%let loc=/mnt/projects/public/offshore\_office/frankji/CE exercise/exercise/sasdata/; \*\* Project folder;

libname lib "&loc"; \*\* SAS dataset libname;

libname out "&loc.ce\_output"; \*\* CE output libname;

\* Comment out any variable types that are not present in the data;

%include "&loc.contvar\_list.txt"; \*\* List of continuous variables to include in analysis;

%include "&loc.ordvar\_list.txt"; \*\* List of ordinal variables to include in analysis;

%include "&loc.nomvar\_list.txt"; \*\* List of nominal variables to include in analysis;

%include "&loc.binvar\_list.txt"; \*\* List of binary variables to include in analysis;

\*\*\* The following are macro variables needed for CE \*\*\*;

\*\*\* Mandatory Macro Variables: These must be set \*\*\*;

\* General Macro Variables \*;

%let path\_output=&loc.ce\_output/; \*\* your output destination folder;

%let inds=lib.randomsample; \*\* your sas input dataset name;

%let id = rid; \*\* Unique ID;

%let dep\_var=dv\_dpa\_purchase\_flag\_in\_attr; \*\* your dependant variable;

%let binary\_dv=Y; \*\* your dependent variable type (Y/N);

%let weight= ; \*\* If you want to use weights, set this variable to the name of your weight variable;

\*\*\* Macro 1: Sampling macro variables \*\*\*;

%let split\_if = if ranuni(12345)<0.7; \*\* your split condition for modeling portion;

%let exclusion\_if=; \*\* Data exclusion if condition applies. e.g. IF &Dep\_var = .;

%let DS\_Present= N; \*\* change to Y only if your data contain standard DS modeling bundle;

\*\*\* Macro 3: Variable Reduction macro variables \*\*\*;

%let fast\_opt = N; \*\* Fast option will turn off all tests except multivariate regression

\*\*\* Optional Macro Variables: These all have defaults that can be used \*\*\*;

\* General Macro Variables \*;

%let prefix=R1\_; \*\* your variable recoding prefix;

%let keep\_list=&id &dep\_var &weight mod\_val\_test; \*\* Add any additional variables you want to keep for analysis purposes;

\*\*\* Macro 1: Sampling macro variables \*\*\*;

%let path\_DS=/mnt/projects/shared/pst\_qmgisi/Modeling/CE/; \*\* Location of standard DS recodes. Do Not Change;

%let bootstrap= N; \*\* request oversamping, bootstrap if responders size is small (Y/N). Binary dv only;

%let oversampled\_rr=0.05; \*\* your desired oversample response rate on the modeling part of data, 0.1-0.5;

%let min\_num\_resp=500; \*\* Minimum number of responders if sampling;

%let seed=123456; \*\* Random selection seed if sampling;

\*\*\* Macro 2: Recoding macro variables \*\*\*;

%let Profiling = Y; \*\* Request profiling report on all variables (Y/N);

%let missrate = .75; \*\* maximum missing rate allowed;

\* Binary and Nominal variable types;

%let concrate = .9; \*\* Maximum amount of file that can be in a single value of an independent variable;

\* Nominal variable types;

%let valcnt = 50; \*\* Maximum number of unique values allowed. Set to 0 to allow any number;

%let minbinnc = 500; \*\* Minimum count in a bin to be usable. set to 0 to use minbinpct only;

%let minbinnp = .05; \*\* Minimum percent of file in a bin to be usable. set to 0 to use minbincnt only;

%let talpha = 0.05; \*\* T Test significance level for collapse of bins;

%let bonfer = N; \*\* Do Bonferoni adjustment for talpha? (Y/N);

%let nom\_method = INDEX; \*\* Recoding method for nominal variables: Binary, Index or Mean;

\* Continuous and Ordinal variable types;

%let pvalue = .05; \*\* Pvalue threshold to include variable;

%let min\_size = 500; \*\* Minimum missing group size for Equal Response imputation;

%let num\_category = 10; \*\* Maximum number of categories for profiling variables;

%let equal\_dist = N; \*\* Use equal distance for dividing variables into groups for profiling (Y/N);

\* Continuous variable types;

%let p\_lo = 1; \*\* Lower percentile for checking constant value;

%let p\_hi = 99; \*\* Upper percentile for checking constant value;

%let impmethodC = median; \*\* What method to use for missing imputation for continuous variables?;

\*\* Options are ER for Equal Reponse or any proc stdize method;

\*\* The location associated with the method will be used for missing values. Be sure to include c or p value if needed;

\*\* Method Location

\*\* ------- ---------

\*\* MEAN Mean

\*\* MEDIAN Median

\*\* SUM 0

\*\* EUCLEN 0

\*\* USTD 0

\*\* STD Mean

\*\* RANGE Minimum

\*\* MIDRANGE Midrange

\*\* MAXABS 0

\*\* IQR Median

\*\* MAD Median

\*\* ABW(c) Biweight one-step M-estimate

\*\* AHUBER(c) Huber one-step M-estimate

\*\* AWAVE(c) Wave one-step M-estimate

\*\* AGK(p) Mean

\*\* SPACING(p) Mid-minimum spacing

\*\* L(p) L(p) ;

%let stdmethodC = STD; \*\* Standardization options: any method allowed in proc stdize or NO to skip;

%let cap\_flrC = Y; \*\* Do you want to do capping / flooring to handle outliers? (Y/N);

%let transformationC = Y; \*\* Include transformed variables in evaluation (Y/N);

\* Ordinal variable types;

%let impmethodO = mean; \*\* What method to use for missing imputation for continuous variables?;

\*\* Options are ER for Equal Reponse or any proc stdize method;

%let stdmethodO = No; \*\* Standardization options: any method allowed in proc stdize or NO to skip;

%let cap\_flrO = N; \*\* Do you want to do capping / flooring to handle outliers? (Y/N);

%let transformationO = N; \*\* Include transformed variables in evaluation? (Y/N);

\*\*\* Macro 3: Variable Reduction macro variables \*\*\*;

%let samplesize = 50000; \*\* Sample size to use for variable reduction;

%let redu\_weight = N; \*\* Use weights in variables reduction? (Y/N);

%let sources = 3; \*\* Minimum number of sources to be selected;

\* Univariate regression option;

%let univ\_reg = Y; \*\* Use univariate regression to choose variables? (Y/N);

%let maxpuni = .0001; \*\* Maximum p value correlation for selecting via univariate regression;

\* Correlation option;

%let correlation = Y; \*\* Use correlation to choose variables? (Y/N);

%let corrcut = .01; \*\* Minimum correlation between independent variable and dependent variable;

\* Principal components option;

%let principal = Y; \*\* Use principal components to choose variables? (Y/N);

%let nprin = 10; \*\* Number of principal components desired;

%let minprin = .5; \*\* Minimum factor correlation for selecting via principal component;

\* Cluster option;

%let cluster = Y; \*\* Use cluster analysis to choose variables? (Y/N);

%let maxc = 20; \*\* Number of clusters desired;

%let maxratio = .5; \*\* Maximum R Squared ratio for selecting variables via clustering;

\* Linear regression option;

%let regression = Y; \*\* Use linear regression to choose variables? (Y/N);

%let alphareg = .05; \*\* Alpha level for forward selection in linear regression;

\* Logistic regression option - only applicable if binary dependent variable;

%let logistic = Y; \*\* Use logistic regression to choose variables? (Y/N);

%let alphalog = .05; \*\* Alpha level for forward selection in logistic regression;

