6.000000	0	1.000000	3.000000	7.000000	15.000000	29.000000	29.000000	1	0.154616	

Notice that there is a row in the data for each level of the target. This needs to be accounted for when doing assessment. The column `_DT_Level_` indicates which value of the target is associated with the given row.

End of Demonstration

ASSESS Statement and Selected Options

```
ASSESS <variable-list> / y=response-variable <option(s)>;
```

Options

- EVENT=“event value” specifies the event value.
- NBINS=*n* specifies the number of bins to summarize the assessment statistics.
- GROUPBY=(*variable-list*) does BY-group processing.
- ODS OUTPUT *object*=*libref.data-set-name* can be used to save tables on the SAS server for reporting and plotting.
- Classification model assessment tables include lift data and ROC plot information.
- Regression models assessment tables include the mean of predicted and actuals for each bin.

91

The GROUPBY option enables you to build and assess models for different segments.



Assessing a Decision Tree

This demonstration illustrates how to assess a decision tree by constructing lift and ROC plots. The demonstration continues with program **mldmbd03d02_DecisionTree.sas**.

The scoring algorithm adds the predicted value in the variable named **_DT_P_**. This is the variable to be assessed. The target variable is assigned using the **Y=** option. In the output data, there is a row for each level of the target. The target value associated with the row is given by the **_DT_Level_** variable. The scored data must be restricted to the row that corresponds to the **EVENT=** value. This is done with the **WHERE** statement. The assessment data are saved in the data set **work.trainlift**.

```
ods output liftinfo=work.trainlift;
ods output rocinfo=work.trainroc;
table LASRlib.&_tempscore_;
assess _DT_P_ /
  y=b tgt
  event='1'
  nbins=20
  step=0.01;
where strip(_DT_Level_) eq '1';
run;
```

The STRIP function removes leading and trailing blanks from a character variable, which is required because **_DT_Level_** is a character variable padded with blanks. The NBINS=20 option specifies 20 bins for calculating lift and other assessment statistics. The STEP=0.01 option specifies an interval step of 0.01 for the ROC chart so that sensitivity and 1 minus specificity pairs are calculated for every 1% of the data ordered by the decision tree scores. Selected lift data are given below. The Depth column in the table below shows the depth associated with each of the 20 bins.

Lift Information for Table _T_A8815008_7F4710120B78																				
Column	Event	Depth	Value	Number of Observations	Number of Events	Number of Events (BEST)	% Captured Response (ACTUAL)	% Captured Response (BEST)	Lift (ACTUAL)	Lift (BEST)	Cumulative % Captured Response (ACTUAL)	Cumulative % Captured Response (BEST)	Cumulative % Lift (ACTUAL)	% Response (ACTUAL)	% Response (BEST)	Cumulative % Response (ACTUAL)	Cumulative % Response (BEST)	Gain (ACTUAL)	Gain (BEST)	
_DT_P_-	1	5.0000	0.4573	26572	14330	26572	13.3875	24.8248	2.8775	4.9650	13.3875	24.8248	2.8775	4.9850	53.9280	100.00	53.9280	100.00	1.6775	3.9650
_DT_P_-	1	10.0000	0.4390	26572	12049	26572	11.2562	24.8248	2.2512	4.9650	24.5437	49.6497	2.4644	4.9850	49.6552	100.00	49.6552	100.00	1.4644	3.9650
_DT_P_-	1	15.0000	0.2747	26572	10740	26572	10.0341	24.8248	2.0068	4.9650	34.5778	74.4745	2.3119	4.9850	40.4195	100.00	46.5633	100.00	1.3119	3.9650
_DT_P_-	1	20.0000	0.1545	26572	4204.30	26572	3.9839	24.8248	0.7968	4.9650	38.6617	99.2993	1.9331	4.9850	15.0481	100.00	38.9345	100.00	0.9331	3.9650
_DT_P_-	1	25.0000	0.1546	26572	4108.46	750.00	3.8383	0.7007	0.7677	0.1401	42.5000	100.00	1.7000	4.0000	15.4616	2.8225	34.2399	80.5645	0.7000	3.0000
_DT_P_-	1	30.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	46.3383	100.00	1.5448	3.3333	15.4616	0	31.1102	67.1371	0.5446	2.3333
_DT_P_-	1	35.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	50.1767	100.00	1.4338	2.8571	15.4616	0	28.8747	57.5481	0.4336	1.8571
_DT_P_-	1	40.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	54.0150	100.00	1.3504	2.5000	15.4616	0	27.1981	50.3524	0.3504	1.8000
_DT_P_-	1	45.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	57.8533	100.00	1.2856	2.2222	15.4616	0	25.8940	44.7581	0.2856	1.2222
_DT_P_-	1	50.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	61.6916	100.00	1.2338	2.0000	15.4616	0	24.8508	40.2823	0.2338	1.0000
_DT_P_-	1	55.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	65.5299	100.00	1.1915	1.8182	15.4616	0	23.9972	36.8202	0.1915	0.8182
_DT_P_-	1	60.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	69.3682	100.00	1.1561	1.6567	15.4616	0	23.2859	33.5685	0.1561	0.6667
_DT_P_-	1	65.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	73.2065	100.00	1.1263	1.5385	15.4616	0	22.0640	30.9683	0.1263	0.5385
_DT_P_-	1	70.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	77.0449	100.00	1.1008	1.4280	15.4616	0	22.1682	28.7730	0.1008	0.4280
_DT_P_-	1	75.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	80.8832	100.00	1.0784	1.3333	15.4616	0	21.7210	26.8548	0.0784	0.3333
_DT_P_-	1	80.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	84.7215	100.00	1.0560	1.2500	15.4616	0	21.3298	25.1784	0.0560	0.2500
_DT_P_-	1	85.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	88.5598	100.00	1.0419	1.1765	15.4616	0	20.9846	23.6954	0.0418	0.1765
_DT_P_-	1	90.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	92.3981	100.00	1.0266	1.1111	15.4616	0	20.6778	22.3790	0.0266	0.1111
_DT_P_-	1	95.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	96.2365	100.00	1.0130	1.0528	15.4616	0	20.4033	21.2012	0.0130	0.0526
_DT_P_-	1	100.0000	0.1364	26555	4026.41	0	3.7635	0	0.7627	0	100.00	100.00	1.0000	15.1701	0	20.1418	20.1418	0	0	

To make the table easier to read, only the leftmost columns are included in the table below.

Lift Information for Tab											
Column	Event	Depth	Value	Number of Observations	Number of Events	Number of Events (BEST)	% Captured Response (ACTUAL)	% Captured Response (BEST)	Lift (ACTUAL)	Lift (BEST)	
_DT_P_	1	5.0000	0.4573	26572	14330	26572	13.3875	24.8248	2.6775	4.9650	
_DT_P_	1	10.0000	0.4360	26572	12048	26572	11.2562	24.8248	2.2512	4.9650	
_DT_P_	1	15.0000	0.2747	26572	10740	26572	10.0341	24.8248	2.0068	4.9650	
_DT_P_	1	20.0000	0.1546	26572	4264.30	26572	3.9839	24.8248	0.7968	4.9650	
_DT_P_	1	25.0000	0.1546	26572	4108.46	750.00	3.8383	0.7007	0.7677	0.1401	
_DT_P_	1	30.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	35.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	40.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	45.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	50.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	55.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	60.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	65.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	70.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	75.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	80.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	85.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	90.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	95.0000	0.1546	26572	4108.46	0	3.8383	0	0.7677	0	
_DT_P_	1	100.00	0.1364	26555	4028.41	0	3.7635	0	0.7527	0	

A partial listing of the ROC data is given below. The cutoff ranges from 0.01 to 1.00, reflecting the STEP=0.01 step size.

ROC Information for Target									
Column	Event	CutOff	True Positives	False Positives	False Negatives	True Negatives	Sensitivity	Specificity	
_DT_P_	1	0	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.01000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.02000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.03000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.04000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.05000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.06000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.07000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.08000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.09000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.1000	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.1100	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.1200	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.1300	107038	424385	0	0	1.0000	0	
_DT_P_	1	0.1400	106460	420724	578.00	3661.00	0.9946	0.008627	
_DT_P_	1	0.1500	106460	420724	578.00	3661.00	0.9946	0.008627	
_DT_P_	1	0.1600	37498	43665	69540	380720	0.3503	0.8971	

The code below scores and assesses the validation data as defined by the WHERE clause. The assessment data are stored in the **work.validlift** data set.

```





```

```





```

The summary statistics table follows:

Summary Statistics for Table _T_71CF089F_7F02C818EB78										
Column	Min	Max	N	Sum	Mean	Std Dev.	Std Error	Coefficient of Variation	Number Missing	
residual	-0.9722	0.8636	528615	-1695.5009	-0.00321	0.3812	0.000524	-11886	0	
res2	0.000772	0.9452	528615	76832.1	0.1453	0.2348	0.000323	161.57	0	
absres	0.02778	0.9722	528615	154459.146	0.2922	0.2449	0.000337	83.8078	0	

Notice that the mean for **res2** is only average squared error.

The following code calculates error diagnostics from the summary table:

```

data SASlib.validstats;
attrib Model length=$20 label="Scoring Model";
set work.summary end=lastobs;
retain Nobs MSE RMSE MEANABS ResVar MeanResid MinResid MaxResid;
keep Model Nobs MSE RMSE MEANABS ResVar MeanResid MinResid MaxResid;
if (upcase(column)='RESIDUAL') then do;
  ResVar=Std*Std;
  MinResid=Min;
  MaxResid=Max;
  MeanResid=Mean;
  Nobs=N;
end;
else if (upcase(column)='RES2') then do;
  MSE=Mean/N;
  RMSE=sqrt(MSE);
end;
else if (upcase(column)='ABSRES') then do;
  MEANABS=Mean/N;
end;
if (lastobs) then do;
  Model="Decision Tree";
  output;
end;
run;

proc print data=SASlib.validstats;
run;

```

The final table of error diagnostics appears below.

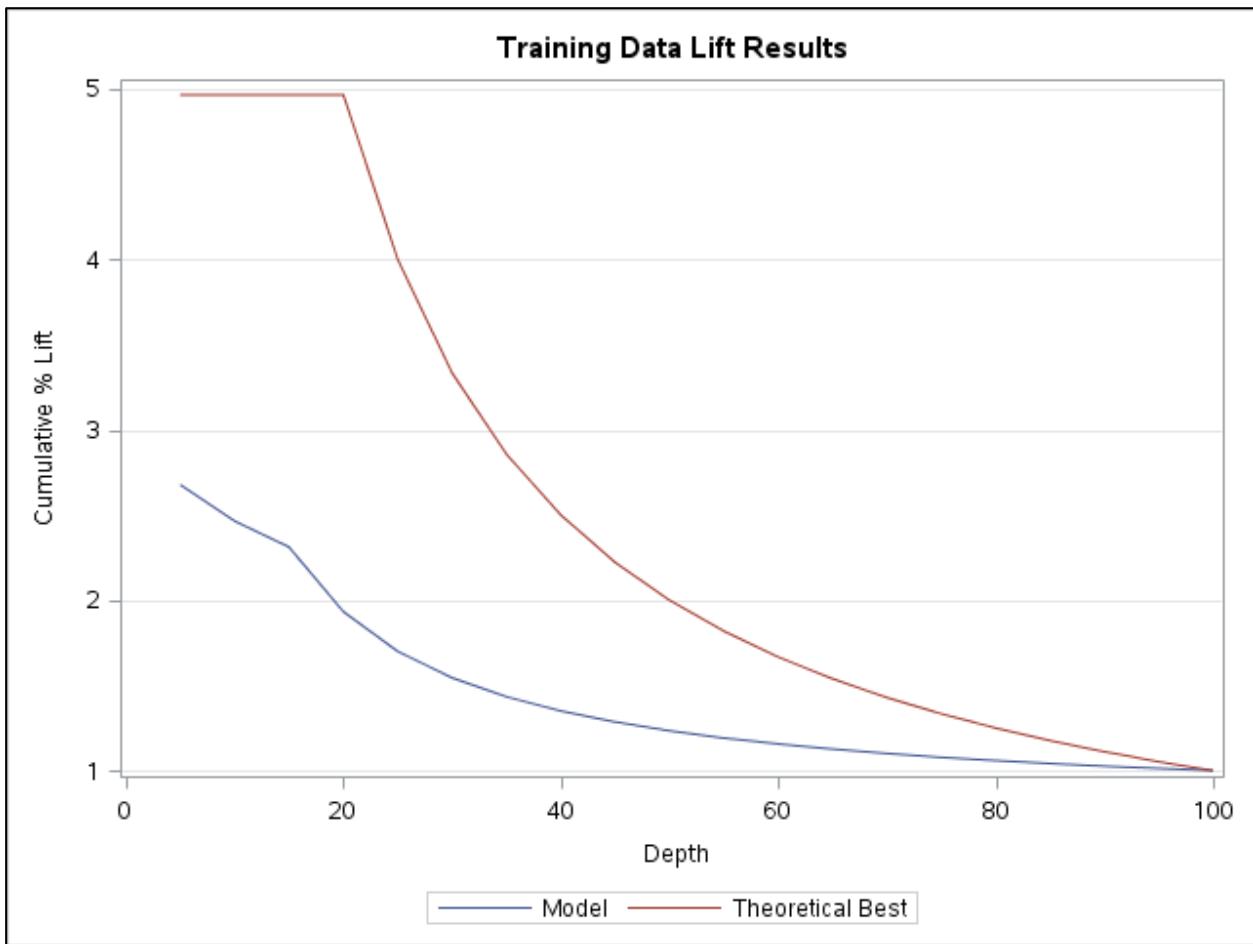
Residual Diagnostics									
Obs	Model	Nobs	MSE	RMSE	MEANABS	ResVar	MeanResid	MinResid	MaxResid
1	Decision Tree	528615	.000000275	.000524363	.000000553	0.14534	-.00320744	-0.97222	0.86365

Because there are no model degrees of freedom, RMSE (Root Mean Square Error) is only the square root of average square error.

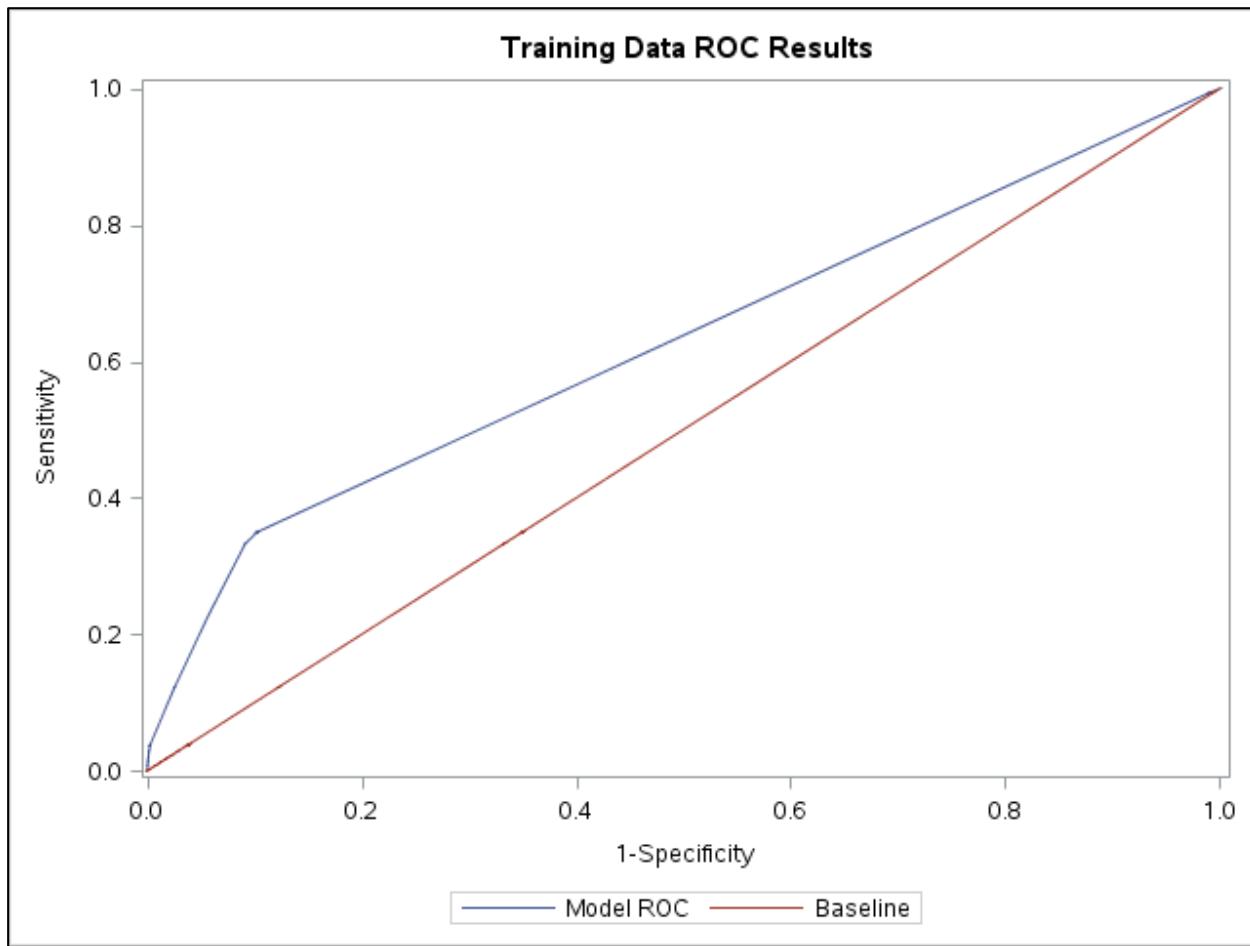
Using the saved assessment data, lift and ROC plots can be produced using the code below.

```
proc sgplot data=work.trainlift;
    series y=Cumlift x=depth / legendlabel="Model" name="line1";
    series y=cumliftbest x=depth /
        legendlabel="Theoretical Best" name="line2";
    yaxis label="Cumulative % Lift" grid;
    keylegend "line1" "line2" / location=outside position=bottom;
    label Cumlift="Cumulative % Lift";
run;
title1 "Training Data ROC Results";
data work.plotroc;
    set work.trainroc;
    onemspcf=1-specificity;
run;
proc sgplot data=work.plotroc;
    series y=sensitivity x=onemspcf / legendlabel="Model ROC"
        name="line1";
    series y=sensitivity x=sensitivity / legendlabel="Baseline"
        name="line2";
    keylegend "line1" "line2" / location=outside position=bottom;
    label sensitivity="Sensitivity"
        onemspcf="1-Specificity";
run;
```

The lift plot for the training data is shown below.



The ROC plot is shown below.

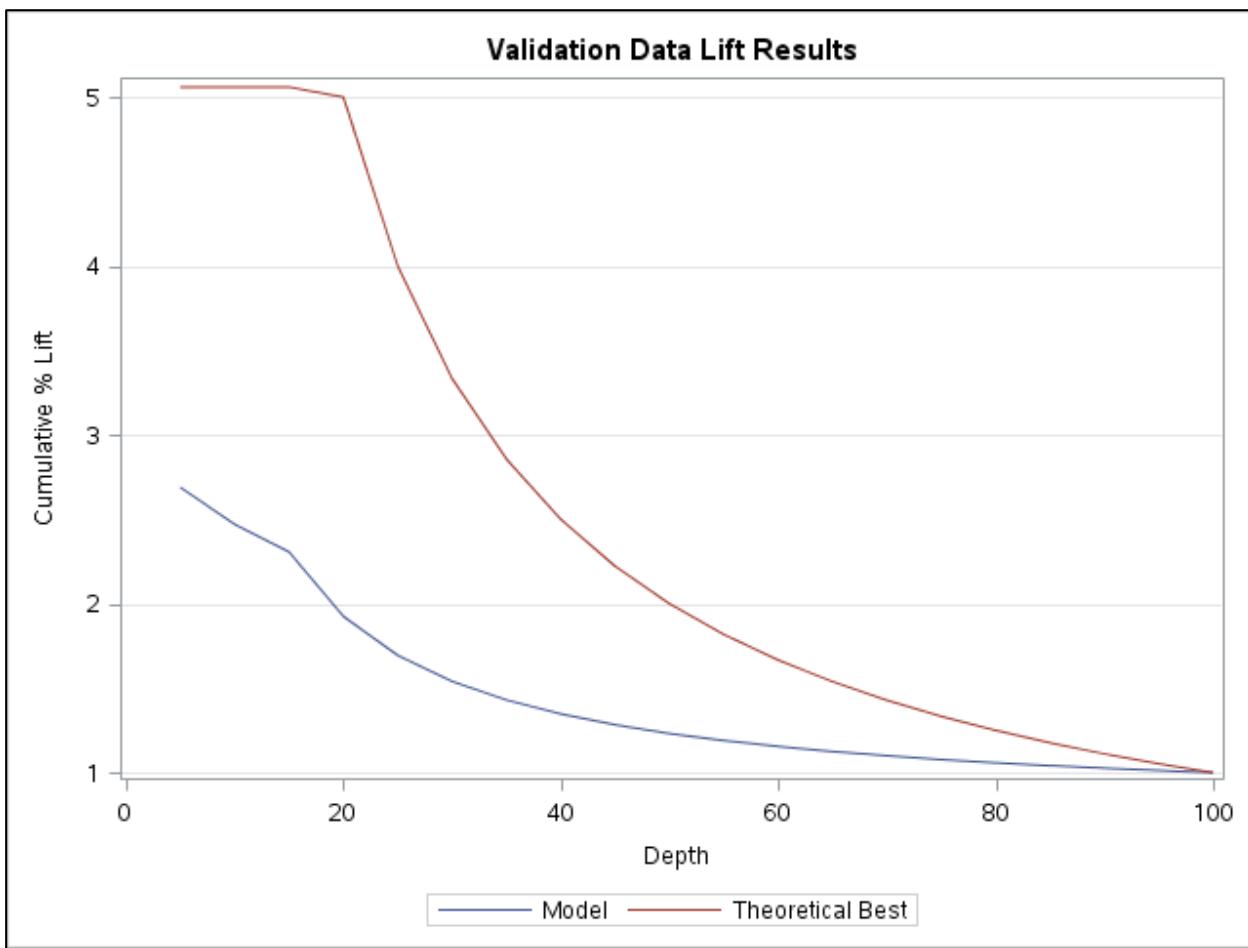


The 45-degree red baseline represents the result for a “random guess” predictive algorithm versus the blue line on top that corresponds to model predictions.

The lift plot for the validation data is produced with the code below.

```
proc sgplot data=work.validlift;
  series y=Cumlift x=depth / legendlabel="Model" name="line1";
  series y=cumliftbest x=depth /
    legendlabel="Theoretical Best" name="line2";
  yaxis label="Cumulative % Lift" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
  label Cumlift="Cumulative % Lift";
run;
```

The lift chart is shown below.



The validation ROC curve is produced by the following code:

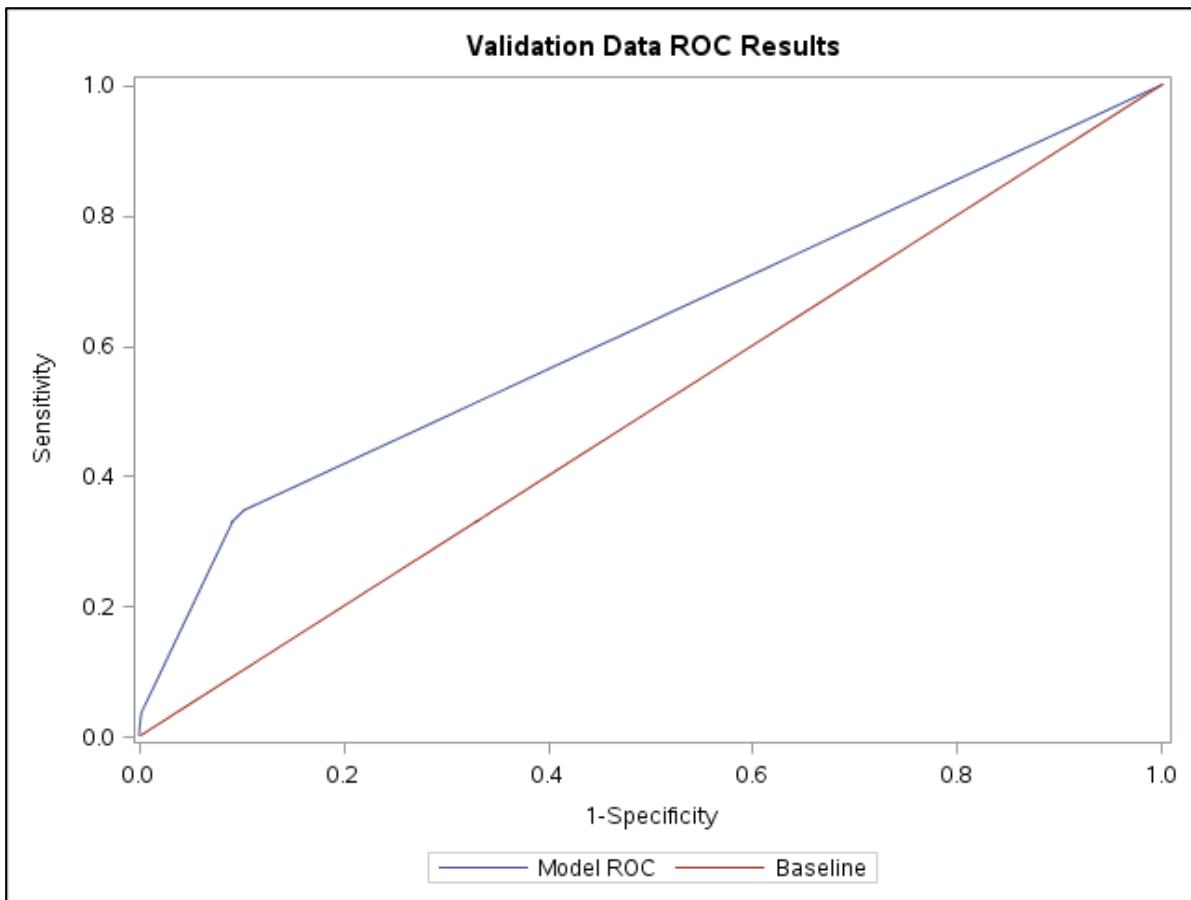
```

title1 "Validation Data ROC Results";
data work.plotroc;
  set work.validroc;
  onemspcf=1-specificity;
run;

proc sgplot data=work.plotroc;
  series y=sensitivity x=onemspcf /
    legendlabel="Model ROC" name="line1";
  series y=sensitivity x=sensitivity /
    legendlabel="Baseline" name="line2";
  keylegend "line1" "line2" / location=outside position=bottom;
  label sensitivity="Sensitivity"
        onemspcf="1-Specificity";
run;

```

The ROC plot appears below.



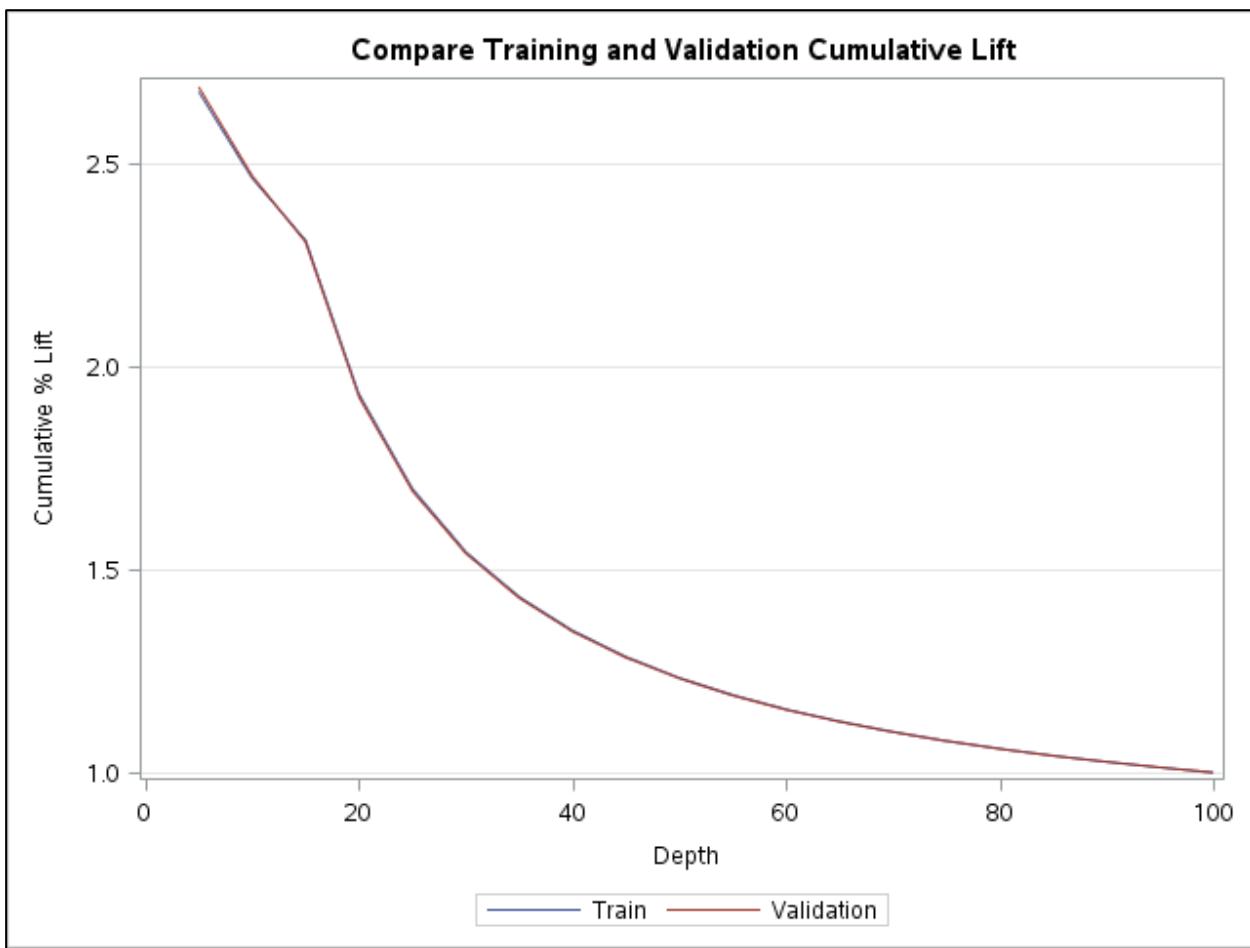
Often, it is useful to overlay the plots for the training and validation data on the same plot. This can be done with the code below.

```

data liftdata;
  merge work.trainlift(keep=depth cumlift
  rename=(cumlift=traincumlift))
    work.validlift(keep=depth cumlift
  rename=(cumlift=validcumlift));
  by depth;
  label traincumlift="Cum % Lift (Train)"
    validcumlift="Cum % Lift (Valid)";
run;
title1 "Compare Training and Validation Cumulative Lift";
proc sgplot data=work.liftdata;
  series y=traincumlift x=depth / legendlabel="Train" name="line1";
  series y=validcumlift x=depth /
    legendlabel="Validation" name="line2";
  yaxis label="Cumulative % Lift" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
  * label traincumlift="Cum % Lift"
    validcumlift="Cum % Lift";
run;
title1;

```

The plot is below.



An alternative to a single decision tree is a forest of trees.

End of Demonstration

An Introduction to Random Forests (Self-Study)

Objectives

- Describe forest predictive models.
- Define variable importance in a forest.

94

There are several random forest algorithms. The approach described in this section is implemented by the HPFOREST procedure in SAS Enterprise Miner. A different algorithm is used by the RANDOMWOODS statement in PROC IMSTAT.

Forest

- An *ensemble model* is an aggregation of more than one model where the final prediction of the model is a combination of the predictions from the component models of the ensemble.
- A *forest model* is an ensemble of classification or regression trees.
- Forest models were developed to overcome the instability that a single classification or regression tree exhibits with minor perturbations of the training data.

95

Seeing the Forest through the Trees...

Trees in the forest differ from each other in two ways:

- Training data for a tree is a sample without replacement from all observations.
- Input variables considered for splitting a node are randomly selected from available inputs. Only the variable most associated with the target is split for that node.

96

The trees that make up a forest differ from each other in two ways:

- The training data for a tree is a sample without replacement from all observations that were originally training data for the forest.
- The input variables considered for splitting for any given node are selected randomly from all available inputs.

Among these variables, only the variable most associated with the target is used when forming a split. This means that each tree is created on a sample of the inputs and from a sample of observations. This process, repeated many times, creates a more stable model than a single tree. The reason for using a sample of the data to construct each tree is because when less than all available observations are used, the generalization error is often improved. A different sample is taken for each tree.

Leaves = Boolean Rules

If $X_1 \in \{values\}$ and $X_2 \in \{values\}$, then $\hat{Y} = value$.

<u>Leaf</u>	<u>X1</u>	<u>X2</u>	<u>Predicted Y</u>
1	<6.5	<.51	.22
2	<6.5	[.51, .63)	.19
3	<6.5	[.63, .67)	.27
4	[6.5, 6.9)	<.67	.27
5	<6.9	$\geq .67$.14
6	[6.9, 7.4)	<.66	.33
7	≥ 7.4	<.66	.46
8	≥ 6.9	$\geq .66$.16

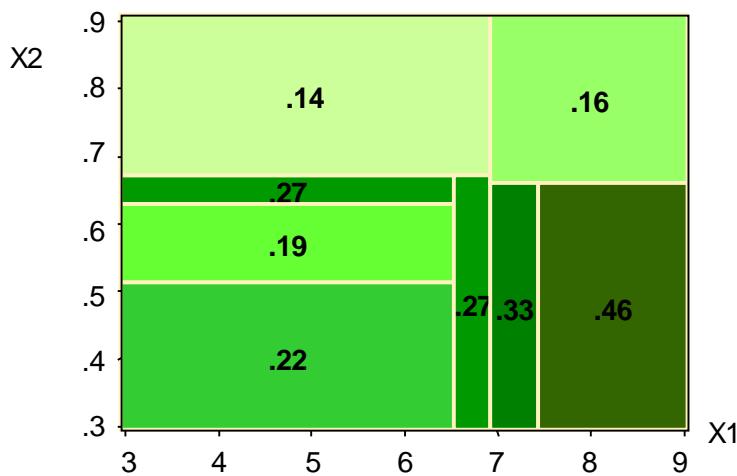
97

The path to each leaf can be expressed as a Boolean rule. The rules take this form:

If the inputs $\in \{region\}$ of the input space, then the predicted value = value.

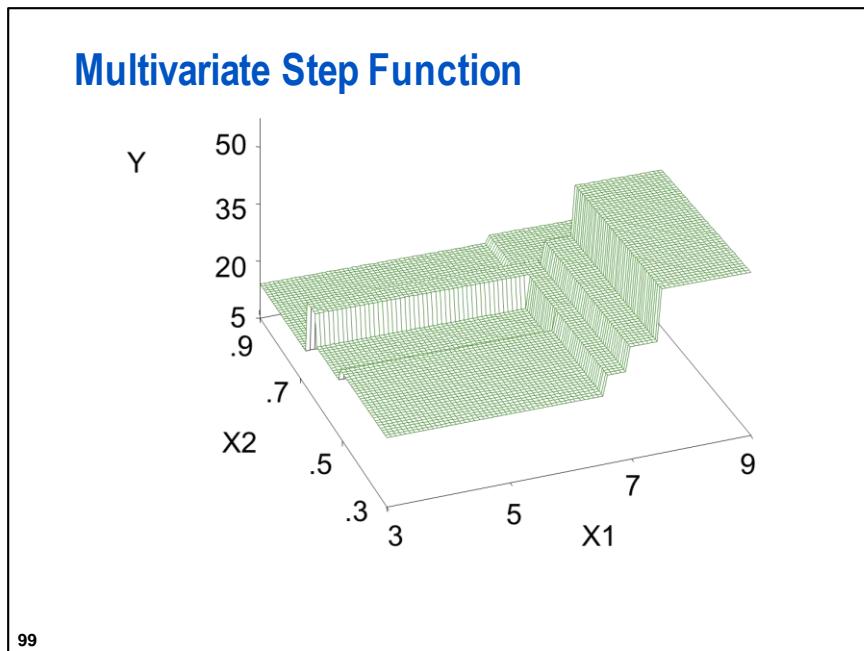
The regions of the input space are determined by the split values. For interval-scaled inputs, the boundaries of the regions are perpendicular to the split variables. Consequently, the regions are intersections of subspaces defined by a single splitting variable.

Partitioned Input Space

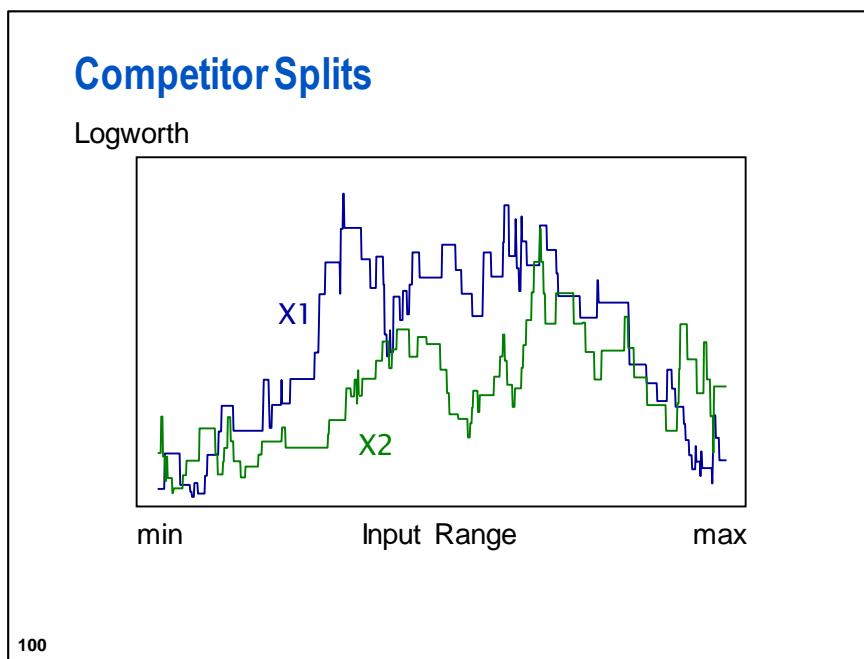


98

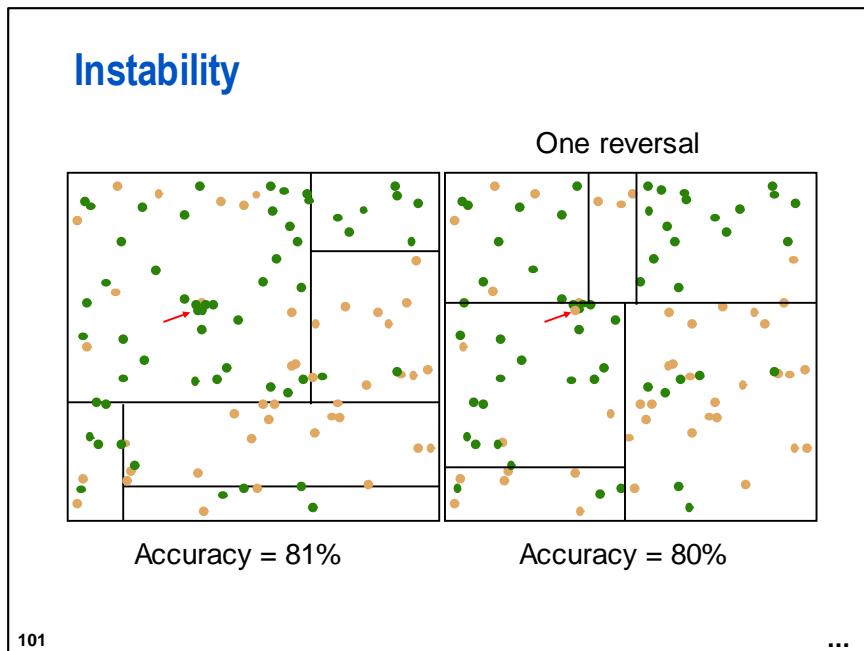
The leaves of the decision tree partition the input space into rectilinear regions. The predicted target has a different constant value in each partition. Consequently, the fitted model is a multivariate step function.



