**Logistic Regression modeling**

In order to compare results easily, we have performed logistic modeling on **SASHELP.HEART** dataset.

Step 1: Fine classing

Step 2: Coarse classing/dummy variable creation (Logistic modeling-example.xls consisting of methodology used for coarse classing from fine classing results generated from SAS output)

Step 3: Development & validation creation

Step 4: Variable reduction in Development dataset (by removing insignificant & multi collinear variables)- Logistic code has been used for removing insignificant & linear code has been used for removing multi-collinear variables

Step 5: Applied the agreed variables on validation dataset to see the performance

Step 6: Creating equation for application and calculating KS, Gini, Goodness of fit

**data** heart(drop = AgeAtDeath DeathCause AgeCHDdiag);

set sashelp.heart;

**run**;

/\* Creating dependent variable indicator \*/

**data** heart2(rename = (weight = bweight));

set heart;

if status = "Alive" then statind = **1**;

else statind = **0**;

**run**;

/\*Fineclassing macro for categorical variables \*/

**%Macro** FClasscat(Data, Y,var );

data \_classifier\_ (keep = &y &var weight);

set &data;

weight = **1**;

run;

proc sql;

create table \_totals\_ as

select sum(weight) as obs, sum(weight\*&Y) as total\_y

from \_classifier\_ ;

quit;

proc sql;

create table \_classifier\_ as

select T1.\*, T2.\*

from \_classifier\_ as T1, \_totals\_ as T2 ;

quit;

proc sql;

create table \_inference\_ as

select &var ,sum(weight\*&y) as good,

**100**\*(sum(weight\*&y))/(sum(total\_y\* weight)/sum(weight))

as p\_good format **8.1**,

sum(weight)-sum(weight\*&y) as bad,

**100**\*(sum(weight)-sum(weight\*&y))/(sum((obs\*weight)-(total\_y\*weight))/sum(weight))

as p\_bad format **8.1**,

**100**\*(((sum(weight\*&y))+(sum(weight)-sum(weight\*&y)))/((sum(total\_y\*weight)/sum(weight))+(sum((obs\*weight)-(total\_y\*weight))/sum(weight)))) as

p\_total format **8.1**,

**100**\*(sum(weight)-sum(weight\*&y))/(sum(weight)) as

bad\_rate format **8.1**

from \_classifier\_

group by &var;

quit;

proc sql;

create table \_inference\_ as

select &var ,good,p\_good,bad,p\_bad,p\_total,bad\_rate,

**100**\*(p\_good/(p\_bad+**0.00005**)) as g\_index format **8.1**,

**100**\*(p\_bad/(p\_good+**0.00005**)) as b\_index format **8.1**,

round(abs((p\_good-p\_bad)/**2**),**.01**) as efficiency format **8.2**,

round(log((p\_good+**0.05**)/(p\_bad+**0.05**))\*(p\_good-p\_bad),**.01**) as i\_value format **8.2**

from \_inference\_;

quit;

proc sql;

create table \_accum\_ as

select sum(efficiency) as efficiency,

sum(i\_value) as i\_value

from \_inference\_;

quit;

data \_inference\_;

set \_inference\_ \_accum\_;

run;

proc export data = \_inference\_

outfile = "/folders/myshortcuts/sasdsets/Fclass\_&var"

dbms = xls replace;

quit;

**%mend**;

%***FClasscat*** (heart2,statind,Sex);

%***FClasscat*** (heart2,statind,Chol\_Status);

%***FClasscat*** (heart2,statind,BP\_Status);

%***FClasscat*** (heart2,statind,Weight\_Status);

%***FClasscat*** (heart2,statind,Smoking\_Status);

