SBA Statistical Business Analyst using SAS

SBA3 Predictive Modeling with Logistic Regression

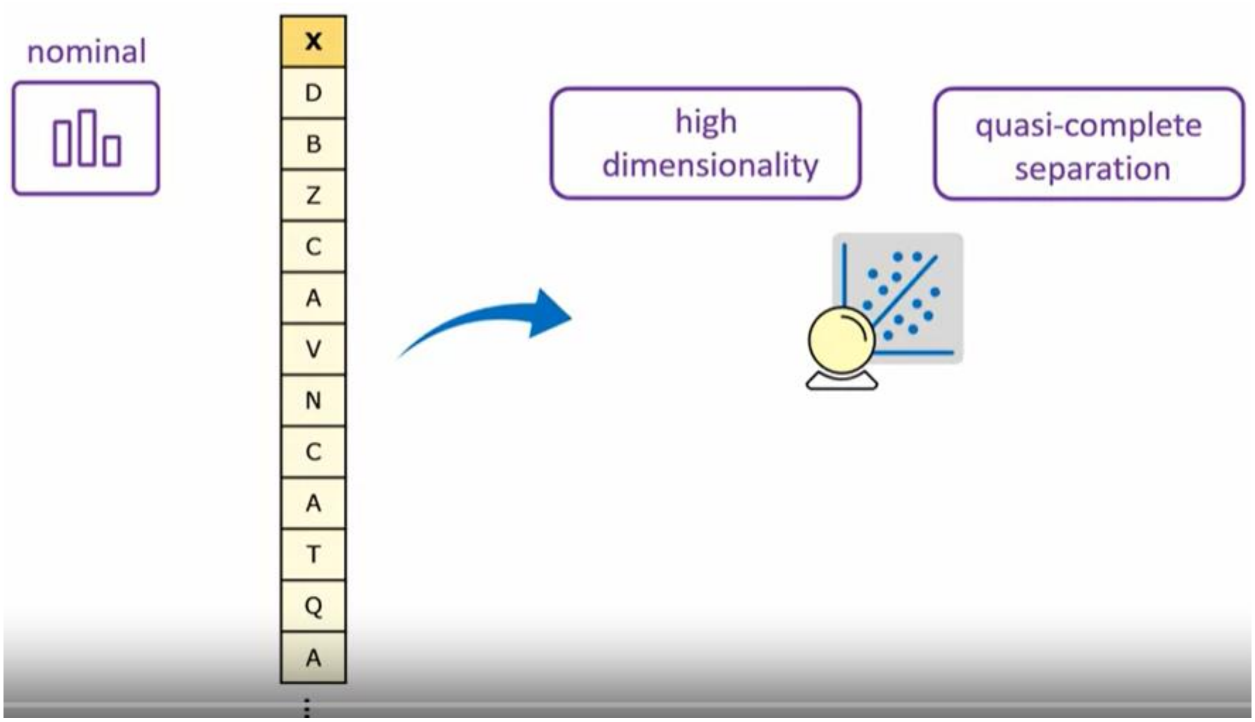
W3b Working with Categorical Inputs

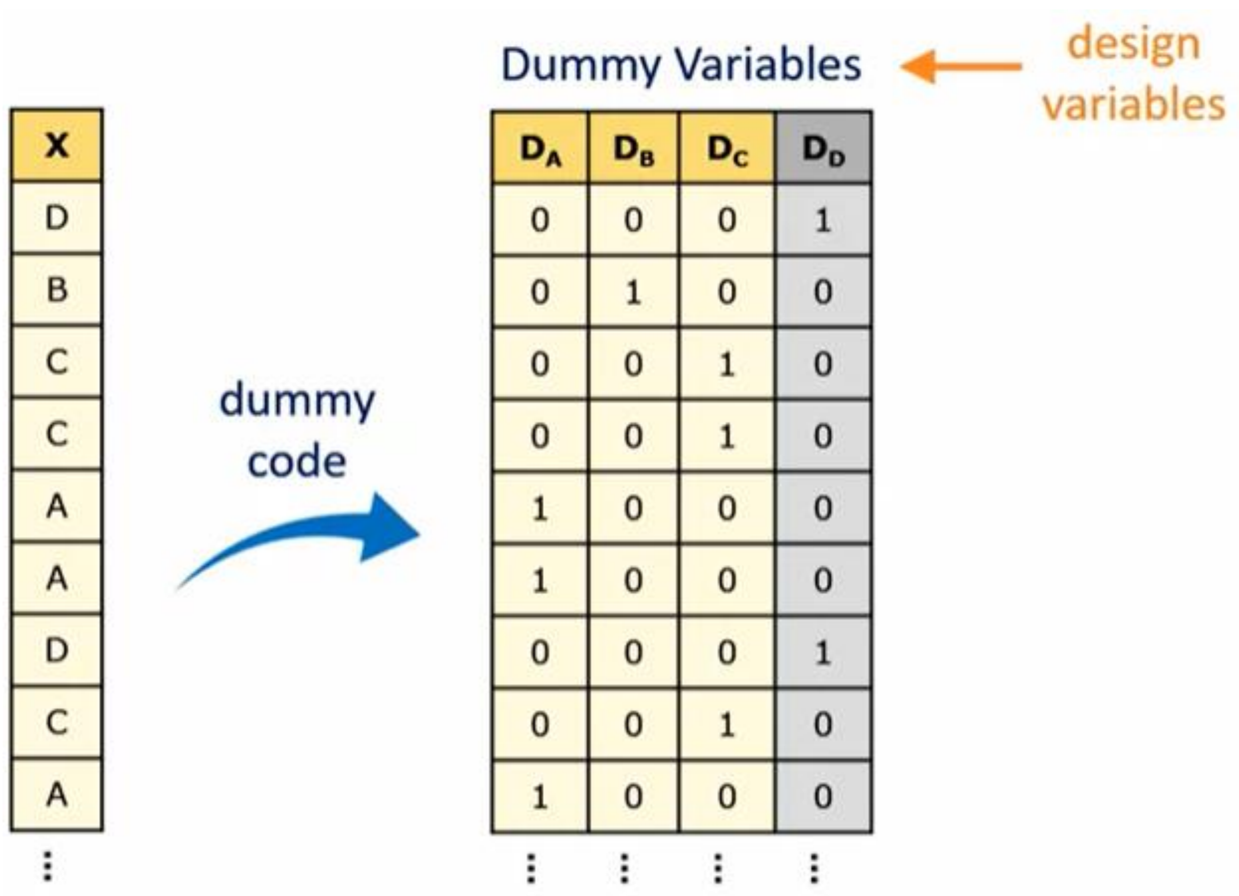
Introduction

In this topic, you learn to do the following:

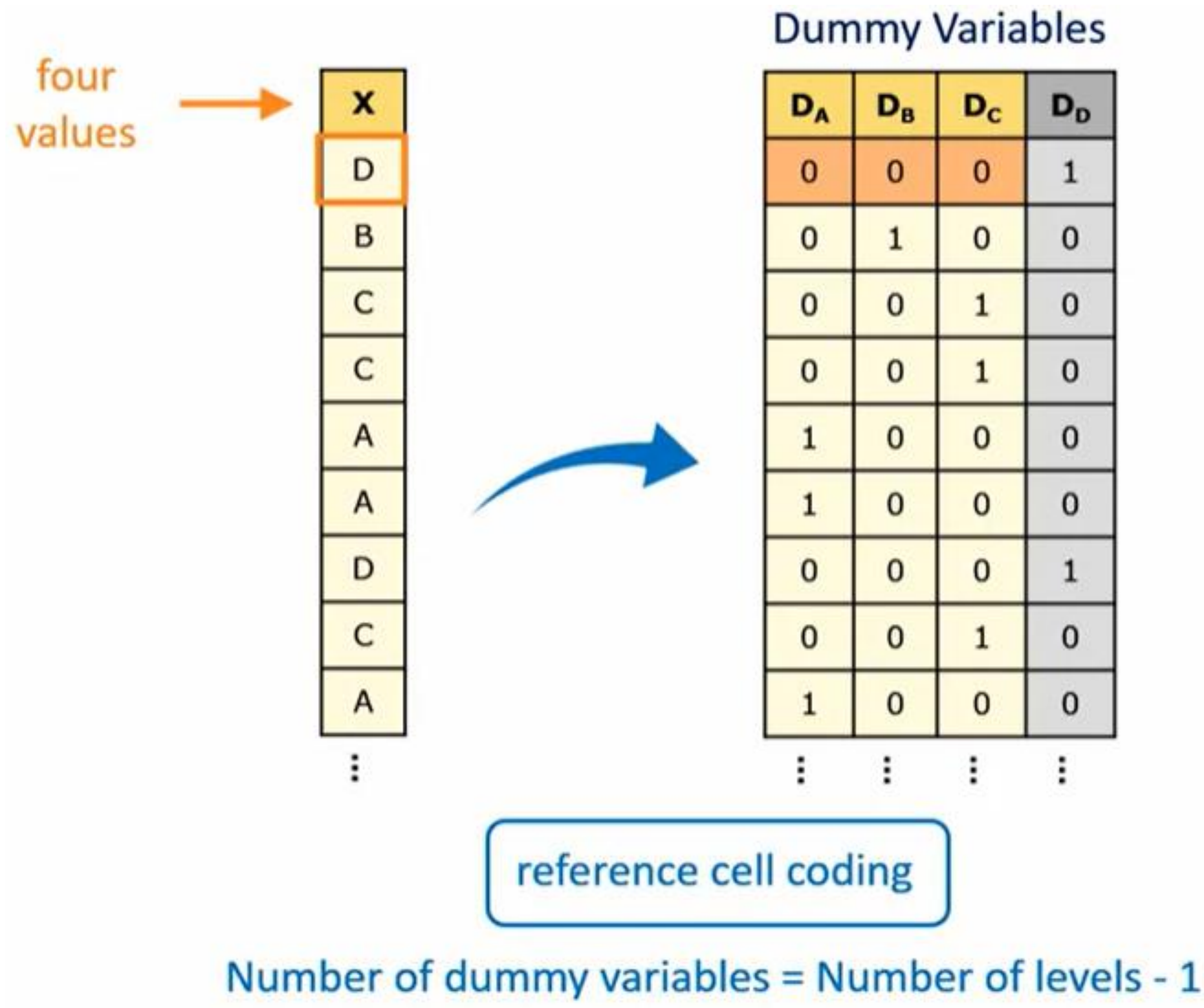
- identify the problems associated with having numerous levels in a categorical input
- identify possible solutions to those problems
- use the CLUSTER procedure to cluster the levels of a categorical input
- use smoothed weight-of-evidence coding to convert a categorical predictor to a continuous predictor

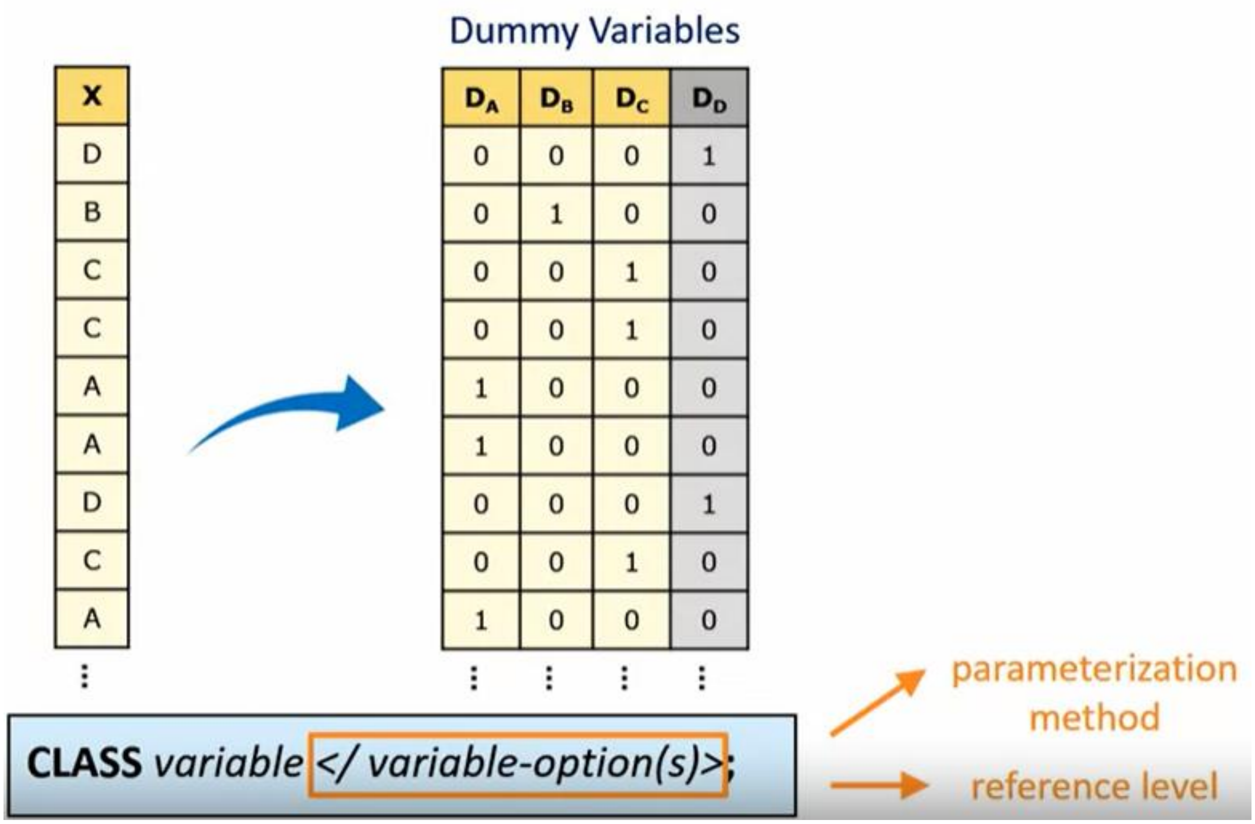
Problems Caused by Categorical Inputs

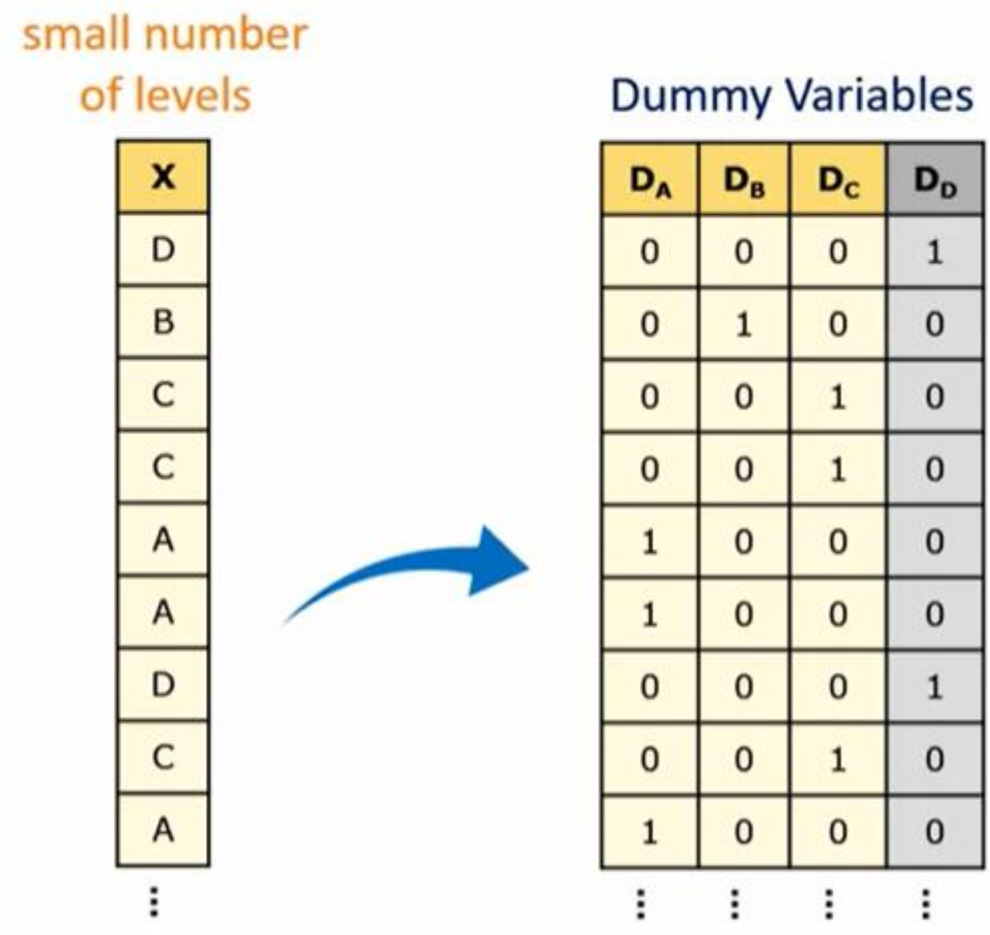


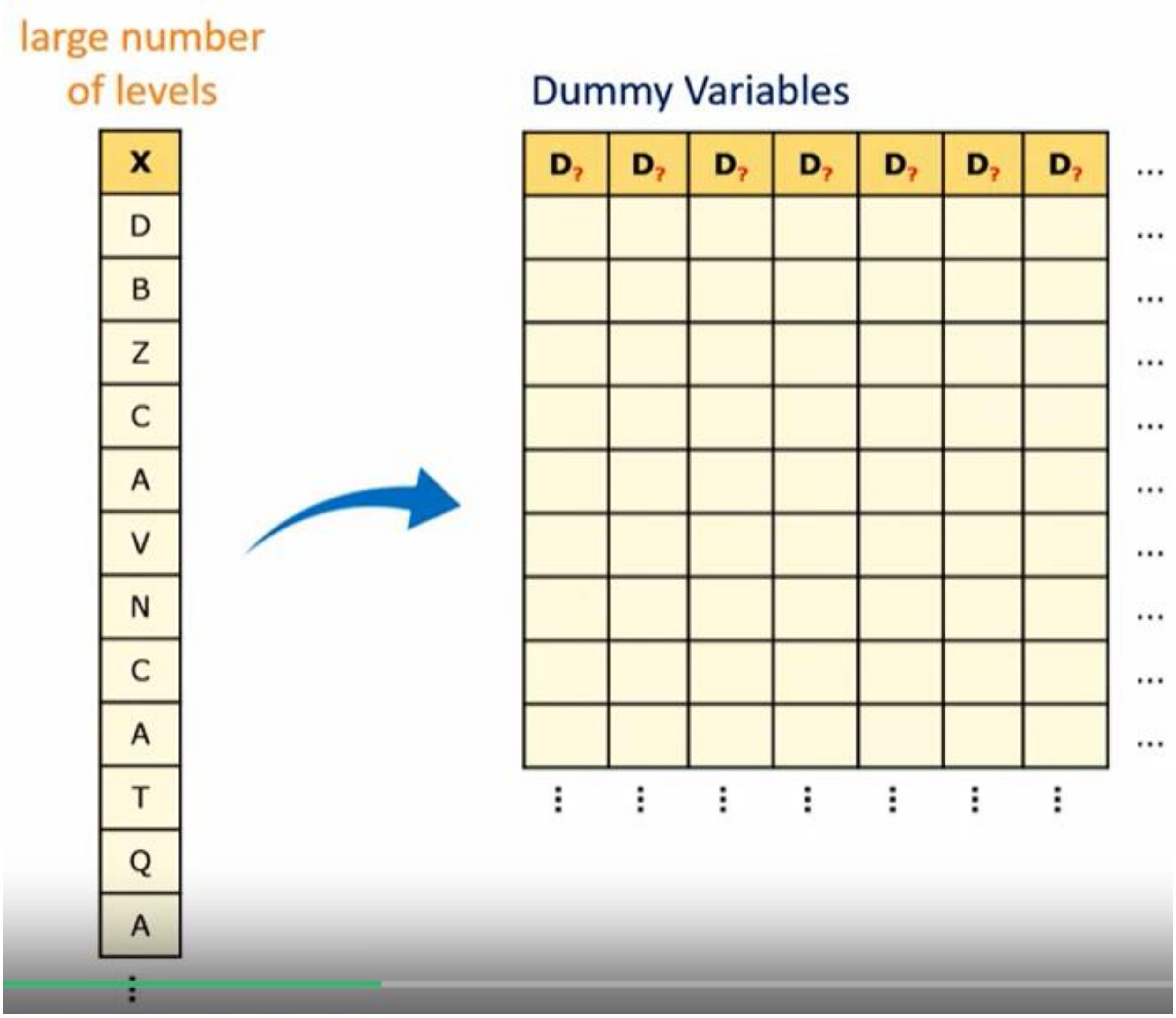


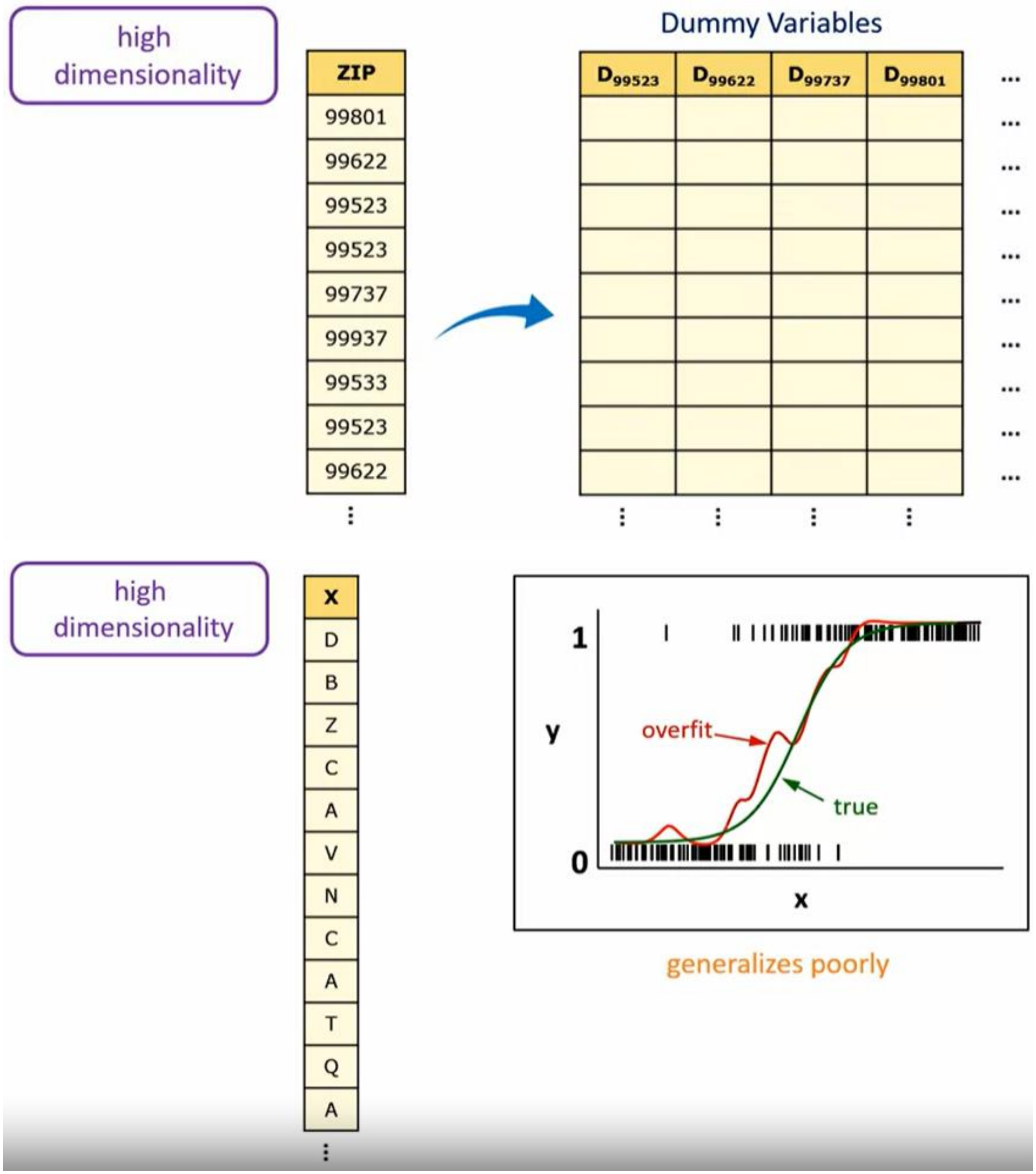
D_A	D_B	D_C	D_D
0	0	0	1
0	1	0	0
0	0	1	0
0	0	1	0
1	0	0	0
1	0	0	0
0	0	0	1
0	0	1	0
1	0	0	0
⋮	⋮	⋮	⋮