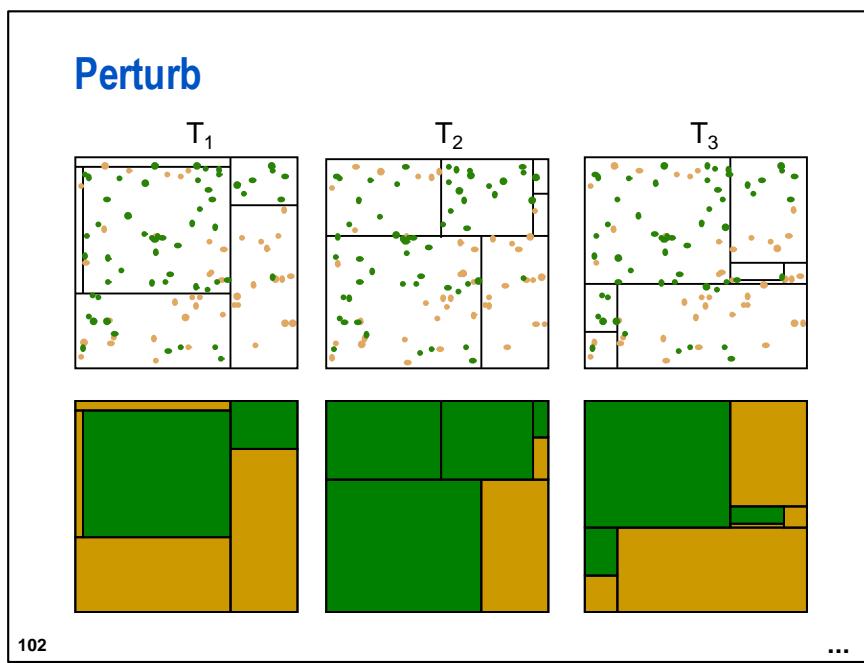
The surface is a piecewise constant and not joined continuously at the boundaries. A step function is highly flexible. It is capable of modeling nonlinear trends.



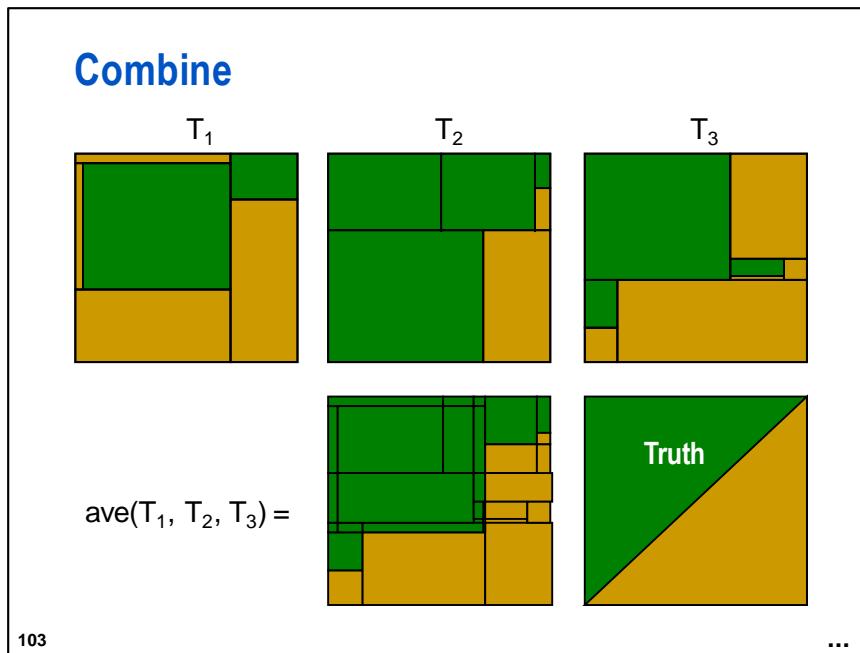
Decision trees are unstable models. That is, small changes in the training data can cause large changes in the topology of the tree. However, the overall performance of the tree remains stable (Breiman et al. 1984). The instability results from the large number of univariate splits considered and the fragmentation of the data. At each split, there are typically a number of splits on the same and different inputs that give similar performance (competitor splits). A small change in the data can easily result in a different split being chosen. This in turn produces different subsets in the child nodes. The changes in the data are even larger in the child nodes. The changes continue to cascade down the tree.



In the above example, changing the class label of one case resulted in a completely different tree with nearly the same accuracy.



Methods have been devised to take advantage of the instability of trees to create models that are more powerful. *Perturb and combine* (P & C) methods generate multiple models by manipulating the distribution of the data or altering the construction method and then averaging the results (Breiman 1998). Any unstable modeling method can be used, but trees are most often chosen because of their speed and flexibility.



An ensemble model is the combination of multiple models. The combinations can be formed by

- voting on the classifications
- weighted voting where some models have more weight
- averaging (weighted or unweighted) the predicted values.

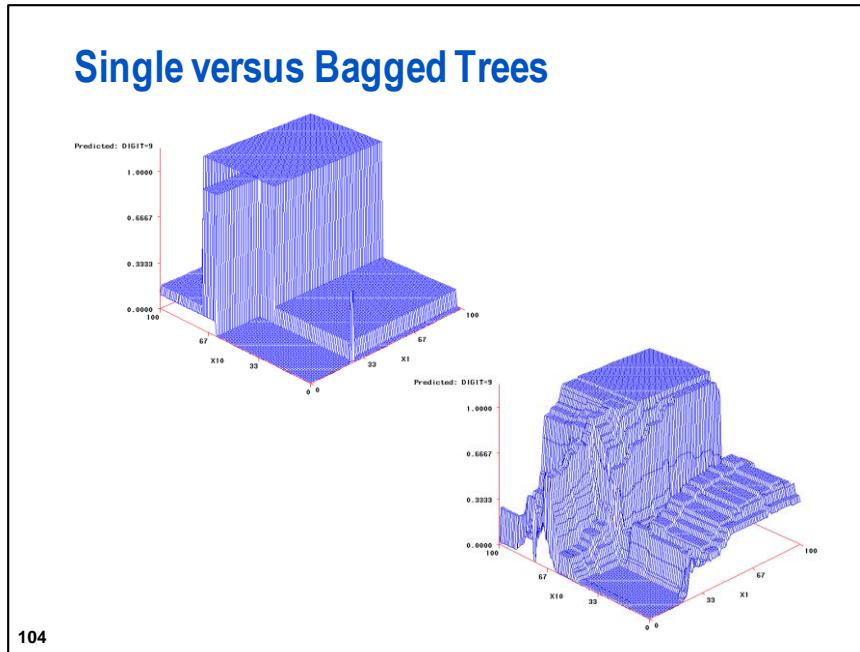
Ensemble methods are a very active area of research in the fields of machine learning and statistics. Many other P & C methods were devised.

The attractiveness of P & C methods is their improved performance over single models. Bauer and Kohavi (1999) demonstrated the superiority of P & C methods with extensive experimentation. One reason why simple P & C methods give improved performance is variance reduction. If the base models have low bias and high variance, then averaging decreases the variance. In contrast, combining stable models can negatively affect performance. The reasons why adaptive P & C methods work go beyond simple variance reduction and are the topic of much current research (Breiman 1998). Graphical explanations show that ensembles of trees have decision boundaries of much finer resolution than would be possible with a single tree (Rao and Potts 1997).

A new case is scored by running it down the multiple trees and averaging the results. Multiple models need to be stored and processed. The simple interpretation of a single tree is lost.

Bagging goes a long way toward making a silk purse out of a sow's ear, especially if the sow's ear is twitchy. ...What one loses, with the trees, is a simple and interpretable structure. What one gains is increased accuracy.

— Breiman (1996)



To smooth the all-or-none bins of a single decisions tree, bagging smooths the prediction surface.

Bagging (bootstrap aggregation) is a P & C method that generally works as follows (Breiman 1996):

1. Draw B bootstrap samples.

A bootstrap sample is a random sample of size n drawn from the empirical distribution of a sample of size N . That is, the training data is resampled with replacement. Some of the cases are omitted from the sample, and some cases are represented more than once.

2. Build a tree on each bootstrap sample.

Pruning can be counterproductive (Bauer and Kohavi 1999). Large trees with low bias and high variance are ideal.

3. Vote or average.

For classification problems, take the mean of the posterior probabilities or take the plurality vote of the predicted class. Bauer and Kohavi (1999) found that averaging the posterior probabilities gave slightly better performance than voting. Take a mean of the predicted values for regression.

Breiman (1996) used 50 bootstrap replicates for classification and 25 for regression and for averaging the posterior probabilities. Bauer and Kohavi (1999) used 25 replicates for both voting and averaging.

Forest Algorithm

- *Bagging* is the term for averaging many trees grown on bootstrap samples of the rows of training data. All columns are considered for splitting at every step.
- The forest algorithm does sampling of the rows **and** sampling of the columns at each step.
- The forest algorithm perturbs the training data more than the bagging algorithm. Thus, it produces more variation among the trees in the ensemble.
- Ensembles of a more diverse set of trees often lead to improved predictive accuracy.

105

In a forest, rather than taking bootstrap samples of only the rows, variables are also randomly sampled. This results in a forest, consisting of trees that use different combinations of rows and variables to determine splits. This additional perturbation (beyond bagging) leads to greater diversity in the trees, and better predictive accuracy.

A New Term Is Needed

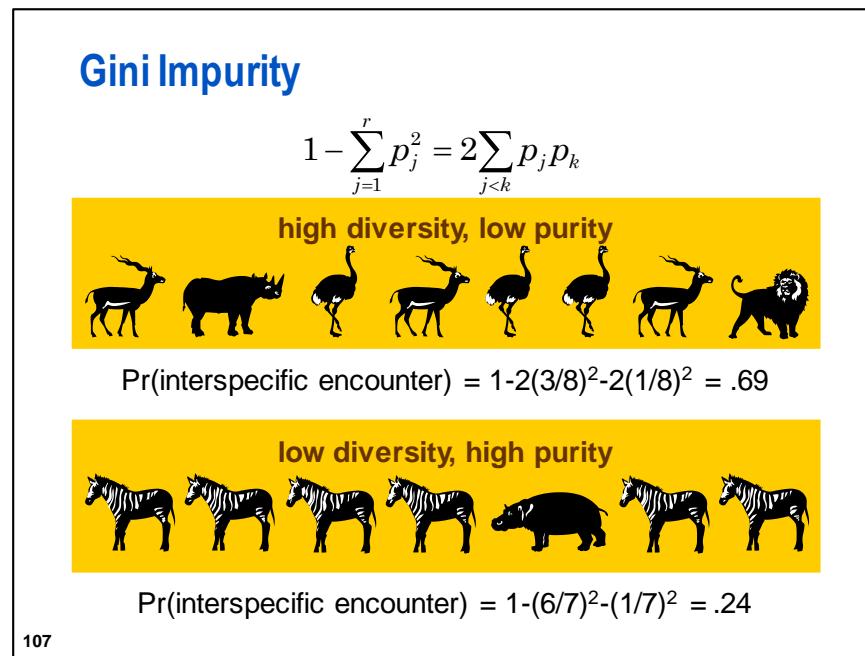
- The out-of-bag sample refers to the training data that is excluded during the construction of an individual tree.
- Observations in the training data that are used to construct an individual tree are the bagged sample.
- Some model assessments, such as the iteration plots, are computed using the out-of-bag sample as well as all the training data.

106

A decision tree in a forest trains on new training data that are derived from the original training data. Training different trees with different training data reduces the correlation of the predictions of the trees, which in turn should improve the predictions of the forest. The training data for an individual tree exclude some of the available data. The data that are withheld from training are called the *out-of-bag sample*. Observations in the training sample are called the *bagged observations*, and the training data for a specific decision are called the *bagged data*. For each individual tree, the out-of-bag sample is used to form predictions. These predictions are more reliable than those from training data.

Model assessment measures, such as misclassification rates, average squared error, and iteration plots, are constructed on both the entire training data set as well as the out-of-bag sample.

Variable Importance in a Forest

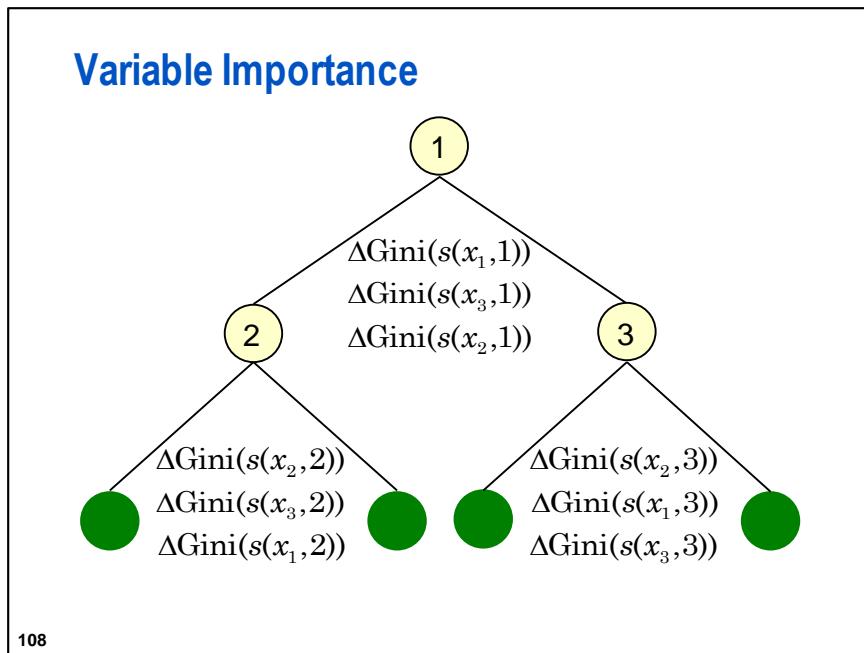


107

The Gini index is a measure of variability for categorical data (developed by the eminent Italian statistician Corrado Gini in 1912). The Gini index can be used as a measure of node impurity where p_1, p_2, \dots, p_r are the relative frequencies of each target class in a node. The ΔGini splitting criteria was proposed by Breiman et al. (1984).

The Gini index can be interpreted as the probability that any two elements of a multi-set, chosen at random (with replacement), are different. A pure node has a Gini index of 0. As the number of evenly distributed classes increases, the Gini index approaches 1.

In mathematical ecology, the Gini index is known as *Simpson's diversity index*. In cryptanalysis, it is 1 minus the *repeat rate* (good discussion of Patil and Taillie, 1982).



Breiman et al (1984) devised a measure of variable importance for trees. It can be particularly useful for tree interpretation.

Let $s(x_j, t)$ be a surrogate split (including the primary split) at the t th internal node using the j th input. Importance is a weighted average of the reduction in impurity for the surrogate splits using the j th input across all the internal nodes in the tree. The weights are the node sizes.

$$\text{Importance}(x_j) = \sum_{t=1}^T \frac{n_t}{n} \Delta(i)(s(x_j), t)$$

In a decision tree, variable importance can be calculated similarly to Breiman et al (1984). The variable importance measure is scaled to be between 0 and 1 by dividing by the maximum importance. Thus, larger values indicate greater importance. Variables that do not appear in any primary or saved surrogate splits have 0 importance.

Summary Points

- Trees automatically handle missing values and variable reduction. Therefore, the input data requires less preparation.
- Forests tend to give better prediction than any specific tree, and often outperform other classes of models.
- Forests are challenging to interpret, but they can be considered an “ideal” model for other models to be compared against.

109

Variable importance measures provide a measure of interpretation for forests.

3.03 Poll

Forests tend to be more stable models than single decision trees.

- True
- False

110

Random Forests in IMSTAT

PROC IMSTAT implements random forests through the RANDOMWOODS statement. A *random forest* is a collection of decision trees. The adjective “random” helps explain that each tree uses a randomization scheme to restrict the choice of inputs, which assures that trees with different inputs are derived. One of the appealing properties of a random forest is that, with the right selection of properties, the resulting predictive algorithm does not overfit the training data.

RANDOMWOODS Algorithm

1. For M input variables, randomly select m variables, and use these m variables to construct the tree. A common practice is to use $m=\text{ceil}(\sqrt{M})$. The value m is supplied using the $M=$ option.
2. For a training data set having N observations, select a fraction f of the N observations at random with replacement. If $f=1$, this omits approximately one-third of the training data not selected, or *out-of-bag* (OOB). The fraction f is supplied using the $\text{BOOTSTRAP}=$ option.
3. Build a tree using the f^*N bootstrap sample and the input variables. Do not prune the tree.
4. Repeat the tree-building steps to derive K different trees. K is specified using the $\text{NTREE}=$ option.
5. To score a new observation, score it with the K trees and then calculate the final score by using a voting scheme.

112

The full tree is determined by stopping rules, such as maximum number of levels and minimum number of observations per leaf.

The number of inputs to select at random, labeled m above, is provided by the $M=$ option. If this option is not specified, the default is as follows:

CEIL(SQRT(*Number of Inputs*))

This is the square root of the number of inputs rounded to the next highest integer.

RANDOMWOODS Statement and Selected Options

RANDOMWOODS *target-variable* / *option(s)*;

- The target is assumed to be binary unless the **REG** option is specified.

Selected Options

- **NOMINAL=(*list*)** specifies the nominal inputs.
- **INPUTS=(*list*)** specifies the input variables, including nominal.
- **M= m** specifies the number of inputs to be randomly selected at each step.
- **SAVE=*data-set-name*** saves the tree details for later scoring and assessment.
- **NTREE= n** specifies the number of trees in the woods.
- **CODE=(*filename*=“*full-path*”)** saves score code.

113

The following RANDOMWOODS syntax specification is extracted from *SAS® LASR Analytic Server 2.5: Reference Guide*.

RANDOMWOODS Statement

The RANDOMWOODS statement builds a random forest of decision trees. Each tree is constructed from a bootstrap sample of the data, drawn with replacement, and is constructed from only a subset of the variables specified in the INPUT= option.

Syntax

RANDOMWOODS *target-variable* </ *options*>;

Required Argument

target-variable

specifies a single column in the in-memory table as the target variable. The variable can be a temporary calculated column.

RANDOMWOODS Statement Options

ADDTREES

requests that the temporary table that is generated by scoring a random forest is enhanced with information about the votes of the individual trees. The process of scoring a random forest means that each tree votes on the predicted value and the predicted value for the forest is obtained by majority vote. This option adds the votes for each tree in the forest. By default, only the overall vote is reported in the temporary table.

ASSESS

specifies that predicted probabilities are added to the temporary result table for the event levels. You can use these predicted probabilities in an ASSESS statement.

BOOTSTRAP=f

specifies the fraction of the data in the bootstrap sample.

Default

$f = 1 - \exp(-1)$

Range

0 to 1

CODE <(code-generation-options)>

requests that the server produce SAS scoring code based on the actions that it performed during the analysis. The server generates DATA step code. By default, the code is replayed as an ODS table by the procedure as part of the output of the statement. More frequently, you might want to write the scoring code to an external file by specifying options.

The scoring code computes the predicted value of the response variable on the data scale (the inverse link scale) and prefixes the name with RF_. For example, if the response variable is Y, the generated code stores the predicted value as RF_Y. The name of the variable is truncated to fit within the SAS name length requirements.

COMMENT

specifies to add comments to the code in addition to the header block. The header block is added by default.

FILENAME='path'

specifies the name of the external file to which the scoring code is written. This suboption applies only to the scoring code itself.

Alias

FILE=

FORMATWIDTH=k

specifies the width to use in formatting derived numbers such as parameter estimates in the scoring code. The server applies the BEST format, and the default format for code generation is BEST20.

Alias

FMTW=

Range

4 to 32

LABELID=ID

specifies a group identifier for group processing. The identifier is an integer and is used to create array names and statement labels in the generated code.

LINESIZE=n

specifies the line size for the generated code.

Alias

LS=

Default

72

Range

64 to 256

NOTRIM

specifies to format the variables using the full format width with padding. By default, leading and trailing blanks are removed from the formatted values.

REPLACE

specifies to overwrite the external file if a file with the specified name already exists. The option has no effect unless you specify the FILENAME= option.

EVENT=("event1" <, "event2">...)

specifies the event names of the target variable. This option is combined with the WEIGHT= option to specify the weight for each specific event. Observations with the specified event are reweighted with the value from the WEIGHT= option. This option is useful for rare-event sampling.

FORMATS="format-specification",...)

specifies the formats for the input variables. If you do not specify the FORMATS= option, the default format is applied for that variable. Enclose each format specification in quotation marks and separate each format specification with a comma.

GAIN

specifies that the splitting criterion is changed to information gain. Typically, this criterion intends to generate trees with more nodes than information gain ratio.

GREEDY

specifies how to perform splitting under specific circumstances.

Assuming that one variable has q levels, when binary splitting is performed and q is less than 15, or option MAXBRANCH > 2 and $q < 12$, all possible binary splits are enumerated and the split with the largest gain or gain ratio is chosen for the variable.

When q is less than 1024 and splitting is not only binary, local greedy searches are applied to determine the optimum local split. Specifically, when the variable is numeric, q levels (similar to q bins) are sorted by value.

When the variable is nominal, the q levels are ordered by random weights. The best binary splitting is applied until the desired number of branches is reached. Only a local optimum can be found with this technique.

For values of $q \geq 1024$, the default k -means clustering algorithm is applied to determine the splits.

IMPUTE

specifies how to treat observations with nonmissing values for the target variable during scoring. When this option is specified, the observed values are used as the predicted values. That is, the observed value is assumed to be known without error. Only the observations with missing values for the target variable are then scored against the random forest, based on their values for the input variables.

This option is useful if you want to replace missing values of a target variable with classified values that are based on the random forest.

INPUT=variable-name**INPUT=(variable-list)**

specifies the variables to use for building the tree. You can add the target variable to the input list if you want to assign a format to the target variable by using the FORMATS= option. Any numeric variable that is not specified in the NOMINAL= option is binned according to the NBINS= specification.

In random forest implementations, all of the input variables do not participate in the construction of the trees. Each tree is built from a subset of the input variables. You can use the M= option to affect the selection of these input variables.

LEAFSIZE= m

specifies the minimal number of observations on each node. When the number of observations on a tree node falls short of the specified leaf size m , the node is changed into a leaf during the building of the tree.

Interaction

Specifying the LEAFSIZE option affects the pruning of the tree.

M= k

specifies the number of input variables used to build a tree. The k variables are selected at random from the list of input variables for each tree. If not specified, then k defaults to the square root of the number of input variables, rounded up to the nearest integer.

MAXBRANCH= n

specifies the maximum number of children (branches) allowed for each level of the tree.

Default

2

MAXLEVEL=*n*

specifies the maximum number of tree levels.

Default

6

NBINS=*k*

specifies the number of bins used in the calculation of the tree. The number of bins affects the accuracy of the tree and increases with *k* at the expense of computing time and memory consumption.

Default

2

NBINSTARGET=*k*

specifies the number of bins to use for a numeric target variable. The number of bins affects the accuracy of the tree. The accuracy increases as values of *k* increase. However, computing time and memory consumption also increase as values of *k* increase. When *k* is greater than zero, the numeric target variable is binned into equally sized bins first and then the bins are used to perform the classification.

Default

0

NOERROR

specifies that the out-of-bag error is not computed when building a random decision forest.

This option is useful to speed up the building process.

NOMINAL=*variable-name***NOMINAL=(***variable-list***)**

specifies the numeric variables to use as nominal variables. Binning is not applied to the specified variables. The target variable is always treated as a nominal variable and does not need to be listed.

NOMISSOBS

specifies to ignore observations that have missing values in the analysis variables when building a decision tree. When scoring a data set, any observations with missing values in the analysis variables for the decision tree are ignored when this option is specified.

When this option is not specified, the RANDOMWOODS statement builds a tree by applying the following policy for missing values:

- For an interval variable, the smallest machine value is assigned.
- For a nominal variable, missing values are represented by a separate level.

NOPREPARSE

prevents pre-parsing and pre-generating the program code that is referenced in the CODE= option. If you know the code is correct, you can specify this option to save resources. The code is always parsed by the server, but you might get more detailed error messages when the procedure parses the code rather than the server. The server assumes that the code is correct. If the code fails to compile, the server indicates that it could not parse the code, but not where the error occurred.

Alias

NOPREP

NTREE=n

specifies the number of trees to build for the random forest.

Default

1

REG

specifies to build the random decision forest using regression trees. Minimal cost-complexity pruning is applied to prune the trees.

SAVE=*table-name*

saves the result table so that you can use it in other IMSTAT procedure statements like STORE, REPLAY, and FREE. The value for *table-name* must be unique within the scope of the procedure execution. The name of a table that is freed with the FREE statement can be used again in subsequent SAVE= options.

SCOREDATA=*table-name*

specifies the in-memory table that contains the scoring data. The table must exist in memory on the server. The RANDOMWOODS statement in the IMSTAT procedure does not transfer a local data set to the server.

If you do not specify a table name for this option, the active table is used as the scoring input.

SEED=s

specifies the random number seed for the random number generator in the server. The default value, zero, implies that the random number stream is based on the computer clock. Negative seed values also lead to random number streams that are based on the computer clock. If you want a reproducible random number sequence between runs, specify a value that is greater than zero.

Default

0

TEMPEXPRESS="*SAS expressions*"**TEMPEXPRESS=***file-reference*

specifies either a quoted string that contains the SAS expression that defines the temporary variables or a file reference to an external file with the SAS statements.

Alias

TE=

TEMPNAMES=*variable-name***TEMPNAMES=(***variable-list***)**

specifies the list of temporary variables for the request. Each temporary variable must be defined through SAS statements that you supply with the TEMPEXPRESS= option.

Alias

TN=

TEMPTABLE

specifies to save the results of the RANDOMWOODS statement in a temporary table.

If you build a random forest, the temporary table contains information about the forest.

If you score a random forest, the temporary table contains the predicted values for the observations in the input table that you scored. The temporary table also contains the variables that you specified to transfer from the input table and other statistics. This option is required when you perform scoring so that you can access the predictions for each observation. The IMSTAT procedure then displays the name of the table and stores it in the **_TEMPSCORE_** macro variable, provided that the scoring action was successful. Observations from the table that you scored can be transferred to the temporary table using the VARS= option.

TIMEOUT=s

specifies the maximum number of seconds that the statement should run in the server. If the computation does not complete before the time-out is reached, execution stops and the server generates an error message. There is no default time-out.

TREEINFO

specifies to display information about individual trees, when you build a tree. For example, the table that is shown can display which variables are used in each tree. The option has no effect if you store the tree in a temporary table.

TREELASR=table-name

specifies the in-memory table that contains the information for the random forest if you want to score a table against the random forest.

The data set whose observations are to be scored is specified in the SCOREDATA= option. If you do not specify the SCOREDATA= option, the active table is used as scoring input.

Alias

LASRTREE=

VARS=variable-name

VARS=(variable-name1 <, variable-name2, ...>)

specifies the variables to transfer from the input table to the temporary table in the server that contains the results of scoring a decision tree. This option has no effect unless you specify the TEMPTABLE option and you score a decision tree.

WEIGHT=

specifies the weight for each corresponding event in the EVENT= option.



Growing a Forest of Trees

This demonstration illustrates how to grow a forest. Open the program **mldmbd03d03_RandomWoods.sas**.



Change to interactive mode. Run the code that defines the system options, library name assignments, and macro variable definitions.

The RANDOMWOODS statement syntax is similar to the DECISIONTREE statement. The major difference is additional options that control the number of trees in the forest.

In this demonstration, score code is generated and used to score an in-memory table. The macro variable **&ScoreFolder** contains the folder where score code is written. This macro variable is defined in **MLDMBD_Macros.sas**.

The code below grows a forest with 51 trees. The target and inputs are the same as those used in the DECISIONTREE statement in the previous example. The WHERE clause subsets the data so that the forest is grown only on the training data.

```
table LASRlib.p_model_bank13(tag="&TagString");
where _PartInd_=1;
randomwoods
  b_tgt / nominal=(&catvars) input=(&rfms &r_demoags &catvars)
        assess
        event=("1")
        ntree=51
        seed=12345
        bootstrap=0.632121
        leafsize=100
        maxbranch=2
        maxlevel=6
        m=5
        nbins=10
        treeinfo
        save=RW1
        code=(filename="&ScoreFolder/RWScoreCode.sas" replace)
        temptable;
run;
quit;
```

The CODE statement saves score code in the file specified by the path. The saved score code is used to score an in-memory table later in the demonstration.

The first part of the results is a summary of the specifications of the woods and the individual trees, as well as the overall Out-Of-Bag (OOB) Error.

The IMSTAT Procedure		
Random Forest Information for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13		
Max Number of Branches		2
Max Number of Levels		6
Number of Bins		10
Number of Trees		51
Number of Variables		22
Number of Selected Variables (M)		5
Random Number Seed		12345
Number of People Who Will Read This		0
Bootstrap Percentage		63.2121
Min Number of Tree Levels (actual)		6
Max Number of Tree Levels (actual)		6
Min Number of Nodes		27
Max Number of Nodes		61
Out-of-Bag Error		0.186695

The second part of the results lists the input variables by their importance.

Variable Importance for Random Forest of USER.STUDENT2.HPADATA.P_MODEL_BANK13		
Variable Name	Importance	Std Dev.
rfm5	7295.87	3965.26
rfm7	3568.34	2916.16
rfm9	2282.38	754.32
r_demog_homeval	1692.29	604.57
rfm8	612.47	1050.09
rfm11	466.58	393.57
rfm6	293.96	788.47
cat_input1	285.77	402.60
cat_input2	219.59	250.24
r_demog_inc	122.48	142.85
rfm10	118.56	278.48
demog_age	25.7508	59.5119
rfm12	19.1087	135.31
demog_pr	4.0766	4.7561
demog_genf	3.3047	4.9756
demog_genm	1.9043	9.3004
demog_ho	0.7078	2.2126
rfm2	0.4899	5.5627
rfm3	0	0
rfm1	0	0
rfm4	0	0

An application of the variable importance measure is for choosing inputs for other types of models such as logistic regression or a neural network. (An example is shown in a later demonstration.)

The results conclude with a summary of each tree in the woods. Only the first 12 trees are shown.

Tree Information for Random Forest of USER.STUDENT2.HPADATA.P_MODEL_BANK13								
Tree Number	Number of Obs	Number of Distinct Obs	Number of Nodes	Number of Leaves	Number of Levels	Mis-class Rate	Cumulative OOB Error	Selected Variables
1	335921	248918	55	28.0000	6	0.001420	0.5662	rfm11 rfm7 r_demog_homeval cat_input1 rfm9
2	335921	249164	41	21.0000	6	0.000214	0.3678	rfm11 demog_age r_demog_inc rfm8 rfm5
3	335921	248762	53	27.0000	6	0.001938	0.2741	rfm5 rfm11 rfm9 cat_input2 r_demog_inc
4	335921	249308	55	28.0000	6	0.001864	0.2266	rfm5 r_demog_homeval rfm10 cat_input2 rfm11
5	335921	249580	33	17.0000	6	0.000140	0.2076	rfm10 rfm7 cat_input2 rfm9 cat_input1
6	335921	249057	51	26.0000	6	0.000324	0.1965	rfm11 rfm9 rfm7 rfm5 cat_input2
7	335921	248873	53	27.0000	6	0.000327	0.1913	rfm11 rfm9 rfm5 rfm8 r_demog_homeval
8	335921	248869	55	28.0000	6	0.001747	0.1893	rfm8 r_demog_homeval rfm5 demog_age rfm7
9	335921	248904	51	26.0000	6	0.005236	0.1866	rfm7 rfm9 r_demog_homeval rfm11 rfm8
10	335921	249031	33	17.0000	6	0.000158	0.1856	rfm10 rfm5 r_demog_inc rfm7 demog_age
11	335921	249023	31	16.0000	6	0.000384	0.1869	rfm12 r_demog_homeval cat_input1 rfm11 rfm7
12	335921	248674	27	14.0000	6	0.000714	0.1882	rfm10 rfm11 rfm9 rfm7 rfm5

End of Demonstration

Scoring with a Random Forest

After the predictive algorithm is derived, PROC IMSTAT facilitates scoring new data in the same way as for ordinary decision trees.

SCORE Statement and Selected Options

```
SCORE CODE=file-reference <options>;
```

Options

- `KEEP=(variable-list)` includes a subset of variables in the output file.
- `OUT=libref.data-set-name` specifies where the scored file is stored.



Scoring Data with the RANDOMWOODS Score Code

This demonstration illustrates how to score an in-memory table using score code. The demonstration continues in the **mldmbd03d03_RandomWoods.sas** program.

The score code generated in the previous demonstration is used to score the validation data in the code below.

```
filename RWscore "&ScoreFolder/RWScoreCode.sas";
proc imstat;
  table LASRlib.p_model_bank13(tag="&TagString");
  where PartInd =2;
  score code=RWscore
    keep=( _ALL_ )
    out=LASRlib.ScoredbyRW;
run;
quit;
```

The scored data is saved on the LASR Analytic Server in the **ScoredbyRW** table.

PROC IMSTAT can be used to examine the scored data as shown below.

```
proc imstat;
  table LASRlib.ScoredbyRW;
  columninfo;
run;
  fetch b_tgt RF_b_tgt _vote_ / from=1 to=5;
run;
  frequency b_tgt: RF_b_tgt;
run;
  crosstab b_tgt*RF_b_tgt;
run;
  crosstab RF_b_tgt*_vote_;
run;
  crosstab b_tgt*_vote_;
run;

quit;
```

The COLUMNINFO statement lists all the columns in the scored data set. Only the columns added by the score code are shown below.

57	<u>_tlevname_0_1</u>	Char	12	\$12.	
58	<u>_tlevname_0_2</u>	Char	12	\$12.	
59	<u>_vote_</u>	Num	8	BEST12.	
60	<u>RF_b_tgt</u>	Char	12	\$12.	

The fetch table and the one-way frequencies are shown below.

Selected Records from Table USER.STUDENT2.HPADATA.SCOREDBYRW			
b_tgt	RF_b_tgt	_vote_	
0	0	51.000000	
0	0	50.000000	
1.000000	0	45.000000	
0	0	51.000000	
1.000000	0	51.000000	

Frequencies for Column b_tgt in Table USER.STUDENT2.HPADATA.SCOREDBYRW			
Level	Formatted	Value	Frequency
1	0	0	424144
2	1	1	104471

The RANDOMWOODS score code does not produce estimates of the predicted probabilities, but it classifies cases based on voting. The predicted class is in the **RF_b_tgt** variable. Each tree in the woods casts a vote. For observation three in the first table above, 45 of the 51 trees voted for zero (0), and six trees voted for one (1), leading to a misclassified observation.

The first crosstab produces a confusion matrix from which the misclassification rate can be calculated.

Cross-tabulation of b_tgt by RF_b_tgt for Table USER.STUDENT2.HPADATA.SCOREDBYRW with Aggregator N		
b_tgt	0	1
0	420964	3180
1	93826	10845

The second crosstab compares the votes with the predicted class.

Cross-tabulation of RF_b_tgt by _vote_ for Table USER.STUDENT2.HPADATA.SCOREDBYRW with Aggregator N																										
RF_b_tgt	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
0	1280	1085	1474	1560	1875	1995	2075	2240	3030	3446	3467	4155	5649	4068	5049	5844	5008	6736	5453	6954	10935	7615	14844	11789	19828	377156
1	1242	1224	1332	1379	1074	868	802	683	617	629	596	496	445	466	465	407	281	243	180	163	93	138	64	55	52	31
Cross-tabulation of b_tgt by _vote_ for Table USER.STUDENT2.HPADATA.SCOREDBYRW with Aggregator N																										
b_tgt	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
0	803	694	810	870	868	870	1001	1043	1416	1644	1805	1962	2697	1961	2575	3047	2860	3729	3187	3964	6859	4977	9792	8424	14856	342230
1	1699	1615	1996	2069	2081	1993	1876	1880	2231	2431	2458	2689	3397	2573	2939	3204	2829	3250	2446	3153	4369	2776	5116	3420	5224	34957

The left portions of the above tables are enlarged below.

Cross-tabulation of RF_b_tgt									
RF_b_tgt	26	27	28	29	30	31	32	33	34
0	1280	1085	1474	1560	1875	1995	2075	2240	3030
1	1242	1224	1332	1379	1074	868	802	683	617

Cross-tabulation of b_tgt b									
b_tgt	26	27	28	29	30	31	32	33	34
0	803	694	810	870	868	870	1001	1043	1416
1	1699	1615	1996	2069	2081	1993	1876	1880	2231

The number of votes is reported for the predicted class. Consequently, the minimum number of votes is 26. Examine the column for 26 votes. For **RF_b_tgt=0**, 1,260 cases had 26 votes out of the possible 51, so these cases are predicted to be class zero. For **RF_b_tgt=1**, 1,242 cases had 26 out of the 51 possible votes for class one, so the predicted class for these cases is 1.

End of Demonstration

Logistic Regression in PROC IMSTAT

Objectives

- Describe the PROC IMSTAT LOGISTIC statement.
- List and explain the logistic regression results.
- Use PROC IMSTAT to create a binary logistic regression.

118

Logistic Regression in PROC IMSTAT

- Logistic regression enables you to investigate the relationship between a discrete target variable and one or more input variables.
- There is only one discrete target variable.
- The focus is on binary logistic regression.
- There can be multiple input variables, which can be the following type:
 - continuous.
 - categorical (also called class variables).
 - interaction terms. The interaction of two variables X1 and X2 is denoted by X1*X2.
 - polynomial terms. For example, a continuous input X1 has a second order or quadratic term that is denoted by X1*X1.

119

Classification models predict the probability of class membership. You try to classify whether someone is likely to leave, whether they might respond to a solicitation, whether they are a good or bad credit risks, and so on. The response variable might be represented as a 0 or 1. Event 1 is the event that you are targeting. You can build a classification model where the response variable has several possible values. You are trying to predict to which group observations belong (such as low, medium, or high) or predict which product someone might buy.



As you already saw, variables used by a predictive model or algorithm have several names. The variable to be predicted has been called a ***response*** variable and a ***target*** variable. Other names include ***dependent*** variable and ***outcome*** or ***output*** variable. The input variables are called ***predictor*** variables. They are also referred to as ***independent*** variables or ***covariates***. The name ***effect*** variable is used by SAS Visual Statistics. This is a popular terminology for many textbooks that focus on the inferential applications of logistic regression.

Logistic Regression in PROC IMSTAT

- If the target variable is binary, those values are used by PROC IMSTAT as a binary (0, 1) outcome.
 - You can specify which level is the event to model.
 - By default, the zero (0) event is modeled.
- If the target variable is multinomial, you can choose one target level as the ‘event’ and the others are used as ‘non-event.’
- If the target variable is continuous, you can create a calculation rule to convert it into a dichotomized variable for simple binary logistic regression. (For example, Weights > 200 is the ‘event.’) This can be performed in PROC IMSTAT using a COMPUTE statement before specifying the LOGISTIC statement.

120

Of course, there are other ways to model categorical responses with logistic regression, including ordinal and multinomial logistic regression. These models are available for in-memory processing using the HPLOGISTIC or HPGENSELECT procedures.

LOGISTIC Statement and Selected Options

LOGISTIC *target(nominal-variables)=inputs / option(s);*

Options

- ROLEVAR=*variable-name* [The value of the variable must be one of (1, t, or T) for training, and (2, v, or V) for validation.]
- INPUTS=(*list*) is used to specify input variables, including nominal.
- DESCENDING sorts the target in descending order.
- SCORE <(score-statistics)> requests that the input data be scored and stored in a temporary table.
- CODE=(*filename=“full-path”*) saves score code to score holdout data for model assessment or deployment.

continued...

121

You might be familiar with software (for example, SAS/STAT software and PROC LOGISTIC), where you define “success” for a binary outcome. For example, PROC LOGISTIC supports syntax that enables you to identify values of the target variable that correspond to success or to the events of interest. The LOGISTIC statement in PROC IMSTAT does not support this capability.

LOGISTIC Statement and Selected Options

Options

- SHOWSELECTED specifies that a backward selection algorithm be used for variable selection and that results are displayed for the selected variables.
- SLSTAY=*value between 0.0 and 1.0* specifies the *p*-value requirement for a variable to stay in the model. The *p*-value for a variable must be smaller than SLSTAY= for the variable to remain in the model.



Developing and Assessing a Logistic Regression Model

This demonstration illustrates how to develop and assess a logistic regression model for a binary target. Open the program **mldmbd03d04_LogisticRegression.sas**.



Change to interactive mode. Run the code that defines the system options, library name assignments, and macro variable definitions.

