

Predictive Modeling Using Logistic Regression

Course Notes

Predictive Modeling Using Logistic Regression Course Notes was developed by Mike Patetta. Additional contributions were made by Dan Kelly. 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.

Predictive Modeling Using Logistic Regression Course Notes

Copyright © 2008 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 E1329, course code LWPMLR92/PMLR92, prepared date 29Jul2008. LWPMLR92_001

ISBN 978-1-59994-787-7

Table of Contents

Course Description	vi
Prerequisites	vii
Chapter 1 Predictive Modeling.....	1-1
1.1 Introduction.....	1-3
1.2 Analytical Challenges	1-9
1.3 Chapter Summary	1-15
Chapter 2 Fitting the Model.....	2-1
2.1 The Model.....	2-3
2.2 Adjustments for Oversampling	2-21
2.3 Chapter Summary	2-30
2.4 Solutions	2-31
Chapter 3 Preparing the Input Variables	3-1
3.1 Missing Values	3-3
3.2 Categorical Inputs	3-13
3.3 Variable Clustering and Screening.....	3-26
3.4 Subset Selection	3-66
3.5 Chapter Summary	3-78
3.6 Solutions	3-79
Chapter 4 Measuring Classifier Performance	4-1
4.1 Honest Assessment.....	4-3
4.2 Misclassification	4-32

4.3	Allocation Rules	4-49
4.4	Overall Predictive Power	4-55
4.5	Chapter Summary	4-64
4.6	Solutions	4-65
Chapter 5 Generating and Evaluating Many Models		5-1
5.1	Model Selection Plots	5-3
5.2	Chapter Summary	5-22
5.3	Solutions	5-23
Appendix A Exercises and Solutions		A-1
Session 1: Exercises		A-4
Chapter 2 Exercises		A-4
Chapter 3 Exercises		A-4
Chapter 4 Exercises		A-6
Chapter 5 Exercises		A-6
Session 1: Solutions		A-7
Chapter 2 Solutions		A-7
Chapter 3 Solutions		A-14
Chapter 4 Solutions		A-36
Chapter 5 Solutions		A-44
Chapter 2 Additional Resources		2-1
2.1	Sampling Weights	2-2
2.2	References	2-6

Appendix C Index C-1

Appendix D Error! No table of contents entries found.Index D-1

Course Description

This course covers predictive modeling using SAS/STAT software with emphasis on the LOGISTIC procedure. This course also discusses selecting variables, assessing models, treating missing values, and using efficiency techniques for massive data sets.

To learn more...



For information on 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 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 our 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 Publications Catalog on the Web at support.sas.com/pubs for a complete list of books and a convenient order form.

Prerequisites

Before attending this course, you should

- have experience executing SAS programs and creating SAS data sets, which you can gain from the *SAS Programming I: Essentials* course
- have experience building statistical models using SAS software
- have completed a statistics course that covers linear regression and logistic regression, such as the *Statistics I: Introduction to ANOVA, Regression, and Logistic Regression* course.

Chapter 1 Predictive Modeling

1.1	Introduction.....	1-3
1.2	Analytical Challenges	1-9
1.3	Chapter Summary.....	1-15

1.1 Introduction

Supervised Classification

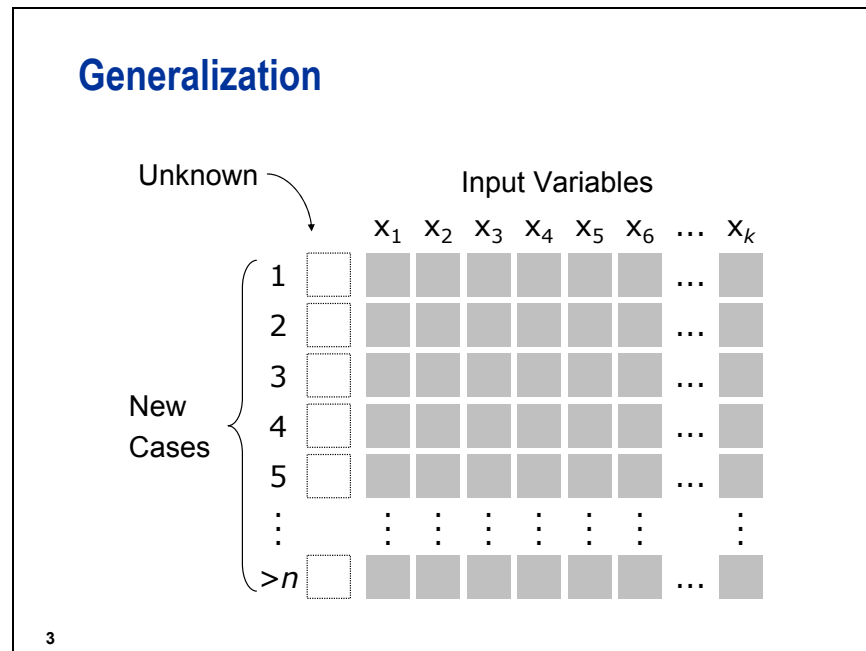
		(Binary) Target									
		Input Variables									
		y	x ₁	x ₂	x ₃	x ₄	x ₅	x ₆	...	x _k	
Cases	1	Blue	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	
	2	Blue	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	
	3	Red	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	
	4	Red	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	
	5	Blue	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
	n	Red	Gray	Gray	Gray	Gray	Gray	Gray	...	Gray	

2

The data used to develop a predictive model consists of a set of *cases* (*observations, examples*). Associated with each case is a vector of *input variables* (*predictors, explanatory variables, features*) and a *target variable* (*outcome, response*). A predictive model maps the vector of input variables to the target. The target is the outcome to be predicted. The cases are the units on which the prediction is made.

In *supervised classification* (Hand 1997), the target is a class label. A predictive model assigns, to each case, a score (or a set of scores) that measures the propensity that the case belongs to a particular class. With two classes, the target is binary and usually represents the occurrence of an event.

The term *supervised* is used when the class label is known for each case. If the label is known, then why build a prediction model?



The prediction model is used on new cases where the values of the input variables are known, but the class labels are unknown. The principal aim of predictive modeling is generalization. *Generalization* means the ability to predict the outcome on novel cases.

In contrast, the principal aim of traditional statistical analysis is inference. Confidence intervals, hypothesis tests, and p -values are the common inferential tools. Similar methods used by predictive modelers (such as logistic regression) may be used to infer how input variables affect the target. The validity of the inference relies on understanding the statistical properties of methods and applying them correctly.

Understanding the relationships between random variables can be important in predictive modeling as well. However, many of the methods used are ad hoc with poorly understood statistical properties. Consequently, the discovery of structure in predictive modeling is informal and exploratory. Some predictive modeling methods (for example, neural networks) are inscrutable yet successful because they generalize well. The validity of predictive modeling methods is assessed empirically. If a model generalizes well, then the method is useful, regardless of its statistical properties.

Applications



Target Marketing



Attrition Prediction



Credit Scoring



Fraud Detection

5

There are many business applications of predictive modeling. *Database marketing* uses customer databases to improve sales promotions and product loyalty. In target marketing, the cases are customers, the inputs are attributes such as previous purchase history and demographics, and the target is often a binary variable indicating a response to a past promotion. The aim is to find segments of customers that are likely to respond to some offer so they can be targeted. Historic customer databases can also be used to predict who is likely to switch brands or cancel services (churn). Loyalty promotions can then be targeted at new cases that are at risk.

Credit scoring (Hand and Henley 1997) is used to decide whether to extend credit to applicants. The cases are past applicants. Most input variables come from the credit application or credit reports. A relevant binary target is whether the case defaulted (charged-off) or paid-off the debt. The aim is to reduce defaults and serious delinquencies on new applicants for credit.

In fraud detection, the cases are transactions (for example, telephone calls, credit card purchases) or insurance claims. The inputs are the particulars and circumstances of the transaction. The binary target is whether that case was fraudulent. The aim is to anticipate fraud or abuse on new transactions or claims so that they can be investigated or impeded.

Supervised classification also has less business-oriented uses. Image classification has applications in areas such as astronomy, nuclear medicine, and molecular genetics (McLachlan 1992; Ripley 1996; Hand 1997).



Exploring the Data

The **develop** data set is a retail-banking example. The data set has 32,264 cases (banking customers) and 47 input variables. The binary target variable **Ins** indicates whether the customer has an insurance product (variable annuity). The 47 input variables represent other product usage and demographics prior to their acquiring the insurance product. Two of the inputs are nominally scaled; the others are interval or binary.

The DATA step reads in the original data and creates a data set in the work library. This prevents writing over the original data.

```
libname pmlr 'SAS-data-library';
data develop;
    set pmlr.develop;
run;
```

The %LET statement enables you to define a macro variable and assign it a value. The statement below creates a macro variable named **inputs** and assigns it a string that contains all the numeric input variable names. This reduces the amount of text you need to enter in other programs.

```
%let inputs=acctage dda ddabal dep depamt cashbk checks
dirdep nsf nsfamt phone teller atm atmamt pos posamt
cd cdbal ira irabal loc locbal inv invbal ils ilsbal
mm mmbal mmcred mtg mtgbal sav savbal cc ccbal
ccpurc sdb income hmown lores hmval age crscore
moved inarea;
```

The MEANS procedure generates descriptive statistics for the numeric variables. The statistics requested below are the number of observations, the number of missing values, the mean, the minimum value, and the maximum value. The macro variable, **inputs**, is referenced in the VAR statement by prefixing an ampersand (&) to the macro variable name. The FREQ procedure examines the values of the target variable and the nominal input variables.

```
proc means data=develop n nmiss mean min max;
    var &inputs;
run;
proc freq data=develop;
    tables ins branch res;
run;
```

The MEANS Procedure						
Variable	Label	N		Mean	Minimum	Maximum
		N	Miss			
AcctAge	Age of Oldest Account	30194	2070	5.9086772	0.3000000	61.5000000
DDA	Checking Account	32264	0	0.8156459	0	1.0000000
DDABal	Checking Balance	32264	0	2170.02	-774.8300000	278093.83
Dep	Checking Deposits	32264	0	2.1346082	0	28.0000000
DepAmt	Amount Deposited	32264	0	2232.76	0	484893.67
CashBk	Number Cash Back	32264	0	0.0159621	0	4.0000000
Checks	Number of Checks	32264	0	4.2599182	0	49.0000000
DirDep	Direct Deposit	32264	0	0.2955616	0	1.0000000
NSF	Number Insufficient Fund	32264	0	0.0870630	0	1.0000000
NSFAmt	Amount NSF	32264	0	2.2905464	0	666.8500000
Phone	Number Telephone Banking	28131	4133	0.4056024	0	30.0000000
Teller	Teller Visits	32264	0	1.3652678	0	27.0000000
Sav	Saving Account	32264	0	0.4668981	0	1.0000000
SavBal	Saving Balance	32264	0	3170.60	0	700026.94
ATM	ATM	32264	0	0.6099368	0	1.0000000
ATMAmt	ATM Withdrawal Amount	32264	0	1235.41	0	427731.26
POS	Number Point of Sale	28131	4133	1.0756816	0	54.0000000
POSAmt	Amount Point of Sale	28131	4133	48.9261782	0	3293.49
CD	Certificate of Deposit	32264	0	0.1258368	0	1.0000000
CDBal	CD Balance	32264	0	2530.71	0	1053900.00
IRA	Retirement Account	32264	0	0.0532792	0	1.0000000
IRABal	IRA Balance	32264	0	617.5704550	0	596497.60
LOC	Line of Credit	32264	0	0.0633833	0	1.0000000
LOCBal	Line of Credit Balance	32264	0	1175.22	-613.0000000	523147.24
Inv	Investment	28131	4133	0.0296826	0	1.0000000
InvBal	Investment Balance	28131	4133	1599.17	-2214.92	8323796.02
ILS	Installment Loan	32264	0	0.0495909	0	1.0000000
ILSBal	Loan Balance	32264	0	517.5692344	0	29162.79
MM	Money Market	32264	0	0.1148959	0	1.0000000
MMBal	Money Market Balance	32264	0	1875.76	0	120801.11
MMCred	Money Market Credits	32264	0	0.0563786	0	5.0000000
MTG	Mortgage	32264	0	0.0493429	0	1.0000000
MTGBal	Mortgage Balance	32264	0	8081.74	0	10887573.28
CC	Credit Card	28131	4133	0.4830969	0	1.0000000
CCBal	Credit Card Balance	28131	4133	9586.55	-2060.51	10641354.78
CCPurc	Credit Card Purchases	28131	4133	0.1541716	0	5.0000000
SDB	Safety Deposit Box	32264	0	0.1086660	0	1.0000000
Income	Income	26482	5782	40.5889283	0	233.0000000
HMOwn	Owns Home	26731	5533	0.5418802	0	1.0000000
LORes	Length of Residence	26482	5782	7.0056642	0.5000000	19.5000000
HMVal	Home Value	26482	5782	110.9121290	67.0000000	754.0000000
Age	Age	25907	6357	47.9283205	16.0000000	94.0000000
CRScore	Credit Score	31557	707	666.4935197	509.0000000	820.0000000
Moved	Recent Address Change	32264	0	0.0296305	0	1.0000000
InArea	Local Address	32264	0	0.9602963	0	1.0000000

The results of PROC MEANS show that several variables have missing values and several variables are (probably) binary.

The FREQ Procedure				
Ins	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	21089	65.36	21089	65.36
1	11175	34.64	32264	100.00

Branch of Bank				
Branch	Frequency	Percent	Cumulative Frequency	Cumulative Percent
B1	2819	8.74	2819	8.74
B10	273	0.85	3092	9.58
B11	247	0.77	3339	10.35
B12	549	1.70	3888	12.05
B13	535	1.66	4423	13.71
B14	1072	3.32	5495	17.03
B15	2235	6.93	7730	23.96
B16	1534	4.75	9264	28.71
B17	850	2.63	10114	31.35
B18	541	1.68	10655	33.02
B19	285	0.88	10940	33.91
B2	5345	16.57	16285	50.47
B3	2844	8.81	19129	59.29
B4	5633	17.46	24762	76.75
B5	2752	8.53	27514	85.28
B6	1438	4.46	28952	89.73
B7	1413	4.38	30365	94.11
B8	1341	4.16	31706	98.27
B9	558	1.73	32264	100.00

Area Classification				
Res	Frequency	Percent	Cumulative Frequency	Cumulative Percent
R	8077	25.03	8077	25.03
S	11506	35.66	19583	60.70
U	12681	39.30	32264	100.00

The results of PROC FREQ show that 34.6 percent of the observations have acquired the insurance product. There are 19 levels of **Branch** and 3 levels under **Res** (R=rural, S=suburb, U=urban).

1.2 Analytical Challenges

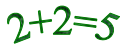
Opportunistic Data



Operational / Observational



Massive



Errors and Outliers









Missing Values

8

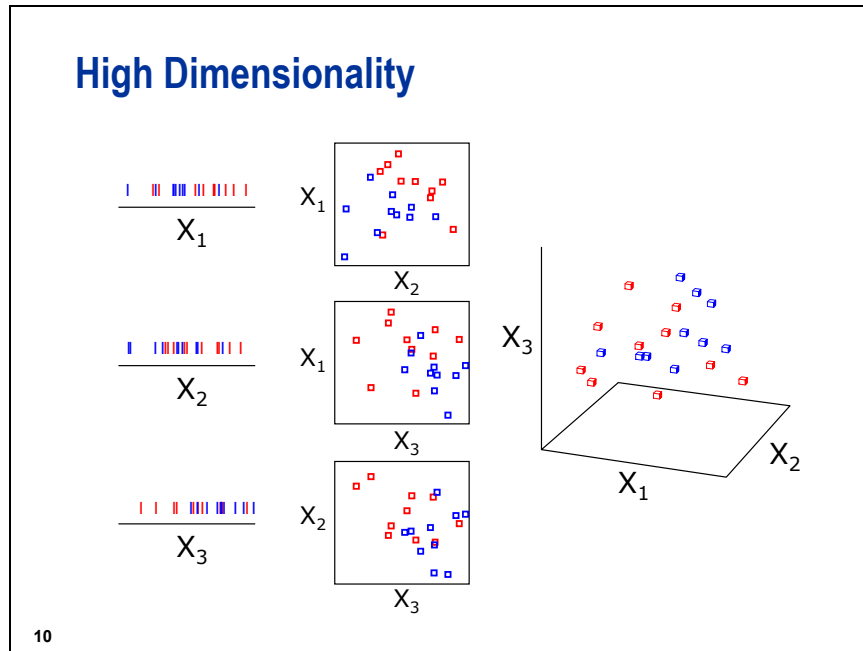
The data that is typically used to develop predictive models can be characterized as *opportunistic*. It was collected for operational purposes unrelated to statistical analysis (Huber 1997). Such data is usually massive, dynamic, and dirty. Preparing data for predictive modeling can be agonizing. For example, the parts of the data that are relevant to the analysis need to be acquired. Furthermore, the relevant inputs usually need to be created from the raw operational fields. Many statistical methods do not scale well to massive data sets because most methods were developed for small data sets generated from designed experiments.

Mixed Measurement Scales

	sales, executive, homemaker, ...
	88.60, 3.92, 34890.50, 45.01, ...
	F, D, C, B, A
	0, 1, 2, 3, 4, 5, 6, ...
	M, F
	27513, 21737, 92614, 10043, ...

9

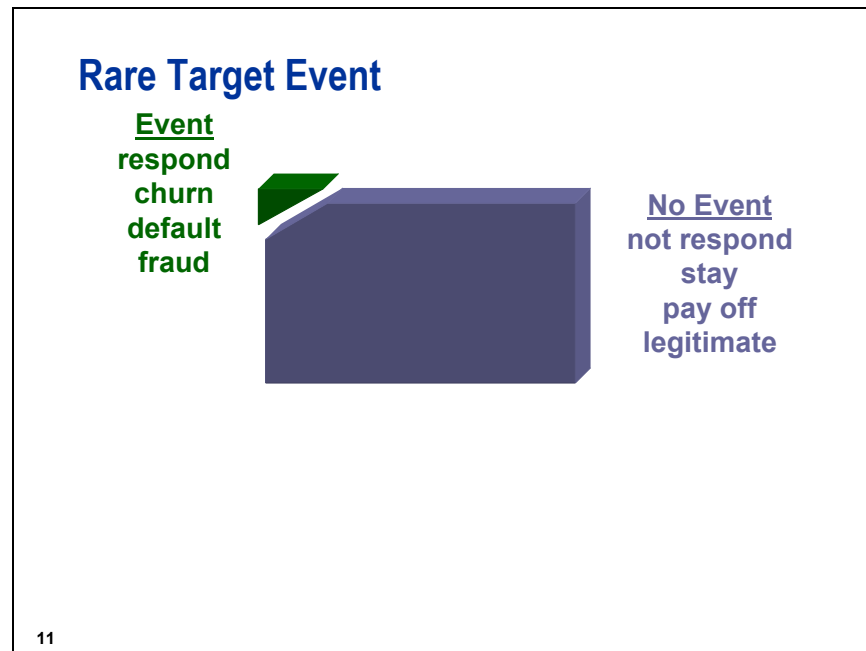
When there are a large number of input variables, there are usually a variety of measurement scales represented. The input variable may be intervally scaled (amounts), binary, nominally scaled (names), ordinally scaled (grades), or counts. Nominal input variables with a large number of levels (such as ZIP code) are commonplace and present complications for regression analysis.



The *dimension* refers to the number of input variables (actually input degrees of freedom). Predictive modelers consider large numbers (hundreds) of input variables. The number of variables often has a greater effect on computational performance than the number of cases. High dimensionality limits the ability to explore and model the relationships among the variables. This is known as the *curse of dimensionality*, which was distilled by Breiman et al. (1984) as

The complexity of a data set increases rapidly with increasing dimensionality.

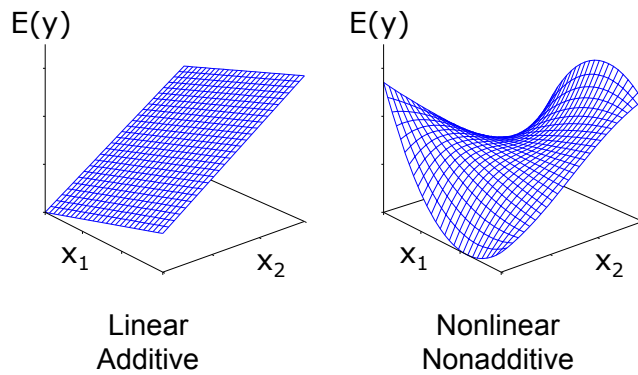
The remedy is dimension reduction; ignore irrelevant and redundant dimensions without inadvertently ignoring important ones.



In predictive modeling, the event of interest is often rare. Usually more data leads to better models, but having an ever-larger number of nonevent cases has rapidly diminishing return and can even have detrimental effects. With rare events, the effective sample size for building a reliable prediction model is closer to 3 times the number of event cases than to the nominal size of the data set (Harrell 1997). A seemingly massive data set might have the predictive potential of one that is much smaller.

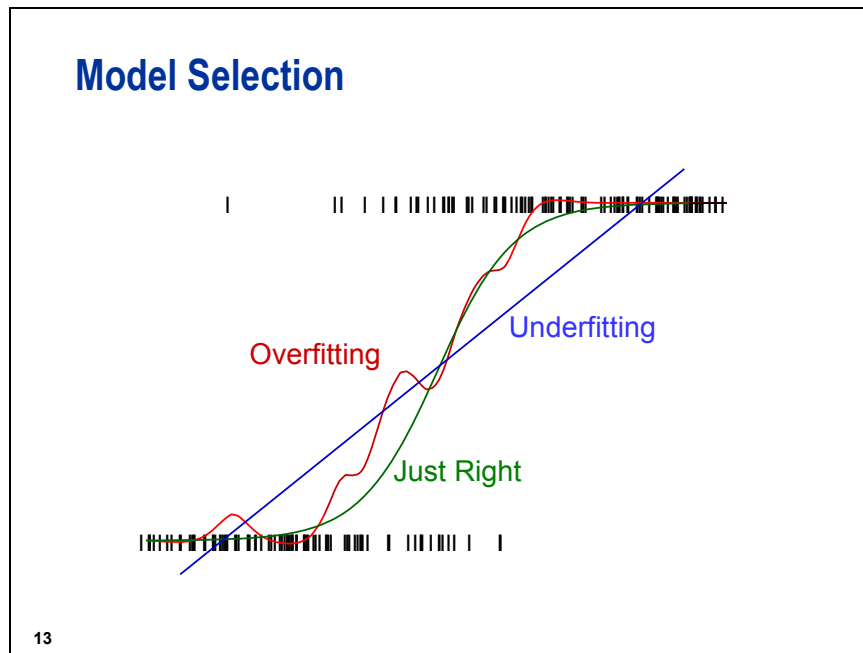
One widespread strategy for predicting rare events is to build a model on a sample that disproportionately over-represents the event cases (for example, an equal number of events and nonevents). Such an analysis introduces biases that need to be corrected so that the results are applicable to the population.

Nonlinearities and Interactions



12

Predictive modeling is a multivariate problem. Each important dimension might affect the target in complicated ways. Moreover, the effect of each input variable might depend on the values of other input variables. The curse of dimensionality makes this difficult to untangle. Many classical modeling methods (including standard logistic regression) were developed for inputs with effects that have a constant rate of change and do not depend on any other inputs.



Predictive modeling typically involves choices from among a set of models. These might be different types of models. These might be different complexities of models of the same type. A common pitfall is to overfit the data; that is, to use too complex a model. An overly complex model might be too sensitive to peculiarities in the sample data set and not generalize well to new data. However, using too simple a model can lead to *underfitting*, where true features are disregarded.

1.3 Chapter Summary

The principal objective of predictive modeling is to predict the outcome on new cases. Some of the business applications include target marketing, attrition prediction, credit scoring, and fraud detection. One challenge in building a predictive model is that the data usually was not collected for purposes of data analysis. Therefore, it is usually massive, dynamic, and dirty. For example, the data usually has a large number of input variables. This limits the ability to explore and model the relationships among the variables. Thus, detecting interactions and nonlinearities becomes a cumbersome problem.

When the target is rare, a widespread strategy is to build a model on a sample that disproportionately over-represents the events. The results will be biased, but they can be easily corrected to represent the population.

A common pitfall in building a predictive model is to overfit the data. An overfitted model will be too sensitive to the nuances in the data and will not generalize well to new data. However, a model that underfits the data will systematically miss the true features in the data.

Chapter 2 Fitting the Model

2.1	The Model.....	2-3
2.2	Adjustments for Oversampling	2-21
2.3	Chapter Summary.....	2-30
2.4	Solutions	2-31
	Solutions to Exercises	2-31
	Solutions to Student Activities (Polls/Quizzes)	2-31

2.1 The Model

Functional Form

$$\text{logit}(p_i) = \beta_0 + \beta_1 x_{1i} + \cdots + \beta_k x_{ki}$$

posterior probability
parameter
input

3

The data set consists of $i=1,2,\dots,n$ cases. Each case belongs to one of two classes. A binary indicator variable represents the class label for each case

$$y_i = \begin{cases} 1 & \text{target event for case } i \\ 0 & \text{no target event for case } i \end{cases}$$

Associated with each case is k -vector of input variables

$$\mathbf{x}_i = (x_{1i}, x_{2i}, \dots, x_{ki})$$

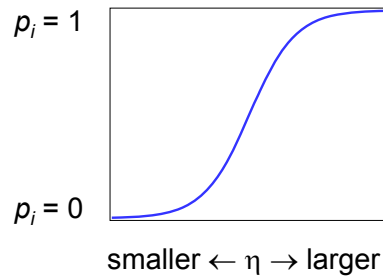
The posterior probability of the target event given the inputs is

$$p_i = E(y_i | \mathbf{x}_i) = \Pr(y_i = 1 | \mathbf{x}_i)$$

The standard logistic regression model assumes that the logit of the posterior probability is a linear combination of the input variables. The parameters, β_0, \dots, β_k , are unknown constants that must be estimated from the data.

The Logit Link Function

$$\text{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \eta \Leftrightarrow p_i = \frac{1}{1+e^{-\eta}}$$



4

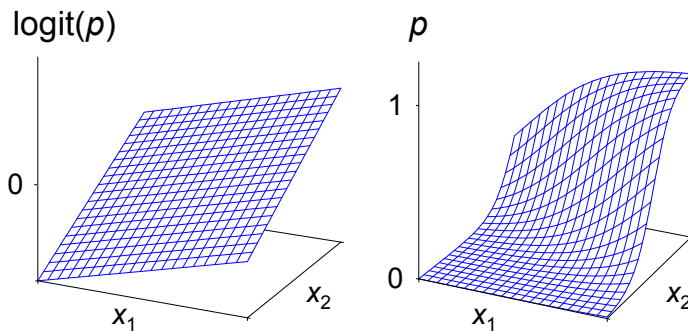
A linear combination can take any value. Probability must be between zero and one. The logit transformation (which is the log of the odds) is a device for constraining the posterior probability to be between zero and one. The logit function transforms the probability scale to the real line $(-\infty, +\infty)$. Therefore, modeling the logit with a linear combination gives estimated probabilities that are constrained to be between zero and one.

Logistic regression is a special case of the *generalized linear model*

$$g(E(y | \mathbf{x})) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

where the expected value of the target is linked to the linear predictor by the function $g(\cdot)$. The link function depends on the scale of the target.

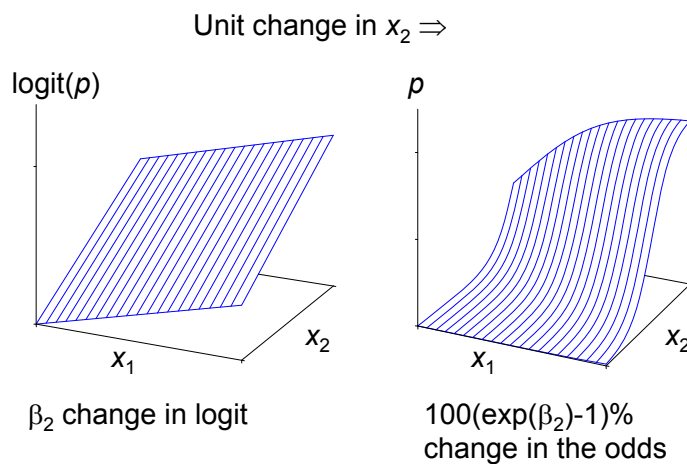
The Fitted Surface



5

The graph of a linear combination on the logit scale is a (hyper)plane. On the probability scale it becomes a sigmoidal surface. Different parameter values give different surfaces with different slopes and different orientations. The nonlinearity is solely due to the constrained scale of the target. The nonlinearity only appears when the fitted values are close to the limits of their sensible range ($>.8$ or $<.2$).

Interpretation

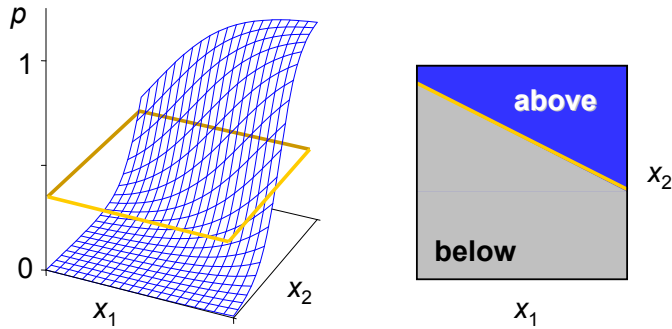


6

A linear-additive model is particularly easy to interpret: each input variable affects the logit linearly. The coefficients are the slopes. Exponentiating each parameter estimate gives the odds ratios, which compares the odds of the event in one group to the odds of the event in another group. For example, the odds ratio for a binary input variable (X) would compare the odds of the event when $X=1$ to the odds of the event when $X=0$. The odds ratio represents the multiplicative effect of each input variable. Moreover, the effect of each input variable does not depend on the values of the other inputs (additivity).

However, this simple interpretation depends on the model being correctly specified. In predictive modeling, you should not presume that the true posterior probability has such a simple form. Think of the model as an approximating (hyper)plane. Consequently, you can determine the extent that the inputs are important to the approximating plane.

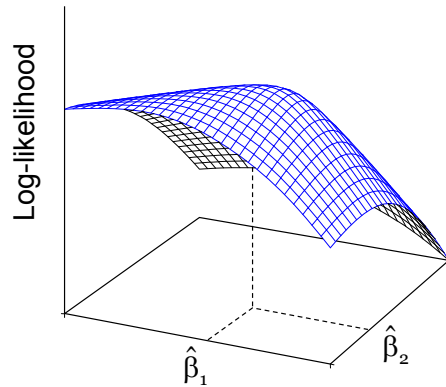
Logistic Discrimination



7

In supervised classification, the ultimate use of logistic regression is to allocate cases to classes. This is more correctly termed logistic discrimination (McClachlan 1989). An allocation rule is merely an assignment of a cutoff probability, where cases above the cutoff are allocated to class 1 and cases below the cutoff are allocated to class 0. The standard logistic discrimination model separates the classes by a linear surface ((hyper)plane). The decision boundary is always linear. Determining the best cutoff is a fundamental concern in logistic discrimination.

Maximum Likelihood Estimation



8

The method of maximum likelihood (ML) is usually used to estimate the unknown parameters in the logistic regression model. The likelihood function is the joint probability density function of the data treated as a function of the parameters. The maximum likelihood estimates are the values of the parameters that maximize the probability of obtaining the sample data.

If you assume that the y_i independently have Bernoulli distributions with probability p_i (which is a function of the parameters), then the log of the likelihood is given by

$$\sum_{i=1}^n (y_i \ln(p_i) + (1 - y_i) \ln(1 - p_i)) = \sum_{i|y=1}^{n_1} \ln(p_i) + \sum_{i|y=0}^{n_0} \ln(1 - p_i)$$

where n_0 and n_1 are the numbers of class 0 and class 1, respectively. The form of the log-likelihood shows the intuitively appealing result that the ML estimates be chosen so that p_i is large when $y_i=1$ and small when $y_i=0$.

In ML estimation the combination of parameter values that maximize the likelihood (or log-likelihood) are pursued. There is, in general, no closed form analytical solution for the ML estimates as there is for linear regression on a normally distributed response. They must be determined using an iterative optimization algorithm. Consequently, logistic regression is considerably more computationally expensive than linear regression.

Software for ML estimation of the logistic model is commonplace. Many SAS procedures can be used; most notable are the LOGISTIC, GENMOD, CATMOD, and DMREG procedures (SAS Enterprise Miner).

2.01 Multiple Choice Poll

The odds ratio for a \$1000 increase in income is 1.074.
This means that for every \$1000 increase in income

- a. the probability of the event increases 7.4%.
- b. the logit increases 7.4%
- c. the odds of the event increases 7.4%
- d. the log of the odds of the event increases 7.4%



Introduction to the LOGISTIC Procedure

The LOGISTIC procedure fits a binary logistic regression model. The seven input variables included in the model were selected arbitrarily. The DES (short for descending) option is used to reverse the sorting order for the levels of the response variable **Ins**. The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement. The PARAM option in the CLASS statement specifies the parameterization method for the classification variable or variables and the REF option specifies the reference level. In this example, the parameterization method is reference cell coding and the reference level is S.

The STB option displays the standardized estimates for the parameters for the continuous input variables. The UNITS statement enables you to obtain an odds ratio estimate for a specified change in an input variable. In this example, the UNITS statement enables you to estimate the change in odds for a 1000-unit change in **DDABal** and **DepAmt**.

```
proc logistic data=develop des;
  class res (param=ref ref='S');
  model ins = dda ddabal dep depamt
             cashbk checks res
             / stb;
  units ddabal=1000 depamt=1000;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.DEVELOP
Response Variable	Ins
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	32264
Number of Observations Used	32264

Response Profile

Ordered Value	Ins	Total Frequency
1	1	11175
2	0	21089

Probability modeled is Ins=1.

The results consist of a number of tables. The Response Profile table shows the target variable values listed according to their ordered values. By default, the target-variable values are ordered alphanumerically and PROC LOGISTIC always models the probability of ordered value 1. The DES option reverses the order so that PROC LOGISTIC models the probability that **Ins**=1. Another way to force the LOGISTIC procedure to model the probability of a particular level of the target variable is with **EVENT=** syntax in the MODEL statement.

Class Level Information			
Class	Value	Design Variables	
Res	R	1	0
	S	0	0
	U	0	1

The Class Level Information table shows the **Res** variable was dummy coded into two design variables using reference cell coding and the level S as the reference level.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	41633.206	39565.329
SC	41641.587	39640.765
-2 Log L	41631.206	39547.329

The Model Fit Statistics table contains the Akaike information criteria (AIC) and the Schwarz criterion (SC). These are goodness-of-fit measures you can use to compare one model to another.

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	2083.8761	8	<.0001
Score	1843.2388	8	<.0001
Wald	1768.0578	8	<.0001

The likelihood ratio, Wald, and Score tests all test the null hypothesis that all regression coefficients of the model other than the intercept are 0.

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
DDA	1	633.6092	<.0001
DDABal	1	448.5968	<.0001
Dep	1	42.8222	<.0001
DepAmt	1	30.6227	<.0001
CashBk	1	24.1615	<.0001
Checks	1	1.6168	0.2035
Res	2	2.7655	0.2509

The Type 3 Analysis of Effects table shows which input variables are significant controlling for all of the other input variables in the model.

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Standardized Estimate
Intercept	1	0.1519	0.0304	24.8969	<.0001	
DDA	1	-0.9699	0.0385	633.6092	<.0001	-0.2074
DDABal	1	0.000072	3.39E-6	448.5968	<.0001	0.2883
Dep	1	-0.0714	0.0109	42.8222	<.0001	-0.0678
DepAmt	1	0.000018	3.225E-6	30.6227	<.0001	0.0660
CashBk	1	-0.5629	0.1145	24.1615	<.0001	-0.0408
Checks	1	-0.00402	0.00317	1.6168	0.2035	-0.0114
Res R	1	-0.0467	0.0316	2.1907	0.1388	
Res U	1	-0.0379	0.0280	1.8375	0.1752	

The parameter estimates measure the rate of change in the logit (log odds) corresponding to a one-unit change in input variable, adjusted for the effects of the other inputs. The parameter estimates are difficult to compare because they depend on the units in which the variables are measured. The standardized estimates convert them to standard deviation units. The absolute value of the standardized estimates can be used to give an approximate ranking of the relative importance of the input variables on the fitted logistic model. The variable **Res** has no standardized estimate because it is a class variable.

Odds Ratio Estimates				
Effect		Point Estimate	95% Wald Confidence Limits	
DDA		0.379	0.352	0.409
DDABal		1.000	1.000	1.000
Dep		0.931	0.911	0.951
DepAmt		1.000	1.000	1.000
CashBk		0.570	0.455	0.713
Checks		0.996	0.990	1.002
Res	R vs S	0.954	0.897	1.015
Res	U vs S	0.963	0.911	1.017

The odds ratio measures the effect of the input variable on the target adjusted for the effect of the other input variables. For example, the odds of acquiring an insurance product for DDA (checking account) customers is .379 times the odds for non-DDA customers. Equivalently, the odds of acquiring an insurance product is 1/.379 or 2.64 times more likely for non-DDA customers compared to DDA customers. By default, PROC LOGISTIC reports the 95% Wald confidence interval.

Association of Predicted Probabilities and Observed Responses				
Percent Concordant	66.4	Somers' D	0.350	
Percent Discordant	31.4	Gamma	0.358	
Percent Tied	2.2	Tau-a	0.158	
Pairs	235669575	c	0.675	

The Association of Predicted Probabilities and Observed Responses table lists several measures that assess the predictive ability of the model. For all pairs of observations with different values of the target variable, a pair is concordant if the observation with the outcome has a higher predicted outcome probability (based on the model) than the observation without the outcome. A pair is discordant if the observation with the outcome has a lower predicted outcome probability than the observation without the outcome.

The four rank correlation indexes (Somer's D, Gamma, Tau-a, and c) are computed from the numbers of concordant and discordant pairs of observations. In general, a model with higher values for these indexes (the maximum value is 1) has better predictive ability than a model with lower values for these indexes.

Adjusted Odds Ratios		
Effect	Unit	Estimate
DDABa1	1000.0	1.074
DepAmt	1000.0	1.018

For continuous variables, it may be useful to convert the odds ratio to a percentage increase or decrease in odds. For example, the odds ratio for a 1000-unit change in **DDABa1** is 1.074. Consequently, the odds of acquiring the insurance product increases 7.4% (calculated as $100(1.074-1)$) for every thousand-dollar increase in the checking balance, assuming that the other variables do not change.

Scoring New Cases

$$\mathbf{x} = (1.1, 3.0)$$

$$\text{logit}(\hat{p}) = -1.6 + .14x_1 - .50x_2$$

$$\hat{p} = .05$$

13

The overriding purpose of predictive modeling is to score new cases. Predictions can be made by simply plugging in the new values of the inputs.



Scoring New Cases

The SCORE Statement

The LOGISTIC procedure can score data sets, without using the OUTPUT statement, beginning in SAS[®]9. The SCORE statement in the LOGISTIC procedure applies the model to a new data set. The DATA= option names the data set to be scored, and the OUT= option names the resulting scored data set. The predicted probability that **ins** is 1 is named **P_1**.

```
proc logistic data=develop des;
    model ins=dda ddabal dep depamt cashbk checks;
    score data = pmlr.new out=scored;
run;

proc print data=scored(obs=20);
    var P_1 dda ddabal dep depamt cashbk checks;
run;
```

Obs	P_1	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.27476	1	56.29	2	955.51	0	1
2	0.32343	1	3292.17	2	961.60	0	1
3	0.30275	1	1723.86	2	2108.65	0	2
4	0.53144	0	0.00	0	0.00	0	0
5	0.27180	1	67.91	2	519.24	0	3
6	0.32482	1	2554.58	1	501.36	0	2
7	0.27205	1	0.00	2	2883.08	0	12
8	0.30558	1	2641.33	3	4521.61	0	8
9	0.53144	0	0.00	0	0.00	0	0
10	0.28679	1	52.22	1	75.59	0	0
11	0.37131	1	6163.29	2	2603.56	0	7
12	0.18001	1	431.12	2	568.43	1	2
13	0.53144	0	0.00	0	0.00	0	0
14	0.20217	1	112.82	8	2688.75	0	3
15	0.31330	1	1146.61	3	11224.20	0	2
16	0.53144	0	0.00	0	0.00	0	0
17	0.27549	1	1241.38	3	3538.14	0	15
18	0.30509	1	298.23	0	0.00	0	0
19	0.28299	1	367.04	2	4242.22	0	11
20	0.26508	1	1229.47	4	3514.57	0	10

OUTEST= Option and the SCORE Procedure

The LOGISTIC procedure outputs the final parameter estimates to a data set using the OUTEST= option.

```
proc logistic data=develop des outest=betas1;
  model ins=dda ddabal dep depamt cashbk checks;
run;

proc print data=betas1;
run;
```

Obs	_LINK_	_TYPE_	_STATUS_	_NAME_	Intercept	DDA	DDABa1
1	LOGIT	PARMS	0 Converged	Ins	0.12592	-0.97052	.000071819
Obs	Dep	DepAmt	CashBk	Checks	_LNLIKE_		
1	-0.071531	.000017829	-0.56175	-.003999236	-19775.05		

The output data set contains one observation and a variable for each parameter estimate. The estimates are named corresponding to their input variable.

The SCORE procedure multiplies values from two SAS data sets, one containing coefficients (SCORE=) and the other containing the data to be scored (DATA=). Typically, the data set to be scored would not have a target variable. The OUT= option specifies the name of the scored data set created by PROC SCORE. The TYPE=PARMS option is required for scoring regression models.

```
proc score data=pmlr.new
  out=scored
  score=betas1
  type=parms;
  var dda ddabal dep depamt cashbk checks;
run;
```

The linear combination produced by PROC SCORE (the variable **Ins**) estimates the logit, not the posterior probability. The logistic function (inverse of the logit) needs to be applied to compute the posterior probability.

```
data scored;
  set scored;
  p=1/(1+exp(-ins));
run;

proc print data=scored(obs=20);
  var p ins dda ddabal dep depamt cashbk checks;
run;
```

Obs	p	Ins	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.27476	-0.97058	1	56.29	2	955.51	0	1
2	0.32343	-0.73807	1	3292.17	2	961.60	0	1
3	0.30275	-0.83426	1	1723.86	2	2108.65	0	2
4	0.53144	0.12592	0	0.00	0	0.00	0	0
5	0.27180	-0.98552	1	67.91	2	519.24	0	3
6	0.32482	-0.73172	1	2554.58	1	501.36	0	2
7	0.27205	-0.98425	1	0.00	2	2883.08	0	12
8	0.30558	-0.82087	1	2641.33	3	4521.61	0	8
9	0.53144	0.12592	0	0.00	0	0.00	0	0
10	0.28679	-0.91103	1	52.22	1	75.59	0	0
11	0.37131	-0.52659	1	6163.29	2	2603.56	0	7
12	0.18001	-1.51631	1	431.12	2	568.43	1	2
13	0.53144	0.12592	0	0.00	0	0.00	0	0
14	0.20217	-1.37280	1	112.82	8	2688.75	0	3
15	0.31330	-0.78473	1	1146.61	3	11224.20	0	2
16	0.53144	0.12592	0	0.00	0	0.00	0	0
17	0.27549	-0.96694	1	1241.38	3	3538.14	0	15
18	0.30509	-0.82318	1	298.23	0	0.00	0	0
19	0.28299	-0.92966	1	367.04	2	4242.22	0	11
20	0.26508	-1.01975	1	1229.47	4	3514.57	0	10

Data can also be scored directly in PROC LOGISTIC using the OUTPUT statement. This has several disadvantages over using PROC SCORE: it does not scale well with large data sets, it requires a target variable (or some proxy), and the adjustments for oversampling, discussed in the next section, are not automatically applied.

The Output Delivery System and Automatic Score Code Generation (Self-Study)

The Output Delivery System is an easy way to save results that are typically displayed as output into a data set. The data set **betas2** contains the output listed under the heading “Analysis of Maximum Likelihood Estimates” in the PROC LOGISTIC output.

```
ods output parameterEstimates = betas2;
proc logistic data=develop des;
    model ins=dda ddabal dep depamt cashbk checks;
run;

proc print data=betas2;
    var variable estimate;
run;
```

	Obs	Variable	Estimate
	1	Intercept	0.1259
	2	DDA	-0.9705
	3	DDABal	0.000072
	4	Dep	-0.0715
	5	DepAmt	0.000018
	6	CashBk	-0.5618
	7	Checks	-0.00400

The code to generate score code from the **betas2** data set follows. This code creates a file that contains the code to create the linear predictor. The macro variable **target** is used in the name and label of the predicted probability variable. The code handles categorical inputs as well, even though there are none in this example. Categorical inputs should be numerically coded using reference cell coding. The code essentially automates the tedious task of cutting and pasting parameter estimates from logistic regression results into DATA step code.

```

%let target=INS;

filename scorecd 'c:\temp\logistic score code.sas';

data _null_;
    attrib PREDNAME length=$32
           TARGNAME length=$32
           LastParm length=$32
           ;
    file scorecd;
    set betas2 end=last;
    retain TARGNAME PREDNAME LastParm ' ';
    if (Variable="Intercept") then do;
        TARGNAME=compress("&target");
        PREDNAME="P_"||compress(TARGNAME);
        put "*****";
        put "**** begin scoring code for Logistic Regression;";
        put "*****";

        put "length " PREDNAME "8;";
        put "label " PREDNAME " = 'Predicted: " TARGNAME "+(-1) '";";

        put "**** accumulate XBETA;";
        put "XBETA = " Estimate best20. " ";";
    end;
    else if (ClassVal0=' ') then do;
        put "XBETA = XBETA + (" Estimate best20. ") * " Variable ";";
    end;
    else if (compress(Variable)=compress(LastParm)) then do;
        put "else if (" Variable "='" ClassVal0 "+(-1) "') then do;";
        put "    XBETA = XBETA + (" Estimate best20. ");";
        put "end;";
    end;
    else do;
        put "if (" Variable "='" ClassVal0 "+(-1) "') then do;";
        put "    XBETA = XBETA + (" Estimate best20. ");";
        put "end;";
    end;
    LastParm=Variable;
    if last then do;
        put PREDNAME " = 1/(1+exp(-XBETA));";
    end;
run;

```

The output of the code generator is DATA step code to score a new data set.

```
*****;
*** begin scoring code for Logistic Regression;
*****;
length P_INS 8;
label P_INS = 'Predicted: INS';
*** accumulate XBETA;
XBETA = 0.12592268473366;
XBETA = XBETA + ( -0.97052008343956) * DDA ;
XBETA = XBETA + ( 0.00007181904404) * DDABal ;
XBETA = XBETA + ( -0.07153101800749) * Dep ;
XBETA = XBETA + ( 0.0000178287498) * DepAmt ;
XBETA = XBETA + ( -0.56175142056463) * CashBk ;
XBETA = XBETA + ( -0.00399923596541) * Checks ;
P_INS = 1/(1+exp(-XBETA)) ;
```

The %INCLUDE statement brings a SAS programming statement, data lines, or both, into a current SAS program. Using the SOURCE2 option after a slash treats the code as if it was pasted into the program editor—the text appears in the log. The %INCLUDE statement can use an explicit file name (C:\Temp\logistic score code.sas) or a fileref (scorecd).

```
data scored;
    set pmlr.new;
    %include "c:\temp\logistic score code.sas" /source2;
run;
```

The above code generates the same results as the following code:

```
data scored;
    set pmlr.new;
    %include scorecd /source2;
run;
```

The results are equivalent to the earlier two techniques.

```
proc print data=scored(obs=20);
    var p_ins xbeta dda ddabal dep depamt cashbk checks;
run;
```

In order to use the model, you must create scoring code. This method automatically creates a DATA step that you could use to score new cases, without running the SCORE procedure or having to cut and paste parameter estimates into code by hand.

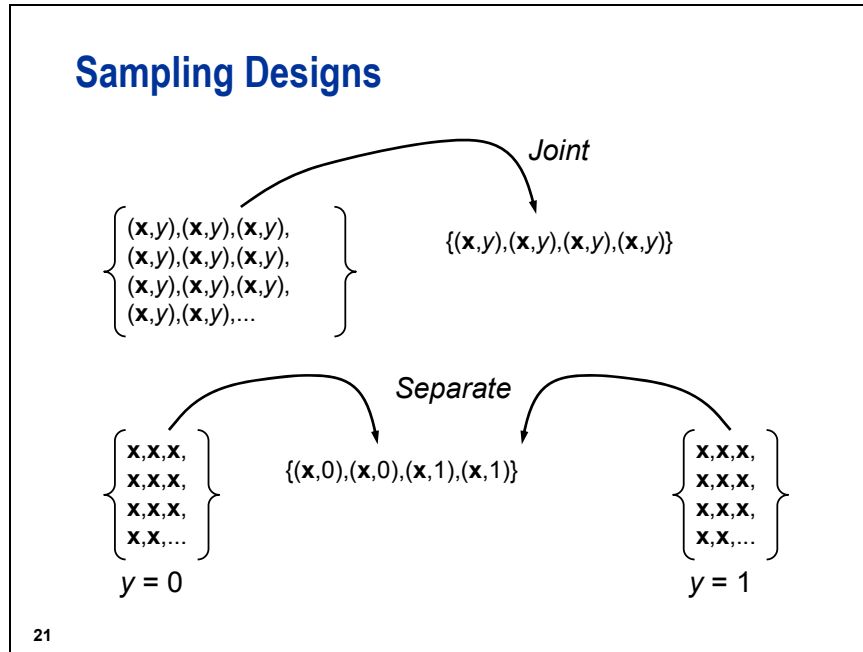
Obs	P_INS	XBETA	DDA	DDABa1	Dep	DepAmt	Cash Bk	Checks
1	0.27476	-0.97058	1	56.29	2	955.51	0	1
2	0.32343	-0.73807	1	3292.17	2	961.60	0	1
3	0.30275	-0.83426	1	1723.86	2	2108.65	0	2
4	0.53144	0.12592	0	0.00	0	0.00	0	0
5	0.27180	-0.98552	1	67.91	2	519.24	0	3
6	0.32482	-0.73172	1	2554.58	1	501.36	0	2
7	0.27205	-0.98425	1	0.00	2	2883.08	0	12
8	0.30558	-0.82087	1	2641.33	3	4521.61	0	8
9	0.53144	0.12592	0	0.00	0	0.00	0	0
10	0.28679	-0.91103	1	52.22	1	75.59	0	0
11	0.37131	-0.52659	1	6163.29	2	2603.56	0	7
12	0.18001	-1.51631	1	431.12	2	568.43	1	2
13	0.53144	0.12592	0	0.00	0	0.00	0	0
14	0.20217	-1.37280	1	112.82	8	2688.75	0	3
15	0.31330	-0.78473	1	1146.61	3	11224.20	0	2
16	0.53144	0.12592	0	0.00	0	0.00	0	0
17	0.27549	-0.96694	1	1241.38	3	3538.14	0	15
18	0.30509	-0.82318	1	298.23	0	0.00	0	0
19	0.28299	-0.92966	1	367.04	2	4242.22	0	11
20	0.26508	-1.01975	1	1229.47	4	3514.57	0	10

2.02 Multiple Choice Poll

How many individuals and responders in the sample have a predicted response rate greater than 0.025?

- 19323 individuals and 4833 responders
- 8161 individuals and 2579 responders
- 2240 individuals and 830 responders
- 1256 individuals and 547 responders

2.2 Adjustments for Oversampling

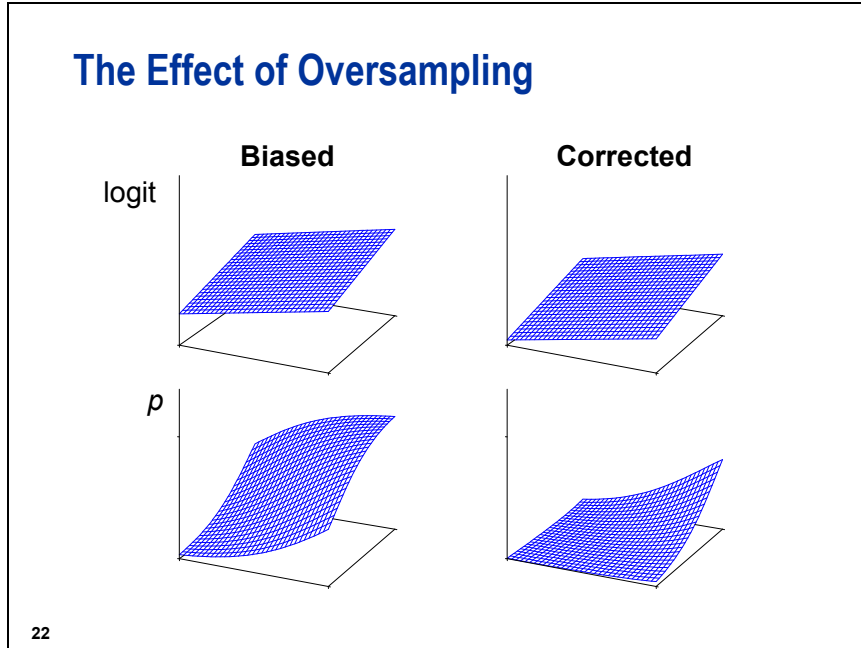


In *joint* (mixture) sampling, the input-target pairs are randomly selected from their joint distribution. In *separate* sampling, the inputs are randomly selected from their distributions within each target class.

Separate sampling is standard practice in supervised classification. When the target event is rare, it is common to oversample the rare event, that is, take a disproportionately large number of event cases. Oversampling rare events is generally believed to lead to better predictions (Scott and Wild 1986). Separate sampling is also known as

- case-control sampling
- choice-based sampling
- stratified sampling on the target, not necessarily taken with proportional allocation
- biased sampling
- y-conditional sampling
- outcome-dependent sampling
- oversampling.

The *priors*, π_0 and π_1 , represent the population proportions of class 0 and 1, respectively. The proportions of the target classes in the sample are denoted ρ_0 and ρ_1 . In separate sampling (nonproportional) $\pi_0 \neq \rho_0$ and $\pi_1 \neq \rho_1$. The adjustments for oversampling require the priors be known a priori.



The maximum likelihood estimates were derived under the assumption that y_i have independent Bernoulli distributions. This assumption is appropriate for joint sampling but not for separate sampling. However, the effects of violating this assumption can be easily corrected. In logistic regression, only the estimate of the intercept, β_0 , is affected by using Bernoulli ML on data from a separate sampling design (Prentice and Pike 1979). If the standard model

$$\text{logit}(p_i) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

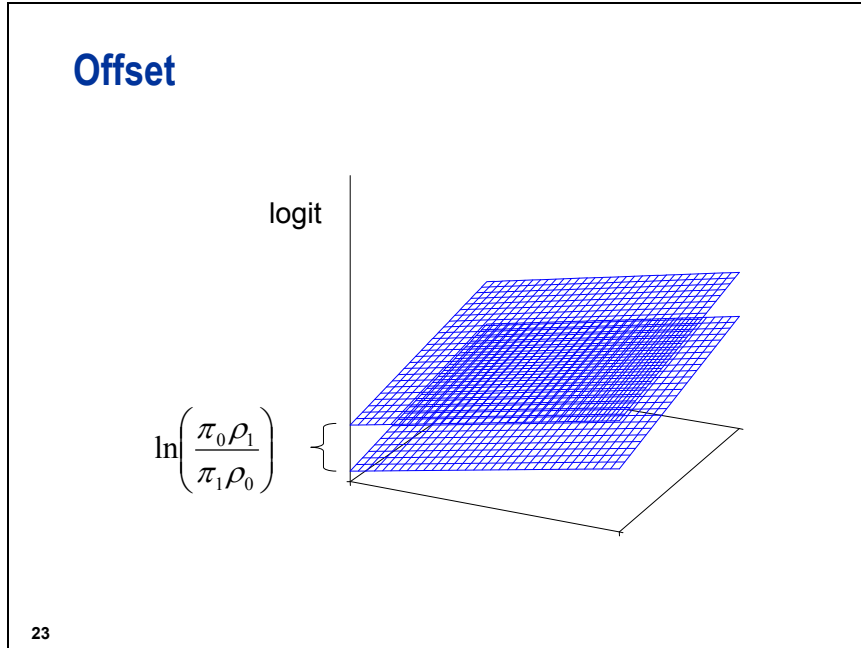
is appropriate for joint sampling, then ML estimates of the parameters under separate sampling can be determined by fitting the *pseudo model* (Scott and Wild 1986, 1997)

$$\text{logit}(p_i^*) = \ln\left(\frac{\rho_1 \pi_0}{\rho_0 \pi_1}\right) + \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

where p^* is the posterior probability corresponding to the biased sample. Consequently, the effect of oversampling is to shift the logits by a constant amount – the *offset*

$$\ln\left(\frac{\rho_1 \pi_0}{\rho_0 \pi_1}\right)$$

When rare events have been oversampled $\pi_0 > \rho_0$ and $\pi_1 < \rho_1$, the offset is positive; that is, the logit is too large. This vertical shift of the logit affects the posterior probability in a corresponding fashion.



The pseudo model can be fitted directly by incorporating the offset into the model. Alternatively, the offset could be applied *after* the standard model is fitted. Subtracting the offset from the predicted values and solving for the posterior probability gives

$$\hat{p}_i = \frac{\hat{p}_i^* \rho_0 \pi_1}{(1 - \hat{p}_i^*) \rho_1 \pi_0 + \hat{p}_i^* \rho_0 \pi_1}$$

where \hat{p}_i^* is the unadjusted estimate of the posterior probability. Both approaches give identical results.

For both types of adjustments, the population priors, π_0 and π_1 , need to be known a priori while the sample priors, ρ_0 and ρ_1 , can be estimated from the data.

Because only the intercept is affected, the adjustments may not be necessary. If the goal of the analysis is to understand the relationships between the inputs and the target, or to rank order the population, then the adjustment is not critical. If the predicted **probabilities** are important, and not just necessary for rank ordering or classification, then the correction for oversampling is necessary.



Correcting for Oversampling

Separate sampling was used to create the **INS** data set. The proportion of the target event in the population was .02, not .346 as appears in the sample. The %LET statement defines the macro variable **p11** for the population prior for class 1 (**Ins=1**).

```
%let p11=.02;          /* supply the prior for class 1 */
```

The SQL procedure can be used to create macro variables as well. The following code is equivalent to **%let rho1 = 0.346361;**

```
proc SQL noprint;
  select mean(INS) into :rho1 from develop;
quit;
```

The SCORE statement in the LOGISTIC procedure will correct predicted probabilities back to the population scale. The option to do this is **PRIOREVENT=**.

```
proc logistic data=develop des;
  model ins=dda ddabal dep depamt cashbk checks;
  score data = pmlr.new out=scored priorevent=&p11;
run;

proc print data=scored(obs=20);
  var P_1 dda ddabal dep depamt cashbk checks;
run;
```

Obs	P_1	DDA	DDABa1	Dep	DepAmt	Cash Bk	Checks
1	0.014381	1	56.29	2	955.51	0	1
2	0.018078	1	3292.17	2	961.60	0	1
3	0.016447	1	1723.86	2	2108.65	0	2
4	0.041854	0	0.00	0	0.00	0	0
5	0.014171	1	67.91	2	519.24	0	3
6	0.018191	1	2554.58	1	501.36	0	2
7	0.014189	1	0.00	2	2883.08	0	12
8	0.016665	1	2641.33	3	4521.61	0	8
9	0.041854	0	0.00	0	0.00	0	0
10	0.015250	1	52.22	1	75.59	0	0
11	0.022241	1	6163.29	2	2603.56	0	7
12	0.008384	1	431.12	2	568.43	1	2
13	0.041854	0	0.00	0	0.00	0	0
14	0.009665	1	112.82	8	2688.75	0	3
15	0.017268	1	1146.61	3	11224.20	0	2
16	0.041854	0	0.00	0	0.00	0	0
17	0.014433	1	1241.38	3	3538.14	0	15
18	0.016628	1	298.23	0	0.00	0	0
19	0.014973	1	367.04	2	4242.22	0	11
20	0.013701	1	1229.47	4	3514.57	0	10

Correcting the Intercept in the Automatic Score Code Generator (Self-Study)

Because the correction for oversampling is simply an adjustment to the intercept, you could build that correction into the score code generator. The following code is similar to the initial score code generator, but the intercept term is corrected by the amount of the offset.

```
ods output parameterEstimates=betas2;
proc logistic data=develop des;
  model ins=dda ddabal dep depamt cashbk checks;
run;

%let target=INS;
filename scorecd 'c:\temp\logistic score code.sas';

data _null_;
  attrib PREDNAME length=$32
         TARGNAME length=$32
         LastParm length=$32
         ;
  file scorecd;
  set betas2 end=last;
  retain TARGNAME PREDNAME LastParm ' ';
  if (Variable="Intercept") then do;
    TARGNAME=compress("&target");
    PREDNAME="P_"||compress(TARGNAME);
    put "*****";
    put "**** begin scoring code for Logistic Regression";
    put "*****";

    put "length " PREDNAME "8";
    put "label " PREDNAME "= 'Predicted: " TARGNAME +(-1) "'";

    put "**** accumulate XBETA";
    Estimate + (-log(((1-&pi1)*&rho1)/(&pi1*(1-&rho1))));
    put "XBETA = " Estimate best20. " ";
  end;
  else if (ClassVal0=' ') then do;
    put "XBETA = XBETA + (" Estimate best20. ") * " Variable " ";
  end;
  else if (compress(Variable)=compress(LastParm)) then do;
    put "else if (" Variable "=" ClassVal0 +(-1) "' ) then do;";
    put "    XBETA = XBETA + (" Estimate best20. ") ";
    put "end;";
  end;
  else do;
    put "if (" Variable "=" ClassVal0 +(-1) "' ) then do;";
    put "    XBETA = XBETA + (" Estimate best20. ") ";
    put "end;";
  end;
  LastParm=Variable;
  if last then do;
    put PREDNAME "= 1/(1+exp(-XBETA)) ";
  end;
run;
```

The score code follows. Note the change in the intercept term, which was 0.12592268473366 before.

```
*****;
*** begin scoring code for Logistic Regression;
*****;
length P_INS 8;
label P_INS = 'Predicted: INS';
*** accumulate XBETA;
XBETA = -3.13082398583352;
XBETA = XBETA + ( -0.97052008343956) * DDA ;
XBETA = XBETA + ( 0.00007181904404) * DDABal ;
XBETA = XBETA + ( -0.07153101800749) * Dep ;
XBETA = XBETA + ( 0.0000178287498) * DepAmt ;
XBETA = XBETA + ( -0.56175142056463) * CashBk ;
XBETA = XBETA + ( -0.00399923596541) * Checks ;
P_INS = 1/(1+exp(-XBETA));
```

Again, you can use the %INCLUDE statement to score data with this code.

```
data scored;
  set pmlr.new;
  %include scorecd /source2;
run;

proc print data=scored(obs=20);
  var p_ins dda ddabal dep depamt cashbk checks;
run;
```

The output is the same as the results of the PRIOREVENT=&pi1 correction, above.

Obs	P_INS	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.014381	1	56.29	2	955.51	0	1
2	0.018078	1	3292.17	2	961.60	0	1
3	0.016447	1	1723.86	2	2108.65	0	2
4	0.041854	0	0.00	0	0.00	0	0
5	0.014171	1	67.91	2	519.24	0	3
6	0.018191	1	2554.58	1	501.36	0	2
7	0.014189	1	0.00	2	2883.08	0	12
8	0.016665	1	2641.33	3	4521.61	0	8
9	0.041854	0	0.00	0	0.00	0	0
10	0.015250	1	52.22	1	75.59	0	0
11	0.022241	1	6163.29	2	2603.56	0	7
12	0.008384	1	431.12	2	568.43	1	2
13	0.041854	0	0.00	0	0.00	0	0
14	0.009665	1	112.82	8	2688.75	0	3
15	0.017268	1	1146.61	3	11224.20	0	2
16	0.041854	0	0.00	0	0.00	0	0
17	0.014433	1	1241.38	3	3538.14	0	15
18	0.016628	1	298.23	0	0.00	0	0
19	0.014973	1	367.04	2	4242.22	0	11
20	0.013701	1	1229.47	4	3514.57	0	10

Other Options (Self-Study)

There are ways to correct for oversampling without the SCORE statement or the code generator.

The DATA step creates a variable to be used as an offset. The probabilities for class 0 in the population and sample are computed as one minus the probabilities in class 1.

```
data develop;
  set develop;
  off=log(((1-&pi1)*&rho1)/(&pi1*(1-&rho1)));
run;
```

The LOGISTIC procedure fits a model with the offset variable and outputs the final parameter estimates. The OFFSET= option in the MODEL statement names the offset variable. The only difference between the parameter estimates with and without the offset variable is the intercept term.

```
proc logistic data=develop des outest=betas2;
  model ins=dda ddabal dep depamt cashbk checks
    / offset=off;
run;
```

Partial Output

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	-3.1308	0.0260	14518.0451	<.0001
DDA	1	-0.9705	0.0385	634.4513	<.0001
DDABal	1	0.000072	3.39E-6	448.8938	<.0001
Dep	1	-0.0715	0.0109	42.9169	<.0001
DepAmt	1	0.000018	3.223E-6	30.5992	<.0001
CashBk	1	-0.5617	0.1145	24.0544	<.0001
Checks	1	-0.00400	0.00317	1.5964	0.2064
off	1	1.0000	0	.	.

The list of parameter estimates contains a new entry for the variable **off**, which has a fixed value of one. The probabilities computed from this model have been adjusted down because the population probability is much lower than the sample probability (.02 versus .346).

The SCORE procedure uses the final parameter estimates from the logistic model with the offset variable to score new data.