/\*Fineclassing macro for continuous variables \*/

**%Macro** FClassnum(Data, Y,var );

data \_classifier\_ (keep = &y &var weight);

set &data;

weight = **1**;

run;

proc sql;

create table \_totals\_ as

select sum(weight) as obs, sum(weight\*&Y) as total\_y

from \_classifier\_ ;

quit;

proc sql;

create table \_classifier\_ as

select T1.\*, T2.\*

from \_classifier\_ as T1, \_totals\_ as T2 ;

quit;

proc rank data = \_classifier\_ out = \_rank\_ groups = **10** ties = mean;

var &var;

ranks r\_&var;

run;

proc sql;

create table \_inference\_ as

select r\_&var ,min(&var) as min,max(&var) as max,sum(weight\*&y) as good,

**100**\*(sum(weight\*&y))/(sum(total\_y\* weight)/sum(weight))

as p\_good format **8.1**,

sum(weight)-sum(weight\*&y) as bad,

**100**\*(sum(weight)-sum(weight\*&y))/(sum((obs\*weight)-(total\_y\*weight))/sum(weight))

as p\_bad format **8.1**,

**100**\*(((sum(weight\*&y))+(sum(weight)-sum(weight\*&y)))/((sum(total\_y\*weight)/sum(weight))+(sum((obs\*weight)-(total\_y\*weight))/sum(weight)))) as

p\_total format **8.1**,

**100**\*(sum(weight)-sum(weight\*&y))/(sum(weight)) as

bad\_rate format **8.1**

from \_rank\_

group by r\_&var;

quit;

proc sql;

create table \_inference\_ as

select r\_&var,min,max ,good,p\_good,bad,p\_bad,p\_total,bad\_rate,

**100**\*(p\_good/(p\_bad+**0.00005**)) as g\_index format **8.1**,

**100**\*(p\_bad/(p\_good+**0.00005**)) as b\_index format **8.1**,

round(abs((p\_good-p\_bad)/**2**),**.01**) as efficiency format **8.2**,

round(log((p\_good+**0.05**)/(p\_bad+**0.05**))\*(p\_good-p\_bad),**.01**) as i\_value format **8.2**

from \_inference\_;

quit;

proc sql;

create table \_accum\_ as

select sum(efficiency) as efficiency,

sum(i\_value) as i\_value

from \_inference\_;

quit;

data \_inference\_;

set \_inference\_ \_accum\_;

run;

proc export data = \_inference\_

outfile = "/folders/myshortcuts/sasdsets/Fclass\_&var"

dbms = xls replace;

quit;

**%mend**;

%***Fclassnum***(heart2,statind,height);

%***Fclassnum***(heart2,statind,bWeight);

%***Fclassnum***(heart2,statind,Diastolic);

%***Fclassnum***(heart2,statind,Systolic);

%***Fclassnum***(heart2,statind,MRW);

%***Fclassnum***(heart2,statind,Smoking);

%***Fclassnum***(heart2,statind,Cholesterol);

/\*Creating dummy variables based on group from IV calculation for independent variables

-refer logistic modeling excel sheet for classification details\*/

**data** heart3;

set heart2;

if Sex = "Female" then d\_sex1 = **1**; else d\_sex1 = **0**;

if Sex = "Male" then d\_sex2 = **1**; else d\_sex2 = **0**;

if Chol\_Status in ("Borderline"," ") then d\_chol1 = **1**; else d\_chol1 = **0**;

if Chol\_Status in ("Desirable","High") then d\_chol2 = **1**; else d\_chol2 = **0**;

if BP\_Status = "Normal" then d\_bp1 = **1**; else d\_bp1 = **0**;

if BP\_Status = "Optimal" then d\_bp2 = **1**; else d\_bp2 = **0**;

if BP\_Status = "High" then d\_bp3 = **1**; else d\_bp3 = **0**;

if Weight\_Status in("Underweight"," ") then d\_w1 = **1**; else d\_w1 = **0**;

if Weight\_Status = "Overweight" then d\_w2 = **1**; else d\_w2 = **0**;

if Weight\_Status = "Normal" then d\_w3 = **1**; else d\_w3 = **0**;

if Smoking\_Status = "Moderate (6-15)" then d\_ss1 = **1**; else d\_ss1 = **0**;

if Smoking\_Status in ("Heavy (16-25)","Light (1-5)","Non-smoker"," ") then d\_ss2 = **1**; else d\_ss2 = **0**;

if Smoking\_Status = "Very Heavy (> 25)" then d\_ss3 = **1**; else d\_ss3 = **0**;