The target is binary, and the DESCENDING option specifies to treat **b_tgt=1** as the event to model. Variables listed after the target in parentheses are declared to be nominal. Candidate inputs are listed after the equal sign. The ROLEVAR= option is used to define the training and validation data: **_partind_=1** is training and **_partind_=2** is validation. The ID variables are included in the scored data set. The link specifies the logit link. The SLSTAY= option invokes the backward selection option and keeps inputs with a *p*-value less than 0.01. The SCORE option requests that all input data are scored with the selected model and saved in a temporary in-memory table. The CODE= option saves the score code for the selected model. The first logistic regression model to be fit uses backward elimination for variable selection.

```
title1 "Logistic Regression Model 1 --- Backward Elimination";
proc imstat;
  table LASRlib.p_model_bank13(tag=&TagString");
  logistic b tgt(&catvars)=&i rfms &ir demogs &catvars /
    rolevar=_partind_ ID=(account _partind_)
    link=logit descending showselected
    score( all ) slstay=0.01
    code=(filename=&ScoreFolder/LR_1_ScoreCode.sas" replace);
run;
  table LASRlib.&_templast_;
  columninfo;
  fetch /from=1 to=10;
run;
  table LASRlib.& temoplast ;
  compute residual "residual=b_tgt-_ilink_";
  compute res2 "res2=(b_tgt-_ilink_)**2";
  compute absres "absres=abs(b_tgt-_ilink_)";
run;
  table LASRlib.&_templast_;
  where _PartInd_=2;
  ods output summary=work.summary_LR1_Valid;
  summary residual res2 absres;
run;
  table LASRlib.&_templast_;
  ods output liftinfo=work.liftdata1;
  ods output rocinfo=work.rocdatal;
  assess _ilink_ / Y=b_tgt event="1"
    groupby=( PartInd )
    nbins=20;      /*---- nbins=20 is default ----*/
run;
quit;
```

Some of the results are listed below. The first table summarizes the model information. The second table lists the nominal inputs and their levels. The third table gives the distribution of the target.

The IMSTAT Procedure		
Model Information		
Data Source	USER.STUDENT2.HPADATA.P_MODEL_BANK13	
Response Variable	b_tgt	
Role Variable	_PartInd_	
Distribution	Binary	
Link Function	Logit	

Class Level Information for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13		
Class	Levels	Values
cat_input1	3	X Y Z
cat_input2	5	A B C D E

Response Profile for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13		
Ordered Value	tgt Binary New Product	Total Frequency
1	0	424385
2	1	107038

The next section of results verifies that the model fit converged and lists the dimensions of the model. All observations are used, and they are enumerated with respect to training and validation partition elements.

Convergence criterion (GCONV=1E-8) satisfied.	
Dimensions	
Number of Model Effects	22
Number of Classification Effects	2
Number of Columns in X	28
Rank of Cross-product Matrix	20
Number of Observations Read	1.06E6
Number of Observations Used	1.06E6
Number of Spurious Rows	1
Number of Training Obs	531423
Number of Validation Obs	528615

The fit statistics and the global test for the model follow.

Fit Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13	
-2 Log Likelihood	385916
AIC	385956
AICC	385956
BIC	386180
R-Square	0.24310
Max-rescaled R-Square	0.38353

Testing Global Null Hypothesis: BETA=0 for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Test	Chi-Square	DF	Pr > ChiSq	
Likelihood Ratio	148016.823	19	<.0001	

Next, the parameter estimates are listed. If the SHOWSELECTED option were not used, the parameter estimates list would include all inputs. Parameter estimates for inputs not selected in the final model are zero and do not need to be listed.

Parameter Estimates for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	0.5088	0.03848	13.22	<.0001
i_rfml2	-0.08485	0.001437	-59.05	<.0001
i_rfml3	-0.03343	0.001192	-28.04	<.0001
i_rfml5	0.3826	0.004235	90.34	<.0001
i_rfml6	-0.01664	0.001299	-12.81	<.0001
i_rfml7	-0.08351	0.005413	-15.43	<.0001
i_rfml8	0.03947	0.002728	14.47	<.0001
i_rfml9	-0.1774	0.001303	-136.14	<.0001
i_rfml0	-0.01668	0.001307	-12.76	<.0001
i_rfml11	0.02091	0.004266	4.90	<.0001
i_rfml2	0.001886	0.000216	8.74	<.0001
ri_demog_homeval	7.009E-6	4.998E-8	140.25	<.0001
demog_pr	0.001721	0.000358	4.81	<.0001
demog_genf	0.02618	0.008263	3.17	0.0015
cat_input1 X	0.4278	0.01915	22.34	<.0001
cat_input1 Y	0.3734	0.02358	15.84	<.0001
cat_input1 Z	0	.	.	.
cat_input2 A	0.2787	0.01392	20.02	<.0001
cat_input2 B	0.2649	0.01240	21.37	<.0001
cat_input2 C	0.1881	0.01262	14.90	<.0001
cat_input2 D	0.1143	0.01441	7.93	<.0001
cat_input2 E	0	.	.	.

The results conclude with the Type III test of model effects and the name of the temporary table that contains the scored data.

Type III Tests of Model Effects for USER.STUDENT2.HPADATA.P_MODEL_BANK13			
Effect	Num DF	Chi-Square	Pr > ChiSq
i_rfml2	1	3486.70	<.0001
i_rfml3	1	786.33	<.0001
i_rfml5	1	8161.44	<.0001
i_rfml6	1	164.04	<.0001
i_rfml7	1	238.01	<.0001
i_rfml8	1	209.38	<.0001
i_rfml9	1	18533.8	<.0001
i_rfml0	1	162.93	<.0001
i_rfml11	1	24.04	<.0001
i_rfml2	1	76.47	<.0001
ri_demog_homeval	1	19668.7	<.0001
demog_pr	1	23.12	<.0001
demog_genf	1	10.04	0.0015
cat_input1	2	499.81	<.0001
cat_input2	4	605.70	<.0001

```

title2 "Residual Diagnostics";

proc contents data=work.summary_LR1_Valid;
run;

data work.validstats;
  attrib Model length=$20 label="Scoring Model";
  set work.summary LR1 Valid end=lastobs;
  retain Nobs MSE RMSE MEANABS ResVar MeanResid MinResid MaxResid;
  keep Model Nobs MSE RMSE MEANABS ResVar MeanResid MinResid MaxResid;
  if (upcase(column)='RESIDUAL') then do;
    ResVar=Std*Std;
    MinResid=Min;
    MaxResid=Max;
    MeanResid=Mean;
    Nobs=N;
  end;
  else if (upcase(column)='RES2') then do;
    MSE=Mean;
    RMSE=sqrt(MSE);
  end;
  else if (upcase(column)='ABSRES') then do;
    MEANABS=Mean;
  end;
  if (lastobs) then do;
    Model="Logistic Reg 1";
    output;
  end;
run;

```

```

data SASlib.validstats;
  set SASlib.validstats work.validstats;
run;

proc print data=SASlib.validstats;
run;

```

The residual diagnostics table appears below. It shows results for the logistic regression model and the decision tree that was derived in a previous section.

Residual Diagnostics

Obs	Model	Nobs	MSE	RMSE	MEANABS	ResVar	MeanResid	MinResid	MaxResid
1	Decision Tree	528615	0.14535	0.38124	0.29220	0.14534	-0.003207440	-0.97222	0.86365
2	Logistic Reg 1	528615	0.11118	0.33344	0.22460	0.11118	-0.002602921	-0.99572	1.00000

As an alternative to backward elimination, the RANDOMWOODS statement can be used as an option to select inputs. The code below illustrates this approach.

The target and candidate inputs are identical to those used in the logistic regression model above. The choice of NTREE=10 is arbitrary for this example.

```

title1 "Logistic Regression Model 2 --- RANDOMWOODS Input Selection";
proc imstat;
  table LASRlib.p_model_bank13(tag=&TagString");
  where _PartInd_=1;

  randomwoods b tgt / nominal=(&catvars)
    input=(&i_rfms &ir_demogs &catvars)
    ntree=10;
  ods output forestvarimpinfo=work.VIMP;
run;
quit;

```

The Variable Importance table produced by the RANDOMWOODS action is saved via the ODS OUTPUT statement to the SAS server. Because the file is small, performance is not an issue. The Variable Importance table is shown below.

Variable Importance for Random Forest of MYHDFS.P_MODEL_BANK13		
Variable Name	Importance	Std Dev.
i_rf7	2151.18	1370.93
i_rf9	2030.02	1096.85
ri_demog_homeval	1213.98	918.80
i_rf11	319.29	237.63
cat_input1	301.25	440.17
i_rf5	221.00	318.94
ri_demog_inc	162.49	146.76
cat_input2	49.5052	45.2821
i_rf8	28.8420	40.7488
i_rf10	23.4106	55.3859
i_demog_age	12.4538	17.2959
i_rf6	11.0675	9.3717
demog_genf	4.2711	8.3942
demog_pv	2.4391	1.8777
demog_ho	1.0275	0.9006
i_rf12	0.7566	0.09049
demog_genm	0.1569	0
i_rf2	0.1485	0.02528
i_rf1	0.09079	0.1377
i_rf4	0.06492	0.1233
i_rf3	0.05755	0.1533

The variables are listed in descending order of importance. Notice that some have very low measures.

The PROC SQL step creates the macro variable **&VIP_Inputs** where their importance is above 3, which is an arbitrary threshold for this example.

```

proc sql;
select VarName into: VIP_Inputs separated by " "
  from work.vimp
  where Importance gt 3;
quit;

%put &VIP_Inputs;

```

The selected variables are shown below.

Variable Name
i_rfmm7
i_rfmm9
ri_demog_homeval
i_rfmm11
cat_input1
i_rfmm5
ri_demog_inc
cat_input2
i_rfmm8
i_rfmm10
i_demog_age
i_rfmm6
demog_genf

The following code fits a logistic regression using the selected variables. Because the inputs were preselected, no variable selection was done by the LOGISTIC statement.

```

proc imstat;
  table LASRlib.p_model_bank13(tag)="&TagString" ;

  logistic b_tgt(&catvars)=&VIP_Inputs /
    rolevar= partind
    ID=(account _partind_)
    score(_all_) descending
    code=(filename="&ScoreFolder/LR_2_ScoreCode.sas" replace) ;
run;

```

Only the fit statistics, global test, and parameters estimates are shown below.

Fit Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
-2 Log Likelihood				404490
AIC				404526
AICC				404526
BIC				404727
R-Square				0.21618
Max-rescaled R-Square				0.34106

Testing Global Null Hypothesis: BETA=0 for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Test	Chi-Square	DF	Pr > ChiSq	
Likelihood Ratio	129442.931	17	<.0001	

Parameter Estimates for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	-0.7166	0.03906	-18.35	<.0001
i_rf7	-0.1451	0.005038	-28.80	<.0001
i_rf9	-0.1883	0.001257	-148.28	<.0001
ri_demog_homeval	5.918E-6	4.973E-8	118.99	<.0001
cat_input1 X	0.6733	0.01750	38.47	<.0001
cat_input1 Y	0.4872	0.02223	21.92	<.0001
cat_input1 Z	0
i_rf5	0.5109	0.004049	126.17	<.0001
i_rf11	-0.02718	0.004146	-6.56	<.0001
cat_input2 A	0.2206	0.01370	16.10	<.0001
cat_input2 B	0.2462	0.01227	20.06	<.0001
cat_input2 C	0.1915	0.01239	15.46	<.0001
cat_input2 D	0.08722	0.01422	6.13	<.0001
cat_input2 E	0
i_rf10	-0.03788	0.001270	-29.83	<.0001
ri_demog_inc	5.92E-7	2.47E-7	2.40	0.0165
i_rf6	-0.01432	0.001285	-11.14	<.0001
i_rf8	0.07242	0.002387	30.34	<.0001
i_demog_age	0.001157	0.000274	4.22	<.0001
demog_genf	0.06698	0.008089	8.28	<.0001

The SCORE statement requested that the input data be scored with the fitted model. The temporary table that was created can be accessed and examined using code similar to that below.

```
table mylasr.&_templatst_;
  columninfo;
  fetch /from=1 to=10;
run;
```

The IMSTAT procedure defines an **&_templat_** macro variable that can be used to access the last created temporary table. In this case, the last table created is the one containing the scored data. The TABLE statement makes the scored table the active table, and the COLUMNINFO and FETCH statements list the columns and the first 10 rows.

The first 16 rows of the COLUMNINFO table contain the target and then the inputs and the variables listed in the ID option.

Column Information for Table _T_BBBA389E_7FED0AFE20C8					
Id	Column	Type	Length	Format	Label
1	account	Char	9	\$9.	account
2	_partind_	Num	8	BEST12.	_partind_
3	cat_input1	Char	5	\$5.	cat_input1
4	cat_input2	Char	1	\$1.	cat_input2
5	b_tgt	Num	8	BEST12.	b_tgt
6	i_rfm7	Num	8	BEST12.	i_rfm7
7	i_rfm9	Num	8	BEST12.	i_rfm9
8	ri_demog_homeval	Num	8	BEST12.	ri_demog_homeval
9	i_rfm11	Num	8	BEST12.	i_rfm11
10	i_rfm5	Num	8	BEST12.	i_rfm5
11	ri_demog_inc	Num	8	BEST12.	ri_demog_inc
12	i_rfm8	Num	8	BEST12.	i_rfm8
13	i_rfm10	Num	8	BEST12.	i_rfm10
14	i_demog_age	Num	8	BEST12.	i_demog_age
15	i_rfm6	Num	8	BEST12.	i_rfm6
16	demog_genf	Num	8	BEST12.	demog_genf

The rest of the columns are added by the scoring process.

17	_PRED_	Num	8	BEST12.	
18	_RESID_	Num	8	BEST12.	
19	_STDP_	Num	8	BEST12.	
20	_ILINK_	Num	8	BEST12.	
21	_LCLM_	Num	8	BEST12.	
22	_UCLM_	Num	8	BEST12.	
23	_LCL_	Num	8	BEST12.	
24	_UCL_	Num	8	BEST12.	
25	_LEVERAGE_	Num	8	BEST12.	
26	_LIKEDIST_	Num	8	BEST12.	
27	_PEARSON_	Num	8	BEST12.	
28	_DEVRESID_	Num	8	BEST12.	
29	_DIFDEV_	Num	8	BEST12.	
30	_DIFCHISQ_	Num	8	BEST12.	
31	_CBAR_	Num	8	BEST12.	
32	_STDRESCHI_	Num	8	BEST12.	

A subset of the results of the FETCH statement is listed below.

Selected Records from Table _T_BBBA389E_7FED0AFE20C8						
i_demog_age	i_rf6	demog_genf	_PRED_	_RESID_	_STDP_	_ILINK_
45.000000	8.000000	1.000000	-1.629889	-1.195951	0.022292	0.163846
45.000000	5.000000	1.000000	-2.484390	-1.083376	0.015447	0.076960
24.000000	8.000000	0	-1.671905	-1.187889	0.017062	0.158170
58.716371	8.000000	1.000000	-2.133718	-1.118396	0.013342	0.105863
69.000000	3.000000	0	-2.248494	-1.105558	0.013958	0.095479
61.000000	7.000000	1.000000	-2.311690	-1.099094	0.012475	0.090159
58.000000	7.000000	0	-0.755185	-1.469924	0.014144	0.319893
58.716371	11.000000	1.000000	-3.976197	-1.018757	0.021480	0.018411

The `_PRED_` column contains the predicted values on the link (logit) scale. The `_ILINK_` column contains the predicted value on the inverse-link scale (in this case, the probability scale). To assess the model, the `_ILINK_` values should be used. The variables `_LCL_` and `_UCL_` are lower and upper 95% confidence limits on the logit prediction, `_PRED_`. The variables `_LCLM_` and `_UCLM_` are lower and upper 95% confidence limits on the probability prediction, `_ILINK_`.

Because the file contains the scored training and validation partitions, the ASSESS statement can be used to calculate the lift data for both using the GROUPBY option.

```
table LASRlib.&_templast_;
compute residual "residual=b tgt- ilink ";
compute res2 "res2=(b_tgt-_ilink_)**2";
compute absres "absres=abs(b_tgt-_ilink_)";
run;
table LASRlib.&_templast_;
where PartInd =2;
ods output summary=work.summary;
summary residual res2 absres;
run;
table LASRlib.& templast ;
ods output liftinfo=work.liftdata;
ods output rocinfo=work.roodata;
assess ilink / Y=b tgt event="1"
            groupby=(_PartInd_)
            nbins=20;
run;
quit;
```

The ODS OUTPUT statement is used to save the assessment data to the SAS server.

The following code creates error diagnostic statistics and combines them with the same statistics calculated for the decision tree:

```
data work.validstats;
attrib Model length=$20 label="Scoring Model";
set work.summary end=lastobs;
retain Nobs MSE RMSE MEANABS ResVar MeanResid MinResid MaxResid;
keep Model Nobs MSE RMSE MEANABS ResVar MeanResid
      MinResid MaxResid;
if (upcase(column)='RESIDUAL') then do;
  ResVar=Std*Std;
  MinResid=Min;
  MaxResid=Max;
  MeanResid=Mean;
  Nobs=N;
end;
else if (upcase(column)='RES2') then do;
  MSE=Mean/N;
  RMSE=sqrt(MSE);
end;
else if (upcase(column)='ABSRES') then do;
  MEANABS=Mean/N;
end;
```

```

if (lastobs) then do;
  Model="Logistic Regression";
  output;
end;
run;

data SASlib.validstats;
  set SASlib.validstats work.validstats;
run;
proc print data=SASlib.validstats;
run;

```

The combined error diagnostic table appears below. It shows results for the three IMSTAT models fit.

Residual Diagnostics									
Obs	Model	Nobs	MSE	RMSE	MEANABS	ResVar	MeanResid	MinResid	MaxResid
1	Decision Tree	528615	0.14535	0.38124	0.29220	0.14534	-.003207440	-0.97222	0.86365
2	Logistic Reg 1	528615	0.11118	0.33344	0.22460	0.11118	-.002602921	-0.99572	1.00000
3	Logistic Reg 2	528615	0.11937	0.34550	0.23773	0.11936	-.002615275	-0.99912	0.99390

The first logistic regression model appears to be superior to the second model. For this data, backward elimination appears to be superior to random forest variable selection. However, what follows examines the latter model.

Code for producing the ROC curve follows:

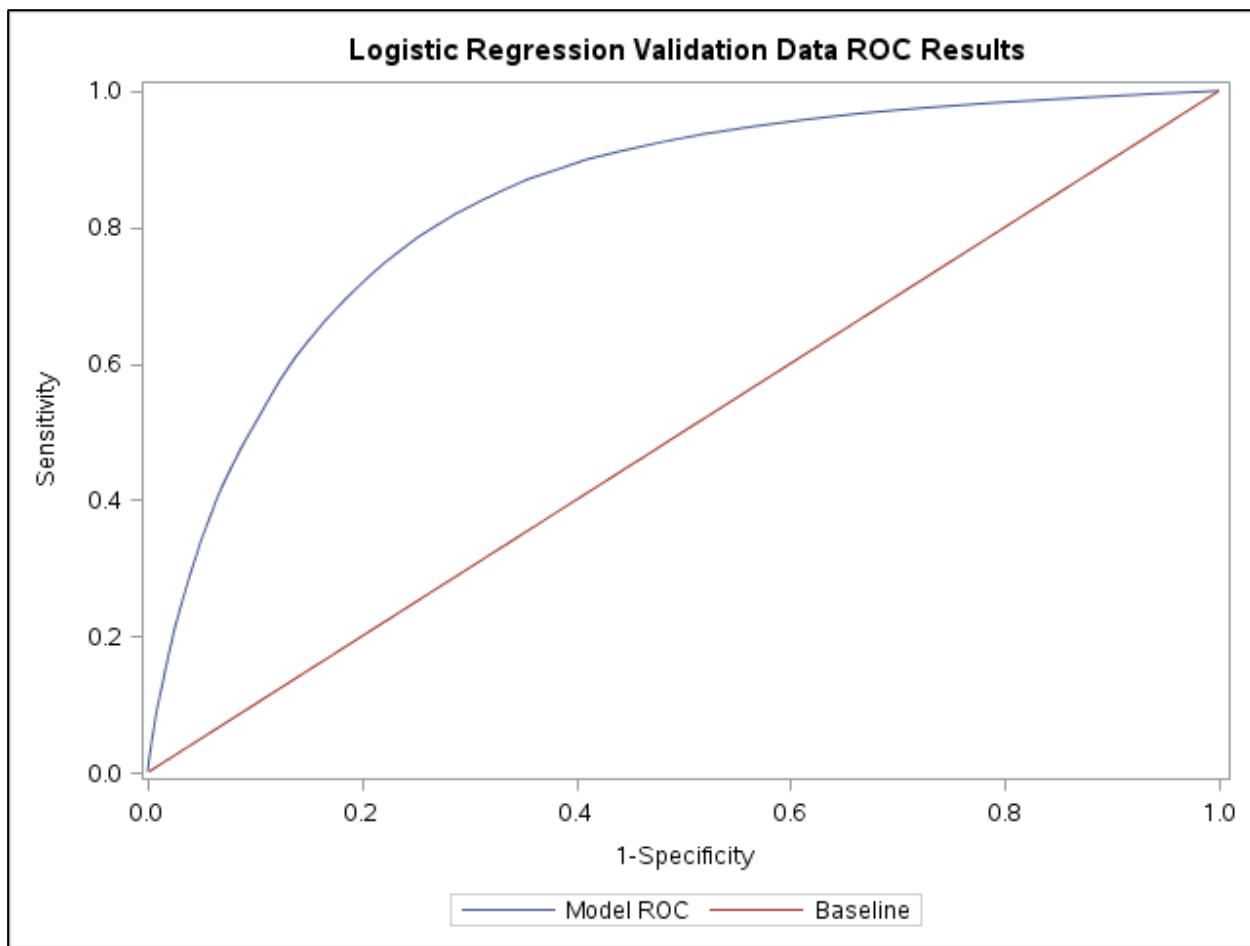
```

data work.rocvalid;
  set work.roodata(where=(strip(_PartInd_)= '2'));
  onemspcf=1-Specificity;
  keep CutOff Specificity Sensitivity onemspcf;
run;

proc sgplot data=work.rocvalid;
  series y=sensitivity x=onemspcf /
    legendlabel="Model ROC" name="line1";
  series y=sensitivity x=sensitivity /
    legendlabel="Baseline" name="line2";
  keylegend "line1" "line2" / location=outside position=bottom;
  label sensitivity="Sensitivity"
        onemspcf="1-Specificity";
run;

```

The results appear below.



A listing of selected columns of the lift data and plots for each partition are produced by the code below.

```

proc print data=work.liftdata;
  by _PartInd_;
  var event depth value NEvents Lift CumResp CumLift Gain;
run;

data work.lifttrain;
  set work.liftdata(where=(strip(_PartInd_)= '1'));
  rename CumLift=CumTrain;
  keep depth CumLift;
run;

data work.liftvalid;
  set work.liftdata(where=(strip(_PartInd_)= '2'));
  rename CumLift=CumValid;
  keep depth CumLift;
run;

data work.liftplot;
  merge work.lifttrain work.liftvalid;

```

```

by depth;
run;
title2 "Compare Training and Validation Cumulative Lift";
proc sgplot data=work.liftplot;
  series y=CumTrain x=depth / legendlabel="Train" name="line1";
  series y=CumValid x=depth / legendlabel="Validation" name="line2";
  yaxis label="Cumulative % Lift" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
run;

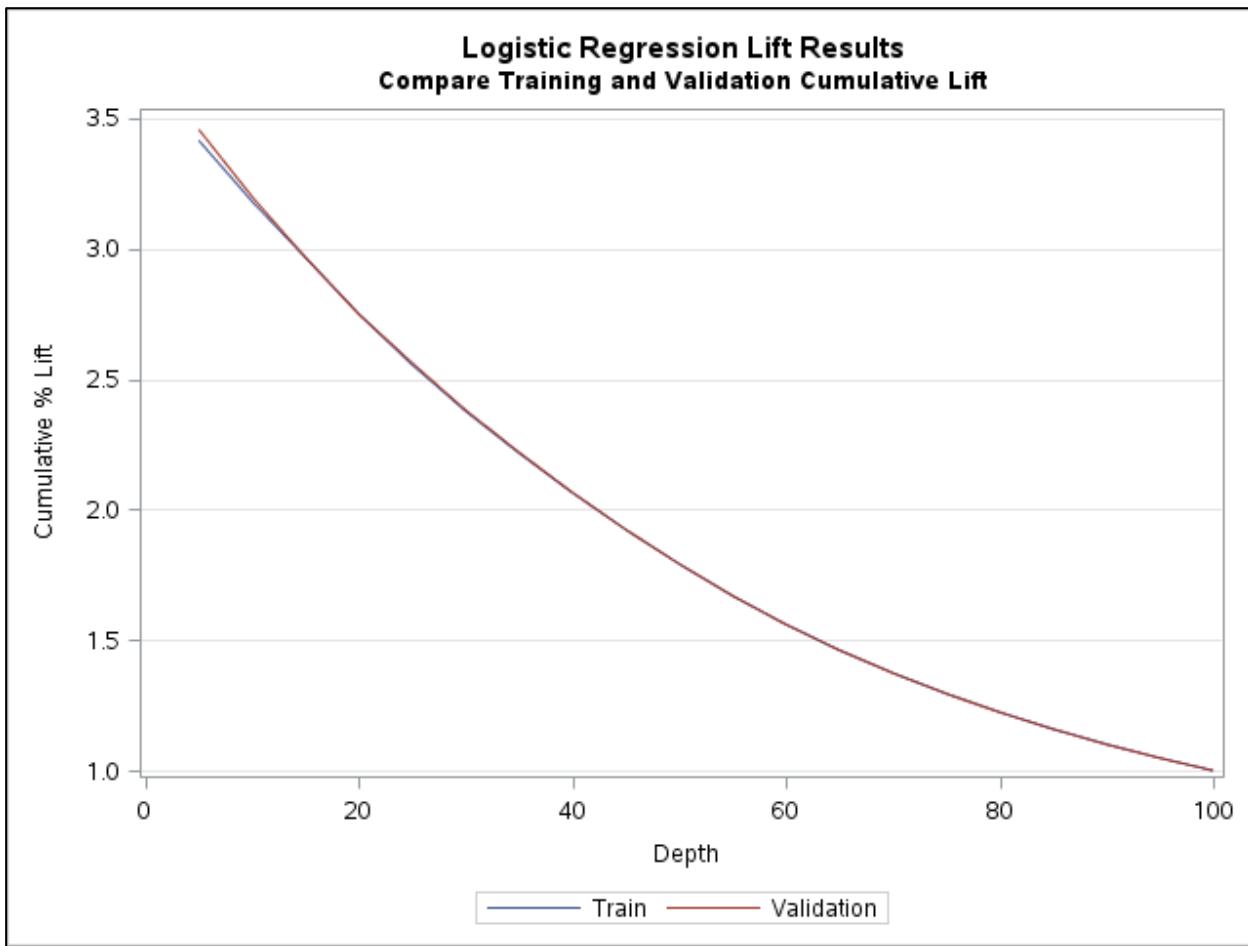
```

Assessment Measure by Data Partitions									
<u>_partind_=1</u>									
Obs	Event	Depth	Value	NEvents	Lift	CumResp	CumLift	Gain	
1	1	5.0000	0.6702	18300	3.4193	17.0967	3.4193	2.4193	
2	1	10.0000	0.5100	15806	2.9533	31.8635	3.1863	2.1863	
3	1	15.0000	0.4045	13553	2.6324	44.6253	2.9684	1.9684	
4	1	20.0000	0.3303	11190	2.0908	54.9795	2.7490	1.7490	
5	1	25.0000	0.2740	9495.00	1.7741	63.8502	2.6540	1.5540	
6	1	30.0000	0.2303	8012.00	1.4970	71.3354	2.3778	1.3778	
7	1	35.0000	0.1960	6737.00	1.2588	77.6294	2.2180	1.2180	
8	1	40.0000	0.1674	5282.00	0.9869	82.5641	2.0641	1.0641	
9	1	45.0000	0.1435	4239.00	0.7921	86.5244	1.9228	0.9228	
10	1	50.0000	0.1233	3189.00	0.6059	89.5037	1.7901	0.7901	
11	1	55.0000	0.1062	2454.00	0.4585	91.7964	1.6690	0.6690	
12	1	60.0000	0.09131	1843.00	0.3444	93.5182	1.5588	0.5588	
13	1	65.0000	0.07796	1539.00	0.2876	94.9560	1.4609	0.4609	
14	1	70.0000	0.06589	1324.00	0.2474	96.1929	1.3742	0.3742	
15	1	75.0000	0.05494	1065.00	0.1990	97.1879	1.2958	0.2958	
16	1	80.0000	0.04492	786.00	0.1469	97.9222	1.2240	0.2240	
17	1	85.0000	0.03564	677.00	0.1265	98.5547	1.1595	0.1595	
18	1	90.0000	0.02714	612.00	0.1144	99.1265	1.1014	0.1014	
19	1	95.0000	0.01850	514.00	0.09604	99.6067	1.0485	0.04849	
20	1	100.00	0.002780	421.00	0.07866	100.00	1.0000	0	

The ASSESS action produces summary statistics for the 20 bins sorted from highest to lowest score. **Lift** is the proportion of events in the bin divided by the proportion of events in the data. **CumResp** is the cumulative number of event cases in the bin and all higher bins. **Gain** is the proportion increase in the number of event cases over what is expected based on random.

<u>_partind_=2</u>									
Obs	Event	Depth	Value	NEvents	Lift	CumResp	CumLift	Gain	
21	1	5.0000	0.6664	18025	3.4507	17.2538	3.4507	2.4507	
22	1	10.0000	0.5074	15399	2.9480	31.9938	3.1994	2.1994	
23	1	15.0000	0.4020	13070	2.5021	44.5042	2.9669	1.9669	
24	1	20.0000	0.3275	10889	2.0846	54.9272	2.7484	1.7484	
25	1	25.0000	0.2716	9540.00	1.8263	64.0589	2.5624	1.5624	
26	1	30.0000	0.2284	7801.00	1.4934	71.5281	2.3842	1.3842	
27	1	35.0000	0.1945	6479.00	1.2403	77.7278	2.2208	1.2208	
28	1	40.0000	0.1662	5099.00	0.9762	82.6086	2.0652	1.0652	
29	1	45.0000	0.1426	4120.00	0.7887	86.5522	1.9234	0.9234	
30	1	50.0000	0.1227	3140.00	0.6011	89.5579	1.7912	0.7912	
31	1	55.0000	0.1056	2374.00	0.4545	91.8303	1.6696	0.6696	
32	1	60.0000	0.09098	1812.00	0.3489	93.5647	1.5594	0.5594	
33	1	65.0000	0.07776	1479.00	0.2831	94.9804	1.4812	0.4812	
34	1	70.0000	0.06666	1270.00	0.2431	96.1961	1.3742	0.3742	
35	1	75.0000	0.05478	997.00	0.1909	97.1504	1.2953	0.2953	
36	1	80.0000	0.04481	781.00	0.1495	97.8980	1.2237	0.2237	
37	1	85.0000	0.03569	632.00	0.1210	98.5029	1.1589	0.1589	
38	1	90.0000	0.02715	559.00	0.1070	99.0380	1.1004	0.1004	
39	1	95.0000	0.01854	587.00	0.1085	99.5807	1.0482	0.04822	
40	1	100.00	0.002792	438.00	0.08385	100.00	1.0000	0	

An overlay plot of the lift curves is shown below.



Training and validation lift curves are almost identical.

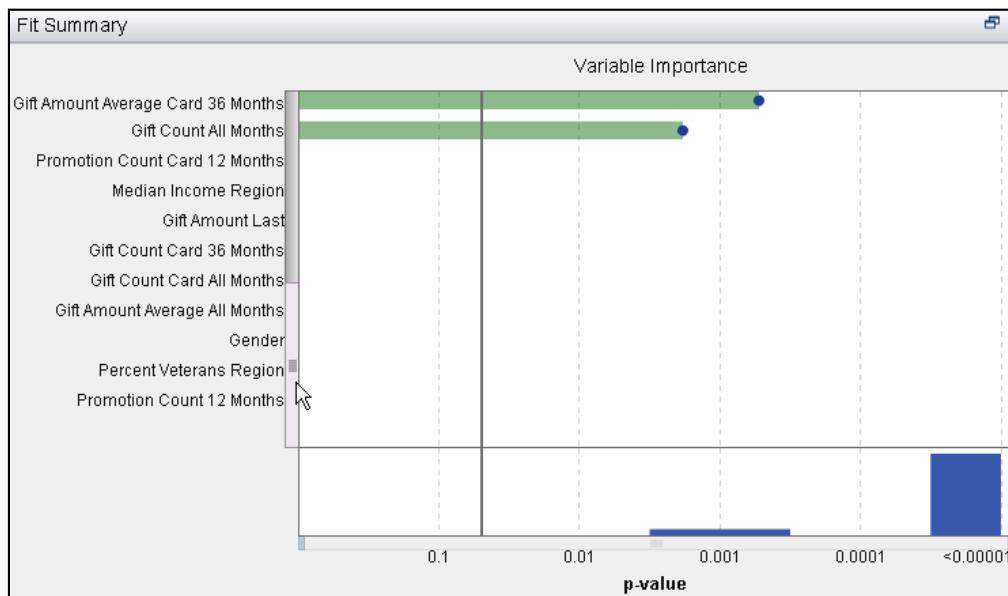
End of Demonstration

3.3 Solutions

Solutions to Exercises

1. Building a Logistic Regression in SAS Visual Analytics

- a. Return to your remote desktop client machine. If your session timed out, sign in and use the information provided by your instructor.
- b. From the SAS Visual Analytics Hub, start a new Data Explore task, and select the **PVA97NK** data source.
- c. Click the **Logistic Regression** tool on the toolbar.
- d. If you did not already do so, in the Measure column, right-click **Target Gift Flag** and select **Category** to create a binary target variable for donations.
- e. Clear the **Auto-Update model** check box at the bottom of the Properties tab.
- f. Select **Target Gift Flag** as the response. Because you want to model customers who make donations, select **1** as the event level under **Advanced** on the Roles tab.
- g. From the Category column, add **Gender**, **Home Owner**, and **Status Category 96NK** as classification effects.
- h. From the Measure column, add all 22 variables *except* **Target Gift Amount** as continuous effects. (You add 21 columns.)
- i. On the Properties tab, click the **Use Variable selection** check box and set the significance level to **.05**.
- j. Create the logistic model by selecting either of the **Auto-update model** check boxes.
 - Maximize the **Fit Summary** panel and scroll to the bottom of the variable importance list. How many variables are not included in this model? **9**



- Are any of the insignificant variables that are not included in this model classification effects? **Gender**

2. Examining Additional Logistic Regression Results

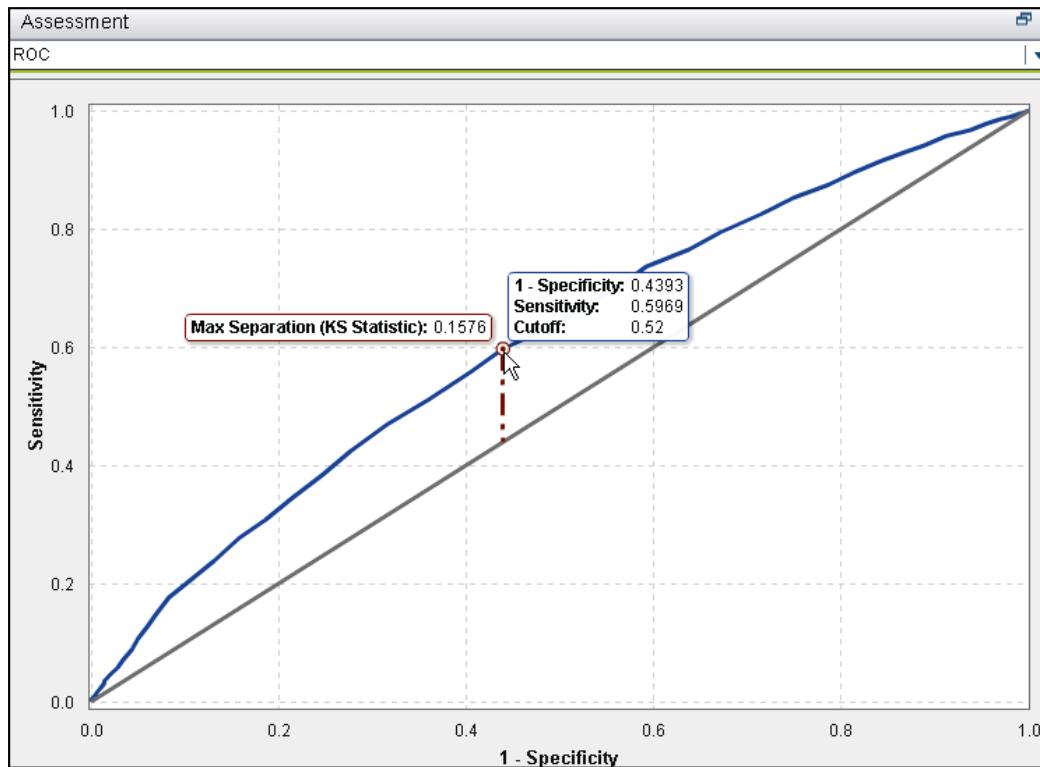
- Open the summary table and click the **Parameter Estimates** tab.
- Select the **Estimate** column heading to sort the parameter estimates and then determine which parameter had the very largest estimate. What was the value?
Status Category 96NK – E, .395

Fit Summary					
Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Parameter		Estimate ▾	Standard Error	z Value	
Status Category 96NK	E	0.395289	0.091768	4.307495	
Intercept		0.214187	0.097098	2.205888	
Status Category Star All Months		0.110559	0.022256	4.967534	
Home Owner	H	0.080017	0.017188	4.65531	
Gift Count 36 Months		0.078822	0.006432	12.25448	
Promotion Count Card 36 Months		0.069532	0.009425	9.124537	

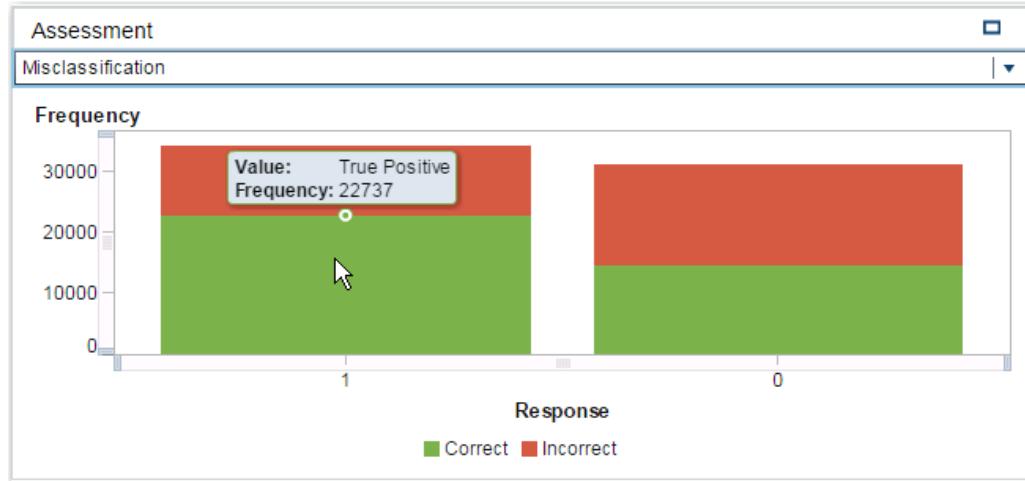
- Click the **Response Profile** tab. How many of the customers made donations? **34111**

Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	Response Profile
Target Gift Flag					Count
0					30877
1					34111

- Hide the summary table.
- Maximize the **Assessment** panel to gain access to the assessment charts.
- Examine the lift chart to determine the advantages of using this model for prediction. How does this model compare to the Best model? **When the Model line is closer to the Best line, especially in the lower percentiles, the model is better. Some of the effects are skewed and some of the effects contain missing values. Performing imputation on the missing values and transforming some of the variables might yield a better model.**
- Select the ROC chart. What are the KS statistic and the cutoff value? **.16 and .52**



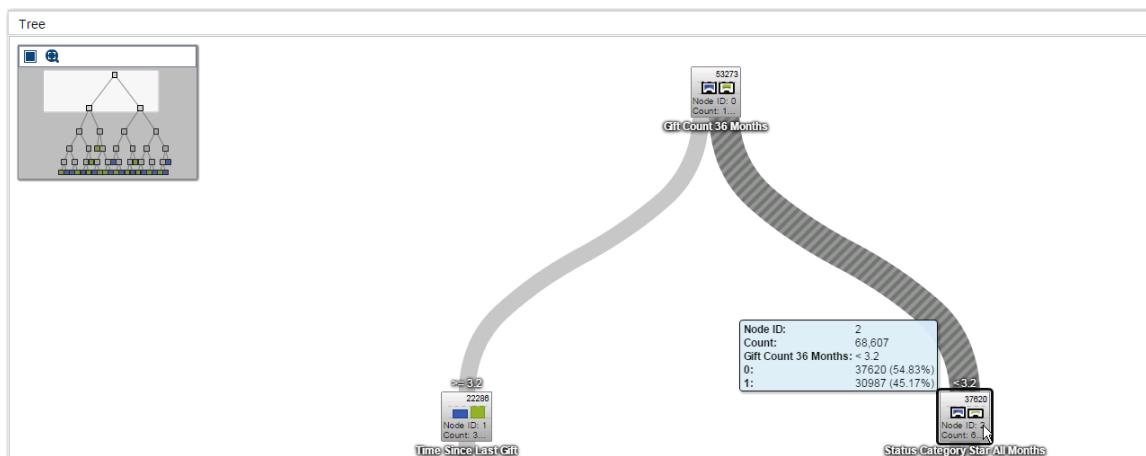
- 3) What is the prediction cutoff value that is used in the current logistic regression model?
Click the Properties tab. The value is specified under the Assessment section at .50.
- 4) Select the misclassification chart to determine whether this model predicts more true positives or more true negatives. **true positives**



- d. Save your project as **VS Logistic Exercise**.
3. **Building a Decision Tree in SAS Visual Statistics**
 - a. Return to your remote desktop client machine. If your session timed out, use the information provided by your instructor to sign in.
 - b. From the SAS Visual Analytics Hub, start a new data exploration and select the **PVA97NK** data source.