\* Information value option;

%let information = Y; \*\* Use information value to choose variables? (Y/N);

%let decile = 20; \*\* Number of groups to use when calculate information values;

%let infvcut = .01; \*\* Minimum information value for selecting via information value;

\* Maximum correlation between independent variables option;

%let ind\_correlation = Y; \*\* Exclude variables with high correlation to others? (Y/N);

%let maxcorr = .7; \*\* Maximum correlation allowed between independent variables;

\* Maximum correlation to dependent variable option;

%let ind\_dv\_corr=Y; \*\* Exclude variables with high correlation to dependent variable? (Y/N);

%let max\_dv\_corr=.7; \*\* Maximum correlation allowed to dependent variable;

\*\*\* Macro 4: Model selection and tuning \*\*\*;

%let sel\_alpha = .05; \*\* Alpha level when selecting/removing variable into the model;

%let includelist = ; \*\* List of variables that has to be in the final model;

%let startlist = ; \*\* List of variables that need to be in the first step;

%let excludelist = ; \*\* List of variables to be excluded from modeling;

%let criteria = c; \*\* Metric to use during variable tuning. Default is c for binary, AdjRsq for other;

%let threshold = 0; \*\* Minimum change in evaluation metric to include variable;

%let SQL\_join = union; \*\* Type of join to use between file portions during variable tuning;

%let minimp = .01; \*\* Minimum relative importance to keep variable;

%let graph\_plot = Y; \*\* Include graphing? (Y/N);

\*\*\* Macro 5: Final Model selection \*\*\*;

%let fin\_alpha = .05; \*\* Alpha level when selecting/removing variable into the model;

%let method = stepwise; \*\* Model build method;

%let fin\_num\_category = 10; \*\* Maximum number of categories for profiling variables;

%let fin\_equal\_dist = N; \*\* Use equal distance for dividing variables into groups for profiling (Y/N);

ods html close;

ods listing;

%inc "/mnt/projects/shared/pst\_qmgisi/Modeling/CE/CE\_Macros\_v5.sas";

\*\*\* 1.Sampling \*\*\*;

%let LogFile = "&path\_output.01\_CE\_Sampling\_Log\_File.log";

%let LstFile = "&path\_output.01\_CE\_Sampling\_LST\_File.lst";

proc printto log=&LogFile new; run;

proc printto print=&LstFile new; run;

%CE\_Sampling(inds=&inds, outds=out.CE1\_Resampled)

proc printto;

run;

\*\*\* 2.EDA, profiling and Recode \*\*\*;

%let LogFile = "&path\_output.02\_CE\_Var\_EDA\_Recode\_Log\_File.log";

%let LstFile = "&path\_output.02\_CE\_Var\_EDA\_Recode\_List\_File.lst";

proc printto log=&LogFile new; run;

proc printto print=&LstFile new; run;

%CE\_EDA\_RECODE(INSDN=out.CE1\_Resampled);

proc printto;

run;

\*\*\* 3.Variable reduction and ranking \*\*\*;

%let LogFile = "&path\_output.03\_Var\_Redu\_Log\_File.log";

%let LstFile = "&path\_output.03\_Var\_Redu\_List\_File.lst";

proc printto log=&LogFile new; run;

proc printto print=&LstFile new; run;

%CE\_Var\_Redu(insdn=out.CE2\_Recoded);

proc printto;

run;

\*\*\* 4.Model selection and tuning \*\*\*;

%let LogFile = "&path\_output.04\_Model\_Selection\_Log\_File.log";

%let LstFile = "&path\_output.04\_Model\_Selection\_List\_File.lst";

proc printto log=&LogFile new; run;

proc printto print=&LstFile new; run;

%inc "&path\_output.CE3\_Varlist\_redu.txt";

%CE\_Model\_Val(out.CE2\_Recoded, &varlist\_redu);

proc printto;

run;

\*\*\* 5.Final Model build and validation on test sample \*\*\*;

%let LogFile = "&path\_output.05\_Final\_Model\_Fit\_Validation\_Log\_File.log";

%let LstFile = "&path\_output.05\_Final\_Model\_Fit\_Validation\_LST\_File.lst";

proc printto log=&LogFile new; run;

proc printto print=&LstFile new; run;

%inc "&path\_output.CE4\_Varlist\_Final.txt";

%CE\_Model\_Lift(insdn=out.CE2\_Recoded, varlist=&varlist\_final)

proc printto;

run;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* 1.Sampling \*\*\*;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\* Macro Sample\_Core: Take a random sample \*\*;

%macro Sample\_Core(in=,out=,response\_var=,samplesize=,ratio=,seed=);

%local default\_seed;

%let default\_seed=;

%if %length(&seed) ne 0 %then %let default\_seed=&seed;

data \_tmp0\_;

min\_num\_nonresp=int((&min\_num\_resp/&Ratio)-&min\_num\_resp);

call symputx('default\_num\_response',&min\_num\_resp);

call symputx('default\_num\_nonresponse',min\_num\_nonresp);

run;

data \_tmp\_;

n1=int(&samplesize\*&ratio);

n2=&samplesize-n1;

if n1 < &default\_num\_response then do;

n1=&default\_num\_response;

/\* output warning message here\*/

put "Warning: Too few responses. Number of Responses is now set to &default\_num\_response..";

end;

if n2 < &default\_num\_nonresponse then do;

n2=&default\_num\_nonresponse;

put "Warning: Too few non-responses. Number of Non-Responses is now set to &default\_num\_response..";

end;

call symputx('\_num\_response\_sample\_',n1);

call symputx('\_num\_nonresponse\_sample\_',n2);

run;

%put &\_num\_response\_sample\_ &\_num\_nonresponse\_sample\_;

data \_in1\_ (compress=yes pointobs=yes reuse=no) \_in2\_ (compress=yes pointobs=yes reuse=no);

set &in;

if &response\_var>0 then output \_in1\_;

else output \_in2\_;

run;

data \_null\_;

set \_in1\_ nobs=n1;

set \_in2\_ nobs=n2;

call symputx('\_num\_response\_original\_',n1);

call symputx('\_num\_nonresponse\_original\_',n2);

stop;

run;

%put &\_num\_response\_original\_ &\_num\_nonresponse\_original\_;

%\* Copy the same data;

%if (&\_num\_response\_original\_=&\_num\_response\_sample\_) %then %do;

data \_out1\_;

set \_in1\_;

run;

%end;

%else %if (&\_num\_response\_original\_>&\_num\_response\_sample\_) %then /\* Sample \*/

%Sample\_Option\_1(in=\_in1\_,out=\_out1\_,samplesize=&\_num\_response\_sample\_,seed=&default\_seed);

%else /\* Bootstrap \*/

%Sample\_Option\_2(in=\_in1\_,out=\_out1\_,samplesize=&\_num\_response\_sample\_,seed=&default\_seed);

%if (&\_num\_nonresponse\_original\_=&\_num\_nonresponse\_sample\_) %then %do;

data \_out2\_;

set \_in2\_;

run;

%end;

%else %if (&\_num\_nonresponse\_original\_>&\_num\_nonresponse\_sample\_) %then /\* Sample \*/

%Sample\_Option\_1(in=\_in2\_,out=\_out2\_,samplesize=&\_num\_nonresponse\_sample\_,seed=&default\_seed);

%else /\* Bootstrap \*/

%Sample\_Option\_2(in=\_in2\_,out=\_out2\_,samplesize=&\_num\_nonresponse\_sample\_,seed=&default\_seed);

data &out;

set \_out1\_ \_out2\_;

run;