/\* 1-red 2-yellow 3-green \*/

if **52** <= Height < **62** then d\_height1 = **1**; else d\_height1 = **0**;

if **64** <= Height < **67** then d\_height1 = **1**;

if Height > **69.5** then d\_height1 = **1**;

if Height = **.** then d\_height2 = **1**; else d\_height2 = **0**;

if **63** <= Height < **64** then d\_height2 = **1**;

if **68** <= Height <= **69.5** then d\_height2 = **1**;

if **62** <= Height < **63** then d\_height3 = **1**; else d\_height3 = **0**;

if **67** <= Height < **68** then d\_height3 = **1**;

if bweight = **.** then d\_bweight1 = **1**; else d\_bweight1 = **0**;

if **158** <= bweight <= **177** then d\_bweight1 = **1**;

if **129** <= bweight <= **157** then d\_bweight2 = **1**; else d\_bweight2 = **0**;

if **67** <= bweight <= **128** then d\_bweight3 = **1**; else d\_bweight3 = **0**;

if bweight > **178** then d\_bweight3 = **1**;

if **82** <= Diastolic <= **90** then d\_Diastolic1 = **1**; else d\_Diastolic1 = **0**;

if **50** <= Diastolic <= **80** then d\_Diastolic2 = **1**; else d\_Diastolic2 = **0**;

if **92** <= Diastolic <= **100** then d\_Diastolic2 = **1**;

if Diastolic > **100** then d\_Diastolic3 = **1**; else d\_Diastolic3 = **0**;

if **130** <= Systolic <= **143** then d\_Systolic1 = **1**; else d\_Systolic1 = **0**;

if **82** <= Systolic <= **128** then d\_Systolic2 = **1**; else d\_Systolic2 = **0**;

if **144** <= Systolic <= **166** then d\_Systolic2 = **1**;

if Systolic > **166** then d\_Systolic3 = **1**;else d\_Systolic3 = **0**;

if MRW = **.** then d\_MRW1 = **1**; else d\_MRW1 = **0**;

if MRW <= **96** then d\_MRW1 = **1**;

if **114** <= MRW <= **133** then d\_MRW1 = **1**;

if **104** <= MRW <= **113** then d\_MRW2 = **1**;else d\_MRW2 = **0**;

if **97** <= MRW <= **103** then d\_MRW3 = **1**;else d\_MRW3 = **0**;

if MRW >= **134** then d\_MRW3 = **1**;

if **10** <= Smoking <= **15** then d\_Smoking1 = **1**;else d\_Smoking1 = **0**;

if Smoking = **.** then d\_Smoking2 = **1**;else d\_Smoking2 = **0**;

if **0** <= Smoking <= **5** then d\_Smoking2 = **1**;

if **20** <= Smoking <= **25** then d\_Smoking2 = **1**;

if Smoking >= **30** then d\_Smoking3 = **1**;else d\_Smoking3 = **0**;

if Cholesterol = **.** then d\_Cholesterol1 = **1**; else d\_Cholesterol1 = **0**;

if **191** <= Cholesterol <= **200** then d\_Cholesterol1 = **1**;

if **213** <= Cholesterol <= **245** then d\_Cholesterol1 = **1**;

if **201** <= Cholesterol <= **212** then d\_Cholesterol2 = **1**; else d\_Cholesterol2 = **0**;

if **246** <= Cholesterol <= **262** then d\_Cholesterol2 = **1**;

if **96** <= Cholesterol <= **190** then d\_Cholesterol3 = **1**; else d\_Cholesterol3 = **0**;

if Cholesterol >**263** then d\_Cholesterol3 = **1**;

serial = \_n\_;

**run**;

/\* Creating development (60 %) and validation (40%) datsets randomly \*/

**proc** **surveyselect** data = heart3 samprate = **.6** seed = **12345** method = SRS out = development;

id \_all\_;

**run**;

**proc** **sql**;

create table validation as select \* from heart3

where serial not in (select serial from development);

**quit**;

/\* Step by step variable reduction process \*/

%let vars2 =

d\_sex1 /\*d\_sex2\*/

/\*d\_chol1 d\_chol2\*/

d\_bp1 d\_bp2 /\*d\_bp3\*/

d\_w1 /\*d\_w2 d\_w3\*/

/\*d\_ss1\*/ d\_ss2 /\*d\_ss3\*/

/\*d\_height1 d\_height2 d\_height3\*/

/\*d\_bweight1 d\_bweight2 d\_bweight3 \*/

/\*d\_Diastolic1 d\_Diastolic2 d\_Diastolic3\*/

d\_Systolic1 /\* d\_Systolic2 d\_Systolic3\*/

d\_MRW1 d\_MRW2 /\*d\_MRW3\*/

/\* d\_Smoking1 d\_Smoking2 d\_Smoking3 \*/

/\*d\_Cholesterol1 d\_Cholesterol2 d\_Cholesterol3\*/ ;