```
proc score data=pmlr.new out=scored score=betas2
  type=parms;
  var dda ddabal dep depamt cashbk checks;
run;

data scored;
  set scored;
  p=1/(1+exp(-ins));
run;

proc print data=scored(obs=20);
  var p ins dda ddabal dep depamt cashbk checks;
run;
```

Obs	p	Ins	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.014381	-4.22733	1	56.29	2	955.51	0	1
2	0.018078	-3.99482	1	3292.17	2	961.60	0	1
3	0.016447	-4.09100	1	1723.86	2	2108.65	0	2
4	0.041854	-3.13082	0	0.00	0	0.00	0	0
5	0.014171	-4.24227	1	67.91	2	519.24	0	3
6	0.018191	-3.98847	1	2554.58	1	501.36	0	2
7	0.014189	-4.24100	1	0.00	2	2883.08	0	12
8	0.016665	-4.07762	1	2641.33	3	4521.61	0	8
9	0.041854	-3.13082	0	0.00	0	0.00	0	0
10	0.015250	-4.16778	1	52.22	1	75.59	0	0
11	0.022241	-3.78334	1	6163.29	2	2603.56	0	7
12	0.008384	-4.77306	1	431.12	2	568.43	1	2
13	0.041854	-3.13082	0	0.00	0	0.00	0	0
14	0.009665	-4.62955	1	112.82	8	2688.75	0	3
15	0.017268	-4.04148	1	1146.61	3	11224.20	0	2
16	0.041854	-3.13082	0	0.00	0	0.00	0	0
17	0.014433	-4.22369	1	1241.38	3	3538.14	0	15
18	0.016628	-4.07993	1	298.23	0	0.00	0	0
19	0.014973	-4.18641	1	367.04	2	4242.22	0	11
20	0.013701	-4.27650	1	1229.47	4	3514.57	0	10

The OFFSET= option is less efficient than fitting an unadjusted model. When the OFFSET= option is used, PROC LOGISTIC uses CPU time roughly equivalent to two logistic regressions. A more efficient way of adjusting the posterior probabilities for the offset is to fit the model without the offset and adjust the fitted posterior probabilities afterwards in a DATA step. The two approaches are statistically equivalent.

```
proc logistic data=develop des outest=betas3;
  model ins=dda ddabal dep depamt cashbk checks;
run;

proc score data=pmlr.new out=scored score=betas3
  type=parms;
  var dda ddabal dep depamt cashbk checks;
run;

data scored;
  set scored;
  off=log(((1-&pi1)*&rho1)/(&pi1*(1-&rho1)));
  p=1/(1+exp(-(ins-off)));
run;

proc print data=scored(obs=20);
  var p ins dda ddabal dep depamt cashbk checks;
run;
```

Obs	p	Ins	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.014381	-0.97058	1	56.29	2	955.51	0	1
2	0.018078	-0.73807	1	3292.17	2	961.60	0	1
3	0.016447	-0.83426	1	1723.86	2	2108.65	0	2
4	0.041854	0.12592	0	0.00	0	0.00	0	0
5	0.014171	-0.98552	1	67.91	2	519.24	0	3
6	0.018191	-0.73172	1	2554.58	1	501.36	0	2
7	0.014189	-0.98425	1	0.00	2	2883.08	0	12
8	0.016665	-0.82087	1	2641.33	3	4521.61	0	8
9	0.041854	0.12592	0	0.00	0	0.00	0	0
10	0.015250	-0.91103	1	52.22	1	75.59	0	0
11	0.022241	-0.52659	1	6163.29	2	2603.56	0	7
12	0.008384	-1.51631	1	431.12	2	568.43	1	2
13	0.041854	0.12592	0	0.00	0	0.00	0	0
14	0.009665	-1.37280	1	112.82	8	2688.75	0	3
15	0.017268	-0.78473	1	1146.61	3	11224.20	0	2
16	0.041854	0.12592	0	0.00	0	0.00	0	0
17	0.014433	-0.96694	1	1241.38	3	3538.14	0	15
18	0.016628	-0.82318	1	298.23	0	0.00	0	0
19	0.014973	-0.92966	1	367.04	2	4242.22	0	11
20	0.013701	-1.01975	1	1229.47	4	3514.57	0	10

2.03 Multiple Choice Poll

If the value for the offset is 3.2567, then the model corrected for oversampling will have

- an intercept that is 3.2567 lower in value compared to the model fitted to the biased sample.
- probabilities that are 3.2567% higher than the probabilities from the model fitted to the biased sample.
- probabilities that are 3.2567% lower than the probabilities from the model fitted to the biased sample.
- parameter estimates that are 3.2567% lower than the parameter estimates from the model fitted to the biased sample.

2.3 Chapter Summary

The standard logistic regression model assumes that the logit of the posterior probability is a linear combination of the input variables. The logit transformation is used to constrain the posterior probability to be between zero and one. The parameter estimates are estimated using the method of maximum likelihood. This method finds the parameter estimates that are most likely given the data. When you exponentiate the slope estimates, you obtain the odds ratio, which compares the odds of the event in one group to the odds of the event in another group.

There are many possibilities for scoring new data, from the SCORE statement in PROC LOGISTIC to DATA step code.

When you oversample rare events, you can use the OFFSET option to adjust the model so that the posterior probabilities reflect the population.

General form of the LOGISTIC procedure:

```
PROC LOGISTIC DATA=SAS-data-set <options>;  
  CLASS variables </option>;  
  MODEL response=predictors </options>;  
  UNITS predictor1=list1 </option>;  
  SCORE <options>;  
RUN;
```

General form of the SCORE procedure:

```
PROC SCORE DATA=SAS-data-set <options>;  
  VAR variables;  
RUN;
```

2.4 Solutions

Solutions to Exercises

Solutions to Student Activities (Polls/Quizzes)

2.01 Multiple Choice Poll – Correct Answer

The odds ratio for a \$1000 increase in income is 1.074.
This means that for every \$1000 increase in income

- a. the probability of the event increases 7.4%.
- b. the logit increases 7.4%
- ☒ c. the odds of the event increases 7.4%
- d. the log of the odds of the event increases 7.4%

11

2.02 Multiple Choice Poll – Correct Answer

How many individuals and responders in the sample have a predicted response rate greater than 0.025?

- ☒ a. 19323 individuals and 4833 responders
- b. 8161 individuals and 2579 responders
- c. 2240 individuals and 830 responders
- d. 1256 individuals and 547 responders

18

2.03 Multiple Choice Poll – Correct Answer

If the value for the offset is 3.2567, then the model corrected for oversampling will have

- ☒ a. an intercept that is 3.2567 lower in value compared to the model fitted to the biased sample.
- b. probabilities that are 3.2567% higher than the probabilities from the model fitted to the biased sample.
- c. probabilities that are 3.2567% lower than the probabilities from the model fitted to the biased sample.
- d. parameter estimates that are 3.2567% lower than the parameter estimates from the model fitted to the biased sample.

Chapter 3 Preparing the Input Variables

3.1	Missing Values	3-3
3.2	Categorical Inputs	3-13
3.3	Variable Clustering and Screening	3-26
3.4	Subset Selection.....	3-66
3.5	Chapter Summary.....	3-78
3.6	Solutions	3-79
	Solutions to Exercises	3-79
	Solutions to Student Activities (Polls/Quizzes)	3-79

3.1 Missing Values

Does $\Pr(\text{missing})$ Depend on the Data?

- No
 - MCAR
- Yes
 - that unobserved value
 - other unobserved values
 - other observed values
 - including the target

14	2	2
67	1	4
?	3	1
33	1	7
18	2	1
6	0	1
31	3	8
51	1	8

3

Missing values of the input variables can arise from several different mechanisms (Little 1992). A value is *missing completely at random* (MCAR) if the probability that it is missing is independent of the data. MCAR is a particularly easy mechanism to manage but is unrealistic in most predictive modeling applications.

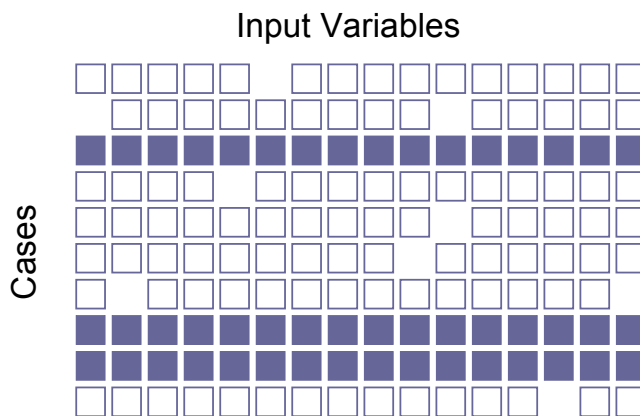
The probability that a value is missing might depend on the unobserved value. Credit applicants with fewer years at their current job might be less inclined to provide this information.

The probability that a value is missing might depend on observed values of other input variables. Customers with longer tenures might be less likely to have certain historic transactional data. Missingness might depend on a combination of values of correlated inputs.

An even more pathological missing-value mechanism occurs when the probability that a value is missing depends on values of unobserved (lurking) predictors. Transient customers might have missing values on a number of variables.

A fundamental concern for predictive modeling is that the missingness is related to the target. The more transient customers may be the best prospects for a new offer.

Complete Case Analysis



5

The default method for treating missing values in most SAS modeling procedures (including the LOGISTIC procedure) is complete-case analysis. In *complete-case analysis*, only those cases without any missing values are used in the analysis.

Complete-case analysis has some moderately attractive theoretical properties even when the missingness depends on observed values of other inputs (Donner 1982; Jones 1996). However, complete-case analysis has serious practical shortcomings with regards to predictive modeling. Even a smattering of missing values can cause an enormous loss of data in high dimensions. For instance, suppose each of the k input variables can be MCAR with probability α ; in this situation, the expected proportion of complete cases is

$$(1 - \alpha)^k$$

Therefore, a 1% probability of missing ($\alpha=.01$) for 100 inputs would leave only 37% of the data for analysis, 200 would leave 13%, and 400 would leave 2%. If the missingness was increased to 5% ($\alpha=.05$), then <1% of the data would be available with 100 inputs.


New Missing Values

Fitted Model:

$$\text{logit}(\hat{p}) = -2.1 + .072x_1 - .89x_2 - 1.4x_3$$

New Case: $(x_1, x_2, x_3) = (2, ?, -.5)$

Predicted Value:

$$\text{logit}(\hat{p}) = -2.1 + .144 - .89() + .7$$


6

Another practical consideration of any treatment of missing values is *scorability* (the practicality of method when it is deployed). The purpose of predictive modeling is scoring new cases. How would a model built on the complete cases score a new case if it had a missing value? To decline to score new incomplete cases would only be practical if there were a very small number of missing values.

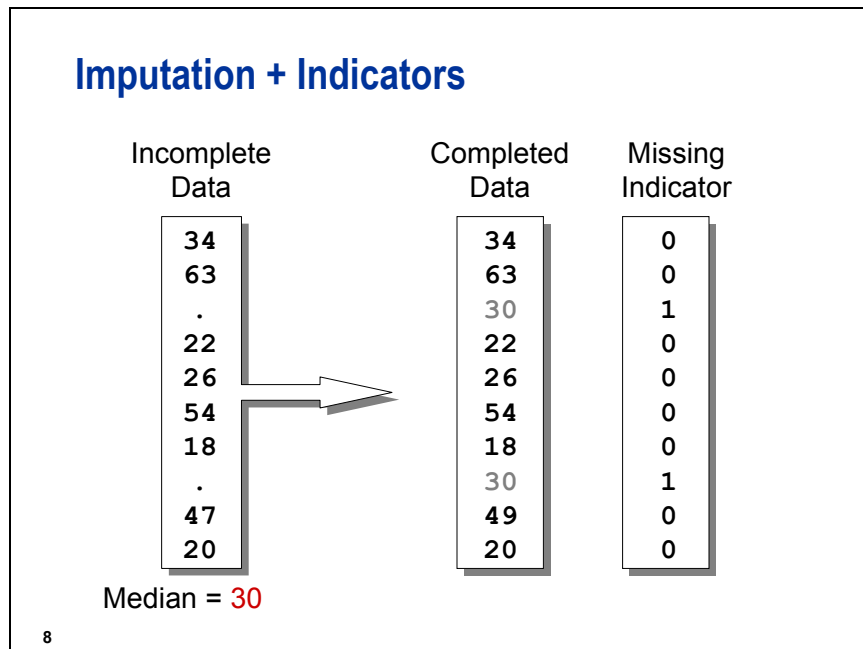
Missing Value Imputation

6	03	2.6	0	8.3	42	66	C03
12	04	1.8	0	0.5	86	65	C14
6.5	01	2.3	.33	4.8	37	66	C00
8	01	2.1	1	4.8	37	64	C08
6	01	2.8	1	9.6	22	66	C99
3	01	2.7	0	1.1	28	64	C00
2	02	2.1	1	5.9	21	63	C03
10	03	2.0	0	0.8	0	63	C99
7	01	2.5	0	5.5	62	67	C12
6.5	01	2.4	0	0.9	29	63	C05

7

Because of the aforementioned drawbacks of complete-case analysis, some type of missing value imputation is necessary. Imputation means filling in the missing values with some reasonable value. Many methods have been developed for imputing missing values (Little 1992). The principal consideration for most methods is getting valid statistical inference on the imputed data, not generalization.

Often, subject-matter knowledge can be used to impute missing data. For example, the missing values might be miscoded zeros.



One reasonable strategy for handling missing values in predictive modeling is to do the following steps.

1. Create missing indicators

$$MI_j = \begin{cases} 1 & \text{if } x_j \text{ is missing} \\ 0 & \text{otherwise} \end{cases}$$

and treat them as new input variables in the analysis.

2. Use median imputation. Fill the missing value of x_j with the median of the complete cases for that variable.
3. Create a new level representing missing (unknown) for categorical inputs.

If a very large percentage of values are missing (>50%), then the variable might be better handled by omitting it from the analysis or by creating the missing indicator only. If a very small percentage of the values are missing (<1%), then the missing indicator is of little value.

This strategy is somewhat unsophisticated but satisfies two of the most important considerations in predictive modeling: scorability and the potential relationship of missingness with the target. A new case is easily scored; first replace the missing values with the medians from the development data and then apply the prediction model.

There is statistical literature concerning different missing value imputation methods, including discussions of the demerits of mean and median imputation and missing indicators (Donner 1982; Jones 1997). Unfortunately, most of the advice is based on considerations that are peripheral to predictive modeling. There is very little advice when the functional form of the model is not assumed to be correct, when the goal is to get good predictions that can be practically applied to new cases, when p -values and hypothesis tests are largely irrelevant, and when the missingness may be highly pathological, in other words, depending on lurking predictors.

3.01 Multiple Choice Poll

Which of the following statements is false regarding missing values in predictive modeling applications?

- a. In complete case analysis, there can be an enormous loss of data when the missing values are spread across many variables.
- b. Observations with missing values will not be scored in PROC LOGISTIC.
- c. Missing indicator variables can be used to capture the relationship between the target variable and missing inputs.
- d. The missing completely at random assumption is valid in most predictive modeling applications.



Imputing Missing Values

The objective of the following program is to create missing value indicator variables and to replace missing values with the variable median.

```
proc print data=develop(obs=30) ;
    var ccbal ccpurc income hmow;
run;
```

Obs	CCBal	CCPurc	Income	HMOW
1	483.65	0	16	1
2	0.00	1	4	1
3	0.00	0	30	1
4	65.76	0	125	1
5	0.00	0	25	1
6	38.62	0	19	0
7	85202.99	0	55	1
8	0.00	0	13	0
9	.	.	20	0
10	0.00	0	54	0
11	0.00	0	.	.
12	0.00	0	25	1
13	.	.	102	1
14	.	.	24	1
15	0.00	0	8	1
16	0.00	0	100	1
17	323.13	0	13	1
18	.	.	17	0
19	.	.	8	1
20	0.00	0	7	1
21	0.00	0	.	.
22	32366.86	0	.	1
23	0.00	0	9	0
24	.	.	45	1
25	.	.	36	1
26	1378.46	1	60	1
27	.	.	35	1
28	17135.95	0	40	1
29	0.00	0	42	0
30	0.00	0	112	1

Fifteen of the input variables were selected for imputation. Two arrays are created, one called MI, which contains the missing value indicator variables, and one called X, which contains the input variables. It is critical that the order of the variables in the array MI matches the order of the variables in array X. Defining the dimension with an asterisk causes the array elements to be automatically counted. In the DO loop, the DIM function returns the dimension of the array. Thus, the DO loop will execute 15 times in this example. The assignment statement inside the DO loop causes the entries of MI to be 1 if the corresponding entry in X is missing, and 0 otherwise.

```

data develop1;
  set develop;
  /* name the missing indicator variables */
  array mi{*} MIAcctAg MIPhone MIPOS MIPOSAmt
              MIInv MIInvBal MICC MICCBal
              MICCPurc MIIncome MIHMOwn MILORes
              MIHMVal MIAge MICRScor;
  /* select variables with missing values */
  array x{*} acctage phone pos posamt
              inv invbal cc ccbal
              ccpurc income hmown lores
              hmval age crscore;
  do i=1 to dim(mi);
    mi{i}=(x{i}=.);
  end;
run;

```

The STDIZE procedure with the REONLY option can be used to replace missing values. The METHOD= option enables you to choose several different location measures such as the mean, median, and midrange. The output data set created by the OUT= option contains all the variables in the input data set where the variables listed in the VAR statement are imputed. Only numeric input variables should be used in PROC STDIZE.

```

proc stdize data=develop1
  reonly
  method=median
  out=imputed;
  var &inputs;
run;

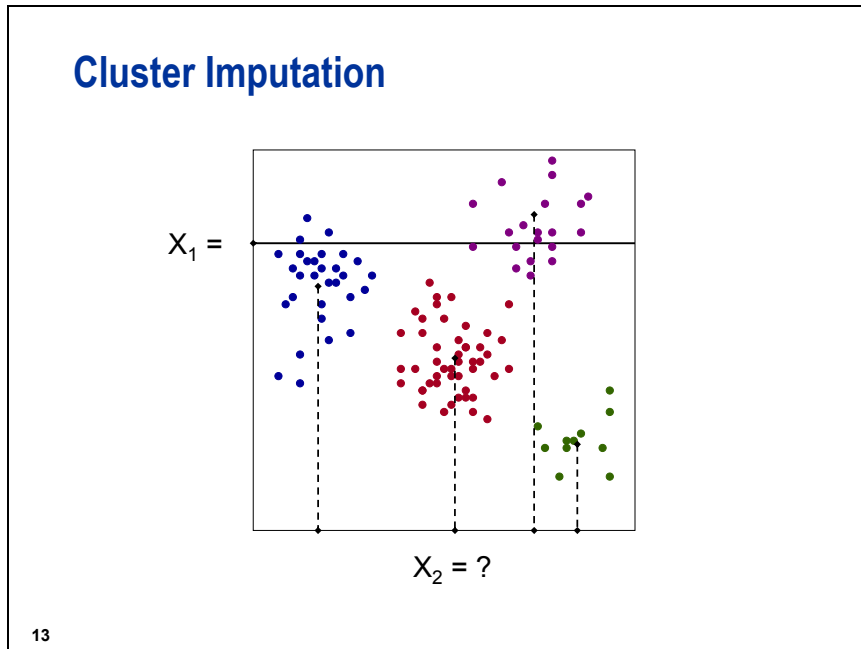
proc print data=imputed(obs=12);
  var ccbal miccbal ccpurc miccpurc
      income miincome hmown mihmown;
run;

```



The REPLACE option in PROC STANDARD can be used to replace missing values with the mean of that variable on the nonmissing cases.

Obs	CCBal	MICCBal	CCPurc	MICCPurc	Income	MIIncome	HMOwn	MIHMOwn
1	483.65	0	0	0	16	0	1	0
2	0.00	0	1	0	4	0	1	0
3	0.00	0	0	0	30	0	1	0
4	65.76	0	0	0	125	0	1	0
5	0.00	0	0	0	25	0	1	0
6	38.62	0	0	0	19	0	0	0
7	85202.99	0	0	0	55	0	1	0
8	0.00	0	0	0	13	0	0	0
9	0.00	1	0	1	20	0	0	0
10	0.00	0	0	0	54	0	0	0
11	0.00	0	0	0	35	1	1	1
12	0.00	0	0	0	25	0	1	0



Mean-imputation uses the unconditional mean of the variable. An attractive extension would be to use the mean conditional on the other inputs. This is referred to as regression imputation. Regression imputation would usually give better estimates of the missing values. Specifically, k linear regression models could be built, one for each input variable using the other inputs as predictors. This would presumably give better imputations and be able to accommodate missingness that depends on the values of the other inputs. An added complication is that the other inputs may have missing values. Consequently, the k imputation regressions also need to accommodate missing values.

Cluster-mean imputation is a somewhat more practical alternative:

1. cluster the cases into relatively homogenous subgroups
2. mean-imputation within each group
3. for new cases with multiple missing values, use the cluster mean that is closest in all the nonmissing dimensions.

This method can accommodate missingness that depends on the other input variables. This method is implemented in SAS Enterprise Miner. For large data sets, the FASTCLUS procedure may also be appropriate.

A simple but less effective alternative is to define a priori segments (for example, high, middle, low, and unknown income), and then do mean or median imputation within each segment (see the exercises in Appendix A).

3.02 Multiple Choice Poll

For the cell `grp_resp=0` and `grp_amt=0`, what was the missing value for `donor_age` replaced with?

- a. 57
- b. 64
- c. 58
- d. 65

3.2 Categorical Inputs

Dummy Variables

X	D _A	D _B	D _C	D _D
D	0	0	0	1
B	0	1	0	0
C	0	0	1	0
C	0	0	1	0
A	1	0	0	0
A	1	0	0	0
D	0	0	0	1
C	0	0	1	0
A	1	0	0	0
⋮	⋮	⋮	⋮	⋮

20

With the CLASS statement, you can use categorical input variables in the LOGISTIC procedure without having to create dummy variables in a DATA step. You can specify the type of parameterization to use, such as effect coding and reference coding, and the reference level. The choice of the reference level is immaterial in predictive modeling because different reference levels give the same predictions.

Smarter Variables

ZIP	HomeVal	Urbanicity	Local ...
99801	75	1	1
99622	100	2	1
99523	150	1	1
99523	150	1	0
99737	150	3	1
99937	75	3	1
99533	100	2	1
99523	150	1	0
99622	100	3	1
⋮	⋮	⋮	⋮

21

Expanding categorical inputs into dummy variables can greatly increase the dimension of the input space. A smarter method is to use subject-matter information to create new inputs that represent relevant sources of variation. A categorical input might be best thought of as a link to other data sets. For example, geographic areas are often mapped to several relevant demographic variables.

Quasi-Complete Separation

	0	1	D _A	D _B	D _C	D _D
A	28	7	1	0	0	0
B	16	0	0	1	0	0
C	94	11	0	0	1	0
D	23	21	0	0	0	1

22

Including categorical inputs in the model can cause quasi-complete separation. *Quasi-complete separation* occurs when a level of the categorical input has a target event rate of 0 or 100%. The coefficient of a dummy variable represents the difference in the logits between that level and the reference level. When quasi-complete separation occurs, one of the logits will be infinite. The likelihood will not have a maximum in at least one dimension, so the ML estimate of that coefficient will be infinite. If the zero-cell category is the reference level, then all the coefficients for the dummy variables will be infinite.

Quasi-complete separation complicates model interpretation. It can also affect the convergence of the estimation algorithm. Furthermore, it might lead to incorrect decisions regarding variable selection.

The most common cause of quasi-complete separation in predictive modeling is categorical inputs with rare categories. The best remedy for sparseness is collapsing levels of the categorical variable.

Clustering Levels

	0	1		0	1		0	1		0	1
A	28	7		28	7		138	18		161	39
B	16	0	→	110	11	→	23	21			
C	94	11									
D	23	21		23	21						

Merged:	B & C	A & BC	ABC & D	
$\chi^2 =$	31.7	30.7	28.6	0
	100%	97%	90%	0%

26

Ideally, subject-matter considerations should be used to collapse levels (reduce the dimension) of categorical inputs. This is not always practical in predictive modeling. A simple data-driven method for collapsing levels of contingency tables was developed by Greenacre (1988, 1993). The levels (rows) are hierarchically clustered based on the reduction in the chi-squared test of association between the categorical variable and the target. At each step, the two levels that give the least reduction in the chi-squared statistic are merged. The process is continued until the reduction in chi-squared drops below some threshold (for example, 99%). This method will quickly throw rare categories in with other categories that have similar marginal response rates. While this method is simple and effective, there is a potential loss of information because only univariate associations are considered.

3.03 Multiple Choice Poll

Which of the following statements is false regarding Greenacre's method for collapsing levels of contingency tables?

- The levels are hierarchically clustered based on the reduction in the chi-squared test of association.
- Levels with similar marginal response rates are merged.
- The method accounts for the sample size in each level.
- The method is appropriate for any categorical input.

28



Clustering Levels of Categorical Inputs

The levels of a categorical input can be clustered using Greenacre's method (1988, 1993) in the CLUSTER procedure. PROC CLUSTER was designed for general clustering applications, but with some simple pre-processing of the data, it can be made to cluster levels of categorical variables.

The variable **Branch** has 19 levels. The first step is to create a data set that contains the proportion of the target event (**ins**) and number of cases in each level. The NWAY option caused the output data set to have 19 observations, one for each of the 19 levels. The output data set also has a variable **prop** for the proportion of target events. It automatically creates the variable **_FREQ_**, which counts the number of cases in each level.

```
proc means data=imputed noprint nway;
  class branch;
  var ins;
  output out=level mean=prop;
run;

proc print data=level;
run;
```

Obs	Branch	_TYPE_	_FREQ_	prop
1	B1	1	2819	0.36999
2	B10	1	273	0.40293
3	B11	1	247	0.35628
4	B12	1	549	0.36430
5	B13	1	535	0.37196
6	B14	1	1072	0.19123
7	B15	1	2235	0.24251
8	B16	1	1534	0.27771
9	B17	1	850	0.37059
10	B18	1	541	0.35675
11	B19	1	285	0.38596
12	B2	1	5345	0.32460
13	B3	1	2844	0.38186
14	B4	1	5633	0.37493
15	B5	1	2752	0.38118
16	B6	1	1438	0.37830
17	B7	1	1413	0.34678
18	B8	1	1341	0.38553
19	B9	1	558	0.37814

Using the FREQ statement and METHOD=WARD in PROC CLUSTER gives identical results to Greenacre's method. The OUTTREE= option creates an output data set that can be used by the TREE procedure to draw a tree diagram. The ID statement specifies a variable that identifies observations in the printed cluster history and in the OUTTREE= data set. The ODS TRACE statement with the LISTING option will show which output objects are associated with each table in the PROC CLUSTER output.


```
ods trace on/listing;

proc cluster data=level method=ward
    outtree=fortree;
    freq _freq_;
    var prop;
    id branch;
run;

ods trace off;
```

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

Output Added:

Name: EigenvalueTable
Label: Eigenvalues of the Covariance Matrix
Template: stat.cluster.EigenvalueTable
Path: Cluster.EigenvalueTable

Eigenvalues of the Covariance Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	0.00245716		1.0000	1.0000

Root-Mean-Square Total-Sample Standard Deviation = 0.04957
Root-Mean-Square Distance Between Observations = 0.070102

Output Added:

Name: ClusterHistory
 Label: Cluster History
 Template: stat.cluster.ClusterHistory
 Path: Cluster.ClusterHistory

Cluster History						T i e
NCL	--Clusters Joined--		FREQ	SPRSQ	RSQ	
18	B6	B9	1996	0.0000	1.00	
17	B11	B18	788	0.0000	1.00	
16	B19	B8	1626	0.0000	1.00	
15	B1	B17	3669	0.0000	1.00	
14	B3	B5	5596	0.0000	1.00	
13	CL15	B13	4204	0.0000	1.00	
12	CL14	CL18	7592	0.0002	1.00	
11	CL13	B12	4753	0.0002	1.00	
10	CL16	CL12	9218	0.0004	.999	
9	CL17	B7	2201	0.0006	.999	
8	CL11	B4	10386	0.0009	.998	
7	B10	CL10	9491	0.0015	.996	
6	CL8	CL7	19877	0.0058	.990	
5	CL9	B2	7546	0.0130	.977	
4	B15	B16	3769	0.0142	.963	
3	B14	CL4	4841	0.0453	.918	
2	CL6	CL5	27423	0.1399	.778	
1	CL2	CL3	32264	0.7779	.000	

The column labeled RSQ in the output is equivalent to the proportion of chi-squared in the 19×2 contingency table remaining after the levels are collapsed. At each step, the levels that give the smallest decrease in chi-squared are merged. The change in chi-squared is listed in the SPRSQ column. The rows in the summary represent the results after the listed clusters were merged. The number of clusters is reduced from 18 to 1. When previously collapsed levels are merged, they are denoted using the CL as the prefix and the number of resulting clusters as the suffix. For example, at the sixth step, CL15 represents B1 and B17 that were merged at the fourth step creating 15 clusters.

To calculate the optimum number of clusters, the chi-square statistic and the associated *p*-value needs to be computed for each collapsed contingency table. This information can be obtained by multiplying the chi-square statistic from the 19×2 contingency table with the proportion of chi-squared remaining after the levels are collapsed. Therefore, the next program converts the output object CLUSTERHISTORY to an output data set. The FREQ procedure is used to compute the chi-square statistic for the 19×2 contingency table.

```
ods listing close;
ods output clusterhistory=cluster;

proc cluster data=level method=ward;
  freq _freq_;
  var prop;
  id branch;
run;
ods listing;

proc print data=cluster;
run;
```

Obs	Number Of Clusters	Idj1	Idj2	FreqOf New Cluster	Semipartial RSq	RSquared	Tie
1	18	B6	B9	1996	0.0000	1.00	
2	17	B11	B18	788	0.0000	1.00	
3	16	B19	B8	1626	0.0000	1.00	
4	15	B1	B17	3669	0.0000	1.00	
5	14	B3	B5	5596	0.0000	1.00	
6	13	CL15	B13	4204	0.0000	1.00	
7	12	CL14	CL18	7592	0.0002	1.00	
8	11	CL13	B12	4753	0.0002	1.00	
9	10	CL16	CL12	9218	0.0004	.999	
10	9	CL17	B7	2201	0.0006	.999	
11	8	CL11	B4	10386	0.0009	.998	
12	7	B10	CL10	9491	0.0015	.996	
13	6	CL8	CL7	19877	0.0058	.990	
14	5	CL9	B2	7546	0.0130	.977	
15	4	B15	B16	3769	0.0142	.963	
16	3	B14	CL4	4841	0.0453	.918	
17	2	CL6	CL5	27423	0.1399	.778	
18	1	CL2	CL3	32264	0.7779	.000	

```
proc freq data=imputed noprint;
  tables branch*ins / chisq;
  output out=chi(keep=_pchi_) chisq;
run;

proc print data=chi;
run;
```

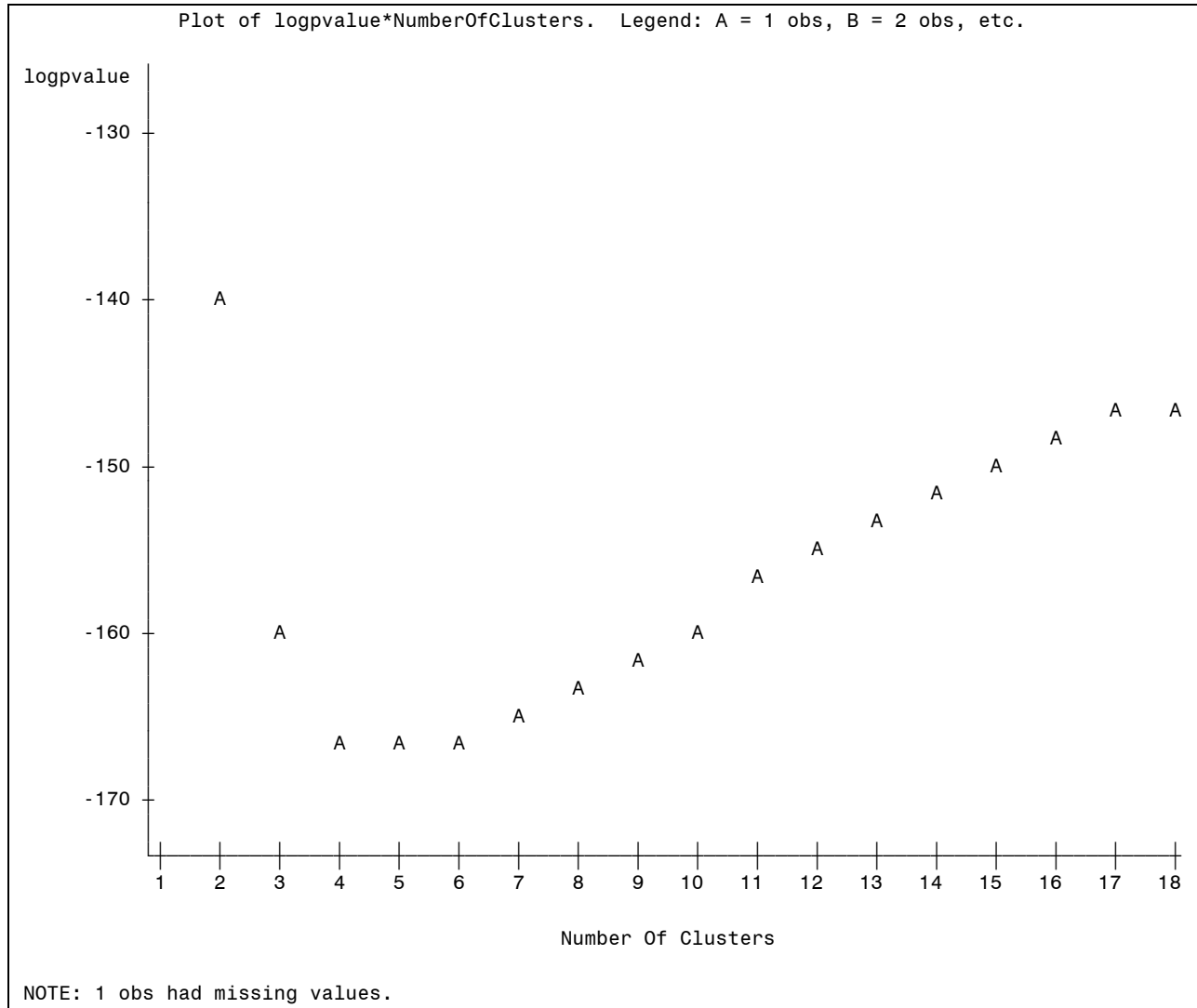
Obs	_PCHI_
1	350.164

The DATA step computes the chi-square statistic for each collapsed contingency table. The `_n_` variable is used to put the overall chi-square value in each observation for the data set `cutoff`. The LOGSDF function computes the log of the probability that an observation from a specified distribution is greater than or equal to a specified value. The arguments for the function are the specified distribution in quotes, the numeric random variable, and the degrees of freedom. The log of the p -value is calculated in order to produce a more visually appealing graph.

```
data cutoff;  
    if _n_ = 1 then set chi;  
    set cluster;  
    chisquare=_pchi_*rsquared;  
    degfree=numberofclusters-1;  
    logpvalue=logsf('CHISQ',chisquare,degfree);  
run;
```

The PLOT procedure plots the log of the p -value by the number of clusters. The VPOS option specifies the number of print positions on the vertical axis.

```
proc plot data=cutoff;
  plot logpvalue*numberofclusters/vpos=30;
run; quit;
```



The graph shows that the 4-, 5-, and 6-cluster solutions had the lowest p -values.



The note at the bottom of the plot refers to an observation with a missing log p -value. Of course, the 1-cluster solution is associated with a χ^2 test with no degrees of freedom. Since this makes no sense, there is no p -value.

To find the cluster solution with the lowest p -value, and assign that value to the macro variable **NCL**, use the SQL procedure.

```
proc sql;
  select NumberOfClusters into :ncl
  from cutoff
  having logpvalue=min(logpvalue);
quit;
```

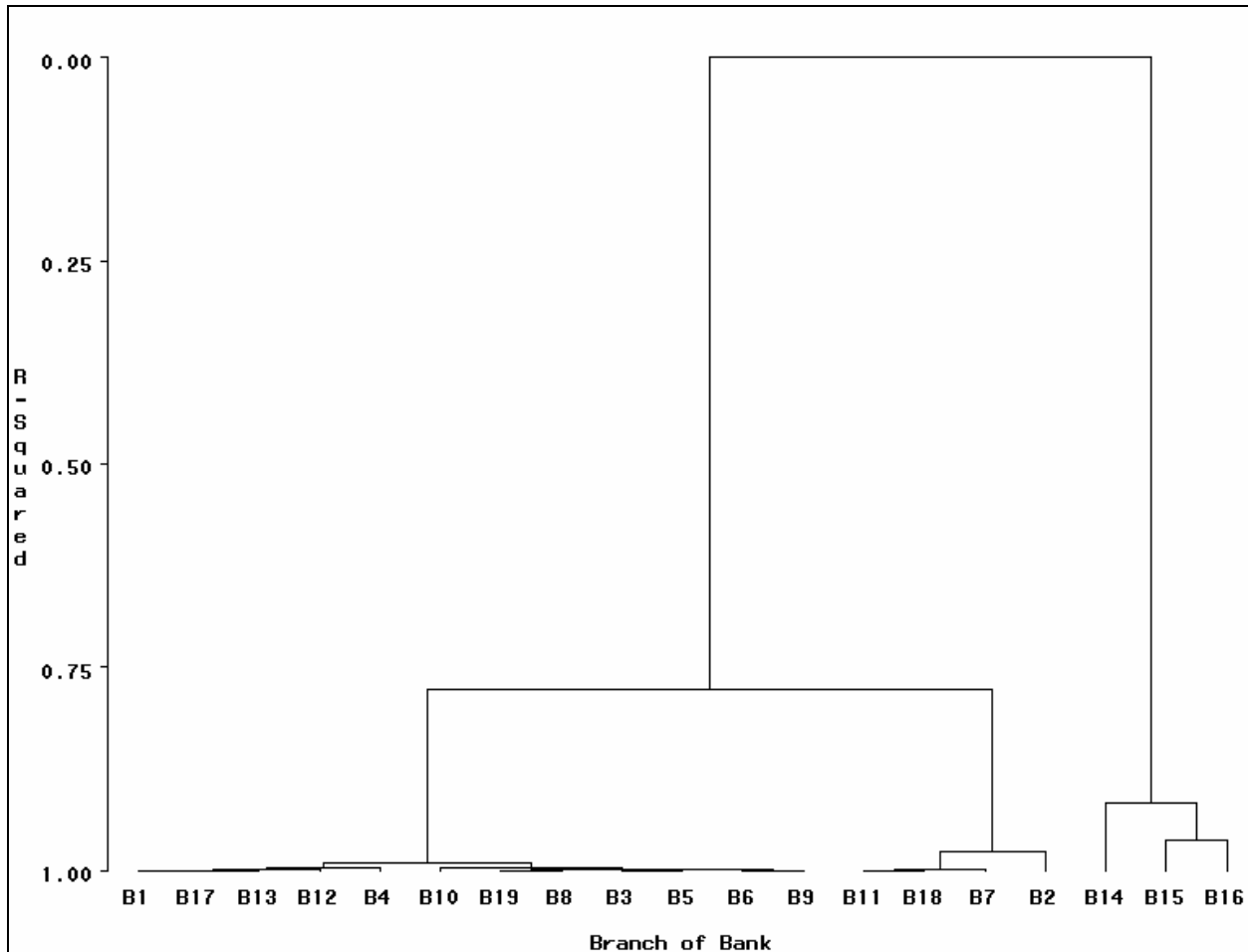
The 5-cluster solution has the smallest log p -value.

Number Of Clusters

5

The TREE procedure produces a high-resolution dendrogram. The H= option specifies the variable to be used as the height axis of the dendrogram. In this example, the proportion of the chi-squared statistic is the vertical axis.

```
proc tree data=fortree h=rsq
  nclusters=&ncl out=clus;
  id branch;
run;
```



The dendrogram shows that several branches can be combined with a miniscule reduction in chi-squared. The horizontal axis is also roughly ordered by the mean proportion of events in each cluster. Choosing a cluster near the center of the tree (for example, B11, B18, B7, and B2) as a reference group may lead to better models if the variable selection method you choose incrementally adds variables to the model (Cohen 1991). The **clus** data set from the OUT= option in PROC TREE shows which levels of **Branch** are associated with each cluster (if the NCLUSTERS= option is correctly specified).

```
proc sort data=clus;
    by clusname;
run;

proc print data=clus;
    by clusname;
    id clusname;
run;
```

CLUSNAME	Branch	CLUSTER
B14	B14	5
B15	B15	3
B16	B16	4
CL5	B11	2
	B18	2
	B7	2
	B2	2
CL6	B6	1
	B9	1
	B19	1
	B8	1
	B1	1
	B17	1
	B3	1
	B5	1
	B13	1
	B12	1
	B4	1
	B10	1

A DATA step is used to assign the branches to dummy variables. Four dummy variables are created with the second cluster designated as the reference level. Note that the dummy variables are numbered sequentially.

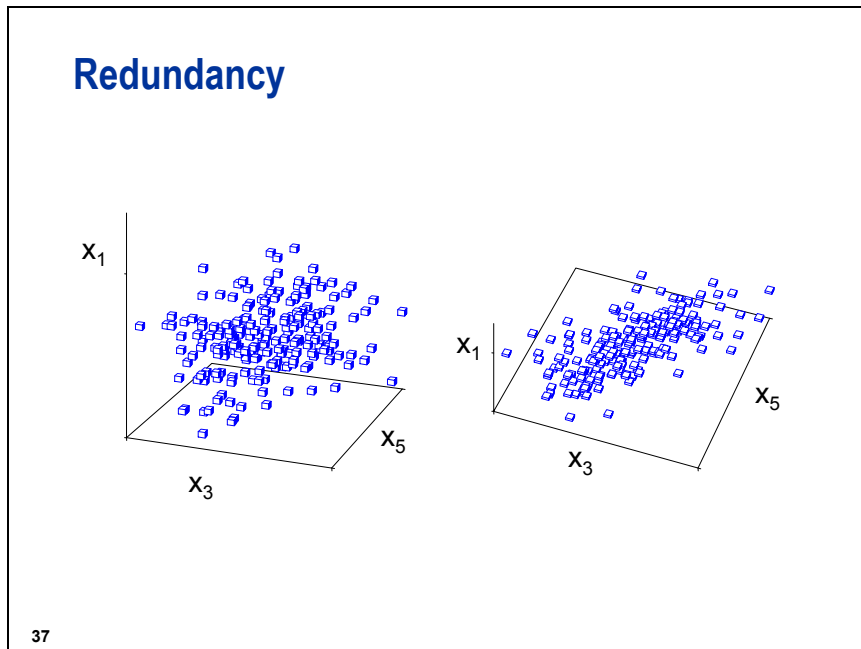
```
data imputed;
  set imputed;
  brclus1=(branch in ('B6','B9','B19','B8','B1','B17',
    'B3','B5','B13','B12','B4','B10'));
  brclus2=(branch='B15');
  brclus3=(branch='B16');
  brclus4=(branch='B14');
run;
```


3.04 Multiple Choice Poll

What is the optimum number of clusters for the cluster_code variable?

- a. 3
- b. 4
- c. 5
- d. 6

3.3 Variable Clustering and Screening



Including redundant inputs can degrade the analysis by

- destabilizing the parameter estimates
- increasing the risk of overfitting
- confounding interpretation
- increasing computation time
- increasing scoring effort
- increasing the cost of data collection and augmentation.

Redundancy is an *unsupervised* concept. It does not involve the target variable. In contrast, irrelevant inputs are not substantially related to the target. In high-dimensional data sets, identifying irrelevant inputs is more difficult than identifying redundant inputs. A good strategy is to first reduce redundancy and then tackle irrelevancy in a lower dimension space.

Principal Components

X_1	X_2	X_3	X_4	X_5	PC_1	PC_2	PC_3	PC_4	PC_5
1	-.11	-.03	-.69	-.04	1.8	0	0	0	0
	1	-.14	.07	.04		1.7	0	0	0
		1	.04	-.73			1.0	0	0
			1	.02				0.3	0
				1					0.2

Correlation Matrix
Covariance Matrix

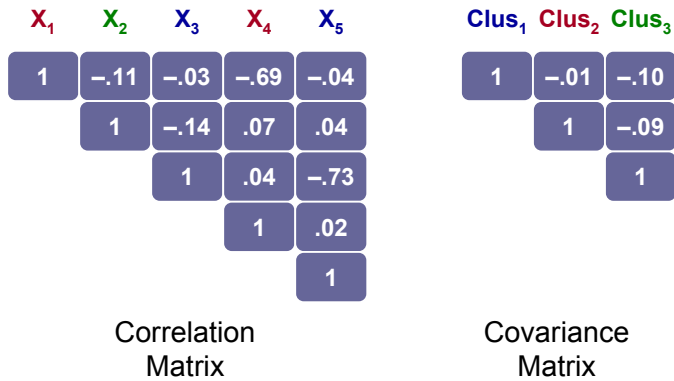
38

Principal components analysis (Jackson 1991) can be used for reducing redundant dimensions. A set of k variables can be transformed into a set of k principal components. The principal components (PCs) are linear combinations of the k variables constructed to be jointly uncorrelated and to explain the total variability among the original (standardized) variables.

The correlation matrix is the covariance matrix of the standardized variables. Because each standardized variable has a variance equal to one, the total variability among the standardized variables is just the number of variables. The principal components are produced by an eigen-decomposition of the correlation matrix. The *eigenvalues* are the variances of the PCs; they sum to the number of variables. The first PC corresponds to the first eigenvalue and explains the largest proportion of the variability. Each PC explains a decreasing amount of the total variability. In the above example, the first three PCs explain 90% of the total variability.

In practice, dimension reduction is achieved by retaining only the first few PCs provided they explain a sufficient proportion of the total variation. The reduced set of PCs might then be used in place of the original variables in the analysis.

Cluster Components



39

Variable clustering as implemented in the VARCLUS procedure (SAS Institute Inc. 1990) is an alternative method for eliminating redundant dimensions that is closely related to principal components analysis. The result of clustering k variables is a set of $\leq k$ cluster components. Like PCs, the cluster components are linear combinations of the original variables. Unlike the PCs, the cluster components are not uncorrelated and do not explain all the variability in the original variables. The cluster components (scores) are standardized to have unit variance.

The three cluster components in the above example correspond to eigenvalues of 1.7, 1.7, and 1.0. Consequently, 88% of variation among the original (standardized) variables is explained by the three clusters.

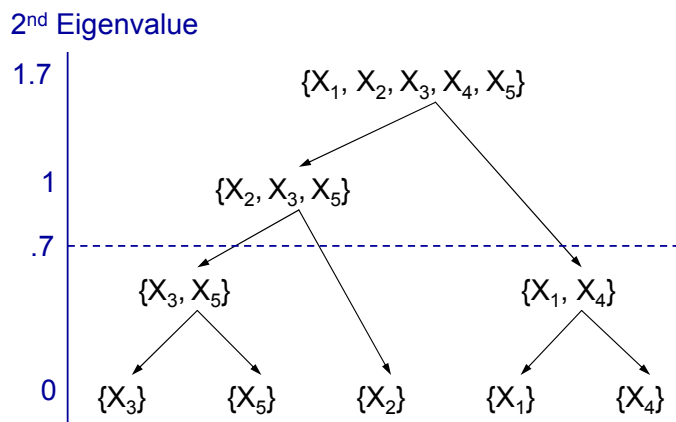
Coefficients

PC ₁	PC ₂	PC ₃	PC ₄	PC ₅		Clus ₁	Clus ₂	Clus ₃
-.28	-.64	.09	.70	.12	x₁	0	.56	0
.22	.08	.97	.03	.11	x₂	0	0	1
-.63	.31	.04	-.09	.70	x₃	.54	0	0
.26	.65	-.14	.70	.04	x₄	0	-.39	0
.64	-.63	-.20	-.08	.69	x₅	-.37	0	0

40

The chief advantage of variable clustering over principal components is the coefficients. The coefficients of the PCs (*eigenvectors*) are usually nonzero for all the original variables. Thus, even if only a few PCs were used, all the inputs would still have to be retained in the analysis. In contrast, the cluster component scores have nonzero coefficients on disjoint subsets of the variables. These subsets correspond to disjoint clusters of the original variables. In the above example, the clusters are $\{x_3, x_5\}$, $\{x_1, x_4\}$, and $\{x_2\}$.

Divisive Clustering



41

Variable clustering finds groups of variables that are as correlated as possible among themselves and as uncorrelated as possible with variables in other clusters. The basic algorithm is binary and divisive. All variables start in one cluster. A principal components analysis is done on the variables in the cluster. If the second eigenvalue is greater than a specified threshold (in other words, there is more than one dominant dimension), then the cluster is split. The PC scores are then rotated obliquely so that the variables can be split into two groups. This process is repeated for the two child clusters until the second eigenvalue drops below the threshold. (By default, PROC VARCLUS does a nonhierarchical version of this algorithm where variables can also be reassigned to other clusters.)

Larger thresholds for the second eigenvalue give fewer clusters and less of the variation is explained. Smaller thresholds give more clusters and more of the variation is explained. The value 1 is a common choice for the threshold because it represents the average size of the eigenvalues. To account for sampling variability, smaller values such as .7 have been suggested (Jackson 1991).

Cluster Representatives

$$1 - R^2 \text{ ratio} = \frac{1 - R_{\text{own cluster}}^2}{1 - R_{\text{next closest}}^2}$$

$$\frac{1-\uparrow}{1-\downarrow} \Rightarrow \frac{\downarrow}{\uparrow} \Rightarrow \downarrow$$

42

As with principal components analysis, dimension reduction could be achieved by replacing the original variables with the cluster scores (components). A simple alternative is to select a representative variable from each cluster. An ideal representative would have high correlation with its own cluster and have a low correlation with the other clusters. Consequently, variables with the lowest $1 - R^2$ ratio (defined above) in each cluster would be good representatives. Of course, subject-matter considerations might dictate the selection of other representatives.

3.05 Multiple Choice Poll

If you had 15 variables broken into 5 clusters, the sum of the eigenvalues over all clusters would be

- a. 5
- b. 10
- c. 15
- d. 20
- e. None of the above

44



Variable Clustering

The VARCLUS procedure¹ clusters numeric variables. The MAXEIGEN= option specifies the largest permissible value of the second eigenvalue in each cluster. The default is 1 (using the correlation matrix). The SHORT option suppresses some of the output. The OUTTREE= option creates an output data set that can be used in the TREE procedure. When the OUTTREE= option is used, the clusters at different levels maintain a hierarchical structure that prevents variables from transferring from one cluster to another after the split is made. If the OUTTREE= option is not used, then use the HI option to get a hierarchical structure. The VAR statement lists the numeric variables to cluster: the original numeric inputs, the missing indicators, and the dummy variables for the collapsed **Branch** (64 total).

```
proc varclus data=imputed
    maxeigen=.7
    outtree=fortree
    short;
var &inputs brclus1-brclus4 miacctag
    miphone mipos miposamt miinv
    miinvbal micc miccbal miccpurc
    miincome mihmown milores mihmval
    miage micrscor;
run;
```

The output from PROC VARCLUS shows the results for each step in the divisive clustering algorithm. Even with the SHORT option, the amount of printed output is voluminous.

Partial Output

Oblique Principal Component Cluster Analysis					
Observations	32264	Proportion	0		
Variables	64	Maxeigen	0.7		
Clustering algorithm converged.					
Cluster Summary for 1 Cluster					
Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	64	64	9.228833	0.1442	5.0980
Total variation explained = 9.228833 Proportion = 0.1442					
Cluster 1 will be split because it has the largest second eigenvalue, 5.097981, which is greater than the MAXEIGEN=0.7 value.					
Clustering algorithm converged.					

¹ The default settings for the VARCLUS procedure changed from SAS Version 8 to SAS[®]9. If you would like to emulate the behavior of SAS Version 8 exactly, specify MAXSEARCH=0 in the PROC VARCLUS statement.

Cluster Summary for 2 Clusters

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	48	48	9.197492	0.1916	3.4316
2	16	16	5.047174	0.3154	2.0471

Total variation explained = 14.24467 Proportion = 0.2226

2 Clusters		R-squared with			Variable Label
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	
Cluster 1	AcctAge	0.0004	0.0002	0.9998	Age of Oldest Account
	Dep	0.0010	0.0001	0.9991	Checking Deposits
	DepAmt	0.0007	0.0000	0.9993	Amount Deposited
	CashBk	0.0003	0.0000	0.9997	Number Cash Back
	Checks	0.0037	0.0001	0.9963	Number of Checks
	DirDep	0.0001	0.0001	0.9999	Direct Deposit
	NSF	0.0002	0.0001	0.9999	Number Insufficient Fund
	NSFAmt	0.0000	0.0000	1.0000	Amount NSF
	Phone	0.0179	0.0001	0.9821	Number Telephone Banking
	Teller	0.0012	0.0001	0.9989	Teller Visits
	Sav	0.0006	0.0000	0.9994	Saving Account
	SavBal	0.0000	0.0000	1.0000	Saving Balance
	ATM	0.0003	0.0000	0.9997	ATM
	ATMAmt	0.0013	0.0000	0.9987	ATM Withdrawal Amount
	POS	0.0257	0.0000	0.9743	Number Point of Sale
	POSAmt	0.0242	0.0001	0.9759	Amount Point of Sale
	CD	0.0002	0.0000	0.9998	Certificate of Deposit
	CDBal	0.0002	0.0000	0.9998	CD Balance
	LOC	0.0066	0.0000	0.9934	Line of Credit
	LOCBal	0.0014	0.0001	0.9987	Line of Credit Balance
	Inv	0.0052	0.0001	0.9949	Investment
	InvBal	0.0002	0.0000	0.9999	Investment Balance
	ILS	0.0026	0.0000	0.9974	Installment Loan
	ILSBal	0.0024	0.0000	0.9977	Loan Balance
	MM	0.0000	0.0000	1.0000	Money Market
	MMBal	0.0001	0.0000	1.0000	Money Market Balance
	MTG	0.0032	0.0003	0.9970	Mortgage
	MTGBal	0.0006	0.0000	0.9995	Mortgage Balance
	CC	0.1279	0.0007	0.8727	Credit Card
	CCBal	0.0019	0.0000	0.9981	Credit Card Balance
	CCPurc	0.0211	0.0000	0.9789	Credit Card Purchases
	SDB	0.0006	0.0000	0.9995	Safety Deposit Box
	CRScore	0.0001	0.0000	0.9999	Credit Score
	Moved	0.0000	0.0000	1.0000	Recent Address Change
	InArea	0.0003	0.0000	0.9997	Local Address
	brclus1	0.2239	0.0535	0.8200	

Oblique Principal Component Cluster Analysis					
2 Clusters		R-squared with		1-R**2 Ratio	Variable Label
Cluster	Variable	Own Cluster	Next Closest		
	brclus2	0.5331	0.0022	0.4679	
	brclus3	0.0061	0.0019	0.9958	
	brclus4	0.2426	0.0010	0.7582	
	MIAcctAg	0.0000	0.0000	1.0000	
	MIPhone	0.9925	0.0044	0.0076	
	MIPOS	0.9925	0.0044	0.0076	
	MIPOSamt	0.9925	0.0044	0.0076	
	MIInv	0.9925	0.0044	0.0076	
	MIInvBal	0.9925	0.0044	0.0076	
	MICC	0.9925	0.0044	0.0076	
	MICCBal	0.9925	0.0044	0.0076	
	MICCPurc	0.9925	0.0044	0.0076	

Cluster 2	DDA	0.0002	0.0001	0.9999	Checking Account
	DDABal	0.0004	0.0001	0.9997	Checking Balance
	IRA	0.0000	0.0000	1.0000	Retirement Account
	IRABal	0.0001	0.0000	0.9999	IRA Balance
	MMCred	0.0000	0.0000	1.0000	Money Market Credits
	Income	0.0113	0.0000	0.9887	Income
	HMOwn	0.1850	0.0000	0.8150	Owns Home
	LORes	0.0005	0.0001	0.9996	Length of Residence
	HMVal	0.0137	0.0000	0.9863	Home Value
	Age	0.0009	0.0000	0.9991	Age
	MIIncome	0.9857	0.0028	0.0143	
	MIHMOwn	0.9541	0.0030	0.0461	
	MILORes	0.9857	0.0028	0.0143	
	MIHMVal	0.9857	0.0028	0.0143	
	MIAge	0.9194	0.0027	0.0808	
	MICRScor	0.0044	0.0000	0.9956	

Cluster 1 will be split because it has the largest second eigenvalue, 3.431637, which is greater than the MAXEIGEN=0.7 value.

Using the Output Delivery System, you can restrict the output to just the objects of interest. This requires knowledge of the object names. These can be found in the documentation for the VARCLUS procedure, or by using the TRACE statement.

```
ods trace on/listing;
proc varclus data=imputed
    maxeigen=.7
    outtree=fortree
    short;
    var &inputs brclus1-brclus4 miacctag
        miphone mipos miposamt miinv
        miinvbal micc miccbal miccpurc
        miincome mihmown milores mihmval
        miage micrscor;
run;
ods trace off;
```

Partial Output

Oblique Principal Component Cluster Analysis

Output Added:

```

-----
Name:      DataOptSummary
Label:     Data and Options Summary
Template:  Stat.Varclus.DataOptSummary
Path:     Varclus.DataOptSummary
-----

```

Observations	32264	Proportion	0
Variables	64	Maxeigen	0.7

Output Added:

```

-----
Name:      ConvergenceStatus
Label:     Convergence Status
Template:  Stat.Varclus.ConvergenceStatus
Path:     Varclus.GROUP.ConvergenceStatus
-----

```

Clustering algorithm converged.

Output Added:

```

-----
Name:      ClusterSummary
Label:     Cluster Summary
Template:  Stat.Varclus.ClusterSummary
Path:     Varclus.GROUP.ClusterSummary
-----

```

Cluster Summary for 1 Cluster

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	64	64	9.228833	0.1442	5.0980

Total variation explained = 9.228833 Proportion = 0.1442

Cluster 1 will be split because it has the largest second eigenvalue, 5.097981, which is greater than the MAXEIGEN=0.7 value.

Output Added:

```
-----
Name:      ConvergenceStatus
Label:     Convergence Status
Template:  Stat.Varclus.ConvergenceStatus
Path:     Varclus.GROUP.ConvergenceStatus
-----
```

Oblique Principal Component Cluster Analysis

Clustering algorithm converged.

Output Added:

```
-----
Name:      ClusterSummary
Label:     Cluster Summary
Template:  Stat.Varclus.ClusterSummary
Path:     Varclus.GROUP.ClusterSummary
-----
```

Cluster Summary for 2 Clusters

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	48	48	9.197492	0.1916	3.4316
2	16	16	5.047174	0.3154	2.0471

Total variation explained = 14.24467 Proportion = 0.2226

Output Added:

Name: RSquare
 Label: R-squared
 Template: Stat.Varclus.RSquare
 Path: Varclus.GROUP.RSquare

2 Clusters		R-squared with			Variable Label
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	
Cluster 1	AcctAge	0.0004	0.0002	0.9998	Age of Oldest Account
	Dep	0.0010	0.0001	0.9991	Checking Deposits
	DepAmt	0.0007	0.0000	0.9993	Amount Deposited
	CashBk	0.0003	0.0000	0.9997	Number Cash Back
	Checks	0.0037	0.0001	0.9963	Number of Checks
	DirDep	0.0001	0.0001	0.9999	Direct Deposit
	NSF	0.0002	0.0001	0.9999	Number Insufficient Fund
	NSFAmt	0.0000	0.0000	1.0000	Amount NSF
	Phone	0.0179	0.0001	0.9821	Number Telephone Banking
	Teller	0.0012	0.0001	0.9989	Teller Visits
	Sav	0.0006	0.0000	0.9994	Saving Account
	SavBal	0.0000	0.0000	1.0000	Saving Balance
	ATM	0.0003	0.0000	0.9997	ATM
	ATMAmt	0.0013	0.0000	0.9987	ATM Withdrawal Amount
	POS	0.0257	0.0000	0.9743	Number Point of Sale
	POSAmt	0.0242	0.0001	0.9759	Amount Point of Sale

Oblique Principal Component Cluster Analysis

2 Clusters		R-squared with			Variable Label
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	
	CD	0.0002	0.0000	0.9998	Certificate of Deposit
	CDBal	0.0002	0.0000	0.9998	CD Balance
	LOC	0.0066	0.0000	0.9934	Line of Credit
	LOCBal	0.0014	0.0001	0.9987	Line of Credit Balance
	Inv	0.0052	0.0001	0.9949	Investment
	InvBal	0.0002	0.0000	0.9999	Investment Balance
	ILS	0.0026	0.0000	0.9974	Installment Loan
	ILSBal	0.0024	0.0000	0.9977	Loan Balance
	MM	0.0000	0.0000	1.0000	Money Market
	MMBal	0.0001	0.0000	1.0000	Money Market Balance
	MTG	0.0032	0.0003	0.9970	Mortgage
	MTGBal	0.0006	0.0000	0.9995	Mortgage Balance
	CC	0.1279	0.0007	0.8727	Credit Card
	CCBal	0.0019	0.0000	0.9981	Credit Card Balance
	CCPurc	0.0211	0.0000	0.9789	Credit Card Purchases
	SDB	0.0006	0.0000	0.9995	Safety Deposit Box
	CRScore	0.0001	0.0000	0.9999	Credit Score
	Moved	0.0000	0.0000	1.0000	Recent Address Change
	InArea	0.0003	0.0000	0.9997	Local Address
	brclus1	0.2239	0.0535	0.8200	

2 Clusters		R-squared with			Variable Label
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	
	brclus2	0.5331	0.0022	0.4679	
	brclus3	0.0061	0.0019	0.9958	
	brclus4	0.2426	0.0010	0.7582	
	MIAcctAg	0.0000	0.0000	1.0000	
	MIPhone	0.9925	0.0044	0.0076	
	MIPOS	0.9925	0.0044	0.0076	
	MIPOSAmt	0.9925	0.0044	0.0076	
	MIInv	0.9925	0.0044	0.0076	
	MIInvBal	0.9925	0.0044	0.0076	
	MICC	0.9925	0.0044	0.0076	
	MICCBal	0.9925	0.0044	0.0076	
	MICCPurc	0.9925	0.0044	0.0076	

Cluster 2	DDA	0.0002	0.0001	0.9999	Checking Account
	DDABal	0.0004	0.0001	0.9997	Checking Balance
	IRA	0.0000	0.0000	1.0000	Retirement Account
	IRABal	0.0001	0.0000	0.9999	IRA Balance
	MMCred	0.0000	0.0000	1.0000	Money Market Credits
	Income	0.0113	0.0000	0.9887	Income
	HMOwn	0.1850	0.0000	0.8150	Owns Home
	LORes	0.0005	0.0001	0.9996	Length of Residence
	HMVal	0.0137	0.0000	0.9863	Home Value
	Age	0.0009	0.0000	0.9991	Age
	MIIncome	0.9857	0.0028	0.0143	
	MIHMOwn	0.9541	0.0030	0.0461	
	MILORes	0.9857	0.0028	0.0143	
	MIHMVal	0.9857	0.0028	0.0143	
	MIAge	0.9194	0.0027	0.0808	
	MICRScor	0.0044	0.0000	0.9956	

Cluster 1 will be split because it has the largest second eigenvalue, 3.431637, which is greater than the MAXEIGEN=0.7 value.

The output object RSQUARE contains the results of each iteration of the algorithm except the first one (the cluster with all of the variables). To print out the last iteration, you need the number of clusters in the last iteration.

Partial Output (continued)

Output Added:

 Name: ClusterQuality
 Label: Cluster Quality
 Template: Stat.Varclus.ClusterQuality
 Path: Varclus.ClusterQuality

Number of Clusters	Total Variation Explained by Clusters	Proportion of Variation Explained by Clusters	Minimum Proportion Explained by a Cluster	Maximum Second Eigenvalue in a Cluster	Minimum R-squared for a Variable	Maximum 1-R**2 Ratio for a Variable
1	9.228833	0.1442	0.1442	5.097981	0.0000	
2	14.244666	0.2226	0.1916	3.431637	0.0000	1.0000
3	17.595965	0.2749	0.1269	2.541211	0.0000	1.4572
4	20.024110	0.3129	0.1269	2.104444	0.0000	1.4572
5	21.979822	0.3434	0.1802	2.047135	0.0000	1.3719
6	23.961430	0.3744	0.1802	2.013311	0.0000	1.3416
7	25.945084	0.4054	0.2023	1.623838	0.0000	1.3428
8	27.446233	0.4288	0.2375	1.449194	0.0000	1.5083
9	28.836743	0.4506	0.2375	1.361978	0.0000	1.5083
10	30.185479	0.4716	0.2504	1.346719	0.0000	1.5083
11	31.531743	0.4927	0.2504	1.293080	0.0003	1.5083
12	32.718882	0.5112	0.2740	1.282023	0.0003	1.4923
13	33.985146	0.5310	0.2915	1.208224	0.0005	1.4923
14	35.066412	0.5479	0.3231	1.161744	0.0005	1.3942
15	36.192826	0.5655	0.3231	1.133198	0.0005	1.1289
16	37.318009	0.5831	0.3231	1.052496	0.0005	1.1289
17	38.322869	0.5988	0.3334	1.051776	0.0022	1.1053
18	39.316470	0.6143	0.3334	1.047393	0.0022	1.1053
19	40.198608	0.6281	0.3334	1.047389	0.0022	1.1053
20	41.003733	0.6407	0.3334	1.008468	0.0022	1.1053
21	42.004913	0.6563	0.3634	1.001662	0.0022	1.1053
22	43.004187	0.6719	0.3634	1.000350	0.0387	1.1053
23	43.994441	0.6874	0.3634	0.993723	0.0387	1.1006
24	44.988164	0.7029	0.3634	0.980590	0.0387	1.1006
25	45.968754	0.7183	0.3634	0.979639	0.0891	1.1006
26	46.820328	0.7316	0.3634	0.975378	0.0891	1.1006
27	47.795707	0.7468	0.3634	0.975309	0.0891	1.1006
28	48.578553	0.7590	0.3634	0.970232	0.0891	1.1006
29	49.547773	0.7742	0.3634	0.955086	0.1885	1.1006
30	50.502860	0.7891	0.3634	0.946039	0.1885	1.1006
31	51.415705	0.8034	0.4554	0.944270	0.1885	1.1006
32	52.303710	0.8172	0.5257	0.874419	0.1885	1.1006
33	53.176667	0.8309	0.5257	0.858297	0.1885	1.1006
34	54.034965	0.8443	0.5257	0.844004	0.1885	1.1006
35	54.878124	0.8575	0.5257	0.839114	0.2933	0.8066
36	55.717238	0.8706	0.5257	0.824195	0.2933	0.8066
37	56.538607	0.8834	0.6174	0.765162	0.4055	0.6283
38	57.303769	0.8954	0.6271	0.745734	0.4055	0.6283
39	58.049504	0.9070	0.6322	0.735552	0.4055	0.6283
40	58.785055	0.9185	0.6414	0.732613	0.4055	0.6283
41	59.513555	0.9299	0.6414	0.687465	0.4865	0.5586

The variable **NumberOfClusters** in the output object ClusterQuality contains the number of clusters for the final iteration. The output object also has information on the proportion of variation explained by the clusters and the maximum second eigenvalue in a cluster. This information can be used to decide whether too few or too many clusters have been formed.

The ODS OUTPUT statement creates a SAS data set named **Summary** from the output object ClusterQuality, and a SAS data set named **Clusters** from each RSQUARE output object. Because there are 40 RSQUARE objects created (one for the 2-cluster solution, one for the 3-cluster solution, and so on up to the 41-cluster solution), the **Clusters** data set concatenates these 40 objects. There is a column called **NumberOfClusters** that indicates which cluster solution each observation in the **Clusters** data set belongs to.

The CALL SYMPUT routine creates the macro variable **nvar**, which contains the value of the number of clusters in the last iteration of the clustering algorithm. The COMPRESS function strips blanks from variables. Because the **Clusters** data set contains the results of all 41-cluster solutions, the **nvar** macro variable is used to restrict focus to the final result of the VARCLUS algorithm; here, this is the 41-cluster solution. The **nvar** macro variable will be useful later, because if you select one representative from each variable cluster, you will have **nvar** inputs for future modeling consideration.