%mend Sample\_Core;

\*\* If the size of desired sample is smaller than the size of the input dataset \*\*;

%macro Sample\_Option\_1(in=,out=,samplesize=,seed=);

%\* if seed not specified;

%if %length(&seed)=0 %then %do;

proc surveyselect data=&in out=&out n=&samplesize noprint;

run;

%end;

%else %do;

proc surveyselect data=&in out=&out n=&samplesize seed=&seed noprint;

run;

%end;

%mend Sample\_Option\_1;

\*\* If the size of the desired sample is greater then the size of input \*\*;

%macro Sample\_Option\_2(in=,out=,samplesize=,seed=);

data \_NULL\_;

set &in nobs=\_nsize\_;

Extra\_num=&samplesize-\_nsize\_;

call symputx('Extra\_num',Extra\_num);

run;

data &out;

set &in nobs=\_nsize\_;

do \_i\_=1 to &Extra\_num;

\_tmp\_index\_=ceil(\_nsize\_\*ranuni(&default\_seed));

set &in point=\_tmp\_index\_;

output;

end;

drop \_i\_;

stop;

run;

data &out;

set &in &out;

run;

%mend Sample\_Option\_2;

\*\* Master Sampling macro \*\*;

%macro CE\_Sampling(inds=, outds=OUT.CE1\_Resampled);

%\*Split the original datasets;

data \_mod \_test;

set &inds;

%if &exclusion\_if NE %then %do;

&exclusion\_if then delete;

%end;

&split\_if then mod\_val\_test=1;

else mod\_val\_test=3;

%if %upcase(&ds\_present)=Y %then %do;

%inc "&path\_DS/DS Recodes.txt";

%end;

if mod\_val\_test=3 then output \_test;

else output \_mod;

run;

%\*bootstrap the modeling portion if requested;

%if %lowcase(&Binary\_dv)=y and %lowcase(&bootstrap)=y %then %do;

%\*Get key components for next sampling macro;

proc means data=\_mod(keep=&dep\_var) n mean noprint;

var &dep\_var;

output out=\_temp n=count1 mean=true\_resp;

run;

data \_NULL\_;

set \_temp;

num\_resp=count1\*true\_resp;

if num\_resp>=&min\_num\_resp then do;

spsize=int((count1\*true\_resp)/&oversampled\_rr);

end;

else if num\_resp<&min\_num\_resp then do;

spsize=int(&min\_num\_resp/&oversampled\_rr);

put "Warning: Too few responses. Number of Responses is now set to &min\_num\_resp...";

end;

call symputx('spsize',spsize);

call symputx('true\_resp',true\_resp);

run;

%put \*\* New modeling sample size: &spsize;

%put \*\* Oversampled Response Rate: &oversampled\_rr;

%put \*\* Original Resp Rate: &true\_resp;

%Sample\_Core(in=\_mod,out=\_mod,response\_var=&dep\_var,samplesize=&spsize,ratio=&oversampled\_rr,seed=&seed);

%\*QC;

proc freq data=&inds;

title "&dep\_var Frequency in Original Dataset";

tables &dep\_var;

run;

proc freq data=\_mod;

title "&dep\_var Frequency in Oversampled Training Dataset";

tables &dep\_var;

run;

proc freq data=\_test;

title "&dep\_var Frequency in unsampled Validation Dataset";

tables &dep\_var;

run;

title ' ';

%end;

%\*combine splitted/oversampled data;

data &outds;

set \_mod \_test;

if mod\_val\_test=1 and ranuni(9545)>0.5 then

mod\_val\_test=2;

run;

%\*Output the sample response rate;

proc means data=&outds nway;

class mod\_val\_test;

var &dep\_var;

output out=out.CE1\_Sample\_Rate mean=;

run;

data out.CE1\_Sample\_Rate;

length partition $ 5;

set out.CE1\_Sample\_Rate (drop=\_type\_);

if mod\_val\_test=1 then partition='Model';

else if mod\_val\_test=2 then partition='Val';

else if mod\_val\_test=3 then partition='Test';

run;

%\*clean up space;

proc datasets lib=work nolist;

delete \_tmp\_ \_in1\_ \_in2\_ \_out1\_ \_out2\_ \_mod \_test;

quit;

%mend;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* 2.EDA,profiling and Recode \*\*\*;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\* Common ordinal and continuous processing macro \*\*;

%macro pnum(insdn,var,nmiss,typ);

%\* Get Non-missing key statistics;

proc means data=&insdn mean median min p1 p25 p75 p99 max NOPRINT;

var &var;

output out=EDA mean=var\_mean median=var\_median min=var\_min p1=var\_p1 p25=var\_p25 p75=var\_p75 p99=var\_p99 max=var\_max / noinherit;

run;

%let skip = 1;

data EDA; set EDA;

%\* calculate upper and lower bounds;

iqr=Max((var\_p75-var\_p25),(var\_p99-var\_p75),(var\_p25-var\_p1));

var\_lb=Min(Max((var\_p25- 1.5\*iqr),var\_min),var\_p1);

var\_ub=Max(Min((var\_p75+ 1.5\*iqr),var\_max),var\_p99);

if var\_lb=var\_ub then do;

var\_lb=var\_min;

var\_ub=var\_max;

end;

var\_mid = (var\_max - var\_min)/2;

CALL SYMPUTX('var\_LB' ,var\_lb);

CALL SYMPUTX('var\_UB' ,var\_ub);

CALL SYMPUTX('var\_median' ,var\_median);

if upcase("&&impmethod&typ") in ('MEAN','STD') then CALL SYMPUTX('var\_miss' ,var\_mean);

else if upcase("&&impmethod&typ") in ('MEDIAN','IQR','MAD') then CALL SYMPUTX('var\_miss' ,var\_median);

else if upcase("&&impmethod&typ") in ('RANGE') then CALL SYMPUTX('var\_miss' ,var\_min);

else if upcase("&&impmethod&typ") in ('MIDRANGE') then CALL SYMPUTX('var\_miss' ,var\_mid);

else if upcase("&&impmethod&typ") in ('SUM','EUCLEN','USTD','MAXABS') then CALL SYMPUTX('var\_miss' ,0);

else CALL SYMPUTX('skip',0);

run;

%\* Cap\_FLR before transformation;

data NONMISSING; set &insdn (where=(missing(&var)=0));

&var=MIN(MAX(&var,&var\_lb),&var\_ub);

%if &&transformation&typ=Y %then %do;

SQ\_&var=&var\*\*2;

SR\_&var=sqrt(max(&var,0));

LN\_&var=log(max(&var,0.00001));

\*IV\_&var= -1/max(&var,0.00001);

\*EP\_&var= -exp(Min(0,-&var));

keep &dep\_var &var SQ\_&var SR\_&var LN\_&var /\*IV\_&var EP\_&var\*/;

%let mod\_list=&var SQ\_&var SR\_&var LN\_&var;

%end;

%else %do;

keep &dep\_var &var;

%let mod\_list=&var;

%end;

run;

%let ck = %sysevalf(&nmiss - &min\_size < 0);

%if &skip = 0 and %upcase(&&impmethod&typ) ^= ER %then %do;

proc stdize data=&insdn out=dummy outstat=tmp method = &&impmethod&typ;

var &var;

run;

data \_NULL\_; set tmp (where=(\_type\_="LOCATION"));

call symputx("var\_miss",&var);

run;

proc datasets nolist; delete tmp dummy; quit; run;