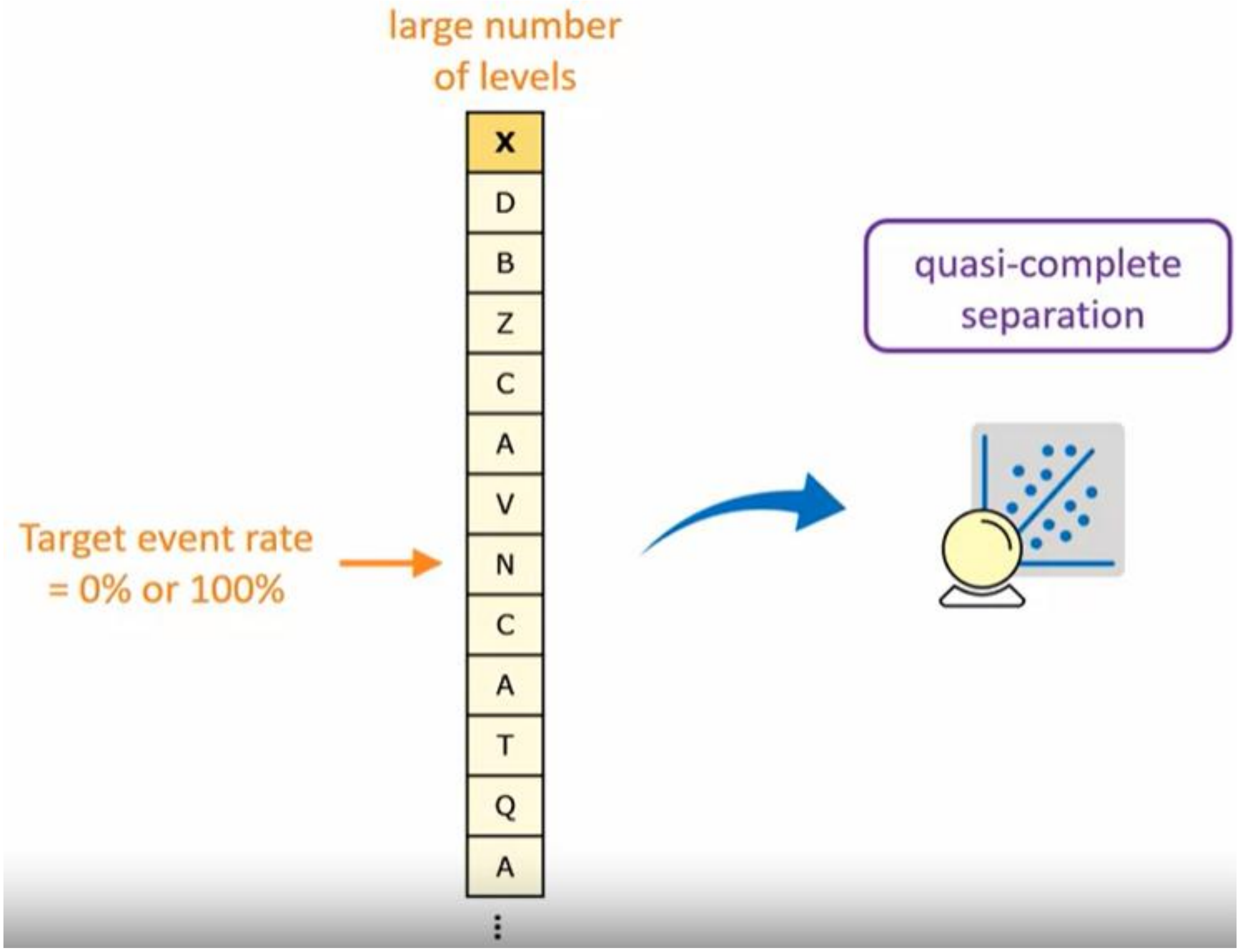

four values
X
D
B
C
C
A
A
D
C
A
Dummy Variables
D_A D_B D_C D_D
0 0 0 1
0 1 0 0
0 0 1 0
0 0 1 0
1 0 0 0
1 0 0 0
0 0 0 1
0 0 1 0
1 0 0 0
reference cell coding
Number of dummy variables = Number of levels - 1

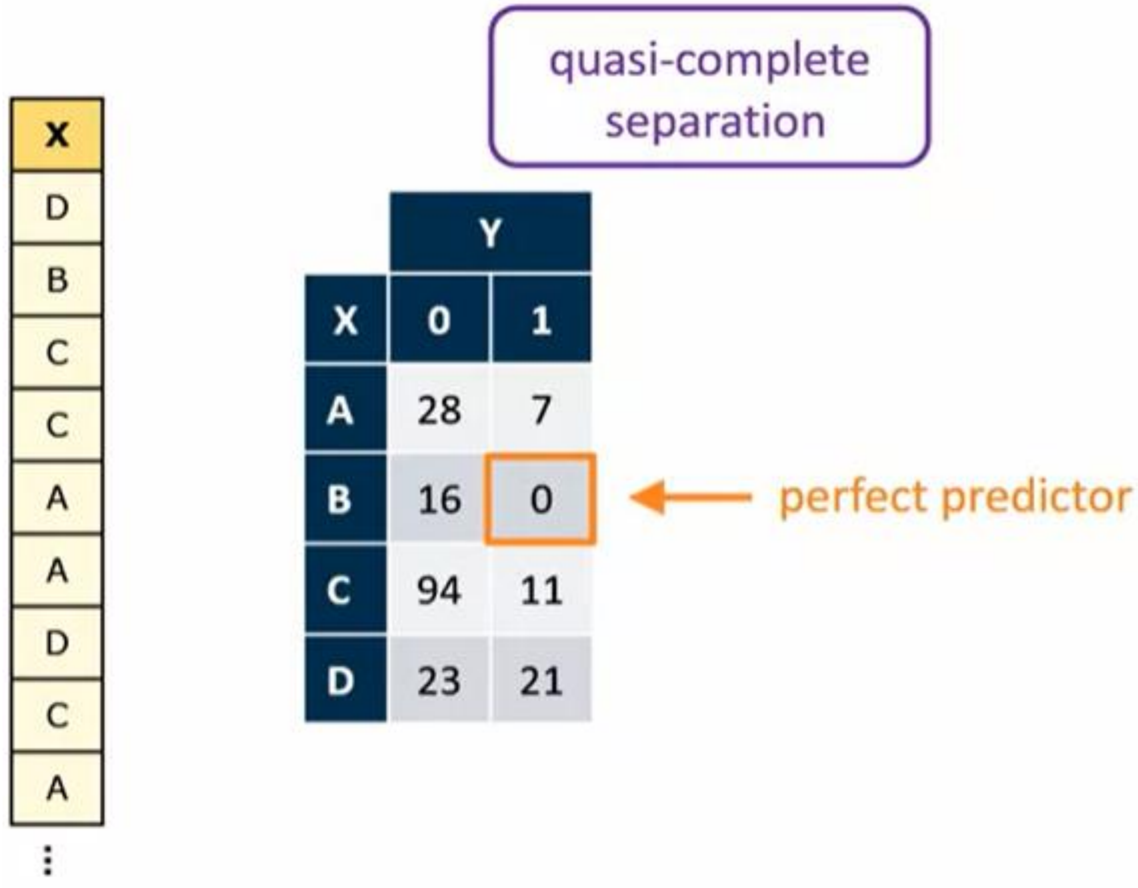

Dummy Variables
X
D
B
C
C
A
A
D
C
A
D_A D_B D_C D_D
0 0 0 1
0 1 0 0
0 0 1 0
0 0 1 0
1 0 0 0
1 0 0 0
0 0 0 1
0 0 1 0
1 0 0 0
CLASS variable </ variable-option(s)>;
parameterization method
reference level


small number
of levels
X
D
B
C
C
A
A
D
C
A
Dummy Variables
D_A D_B D_C D_D
0 0 0 1
0 1 0 0
0 0 1 0
0 0 1 0
1 0 0 0
1 0 0 0
0 0 0 1
0 0 1 0
1 0 0 0


large number
of levels
Dummy Variables
X
D
B
Z
C
A
V
N
C
A
T
Q
A
D?
D?
D?
D?
D?
D?
D?


high dimensionality
ZIP
99801
99622
99523
99523
99737
99937
99533
99523
99622
Dummy Variables
D99523
D99622
D99737
D99801
high dimensionality
x
D
B
Z
C
A
V
N
C
A
T
Q
A
1
0
y
x
overfit
true
generalizes poorly


large number
of levels
X
D
B
Z
C
A
V
N
C
A
T
Q
A
Target event rate
= 0% or 100%
quasi-complete
separation



quasi-complete separation
X
D
B
C
C
A
A
D
C
A
Y
X 0 1
A 28 7
B 16 0
C 94 11
D 23 21
perfect predictor

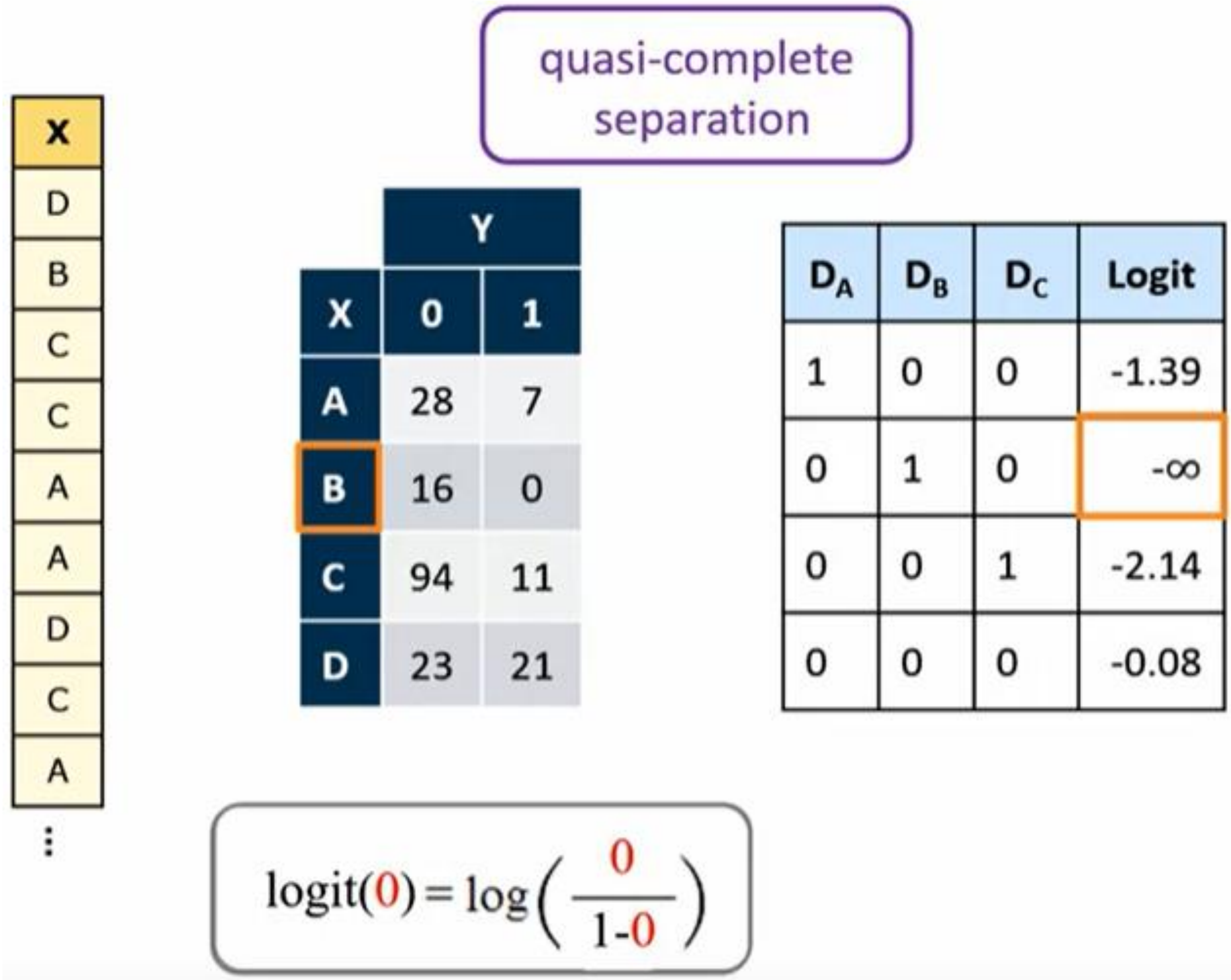
quasi-complete separation

X
D
B
C
C
A
A
D
C
A
⋮

	Y	
X	0	1
A	28	7
B	16	0
C	94	11
D	23	21

D_A	D_B	D_C	Logit
1	0	0	-1.39
0	1	0	$-\infty$
0	0	1	-2.14
0	0	0	-0.08

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$$



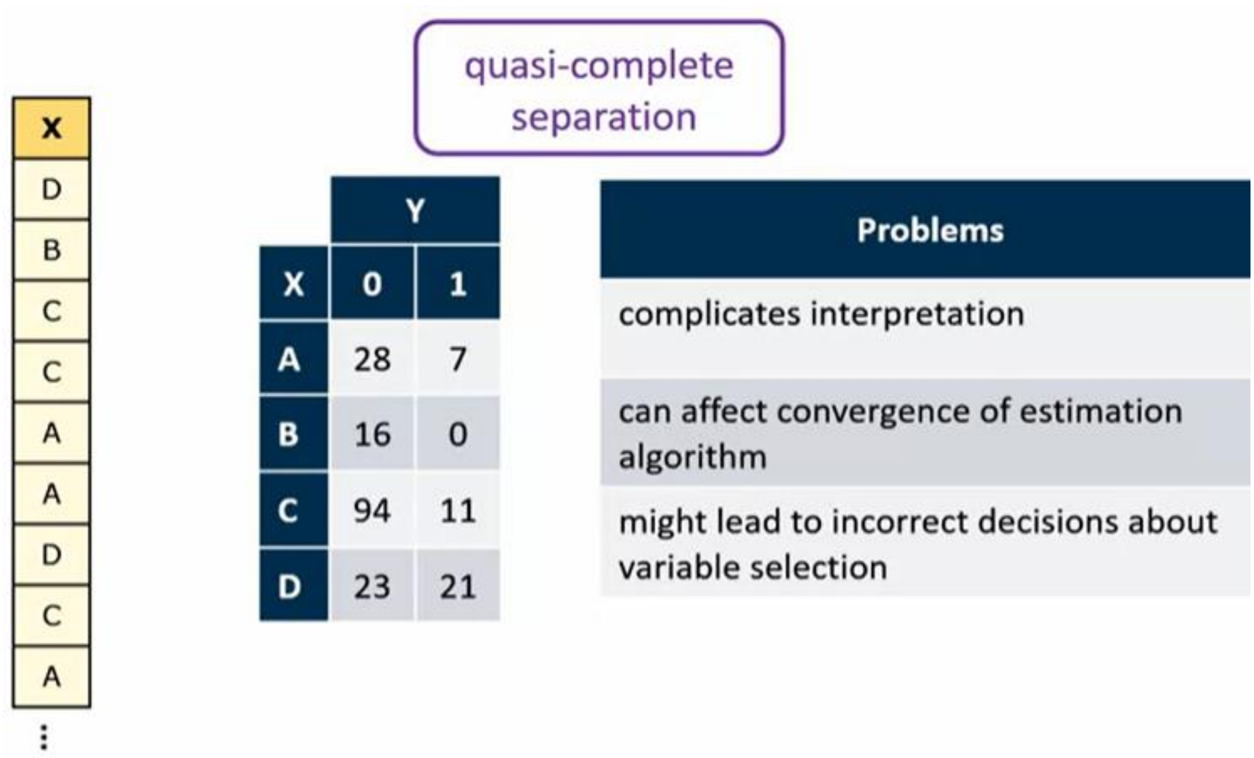
quasi-complete separation

X
D
B
C
C
A
A
D
C
A
⋮

	Y	
X	0	1
A	28	7
B	16	0
C	94	11
D	23	21

D_A	D_B	D_C	Logit
1	0	0	-1.39
0	1	0	$-\infty$
0	0	1	-2.14
0	0	0	-0.08

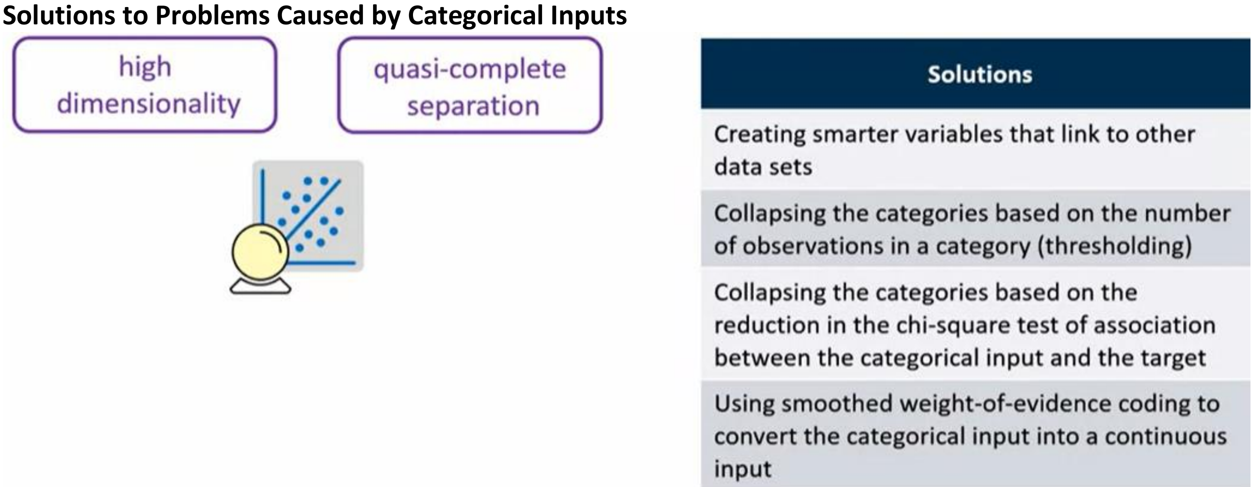
$$\text{logit}(0) = \log\left(\frac{0}{1-0}\right)$$