- c. Click the **Decision Tree** tool on the toolbar.
- d. If you did not already do so, in the Measure column, right-click **Target Gift Flag** and select **Category** to create a binary target variable for donations.
- e. Select **Target Gift Flag** as the response. Because you want to model customers who make donations, select **1** as the event level under **Advanced** on the Roles tab.
 - How many customers made donations? **53273**
 - What proportion of individuals does that represent in the target node? **50%**
- f. Clear the **Auto-Update model** check box at the bottom of the Properties tab.
- g. From the Category column, add **Gender**, **Home Owner**, and **Status Category 96NK** as predictors.
- h. From the Measure column, add all 22 variables *except* **Target Gift Amount** as predictors. (You add 21 columns.)
- i. Create the decision tree by selecting either of the **Auto-update model** check boxes.

Zoom in toward the root node. (You can use either the wheel on your mouse or right-click in the tree background and select **Show tree overview**.)



- On what column does the top split occur? **Gift Count 36 Months**
- What is the split point that determines to which branch a customer belongs? **3.2**
- In which branch do the majority of customers fall at this split point?
Gift Count 36 Months < 3.2.
- How many customers were less than this value and belong to Node 2? **68607**
- What proportion of the customers in Node 2 made donations? **45.2**

4. Examining Additional Decision Tree Results

- a. Select the **Show Details** button to examine the node statistics.
 - 1) Examine the last column to see whether there are any 100% donator nodes. If so, which node or nodes? **nodes 24 and 38**

Node Statistics		Node Rules		Depth	Parent ID	N Children	Type	Observations	% Observations	Percent of Parent	Gain	Predicted Value	0	1
Node ID	Class	Rule	Condition											
23	4	12	2	Class				33	0.03%	13.64%	0.9183	1	11 (33.33%)	22 (66.67%)
38	5	23	0	Leaf				22	0.02%	66.67%	0.0000	1	22 (100.00%)	
24	4	12	0	Leaf				209	0.20%	86.36%	0.0000	1	209 (100.00%)	
3	2	1	2	Class				35068	32.91%	92.43%	0.0072	1	14982 (42.72%)	20086 (57.28%)

- 2) Click the **Node Rules** tab. Is Node 29 a class node or a leaf node? **leaf**

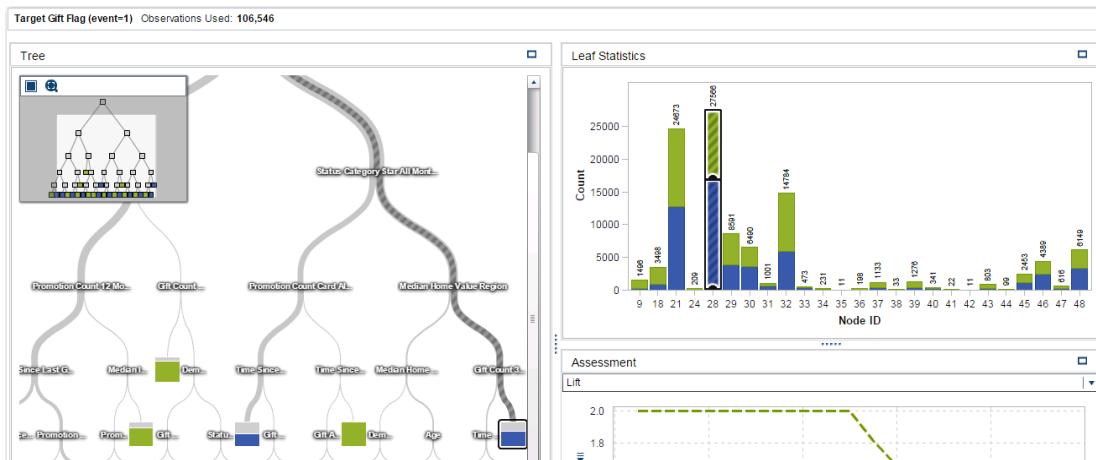
Node Statistics		Node Rules											
Node ID	Parent ID	Type	Gender	Home Owner	Status Categor...	Status Categor...	Promotion Cou...	Percent Vote					
24	12	Leaf					≥ 0.05	< 7.4					
25	13	Class					< 0.05						
26	13	Class					≤ 0.05						
27	14	Class					≤ 0.05						
28	14	Leaf					≤ 0.05						
29	15	Leaf										≥ 10.55	
30	15	Leaf										≥ 10.55	

- 3) Use the node rules to describe the customers in Node 29 in laymen's terms. **In the past 36 months, Node 29 customers gave more than three donations. They received approximately 11 promotions in the last year. It is at least 64 months since they made their first donations. They did not make a donation in approximately the past 18 months.**

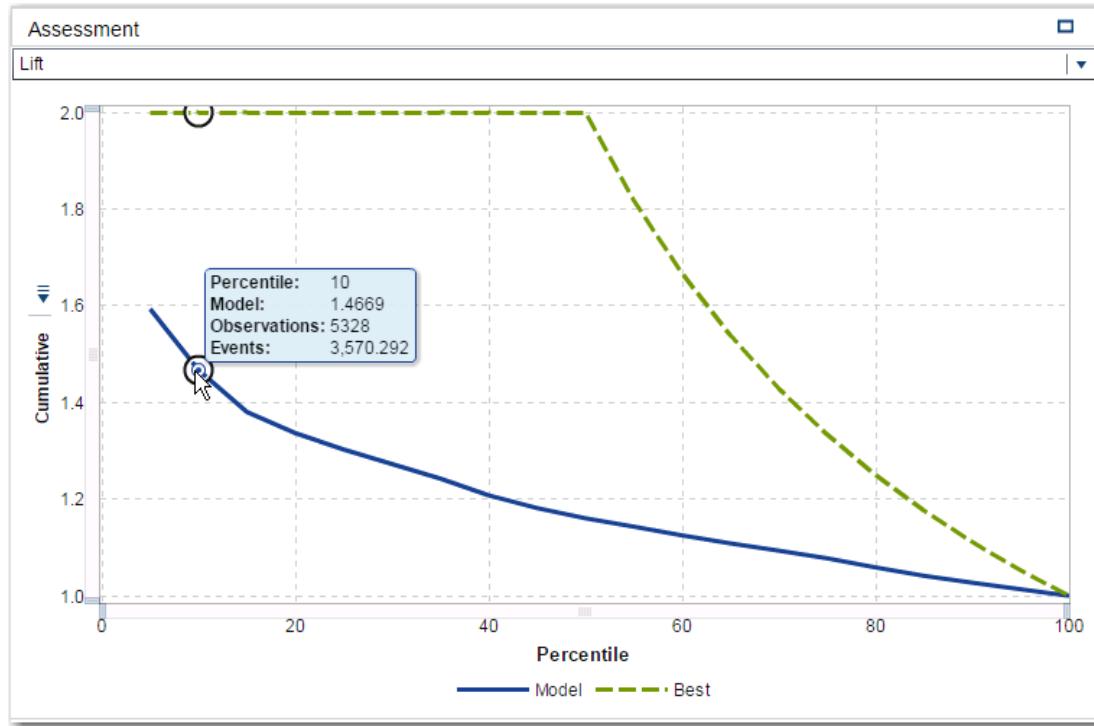
b. Hide the summary table.

c. Right-click in the tree background to open the diagnostic plots.

- 1) Maximize the **Leaf Statistics** panel. Which node has the most customers? **28**
- 2) Select the node with the most customers and examine either the tree or the summary table. What is the proportion of donors? **38.7% (or 10670 count)**



- 3) Maximize the **Assessment** panel and examine the lift chart. What can you determine about the top 10% (percentile) of the data? **When ranked-ordered by this model, the proportion of customers in the top 10% are approximately 1.5 times more likely to make a donation than a randomly selected 10% of the cases.**



- d. Save your project as **VS Tree Exercise**.

End of Solutions

Solutions to Student Activities (Polls/Quizzes)

3.01 Multiple Choice Poll – Correct Answer

Which of the following are true regarding measure variables?

- a. A measure variable must be a cluster or group by variable.
- b. Measure variables can be used to rank-order the clusters.**
- c. All clusters must be displayed in an advanced group by.

42

3.02 Multiple Choice Poll – Correct Answer

Which of the following is false regarding decision trees in Visual Statistics?

- a. An input variable can appear in only one split of the tree, by default.**
- b. Probabilities can be derived from the proportion of responders in each (terminal) leaf.
- c. Trees are relatively easy to interpret and explain.

52

3.03 Poll – Correct Answer

Forests tend to be more stable models than single decision trees.

- True
- False

Chapter 4 Analysis Methods for Interval Targets

4.1 Interval Targets (SAS Visual Statistics)	4-3
Linear Regression.....	4-3
Demonstration: Building a Linear Regression.....	4-20
Exercises.....	4-32
Generalized Linear Models.....	4-34
Demonstration: Building a GLM Model.....	4-46
Exercises.....	4-51
4.2 Interval Targets (SAS In-Memory Statistics).....	4-52
Demonstration: Generalized Linear Models for Count Data.....	4-55
Demonstration: Generalized Linear Models for Interval Targets.....	4-72
Demonstration: Fitting a Lognormal Model to an Interval Target	4-81
4.3 Interval Targets (SAS Enterprise Miner)	4-84
Demonstration: Fitting a Zero-Inflated Poisson Model Using the HP GLM Node in SAS Enterprise Miner.....	4-86
4.4 Solutions	4-90
Solutions to Exercises	4-90
Solutions to Student Activities (Polls/Quizzes)	4-93

4.1 Interval Targets (SAS Visual Statistics)

Objectives

- Create a linear regression model in SAS Visual Statistics.
- Implement predictor variable selection for a linear regression model.
- Create interaction variables.
- Explore the impact of outlier observations on the fitted model.
- Use generated statistics and diagnostic plots to assess model goodness of fit.

3

Linear Regression

Regression Models

- Regression models enable you to predict the value of a continuously valued response variable (dependent variable) as a linear function of one or more effects (independent or predictor variables).
- SAS Visual Statistics provides the following regression model types:
 - linear regression
 - generalized linear model

4

Regression techniques are useful for investigating the relationships between variables. The application of regression models is widespread.

Examples include the following:

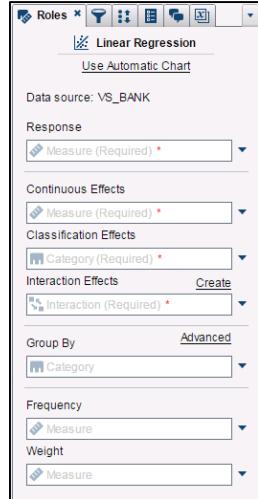
- to determine trends in data
- in health care, to study observational data and determine what factors put people at risk for certain diseases
- in finance, for risk assessment
- in economics, for modeling inventory, labor supply, and demand, and for developing spending models

Linear Regression

- Linear regression models assume that the relationship between the response variable and the input variables is linear.
 - $Y_i = \alpha + \sum \beta_i X_{ij} + \varepsilon_i$
- Maximum likelihood is the preferred method for generating parameter estimates.
- There is only one continuous response variable (**Response**).
- Model effects or explanatory variables can be
 - continuous (**Continuous Effects**)
 - categorical (**Classification Effects**)
 - interaction terms (**Interactions**).

Linear Regression Roles

- Response – one (continuous) measure
- Continuous Effects – one or more (continuous) measures
- Classification Effects – one or more categories
- Interaction Effects – one or more interaction terms (Create them first.)
- Group By – defines separately modeled groups
- Frequency – only one measure
- Weight – only one measure



6

You need to create the interaction terms before you can assign them to the Interactions role.

If you specify more than one group-by variable, the groups are combined to form compound groups.

Frequency – specifies the variable that is used to perform the frequency analysis for each effect. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used.

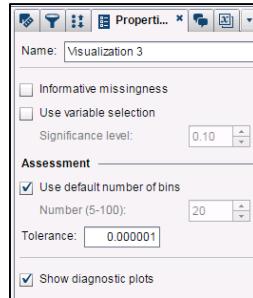
Weight – specifies the numeric variable to use as a weight variable when you solve the linear model.

 Only the observations with a weight value that is not missing and greater than zero are used in the analysis.

Adding roles to the model automatically updates the model. If you do not want the model to be updated automatically when you add roles, clear the **Auto-update model** check box at the bottom of the Roles tab. After you define all of the roles, you can update the model by clicking **Update** at the bottom of the Roles tab.

Linear Regression Properties

- Name (of model)
- Informative missingness
- Use variable selection
 - Significance level
- Assessment
 - Number of bins
 - default
 - user-specified number
 - Tolerance
- Show diagnostic plots
(residual plot, assessment, influence plot)



7

Linear regression, logistic regression, and GLM models in SAS Visual Statistics handle missing values by dropping all observations that contain a missing value in any assigned role variable. However, these models do provide the *Informative missingness* property. Informative missingness requests that missing values be handled by modeling them through extra model effects. These effects consist of dummy variables that have a value of 1 when the value of a continuous model variable that is involved in the effect is missing. Otherwise, they are assigned the value of 0. For continuous variables, missing values are imputed with the observed mean, and an indicator variable is created to denote missingness. For category variables, missing values are considered a distinct level.

 The default method for handling missing values varies across model types in SAS Visual Statistics. In contrast to the models listed above, the decision tree accommodates missing values automatically and does not delete missing values and corresponding rows by default. Also, missing values of group-by variables are retained and given their own group. Computed columns can also be added to the data in Visual Analytics and used in models.

Use variable selection – uses a fast backward-selection algorithm to determine whether effects should stay in the model. If you enable **Use variable selection**, you can also set the significance level at which the effects remain in the model. Variable selection attempts to reduce the number of input variables and include only the most important variables in the model.

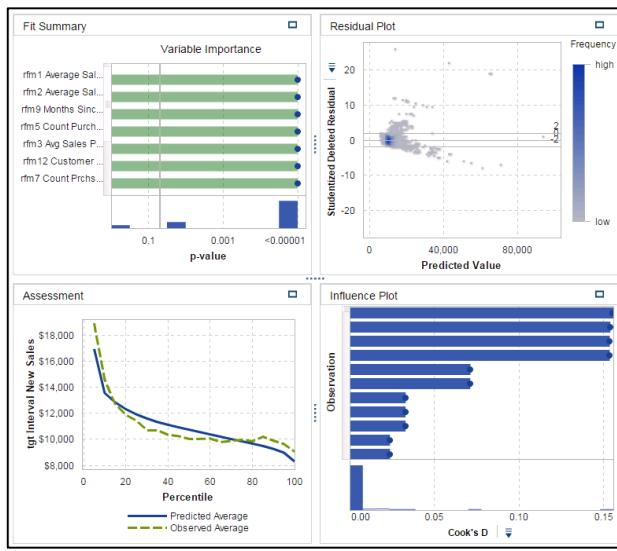
Assessment

- **Use default number of bins** – specifies the number of bins to use in the lift calculations in the assessment plot. It is enabled by default and set at 20. Clear the field and enter your own number of bins if you want to change the default. Increasing the number of bins increases the accuracy of the assessment at the expense of computing time.
- **Tolerance** – specifies the tolerance value that is used to determine the convergence of the iterative algorithm that estimates the percentiles. Specify a smaller value to increase the algorithmic precision.

Show diagnostic plots – displays the residual plot, assessment, and influence plot. It is enabled by default. Clear the check box if you do not want the diagnostic plots to appear.

Linear Regression Results

8



Analyzing Linear Regression Results

Four panes appear. They can help you analyze the results of the linear regression model.

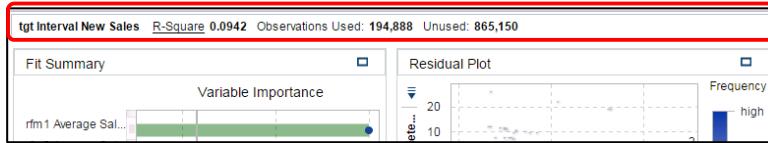
- Fit Summary – displays how significant the effect variables are to the response variable.
- Residual Plot – displays the difference between the predicted and the actual data.
- Assessment – displays the values for the observed response along with the model's predicted response.
- Influence Plot – displays the observations that might influence the overall analysis.

9

Summary Bar

For all models except a cluster model, general model information appears at the top of the Model pane.

- name of response variable
- model evaluation criteria (selected from drop-down list)
- number of observations used to build the model
- number of observations not used



10

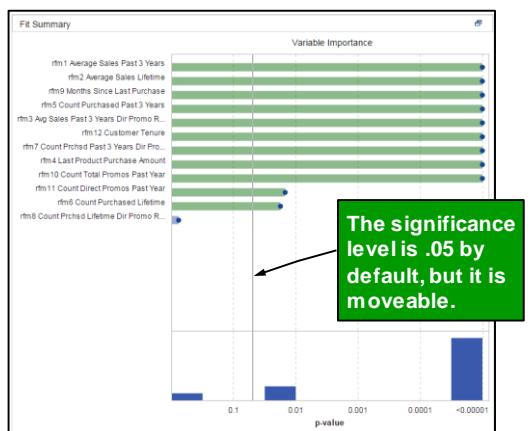
You can click the model evaluation criteria in the summary bar to select a different choice from the pop-up menu. The choices depend on the model type that is used. For a linear regression, the evaluation criteria choices are as follows:

- Adjusted R-square
- AIC
- AICC
- Average Squared Error
- F Value for Model
- Mean Square Error
- Pr > F
- R-Square
- Root MSE
- SBC

If a group-by variable is assigned to the model, the number of observations that appears at the top represents the observations for the group selected in the Variable Importance pane.

Fit Summary

- The *Fit Summary pane* shows how significant the effects (predictor) variables are to the response variable by displaying a Variable Importance pane.
- The pane is displayed for
 - linear regression
 - logistic regression
 - generalized linear model.



11

The Fit Summary pane is used to determine the most significant predictor variables that affect the response variable.

The *variable importance plot* displays the effects on the Y axis and the *p*-values on the X axis. The variable importance is based on the negative log of the *p*-value ($-\log(p\text{-value})$). A larger $-\log(p\text{-value})$ indicates a more important variable.

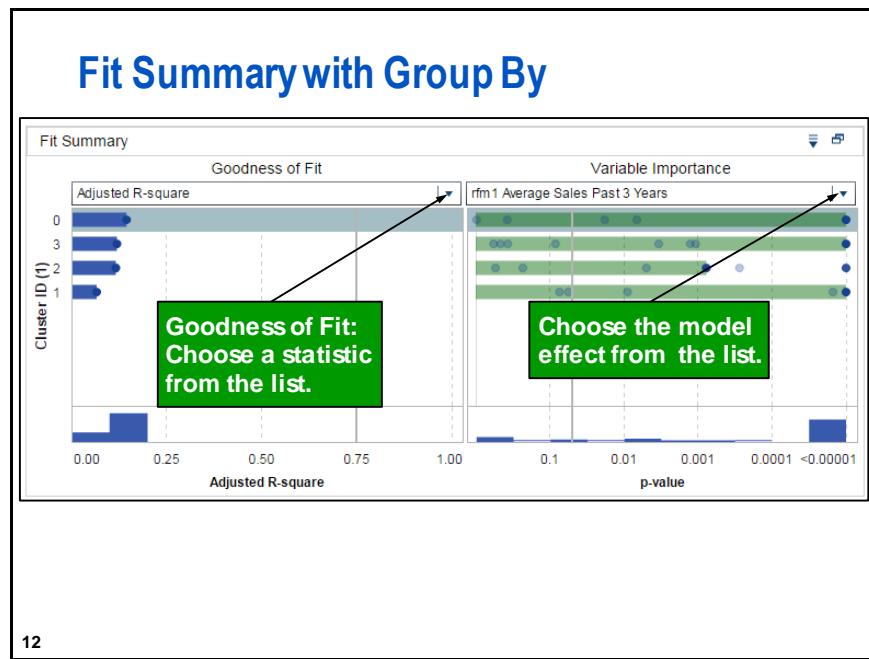
The significance of the effect is shown by the color and length of the horizontal bars. A green horizontal bar shows that the variable importance is above the significance level. The length of the green bar indicates how meaningful the variable is. A blue horizontal bar shows that the variable importance is below the significance level.

The significance level (or alpha value, which is plotted as $-\log(\alpha)$) is set to 0.05 and displayed as a black vertical line in the pane. Place the mouse pointer on the line to see the alpha and $-\log(\alpha)$ values. Click the line and you can move it to the left or right to change the significance level.

Position your pointer over the horizontal bars to see the *p*-value and the $-\log(p\text{-value})$.

Placing the pointer over the histogram bars at the bottom of the graph displays the percent of the data that is within the range.

To remove an effect from the model, click a horizontal bar, right-click, and then select **Remove**.



Fitting the model with a group-by variable in Visual Statistics is extremely efficient because the data are already distributed in memory, and per-group models can be developed without the added overhead time to access the data repeatedly.

Goodness of fit measures the model fit for each variable. The X axis displays the values for the model evaluation criteria that are selected. The Y axis displays the values of the group-by variable. You can change the model evaluation criteria by selecting different criteria from the drop-down list at the top of the Goodness of Fit pane. These are the choices:

- Adjusted R-square
- R-Square
- AIC
- AICC
- SBC
- Average Squared Error
- Observations

Place the pointer on the horizontal bars in the Goodness of Fit pane to display the group-by value, the model evaluation criteria, and the number of observations.

Position the pointer over the histogram bars at the bottom of the Goodness of Fit pane to display the range of values represented by the bar for the selected model evaluation criteria and the percentage of observations included in the bar.

In the Goodness of Fit pane, selecting and highlighting a group-by value (clicking a horizontal bar) updates the Residual Plot pane and Influence Plot pane to display only the observations in that segment.

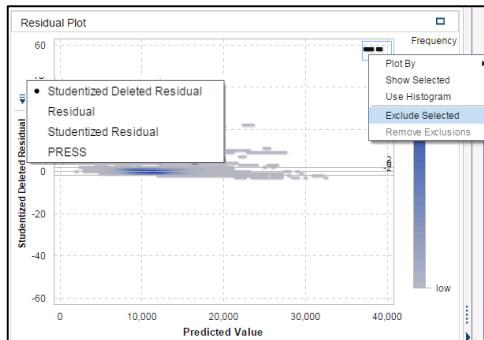
When you assign a group-by variable, the Variable Importance pane displays the p -values for a selected model effect for each value of the group-by variable. You see what equates to a linear scatter plot of the p -values. You can position your pointer on the points to display the BY-group value, the effect name, the p -value, and the $-\log(p\text{-value})$.

Place the pointer on the histogram bars at the bottom of the Variable Importance pane to display the range of all p -values represented by the bar for the selected model effect and the percentage of observations included in the bar.

You can sort the order of the variables by either Goodness of Fit or Variable Importance by selecting the icon in the top right of the pane.

Residual Plot

- The *Residual Plot* pane displays the difference between the predicted data and the actual data using a scatter plot or a heat map.
- The pane is displayed for
 - linear regression
 - logistic regression
 - generalized linear model.



13

The residual plot is used to assess the quality of the model and to identify outlier observations. The plot appears as either a scatter plot for smaller data or as a heat map when used with larger data. If it is displayed as a heat map, a color-coded frequency bar appears on the right side of the graph. The bar represents the frequency of each intersection of values. As the values become darker, the frequency increases.

The residual plot displays the predicted value of the response variable on the X axis and the studentized deleted residual statistic on the Y axis. You can choose a different statistic for the Y axis by clicking **Studentized Deleted Residual** and selecting another from the pop-up menu. (See the slide.) These are the choices:

- Studentized Deleted Residual
- Residual
- Studentized Residual
- PRESS

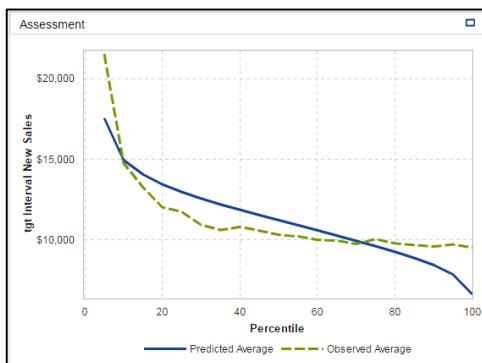
If the residual plot shows most of the points between -2 and +2 (hence, the reference lines on the plot), then the linear model is a good fit.

Selecting one or more observations on the residual plot highlights the corresponding bars in the influence plot if the observations have a significant influence on the model.

Residual plots have several uses when you examine a model. First, obvious patterns in the residual plot indicate that the model might not fit the data. Second, residual plots can detect nonconstant variance in the input data when you plot the residuals against the predicted value. Nonconstant variance is evident when the relative spread of the residual values changes as the predicted values change. Third, in combination with other methods, the residual plot can help identify outliers in your data.

Assessment

- The *Assessment pane* displays the values for the observed response along with the model's predicted response.
- The pane is displayed for
 - linear regression
 - logistic regression
 - generalized linear model
 - decision tree.



14

The *assessment plot* is another indicator of how well the model fits the data.

The assessment plot displays the values for the observed response and the model's predicted response on the Y axis, and the cumulative percentage of observations along the X axis. When the Y-axis values mirror each other more closely, the model fit is better.

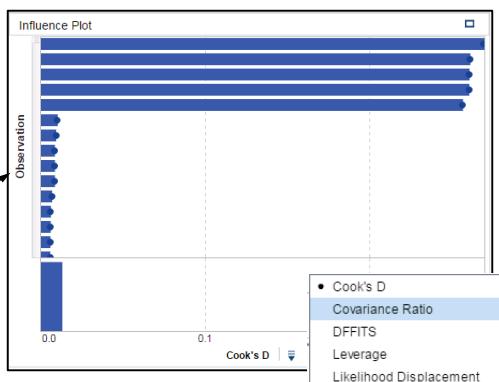
If a group-by variable is assigned, the assessment plot displays the data from the group selected in the Fit Summary pane.

Place your pointer on the plotted lines at every fifth percentile (5, 10, 15, and so on) to view the data for the selected points.

Influence Plot

- The *Influence Plot pane* displays the observations that might influence the overall analysis.
- The pane is displayed for
 - linear regression
 - logistic regression.

Only the top 2000 observations are shown.



15

The influence plot is used to determine which observations yield a high level of influence on the model. Observations are influential if they are outliers and exhibit leverage. *Leverage* refers to an observation with an extreme value for a predictor variable. An observation is said to be *influential* if removing it from the model substantially changes the regression parameter estimates. An influence plot can help determine whether outliers should be removed from the model or at least deserve more scrutiny.

Only the top 2000 observations are included in the influence plot by default. If you exclude any, the display is refreshed to keep the number of observations that are displayed to 2000.

The influence plot displays the individual observations on the Y axis and the estimate of the influence criteria on the X axis, which is Cook's D by default. You can choose a different influence statistic for the X axis by clicking **Cook's D** and choosing another statistic from the pop-up menu. (See the slide.) These are the choices:

- Cook's D
- Covariance Ratio
- DFFITS
- Leverage
- Likelihood Displacement

The horizontal blue bars can represent either individual observations or multiple observations that have the same values for all effects variables. The length of the bar shows how far the observations are from the normal. Selecting one or more bars displays the corresponding points (or bins) in the Residual Plot pane. Positioning your pointer on a bar displays the values for all of the variables, as well as the value for the estimate of influence criteria.

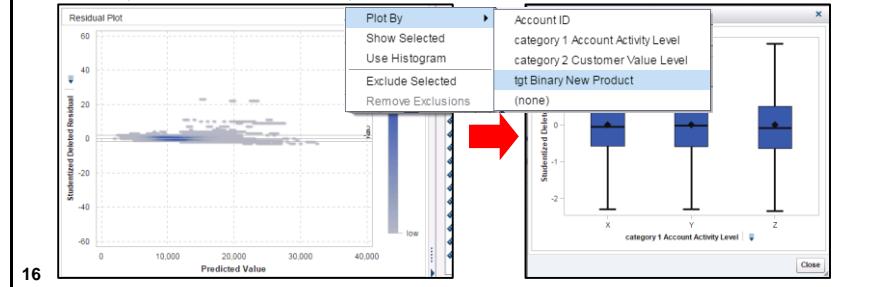
Select one or more observations, right-click, and select **Exclude Selected** to remove them from the model. The model is refitted automatically (unless you cleared the **Auto-update** check box).

Clicking any of the histogram bars at the bottom of the Influence Plot pane displays the observations that are represented by the bar.

Residual Plot and Influence Plot: Viewing Options

You can do the following:

- display a box plot of all observations against a categorical variable (**Plot By** $\Rightarrow \{category\}$)
- select data points and display additional data (**Show Selected**)
- display a histogram of all observations (**Use Histogram**)
- exclude selected data points (**Exclude Selected**) or not (**Remove Exclusion**)



16

Right-click either a residual plot or influence plot and select **Plot by**. This opens a submenu from which you select a categorical variable. After that selection, a box plot appears that has the model evaluation criteria on the Y axis and the categorical variable on the X axis. The box plot shows the distribution of values using a rectangular box and lines called *whiskers*. Position your pointer on any part of the individual box plots to display the minimum, maximum, first and third quartiles, mean, median, and standard deviation. Right-click anywhere on the box plot, and a pop-up menu appears with options to assign the categorical variable as either a classification effect (**Assign to Classification**) or a group-by variable (**Assign to Group By**). You can also select **Show outliers** on the box plot. The categorical variable on the X axis can be changed by clicking the category at the bottom of the box plot and selecting another value from the pop-up menu. To close the box plot, select **Close**.

Selecting **Show selected** from the pop-up menu displays additional data for the selected observations in a table view.

Right-click and select **Use histogram** to display the distribution of values for the model evaluation criteria along the X axis and the count of observations along the Y axis. Position your pointer on any of the histogram bars to view the model evaluation criteria value, count, and percent represented by the bar. To change the model evaluation criteria used on the X axis, click it at the bottom of the histogram and select another value from the pop-up menu. To close the histogram view and return to the residual or influence plot, right-click anywhere on the histogram and clear the **Use histogram** check box from the pop-up menu.

You can select one or more observations and remove them from the model by selecting them. Then right-click and select **Exclude selected** from the pop-up list. The observation number at the top of the model is updated to reflect the data that are still included in the model. The Filters tab also indicates that data were excluded from the results. To re-include the removed observations, right-click **Residual Plot** or **Influence Plot** and select **Remove exclusion** or click the **Filters** tab. Then select **Remove All** from the drop-down list for excluded data.

Show Details

- The *Summary Table pane* provides detailed statistics about the model via the tabs, which are model dependent.
 - Overall ANOVA
 - Dimensions
 - Fit Statistics
 - Model ANOVA
 - Type III Test
 - Parameter Estimates
 - Iteration History
 - Convergence
 - Response Profile
- To display the summary table, click **Show details** in the menu in the upper right of the Model pane.

Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates	
Source	Deg Freedom	Sum of Squares	Mean Square	F Value	Pr > F	R-Square
Model	18	1.423E12	7.907E10	1209.329	<0.0001	0.093321
Error	211490	1.383E13	65366826			
Corrected Total	211508	1.525E13				

17

The specific information in each summary table varies based on the model. The display above is from a linear regression.

To hide the summary table, click **Hide Summary Table** in the upper right of the Model pane.

If the summary table is enabled and you maximize any of the individual panes, the summary table remains open at the bottom of the Model pane.

Linear Regression: Summary Table

- Overall ANOVA
- Dimensions
- Fit Statistics
- Model ANOVA
- Type III Test
- Parameter Estimates

18

Linear Regression: Overall ANOVA

The *Overall ANOVA* tab provides information about how well the model fits the data.

- Source – source of the variation in the data
- Deg Freedom – degrees of freedom associated with the source
- Sum of Squares – sum of squared errors of prediction
- Mean Square – Sum of Squares \div Deg Freedom
- F Value – is Model Mean Square \div Error Mean Square
- Pr > F – p-value
- R-Square – the proportion of variation in the response variable explained by the factors in the model

Overall ANOVA						
Source	Deg Freedom	Sum of Squares	Mean Square	FValue	Pr > F	R-Square
Model	18	1.423E12	7.907E10	1209.329	<0.0001	
Error	214490	1.383E13	65366826	.	.	
Corrected Total	211508	1.525E13	.	.	.	

19

The Overall ANOVA tab provides analysis-of-variance results for the model, error, and corrected total.

The model fits the data well when the p-value (Pr > F) is less than .05 and the R-square is high.

Linear Regression: Dimensions

The *Dimensions* tab displays information about the number of effects and observations included in the model.

Overall ANOVA		Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Description						Value
Number of Model Effects						15
Number of Classification Effects						2
Number of Columns in X						21
Rank of Cross-product Matrix						19
Number of Observations Read						1,060,038
Number of Observations Used						211,509

20

The Dimensions tab provides an overview of the effect variables used in the model. This tab identifies how many model and classification effects were chosen for the model, the rank of the cross-product matrix, how many observations were read, and how many observations were used in the model.

If the rank of the cross-product matrix does not equal the number of columns in X, then the difference is equal to the number of parameters set to 0.

Linear Regression: Fit Statistics

The *Fit Statistics* tab displays statistics about the estimated model.

Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Statistic					Value
Mean Square Error					65386826
Root MSE					8086.212
F Value for Model					1209.329
Pr > F					0
R-Square					0.093321
Adjusted R-square					0.093244
AIC					4017810
AICC					4017810
SBC					3806494
Average Squared Error					65380952

21

Linear Regression: Model ANOVA

The *Model ANOVA* tab displays the *p*-value for testing the significance of each model effect considering all other effects in the model.

Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Source	Deg Freedom	Sum of Squares	Mean Square	F Value	Pr > F
logi_rf1 Average Sales Past 3 Years	1	4.745E10	4.745E10	725.6634	<0.0001
logi_rf10 Count Total Promos Past ...	1	44153768	44153768	0.67527	0.4112
logi_rf11 Count Direct Promos Pas...	1	3562368	3562368	0.054328	0.8157
logi_rf12 Customer Tenure	1	6.8191E8	6.8191E8	10.42883	0.0012
logi_rf2 Average Sales Lifetime	1	7.365E10	7.365E10	1126.386	<0.0001
logi_rf3 Avg Sales Past 3 Years Di...	1	5.435E10	5.435E10	831.2135	<0.0001
logi_rf4 Last Product Purchase Am...	1	8.7331E9	8.7331E9	133.561	<0.0001
logi_rf5 Count Purchased Past 3 Y...	1	2.362E10	2.362E10	361.2471	<0.0001
logi_rf6 Count Purchased Lifetime	1	10348511	10348511	0.158266	0.6908
logi_rf7 Count PrchsD Past 3 Years...	1	16458828	16458828	0.251715	0.6159
logi_rf8 Count PrchsD Lifetime Dir...	1	1.2262E9	1.2262E9	18.75241	<0.0001
logi_rf9 Months Since Last Purchase	1	3.423E10	3.423E10	523.4444	<0.0001
category 1 Account Activity Level	2	4.4634E9	2.2317E9	34.13108	<0.0001
category 2 Customer Value Level	4	4.777E10	1.194E10	182.637	<0.0001

22

Linear Regression: Type III Test

The *Type III Test* tab displays the significance of each partial effect considering all other effects in the model.

Effect	DF	DenDF	F Value	Pr > F
logi_rf1 Average Sales Past 3 Years	1	211490	725.6634	<0.0001
logi_rf10 Count Total Promos Past...	1	211490	0.67527	0.4112
logi_rf11 Count DirectPromos Pas...	1	211490	0.054328	0.8157
logi_rf12 Customer Tenure	1	211490	10.42883	0.0012
logi_rf2 Average Sales Lifetime	1	211490	1126.386	<0.0001
logi_rf3 Avg Sales Past 3 Years Di...	1	211490	831.2135	<0.0001
logi_rf4 Last Product Purchase Am...	1	211490	133.561	<0.0001
logi_rf5 Count Purchased Past 3 Y...	1	211490	361.2471	<0.0001
logi_rf6 Count Purchased Lifetime	1	211490	0.158266	0.6908
logi_rf7 Count Prchsld Past 3 Years...	1	211490	0.251715	0.6159
logi_rf8 Count Prchsld Lifetime Dir...	1	211490	18.75241	<0.0001
logi_rf9 Months Since Last Purchase	1	211490	523.4444	<0.0001
category 1 Account Activity Level	2	211490	34.13108	<0.0001
category 2 Customer Value Level	4	211490	182.637	<0.0001

23

The Type III Test tab examines the significance of each partial effect with all other effects in the model. For more information, see the chapter “The Four Types of Estimable Functions” in *SAS/STAT® 13.2 User’s Guide*.

Linear Regression: Parameter Estimates

The *Parameter Estimates* tab displays the parameter estimates (coefficients) of each model effect and their associated statistics.

Parameter	Estimate	Standard Error	t Value	Pr > t
Intercept	-1539.02	395.0642	-3.89561	<0.0001
logi_rf1 Average Sales Past 3 Years	3157.097	117.198	26.93814	<0.0001
logi_rf10 Count Total Promos Past...	-85.1537	103.6251	-0.82175	0.4112
logi_rf11 Count DirectPromos Pas...	-28.8847	123.9234	-0.23308	0.8157
logi_rf12 Customer Tenure	246.8584	76.44165	3.22937	0.0012
logi_rf2 Average Sales Lifetime	3690.922	109.9743	33.56167	<0.0001
logi_rf3 Avg Sales Past 3 Years Di...	-2542.11	88.17358	-28.8308	<0.0001
logi_rf4 Last Product Purchase Am...	601.1123	52.01347	11.55686	<0.0001
logi_rf5 Count Purchased Past 3 Y...	-1667.51	87.73354	-19.0065	<0.0001
logi_rf6 Count Purchased Lifetime	-35.1153	88.26786	-0.39783	0.6908
logi_rf7 Count Prchsld Past 3 Years...	34.57096	68.90601	0.501712	0.6159
logi_rf8 Count Prchsld Lifetime Dir...	364.5495	84.18368	4.330406	<0.0001
logi_rf9 Months Since Last Purchase	1495.977	65.38674	22.87891	<0.0001
category 1 Account Activity Level	-744.835	96.22958	-7.74019	<0.0001
category 1 Account Activity Level	-903.505	114.9419	-7.86054	<0.0001
category 1 Account Activity Level	0	.	.	.
category 2 Customer Value Level	-1277.33	49.91534	-25.59	<0.0001
category 2 Customer Value Level	-695.821	51.89858	-13.4073	<0.0001
category 2 Customer Value Level	-210.774	55.60406	-3.79063	0.0002
category 2 Customer Value Level	-428.271	64.57968	-6.63167	<0.0001
category 2 Customer Value Level	0	.	.	.

24

The actual model equation can be completed using the Estimate column, where the numbers represent the coefficients for each of the variables in the model.

4.01 Poll

Linear regression models are appropriate for any continuous response variable.

- True
- False



Building a Linear Regression

This demonstration illustrates using the **VS_BANK** data set to build, explore, and refine a linear regression model in SAS Visual Statistics.

Creating and Exploring the Linear Regression Model

1. Return to a previous exploration in which the **VS_BANK** data source is loaded into memory.
2. Click the **New Visualization** icon and select **Linear Regression** to start a new linear regression analysis.
3. Clear the **Auto-update model** check box at the bottom right of the window.
4. Assign **tgt interval New Sales** as the response variable.