/\* Development dataset modeling \*/

**proc** **logistic** data = development descending;

model statind = &vars2;

**quit**;

**proc** **reg** data = development;

model statind = &vars2./vif;

**quit**;

/\*validation datset modeling \*/

**proc** **logistic** data = validation descending;

model statind = &vars2;

**quit**;

**MACRO FOR calculating KS, GINI, RANK ORDERING & Goodness of fit – for Logistic model validation**

**%macro** validation (set,score,resp,group,title);

data table1;

set &set(keep = &score &resp);

nonresp = **1** - &resp;

run;

proc sort data = table1; by descending &score; run;

data \_null\_;

set table1;

call symput('size',\_n\_);

run;

data table1;

set table1;

decile = floor(**1**+&group \* (\_n\_ - **1**)/&size);

run;

proc summary data = table1 nway missing;

var &score;

class decile;

output out = table2 (drop = \_type\_ \_freq\_ ) sum(nonresp &resp) = nonresp resp

min(&score) = minscore max(&score) = maxscore

mean(&score) = meanscore;

run;

proc summary data = table1 nway missing;

var &score;

output out = table11(drop = \_type\_ \_freq\_ ) min (&score) = minscore

max(&score) = maxscore

mean(&score) = meanscore;

run;

data table2;

set table2 end = last;

minscore = minscore\***1000**; format minscore **8.0**;

maxscore = maxscore\***1000**; format maxscore **8.0**;

predrespr = meanscore\***100**;format predrespr **8.1**;

actrespr = (resp/(resp+nonresp)) \***100**; format actrespr **8.1**;

overall\_odds = nonresp/resp;

cumnonresp+nonresp;

cumresp+resp;

cumtotal = cumnonresp + cumresp;

cumrespr = (cumresp/cumtotal)\***100**; format cumrespr **8.1**;

cumodds = cumnonresp/cumresp; format cumodds **8.2**;

if last then do;

call symput ('sumodds',cumodds);

call symput('sumnonresp',cumnonresp);

call symput('sumresp',cumresp);

call symput('sumcumresprate',cumrespr);

end;

run;

data table3(drop = cumresp cumnonresp cumodds y cumgini);

set table2 end = final;

info\_odds = overall\_odds/&sumodds; format info\_odds **8.1**;

format overall\_odds **8.1**;

log\_odds = log(info\_odds); format log\_odds **8.1**;

prob\_nonresp = nonresp/(nonresp+resp);format prob\_nonresp **8.2**;

chi\_sq = ((nonresp-

(&sumnonresp/(&sumnonresp + &sumresp)) \* (nonresp + resp)) \*\***2**)/((&sumnonresp/(&sumnonresp+&sumresp))\*(nonresp+resp))+

((resp-(&sumresp/(&sumnonresp+&sumresp))\*(nonresp+resp))\*\***2**)/((&sumresp/(&sumnonresp+sumresp))\*(nonresp+resp)); format chi\_sq **8.1**;

cumchi\_sq + chi\_sq;

pernonresp = (nonresp/&sumnonresp)\***100**; format pernonresp **8.1**;

perresp = (resp/&sumresp)\***100**; format perresp **8.1**;

cumpernonresp+pernonresp; format cumpernonresp **8.1**;

cumperresp + perresp; format cumperresp **8.1**;

perobs = ((nonresp+resp)/(&sumnonresp+&sumresp))\***100**; format perobs **8.0**;

cumperobs+perobs; format cumperobs **8.0**;

lift = (actrespr/&sumcumresprate)\***100**; format lift **8.0**;

cumlift = (cumrespr/&sumcumresprate)\***100**; format cumlift **8.0**;

ks = abs(cumperresp - cumpernonresp); format ks **8.1**;

lg1 = lag(resp);

if \_n\_ = **1** then flag = **1**; else flag = (resp<=lg1);

if flag = **0** then break = \_n\_;

y = lag(cumperresp);