```
ods listing close;
ods output clusterquality=summary
           rsquare=clusters;

proc varclus data=imputed
           maxeigen=.7
           short
           hi;
    var &inputs brclus1-brclus4 miacctag
        miphone mipos miposamt miinv
        miinvbal micc miccbal miccpurc
        miincome mihmown milores mihmval
        miage micrscor;
run;
ods listing;

data _null_;
    set summary;
    call symput('nvar',compress(NumberOfClusters));
run;

proc print data=clusters noobs;
    where NumberOfClusters=&nvar;
    var Cluster Variable RSquareRatio VariableLabel;
run;
```

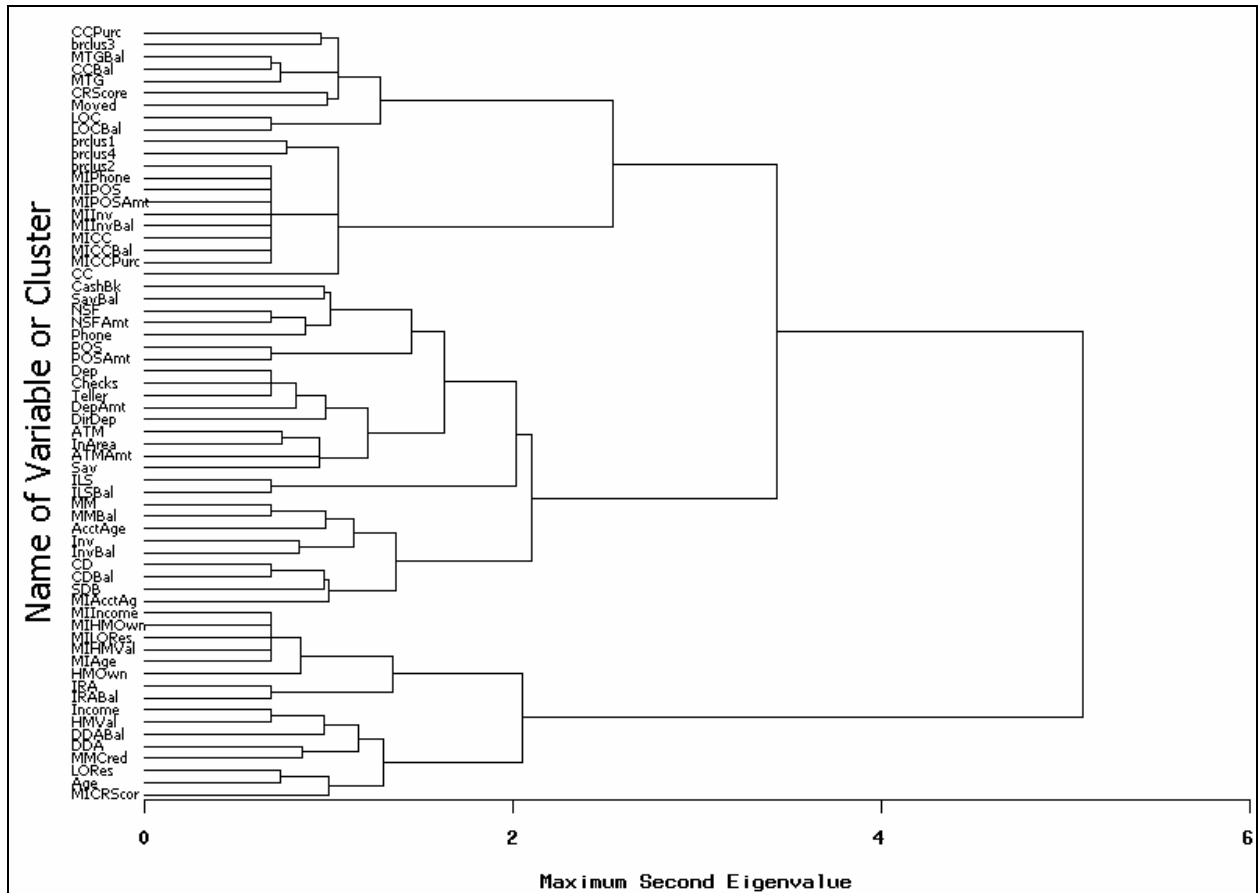

Cluster	Variable	RSquare Ratio	VariableLabel
Cluster 1	brclus2	0.4901	
	MIPhone	0.0050	
	MIPOS	0.0050	
	MIPOSamt	0.0050	
	MIInv	0.0050	
	MIInvBal	0.0050	
	MICC	0.0050	
	MICCBal	0.0050	
	MICCPurc	0.0050	
Cluster 2	MIIncome	0.0074	
	MIHMOwn	0.0431	
	MILORes	0.0074	
	MIHMVal	0.0074	
	MIAge	0.0832	
Cluster 3	Dep	0.4235	Checking Deposits
	Checks	0.3383	Number of Checks
	Teller	0.5586	Teller Visits
Cluster 4	MTGBal	0.0390	Mortgage Balance
	CCBal	0.0361	Credit Card Balance
Cluster 5	MM	0.0360	Money Market
	MMBal	0.0348	Money Market Balance
Cluster 6	Income	0.1625	Income
	HMVal	0.2332	Home Value
Cluster 7	ILS	0.0044	Installment Loan
	ILSBal	0.0044	Loan Balance
Cluster 8	POS	0.0873	Number Point of Sale
	POSamt	0.0864	Amount Point of Sale
Cluster 9	NSF	0.2541	Number Insufficient Fund
	NSFamt	0.2503	Amount NSF
Cluster 10	CD	0.2816	Certificate of Deposit
	CDBal	0.2823	CD Balance
Cluster 11	IRA	0.3426	Retirement Account
	IRABal	0.3341	IRA Balance
Cluster 12	Age	0.0000	Age
Cluster 13	LOC	0.2740	Line of Credit
	LOCBal	0.2827	Line of Credit Balance
Cluster 14	Sav	0.0000	Saving Account
Cluster 15	DDA	0.0000	Checking Account
Cluster 16	InvBal	0.0000	Investment Balance
Cluster 17	CRScore	0.0000	Credit Score
Cluster 18	brclus3	0.0000	
Cluster 19	CC	0.0000	Credit Card
Cluster 20	brclus1	0.0000	
Cluster 21	CashBk	0.0000	Number Cash Back
Cluster 22	MIAcctAg	0.0000	
Cluster 23	MICRScor	0.0000	
Cluster 24	Moved	0.0000	Recent Address Change
Cluster 25	AcctAge	0.0000	Age of Oldest Account
Cluster 26	DirDep	0.0000	Direct Deposit
Cluster 27	SavBal	0.0000	Saving Balance

Cluster	Variable	RSquare Ratio	VariableLabel
Cluster 28	DDABal	0.0000	Checking Balance
Cluster 29	SDB	0.0000	Safety Deposit Box
Cluster 30	CCPurc	0.0000	Credit Card Purchases
Cluster 31	InArea	0.0000	Local Address
Cluster 32	ATMAmt	0.0000	ATM Withdrawal Amount
Cluster 33	Phone	0.0000	Number Telephone Banking
Cluster 34	MMCred	0.0000	Money Market Credits
Cluster 35	HMOwn	0.0000	Owens Home
Cluster 36	Inv	0.0000	Investment
Cluster 37	DepAmt	0.0000	Amount Deposited
Cluster 38	brclus4	0.0000	
Cluster 39	ATM	0.0000	ATM
Cluster 40	LORes	0.0000	Length of Residence
Cluster 41	MTG	0.0000	Mortgage

The output shows the cluster number, the names of the variables in each cluster, and the $1-R^2$ ratio (**RSquareRatio**).

PROC TREE draws a dendrogram of the divisive variable clustering process. The AXIS statements are used to control the plotting and printing of axes and their labels. The HORIZONTAL option specifies that the dendrogram should be plotted horizontally. The HAXIS= and VAXIS= statements tie the axis definitions in the AXIS statements to the axes of the plot. The HEIGHT statement plots the second eigenvalue on the horizontal axis. For example, when a principal components analysis is done on all the variables (the first cluster), the second eigenvalue is 5.10. When this cluster is split, the second eigenvalue for one child cluster is 3.17 and 2.06 for the other.

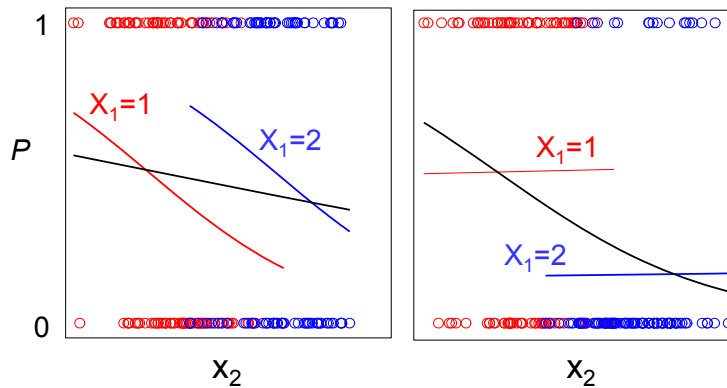
```
axis1 value=(font = tahoma color=blue rotate=0 height=.8)
      label=(font = tahoma angle=90 height=2);
axis2 order=(0 to 6 by 2);
proc tree data=fortree
      horizontal
      vaxis=axis1
      haxis=axis2;
      height _MAXEIG_;
run;
```



To reduce the amount of text that needs to be entered, a new macro variable named **Reduced** is created that contains the names of all the variables selected from PROC VARCLUS. As was mentioned above, subject-matter considerations should play a part in deciding which inputs become the cluster representatives. Consider the first cluster, which consists of the variables **brclus2**, **MIphone**, **MIPOS**, **MIPOSamt**, **MIInv**, **MIInvBal**, **MICC**, **MICCBal**, and **MICCPurc**. According to the 1-R-Square ratio, any of the missing indicators would be the best cluster representative. However, closer investigation of the development data set might reveal that the missingness of those eight inputs has more to do with poor data quality than any other source of missing data. If these missing indicators in actuality indicate poor data quality, then their usefulness as predictors with good generalizing power is suspect. With that subject-matter consideration in mind, perhaps **brclus2** is the best cluster representative from cluster 1.

```
%let reduced=
brclus2 miincome checks ccbal
mmbal income ilsbal posamt
nsfamt cd irabal age
loc sav dda invbal
crscore brclus3 cc brclus1
cashbk miacctag micrscor moved
acctage dirdep savbal ddabal
sdb ccpurc inarea atmamt
phone mmcred hmown inv
depamt brclus4 atm lores
mtg;
```

Univariate Screening



51

It is tempting to use univariate associations to detect irrelevant input variables. Each input variable is screened individually versus the target (chi-squared tests, correlation coefficients, two-sample tests, and so on). Only the most important inputs are retained in the analysis. This method does not account for partial associations among the inputs. Inputs could be erroneously omitted or erroneously included. Partial association occurs when the effect of one variable changes in the presence of another variable. Multivariate methods that consider subsets of variables jointly are preferable. The best k inputs in a univariate sense would not necessarily be the best k -element subset. The presence of interactions can also give misleading univariate associations.



Variable Screening

Even after variable clustering, some further variable reduction may be needed prior to using the variable selection techniques in the LOGISTIC procedure. Very liberal univariate screening may be helpful when the number of clusters created in the VARCLUS procedure is still relatively large. Because some of the variable selection techniques use the full model, eliminating clearly irrelevant variables (for example, p -values greater than .50) will stabilize the full model and may improve the variable selection technique without much risk of eliminating important input variables. Keep in mind that univariate screening can give misleading results when there are partial associations. This problem is minimized because the screening is done after PROC VARCLUS, and is used in eliminating clearly irrelevant variables, rather than searching for the best predictors.

The CORR procedure can be used for univariate screening. The SPEARMAN option requests Spearman correlation statistics, which is a correlation of the ranks of the input variables with the binary target. The Spearman correlation statistic was used rather than the Pearson correlation statistic because Spearman is less sensitive to nonlinearities and outliers. However, when variables are not monotonically related to each other, the Spearman correlation statistic can miss important associations. A general and robust similarity measure is Hoeffding's D (requested by the Hoeffding option) which will detect a wide variety of associations between two variables. Hoeffding's D statistic has values between -0.5 to 1 , but if there are many ties, smaller values may result. The RANK option prints the correlation coefficients for each variable in order from highest to lowest.

A useful table (or plot) would compare the rank order of the Spearman correlation statistic to the rank order of the Hoeffding's D statistic. If the Spearman rank is high but the Hoeffding's D rank is low, then the association is probably not monotonic. Empirical logit plots could be used to investigate this type of relationship.

The output object for the table of Spearman correlation statistics is called SPEARMANCORR, and the output object for the table of Hoeffding's D statistics is called HoeffdingCORR.

```
ods listing close;
ods output spearmancorr=spearman
           hoeffdingcorr=hoeffding;

proc corr data=imputed spearman hoeffding rank;
  var &reduced;
  with ins;
run;

ods listing;
```

The variable names in the SAS data sets **Spearman** and **Hoeffding** are in the variables **Best1** through **Best41**, the correlation statistics are in the variables **R1** through **R41**, and the p -values are in the variables **P1** through **P41**. The macro variable **nvar** was created in the last demonstration. It points to the number of clusters in the final cluster solution from the VARCLUS procedure (here it is 41). In order to make a more useful table, a DATA step is used to transpose the variables to observations.

```
data spearman1(keep=variable scorr spvalue ranksp);
    length variable $ 8;
    set spearman;
    array best(*) best1--best&nvar;
    array r(*) r1--r&nvar;
    array p(*) p1--p&nvar;
    do i=1 to dim(best);
        variable=best(i);
        scorr=r(i);
        spvalue=p(i);
        ranksp=i;
        output;
    end;
run;

data hoeffding1(keep=variable hcorr hpvalue rankho);
    length variable $ 8;
    set hoeffding;
    array best(*) best1--best&nvar;
    array r(*) r1--r&nvar;
    array p(*) p1--p&nvar;
    do i=1 to dim(best);
        variable=best(i);
        hcorr=r(i);
        hpvalue=p(i);
        rankho=i;
        output;
    end;
run;
```

The two data sets are then sorted by the variable names and merged by the variable names.

```
proc sort data=spearman1;
    by variable;
run;

proc sort data=hoeffding1;
    by variable;
run;

data correlations;
    merge spearman1 hoeffding1;
    by variable;
run;
```

The final data set is sorted by the rank of the Spearman correlation statistic and then a table is generated showing the rank and associated statistics of the Spearman correlations and Hoeffding's D statistics.

```
proc sort data=correlations;
  by ranksp;
run;

proc print data=correlations label split='*';
  var variable ranksp rankho scorr spvalue hcorr hpvalue;
  label ranksp = 'Spearman rank*of variables'
        scorr = 'Spearman Correlation'
        spvalue = 'Spearman p-value'
        rankho = 'Hoeffding rank*of variables'
        hcorr = 'Hoeffding Correlation'
        hpvalue = 'Hoeffding p-value';
run;
```

Obs	variable	Spearman rank of variables	Hoeffding rank of variables	Spearman Correlation	Spearman p-value	Hoeffding Correlation	Hoeffding p-value
1	SavBal	1	1	0.25031	0.00000	0.009764437	0.00001
2	CD	2	6	0.20264	0.00000	0.001876987	0.00001
3	DDA	3	5	-0.18493	0.00000	0.002144155	0.00001
4	MMBal	4	11	0.16066	0.00000	0.001117737	0.00001
5	Sav	5	3	0.15164	0.00000	0.002395734	0.00001
6	CC	6	4	0.14683	0.00000	0.002196920	0.00001
7	ATM	7	7	-0.12247	0.00000	0.001480339	0.00001
8	IRABal	8	18	0.10995	0.00000	0.000198562	0.00001
9	Phone	9	14	-0.10400	0.00000	0.000615874	0.00001
10	Inv	10	22	0.09838	0.00000	0.000059608	0.00761
11	MMCred	11	21	0.09509	0.00000	0.000119558	0.00025
12	Checks	12	9	-0.09015	0.00000	0.001239866	0.00001
13	brclus1	13	12	0.08189	0.00000	0.000638991	0.00001
14	CCPurc	14	17	0.08173	0.00000	0.000240566	0.00001
15	POSAmt	15	15	-0.07804	0.00000	0.000445904	0.00001
16	InvBal	16	28	0.07277	0.00000	-0.000011379	0.83466
17	SDB	17	19	0.07256	0.00000	0.000175689	0.00001
18	CCBal	18	13	0.07147	0.00000	0.000632818	0.00001
19	DirDep	19	16	-0.07037	0.00000	0.000402104	0.00001
20	ATMAmt	20	8	-0.06809	0.00000	0.001410049	0.00001
21	NSFAmt	21	20	-0.06804	0.00000	0.000119878	0.00025
22	InArea	22	24	-0.06049	0.00000	0.000015869	0.11424
23	brclus4	23	25	-0.06044	0.00000	0.000006379	0.22409
24	brclus2	24	23	-0.05955	0.00000	0.000055090	0.00993
25	DDABal	25	2	0.05611	0.00000	0.003101822	0.00001
26	DepAmt	26	10	-0.04369	0.00000	0.001179781	0.00001
27	CashBk	27	33	-0.04159	0.00000	-0.000033143	1.00000
28	brclus3	28	31	-0.03224	0.00000	-0.000022896	1.00000
29	Income	29	26	0.01441	0.00966	0.000002916	0.29081
30	ILSBal	30	35	-0.01413	0.01113	-0.000037882	1.00000
31	Age	31	27	0.01252	0.02457	-0.000000997	0.39375
32	micrscor	32	40	0.00895	0.10779	-0.000043281	1.00000
33	miacctag	33	37	0.00533	0.33881	-0.000041551	1.00000
34	Moved	34	41	-0.00466	0.40294	-0.000043400	1.00000
35	AcctAge	35	29	-0.00459	0.40928	-0.000015364	0.97453

36	LORes	36	32	0.00408	0.46371	-.000026362	1.00000
37	CRScore	37	30	0.00407	0.46454	-.000017633	0.99778
38	LOC	38	38	0.00232	0.67642	-.000042162	1.00000
39	miincome	39	36	0.00159	0.77572	-.000038195	1.00000
40	HMOwn	40	34	-0.00060	0.91437	-.000034339	1.00000
41	MTG	41	39	-0.00042	0.93941	-.000042850	1.00000

A graphical representation of this table may aid decision making. The PLOT procedure is used to create a scatter plot of Spearman ranks against Hoeffding ranks. For ease of interpretation, the variable names are placed next to the point markers.

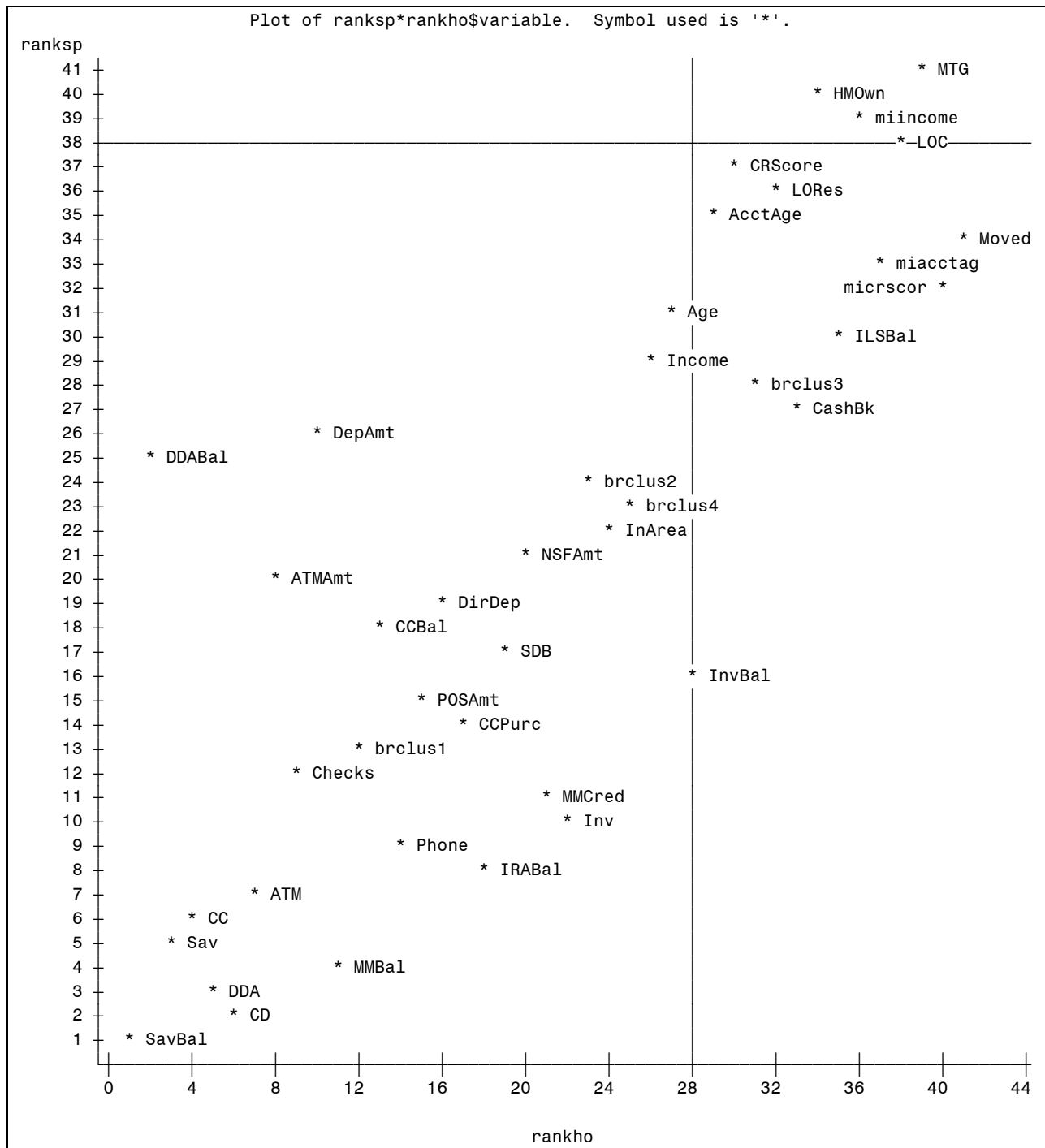


In order to have reference lines on the plot, the SQL procedure is used to create macro variables that point to the smallest Spearman rank and Hoeffding rank associated with a p -value greater than 0.5. This is just for presentation.

In general, the upper-right corner of the plot contains the names of variables that could reasonably be excluded from further analysis, due to their poor rank on both metrics. The criterion to use in eliminating variables is a subjective decision. Thus, four variables are eliminated from the analysis.

```
proc sql noprint;
  select min(ranksp) into :vref
  from (select ranksp
        from correlations
        having spvalue > .5);
  select min(rankho) into :href
  from (select rankho
        from correlations
        having hpvalue > .5);
quit;

proc plot data = correlations;
  plot ranksp*rankho $ variable="*"
    /vref=&vref href=&href vpos=&nvar;
run; quit;
```

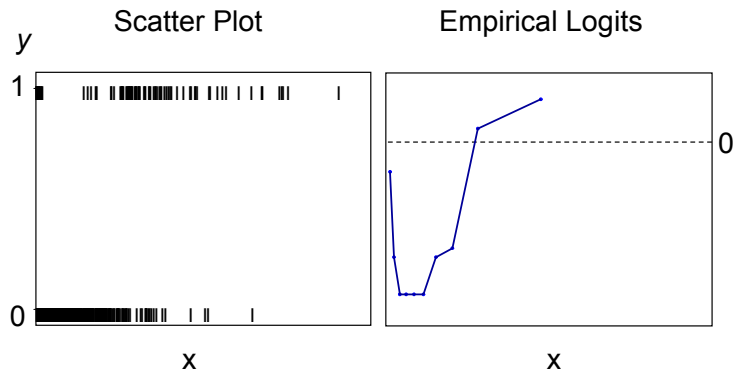



In addition to being a useful screening tool, this analysis may point toward further analyses. High ranks for Spearman and low ranks for Hoeffding's D are found for the variables **DDABal**, **DepAmt**, and **ATMAmt**. Even though these variables do not have a monotonic relationship with **Ins**, some other type of relationship is detected by Hoeffding's D statistic. Empirical logit plots should be used to examine these relationships.

The %LET statement creates a macro variable called **screened** that has the names of the variables remaining after the univariate screening method.

```
%let screened =  
brclus2 checks ccbal  
mmbal income ilsbal posamt  
nsfamt cd irabal age  
sav dda invbal  
crscore brclus3 cc brclus1  
cashbk miacctag micrscor moved  
acctage dirdep savbal ddabal  
sdb ccpurc inarea atmamt  
phone mmcred inv  
depamt brclus4 atm lores;
```

Univariate Smoothing



55

In regression analysis, it is standard practice to examine scatter plots of the target versus each input variable. When the target is binary, these plots are not very enlightening. A useful plot to detect nonlinear relationships is a plot of the empirical logits.

Clustering Levels

	0	1		0	1
A	28	7		28	7
B	16	0	→	110	11
C	94	11		23	21
D	23	21			

Merged: B & C

$\chi^2 =$	31.7	30.7
	100%	97%

24

...

Univariate plots of binary data need to be smoothed. A simple, scalable, and robust smoothing method is to plot empirical logits for quantiles of the input variables. These logits use a minimax estimate of the proportion of events in each bin (Duffy and Santner 1989). This eliminates the problem caused by zero counts and reduces variability.

The number of bins determines the amount of smoothing (for example, the fewer bins, the more smoothing). One large bin would give a constant logit. For very large data sets and intervally scaled inputs, 100 bins often works well. If the standard logistic model were true, then the plots should be linear. Sample variability can cause apparent deviations, particularly when the bin size is too small. However, serious nonlinearities, such as nonmonotonicity, are usually easy to detect.



Empirical Logit Plots

To create a plot of the empirical logits versus a continuous input variable, the input variable first needs to be binned. Use the RANK procedure with the GROUPS= option to bin variables automatically. The bins will be equal size (quantiles) except when the number of tied values exceeds the bin size, in which case the bin will be enlarged to contain all the tied values. The VAR statement lists the variable in the DATA= data set to be grouped. The RANKS statement names the variable representing the groups in the OUT= data set. If the RANKS statement is not used, the VAR variable is replaced by the groups.

```
%let var=DDABal;

proc rank data=imputed groups=100 out=out;
    var &var;
    ranks bin;
run;

proc print data=out(obs=10);
    var &var bin;
run;
```

	Obs	DDABal	bin
	1	419.27	44
	2	1986.81	76
	3	0.00	9
	4	1594.84	72
	5	2813.45	82
	6	1069.78	63
	7	1437.57	69
	8	1683.28	73
	9	190.03	33
	10	462.12	46

To compute the empirical logit, the number of target event cases (**ins=1**) and total cases in each bin needs to be computed. The empirical logits are plotted against the mean of the input variable in each bin. This needs to be computed as well. Both tasks can be done in the MEANS procedure using the CLASS statement. The appropriate statistics (SUM and MEAN) need to be specified in the OUTPUT statement.

```
proc means data=out noprint nway;
    class bin;
    var ins &var;
    output out=bins sum(ins)=ins mean(&var)=&var;
run;

proc print data=bins(obs=10);
run;
```

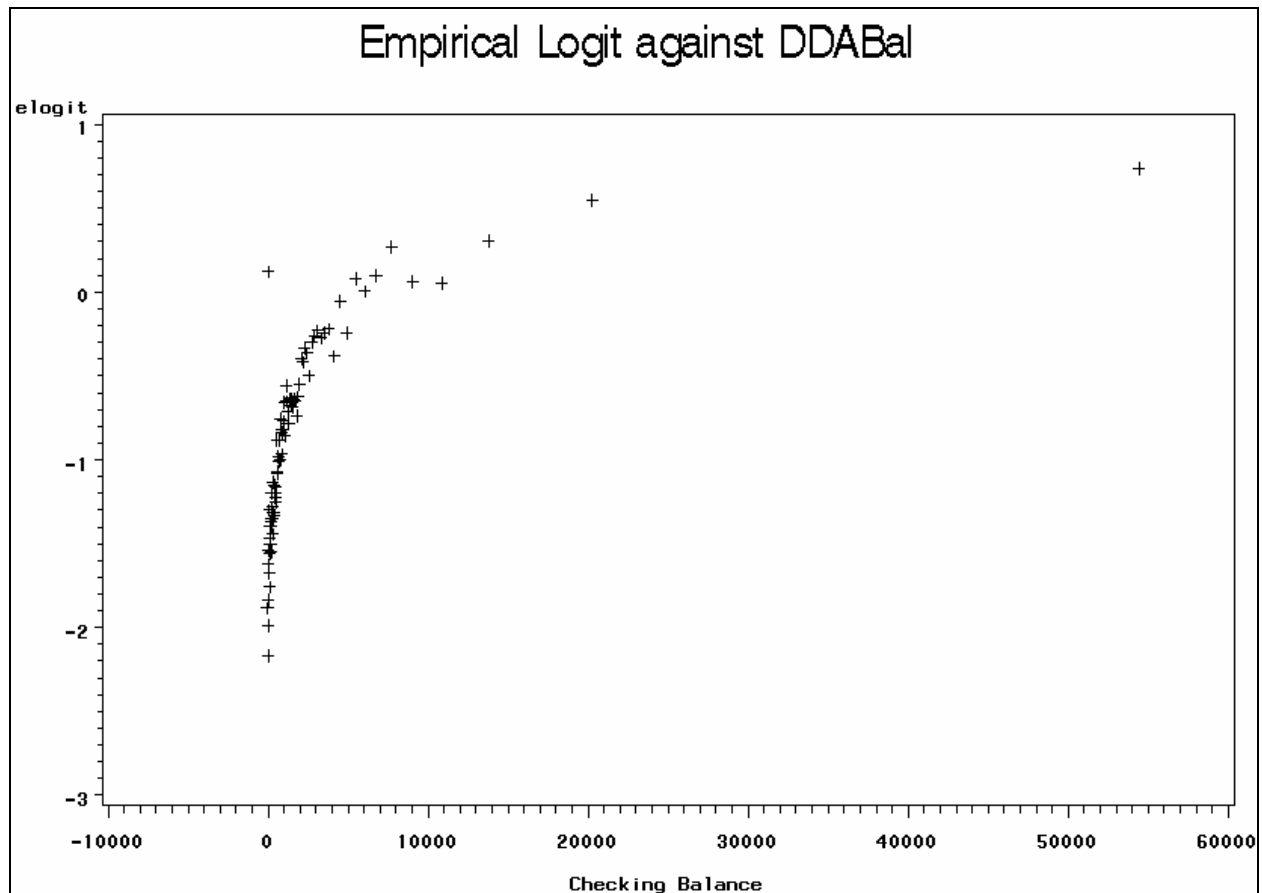
	Obs	bin	_TYPE_	_FREQ_	ins	DDABal
	1	0	1	198	21	-31.8287
	2	9	1	5987	3179	0.0000
	3	19	1	267	26	2.7617
	4	20	1	323	26	9.0509
	5	21	1	323	38	17.2157
	6	22	1	322	51	28.1966
	7	23	1	323	38	40.9160
	8	24	1	322	47	53.4157
	9	25	1	323	45	66.7146
	10	26	1	323	64	81.9273

The variable **ins** contains the number of events while the automatic variable **_FREQ_** contains the bin size.

```
data bins;
  set bins;
  elogit=log((ins+(sqrt(_FREQ_)/2))/
             (_FREQ_-ins+(sqrt(_FREQ_)/2)));
run;
```

The empirical logits can be plotted using the GPLOT procedure. It is easy to get a simple plot. The statement **PLOT Y * X** requests that the variable **Y** be plotted on the vertical axis and that the variable **X** be plotted on the horizontal axis. The default picture is a scatter plot of **Y** by **X**. The following code plots **elogit** against **&var**, here **DDABal**.

```
proc gplot data = bins;
  title "Empirical Logit against &var";
  plot elogit * &var;
run; quit;
```



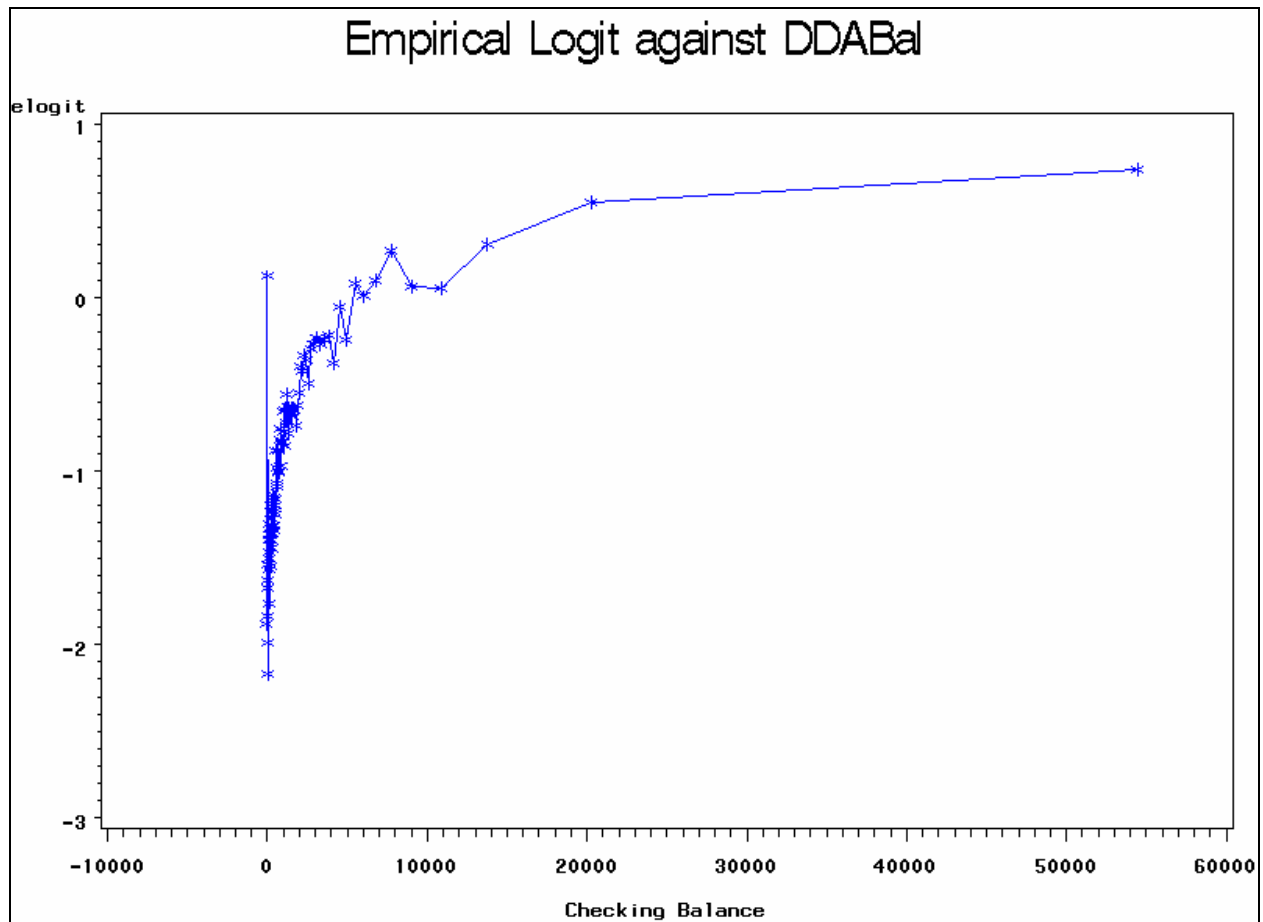
Using a few more commands², it is possible to control many features of the plot. The code below uses a SYMBOL statement to produce the following:

- a line that interpolates between the points on the scatter plot (I=JOIN)
- a blue line (C=BLUE)
- stars as the plotting symbol (V=STAR).

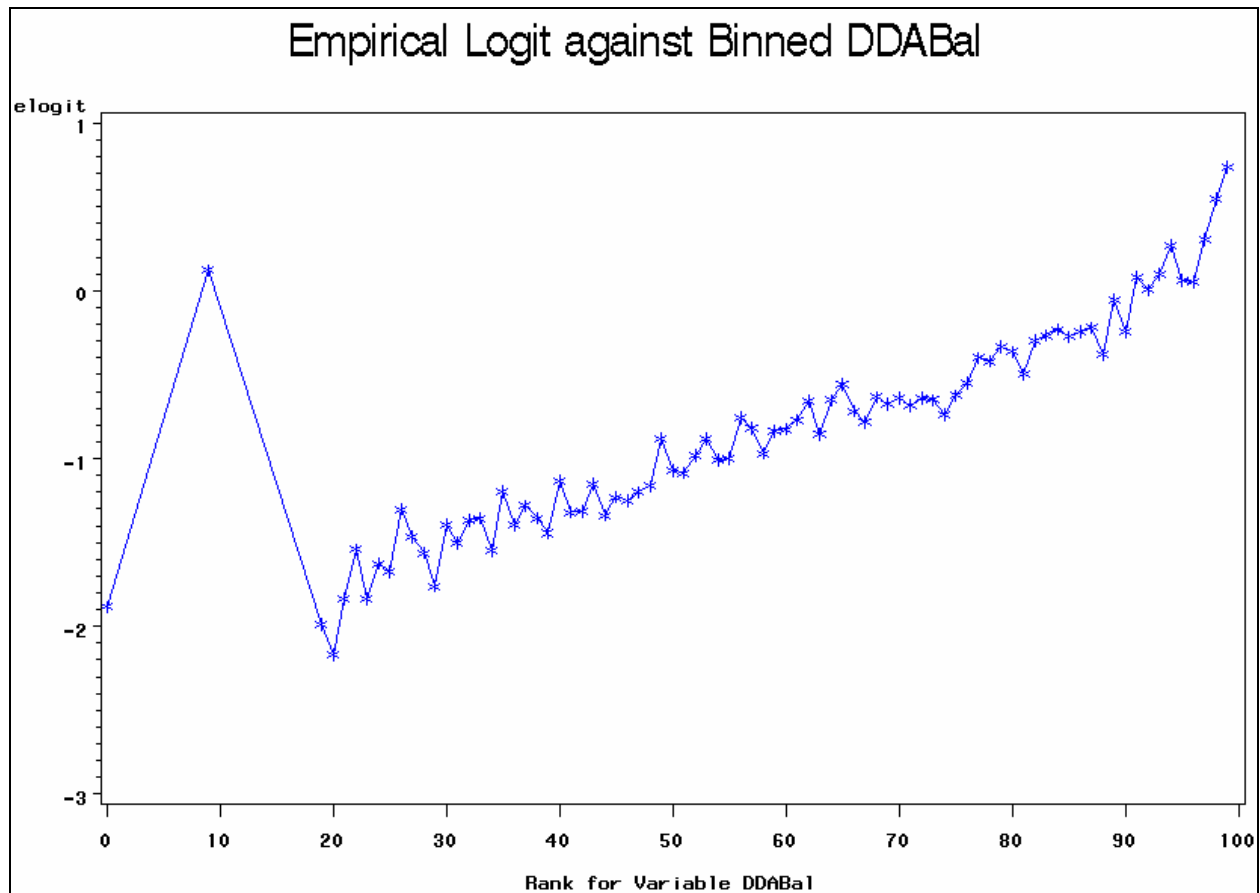
In addition to plotting the empirical logits against the DDA balance, this code plots the logits against bin number.

```
symbol i=join c=blue v=star;
proc gplot data = bins;
  title "Empirical Logit against &var";
  plot elogit * &var;
  title "Empirical Logit against Binned &var";
  plot elogit * bin;
run; quit;
```

² If you choose a point-and-click interface for creating graphs, the Graph-N-Go utility is supplied in SAS/GRAPH software. However, pointing and clicking will require much more user interaction.



The pattern made by plotting logit against checking account balance has two striking features. There is a spike in the logits at the \$0 balance level. Aside from that spike, the trend is monotonic but certainly not linear.



The pattern made by plotting logit against the rank of checking account balance seems to accommodate the nonlinearity of the last plot, but the spike indicates a portion of the population who are not behaving as their balance would lead one to believe. The trend in checking account balance is clear, and this is probably a very good input. How, though, can one account for the nonlinear relationship between response behavior and balance amount?

Clustering Levels

	0	1		0	1		0	1
A	28	7		28	7		138	18
B	16	0		110	11		23	21
C	94	11	→			→		
D	23	21		23	21			
Merged:	B & C			A & BC				
$\chi^2 =$	31.7			30.7			28.6	
	100%			97%			90%	

25

...

1. Skilled and patient data analysts can accommodate nonlinearities in a logistic regression model by transforming or discretizing the input variables. This can become impractical with high-dimensional data and increases the risk of overfitting.
2. Polynomial terms may improve model fit but hamper model interpretation and generalization. Quadratic terms and two-factor interactions can be useful additions to a modeler's tool kit but are no panacea. Higher-order polynomials are not reliable smoothers.
3. Methods such as classification trees, generalized additive models, projection pursuit, multivariate adaptive regression splines, radial basis function networks, and multilayer perceptrons are flexible alternatives to logistic regression (Hastie and Tibshirani 1990, Ripley 1996).
4. Standard (linear) logistic regression can produce powerful and useful classifiers even when the estimates of the posterior probabilities are poor. Often more flexible approaches do not show enough improvement to warrant the effort.



Accommodating Nonlinearities

In order to use the checking account balance, you might consider taking a logarithmic transformation, a square root transformation, or some other transformation in an attempt to linearize the relationship between the logit and the account balances. This linearization will require two steps. First, that spike at \$0 needs to be accounted for. Second, you can transform the balances to some scale that reflects the behavior exhibited in the data.

It seems suspicious that a large portion of the population has exactly \$0 in their checking accounts. Investigation (of the data set or of the people who constructed the data set) will show that most of the individuals with exactly \$0 balances do not have checking accounts. Their balances have been set to \$0 as part of the data pre-processing. This rule seems reasonable from a logical imputation standpoint. How can someone with no checking account have a checking balance? But it is clear from the logit plots that those individuals are behaving like people with much more than \$0 in their checking accounts. The CLASS statement yields results for each level of the **DDA** variable; the results of interest are the mean, minimum, and maximum for the **DDABal** and **Ins** variables.

```
title;
proc means data=imputed mean min max;
  class dda;
  var ddabal ins;
run;
```

From the output, it is clear that the individuals without checking accounts have had \$0 balances imputed for them, and those individuals respond at a higher rate than individuals with checking accounts. If this seems unreasonable, consider that the individuals without checking accounts presumably do their everyday banking with a different bank altogether, and treat this bank as an investment institution.

The MEANS Procedure							
Checking Account	N Obs	Variable	Label	Mean	Minimum	Maximum	
0	5948	DDABal	Checking Balance	0	0	0	
		Ins		0.5314391	0	1.0000000	
1	26316	DDABal	Checking Balance	2660.49	-774.8300000	278093.83	
		Ins		0.3045296	0	1.0000000	

There are several possible ways to account for this discrepancy between the **DDABal** value and the empirically observed response behavior. One of the most straightforward is to re-impute. The following code creates a macro variable mean that has the value of the checking account balances of those customers who actually have checking accounts. Then that mean is substituted for the \$0 balances³ of those who do not have checking account balances. The SELECT INTO statement in the SQL procedure creates the macro variable, and the following DATA step fills in the \$0 balances with the balance of customers who have their accounts at this bank.

³ Arguably, most, if not all, of these customers have checking accounts somewhere, and therefore do have balances.

```
proc sql;
    select mean(ddabal) into :mean
    from imputed where dda;
quit;
```

2660.489

```
data imputed;
    set imputed;
    if not dda then ddabal = &mean;
run;
```

To evaluate the effectiveness of this re-imputation, take another look at the logit plots.

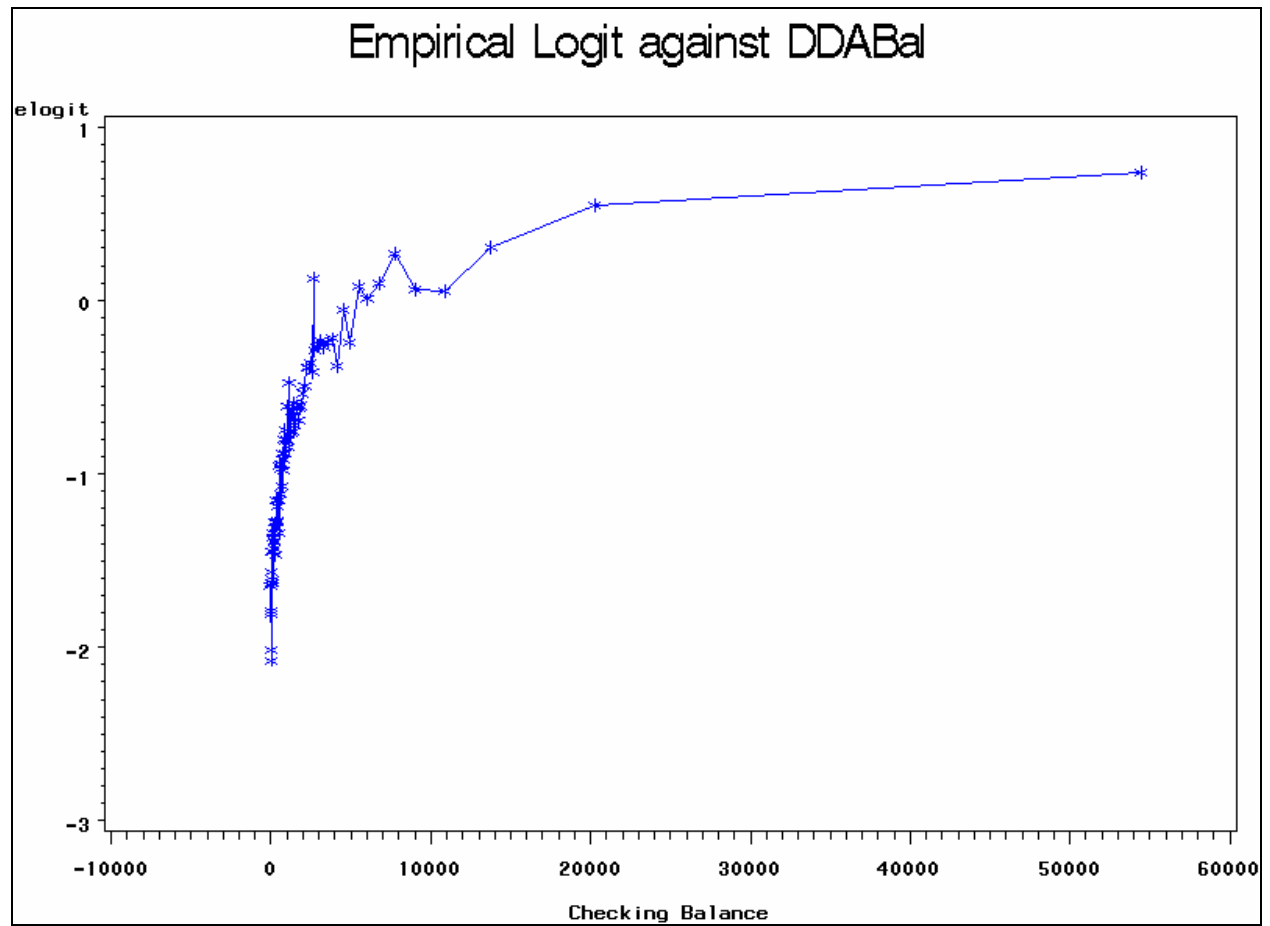
```
%let var=DDABal;

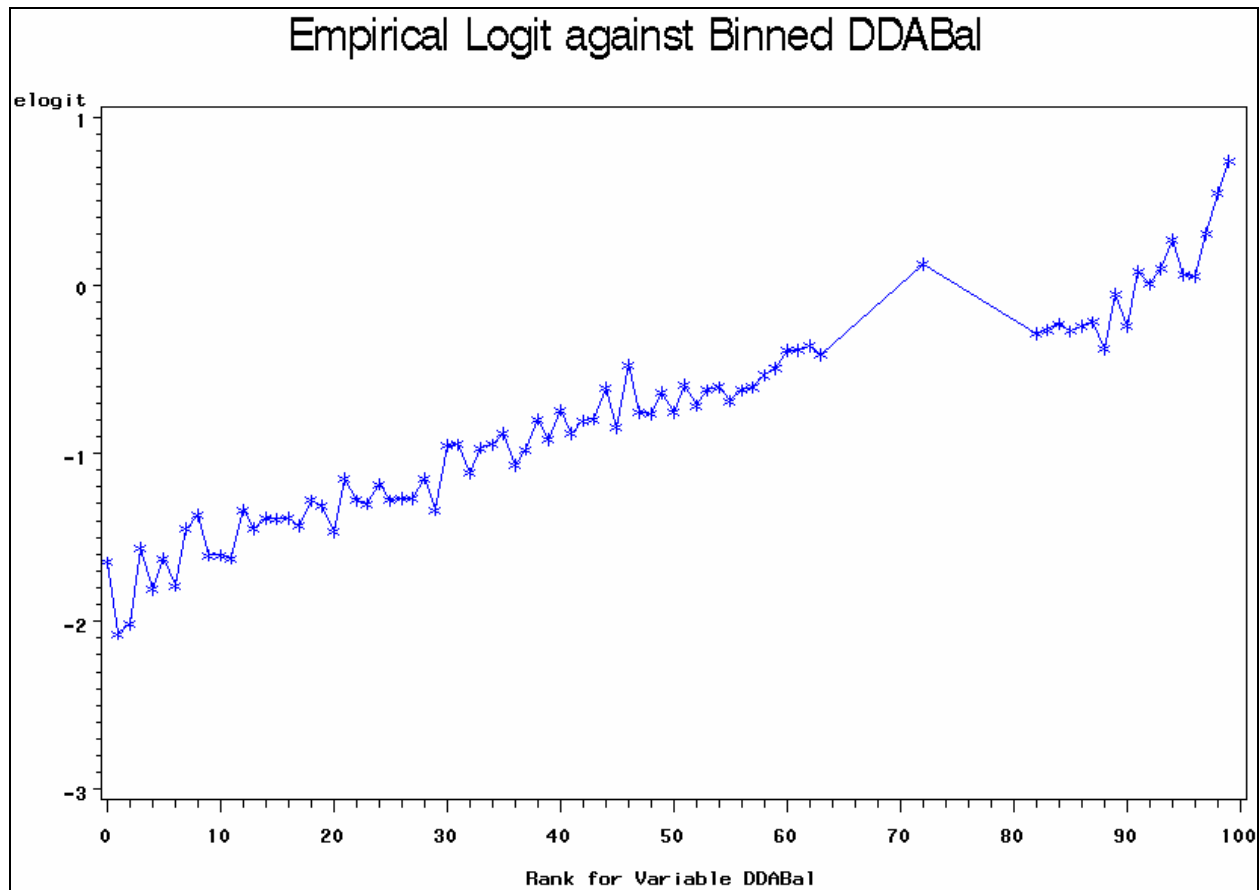
proc rank data=imputed groups=100 out=out;
    var &var;
    ranks bin;
run;

proc means data=out noprint nway;
    class bin;
    var ins &var;
    output out=bins sum(ins)=ins mean(&var)=&var;
run;

data bins;
    set bins;
    elogit=log((ins+(sqrt(_FREQ_)/2))/
        (_FREQ_-ins+(sqrt(_FREQ_)/2)));
run;

symbol i=join c=blue v=star;
proc gplot data = bins;
    title "Empirical Logit against &var";
    plot elogit * &var;
run;
    title "Empirical Logit against Binned &var";
    plot elogit * bin;
run; quit;
```





The spike seems to be accounted for. Now, the question that remains is that of transformation. Which transformation does the best job of linearizing the relationship? You may have several of your favorite transformations; the limitations are usually only the analyst's imagination and the software on the server that houses the database. Here, it seems like the bins do a very good job of capturing the way that response behavior changes with respect to checking balances in a fairly linear way. Hence, the following code creates a new input, **B_DDABal**, which represents the binned checking account balances. Because you would need this recipe to create the **B_DDABal** input if it eventually is part of your model, the following code generates DATA step code to perform the binning.



Some analysts might reserve the term binning for the practice of creating a categorical predictor from a continuous input. Therefore, it might be more appropriate to label this a *rank-group* or *percentile* transformation.

Regardless of the input's original distribution, the percentiles have a uniform distribution. This uniformly distributed version of **DDABal** has a linear association with the logit. Even better, unlike the log transformation, the percentile transformation affords easy interpretability with odds ratios: a one-unit change in the percentiles results in the corresponding change in the odds of response.

The RANK procedure assigns observations to 100 groups, according to **DDABal**. The MEANS procedure creates a data set that contains the maximum **DDABal** value in each bin. This information can be used to assign a new data set its own bins, without running the RANK procedure.

```
proc rank data=imputed groups=100 out=out;
    var ddabal;
    ranks bin;
run;

title;
proc means data = out noprint nway;
    class bin;
    var ddabal;
    output out=endpts max=max;
run;

proc print data = endpts(obs=10);
run;
```

	Obs	bin	_TYPE_	_FREQ_	max
	1	0	1	323	1.72
	2	1	1	322	8.52
	3	2	1	322	16.38
	4	3	1	323	27.12
	5	4	1	323	40.27
	6	5	1	322	52.51
	7	6	1	323	65.29
	8	7	1	323	81.54
	9	8	1	322	94.38
	10	9	1	323	110.38

Rather than writing out, by hand, a series of rules like IF **DDABal** <= 1.72 then bin =0; ELSE IF... for many bins, you can use a DATA step to write the rules. An easy way to do this is to use a FILENAME statement to point to a file and then write the rules to that file using the PUT statement.

The FILENAME statement creates a *fileref*, Rank, which points to the physical file C:\temp\rank.sas. The DATA _NULL_ statement enables you to use the DATA step processing without being required to write out a data set. The FILE statement specifies where you want to put the output of the DATA step (much as the INFILE statement would enable you to specify the input source for a DATA step.) The SET statement brings in the data set **endpts**, which has the maximum balance in each bin. The option END= creates an internal flag that you can use to identify the last record of the data set. You could write IF...THEN...ELSE syntax to do the bin assignment. The following example uses SELECT...WHEN...OTHERWISE to perform the same task. The last record captures everyone with a balance larger than the maximum in the penultimate bin.

```
filename rank "C:\temp\rank.sas";

data _null_;
  file rank;
  set endpts end=last;
  if _n_ = 1 then put "select;";
  if not last then do;
    put "  when (ddabal <= " max ") B_DDABal =" bin ";";
    end;
  else if last then do;
    put "  otherwise B_DDABal =" bin ";";
    put "end;";
    end;
run;
```

Partial Listing of RANK.SAS

```
select;
  when (ddabal <= 1.72 ) B_DDABal =0 ;
  when (ddabal <= 8.52 ) B_DDABal =1 ;
  when (ddabal <= 16.38 ) B_DDABal =2 ;
  when (ddabal <= 27.12 ) B_DDABal =3 ;
  when (ddabal <= 40.27 ) B_DDABal =4 ;
  when (ddabal <= 52.51 ) B_DDABal =5 ;
  when (ddabal <= 65.29 ) B_DDABal =6 ;
  when (ddabal <= 81.54 ) B_DDABal =7 ;
  when (ddabal <= 94.38 ) B_DDABal =8 ;
  when (ddabal <= 110.38 ) B_DDABal =9 ;
```

To use this code, you can use the %INCLUDE statement embedded in a DATA step. For example, the following code puts **B_DDABal** on the **Imputed** data set. The **SOURCE2** option requests that the code be included in the log as well.

```
data imputed;
  set imputed;
  %include rank /source2;
run;
```

To see that the recoding worked, evaluate the minimum and maximum checking account balances in each of the bins. The MEANS procedure with a CLASS statement will do this.

```
proc means data = imputed min max;
  class B_DDABal;
  var DDABal;
run;
```


Partial Output

The MEANS Procedure				
Analysis Variable : DDABal Checking Balance				
B_DDABal	N Obs	Minimum	Maximum	
0	323	-774.8300000	1.7200000	
1	322	1.7300000	8.5200000	
2	322	8.5300000	16.3800000	
3	323	16.4000000	27.1200000	
4	323	27.2400000	40.2700000	
5	322	40.4400000	52.5100000	
6	323	52.5400000	65.2900000	
7	323	65.3700000	81.5400000	
8	322	81.5600000	94.3800000	
9	323	94.4200000	110.3800000	
10	323	110.4000000	125.0000000	

Because the binned balance input is a replacement for the **DDABal** column, switch **B_DDABal** for **DDABal** in the **Screened** macro variable.

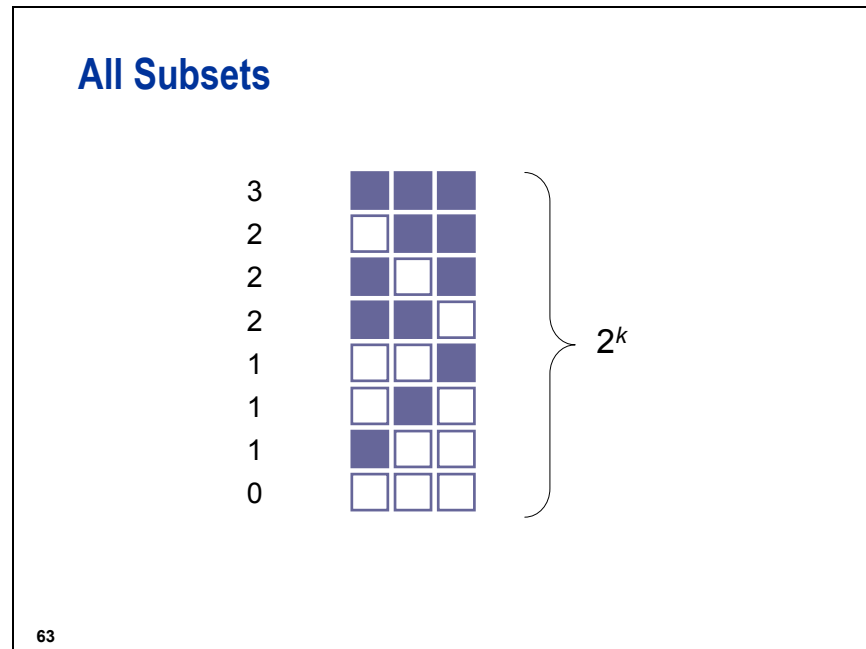
```
%let screened =
brclus2 checks ccbal
mmbal income ilsbal posamt
nsfamt cd irabal age
sav dda invbal
crscore brclus3 cc brclus1
cashbk miacctag micrscor moved
acctage dirdep savbal B_DDABAL
sdb ccpurc inarea atmamt
phone mmcred inv
depamt brclus4 atm lores;
```



Binning is not the only option. Other popular techniques for hand-crafting inputs include the transformations mentioned above as well as splines. Notice that **B_DDABal** is not a categorical input. The point of the linearization exercise is to take advantage of the linear relationship between logits and bins, not add 100 dummy variables to the list of inputs.

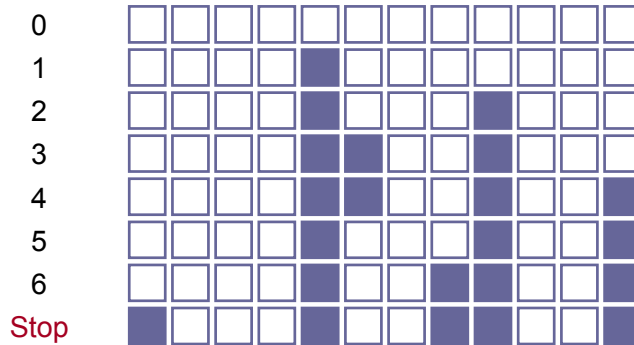
If there was any question about data quality, the binning code would have to be much more robust.

3.4 Subset Selection



Variable selection methods in regression are concerned with finding subsets of the inputs that are jointly important in predicting the target. The most thorough search would consider all possible subsets. This can be prohibitively expensive when the number of inputs, k , is large, as there are 2^k possible subsets to consider.

Stepwise Selection

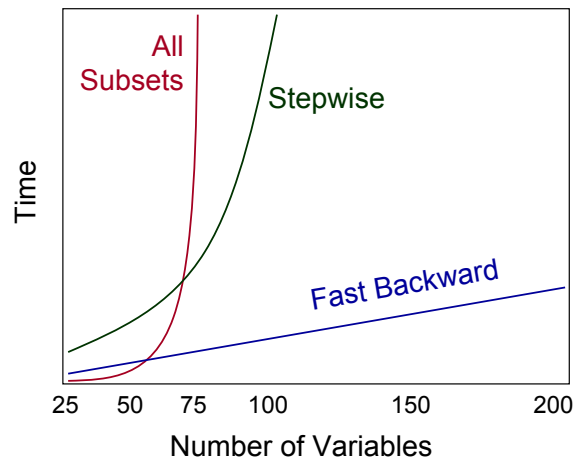


64

Stepwise variable selection is a much-maligned yet heavily used subset selection method. Stepwise selection first searches the 1-input models and selects the best. It then searches the 2-input models that contain the input selected in the first step and selects the best. The model is incrementally built in this fashion until no improvement is made. There is also a backward portion of the algorithm where at each step, the variables in the current model can be removed if they have become unimportant. The usual criterion used for entry and removal from the model is the p -value from a significance test that the coefficient is zero, although other criteria can also be used. Note that in subset selection, the p -value is merely a tuning parameter that measures the relative strength of the candidate variables.

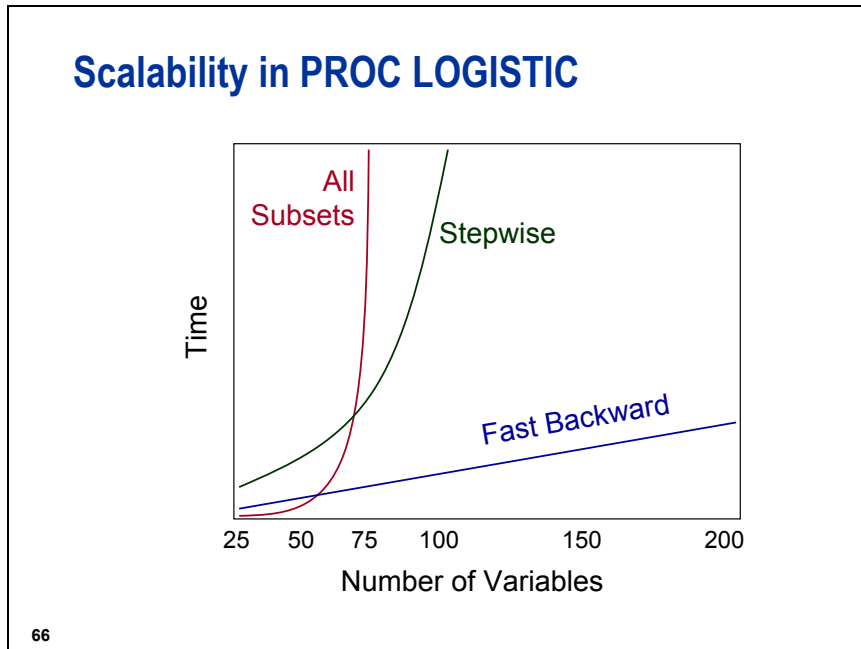
Stepwise selection was devised to give a computationally efficient alternative to examining all subsets. It is not guaranteed to find the best subset and it can be shown to perform badly in many situations (Harrell 1997).

Scalability in PROC LOGISTIC



66

Backward variable selection starts with all the candidate variables in the model simultaneously. At each step, the least important input variable is removed (as determined by the p -value). Backward elimination is less inclined to exclude important inputs or include spurious inputs than forward (stepwise) methods (Mantel 1970; Harrell 1997). However, it is considered more computationally expensive than stepwise because more steps are usually required and they involve larger models.



Most of the literature on the different subset selection methods has considered linear rather than logistic regression. The conventional wisdom regarding computation time is that stepwise < backwards < all subsets.

However, logistic regression (as implemented by PROC LOGISTIC) gives a different story. For up to ≈ 60 inputs, the results are reversed
all subsets < backwards < stepwise.

For any number of inputs, backward elimination (with the FAST option) is more efficient than stepwise. (The above simulation was conducted with 50,000 cases and 200 intercorrelated inputs; 16 of the inputs were important, 6 strongly so.)

Logistic regression requires an iterative optimization algorithm. Each model fit is much more expensive than with linear regression. Each step in the stepwise algorithm requires iterative optimization. To find 16 variables (considering only the forward part of stepwise) would take 16k–120 separate nonlinear regressions, each of which might require several iterations. Stepwise logistic regression is a poor performer. Because its only universally acknowledged advantage is speed, there is little reason to prefer it for logistic regression (using PROC LOGISTIC).

All-subsets selection is executed in PROC LOGISTIC with the SELECTION=SCORE option (SAS Institute Inc. 1997). This method only requires that one model be fit (the full model). The results are then manipulated to calculate a score test for each possible combination of input variables. It also uses a branch and bound method for efficiently searching the many combinations. This method is the fastest until the number of possible combinations becomes unmanageable, at which point the performance acutely deteriorates. If redundant inputs are eliminated first (using variable clustering), then all-subsets selection can be a practical method for predictive modeling.

When combined with the FAST option, backward variable selection requires only a single logistic regression (SAS Institute Inc. 1997). PROC LOGISTIC uses the method of Lawless and Singhal (1978) to manipulate the full model fit to approximate fitting reduced models. The FAST option is extremely efficient because the model is not refitted for every variable removed. Fast backward elimination had the best overall performance, a linear increase in time as the number of inputs increased. Note that ordinary backwards (without the FAST option) would have been slower than stepwise.

3.09 Multiple Choice Poll

Which of the following statements is true regarding best subsets selection?

- a. The models are ranked by the likelihood ratio chi-square.
- b. The method uses a forward selection to find the best model.
- c. The method is relatively efficient for a small number of variables.
- d. None of the above



Automatic Subset Selection

PROC LOGISTIC can be used to further reduce the number of input variables. In the MODEL statement, the SELECTION= option specifies the method, in this example, the backward elimination method. The FAST option uses the full model fit to approximate the remaining slope estimates for each subsequent elimination of a variable from the model. The SLSTAY option specifies the significance level for a variable to stay in the model in a backward elimination step. The significance level⁴ was chosen arbitrarily for illustrative purposes.