You can also drag and drop **tgt interval New Sales** into the Linear Regression workspace.
5. Assign the **logi_rfm_nn** variables as continuous effects.
6. Assign **category 1 Account Activity Level** and **category 2 Customer Value Level** as classification effects.

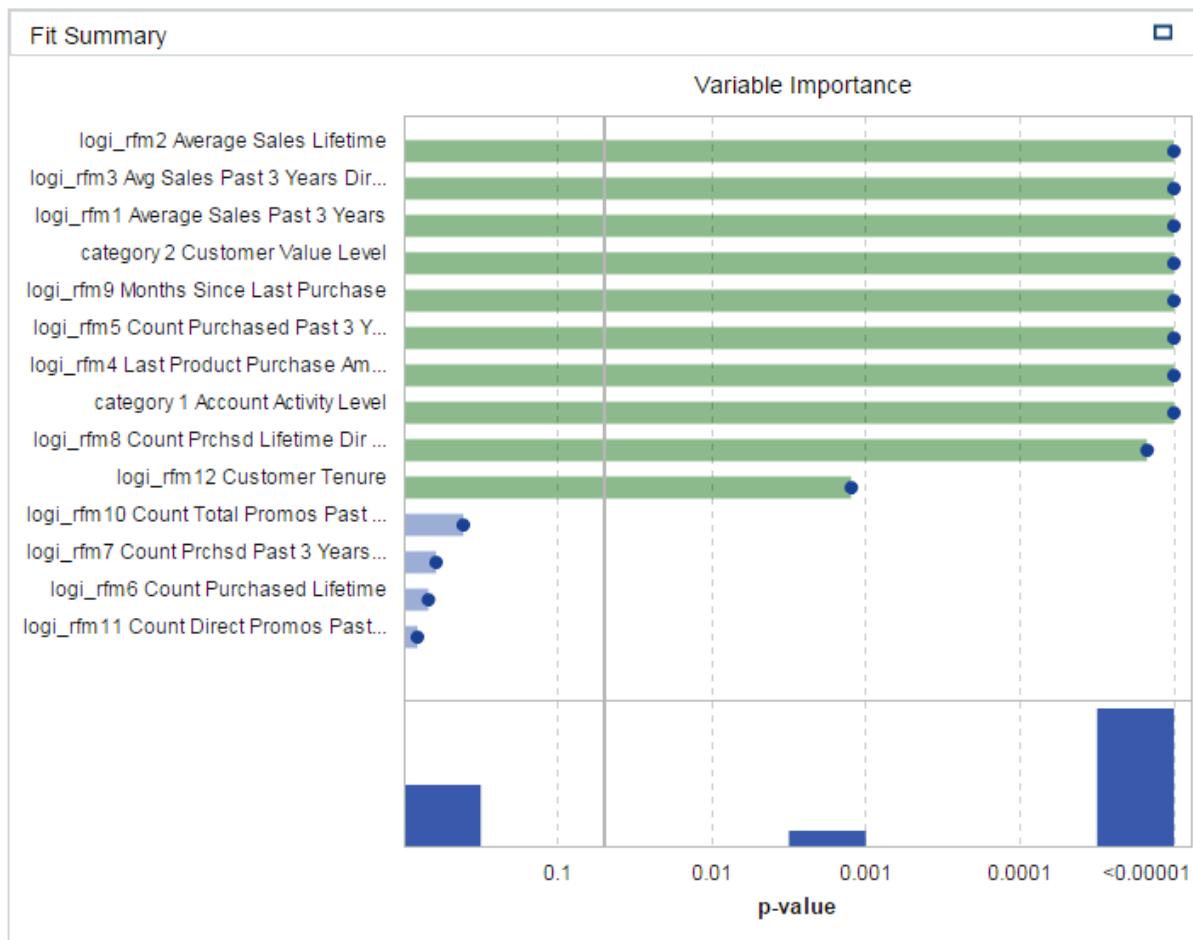
Target and input variable roles for the model are summarized below.

The screenshot shows the 'Roles' pane of the SAS Visual Statistics interface. At the top, it says 'Linear Regression' and 'Use Automatic Chart'. Below that, 'Data source: VS_BANK' is listed. Under 'Response', the target variable is 'tgt Interval New Sales'. In the 'Continuous Effects' section, there is a long list of variables starting with 'logi_rfm1 Average Sales Past 3 Years' and ending with 'logi_rfm9 Months Since Last Purchase'. In the 'Classification Effects' section, there are two categories: 'category 1 Account Activity Level' and 'category 2 Customer Value Level'.

7. Click the **Update** button to run the model.
8. The R-square is .0933. The R-square measure seems low, but further model refinement, illustrated below, might improve it. The model uses 211,509 observations.

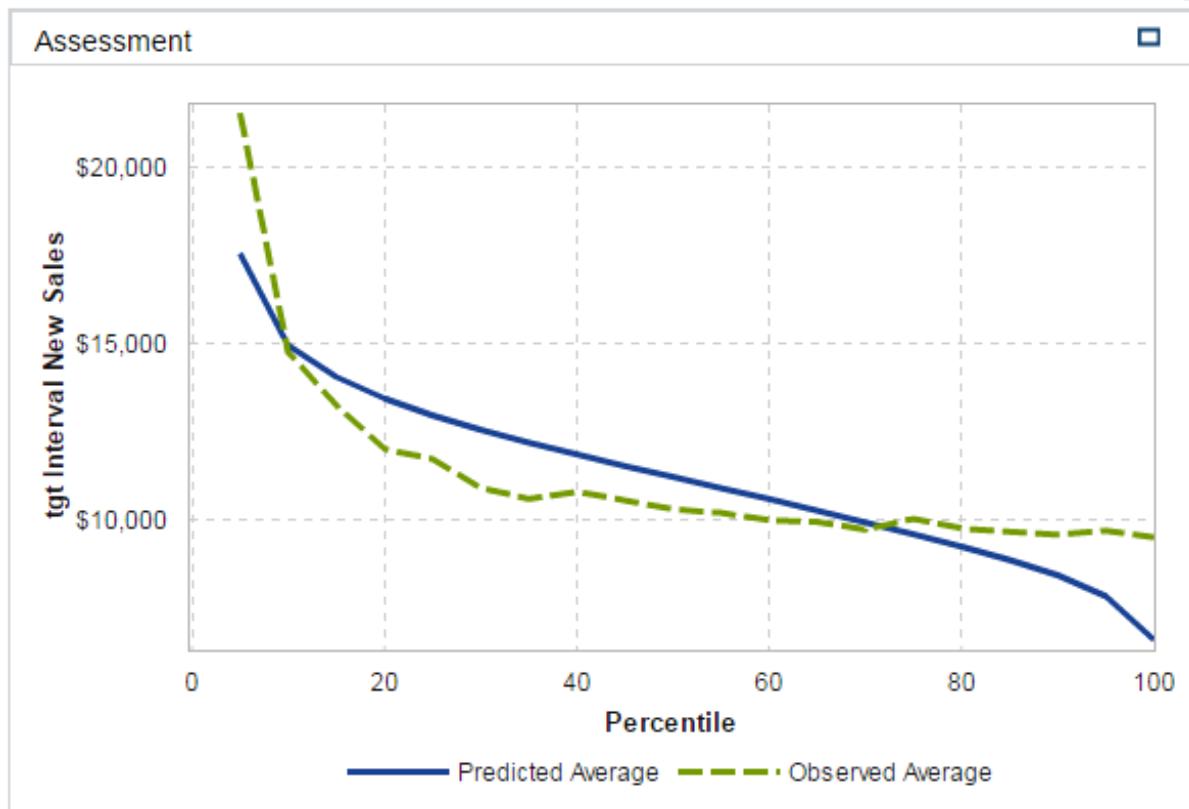
The number of observations that is used seems low. The data set contains more than a million accounts or customers. Recall that the data has a 20% response rate. For the interval-valued target, used in this analysis, non-responders are coded with a missing value. The linear regression model is fitted using only responders in the data.
9. You can investigate other fit measures in addition to R-square. Click **R-Square** and select **Root Mean Square Error**. The RMSE is 8086.21. This statistic tells you that, on average and over the range of the data, the difference between the prediction and the actual value is approximately \$8,086. This seems imprecise and might be an artifact of how the data were collected. Ways to improve this measure are explored in the model refinement part of this demonstration.
10. Maximize the Fit Summary pane.

A default criterion, $p\text{-value} < .05$, is used as a threshold for variable importance. Effects with p -values less than .05 have green horizontal bars extending to the right side of the plot, and effects with p -values greater than .05 have blue bars. A bar chart below shows the distribution of effects in various ranges of p -values on the negative \log_{10} scale.



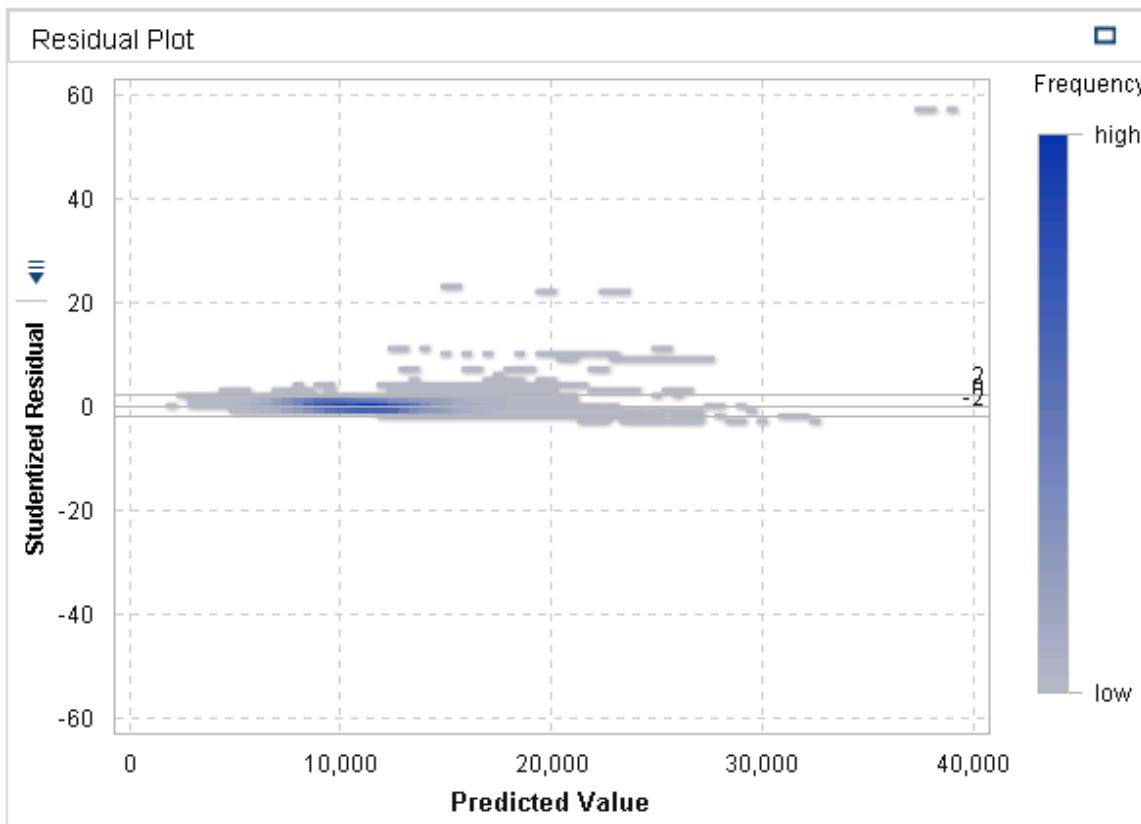
Some of the logged, imputed RFM variables (1, 2, 3, 4, 5, 8, 9, and 12), as well as both categorical inputs, are important predictors in the model according to the criterion listed above. (Functionality for removing irrelevant input variables from the model is discussed below.)

11. Restore the Fit Summary pane and maximize the assessment plot.



The line plots show model predictions versus actual response amount in the data. To create the lines, both outcomes and predictions are binned into percentiles. Although the model's predictions seem consistent with responder outcomes over the middle range of the plot, the model seems to under-predict the response amount at the high and low end of the response range.

12. Restore the assessment plot and maximize the residual plot.
13. Change the residual measure to studentized residual. Use the button adjacent to the vertical axis in the plot.



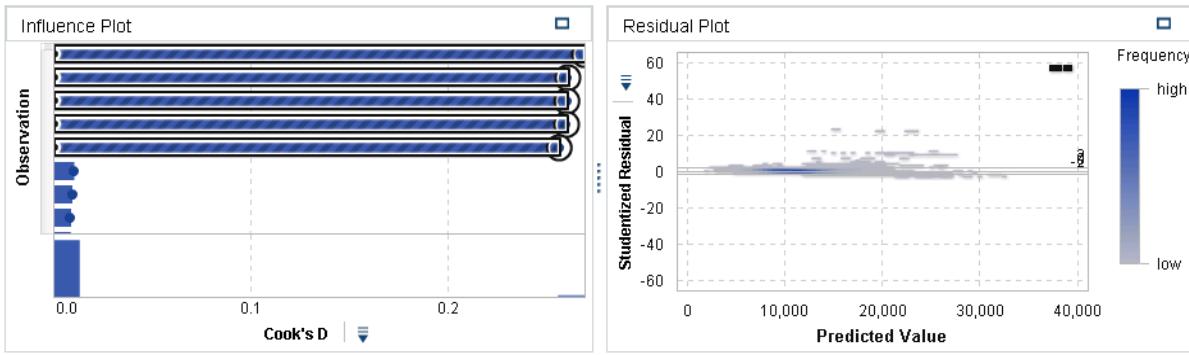
A studentized residual is a raw residual that was divided by its estimated standard deviation.

Two features of this plot merit further investigation:

- The variance of the residuals seems to increase as a function of the predicted value. (Causes and possible remedies for heteroscedasticity are explored later in this demonstration.)
- There are some very large, positive outliers in the residuals. The influence plot can help you explore the effect of these outliers on the fitted model.

14. Restore the residual plot and drag the influence plot around so that the two plots are shown side-by-side.
15. Change the influence measure to likelihood displacement and then click the top bar in the influence plot. Drag the mouse pointer over the tallest adjacent bars to select them.

Notice the interaction between the influence plot and the residual plot.



The bars in the influence plot correspond to observations (accounts) in the data. The selected observations are *high leverage* in the sense that, when they are removed from the data, the maximized value of the function used to generate the parameter estimates changes substantially, which results in likelihood displacement.

- Each bar on the influence plot can represent more than one observation. This happens when two or more observations are influential, and they have the same column (target and predictor) values.

16. Right-click the selected bars and select **Show Selected**. Column values associated with these observations are listed.

Selected Points							
Account ID	category 1 Acc...	category 2 Cus...	demog Custom...	demog Female ...	demog Home V...	demog Homeo...	demog Incon...
100881582	Y	E			\$65,021		\$64,7
100303380	X	E			\$65,023		\$64,7
100399747	X	E			\$65,017		\$64,7
100110646	X	E			\$64,995		\$64,7
100785215	X	E			\$64,998		\$64,7
100977949	X	E			\$65,007		\$64,7
100592481	X	E		1	\$65,003		\$64,7
100014279	X	E			\$64,972		\$64,7
100207013	X	E		1	\$64,996		\$64,7
100496114	X	E		1	\$65,028		\$64,7
100688848	X	E			\$64,988		\$64,7

17. Close the Selected Points window.
18. Right-click the selected bars and select **Exclude Selected**.
19. Select **Update** to incorporate the changes into the model.

Although the R-squared measure for the model improves slightly when these observations are removed from the analysis, the main model problems that are identified (bias and heteroscedasticity) are not substantially impacted.

20. Right-click the influence plot and select **Remove Exclusion**.
21. Click the **Update** button to restore the original model.

Refining the Linear Regression Model

Model Selection

1. Select **Show Details** from the triangle menu at the top right of the main window.
2. Click the **Type III Test** tab.

Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Effect		DF	DenDF	F Value	Pr > F
logi_rfml Average Sales Past 3 Years		1	211490	725.6634	<0.0001
logi_rfml0 Count Total Promos Past ...		1	211490	0.67527	0.4112
logi_rfml1 Count Direct Promos Pas...		1	211490	0.054328	0.8157
logi_rfml2 Customer Tenure		1	211490	10.42883	0.0012
logi_rfml3 Average Sales Lifetime		1	211490	1126.386	<0.0001
logi_rfml4 Avg Sales Past 3 Years Di...		1	211490	831.2135	<0.0001
logi_rfml5 Last Product Purchase Am...		1	211490	133.561	<0.0001
logi_rfml6 Count Purchased Past 3 Y...		1	211490	361.2471	<0.0001
logi_rfml7 Count Purchased Lifetime		1	211490	0.158266	0.6908
logi_rfml8 Count Purchased Past 3 Years...		1	211490	0.251715	0.8159
logi_rfml9 Count Purchased Lifetime Dir...		1	211490	18.75241	<0.0001
logi_rfml0 Months Since Last Purchase		1	211490	523.4444	<0.0001
category 1 Account Activity Level		2	211490	34.13108	<0.0001
category 2 Customer Value Level		4	211490	182.637	<0.0001

The Type III test represents the relative importance of the input variables included in the model. The *p*-value measures the significance of each variable, given that all other variables are included in the model. The results indicate that at least two of the input variables might be irrelevant for predicting loan amount.



The widely used 0.05 rule-of-thumb threshold for variable significance was conceived on much smaller data sets than the one used in this course. A best practice for building models on “big data” is to use a much stricter *p*-value threshold. A 0.01 *p*-value threshold is used to infer input variable significance. Further information about this topic is provided in Rafferty (1995)¹.

3. Click the **Properties** tab.
4. Select the **Use variable selection** check box, and set the significance level to **0.01**.

Informative missingness

Use variable selection

Significance level: 0.01 ▲ ▼



A fast backward-selection algorithm is used for model selection in SAS Visual Statistics.

¹ Rafferty, A.E. 1995. “Bayesian model selection in social research.” *Sociological Methodology* 111-196.

5. Select **Update** to run the model.

The Type III test results summarize the effects of variable selection. Variables listed with a missing *p*-value have zero associated degrees of freedom and do not contribute to the fitted model.

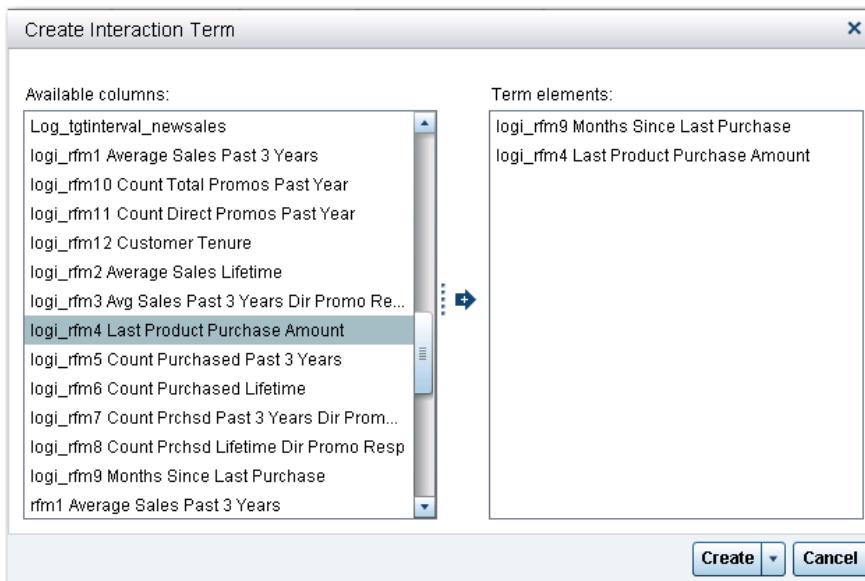
Overall ANOVA	Dimensions	Fit Statistics	Model ANOVA	Type III Test	Parameter Estimates
Effect	DF	DenDF	F Value		Pr > F
logi_rfim1 Average Sales Past 3 Years	1	211494	725.4979		<0.0001
logi_rfim10 Count Total Promos Past ...	0	211494	.		.
logi_rfim11 Count Direct Promos Pas...	0	211494	.		.
logi_rfim12 Customer Tenure	1	211494	11.45196		0.0007
logi_rfim2 Average Sales Lifetime	1	211494	1168.601		<0.0001
logi_rfim3 Avg Sales Past 3 Years Di...	1	211494	849.6745		<0.0001
logi_rfim4 Last Product Purchase Am...	1	211494	132.8222		<0.0001
logi_rfim5 Count Purchased Past 3 Y...	1	211494	822.1579		<0.0001
logi_rfim6 Count Purchased Lifetime	0	211494	.		.
logi_rfim7 Count Prchsd Past 3 Years..	0	211494	.		.
logi_rfim8 Count Prchsd Lifetime Dir...	1	211494	51.68762		<0.0001
logi_rfim9 Months Since Last Purchase	1	211494	845.2467		<0.0001
category 1 Account Activity Level	2	211494	33.58721		<0.0001
category 2 Customer Value Level	4	211494	182.645		<0.0001

Although the variable selection process did not substantially improve the main problems in the model (noted above), it did result in a more defensible, and from an exploratory point of view, useful specification.

Interactions

Domain experts hypothesize interactions between input variables in the data. The Interaction functionality in SAS Visual Statistics is accessed under the Roles portion of the interface.

1. Click the **Roles** tab. Select the **Create** toggle next to **Interactions**.
2. Highlight **logi_rfim9 Months Since Last Purchase**, and select the arrow to move it under the Term Elements side of the table. Do the same for **logi_rfim4 Last Product Purchase Amount**.



3. Click the **Create** button. The new interaction term appears at the bottom of the list of variables.



Domain experts suggest that there is a negative correlation between purchase amount and purchase recency. That is, customers who purchased recently tend to buy smaller amounts (obtain smaller loans) than customer who purchased less recently.

4. Click **Update** to refresh the model.

The Type III Test table provides evidence in favor of the hypothesis regarding purchase recency and amount. The interaction term is selected and included in the model.

Filtering

The majority of accounts in the data are consumers of short- to medium-term loans. These include home equity lines of credit and automobile loans. Historically, these are in the \$2,000 to \$50,000 range for the bank. However, some loans in the data are less than \$2,000 and are similar to “payday” loans. Also, some loan amounts greatly exceed \$50,000. If accounts at the high and low end of the target variable range are very different, in terms of systematic relationships between predictors and loan quantity demanded, then bias and heteroscedasticity could result. The following analysis assesses the impact of observations in the tails of the target variable on linear regression model results:

1. Click the **Filters** tab. (Clear the **Auto-update** check box if it is selected.)
2. Select and drag the **tgt Interval New Sales** variable to the Visualization area of the Filters tab.

The existing range for new sales amounts is \$0 to \$500,000. To be consistent with the loan profile for this portfolio, modify the filter range so that it is similar to the range shown below. Enter the filter values at **2000** and **50000**.



3. Click **Update** to apply the filter.

The diagnostics indicate these results:

- The filter greatly reduces the range of the target variable. However, only approximately 10% of the responders are excluded by the filter.
- Root Mean Square Error is reduced to 6031.4372.
- Problems with bias and violations of error assumptions, noted earlier, are partially mitigated.

Further exploration using filtering can help analysts understand whether homogeneous groups of responder accounts exist within the distribution of the target variable. Subsequent prediction quality can be improved if separate modeling exercises are performed for different groups of responders.

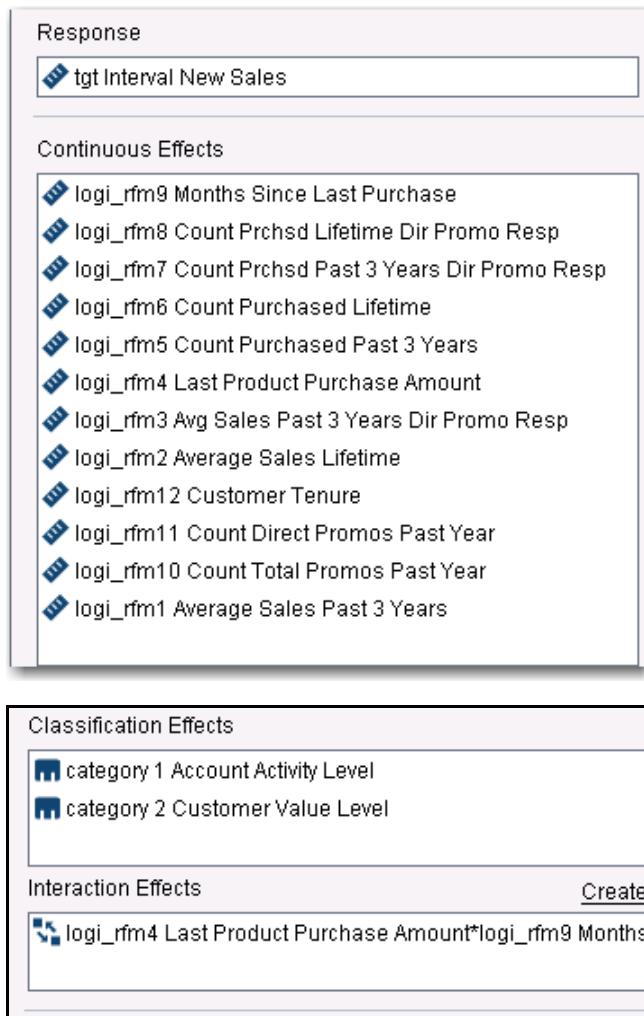
Generalizing the Linear Model

A final exploration on the interval target variable considers its distribution. Under the linear model assumptions, errors are normal and independent, and have constant variance. That is, the variance does not change as a function of the expected value of the target.

The normal distribution is symmetric and has the real number line as its domain. The target variable for the analysis is right skewed and is restricted to be nonnegative. Also, the residual plot shows evidence that the variance of the residuals increases as the prediction quantity increases. Modifying the target or generalizing the model to more closely match the assumptions that regulate the analysis should improve the quality of the model and make it more useful for understanding the relationships in the data.

In this demonstration, the target variable is modified. Alternative models are considered in the next section. A widely used approach for modeling continuous, nonnegative, and right-skewed variables is to assume that they have a log-normal distribution. In applications, this approach is implemented by transforming the target variable using a natural log.

-  If a variable has a log-normal distribution, then its log has a normal distribution.
1. Click **Visualization** \Rightarrow **New**. Maximize the new visualization window.
 2. Click the **Generalized linear model** button on the toolbar.
 3. Clear the **Auto-update** check box.
 4. Assign variable roles for the model as you did for the linear model. These are summarized below.



Response

tgt Interval New Sales

Continuous Effects

- logi_rfm9 Months Since Last Purchase
- logi_rfm8 Count Prchsd Lifetime Dir Promo Resp
- logi_rfm7 Count Prchsd Past 3 Years Dir Promo Resp
- logi_rfm6 Count Purchased Lifetime
- logi_rfm5 Count Purchased Past 3 Years
- logi_rfm4 Last Product Purchase Amount
- logi_rfm3 Avg Sales Past 3 Years Dir Promo Resp
- logi_rfm2 Average Sales Lifetime
- logi_rfm12 Customer Tenure
- logi_rfm11 Count Direct Promos Past Year
- logi_rfm10 Count Total Promos Past Year
- logi_rfm1 Average Sales Past 3 Years

Classification Effects

- category 1 Account Activity Level
- category 2 Customer Value Level

Interaction Effects

Create

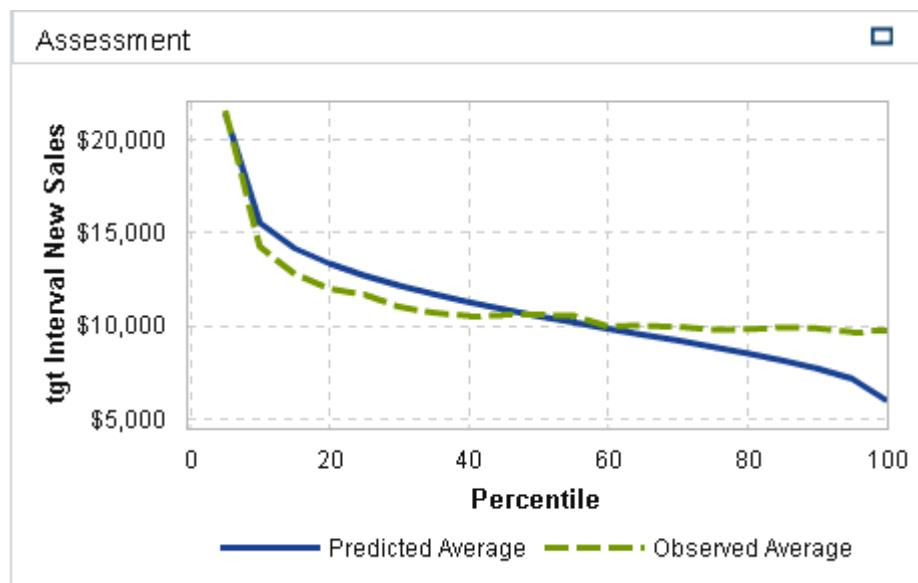
logi_rfm4 Last Product Purchase Amount*logi_rfm9 Months

5. Click the **Properties** tab, and change the **Link function** field to **Log**.
6. Click **Update** to run the model.

The Type III Test table suggests that, under the modified modeling assumptions, all included input effects are significant for predicting **tgt Interval New Sales**.

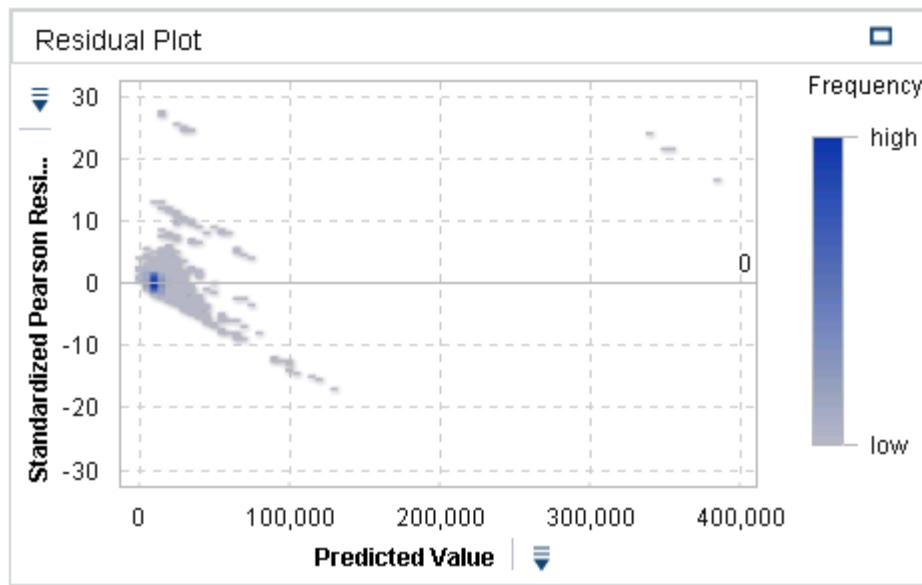
Effect	DF	Chi-Square	Pr > ChiSq
logi_rf9 Months Since Last Purchase	1	794.9024	<0.0001
logi_rf8 Count Prchsd Lifetime Dir ...	1	467.3419	<0.0001
logi_rf7 Count Prchsd Past 3 Years ...	1	98.85134	<0.0001
logi_rf6 Count Purchased Lifetime	1	451.7614	<0.0001
logi_rf5 Count Purchased Past 3 Ye...	1	871.1604	<0.0001
logi_rf4 Last Product Purchase Amo...	1	358.098	<0.0001
logi_rf3 Avg Sales Past 3 Years Dir...	1	18490.29	<0.0001
logi_rf2 Average Sales Lifetime	1	854.3417	<0.0001
logi_rf12 Customer Tenure	1	122.9872	<0.0001
logi_rf11 Count Direct Promos Past...	1	179.7251	<0.0001
logi_rf10 Count Total Promos Past ...	1	18.18943	<0.0001
logi_rf1 Average Sales Past 3 Years	1	8359.256	<0.0001
category 1 Account Activity Level	2	144.6118	<0.0001

The assessment plot shows evidence that model-prediction bias, at least at the lower end of the target variable range, is mitigated.



The residual plot still shows signs of heteroscedastic residual variance. This and the remaining bias, seen above, might be a result of the very large target observations, noted earlier.

Look at the residuals plot. Change it to *Standardized Pearson Residuals*.



The plot indicates some possible problems with the fitted model.

7. Save the project as **VA_BankInterval**.

End of Demonstration



Exercises

1. Building a Linear Model to Predict the Amount of a Donation

In this exercise, you use the **PVA97NK** data to build and refine a linear regression model. This model attempts to predict the amount of donation given to the PVA.

The exercises provide practice for using SAS Visual Statistics to do exploratory modeling.

- Exercise setup: Create the exercise project and linear model analysis.

- 1) Launch SAS Visual Analytics.
- 2) Select the **PVA97NK** data source and open it.
- 3) Click the **Linear Regression** tab, clear the **Auto-Update** box, and assign the variable roles as shown below.

The screenshot shows the SAS Visual Analytics interface with the 'Linear Regression' tab selected. The 'Response' field is set to 'Target Gift Amount'. The 'Continuous Effects' section contains a long list of variables, many of which are related to gift amounts and counts over different time periods (e.g., 'Age', 'Gift Amount Average 36 Months', 'Gift Count 36 Months'). Other variables include 'Median Home Value Region', 'Median Income Region', and various promotion and status categories. The 'Classification Effects' section includes 'Home Owner' and 'Gender'.

- 4) Click **Update** to run the model.
- b. How many observations were used to build the linear regression model?
- c. What is the R-square value of the model?
- d. Modify the variable measures.
 - 1) In the Data pane, change **Status Category Star All Months** from a measure to a *category*.
 - 2) Assign **Status Category 96NK** and **Status Category Star All Months** as classification effects.
- e. On the Properties tab, select both the **Informative missingness** and **Use variable selection** check boxes. Change the significance level to **0.01**. Click **Update** to rerun the model.
 - 1) How many observations were used to build the model?
 - 2) What is the R-square value of the model now?
 - 3) Does variable selection exclude any input variables?
 - 4) How many dummy variables does the informative missingness property add to the model?

Hint: These variables have a suffix of **_miss**.
- f. Domain experts hypothesize an interaction between gift frequency and gift amount. Explore this hypothesis using the **Gift Count All Months** and **Gift Amount Last** variables. Is the interaction significant?
- g. Save the project as **VS Interval Target Exercise**.

End of Exercises

Generalized Linear Models

Objectives

- Perform an overview of generalized linear models.
- Create a generalized linear regression model in SAS Visual Statistics.
- List available distributions and link functions for generalized linear models.
- Assess residuals and other model diagnostics to choose an appropriate distribution and link function.

32

Generalized Linear Model Overview

Generalized Linear Models

$$g(E(y_i)) = \beta_0 + \beta_1 x_{1i} + \dots + \beta_k x_{ki} = X\beta$$

- The distribution of the observations can come from the exponential family of distributions.
- The variance of the response variable can be expressed as a function of its mean.
- $X\beta$ is fit to a function of $E(y)$ (called a link function) suggested by the distribution of the observations:

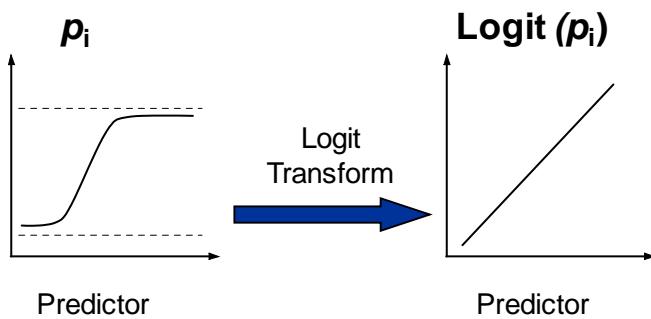
$$g(E(y)) = g(\mu) = X\beta$$

Link function

33

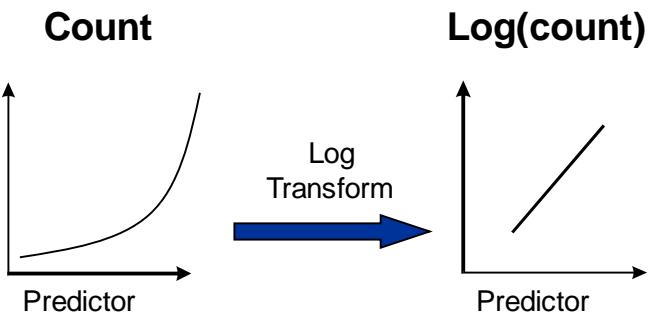
Logit Link Function for Binary Response

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$$



34

Log Link Function for Count Data



35

Examples of Generalized Linear Models

Model	Response	Distribution	Mean	Variance	Canonical Link
Linear Regression	Continuous	Normal	μ	σ^2	identity μ
Logistic regression	Dichotomous	Binomial	π	$\pi(1-\pi)/n$	logit $\log[\pi/(1-\pi)]$
Poisson Regression	Count	Poisson	λ	λ	log $\log(\lambda)$
Gamma Regression	Continuous	Gamma	μ	μ^2/ν	*inverse $1/\mu$

* Models often use the LOG link in practice.

36

Generalized Linear Models in SAS Visual Statistics

Generalized Linear Model

- Generalized linear models extend the theory and methods of linear models to data that is not normally distributed. A Link function is used to take into account the distribution of the response variable.
- There is only one continuous response variable (**Response**).
- Multiple effects (independent or predictor) variables can be any of the following:
 - continuous (**Continuous Effects**)
 - categorical (**Classification Effects**)
 - interaction terms (**Interactions**)

37

The distribution of the response variable imposes range requirements on the response variable.

The requirements are as follows:

Distribution	Range Requirements
Beta	Values must be between 0 and 1, exclusive.
Binary	Exactly two distinct values
Exponential	Nonnegative real values
Gamma	Nonnegative real values
Geometric	Positive integers
Inverse Gaussian	Positive real values
Negative Binomial	Nonnegative integers
Normal	Real values
Poisson	Nonnegative integers

4.02 Poll

You can perform linear regression analysis with a generalized linear model.

- True
- False

4.03 Poll

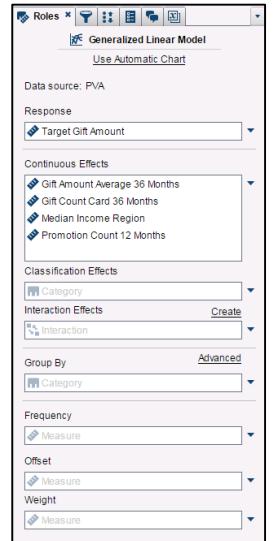
You can perform logistic regression analysis with a generalized linear model.

- True
- False

40

Generalized Linear Model Roles

- Response – assign only one measure
- Continuous Effects – assign one or more measures
- Classification Effects – assign one or more categories
- Interactions – assign one or more interaction terms
- Group By – can assign one or more categories as group-by variables, if the user chooses
- Frequency – only one measure
- Offset – only one measure
- Weight – only one measure



42

You need to create the interaction terms before you can assign them to the Interactions role.

If you specify more than one group-by variable, the groups are combined to form compound groups. (See the slides regarding the Group By role in the “Linear Regression” section for more information about the Group By role.)

Frequency – specifies the variable that is used to perform the frequency analysis for each effect. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used.

Offset – is often used in Poisson regression with the log link function to account for exposure. An offset variable is one that is treated like a regression covariate whose parameter is fixed to be 1.0.

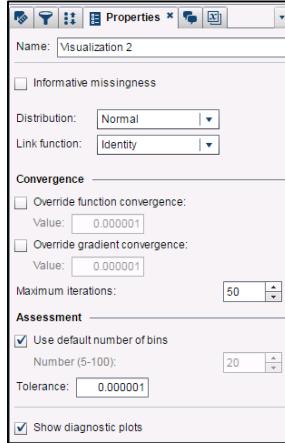
Weight – specifies the numeric variable to use as a weight variable in solving the linear model.

 Only the observations with a weight value that is not missing and greater than zero are used in the analysis.

Adding roles to the model automatically updates the model. If you do not want the model to automatically be updated when you add roles, clear the **Auto-update model** check box on the Roles tab. After you define all of the roles, you can update the model by clicking **Update** at the bottom of the Roles tab.