	Y	
X	0	1
A	28	7
B	16	0
C	94	11
D	23	21

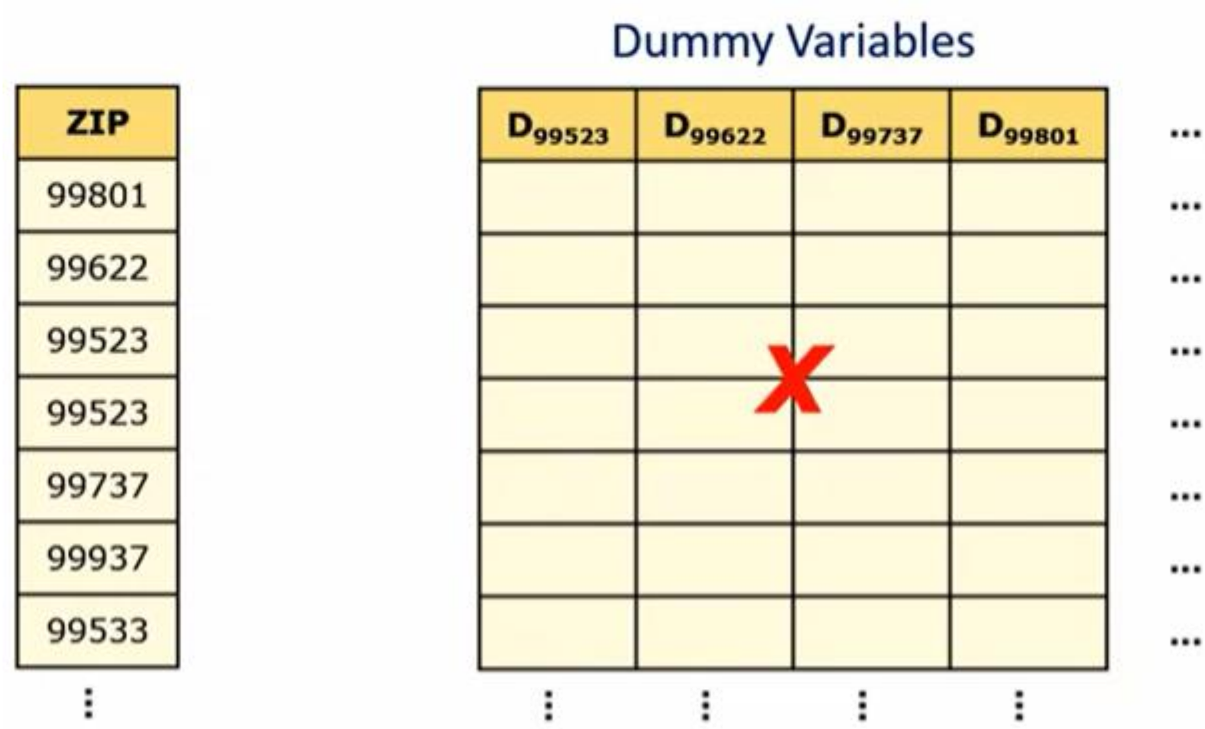
Problems
complicates interpretation
can affect convergence of estimation algorithm
might lead to incorrect decisions about variable selection

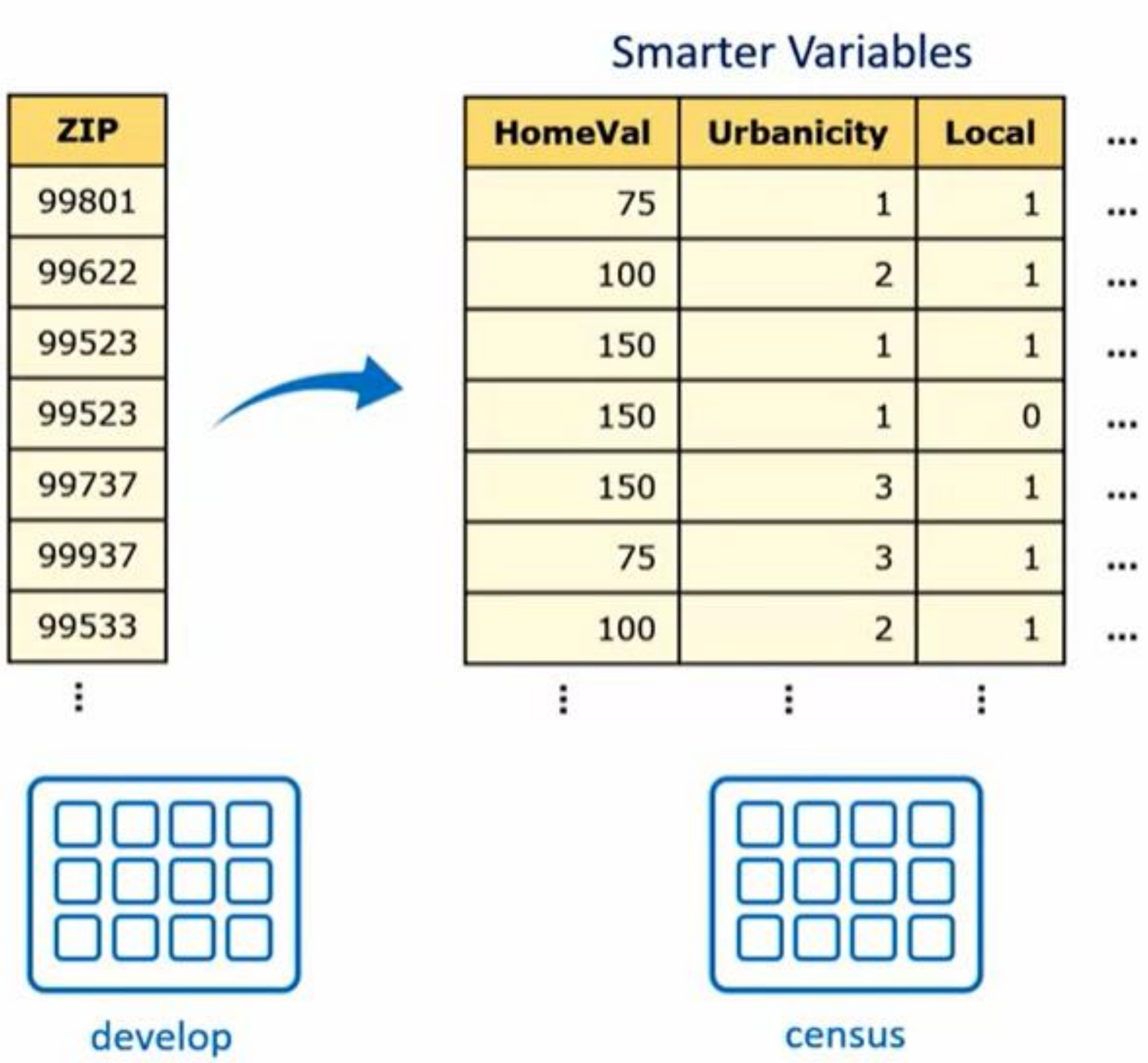
Solutions to Problems Caused by Categorical Inputs



Solutions
Creating smarter variables that link to other data sets
Collapsing the categories based on the number of observations in a category (thresholding)
Collapsing the categories based on the reduction in the chi-square test of association between the categorical input and the target
Using smoothed weight-of-evidence coding to convert the categorical input into a continuous input

Linking to Other Data Sets





ZIP
99801
99622
99523
99523
99737
99937
99533
⋮

HomeVal	Urbanicity	Local	...
75	1	1	...
100	2	1	...
150	1	1	...
150	1	0	...
150	3	1	...
75	3	1	...
100	2	1	...
⋮	⋮	⋮	

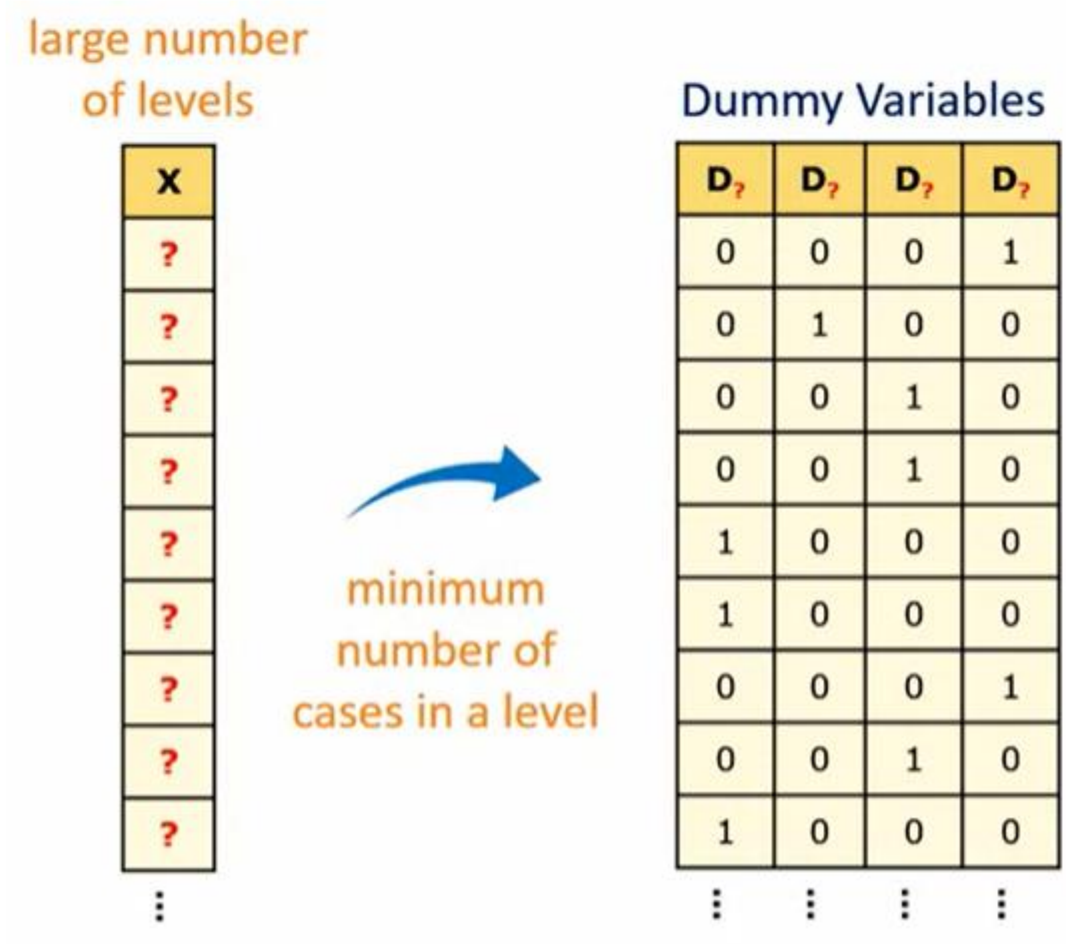
Collapsing Categories by Thresholding

threshold = 50

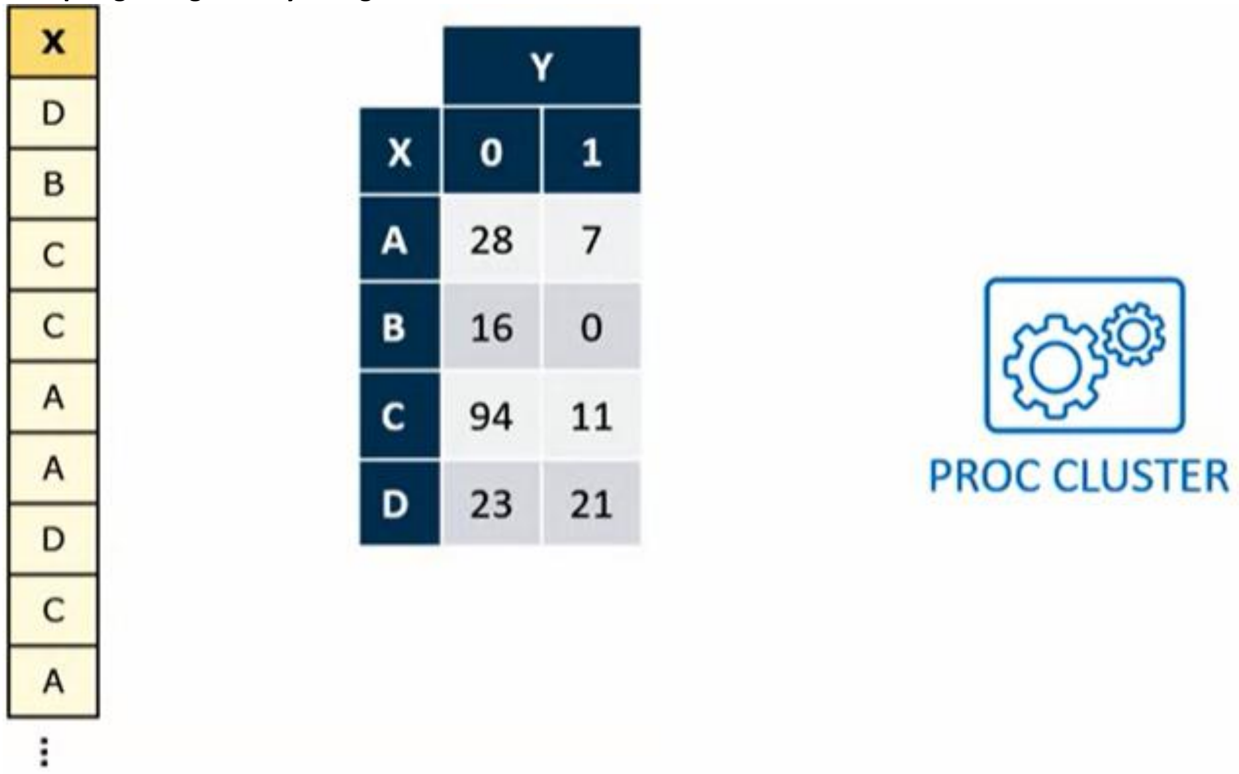
Level	Number of Cases
A	1562
B	970
C	223
D	111
E	85
F	23
G	17
H	12
I	5

Other

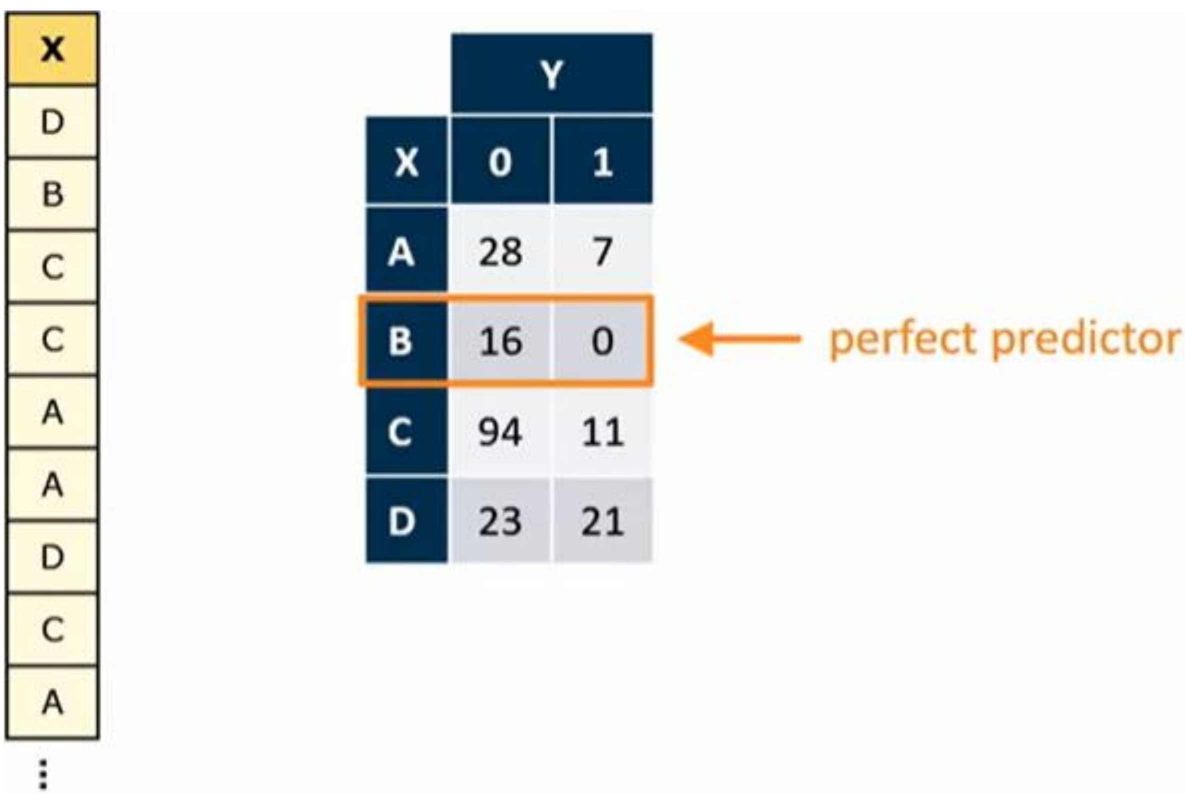
limits spurious input-target associations


large number of levels
X
?
?
?
?
?
?
?
?
?
minimum number of cases in a level
Dummy Variables
D?
D?
D?
D?
0 0 0 1
0 1 0 0
0 0 1 0
0 0 1 0
1 0 0 0
1 0 0 0
0 0 0 1
0 0 1 0
1 0 0 0

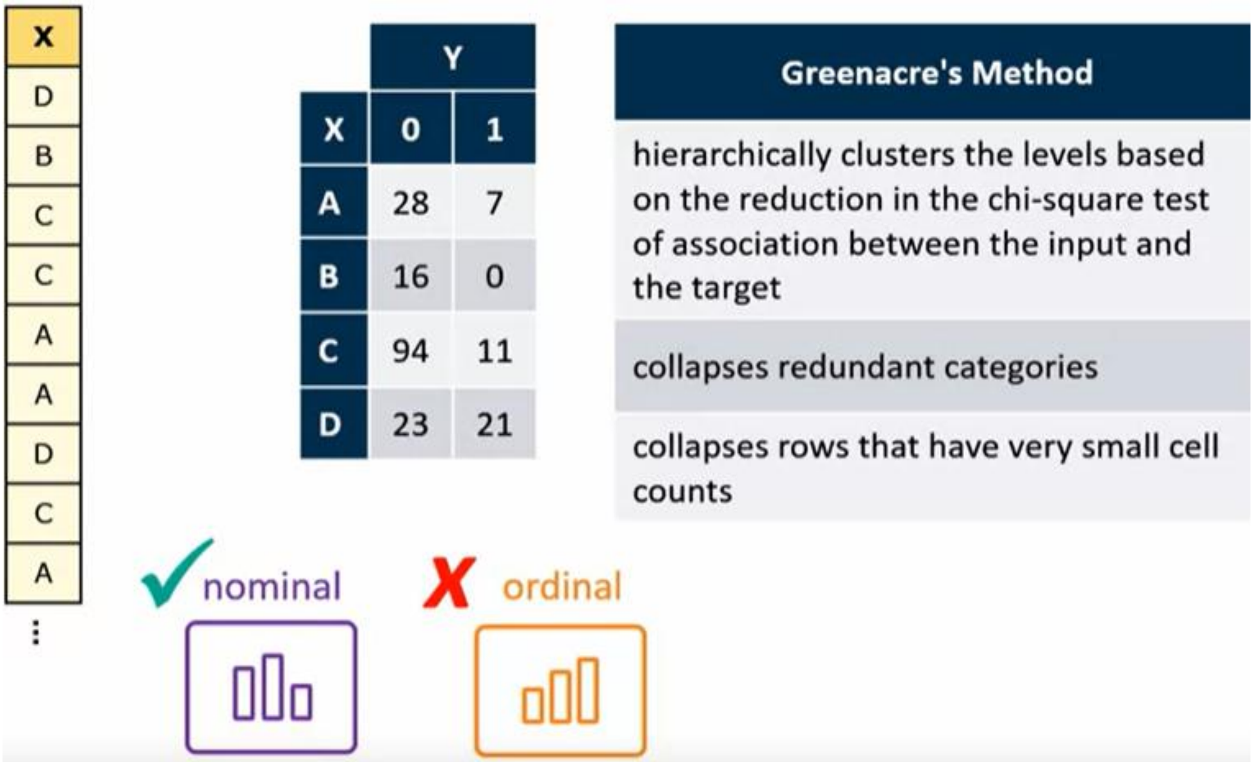
Collapsing Categories by Using Greenacre's Method

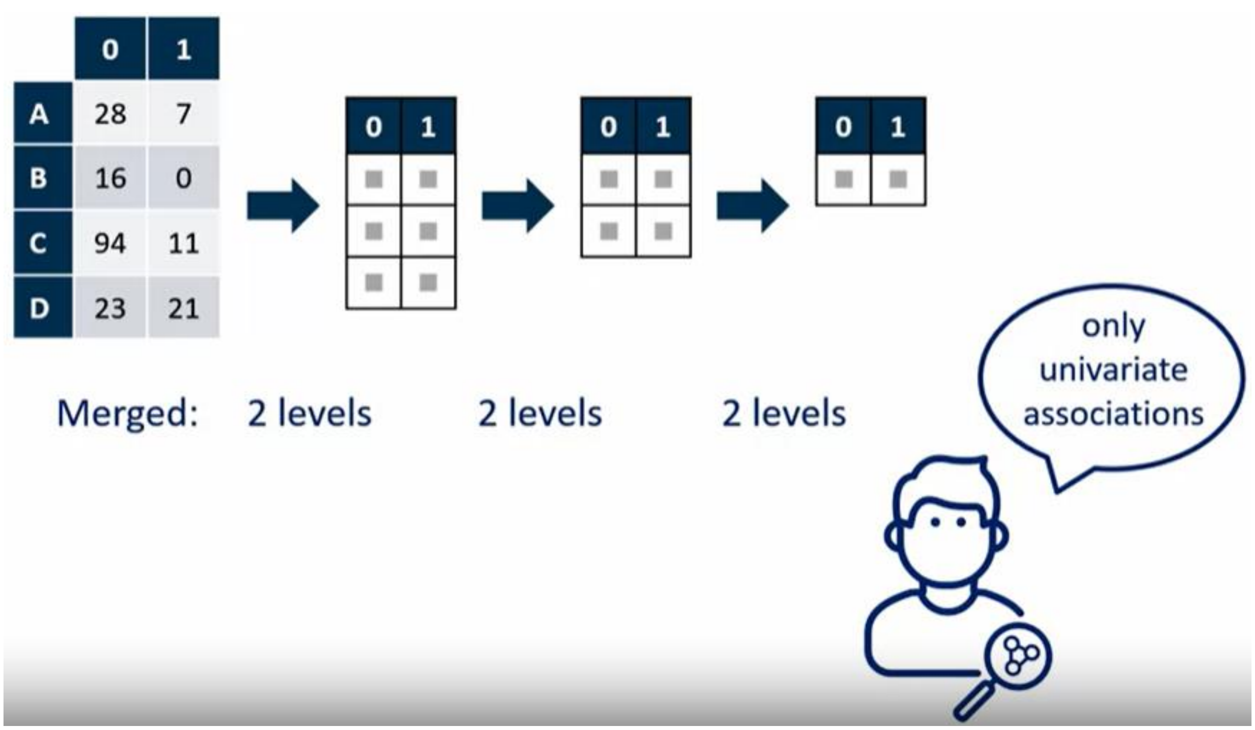


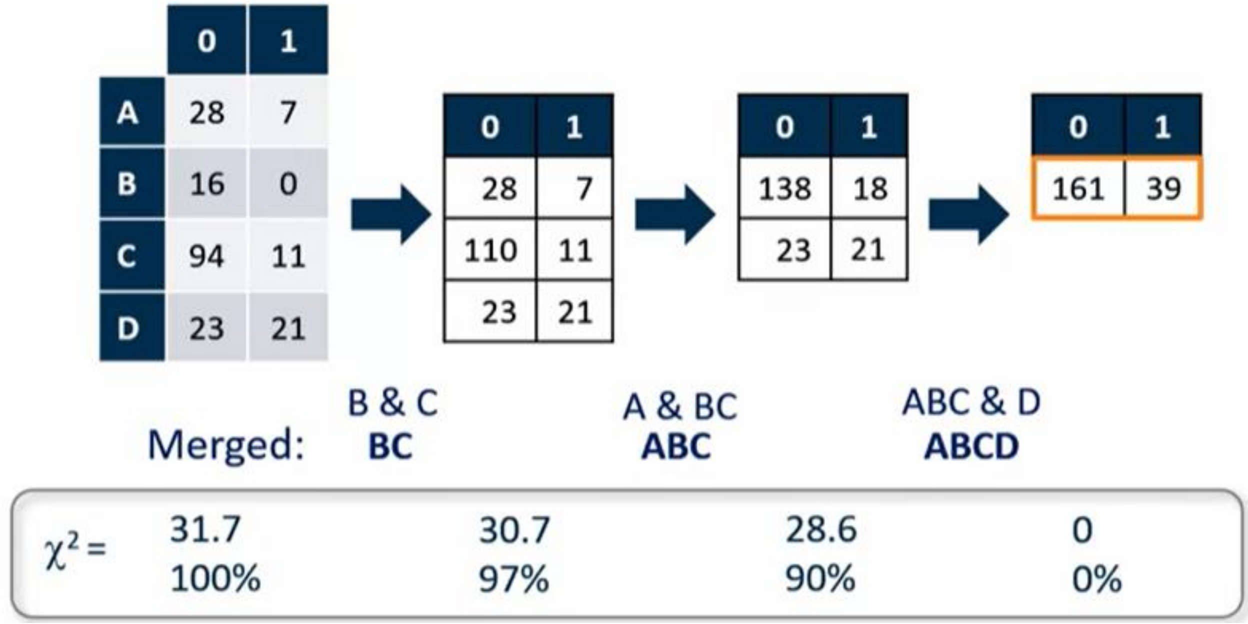
	Y	
X	0	1
A	28	7
B	16	0
C	94	11
D	23	21



	Y	
X	0	1
A	28	7
B	16	0
C	94	11
D	23	21


X
D
B
C
C
A
A
D
C
A
Y
X 0 1
A 28 7
B 16 0
C 94 11
D 23 21
Greenacre's Method
hierarchically clusters the levels based on the reduction in the chi-square test of association between the input and the target
collapses redundant categories
collapses rows that have very small cell counts
nominal
ordinal


0 1
A 28 7
B 16 0
C 94 11
D 23 21
0 1
0 1
0 1
Merged: 2 levels 2 levels 2 levels
only univariate associations



Demo Collapsing the Levels of a Nominal Input, Part 1

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

* Create an output data set that contains the proportion of events and the number of cases in each branch.