%end;

%else %if %upcase(&&impmethod&typ) = ER and &ck = 0 %then %do;

%\* Get missing data average target;

proc means data=&insdn mean noprint;

var &dep\_var;

where missing(&var);

output out=M\_RR mean=Missing\_RR;

run;

data \_NULL\_; set M\_RR;

%if %upcase(&Binary\_dv) = Y %then %do;

if Missing\_RR=0 then Missing\_RR=0.0001;

else if Missing\_RR=1 then Missing\_RR=0.9999;

%end;

call symputx('Missing\_RR' ,Missing\_RR);

run;

proc datasets nolist; delete M\_RR; quit; run;

%end;

%if %upcase(&Binary\_dv) = Y %then %do;

%\* Run univariate logistic regression;

ods listing close;

ods output parameterestimates=parm association=assoc;

proc logistic data=NONMISSING desc namelen=32;

model &dep\_var=&mod\_list /PARMLABEL selection=forward STOP=1 slentry=1;

run;

ods listing;

%if %sysfunc(exist(assoc)) %then %do;

data \_NULL\_; set assoc(keep=cvalue1 cvalue2);

if \_n\_ = 1 then call symputx("Concordant",cvalue1);

else if \_N\_ = 4 then call symputx("CValue",cvalue2);

run;

proc datasets nolist; delete assoc; quit; run;

%end;

%else %do;

%let Concordant = . ;

%let CValue = . ;

%end;

%end;

%else %do;

%\* Run univariate simple regression;

ods listing close;

ods output SelParmEst=parm SelectionSummary=SelectionSummary;

proc reg data=NONMISSING;

model &dep\_var=&mod\_list/selection=forward MAXSTEP=1 slentry=0.999;

run;

ods listing;

%if %sysfunc(exist(SelectionSummary)) %then %do;

data \_NULL\_; set SelectionSummary(keep=ModelRsquare obs=1);

call symputx("RSquare",ModelRsquare);

run;

proc datasets nolist; delete SelectionSummary; quit; run;

%end;

%else %do;

%let RSquare = . ;

%end;

%end;

data \_NULL\_; set parm(obs=1);

call symputx('Intercept',Estimate);

run;

data \_NULL\_; set parm (firstobs=2 obs=2);

%if %upcase(&Binary\_dv) = Y %then %do;

call symputx('prob',ProbChiSq);

%end;

%else %do;

call symputx('prob',ProbF);

%end;

run;

%let ckP = %sysevalf(&prob - .05 > 0);

data parm; set parm (firstobs=2 obs=2);

length relationship $ 6 ck 3.;

\_Trans=substr(variable,1,3);

if \_Trans not in ('SQ\_','SR\_','LN\_','IV\_','EP\_') or (variable = "&var") then \_Trans='';

%if %upcase(&&impmethod&typ) = ER %then %do;

%if %upcase(&Binary\_dv) = Y %then %do;

%\*missing imputation based on missing target rate;

%if &ck = 1 or &ckP = 1 %then %do;

miss\_impute=&var\_median;

%end;

%else %do;

if \_Trans='SQ\_' then miss\_impute=SQRT(Max((log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate,0));

else if \_Trans='SR\_' then miss\_impute=((log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate)\*\*2;

else if \_Trans='LN\_' then miss\_impute=exp((log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate);

else if \_Trans='IV\_' then miss\_impute=-1/((log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate);

else if \_Trans='EP\_' then miss\_impute=-log(Max(-(log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate,0.00001));

else miss\_impute=(log(&Missing\_RR/(1-&Missing\_RR))-&Intercept)/Estimate;

%end;

%end;

%else %do;

%\*missing imputation based on missing target rate;

%if &ck = 1 or &ckP = 1 %then %do;

miss\_impute=&var\_median;

%end;

%else %do;

if \_Trans='SQ\_' then miss\_impute= SQRT(Max((&Missing\_RR-&Intercept)/Estimate,0));

else if \_Trans='SR\_' then miss\_impute= ((&Missing\_RR-&Intercept)/Estimate)\*\*2;

else if \_Trans='LN\_' then miss\_impute= exp((&Missing\_RR-&Intercept)/Estimate);

else if \_Trans='IV\_' then miss\_impute= -1/((&Missing\_RR-&Intercept)/Estimate);

else if \_Trans='EP\_' then miss\_impute= -log(Max(-(&Missing\_RR-&Intercept)/Estimate,0.00001));

else miss\_impute= (&Missing\_RR-&Intercept)/Estimate;

%end;

%end;

if miss\_impute<&var\_LB then miss\_impute=&var\_LB;

else if miss\_impute>&var\_UB then miss\_impute=&var\_UB;

call symputx("var\_miss",miss\_impute);

%end;

if Estimate>0 then sign='(+)'; else sign='(-)';

relationship=compress(\_Trans||sign,'');

\*\* Output fields to append to the vars table;

call symputx("New\_var",variable);

call symputx("Sign",sign);

call symputx("Relation",relationship);

%if %upcase(&Binary\_dv) = Y %then %do;

call symputx("Prob",ProbChiSq);

call symputx("Chisq",WaldChiSq);

%end;

%else %do;

call symputx("Prob",ProbF);

call symputx("FValue",FValue);

%end;

run;

%\*\* Update vars table \*\*;

data vars2; set vars2;

if name = "&var" then do;

var\_lb = &var\_lb;

var\_ub = &var\_ub;

var\_median = &var\_median;

miss\_impute = &var\_miss;

%if %upcase(&Binary\_dv) = Y %then %do;

Concordant = &Concordant;

CValue = &CValue;

Chisq = &Chisq;

PValue = &Prob;

%end;

%else %do;

RSquare = &RSquare;

FValue = &FValue;

PValue = &Prob;

%end;

if PValue <= &PValue then do;

new\_var = "&Prefix"||"&New\_var";

Sign = "&Sign";

Relationship = "&Relation";

end;

end;

run;

%if %sysevalf(&Pvalue-&prob>=0) %then %do;

%if %upcase("&&stdmethod&typ") ^= "NO" %then %do;

%\* Standardization;

data temp; set &insdn (keep=&var);

\_Trans=substr("&Relation",1,3);

&New\_var = &var;

if missing(&New\_var) then &New\_var = &var\_miss;

%if %upcase(&&cap\_flr&typ) = Y %then %do;

&New\_var = MIN(MAX(&New\_var, &Var\_LB), &var\_UB);

%end;

\*standard transformation;

if \_Trans='SQ\_' then &New\_var = &New\_var \*\*2;

else if \_Trans='SR\_' then &New\_var = SQRT(MAX(&New\_var,0));

else if \_Trans='LN\_' then &New\_var = LOG(MAX(&New\_var,0.00001));

else if \_Trans='IV\_' then &New\_var = -1/(MAX(&New\_var,0.00001));

else if \_Trans='EP\_' then &New\_var = -EXP(MIN(-&New\_var,0));

run;

proc stdize data=temp out=dummy outstat=std\_tmp method = &&stdmethod&typ;

var &New\_var;

run;

data \_NULL\_; set std\_tmp;

if \_type\_ = 'LOCATION' then call symputx("loc",&new\_var);

if \_type\_ = 'SCALE' then call symputx("scale",&new\_var);

run;

%\*\* Update vars table \*\*;

data vars2; set vars2;

if name = "&var" then do;