```
proc logistic data=imputed des;
  class res;
  model ins=&screened res / selection=backward fast
    slstay=.001;
run;
```

Partial Output

Summary of Backward Elimination						
Step	Effect Removed	DF	Number In	Wald Chi-Square	Pr > ChiSq	Variable Label
1	CCPurc	1	37	0.0020	0.9646	Credit Card Purchases
1	POSAmt	1	36	0.0219	0.8824	Amount Point of Sale
1	MMCred	1	35	0.0292	0.8643	Money Market Credits
1	Age	1	34	0.0707	0.7904	Age
1	Res	2	33	1.1392	0.5657	Area Classification
1	InvBal	1	32	0.3452	0.5569	Investment Balance
1	CRScore	1	31	0.3567	0.5504	Credit Score
1	Moved	1	30	0.4602	0.4975	Recent Address Change
1	Income	1	29	0.7832	0.3762	Income
1	LORes	1	28	0.6500	0.4201	Length of Residence
1	InArea	1	27	1.5256	0.2168	Local Address
1	DDA	1	26	1.9730	0.1601	Checking Account
1	DepAmt	1	25	3.6400	0.0564	Amount Deposited
1	CashBk	1	24	3.7577	0.0526	Number Cash Back
1	MICRScor	1	23	3.8363	0.0502	
1	SDB	1	22	4.9697	0.0258	Safety Deposit Box
1	IRABal	1	21	10.0864	0.0015	IRA Balance
1	brclus1	1	20	10.1954	0.0014	
1	CCBal	1	19	10.4551	0.0012	Credit Card Balance
1	MIAcctAg	1	18	10.3531	0.0013	

⁴ The choice of significance level is a multi-faceted question. Analysts with experience may be able to posit a significance level that will secure a model with good generalizing power. If not, the information criterion-based method that follows or the techniques of the following chapters can be used to find a model that generalizes well.

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	
Intercept	1	-1.9129	0.0414	2134.7511	<.0001	
brclus2	1	-0.6501	0.0587	122.7555	<.0001	
Checks	1	-0.0270	0.00299	81.5434	<.0001	
MMBal	1	0.000043	2.449E-6	305.2159	<.0001	
ILSBal	1	-0.00002	5.881E-6	15.7352	<.0001	
NSFAmt	1	0.00387	0.000977	15.7276	<.0001	
CD	1	0.9106	0.0379	578.1748	<.0001	
Sav	1	0.5681	0.0298	363.1238	<.0001	
brclus3	1	-0.6280	0.0650	93.3773	<.0001	
CC	1	0.2852	0.0286	99.4085	<.0001	
AcctAge	1	-0.0177	0.00217	66.4155	<.0001	
DirDep	1	-0.1716	0.0315	29.6149	<.0001	
SavBal	1	0.000044	2.243E-6	382.7221	<.0001	
B_DDABal	1	0.0190	0.000510	1391.6548	<.0001	
ATMAmt	1	0.000036	4.642E-6	60.6472	<.0001	
Phone	1	-0.0643	0.0150	18.4484	<.0001	
Inv	1	0.6106	0.0803	57.7643	<.0001	
brclus4	1	-0.8812	0.0874	101.7003	<.0001	
ATM	1	-0.2338	0.0309	57.2571	<.0001	

The results show that PROC LOGISTIC using the backward elimination method reduced the number of variables down from 38 to 18.

The SELECTION=SCORE option finds the best subsets of each model size. The number of models printed of each size is controlled by the BEST= option. Because the best subsets method does not support class variables, dummy variables for **Res** are created in a DATA step.

```
data imputed;
  set imputed;
  resr=(res='R');
  resu=(res='U');
run;

proc logistic data=imputed des;
  model ins=&screened resr resu
    / selection=score best=1;
run;
```


Partial Output

Regression Models Selected by Score Criterion		
Number of Variables	Score Chi-Square	Variables Included in Model
1	2781.3404	B_DDABal
2	3711.5967	CD B_DDABal
3	4380.4694	CD SavBal B_DDABal
4	4763.8784	CD Sav SavBal B_DDABal
5	5147.5244	MMBal CD Sav SavBal B_DDABal
6	5376.0884	Checks MMBal CD Sav SavBal B_DDABal
7	5567.6038	Checks MMBal CD Sav CC SavBal B_DDABal
8	5681.4645	Checks MMBal CD Sav CC brclus1 SavBal B_DDABal
9	5749.8760	Checks MMBal CD Sav CC brclus1 SavBal B_DDABal Inv
10	5807.1711	brclus2 Checks MMBal CD Sav brclus3 CC SavBal B_DDABal brclus4
11	5872.1641	brclus2 Checks MMBal CD Sav brclus3 CC SavBal B_DDABal Inv brclus4
12	5924.1821	brclus2 Checks MMBal CD Sav brclus3 CC DirDep SavBal B_DDABal Inv brclus4
13	5961.7085	brclus2 Checks MMBal CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal Inv brclus4
14	6009.4822	brclus2 Checks MMBal CD Sav brclus3 CC AcctAge SavBal B_DDABal ATMamt Inv brclus4 ATM

The score test statistic increases with model size. The Schwarz Bayes criterion (SBC) is often used for model selection. The SBC is essentially the $-2 \log$ likelihood plus a penalty term that increases as the model gets bigger. The penalty term for SBC is $(k+1) \cdot \ln(n)$, where k is the number of variables in the model and n is the sample size. Smaller values of SBC are preferable. The score test statistic is asymptotically equivalent to the likelihood ratio statistic. The $-2 \log$ likelihood is a constant minus the likelihood ratio statistic. Thus, an SBC type statistic could be computed from the score statistic as $-(\text{score}) + (k+1) \cdot \ln(n)$, where smaller values would be preferable.

When the SELECTION=SCORE option is used, output data sets are not available. Therefore, the Output Delivery System is used to create an output data set with the score statistic and the number of variables. The output objects with this information are called NOBS and BESTSUBSETS.

```
ods listing close;
ods output NObs=NObs
           bestsubsets=score;

proc logistic data=imputed des;
  model ins=&screened resr resu
        / selection=score best=1;
run;

ods listing;

proc print data=NObs;
run;
```

				N	NObs Read	NObs Used	Sum Freqs Read	Sum Freqs Used
Obs	Label							
1	Number of Observations Read			32264	32264	32264	32264	32264
2	Number of Observations Used			32264	32264	32264	32264	32264

The value of interest is the number of observations.

```
proc print data=score;
run;
```

Partial Output

Obs	control_ var	NumberOf Variables	ScoreChiSq
1		1	2781.3404
2	1	2	3711.5967
3	1	3	4380.4694
4	1	4	4763.8784
5	1	5	5147.5244

Obs VariablesInModel

```
1 B_DDABal
2 CD B_DDABal
3 CD SavBal B_DDABal
4 CD Sav SavBal B_DDABal
5 MMBal CD Sav SavBal B_DDABal
```

The variables of interest to calculate SBC are **NumberOfVariables** and **ScoreChiSq** from the **score** data set and **N** from the **NObs** data set.

The first DATA step selects one observation (the number of observations) and creates a macro variable **obs** that contains the number of observations. The second DATA step computes the SBC statistic.

```
data _NULL_;  
    set NObs;  
    where label = 'Number of Observations Used';  
    call symput('obs',n);  
run;  
  
data subset;  
    set score;  
    sbc=-scorechisq+log(&obs)*(numberofvariables+1);  
run;  
  
proc print data=subset;  
    var sbc variablesinmodel;  
run;
```

Partial Output

```

Obs      sbc

  1 -2760.58
  2 -3680.45
  3 -4338.94
  4 -4711.97
  5 -5085.23
  6 -5303.42
  7 -5484.55
  8 -5588.03
  9 -5646.06
 10 -5692.97
 11 -5747.58
 12 -5789.22
 13 -5816.36
 14 -5853.76
 15 -5875.59
 16 -5879.15
 17 -5879.46
 18 -5879.71
 19 -5879.35
 20 -5878.21

Obs VariablesInModel

  1 B_DDABal
  2 CD B_DDABal
  3 CD SavBal B_DDABal
  4 CD Sav SavBal B_DDABal
  5 MMBal CD Sav SavBal B_DDABal
  6 Checks MMBal CD Sav SavBal B_DDABal
  7 Checks MMBal CD Sav CC SavBal B_DDABal
  8 Checks MMBal CD Sav CC brclus1 SavBal B_DDABal
  9 Checks MMBal CD Sav CC brclus1 SavBal B_DDABal Inv
 10 brclus2 Checks MMBal CD Sav brclus3 CC SavBal B_DDABal brclus4
 11 brclus2 Checks MMBal CD Sav brclus3 CC SavBal B_DDABal Inv brclus4
 12 brclus2 Checks MMBal CD Sav brclus3 CC DirDep SavBal B_DDABal Inv brclus4
 13 brclus2 Checks MMBal CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal Inv brclus4
 14 brclus2 Checks MMBal CD Sav brclus3 CC AcctAge SavBal B_DDABal ATMamt Inv brclus4 ATM
 15 brclus2 Checks MMBal CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal ATMamt Inv brclus4 ATM
 16 brclus2 Checks MMBal ILSBal CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal ATMamt Inv brclu
 17 brclus2 Checks MMBal ILSBal CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal ATMamt Phone Inv
 18 brclus2 Checks MMBal ILSBal NSFamt CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal ATMamt Ph
 19 brclus2 Checks MMBal ILSBal NSFamt CD IRABal Sav brclus3 CC AcctAge DirDep SavBal B_DDABal AT
 20 brclus2 Checks MMBal ILSBal NSFamt CD IRABal Sav brclus3 CC brclus1 AcctAge DirDep SavBal B_D

```

Careful inspection reveals that the 18-input model has the smallest value of **sbc**. The following SQL code creates a macro variable **selected** that contains the names of the inputs in that model.

```

proc sql;
  select VariablesInModel into :selected
  from subset
  having sbc=min(sbc);
quit;

```

Variables Included in Model

```
brclus2 Checks MMBal ILSBal NSFamt CD Sav brclus3 CC AcctAge DirDep SavBal B_DDABal ATMamt Phone
Inv brclus4 ATM
```

These automatic selection routines raise several questions. For techniques like stepwise selection and backward elimination, what are good stopping rules? For best subsets, what number of inputs yields the best model?

The answers to these questions lie in the purposes of the models. The goal of most predictive modeling is generalization. Hence, the best model is the model that generalizes the best. How does one measure generalizing ability of a model? What are some statistics that summarize a model's performance? The next chapters offer several suggestions for comparing and selecting models.

3.10 Multiple Choice Poll

Which model had the highest c statistic from the models generated in PROC LOGISTIC and PROC REG in Exercises 3 and 4?

- a. Logistic Regression model with the FAST BACKWARD method using only the selected cluster representatives as inputs.
- b. Logistic Regression model with the STEPWISE method using all numeric variables and 3 categorical variables.
- c. Logistic Regression model with the FAST BACKWARD method using all numeric variables and 3 categorical variables.
- d. Logistic Regression model with the lowest SBC selected by the SCORE method using all numeric variables and 3 categorical variables.
- e. Linear Regression model with the STEPWISE method using all numeric variables and the dummy codes for the categorical variables.

3.5 Chapter Summary

Preparing the data for predictive modeling can be laborious. First, missing values need to be replaced with reasonable values. Missing indicator variables are also needed if missingness is related to the target. If there are nominal input variables with numerous levels, the levels should be collapsed to reduce the likelihood of quasi-complete separation and to reduce the redundancy among the levels. Furthermore, if there are numerous input variables, variable clustering should be performed to reduce the redundancy among the variables. Additionally, there are several selection methods in the LOGISTIC procedure to select a subset of variables.

To assist in identifying nonlinear associations, the Hoeffding's D statistic can be used. A variable with a low rank in the Spearman correlation statistic but with a high rank in the Hoeffding's D statistic may indicate that the association with the target is nonlinear.

General form of the STDIZE procedure:

```
PROC STDIZE DATA=SAS-data-set <options>;  
    VAR variables;  
RUN;
```

General form of the CLUSTER procedure:

```
PROC CLUSTER DATA=SAS-data-set <options>;  
    FREQ variable;  
    VAR variable;  
    ID variable;  
RUN;
```

General form of the VARCLUS procedure:

```
PROC VARCLUS DATA=SAS-data-set <options>;  
    VAR variables;  
RUN;
```

3.6 Solutions

Solutions to Exercises

Solutions to Student Activities (Polls/Quizzes)

3.01 Multiple Choice Poll – Correct Answer

Which of the following statements is false regarding missing values in predictive modeling applications?

- a. In complete case analysis, there can be an enormous loss of data when the missing values are spread across many variables.
- b. Observations with missing values will not be scored in PROC LOGISTIC.
- c. Missing indicator variables can be used to capture the relationship between the target variable and missing inputs.
- ☒ d. The missing completely at random assumption is valid in most predictive modeling applications.

11

3.02 Multiple Choice Poll – Correct Answer

For the cell grp_resp=0 and grp_amt=0, what was the missing value for donor_age replaced with?

- a. 57
- ☒ b. 64
- c. 58
- d. 65

17

3.03 Multiple Choice Poll – Correct Answer

Which of the following statements is false regarding Greenacre's method for collapsing levels of contingency tables?

- a. The levels are hierarchically clustered based on the reduction in the chi-squared test of association.
- b. Levels with similar marginal response rates are merged.
- c. The method accounts for the sample size in each level.
- ☒ d. The method is appropriate for any categorical input.

29

3.04 Multiple Choice Poll – Correct Answer

What is the optimum number of clusters for the cluster_code variable?

- a. 3
- b. 4
- ☒ c. 5
- d. 6

34

3.05 Multiple Choice Poll – Correct Answer

If you had 15 variables broken into 5 clusters, the sum of the eigenvalues over all clusters would be

- a. 5
- b. 10
- ☒ c. 15
- d. 20
- e. None of the above

45

3.09 Multiple Choice Poll – Correct Answer

Which of the following statements is true regarding best subsets selection?

- a. The models are ranked by the likelihood ratio chi-square.
- b. The method uses a forward selection to find the best model.
- ☒ c. The method is relatively efficient for a small number of variables.
- d. None of the above

69

3.10 Multiple Choice Poll – Correct Answer

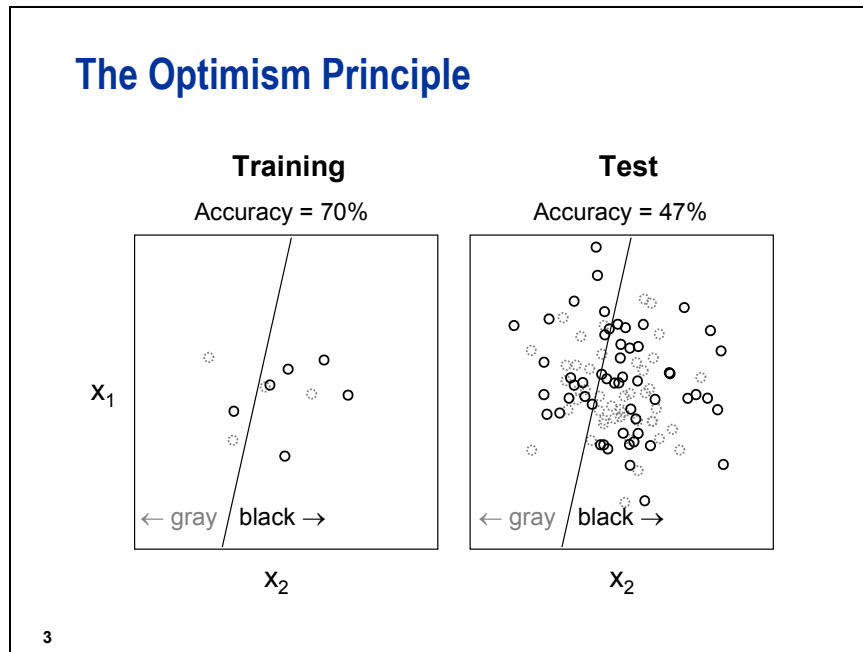
Which model had the highest c statistic from the models generated in PROC LOGISTIC and PROC REG in Exercises 3 and 4?

- ☒ a. Logistic Regression model with the FAST BACKWARD method using only the selected cluster representatives as inputs.
- b. Logistic Regression model with the STEPWISE method using all numeric variables and 3 categorical variables.
- c. Logistic Regression model with the FAST BACKWARD method using all numeric variables and 3 categorical variables.
- d. Logistic Regression model with the lowest SBC selected by the SCORE method using all numeric variables and 3 categorical variables.
- e. Linear Regression model with the STEPWISE method using all numeric variables and the dummy codes for the categorical variables.

Chapter 4 Measuring Classifier Performance

4.1	Honest Assessment	4-3
4.2	Misclassification.....	4-32
4.3	Allocation Rules	4-49
4.4	Overall Predictive Power	4-55
4.5	Chapter Summary.....	4-64
4.6	Solutions	4-65
	Solutions to Exercises	4-65
	Solutions to Student Activities (Polls/Quizzes)	4-65

4.1 Honest Assessment

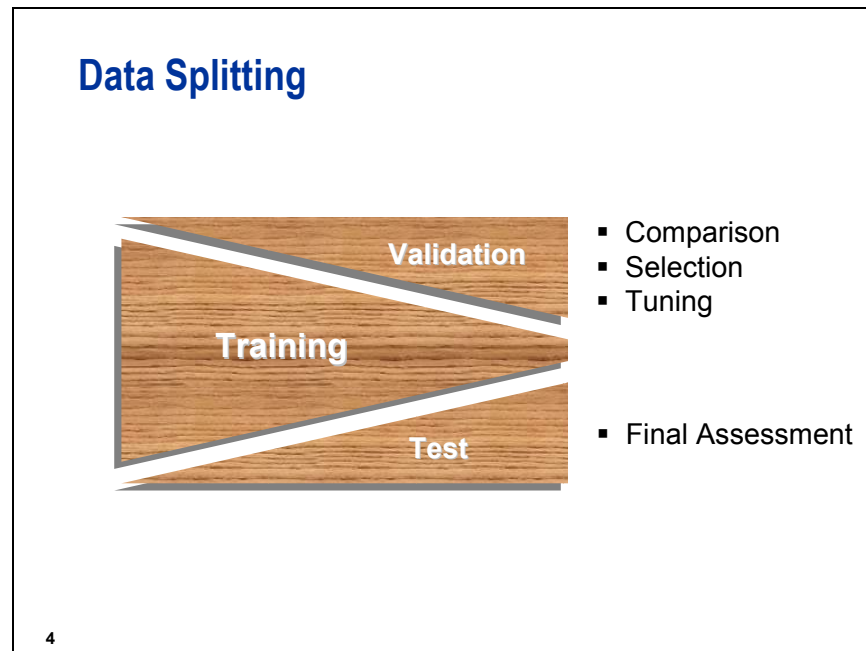


Evaluating the performance of a classifier on the same data used to train the classifier usually leads to an optimistically biased assessment.

For example, the above classifier was fit (or more properly *overfit*) to a 10-case data set. It correctly classified 70% of the cases. However, when the same classification rule was applied to 100 new cases from the same distribution, only 47% were correctly classified. This is called *overfitting*. The model was overly sensitive to peculiarities of the particular training data, in addition to true features of their joint distribution.

The more flexible the underlying model and less plentiful the data, the more that overfitting is a problem. When a relatively inflexible model like (linear) logistic regression is fitted to massive amounts of data, overfitting may not be a problem (Hand 1997). However, the chance of overfitting is increased by variable selection methods and supervised input preparation (such as collapsing levels of nominal variables based on associations with the target). It is prudent to assume overfitting until proven otherwise.

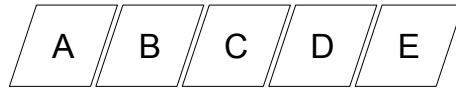
Large differences between the performance on the training and test sets usually indicate overfitting.



The simplest strategy for correcting the optimistic bias is to holdout a portion of the development data for assessment. The model is fit to the remainder (*training data set*) and performance is evaluated on the holdout portion (*test data set*). Usually from one-fourth to one-half of the development data is used as a test set (Picard and Berk 1990). After assessment, it is common practice to refit the final model on the entire undivided data set.

When the holdout data is used for comparing, selecting, and tuning models and the chosen model is assessed on the same data set that was used for comparison, then the optimism principle again applies. In this situation, the holdout sample is more correctly called a *validation data set*, not a test set. The test set is used for a final assessment of a fully specified classifier (Ripley 1996). If model tuning and a final assessment are both needed, then the data should be split three ways into training, validation, and test sets.

Other Approaches



	Train	Validate
1)	BCDE	A
2)	ACDE	B
3)	ABDE	C
4)	ABCE	D
5)	ABCD	E

5

Data splitting is a simple but costly technique. When data is scarce, it is inefficient to use only a portion for training. Furthermore, when the test set is small, the performance measures may be unreliable because of high variability. For small and moderate data sets, v -fold cross-validation (Breiman et al. 1984; Ripley 1996; Hand 1997) is a better strategy. In 5-fold cross-validation, for instance, the data would be split into five equal sets. The entire modeling process would be redone on each four-fifths of the data using the remaining one-fifth for assessment. The five assessments would then be averaged. In this way, all the data is used for both training and assessment.

Another approach that is frugal with the data is to assess the model on the same data set that was used for training but to penalize the assessment for optimism (Ripley 1996). The appropriate penalty can be determined theoretically or using computationally intensive methods such as the bootstrap.



Honest Model Assessment

The objective is to split the data into training and validation sets. The model and all the input preparation steps then need to be redone on the training set. The validation data will be used for assessment. Consequently, it needs to be treated as if it were truly new data where the target is unknown. The results of the analysis on the training data need to be applied to the validation data, not recalculated.

Several input-preparation steps can be done before the data is split. Creating missing indicators should be done on the full development data because the results will not change. The `rho1` macro variable is also created before the data is split to get the best estimate of the proportion of events.

```
%let pi1=0.02;

proc sql noprint;
    select mean(ins) into :rho1 from pmlr.develop;
quit;

%let inputs=ACCTAGE DDA DDABAL DEP DEPAMT CASHBK
            CHECKS DIRDEP NSF NSFAMT PHONE TELLER
            SAV SAVBAL ATM ATMAMT POS POSAMT CD
            CDBAL IRA IRABAL LOC LOCBAL INV
            INVBAL ILS ILSBAL MM MMBAL MMCRED MTG
            MTGBAL CC CCBAL CCPURC SDB INCOME
            HMOWN LORES HMVAL AGE CRSCORE MOVED
            INAREA;

data develop(drop=i);
    set pmlr.develop;
    /* name the missing indicator variables */
    array mi{*} MIAcctAg MIPhone MIPOS MIPOSamt
               MIIInv MIIInvBal MICC MICCBal
               MICCPurc MIIncome MIHMOwn MILOres
               MIHMVal MIAge MICRScor;
    /* select variables with missing values */
    array x{*} acctage phone pos posamt
               inv invbal cc ccbal
               ccpurc income hmown lores
               hmval age crscore;
    do i=1 to dim(mi);
        mi{i}=(x{i}=.);
    end;
run;
```


The SURVEYSELECT procedure can be used to select the records for the training and validation data sets. To create a stratified sample, the data must be sorted by the stratum variable. The SAMPRATE= option specifies what proportion of the **develop** data set should be selected. The default behavior of PROC SURVEYSELECT is to output the sample, not the entire data set, so the OUTALL option can be used to return the initial data set augmented by a flag to indicate selection in the sample.

Of course, this flag indicates membership in the training and validation data sets in this context. The FREQ procedure verifies the stratification. The SEED= option enables the user to control what series of pseudo-random numbers is generated to do the partitioning. A particular number, greater than zero, will produce the same split each time the SURVEYSELECT procedure was run. If the seed were zero, then the data would be split differently each time the procedure was run.

```
proc sort data=develop out=develop;
  by ins;
run;

proc surveyselect noprint
  data = develop
  samprate=.6667
  out=develop
  seed=44444
  outall;

  strata ins;
run;

proc freq data = develop;
  tables ins*selected;
run;
```

Notice that the proportion of responders is the same for both levels of selected, and the data has been partitioned into two-thirds for training and one-third for validation.

Table of Ins by Selected

Ins	Selected(Selection Indicator)		
Frequency	0	1	Total
Percent			
Row Pct			
Col Pct			
0	7028	14061	21089
	21.78	43.58	65.36
	33.33	66.67	
	65.36	65.36	
1	3724	7451	11175
	11.54	23.09	34.64
	33.32	66.68	
	34.64	34.64	
Total	10752	21512	32264
	33.33	66.67	100.00

The next DATA step creates the two data sets, **Train** and **Valid**. All other input-preparation steps are done after the split, on the training data only, because the validation data is treated as if it were truly new cases. This is imperative for supervised methods such as collapsing the levels of **Branch** based on its association with **Ins**. It could be argued that unsupervised methods, such as median imputation, could be done on **Develop** because they do not involve **Ins**. This example presents a cautious approach and does not involve the validation data at all.



This conservative approach also permits the comparison of models based on different imputation schemes.

```
data train valid;
    set develop;
    if selected then output train;
    else output valid;
run;
```

The STDIZE procedure imputes missing values. Here, as before, the median is used.

```
proc stdize data=train
    reponly
    method=median
    out=train1;
    var &inputs;
run;
```

The CLUSTER procedure is used to perform Greenacre's correspondence analysis; this is the method used to reduce the cardinality of the nominal input **Branch**.

```
proc means data=train1 noprint nway;
    class branch;
    var ins;
    output out=level mean=prop;
run;

ods listing close;
ods output clusterhistory=cluster;

proc cluster data=level
    method=ward
    outtree=fortree;
    freq _freq_;
    var prop;
    id branch;
run;

ods listing;

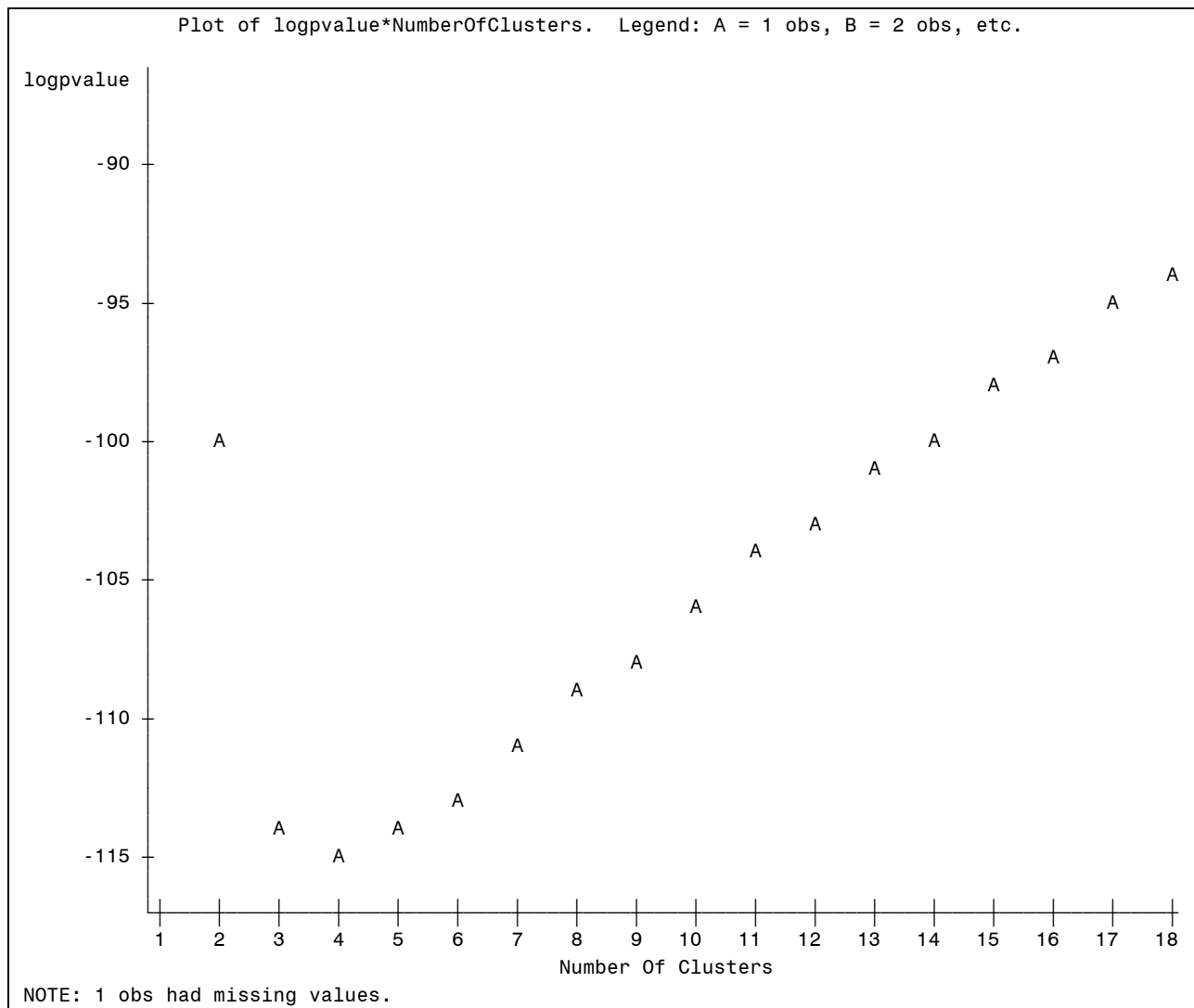
proc freq data=train1 noprint;
    tables branch*ins / chisq;
    output out=chi(keep=_pchi_) chisq;
run;
```

```

data cutoff;
  if _n_ = 1 then set chi;
  set cluster;
  chisquare=_pchi_*rsquared;
  degfree=numberofclusters-1;
  logpvalue=logsf('CHISQ',chisquare,degfree);
run;

proc plot data=cutoff;
  plot logpvalue*numberofclusters/vpos=30;
run; quit;

```



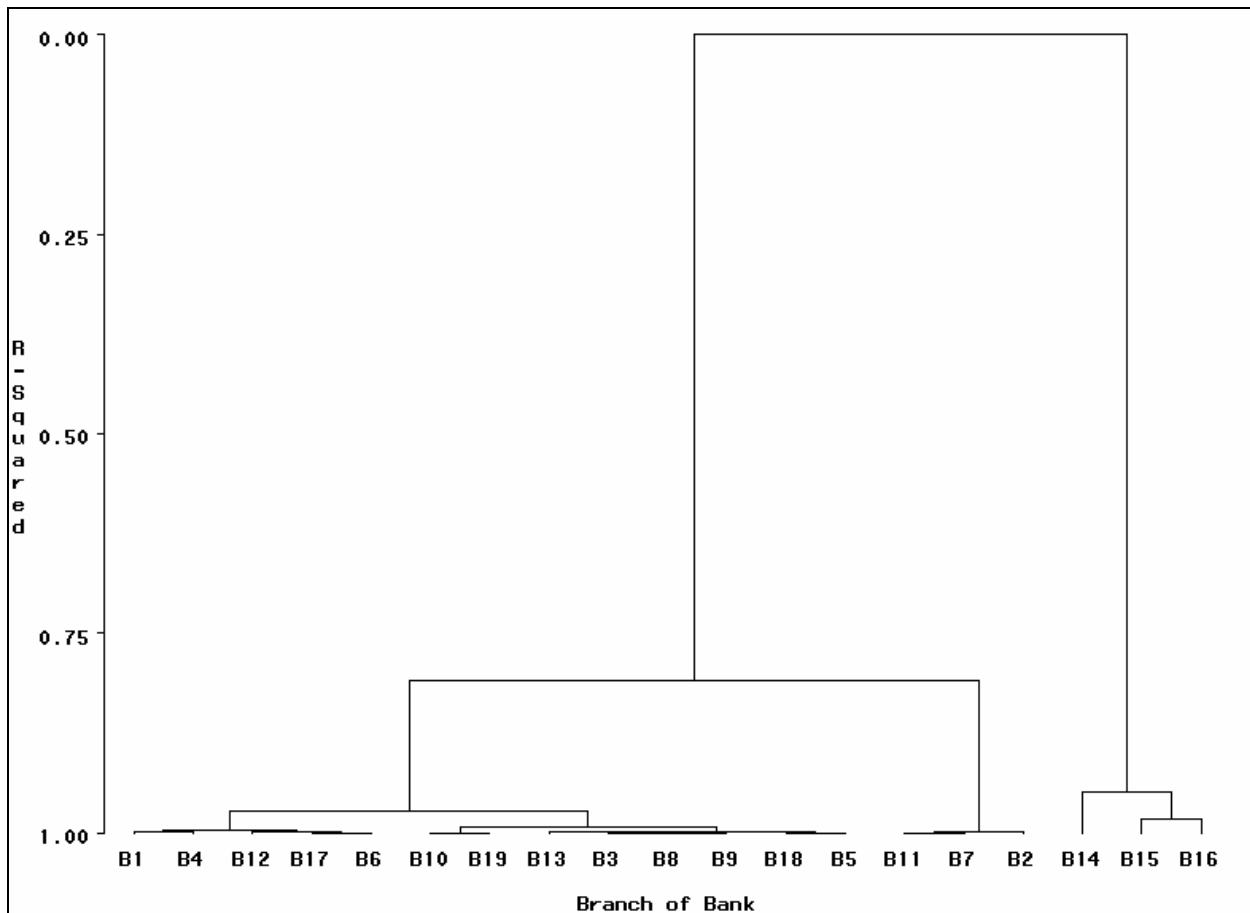
A macro variable is created using the SQL procedure that contains the number of clusters associated with the smallest p -value. The TREE procedure then creates a data set containing the information about that cluster solution.

```
proc sql;
  select NumberOfClusters into :ncl
    from cutoff
    having logpvalue=min(logpvalue);
quit;
```

Number
Of
Clusters

4

```
proc tree data=fortree h=rsq
  nclusters=&ncl out=clus;
  id branch;
run;
```



```

proc sort data=clus;
  by clusname;
run;

proc print data=clus;
  by clusname;
  id clusname;
run;

```

CLUSNAME	Branch	CLUSTER
B14	B14	4
CL4	B17	1
	B6	1
	B10	1
	B19	1
	B3	1
	B8	1
	B18	1
	B5	1
	B9	1
	B12	1
	B13	1
	B1	1
	B4	1
CL5	B15	3
	B16	3
CL8	B11	2
	B7	2
	B2	2

Three dummy variables are created, and the cluster with the response rate closest to p_1 is made the reference level.

```

data train1;
  set train1;
  brclus1=(branch='B14');
  brclus2=(branch in ('B12','B5','B8',
                     'B3','B18','B19','B17',
                     'B4','B6','B10','B9',
                     'B1','B13'));
  brclus3=(branch in ('B15','B16'));
run;

```

The VARCLUS procedure is used to group correlated inputs with one another. This allows variable reduction based on redundancy within the inputs.

```
ods listing close;
ods output clusterquality=summary
            rsquare=clusters;

proc varclus data=train1
            maxeigen=.7
            short
            hi;
    var &inputs brclus1-brclus3 miacctag
        miphone mipos miposamt miinv
        miinvbal micc miccbal miccpurc
        miincome mihmown milores mihmval
        miage micrscor;
run;
ods listing;

data _null_;
    set summary;
    call symput('nvar',compress(NumberOfClusters));
run;

proc print data=clusters noobs;
    where NumberOfClusters=&nvar;
    var Cluster Variable RSquareRatio VariableLabel;
run;
```

Cluster	Variable	RSquare Ratio	VariableLabel
Cluster 1	MIPhone	0.0000	
	MIPOS	0.0000	
	MIPOSAmt	0.0000	
	MIInv	0.0000	
	MIInvBal	0.0000	
	MICC	0.0000	
	MICCBal	0.0000	
	MICCPurc	0.0000	
	MIIncome	0.0074	
Cluster 2	MIHMown	0.0422	
	MILOres	0.0074	
	MIHMVal	0.0074	
	MIAge	0.0849	
Cluster 3	Teller	0.0000	Teller Visits
Cluster 4	MM	0.0890	Money Market
	MMBal	0.1053	Money Market Balance
	MMCred	0.4513	Money Market Credits
Cluster 5	Income	0.1626	Income
	HMVal	0.2372	Home Value

Cluster	Variable	RSquare Ratio	VariableLabel
Cluster 6	ILS	0.0041	Installment Loan
	ILSBal	0.0041	Loan Balance
Cluster 7	LOC	0.2802	Line of Credit
	LOCBal	0.2974	Line of Credit Balance
Cluster 8	POS	0.0864	Number Point of Sale
	POSAmt	0.0858	Amount Point of Sale
Cluster 9	NSF	0.2494	Number Insufficient Fund
	NSFAmt	0.2464	Amount NSF
Cluster 10	CD	0.2875	Certificate of Deposit
	CDBal	0.3037	CD Balance
Cluster 11	LORes	0.0000	Length of Residence
Cluster 12	CC	0.3523	Credit Card
	CCPurc	0.3321	Credit Card Purchases
Cluster 13	ATMAmt	0.0000	ATM Withdrawal Amount
Cluster 14	brclus2	0.2961	
	brclus3	0.3480	
Cluster 15	Inv	0.0000	
Cluster 16	DDA	0.4446	Checking Account
	Dep	0.2571	
	Checks	0.4117	Number of Checks
Cluster 17	CashBk	0.0000	Number Cash Back
Cluster 18	Moved	0.0000	Recent Address Change
Cluster 19	IRA	0.0000	Retirement Account
Cluster 20	CRScore	0.0000	Credit Score
Cluster 21	MIAcctAg	0.0000	
Cluster 22	IRABal	0.0000	IRA Balance
Cluster 23	MICRScor	0.0000	
Cluster 24	MTGBal	0.0000	Mortgage Balance
Cluster 25	AcctAge	0.0000	Age of Oldest Account
Cluster 26	SavBal	0.0000	Saving Balance
Cluster 27	DDABal	0.0000	Checking Balance
Cluster 28	SDB	0.0000	Safety Deposit Box
Cluster 29	InArea	0.0000	Local Address
Cluster 30	Sav	0.0000	Saving Account
Cluster 31	Phone	0.0000	Number Telephone Banking
Cluster 32	CCBal	0.0000	Credit Card Balance
Cluster 33	InvBal	0.0000	
Cluster 34	MTG	0.0000	Mortgage
Cluster 35	HMOwn	0.0000	Owns Home
Cluster 36	DepAmt	0.0000	
Cluster 37	DirDep	0.0000	Direct Deposit
Cluster 38	ATM	0.0000	ATM
Cluster 39	brclus1	0.0000	
Cluster 40	Age	0.0000	Age

A total of 40 clusters are formed.

A macro variable **reduced** is created with the names of the selected cluster representatives.

```
%let reduced=
MIPhone MIIncome Teller MM
Income ILS LOC POSAmt
NSFAmt CD LORes CCPurc
ATMAmt brclus2 Inv Dep
CashBk Moved IRA CRScore
MIACctAg IRABal MICRScor MTGBal
AcctAge SavBal DDABal SDB
InArea Sav Phone CCBal
InvBal MTG HMOwn DepAmt
DirDep ATM brclus1 Age;
```

Using correlation coefficients, the list of potential inputs can be further screened. Recall that the purpose of this step is to eliminate poor performers, **not** select the top inputs.

```
ods listing close;
ods output spearmancorr=spearman
      hoeffdingcorr=hoeffding;

proc corr data=train1 spearman hoeffding rank;
  var &reduced;
  with ins;
run;

ods listing;

data spearman1(keep=variable scorrr spvalue ranksp);
  length variable $ 8;
  set spearman;
  array best(*) best1--best&nvar;
  array r(*) r1--r&nvar;
  array p(*) p1--p&nvar;
  do i=1 to dim(best);
    variable=best(i);
    scorrr=r(i);
    spvalue=p(i);
    ranksp=i;
    output;
  end;
run;
```



```
data hoeffding1(keep=variable hcorr hpvalue rankho);
  length variable $ 8;
  set hoeffding;
  array best(*) best1--best&nvar;
  array r(*) r1--r&nvar;
  array p(*) p1--p&nvar;
  do i=1 to dim(best);
    variable=best(i);
    hcorr=r(i);
    hpvalue=p(i);
    rankho=i;
    output;
  end;
run;

proc sort data=spearman1;
  by variable;
run;

proc sort data=hoeffding1;
  by variable;
run;

data correlations;
  merge spearman1 hoeffding1;
  by variable;
run;

proc sort data=correlations;
  by ranksp;
run;

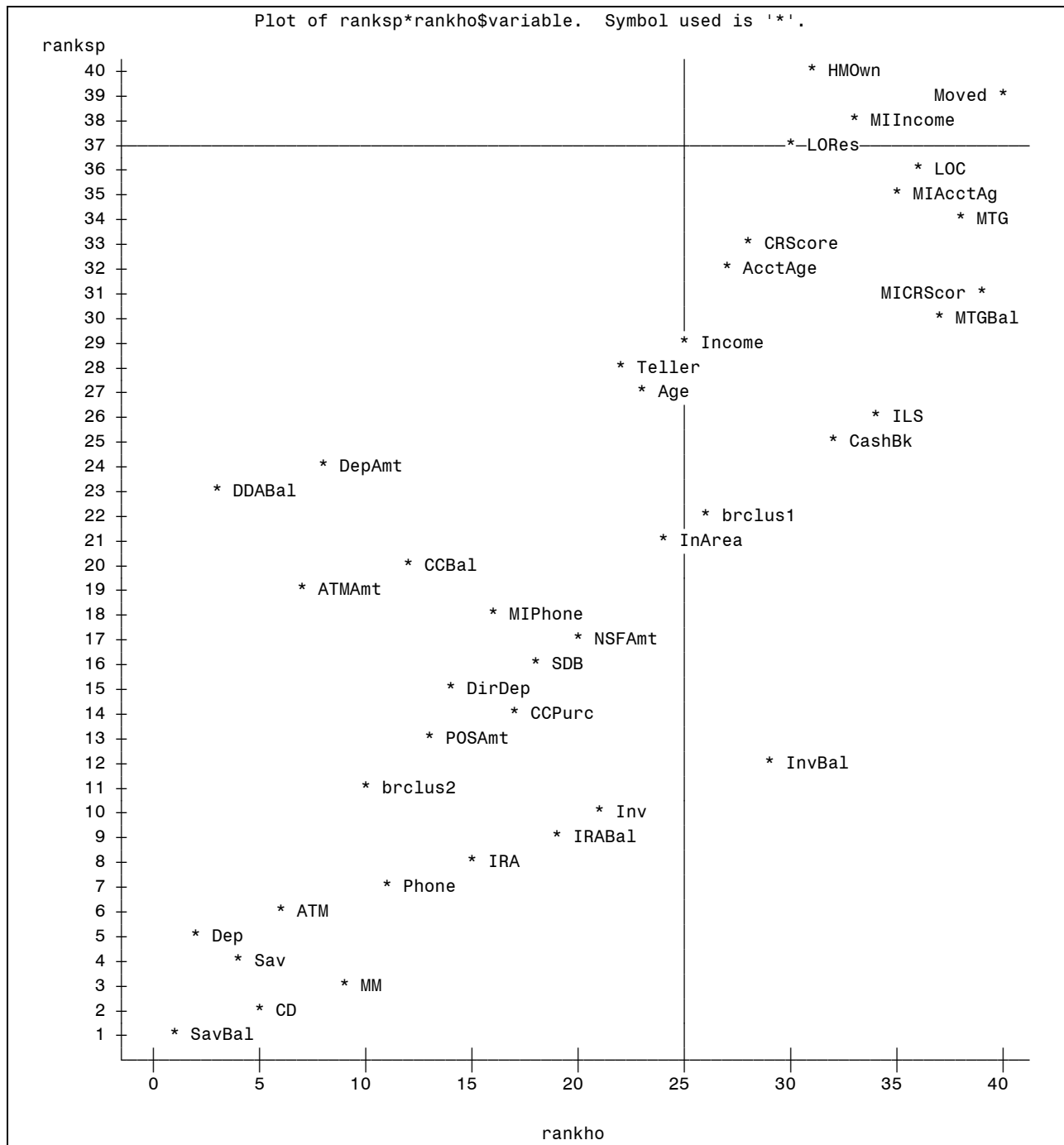
proc print data=correlations label split='*';
  var variable ranksp rankho scorr spvalue hcorr hpvalue;
  label ranksp = 'Spearman rank*of variables'
        scorr = 'Spearman Correlation'
        spvalue = 'Spearman p-value'
        rankho = 'Hoeffding rank*of variables'
        hcorr = 'Hoeffding Correlation'
        hpvalue = 'Hoeffding p-value';
run;
```

Obs	variable	Spearman rank of variables	Hoeffding rank of variables	Spearman Correlation	Spearman p-value	Hoeffding Correlation	Hoeffding p-value
1	SavBal	1	1	0.25272	0.00000	0.010014	0.00001
2	CD	2	5	0.21013	0.00000	0.001984	0.00001
3	MM	3	9	0.15958	0.00000	0.001045	0.00001
4	Sav	4	4	0.15145	0.00000	0.002373	0.00001
5	Dep	5	2	-0.15102	0.00000	0.003463	0.00001
6	ATM	6	6	-0.12047	0.00000	0.001414	0.00001
7	Phone	7	11	-0.10721	0.00000	0.000639	0.00001
8	IRA	8	15	0.10603	0.00000	0.000177	0.00027
9	IRABal	9	19	0.10416	0.00000	0.000153	0.00066
10	Inv	10	21	0.09940	0.00000	0.000040	0.05602
11	brclus2	11	10	0.08566	0.00000	0.000673	0.00001
12	InvBal	12	29	0.07762	0.00000	-0.000028	0.99969
13	POSAmt	13	13	-0.07749	0.00000	0.000422	0.00001
14	CCPurc	14	17	0.07454	0.00000	0.000171	0.00034
15	DirDep	15	14	-0.07334	0.00000	0.000422	0.00001
16	SDB	16	18	0.07222	0.00000	0.000154	0.00064
17	NSFAmt	17	20	-0.07122	0.00000	0.000114	0.00292
18	MIPhone	18	16	-0.07046	0.00000	0.000176	0.00028
19	ATMAmt	19	7	-0.06790	0.00000	0.001336	0.00001
20	CCBal	20	12	0.06461	0.00000	0.000529	0.00001
21	InArea	21	24	-0.06090	0.00000	-0.000005	0.46212
22	brclus1	22	26	-0.05666	0.00000	-0.000021	0.92877
23	DDABal	23	3	0.05296	0.00000	0.003045	0.00001
24	DepAmt	24	8	-0.04935	0.00000	0.001214	0.00001
25	CashBk	25	32	-0.04212	0.00000	-0.000055	1.00000
26	ILS	26	34	-0.01885	0.00569	-0.000057	1.00000
27	Age	27	23	0.01501	0.02775	0.000003	0.31614
28	Teller	28	22	-0.01331	0.05091	0.000028	0.09305
29	Income	29	25	0.01305	0.05571	-0.000019	0.88381
30	MTGBal	30	37	-0.00814	0.23228	-0.000063	1.00000
31	MICRScor	31	39	0.00763	0.26310	-0.000065	1.00000
32	AcctAge	32	27	-0.00751	0.27048	-0.000021	0.94154
33	CRScore	33	28	0.00710	0.29760	-0.000021	0.94231
34	MTG	34	38	-0.00702	0.30299	-0.000063	1.00000
35	MIAcctAg	35	35	0.00597	0.38158	-0.000062	1.00000
36	LOC	36	36	-0.00540	0.42846	-0.000063	1.00000
37	LORes	37	30	0.00438	0.52027	-0.000040	1.00000
38	MIIncome	38	33	0.00398	0.55921	-0.000057	1.00000
39	Moved	39	40	0.00061	0.92861	-0.000066	1.00000
40	HMOwn	40	31	-0.00014	0.98360	-0.000052	1.00000

As before, a plot may be useful.

```
proc sql noprint;
  select min(ranksp) into :vref
  from (select ranksp
        from correlations
        having spvalue > .5);
  select min(rankho) into :href
  from (select rankho
        from correlations
        having hpvalue > .5);
run; quit;

proc plot data = correlations;
  plot ranksp*rankho $ variable="*"
    /vref=&vref href=&href vpos=&nvar;
run; quit;
```



Dropping any variable that has both the Hoeffding and the Spearman p -value above .5 leaves 36 inputs.

Put their names into a macro variable named **Screened**.

```
%let screened =
MIPhone Teller MM
Income ILS LOC POSAmt
NSFAmt CD CCPurc
ATMAmt brclus2 INV DEP
CashBk IRA CRScore
MIACctAg IRABal MICRScor MTGBal
AcctAge SavBal DDABal SDB
InArea Sav Phone CCBal
INVBal MTG DEPAmt
DirDep ATM brclus1 Age;
```

Create logit plots and assess the linearity assumption for **DDABal** and **SavBal**, two skewed inputs.¹ You can do this by running the code below, which generates plots for **DDABal**, and then re-running the code from the line after *********. That second run will generate comparable plots for **SavBal**.

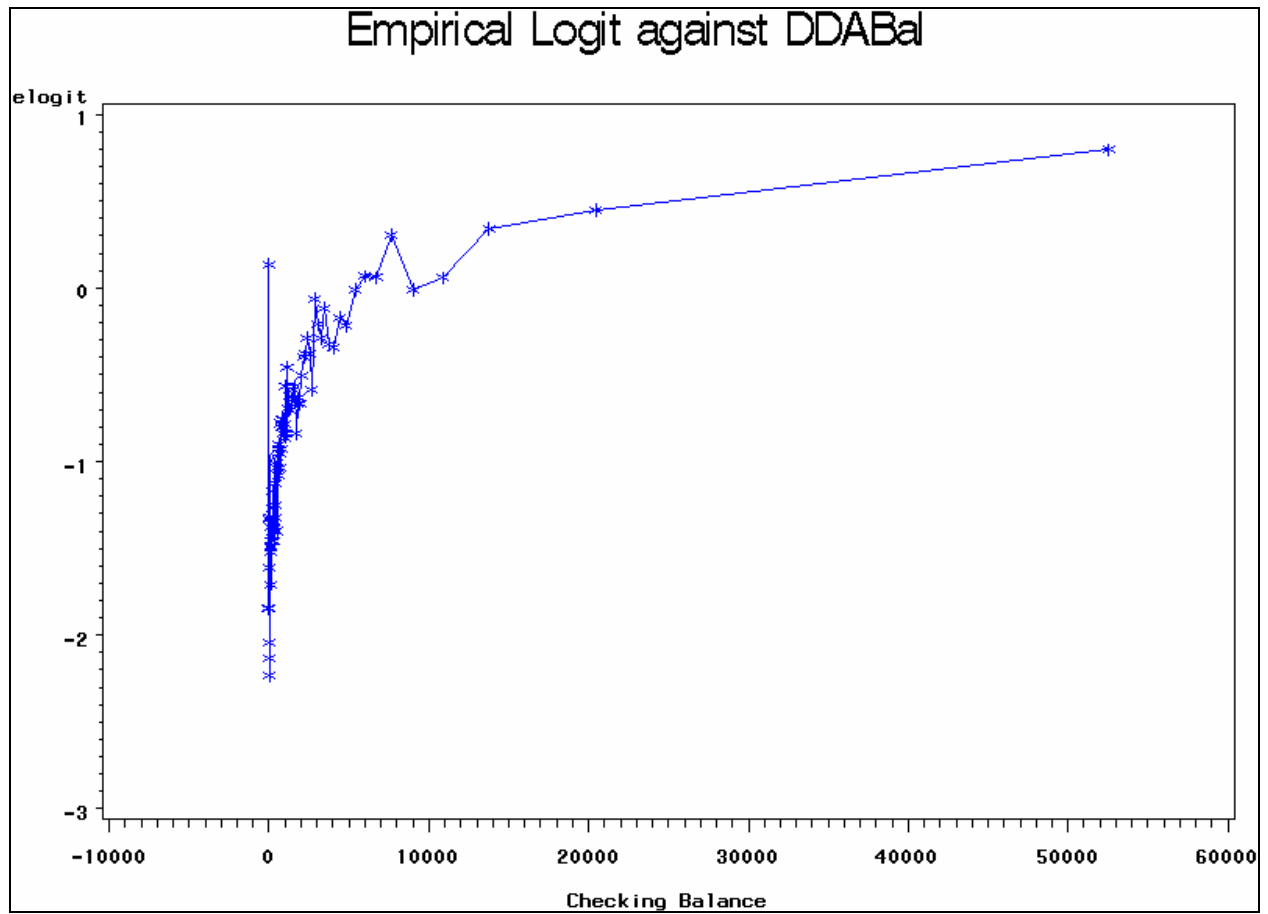
```
%let var=DDABal;
*****
%let var = SavBal;
proc rank data=train1 groups=100 out=out;
    var &var;
    ranks bin;
run;

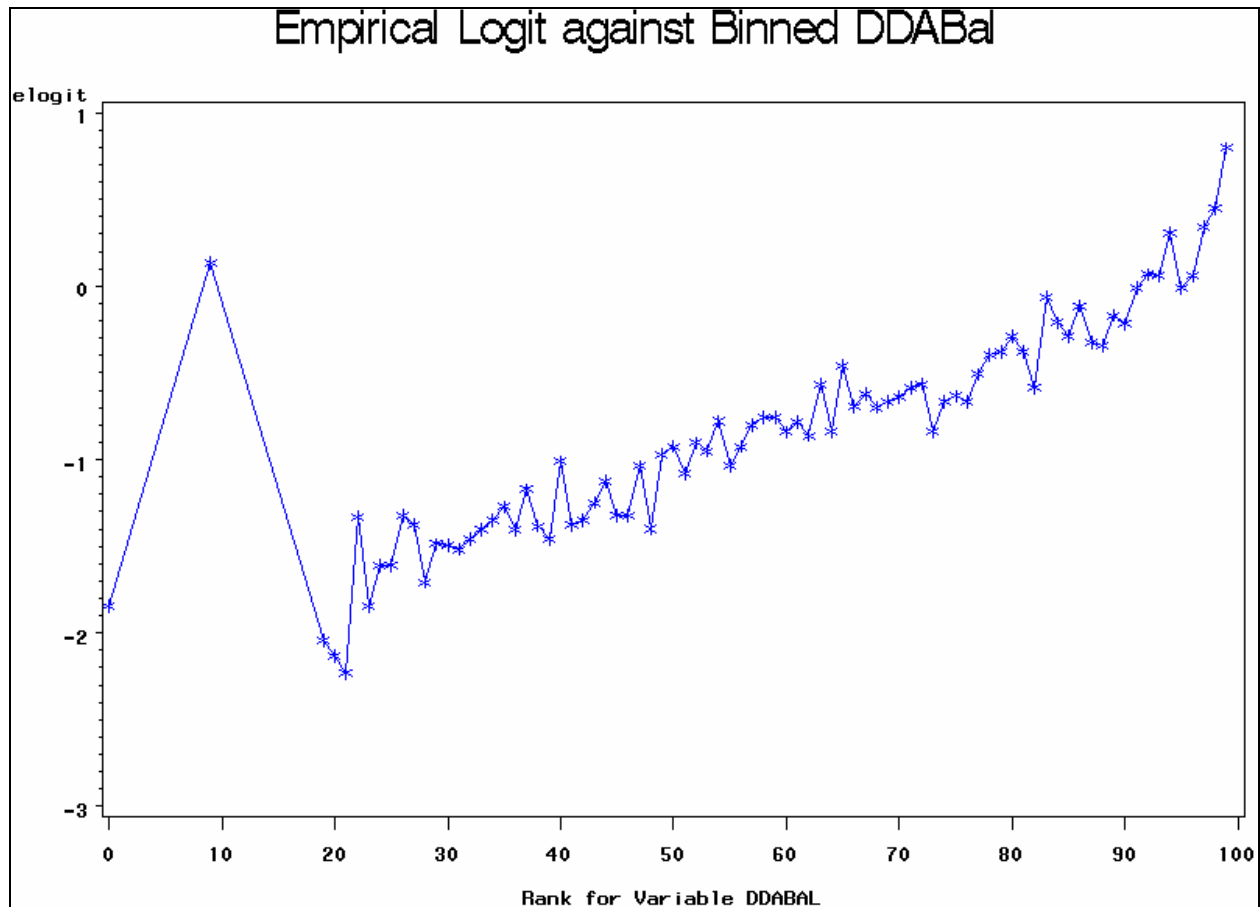
proc means data=out noprint nway;
    class bin;
    var ins &var;
    output out=bins sum(ins)=ins mean(&var)=&var;
run;

data bins;
    set bins;
    elogit=log((ins+(sqrt(_FREQ_)/2))/
                (_FREQ_-ins+(sqrt(_FREQ_)/2)));
run;

symbol i=join c=blue v=star;
proc gplot data = bins;
    title "Empirical Logit against &var";
    plot elogit * &var;
run;
    title "Empirical Logit against Binned &var";
    plot elogit * bin;
run; quit;
title;
```

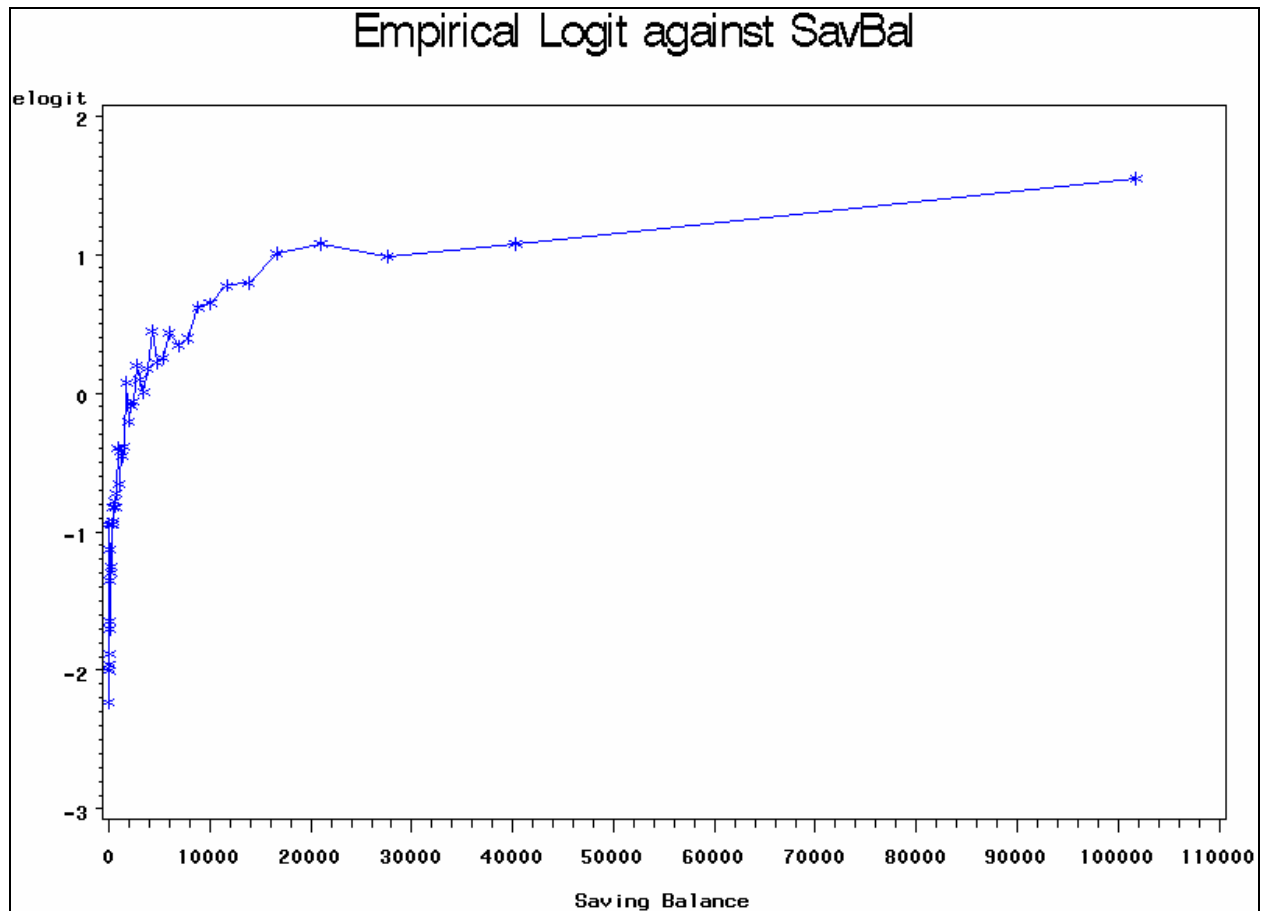
¹ You might also consider **DepAmt** and **ATMAmt** as skewed inputs.





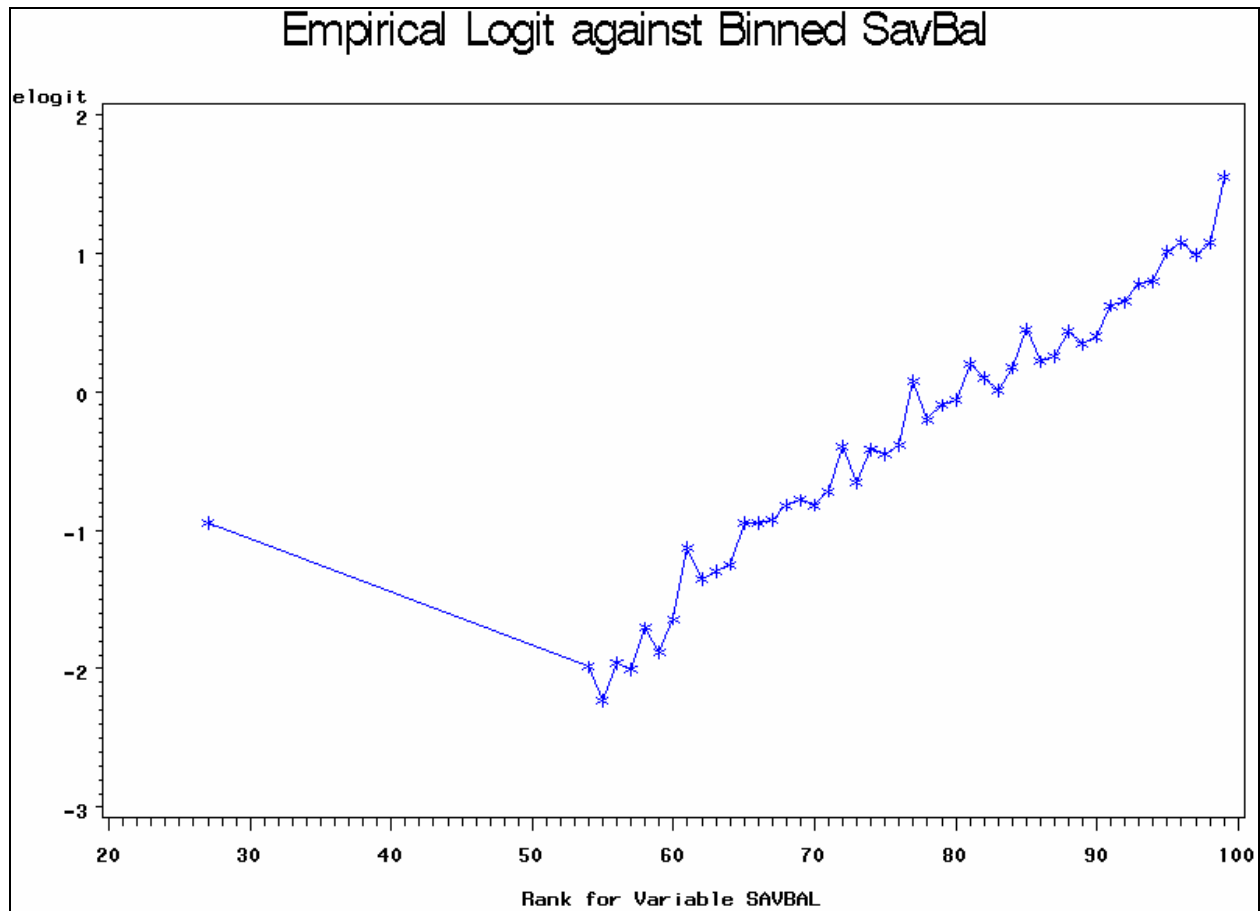
As before, the logical imputation of a \$0 balance for customers without a DDA account will be replaced with mean imputation to account for the spike. Also, the rank-transformed **DDABal** looks like it has a more linear relationship with the target, so that is an attractive option.

Performing a similar evaluation on **SavBal** reveals two things: the spike induced by logical imputation of \$0 balances for customer without saving accounts is not as drastic as it was for **DDABal**, and the relationship would be (roughly) linear if the top 5% or 10% of individuals had their balances capped at the 95th or 90th percentile. Hence, a reasonable “transformation” of **SavBal** might be simply replacing every balance greater than some number with that number.



This capping could be handled using the UNIVARIATE procedure (or the MEANS procedure) to find the percentile and using macro variables to pass that information into DATA step processing, but rudimentary visual inspection indicates that the 95th percentile of savings balance is around \$16,000. This appears to be a point where the relationship is turning a corner; hence, it seems like a good value for the cap.

The binned savings balance does not have the overwhelming linear relationship that was seen in the earlier **DDABal** plots. Hence, the transformation for the **SavBal** input is probably best left at this capping.



To handle these transformations, first tend to the **DDABal** variable. Replace the \$0 balances with the mean of those who have checking accounts for individuals who have no checking accounts.