gini = ((sum(cumperresp,y)/**2**)\*pernonresp)/(**100**\*\***2**); format gini **8.2**;

cumgini+gini;

if final then do;

call symput('sumgini',cumgini);

end;

predresp = meanscore \* (resp+nonresp);

diff = resp=predresp;

gofcell = (diff\*\***2**)/((resp+nonresp)\*meanscore\*(**1**-meanscore));

gof+gofcell; format gof **8.1**;

meanscore = meanscore\***1000**; format meanscore **8.0**;

run;

data table3;

set table3;

totalgini = &sumgini - **0.5**; format totalgini **8.4**;

run;

proc transpose data = table3 out = table4; var break; run;

data table4 (keep = sat\_rank ranking);

set table4 (drop = \_NAME\_);

array a{&group} col1 - col&group;

do i =**1** to &group;

if a{i} > **0** then leave;

end;

if i = (&group+**1**) then sat\_rank = 'all';else sat\_rank = i-**1**;

if sat\_rank = 'all' then ranking = 'SATISFACTORY ';

else ranking = 'NOT SATISFACTORY';

run;

proc transpose data = table3 out = table5; var ks; run;

data table5(keep = ks maxksdec);

set table5(drop = \_NAME\_);

array ks1{&group} col1 - col&group;

format ks **8.1**;

ks = **0**;

do i = **1** to &group;

if ks1{i} > ks then ks = ks1{i}; else leave;

end;

maxksdec = i-**1**;

run;

proc transpose data = table3 out = table6; var totalgini;run;

data table6;

set table6 (keep = col1);

rename col1 = gini;

run;

proc transpose data = table3 out = table7; var gof; run;

data table7;

set table7(keep = col&group);

rename col&group = gof;

run;

data mrg;

set table4 table5 table6 table7;

run;

data total(drop = cumrespr);

set table2(obs = **1**);

nonresp = &sumnonresp;

resp = &sumresp;

actrespr = (resp/(resp+nonresp))\***100**; format actrespr **8.1**;

overall\_odds = nonresp/resp;

info\_odds = overall\_odds/&sumodds;

format overall\_odds **8.1**;

format info\_odds **8.1**;

log\_odds = log(info\_odds); format log\_odds **8.1**;

prob\_nonresp = nonresp/(nonresp+resp); format prob\_nonresp **8.2**;

pernonresp = (nonresp/nonresp)\***100**; format pernonresp **8.1**;

perresp = (resp/resp)\***100**; format perresp **8.1**;

perobs = ((nonresp+resp)/(&sumnonresp+&sumresp))\***100**; format perobs **8.0**;

run;

data tone;

set table11;

minscore = minscore \***1000**; format minscore **8.0**;

maxscore = maxscore\***1000**; format maxscore **8.0**;

predrespr = meanscore \* **1000**; format predrespr **8.1**;

meanscore = meanscore\***1000**; format meanscore **8.0**;

run;

data ttwo(keep = cumchi\_sq cumgini);

set table3;

cumchisq + chi\_sq;

cumgini+gini;

if \_n\_ = &group;

run;

data tthree;

set table5(keep = ks);

run;

data tfour;

set table7 (keep = gof);

run;

data total;

merge total tone ttwo tthree tfour;

rename cumchi\_sq = chi\_sq;

rename cumgini = gini;

run;

data table3(drop = decile);

set table3 total;

if \_n\_ = &group+**1** then decile = **9999**;

decile1 = put(decile,**8.**); format decile1 $8.;

run;

data table3;

set table3;

rename decile1 = decile;

decile1 = left(trim(decile1));

if decile1 = "9999" then decile1 = "Total";

run;

proc print data = table3 noobs;

title "&title";

var decile nonresp resp actrespr predrespr minscore maxscore meanscore overall\_odds info\_odds log\_odds

prob\_nonresp chi\_sq pernonresp perresp cumpernonresp cumperresp perobs cumperobs cumrespr lift cumlift gini

ks gof;

run;

proc print data = mrg noobs;

var ranking sat\_rank ks maxksdec gini gof;

run;

**%mend**;

%***validation***(development2,prob\_score,statind,**10**,KSTable\_Development);

%***validation***(validation2,prob\_score,statind,**10**,KSTable\_Validation);