* Cluster the branches based on the reduction of the chi-square statistic using Greenacre's method.

pmlr03d02.sas

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

title1 "Proportion of Events by Level";
proc print data=work.level;
run;
```

FREQ

Proportion of Events by Level

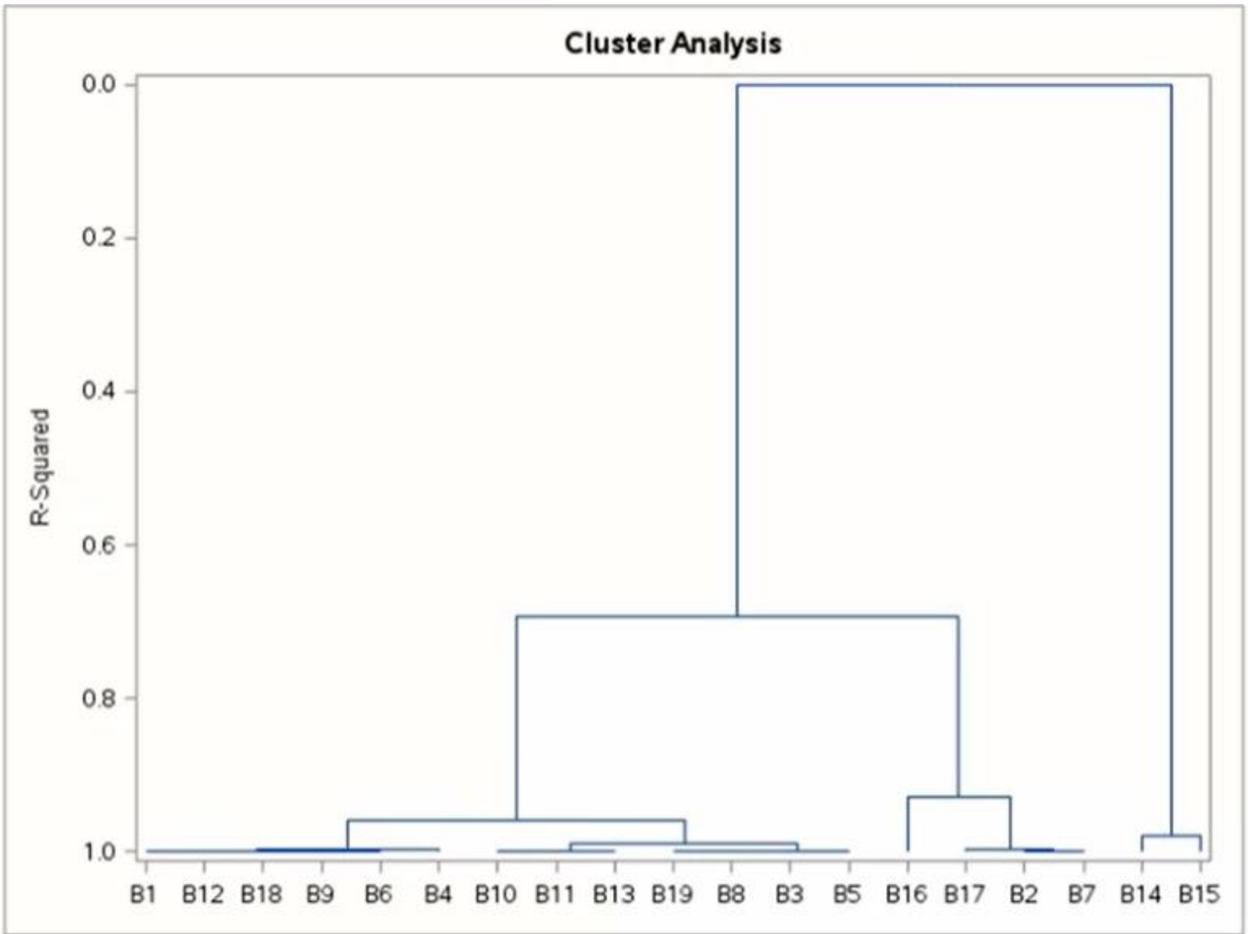
Obs	Branch	_TYPE_	_FREQ_	prop
1	B1	1	1930	0.36995
2	B10	1	182	0.41758
3	B11	1	160	0.41875
4	B12	1	368	0.36957
5	B13	1	369	0.40650
6	B14	1	712	0.19663
7	B15	1	1510	0.23179
8	B16	1	1040	0.28558
9	B17	1	544	0.34007
10	B18	1	370	0.36757
11	B19	1	191	0.38743
12	B2	1	3543	0.32882

```
/* Use ODS to output the ClusterHistory output object into a data set
   named "cluster." */

ods output clusterhistory=work.cluster;

proc cluster data=work.level method=ward outtree=work.fortree
        plots=(dendrogram(vertical height=rsq));
   freq _freq_;
   var prop;
   id branch;
run;
```

Cluster History						
Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Tie
18	B1	B12	2298	0.0000	1.00	
17	B18	B9	738	0.0000	1.00	
16	B10	B11	342	0.0000	1.00	
15	B19	B8	1069	0.0000	1.00	
14	CL17	B6	1676	0.0000	1.00	
13	B2	B7	4491	0.0000	1.00	
12	CL15	B3	2974	0.0001	1.00	
11	CL18	CL14	3974	0.0002	1.00	
10	CL16	B13	711	0.0004	.999	
9	CL12	B5	4793	0.0006	.999	
8	CL11	B4	7711	0.0008	.998	
7	B17	CL13	5035	0.0012	.997	
6	CL10	CL9	5504	0.0072	.989	
5	B14	B15	2222	0.0106	.979	
4	CL8	CL6	13215	0.0204	.958	
3	B16	CL7	6075	0.0297	.929	
2	CL4	CL3	19290	0.2350	.694	
1	CL2	CL5	21512	0.6937	.000	




```
/* Run this code before demo l3d2a */


/* ===================================================== */
/* Lesson 1, Section 1: l1d1.sas
   Demonstration: Examining the Code for Generating
   Descriptive Statistics and Frequency Tables        */
/* ===================================================== */


data work.develop;
   set pmlr.develop;
run;


%global inputs;
```

```
%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;

proc means data=work.develop n nmiss mean min max;
  var &inputs;
run;

proc freq data=work.develop;
  tables ins branch res;
run;

/* ===================================================== */
/* Lesson 1, Section 2: l1d2.sas
  Demonstration: Splitting the Data   */
/* ===================================================== */


/* Sort the data by the target in preparation for stratified sampling. */

proc sort data=work.develop out=work.develop_sort;
  by ins;
run;
```

```
/* The SURVEYSELECT procedure will perform stratified sampling
  on any variable in the STRATA statement. The OUTALL option
  specifies that you want a flag appended to the file to
  indicate selected records, not simply a file comprised
  of the selected records. */

proc surveyselect noprint data=work.develop_sort
            samprate=.6667 stratumseed=restore
            out=work.develop_sample
            seed=44444 outall;
  strata ins;
run;

/* Verify stratification. */

proc freq data=work.develop_sample;
  tables ins*selected;
run;

/* Create training and validation data sets. */

data work.train(drop=selected SelectionProb SamplingWeight)
   work.valid(drop=selected SelectionProb SamplingWeight);
  set work.develop_sample;
  if selected then output work.train;
  else output work.valid;
run;
```

```
/* ======================================================= */
/* Lesson 2, Section 1: l2d1.sas
   Demonstration: Fitting a Basic Logistic
   Regression Model, Parts 1 and 2                    */
/* ======================================================= */

title1 "Logistic Regression Model for the Variable Annuity Data Set";
proc logistic data=work.train
          plots(only maxpoints=none)=(effect(clband x=(ddabal depamt checks res))
          oddsratio (type=horizontalstat));
   class res (param=ref ref='S') dda (param=ref ref='0');
   model ins(event='1')=dda ddabal dep depamt
           cashbk checks res / stb clodds=pl;
   units ddabal=1000 depamt=1000 / default=1;
   oddsratio 'Comparisons of Residential Classification' res / diff=all cl=pl;
   effectplot slicefit(sliceby=dda x=ddabal) / noobs;
   effectplot slicefit(sliceby=dda x=depamt) / noobs;
run;
title1;


/* ======================================================= */
/* Lesson 2, Section 1: l2d2.sas
   Demonstration: Scoring New Cases                   */
/* ======================================================= */

/* Score a new data set with one run of the LOGISTIC procedure with the
   SCORE statement.  */
```

```
proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
  score data = pmlr.new out=work.scored1;
run;

title1 "Predicted Probabilities from Scored Data Set";
proc print data=work.scored1(obs=10);
  var p_1 dda ddabal dep depamt cashbk checks res;
run;

title1 "Mean of Predicted Probabilities from Scored Data Set";
proc means data=work.scored1 mean nolabels;
  var p_1;
run;

/* Score a new data set with the OUTMODEL= amd INMODEL= options */

proc logistic data=work.train outmodel=work.scoredata noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
run;

proc logistic inmodel=work.scoredata noprint;
  score data = pmlr.new out=work.scored2;
run;

title1 "Predicted Probabilities from Scored Data Set";
```

```
proc print data=work.scored2(obs=10);
  var p_1 dda ddabal dep depamt cashbk checks res;
run;

/* Score a new data set with the CODE Statement  */

proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
  code file="&PMLRfolder/pmlr_score.txt";
run;

data work.scored3;
  set pmlr.new;
  %include "&PMLRfolder/pmlr_score.txt";
run;

title1 "Predicted Probabilities from Scored Data Set";
proc print data=work.scored3(obs=10);
  var p_ins1 dda ddabal dep depamt cashbk checks res;
run;
title1 ;



/* ===================================================== */
/* Lesson 2, Section 2: l2d3.sas
  Demonstration: Correcting for Oversampling          */
/* ===================================================== */
```

```
/* Specify the prior probability to correct for oversampling.  */
%global pi1;
%let pi1=.02;

/* Correct predicted probabilities */

proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')=dda ddabal dep depamt cashbk checks res;
  score data=pmlr.new out=work.scored4 priorevent=&pi1;
run;

title1 "Adjusted Predicted Probabilities from Scored Data Set";
proc print data=work.scored4(obs=10);
  var p_1 dda ddabal dep depamt cashbk checks res;
run;

title1 "Mean of Adjusted Predicted Probabilities from Scored Data Set";
proc means data=work.scored4 mean nolabels;
  var p_1;
run;
title1 ;

/* Correct probabilities in the Score Code */

proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')=dda ddabal dep depamt cashbk checks res;
  /* File suffix "txt" is used so you can view the file */
```

```
  /* with a native text editor. SAS prefers "sas", but  */
  /* when specified as a filename, SAS does not care.   */
  code file="&PMLRfolder/pmlr_score_adj.txt";
run;

%global rho1;
proc SQL noprint;
  select mean(INS) into :rho1
  from work.train;
quit;

data new;
  set pmlr.new;
  off=log(((1-&pi1)*&rho1)/(&pi1*(1-&rho1)));
run;

data work.scored5;
  set work.new;
  %include "&PMLRfolder/pmlr_score_adj.txt";
  eta=log(p_ins1/p_ins0) - off;
  prob=1/(1+exp(-eta));
run;

title1 "Adjusted Predicted Probabilities from Scored Data Set";
proc print data=scored5(obs=10);
  var prob dda ddabal dep depamt cashbk checks res;
run;
title1 ;
```

```
/* ===================================================== */
/* Lesson 3, Section 1: l3d1.sas
  Demonstration: Imputing Missing Values
/* ===================================================== */

title1 "Variables with Missing Values";
proc print data=work.train(obs=15);
  var ccbal ccpurc income hmown;
run;
title1 ;

/* Create missing indicators */
data work.train_mi(drop=i);
  set work.train;
  /* 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}=.);
    nummiss+mi{i};
  end;
```

```
run;

/* Impute missing values with the median */
proc stdize data=work.train_mi reponly method=median out=work.train_imputed;
  var &inputs;
run;

title1 "Imputed Values with Missing Indicators";
proc print data=work.train_imputed(obs=12);
  var ccbal miccbal ccpurc miccpurc income miincome hmown mihmown nummiss;
run;
title1;

/* ===================================================== */
/* Lesson 3, Section 2: l3d2a.sas
  Demonstration: Collapsing the Levels of a Nominal Input,
  Part 1
  [m643_2_g1; derived from pmlr03d02.sas]             */
/* ===================================================== */

proc means data=work.train_imputed noprint nway;
  class branch;
  var ins;
  output out=work.level mean=prop;
run;

title1 "Proportion of Events by Level";
proc print data=work.level;
run;
```

```
/* Use ODS to output the ClusterHistory output object into a data set
  named "cluster." */

ods output clusterhistory=work.cluster;

proc cluster data=work.level method=ward outtree=work.fortree
       plots=(dendrogram(vertical height=rsq));
  freq _freq_;
  var prop;
  id branch;
run;
```

Demo Collapsing the Levels of a Nominal Input, Part 2

* Compute the log of the *p*-value for each cluster solution, and plot the log of the *p*-value by the number of clusters.
* Create an output data set that shows which branches were assigned to each cluster.
* Assign the branches to dummy variables.

pmlr03d02.sas

```
/* Use the FREQ procedure to get the Pearson Chi^2 statistic of the
   full BRANCH*INS table. */

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

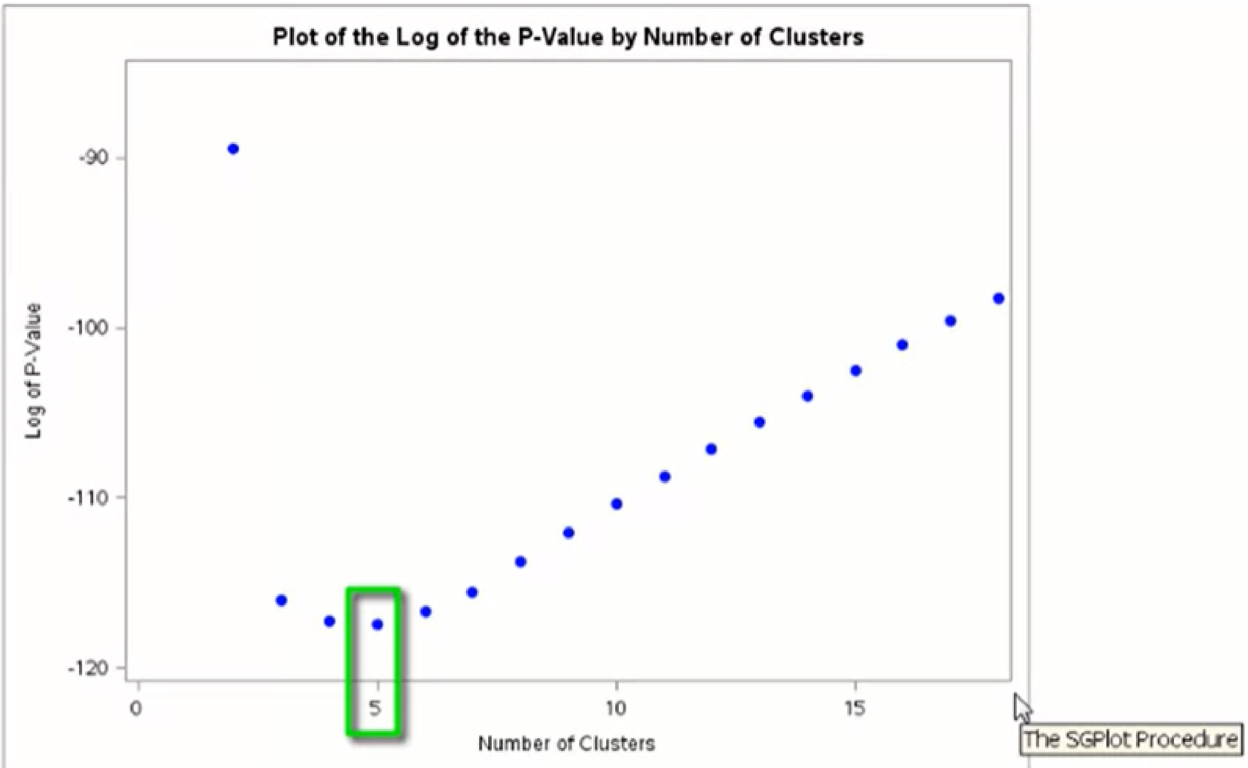
pmlr03d02.sas

```
/* Use a one-to-many merge to put the Chi^2 statistic onto the clustering
   results. Calculate a (log) p-value for each level of clustering. */

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

```
/* Plot the log p-values against number of clusters. */

title1 "Plot of the Log of the P-Value by Number of Clusters";
proc sgplot data=work.cutoff;
   scatter y=logpvalue x=numberofclusters
           / markerattrs=(color=blue symbol=circlefilled);
   xaxis label="Number of Clusters";
   yaxis label="Log of P-Value" min=-120 max=-85;
run;
title1 ;
```




```
/* Create a macro variable (&ncl) that contains the number of clusters
   associated with the minimum log p-value. */

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

Number of Clusters
5

```
proc tree data=work.fortree nclusters=&ncl out=work.clus noprint;
   id branch;
run;
```

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

```
title1 "Levels of Branch by Cluster";
proc print data=work.clus;
   by clusname;
   id clusname;I
run;
title1 ;
```

Levels of Branch by Cluster

CLUSNAME	Branch	CLUSTER
B16	B16	5

CLUSNAME	Branch	CLUSTER
CL5	B14	4
	B15	4

CLUSNAME	Branch	CLUSTER
CL6	B10	2
	B11	2
	B19	2
	B8	2
	B3	2

CLUSNAME	Branch	CLUSTER
CL7	B2	3
	B7	3
	B17	3

CLUSNAME	Branch	CLUSTER
CL8	B1	1
	B12	1
	B18	1
	B9	1
	B6	1
	B4	1

```
/* The DATA Step creates the scoring code to assign the branches to a clu

filename brclus "&PMLRfolder\branchclus.sas";

data _null_;
   file brclus;
   set work.clus end=last;
   if _n_=1 then put "select (branch);";
   put "  when ('" branch +(-1) "') branch_clus = '" cluster +(-1) "';";
   if last then do;
      put "  otherwise branch_clus = 'U';" / "end;";
   end;
run;
```

```
data work.train_imputed_greenacre;
   set work.train_imputed;
   %include brclus / source2;
run;
```

```
Log - (Untitled)
OTE: %INCLUDE (level 1) file BRCLUS is file S:\workshop\branchclus.sas.
561 +select (branch);
562 +  when ('B16') branch_clus = '5';
563 +  when ('B14') branch_clus = '4';
564 +  when ('B15') branch_clus = '4';
565 +  when ('B10') branch_clus = '2';
566 +  when ('B11') branch_clus = '2';
567 +  when ('B19') branch_clus = '2';
568 +  when ('B8') branch_clus = '2';
569 +  when ('B3') branch_clus = '2';
570 +  when ('B13') branch_clus = '2';
571 +  when ('B5') branch_clus = '2';
572 +  when ('B2') branch_clus = '3';
573 +  when ('B7') branch_clus = '3';
574 +  when ('B17') branch_clus = '3';
575 +  when ('B1') branch_clus = '1';
576 +  when ('B12') branch_clus = '1';
577 +  when ('B18') branch_clus = '1';
578 +  when ('B9') branch_clus = '1';
579 +  when ('B6') branch_clus = '1';
580 +  when ('B4') branch_clus = '1';
581 +  otherwise branch_clus = 'U';
582 +end;
```

```
561 +select (branch);
562 +  when ('B16') branch_clus = '5';
563 +  when ('B14') branch_clus = '4';
564 +  when ('B15') branch_clus = '4';
565 +  when ('B10') branch_clus = '2';
566 +  when ('B11') branch_clus = '2';
```

```
data _null_;
   file brclus;
   set work.clus end=last;
   if _n_=1 then put "select (branch);";
   put "  when ('" branch +(-1) "') branch_clus = '" cluster +(-1) "';";
   if last then do;
      put "  otherwise branch_clus = 'U';" / "end;";
   end;
run;
```