```
proc sql;
  select mean(ddabal) into :mean
  from train1 where dda;
quit;
```

2620.85

```
data train1;
  set train1;
  if not dda then ddabal = &mean;
run;
```

The rank-group or percentile transformation looks like it will help linearize the relationship between the logits and the predictor. The MEANS procedure calculates the endpoints of the bins, and the DATA step is used to create DATA step code that will perform the rank-group transformation. The final DATA step uses the binning code to create **B_DDABal** and to truncate the larger values of **SavBal**.

```

proc rank data=train1 groups=100 out=out;
    var ddabal;
    ranks bin;
run;

proc means data = out noprint nway;
    class bin;
    var ddabal;
    output out=endpts max=max;
run;

filename rank "C:\temp\rank.sas";

data _null_;
    file rank;
    set endpts end=last;
    if _n_ = 1 then put "select;";
    if not last then do;
        put "    when (ddabal <= " max ") B_DDABal =" bin ";";
    end;
    else if last then do;
        put "otherwise B_DDABal =" bin ";";
        put "end;";
    end;
run;

data train1;
    set train1;
    %include rank /source2;
    if savbal > 16000 then savbal=16000;
run;

```

After performing the transformation, replace **DDABal** with the transformed input **B_DDABal** in the **screened** macro variable.

```

%let screened =
MIPhone Teller MM
Income ILS LOC POSAmt
NSFAmt CD CCPurc
ATMAmt brclus2 INV DEP
CashBk IRA CRScore
MIAcctAg IRABal MICRScor MTGBal
AcctAge SavBal B_DDABal SDB
InArea Sav Phone CCBal
INVBal MTG DEPAmt
DirDep ATM brclus1 Age;

```

Using, as an example, the fast backward elimination input selection algorithm, a model with 19 inputs is selected.

```

proc logistic data = train1 des;
  class res;
  model ins=&screened res
    / selection=backward fast slstay=.001;
run;

```

Partial Output

Summary of Backward Elimination

Step	Effect Removed	DF	Number In	Wald Chi-Square	Pr > ChiSq	Variable Label
1	CRScore	1	36	0.0006	0.9811	Credit Score
1	Income	1	35	0.0982	0.7541	Income
1	Age	1	34	0.1830	0.6688	Age
1	IRABal	1	33	0.2870	0.5922	IRA Balance
1	InvBal	1	32	0.3169	0.5735	
1	POSAmt	1	31	1.1900	0.2753	Amount Point of Sale
1	DepAmt	1	30	1.1876	0.2758	
1	InArea	1	29	1.2475	0.2640	Local Address
1	SDB	1	28	1.7789	0.1823	Safety Deposit Box
1	MICRScor	1	27	2.2677	0.1321	
1	Res	2	26	4.8357	0.0891	Area Classification
1	CashBk	1	25	3.1631	0.0753	Number Cash Back
1	CCPurc	1	24	4.1302	0.0421	Credit Card Purchases
1	NSFAmt	1	23	6.1459	0.0132	Amount NSF
1	ILS	1	22	6.2171	0.0127	Installment Loan
1	MTG	1	21	8.1064	0.0044	Mortgage
1	MIAcctAg	1	20	9.6674	0.0019	
1	brclus1	1	19	10.7599	0.0010	

Type 3 Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
MIPhone	1	64.0823	<.0001
Teller	1	46.8466	<.0001
MM	1	254.0652	<.0001
LOC	1	21.3013	<.0001
CD	1	414.9625	<.0001
ATMAmt	1	22.4368	<.0001
brclus2	1	56.1960	<.0001
Inv	1	28.0700	<.0001
Dep	1	27.7769	<.0001
IRA	1	19.4438	<.0001
MTGBal	1	14.8329	0.0001
AcctAge	1	55.4224	<.0001
SavBal	1	682.1551	<.0001
B_DDABal	1	803.8970	<.0001
Sav	1	36.6640	<.0001
Phone	1	11.2653	0.0008
CCBal	1	12.2771	0.0005
DirDep	1	20.1288	<.0001
ATM	1	12.9959	0.0003

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-2.0002	0.0578	1197.1425	<.0001
MIPhone	1	-0.4516	0.0564	64.0823	<.0001
Teller	1	0.0552	0.00806	46.8466	<.0001
MM	1	0.7898	0.0495	254.0652	<.0001
LOC	1	-0.3191	0.0691	21.3013	<.0001
CD	1	0.9625	0.0472	414.9625	<.0001
ATMAmt	1	0.000026	5.511E-6	22.4368	<.0001
brclus2	1	0.2736	0.0365	56.1960	<.0001
Inv	1	0.5373	0.1014	28.0700	<.0001
Dep	1	-0.0733	0.0139	27.7769	<.0001
IRA	1	0.3149	0.0714	19.4438	<.0001
MTGBal	1	-3.11E-6	8.063E-7	14.8329	0.0001
AcctAge	1	-0.0200	0.00269	55.4224	<.0001
SavBal	1	0.000129	4.955E-6	682.1551	<.0001
B_DDABal	1	0.0179	0.000631	803.8970	<.0001
Sav	1	0.2387	0.0394	36.6640	<.0001
Phone	1	-0.0631	0.0188	11.2653	0.0008
CCBal	1	2.897E-6	8.268E-7	12.2771	0.0005
DirDep	1	-0.1799	0.0401	20.1288	<.0001
ATM	1	-0.1455	0.0404	12.9959	0.0003

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
MIPhone	0.637	0.570	0.711
Teller	1.057	1.040	1.074
MM	2.203	1.999	2.428
LOC	0.727	0.635	0.832
CD	2.618	2.387	2.872
ATMAmt	1.000	1.000	1.000
brclus2	1.315	1.224	1.412
Inv	1.711	1.403	2.088
Dep	0.929	0.904	0.955
IRA	1.370	1.191	1.576
MTGBal	1.000	1.000	1.000
AcctAge	0.980	0.975	0.985
SavBal	1.000	1.000	1.000
B_DDABal	1.018	1.017	1.019
Sav	1.270	1.175	1.372
Phone	0.939	0.905	0.974
CCBal	1.000	1.000	1.000
DirDep	0.835	0.772	0.904
ATM	0.865	0.799	0.936

Association of Predicted Probabilities and Observed Responses

Percent Concordant	78.3	Somers' D	0.569
Percent Discordant	21.4	Gamma	0.570
Percent Tied	0.2	Tau-a	0.258
Pairs	104768511	c	0.784

The 19 inputs in the model are named in the macro variable **selected**.

```
%let selected =
MIPhone TELLER MM LOC CD
ATMAMT brclus2 INV DEP IRA
MTGBAL ACCTAGE SAVBAL B_DDABal SAV
PHONE CCBAL DIRDEP ATM;
```

To assess the model generalization performance, the validation data need to be prepared for scoring the same way that the training data was prepared for model building. Missing values need to be imputed, new inputs need to be created, and any transformations need to be applied. The MEANS procedure can be used to see which inputs need imputation on this **valid** data set.

```
proc means data = valid nmiss;
  var MIPhone Teller MM LOC CD
      ATMamt INV DEP IRA MTGBal
      AcctAge SavBal DDABal Sav
      Phone CCBal DirDep ATM;
run;
```

The MEANS Procedure

Variable	Label	N Miss
MIPhone		0
Teller	Teller Visits	0
MM	Money Market	0
LOC	Line of Credit	0
CD	Certificate of Deposit	0
ATMAmt	ATM Withdrawal Amount	0
Inv		1369
Dep		0
IRA	Retirement Account	0
MTGBal	Mortgage Balance	0
AcctAge	Age of Oldest Account	687
SavBal	Saving Balance	0
DDABal	Checking Balance	0
Sav	Saving Account	0
Phone	Number Telephone Banking	1369
CCBal	Credit Card Balance	1369
DirDep	Direct Deposit	0
ATM	ATM	0

In the validation data set, missing values should be replaced with the medians from the training data set. Because the variables **AcctAge**, **Phone**, **Inv** and **CCBal** have missing values, PROC UNIVARIATE is used to create an output data set with the medians of those variables. The PCTLPTS option requests the 50th percentile and the PCTLPRE option specifies the prefix for the variable names in the output data set.

```

proc univariate data=train1 noprint;
  var acctage phone inv ccbal;
  output out=medians
        pctlpts=50
        pctlpre=acctage phone inv ccbal;
run;

```

The DATA step first combines values from a single observation in one data set (**Medians**) with all of the observations in another data set (**Valid1**). Array X contains the variables with missing values and array MED contains the variables with the medians. The DO loop replaces the missing values with the medians. The necessary **BrClus2** variable, the **B_DDABal** rank-transformed input, and the truncated **SavBal** are all added at this stage.

```

data valid1(drop=acctage50 phone50 inv50 ccbal50 i);
  if _N_ = 1 then set medians;
  set valid;
  array x(*) acctage phone inv ccbal;
  array med(*) acctage50 phone50 inv50 ccbal50;
  do i = 1 to dim(x);
    if x(i)=. then x(i)=med(i);
  end;
  if not dda then ddabal = &mean;
  brclus1=(branch='B14');
  brclus2=(branch in ('B12','B5','B8',
                    'B3','B18','B19','B17',
                    'B4','B6','B10','B9',
                    'B1','B13'));
  brclus3=(branch in ('B15','B16'));
  %include rank;
  if savbal > 16000 then savbal=16000;
run;

```



BrClus1 and **BrClus3** are not used by this model, but if you are considering several possible models, it may be easier to create one validation data set that has received exactly the same treatment as the training data set so that assessing model performance is simplified. To that end, it may be useful to know about the code that follows.

Now that the validation data is prepared for scoring, you could use any of the techniques discussed in Chapter 2 to score it. However, this begs a question: what metrics can you use to measure model performance? This question and the related question of optimal use of a model are the foci of the following sections.

Imputation for All Inputs

The STDIZE procedure can be used to output a data set that contains the relevant information about the imputed values for every input. In addition, the STDIZE procedure can also be used to take that information and use it to impute the training data set values in the validation data set. An example follows.

As before, use the STDIZE procedure with the REONLY option to impute missing values on the training data. In addition, specify the OUTSTAT= option to save the imputed values in a separate data set, called **med**.

```
proc stdize data = train out=train2
            method=median reonly
            OUTSTAT=med;
    var &inputs;
run;
```

After the **med** data set has been created, it can be used to impute for missing values in a different data set. The code below creates a data set, **valid2**, based on the unimputed **valid** data, with the medians from the training data imputed. The option to specify the data set with the median information is METHOD=IN(*data-set-name*).

```
proc stdize data=valid out=valid2
            reonly method=in(med) ;
    var &inputs;
run;
```

To see that the values imputed for **AcctAge**, **Phone**, **Inv**, and **CCBal** are the same in **valid1** (created above) as in **valid2** (created here) you can use the COMPARE procedure.

```
proc compare base= valid1 compare=valid2;
    var acctage phone inv ccbal;
run;
```

The COMPARE Procedure
Comparison of WORK.VALID1 with WORK.VALID2
(Method=EXACT)

Data Set Summary

Dataset	Created	Modified	NVar	NObs
WORK.VALID1	30DEC04:12:34:47	30DEC04:12:34:47	70	10752
WORK.VALID2	30DEC04:12:34:34	30DEC04:12:34:34	66	10752

Variables Summary

Number of Variables in Common: 66.
Number of Variables in WORK.VALID1 but not in WORK.VALID2: 4.
Number of VAR Statement Variables: 4.

Observation Summary

Observation	Base	Compare
First Obs	1	1
Last Obs	10752	10752

Number of Observations in Common: 10752.
Total Number of Observations Read from WORK.VALID1: 10752.
Total Number of Observations Read from WORK.VALID2: 10752.

Number of Observations with Some Compared Variables Unequal: 0.
Number of Observations with All Compared Variables Equal: 10752.

NOTE: No unequal values were found. All values compared are exactly equal.

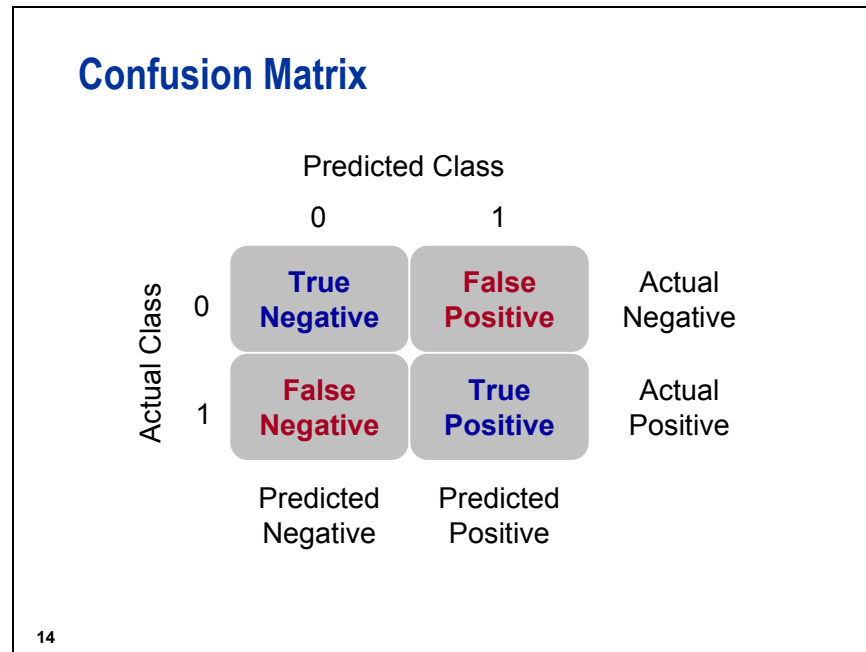
The **train2** and **valid2** data sets will be used in the next chapter; in order to generate a series of models, you probably want to impute for all inputs first. This way, you can evaluate model performance on the validation data without having to intervene by hand to impute for missing values.

4.02 Multiple Choice Poll

Which variables seem to have a nonlinear relationship with the target?

- a. RECENT_AVG_CARD_GIFT_AMT and LIFETIME_GIFT_RANGE
- b. DONOR_AGE and PER_CAPITA_INCOME
- c. RECENT_RESPONSE_COUNT and RECENT_STAR_STATUS
- d. INCOME_GROUP and PCT_MALE_MILITARY

4.2 Misclassification



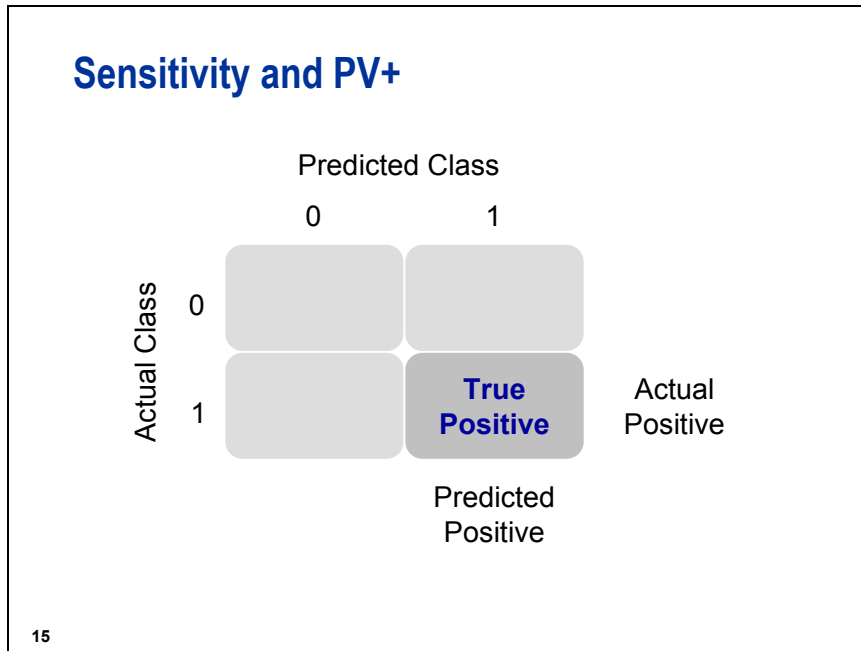
Supervised classification does not usually end with an estimate of the posterior probability. An allocation rule corresponds to a threshold value (cutoff) of the posterior probability. For example, all cases with probabilities of default greater than .04 might be rejected for a loan. For a given cutoff, how well does the classifier perform?

The fundamental assessment tool is the confusion matrix. The *confusion matrix* is a crosstabulation of the actual and predicted classes. It quantifies the confusion of the classifier. The event of interest, whether it is unfavorable (like fraud, churn, or default) or favorable (like response to offer), is often called a positive, although this convention is arbitrary. The simplest performance statistics are *accuracy*

$$(\text{true positives and negatives}) / (\text{total cases})$$

and *error rate*

$$(\text{false positives and negatives}) / (\text{total cases}).$$



Two specialized measures of classifier performance are *sensitivity*

$$(\text{true positives}) / (\text{total actual positives})$$

and *positive predicted value* (PV+)

$$(\text{true positives}) / (\text{total predicted positives}).$$

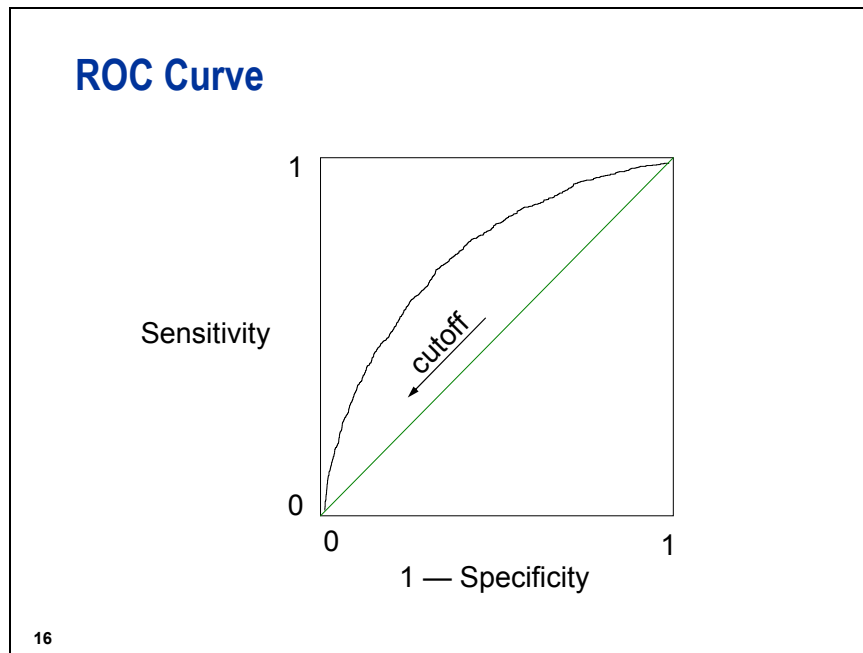
The analogues to these measures for true negatives are *specificity*

$$(\text{true negatives}) / (\text{total actual negatives})$$

and *negative predicted value* (PV−)

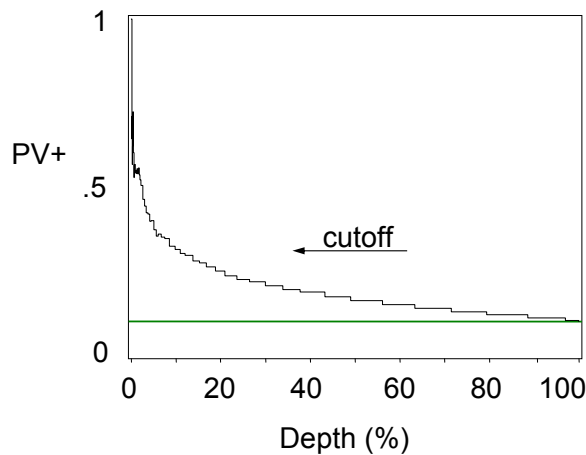
$$(\text{true negatives}) / (\text{total predicted negatives}).$$

Large sensitivities do not necessarily correspond to large values of PV+. Ideally, one would like large values of all these statistics. The context of the problem determines which of these measures is the primary concern. For example, a database marketer might be most concerned with PV+ because it gives the response rate for the customers that receive an offer. In contrast, a fraud investigator might be most concerned with sensitivity because it gives the proportion of frauds that would be detected.



The *receiver operating characteristic* (ROC) curve was adapted from signal detection theory for the assessment of classifiers. The ROC curve displays the sensitivity and specificity for the entire range of cutoff values. As the cutoff decreases, more and more cases are allocated to class 1; hence, the sensitivity increases and specificity decreases. As the cutoff increases, more and more cases are allocated to class 0, hence the sensitivity decreases and specificity increases. Consequently, the ROC curve intersects (0,0) and (1,1). If the posterior probabilities were arbitrarily assigned to the cases, then the ratio of false positives to true positives would be the same as the ratio of the total actual negatives to the total actual positives. Consequently, the baseline (random) model is a 45° angle going through the origin. As the ROC curve bows above the diagonal, the predictive power increases. A perfect model would reach the (0,1) point where both sensitivity and specificity equal 1.

Gains Chart



17

The *depth* of a classification rule is the total proportion of cases that were allocated to class 1. The (cumulative) *gains chart* displays the positive predicted value and depth for a range of cutoff values. As the cutoff decreases, more and more cases are allocated to class 1; hence, the depth increases and the PV+ approaches the marginal event rate. When the cutoff is minimum, then 100% of the cases are selected and the response rate is p_1 . As the cutoff increases the depth decreases. A model with good predictive power would have increasing PV+ (response rate) as the depth decreases. If the posterior probabilities were arbitrarily assigned to the cases, then the gains chart would be a horizontal line at p_1 .

The gains chart is widely used in database marketing to decide how deep in a database to go with a promotion. The simplest way to construct this curve is to sort and bin the predicted posterior probabilities (for example, deciles). The gains chart is easily augmented with revenue and cost information. The *lift* is $PV+/p_1$, so for a given depth, there are (lift)× more responders targeted by the model than by random chance.

A plot of sensitivity versus depth is sometimes called a *Lorentz curve*, *concentration curve*, or a *lift curve* (although lift value is not explicitly displayed). This plot and the ROC curve are very similar because depth and 1-specificity are monotonically related.

Oversampled Test Set

		Predicted		
		0	1	
Actual	0	29	21	50
	1	17	33	50
		46	54	
Sample				

		Predicted		
		0	1	
Actual	0	56	41	97
	1	1	2	3
		57	43	
Population				

18

If the holdout data was obtained by splitting oversampled data, then it is oversampled as well. If the proper adjustments were made when the model was fitted, then the predicted posterior probabilities are correct. However, the confusion matrices would be incorrect (with regard to the population) because the event cases are over-represented. Consequently, PV+ (response rate) might be badly overestimated. Sensitivity and specificity, however, are not affected by separate sampling because they do not depend on the proportion of each class in the sample.

Adjustments for Oversampling

		Predicted Class		
		0	1	
Actual Class	0	$\pi_0 \cdot Sp$	$\pi_0(1 - Sp)$	π_0
	1	$\pi_1(1 - Se)$	$\pi_1 \cdot Se$	π_1

19

Knowing sensitivity, specificity, and the priors is sufficient for adjusting the confusion matrix for oversampling. For example, if the sample represented the population, then $n\pi_1$ cases are in class 1. The proportion of those that were allocated to class 1 is Se . Thus, there are $n\pi_1 \cdot Se$ true positives. Note that these adjustments are equivalent to multiplying the cell counts by their sample weights, for example

$$TP_{\text{sample}} \cdot wt_1 = TP_{\text{sample}} \frac{\pi_1}{\rho_1} = TP_{\text{sample}} \cdot \pi_1 \frac{n}{\text{Tot Pos}_{\text{sample}}} = n \cdot \pi_1 \cdot Se$$

where **TP** is the proportion of true positives, and sample weights are defined as π_i / ρ_i for Class i .

4.03 Multiple Choice Poll

The formula for sensitivity is

- true positives / total actual positives
- true positives / total predicted positives
- true negatives / total actual negatives
- true negatives / total predicted negatives

21



Assessing Classifier Performance

The performance measures need to be calculated on the validation data. One approach would be to use the SCORE procedure to score the validation data and then use DATA steps and the FREQ procedure to calculate misclassification measures for different cutoffs. An easier approach is to score the validation data inside the LOGISTIC procedure and use the OUTROC= data set, which contains many of the statistics necessary for assessment. The SCORE statement allows for you to output this data set.

The OUTROC= option creates an output data set with sensitivity (**_SENSIT_**) and one minus specificity (**_1MSPEC_**) calculated for a full range of cutoff probabilities (**_PROB_**). The other statistics in the OUTROC= data set are not useful when the data is oversampled. The two variables **_SENSIT_** and **_1MSPEC_** in the OUTROC= data set are correct whether or not the validation data is oversampled. The variable **_PROB_** is correct, provided the PRIOREVENT= was set to π_1 . If they were not corrected, then **_PROB_** needs to be adjusted using the formula

$$\hat{p}_i = \frac{\hat{p}_i^* \rho_o \pi_1}{(1 - \hat{p}_i^*) \rho_1 \pi_o + \hat{p}_i^* \rho_o \pi_1}$$

where \hat{p}_i^* is the unadjusted estimate of the posterior probability (**_PROB_**). The **Scoval** (scored validation) data set will be used later.

```
proc logistic data=train1 des;
  model ins=&selected;
  score data=valid1 out=scoval
        priorevent=&pil outroc=roc;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.TRAIN1
Response Variable	Ins
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	21512
Number of Observations Used	21512

Response Profile

Ordered Value	Ins	Total Frequency
1	1	7451
2	0	14061

Probability modeled is Ins=1.

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	27759.675	22896.395
SC	27767.651	23055.922
-2 Log L	27757.675	22856.395

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	4901.2800	19	<.0001
Score	4644.4110	19	<.0001
Wald	3609.9677	19	<.0001

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-2.0002	0.0578	1197.1425	<.0001
MIPhone	1	-0.4516	0.0564	64.0823	<.0001
Teller	1	0.0552	0.00806	46.8466	<.0001
MM	1	0.7898	0.0495	254.0652	<.0001
LOC	1	-0.3191	0.0691	21.3013	<.0001
CD	1	0.9625	0.0472	414.9625	<.0001
ATMAmt	1	0.000026	5.511E-6	22.4368	<.0001
brclus2	1	0.2736	0.0365	56.1960	<.0001
Inv	1	0.5373	0.1014	28.0700	<.0001
Dep	1	-0.0733	0.0139	27.7769	<.0001
IRA	1	0.3149	0.0714	19.4438	<.0001
MTGBal	1	-3.11E-6	8.063E-7	14.8329	0.0001
AcctAge	1	-0.0200	0.00269	55.4224	<.0001
SavBal	1	0.000129	4.955E-6	682.1551	<.0001
B_DDABal	1	0.0179	0.000631	803.8970	<.0001
Sav	1	0.2387	0.0394	36.6640	<.0001
Phone	1	-0.0631	0.0188	11.2653	0.0008
CCBal	1	2.897E-6	8.268E-7	12.2771	0.0005
DirDep	1	-0.1799	0.0401	20.1288	<.0001
ATM	1	-0.1455	0.0404	12.9959	0.0003

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
MIPhone	0.637	0.570	0.711
Teller	1.057	1.040	1.074
MM	2.203	1.999	2.428
LOC	0.727	0.635	0.832
CD	2.618	2.387	2.872
ATMAmt	1.000	1.000	1.000
brclus2	1.315	1.224	1.412
Inv	1.711	1.403	2.088
Dep	0.929	0.904	0.955
IRA	1.370	1.191	1.576
MTGBal	1.000	1.000	1.000
AcctAge	0.980	0.975	0.985
SavBal	1.000	1.000	1.000
B_DDABal	1.018	1.017	1.019
Sav	1.270	1.175	1.372
Phone	0.939	0.905	0.974
CCBal	1.000	1.000	1.000
DirDep	0.835	0.772	0.904
ATM	0.865	0.799	0.936
Association of Predicted Probabilities and Observed Responses			
Percent Concordant	78.3	Somers' D	0.569
Percent Discordant	21.4	Gamma	0.570
Percent Tied	0.2	Tau-a	0.258
Pairs	104768511	c	0.784

```
proc print data=roc(obs=25);
  var _prob_ _sensit_ _lmspec_;
run;
```

Obs	_PROB_	_SENSIT_	_1MSPEC_
1	0.99994	.000268528	.000000000
2	0.75575	.000537057	.000000000
3	0.73370	.000805585	.000000000
4	0.72118	.001074114	.000000000
5	0.67843	.001074114	.000142288
6	0.67313	.001342642	.000142288
7	0.66172	.001611171	.000142288
8	0.64687	.001879699	.000142288
9	0.59700	.002148228	.000142288
10	0.57014	.002685285	.000142288
11	0.56990	.002953813	.000142288
12	0.56719	.003222342	.000142288
13	0.56610	.003222342	.000284576
14	0.56188	.003490870	.000284576
15	0.56174	.003759398	.000284576
16	0.55953	.003759398	.000426864
17	0.55464	.004027927	.000426864
18	0.53059	.004296455	.000426864
19	0.52356	.004564984	.000426864
20	0.52049	.004564984	.000569152
21	0.50546	.005102041	.000569152
22	0.49185	.005370569	.000569152
23	0.48751	.005370569	.000711440
24	0.47946	.005639098	.000711440
25	0.47150	.005907626	.000711440

Knowledge of the population priors and sensitivity and specificity is sufficient to fill in the confusion matrices. Several additional statistics can be calculated in a DATA step:

- TP = proportion of true positives
- FN = proportion of false negatives
- TN = proportion of true negatives
- FP = proportion of false positives
- POSPV = positive predicted value
- NEGPPV = negative predicted value
- ACC = accuracy
- DEPTH = proportion allocated to class 1
- LIFT = positive predicted value/ π_1 .

Each row in the OUTROC= data set corresponds to a cutoff (**_PROB_**). The selected cutoffs occur where values of the estimated posterior probability change, provided the posterior probabilities are more than .0001 apart, otherwise they are grouped. Consequently, the maximum number of rows in the OUTROC= data set is 9999, but it is usually much less. The grouping can be made coarser by using the ROCEPS= option in the MODEL statement.

```

data roc;
  set roc;
  cutoff=_PROB_;
  specif=1-_1MSPEC_;
  tp=&pi1*_SENSIT_;
  fn=&pi1*(1-_SENSIT_);
  tn=(1-&pi1)*specif;
  fp=(1-&pi1)*_1MSPEC_;
  depth=tp+fp;
  pospv=tp/depth;
  negpv=tn/(1-depth);
  acc=tp+tn;
  lift=pospv/&pi1;
  keep cutoff tn fp fn tp
      _SENSIT_ _1MSPEC_ specif depth
      pospv negpv acc lift;
run;

```

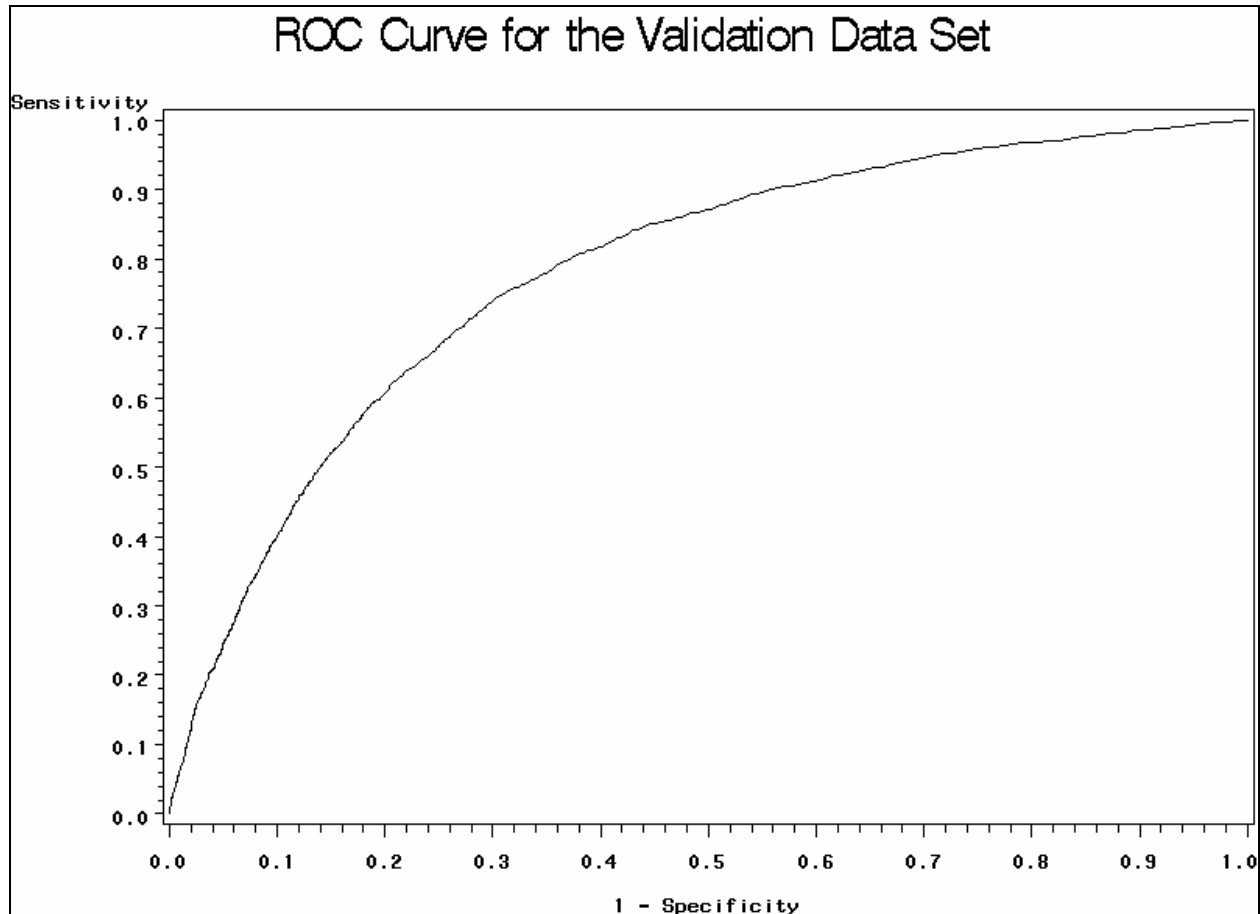
ROC curves can be created by plotting **_SENSIT_** versus **_1MSPEC_**. Gains charts can be created by plotting **PosPV** versus **Depth** or **Lift** versus **Depth**. Often, overlying many statistics versus **Depth** is informative. These plots follow.

To generate the ROC curve, use the GPLOT procedure. The SYMBOL statement defines the line that will connect the points: join the points (I=JOIN), use no symbol to plot each point (V=NONE), and color the line black.

```

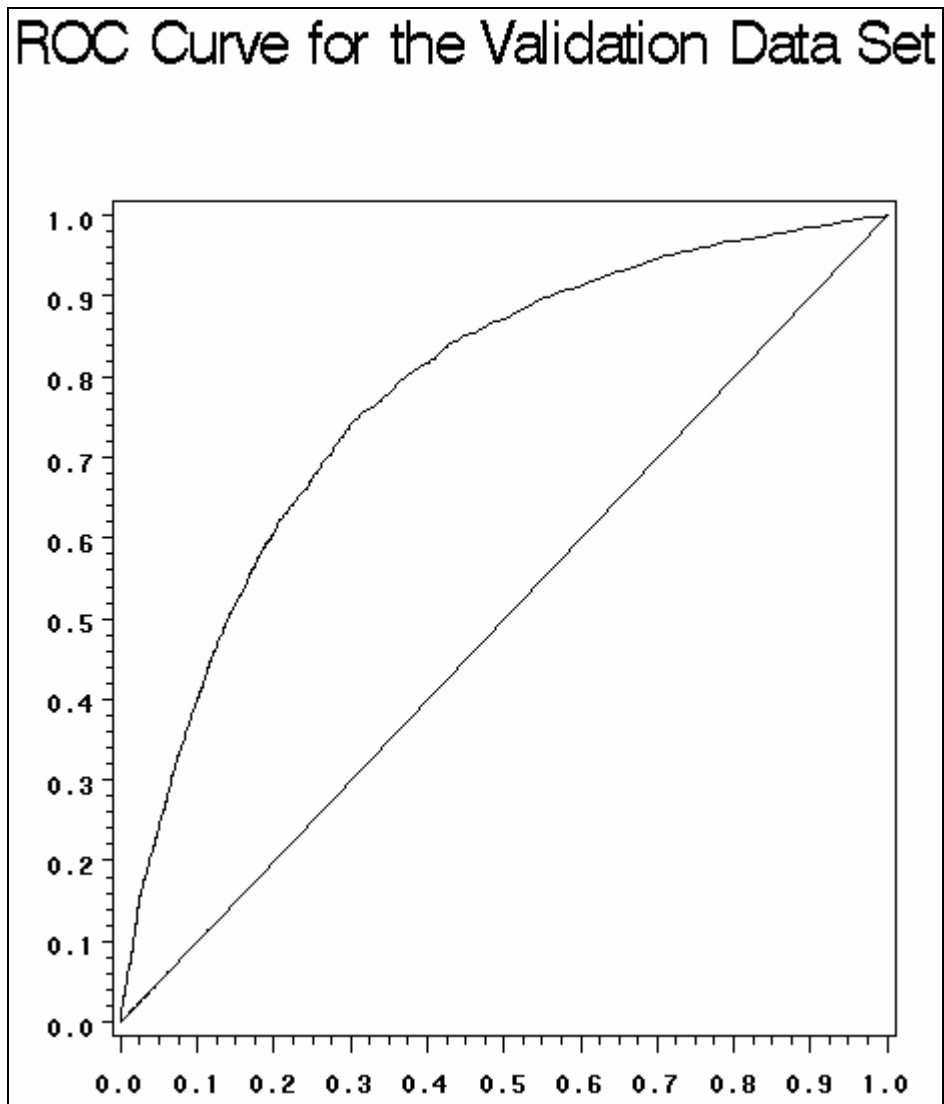
symbol i=join v=none c=black;
proc gplot data = roc;
  title "ROC Curve for the Validation Data Set";
  plot _SENSIT_*_1MSPEC_;
run; quit;

```



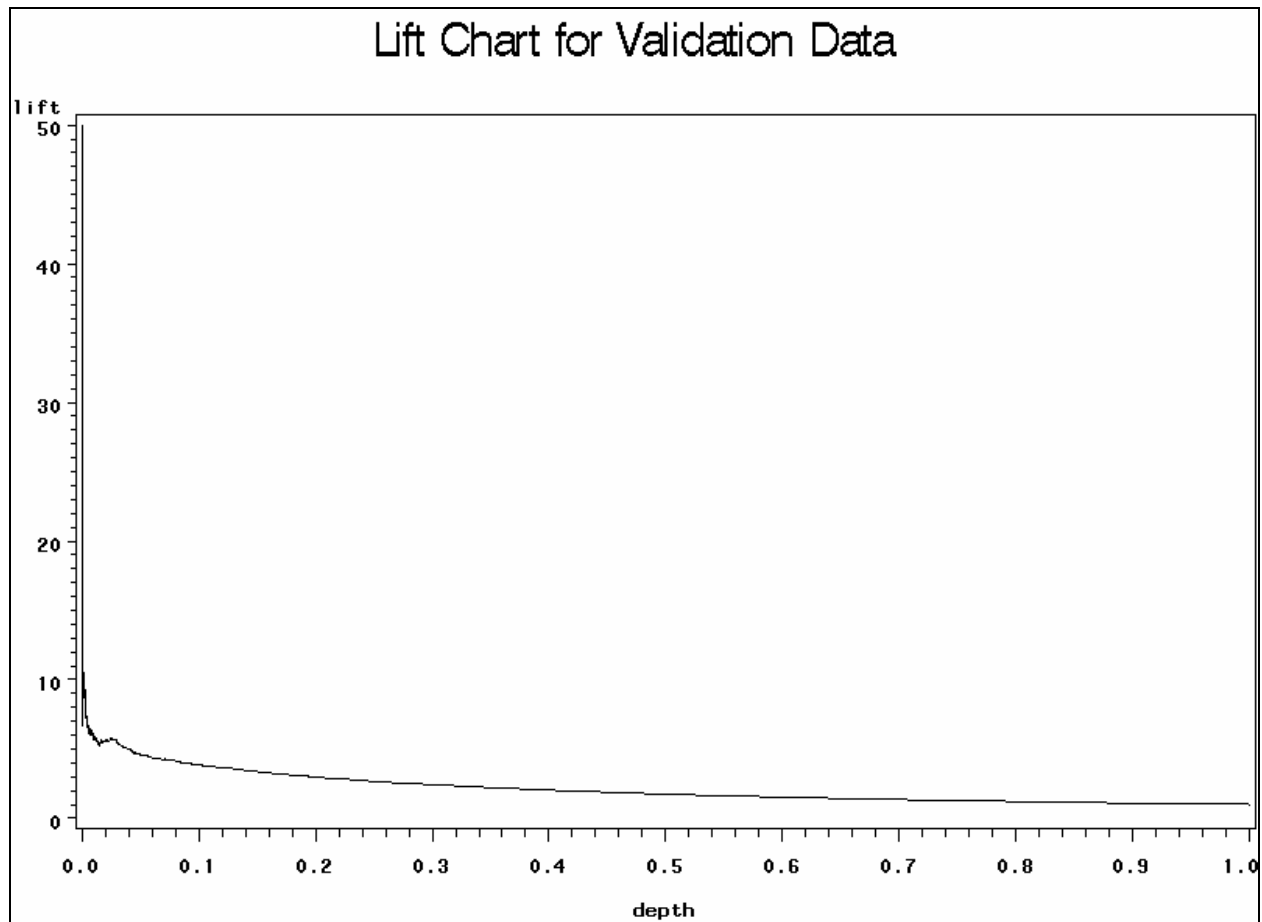
This plot could be improved, if desired, by making the axes of equal length. Also, the baseline through (0,0) and (1,1) is a useful addition. Finally, the labels on the axes are redundant. An ROC curve is defined by the values plotted, so you could remove the labels. The length of the axes can be controlled in the `AXIS` statement with the `LENGTH=` option and the labels can be removed from the axes by specifying `LABEL=NONE`. The `ORDER=` option in the `AXIS` statement enables you to control the major tick marks. To overlay the ROC curve with the straight line, you can add `_1MSPEC_ * _1MSPEC_` to the `PLOT` statement and specify the `OVERLAY` option following a slash. Since you will be plotting two sets of points, a `SYMBOL2` statement specifies the plotting options for the second set of points listed.

```
axis order=(0 to 1 by .1) label=None length=4in;
symbol i=join v=None c=black;
symbol2 i=join v=None c=black;
proc gplot data = roc;
    title "ROC Curve for the Validation Data Set";
    plot _SENSIT_ * _1MSPEC_ _1MSPEC_ * _1MSPEC_
        / overlay vaxis=axis haxis=axis;
run; quit;
```



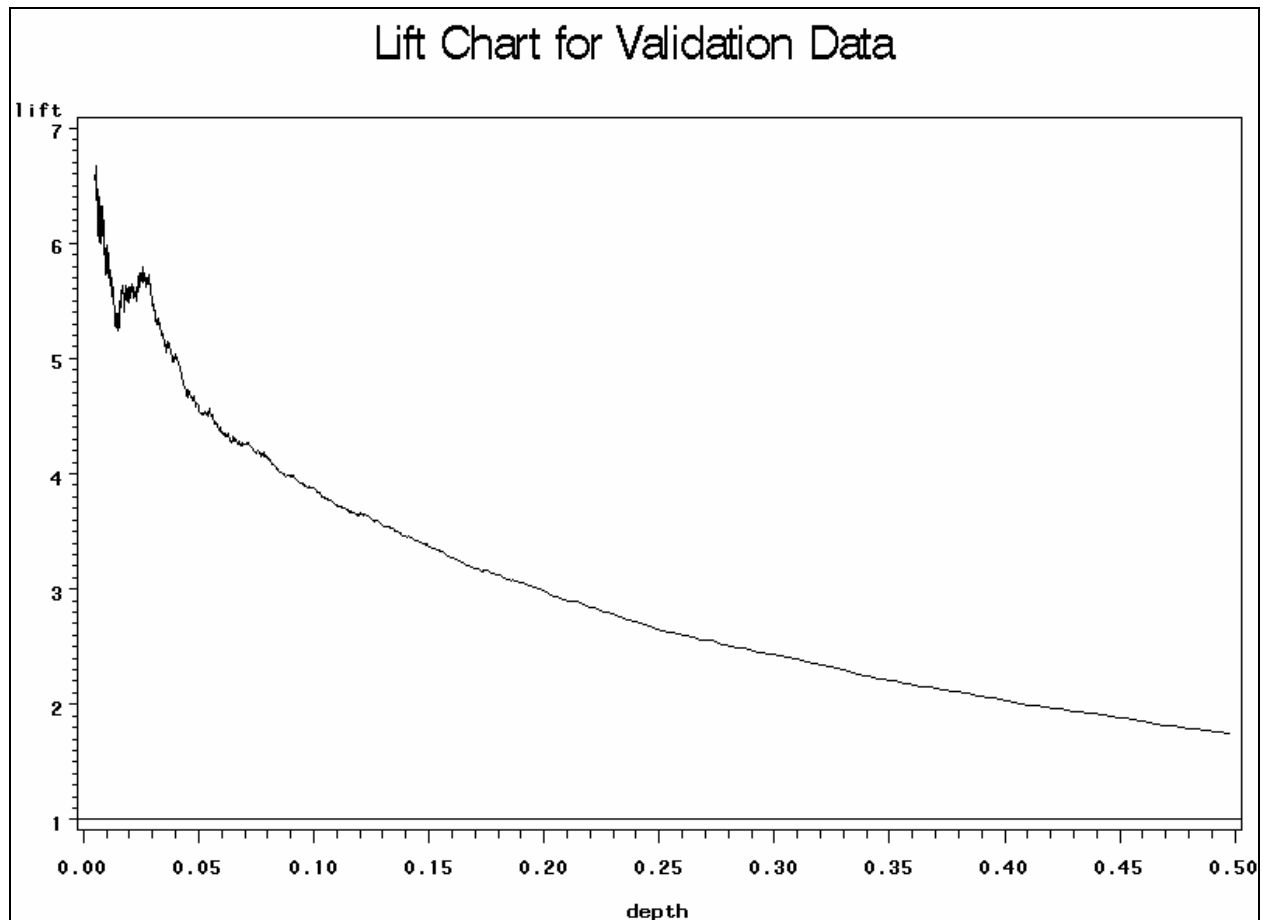
To create a lift chart, plot **Lift** against **Depth**.

```
symbol i=join v=none c=black;  
proc gplot data=roc;  
    title "Lift Chart for Validation Data";  
    plot lift*depth;  
run; quit;
```



To improve this graph, consider adding a reference line at the base line (a lift of 1) and restricting the focus to the region where depth is greater than 0.5% (less erratic) and less than 50%. The final TITLE statement, with no options, clears the title. The VREF= option in the PLOT statement puts a reference line on the vertical axis.

```
symbol i=join v=none c=black;
proc gplot data=roc;
  where 0.005 < depth < 0.50;
  title "Lift Chart for Validation Data";
  plot lift*depth / vref=1;
run; quit;
title;
```

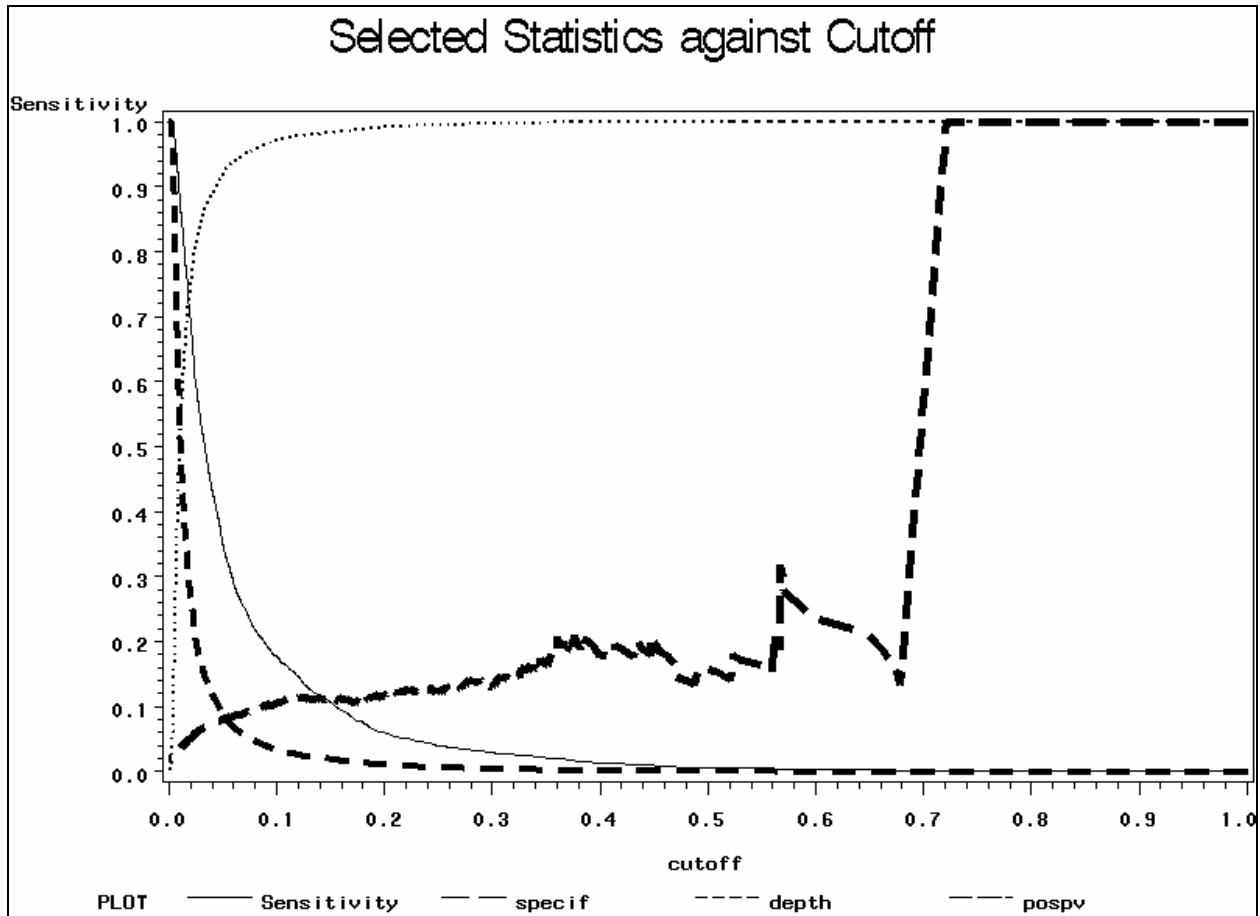


Finally, a plot of sensitivity, specificity, positive predicted value, and depth against cutoff may help guide you to an appropriate cutoff.² Allocation rules are discussed in more detail in the next section, but considering a plot that shows how several statistics of interest react to changes in the cutoff may be useful.

Specify the plotting controls in SYMBOL statements, as before. To be able to discriminate between lines, you can specify L= for different line types (from 1 to 46), and you can change the width of the lines with the W= option. Finally, to include a legend with your plot, include the LEGEND option in the PLOT statement.

```
symbol i=join v=none c=black l=1 w=1;
symbol2 i=join v=none c=black l=2 w=2;
symbol3 i=join v=none c=black l=3 w=3;
symbol4 i=join v=none c=black l=4 w=4;
proc gplot data=roc;
  title "Selected Statistics against Cutoff";
  plot (_SENSIT_ specif depth pospv)*cutoff
    / overlay legend;
run; quit;
```

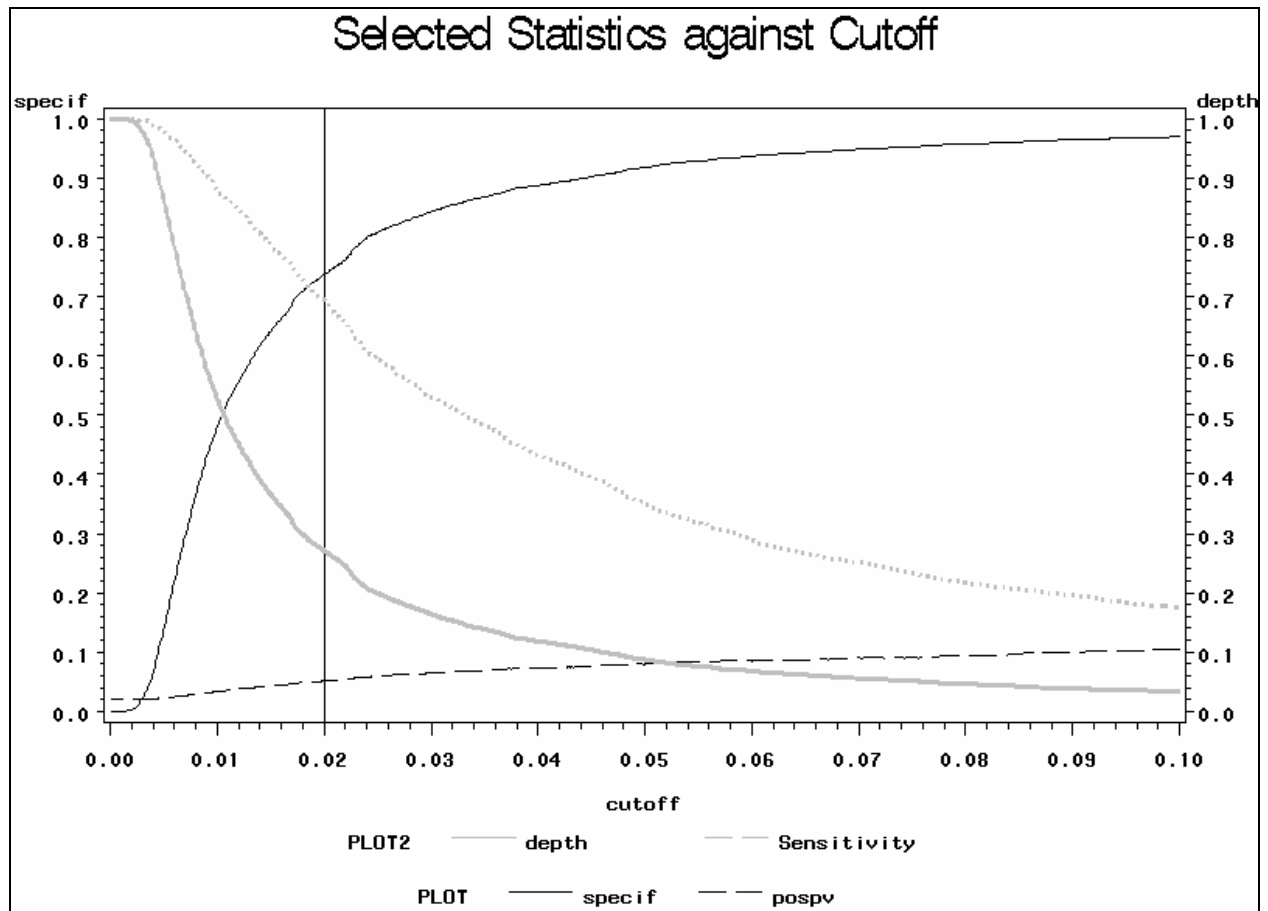
² Recall that cutoff refers to the probability that defines your decision rule. Individuals with a posterior probability greater than the cutoff are predicted to be events.



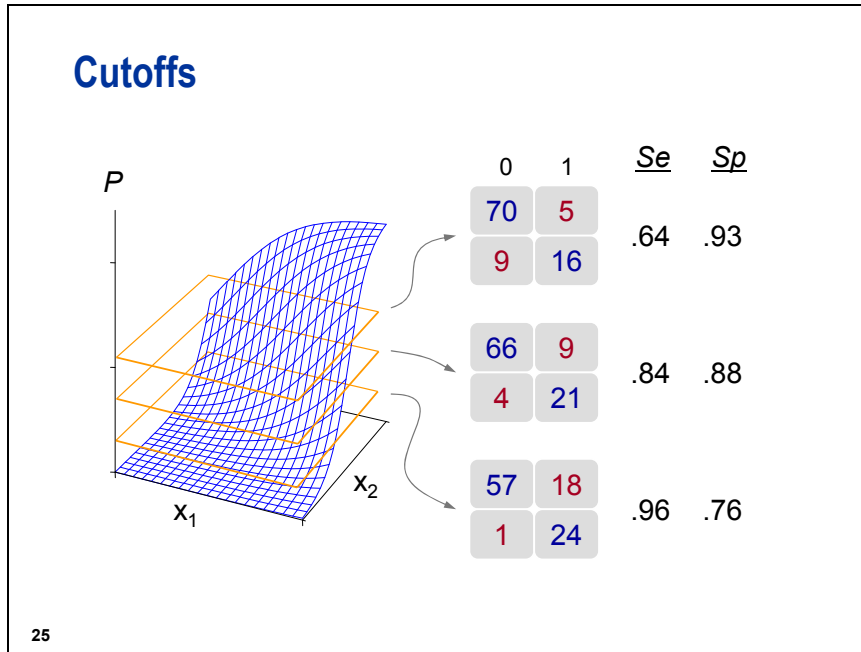
This plot could be improved by a few simple steps. With a target this rare, evaluating cutoffs above 0.1 makes little sense; restrict the focus. In that range, the depth and sensitivity lines are on a different scale than the specificity and positive predicted value. Hence, plot depth and sensitivity on the right axis. Finally, add a reference line to the horizontal axis at a cutoff of π_1 .

The SYMBOL statements can be used to differentiate lines. In the GPLOT procedure, the PLOT2 statement has syntax just like the PLOT statement, except it plots against the right side axis. The HREF= option specifies the point at which to draw a reference line on the horizontal axis. C=LIGR specifies that those lines should be drawn in light gray.

```
symbol i=join v=None c=black l=1 w=1;
symbol2 i=join v=None c=black l=2 w=1;
symbol3 i=join v=None c=ligr l=1 w=3;
symbol4 i=join v=None c=ligr l=2 w=3;
proc gplot data=roc;
  where cutoff < 0.1;
  title "Selected Statistics against Cutoff";
  plot (specif pospv)*cutoff
    / overlay legend href=&pi1;
  plot2 (depth_sensit_)*cutoff
    /overlay legend;
run; quit;
title;
```



4.3 Allocation Rules



Different cutoffs produce different allocations and different confusion matrices. To determine the optimal cutoff, a performance criterion needs to be defined. If the goal were to increase the sensitivity of the classifier, then the optimal classifier would allocate all cases to class 1. If the goal were to increase specificity, then the optimal classifier would be to allocate all cases to class 0. For realistic data, there is a trade-off between sensitivity and specificity. Higher cutoffs decrease sensitivity and increase specificity. Lower cutoffs decrease specificity and increase sensitivity.

Misclassification Costs

		Predicted				Total Cost
		0	1			
Actual	0	0	1	70	5	$9 \cdot 4 + 5 = 41$
	1	4	0	9	16	
	0			66	9	$4 \cdot 4 + 9 = 25$
	1			4	21	
				57	18	$1 \cdot 4 + 18 = 22$
				1	24	

A formal approach to determining the optimal cutoff uses statistical decision theory (McLachlan 1992; Ripley 1996; Hand 1997). The decision-theoretic approach starts by assigning *misclassification costs* (losses) to each type of error (false positives and false negatives). The optimal decision rule minimizes the total expected cost (*risk*).

Bayes Rule

Allocate to class 1 if

$$p_i > \frac{1}{1 + \left(\frac{\text{cost}_{\text{FN}}}{\text{cost}_{\text{FP}}} \right)}$$

Allocate to class 0, otherwise.

27

The *Bayes rule* is the decision rule that minimizes the expected cost. In the two-class situation, the Bayes rule can be determined analytically. If you classify a case into class 1, then the cost is

$$(1 - p) \text{cost}_{\text{FP}}$$

where p is the true posterior probability that a case belongs to class 1. If you classify a case into class 0, then the cost is

$$p \cdot \text{cost}_{\text{FN}}$$

Therefore, the optimal rule allocates a case to class 1 if

$$(1 - p) \text{cost}_{\text{FP}} < p \cdot \text{cost}_{\text{FN}}$$

otherwise allocate the case to class 0. Solving for p gives the optimal cutoff probability. Because p must be estimated from the data, the *plug-in Bayes rule* is used in practice

$$\hat{p} > \frac{1}{1 + \left(\frac{\text{cost}_{\text{FN}}}{\text{cost}_{\text{FP}}} \right)}$$

Consequently, the plug-in Bayes rule may not achieve the minimum cost if the estimate of the posterior probability is poorly estimated.

Note that the Bayes rule only depends on the ratio of the costs, not on their actual values.

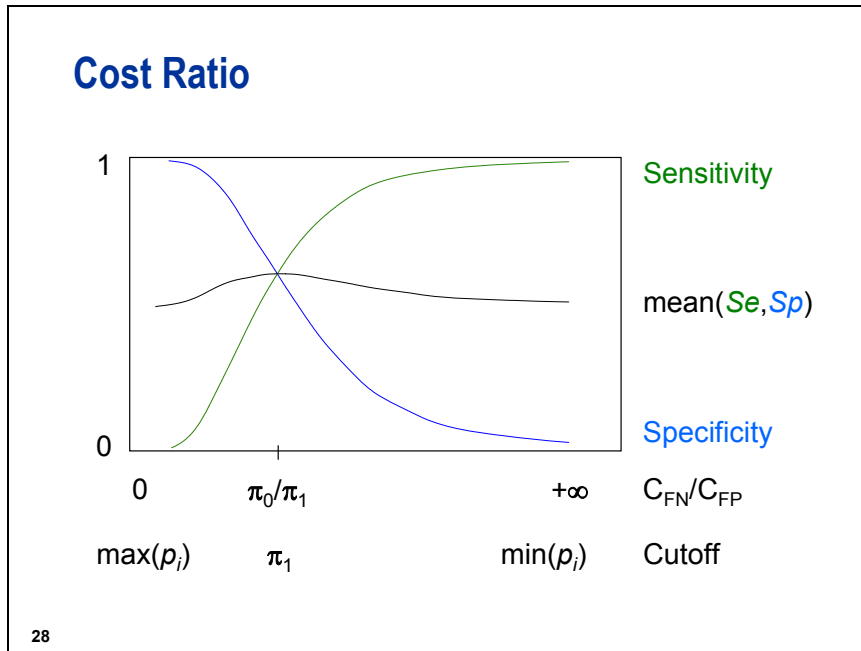
If the misclassification costs are equal, then the Bayes rule corresponds to a cutoff of 0.5. The expected cost (risk) equals

$$\text{cost}_{\text{FP}} \pi_0 (1 - Sp) + \text{cost}_{\text{FN}} \pi_1 (1 - Se)$$

When the cost ratio equals one, the expected cost is proportional to the error rate. A .5 cutoff tends to minimize error rate (maximize accuracy). Hand (1997) commented that

The use of error rate often suggests insufficiently careful thought about the real objectives....

When the target event is rare, the cost of a false negative is usually greater than the cost of a false positive. The cost of not soliciting a responder is greater than the cost of sending a promotion to someone who does not respond. The cost of accepting an applicant who will default is greater than the cost of rejecting someone who would pay off the loan. The cost of approving a fraudulent transaction is greater than the cost of denying a legitimate one. Such considerations dictate cutoffs that are less (often much less) than .5.



In many situations, it is difficult to precisely quantify the cost ratio, $\text{cost}_{FN}/\text{cost}_{FP}$. Examining the performance of a classifier over a range of cost ratios can be useful. A pertinent range of cost ratios is usually centered on the ratio of priors π_0/π_1 . This corresponds to a cutoff of π_1

$$p > \frac{1}{1 + \left(\frac{\pi_0}{\pi_1} \right)} = \pi_1$$

For instance, if the nonevent cases were nine times more prevalent than the event cases, then a false negative would be nine times more costly than a false positive and the cutoff would be 0.1. When the cost ratio equals π_0/π_1 , the expected cost is equivalent to the negative sum of sensitivity and specificity. The central cutoff, π_1 , tends to maximize the mean of sensitivity and specificity. Because increasing sensitivity usually corresponds to decreasing specificity, the central cutoff tends to equalize sensitivity and specificity.