Generalized Linear Model Properties

- Name (of model)
- Informative missingness
- Distribution
- Link function
- Convergence
 - Override function convergence
 - Override gradient convergence
 - Maximum iterations
- Assessment
 - Use default bin count
 - Number
 - Tolerance
- Show diagnostic plots (Residual Plot, Assessment)



Informative missingness – requests that missing values be handled by modeling them through extra model effects. These effects consist of dummy variables that take on the value *I* when the value of a continuous model variable that is involved in the effect is missing. Otherwise, they are assigned the value of 0. For continuous variables, missing values are imputed with the observed mean, and an indicator variable is created to denote missingness. For category variables, missing values are considered a distinct level. Missing levels of group-by variables are assigned to their own level.

Distribution – specifies the response probability distribution.

Link function – specifies the link function to use with the specified distribution. Each distribution has a default link function that is assigned automatically, based on the distribution that is chosen. You can change this in the drop-down list.

Use the drop-down list to select a distribution and then a link function. The choices are as follows:

Distribution	Available Link Functions (default listed first)
Beta	Logit, Probit, Log-log, C-log-log
Binary	Logit, Probit, Log-log, C-log-log
Exponential	Log, Identity
Gamma	Log, Identity, Recip
Geometric	Log, Identity
Inverse Gaussian	Power(-2), Log, Identity
Negative Binomial	Log, Identity
Normal (default)	Identity, Log
Poisson	Log, Identity

- ✍ A GLM model with the distribution set to **Normal** and the link function set to **Identity** is the same as the linear regression model.
- ✍ A GLM model with the distribution set to **Binary** and the link function set to **Logit** results in the same specification as the logistic regression model. However, the target variable measurement for the logistic model is categorical, and the target variable measurement for GLM models is measure.

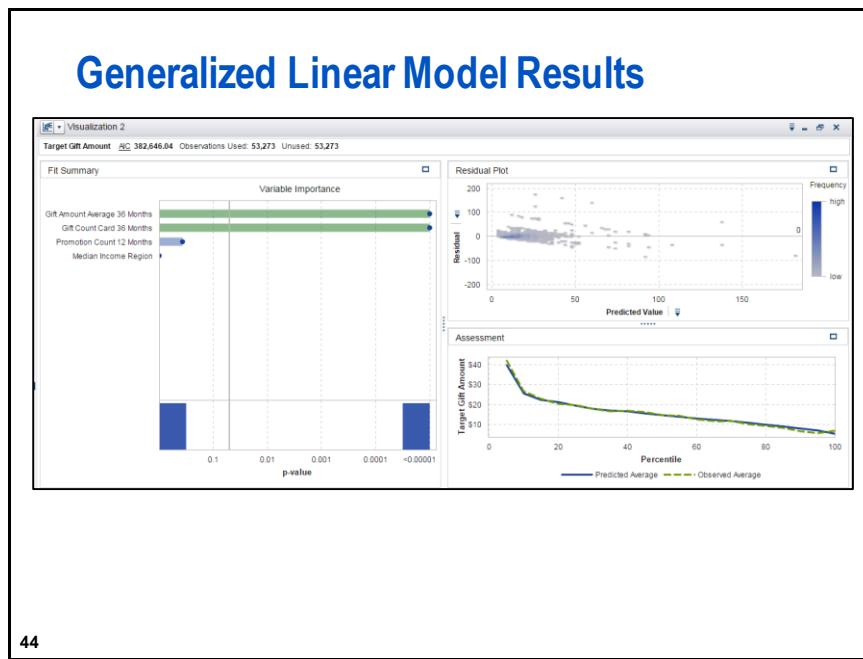
The Convergence properties enable you to enter values that control how quickly the model converges or finds a satisfactory solution.

- **Override function convergence** – Entering a large value reduces the time to train the model but can create a suboptimal model.
- **Override gradient convergence** – Entering a large value reduces the time to train the model but can create a suboptimal model.
- **Maximum iterations** – Entering a small value can reduce the time to train the model but can create a suboptimal model. The default is 50, and the maximum is 100.

- ✍ When you specify a gradient convergence or function convergence criterion, it is possible for the model to converge based on an internal convergence criterion before your criterion is reached. The reason for convergence can be found on the Convergence tab of the summary table.

Assessment

- **Use default bins count** – specifies the number of bins to use in the lift calculations in the assessment plot. It is enabled by default and set at 20. Clear the check box and enter your own number of bins if desired. Increasing the number of bins increases the accuracy of the assessment at the expense of computing time.
- **Tolerance** – specifies the tolerance value that is used to determine the convergence of the iterative algorithm that estimates the percentiles. Specify a smaller value to increase the algorithmic precision.
- **Show diagnostic plots** – displays the residual plot and assessment. It is enabled by default. Clear the check box if you do not want the diagnostic plots to appear.



Although the results panes for the general linear model look essentially the same as for linear regression, there are some differences in the statistics and criteria for a GLM.

Notice that there is no influence plot.

Notice the different model criteria selections that are available on the summary bar.

- -2 Log Likelihood
- AIC
- AICC
- BIC
- Observations

The different selections for the Y axis for the residual plot are as follows:

- Residual
- Standardized Pearson Residual

You can also switch to **Linear Predictor** (versus **Predicted Probability**) on the X axis of the residual plot to display the predicted values rather than the probabilities.

Analyzing Generalized Linear Model Results

Three panes help you analyze the results of the generalized linear model.

- Fit Summary – displays how significant the effect variables are to the response variable.
- Residual Plot – displays the difference between the predicted and the actual data.
- Assessment – displays the values for the observed response along with the model's predicted response.

45

Fit Summary, Residual Plot, and Assessment all work the same way they do for the linear regression model. (Refer to the slides above for explanations about how to use them.)

Generalized Linear Model: Details Table

- Dimensions
- Iteration History
- Convergence
- Fit Statistics
- Type III Test
- Parameter Estimates

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates	
Description						Value
Number of Model Effects						5
Number of Classification Effects						0
Number of Columns in X						5
Rank of Cross-product Matrix						5
Number of Observations Read						106,546
Number of Observations Used						53,273

46

The Dimensions tab provides an overview of the effect variables that are used in the model. This tab identifies the following:

- how many model and classification effects were chosen for the model
- the rank of the cross-product matrix
- how many observations were read
- how many observations were used in the model

Generalized Linear Model: Iteration History

The *Iteration History* tab displays the progress of the estimation results.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Iteration	Evaluations	Objective	Change		Max Gradient
0	5	1.610502	.		9649.065
1	2	1.592	0.018503		1355.967
2	2	1.591427	0.000572		59.3801
3	2	1.591426	1.00E-6		0.140464
4	2	1.591426	3.46E-12		2.94E-7

47

The Iteration History tab shows the progress of the iterative optimization process. This tab also lists the value of the objective function, its change in value, and its maximum gradient.

Generalized Linear Model: Convergence

The *Convergence* tab displays the reason and convergence status of the model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Reason		Status			
Convergence criterion (GCONV=1E-8) satisfied.		0			

48

Models converge when there is no longer a significant improvement between iterations. On the Properties tab of the GLM model, you have the option to override the default function convergence (FCONV) and gradient convergence (GCONV) numbers if you choose. Users should be aware that, even though the user specifies convergence criteria on the Properties tab, it is possible that some of the internal controls are met first (such as XCONV, the relative parameter convergence criterion). (See the slide above.)

In some instances, the model might not reach convergence. When a model does not converge, this indicates that the coefficients are not meaningful because the iterative process was unable to find appropriate solutions. A failure to converge might occur for a number of reasons, such as having a large proportion of predictors to cases, multicollinearity, sparseness, or complete separation.

Generalized Linear Model: Fit Statistics

The *Fit Statistics* tab displays statistics about the estimated model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Statistic					Value
-2 Log Likelihood					339120.2
AIC					339132.2
AICC					339132.2
BIC					339185.5

49

Generalized Linear Model: Type III Test

The *Type III Test* tab displays the significance of each partial effect considering all other effects in the model.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Effect	DF	Chi-Square			Pr > ChiSq
Gift Amount Average 36 Months	1	29126.55			<0.0001
Gift Count Card 36 Months	1	2995.133			<0.0001
Median Income Region	1	34.9036			<0.0001
Promotion Count 12 Months	1	28.04092			<0.0001

50

The Type III Test tab examines the significance of each partial effect with all other effects in the model. For more information, see the “The Four Types of Estimable Functions” chapter in *SAS/STAT® 13.2 User’s Guide*.

Generalized Linear Model: Parameter Estimates

The *Parameter Estimates* tab displays the parameter estimates (coefficients) of each model effect and their associated statistics.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Parameter	Estimate	Standard Error	z Value		Pr > z
Intercept	2.080903	0.007196	289.177		<0.0001
Gift Amount Ave...	0.046959	0.000275	170.665		<0.0001
Gift Count Card...	-0.06756	0.001234	-54.7278		<0.0001
Median Income...	3.92E-7	6.63E-8	5.907927		<0.0001
Promotion Cou...	0.00197	0.000372	5.295367		<0.0001
Scale Parameter	5.052732	0.029993	.		.

51

A partial display is shown above. The Parameter column width is expanded.



Building a GLM Model

This demonstration illustrates using the **VS_BANK** data to build, explore, and refine a GLM model in SAS Visual Statistics. The dependent variable for the analysis is **tgt Count Number New Products**. This variable consists of the number of products with which each account enters the current campaign season. The range is from 0 to 8 and is measured on a count or integer scale.

Similar to the interval-valued target considered in the previous demonstration, the distribution of the count target is not a good fit for the assumption of normally distributed errors embedded in the linear model. It is restricted to be nonnegative. It also has right skew. Not many customers come into the current campaign season with more than three or four products.

The demonstration begins under the linear model assumptions. The effects of violating the linear model's assumptions are explored. The modeling framework is generalized to find a distribution and support that are compatible with the measurement of the dependent variable and to enhance the usefulness of the model.

Fitting a Linear Model to the Count Target

1. Open the **VA_BankInterval** project that was saved in the previous demonstration.
2. Create a new visualization. Click the **Generalized linear model** icon on the toolbar.
3. Deselect the **Auto-Update** button.
4. Assign **tgt Count Number New Products** to the Response role. Assign the continuous effects, the categorical effects, and the interaction as shown below.

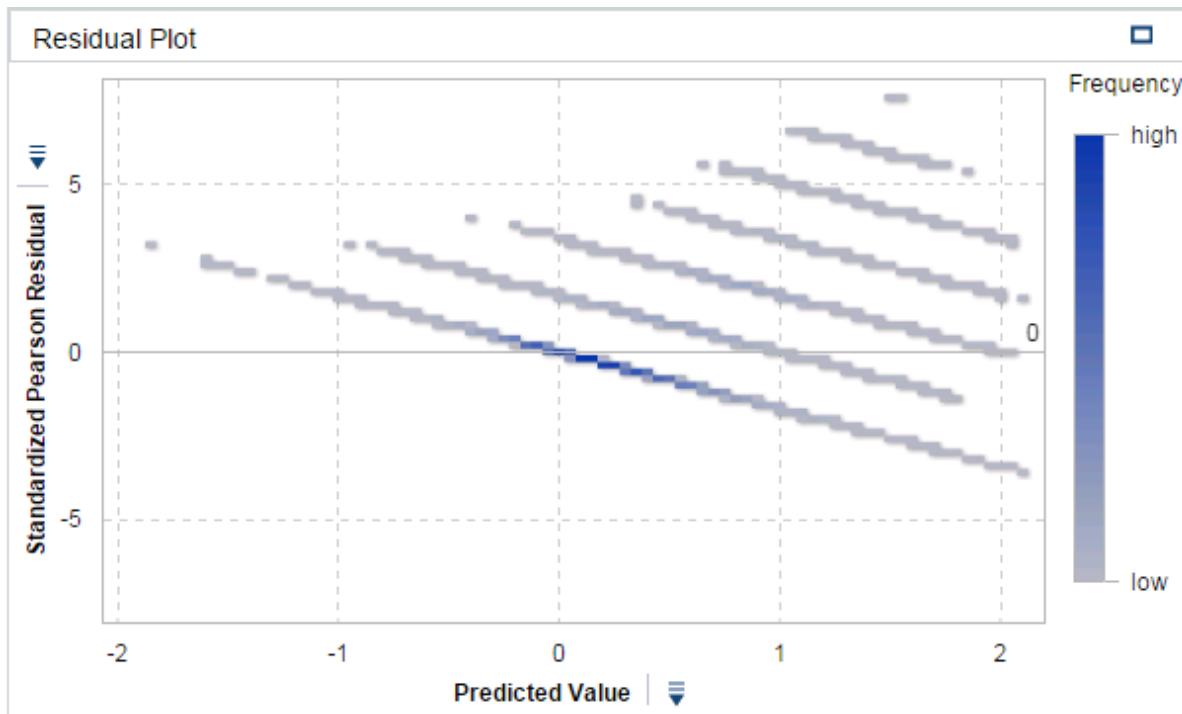
The screenshot shows the configuration interface for a Generalized Linear Model. The 'Response' role is set to 'tgt Count Number New Products'. The 'Continuous Effects' section contains a large list of RFM-related variables. The 'Classification Effects' section includes two categorical variables. The 'Interaction Effects' section displays a selected interaction term.

5. Click the **Properties** tab.

The default distribution is **Normal**, and the default link function is **Identity**.

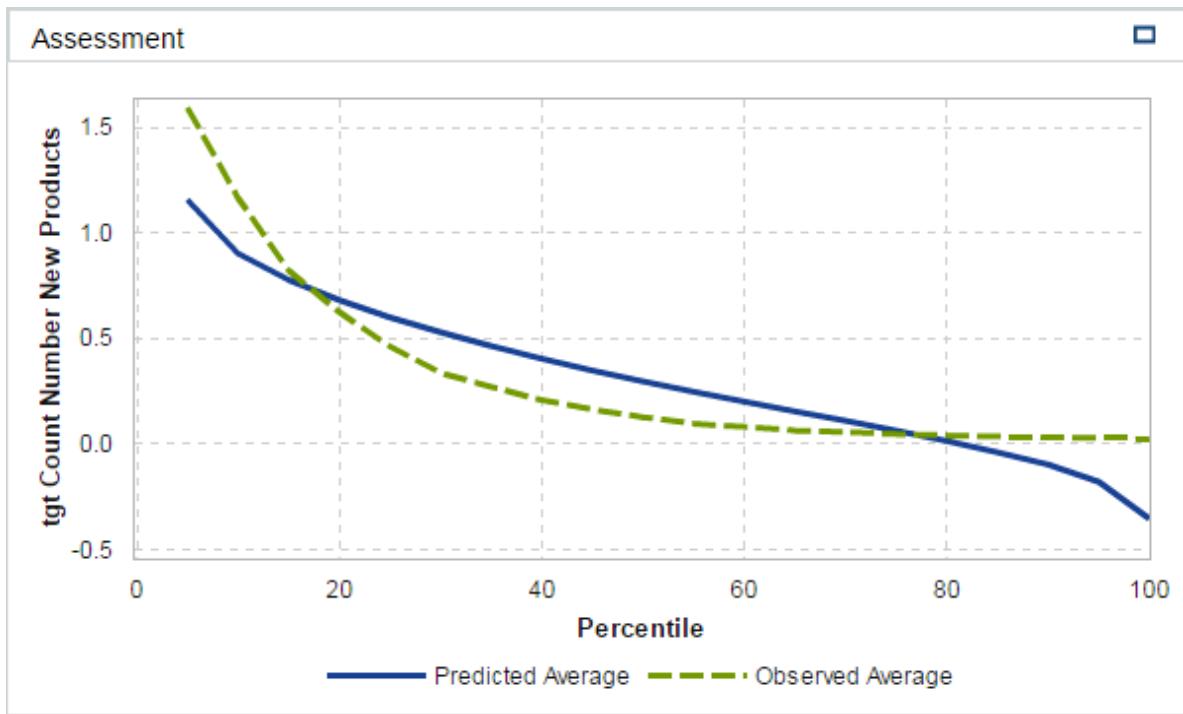
An ordinary linear model is fit under the default settings of the generalized linear model.

6. Click **Update** to run the model.
7. Maximize the residual plot, and change the measure on the vertical axis to **Standardized Pearson Residual**.



The linear modeling assumption of constant error variance seems to be violated. The variance of the residuals is negatively correlated with the magnitude of the prediction for the count-valued target. Another problematic aspect of this plot is that there seems to be a pattern in the residuals. This might be an indication that the model is not picking up all of the available signals in the data. That is, some of the systematic variation in the data is not incorporated in the “explained” variation of the fitted model, and is in what is supposed to be the random or unexplainable variation: the residuals.

8. Restore the residuals plot and maximize the assessment plot.



Prediction bias in the tails of the count target variable distribution is evident in the assessment plot.

9. Restore the assessment plot, and select **Show Details**.

All input variables included in the model seem to be relevant for predicting the number of products purchased in the linear model framework.

Generalizing the Count Target Model

Two candidate distributions that are consistent with the measurement scale of the count target are Poisson and Negative Binomial. Poisson is the simpler distribution in that its mean and variance are restricted to be equal.

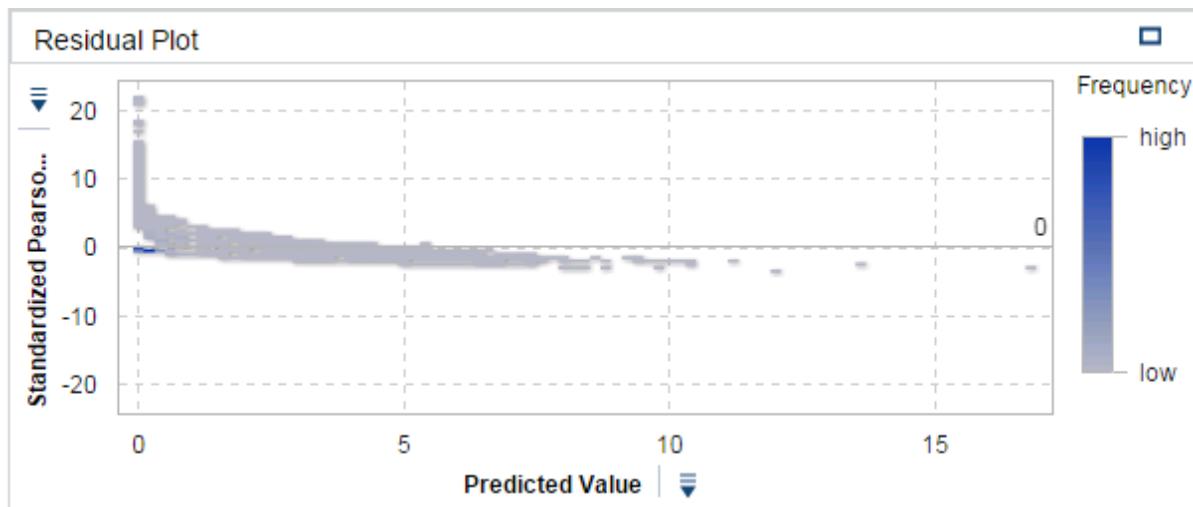
Poisson Model

1. Click the **Properties** tab. Change the distribution to **Poisson**. The link function is **Log**.

Following convention, a log-linear (in parameters) model is specified above. For more details about Poisson and other regression models considered in this demonstration, see Greene (1997)².

2. Click **Update** to run the model.
3. Investigate the standardized Pearson residuals plot.

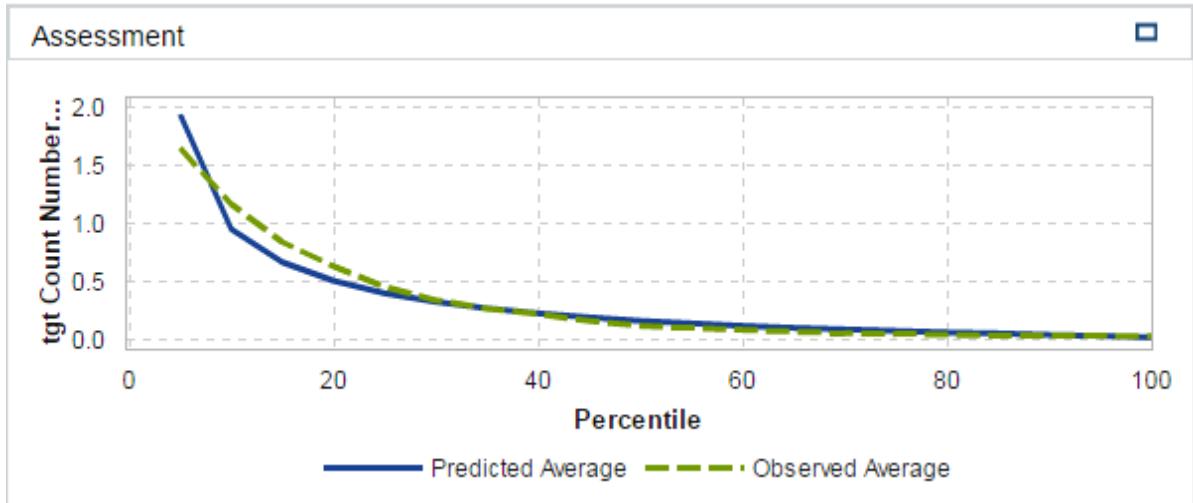
² Greene, William H. 1997. *Econometric Analysis*, 3rd edition, London: Prentice Hall.



Aside from an anomaly around zero (see the note below), the variance in the residuals is closer to satisfying the homoscedasticity assumption than residuals produced in the linear model. The pattern noted in the residuals of the linear model largely disappeared.

- Non-responders are coded with a zero for the count target. Because the response rate is approximately 20%, the count target mainly consists of zeros. Zero Inflated Poisson (ZIP) models, two-stage modeling approaches, and Tweedie distribution models are all successfully used to model count variables with this characteristic.

4. Restore the residual plot and maximize the assessment plot.



The prediction bias produced under the linear model assumptions is largely corrected using a distribution more consistent with the measurement scale of the target.

5. Select **Show details** and click the **Fit Statistics** tab.

Dimensions	Iteration History	Convergence	Fit Statistics	Type III Test	Parameter Estimates
Statistic					Value
-2 Log Likelihood					1157529
AIC					1157569
AICC					1157569
BIC					1157807

6. Save the **VA_BankInterval** project.

End of Demonstration



Exercises

2. Building a GLM Model to Predict the Amount of Donations

In this exercise, you use the **PVA97NK** data to build a GLM model. The target variable is transformed to be more consistent with the restrictions that regulate the analysis.

- a. Open **VS Interval Target Exercise**, which was saved earlier in this chapter. Examine the results of the linear model.
 - 1) Does there seem to be a problem with model bias?
 - 2) Do the residuals seem to satisfy the assumption of constant variance?
- b. Build an appropriate GLM model for the continuous target under the assumption that it follows a log-normal distribution. Use the same variable roles that you used for the linear model, excluding the interaction term. Select the **Informative missingness** property.
- c. Does the GLM model solve the problems that you found in the linear model, if any?
- d. Save the changes to **VS Interval Target Exercise**.

End of Exercises

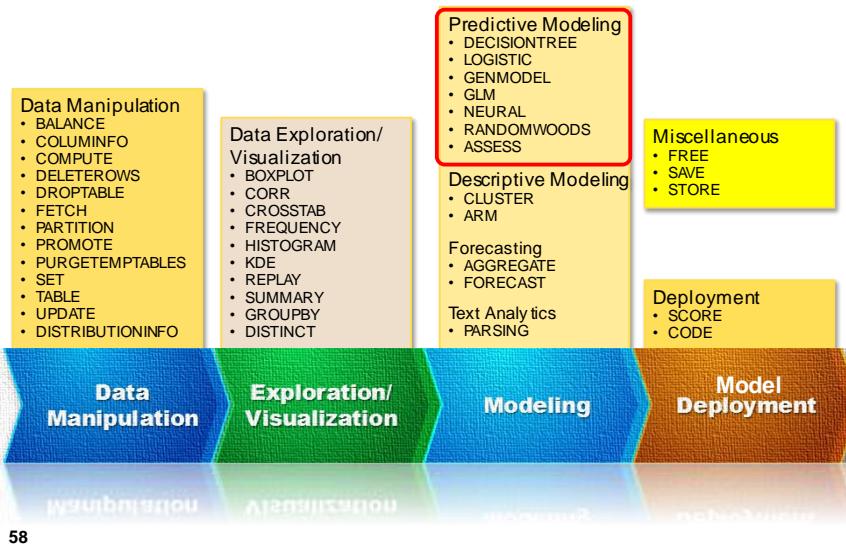
4.2 Interval Targets (SAS In-Memory Statistics)

Objectives

- Describe the syntax for modeling an interval target using the GENMODEL and GLM statements.
- Illustrate different approaches for scoring input and holdout data for model assessment.
- Calculate assessment summary statistics and generate assessment plots.

57

PROC IMSTAT Statements



PROC IMSTAT supports a variety of modeling techniques for predicting interval targets. The GENMODEL and GLM statements are discussed in this section. PROC IMSTAT also supports neural network modeling with the NEURAL statement. (Neural networks are not addressed in this course.)

The generalized linear model was introduced previously.

Generalized Linear Models (GzLM)

- *Generalization* is from linear models.
 - Allow data from exponential family of distributions.
$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)\right\}$$
 - discrete response: Bernoulli, binomial, Poisson, negative binomial, geometric
 - continuous response: normal, gamma, inverse Gaussian, beta
- *Linearity* is achieved through the link function.
 - A transformation of the mean (link function) is linearly related to the independent variables.
 - Link functions are monotonic.

59

PROC IMSTAT implements generalized linear models through the GENMODEL statement.

GENMODEL Statement and Selected Options

```
GENMODEL dependent-variable  
(class-variables)= model-effects / option(s);
```

Options

- DIST=*distribution* – 12 choices, including gamma, normal, and Poisson
- LINK=*function* – nine choices, including identity, logit, and log
- CODE=(*filename*=“*full path*”) – saves score code to score the holdout data for model assessment or deployment

60

 You should choose appropriate distribution and link combinations.

The GENMODEL action has the following options for discrete distributions: BINARY | BERNOUILLI, GENPOISSON, GEOMETRIC | GEOM, NEGBINOMIAL | NEGBIN | NB, and POISSON | PO.

The GENMODEL action has the following options for continuous distributions: BETA, EXPONENTIAL | EXPO, GAMMA | GAM, INVGAUSS | IGAUSSIAN | IG, NORMAL | GAUSSIAN | GAUSS, T | STUDENT, and WEIBULL.

The options for link functions are IDENTITY, LOGIT, PROBIT, LOG, LOGLOG, CLOGLOG, RECIP, POWMINUS2, and POWER(v) | POW(v) | POM(v).



Generalized Linear Models for Count Data

This demonstration illustrates how to fit generalized linear models to the **cnt_tgt** target. Open the program **mldmbd04d01_GeneralizedLinearModel_Counts.sas**.



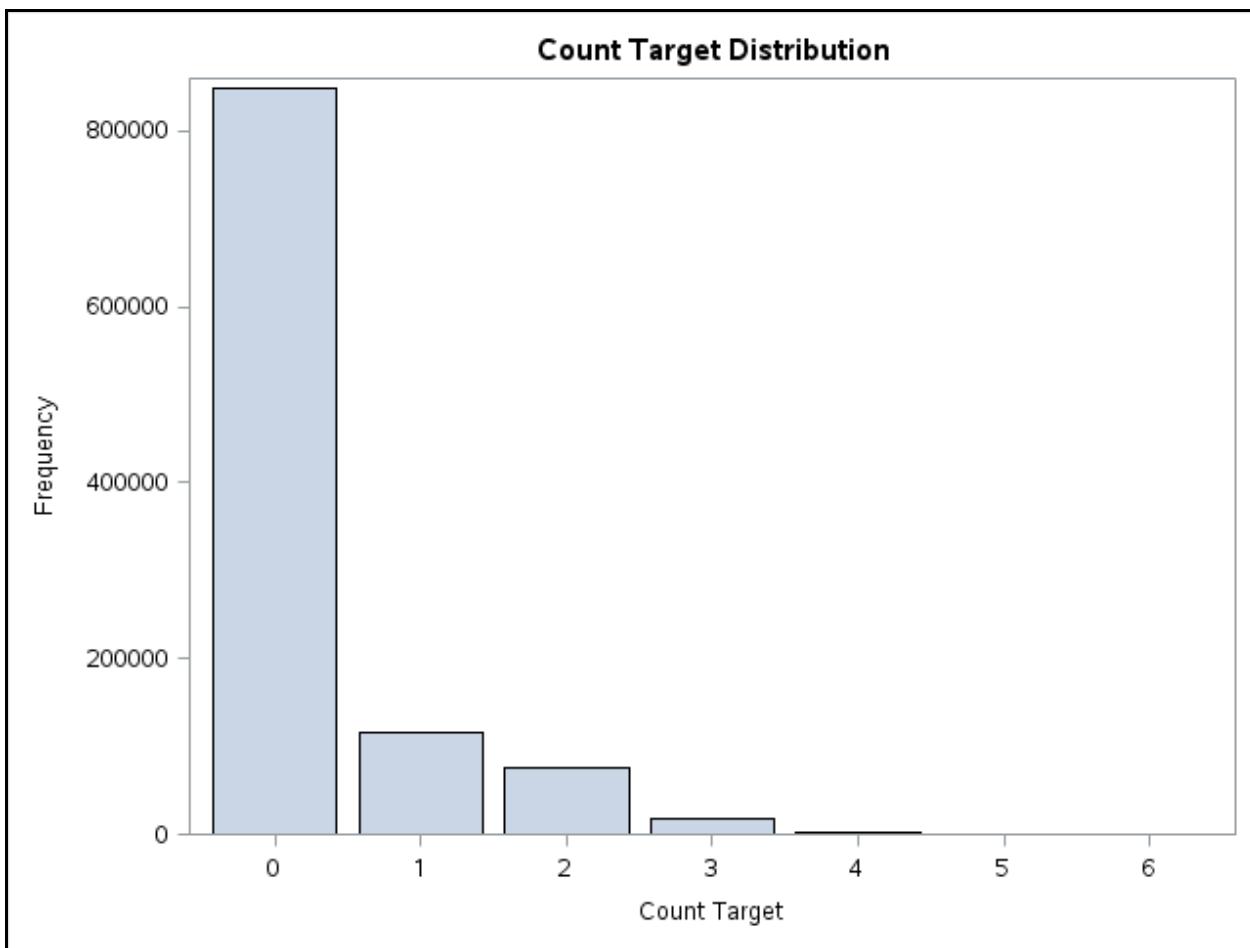
Change to interactive mode. Run the code that defines the system options, library name assignments, and macro variable definitions.

The first section of this demonstration is to review the distribution of the count target. Based on the plots, candidate models are fit and assessed.

The program below uses the HISTOGRAM statement in PROC IMSTAT and the SGPLOT procedure (a Base SAS procedure) to produce a plot of the **cnt_tgt** target variable. Refer to the first demonstration in this chapter for details about generating histogram data and plots.

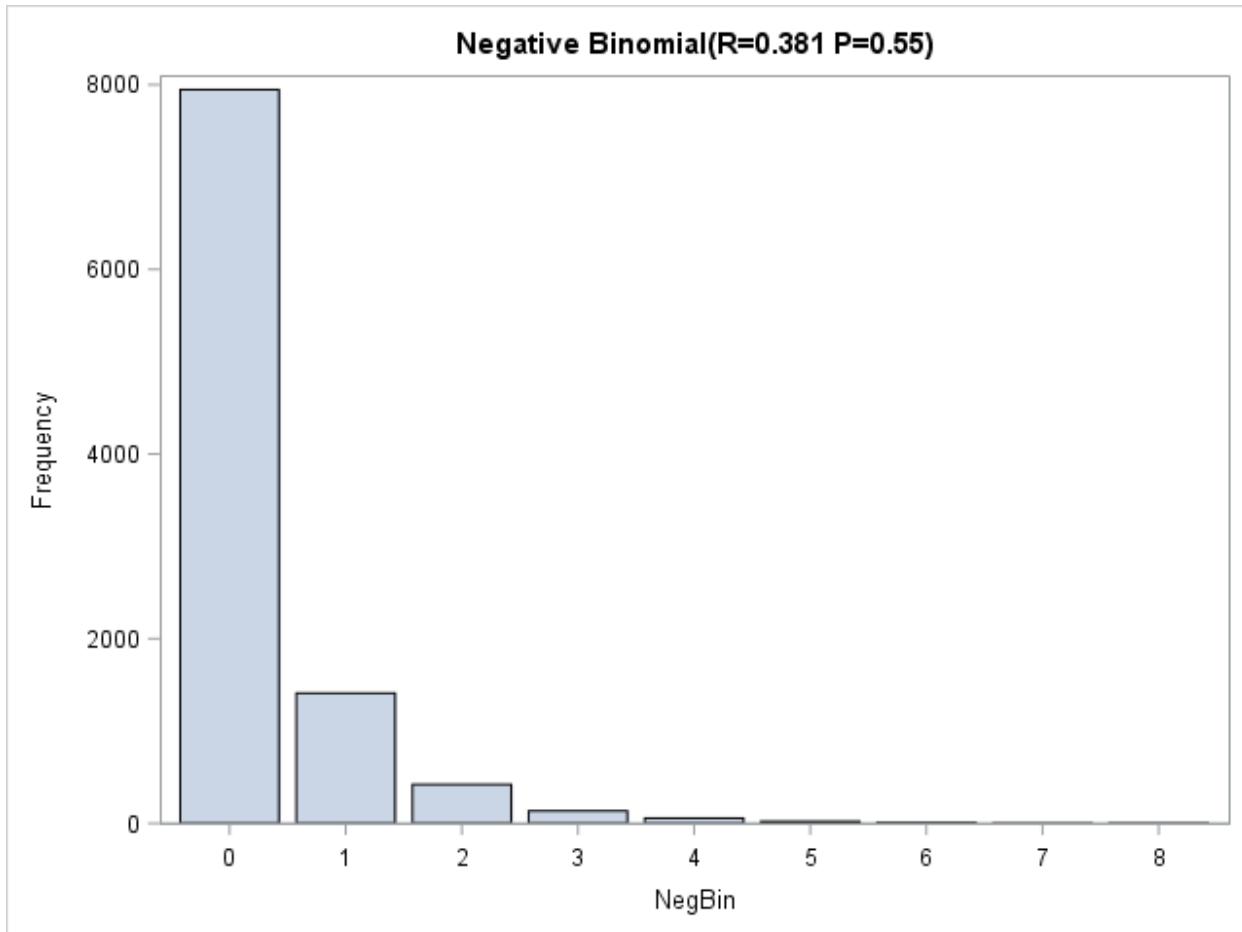
```
proc imstat;
  tableinfo / port=&SessionPort;
run;
  table LASRlib.p_model_bank13(tag="&TagString");
run;
ods output histogram = work.cnt_histogramdata;
histogram cnt_tgt/nbins=8;
run;
quit;
title1 "Count Target Distribution";
proc sgplot data=work.cnt_histogramdata;
  vbar binmid / freq=frequency;
  label binmid="Count Target";
run;
title1;
```

The plot of the **cnt_tgt** target is shown below.



The **cnt_tgt** target has only seven levels, integers from 0 to 6 inclusive. Targets with this distribution are often modeled using Poisson regression. Recall that a Poisson distribution is used to model the number of arrivals in a given time interval where the mean and variance are the same. If you expect greater variance than the mean, then a negative binomial distribution might be appropriate. The negative binomial distribution is usually described in relation to modeling the number of failures before the R-th success occurs. As an error distribution, R becomes a parameter of the distribution along with p , the probability of success. The general formulation for the negative binomial distribution enables R to be any positive number.

The following bar chart was produced by simulating a negative binomial distribution with the indicated parameters:



In the following code, the WHERE clause subsets the active table to the training partition. The nominal and input variables are defined using syntax identical to that used for the LOGISTIC statement. The DISTRIBUTION=POISSON and LINK=LOG options instruct the GENMODEL statement to fit a Poisson regression model. The SCORE statement scores all of the input data. The CODE statement generates code that is saved in the defined path.

- ✍ The GENMODEL statement option DISTRIBUTION= supports 12 distributions, five discrete distributions, and seven continuous distributions. (The keywords for these distributions were given prior to the demonstration.)

Because counts are discrete, one of the five discrete distributions is appropriate.

```

proc imstat;
  table LASRlib.p model_bank13(tag="&TagString");
  where _partind_=1;
  genmodel cnt tgt(&catvars) = &i_rfms &ir_demogs &catvars /
    ID=(account)
    dist=poisson
    link=log
    score(_all_)
  code=(filename=&ScoreFolder/cnt_tgt_Poisson_ScoreCode.sas"
        replace);
run;

```

Like the LOGISTIC statement, the GENMODEL statement supports the SHOWSELECTED keyword, which requests backward elimination variable selection.

The first sections of the results list the model information, class level information, and iteration history.

The IMSTAT Procedure				
Model Information				
Data Source	USER.STUDENT2.HPADATA.P_MODEL_BANK13			
Response Variable	cnt_tgt			
Distribution	Poisson			
Link Function	Log			
Class Level Information for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Class		Levels	Values	
cat_input1		3	X Y Z	
cat_input2		5	A B C D E	
Iteration History for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Iteration	Function Calls	Objective Function	Change	Maximum Gradient
0	5	0.6193459838	.	17707.77
1	2	0.5631918305	0.05615403	4062.615
2	2	0.5540924834	0.00909935	942.0769
3	2	0.5535081484	0.00058434	78.25225
4	2	0.5534811155	0.00002703	4.118889
5	2	0.5534810883	0.00000003	0.004346
6	2	0.5534810883	0.00000000	4.249E-9
Convergence criterion (GCONV=1E-8) satisfied.				

The dimensions and fit statistics are shown next.

Dimensions	
Number of Model Effects	22
Number of Classification Effects	2
Number of Columns in X	28
Rank of Cross-product Matrix	25
Number of Observations Read	531423
Number of Observations Used	531423

Fit Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13	
-2 Log Likelihood	588265
AIC	588315
AICC	588315
BIC	588595

The parameter estimates are displayed next.

Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	-0.4064	0.026885	-15.14	<.0001
i_rfml	-0.00711	0.001221	-5.82	<.0001
i_rfml2	-0.05603	0.001323	-42.34	<.0001
i_rfml3	-0.02364	0.001028	-23.00	<.0001
i_rfml4	-0.00207	0.000631	-3.28	0.0010
i_rfml5	0.2872	0.002143	134.04	<.0001
i_rfml6	-0.01422	0.000659	-21.60	<.0001
i_rfml7	-0.03815	0.002936	-12.99	<.0001
i_rfml8	0.01769	0.001469	12.05	<.0001
i_rfml9	-0.09316	0.000690	-135.04	<.0001
i_rfml10	-0.00415	0.000716	-5.79	<.0001
i_rfml11	-0.02158	0.002293	-9.41	<.0001
i_rfml12	0.001596	0.000130	12.28	<.0001
i_demog_age	-0.00029	0.000167	-1.72	0.0882
demog_ho	0.002959	0.005180	0.57	0.5678
ri_demog_homeval	2.952E-6	0	Infty	<.0001
ri_demog_inc	3.451E-7	1.511E-7	2.28	0.0224
demog_pv	0.000803	0.000211	3.80	0.0001
demog_genf	0.02483	0.005013	4.95	<.0001
demog_genm	0	.	.	.
cat_input1 X	0.4942	0.01472	33.58	<.0001
cat_input1 Y	0.4489	0.01719	26.12	<.0001
cat_input1 Z	0	.	.	.
cat_input2 A	0.1722	0.008307	20.74	<.0001
cat_input2 B	0.1847	0.007519	24.56	<.0001
cat_input2 C	0.1129	0.007702	14.86	<.0001
cat_input2 D	0.07525	0.008876	8.48	<.0001
cat_input2 E	0	.	.	.

The Type III test is shown next.

Type III Tests of Model Effects for USER.STUDENT2.HPADATA.P_MODEL_BANK13			
Effect	Num DF	Chi-Square	Pr > ChiSq
i_rfm1	1	33.88	<.0001
i_rfm2	1	1792.58	<.0001
i_rfm3	1	528.92	<.0001
i_rfm4	1	10.75	0.0010
i_rfm5	1	17966.4	<.0001
i_rfm6	1	466.40	<.0001
i_rfm7	1	168.81	<.0001
i_rfm8	1	145.10	<.0001
i_rfm9	1	18235.0	<.0001
i_rfm10	1	33.58	<.0001
i_rfm11	1	88.57	<.0001
i_rfm12	1	150.91	<.0001
i_demog_age	1	2.94	0.0862
demog_ho	1	0.33	0.5678
ri_demog_homeval	0	Infty	<.0001
ri_demog_inc	1	5.22	0.0224
demog_pr	1	14.47	0.0001
demog_genf	0
demog_genm	0
cat_input1	2	1131.08	<.0001
cat_input2	4	715.75	<.0001

Some of the inputs are not statistically significant. This implies that a more parsimonious model might be appropriate. You could eliminate non-significant terms or use a decision tree or a forest to screen inputs (as shown in a previous demonstration) to develop candidate models with fewer inputs.