/* Run this code before demo l3d2b */

/* === */

/* Lesson 1, Section 1: l1d1.sas

Demonstration: Examining the Code for Generating

Descriptive Statistics and Frequency Tables */

/* === */

```
data work.develop;
  set pmlr.develop;
run;

%global inputs;
%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;

proc means data=work.develop n nmiss mean min max;
  var &inputs;
run;

proc freq data=work.develop;
  tables ins branch res;
run;

/* ===================================================== */
/* Lesson 1, Section 2: l1d2.sas
  Demonstration: Splitting the Data   */
/* ===================================================== */


/* Sort the data by the target in preparation for stratified sampling. */
```

```
proc sort data=work.develop out=work.develop_sort;
  by ins;
run;

/* The SURVEYSELECT procedure will perform stratified sampling
  on any variable in the STRATA statement. The OUTALL option
  specifies that you want a flag appended to the file to
  indicate selected records, not simply a file comprised
  of the selected records. */

proc surveyselect noprint data=work.develop_sort
            samprate=.6667 stratumseed=restore
            out=work.develop_sample
            seed=44444 outall;
  strata ins;
run;

/* Verify stratification. */

proc freq data=work.develop_sample;
  tables ins*selected;
run;

/* Create training and validation data sets. */

data work.train(drop=selected SelectionProb SamplingWeight)
   work.valid(drop=selected SelectionProb SamplingWeight);
  set work.develop_sample;
```

```
  if selected then output work.train;
  else output work.valid;
run;




/* ===================================================== */
/* Lesson 2, Section 1: l2d1.sas
   Demonstration: Fitting a Basic Logistic
   Regression Model, Parts 1 and 2                    */
/* ===================================================== */

title1 "Logistic Regression Model for the Variable Annuity Data Set";
proc logistic data=work.train
         plots(only maxpoints=none)=(effect(clband x=(ddabal depamt checks res))
         oddsratio (type=horizontalstat));
  class res (param=ref ref='S') dda (param=ref ref='0');
  model ins(event='1')=dda ddabal dep depamt
          cashbk checks res / stb clodds=pl;
  units ddabal=1000 depamt=1000 / default=1;
  oddsratio 'Comparisons of Residential Classification' res / diff=all cl=pl;
  effectplot slicefit(sliceby=dda x=ddabal) / noobs;
  effectplot slicefit(sliceby=dda x=depamt) / noobs;
run;
title1;



/* ===================================================== */
/* Lesson 2, Section 1: l2d2.sas
```

```
  Demonstration: Scoring New Cases                    */
/* ====================================================== */

/* Score a new data set with one run of the LOGISTIC procedure with the
  SCORE statement.  */

proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
  score data = pmlr.new out=work.scored1;
run;

title1 "Predicted Probabilities from Scored Data Set";
proc print data=work.scored1(obs=10);
  var p_1 dda ddabal dep depamt cashbk checks res;
run;

title1 "Mean of Predicted Probabilities from Scored Data Set";
proc means data=work.scored1 mean nolabels;
  var p_1;
run;

/* Score a new data set with the OUTMODEL= amd INMODEL= options */

proc logistic data=work.train outmodel=work.scoredata noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
run;
```

```
proc logistic inmodel=work.scoredata noprint;
  score data = pmlr.new out=work.scored2;
run;

title1 "Predicted Probabilities from Scored Data Set";
proc print data=work.scored2(obs=10);
  var p_1 dda ddabal dep depamt cashbk checks res;
run;

/* Score a new data set with the CODE Statement  */

proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')= res dda ddabal dep depamt cashbk checks;
  code file="&PMLRfolder/pmlr_score.txt";
run;

data work.scored3;
  set pmlr.new;
  %include "&PMLRfolder/pmlr_score.txt";
run;

title1 "Predicted Probabilities from Scored Data Set";
proc print data=work.scored3(obs=10);
  var p_ins1 dda ddabal dep depamt cashbk checks res;
run;
title1 ;
```

```
/* ===================================================== */
/* Lesson 2, Section 2: l2d3.sas
   Demonstration: Correcting for Oversampling          */
/* ===================================================== */

/* Specify the prior probability to correct for oversampling.  */
%global pi1;
%let pi1=.02;

/* Correct predicted probabilities */

proc logistic data=work.train noprint;
   class res (param=ref ref='S');
   model ins(event='1')=dda ddabal dep depamt cashbk checks res;
   score data=pmlr.new out=work.scored4 priorevent=&pi1;
run;

title1 "Adjusted Predicted Probabilities from Scored Data Set";
proc print data=work.scored4(obs=10);
   var p_1 dda ddabal dep depamt cashbk checks res;
run;

title1 "Mean of Adjusted Predicted Probabilities from Scored Data Set";
proc means data=work.scored4 mean nolabels;
   var p_1;
run;
title1 ;

/* Correct probabilities in the Score Code */
```

```
proc logistic data=work.train noprint;
  class res (param=ref ref='S');
  model ins(event='1')=dda ddabal dep depamt cashbk checks res;
  /* File suffix "txt" is used so you can view the file */
  /* with a native text editor. SAS prefers "sas", but  */
  /* when specified as a filename, SAS does not care.   */
  code file="&PMLRfolder/pmlr_score_adj.txt";
run;

%global rho1;
proc SQL noprint;
  select mean(INS) into :rho1
  from work.train;
quit;

data new;
  set pmlr.new;
  off=log(((1-&pi1)*&rho1)/(&pi1*(1-&rho1)));
run;

data work.scored5;
  set work.new;
  %include "&PMLRfolder/pmlr_score_adj.txt";
  eta=log(p_ins1/p_ins0) - off;
  prob=1/(1+exp(-eta));
run;

title1 "Adjusted Predicted Probabilities from Scored Data Set";
```

```
proc print data=scored5(obs=10);
  var prob dda ddabal dep depamt cashbk checks res;
run;
title1 ;


/* ====================================================== */
/* Lesson 3, Section 1: l3d1.sas
  Demonstration: Imputing Missing Values
/* ====================================================== */

title1 "Variables with Missing Values";
proc print data=work.train(obs=15);
  var ccbal ccpurc income hmown;
run;
title1 ;

/* Create missing indicators */
data work.train_mi(drop=i);
  set work.train;
  /* 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}=.);
    nummiss+mi{i};
  end;
run;

/* Impute missing values with the median */
proc stdize data=work.train_mi reponly method=median out=work.train_imputed;
  var &inputs;
run;

title1 "Imputed Values with Missing Indicators";
proc print data=work.train_imputed(obs=12);
  var ccbal miccbal ccpurc miccpurc income miincome hmown mihmown nummiss;
run;
title1;



/* ===================================================== */
/* Lesson 3, Section 2: l3d2a.sas
  Demonstration: Collapsing the Levels of a
  Nominal Input, Part 1                         */
/* ===================================================== */

proc means data=work.train_imputed noprint nway;
  class branch;
  var ins;
```

```
  output out=work.level mean=prop;
run;

title1 "Proportion of Events by Level";
proc print data=work.level;
run;

/* Use ODS to output the ClusterHistory output object into a data set
  named "cluster." */

ods output clusterhistory=work.cluster;

proc cluster data=work.level method=ward outtree=work.fortree
     plots=(dendrogram(vertical height=rsq));
  freq _freq_;
  var prop;
  id branch;
run;

/* =================================================== */
/* Lesson 3, Section 2: l3d2b.sas
  Demonstration: Collapsing the Levels of a Nominal Input,
  Part 2
  [m643_2_g2; derived from pmlr03d02.sas]          */
/* =================================================== */

/* Use the FREQ procedure to get the Pearson Chi^2 statistic of the
  full BRANCH*INS table. */
```

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

/* Use a one-to-many merge to put the Chi^2 statistic onto the clustering
  results. Calculate a (log) p-value for each level of clustering. */

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

/* Plot the log p-values against number of clusters. */

title1 "Plot of the Log of the P-Value by Number of Clusters";
proc sgplot data=work.cutoff;
  scatter y=logpvalue x=numberofclusters
       / markerattrs=(color=blue symbol=circlefilled);
  xaxis label="Number of Clusters";
  yaxis label="Log of P-Value" min=-120 max=-85;
run;
title1 ;

/* Create a macro variable (&ncl) that contains the number of clusters
  associated with the minimum log p-value. */
```

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

proc tree data=work.fortree nclusters=&ncl out=work.clus noprint;
  id branch;
run;

proc sort data=work.clus;
  by clusname;
run;

title1 "Levels of Branch by Cluster";
proc print data=work.clus;
  by clusname;
  id clusname;
run;
title1 ;

/* The DATA Step creates the scoring code to assign the branches to a cluster. */

filename brclus "&PMLRfolder/branch_clus.sas";

data _null_;
  file brclus;
  set work.clus end=last;
```

```
  if _n_=1 then put "select (branch);";
  put "  when ('" branch +(-1) "') branch_clus = '" cluster +(-1) "';";
  if last then do;
    put "  otherwise branch_clus = 'U';" / "end;";
  end;
run;

data work.train_imputed_greenacre;
  set work.train_imputed;
  %include brclus / source2;
run;
-----------------------------------------------------------------------------------------
/* Run this code before doing practice l3p2 */

/* ===================================================== */
/* Lesson 1, Practice 1
  Practice: Exploring the Veterans' Organization Data
  Used in the Practices                    */
/* ===================================================== */

data pmlr.pva(drop=control_number
            MONTHS_SINCE_LAST_PROM_RESP
            FILE_AVG_GIFT
            FILE_CARD_GIFT);
  set pmlr.pva_raw_data;
  STATUS_FL=RECENCY_STATUS_96NK in("F","L");
  STATUS_ES=RECENCY_STATUS_96NK in("E","S");
  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;

proc contents data=pmlr.pva;
run;

proc means data=pmlr.pva mean nmiss max min;
  var _numeric_;
run;

proc freq data=pmlr.pva nlevels;
  tables _character_;
run;


/* ======================================================== */
/* Lesson 1, Practice 2
  Practice: Splitting the Data                    */
/* ======================================================== */

proc sort data=pmlr.pva out=work.pva_sort;
  by target_b;
```

```
run;

proc surveyselect noprint data=work.pva_sort
            samprate=0.5 out=pva_sample seed=27513
            outall stratumseed=restore;
  strata target_b;
run;

data pmlr.pva_train(drop=selected SelectionProb SamplingWeight)
   pmlr.pva_valid(drop=selected SelectionProb SamplingWeight);
  set work.pva_sample;
  if selected then output pmlr.pva_train;
  else output pmlr.pva_valid;
run;


/* ===================================================== */
/* Lesson 2, Practice 1
  Practice: Fitting a Logistic Regression Model       */
/* ===================================================== */

/* Modifications for your SAS software:
  -----------------------------------------------------
  (Optional) To avoid a warning in the log about the
  suppression of plots that have more than 5000
  observations, you can add the MAXPOINTS= option
  to the PROC LOGISTIC statement like this:
  plots(maxpoints=none only). Omitting the
  MAXPOINTS= option does not affect the results
```

```
  of the practices in this course.
*/

%global ex_pi1;
%let ex_pi1=0.05;

title1 "Logistic Regression Model of the Veterans' Organization Data";
proc logistic data=pmlr.pva_train plots(only)=
          (effect(clband x=(pep_star recent_avg_gift_amt
          frequency_status_97nk)) oddsratio (type=horizontalstat));
  class pep_star (param=ref ref='0');
  model target_b(event='1')=pep_star recent_avg_gift_amt
            frequency_status_97nk / clodds=pl;
  effectplot slicefit(sliceby=pep_star x=recent_avg_gift_amt) / noobs;
  effectplot slicefit(sliceby=pep_star x=frequency_status_97nk) / noobs;
  score data=pmlr.pva_train out=work.scopva_train priorevent=&ex_pi1;
run;

title1 "Adjusted Predicted Probabilities of the Veteran's Organization Data";
proc print data=work.scopva_train(obs=10);
  var p_1 pep_star recent_avg_gift_amt frequency_status_97nk;

run;
title;



/* ===================================================== */
/* Lesson 3, Practice 1
  Practice: Imputing Missing Values                */
```

```
/* ==================================================== */

data pmlr.pva_train_mi(drop=i);
  set pmlr.pva_train;
  /* 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}=.);
    nummiss+mi{i};
  end;
run;

proc rank data=pmlr.pva_train_mi out=work.pva_train_rank
       groups=3;
  var recent_response_prop recent_avg_gift_amt;
  ranks grp_resp grp_amt;
run;

proc sort data=work.pva_train_rank out=work.pva_train_rank_sort;
  by grp_resp grp_amt;
run;

proc stdize data=work.pva_train_rank_sort method=median
        reponly out=pmlr.pva_train_imputed;
  by grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
```

```
run;

options nolabel;
proc means data=pmlr.pva_train_imputed median;
  class grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;
options label;

/* solution for l3p2 */

/* step 2 */

proc means data=pmlr.pva_train_imputed noprint nway;
  class cluster_code;
  var target_b;
  output out=work.level mean=prop;
run;



/* step 3 */

ods output clusterhistory=work.cluster;

proc cluster data=work.level method=ward
        outtree=work.fortree
        plots=(dendrogram(horizontal height=rsq));
  freq _freq_;
  var prop;
```

```
  id cluster_code;
run;


/* step 4 */

proc freq data=pmlr.pva_train_imputed noprint;
  tables cluster_code*target_b / chisq;
  output out=work.chi(keep=_pchi_) chisq;
run;

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

title1 "Plot of the Log of the P-Value by Number of Clusters";
proc sgplot data=work.cutoff;
  scatter y=logpvalue x=numberofclusters
        / markerattrs=(color=blue symbol=circlefilled);
  xaxis label="Number of Clusters";
  yaxis label="Log of P-Value" min=-40 max=0;
run;
title1;
```

```
/* step 5 */

%global ncl;

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


/* step 6 */

proc tree data=work.fortree nclusters=&ncl
       out=work.clus noprint;
  id cluster_code;
run;

proc sort data=work.clus;
  by clusname;
run;

title1 "Cluster Assignments";
proc print data=work.clus;
  by clusname;
  id clusname;
run;
```

```
/* step 7 */

filename clcode "&PMLRfolder/cluster_code.sas";

data _null_;
  file clcode;
  set work.clus end=last;
  if _n_=1 then put "select (cluster_code);";
  put "  when ('" cluster_code +(-1) "') cluster_clus='" cluster +(-1) "';";
  if last then do;
    put "  otherwise cluster_clus='U';" / "end;";
  end;
run;

data pmlr.pva_train_imputed_clus;
  set pmlr.pva_train_imputed;
  %include clcode;
run;
```

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

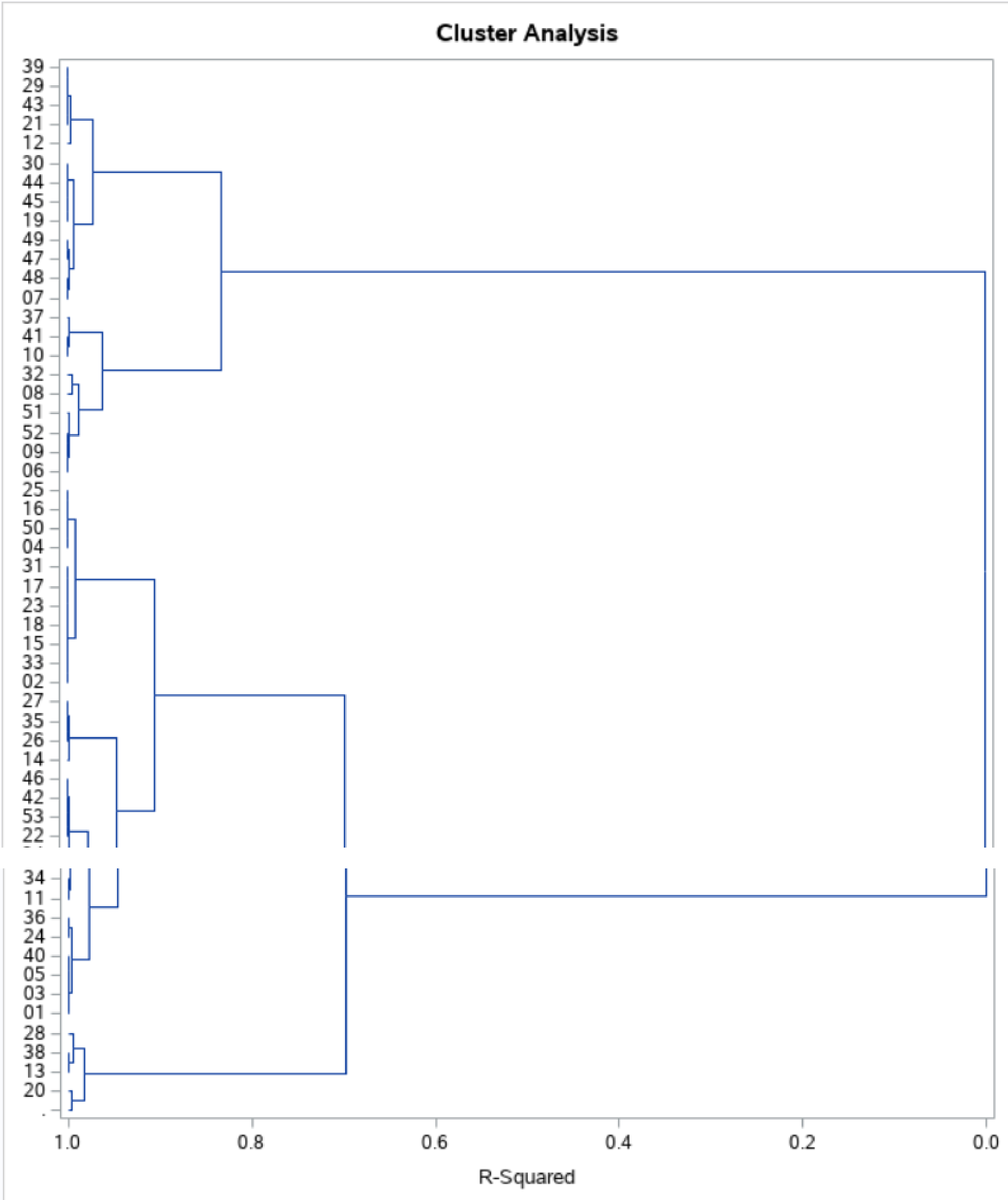
Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	0.00145225		1.0000	1.0000

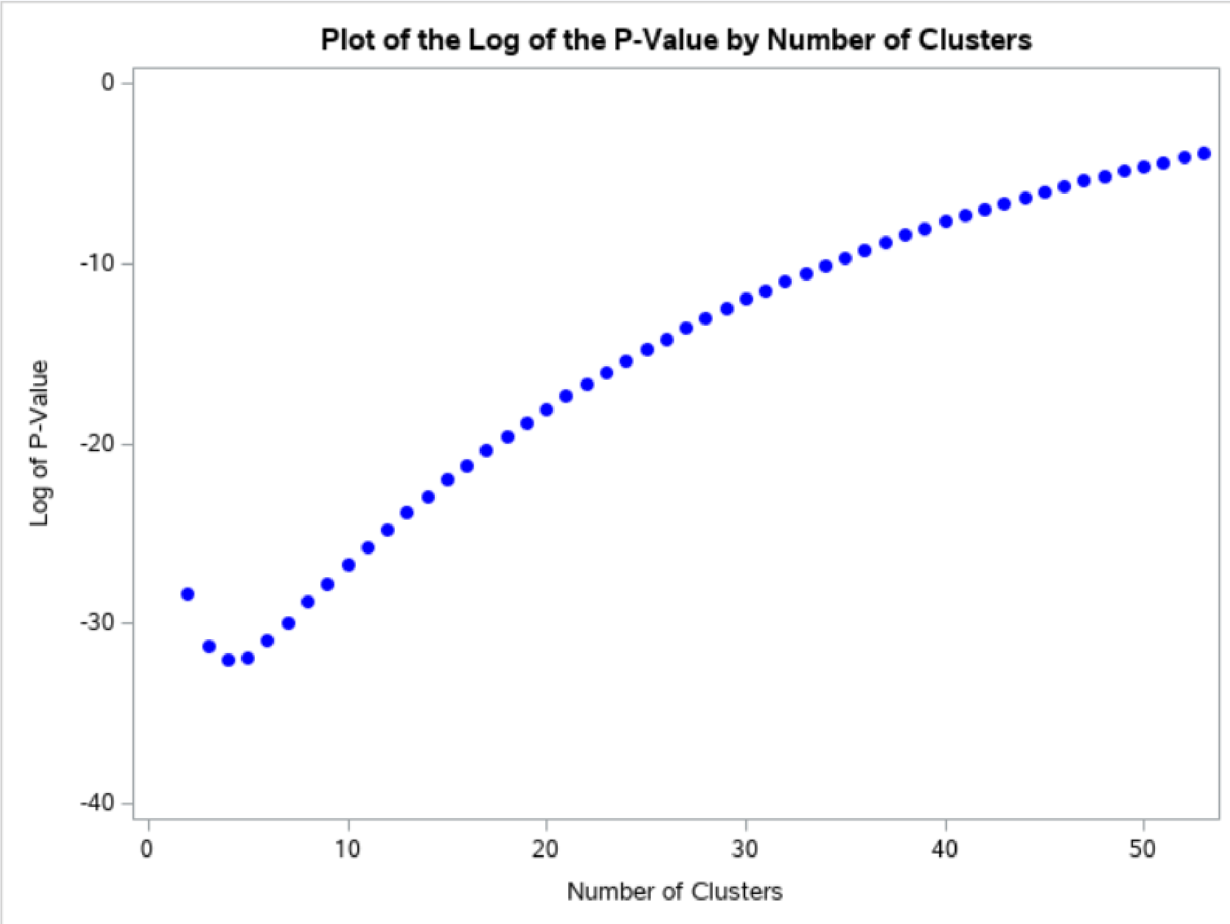
Root-Mean-Square Total-Sample Standard Deviation	0.038108