Location = &loc;

Scale = &scale;

end;

run;

proc datasets nolist; delete temp dummy std\_tmp; quit; run;

%end;

data \_NULL\_; set vars2 (where=(name="&var"));

\_Trans=substr(relationship,1,3);

if missing(label) then lab2 = name;

else lab2 = label;

%if &typ = O %then %do;

FILE "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt" mod;

%end;

%else %do;

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" mod;

%end;

PUT ' ' '0d'x;

PUT "\*\*\* RECODE " name ": " label " \*\*\*;" '0d'x;

\* Missing imputation;

Put "IF missing(" name +(-1) ") THEN &PREFIX" name " = " Miss\_Impute " ; " '0d'x;

\* Valid values;

PUT @2 "ELSE &PREFIX" name " = " name ";" '0d'x;

\* Capping and flooring;

%if %upcase(&&cap\_flr&typ) = Y %then %do;

Put "&PREFIX" name " = MIN(MAX(&PREFIX" name ", " Var\_LB +(-1) "), " var\_UB +(-1) ");" '0d'x;

%end;

\* Untransformed variable label;

%if &typ = O %then %do;

PUT @2 "Label &PREFIX" name ' = "' lab2 ': Ordinal Recode ' Sign '";' '0d'x;

%end;

%else %do;

PUT @2 "Label &PREFIX" name ' = "' lab2 ': Continuous Recode ' Sign '";' '0d'x;

%end;

\*standard transformation;

if \_Trans='SQ\_' then do; Put new\_var "= &PREFIX" name "\*\*2;" '0d'x;

Put @2 " Label " new\_var '= "' lab2 'SQUARE ' Sign '";' '0d'x; end;

else if \_Trans='SR\_' then do; Put new\_var " = SQRT(MAX(&PREFIX" name ",0));" '0d'x;

Put @2 " Label " new\_var '= "' lab2 'SQRT ' Sign '";' '0d'x; end;

else if \_Trans='LN\_' then do; Put new\_var "= LOG(MAX(&PREFIX" name ",0.00001));" '0d'x;

Put @2 " Label " new\_var '= "' lab2 'LOG ' Sign '";' '0d'x; end;

else if \_Trans='IV\_' then do; Put new\_var "= -1/(MAX(&PREFIX" name ",0.00001));" '0d'x;

Put @2 " Label " new\_var '= "' lab2 'NEGATIVE INVERSE ' Sign '";' '0d'x; end;

else if \_Trans='EP\_' then do; Put new\_var "= -EXP(MIN(-&PREFIX" name ",0));" '0d'x;

Put @2 " Label " new\_var '= "' lab2 'EXPONENTIAL ' Sign '";' '0d'x; end;

\* Standardization;

%if %upcase("&&stdmethod&typ") ^= "NO" %then %do;

if (location ^= . and scale ^= .) then put new\_var "= (" new\_var "- (" location +(-1) ") ) / " scale ";" '0d'x;

%end;

run;

%end;

proc datasets nolist; delete parm nonmissing; quit; run;

%mend pnum;

\*\* Ordinal and continuous profiling macro 1 \*\*;

%macro prof1(insdn,var);

proc summary data=&insdn nway missing;

var &dep\_var;

class &var;

output out=prof (drop=\_type\_ rename=\_freq\_=xcount) mean=xmean;

run;

data prof; set prof;

length xcategory $256.;

if missing(&var) then xcategory = 'Missing';

else xcategory = trim(left(put(&var,best8.)));

run;

%mend;

\*\* Ordinal and continuous profiling macro 2 \*\*;

%macro prof2(insdn,var);

%if %upcase(&equal\_dist) = Y %then %do;

data \_NULL\_; set EDA;

range = (var\_p99 - var\_p1)/&num\_category;

call symputx('cut\_lo',var\_p1);

call symputx('cut\_hi',var\_p99);

call symputx('range',range);

run;

%put &range;

data tmp; set &insdn (keep=&dep\_var &var);

if missing(&var) then bin = .;

else if &var < &cut\_lo then bin = 1;

else if &var >= &cut\_hi then bin = input("&num\_category",best10.);

else do;

do k = 1 to &num\_category;

if &var>= &cut\_lo+(k-1)\*&range and &var< &cut\_lo+k\*&range then bin=k;

end;

end;

run;

%end;

%else %do;

proc rank data=&insdn (keep=&dep\_var &var) out=tmp ties=High group=&num\_category;

var &var;

ranks bin;

run;

%end;

proc summary data=tmp nway missing;

var &var &dep\_var;

class bin;

output out=prof (drop=\_type\_ rename=\_freq\_=xcount) min(&var)=lo max(&var)=hi mean(&dep\_var)=xmean;

run;

data prof (drop=tag); set prof end=eof;

length xcategory $256.;

retain tag 0;

if missing(bin) then do;

xcategory = 'Missing';

tag = 1;

end;

else if \_N\_ = 1 or (\_N\_ = 2 and tag = 1) then xcategory = "Low to " || trim(left(put(hi,best8.)));

else if eof then xcategory = trim(left(put(lo,best8.))) || " to High";

else xcategory = trim(left(put(lo,best8.))) || " to " || trim(left(put(hi,best8.)));

run;

proc datasets nolist; delete tmp eda; run;

%mend;

\*\* Ordinal and continuous profiling macro 3 \*\*;

%macro prof3(typ);

proc sql;

create table prof2 as

select a.name as variable, a.label, b.xcategory,

b.xcount, b.xmean

from vars2 a, prof b

where a.name = "&var";

quit;

data prof3 (drop=xcount xmean xcategory); set prof2 end=eof;

length category $256.;

length type $10.;

length star $8.;

if "&typ" = "C" then type = 'Continuous';

else type = 'Ordinal';

if \_N\_ = 1 then do;

category = "Overall";

Average\_DV = &overall\_avg;

Count = &nobs;

Percent = 1;

index = 100;

output;

end;

category = xcategory;

count = xcount;

percent = xcount / &nobs;

Average\_DV = xmean;

index = (Average\_DV / &overall\_avg)\*100;

if index >= 110 then star = '\* (+)';

else if index > 100 then star = ' (+)';

else if index <= 90 then star = '\* (-)';

else if index <= 100 then star = ' (-)';

else star = ' (0)';

output;

run;

data out.CE2\_profile; set out.CE2\_profile prof3; run;

proc datasets nolist; delete prof prof2 prof3; run;

%mend;

\*\* Binary variable processing macro \*\*;

%macro pbin(insdn,var,typ);

%\*\* Get missing count \*\*;

proc sql noprint;

select max(sum(missing(&var)),0) into : miss\_ck from &insdn;

quit;

%put processing binary variable &var with type &typ;

%if %sysevalf((&nobs\*&missrate)>=&miss\_ck) %then %put missing count is &miss\_ck;

%else %put missing count is &miss\_ck which is too high;

%\*\* If missing count is acceptable \*\*;

%if %sysevalf((&nobs\*&missrate)>=&miss\_ck) %then %do;

%\*\* Get counts of values that should be coded as 1 \*\*;

%if &typ = 1 %then %do; \*\* Numeric variables \*\*;

proc sql noprint;

select sum(case when &var = 1 then 1 else 0 end) into : cnt\_ck from &insdn;

quit;

%end;

%else %do; %\*\* Character variables \*\*;

proc sql noprint;

select sum(case when strip(&var) in ('1','Y','y') then 1 else 0 end) into : cnt\_ck from &insdn;

quit;