The fit of the model can be assessed on the training data using the code below.

```
table LASRlib.&_templat_<br/>
columninfo;<br/>
run;<br/>
fetch account cnt_tgt _pred_ _ilink_ _resid_ /<br/>
from=1 to=10;<br/>
run;<br/>
title1 "Train Lift on Original Scale";<br/>
ods output liftreginfo=work.TrainPoiRegLift;<br/>
assess _ilink_ / y=cnt_tgt;<br/>
run;<br/>
quit;
```

The scored input data table was made active using the `&_templast_` macro variable name. The columns of the scored table are listed with the COLUMNINFO statement. Only the columns added by the scoring process are listed below.

24	_PRED_	Num	8	BEST12.	
25	_RESID_	Num	8	BEST12.	
26	_ILINK_	Num	8	BEST12.	
27	_PEARSON_	Num	8	BEST12.	

Selected variables of the first 10 rows of the scored table are listed below.

Selected Records from Table _T_A8D50C63_7FDDC84ACFF8					
account	cnt_tgt	_PRED_	_ILINK_	_RESID_	
100588679	0	-1.007847	0.385004	-0.385004	
100588687	1.000000	-1.560034	0.212241	0.787759	
100588699	0	-2.397997	0.090900	-0.090900	
100588703	0	-1.186692	0.311395	-0.311395	
100588715	0	-3.691963	0.024923	-0.024923	
100588719	0	-0.696595	0.498279	-0.498279	
100588723	1.000000	-1.612761	0.199337	0.800663	
100588727	2.000000	-0.914857	0.400574	1.599426	
100588731	0	-2.084072	0.124422	-0.124422	
100588735	0	-1.662137	0.189733	-0.189733	

The `_PRED_` variable is the predicted value on the link (log) scale. The `_ILINK_` variable is the predicted value on the original count scale. In the ASSESS statement, the `_ILINK_` value is used to assess how well the model predictions agree with the target. The assessment statistics are saved to the SAS server using the ODS OUTPUT statement.

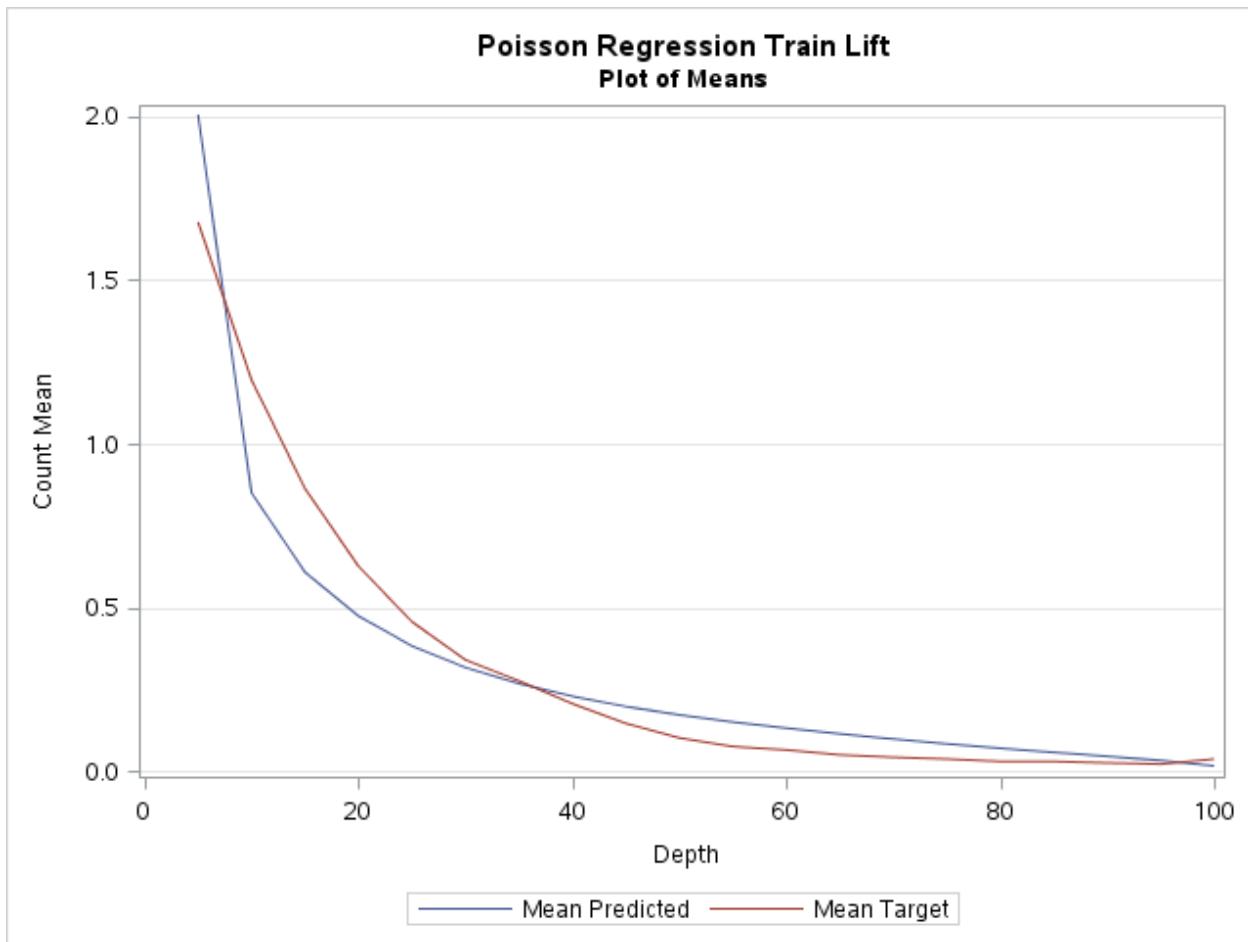
The assessment statistics are listed below.

Lift Information for Table _T_A8D50C63_7FDDC84ACFF8									
Column	Depth	Value	Number of Observations	Mean Predicted	Min Predicted	Max Predicted	Mean Response	Min Response	Max Response
LINK	5.0000	1.0608	26572	2.0056	1.0608	29.6005	1.6769	0	6.0000
LINK	10.0000	0.6989	26572	0.8496	0.6989	1.0608	1.1942	0	4.0000
LINK	15.0000	0.5311	26572	0.6076	0.5311	0.6989	0.8629	0	4.0000
LINK	20.0000	0.4226	26572	0.4734	0.4226	0.5311	0.6259	0	3.0000
LINK	25.0000	0.3458	26572	0.3820	0.3458	0.4226	0.4561	0	3.0000
LINK	30.0000	0.2899	26572	0.3163	0.2899	0.3458	0.3395	0	4.0000
LINK	35.0000	0.2487	26572	0.2874	0.2487	0.2899	0.2756	0	4.0000
LINK	40.0000	0.2122	26572	0.2286	0.2122	0.2487	0.2059	0	2.0000
LINK	45.0000	0.1834	26572	0.1973	0.1834	0.2122	0.1456	0	3.0000
LINK	50.0000	0.1597	26572	0.1712	0.1597	0.1834	0.1013	0	3.0000
LINK	55.0000	0.1397	26572	0.1495	0.1397	0.1697	0.07519	0	3.0000
LINK	60.0000	0.1220	26572	0.1308	0.1220	0.1397	0.06458	0	3.0000
LINK	65.0000	0.1056	26573	0.1137	0.1056	0.1220	0.04956	0	2.0000
LINK	70.0000	0.09041	26571	0.09793	0.09041	0.1056	0.04226	0	2.0000
LINK	75.0000	0.07654	26572	0.08338	0.07654	0.09041	0.03688	0	2.0000
LINK	80.0000	0.06331	26572	0.06984	0.06331	0.07654	0.02928	0	2.0000
LINK	85.0000	0.05098	26572	0.05707	0.05098	0.06331	0.02913	0	2.0000
LINK	90.0000	0.03895	26572	0.04495	0.03895	0.05098	0.02533	0	3.0000
LINK	95.0000	0.02594	26572	0.03264	0.02594	0.03895	0.02145	0	2.0000
LINK	100.00	2.27E-59	26555	0.01571	2.27E-59	0.02594	0.03702	0	2.0000

The means plot (similar to a lift plot) is produced by the code below.

```
title1 "Poisson Regression Train Lift";
title2 "Plot of Means";
proc sgplot data=work.trainpoireglift;
  series y=meanp x=depth / legendlabel="Mean Predicted" name="line1";
  series y=meant x=depth / legendlabel="Mean Target" name="line2";
  yaxis label="Count Mean" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
run;
title2;
title1;
```

The plot is shown.



The program below fits another candidate model to the count target. The only difference is that the generalized Poisson distribution is used.

```
proc imstat;
  table LASRlib.p_model_bank13(tag=&TagString");
  where partind =1;
  genmodel cnt_tgt(&catvars)= &i_rfms &ir_demogs &catvars /
    ID=(account)
    dist=genpoisson
    link=log
    score(_all_)
  code=(filename=&ScoreFolder/cnt_tgt_GenPoisson_ScoreCode.sas"
        replace);
run;
```

Only the fit statistics and parameter estimates are shown.

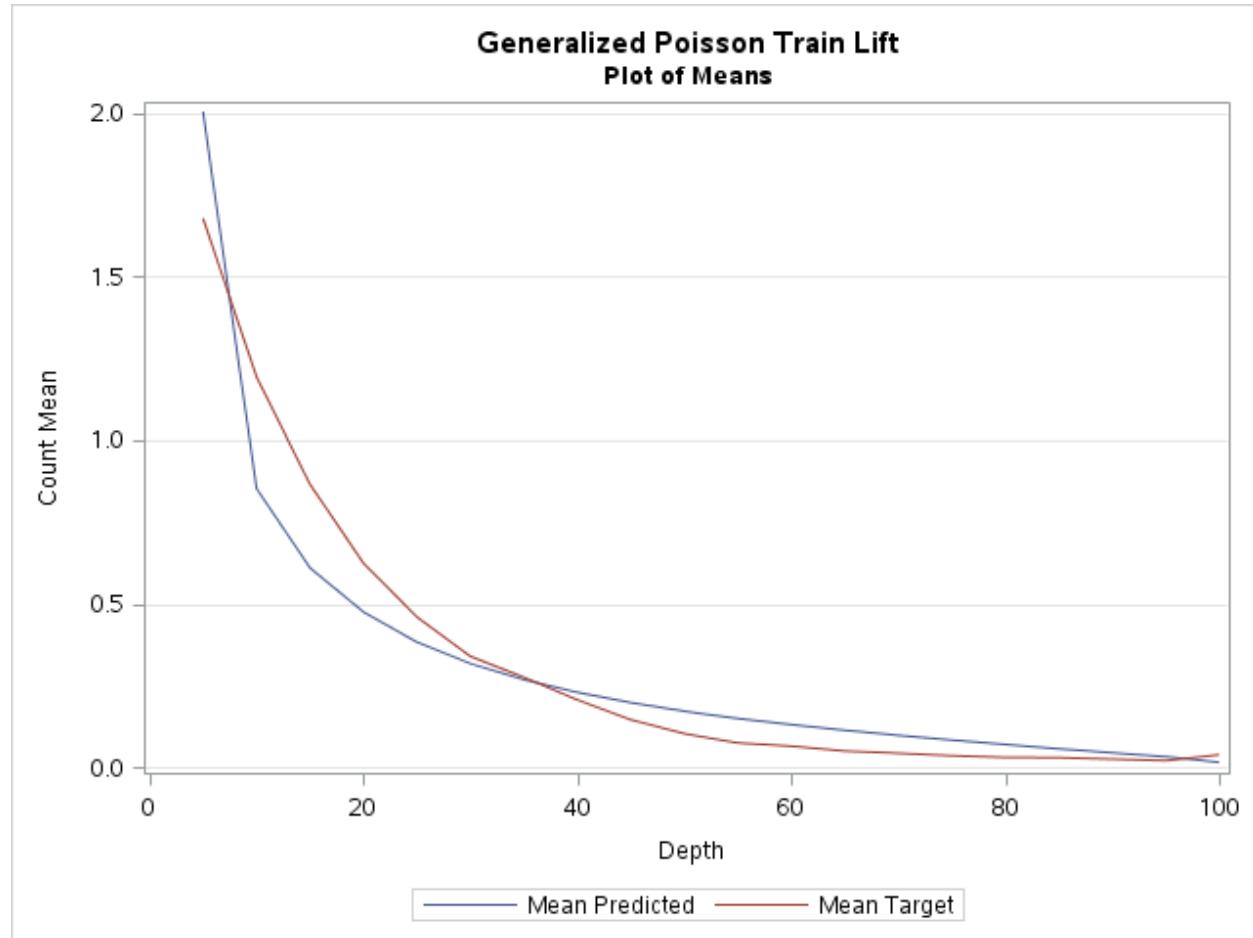
Fit Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13	
-2 Log Likelihood	587698
AIC	587750
AICC	587750
BIC	588040

Based on BIC, this model is better than the previous candidate.

Parameter Estimates for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	-0.3627	0.02755	-13.16	<.0001
i_rfml	-0.00577	0.001253	-4.60	<.0001
i_rfml2	-0.05747	0.001363	-42.16	<.0001
i_rfml3	-0.02506	0.001053	-23.80	<.0001
i_rfml4	-0.00271	0.000668	-4.05	<.0001
i_rfml5	0.2840	0.002213	128.33	<.0001
i_rfml6	-0.01435	0.000679	-21.13	<.0001
i_rfml7	-0.03892	0.003026	-12.88	<.0001
i_rfml8	0.01792	0.001511	11.86	<.0001
i_rfml9	-0.09392	0.000708	-132.73	<.0001
i_rfml10	-0.00457	0.000736	-6.21	<.0001
i_rfml11	-0.01970	0.002356	-8.38	<.0001
i_rfml12	0.001625	0.000133	12.18	<.0001
i_demog_age	-0.00028	0.000171	-1.64	0.1007
demog_ho	0.004132	0.005326	0.78	0.4379
ri_demog_homeval	2.893E-6	0	Infty	<.0001
ri_demog_inc	3.574E-7	1.542E-7	2.32	0.0204
demog_pr	0.000818	0.000217	3.78	0.0002
demog_genf	0.02464	0.005149	4.78	<.0001
demog_genm	0	-	-	-
cat_input1 X	0.4837	0.01503	32.19	<.0001
cat_input1 Y	0.4409	0.01756	25.10	<.0001
cat_input1 Z	0	-	-	-
cat_input2 A	0.1788	0.008529	20.96	<.0001
cat_input2 B	0.1906	0.007732	24.66	<.0001
cat_input2 C	0.1173	0.007934	14.78	<.0001
cat_input2 D	0.07782	0.009152	8.50	<.0001
cat_input2 E	0	-	-	-
Scale Parameter	0.02828	0.001391	-	-

The parameters and their p -values are also different from the previous candidate, but not substantially. One additional parameter is estimated compared to the previous model, the scale parameter. This additional parameter enables for the model to account for over- or under-dispersion.

The means plot for the generalized Poisson model is below.



To assess the model performance on the validation data, the score code saved with the SCORE statement can be applied to the validation data. After it is scored, the model can be assessed using the approach above. The code below accomplishes this goal.

```

proc imstat;
  table LASRlib.p_model_bank13(tag="&TagString");
  where _PartInd_=2;
  filename GPscore "&ScoreFolder/cnt_tgt_GenPoisson_ScoreCode.sas";
  score code=GPscore keep=(_ALL_)
    out=LASRlib.ScoredbyGP;
run;
quit;

proc imstat;
  table LASRlib.ScoredbyGP;
  columninfo;
run;

```

```

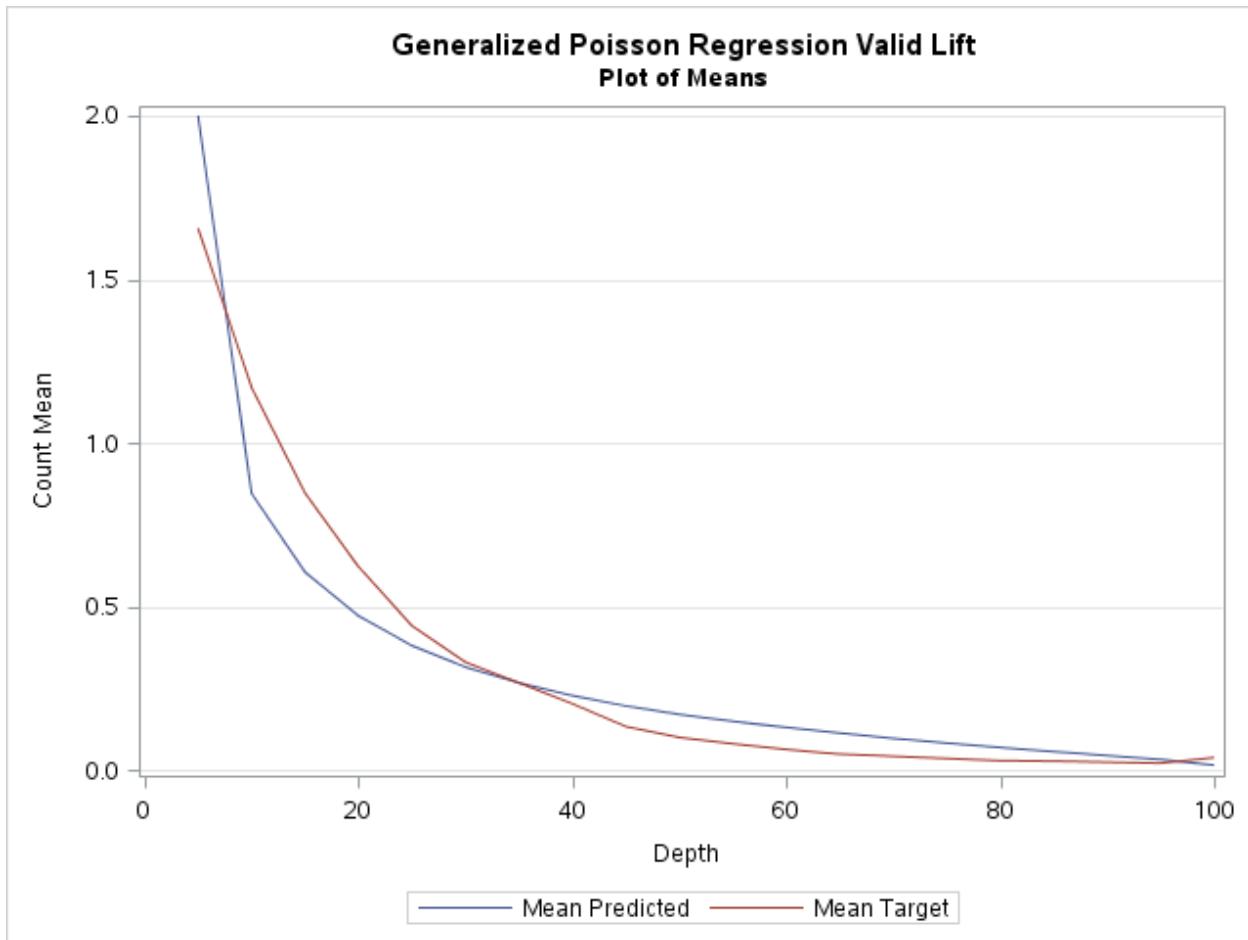
fetch cnt_tgt p_cnt_tgt;
run;
title1 "Valid Lift on Original Scale";
ods output liftreginfo=work.ValidGenPoiRegLift;
assess P_cnt_tgt /y=cnt_tgt;
run;
title;
quit;
title1 "Poisson Regression Valid Lift";
title2 "Plot of Means";
proc sgplot data=work.ValidGenPoiReglift;
series y=meanp x=depth;
series y=meant x=depth;
run;
title2;
title1;

```

The score code labels the predicted value on the original scale by appending the prefix **P_** to the name of the target variable. In this example, **P_cnt_tgt** is the predicted value that you assess. This value corresponds to the **_ILINK_** value that is calculated by the SCORE option.

The means tables and plots are shown below.

Lift Information for Table USER.STUDENT2.HPADATA.SCOREDBYGP									
Column	Depth	Value	Number of Observations	Mean Predicted	Min Predicted	Max Predicted	Mean Response	Min Response	Max Response
P_cnt_tgt	5.0000	1.0528	26430	2.0005	1.0528	26.4168	1.6557	0	6.0000
P_cnt_tgt	10.0000	0.6961	26431	0.8455	0.6961	1.0528	1.1695	0	4.0000
P_cnt_tgt	15.0000	0.5287	26431	0.6051	0.5287	0.6961	0.8468	0	4.0000
P_cnt_tgt	20.0000	0.4211	26431	0.4712	0.4211	0.5287	0.6214	0	3.0000
P_cnt_tgt	25.0000	0.3438	26431	0.3802	0.3438	0.4211	0.4410	0	3.0000
P_cnt_tgt	30.0000	0.2890	26431	0.3147	0.2890	0.3438	0.3297	0	3.0000
P_cnt_tgt	35.0000	0.2460	26431	0.2667	0.2460	0.2890	0.2667	0	4.0000
P_cnt_tgt	40.0000	0.2105	26431	0.2275	0.2105	0.2460	0.2026	0	2.0000
P_cnt_tgt	45.0000	0.1823	26431	0.1959	0.1823	0.2105	0.1329	0	3.0000
P_cnt_tgt	50.0000	0.1591	26432	0.1703	0.1591	0.1823	0.0990	0	3.0000
P_cnt_tgt	55.0000	0.1390	26427	0.1488	0.1390	0.1591	0.08030	0	3.0000
P_cnt_tgt	60.0000	0.1214	26434	0.1302	0.1214	0.1390	0.06284	0	3.0000
P_cnt_tgt	65.0000	0.1049	26431	0.1130	0.1049	0.1214	0.04862	0	3.0000
P_cnt_tgt	70.0000	0.08976	26431	0.09713	0.08977	0.1049	0.04245	0	2.0000
P_cnt_tgt	75.0000	0.07574	26431	0.08268	0.07574	0.08976	0.03511	0	2.0000
P_cnt_tgt	80.0000	0.06256	26431	0.06908	0.06256	0.07574	0.02883	0	2.0000
P_cnt_tgt	85.0000	0.05032	26431	0.05637	0.05032	0.06256	0.02713	0	2.0000
P_cnt_tgt	90.0000	0.03840	26431	0.04434	0.03840	0.05032	0.02437	0	3.0000
P_cnt_tgt	95.0000	0.02547	26431	0.03216	0.02547	0.03840	0.02100	0	3.0000
P_cnt_tgt	100.00	1.01E-60	26426	0.01631	1.01E-60	0.02547	0.03837	0	2.0000



The following code fits a generalized linear model using the negative binomial distribution.

```

proc imstat;
  table LASRlib.p_model_bank13(tag="&TagString") ;
  where partind =1;
  genmodel cnt_tgt(&catvars)= &i_rfms &ir_demo &catvars /
    ID=(account) dist=negbin link=log
    score(_all_)
    code=(filename="&ScoreFolder/cnt_tgt_NegBin_ScoreCode.sas"
    replace);
run;
  table LASRlib.&_templatst_;
  columninfo;
run;
  fetch account cnt_tgt _pred_ _ilink_ _resid_ / from=1 to=10;
run;
  ods output liftreginfo=work.TrainNegBinRegLift;
  assess _ilink_ /y=cnt_tgt;
run;
quit;

```

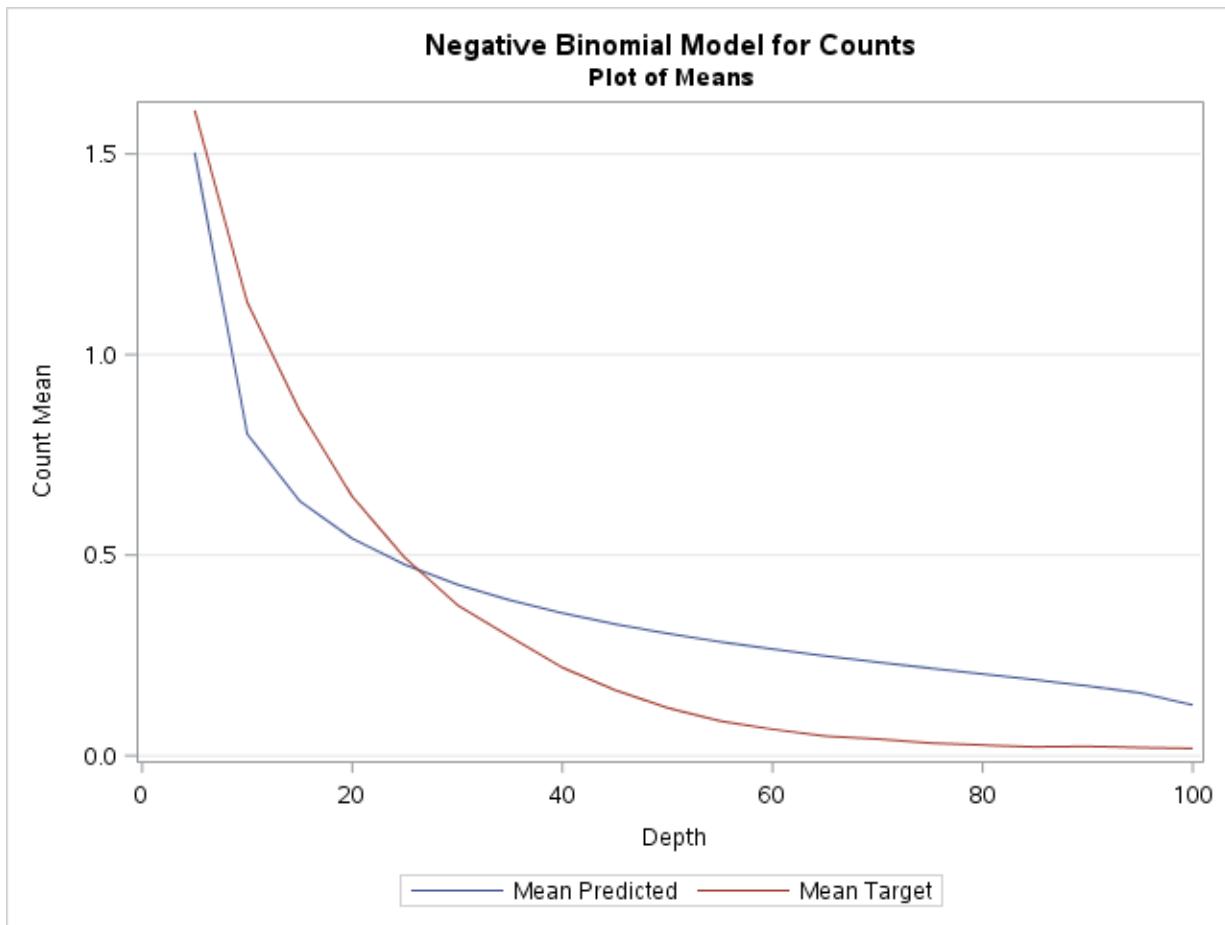
The parameter estimates table is shown here.

Parameter Estimates for USER.STUDENT.P_MODEL_BANK13				
Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	-0.6925	0.02161	-32.05	<.0001
i_rfml	0.001077	0.000404	2.66	0.0077
i_rfml2	-0.01199	0.000355	-33.77	<.0001
i_rfml3	-0.00147	0.000301	-4.89	<.0001
i_rfml4	0.000135	0.000119	1.13	0.2576
i_rfml5	0.2453	0.002359	103.98	<.0001
i_rfml6	-0.00223	0.000745	-2.99	0.0028
i_rfml7	-0.03370	0.003177	-10.61	<.0001
i_rfml8	0.01300	0.001574	8.26	<.0001
i_rfml9	-0.05116	0.000641	-79.84	<.0001
i_rfml10	0.000395	0.000725	0.54	0.5864
i_rfml11	-0.05747	0.002305	-24.93	<.0001
i_rfml12	0.000200	0.000121	1.66	0.0972
i_demog_age	-0.00008	0.000156	-0.48	0.6292
demog_ho	-0.00215	0.004661	-0.46	0.6450
ri_demog_homeval	1.941E-6	2.986E-8	65.02	<.0001
ri_demog_inc	9.886E-8	1.514E-7	0.65	0.5138
demog_pr	0.000104	0.000197	0.53	0.5966
demog_genf	0.01269	0.004491	2.83	0.0047
demog_genm	0	.	.	.
cat_input1 X	-0.00060	0.008377	-0.07	0.9429
cat_input1 Y	-0.06100	0.01109	-5.50	<.0001
cat_input1 Z	0	.	.	.
cat_input2 A	0.05481	0.007798	7.03	<.0001
cat_input2 B	0.06327	0.006878	9.20	<.0001
cat_input2 C	0.04172	0.006830	6.11	<.0001
cat_input2 D	0.01905	0.007617	2.50	0.0124
cat_input2 E	0	.	.	.
Scale Parameter	0.2332	0		

The following code produces a plot of means for the training data:

```
proc sgplot data=work.trainNegBinreglift;
  series y=meanp x=depth / legendlabel="Mean Predicted" name="line1";
  series y=meant x=depth / legendlabel="Mean Target" name="line2";
  yaxis label="Count Mean" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
run;
```

The code produces the following plot:



The model appears to be inferior to the generalized Poisson model, particularly in the upper tail.

To show the mechanics for producing validation plots, consider the following code:

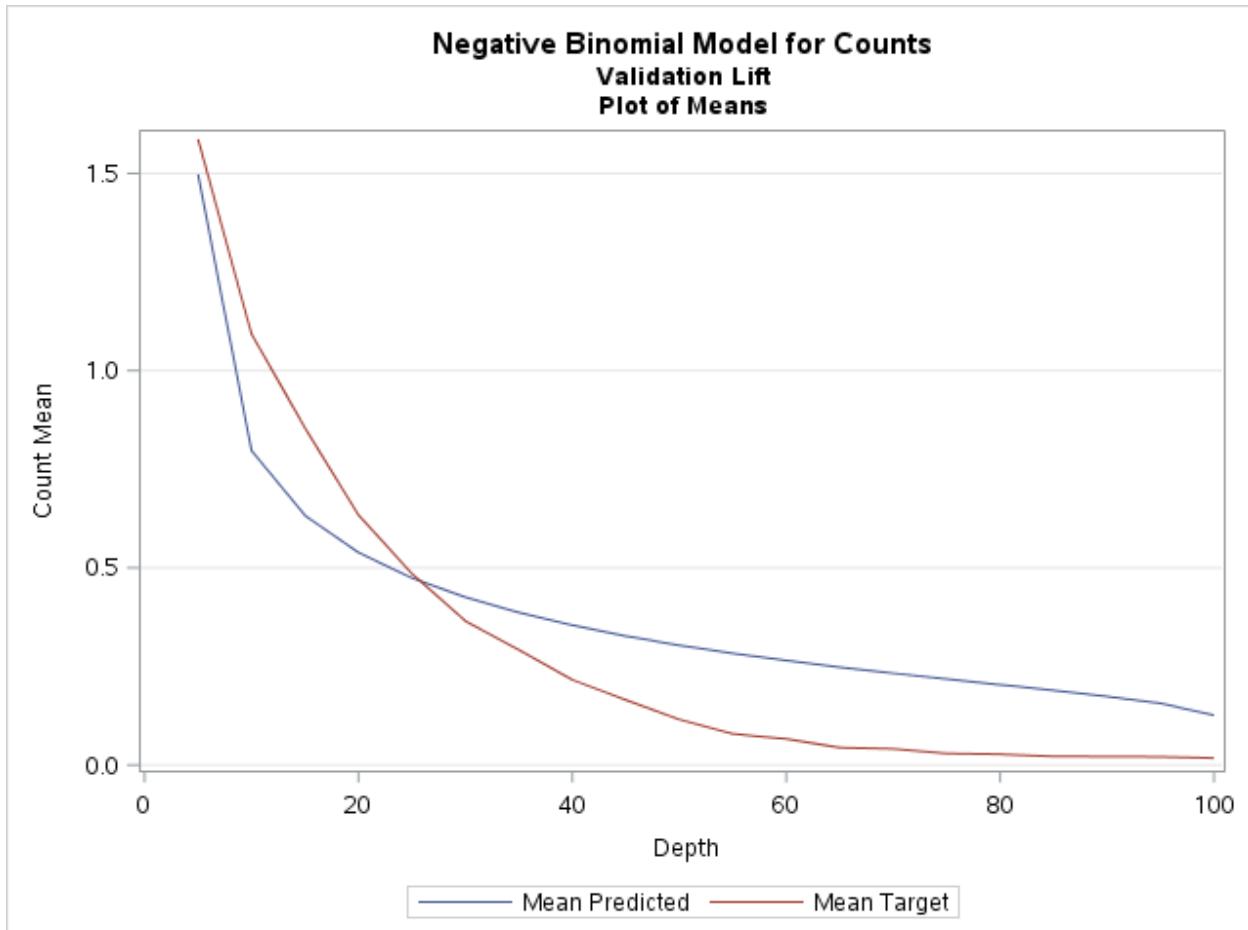
```

proc imstat;
  table LASRlib.p_model_bank13(tag="&TagString") ;
  where PartInd =2;
  filename GPscore "&ScoreFolder/cnt_tgt_NegBin_ScoreCode.sas";
  score code=GPscore keep=( ALL )
    out=LASRlib.ScoredbyNegBin;
run;
quit;
proc imstat;
  table LASRlib.ScoredbyNegBin;
  columninfo;
run;
  fetch cnt_tgt p_cnt_tgt;
run;
  ods output liftreginfo=work.ValidNegBinRegLift;
  assess P_cnt_tgt /y=cnt_tgt;
run;
quit;

```

```
proc sgplot data=work.ValidNegBinReglift;
  series y=meanp x=depth / legendlabel="Mean Predicted" name="line1";
  series y=meant x=depth / legendlabel="Mean Target" name="line2";
  yaxis label="Count Mean" grid;
  keylegend "line1" "line2" / location=outside position=bottom;
run;
```

Here is the assessment plot for the validation data:





Generalized Linear Models for Interval Targets

This demonstration illustrates how to fit generalized linear models to the **int_tgt** target. You review the distribution of the interval target **int_tgt** and fit several candidate models using the GENMODEL statement with select distributions.

Open the program **mldmbd04d02_GeneralizedLinearModel_IntervalTargets.sas**.

Recall from earlier exploration that **int_tgt** is missing whenever **b_tgt=0**. Consequently, your analysis is restricted to only cases where **int_tgt** is greater than 0. (Missing is treated as the smallest numeric variable by SAS.) Also, for skewed data such as the interval target, modeling the log of the target can lead to better models. Using the COMPUTE statement, you can add new columns to an in-memory table on the LASR Analytic Server. In this example, a new column, **log_int_tgt**, is calculated. To use a computed column in subsequent actions, you must use the TABLE statement and set the modified table to active. Run the program and look at the histogram plots.

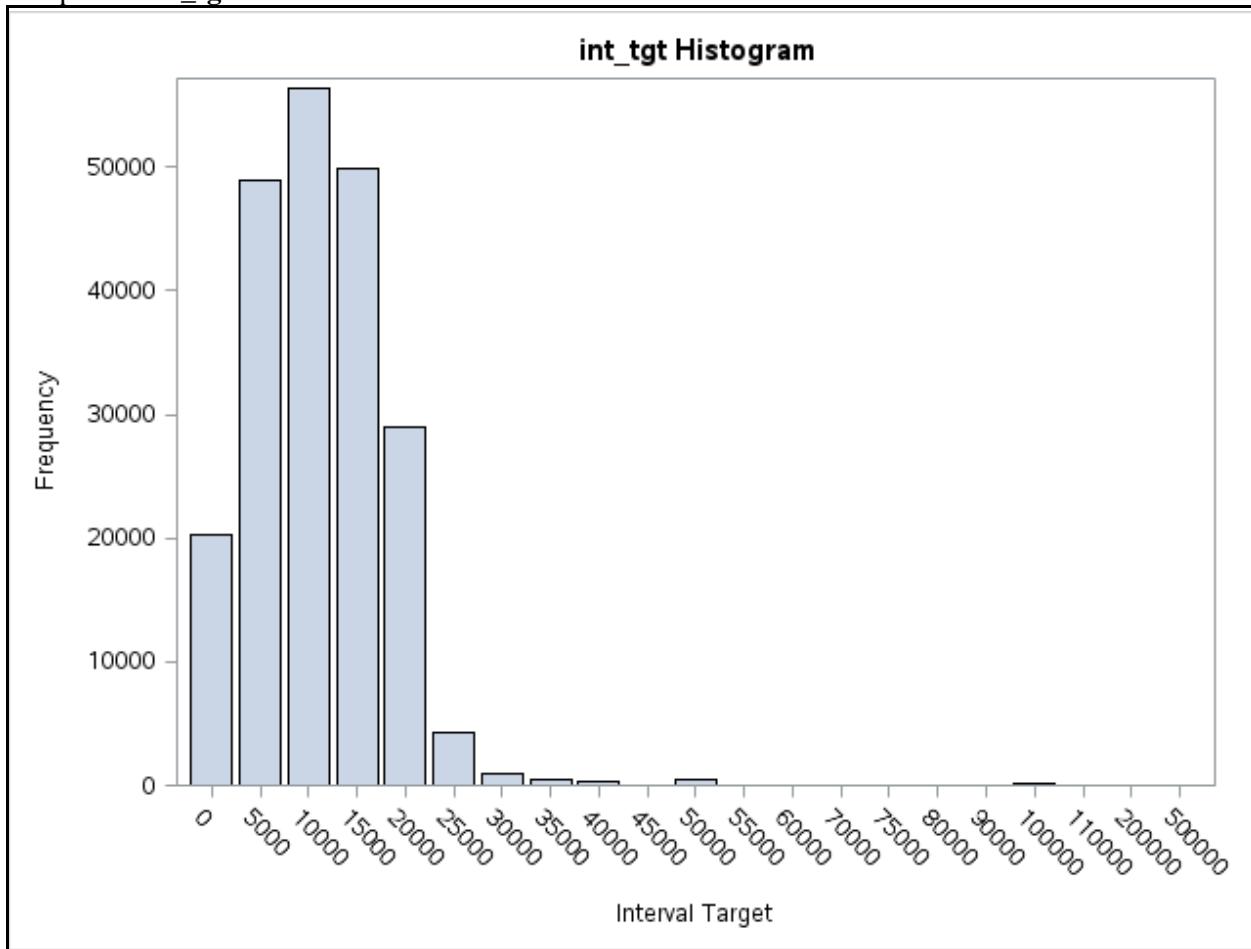
```

proc imstat;
  table LASRlib.p model_bank13(tag)="&TagString";
  where int_tgt gt 0;
  compute Log_int_tgt "Log_int_tgt=log(int_tgt);";
run;
/*---- Check to ensure computed variable ----*/
/*---- is available and correct           ----*/
table LASRlib.p_model_bank13(tag)="&TagString";
columninfo;
run;
ods output histogram = work.logint histogramdata;
histogram log_int_tgt/nbins=200 noemptybin;
run;
ods output histogram = work.int_histogramdata;
histogram int_tgt/nbins=200 noemptybin;
run;
quit;
title1 "int tgt Histogram";
proc sgplot data=work.int_histogramdata;
vbar binmid / freq=frequency;
label binmid="Interval Target";
run;

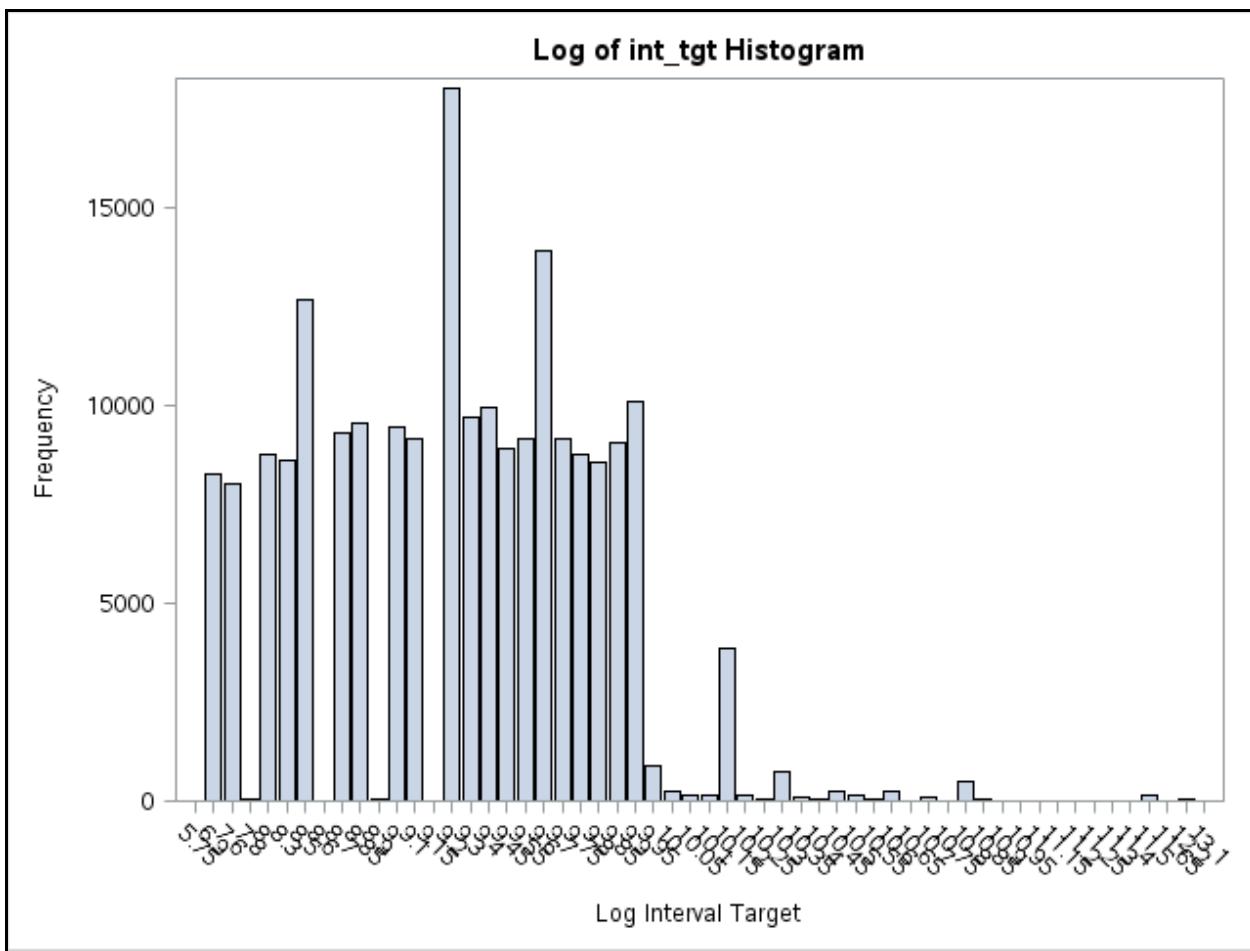
title1 "Log of int_tgt Histogram";
proc sgplot data=work.logint histogramdata;
vbar binmid / freq=frequency;
label binmid="Log Interval Target";
run;

```

The plot of **int_tgt** is shown below.