Root-Mean-Square Distance Between Observations	0.053893

Cluster History						
Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Tie
53	16	25	325	0.0000	1.00	
52	19	45	291	0.0000	1.00	
51	03	05	254	0.0000	1.00	
50	18	23	455	0.0000	1.00	
49	CL52	44	473	0.0000	1.00	
48	47	49	432	0.0000	1.00	
47	22	53	265	0.0000	1.00	
46	15	CL50	565	0.0000	1.00	
45	26	35	472	0.0000	1.00	
44	07	48	192	0.0000	1.00	
43	09	52	107	0.0000	1.00	
42	17	31	296	0.0000	1.00	
41	02	33	247	0.0000	1.00	
40	11	34	374	0.0000	1.00	
39	CL47	42	397	0.0000	1.00	

28	CL39	46	580	0.0000	1.00	
27	CL38	CL53	451	0.0001	1.00	
26	CL49	30	735	0.0001	1.00	
25	24	36	769	0.0001	.999	
24	CL37	CL36	751	0.0001	.999	
23	CL41	CL32	1108	0.0002	.999	
22	CL34	51	413	0.0002	.999	
21	CL44	CL48	624	0.0002	.999	
20	CL33	37	521	0.0003	.999	
19	CL40	CL28	954	0.0003	.998	
18	14	CL31	1045	0.0005	.998	
17	.	20	407	0.0006	.997	
16	CL30	CL25	1532	0.0006	.997	
15	12	CL24	1062	0.0007	.996	
14	CL29	28	574	0.0008	.995	
13	08	32	261	0.0009	.994	
12	CL21	CL26	1359	0.0014	.993	
11	CL23	CL27	1559	0.0021	.991	
10	CL22	CL13	674	0.0032	.987	
9	CL17	CL14	981	0.0043	.983	
8	CL16	CL19	2486	0.0049	.978	
7	CL12	CL15	2421	0.0054	.973	
6	CL10	CL20	1195	0.0107	.962	
5	CL8	CL18	3531	0.0159	.946	
4	CL5	CL11	5090	0.0409	.905	
3	CL6	CL7	3616	0.0722	.833	
2	CL9	CL4	6071	0.1358	.697	
1	CL2	CL3	9687	0.6973	.000	


Cluster Analysis
39
29
43
21
12
30
44
45
19
49
47
48
07
37
41
10
32
08
51
52
09
06
25
16
50
04
31
17
23
18
15
33
02
27
35
26
14
46
42
53
22
34
11
36
24
40
05
03
01
28
38
13
20
1.0
0.8
0.6
0.4
0.2
0.0
R-Squared


Plot of the Log of the P-Value by Number of Clusters
Log of P-Value
Number of Clusters
0
-10
-20
-30
-40
0
10
20
30
40
50

Number of Clusters
4

Cluster Assignments

CLUSNAME	CLUSTER_CODE	CLUSTER
CL4	16	1
	25	1
	03	1
	05	1
	18	1
	23	1
	22	1
	53	1
	15	1
	26	1
	35	1
	17	1
	31	1
	02	1
	33	1
	11	1
	34	1
	42	1
	04	1
	50	1
	40	1
	27	1
	01	1
	46	1
	24	1
	36	1
	14	1

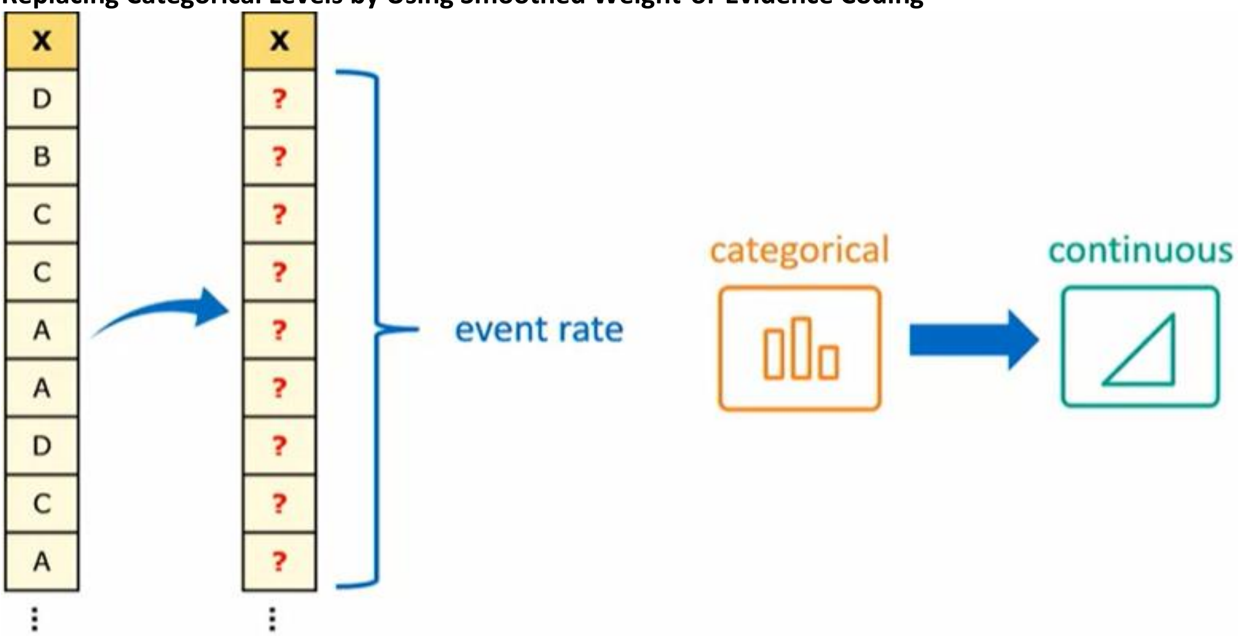
CLUSNAME	CLUSTER_CODE	CLUSTER
CL6	09	3
	52	3
	06	3
	10	3
	41	3
	51	3
	37	3
	08	3
	32	3

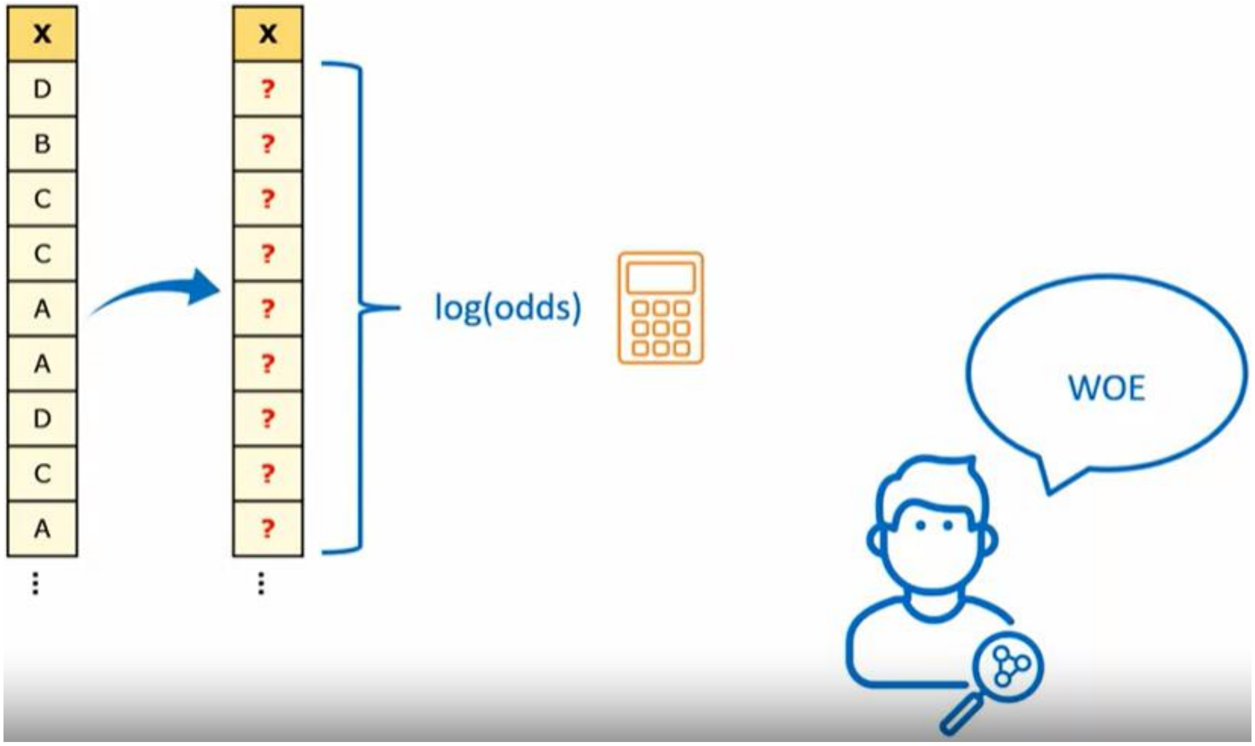
CLUSNAME	CLUSTER_CODE	CLUSTER
CL7	19	2
	45	2
	44	2
	47	2
	49	2
	07	2
	48	2
	21	2

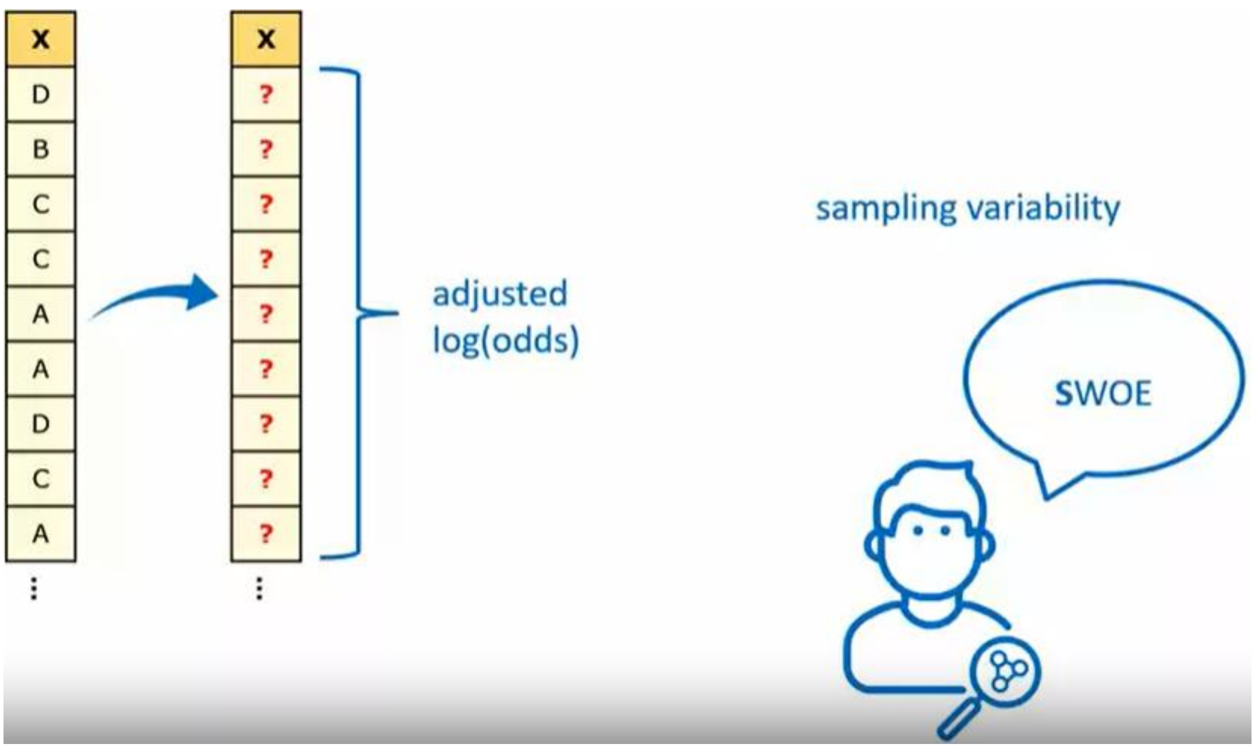
	43	2
	29	2
	39	2
	30	2
	12	2

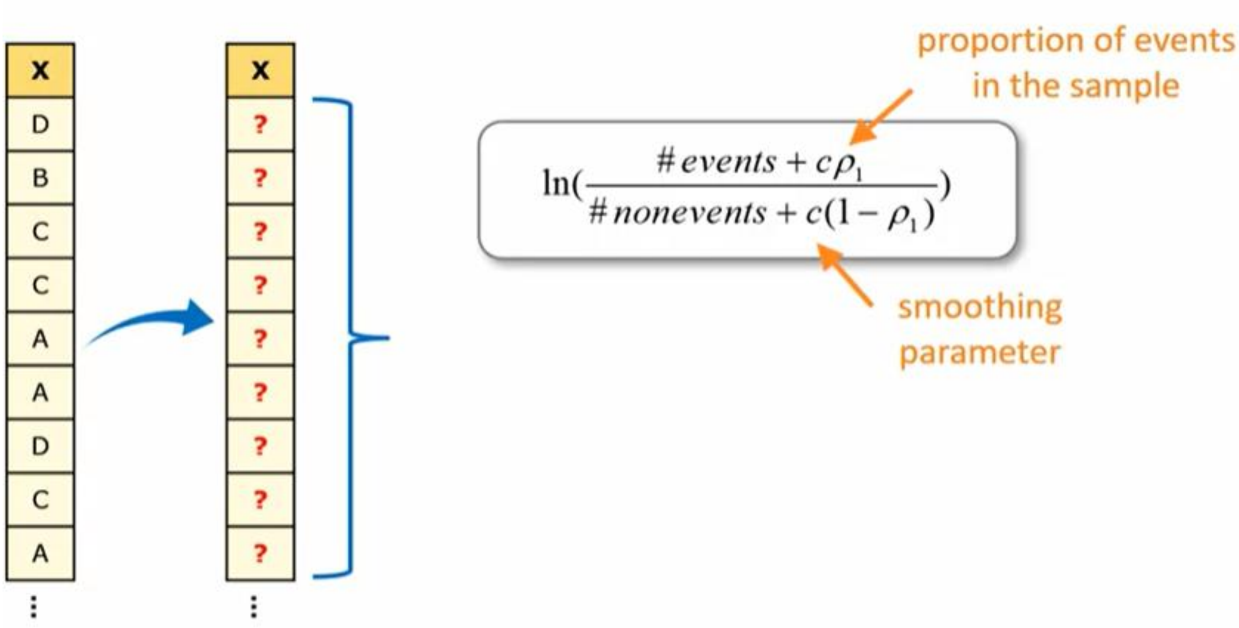
CLUSNAME	CLUSTER_CODE	CLUSTER
CL9	13	4
	38	4
	.	4
	20	4
	28	4

Replacing Categorical Levels by Using Smoothed Weight-of-Evidence Coding

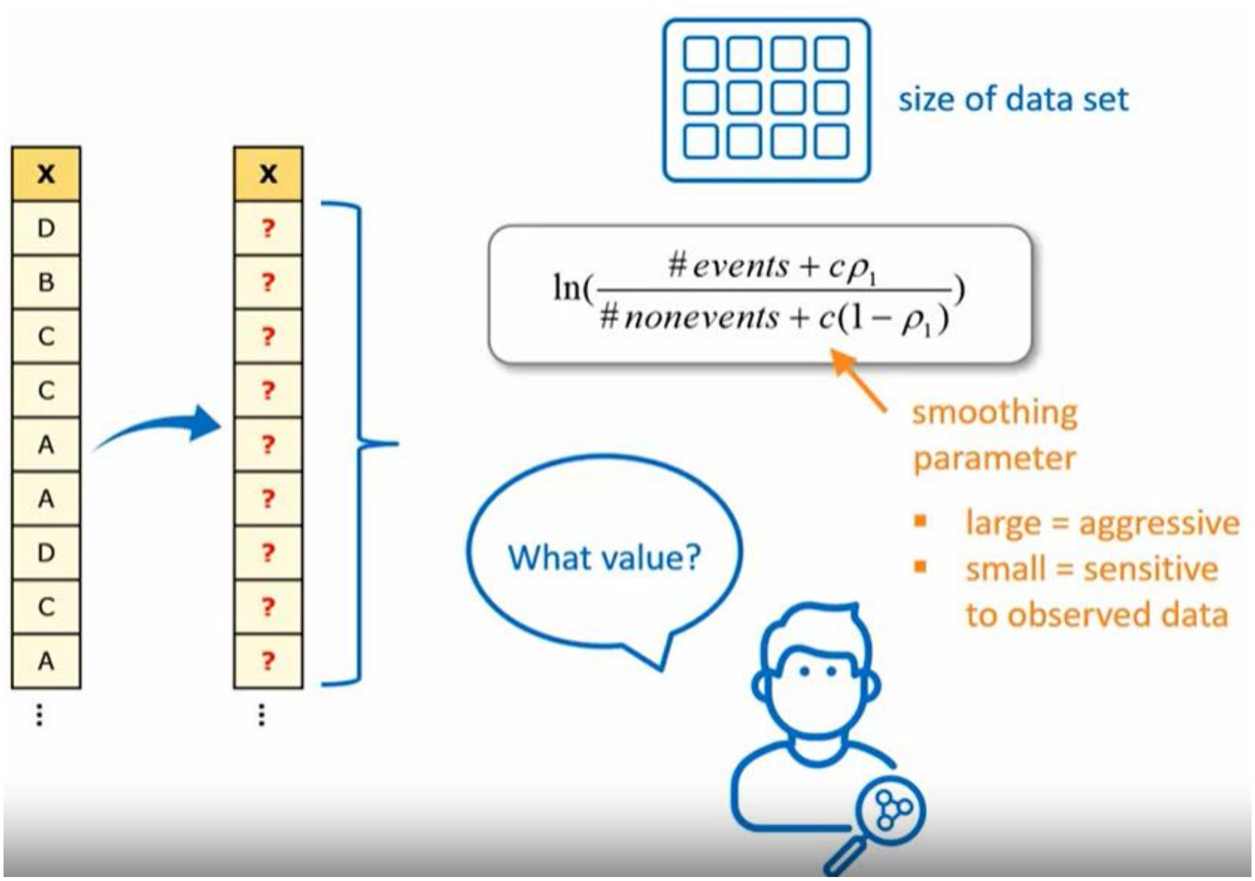



X
D
B
C
C
A
A
D
C
A
X
?
?
?
?
?
?
?
?
?
log(odds)
WOE


X
D
B
C
C
A
A
D
C
A
X
?
?
?
?
?
?
?
?
?
adjusted
log(odds)
sampling variability
SWOE



X
D
B
C
C
A
A
D
C
A
X
proportion of events in the sample
$$\ln\left(\frac{\#events + c\rho_1}{\#nonevents + c(1-\rho_1)}\right)$$
smoothing parameter



size of data set
X
D
B
C
C
A
A
D
C
A
X
$$\ln\left(\frac{\#events + c\rho_1}{\#nonevents + c(1-\rho_1)}\right)$$
smoothing parameter
large = aggressive
small = sensitive to observed data
What value?