%end;

%if %sysevalf((&nobs\*(1-&concrate)) <= &cnt\_ck and &cnt\_ck <= (&nobs\*&concrate)) %then %do;

%\*\* If count is okay, write out code to file \*\*;

data \_null\_; set vars (where=(name="&var"));

FILE "&Path\_output.CE2\_Binary\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT "\*\*\* RECODE " name ": " label " \*\*\*;" '0d'x;

%if &typ = 1 %then %do;

PUT "if " name " = 1 then &PREFIX" name " = 1; else &PREFIX" name " = 0;" '0d'x;

%end;

%else %do;

PUT "if " name " in ('1','Y','y') then &PREFIX" name " = 1; else &PREFIX" name " = 0;" '0d'x;

%end;

if missing(label) then PUT @2 "Label &PREFIX" name ' = "' name ': Binary Recode";' '0d'x;

else PUT @2 "Label &PREFIX" name ' = "' label ': Binary Recode";' '0d'x;

run;

%if %upcase(&profiling) = Y %then %do;

%if &typ = 1 %then %do;

data tmp1; set &insdn (keep=&dep\_var &var);

if &var = 1 then newvar = 1;

else newvar = 0;

run;

%end;

%else %do;

data tmp1; set &insdn;

if &var in ('1','Y','y') then newvar = 1;

else newvar = 0;

run;

%end;

proc sql;

create table tmp2 as

select a.name as variable, a.label, b.newvar,

count(\*) as xcount,

mean(b.&dep\_var) as xmean

from vars a, tmp1 b

where a.name = "&var"

group by a.name, a.label, b.newvar

order by a.name, a.label, b.newvar;

quit;

data tmp3 (drop=xcount xmean newvar); set tmp2 end=eof;

by newvar;

length type $10.;

length category $256.;

length star $8.;

type = 'Binary';

if \_N\_ = 1 then do;

category = "Overall";

Average\_DV = &overall\_avg;

Count = &nobs;

Percent = 1;

index = 100;

output;

end;

count = xcount;

percent = xcount / &nobs;

Average\_DV = xmean;

index = (Average\_DV / &overall\_avg)\*100;

%if &typ = 1 %then %do;

if newvar=0 then category = "Missing,0";

else category = "1";

%end;

%else %do;

if newvar=0 then category = "Missing,0,N";

else category = "1,Y";

%end;

if index >= 110 then star = '\* (+)';

else if index > 100 then star = ' (+)';

else if index <= 90 then star = '\* (-)';

else if index <= 100 then star = ' (-)';

else star = ' (0)';

output;

run;

data out.CE2\_profile; set out.CE2\_profile tmp3; run;

proc datasets nolist; delete tmp: ; run;

%end;

%end;

%else %do; %\*\* If variable is too concentrated, write note to log \*\*;

%put variable &var is too concentrated to use;

%end;

%\*\* Add counts to summary file \*\*;

data vars; set vars;

if name = "&var" then do;

miss\_cnt = &miss\_ck;

ones\_cnt = &cnt\_ck;

%if %sysevalf((&nobs\*(1-&concrate)) <= &cnt\_ck and &cnt\_ck <= (&nobs\*&concrate)) %then %do;

good = 1;

%end;

end;

run;

%end;

%\*\* If missing count is too high \*\*;

%else %do;

data vars; set vars;

if name = "&var" then do;

miss\_cnt = &miss\_ck;

end;

run;

%end;

%mend pbin;

\*\* Binary processing control macro \*\*;

%macro bin\_cntl(insdn);

%\*\* Initialize code file \*\*;

data \_null\_;

FILE "&Path\_output.CE2\_Binary\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_BINARY\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

run;

%\*\* Get variables to process \*\*;

proc contents data=&insdn (keep=&binvar)

out=vars (keep=varnum name type length label)

noprint;

run;

proc sql noprint;

select count(\*) into : ck from vars;

quit;

%put number of binary variables = &ck;

%\*\* If there are binary variables to process \*\*;

%if &ck > 0 %then %do;

%\*\* Add new variables to summary file \*\*;

data vars; set vars;

length miss\_cnt ones\_cnt good 8.;

run;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set vars (where=(length(strip(name)) + length(strip("&prefix")) <= 32)) end=eof;

if eof then call symputx("VARCNT",\_N\_);

call symputx("IV"|| trim(left(put(\_N\_,4.))) ,name);

call symputx("typ"|| trim(left(put(\_N\_,4.))) ,type);

run;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%pbin(&insdn,&&iv&\_I\_,&&typ&\_I\_);

%end;

%\*\* Create new variable name for surviving variables \*\*;

data out.CE2\_binary\_vars (drop=good); set vars;

length new\_var $32.;

if good = 1 then new\_var = "&prefix"||name;

run;

%\*\* Write out list of variables to keep to code file \*\*;

%let rck = 0;

data \_NULL\_; set out.CE2\_binary\_vars (where=(missing(new\_var)=0)) end=eof;

m = mod(\_N\_,7);

length v $256.;

retain v ;

if m = 1 then v = new\_var;

else v = strip(v) || " " || strip(new\_var);

FILE "&Path\_output.CE2\_Binary\_Var\_Recode.txt" mod;

if \_N\_=1 then do;

PUT ' ' '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_B =';

end;

if m = 0 or eof then PUT v '0d'x;

if eof then PUT ';';

if eof then call symputx("rck" ,\_N\_);

run;

%put number of recoded binary variables = &rck;

%\* Capture old and new variable name;

data tmp (keep=orig\_var orig\_label variable);

length variable orig\_var $32.;

length orig\_label $256.;

set out.CE2\_binary\_vars (where=(missing(variable)=0) rename=(name=orig\_var label=orig\_label new\_var=variable));

run;

data keep\_vars; set keep\_vars tmp; run;

proc datasets nolist; delete vars tmp; quit; run;

%end;

%\*\* If there are no binary variables to process \*\*;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Binary\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_B = ;' '0d'x;

run;

%end;

%mend bin\_cntl;

\*\* Nominal variable processing macro \*\*;

%macro pnom(insdn,var,typ);

%\*\* Summarize \*\*;

proc sql;

create table tmp as

select &var, count(\*) as dcount, mean(&dep\_var) as dmean, var(&dep\_var) as dvar

from &insdn

group by &var

order by dmean;

quit;

proc sql noprint;

select sum(case when missing(&var) then dcount else 0 end) into : misscnt from tmp;

select max(dcount) into : maxcnt from tmp;

select count(\*) into : unqcnt from tmp;

quit;

%\*\* Test \*\*;

%put processing nominal variable &var with type &typ;

%if %sysevalf((&nobs\*&missrate)<&misscnt) %then %put missing count is &misscnt which is too high;

%else %do;

%put missing count is &misscnt;

%if %sysevalf((&nobs\*&concrate)<&maxcnt) %then %put variable &var is too concentrated to use;

%else %do;

%if %sysevalf(&valcnt^=0 and &valcnt<=&unqcnt) %then %put variable &var has too many values to use;

%else %do; %\*\* Variable is usable;

%\*\* Collapse values based on counts;

DATA tmp1; SET tmp END=eof;

LENGTH tcount tmean tvar tgroup cumcnt tgrpcnt 8.;

RETAIN tcount tmean tvar tgroup cumcnt tgrpcnt;

cumcnt + dcount;

IF \_n\_ = 1 THEN DO;

tcount = dcount;

tmean = dmean;

tvar = dvar;

tgroup = 1;

tgrpcnt = 1;