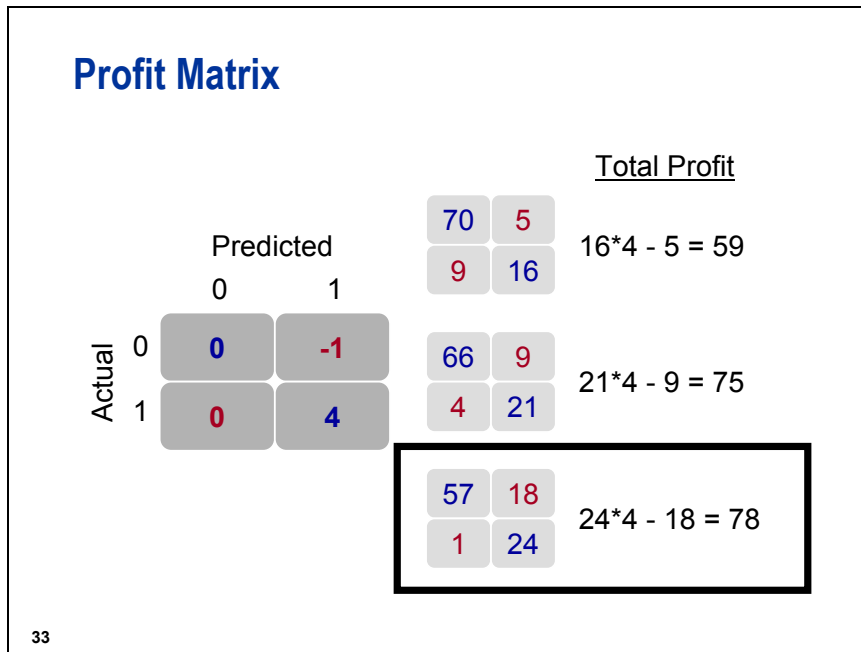
If separate samples were taken with equal allocation (50% events and 50% nonevents), then using the unadjusted cutoff of 0.5 on the biased sample is equivalent to using the central cutoff, π_1 , on the population.

4.04 Multiple Choice Poll

If the cost of false negatives is 9 times higher than the cost of false positives, then according to Bayes rule what is the cutoff that minimizes the expected cost?

- a. 0.90
- b. 0.09
- c. $1/9$
- d. 0.10

4.4 Overall Predictive Power



Defining a *profit matrix* (instead of a cost matrix, as in the last section) will not lead to a different classification rule. It does point to a useful statistic for measuring classifier performance. The model yields posterior probabilities, and those probabilities (in conjunction with a profit or cost matrix) classify individuals into likely positives and likely negatives. On the validation data, the behavior of these individuals is known; hence, it is feasible to calculate each individual's expected profit, and hence it is also feasible to calculate a total profit. This total profit can be used as a model selection and assessment criterion.



Using Profit to Assess Fit

Gathering profit information can be difficult. If profit information can be used, however, it permits a more familiar scale for comparing models. Consumers of models may not have a feel for what kind of lift they expect, or what constitutes a good value for sensitivity. Using total or average profit as an assessment statistic may skirt those issues.

Consider the following profit matrix:

		Predicted Class	
		0	1
Actual Class	0	\$0	-\$1
	1	\$0	\$99

This profit matrix is consistent with a marketing effort that costs \$1, and, when successful, garners revenue of \$100. Hence, the profit for soliciting a non-responder is -\$1, and the profit for soliciting a responder is \$100-\$1 = \$99. Given that each individual has a posterior probability p_i , you can resort either to the Bayes rule or simple algebra to find the optimum cutoff.

A typical decision rule would be: Solicit if the expected profit for soliciting, given the posterior probability, is higher than the expected profit for ignoring the customer.

Solicit if:

$$\begin{aligned}
 E(\text{Profit} \mid p_i, \text{solicit}) &> E(\text{Profit} \mid p_i, \text{do not solicit}) \\
 p_i * 99 + (1 - p_i) * (-1) &> p_i * 0 + (1 - p_i) * (0) \\
 99 * p_i - 1 + p_i &> 0 \\
 100 * p_i - 1 &> 0 \\
 p_i &> 0.01.
 \end{aligned}$$

This cutoff of 0.01 can be used to calculate the expected profit of using this rule with the current model. In order to calculate total and average profit comparable to what would be achieved in the population, weights must be calculated. The decision variable is created as a flag indicating whether the predicted probability is greater than the cutoff, 0.01. Using the information about decision and response, the profit per individual is calculated. These profits are summed and averaged by the MEANS procedure.

```

data scoval;
  set scoval;
  sampwt = (&pi1/&rho1)*(INS)
          + ((1-&pi1)/(1-&rho1))*(1-INS);
  decision = (p_1 > 0.01);
  profit = decision*INS*99
          - decision*(1-INS)*1;
run;

```

```
proc means data=scoval sum mean;
  weight sampwt;
  var profit;
run;
```

The MEANS Procedure

Analysis Variable : profit

Sum	Mean
13269.60	1.2341397

Using this model to score a population of, say, 1,000,000 individuals and soliciting only those with p_i greater than 0.01 would yield a total expected profit of \$1,234,139.70. With the above profit matrix and $\pi_1=0.02$, the “solicit everyone” rule generates a profit of $1,000,000*0.02*99-1,000,000*.98*1=\$1,000,000$. Using the model and some elementary decision theory leads to better decisions and more profit. Other models can be compared to this current model with this statistic.

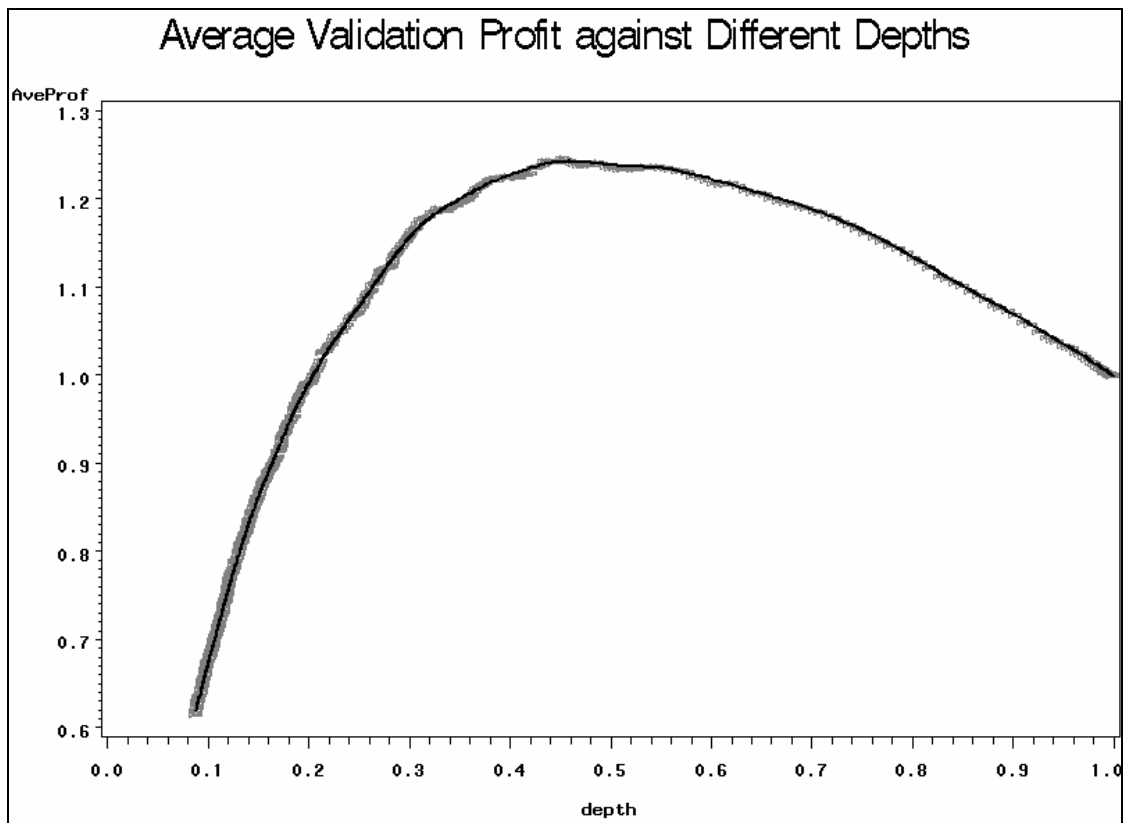
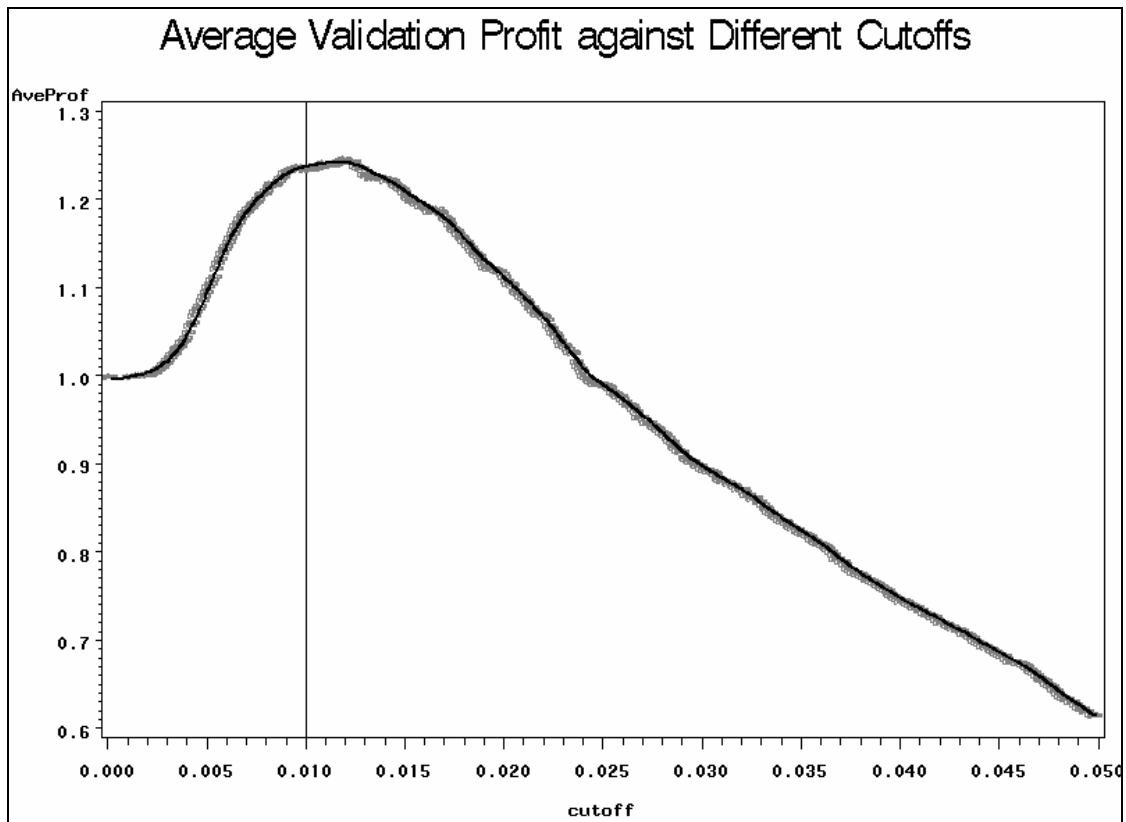
To see how other cutoffs fare, you can use the information in the **ROC** data set to draw a plot of how average profit changes as a function of the cutoff (or the solicited depth).

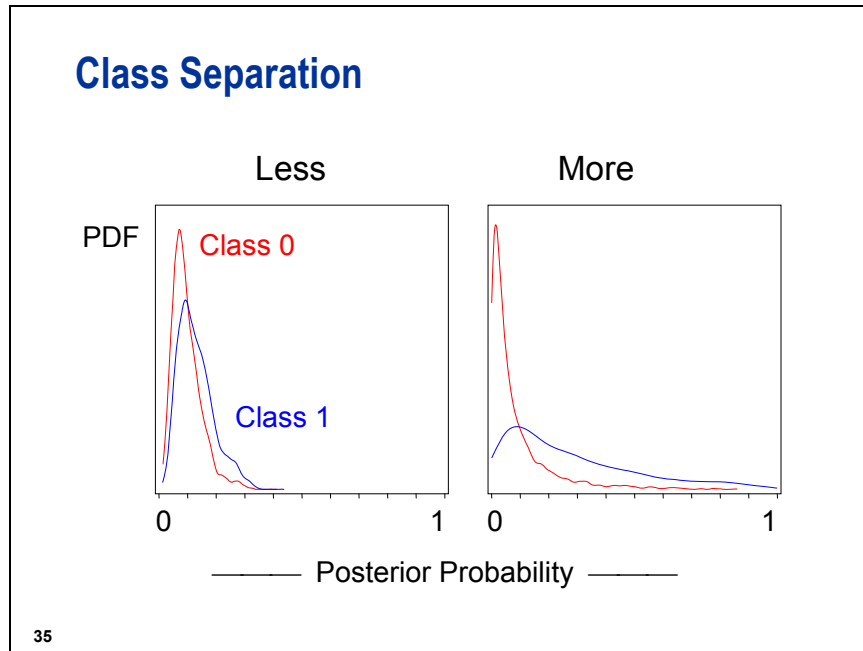
Using the true positive and false positive rates calculated earlier, you can calculate the average profit for each of the cutoffs considered in the **ROC** data set. A false positive (**fp**) is an individual who is solicited but does not respond. Hence, the cost of soliciting the false positives, on average, is the \$1 solicitation cost times the false positive rate. Likewise, the profit associated with individuals who are solicited and respond is \$99 times the true positive rate, **tp**. The difference of these two terms is the average profit.

Using the SYMBOL statements, specify two rules for plotting the points: the first rule plots short (H=.5), light gray (C=GRAY) plus signs (V=PLUS) for the points; the second line joins the points with a thick (W=2), smoothed spline (I=SM##) that is colored black (C=BLACK). The plot is restricted to the region around the cutoff of 0.01.

```
data roc;
  set roc;
  AveProf = 99*tp - 1*fp;
run;

symbol1 i=none v=plus c=gray h=.5;
symbol2 i=sm25 v=none c=black w=2 l=1;
proc gplot data=roc;
  where cutoff < .05;
  title "Average Validation Profit against Different Cutoffs";
  plot aveProf*cutoff aveProf*cutoff/overlay href=.01;
run;
  title "Average Validation Profit against Different Depths";
  plot aveProf*depth aveProf*depth/overlay;
run; quit;
title;
```

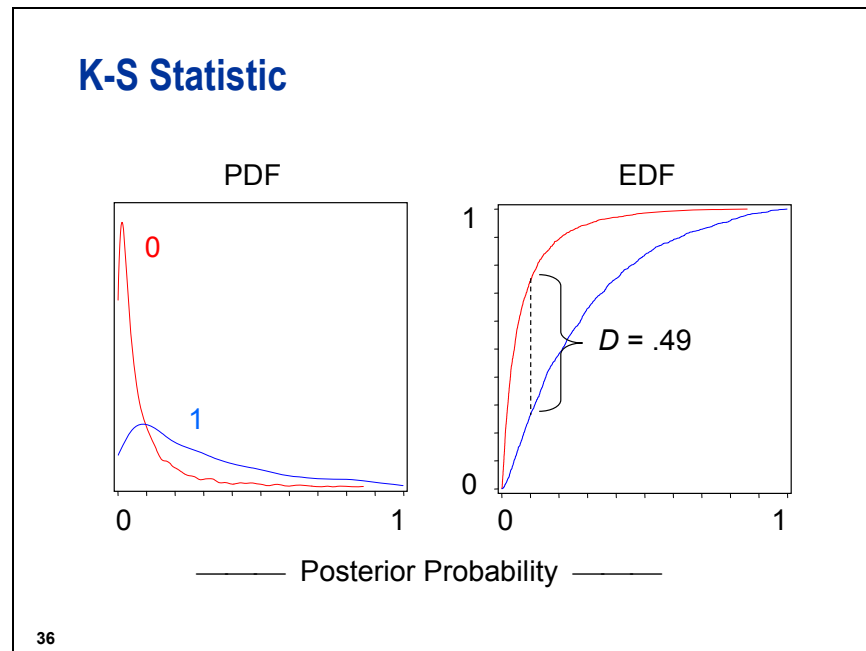




Statistics such as sensitivity, positive predictive value, and risk depend on the choice of cutoff value. Statistics that summarize the performance of a classifier across a range of cutoffs can also be useful for assessing global discriminatory power. One approach is to measure the separation between the predicted posterior probabilities for each class. The more that the distributions overlap, the weaker the model.

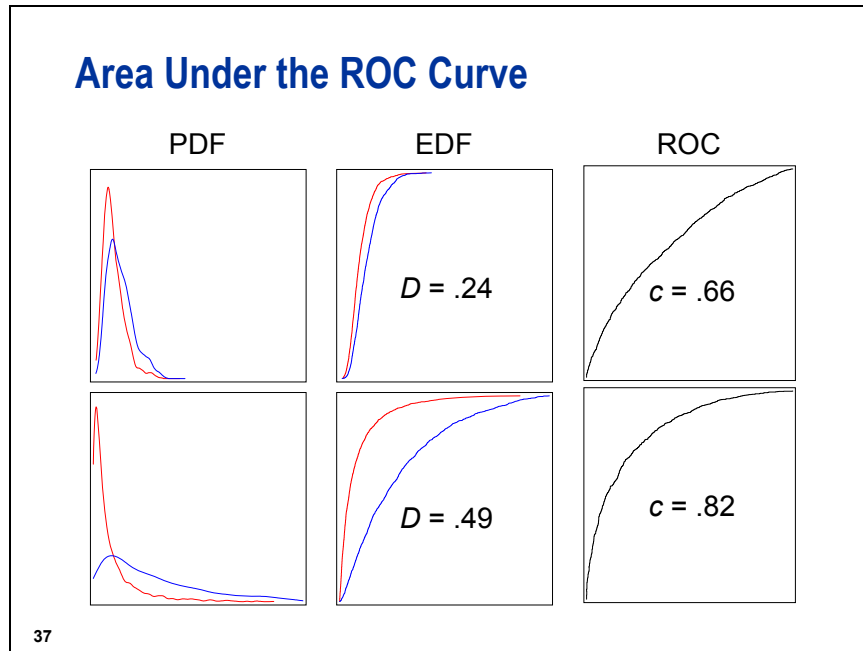
The simplest statistics are based on the difference between the means of the two distributions. In credit scoring, the *divergence* statistic is a scaled difference between the means (Nelson 1997). Hand (1997) discusses several summary measures based on the difference between the means.

The well-known *t*-test for comparing two distributions is based on the difference between the means. The *t*-test has many optimal properties when the two distributions are symmetric with equal variance (and have light tails). However, the distributions of the predicted posterior probabilities are typically asymmetric with unequal variance. Many other two-sample tests have been devised for nonnormal distributions (Conover 1980).



The Kolmogorov-Smirnov two-sample test is based on the distance between the empirical distribution functions (Conover 1980). The test statistic, D , is the maximum vertical difference between the cumulative distributions. If D equals zero, the distributions are everywhere identical. If $D > 0$, then there is some posterior probability where the distributions differ. The maximum value of the K-S statistic, 1, occurs when the distributions are perfectly separated. Use of the K-S statistic for comparing predictive models is popular in database marketing.

An oversampled validation data set does not affect D because the empirical distribution function is unchanged if each case represents more than one case in the population.



The Kolmogorov-Smirnov two-sample test is sensitive to all types of differences between the distributions – location, scale, and shape. In the predictive modeling context, it could be argued that location differences are paramount. Because of its generality, the K-S test is not particularly powerful at detecting location differences. The most powerful nonparametric two-sample test is the Wilcoxon-Mann-Whitney test. Remarkably, the Wilcoxon-Mann-Whitney test statistic is also equivalent to the area under the ROC curve (Hand 1997).

The Wilcoxon version of this popular two-sample test is based on the ranks of the data. In the predictive modeling context, the predicted posterior probabilities would be ranked from smallest to largest. The test statistic is based on the sum of the ranks in the classes. The area under the ROC curve, c , can be determined from the rank-sum in class 1.

$$c = \frac{\sum_{\{i|y=1\}}^{n_1} R_i - \frac{1}{2} n_1 (n_1 + 1)}{n_1 \cdot n_o}$$

The first term in the numerator is the sum of the ranks in class 1.

A perfect ROC curve would be a horizontal line at one – that is, sensitivity and specificity would both equal one for all cutoffs. In this case, the c statistic would equal one. The c statistic technically ranges from zero to one, but in practice, it should not get much lower than one-half. A perfectly random model, where the posterior probabilities were assigned arbitrarily, would give a 45° angle straight ROC curve that intersects the origin; hence, it would give a c statistic of 0.5.

Oversampling does not affect the area under the ROC curve because sensitivity and specificity are unaffected. The area under the ROC curve is also equivalent to the Gini coefficient, which is used to summarize the performance of a Lorentz curve (Hand 1997).



Calculating the K-S and c Statistics

The K-S statistic can be computed in the NPAR1WAY procedure using the scored validation data set. The EDF and WILCOXON options in PROC NPAR1WAY request the Kolmogorov-Smirnov and Wilcoxon tests, respectively. The tests compare the values of the variable listed in the VAR statement between the groups listed in the CLASS statement.

```
proc npar1way edf wilcoxon data=scoval;
  class ins;
  var p_1;
run;
```

Partial Output

Kolmogorov-Smirnov Test for Variable P_1 Classified by Variable Ins			
INS	N	EDF at Maximum	Deviation from Mean at Maximum
0	7028	0.698207	12.827001
1	3724	0.256445	-17.621228
Total	****	0.545201	
Maximum Deviation Occurred at Observation 6855 Value of P_1 at Maximum = 0.017303			
Kolmogorov-Smirnov Two-Sample Test (Asymptotic)			
KS	0.210194	D	0.441762
KSa	21.795404	Pr > KSa	<.0001

The results show that the two-sample Kolmogorov statistic D is 0.44, which represents the largest separation between the two cumulative distributions. The results of the Wilcoxon test can be used to compute the c statistic. However, this is automatically computed in the LOGISTIC procedure.

```
proc logistic data=scoval des;
  model ins=p_1;
run;
```

Partial Output

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	77.6	Somers' D	0.560
Percent Discordant	21.5	Gamma	0.565
Percent Tied	0.9	Tau-a	0.254
Pairs	26172272	c	0.780

Comparing this model to a different model, say the **DDA**, **DDABal**, **Dep**, **DepAmt**, **CashBk**, **Checks**, and **Res** model posited in Chapter 2, you would find that this model performs much better on the validation data and hence would be the model to use.


```

proc logistic data=train1 des;
  class res;
  model ins=dda ddabal dep depamt checks res;
  score data=valid1 out=cand
    priorevent=&pil;
run;
proc logistic data=cand des;
  model ins=p_1;
run;

```

The validation results show that the c statistic is much lower for this model. Of course, because these inputs were selected arbitrarily, it would be highly unlikely that this model would outperform the previous model.

Partial Output

Association of Predicted Probabilities and Observed Responses				
Percent Concordant	63.2	Somers' D	0.344	
Percent Discordant	28.8	Gamma	0.373	
Percent Tied	7.9	Tau-a	0.156	
Pairs	26172272	c	0.672	

This process of selecting inputs for a model, fitting that model, and evaluating that model's fit on the validation data can be automated with macro programming. This will allow you to consider many candidate models in a small time frame; this should lead to better model generalization.

The next chapter features an example of a macro program that takes the series of models generated by best subsets logistic regression and compares them on the validation data performance. A plot shows the performance gains as a function of model complexity, which should be a very useful tool for final model selection.

4.05 Multiple Choice Poll

Which of the following statements is false regarding the K-S statistic?

- The test statistic, D, is the maximum vertical distance between the cumulative distributions.
- An oversampled validation data set does not affect the test statistic D.
- The K-S statistic is statistically equivalent to the area under the ROC curve.
- The K-S statistic is not as powerful at detecting location differences as the Wilcoxon-Mann-Whitney test.

4.5 Chapter Summary

One of the most important steps in predictive modeling is to assess the performance of the model. To correct for the optimistic bias, a common strategy is to holdout a portion of the development data for assessment. The LOGISTIC procedure can then be used to score the data set used for assessment. Statistics that measure the predictive accuracy of the model include sensitivity and positive predicted value. Graphics such as the ROC curve, the gains chart, and the lift chart can also be used to assess the performance of the model.

If the assessment data set was obtained by splitting oversampled data, then the assessment data set needs to be adjusted. This can be accomplished by using the sensitivity, specificity, and the prior probabilities.

In predictive modeling, the ultimate use of logistic regression is to allocate cases to classes. To determine the optimal cutoff probability, the plug-in Bayes rule can be used. The information you need is the ratio of the costs of false negatives to the cost of false positives. This optimal cutoff will minimize the total expected cost (or maximize the total expected profit).

The profit itself can be used as an assessment statistic, presuming that some reasonable estimate of the appropriate financial figures can be found.

When the target event is rare, the cost of a false negative is usually greater than the cost of a false positive. For example, the cost of not soliciting a responder is greater than the cost of sending a promotion to someone who does not respond. Such considerations dictate cutoffs that are usually much less than .50, the cutoff that maximizes accuracy.

A popular statistic that summarizes the performance of a model across a range of cutoffs is the Kolmogorov-Smirnov statistic. However, this statistic is not as powerful in detecting location differences as the Wilcoxon-Mann-Whitney test. Furthermore, the Wilcoxon-Mann-Whitney test statistic is equivalent to the area under the ROC curve (the c statistic). Thus, the c statistic should be used to assess the performance of a model across a range of cutoffs.

General form of PROC NPAR1WAY:

```
PROC NPAR1WAY DATA=SAS-data-set <options>;  
    CLASS variable;  
    VAR variable;  
RUN;
```

4.6 Solutions

Solutions to Exercises

Solutions to Student Activities (Polls/Quizzes)

4.02 Multiple Choice Poll – Correct Answer

Which variables seem to have a nonlinear relationship with the target?

- ☒ a. RECENT_AVG_CARD_GIFT_AMT and LIFETIME_GIFT_RANGE
- b. DONOR_AGE and PER_CAPITA_INCOME
- c. RECENT_RESPONSE_COUNT and RECENT_STAR_STATUS
- d. INCOME_GROUP and PCT_MALE_MILITARY

12

4.03 Multiple Choice Poll – Correct Answer

The formula for sensitivity is

- ☒ a. true positives / total actual positives
- b. true positives / total predicted positives
- c. true negatives / total actual negatives
- d. true negatives / total predicted negatives

22

4.04 Multiple Choice Poll – Correct Answer

If the cost of false negatives is 9 times higher than the cost of false positives, then according to Bayes rule what is the cutoff that minimizes the expected cost?

- a. 0.90
- b. 0.09
- c. 1/9
- ☒ d. 0.10

31

4.05 Multiple Choice Poll – Correct Answer

Which of the following statements is false regarding the K-S statistic?

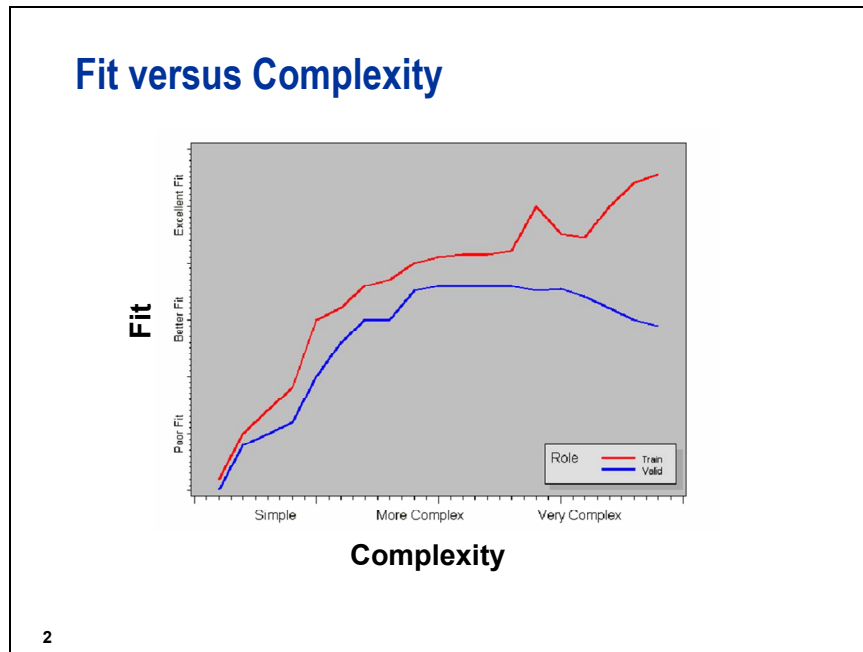
- a. The test statistic, D , is the maximum vertical distance between the cumulative distributions.
- b. An oversampled validation data set does not affect the test statistic D .
- ☒ c. The K-S statistic is statistically equivalent to the area under the ROC curve.
- d. The K-S statistic is not as powerful at detecting location differences as the Wilcoxon-Mann-Whitney test.

40

Chapter 5 Generating and Evaluating Many Models

5.1	Model Selection Plots.....	5-3
5.2	Chapter Summary.....	5-22
5.3	Solutions	5-23
	Solutions to Exercises	5-23
	Solutions to Student Activities (Polls/Quizzes)	5-23

5.1 Model Selection Plots



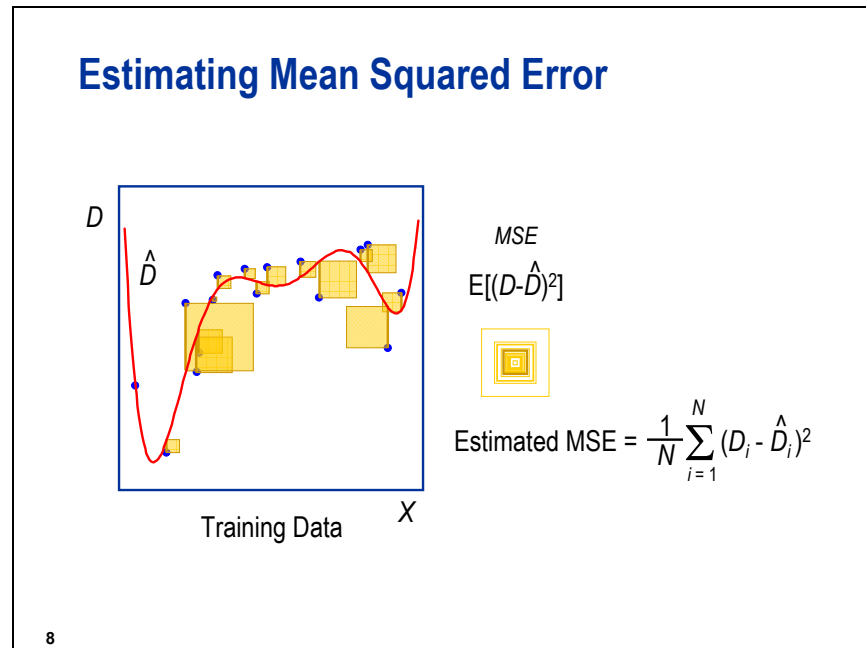
To compare many models, an appropriate fit statistic (or statistics) must be selected. For statistics like average profit, c , and Kolmogorov-Smirnov's D , higher values mean better fitting models. Because the goal of most predictive modeling efforts is a model that generalizes well, these statistics are typically measured on the validation data set. For a series of models, which could be generated by an automatic selection routine, it is conceivable to plot a fit measure against some index of complexity. For standard logistic regression models, this index is likely equivalent to the degrees of freedom in the model.

Typically, model performance follows a fairly straightforward trend. As the complexity increases (that is, as terms are added) the fit on the training data gets better. After a point, the fit may plateau, but on the training data, the fit gets better as model complexity increases. Some of this increase is attributable to the model capturing relevant trends in the data. Detecting these trends is the goal of modeling. Some of the increase, however, is due to the model identifying vagaries of the training data set. This behavior has been called overfitting. Because these vagaries are not likely to be repeated, in the validation data or in future observations, it is reasonable to want to eliminate those models. Hence, the model fit on the validation data, for models of varying complexity, is also plotted. The typical behavior of the validation fit line is an increase (as more complex models detect more usable patterns) followed by a plateau, which may finally result in a decline¹ in performance. The decline in performance is due to overfitting. The plateau just indicates more complicated models that have no fit-based arguments for their use. A reasonable rule would be to select the model associated with the complexity that has the highest validation fit statistic.

¹ This behavior is sometimes called *peaking*.

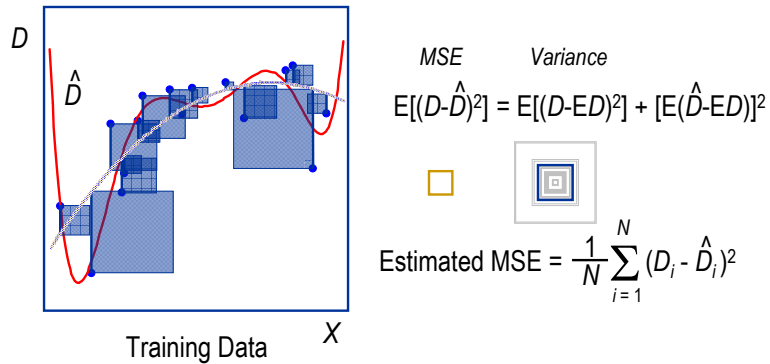
Plotting the training and validation results together permits a further assessment of the model's generalizing power. Typically, the performance will deteriorate from the training data to the validation data. This phenomenon, sometimes known as *shrinkage*, is an additional statistic that some modelers use to get a measure of the generalizing power of the selected model. For example, the rule used to select a model from the many different models plotted might be: "Choose the simplest model that has the highest validation fit measure, with no more than 10% shrinkage from the training to the validation results."

If the measure of model fit is some sort of error rate, then the plot looks like the above but flipped about the horizontal axis. In the absence of profit or cost information, the Mean Squared Error (MSE) is one such fitness statistic that measures how poorly a model fits; that is, smaller is better.



Mean squared error (MSE) is a commonly employed method of establishing and assessing the relationship between inputs and the expected value of the target. MSE is estimated from a sample by differencing model predictions from observed target values, squaring these differences, and averaging across all data points in the sample.

MSE Decomposition: Variance

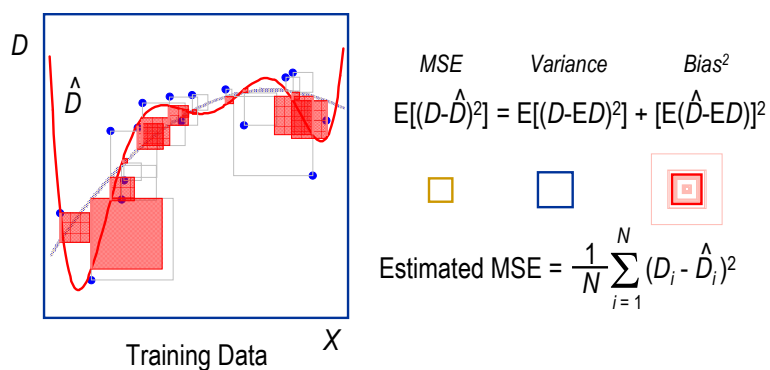


12

In theory, the MSE can be decomposed into two components, each involving a deviance from the true (but unknown) expected value of the target variable.

The first of these components is the residual variance of the target variable. This term quantifies the theoretical limit of prediction accuracy and the absolute lower bound for the MSE. The variance component is independent of any fitted model.

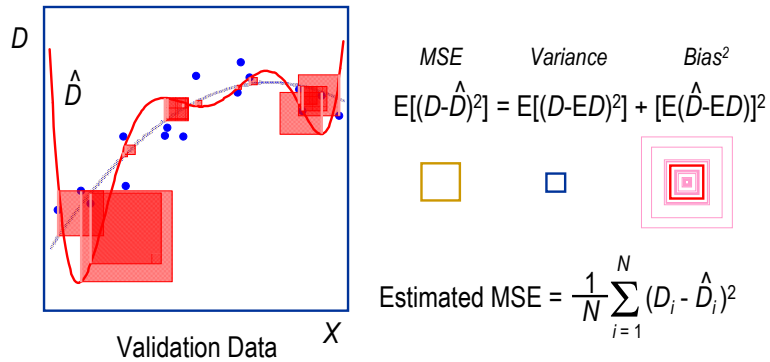
MSE Decomposition: Squared Bias



14

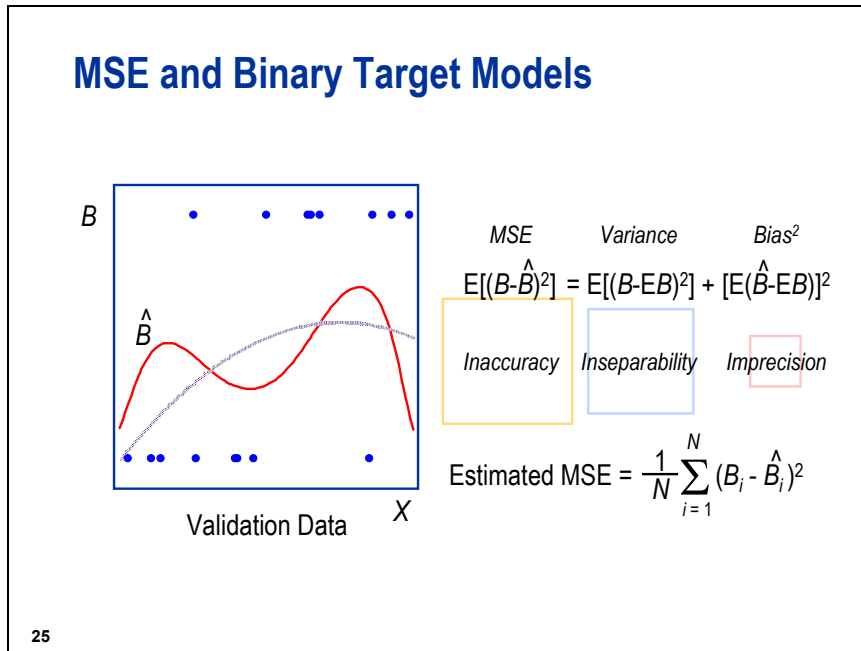
The second MSE component is the average prediction bias squared. This term quantifies the difference between the predicted and actual expected value of the target.

Honest MSE Estimation



21

As always, you must be careful to obtain an unbiased estimate of MSE. MSE estimates obtained from the data used to fit the model will almost certainly be overly optimistic. Estimates of MSE from an independent validation data set allow for an honest assessment of model performance.



While MSE is an obvious choice for comparing interval target models, it is also useful for assessing binary target models (Hand 1997). The estimated MSE can be thought of as measuring the overall *inaccuracy* of model prediction. This inaccuracy estimate can be decomposed into a term related to the *inseparability* of the two-target level (corresponding to the variance component) plus a term related to the *imprecision* of the model estimate (corresponding to the bias-squared component). In this way, the model with the smallest estimated MSE will also be the least imprecise.

Whether you choose the MSE, the c statistic, the average profit, Kolmogorov-Smirnov's D or any other measure of model fit, the model selection plot should help point you toward the appropriate level of complexity to achieve good validation generalization.

Likewise, there are many ways to consider generating a series of models of different complexity. Consider any of the automatic variable selection routines described in Chapter 3. Backward elimination of variables creates a series of models of decreasing complexity. Stepwise selection, in general, generates a series of models of increasing complexity. All subsets selection techniques likewise create a series of increasingly complex models. Any of these techniques lends itself to model selection based on validation data performance; the next demonstration shows an example of using all subsets regression.

5.01 Multiple Choice Poll

Which of the following statements is false regarding model assessment statistics?

- a. The shrinkage statistic is a measurement of the difference between the training model results and the validation model results.
- b. The model with the smallest estimated mean squared error will also be the least imprecise.
- c. The variance component of the mean squared error is independent of any fitted model.
- d. The bias component of the mean squared error is independent of any fitted model.



All Subsets Regression

This demonstration uses the validation performance of many different models as a guide to selecting models that will generalize well. Hence, the validation data must be prepared in the same way as the training data. To this end, the data sets **train2** and **valid2**, which were created by imputing for all numeric variables on the **train** and **valid** data sets, are augmented with the modified **DDABal** and **SavBal** variables. Moreover, because the all subsets algorithm in the LOGISTIC procedure cannot be used with the CLASS statement, the dummy variables for **Res** are created on these data sets as well.

```
data train2;
  set train2;
  resr=(res="R");
  resu=(res="U");
  if not dda then ddabal = &mean;
  brclus1=(branch='B14');
  brclus2=(branch in ('B12','B5','B8',
                     'B3','B18','B19','B17',
                     'B4','B6','B10','B9',
                     'B1','B13'));
  brclus3=(branch in ('B15','B16'));
  %include rank;
  if savbal > 16000 then savbal=16000;
run;

data valid2;
  set valid2;
  resr=(res="R");
  resu=(res="U");
  if not dda then ddabal = &mean;
  brclus1=(branch='B14');
  brclus2=(branch in ('B12','B5','B8',
                     'B3','B18','B19','B17',
                     'B4','B6','B10','B9',
                     'B1','B13'));
  brclus3=(branch in ('B15','B16'));
  %include rank;
  if savbal > 16000 then savbal=16000;
run;
```

The LOGISTIC procedure with SELECTION equal to SCORE is called. The number of models per complexity is specified by the BEST= option. The ODS output data set **score** contains the inputs used in at each complexity level. Setting BEST=2 will generate twice as many candidate models as BEST=1; this may prove useful in finding a model that fits the validation data, and hence the population, well.

```
ods listing close;
ods output bestsubsets=score;

proc logistic data=train2 des;
  model ins=&screened
  / selection=SCORE best=2;
run;

ods listing;
proc print data=score(obs=12);
run;
```

Obs	control_ var	NumberOf Variables	ScoreChiSq	VariablesInModel
1		1	2085.5103	SavBal
2		1	1870.2276	B_DDABal
3	1	2	3316.1397	SavBal B_DDABal
4		2	2794.5356	CD SavBal
5	1	3	3856.4605	CD SavBal B_DDABal
6		3	3691.8664	MM SavBal B_DDABal
7	1	4	4153.3824	MM CD SavBal B_DDABal
8		4	3977.5128	MIPhone CD SavBal B_DDABal
9	1	5	4278.3879	MIPhone MM CD SavBal B_DDABal
10		5	4272.2890	MM CD brclus2 SavBal B_DDABal
11	1	6	4352.5207	MIPhone MM CD SavBal B_DDABal DirDep
12		6	4347.5405	MM CD brclus2 SavBal B_DDABal DirDep



Recall that the **screened** macro variable contains the list of inputs created by making missing indicators (**MIPhone**, and so forth), clustering categorical input levels (**brclus1**, and so forth), and then clustering variables and choosing representatives. The inputs deemed to have low correlation with the target according to the Spearman and Hoeffding measures have been excluded. All in all, this leaves 36 inputs.

The inputs placed in **score** by the LOGISTIC procedure are transferred to macro variables using an SQL SELECT INTO function. The automatic macro variable **SQLOBS** gives the number of models produced overall.

```
proc sql noprint;
  select variablesinmodel into :inputs1 - :inputs99999
  from score;
  select NumberOfVariables into :ic1 - :ic99999
  from score;
quit;

%let lastindx = &SQLOBS;
```

The variables π_1 , ρ_1 , π_0 , and ρ_0 are necessary to calculate sampling weights. π_1 must be known a priori, and ρ_1 was automatically calculated in an earlier demonstration. To simplify the code, create macro variables to house π_0 and ρ_0 as well. The %SYSEVALF function takes the text string in its argument and performs the calculation. For example, %SYSEVALF(1-&pi1) will resolve to 0.98 in this instance.

```
%let rho0 = %sysevalf(1-&rho1);
%let pi0 = %sysevalf(1-&pi1);
```

Likewise, the profit matrix is placed into macro variables.

```
%let pf11 = 99; %let pf10 = 0;
%let pf01 = -1; %let pf00 = 0;
```

After these preliminaries have been established, it is easy to imagine the data set necessary to generate the model selection plot. It requires a summary of the performance for each model on the training and on the validation data sets. The particular data wrangling that needs to take place depends on what statistics are of interest and how you choose to calculate them. The following code calculates c, ASE, and average overall profit. ASE, the average squared error, is related to MSE but has a different divisor.

Two macros will be compiled. One, %ASSESS, will assess the performance of a model on a particular data set, and append a record summarizing that model's performance to the **Results** data set. The %FITANDSCORE macro will fit and score many different models; here, the series of models created by the all subsets algorithm.



These macros are predicated on assumptions of modeling data structure and the macro variables created by the SQL SELECT INTO call above. Applying these techniques, or similar techniques, on your data may require careful consideration of the data.

The %ASSESS macro is defined. This macro simplifies the assessment of each of the models found by the best subsets procedure. Data for assessment are assumed stored in a data set named SCORE&DATA, where DATA is a macro parameter with anticipated values TRAIN, VALID, and potentially TEST. The INPUTCOUNT= macro parameter points to the number of parameters in the model in question. The INPUTSINMODEL= macro parameter points to the list of parameters in the model. The INDEX= macro parameter is a unique index for each model in the series. This allows you to have more than one model with, say, 16 inputs.

```
%macro assess(data=,inputcount=,inputsinmodel=,index=);
```

The assessment data is sorted by descending posterior probability. This sorting allows the calculation of a c statistic by the **ASSESS** DATA step below.

```
proc sort data=scored&data;
  by descending p_1;
run;
```

The main assessment DATA step begins. The **DATAROLE** variable is defined to contain the type of assessment (for example, VALID). A two-by-two temporary array, **n**, is defined and initialized to all zeros. The **n** array will contain the confusion matrix based on the profit matrix specified by decision processing.

A two-by-one temporary array, **w**, is defined and initialized to the ratio of the population and sample marginal averages (π_0/ρ_0) and (π_1/ρ_1). The **w** array will be used to adjust the model posterior probabilities as well as to calculate total profit and the confusion matrix.

```
data assess;
  attrib DATAROLE length=$5;
  retain sse 0 csum 0 DATAROLE "&data";

  array n[0:1,0:1] _temporary_ (0 0 0 0);
  array w[0:1] _temporary_
    (%sysevalf(&pi0/&rho0) %sysevalf(&pi1/&rho1));
  keep DATAROLE INPUT_COUNT INDEX
    TOTAL_PROFIT OVERALL_AVG_PROFIT ASE C;
```

An observation is read from the assessment data.

```
set scored&data end=last;
```



Assuming that the LOGISTIC call that generates the scored data sets will correct for oversampling, you can use the predicted probabilities. If the probabilities are biased due to oversampling, and you do not correct for this before the assessment, you could do that here.

Profits for each decision alternative are calculated, based on the user-specified decision data.

```
d1=&PF11*p_1+&PF01*p_0;
d0=&PF10*p_1+&PF00*p_0;
```

Variable **t** equals 1 if the target equals the primary target value and zero otherwise; that is, **t** is a flag for response. Variable **d** equals 1 if the profit for decision 1 exceeds the profit for decision 0; that is, **d** is the profit maximizing decision. The STRIP function removes trailing and leading blanks. Contrast this with the default behavior of the COMPRESS function, which removes all blanks in a field.



These variables may be defined differently in the presence of different decision rules or different values for the target classes; for example a different profit matrix or a Y/N target instead of a 1/0 target.

```
t=(strip(ins)="1");
d=(d1>d0);
```

The appropriate cell of the confusion matrix array **n** is incremented by the appropriate element of the weight vector. The sum of squared error, **sse**, is incremented. The sum used to calculate the c statistic, **csum**, is incremented. The increment will be positive only when the target value is zero. This is in effect a Riemann sum of the Receiver Operating Characteristic (ROC) curve.

```
n[t,d] + w[t];
sse + (ins-p_1)**2;
csum + ((n[1,1]+n[1,0])*(1-t)*w[0]);
```


On the last time through the DATA step, finalize the fit statistics and output. To finalize the ASE, divide the sum of squared errors by the sample size. Total profit is the product of the sample sizes in each cell of **n** with the appropriate profit amount. The c statistic is based on the sum of the area of rectangles that approximate the ROC curve, but those areas are based on the sample size. To restrict this area to be between 0 and 1, the range of the c statistic, divide through by the product of n_1 and n_0 .

```

if last then do;
  INPUT_COUNT=&inputcount;
  TOTAL_PROFIT =
    sum(&PF11*n[1,1], &PF10*n[1,0], &PF01*n[0,1], &PF00*n[0,0]);
  OVERALL_AVG_PROFIT =
    TOTAL_PROFIT/sum(n[0,0], n[1,0], n[0,1], n[1,1]);
  ASE = sse/sum(n[0,0], n[1,0], n[0,1], n[1,1]);
  C = csum/(sum(n[0,0], n[0,1])*sum(n[1,0], n[1,1]));
  index=&index;
  output;
end;
run;

```

Update the **Results** data set with this data's assessment results.

```

proc append base=results data=assess force;
run;

```

End the %ASSESS macro.

```
%mend assess;
```

Define another macro %FITANDSCORE. This macro will refit the series of logistic regression models, and create the scored data sets that the %ASSESS macro needs.

```
%macro fitandscore();
```

Any previous version of the **Results** data set is removed using the DATASETS procedure.

```

proc datasets
  library=work
  nodetails
  nolist;
  delete results;
run;

```

A loop begins over all models fit by the original LOGISTIC procedure. The macro variable **IM** is set equal to the inputs in the **MODEL_INDX** model. **IC** is set equal to the number of inputs in the **MODEL_INDX** model.

```

%do model_indx=1 %to &lastindx;

%let im=&&inputs&model_indx;
%let ic=&&ic&model_indx;

```



The macro variable **LASTINDX** was created earlier; it points to the total number of models in the series of models to be considered.

The LOGISTIC procedure refits the model with inputs specified in **IM**. The SCORE option is used to score the training and validation data sets and place the result in **ScoredTrain** and **ScoredValid**, respectively. Only the target value and posterior probabilities are kept. The PRIOREVENT= option corrects the predicted probabilities for oversampling. If you have many models, or if you are not interested in the output, you can use the Output Delivery System to eliminate the printed listing of the LOGISTIC procedure results.

```
proc logistic data=train2 des;
  model ins=&im;
  score data=train2
    out=scoredtrain(keep=ins p_1 p_0)
    priorevent=&pil;
  score data=valid2
    out=scoredvalid(keep=ins p_1 p_0)
    priorevent=&pil;
run;
```

The %ASSESS macro, discussed above, is called for each of the scored data sets. The values of the parameters in the call (DATA=, INPUTCOUNT=, and so forth) become macro variables in the %ASSESS macro.

```
%assess (data=TRAIN,
         inputcount=&ic,
         inputsinmodel=&im,
         index=&model_idx);
%assess (data=VALID,
         inputcount=&ic,
         inputsinmodel=&im,
         index=&model_idx);
```

The loop over all models fit by the LOGISTIC procedure with the SCORE option ends.

```
%end;
```

The %FITANDSCORE macro definition ends.

```
%mend fitandscore;
```

The %FITANDSCORE macro is called.

```
%fitandscore;
```

The result of the %FITANDSCORE macro is a results data set.

```
proc print data = results(obs=10);
run;
```

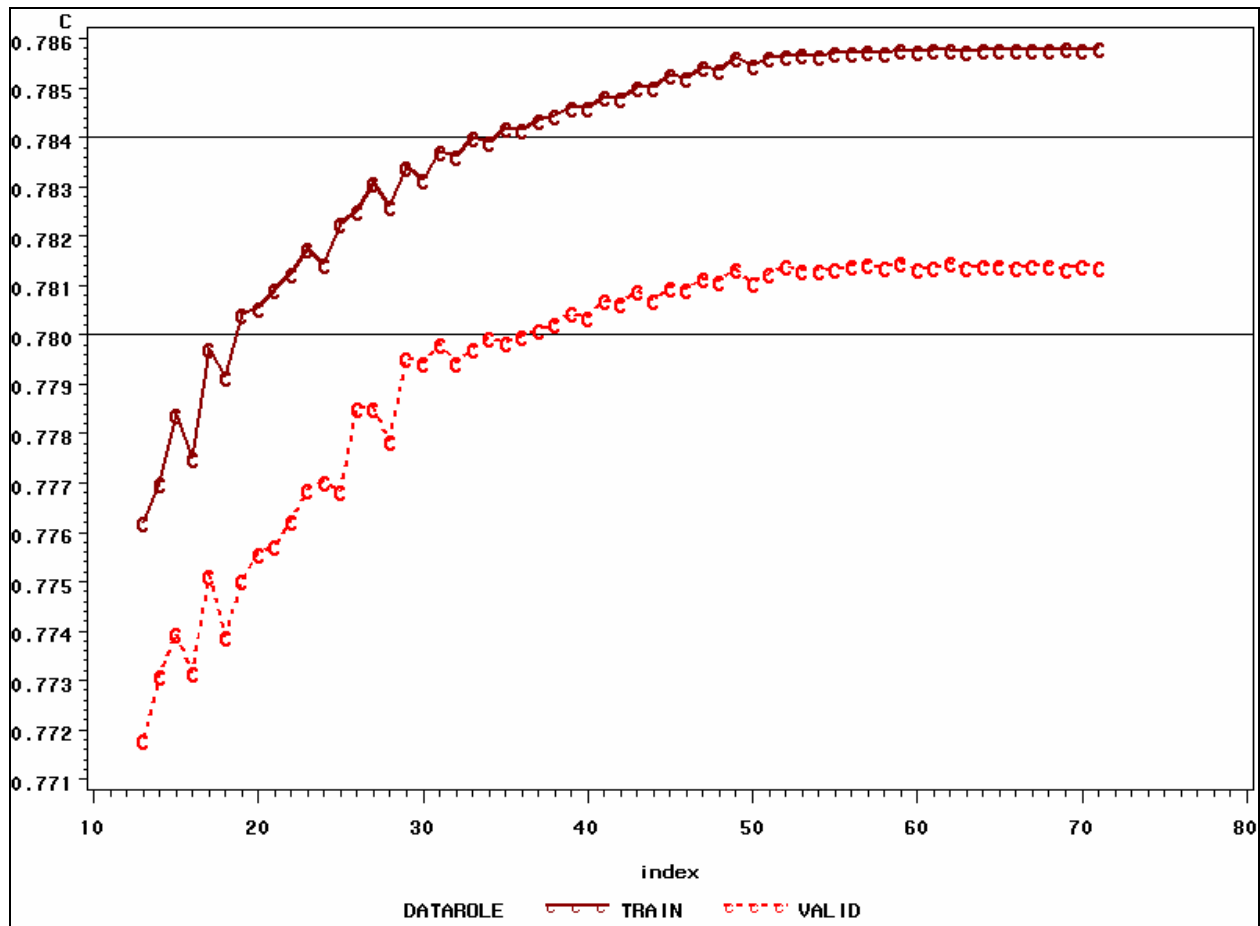
Obs	DATAROLE	INPUT_ COUNT	TOTAL_ PROFIT	OVERALL_ AVG_PROFIT	ASE	C	index
1	TRAIN	1	21512.59	1.00003	0.32030	0.50975	1
2	VALID	1	10751.47	0.99994	0.32014	0.50341	1
3	TRAIN	1	23137.64	1.07557	0.32754	0.65604	2
4	VALID	1	11472.91	1.06704	0.32735	0.65324	2
5	TRAIN	2	25396.83	1.18060	0.31592	0.73248	3
6	VALID	2	12459.36	1.15878	0.31562	0.72680	3
7	TRAIN	2	21512.59	1.00003	0.31645	0.58425	4
8	VALID	2	10751.47	0.99994	0.31651	0.57216	4
9	TRAIN	3	25765.88	1.19775	0.31246	0.75293	5
10	VALID	3	12747.68	1.18560	0.31240	0.74437	5

Visual inspection of these figures may make model selection easier.

The SYMBOL1 statement specifies a dark red (C=DARKRED) line with no breaks (L=1) connecting points indicated by the letter c (V=c) plotted 1.5 times higher than a point marker usually is (H=1.5). The F=ARIAL option specifies the font for printing the character. This symbol definition is associated with the first series of points plotted, which will be the c statistic for each model for the training data. The SYMBOL2 statement specifies a red line with breaks in it (L=2) for the validation data points. In the PLOT statement, the reference lines on the vertical axis are set to the training and validation c statistics of the model selected in Chapter 4.

```
symbol1 i=join f=arial v=c h=1.5 c=darkred l=1;
symbol2 i=join f=arial v=c h=1.5 c=red l=2;
proc gplot data = results;
  where index > 12;
  plot C*Index=datarole / vref=.784 .780;
run; quit;
```

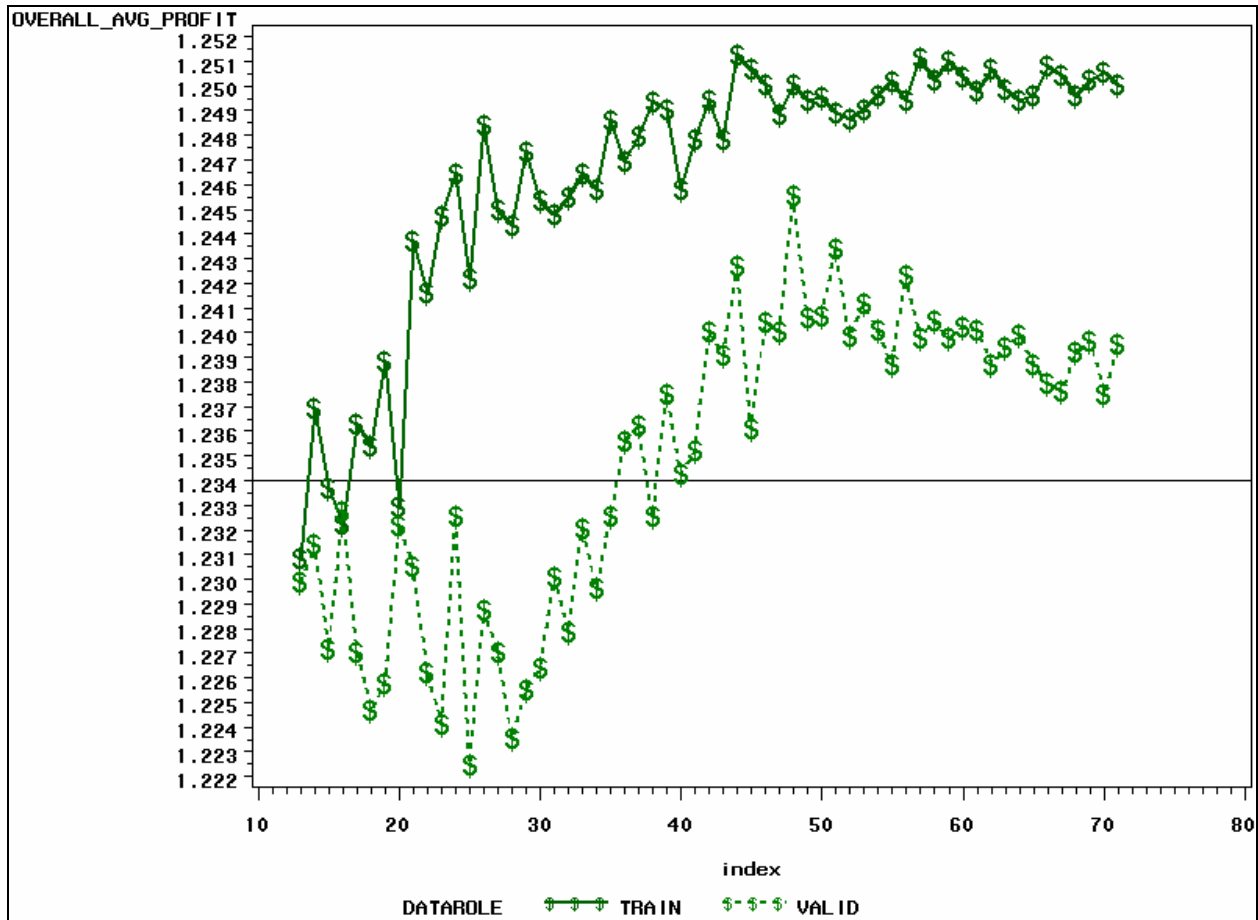
The plot seems to reach a plateau around index 45 or 50. Because the goal is to find a model that generalizes well, the simplest model that has good performance on the validation data is probably the best candidate.



Of course, it is difficult to quantify the trade-off between a slightly more complex model and a slightly higher c statistic. The trade-off may be easier to think about in terms of profit. What is it worth to the organization to use a more complex model? A plot of profit may help to visualize, and quantify, this trade-off.

This plot uses SYMBOL and GPLOT procedure syntax that is similar to earlier examples. The reference line is the average profit achieved using the model built in Chapter 4.

```
symbol1 i=join f=arial v=$ h=1.5 c=darkgreen w=2 l=1;
symbol2 i=join f=arial v=$ h=1.5 c=green w=2 l=2;
proc gplot data = results;
  where index > 12;
  plot OVERALL_AVG_PROFIT*Index=datarole / vref=1.234;
run; quit;
```



Again, the plot seems to show that the validation data performance peaks near index 50. Plotting the ASE will show similar results as well. Of course, you could use the MEANS procedure or the SQL procedure to find the model index associated with the highest validation profit, or c, or ASE. To see the results of, say, the highest profit model, submit the following code. Because the highest validation performance is achieved by the model with index 48, that model is a good candidate.

```
proc logistic data=train2 des;
  model ins=&inputs48;
  score data=valid2 out=scoval2;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.TRAIN2	
Response Variable	INS	Insurance Product
Number of Response Levels	2	
Model	binary logit	
Optimization Technique	Fisher's scoring	

Number of Observations Read	21512
Number of Observations Used	21512

Response Profile

Ordered Value	INS	Total Frequency
1	1	7451
2	0	14061

Probability modeled is INS=1.

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	27759.675	22866.682
SC	27767.651	23066.092
-2 Log L	27757.675	22816.682

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	4940.9925	24	<.0001
Score	4674.1126	24	<.0001
Wald	3628.2749	24	<.0001

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.9715	0.0581	1151.4808	<.0001
MIPhone	1	-0.3425	0.0623	30.2307	<.0001
TELLER	1	0.0559	0.00808	47.8327	<.0001
MM	1	0.7747	0.0505	235.7065	<.0001
ILS	1	-0.2186	0.0810	7.2894	0.0069
LOC	1	-0.2760	0.0707	15.2462	<.0001
CD	1	0.9614	0.0473	412.6956	<.0001
CCPURC	1	0.0809	0.0400	4.0812	0.0434
ATMAMT	1	0.000027	5.567E-6	22.8649	<.0001
brclus2	1	0.2586	0.0366	49.8116	<.0001
INV	1	0.5426	0.1015	28.6009	<.0001
DEP	1	-0.0725	0.0139	27.0357	<.0001
IRA	1	0.3152	0.0714	19.4742	<.0001
MIAcctAg	1	-0.2076	0.0671	9.5746	0.0020
MTGBAL	1	-2.87E-6	8.771E-7	10.6979	0.0011
ACCTAGE	1	-0.0208	0.00271	58.9850	<.0001
SAVBAL	1	0.000130	4.964E-6	683.6868	<.0001
B_DDABal	1	0.0180	0.000633	811.5302	<.0001
SAV	1	0.2375	0.0396	36.0415	<.0001
PHONE	1	-0.0642	0.0188	11.6484	0.0006
CCBAL	1	2.834E-6	8.839E-7	10.2802	0.0013
MTG	1	-0.2637	0.0866	9.2753	0.0023
DIRDEP	1	-0.1728	0.0402	18.4607	<.0001
ATM	1	-0.1556	0.0405	14.7301	0.0001
brclus1	1	-0.3878	0.1171	10.9629	0.0009

Odds Ratio Estimates				
Effect	Point Estimate	95% Wald Confidence Limits		
MIPhone	0.710	0.628	0.802	
TELLER	1.057	1.041	1.074	
MM	2.170	1.966	2.396	
ILS	0.804	0.686	0.942	
LOC	0.759	0.661	0.872	
CD	2.615	2.384	2.870	
CCPURC	1.084	1.002	1.173	
ATMAMT	1.000	1.000	1.000	
brclus2	1.295	1.205	1.392	
INV	1.720	1.410	2.099	
DEP	0.930	0.905	0.956	
IRA	1.371	1.191	1.576	
MIAcctAg	0.813	0.712	0.927	
MTGBAL	1.000	1.000	1.000	
ACCTAGE	0.979	0.974	0.985	
SAVBAL	1.000	1.000	1.000	
B_DDABal	1.018	1.017	1.019	
SAV	1.268	1.173	1.370	
PHONE	0.938	0.904	0.973	
CCBAL	1.000	1.000	1.000	
MTG	0.768	0.648	0.910	
DIRDEP	0.841	0.778	0.910	
ATM	0.856	0.791	0.927	
brclus1	0.679	0.539	0.854	

Association of Predicted Probabilities and Observed Responses				
Percent Concordant	78.4	Somers' D	0.571	
Percent Discordant	21.4	Gamma	0.572	
Percent Tied	0.2	Tau-a	0.258	
Pairs	104768511	c	0.785	

The c statistic on the validation data is easy to obtain.