	0	1	SWOE
A	28	7	-1.399
B	16	0	$\boxed{-2.021}$
C	94	11	-1.978
D	23	21	-0.499

$C = 24$

$\rho_1 = 0.195$

$$\ln\left(\frac{\#events + c\rho_1}{\#nonevents + c(1-\rho_1)}\right) = \ln\left(\frac{0 + 24 * 0.195}{16 + 24(1 - 0.195)}\right)$$

Demo Computing the Smoothed Weight of Evidence

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

pmlr03d03.sas

```
/* Rho1 is the proportion of events in the training data set. */
%global rho1;
proc sql noprint;
   select mean(ins) into :rho1
   from work.train_imputed;
quit;

/* The output data set from PROC MEANS will have the number of
   observations and events for each level of branch. */

proc means data=work.train_imputed sum nway noprint;
   class branch;
   var ins;
   output out=work.counts sum=events;
run;
```

```
/* The DATA Step creates the scoring code that assigns each branch to
   a value of the smoothed weight of evidence. */


filename brswoe "&PMLRfolder\swoe_branch.sas";

data _null_;
   file brswoe;
   set work.counts end=last;
   logit=log((events + &rho1*24)/(_FREQ_ - events + (1-&rho1)*24));
   if _n_=1 then put "select (branch);" ;
   put "  when ('" branch +(-1) "') branch_swoe = " logit ";" ;
   if last then do;
   logit=log(&rho1/(1-&rho1));
   put "  otherwise branch_swoe = " logit ";" / "end;";
   end;
run;
```

```
data work.train_imputed_swoe;
   set work.train_imputed;
   %include brswoe / source2;
run;
```

swoe_branch.sas

```
select (branch);
  when ('B1') branch_swoe = -0.533682018 ;
  when ('B10') branch_swoe = -0.366920929 ;
  when ('B11') branch_swoe = -0.366824824 ;
  when ('B12') branch_swoe = -0.540183932 ;
  when ('B13') branch_swoe = -0.393681164 ;
  when ('B14') branch_swoe = -1.376871614 ;
  when ('B15') branch_swoe = -1.188201904 ;
  when ('B16') branch_swoe = -0.91025293 ;
  when ('B17') branch_swoe = -0.661782256 ;
  when ('B18') branch_swoe = -0.548226206 ;
  when ('B19') branch_swoe = -0.477468642 ;
  when ('B2') branch_swoe = -0.713003789 ;
```

```
  when ('B18') branch_swoe = -0.548226206 ;
  when ('B19') branch_swoe = -0.477468642 ;
  when ('B2') branch_swoe = -0.713003789 ;
  when ('B3') branch_swoe = -0.477918403 ;
  when ('B4') branch_swoe = -0.518845457 ;
  when ('B5') branch_swoe = -0.450637056 ;
  when ('B6') branch_swoe = -0.552908065 ;
  when ('B7') branch_swoe = -0.719594589 ;
  when ('B8') branch_swoe = -0.468178724 ;
  when ('B9') branch_swoe = -0.551166668 ;
  otherwise branch_swoe = -0.635055959 ;
end;
```

```
/* Run this code before doing practice l3p3 */

/* ===================================================== */
/* Lesson 1, Practice 1
   Practice: Exploring the Veterans' Organization Data
   Used in the Practices                              */
/* ===================================================== */

data pmlr.pva(drop=control_number
             MONTHS_SINCE_LAST_PROM_RESP
             FILE_AVG_GIFT
             FILE_CARD_GIFT);
   set pmlr.pva_raw_data;
   STATUS_FL=RECENCY_STATUS_96NK in("F","L");
   STATUS_ES=RECENCY_STATUS_96NK in("E","S");
   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;

proc contents data=pmlr.pva;
run;
```

```
proc means data=pmlr.pva mean nmiss max min;
  var _numeric_;
run;

proc freq data=pmlr.pva nlevels;
  tables _character_;
run;


/* ===================================================== */
/* Lesson 1, Practice 2
  Practice: Splitting the Data                */
/* ===================================================== */

proc sort data=pmlr.pva out=work.pva_sort;
  by target_b;
run;

proc surveyselect noprint data=work.pva_sort
            samprate=0.5 out=pva_sample seed=27513
            outall stratumseed=restore;
  strata target_b;
run;

data pmlr.pva_train(drop=selected SelectionProb SamplingWeight)
    pmlr.pva_valid(drop=selected SelectionProb SamplingWeight);
  set work.pva_sample;
  if selected then output pmlr.pva_train;
```

```
  else output pmlr.pva_valid;
run;



/* ===================================================== */
/* Lesson 2, Practice 1
  Practice: Fitting a Logistic Regression Model        */
/* ===================================================== */

/* Modifications for your SAS software:
  ---------------------------------------------------
  (Optional) To avoid a warning in the log about the
  suppression of plots that have more than 5000
  observations, you can add the MAXPOINTS= option
  to the PROC LOGISTIC statement like this:
  plots(maxpoints=none only). Omitting the
  MAXPOINTS= option does not affect the results
  of the practices in this course.
*/

%global ex_pi1;
%let ex_pi1=0.05;

title1 "Logistic Regression Model of the Veterans' Organization Data";
proc logistic data=pmlr.pva_train plots(only)=
          (effect(clband x=(pep_star recent_avg_gift_amt
          frequency_status_97nk)) oddsratio (type=horizontalstat));
  class pep_star (param=ref ref='0');
  model target_b(event='1')=pep_star recent_avg_gift_amt
```

```
            frequency_status_97nk / clodds=pl;
  effectplot slicefit(sliceby=pep_star x=recent_avg_gift_amt) / noobs;
  effectplot slicefit(sliceby=pep_star x=frequency_status_97nk) / noobs;
  score data=pmlr.pva_train out=work.scopva_train priorevent=&ex_pi1;
run;

title1 "Adjusted Predicted Probabilities of the Veteran's Organization Data";
proc print data=work.scopva_train(obs=10);
  var p_1 pep_star recent_avg_gift_amt frequency_status_97nk;

run;
title;



/* ===================================================== */
/* Lesson 3, Practice 1
  Practice: Imputing Missing Values                */
/* ===================================================== */

data pmlr.pva_train_mi(drop=i);
  set pmlr.pva_train;
  /* 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}=.);
    nummiss+mi{i};
```

```
  end;
run;

proc rank data=pmlr.pva_train_mi out=work.pva_train_rank
        groups=3;
  var recent_response_prop recent_avg_gift_amt;
  ranks grp_resp grp_amt;
run;

proc sort data=work.pva_train_rank out=work.pva_train_rank_sort;
  by grp_resp grp_amt;
run;

proc stdize data=work.pva_train_rank_sort method=median
          reponly out=pmlr.pva_train_imputed;
  by grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;

options nolabel;
proc means data=pmlr.pva_train_imputed median;
  class grp_resp grp_amt;
  var DONOR_AGE INCOME_GROUP WEALTH_RATING;
run;
options label;


/* ====================================================== */
/* Lesson 3, Practice 2
```

```
  Practice: Collapsing the Levels of a Nominal Input

  Note: After you submit this code, a note in the log
  indicates that argument 3 to the LOGSDF function
  is invalid. You can ignore this note; it is not
  important for this analysis. The note pertains
  to the situation in which the number of clusters is 1.
  In this case, the degrees of freedom is 0 (degrees of
  freedom is equal to the number of clusters minus 1) and
  the mathematical operation cannot be performed in the
  LOGSDF function. Therefore, the log of the p-value is
  set to missing.                          */
/* ===================================================== */

proc means data=pmlr.pva_train_imputed noprint nway;
  class cluster_code;
  var target_b;
  output out=work.level mean=prop;
run;

ods output clusterhistory=work.cluster;

proc cluster data=work.level method=ward
         outtree=work.fortree
         plots=(dendrogram(horizontal height=rsq));
  freq _freq_;
  var prop;
  id cluster_code;
run;
```

```
proc freq data=pmlr.pva_train_imputed noprint;
  tables cluster_code*target_b / chisq;
  output out=work.chi(keep=_pchi_) chisq;
run;

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

title1 "Plot of the Log of the P-Value by Number of Clusters";
proc sgplot data=work.cutoff;
  scatter y=logpvalue x=numberofclusters
        / markerattrs=(color=blue symbol=circlefilled);
  xaxis label="Number of Clusters";
  yaxis label="Log of P-Value" min=-40 max=0;
run;

title1;

%global ncl;

proc sql;
  select NumberOfClusters into :ncl
  from work.cutoff
```

```
  having logpvalue=min(logpvalue);
quit;

proc tree data=work.fortree nclusters=&ncl
        out=work.clus noprint;
  id cluster_code;
run;

proc sort data=work.clus;
  by clusname;
run;

title1 "Cluster Assignments";
proc print data=work.clus;
  by clusname;
  id clusname;
run;

filename clcode "&PMLRfolder/cluster_code.sas";

data _null_;
  file clcode;
  set work.clus end=last;
  if _n_=1 then put "select (cluster_code);";
  put "  when ('" cluster_code +(-1) "')
        cluster_clus='" cluster +(-1) "';";
  if last then do;
    put "  otherwise cluster_clus='U';" / "end;";
  end;
```

```
run;

data pmlr.pva_train_imputed_clus;
  set pmlr.pva_train_imputed;
  %include clcode;
run;
```

The MEANS Procedure

Variable	Mean	N Miss	Maximum	Minimum
TARGET_B	0.2500000	0	1.0000000	0
TARGET_D	15.6243444	14529	200.0000000	1.0000000
MONTHS_SINCE_ORIGIN	73.4099732	0	137.0000000	5.0000000
DONOR_AGE	58.9190506	4795	87.0000000	0
IN_HOUSE	0.0731984	0	1.0000000	0
INCOME_GROUP	3.9075434	4392	7.0000000	1.0000000
PUBLISHED_PHONE	0.4977287	0	1.0000000	0
MOR_HIT_RATE	3.3616560	0	241.0000000	0
WEALTH_RATING	5.0053967	8810	9.0000000	0

The FREQ Procedure

Number of Variable Levels	
Variable	Levels
URBANICITY	6
SES	5
CLUSTER_CODE	54
HOME_OWNER	2
DONOR_GENDER	4
OVERLAY_SOURCE	4
RECENCY_STATUS_96NK	6

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

RECENCY_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

Logistic Regression Model of the Veterans' Organization Data

The LOGISTIC Procedure

Model Information	
Data Set	PMLR.PVA_TRAIN
Response Variable	TARGET_B
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	9687
Number of Observations Used	9687

Response Profile		
Ordered Value	TARGET_B	Total Frequency
1	0	7265
2	1	2422

Probability modeled is TARGET_B=1.

Class Level Information		
Class	Value	Design Variables
PEP_STAR	0	0
	1	1

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

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	10897.230	10663.061
SC	10904.409	10691.776
-2 Log L	10895.230	10655.061

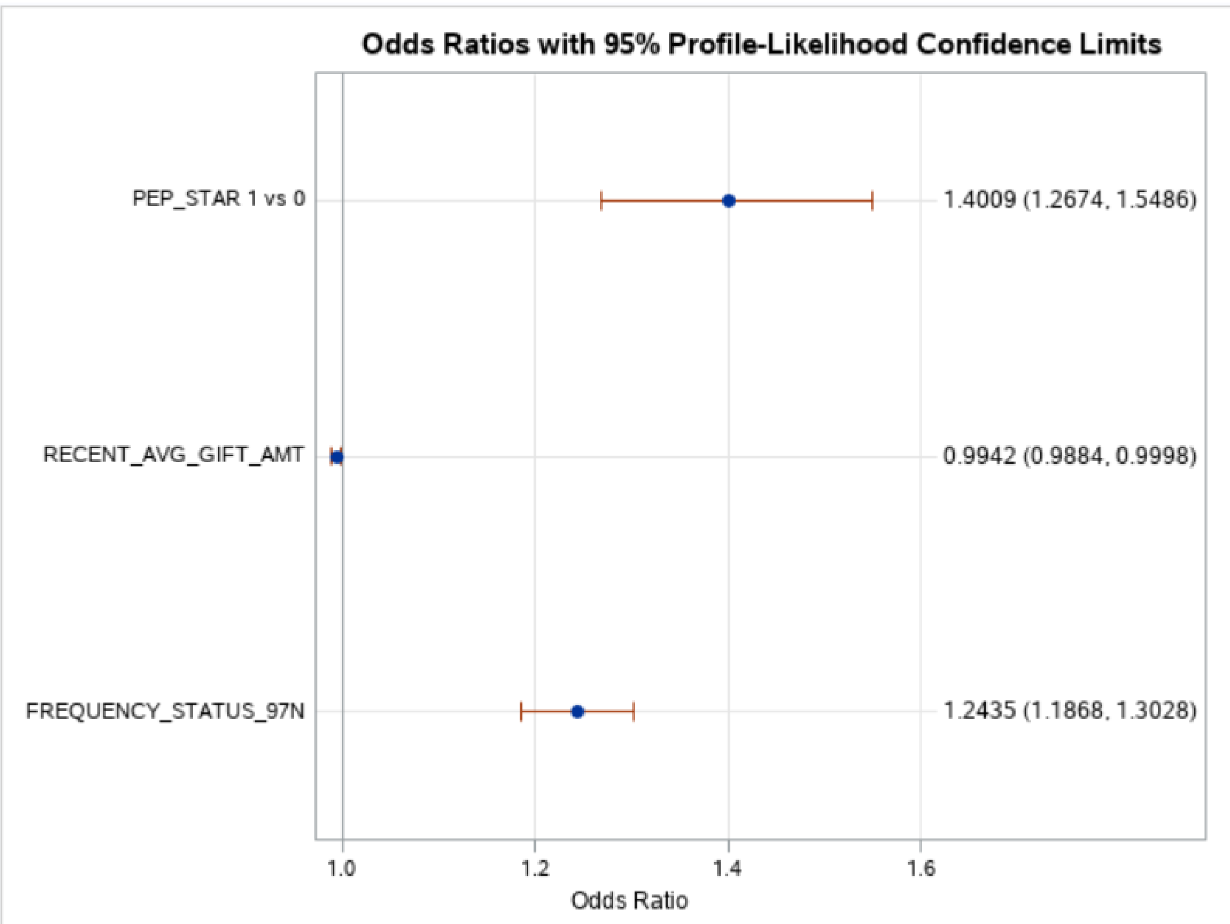
Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	240.1690	3	<.0001
Score	242.9486	3	<.0001
Wald	237.2875	3	<.0001

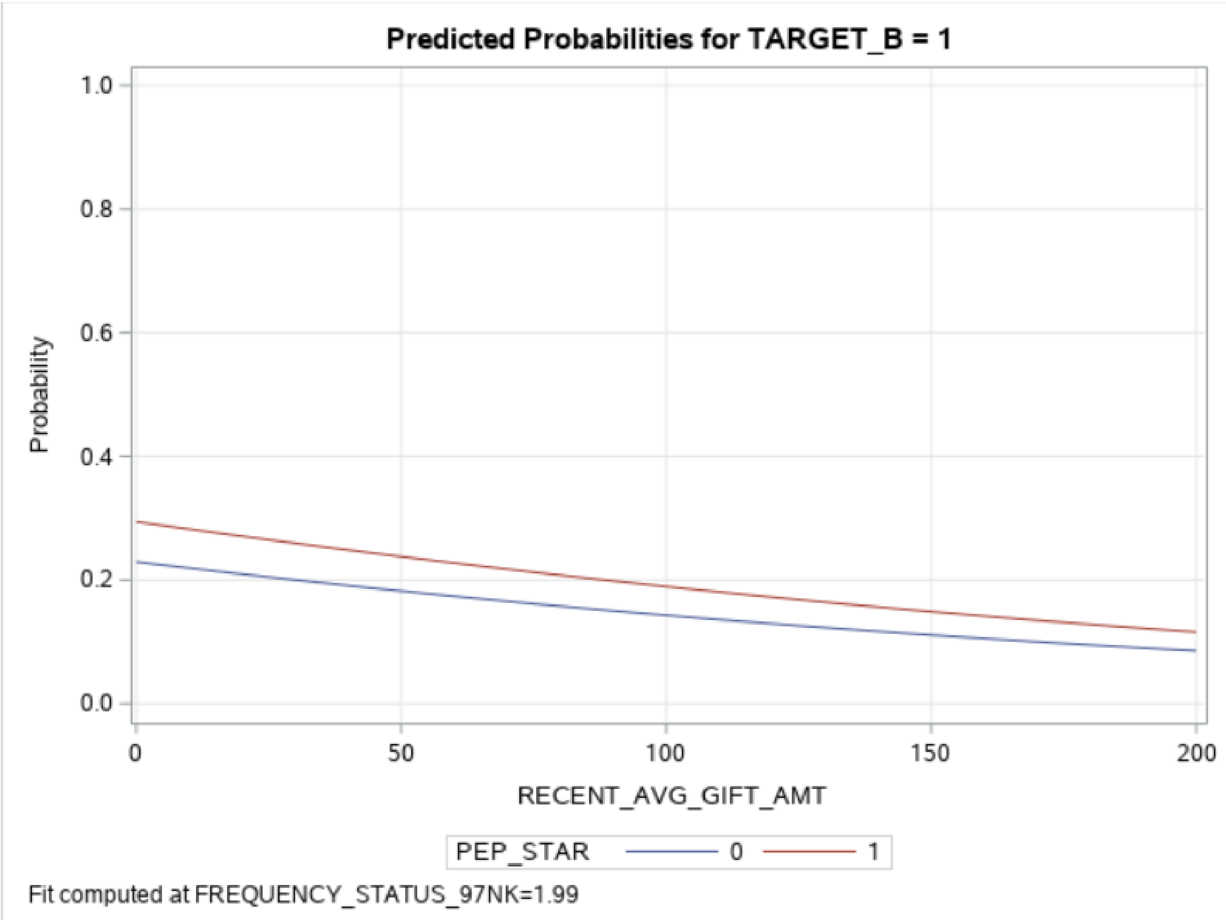
Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
PEP_STAR	1	43.4902	<.0001
RECENT_AVG_GIFT_AMT	1	3.9559	0.0467
FREQUENCY_STATUS_97N	1	83.8209	<.0001

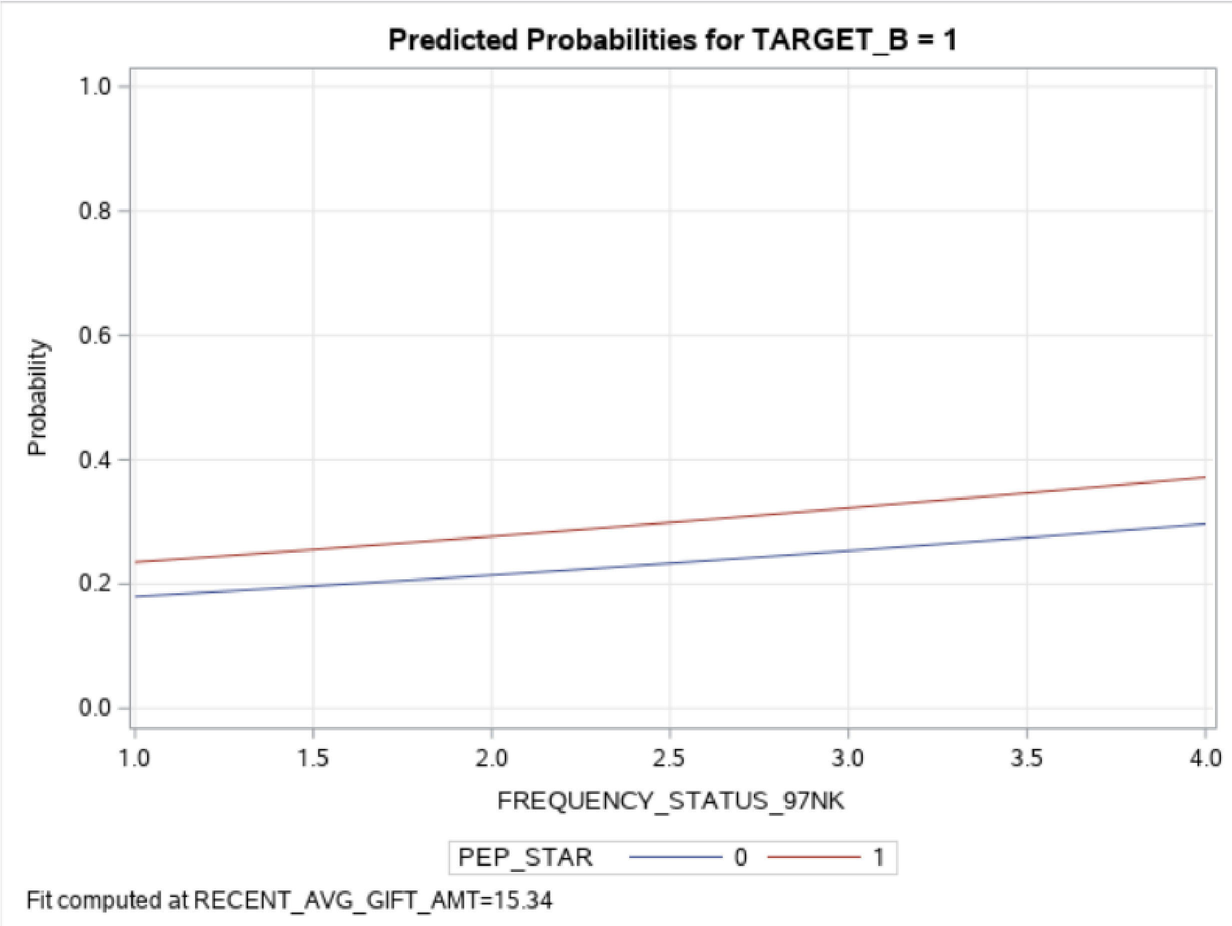
Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-1.6454	0.0831	392.4480	<.0001
PEP_STAR	1	1	0.3371	0.0511	43.4902	<.0001
RECENT_AVG_GIFT_AMT		1	-0.00579	0.00291	3.9559	0.0467
FREQUENCY_STATUS_97N		1	0.2179	0.0238	83.8209	<.0001

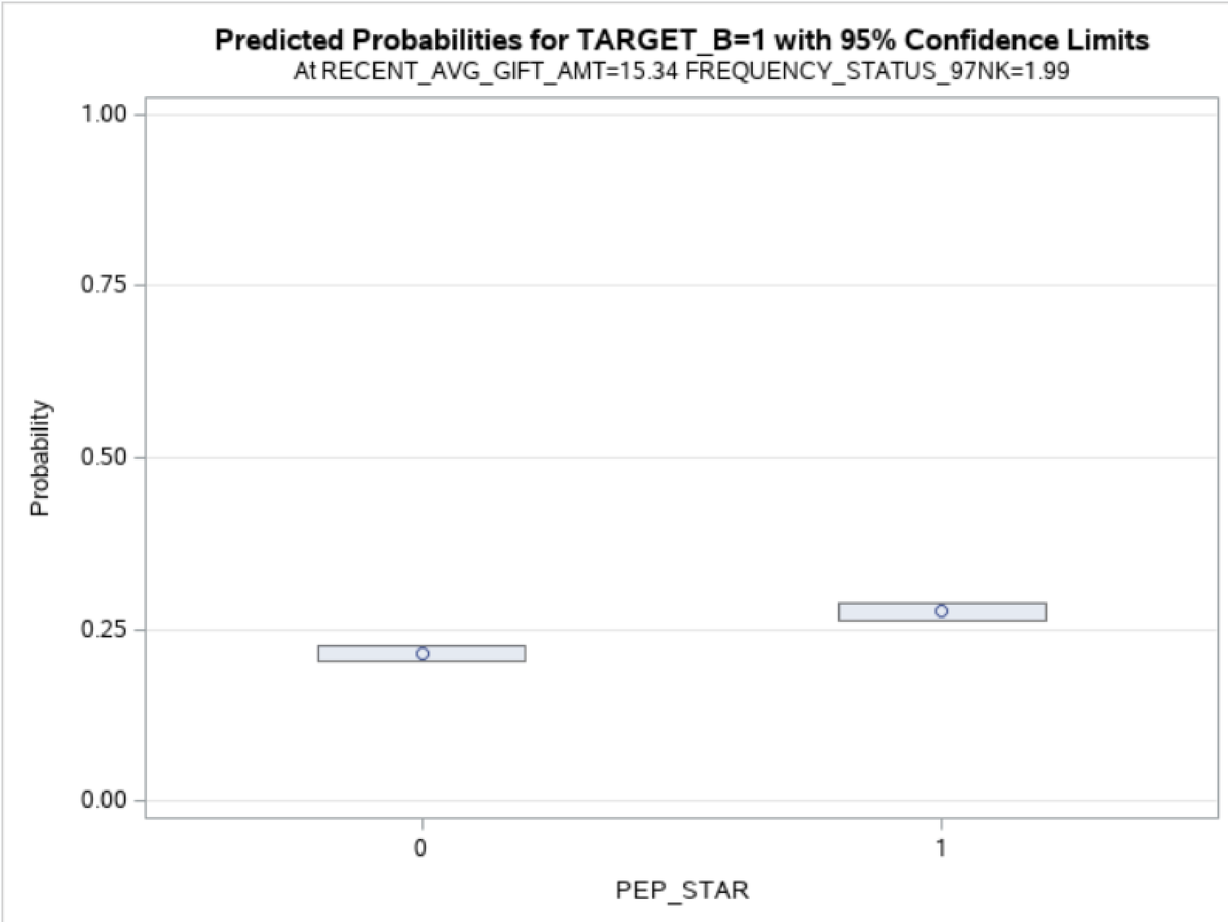
Association of Predicted Probabilities and Observed Responses			
Percent Concordant	59.9	Somers' D	0.208
Percent Discordant	39.0	Gamma	0.211
Percent Tied	1.1	Tau-a	0.078
Pairs	17595830	c	0.604