Here is the plot of **log_int_tgt**.



The distribution of **log_int_tgt** is much less skewed than the original target. This suggests that a log link with an appropriate distribution might be suitable for modeling.

A gamma regression with the log link is fit using the code below.

The WHERE clause restricts the data to the training partition and only cases where **int_tgt** is greater than zero.

```
proc imstat;
  table LASRlib.p model bank13(tag=&TagString");
  where _partind_=1 and int_tgt gt 0;
  genmodel int_tgt(&catvars)= &i_rfms &i_rfms &catvars /
    ID=(account)
    dist=gamma
    link=log
    score(_all_)
    code=(filename=&ScoreFolder/int_tgt_Gamma_ScoreCode.sas"
          replace);
run;
```

The fit statistics are shown below.

Fit Statistics for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13	
-2 Log Likelihood	2131554
AIC	2131606
AICC	2131606
BIC	2131854

The parameter estimates are listed below.

Parameter Estimates for USER.STUDENT2.HPADATA.P_MODEL_BANK13				
Parameter	Estimate	Standard Error	z Value	Pr > z
Intercept	9.1842	0.02072	443.23	<.0001
i_rfml	0.007012	0.000734	9.55	<.0001
i_rfml2	0.007675	0.000858	8.95	<.0001
i_rfml3	0.004072	0.000545	7.47	<.0001
i_rfml4	0.001810	0.000365	4.98	<.0001
i_rfml5	-0.02760	0.001882	-14.67	<.0001
i_rfml6	0.001101	0.000550	2.00	0.0454
i_rfml7	0.008235	0.002405	3.42	0.0006
i_rfml8	-0.00097	0.001189	-0.81	0.4167
i_rfml9	0.008720	0.000580	15.03	<.0001
i_rfml10	-0.00294	0.000600	-4.91	<.0001
i_rfml11	0.007668	0.001933	3.97	<.0001
i_rfml12	-0.00015	0.000104	-1.47	0.1429
i_demog_age	-0.00026	0.000135	-1.90	0.0571
demog_ho	-0.00433	0.004212	-1.03	0.3042
ri_demog_homeval	-3.11E-7	0	Infty	<.0001
ri_demog_inc	8.227E-8	1.185E-7	0.69	0.4875
demog_pr	-0.00011	0.000172	-0.65	0.5182
demog_genf	-0.01028	0.004033	-2.55	0.0108
demog_genm	0	-	-	-
cat_input1 X	-0.05972	0.01044	-5.72	<.0001
cat_input1 Y	-0.04454	0.01258	-3.54	0.0004
cat_input1 Z	0	-	-	-
cat_input2 A	0.002061	0.006673	0.31	0.7575
cat_input2 B	0.008180	0.006145	1.33	0.1831
cat_input2 C	0.02474	0.006319	3.92	<.0001
cat_input2 D	-0.00905	0.007281	-1.24	0.2138
cat_input2 E	0	-	-	-
Scale Parameter	2.4506	0.01005		

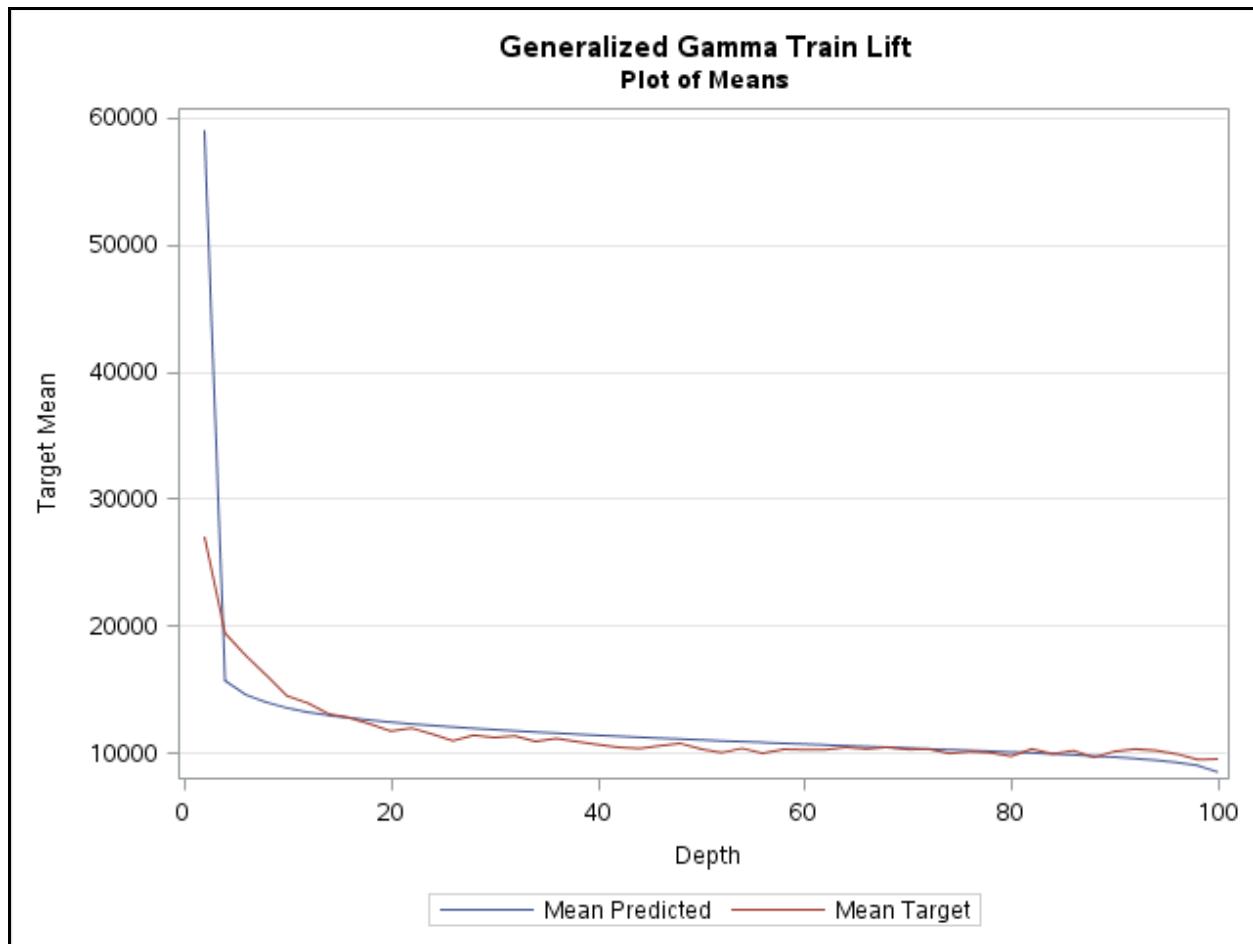
Some of the inputs are not significantly different from zero. This implies that a more parsimonious model might be appropriate. Other candidate models are fit and examined below.

The code below is very similar to that shown in earlier demonstrations. Run the code and examine the means plot.

```
table LASRlib.&_templast_;
columninfo;
run;
fetch account int_tgt _pred_ _ilink_ _resid_ / from=1 to=10;
run;
title1 "Train Lift on Original Scale";
ods output liftreginfo=work.TrainGammaRegLift;
assess _ilink_ /y=int_tgt nbins=50;
run;
quit;

title1 "Generalized Gamma Train Lift";
title2 "Plot of Means";
proc sgplot data=work.trainGammaReglift;
series y=meanp x=depth / legendlabel="Mean Predicted" name="line1";
series y=meant x=depth / legendlabel="Mean Target" name="line2";
yaxis label="Target Mean" grid;
keylegend "line1" "line2" / location=outside position=bottom;
run;
```

The means plot is shown.



The model severely over-predicts for the highest values of the target. This suggests fitting a model with a distribution that is less skewed.

The code below fits a Poisson regression to the interval target.

Run the program and examine the means plot.

```
proc imstat;
  table LASRlib.p_model_bank13(tag="&TagString") ;
  where partind =1 and int tgt gt 0;
  genmodel int_tgt(&catvars)= &i_rfms &ir_demogs &catvars /
    ID=(account) dist=poisson link=log
    score( all ) code=(filename=
      "&ScoreFolder/int_tgt_Poisson_ScoreCode.sas" replace);
run;

  table LASRlib.&_templatst_;
  columninfo;
run;

  fetch account int_tgt _pred_ _ilink_ _resid_ / from=1 to=10;
run;
```

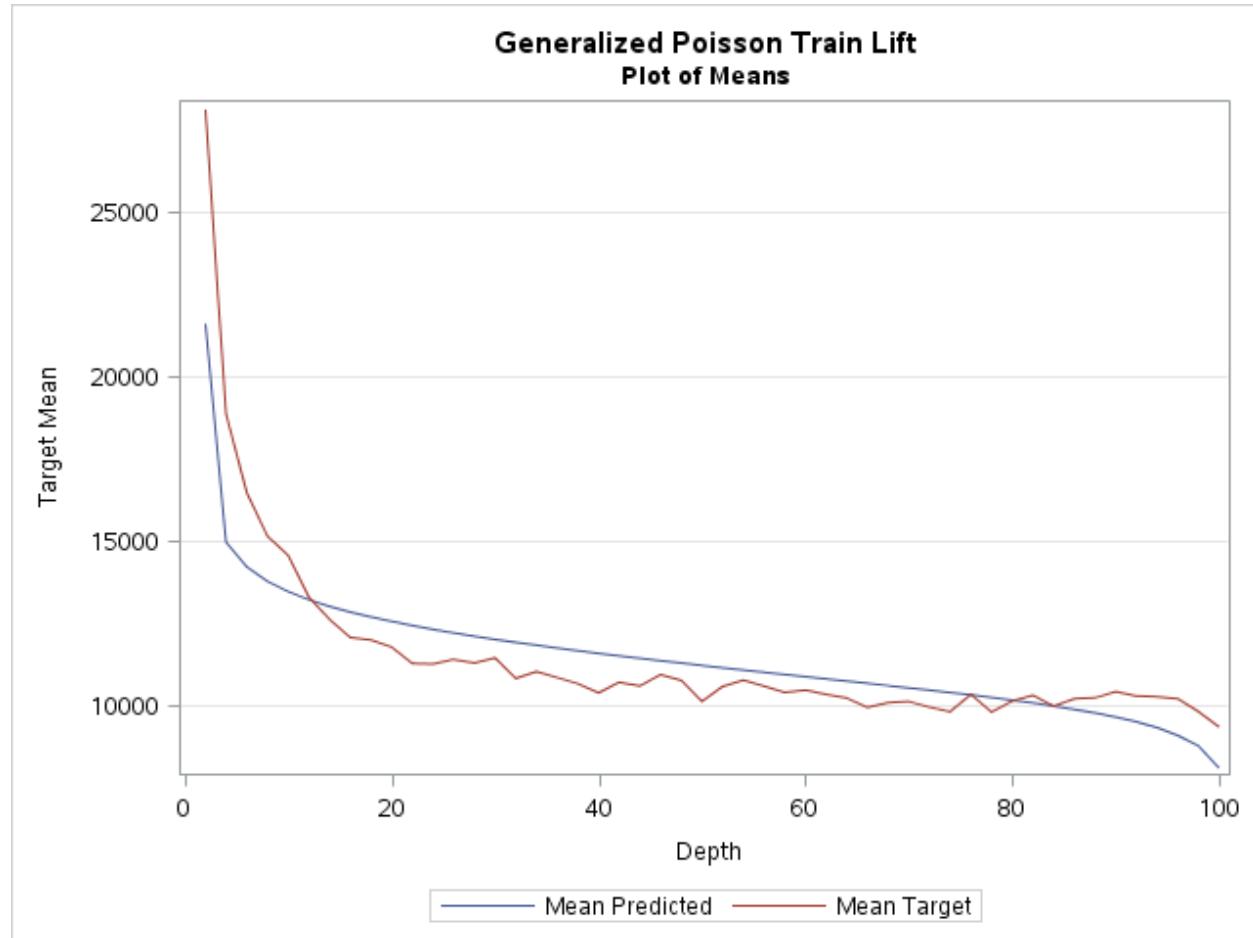
```

title1 "Train Lift on Original Scale";
ods trace on;
ods output liftreginfo=work.TrainPoissonRegLift;
assess _ilink_ / y=int_tgt nbins=50;
run;
quit;

title1 "Generalized Poisson Train Lift";
title2 "Plot of Means";
proc sgplot data=work.trainPoissonReglift;
series y=meanp x=depth / legendlabel="Mean Predicted" name="line1";
series y=meant x=depth / legendlabel="Mean Target" name="line2";
yaxis label="Target Mean" grid;
keylegend "line1" "line2" / location=outside position=bottom;
run;

```

The means plot is shown.



The plot shows that this model under-predicts the highest values of the target.

Another candidate model uses the lognormal distribution. A lognormal model can be fit using a linear model on the log of a target. (This is shown in the next demonstration.)

End of Demonstration

GLM Statement Procedure Model

$$y = X\beta + \varepsilon$$

Assume $E[\varepsilon] = 0, Var[\varepsilon] = \sigma^2 I_n$

Therefore, $E[y] = X\beta, Var[y] = \sigma^2 I_n$

63

Lognormal Model

When a response variable follows the lognormal distribution, it is possible to fit lognormal regression by taking the natural log of the response and estimating parameters as usual with OLS.

$$\ln(y) = X\beta + \varepsilon$$

The lognormal distribution is not a member of the exponential family.

64

A Note about the Lognormal Model

Unlike a generalized linear model, exponentiating the estimate on the log scale does not result in an unbiased estimate of the mean.

The reverse-transformation $e^{X\beta}$ results in an unbiased estimate of the **median** of the original data.

To estimate the mean, you must apply an adjustment:

$e^{X\beta + \frac{\hat{\sigma}^2}{2}}$ where $\hat{\sigma}^2$ is the variance estimate on the log scale.

The GLM action can be used on a logged variable to fit a lognormal model.

65

GLM Statement and Selected Options

GLM dependent-variable
(class-variables)=model-effects / option(s);

Options

- ROLEVAR=variable-name has a value of 1, t, or T for training or 2, v, or V for validation.
- VARSELECTION specifies to do backward selection of effects.
- SLSTAY=alpha specifies the upper-bound *p*-value to retain effects in the selection process.
- CODE=(filename="full path") saves score code to score holdout data for model assessment or deployment.

66



Fitting a Lognormal Model to an Interval Target

This demonstration illustrates how to fit a linear model to the log of the `int_tgt` target. The demonstration continues with the `6_6_Generalized_Linear_Models_for_Interval_Targets.sas` program.

The code below uses the GLM statement to fit a model to the log of `int_tgt`.

-  When SAS was first developed in the late 1960s and early 1970s, the acronym GLM was commonly used for the general linear model. One of the oldest procedures in the SAS System is PROC GLM. The GLM procedure fits general linear models that are still commonly used in the analysis of experimental data generated from designed experiments. The GLM statement in PROC IMSTAT is consistent with the models that are fit with the GLM procedure.

```
proc imstat;
  table LASRlib.p model bank13(tag="&TagString") ;
  where _partind_=1 and int_tgt gt 0 ;
  glm log int tgt(&catvars)= &i_rfms &ir_demoags &catvars /
    ID=(account)
    score(_all_)
    code=(filename="&ScoreFolder/int_tgt_LogNorm_ScoreCode.sas"
          replace);
run;
```

The GLM results have some differences from the results of other models that are demonstrated.

One table is the overall analysis of variance.

Overall Analysis of Variance for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13						
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F	R-Square
Model	24	2550.11	106.2544474	185.68	<.0001	0.040709
Error	105013	60092.40	0.5722378			
Corrected Total	105037	62642.51				

Another difference is that the parameters estimates are not shown.

The Type III test for the effects is shown here.

Model Analysis of Variance (Type III) for Table USER.STUDENT2.HPADATA.P_MODEL_BANK13					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
i_rfm1	1	71.7284937	71.7284937	125.35	<.0001
i_rfm2	1	0.4437850	0.4437850	0.78	0.3785
i_rfm3	1	27.4460266	27.4460266	47.96	<.0001
i_rfm4	1	9.0364308	9.0364308	15.79	<.0001
i_rfm5	1	234.5415524	234.5415524	409.87	<.0001
i_rfm6	1	11.0133956	11.0133956	19.25	<.0001
i_rfm7	1	28.1921074	28.1921074	49.27	<.0001
i_rfm8	1	11.7691587	11.7691587	20.57	<.0001
i_rfm9	1	130.8631051	130.8631051	228.69	<.0001
i_rfm10	1	12.2922817	12.2922817	21.48	<.0001
i_rfm11	1	19.5980355	19.5980355	34.25	<.0001
i_rfm12	1	0.000587857	0.000587857	0.00	0.9744
i_demog_age	1	1.4389554	1.4389554	2.51	0.1128
demog_ho	1	0.0479823	0.0479823	0.08	0.7721
ri_demog_homeval	1	120.2802840	120.2802840	210.19	<.0001
ri_demog_inc	1	0.4309184	0.4309184	0.75	0.3855
demog_pr	1	0.1432416	0.1432416	0.25	0.6169
demog_genf	0	-	-	-	-
demog_genm	0	-	-	-	-
cat_input1	2	33.3084096	16.6542048	29.10	<.0001
cat_input2	4	19.1956873	4.7989218	8.39	<.0001

This program assesses the model fit above and produces a means plot. The code is essentially the same as in previous demonstrations. Run the program and examine the means plot.

```





```

The means plot is shown below. It is important to notice the scale in the plot. The predictions are on the log scale. In the previous demonstration, the means plots were on the original scale. Because the log transformation was done internally in the statement, the `_ILINK_` variable was available to plot the means on the original scale. To produce a comparable plot, you can use a computed column to transform to either the median or mean prediction and generate plots.



End of Demonstration

4.3 Interval Targets (SAS Enterprise Miner)

In this section, a mixture distribution model is fitted to the count target in the **VS_BANK** data set. Mixture distributions, as the name suggests, are a combination or mixture of different parametric distributions. Zero-inflated distributions, such as zero-inflated Poisson (ZIP), are a commonly used class of mixture distributions that combine a binomial distribution with another distribution (such as a Poisson, in the case of ZIP). The HP GLM node in Enterprise Miner is chosen for the modeling task, because it has the functionality to accommodate the large proportion of zeros that were seen in the count target in a previous demonstration.

Interval Target Distributions in HP GLM

- Normal
- Gamma
- Negative Binomial
- Poisson
- Tweedie
- Zero-inflated Poisson
- Zero-inflated Negative Binomial
- Inverse Gaussian

69

Different distributions are supported for different applications. For example, the inverse Gaussian distribution is popular in reliability and survival analysis. The Tweedie distribution arises in diverse areas ranging from economics to life sciences. For example, the Tweedie distribution was applied to fisheries science related to zero-catch situations.

Zero-Inflated Poisson (ZIP) Distribution

$$pr(Y = y) = \begin{cases} p + (1-p)e^{-\lambda} & y = 0 \\ (1-p)\frac{e^{-\lambda}\lambda^y}{y!} & y > 0 \end{cases}$$

pdf(Poisson(y=0))
pdf(Poisson(y>0))

λ is the Poisson mean and $\log(\lambda) = X\beta + Z\gamma = \eta$.

p is the probability of the state where only zero count is possible. p can be the following:

- dependent on the same or different covariates that affect λ : $\text{logit}(p) = X_p\beta_p = \eta_p$
- a constant parameter
- a decreasing function of λ

70

The likelihood is defined conditioned on the target value. When the target = 0, the probability is the sum of the probability of the value being zero from the binary distribution and from the Poisson. Otherwise, the probability equals the probability that the binary distribution equals one times the Poisson distribution probability.

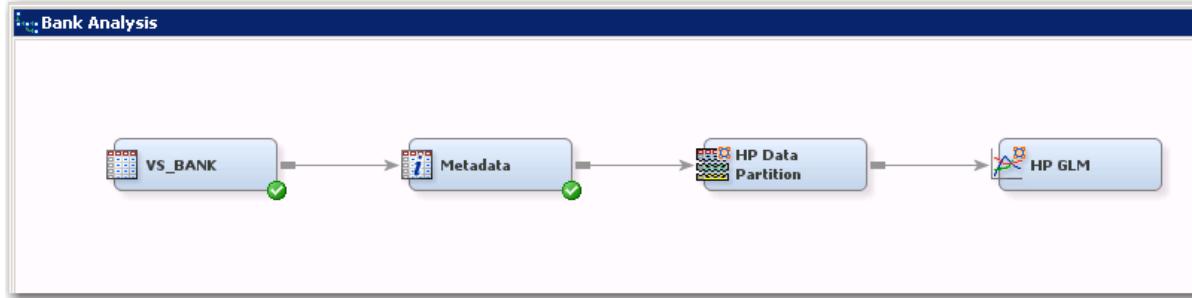


Fitting a Zero-Inflated Poisson Model Using the HP GLM Node in SAS Enterprise Miner

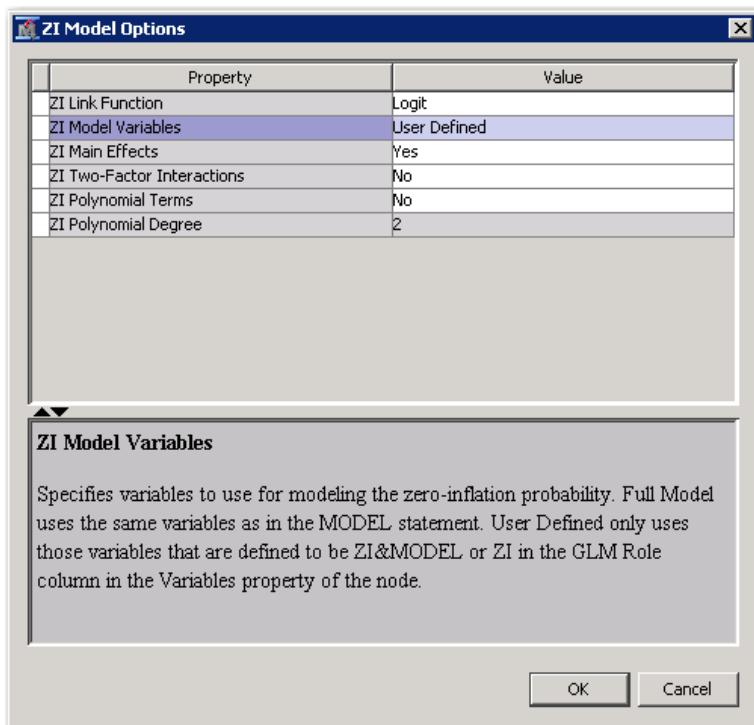
Recall that the count target (**cnt_tgt**) variable in the **VS_BANK** data is integer valued and primarily comprised of zeros. For data with this property, the generalized linear model with Poisson distribution (fitted in a previous demonstration) tends to under-predict the likelihood of a zero outcome. A zero-inflated Poisson model is fit to accommodate the “spike” in the count target variable’s distribution at zero.

1. Open the **EM_Project_Bank** project and **Bank_Analysis** diagram that you worked with in a previous Enterprise Miner demonstration.
2. Navigate to the **HPDM** tab, and click and drag an **HP Data Partition** node into the diagram. Connect it to the **Metadata** node. Change the Data Set Allocation property in the HP Data Partition node to **50** for both the Validation and Train data sets.
3. Navigate to the **HPDM** tab, and click and drag an **HP GLM** node into the flow diagram. Then, connect it to the **HP Data Partition** node.

Your Bank_Analysis diagram should look similar to the following display:

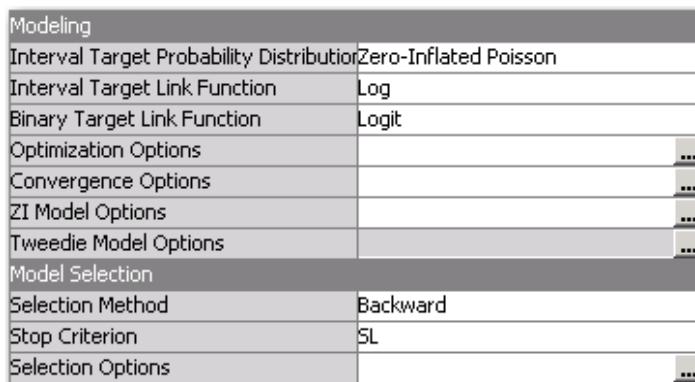


4. In the properties of the HP GLM node, change the Interval Target Probability Distribution to **Zero-Inflated Poisson**.
5. Select the ellipsis next to the **ZI Model Options** property, and set ZI Model Variables to **User Defined**. This setting enables the user to restrict variable selection for the binary and Poisson models to subsets of candidate input variables.

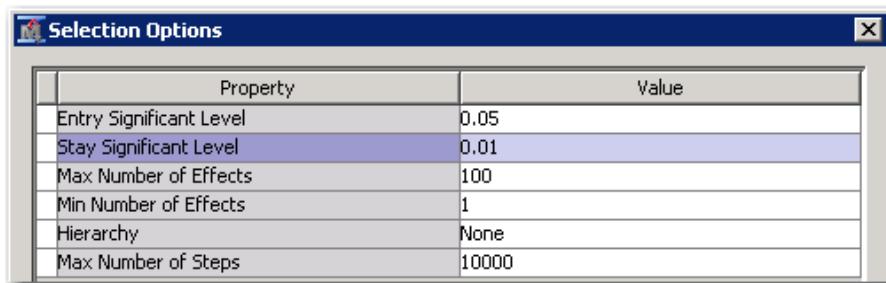


6. Scroll down to the **Model Selection** properties. Set the Selection Method to **Backward**, and the Stop Criterion to **SL**. The default is SL only.

SL indicates that selection is based on a *p*-value significance level.



7. Select the ellipsis next to **Selection Options**. Change the default values for the Stay Significant Level, Max Number of Effect, Min Number of Effects, and Max Number of Steps to those shown below.



8. Select the ellipsis next to the **Variables** property. Set Use to **Yes** for **cnt_tgt**, and set Use to **No** for **b_tgt**.

The measurement level of the target must be *Interval* for this model.

9. Demographic and categorical candidate input variables are hypothesized to influence the decision to buy or not to buy (be in the zero state or not). Set the GLM Role of these candidate input variables to **ZI Model**.

The GLM Role ZI Model restricts variable selection for the binary (ZI) model to the designated subset of candidate input variables.

Name	Use	GLM Role	Role	Level
cat_input1	Default	ZI Model	Input	Nominal
cat_input2	Default	ZI Model	Input	Nominal
demog_genm	Default	ZI Model	Input	Binary
demog_ho	Default	ZI Model	Input	Binary
ri_demog_jnc	Default	ZI Model	Input	Interval
ri_demog_homev	Default	ZI Model	Input	Interval
demog_pr	Default	ZI Model	Input	Interval
i_demog_age	Default	ZI Model	Input	Interval
logi_rfml1	Default	Default	Input	Interval
cnt_tgt	Yes	Default	Target	Interval
logi_rfml1	Default	Default	Input	Interval
logi_rfml2	Default	Default	Input	Interval
logi_rfml0	Default	Default	Input	Interval
logi_rfml8	Default	Default	Input	Interval
logi_rfml2	Default	Default	Input	Interval
logi_rfml3	Default	Default	Input	Interval
logi_rfml4	Default	Default	Input	Interval
logi_rfml6	Default	Default	Input	Interval
logi_rfml9	Default	Default	Input	Interval
logi_rfml5	Default	Default	Input	Interval
logi_rfml7	Default	Default	Input	Interval
b_tgt	No	Default	Target	Binary

10. Right-click and click **Run** on the HP GLM node. Open the results after it runs.

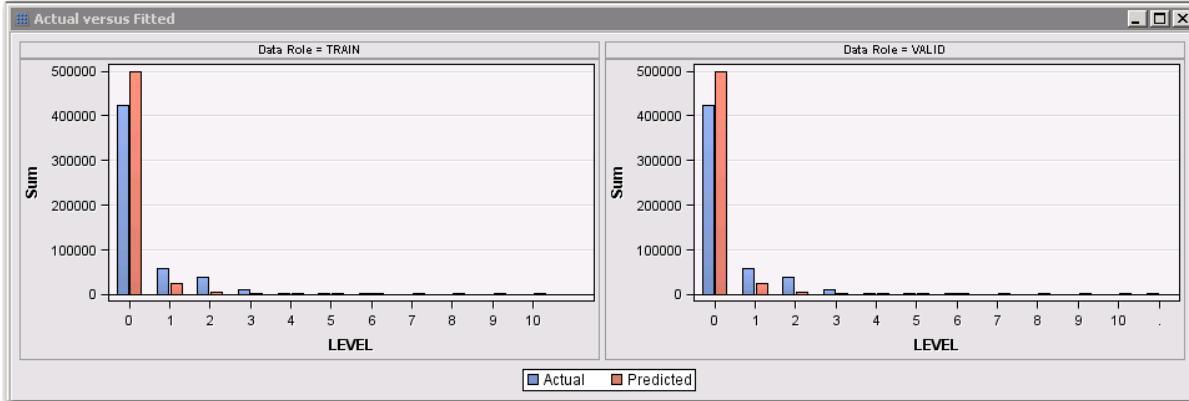
The ZI Parameter Estimates table lists the variables selected in the ZI or binomial model. This model is used to predict zero-state membership. The variables that are selected for this model contain information about credit worthiness and demographics.

ZI Parameter Estimates				
Effect	Parameter	Level	Estimate	Pr > Chi-Square
Intercept_Zero	Intercept_Zero		1.186168	<.0001
i_demog_age_...	i_demog_age_...		-0.006683	<.0001
ri_demog_ho...	ri_demog_ho...		-0.000018280	<.0001
cat_input1_Zero	cat_input1_Zer... Y		0.076731	0.0178
cat_input1_Zero	cat_input1_Zer... Z		0.750313	<.0001
cat_input1_Zero	cat_input1_Zer... X		0	.
cat_input2_Zero	cat_input2_Zer... D		0.036079	0.1666
cat_input2_Zero	cat_input2_Zer... C		-0.090229	0.0001
cat_input2_Zero	cat_input2_Zer... A		-0.245073	<.0001
cat_input2_Zero	cat_input2_Zer... B		-0.071303	0.0115
cat_input2_Zero	cat_input2_Zer... E		0	.

The Parameter Estimates table provides information about the variables that are selected for the Poisson regression model. This model predicts the number of purchases.

Effect	Parameter	Estimate	Pr > Chi-Square
Intercept	Intercept	1.185241	<.0001
logi_rf1	logi_rf1	-0.098597	<.0001
logi_rf10	logi_rf10	0.097261	<.0001
logi_rf11	logi_rf11	-0.441309	<.0001
logi_rf12	logi_rf12	-0.013413	0.0205
logi_rf2	logi_rf2	-0.650258	<.0001
logi_rf4	logi_rf4	-0.029426	0.0001
logi_rf5	logi_rf5	1.844148	<.0001
logi_rf7	logi_rf7	-0.075818	<.0001
logi_rf9	logi_rf9	-0.759222	<.0001

The Actual versus Fitted histogram indicates that the model over-predicts the number of zeros, and also tends to under-predict the nonzero outcomes.



11. Close the Results window. Then navigate to the **Exported Data** property of the HP GLM node. Select the ellipsis next to **Exported Data**, and then highlight the **Train** data set listed in the table.
12. Select **Explore**.

The **Predicted: cnt_tgt** column contains the model-based prediction of the number of purchases for each case. This prediction is conditioned on both the covariates in the Poisson regression model and the probability of zero-state membership.

Predicted: cnt_tgt	Residual: cnt_tgt
3.236589	-1.23659
0.210357	-0.21036
0.114905	-0.11491
0.24674	-0.24674
0.400418	1.599582
0.22913	-0.22913
0.264193	0.735807
0.059097	-0.0591
0.402841	0.597159

End of Demonstration

4.4 Solutions

Solutions to Exercises

1. Building a Linear Model to Predict the Amount of a Donation

In this exercise, you use the **PVA97NK** data to build and refine a linear regression model. This model attempts to predict the amount of donation given to the PVA.

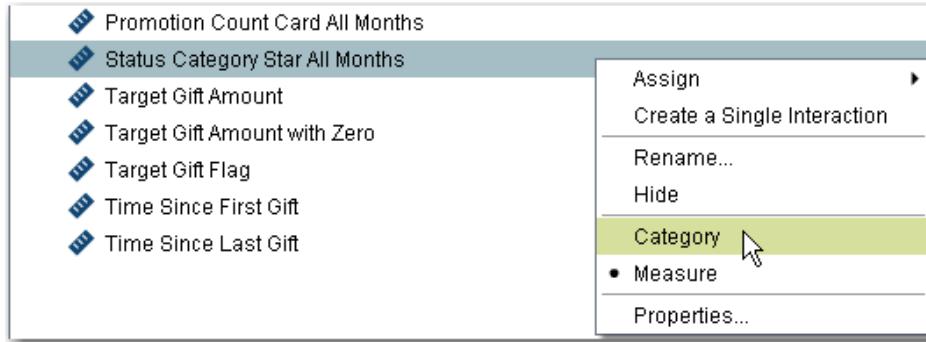
a. Exercise setup: Create the exercise project and linear model analysis.

- 1) From the SAS Visual Analytics Home Page, select **Data Exploration**.
- 2) Select the **PVA97NK** data source and open it.
- 3) Click the **Linear Regression** tab. Clear the **Auto-Update** check box, and assign variable roles as shown below.



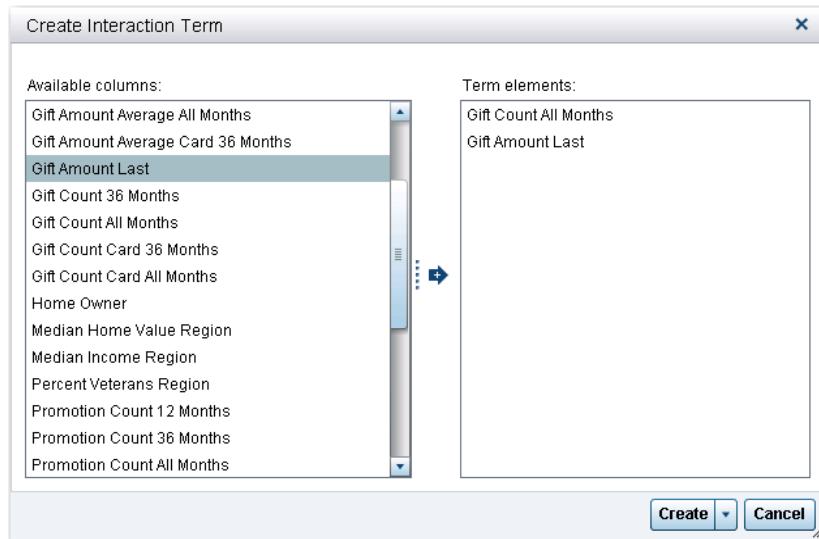
- 4) Click **Update** to run the model.
- How many observations were used to build the linear regression model? **34,111**
 - What is the R-square value of the model? **.5651**
 - Modify the variable measures.

1) In the Data pane, change **Status Category Star All Months** from a measure to a *category*.



- Assign **Status Category 96NK** as a classification effect. **Status Category Star All Months** is automatically reassigned as a classification effect.
- On the Properties tab, select both the **Informative missingness** and **Use variable selection** check boxes. Change the significance level to **0.01**. Click **Update** to rerun the model.
 - How many observations were used to build the model? **53,273**
 - What is the R square value of the model now? **.562**
 - Does variable selection exclude any input variables from the model?
Select **Show Details** and click the **Type III Test** tab. More than half of the candidate input variables are excluded from the final model. This is indicated by zero degrees of freedom associated with these variables.
 - How many dummy variables does the informative missingness property add to the model?
Click the **Parameter Estimates** tab under the **Show Summary Tables** results. Twenty missing value indicator variables are created (variables with the _miss suffix). That is, 20 of the candidate input variables have missing values.
- Domain experts hypothesize an interaction between gift frequency and gift amount. Explore this hypothesis using the **Gift Count All Months** and **Gift Amount Last** variables. Is the interaction significant?
 - Select **Create** next to **Interactions** on the Roles tab.

- 2) Create the interaction variable by highlighting the variables listed below and selecting the arrow. Then select **Create**.



- 3) Click **Update** to run the model.
- 4) Open the Type III Test table. The interaction term is included in the final model. It appears to be a relevant predictor of the interval target variable.
- g. Save the project as **VS Interval Target Exercise**.
- 2. Building a GLM Model to Predict the Amount of Donations**
- In this exercise, you use the **PVA97NK** data to build a GLM model. The target variable is transformed to be more consistent with the restrictions that regulate the analysis.
- Open **VS Interval Target Exercise**, which was saved earlier in this chapter. Examine the results of the linear model.
 - Does there seem to be a problem with model bias?
No, there does not appear to be a pattern in the differences between the prediction and outcome lines in the assessment plot.
 - Do the residuals seem to satisfy the assumption of constant variance?
No, the residuals plot indicates that the residuals variance increases with the magnitude of the prediction.
 - Build an appropriate GLM model for the continuous target under the assumption that it follows a log-normal distribution. Use the same variable roles that you used for the linear model, excluding the interaction term. Select the **Informative missingness** property.
 - Does the GLM model solve the problems that you found in the linear model, if any?
Yes, the residuals' variance seems more consistent with the assumption that it is constant. There are several large outlier residuals found in the plot.
 - Save the changes to **VS Interval Target Exercise**. Select **File** \Rightarrow **Save As** \Rightarrow **VS Interval Target Exercise**.

End of Solutions

Solutions to Student Activities (Polls/Quizzes)

4.01 Poll – Correct Answer

Linear regression models are appropriate for any continuous response variable.

- True
- False

26

4.02 Poll – Correct Answer

You can perform linear regression analysis with a generalized linear model.

- True
- False

39

4.03 Poll – Correct Answer

You can perform logistic regression analysis with a generalized linear model.

- True
- False