```
proc logistic data=scoval2 des;
  model ins=p_1;
run;
```

Partial Output

Association of Predicted Probabilities and Observed Responses				
Percent Concordant	78.0	Somers' D	0.562	
Percent Discordant	21.8	Gamma	0.564	
Percent Tied	0.3	Tau-a	0.255	
Pairs	26172272	c	0.781	

This model seems like a very good performer.

5.02 Multiple Choice Poll

On the validation data set, which model had the highest overall average profit?

- a. model 10
- b. model 11
- c. model 12
- d. model 13

5.2 Chapter Summary

The output of a predictive model should be predictions that generalize well over time. Using the validation data set for honest assessment, a series of models can be compared according to their fitness to this task. Appropriate measures of fitness and complexity may vary.

In the absence of profit information, the MSE (or, equivalently, ASE) can be used as a model fitness metric to detect models with low bias, or lack-of-fit.

5.3 Solutions

Solutions to Exercises

Solutions to Student Activities (Polls/Quizzes)

5.01 Multiple Choice Poll – Correct Answer

Which of the following statements is false regarding model assessment statistics?

- a. The shrinkage statistic is a measurement of the difference between the training model results and the validation model results.
- b. The model with the smallest estimated mean squared error will also be the least imprecise.
- c. The variance component of the mean squared error is independent of any fitted model.
- ☒ d. The bias component of the mean squared error is independent of any fitted model.

28

5.02 Multiple Choice Poll – Correct Answer

On the validation data set, which model had the highest overall average profit?

- a. model 10
- ☒ b. model 11
- c. model 12
- d. model 13

33

Appendix A Exercises and Solutions

Session 1: Exercises **A-4**

Session 1: Solutions..... **A-7**

A national veterans' organization seeks to better target its solicitations for donation. By only soliciting the most likely donors, less money will be spent on solicitation efforts and more money will be available for charitable concerns. Solicitations involve sending a small gift to an individual together with a request for donation. Gifts include mailing labels and greeting cards.

The organization has more than 3.5 million individuals in its mailing database. These individuals have been classified by their response behavior to previous solicitation efforts. Of particular interest is the class of individuals identified as lapsing donors. These individuals made their most recent donation between 12 and 24 months ago. The organization found that by predicting the response behavior of this group, they could use the model to rank all 3.5 million individuals in their database. With this ranking, a decision can be made to either solicit or ignore an individual in the current solicitation campaign. The current campaign refers to a greeting card mailing sent in June of 1997. It is identified in the raw data as the 97NK campaign.

The raw analysis data has been reduced for the purpose of this course. A subset of slightly over 19,000 records has been selected for modeling. As will be seen, this subset was not chosen arbitrarily. In addition, the 481 fields have been reduced to 47. Considering their potential association with the analysis objective eliminated some of the fields (for example, it is doubtful that CD player ownership is strongly correlated with donation potential). Other fields were combined to form summaries of a particular customer behavior. The following table details the variables and descriptions:

Variable	Description
CARD_PROM_12	Count of card promotions in the last 12 months
CLUSTER_CODE	Socio-Economic Cluster Code
CONTROL_NUMBER	ID
DONOR_AGE	Donor Age
DONOR_GENDER	Donor Gender
FREQUENCY_STATUS_97NK	Count of Donations between June 1995 and June 1996 (capped at 4)
HOME_OWNER	Home Owner flag
INCOME_GROUP	Income Bracket, from 1 to 7
IN_HOUSE	Flag for <i>In-House</i> donor program
LAST_GIFT_AMT	Amount of most recent donation
LIFETIME_AVG_GIFT_AMT	Average donation amount, ever
LIFETIME_CARD_PROM	Number of card promotions, ever
LIFETIME_GIFT_AMOUNT	Total donation amount, ever
LIFETIME_GIFT_COUNT	Total number of donations, ever
LIFETIME_GIFT_RANGE	Maximum gift amount less minimum gift amount
LIFETIME_MAX_GIFT_AMT	Maximum gift amount, ever
LIFETIME_MIN_GIFT_AMT	Minimum gift amount, ever
LIFETIME_PROM	Count of solicitations ever sent

MEDIAN_HOME_VALUE	Census data
MEDIAN_HOUSEHOLD_INCOME	Census data
MONTHS_SINCE_FIRST_GIFT	Months since first donation
MONTHS_SINCE_LAST_GIFT	Months since most recent donation
MONTHS_SINCE_ORIGIN	Months since entry onto the file
MOR_HIT_RATE	Data recorded by a third party-Mail Order Response rate
NUMBER_PROM_12	Count of promotions in the last 12 months
OVERLAY_SOURCE	Source of Demographic overlay
PCT_MALE_MILITARY	Census data
PCT_MALE_VETERANS	Census data
PCT_OWNER_OCCUPIED	Census data
PCT_VIETNAM_VETERANS	Census data
PCT_WWII_VETERANS	Census data
PEP_STAR	Flag to identify consecutive donors
PER_CAPITA_INCOME	Census data
PUBLISHED_PHONE	Flag
REGENCY_STATUS_96NK	Categorization of donation patterns
RECENT_AVG_CARD_GIFT_AMT	Average donation amount to card promotions since June 1994
RECENT_AVG_GIFT_AMT	Average donation amount to promotions since June 1994
RECENT_CARD_RESPONSE_COUNT	Count of responses to card promotions since June 1994
RECENT_CARD_RESPONSE_PROP	Proportion of responses to card promotions since June 1994
RECENT_RESPONSE_COUNT	Count of responses to promotions since June 1994
RECENT_RESPONSE_PROP	Proportion of responses to promotions since June 1994
RECENT_STAR_STATUS	STAR status flag, since June 1994
SES	A clustering of the levels of CLUSTER_CODE
TARGET_B	B=Binary, flag for response to 97NK—Target Variable
TARGET_D	Dollar amount of response to 97NK
URBANICITY	Categorization of residency
WEALTH_RATING	Measures wealth relative to others within state

Session 1: Exercises

Chapter 2 Exercises

1. Investigate the inputs in the file **PMLR.PVA_RAW_DATA**. You may choose to create a temporary copy of the data. Investigate a preliminary model, for example, **PVA**.
 - a. Create a table of the mean, minimum, maximum, and count of missing for each numeric input.
 - b. Create tables of the categorical inputs. Do **not** create a table using **CONTROL_NUMBER**, the identification key.
 - c. Create a macro variable to store π_1 , the proportion of responders in the population. This value is 0.05.
 - d. The current model consists of **PEP_STAR**, **RECENT_AVG_GIFT_AMT**, and **FREQUENCY_STATUS_97NK**. Fit this model to the data. Use the **SCORE** statement to append the predicted probability to the data, correcting for oversampling. Investigate the minimum, maximum, and average predicted probability of response based on this model.
 - e. How many individuals in the sample have a predicted response rate greater than 0.025? 0.05? 0.075? How many responders?


Chapter 3 Exercises

1. Replace missing values using group-median imputation.
 - a. Create missing value indicators for inputs that have missing values.
 - b. Submit the program below to group the variables **RECENT_RESPONSE_PROP** and **RECENT_AVG_GIFT_AMT** into three groups. The **RANK** procedure with the **GROUPS=** option bins variables into quantiles. The **VAR** statement lists the variables to be grouped. The **RANKS** statement names the group indicators in the **OUT=** data set. If the **RANKS** statement is omitted, then the group indicators replace the **VAR** variables in the **OUT=** data set.

```
proc rank data=pva out=pva groups=3;  
  var recent_response_prop recent_avg_gift_amt;  
  ranks grp_resp grp_amt;  
run;
```

- c. Sort **PVA** by **grp_resp** and **grp_amt**.
 - d. Use the **STDIZE** procedure with a **BY** statement to impute missing values for each **BY** group and output the completed data set. Name the output data set **PVA1**.
 - e. Use the **MEANS** procedure to determine the values that the missing values were replaced with.
 - f. (Optional) If you use the **OUTSTAT=** option in the **PROC STDIZE** statement, you can save the information used by the procedure for imputation. Print out the **OUTSTAT=** data set for a more compact view of the information.

2. Use Greenacre's correspondence analysis to cluster levels of **CLUSTER_CODE** and **RECENCY_STATUS_96NK**.
 - a. Use the MEANS procedure to generate a data set with information about the average response rate and sample size for each level of **CLUSTER_CODE**.
 - b. Use the CLUSTER procedure to group those levels together.
 - c. Use the p -value of the appropriate χ^2 test to determine which level of clustering is appropriate.
 - d. Create indicators to flag membership in those clusters.
 - e. Repeat steps a and b with **RECENCY_STATUS_96NK**, but choose the appropriate level of clustering by selecting the level of clustering associated with an R^2 of at least 99%.
 - f. Create indicator flags for this cluster solution as well.
3. Perform variable clustering to stem multicollinearity.
 - a. The MAXEIGEN= option in the PROC VARCLUS statement establishes a stopping rule for the divisive clustering procedure. A similar stopping criterion that may be more intuitive is the PERCENT= option, which specifies the percent of variation that must be explained by the first eigenvalue. Use the VARCLUS procedure to cluster all numeric variables, but use PERCENT=80 instead of MAXEIGEN= as your stopping criterion.

 If you do not specify OUTTREE=, use the HI option in the PROC VARCLUS statement to force divisive clustering and save some processing time.
 - b. Select one variable from each cluster as a representative. Did you choose variables according to the smallest $1-R^2$ ratio, or did you choose some according to subject-matter considerations?
 - c. Fit a logistic regression model with the FAST BACKWARD method using only the selected cluster representatives as inputs. How will you assess this model's fit?
4. Compare variable selection methods.
 - a. Fit a logistic regression model with the STEPWISE method. Use all of the numeric variables and the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Set SLSTAY and SLENTY equal to .001.
 - b. Fit a logistic regression model with the FAST BACKWARD method. Use all of the numeric variables and the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Set SLSTAY equal to .001.
 - c. Fit a logistic regression model with the SCORE method and the BEST=1 option. Use all of the numeric variables and create dummies for the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Use the Output Delivery System to determine which model is the best according to the SBC criterion. (How did you decide what level of the dummies to make the reference level?)

- d. Fit a stepwise linear regression model using the REG procedure:

```
proc reg;  
  model target=list numeric inputs / selection=stepwise  
                                     slstay=.001 slentry=.001;  
run;
```


- e. How does the c statistic computed from the model generated in PROC REG compare with the c statistics from the models generated in PROC LOGISTIC? What reasons might one have for choosing PROC LOGISTIC over PROC REG as a modeling tool? What reasons might one have for choosing PROC REG over PROC LOGISTIC as a modeling tool?

Chapter 4 Exercises

1. Assess some logistic regression models using a validation data set.
 - a. Split the imputed data set into training and validation data sets. Use 50% of the data for each data set role. Stratify on the target variable.
 - b. (Optional) Use the Spearman and Hoeffding correlation coefficients to screen the inputs with the least evidence of a relationship with the target. If necessary, consider transforming some inputs that do well according to the Hoeffding correlation measure.

Chapter 5 Exercises

1. Assess some logistic regression models using a validation data set.
 - a. Using the macros outlined in Chapter 5, score and evaluate the performance of the best subsets models on the training and validation data sets.



If you want to use average profit, you can calculate the profit matrix from the following information: The average responder donates \$15.62, and the cost of a solicitation is \$0.68.
 - b. Use graphical summaries to determine what models are of interest. If these are inconclusive, what measures will you take to choose the best model? You might consider lift charts or ROC curves.
2. (Optional) Prepare scoring code for your selected model. Accommodate any specialized input that you created; for example, clustered levels of **CLUSTER_CODE**.

Session 1: Solutions

Chapter 2 Solutions

1. Investigate the inputs in the file, `PMLR.PVA_RAW_DATA`.

```
data pva(drop=control_number);
  set pmlr.pva_raw_data;
run;

%let ex_inputs= MONTHS_SINCE_ORIGIN
DONOR_AGE IN_HOUSE INCOME_GROUP PUBLISHED_PHONE
MOR_HIT_RATE WEALTH_RATING MEDIAN_HOME_VALUE
MEDIAN_HOUSEHOLD_INCOME PCT_OWNER_OCCUPIED
PER_CAPITA_INCOME PCT_MALE_MILITARY
PCT_MALE_VETERANS PCT_VIETNAM_VETERANS
PCT_WWII_VETERANS PEP_STAR RECENT_STAR_STATUS
FREQUENCY_STATUS_97NK RECENT_RESPONSE_PROP
RECENT_AVG_GIFT_AMT RECENT_CARD_RESPONSE_PROP
RECENT_AVG_CARD_GIFT_AMT RECENT_RESPONSE_COUNT
RECENT_CARD_RESPONSE_COUNT LIFETIME_CARD_PROM
LIFETIME_PROM LIFETIME_GIFT_AMOUNT
LIFETIME_GIFT_COUNT LIFETIME_AVG_GIFT_AMT
LIFETIME_GIFT_RANGE LIFETIME_MAX_GIFT_AMT
LIFETIME_MIN_GIFT_AMT LAST_GIFT_AMT
CARD_PROM_12 NUMBER_PROM_12 MONTHS_SINCE_LAST_GIFT
MONTHS_SINCE_FIRST_GIFT;
```

- a. Create a table of the mean, minimum, maximum, and count of missing for each numeric input.

```
proc means data=pva n nmiss mean std min max;
  var &ex_inputs;
run;
```

Variable	N		Mean	Std Dev	Minimum	Maximum
	N	Miss				
MONTHS_SINCE_ORIGIN	19372	0	73.4099732	41.2555742	5.0000000	137.0000000
DONOR_AGE	14577	4795	58.9190506	16.6693824	0	87.0000000
IN_HOUSE	19372	0	0.0731984	0.2604687	0	1.0000000
INCOME_GROUP	14980	4392	3.9075434	1.8647962	1.0000000	7.0000000
PUBLISHED_PHONE	19372	0	0.4977287	0.5000077	0	1.0000000
MOR_HIT_RATE	19372	0	3.3616560	9.5034812	0	241.0000000
WEALTH_RATING	10562	8810	5.0053967	2.8153860	0	9.0000000
MEDIAN_HOME_VALUE	19372	0	1079.87	960.7534484	0	6000.00
MEDIAN_HOUSEHOLD_INCOME	19372	0	341.9702147	164.2078074	0	1500.00
PCT_OWNER_OCCUPIED	19372	0	69.6989986	21.7110186	0	99.0000000
PER_CAPITA_INCOME	19372	0	15857.33	8710.63	0	174523.00
PCT_MALE_MILITARY	19372	0	1.0290109	4.9182974	0	97.0000000
PCT_MALE_VETERANS	19372	0	30.5739211	11.4214714	0	99.0000000
PCT_VIETNAM_VETERANS	19372	0	29.6032934	15.1203598	0	99.0000000
PCT_WWII_VETERANS	19372	0	32.8524675	17.8397648	0	99.0000000
PEP_STAR	19372	0	0.5044394	0.4999932	0	1.0000000
RECENT_STAR_STATUS	19372	0	0.9311377	2.5455850	0	22.0000000
FREQUENCY_STATUS_97NK	19372	0	1.9839975	1.0993458	1.0000000	4.0000000
RECENT_RESPONSE_PROP	19372	0	0.1901275	0.1139467	0	1.0000000
RECENT_AVG_GIFT_AMT	19372	0	15.3653959	10.1674849	0	260.0000000
RECENT_CARD_RESPONSE_PROP	19372	0	0.2308077	0.1862301	0	1.0000000
RECENT_AVG_CARD_GIFT_AMT	19372	0	11.6854703	10.8341202	0	300.0000000
RECENT_RESPONSE_COUNT	19372	0	3.0431034	2.0464006	0	16.0000000
RECENT_CARD_RESPONSE_COUNT	19372	0	1.7305389	1.5355208	0	9.0000000
LIFETIME_CARD_PROM	19372	0	18.6680776	8.5587782	2.0000000	56.0000000
LIFETIME_PROM	19372	0	47.5705141	22.9501581	5.0000000	194.0000000
LIFETIME_GIFT_AMOUNT	19372	0	104.4257165	105.7224599	15.0000000	3775.00
LIFETIME_GIFT_COUNT	19372	0	9.9797646	8.6881633	1.0000000	95.0000000
LIFETIME_AVG_GIFT_AMT	19372	0	12.8583383	8.7877579	1.3600000	450.0000000
LIFETIME_GIFT_RANGE	19372	0	11.5878758	15.1168929	0	997.0000000
LIFETIME_MAX_GIFT_AMT	19372	0	19.2088081	16.1011278	5.0000000	1000.00
LIFETIME_MIN_GIFT_AMT	19372	0	7.6209323	7.9597857	0	450.0000000
LAST_GIFT_AMT	19372	0	16.5841988	11.9775577	0	450.0000000
CARD_PROM_12	19372	0	5.3671278	1.2642046	0	17.0000000
NUMBER_PROM_12	19372	0	12.9018687	4.6420721	2.0000000	64.0000000
MONTHS_SINCE_LAST_GIFT	19372	0	18.1911522	4.0330648	4.0000000	27.0000000
MONTHS_SINCE_FIRST_GIFT	19372	0	69.4820875	37.5681693	15.0000000	260.0000000

- b. Create tables of the categorical inputs. Do **not** create a table using **CONTROL_NUMBER**, the identification key.

```
proc freq data=pva;
  tables _character_ target_b / missing;
run;
```

URBANICITY	Frequency	Percent	Cumulative Frequency	Cumulative Percent
?	454	2.34	454	2.34
C	4022	20.76	4476	23.11
R	4005	20.67	8481	43.78
S	4491	23.18	12972	66.96
T	3944	20.36	16916	87.32
U	2456	12.68	19372	100.00

SES	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	5924	30.58	5924	30.58
2	9284	47.92	15208	78.51
3	3323	17.15	18531	95.66
4	387	2.00	18918	97.66
?	454	2.34	19372	100.00

CLUSTER_ CODE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
.	454	2.34	454	2.34
01	239	1.23	693	3.58
02	380	1.96	1073	5.54
03	300	1.55	1373	7.09
04	113	0.58	1486	7.67
05	199	1.03	1685	8.70
06	123	0.63	1808	9.33
07	184	0.95	1992	10.28
08	378	1.95	2370	12.23
09	153	0.79	2523	13.02
10	387	2.00	2910	15.02
11	484	2.50	3394	17.52
12	631	3.26	4025	20.78
13	579	2.99	4604	23.77
14	454	2.34	5058	26.11
15	223	1.15	5281	27.26
16	384	1.98	5665	29.24
17	349	1.80	6014	31.04
18	619	3.20	6633	34.24
19	98	0.51	6731	34.75
20	317	1.64	7048	36.38
21	353	1.82	7401	38.20
22	251	1.30	7652	39.50
23	293	1.51	7945	41.01
24	795	4.10	8740	45.12
25	273	1.41	9013	46.53
26	202	1.04	9215	47.57
27	666	3.44	9881	51.01
28	343	1.77	10224	52.78
29	170	0.88	10394	53.65
30	519	2.68	10913	56.33
31	249	1.29	11162	57.62
32	152	0.78	11314	58.40
33	109	0.56	11423	58.97
34	284	1.47	11707	60.43

35	727	3.75	12434	64.19
36	716	3.70	13150	67.88
37	204	1.05	13354	68.93
38	240	1.24	13594	70.17
39	512	2.64	14106	72.82
40	830	4.28	14936	77.10
41	431	2.22	15367	79.33
42	284	1.47	15651	80.79
43	468	2.42	16119	83.21
44	383	1.98	16502	85.18
45	482	2.49	16984	87.67
46	369	1.90	17353	89.58
47	185	0.95	17538	90.53
48	180	0.93	17718	91.46
49	675	3.48	18393	94.95
50	156	0.81	18549	95.75
51	460	2.37	19009	98.13
52	60	0.31	19069	98.44
53	303	1.56	19372	100.00

HOME_ OWNER	Frequency	Percent	Cumulative Frequency	Cumulative Percent
H	10606	54.75	10606	54.75
U	8766	45.25	19372	100.00

DONOR_ GENDER	Frequency	Percent	Cumulative Frequency	Cumulative Percent
A	1	0.01	1	0.01
F	10401	53.69	10402	53.70
M	7953	41.05	18355	94.75
U	1017	5.25	19372	100.00

OVERLAY_ SOURCE	Frequency	Percent	Cumulative Frequency	Cumulative Percent
B	8732	45.08	8732	45.08
M	1480	7.64	10212	52.72
N	4392	22.67	14604	75.39
P	4768	24.61	19372	100.00

REGENCY_ STATUS_ 96NK	Frequency	Percent	Cumulative Frequency	Cumulative Percent
A	11918	61.52	11918	61.52
E	427	2.20	12345	63.73
F	1521	7.85	13866	71.58
L	93	0.48	13959	72.06
N	1192	6.15	15151	78.21
S	4221	21.79	19372	100.00

Target Variable Indicates for Response to 97NK Mailing

TARGET_B	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	14529	75.00	14529	75.00
1	4843	25.00	19372	100.00

- c. Create a macro variable to store π_1 , the proportion of responders in the population. This value is 0.05.

```
%let ex_pil=0.05;
```

- d. The current model consists of **PEP_STAR**, **RECENT_AVG_GIFT_AMT**, and **FREQUENCY_STATUS_97NK**. Fit this model to the data. Use the **SCORE** statement to append the predicted probability to the data, correcting for oversampling. Investigate the minimum, maximum, and average predicted probability of response based on this model.

```
proc logistic data=pva des;
  model target_b=PEP_STAR
          RECENT_AVG_GIFT_AMT
          FREQUENCY_STATUS_97NK;
  score data=PMLR.PVA_RAW_DATA
        out=scopva
        priorevent=&ex_pil;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.PVA
Response Variable	TARGET_B
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Model Information

Target Variable Indicates for Response to 97NK Mailing

Number of Observations Read	19372
Number of Observations Used	19372

Response Profile

Ordered Value	TARGET_B	Total Frequency
1	1	4843
2	0	14529

Probability modeled is TARGET_B=1.

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	21789.113	21351.668
SC	21796.984	21383.154
-2 Log L	21787.113	21343.668

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	443.4451	3	<.0001
Score	448.9127	3	<.0001
Wald	439.1624	3	<.0001

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.6241	0.0584	773.5119	<.0001
PEP_STAR	1	0.3078	0.0361	72.8485	<.0001
RECENT_AVG_GIFT_AMT	1	-0.00555	0.00204	7.3829	0.0066
FREQUENCY_STATUS_97N	1	0.2147	0.0168	163.3538	<.0001

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
PEP_STAR	1.360	1.268	1.460
RECENT_AVG_GIFT_AMT	0.994	0.990	0.998
FREQUENCY_STATUS_97N	1.239	1.199	1.281

Association of Predicted Probabilities and Observed Responses

Percent Concordant	58.9	Somers' D	0.201
Percent Discordant	38.8	Gamma	0.206
Percent Tied	2.3	Tau-a	0.075
Pairs	70363947	c	0.601

```
proc means data=scopva min mean max;
  var p_1;
run;
```


The MEANS Procedure

Analysis Variable : P_1 Predicted Probability: TARGET_B=1

Minimum	Mean	Maximum
0.0090269	0.0512445	0.0901428

- e. How many individuals in the sample have a predicted response rate greater than 0.025? 0.05? 0.075? How many responders?

```
proc means data=scopva n sum;
  where p_1 > .025;
  var target_b;
run;
```

N	Sum
19323	4833.00

```
proc means data=scopva n sum;
  where p_1 > .05;
  var target_b;
run;
```

N	Sum
8161	2579.00

```
proc means data=scopva n sum;
  where p_1 > .075;
  var target_b;
run;
```

N	Sum
2240	830.0000000

Chapter 3 Solutions

1. Replace missing values using group-median imputation.

- a. Create missing value indicators for inputs that have missing values.

```
data pva(drop=i);
  set pva;
  /* name the missing indicator variables */
  array mi{*} mi_DONOR_AGE mi_INCOME_GROUP
             mi_WEALTH_RATING;
  /* select variables with missing values */
  array x{*} DONOR_AGE INCOME_GROUP WEALTH_RATING;
  do i=1 to dim(mi);
    mi{i}=(x{i}=.);
  end;
run;
```

- b. Group the variables **RECENT_RESPONSE_PROP** and **RECENT_AVG_GIFT_AMT** into three groups.

```
proc rank data=pva out=pva groups=3;
  var recent_response_prop recent_avg_gift_amt;
  ranks grp_resp grp_amt;
run;

proc freq data=pva;
  tables grp_resp*grp_amt;
run;
```

grp_resp(Rank for Variable RECENT_RESPONSE_PROP)				
grp_amt(Rank for Variable RECENT_AVG_GIFT_AMT)				
Frequency				
Percent				
Row Pct				
Col Pct	0	1	2	Total
0	969	2325	3216	6510
	5.00	12.00	16.60	33.61
	14.88	35.71	49.40	
	14.66	36.83	49.85	
1	1396	2557	2394	6347
	7.21	13.20	12.36	32.76
	21.99	40.29	37.72	
	21.12	40.51	37.11	
2	4244	1430	841	6515
	21.91	7.38	4.34	33.63
	65.14	21.95	12.91	
	64.22	22.66	13.04	
Total	6609	6312	6451	19372
	34.12	32.58	33.30	100.00

c. Sort **PVA** by **grp_resp** and **grp_amt**.

```
proc sort data=pva out=pva;
  by grp_resp grp_amt;
run;
```

d. Use the **STDIZE** procedure with a **BY** statement to impute missing values for each **BY** group and output the completed data set. Name the output data set **PVA1**.

```
proc stdize data=pva method=median
  reponly out=pva1;
  by grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;
```

e. Use the **MEANS** procedure to determine the values that the missing values were replaced with.

```
proc means data=pva1 median;
  class grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;
```

Rank for Variable RECENT_RESPONSE_ PROP	Rank for Variable RECENT_AVG_ GIFT_AMT	N Obs	Variable	Median
0	0	969	DONOR_AGE	64.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	1	2325	DONOR_AGE	59.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	2	3216	DONOR_AGE	57.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	6.0000000
1	0	1396	DONOR_AGE	65.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	1	2557	DONOR_AGE	58.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	2	2394	DONOR_AGE	57.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	6.0000000

2	0	4244	DONOR_AGE	63.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	1	1430	DONOR_AGE	61.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	5.0000000
	2	841	DONOR_AGE	59.0000000
			INCOME_GROUP	4.0000000
			WEALTH_RATING	6.0000000

- f. (Optional) If you use the OUTSTAT= option in the PROC STDIZE statement, you can save the information used by the procedure for imputation. Print out the OUTSTAT= data set for a more compact view of the information.

```
proc stdize data=pva method=median
    reonly out=pval outstat=med;
    by grp_resp grp_amt;
    var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;

proc print data=med;
    where _TYPE_ = "LOCATION";
run;
```

Obs	grp_resp	grp_amt	_TYPE_	DONOR_AGE	INCOME_GROUP	WEALTH_RATING
1	0	0	LOCATION	64	4	5
9	0	1	LOCATION	59	4	5
17	0	2	LOCATION	57	4	6
25	1	0	LOCATION	65	4	5
33	1	1	LOCATION	58	4	5
41	1	2	LOCATION	57	4	6
49	2	0	LOCATION	63	4	5
57	2	1	LOCATION	61	4	5
65	2	2	LOCATION	59	4	6

2. Use Greenacre's correspondence analysis to cluster levels of **CLUSTER_CODE** and **REGENCY_STATUS_96NK**.

- a. Use the MEANS procedure to generate a data set with information about the average response rate and sample size for each level of **CLUSTER_CODE**.

```
proc means data=pval noprint nway;
    class CLUSTER_CODE;
    var target_b;
    output out=level mean=prop;
run;
```

- b. Use the CLUSTER procedure to group those levels together.

```
ods listing close;
ods output clusterhistory=cluster;

proc cluster data=level
            method=ward
            outtree=fortree;
    freq _freq_;
    var prop;
    id CLUSTER_CODE;
run;

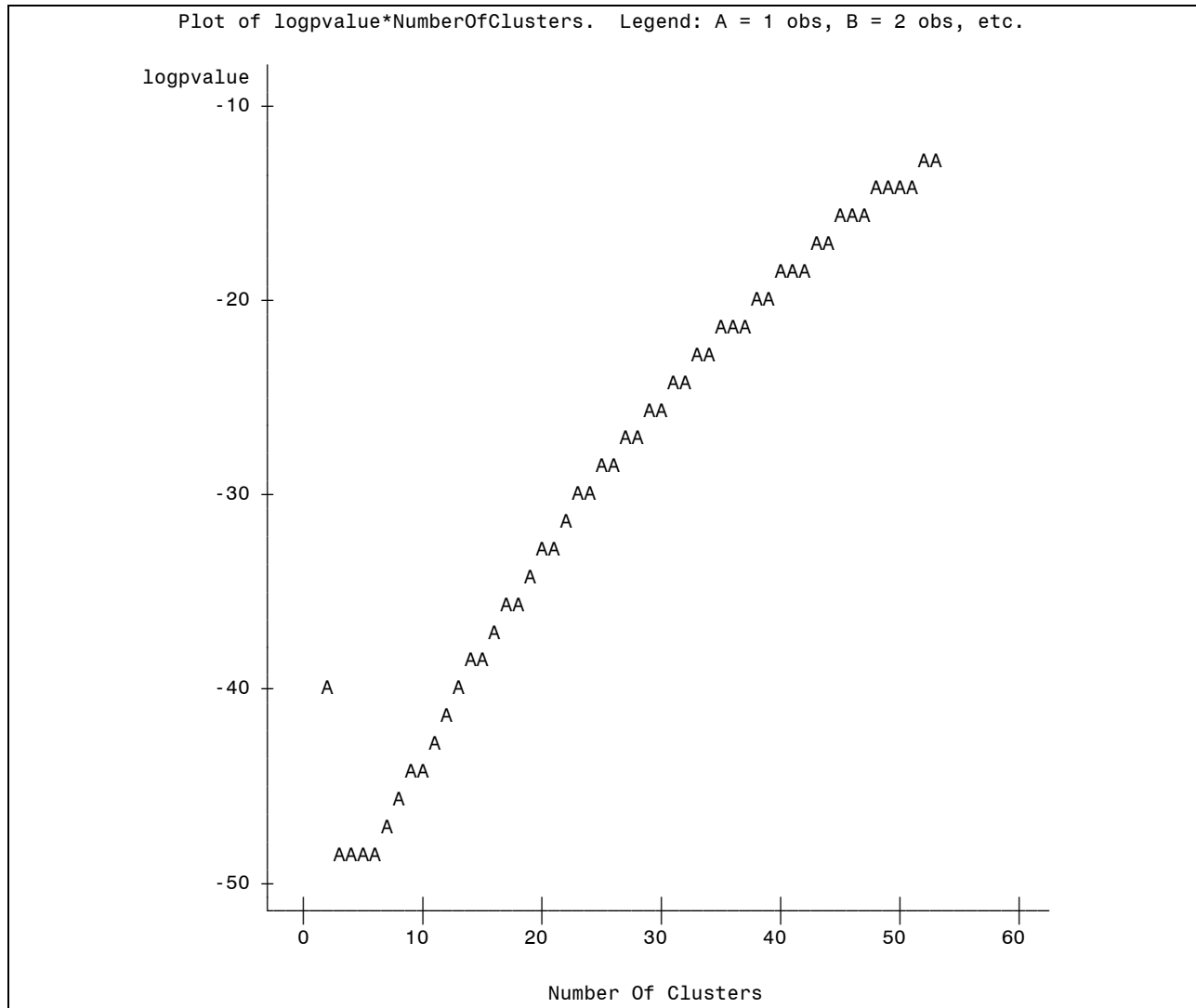
ods listing;
```

- c. Use the p -value of the appropriate χ^2 test to determine which level of clustering is appropriate.

```
proc freq data=pval noprint;
    tables CLUSTER_CODE*TARGET_B / chisq;
    output out=chi(keep=_pchi_) chisq;
run;

data cutoff;
    if _n_ = 1 then set chi;
    set cluster;
    chisquare=_pchi_*rsquared;
    degfree=numberofclusters-1;
    logpvalue=logsf('CHISQ',chisquare,degfree);
run;

proc plot data=cutoff;
    plot logpvalue*numberofclusters/vpos=30;
run; quit;
```

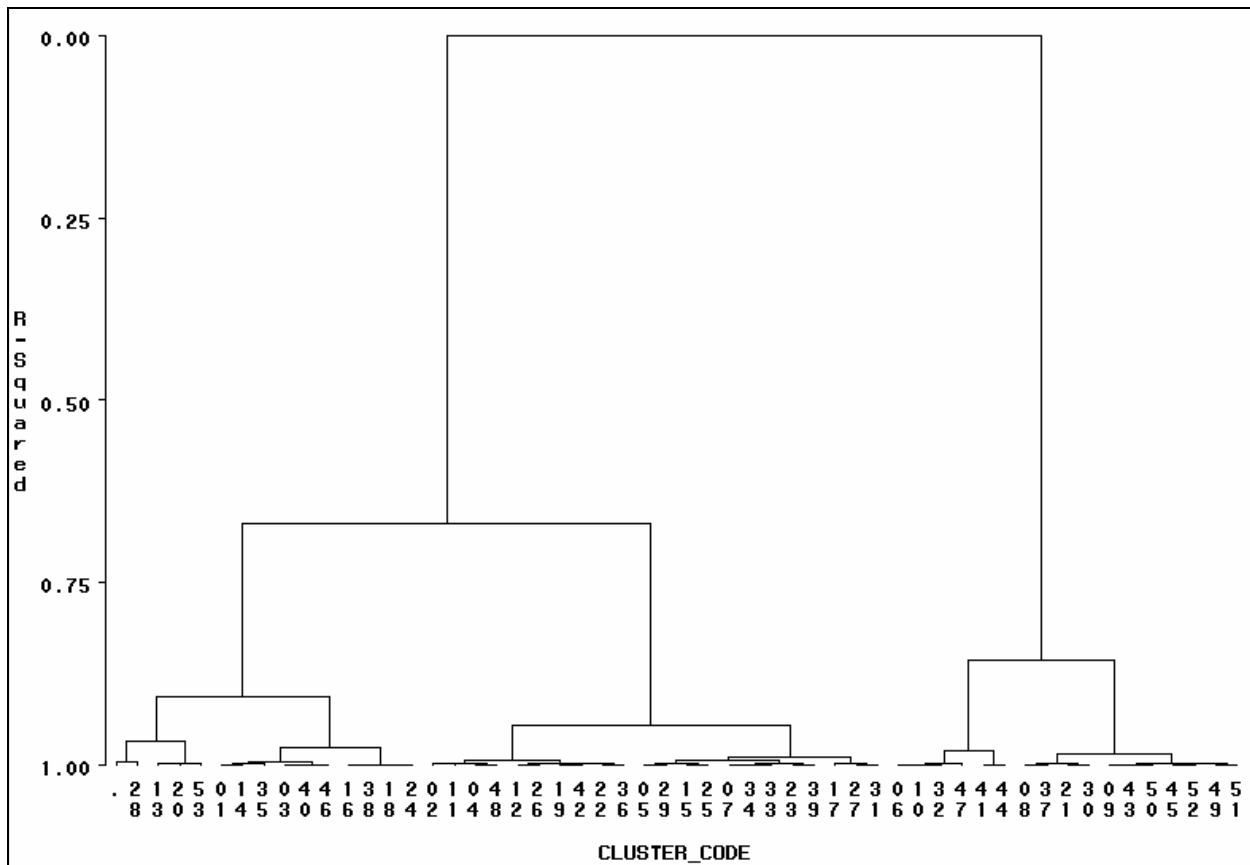


```
proc sql;
  select NumberOfClusters into :ncl
  from cutoff
  having logpvalue=min(logpvalue);
quit;
```

Number
Of
Clusters

5

```
proc tree data=fortree h=rsq
      nclusters=&ncl out=clus;
      id CLUSTER_CODE;
run;
```



```
proc sort data=clus;
      by clusname;
run;

proc print data=clus;
      by clusname;
      id clusname;
run;
```

CLUSNAME	CLUSTER_	CLUSTER
	CODE	
CL5	23	3
	39	3
	07	3
	34	3
	27	3
	31	3
	33	3
	22	3
	36	3
	05	3
	29	3
	04	3
	48	3
	12	3
	26	3
	15	3
	25	3
	19	3
	42	3
	02	3
	11	3
	17	3
CL6	13	5
	20	5
	53	5
	.	5
	28	5
CL7	16	1
	38	1
	03	1
	40	1
	18	1
	24	1
	01	1
	14	1
	46	1
	35	1

CL8	06	4
	10	4
	32	4
	41	4
	44	4
	47	4
CL9	09	2
	43	2
	49	2
	51	2
	21	2
	30	2
	45	2
	52	2
	08	2
	37	2
	50	2

d. Create indicators to flag membership in those clusters.

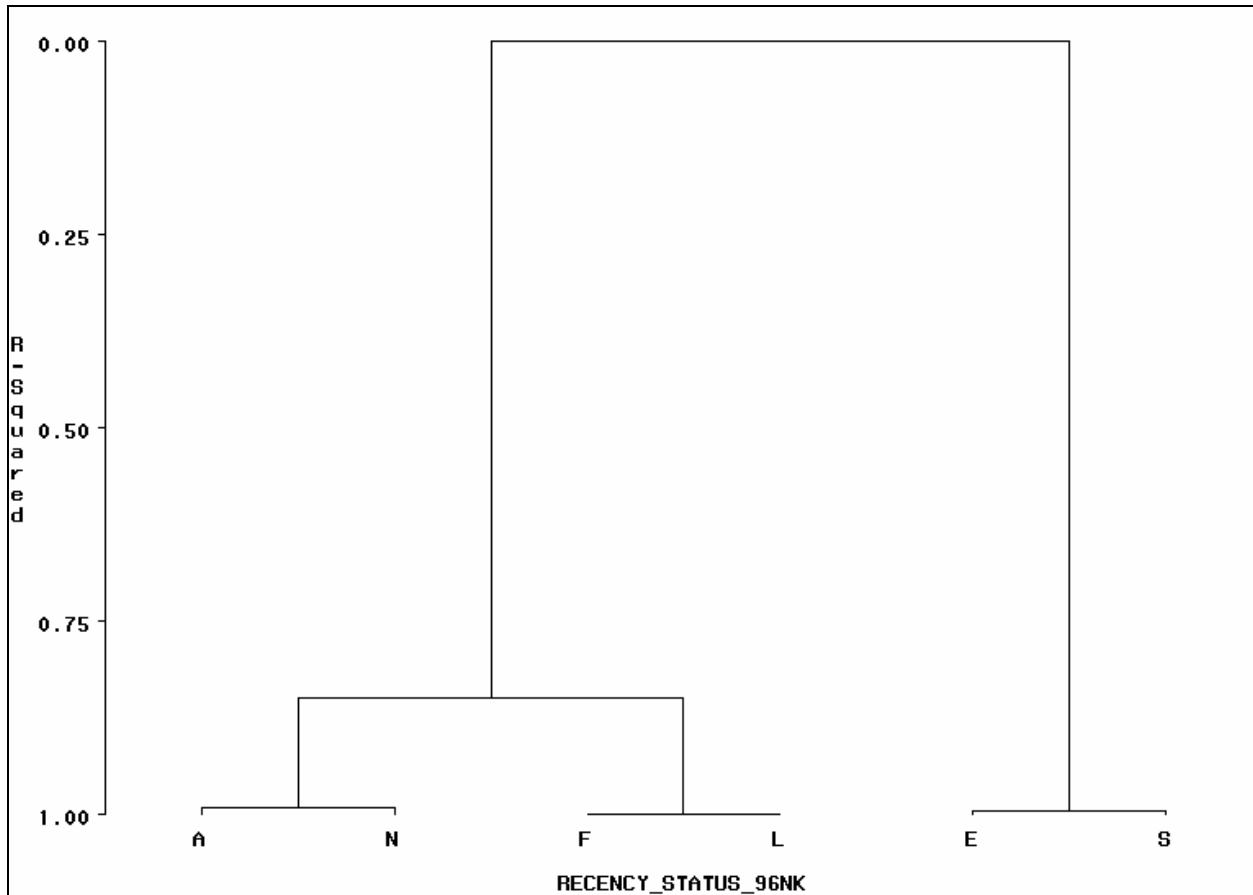
```
data pval;
  set pval;
  ClusCdGrp1 = CLUSTER_CODE in("13", "20", "53", ".", "28");
  ClusCdGrp2 = CLUSTER_CODE in("16", "38", "03", "40", "18",
                                "24", "01", "14", "46", "35");
  ClusCdGrp3 = CLUSTER_CODE in("06", "10", "32", "41", "44", "47");
  ClusCdGrp4 = CLUSTER_CODE in("09", "43", "49", "51",
                                "21", "30", "45", "52",
                                "08", "37", "50");
run;
```

e. Repeat steps a and b with **RECENCY_STATUS_96NK**, but choose the appropriate level of clustering by selecting the level of clustering associated with an R^2 of at least 99%.

```
proc means data=pval mean nway;
  class RECENCY_STATUS_96NK;
  var target_b;
  output out=level mean=prop;
run;

proc cluster data=level
             method=ward
             outtree=fortree;
  freq _freq_;
  var prop;
  id RECENCY_STATUS_96NK;
run;

proc tree data=fortree h=rsq;
  id RECENCY_STATUS_96NK;
run;
```



f. Create indicator flags for this cluster solution as well.

```
data pval;
  set pval;
  statusFL=RECENCY_STATUS_96NK in("F","L");
  statusES=RECENCY_STATUS_96NK in("E","S");
run;
```

3. Perform variable clustering to stem multicollinearity.

a. Use the VARCLUS procedure to cluster all numeric variables, but use PERCENT=80 instead of MAXEIGEN= as your stopping criterion.

```
proc varclus data=pval
  short hi
  percent=80;
  var &ex_inputs ClusCd: mi_: statusFL statusES;
run;
```

- b. Select one variable from each cluster as a representative. Using only the $1-R^2$ ratio as a criterion, you might get:

```
%let ex_screened=
LIFETIME_CARD_PROM      LIFETIME_MIN_GIFT_AMT      PER_CAPITA_INCOME
CARD_PROM_12            mi_INCOME_GROUP        ClusCdGrp1
RECENT_RESPONSE_COUNT   ClusCdGrp3              PCT_MALE_MILITARY
DONOR_AGE                PCT_VIETNAM_VETERANS      MOR_HIT_RATE
PCT_OWNER_OCCUPIED      PCT_MALE_VETERANS        PUBLISHED_PHONE
ClusCdGrp4              WEALTH_RATING          ClusCdGrp2
MONTHS_SINCE_LAST_GIFT  RECENT_STAR_STATUS      LIFETIME_GIFT_RANGE
INCOME_GROUP            IN_HOUSE                statusFL
statusES                RECENT_AVG_GIFT_AMT      PCT_WWII_VETERANS
LIFETIME_GIFT_AMOUNT    PEP_STAR                mi_DONOR_AGE
RECENT_AVG_CARD_GIFT_AMT RECENT_CARD_RESPONSE_PROP
;
```

- c. Fit a logistic regression model with the FAST BACKWARD method using only the selected cluster representatives as inputs. How will you assess this model's fit?

```
proc logistic data=pval des namelen=32;
  model target_b = &ex_screened
    /selection=backward fast;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.PVA1
Response Variable	TARGET_B
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	19372
Number of Observations Used	19372

Response Profile

Ordered Value	TARGET_B	Total Frequency
1	1	4843
2	0	14529

Probability modeled is TARGET_B=1.

Backward Elimination Procedure

Step 0. The following effects were entered:

Intercept LIFETIME_CARD_PROM LIFETIME_MIN_GIFT_AMT PER_CAPITA_INCOME CARD_PROM_12
 mi_INCOME_GROUP ClusCdGrp1 RECENT_RESPONSE_COUNT ClusCdGrp3 PCT_MALE_MILITARY DONOR_AGE
 PCT_VIETNAM_VETERANS MOR_HIT_RATE PCT_OWNER_OCCUPIED PCT_MALE_VETERANS PUBLISHED_PHONE
 ClusCdGrp4 WEALTH_RATING ClusCdGrp2 MONTHS_SINCE_LAST_GIFT RECENT_STAR_STATUS
 LIFETIME_GIFT_RANGE INCOME_GROUP IN_HOUSE statusFL statusES RECENT_AVG_GIFT_AMT
 PCT_WWII_VETERANS LIFETIME_GIFT_AMOUNT PEP_STAR mi_DONOR_AGE RECENT_AVG_CARD_GIFT_AMT
 RECENT_CARD_RESPONSE_PROP

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	21789.113	21159.997
SC	21796.984	21419.759
-2 Log L	21787.113	21093.997

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	693.1162	32	<.0001
Score	691.1088	32	<.0001
Wald	662.3084	32	<.0001

Step 1. Fast Backward Elimination:

Analysis of Effects Removed by Fast Backward Elimination

Effect Removed	Chi-Square	DF	Pr > ChiSq	Residual Chi-Square	DF	Pr > Residual ChiSq
PCT_OWNER_OCCUPIED	0.0018	1	0.9660	0.0018	1	0.9660
PCT_MALE_MILITARY	0.0037	1	0.9517	0.0055	2	0.9973
PCT_WWII_VETERANS	0.0821	1	0.7744	0.0876	3	0.9933
WEALTH_RATING	0.1630	1	0.6864	0.2506	4	0.9928
mi_INCOME_GROUP	0.2911	1	0.5895	0.5417	5	0.9905
PCT_MALE_VETERANS	0.3511	1	0.5535	0.8928	6	0.9894
MOR_HIT_RATE	0.3982	1	0.5280	1.2909	7	0.9887
RECENT_AVG_CARD_GIFT_AMT	0.5050	1	0.4773	1.7959	8	0.9866
LIFETIME_GIFT_AMOUNT	0.5574	1	0.4553	2.3533	9	0.9846
LIFETIME_GIFT_RANGE	0.7367	1	0.3907	3.0900	10	0.9792
statusES	1.0369	1	0.3085	4.1269	11	0.9661
IN_HOUSE	1.2613	1	0.2614	5.3882	12	0.9437
PCT_VIETNAM_VETERANS	1.4606	1	0.2268	6.8488	13	0.9098
LIFETIME_MIN_GIFT_AMT	1.7580	1	0.1849	8.6068	14	0.8554
mi_DONOR_AGE	1.8035	1	0.1793	10.4104	15	0.7932
PUBLISHED_PHONE	2.1297	1	0.1445	12.5400	16	0.7060

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	21789.113	21140.627
SC	21796.984	21274.443
-2 Log L	21787.113	21106.627

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	680.4863	16	<.0001
Score	677.0495	16	<.0001
Wald	650.5041	16	<.0001

Residual Chi-Square Test

Chi-Square	DF	Pr > ChiSq
12.9763	16	0.6745

Summary of Backward Elimination

Step	Effect Removed	DF	Number In	Wald Chi-Square	Pr > ChiSq
1	PCT_OWNER_OCCUPIED	1	31	0.0018	0.9660
1	PCT_MALE_MILITARY	1	30	0.0037	0.9517
1	PCT_WWII_VETERANS	1	29	0.0821	0.7744
1	WEALTH_RATING	1	28	0.1630	0.6864
1	mi_INCOME_GROUP	1	27	0.2911	0.5895
1	PCT_MALE_VETERANS	1	26	0.3511	0.5535
1	MOR_HIT_RATE	1	25	0.3982	0.5280
1	RECENT_AVG_CARD_GIFT_AMT	1	24	0.5050	0.4773
1	LIFETIME_GIFT_AMOUNT	1	23	0.5574	0.4553
1	LIFETIME_GIFT_RANGE	1	22	0.7367	0.3907
1	statusES	1	21	1.0369	0.3085
1	IN_HOUSE	1	20	1.2613	0.2614
1	PCT_VIETNAM_VETERANS	1	19	1.4606	0.2268
1	LIFETIME_MIN_GIFT_AMT	1	18	1.7580	0.1849
1	mi_DONOR_AGE	1	17	1.8035	0.1793
1	PUBLISHED_PHONE	1	16	2.1297	0.1445

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.9952	0.1684	34.9119	<.0001
LIFETIME_CARD_PROM	1	0.00727	0.00292	6.1852	0.0129
PER_CAPITA_INCOME	1	4.181E-6	2.112E-6	3.9184	0.0478
CARD_PROM_12	1	-0.0468	0.0157	8.9016	0.0028
ClusCdGrp1	1	0.2024	0.0638	10.0736	0.0015
RECENT_RESPONSE_COUNT	1	0.0380	0.0124	9.3469	0.0022
ClusCdGrp3	1	-0.3301	0.0705	21.9487	<.0001
DONOR_AGE	1	0.00351	0.00121	8.3776	0.0038
ClusCdGrp4	1	-0.1798	0.0495	13.1945	0.0003
ClusCdGrp2	1	0.1083	0.0426	6.4461	0.0111
MONTHS_SINCE_LAST_GIFT	1	-0.0403	0.00450	80.2425	<.0001
RECENT_STAR_STATUS	1	-0.0215	0.00777	7.6674	0.0056
INCOME_GROUP	1	0.0547	0.0112	23.9968	<.0001
statusFL	1	-0.3387	0.0783	18.7267	<.0001
RECENT_AVG_GIFT_AMT	1	-0.00861	0.00209	16.9845	<.0001
PEP_STAR	1	0.2546	0.0471	29.1885	<.0001
RECENT_CARD_RESPONSE_PROP	1	0.7189	0.1247	33.2489	<.0001

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
LIFETIME_CARD_PROM	1.007	1.002	1.013
PER_CAPITA_INCOME	1.000	1.000	1.000
CARD_PROM_12	0.954	0.925	0.984
ClusCdGrp1	1.224	1.080	1.387
RECENT_RESPONSE_COUNT	1.039	1.014	1.064
ClusCdGrp3	0.719	0.626	0.825
DONOR_AGE	1.004	1.001	1.006
ClusCdGrp4	0.835	0.758	0.921
ClusCdGrp2	1.114	1.025	1.211
MONTHS_SINCE_LAST_GIFT	0.961	0.952	0.969
RECENT_STAR_STATUS	0.979	0.964	0.994
INCOME_GROUP	1.056	1.033	1.080
statusFL	0.713	0.611	0.831
RECENT_AVG_GIFT_AMT	0.991	0.987	0.995
PEP_STAR	1.290	1.176	1.415
RECENT_CARD_RESPONSE_PROP	2.052	1.607	2.620

Association of Predicted Probabilities and Observed Responses

Percent Concordant	61.8	Somers' D	0.242
Percent Discordant	37.6	Gamma	0.244
Percent Tied	0.7	Tau-a	0.091
Pairs	70363947	c	0.621

4. Compare variable selection methods.

- a. Fit a logistic regression model with the STEPWISE method. Use all of the numeric variables and the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Set **SLSTAY** and **SLENTY** equal to .001.

```
proc logistic data=pval des namelen=32;
  class URBANICITY SES HOME_OWNER;
  model target_b = &ex_screened URBANICITY
                  SES HOME_OWNER
                  /selection=stepwise slentry=.001 slstay=.001;
run;
```

Partial Output

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.1198	0.1367	67.0677	<.0001
RECENT_RESPONSE_COUNT	1	0.0377	0.0113	11.1164	0.0009
ClusCdGrp3	1	-0.4168	0.0671	38.5417	<.0001
DONOR_AGE	1	0.00413	0.00120	11.8532	0.0006
ClusCdGrp4	1	-0.2659	0.0448	35.2724	<.0001
MONTHS_SINCE_LAST_GIFT	1	-0.0369	0.00426	74.8419	<.0001
INCOME_GROUP	1	0.0623	0.0108	33.5689	<.0001
statusFL	1	-0.3569	0.0771	21.4193	<.0001
RECENT_AVG_GIFT_AMT	1	-0.00865	0.00205	17.7715	<.0001
PEP_STAR	1	0.2635	0.0389	45.9609	<.0001
RECENT_CARD_RESPONSE_PROP	1	0.7048	0.1142	38.0618	<.0001

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
RECENT_RESPONSE_COUNT	1.038	1.016	1.062
ClusCdGrp3	0.659	0.578	0.752
DONOR_AGE	1.004	1.002	1.006
ClusCdGrp4	0.767	0.702	0.837
MONTHS_SINCE_LAST_GIFT	0.964	0.956	0.972
INCOME_GROUP	1.064	1.042	1.087
statusFL	0.700	0.602	0.814
RECENT_AVG_GIFT_AMT	0.991	0.987	0.995
PEP_STAR	1.301	1.206	1.405
RECENT_CARD_RESPONSE_PROP	2.023	1.617	2.531

Association of Predicted Probabilities and Observed Responses

Percent Concordant	61.4	Somers' D	0.235
Percent Discordant	37.9	Gamma	0.237
Percent Tied	0.7	Tau-a	0.088
Pairs	70363947	c	0.617

- b. Fit a logistic regression model with the FAST BACKWARD method. Use all of the numeric variables and the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Set **SLSTAY** equal to .001.

```
proc logistic data=pval des namelen=32;
  class URBANICITY SES HOME_OWNER;
  model target_b = &ex_screened URBANICITY
                  SES HOME_OWNER
    /selection=backward FAST slstay=.001;
run;
```

Partial Output

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.1198	0.1367	67.0677	<.0001
RECENT_RESPONSE_COUNT	1	0.0377	0.0113	11.1164	0.0009
ClusCdGrp3	1	-0.4168	0.0671	38.5417	<.0001
DONOR_AGE	1	0.00413	0.00120	11.8532	0.0006
ClusCdGrp4	1	-0.2659	0.0448	35.2724	<.0001
MONTHS_SINCE_LAST_GIFT	1	-0.0369	0.00426	74.8419	<.0001
INCOME_GROUP	1	0.0623	0.0108	33.5689	<.0001
statusFL	1	-0.3569	0.0771	21.4193	<.0001
RECENT_AVG_GIFT_AMT	1	-0.00865	0.00205	17.7715	<.0001
PEP_STAR	1	0.2635	0.0389	45.9609	<.0001
RECENT_CARD_RESPONSE_PROP	1	0.7048	0.1142	38.0618	<.0001

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
RECENT_RESPONSE_COUNT	1.038	1.016	1.062
ClusCdGrp3	0.659	0.578	0.752
DONOR_AGE	1.004	1.002	1.006
ClusCdGrp4	0.767	0.702	0.837
MONTHS_SINCE_LAST_GIFT	0.964	0.956	0.972
INCOME_GROUP	1.064	1.042	1.087
statusFL	0.700	0.602	0.814
RECENT_AVG_GIFT_AMT	0.991	0.987	0.995
PEP_STAR	1.301	1.206	1.405
RECENT_CARD_RESPONSE_PROP	2.023	1.617	2.531

Association of Predicted Probabilities and Observed Responses

Percent Concordant	61.4	Somers' D	0.235
Percent Discordant	37.9	Gamma	0.237
Percent Tied	0.7	Tau-a	0.088
Pairs	70363947	c	0.617

- c. Fit a logistic regression model with the SCORE method and the BEST=1 option. Use all of the numeric variables and create dummies for the **URBANICITY**, **SES**, and **HOME_OWNER** categorical variables. Use the Output Delivery System to determine which model is the best according to the SBC criterion.

```
proc freq data=pval;
  tables (URBANICITY SES HOME_OWNER)*target_b;
run;
```

Table of URBANICITY by TARGET_B

URBANICITY	TARGET_B		
	0	1	Total
Frequency			
Percent			
Row Pct			
Col Pct			
?	314	140	454
	1.62	0.72	2.34
	69.16	30.84	
	2.16	2.89	
C	3016	1006	4022
	15.57	5.19	20.76
	74.99	25.01	
	20.76	20.77	
R	3078	927	4005
	15.89	4.79	20.67
	76.85	23.15	
	21.19	19.14	
S	3304	1187	4491
	17.06	6.13	23.18
	73.57	26.43	
	22.74	24.51	
T	2938	1006	3944
	15.17	5.19	20.36
	74.49	25.51	
	20.22	20.77	
U	1879	577	2456
	9.70	2.98	12.68
	76.51	23.49	
	12.93	11.91	
Total	14529	4843	19372
	75.00	25.00	100.00

Table of SES by TARGET_B

SES	TARGET_B		
Frequency			
Percent			
Row Pct			
Col Pct	0	1	Total
1	4341	1583	5924
	22.41	8.17	30.58
	73.28	26.72	
	29.88	32.69	
2	6966	2318	9284
	35.96	11.97	47.92
	75.03	24.97	
	47.95	47.86	
3	2590	733	3323
	13.37	3.78	17.15
	77.94	22.06	
	17.83	15.14	
4	318	69	387
	1.64	0.36	2.00
	82.17	17.83	
	2.19	1.42	
?	314	140	454
	1.62	0.72	2.34
	69.16	30.84	
	2.16	2.89	
Total	14529	4843	19372
	75.00	25.00	100.00

Table of HOME_OWNER by TARGET_B

HOME_OWNER	TARGET_B		
Frequency			
Percent			
Row Pct			
Col Pct	0	1	Total
H	7898 40.77 74.47 54.36	2708 13.98 25.53 55.92	10606 54.75
U	6631 34.23 75.64 45.64	2135 11.02 24.36 44.08	8766 45.25
Total	14529 75.00	4843 25.00	19372 100.00

```

data pva2;
  set pva1;
  home01=(HOME_OWNER="H");
  nses1=(SES="1");
  nses3=(SES="3");
  nses4=(SES="4");
  nses_=(SES="?");
  nurbr=(URBANICITY="R");
  nurbu=(URBANICITY="U");
  nurbs=(URBANICITY="S");
  nurbt=(URBANICITY="T");
  nurb_=(URBANICITY="?");
run;

ods listing close;
ods output NObs=NObs
          bestsubsets=score;
proc logistic data=pva2 des namelen=32;
  model target_b= &ex_screened
              nses1 nses3 nses4 nses_
              nurbr nurbu nurbs nurbt nurb_
              home01
          /selection=score best=1;
run;
ods listing;

data _NULL_;
  set NObs;
  where label = 'Number of Observations Used';
  call symput('obs',n);
run;

/* Calculate the SBC-like measure of model fit */
data subset;
  set score;
  sbc=-scorechisq+log(&obs)*(numberofvariables+1);
run;

proc sql;
  select VariablesInModel into :ex_selected
  from subset
  having sbc=min(sbc);
quit;

```

Variables Included in Model

RECENT_RESPONSE_COUNT ClusCdGrp3 DONOR_AGE ClusCdGrp4 MONTHS_SINCE_LAST_GIFT INCOME_GROUP
statusFL RECENT_AVG_GIFT_AMT PEP_STAR RECENT_CARD_RESPONSE_PROP

```
proc logistic data=pva2 des namelen=32;
  model target_b = &ex_selected;
run;
```

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.1198	0.1367	67.0677	<.0001
RECENT_RESPONSE_COUNT	1	0.0377	0.0113	11.1164	0.0009
ClusCdGrp3	1	-0.4168	0.0671	38.5417	<.0001
DONOR_AGE	1	0.00413	0.00120	11.8532	0.0006
ClusCdGrp4	1	-0.2659	0.0448	35.2724	<.0001
MONTHS_SINCE_LAST_GIFT	1	-0.0369	0.00426	74.8419	<.0001
INCOME_GROUP	1	0.0623	0.0108	33.5689	<.0001
statusFL	1	-0.3569	0.0771	21.4193	<.0001
RECENT_AVG_GIFT_AMT	1	-0.00865	0.00205	17.7715	<.0001
PEP_STAR	1	0.2635	0.0389	45.9609	<.0001
RECENT_CARD_RESPONSE_PROP	1	0.7048	0.1142	38.0618	<.0001

Odds Ratio Estimates

Effect	Point Estimate	95% Wald Confidence Limits	
RECENT_RESPONSE_COUNT	1.038	1.016	1.062
ClusCdGrp3	0.659	0.578	0.752
DONOR_AGE	1.004	1.002	1.006
ClusCdGrp4	0.767	0.702	0.837
MONTHS_SINCE_LAST_GIFT	0.964	0.956	0.972
INCOME_GROUP	1.064	1.042	1.087
statusFL	0.700	0.602	0.814
RECENT_AVG_GIFT_AMT	0.991	0.987	0.995
PEP_STAR	1.301	1.206	1.405
RECENT_CARD_RESPONSE_PROP	2.023	1.617	2.531

Association of Predicted Probabilities and Observed Responses

Percent Concordant	61.4	Somers' D	0.235
Percent Discordant	37.9	Gamma	0.237
Percent Tied	0.7	Tau-a	0.088
Pairs	70363947	c	0.617

d. Fit a stepwise linear regression model using the REG procedure:

```
proc reg data=pva2;
  model target_b = &ex_screened
                 nses1 nses3 nses4 nses_
                 nurbr nurbu nurbs nurbt nurb_
                 home01
    / selection=stepwise slstay=.001 slentry=.001;
  output out=out1 p=p;
run; quit;
```

Partial Output

```

                                The REG Procedure
                                Model: MODEL1
Dependent Variable: TARGET_B Target Variable Indicates for Response to 97NK Mailing

                                Number of Observations Read      19372
                                Number of Observations Used      19372

                                Stepwise Selection: Step 1

Variable RECENT_RESPONSE_COUNT Entered: R-Square = 0.0166 and C(p) = 349.3408

                                Analysis of Variance

Source              DF          Sum of Squares          Mean Square          F Value          Pr > F
Model                1             60.22121             60.22121             326.56          <.0001
Error              19370          3572.02879             0.18441
Corrected Total    19371          3632.25000

Variable              Parameter Estimate      Standard Error      Type II SS      F Value      Pr > F
Intercept              0.16709          0.00553          168.40627      913.21      <.0001
RECENT_RESPONSE_COUNT  0.02725          0.00151          60.22121      326.56      <.0001

                                Bounds on condition number: 1, 1
-----
```

Stepwise Selection: Step 2

Variable MONTHS_SINCE_LAST_GIFT Entered: R-Square = 0.0205 and C(p) = 272.4543

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	74.51243	37.25621	202.83	<.0001
Error	19369	3557.73757	0.18368		
Corrected Total	19371	3632.25000			

The REG Procedure

Model: MODEL1

Dependent Variable: TARGET_B Target Variable Indicates for Response to 97NK Mailing

Stepwise Selection: Step 2

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	0.30200	0.01626	63.36215	344.96	<.0001
RECENT_RESPONSE_COUNT	0.02421	0.00154	45.18640	246.00	<.0001
MONTHS_SINCE_LAST_GIFT	-0.00691	0.00078324	14.29122	77.80	<.0001

Bounds on condition number: 1.0523, 4.2092

Stepwise Selection: Step 3

Variable PEP_STAR Entered: R-Square = 0.0233 and C(p) = 219.1810

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	84.52585	28.17528	153.82	<.0001
Error	19368	3547.72415	0.18317		
Corrected Total	19371	3632.25000			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	0.29189	0.01630	58.77461	320.87	<.0001
RECENT_RESPONSE_COUNT	0.01867	0.00171	21.71513	118.55	<.0001
MONTHS_SINCE_LAST_GIFT	-0.00684	0.00078222	13.99109	76.38	<.0001
PEP_STAR	0.05088	0.00688	10.01342	54.67	<.0001

Bounds on condition number: 1.3015, 10.818

Partial Output

All variables left in the model are significant at the 0.0010 level.

No other variable met the 0.0010 significance level for entry into the model.

Summary of Stepwise Selection

Variable Step Entered	Variable Removed
1 RECENT_RESPONSE_COUNT	
2 MONTHS_SINCE_LAST_GIFT	
3 PEP_STAR	
4 PER_CAPITA_INCOME	
5 RECENT_CARD_RESPONSE_PROP	
6 INCOME_GROUP	
7 statusFL	
8 ClusCdGrp3	
9 ClusCdGrp4	
10	PER_CAPITA_INCOME
11 RECENT_AVG_GIFT_AMT	
12 DONOR_AGE	

```
proc means data=out1 min mean max;
  var p;
run;
```

Analysis Variable : p Predicted Value of TARGET_B

Minimum	Mean	Maximum
-0.1063936	0.2500000	0.5804033

- e. How does the c statistic computed from the model generated in PROC REG compare with the c statistics from the models generated in PROC LOGISTIC? What reasons might one have for choosing PROC LOGISTIC over PROC REG as a modeling tool? What reasons might one have for choosing PROC REG over PROC LOGISTIC as a modeling tool?

```
proc logistic data=out1;
  model target_b=p;
run;
```

Partial Output

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	61.4	Somers' D	0.235
Percent Discordant	37.9	Gamma	0.236
Percent Tied	0.7	Tau-a	0.088
Pairs	70363947	c	0.617

A possible reason to consider PROC REG as a modeling tool is its speed. A reason to consider PROC LOGISTIC is the fact that REG may yield negative predicted probabilities or probabilities greater than one.

Chapter 4 Solutions

1. Assess some logistic regression models using a validation data set.
 - a. Split the imputed data set into training and validation data sets. Use 50% of the data for each data set role. Stratify on the target variable.

```
proc sort data=pva2 out=pva2;
  by target_b;
run;

proc surveyselect noprint
  data=pva2
  samprate=.5
  out=pva2
  seed=27513
  outall;
  strata target_b;
run;

data pva_train pva_valid;
  set pva2;
  if selected then output pva_train;
  else output pva_valid;
run;
```


- b. (Optional) Use the Spearman and Hoeffding correlation coefficients to screen the inputs with the least evidence of a relationship with the target. If necessary, consider transforming some inputs that do well according to the Hoeffding correlation measure.