Odds Ratio Estimates and Profile-Likelihood Confidence Intervals				
Effect	Unit	Estimate	95% Confidence Limits	
PEP_STAR 1 vs 0	1.0000	1.401	1.267	1.549
RECENT_AVG_GIFT_AMT	1.0000	0.994	0.988	1.000
FREQUENCY_STATUS_97N	1.0000	1.243	1.187	1.303

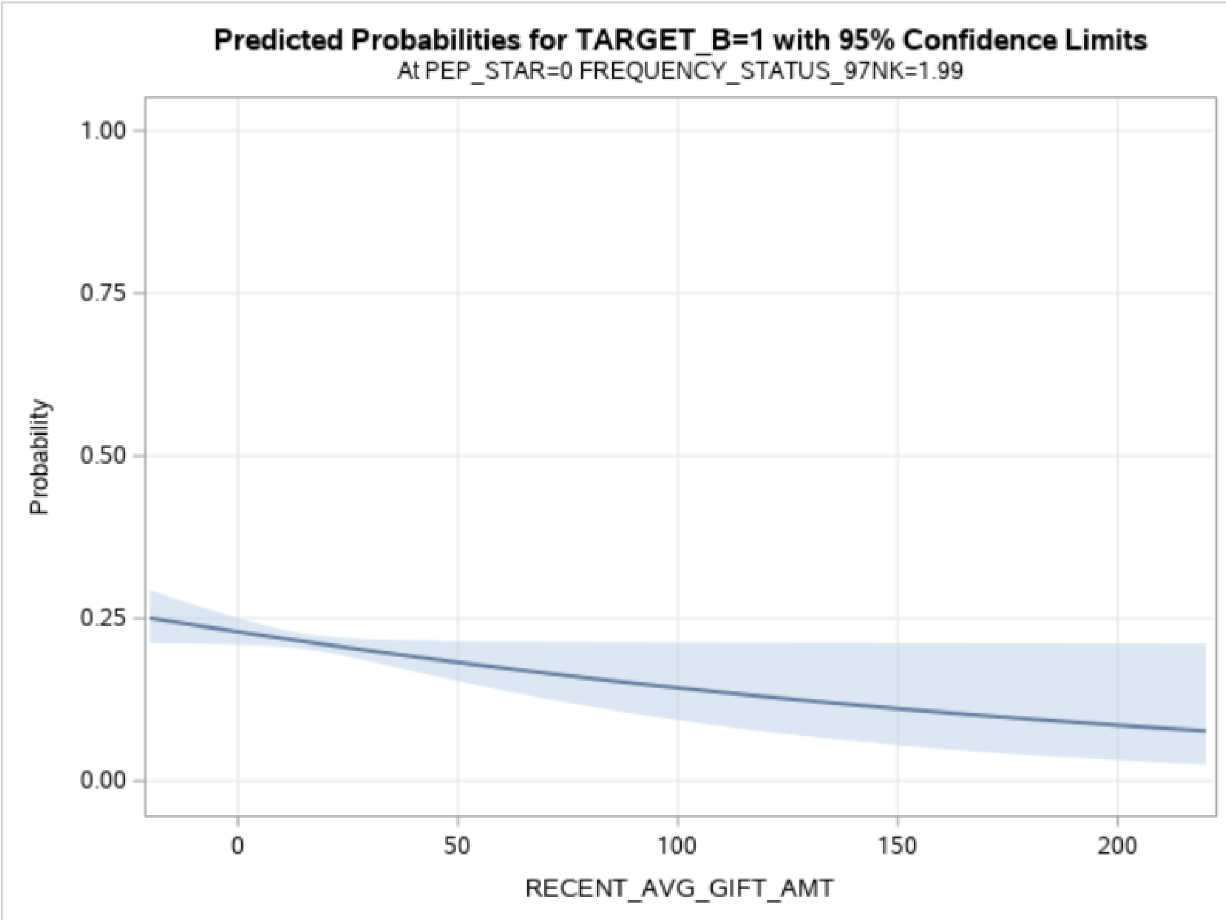


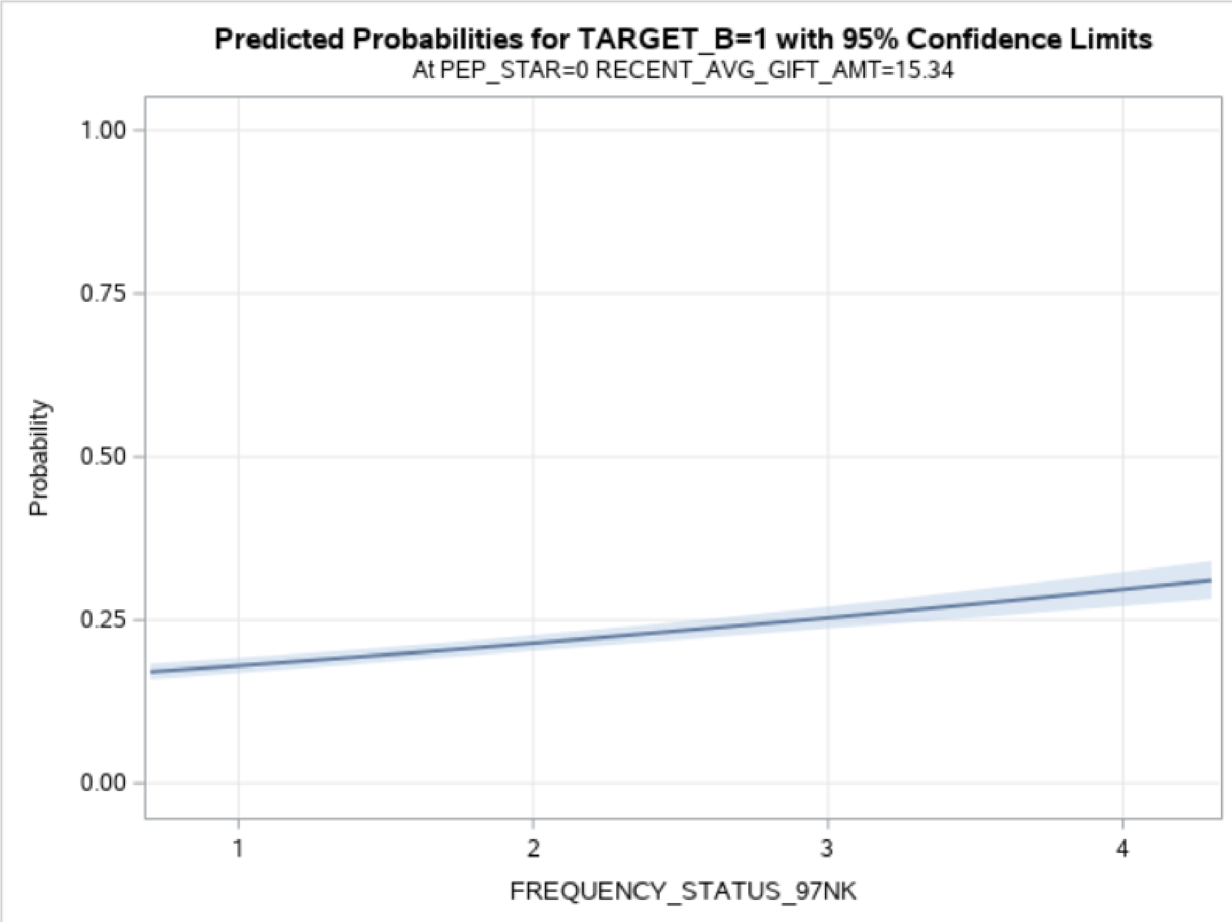

Predicted Probabilities for TARGET_B = 1
1.0
0.8
0.6
0.4
0.2
0.0
Probability
0
50
100
150
200
RECENT_AVG_GIFT_AMT
PEP_STAR
0
1
Fit computed at FREQUENCY_STATUS_97NK=1.99


Predicted Probabilities for TARGET_B = 1
1.0
0.8
0.6
0.4
0.2
0.0
Probability
1.0
1.5
2.0
2.5
3.0
3.5
4.0
FREQUENCY_STATUS_97NK
PEP_STAR
0
1
Fit computed at RECENT_AVG_GIFT_AMT=15.34



Predicted Probabilities for TARGET_B=1 with 95% Confidence Limits
At RECENT_AVG_GIFT_AMT=15.34 FREQUENCY_STATUS_97NK=1.99
Probability
1.00
0.75
0.50
0.25
0.00
0
1
PEP_STAR


Predicted Probabilities for TARGET_B=1 with 95% Confidence Limits
At PEP_STAR=0 FREQUENCY_STATUS_97NK=1.99
Probability
1.00
0.75
0.50
0.25
0.00
0
50
100
150
200
RECENT_AVG_GIFT_AMT



Adjusted Predicted Probabilities of the Veteran's Organization Data

Obs	P_1	PEP_STAR	RECENT_AVG_GIFT_AMT	FREQUENCY_STATUS_97NK
1	0.046390	1	15.00	1
2	0.033094	0	17.50	1
3	0.064890	0	8.33	4
4	0.090167	1	5.00	4
5	0.059152	1	8.33	2
6	0.058117	1	11.57	2
7	0.046941	1	12.86	1
8	0.031733	0	25.00	1
9	0.045126	1	20.00	1
10	0.032091	0	23.00	1

The MEANS Procedure

grp_resp	grp_amt	N Obs	Variable	Median
0	0	487	DONOR_AGE INCOME_GROUP WEALTH_RATING	65.0000000 4.0000000 5.0000000
	1	1147	DONOR_AGE INCOME_GROUP WEALTH_RATING	58.0000000 4.0000000 5.0000000
	2	1612	DONOR_AGE INCOME_GROUP WEALTH_RATING	58.0000000 4.0000000 6.0000000
1	0	671	DONOR_AGE INCOME_GROUP WEALTH_RATING	65.0000000 4.0000000 4.5000000
	1	1270	DONOR_AGE INCOME_GROUP WEALTH_RATING	59.0000000 4.0000000 5.0000000
	2	1202	DONOR_AGE INCOME_GROUP WEALTH_RATING	57.0000000 4.0000000 5.0000000
2	0	2155	DONOR_AGE INCOME_GROUP WEALTH_RATING	63.0000000 4.0000000 5.0000000
	1	733	DONOR_AGE INCOME_GROUP WEALTH_RATING	61.0000000 4.0000000 6.0000000
	2	410	DONOR_AGE INCOME_GROUP WEALTH_RATING	58.5000000 4.0000000 6.0000000

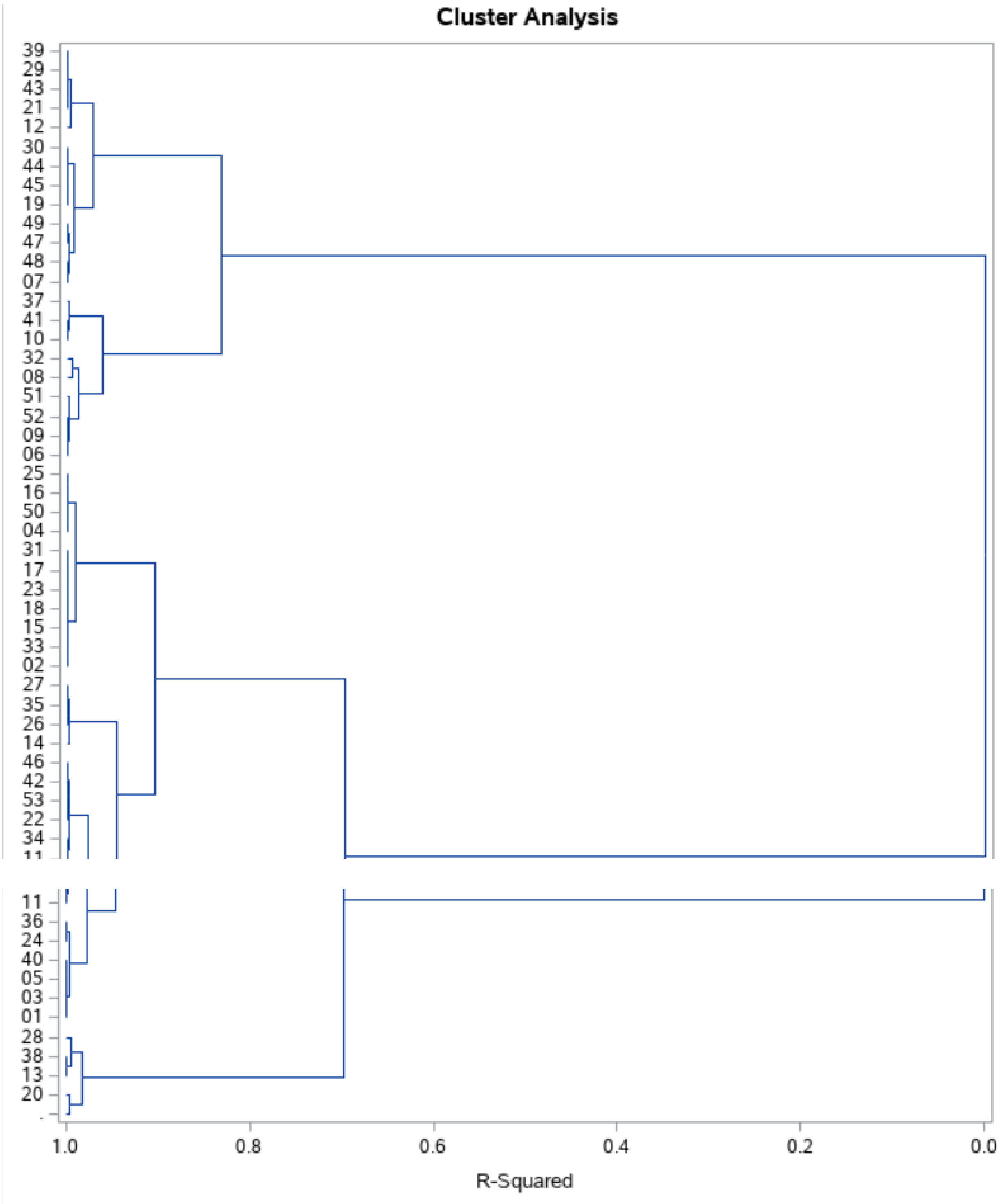
The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

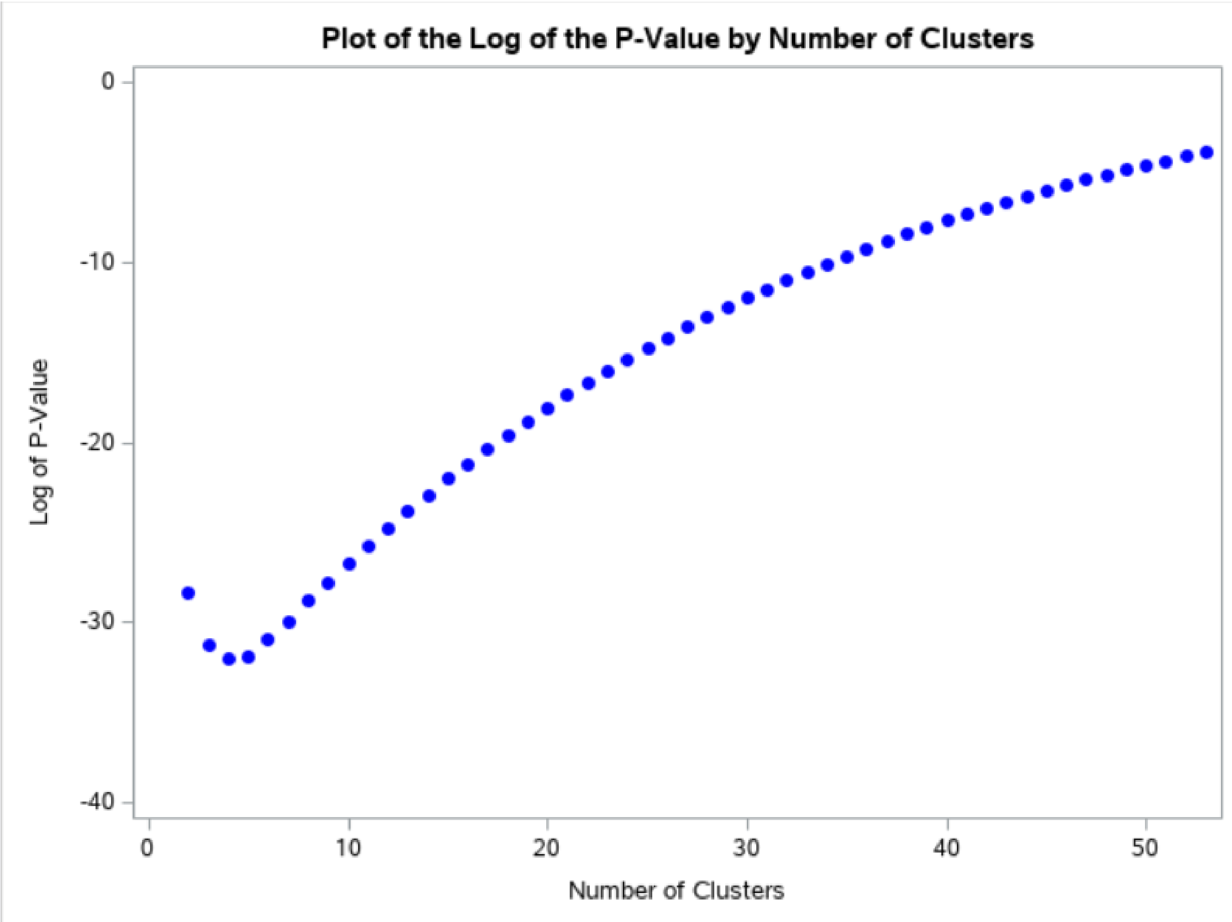
Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	0.00145225		1.0000	1.0000

Root-Mean-Square Total-Sample Standard Deviation	0.038108

Root-Mean-Square Distance Between Observations	0.053893

Cluster History						
Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Tie
53	16	25	325	0.0000	1.00	
52	19	45	291	0.0000	1.00	
51	03	05	254	0.0000	1.00	
50	18	23	455	0.0000	1.00	
49	CL52	44	473	0.0000	1.00	
48	47	49	432	0.0000	1.00	
47	22	53	265	0.0000	1.00	
46	15	CL50	565	0.0000	1.00	
45	26	35	472	0.0000	1.00	
44	07	48	192	0.0000	1.00	
43	09	52	107	0.0000	1.00	
42	17	31	296	0.0000	1.00	
41	02	33	247	0.0000	1.00	
40	11	34	374	0.0000	1.00	


Cluster Analysis
39
29
43
21
12
30
44
45
19
49
47
48
07
37
41
10
32
08
51
52
09
06
25
16
50
04
31
17
23
18
15
33
02
27
35
26
14
46
42
53
22
34
11
36
24
40
05
03
01
28
38
13
20
1.0
0.8
0.6
0.4
0.2
0.0
R-Squared


Plot of the Log of the P-Value by Number of Clusters
Log of P-Value
0
-10
-20
-30
-40
0
10
20
30
40
50
Number of Clusters

Number of Clusters
4

Cluster Assignments

CLUSNAME	CLUSTER_CODE	CLUSTER
CL4	16	1
	25	1
	03	1
	05	1
	18	1
	23	1
	22	1
	53	1
	15	1
	26	1
	35	1
	17	1
	31	1
	02	1
	33	1
	11	1
	34	1
	42	1
	04	1
	50	1

CLUSNAME	CLUSTER_CODE	CLUSTER
CL6	09	3
	52	3
	06	3
	10	3
	41	3
	51	3
	37	3
	08	3
	32	3

CLUSNAME	CLUSTER_CODE	CLUSTER
CL7	19	2
	45	2
	44	2
	47	2
	49	2
	07	2
	48	2
	21	2
	43	2
	29	2
	39	2
	30	2
	12	2

CLUSNAME	CLUSTER_CODE	CLUSTER
CL9	13	4
	38	4
	.	4
	20	4
	28	4

```
/* Solution for l3p3.sas */

/* step 2 */

%global rho1_ex;

proc sql noprint;
  select mean(target_b) into :rho1_ex
  from pmlr.pva_train_imputed;
run;

proc means data=pmlr.pva_train_imputed
        sum nway noprint;
  class cluster_code;
  var target_b;
  output out=work.counts sum=events;
run;


/* step 3 */

filename clswoe "&PMLRfolder/swoe_cluster.sas";

data _null_;
  file clswoe;
  set work.counts end=last;
    logit=log((events + &rho1_ex*24)/
        (_FREQ_ - events + (1-&rho1_ex)*24));
  if _n_=1 then put "select (cluster_code);" ;
```

```
  put "  when ('" cluster_code +(-1) "')
     cluster_swoe=" logit ";" ;
  if last then do;
    logit=log(&rho1_ex/(1-&rho1_ex));
    put "  otherwise cluster_swoe=" logit ";" / "end;";
  end;
run;

data pmlr.pva_train_imputed_swoe;
  set pmlr.pva_train_imputed;
  %include clswoe;
run;

title;

proc print data=pmlr.pva_train_imputed_swoe(obs=1);
  where cluster_code = "01";
  var cluster_code cluster_swoe;
run;
```

Obs	CLUSTER_CODE	cluster_swoe
267	01	-0.98447