END;

ELSE DO;

IF tcount <= &minbinn or (cumcnt - dcount) >= (&nobs - &minbinn) THEN DO;

tmean = ((tcount \* tmean) + (dcount \* dmean))/(tcount + dcount);

tvar = (((tcount - 1) \* tvar) + ((dcount - 1) \* dvar))/(tcount + dcount - 2);

tcount = tcount + dcount;

tgroup = tgroup;

tgrpcnt = tgrpcnt + 1;

END;

ELSE DO;

tcount = dcount;

tmean = dmean;

tvar = dvar;

tgroup = tgroup+1;

tgrpcnt = 1;

END;

END;

KEEP &var dcount dmean dvar tgroup tcount tmean tvar tgrpcnt;

RUN;

%\*\* Create dataset with final record for group only;

data tmp2; set tmp1 (drop=&var dcount dmean dvar);

by tgroup;

if last.tgroup;

run;

%\*\* Calculate bonferroni adjustment on alpha value;

%IF %UPCASE(&bonfer) = Y %THEN %DO;

PROC SQL NOPRINT;

SELECT COUNT(DISTINCT group)

INTO :ncomps

FROM tmp2;

QUIT;

%LET ncomps = %SYSEVALF(&ncomps - 1);

%IF &ncomps > 1 %THEN %LET ftalpha = %SYSEVALF(&talpha./&ncomps);

%ELSE %LET ftalpha = &talpha;

%END;

%ELSE %LET ftalpha = &talpha;

%\*\* Collapse values based on variance;

DATA tmp3; SET tmp2 END=eof;

LENGTH row fcount fmean fvar fgroup grpcnt 8.;

RETAIN fcount fmean fvar fgroup grpcnt;

row = \_N\_;

IF \_n\_ = 1 THEN DO;

fcount = tcount;

fmean = tmean;

fvar = tvar;

fgroup = 1;

grpcnt = tgrpcnt;

END;

ELSE DO;

pvar = (((fcount - 1) \* fvar) + ((tcount - 1) \* tvar))/(fcount + tcount - 2);

t = (fmean - tmean)/SQRT(pvar \* ((1/fcount) + (1/tcount)));

df = (fcount + tcount - 2);

prob = (1-PROBT(abs(t),df));

IF prob <= &ftalpha THEN DO;

fcount = tcount;

fmean = tmean;

fvar = tvar;

fgroup = fgroup+1;

grpcnt = tgrpcnt;

END;

ELSE DO;

fmean = ((fcount \* fmean) + (tcount \* tmean))/(fcount + tcount);

fvar = (((fcount - 1) \* fvar) + ((tcount - 1) \* tvar))/(fcount + tcount - 2);

fcount = fcount + tcount;

fgroup = fgroup;

grpcnt = grpcnt + tgrpcnt;

END;

END;

KEEP row tgroup fgroup fcount fmean grpcnt;

RUN;

PROC SORT DATA = tmp3 out=tmp4; BY fgroup DESCENDING row; RUN;

DATA tmp4 (drop=fcount fmean row grpcnt); SET tmp4;

LENGTH xmean xcount group\_size 8. name $32.;

RETAIN xmean xcount group\_size;

BY fgroup DESCENDING row;

IF FIRST.fgroup THEN DO;

xmean = fmean;

xcount = fcount;

group\_size = grpcnt;

END;

Index = (xmean/&overall\_Avg)\*100 ;

diff = abs(xmean-&overall\_Avg);

name="&var";

RUN;

%\* Sort to keep smaller groups first;

PROC SORT DATA = tmp4; BY group\_size DESCENDING diff xcount ; RUN;

DATA \_NULL\_; SET tmp4 END=eof;

IF eof THEN call symputx('last\_group',fgroup);

RUN;

%\* Combine data;

proc sql;

create table tmp5 as

select b.name, c.label, a.&var, b.fgroup, a.dmean, a.dcount, b.xmean,

b.xcount, b.diff, b.Index, b.group\_size

from tmp1 a, tmp4 b, vars c

where a.tgroup = b.tgroup

and b.name = c.name

order by b.group\_size, b.diff desc, b.xcount;

quit;

%\* Write out recode code;

%if %upcase(&nom\_method) = BINARY %then %do;

DATA \_null\_; SET tmp5 (where=(fgroup ^= &last\_group)) END=eof;

BY group\_size DESCENDING diff ;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

IF \_n\_ = 1 THEN DO;

PUT ' ' '0d'x;

PUT "\*\*\* RECODE " name ": " label "\*\*\*;" '0d'x;

END;

%if &typ = 1 %then %do;

IF FIRST.diff THEN PUT "&PREFIX" name +(-1) "\_X" fgroup "= " name "in (" &var +(-1)@;

IF FIRST.diff NE 1 THEN PUT "," &var +(-1)@;

IF LAST.diff THEN PUT ");" '0d'x;

%end;

%else %do;

IF FIRST.diff THEN PUT "&PREFIX" name +(-1) "\_X" fgroup "= " name "in ('" &var +(-1)@;

IF FIRST.diff NE 1 THEN PUT "','" &var +(-1)@;

IF LAST.diff THEN PUT "');" '0d'x;

%end;

RUN;

DATA \_null\_; SET tmp5 (where=(fgroup ^= &last\_group)) END=eof;

BY group\_size DESCENDING diff ;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

IF FIRST.diff THEN DO;

IF missing(label) then PUT @2 "Label &PREFIX" name +(-1) "\_X" fgroup '= "' name ': Values ' &var +(-1)@;

ELSE PUT @2 "Label &PREFIX" name +(-1) "\_X" fgroup '= "' label ': Values ' &var +(-1)@;

END;

IF FIRST.diff NE 1 THEN PUT "," &var +(-1)@;

IF LAST.diff THEN PUT '";' '0d'x;

RUN;

data nv (keep=name new\_var); set tmp5 (where=(fgroup ^= &last\_group));

by group\_size DESCENDING diff ;

if first.diff;

length new\_var $32.;

new\_var = "&prefix" || strip(name) || "\_X" || strip(left(put(fgroup,best2.)));

run;

data new\_vars; set new\_vars nv; run;

%end;

%else %do;

%if %upcase(&nom\_method) = INDEX %then %let val =index;

%else %let val = xmean;

proc sort data=tmp5; by fgroup group\_size &var; run;

proc sql noprint;

select max(fgroup) into : last\_group from tmp5;

quit;

DATA \_null\_; SET tmp5 END=eof;

BY fgroup group\_size;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

IF \_n\_ = 1 THEN DO;

IF fgroup = &last\_group THEN STOP;

ELSE DO;

PUT ' ' '0d'x;

PUT "\*\*\* RECODE " name ": " label "\*\*\*;" '0d'x;

END;

END;

IF fgroup = &last\_group THEN DO;

IF first.group\_size THEN PUT "else &PREFIX" name "= " &val ";" '0d'x;

END;

ELSE DO;

%if &typ = 1 %then %do;

IF \_n\_ = 1 AND FIRST.group\_size THEN PUT "if " name "in (" &var +(-1)@;

ELSE IF FIRST.group\_size THEN PUT "else if " name "in (" &var +(-1)@;

IF FIRST.group\_size NE 1 THEN PUT "," &var +(-1)@;

IF LAST.group\_size THEN PUT ") then &PREFIX" name "= " &val ";" '0d'x;

%end;

%else %do;