```
ods listing close;
ods output spearmancorr=spearman
           hoeffdingcorr=hoeffding;

proc corr data=pva_train spearman hoeffding rank;
  var &ex_screened
      nses1 nses3 nses4 nses_
      nurbr nurbu nurbs nurbt nurb_
      home01;
  with target_b;
run;

ods listing;

%let nvar = 42;

data spearman1(keep=variable scorr spvalue ranksp);
  length variable $ 32;
  set spearman;
  array best(*) best1--best&nvar;
  array r(*) r1--r&nvar;
  array p(*) p1--p&nvar;
  do i=1 to dim(best);
    variable=best(i);
    scorr=r(i);
    spvalue=p(i);
    ranksp=i;
    output;
  end;
run;

data hoeffding1(keep=variable hcorr hpvalue rankho);
  length variable $ 32;
  set hoeffding;
  array best(*) best1--best&nvar;
  array r(*) r1--r&nvar;
  array p(*) p1--p&nvar;
  do i=1 to dim(best);
    variable=best(i);
    hcorr=r(i);
    hpvalue=p(i);
    rankho=i;
    output;
  end;
run;
```

```

proc sort data=spearman1;
  by variable;
run;

proc sort data=hoeffding1;
  by variable;
run;

data correlations;
  merge spearman1 hoeffding1;
  by variable;
run;

proc sort data=correlations;
  by ranksp;
run;

proc print data=correlations label split='*';
  var variable ranksp rankho;
  label ranksp = 'Spearman rank*of variables'
        rankho = 'Hoeffding rank*of variables';
run;

```

Obs	variable	Spearman rank of variables	Hoeffding rank of variables
1	RECENT_RESPONSE_COUNT	1	1
2	RECENT_AVG_GIFT_AMT	2	2
3	PEP_STAR	3	6
4	RECENT_CARD_RESPONSE_PROP	4	3
5	LIFETIME_MIN_GIFT_AMT	5	4
6	statusES	6	7
7	MONTHS_SINCE_LAST_GIFT	7	5
8	RECENT_STAR_STATUS	8	12
9	LIFETIME_CARD_PROM	9	8
10	LIFETIME_GIFT_AMOUNT	10	9
11	PER_CAPITA_INCOME	11	10
12	statusFL	12	21
13	ClusCdGrp4	13	17
14	ClusCdGrp3	14	25
15	IN_HOUSE	15	31
16	DONOR_AGE	16	13
17	CARD_PROM_12	17	15
18	nses3	18	20
19	nurbr	19	19
20	INCOME_GROUP	20	16
21	nurb_	21	40
22	nses_	22	41
23	nses4	23	42
24	MOR_HIT_RATE	24	18
25	ClusCdGrp1	25	36
26	ClusCdGrp2	26	26
27	nses1	27	23
28	mi_DONOR_AGE	28	28
29	LIFETIME_GIFT_RANGE	29	14

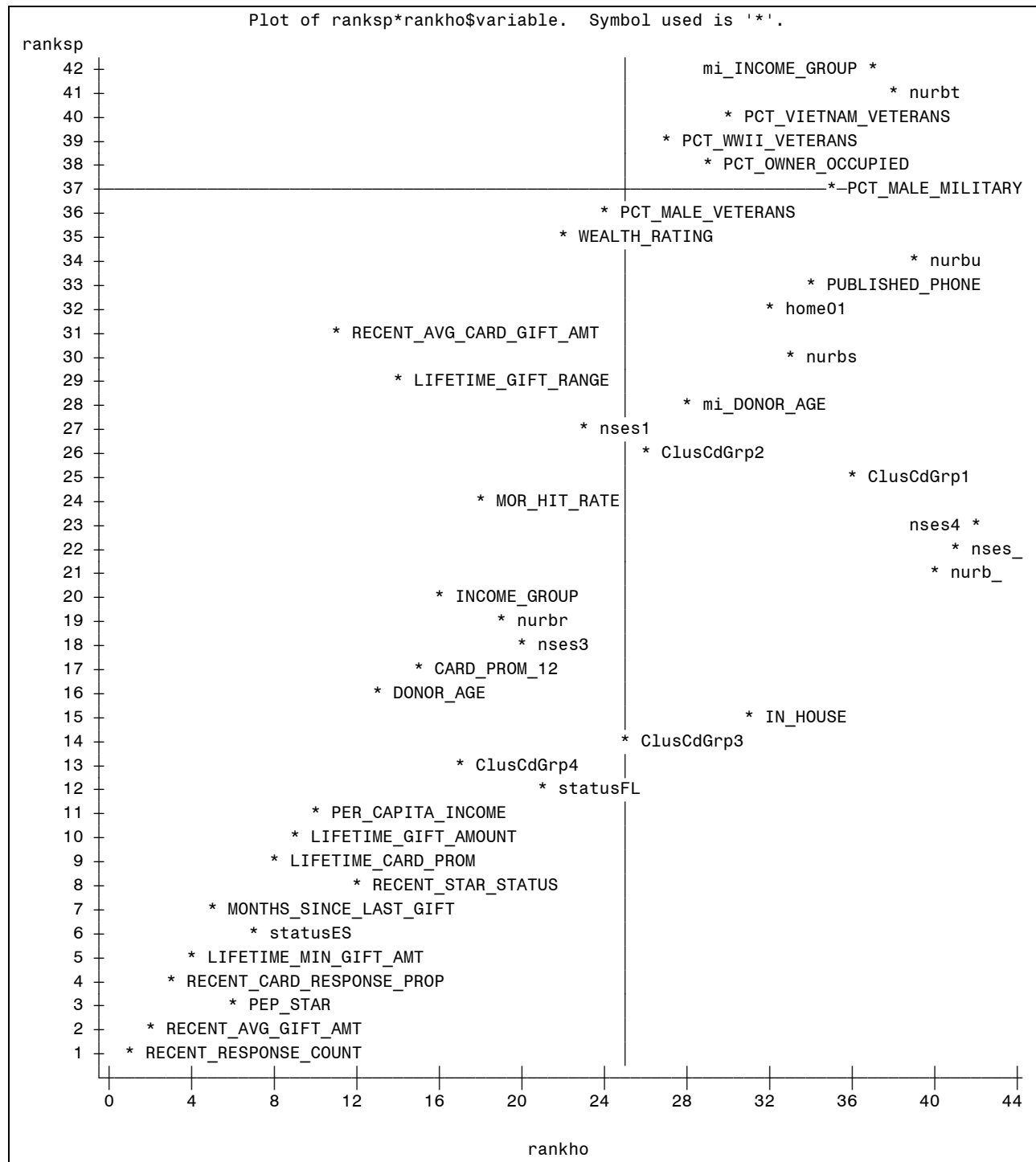
30	nurbs	30	33
31	RECENT_AVG_CARD_GIFT_AMT	31	11
32	home01	32	32
33	PUBLISHED_PHONE	33	34
34	nurbu	34	39
35	WEALTH_RATING	35	22
36	PCT_MALE_VETERANS	36	24
37	PCT_MALE_MILITARY	37	35
38	PCT_OWNER_OCCUPIED	38	29
39	PCT_WWII_VETERANS	39	27
40	PCT_VIETNAM_VETERANS	40	30
41	nurbt	41	38
42	mi_INCOME_GROUP	42	37

```

proc sql noprint;
  select min(ranksp) into :vref
  from (select ranksp
        from correlations
        having spvalue > .5);
  select min(rankho) into :href
  from (select rankho
        from correlations
        having hpvalue > .5);
run; quit;

proc plot data = correlations;
  plot ranksp*rankho $ variable="*"
    /vref=&vref href=&href vpos=&nvar;
run; quit;

```



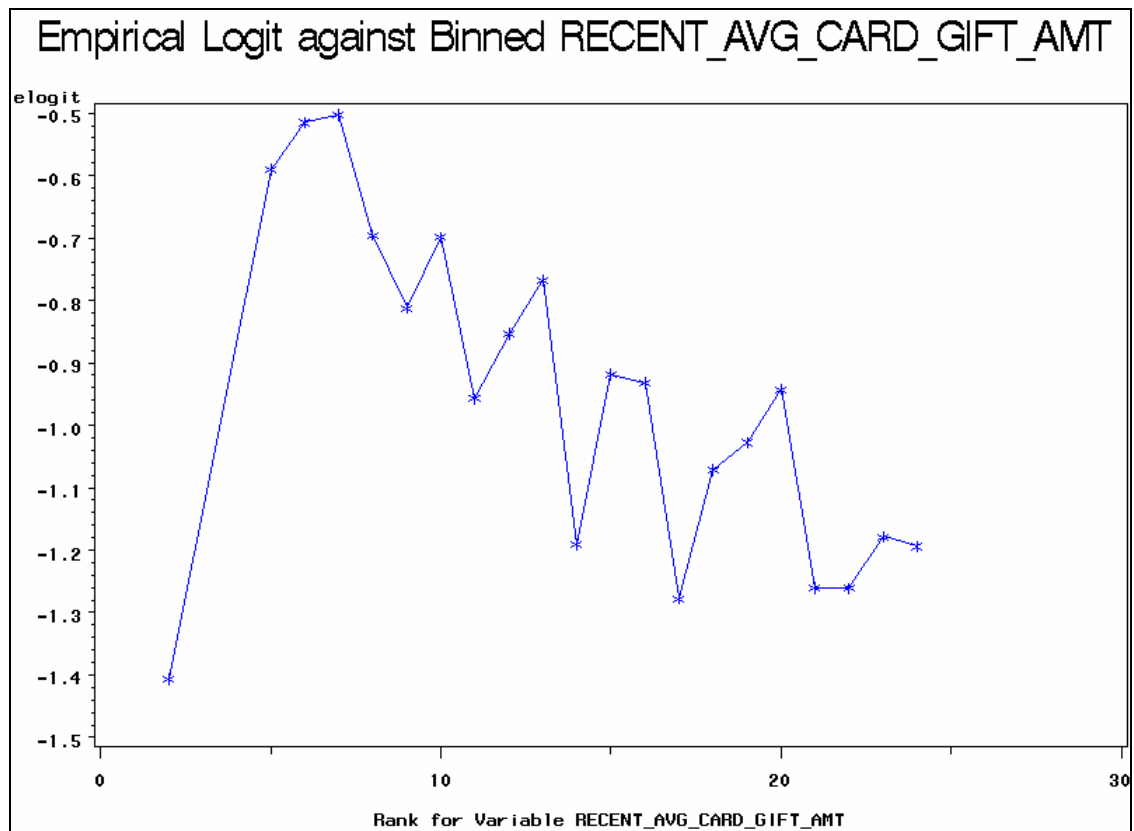
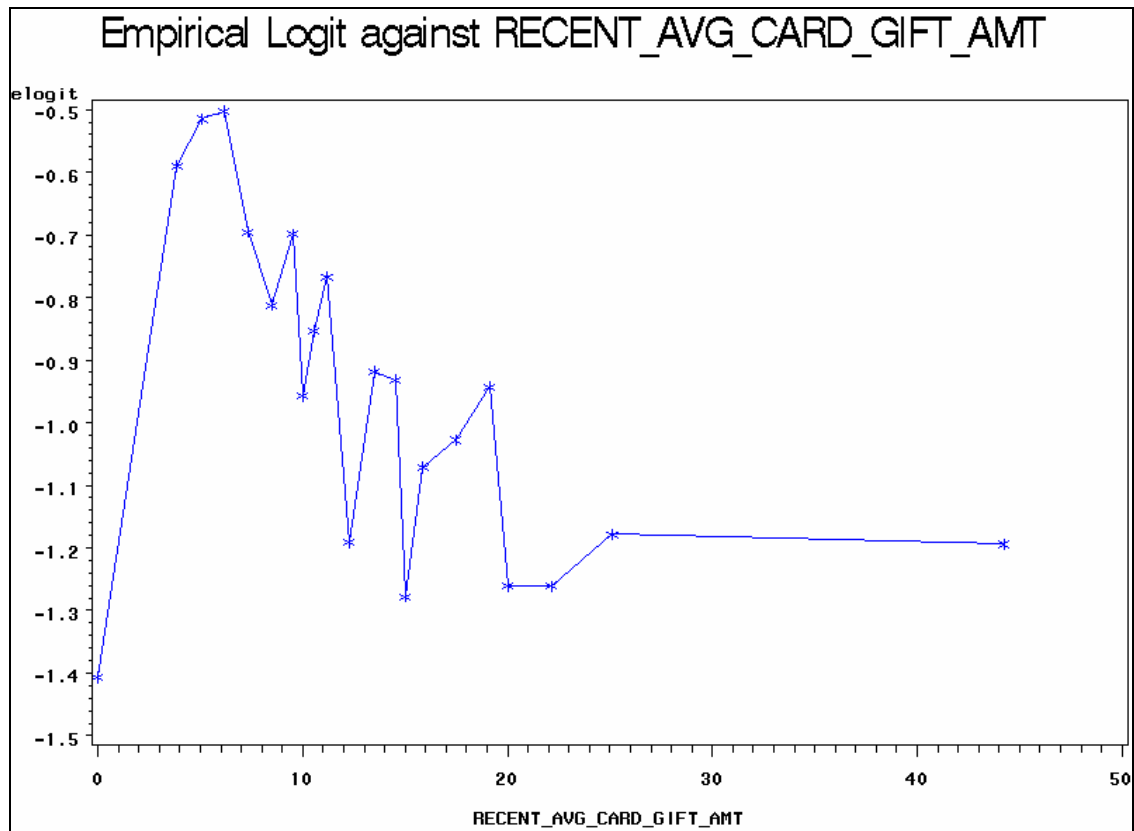
- 1) It seems like potential nonlinear inputs include **RECENT_AVG_CARD_GIFT_AMT** and **LIFETIME_GIFT_RANGE**.

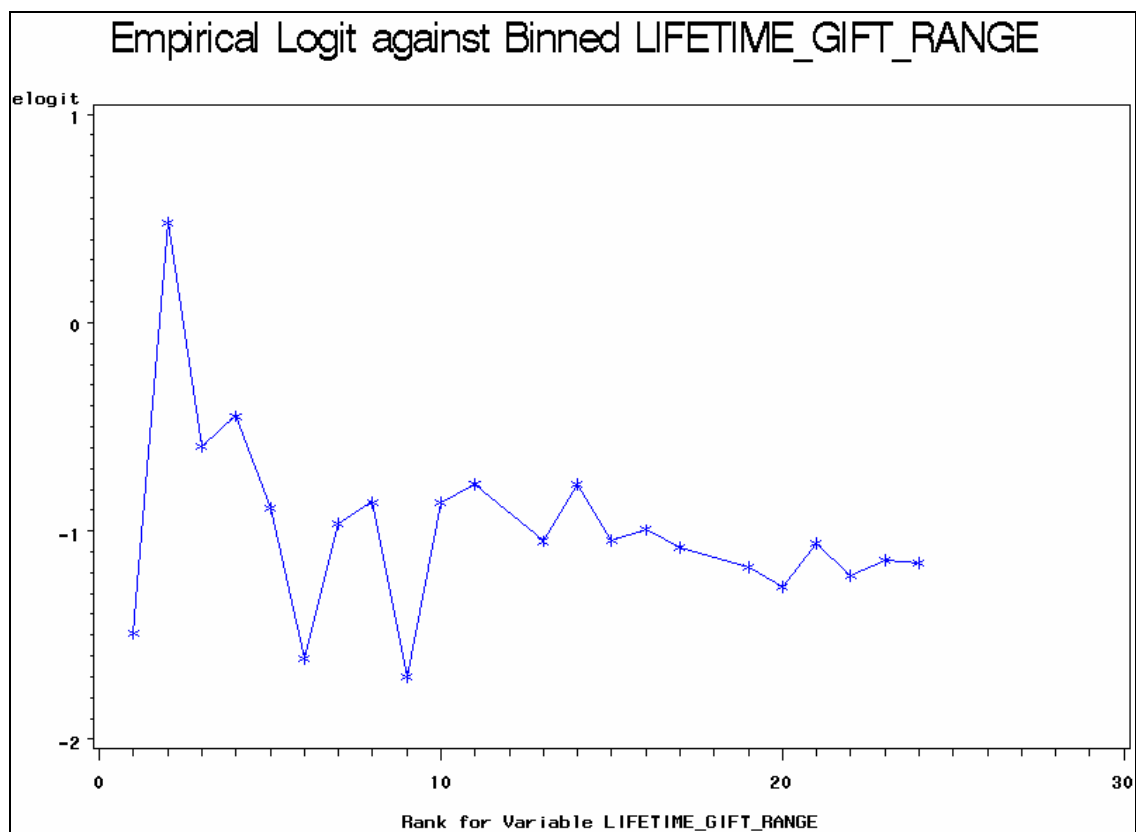
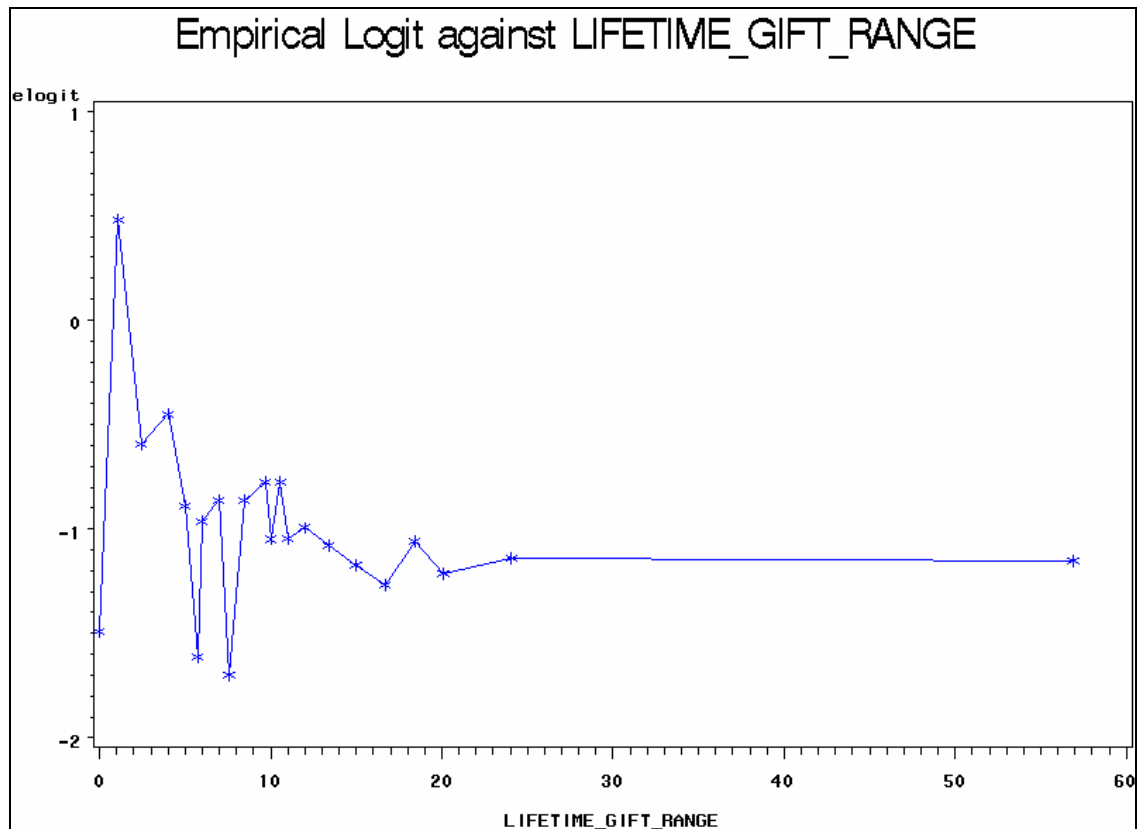
```
%let var=RECENT_AVG_CARD_GIFT_AMT;
*****
%let var = LIFETIME_GIFT_RANGE;
proc rank data=pva_train groups=25 out=out;
    var &var;
    ranks bin;
run;

proc means data=out noprint nway;
    class bin;
    var target_b &var;
    output out=bins sum(target_b)=target_b
        mean(&var)=&var;
run;

data bins;
    set bins;
    elogit=log((target_b+(sqrt(_FREQ_)/2))/
        (_FREQ_-target_b+(sqrt(_FREQ_)/2)));
run;

goptions reset=all;
symbol i=join c=blue v=star;
proc gplot data = bins;
    title "Empirical Logit against &var";
    plot elogit * &var;
run;
    title "Empirical Logit against Binned &var";
    plot elogit * bin;
run; quit;
title;
```





- 2) You can consider different inputs to account for the nonlinear relationships among these inputs. Try both quadratic terms.

Chapter 5 Solutions

1. Assess some logistic regression models using a validation data set.
 - a. Using the macros outlined in Chapter 5, score and evaluate the performance of the best subsets models on the training and validation data sets.

```
ods listing close;
ods output bestsubsets=score;
proc logistic data=pva_train des;
  model target_b = &ex_screened
                 nses1 nses3 nses4 nses_
                 nurbr nurbu nurbs nurbt nurb_
                 home01
                 RECENT_AVG_CARD_GIFT_AMT*RECENT_AVG_CARD_GIFT_AMT
                 LIFETIME_GIFT_RANGE*LIFETIME_GIFT_RANGE
  /selection=score best=1;
run;
ods listing;
```



If you elected to skip 6.b. then you can ignore the two squared terms in the above PROC LOGISTIC call.

```
proc sql noprint;
  select variablesinmodel into :inputs1 - :inputs99999
  from score;
  select NumberOfVariables into :ic1 - :ic99999
  from score;
quit;
%let lastindx = &SQLOBS;

* Save the observed response rate, for the weighted sums of profit
etc.;
proc sql;
  select mean(target_b) into :ex_rho1
  from pva;
quit;

%let ex_rho0 = %sysevalf(1-&ex_rho1);
%let ex_pi0 = %sysevalf(1-&ex_pi1);

%let ex_pf11 = 14.94; %let ex_pf10 = 0;
%let ex_pf01 = -0.68; %let ex_pf00 = 0;
```



The macro variables defined above make the comparisons of literally hundreds of models easier.


```

data train2;
  set pva_train;
  RECENT_AVG_CARD_GIFT_AMTRECENT_A=
    RECENT_AVG_CARD_GIFT_AMT*RECENT_AVG_CARD_GIFT_AMT;
  LIFETIME_GIFT_RANGELIFETIME_GIFT=
    LIFETIME_GIFT_RANGE*LIFETIME_GIFT_RANGE;
run;

data valid2;
  set pva_valid;
  RECENT_AVG_CARD_GIFT_AMTRECENT_A=
    RECENT_AVG_CARD_GIFT_AMT*RECENT_AVG_CARD_GIFT_AMT;
  LIFETIME_GIFT_RANGELIFETIME_GIFT=
    LIFETIME_GIFT_RANGE*LIFETIME_GIFT_RANGE;
run;

```



Any new inputs need to be created on the training and validation data sets. These variables are named according to the rules used to refer to polynomial terms in the best subsets output.

```

%macro ex_assess(data=,inputcount=,inputsinmodel=,index=) ;
/* sort data set from likely to unlikely to respond */
proc sort data=scored&data;
  by descending p_1;
run;

```

```

/* create assessment data set */
data assess;
  attrib DATAROLE length=$5;
  retain sse 0 csum 0 DATAROLE "&data";

  /* 2 x 2 count array, or count matrix */
  array n[0:1,0:1] _temporary_ (0 0 0 0);
  /* sample weights array */
  array w[0:1] _temporary_
    (%sysevalf(&ex_pi0/&ex_rho0) %sysevalf(&ex_pi1/&ex_rho1));
  keep DATAROLE INPUT_COUNT INDEX
    TOTAL_PROFIT OVERALL_AVG_PROFIT ASE C;

  set scored&data end=last;
  /* profit associated with each decision */
  d1=&ex_PF11*p_1+&ex_PF01*p_0;
  d0=&ex_PF10*p_1+&ex_PF00*p_0;

  /* T is a flag for response */
  t=(strip(target_b)="1");
  /* D is the decision, based on profit. */
  d=(d1>d0);

  /* update the count matrix, sse, and c */
  n[t,d] + w[t];
  sse + (target_b-p_1)**2;
  csum + ((n[1,1]+n[1,0])*(1-t)*w[0]);

  if last then do;
    INPUT_COUNT=&inputcount;
    TOTAL_PROFIT =
sum(&ex_PF11*n[1,1],&ex_PF10*n[1,0],&ex_PF01*n[0,1],&ex_PF00*n[0,0]);
    OVERALL_AVG_PROFIT =
TOTAL_PROFIT/sum(n[0,0],n[1,0],n[0,1],n[1,1]);
    ASE = sse/sum(n[0,0],n[1,0],n[0,1],n[1,1]);
    C = csum/(sum(n[0,0],n[0,1])*sum(n[1,0],n[1,1]));
    index=&index;
    output;
  end;
run;

proc append base=results data=assess force;
run;

```

```

%mend ex_assess;

%macro ex_fitandscore();
proc datasets
    library=work
    nodetails
    nolist;
    delete results;
run;

%do model_indx=1 %to &lastindx;

%let im=&&inputs&model_indx;
%let ic=&&ic&model_indx;

proc logistic data=train2 des;
    model target_b=&im;
    score data=train2
        out=scoredtrain(keep=target_b p_1 p_0)
        priorevent=&ex_pil;
    score data=valid2
        out=scoredvalid(keep=target_b p_1 p_0)
        priorevent=&ex_pil;
run;
%ex_assess(data=TRAIN,
    inputcount=&ic,
    inputsinmodel=&im,
    index=&model_indx);
%ex_assess(data=VALID,
    inputcount=&ic,
    inputsinmodel=&im,
    index=&model_indx);
%end;
%mend ex_fitandscore;

```



The EX_ASSESS and EX_FITANDSCORE macros are updated versions of the model fitting and assessing macros discussed in Chapter 5.

```

%ex_fitandscore;

proc print data = results(obs=10);
run;

```

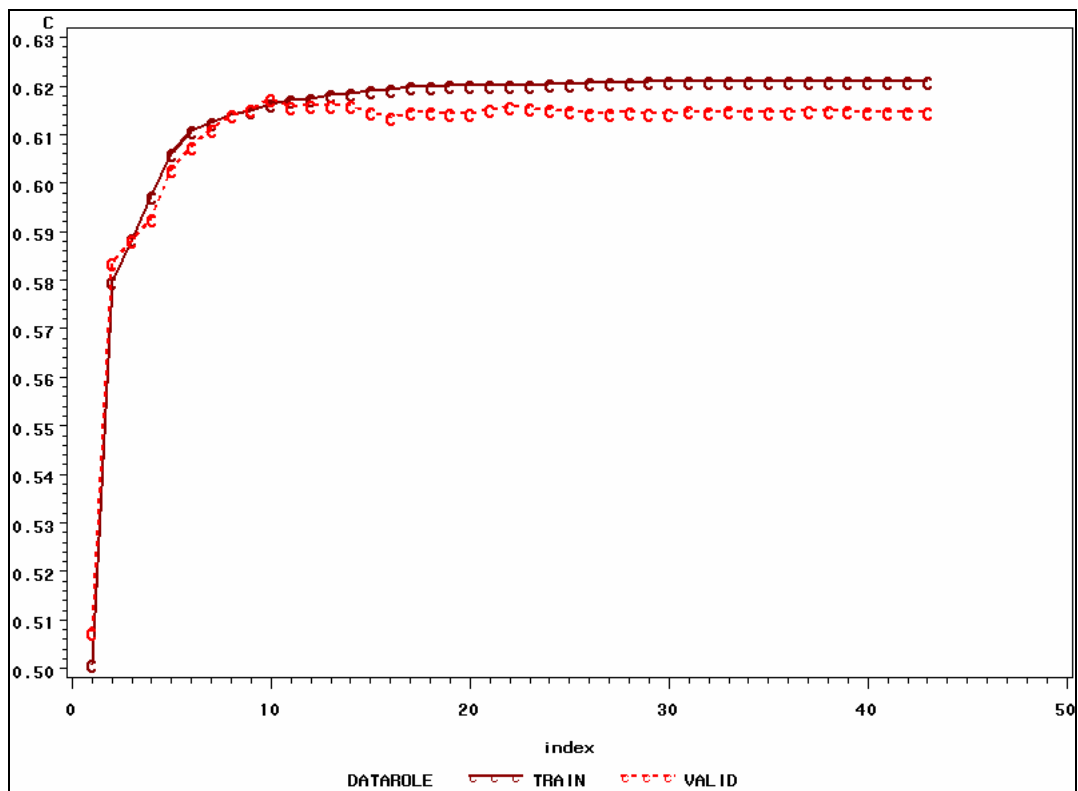
Obs	DATAROLE	INPUT_ COUNT	TOTAL_ PROFIT	OVERALL_ AVG_PROFIT	ASE	C	index
1	TRAIN	1	1348.75	0.13924	0.22577	0.50089	1
2	VALID	1	1462.51	0.15100	0.22585	0.50756	1
3	TRAIN	2	1372.78	0.14172	0.22512	0.57980	2
4	VALID	2	1423.12	0.14694	0.22531	0.58346	2
5	TRAIN	3	1409.51	0.14551	0.22485	0.58847	3
6	VALID	3	1380.63	0.14255	0.22513	0.58840	3
7	TRAIN	4	1474.30	0.15220	0.22455	0.59723	4
8	VALID	4	1404.31	0.14499	0.22489	0.59279	4
9	TRAIN	5	1566.99	0.16177	0.22427	0.60610	5
10	VALID	5	1633.05	0.16861	0.22455	0.60276	5

- b. Use graphical summaries to determine what models are of interest. If these are inconclusive, what measures will you take to choose the best model? You might consider lift charts or ROC curves.

```

symbol1 i=join f=arial v=c h=1.5 c=darkred l=1;
symbol2 i=join f=arial v=c h=1.5 c=red l=2;
proc gplot data = results;
  plot C*Index=datarole;
run; quit;

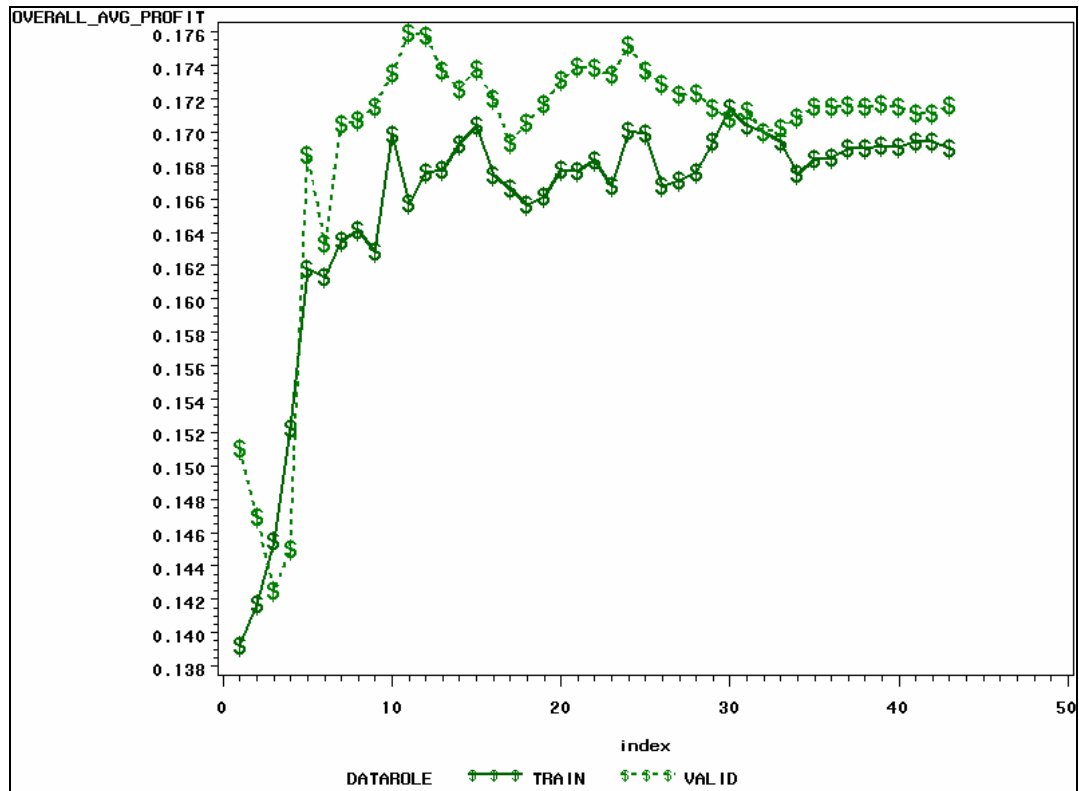
```



```

symbol1 i=join f=arial v=$ h=1.5 c=darkgreen w=2 l=1;
symbol2 i=join f=arial v=$ h=1.5 c=green w=2 l=2;
proc gplot data = results;
    plot OVERALL_AVG_PROFIT*Index=datarole;
run; quit;

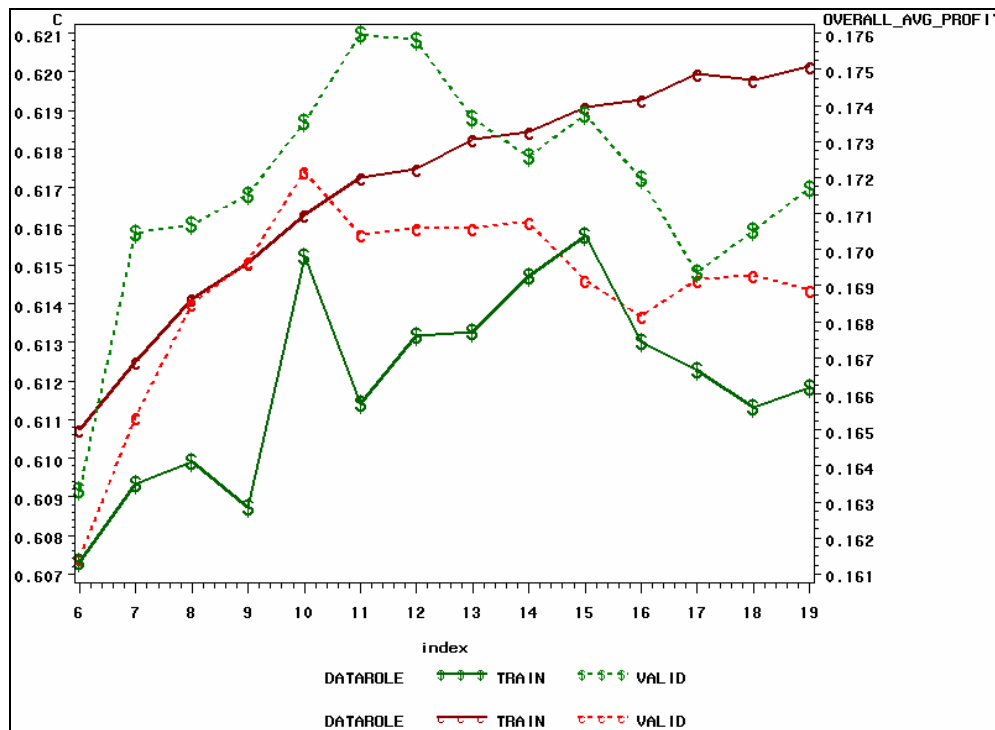
```



```

symbol1 i=join f=arial v=c h=1.5 c=darkred l=1;
symbol2 i=join f=arial v=c h=1.5 c=red l=2;
symbol3 i=join f=arial v=$ h=1.5 c=darkgreen w=2 l=1;
symbol4 i=join f=arial v=$ h=1.5 c=green w=2 l=2;
proc gplot data = results;
  where 5< index < 20;
  plot C*Index=datarole;
  plot2 OVERALL_AVG_PROFIT*Index=datarole;
run; quit;

```



- 1) On validation profit, the model with index 11 seems superior. By the c statistic, model 10 seems best. You could compare these using the ROC curve or lift charts to see if they seem to perform differently.

```

proc logistic data=train2 des namelen=32 noprint;
  model target_b=&inputs10;
  score data=valid2 out=scoval10 outroc=outroc10 priorevent=&ex_pil;
run;

proc logistic data=train2 des namelen=32 noprint;
  model target_b=&inputs11;
  score data=valid2 out=scoval11 outroc=outroc11 priorevent=&ex_pil;
run;

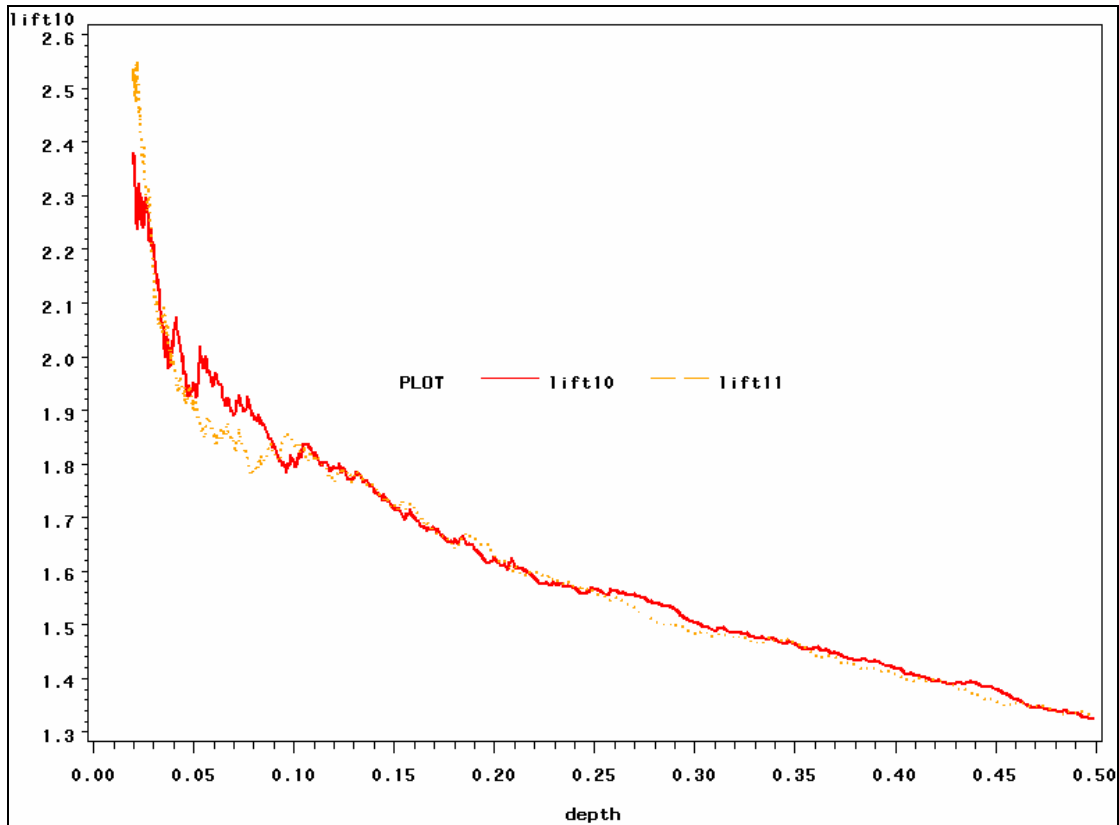
```

```
data outroc10;
  set outroc10;
  cutoff=_PROB_;
  specif10=1-_1MSPEC_;
  sensit10=_SENSIT_;
  tp10=&ex_pi1*_SENSIT_;
  fn10=&ex_pi1*(1-_SENSIT_);
  tn10=(1-&ex_pi1)*specif10;
  fp10=(1-&ex_pi1)*_1MSPEC_;
  depth=tp10+fp10;
  pospv10=tp10/depth;
  negpv10=tn10/(1-depth);
  acc10=tp10+tn10;
  lift10=pospv10/&ex_pi1;
run;

data outroc11;
  set outroc11;
  cutoff=_PROB_;
  specif11=1-_1MSPEC_;
  sensit11=_SENSIT_;
  tp11=&ex_pi1*_SENSIT_;
  fn11=&ex_pi1*(1-_SENSIT_);
  tn11=(1-&ex_pi1)*specif11;
  fp11=(1-&ex_pi1)*_1MSPEC_;
  depth=tp11+fp11;
  pospv11=tp11/depth;
  negpv11=tn11/(1-depth);
  acc11=tp11+tn11;
  lift11=pospv11/&ex_pi1;
run;

data ex_roc;
  merge outroc10 outroc11;
  by descending _PROB_;
run;
```

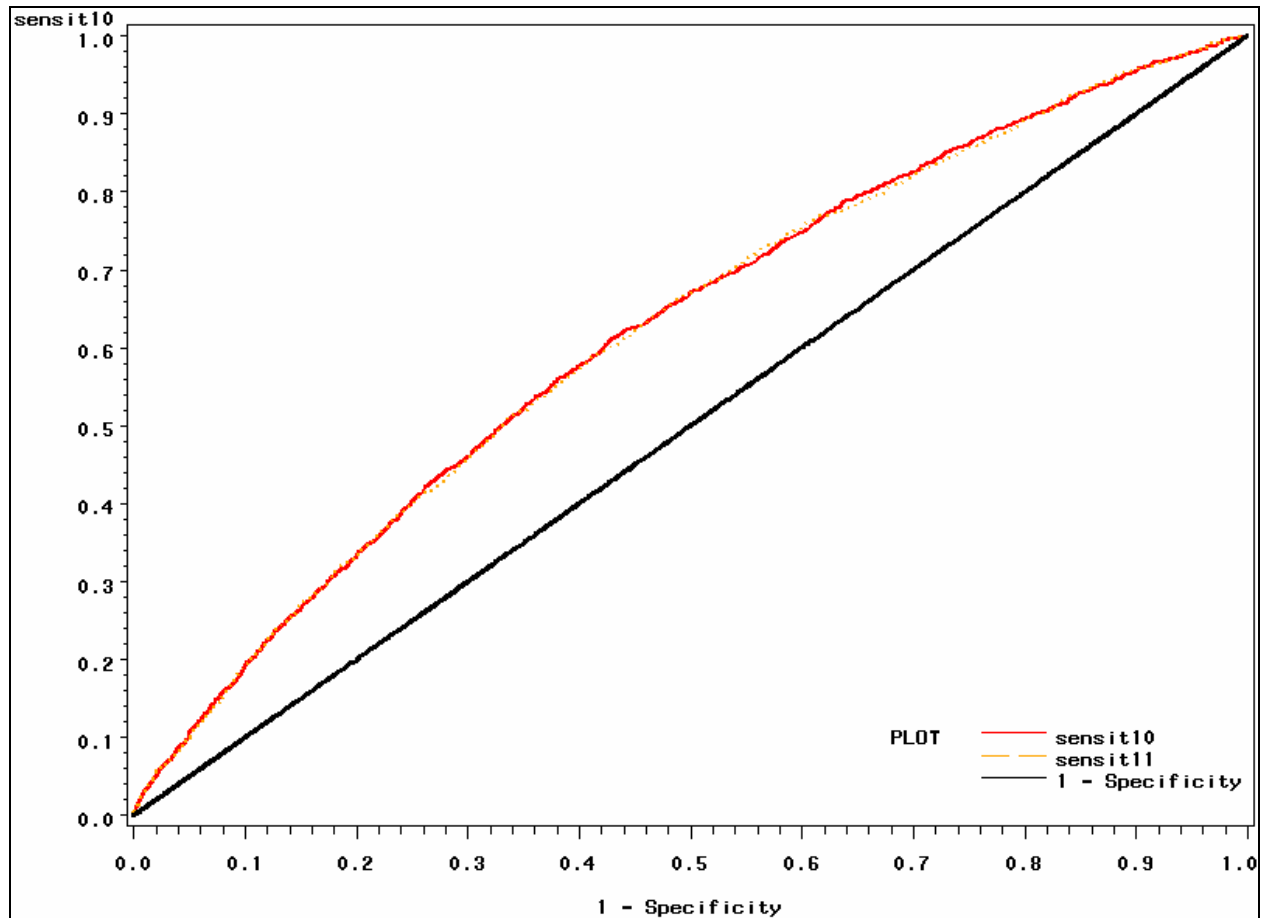
```
symbol i = join v=none color=red w=2 l=1;  
symbol2 i = join v=none color=orange w=2 l=2;  
legend position=(middle center inside);  
proc gplot data=ex_roc;  
  where .02 < depth < .5;  
  plot (lift10 lift11)*depth / overlay legend=legend;  
run; quit;
```




```

symbol3 i=join color=black w=3 v=none l=1;
legend position=(bottom right inside) down=4;
proc gplot data = ex_roc;
  plot (sensit10 sensit11 _1MSPEC_)*_1MSPEC_ / overlay legend=legend;
run; quit;

```



Because it seems like there is little difference in performance, and because a 10-input model will be easier to use than an 11-input model, you might consider using the 10-input model.

2. (Optional) Prepare scoring code for your selected model. Accommodate any specialized input that you created; for example, clustered levels of `CLUSTER_CODE`.

```
ods output parameterEstimates = betas2;
proc logistic data=train2 des namelen=32;
  model target_b=&inputs10;
run;

%let target=TARGET_B;

filename scored 'c:\temp\logistic score code.sas';

data _null_;
  attrib PREDNAME length=$32
         TARGNAME length=$32
         LastParm length=$32
         ;
  file scored;
  set betas2 end=last;
  retain TARGNAME PREDNAME LastParm ' ';
  if (Variable="Intercept") then do;
    TARGNAME=compress("&target");
    PREDNAME="P_"||compress(TARGNAME);
    put "*****";
    put "**** begin scoring code for Logistic Regression;";
    put "*****";

    put "length " PREDNAME "8;";
    put "label " PREDNAME "= 'Predicted: " TARGNAME +(-1) "'";

    put "**** accumulate XBETA;";
    Estimate + (-log(((1-&ex_pil)*&ex_rho1)/(&ex_pil*(1-&ex_rho1)))));
    put "XBETA = " Estimate best20. ";";
  end;
  else if (ClassVal0=' ') then do;
    put "XBETA = XBETA + (" Estimate best20. ") * " Variable ";";
  end;
  else if (compress(Variable)=compress(LastParm)) then do;
    put "else if (" Variable "=" ClassVal0 +(-1) "' ) then do;";
    put "  XBETA = XBETA + (" Estimate best20. ");";
    put "end;";
  end;
  else do;
    put "if (" Variable "=" ClassVal0 +(-1) "' ) then do;";
    put "  XBETA = XBETA + (" Estimate best20. ");";
    put "end;";
  end;
  LastParm=Variable;
  if last then do;
    put PREDNAME "= 1/(1+exp(-XBETA));";
  end;
run;
```

```
data your_scored_data_set;  
  set your_data_set to be_scored;  
  if income_group = . then income_group = 4;  
  ClusCdGrp3 = CLUSTER_CODE in("06", "10", "32", "41", "44", "47");  
  ClusCdGrp4 = CLUSTER_CODE in("09", "43", "49", "51",  
                                "21", "30", "45", "52",  
                                "08", "37", "50");  
  statusFL=RECENCY_STATUS_96NK in("F","L");  
  %include scorecd;  
run;
```

The missing piece is the **DONOR_AGE** imputation, which requires the DATA step to assign individuals to the appropriate group based on recent response activity and recent donation amounts.

Appendix B Additional Resources

B.1	Sampling Weights.....	B-2
B.2	References	B-6

B.1 Sampling Weights

$$weight_i = \begin{cases} \frac{\pi_1}{\rho_1} & \text{if } y_i = 1 \\ \frac{\pi_0}{\rho_0} & \text{if } y_i = 0 \end{cases}$$

Another method for adjusting for oversampling is to incorporate sampling weights. Sampling weights adjust the data so that it better represents the true population. When a rare target event has been oversampled, class 0 is under-represented in the sample. Consequently, a class-0 case should actually count more in the analysis than a class-1 case. The predicted values will be properly corrected by using weights that are inversely proportional to selection probabilities (for each class, the number of cases in the sample divided by the number of cases in the population). It is convenient to use the normalized sample weights because they sum to the original sample size

$$\sum_{i=1}^n weight_i = n_0 \frac{\pi_0}{\rho_0} + n_1 \frac{\pi_1}{\rho_1} = n \pi_0 + n \pi_1 = n$$

The weights adjust the number of cases in the sample to be $n\pi_0$ and $n\pi_1$, in class 0 and 1 respectively. The classes now are in the same proportion in the adjusted sample as they are in the population. The normalization causes less distortion in standard errors and p -values. While statistical inference is not the goal of the analysis, p -values are used as tuning parameters in variable selection algorithms.

The offset method and the weighted method are not statistically equivalent. The parameter estimates are not exactly the same, but they have the same large-sample statistical properties. When the linear-logistic model is correctly specified, the offset method (un-weighted) analysis is considered superior. However, when the logistic model is merely an approximation to some nonlinear model, weighted analysis has advantages (Scott and Wild 1986).

$$\text{Sampling Weight} = \begin{cases} \frac{0.02}{0.35} = 0.58 & \text{if } \text{Ins} = 1 \\ \frac{0.98}{0.65} = 1.46 & \text{if } \text{Ins} = 0 \end{cases}$$



Sampling Weights

The DATA step adds the sampling weights to the data set DEVELOP. The weights are .058 (.02/.346) for class 1 and 1.5 (.98/.654) for class 0. They could have been assigned manually without having to reference macro variables. The logical expressions (**Ins=1**) and (**Ins=0**) in the assignment statement return the value one when true and zero when false. Consequently, this syntax is a more compact way of expressing a conditional.

```
libname pmlr "c:\workshop\winsas\pmlr";

%let pil = 0.02;

proc sql noprint;
    select mean(Ins) into :rho1 from pmlr.develop;
quit;

data develop;
    set pmlr.develop;
    sampwt=( (1-&pil) / (1-&rho1) ) * (Ins=0) + (&pil/&rho1) * (Ins=1) ;
run;
```

The WEIGHT statement in PROC LOGISTIC weights each observation in the input data set by the value of the WEIGHT variable.

```
proc logistic data=develop des;
    weight sampwt;
    model Ins = dda ddabal dep depamt cashbk checks / stb;
    score data=pmlr.new out=scored;
run;
```

The LOGISTIC Procedure

Model Information

Data Set	WORK.DEVELOP
Response Variable	Ins
Number of Response Levels	2
Weight Variable	sampwt
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	32264
Number of Observations Used	32264
Sum of Weights Read	32263.99
Sum of Weights Used	32263.99

Response Profile			
Ordered Value	Ins	Total Frequency	Total Weight
1	1	11175	645.281
2	0	21089	31618.707
Probability modeled is Ins=1.			

The results show that the response profile table has a new column named Total Weight. The figures in the column represent the sample sizes adjusted to the population proportions. Note that the Sum of the Weights equals the total sample size.

Model Convergence Status						
Convergence criterion (GCONV=1E-8) satisfied.						
Model Fit Statistics						
Criterion	Intercept Only	Intercept and Covariates				
AIC	6328.271	6194.870				
SC	6336.653	6253.542				
-2 Log L	6326.271	6180.870				
Testing Global Null Hypothesis: BETA=0						
Test	Chi-Square	DF	Pr > ChiSq			
Likelihood Ratio	145.4014	6	<.0001			
Score	230.9722	6	<.0001			
Wald	156.2504	6	<.0001			
The LOGISTIC Procedure						
Analysis of Maximum Likelihood Estimates						
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Standardized Estimate
Intercept	1	-3.1308	0.0756	1714.2522	<.0001	
DDA	1	-0.8398	0.1207	48.4361	<.0001	-0.1583
DDABal	1	0.000024	4.241E-6	32.6018	<.0001	0.0669
Dep	1	-0.0756	0.0376	4.0510	0.0441	-0.0718
DepAmt	1	2.807E-6	6.199E-6	0.2050	0.6507	0.00814
CashBk	1	-0.5691	0.4240	1.8016	0.1795	-0.0461
Checks	1	0.00479	0.0105	0.2080	0.6484	0.0139

Odds Ratio Estimates				
Effect	Point Estimate	95% Wald Confidence Limits		
DDA	0.432	0.341	0.547	
DDABal	1.000	1.000	1.000	
Dep	0.927	0.861	0.998	
DepAmt	1.000	1.000	1.000	
CashBk	0.566	0.247	1.299	
Checks	1.005	0.984	1.026	
Association of Predicted Probabilities and Observed Responses				
Percent Concordant	53.6	Somers' D	0.282	
Percent Discordant	25.3	Gamma	0.358	
Percent Tied	21.1	Tau-a	0.128	
Pairs	235669575	c	0.641	

```
proc print data=scored(obs=20);
  var p_1 dda ddabal dep depamt cashbk checks;
run;
```

Obs	P_1	DDA	DDABal	Dep	DepAmt	Cash Bk	Checks
1	0.016095	1	56.29	2	955.51	0	1
2	0.017385	1	3292.17	2	961.60	0	1
3	0.016880	1	1723.86	2	2108.65	0	2
4	0.041854	0	0.00	0	0.00	0	0
5	0.016233	1	67.91	2	519.24	0	3
6	0.018463	1	2554.58	1	501.36	0	2
7	0.017019	1	0.00	2	2883.08	0	12
8	0.016586	1	2641.33	3	4521.61	0	8
9	0.041854	0	0.00	0	0.00	0	0
10	0.017213	1	52.22	1	75.59	0	0
11	0.019232	1	6163.29	2	2603.56	0	7
12	0.009292	1	431.12	2	568.43	1	2
13	0.041854	0	0.00	0	0.00	0	0
14	0.010447	1	112.82	8	2688.75	0	3
15	0.015850	1	1146.61	3	11224.20	0	2
16	0.041854	0	0.00	0	0.00	0	0
17	0.016535	1	1241.38	3	3538.14	0	15
18	0.018644	1	298.23	0	0.00	0	0
19	0.017152	1	367.04	2	4242.22	0	11
20	0.014986	1	1229.47	4	3514.57	0	10

The probabilities from the weighted analysis are similar but not equivalent to the probabilities estimated by the offset method (pseudo model).

B.2 References

- Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984), *Classification and Regression Trees*, Chapman and Hall.
- Cohen, A. (1991), "Dummy Variables in Stepwise Regression," *The American Statistician*, 45, 226-228.
- Conover, W. J. (1980), *Practical Nonparametric Statistics*, New York: John Wiley & Sons.
- Donner, A. (1982), "The Relative Effectiveness of Procedures Commonly Used in Multiple Regression Analysis for Dealing with Missing Values," *The American Statistician*, 36, 378-381.
- Duffy, T. J. and Santner, D. E. (1989), *The Statistical Analysis of Discrete Data*, New York: Springer-Verlag.
- Georges, J.E. (2004), *Advanced Predictive Modeling Using SAS® Enterprise Miner 5.1 Course Notes*, Cary, NC: SAS Institute Inc.
- Greenacre, M. J. (1988), "Clustering Rows and Columns of a Contingency Table," *Journal of Classification*, 5, 39-51.
- Greenacre, M. J. (1993), *Correspondence Analysis in Practice*, San Diego, CA: Academic Press.
- Hand, D. J. (1997), *Construction and Assessment of Classification Rules*, New York: John Wiley & Sons.
- Hand, D. J. and Henley, W. E. (1997), "Statistical Classification Methods in Consumer Credit Scoring: A Review," *Journal of the Royal Statistical Society A*, 160, 153-541.
- Harrell, F. E. (1997), *Predicting Outcomes: Applied Survival Analysis and Logistic Regression*, Charlottesville Virginia: School of Medicine, University of Virginia.
- Hastie, T. J. and Tibshirani, R. J. (1990), *Generalized Additive Models*, London: Chapman and Hall.
- Huber, P. J. (1997), "From Large to Huge: A Statistician's Reactions to KDD & DM," *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, Menlo Park, CA: AAAI Press.
- Jackson, J. E. (1991), *A Users Guide to Principal Components*, New York: John Wiley & Sons.
- Jones, M. P. (1996), "Indicator and Stratification Methods for Missing Explanatory Variables in Multiple Linear Regression," *Journal of the American Statistical Association*, 91, 222-230.
- Lawless, J. F. and Singhal, K. (1978), "Efficient Screening of Nonnormal Regression Models," *Biometrics*, 34, 318-327.
- Little, R. J. A. (1992), "Regression with Missing X's: A Review," *Journal of the American Statistical Association*, 87, 1227-1237.
- Magee, L. (1998), "Nonlocal Behavior in Polynomial Regressions," *The American Statistician*, 52, 20-22.
- Mantel, N. (1970), "Why Stepdown Procedures in Variable Selection," *Technometrics*, 12, 621-625.

- McLachlan, G. J. (1992), *Discriminant Analysis and Statistical Pattern Recognition*, New York: John Wiley & Sons.
- Nelson, R. W. (1997), *Credit Card Risk Management*, Warren Taylor Publications.
- Prentice, R. L. and Pike, R. (1979), "Logistic Disease Incidence Models and Case-Control Studies," *Biometrika*, 66, 403-411.
- Ripley, B. D. (1996), *Pattern Recognition and Neural Networks*, Cambridge University Press.
- Sarle, W. S. (1994), "Neural Networks and Statistical Models," *Proceedings of the 19th Annual SUGI*, Cary, NC: SAS Institute Inc.
- SAS Institute Inc., *SAS[®] 9.1 Language Reference: Concepts, Third Edition*, Cary, NC: SAS Institute Inc., 2005.
- SAS Institute Inc., *Base SAS[®] 9.1 Procedures Guide, Second Edition*, Cary, NC: SAS Institute Inc., 2006.
- SAS Institute Inc., *SAS/STAT[®] 9.1 Users Guide*, Cary, NC: SAS Institute Inc., 2004.
- SAS Institute Inc., *Logistic Regression Examples Using the SAS System, Version 6, First Edition*, Cary, NC: SAS Institute Inc., 1995.
- Scott, A. J. and Wild, C. J. (1986), "Fitting Logistic Regression Models under Case-Control or Choice Based Sampling," *Journal of the Royal Statistical Society B*, 48, 170-182.
- Scott, A. J. and Wild, C. J. (1997), "Fitting Regression Models to Case-Control Data by Maximum Likelihood," *Biometrika*, 84, 57-71.

Appendix C Index

Appendix D **Error! No table of contents entries found.**Index

%

%LET statement, 1-6

A

accuracy, 4-32

all subsets regression, 5-9–5-20

ASSESS macro, 5-11

assessing fit

 using profit, 4-56–4-58

attrition predicting, 1-5

automatic score code generation

 correcting the intercept, 2-25–2-26

 Output Delivery System, 2-13–2-20

B

backward elimination method, 3-68

Bayes rule, 4-51

binary indicator variables, 2-3

binary logistic regression model, 2-10–2-13

bins, 3-52

C

c statistic, 4-61, 4-62

cases, 1-3

categorical inputs, 3-14

class separation, 4-59

CLASS statement, LOGISTIC procedure

 PARAM option, 2-10

 REF option, 2-10

CLASS statement, MEANS procedure, 3-53

classification rules

 depth, 4-35

classifier performance, 4-38–4-47

cluster components, 3-28

cluster imputation, 3-11

CLUSTER procedure, 4-8

FREQ statement, 3-16

cluster representatives, 3-31

clustering, 3-32–3-43

clustering levels, 3-15, 3-16–3-24

cluster-mean imputation, 3-11

coefficients, 3-29

complete-case analysis, 3-4

complexity, 5-3

concentration curves, 4-35

confusion matrix, 4-32, 4-49

cost ratios, 4-53

credit scoring, 1-5

cross-validation, 4-5

curse of dimensionality, 1-11

cutoff values, 4-49

D

data sets

 test, 4-4

 training, 4-4

 validation, 4-4

data splitting, 4-4

database marketing, 1-5

depth

 classification rules, 4-35

DES option

 PROC LOGISTIC statement, 2-10

dimensions, 1-11

divergence statistics, 4-59

divisive clustering, 3-30

dummy variables, 3-13

E

EDF option

 PROC NPAR1WAY statement, 4-62

eigenvalues, 3-27

eigenvectors, 3-29

empirical logits, 3-52–3-57

error rate, 4-32

estimators

flexible multivariate function, 3-58

F

FAST option

MODEL statement (LOGISTIC), 3-69

FITANDSCORE macro, 5-13

fitted surface, 2-5

flexible multivariate function estimators, 3-58

forward selection method, 3-68

fraud detection, 1-5

FREQ procedure, 1-6

FREQ statement

CLUSTER procedure, 3-16

G

gains charts, 4-35

generalization, 1-4

generalized linear model, 2-4

Greenacre's method, 3-16

GROUPS= option

PROC RANK statement, 3-53

H

H= option

PROC TREE statement, 3-22

hand-crafted

input variables, 3-58

HEIGHT statement

TREE procedure, 3-42

HI option

PROC VARCLUS statement, 3-32

high dimensionality, 1-11

HOEFFDING option

PROC CORR statement, 3-45

Hoeffding's D statistics, 3-45

I

imputation

indicators, 3-7

missing values, 3-6

imputing missing values, 3-9–3-10

indicators

imputation, 3-7

input variables, 1-3

binning, 3-53

hand-crafted, 3-58

transforming, 3-58

interactions, 1-13

J

joint sampling, 2-21

K

Kolmogorov-Smirnov test statistics (K-S), 4-60, 4-62

L

lift charts, 4-35

logistic discrimination, 2-7

LOGISTIC procedure, 5-10

CLASS statement, 2-10

MODEL statement, 2-10

scalability, 3-69

UNITS statement, 2-10

logistic regression model, 3-58

binary, 2-10–2-13

logit link function, 2-4

Lorentz curves, 4-35

M

MAXEIGEN= option

PROC VARCLUS statement, 3-32

maximum likelihood estimates, 2-8

mean squared error. See MSE

mean-imputation, 3-11

MEANS procedure, 1-6

CLASS statement, 3-53

OUTPUT statement, 3-53

measurement scales, 1-10

METHOD= option

PROC CLUSTER statement, 3-16

PROC STDIZE statement, 3-10

misclassification costs, 4-50

missing completely at random (MCAR), 3-3

missing values, 3-3

imputation, 3-6, 3-9–3-10

model assessment, 4-6–4-29

model interpretation, 2-6

model selection

overfitting, 1-14

underfitting, 1-14

MODEL statement, LOGISTIC procedure

FAST option, 3-69

OFFSET= option, 2-27

ROCEPS= option, 4-41
 SELECTION= option, 3-69
 SLSTAY option, 3-71
 STB option, 2-10
MSE
 binary target models, 5-7
 estimating, 5-4, 5-6
MSE decomposition
 squared bias, 5-5
 variance, 5-5

N
NCLUSTERS= option
 PROC TREE statement, 3-23
 nonlinearities, 1-13, 3-58–3-65
NWAY option
 PROC MEANS statement, 3-16

O
OFFSET option
 MODEL statement (LOGISTIC), 2-27
 offsets, 2-22–2-23
 opportunistic data, 1-9
OUT= option
 PROC SCORE statement, 2-15
 PROC STDIZE statement, 3-10
OUTEST= option
 PROC LOGISTIC statement, 2-15
Output Delivery System
 automatic score code generation, 2-13–2-20
 OUTPUT statement, 3-40
OUTPUT statement, MEANS procedure, 3-53
OUTPUT statement, Output Delivery System, 3-40
OUTROC= option
 SCORE statement (LOGISTIC), 4-38
OUTTREE= option
 PROC CLUSTER statement, 3-16
 PROC VARCLUS statement, 3-32
 overfitting, 4-3, 5-3
 model selection, 1-14
 oversampled test set, 4-36
 oversampling, 2-22
 adjustments, 4-37
 correcting for, 2-24–2-28

P

PARAM option

CLASS statement (LOGISTIC), 2-10
 PCTLPRE option
 OUTPUT statement (UNIVARIATE), 4-27
 polynomial models, 3-58
 posterior probability, 2-3, 2-23
 predicted values, 4-33
 predictive model, 1-3
 business applications, 1-5
 principal components analysis, 3-27
 prior probabilities, 4-37
 priors, 2-21
PROC CLUSTER statement
 METHOD= option, 3-16
 OUTTREE= option, 3-16
PROC CORR statement
 HOEFFDING option, 3-45
 RANK option, 3-45
 SPEARMAN option, 3-45
PROC LOGISTIC statement
 DES option, 2-10
 OUTEST= option, 2-15
PROC MEANS statement
 NWAY option, 3-16
PROC NPAR1WAY statement
 EDF option, 4-62
 WILCOXON option, 4-62
PROC RANK statement
 GROUPS= option, 3-53
PROC SCORE statement
 OUT= option, 2-15
 TYPE= option, 2-15
PROC STDIZE statement
 METHOD= option, 3-10
 OUT= option, 3-10
 REONLY option, 3-10
PROC TREE statement
 H= option, 3-22
 NCLUSTERS= option, 3-23
PROC VARCLUS statement
 HI= option, 3-32
 MAXEIGEN= option, 3-32
 OUTTREE= option, 3-32
 SHORT option, 3-32
 profit matrix, 4-56–4-58

Q

quasi-complete separation, 3-14

R

- RANK option
 - PROC CORR statement, 3-45
- RANK procedure
 - RANKS statement, 3-53
 - VAR statement, 3-53
- RANKS statement
 - RANK procedure, 3-53
- rare target event, 1-12
- receiver operating characteristic (ROC), 4-34
 - area under the curve, 4-61
- redundancy, 3-26
- REF option
 - CLASS statement (LOGISTIC), 2-10
- REONLY option
 - METHOD statement (STDIZE), 4-29
 - PROC STDIZE statement, 3-10
- retail banking example, 1-6–1-8
- ROC curve, 4-61
- ROCEPS = option
 - MODEL statement (LOGISTIC), 4-41

S

- sampling
 - joint, 2-21
 - separate, 2-21
- scalability
 - LOGISTIC procedure, 3-69
- scatter plots, 3-51
- Schwarz Bayes criterion (SBC), 3-73
- scorability, 3-5
- SCORE procedure
 - VAR statement, 2-15
- SCORE statement, LOGISTIC procedure
 - OUTROC= option, 4-38
- scoring new cases, 2-13–2-20
- selecting subsets, 3-70–3-76
- SELECTION= option
 - MODEL statement (LOGISTIC), 3-69
- sensitivity, 4-33
- separate sampling, 2-21, 2-24–2-28
- SHORT option
 - PROC VARCLUS statement, 3-32
- SLSTAY option
 - MODEL statement (LOGISTIC), 3-71
- Spearman correlation statistics, 3-45
- SPEARMAN option

- PROC CORR statement, 3-45
- specificity, 4-33
- STB option
 - MODEL statement (LOGISTIC), 2-10
- STDIZE procedure, 4-8, 4-28
 - VAR statement, 3-10
- stepwise selection method, 3-67
- subsets selection, 3-66, 3-70–3-76
- supervised classification, 1-3
- SURVEYSELECT procedure, 4-7
- SYMPUT routine, 3-40

T

- target marketing, 1-5
- target variables, 1-3
- test data set, 4-4
- training data set, 4-4
- transforming
 - input variables, 3-58
- TREE procedure, 4-9
 - HEIGHT statement, 3-42
- TYPE= option
 - PROC SCORE statement, 2-15

U

- underfitting
 - model selection, 1-14
- UNITS statement
 - LOGISTIC procedure, 2-10
- univariate screening, 3-44–3-50
- univariate smoothing, 3-51

V

- validation data set, 4-4
- VAR statement, RANK procedure, 3-53
- VAR statement, SCORE procedure, 2-15
- VAR statement, STDIZE procedure, 3-10
- VAR statement, VARCLUS procedure, 3-32
- VARCLUS procedure, 4-12
 - VAR statement, 3-32
- variable clustering, 3-32–3-43
- variable screening, 3-44–3-50

W

- WILCOXON option
 - PROC NPAR1WAY statement, 4-62
- Wilcoxon test statistics, 4-62