IF \_n\_ = 1 AND FIRST.group\_size THEN PUT "if " name "in ('" &var +(-1)@;

ELSE IF FIRST.group\_size THEN PUT "else if " name "in ('" &var +(-1)@;

IF FIRST.group\_size NE 1 THEN PUT "','" &var +(-1)@;

IF LAST.group\_size THEN PUT "') then &PREFIX" name "= " &val ";" '0d'x;

%end;

END;

RUN;

DATA \_null\_; SET tmp5 (where=(fgroup ^= &last\_group) obs=1) END=eof;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

IF missing(label) then PUT @2 "Label &PREFIX" name ' = "' name ': Nominal Recode";' '0d'x;

ELSE PUT @2 "Label &PREFIX" name ' = "' label ': Nominal Recode";' '0d'x;

RUN;

data nv (keep=name new\_var); set tmp5 (where=(fgroup ^= &last\_group) obs=1);

length new\_var $32.;

new\_var = "&prefix" || strip(name);

run;

data new\_vars; set new\_vars nv; run;

%end;

%if %upcase(&Profiling) = Y %then %do;

proc sort data=tmp5; by fgroup group\_size &var; run;

data tmp6 (keep=type Variable label category count percent Average\_DV index star);

set tmp5 (rename=(name=Variable)) end=eof;

by fgroup;

length type $10.;

length category $256.;

length star $8.;

retain category;

type = 'Nominal';

if \_N\_ = 1 then do;

category = "Overall";

Average\_DV = &overall\_avg;

Count = &nobs;

Percent = 1;

index = 100;

output;

end;

count = xcount;

percent = xcount / &nobs;

Average\_DV = xmean;

Index = (Average\_DV / &overall\_avg)\*100;

if first.fgroup and missing(&var) then category = "Missing";

else if first.fgroup then category = strip(&var);

else category = strip(category) || "," || strip(&var);

if index >= 110 then star = '\* (+)';

else if index > 100 then star = ' (+)';

else if index <= 90 then star = '\* (-)';

else if index <= 100 then star = ' (-)';

else star = ' (0)';

if last.fgroup then output;

run;

data out.CE2\_profile; set out.CE2\_profile tmp6; run;

%end;

%end;

%end;

%end;

%\*\* Add counts to summary file \*\*;

data vars; set vars;

if name = "&var" then do;

unique\_cnt = &unqcnt;

miss\_cnt = &misscnt;

max\_cnt = &maxcnt;

end;

run;

proc datasets nolist; delete tmp: nv; quit; run;

%mend pnom;

\*\* Nominal processing control macro \*\*;

%macro nom\_cntl(insdn);

%\*\* Initialize code file \*\*;

data \_null\_;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_NOMINAL\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

run;

%\*\* Get variables to process \*\*;

proc contents data=&insdn (keep=&nomvar)

out=vars (keep=varnum name type length label)

noprint;

run;

proc sql noprint;

select count(\*) into : ck from vars;

quit;

%put number of nominal variables = &ck;

%\*\* If there are nominal variables to process \*\*;

%if &ck > 0 %then %do;

%\*\* Add new variables to summary file \*\*;

data vars; set vars;

length unique\_cnt miss\_cnt max\_cnt 8.;

run;

%if %upcase(&nom\_method) = BINARY %then %let maxlen=29;

%else %let maxlen=32;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set vars (where=(length(strip(name)) + length(strip("&prefix")) <= &maxlen)) end=eof;

if eof then call symputx("VARCNT",\_N\_);

call symputx("IV"|| trim(left(put(\_N\_,4.))) ,name);

call symputx("typ"|| trim(left(put(\_N\_,4.))) ,type);

run;

data new\_vars; set \_NULL\_;

length name new\_var $32.;

run;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%pnom(&insdn,&&iv&\_I\_,&&typ&\_I\_);

%end;

%\*\* Create new variable name for surviving variables \*\*;

proc sort data=new\_vars; by name; run;

data nv; set new\_vars;

by name;

length new\_vars $256.;

retain new\_vars;

if first.name then new\_vars = new\_var;

else new\_vars = strip(new\_vars)||","||strip(new\_var);

if last.name then output;

run;

proc sql;

create table out.CE2\_nominal\_vars as

select a.\*, b.new\_vars

from vars a

left join nv b on a.name = b.name;

quit;

%\*\* Write out list of variables to keep to code file \*\*;

%let rck = 0;

data \_NULL\_; set new\_vars (where=(missing(new\_var)=0)) end=eof;

m = mod(\_N\_,7);

length v $256.;

retain v ;

if m = 1 then v = new\_var;

else v = strip(v) || " " || strip(new\_var);

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

if \_N\_=1 then do;

PUT ' ' '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_N =';

end;

if m = 0 or eof then PUT v '0d'x;

if eof then PUT ';';

if eof then call symputx("rck" ,\_N\_);

run;

%put number of recoded nominal variables = &rck;

%\* Capture old and new variable name;

proc sql;

create table tmp as

select a.name as orig\_var, a.label as orig\_label, b.new\_var as variable

from vars a, new\_vars b

where a.name = b.name;

quit;

data keep\_vars; set keep\_vars tmp; run;

proc datasets nolist; delete vars new\_vars nv tmp; quit; run;

%end;

%\*\* If there are no nominal variables to process \*\*;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_N = ;' '0d'x;

run;

%end;

%mend nom\_cntl;

\*\* Ordinal variable processing macro \*\*;

%macro pord(insdn,var,nmiss);

%\*\* Summarize \*\*;

proc sql;

create table tmp as

select &var, count(\*) as count

from &insdn

group by &var;

quit;

proc sql noprint;

select max(count) into : maxcnt from tmp;

select count(\*) into : unqcnt from tmp;

quit;

%\* check if dep\_var has the constant value in nonmissing part;

proc sql noprint;

select min(&dep\_var)=max(&dep\_var) into : cons\_dep from &insdn

where missing(&var)=0;

quit;

proc datasets nolist; delete tmp; quit; run;

%\* add to vars file;

data vars2; set vars2;

if name = "&var" then do;

unique\_cnt = &unqcnt;

max\_cnt = &maxcnt;

end;

run;

%\*\* Test \*\*;

%if %sysevalf((&nobs\*&concrate)<&maxcnt) %then %put variable &var is too concentrated to use;

%else %if &cons\_dep = 1 %then %put &var has constant dependent value in nonmissing part and will be excluded;

%else %do;

%pnum(&insdn,&var,&nmiss,O);

%\*\* Profiling \*\*;

%if %upcase(&profiling) = Y %then %do;

%if &unqcnt <= &num\_category %then %do;

%prof1(&insdn,&var);

%end;

%else %do;

%prof2(&insdn,&var);

%end;

%prof3(O);

%end;

%end;

%mend pord;

\*\* Ordinal processing control macro \*\*;

%macro ord\_cntl(insdn);

%\*\* Initialize code file \*\*;

data \_null\_;

FILE "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_ORDINAL\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

run;

%\*\* Get variables to process \*\*;

proc contents data=&insdn (keep=&ordvar)

out=vars (keep=varnum name type length label)

noprint;

run;

proc sql noprint;

select max(type) into : ck\_type from vars;

quit;

%if &ck\_type = 2 %then %do;

proc sql noprint; select name into : bad separated by ',' from vars where type = 2; quit;

%put The following character variables were dropped from the ordinal variable list:;

%put &bad;

proc sql noprint;

select name into : ordvar separated by ' ' from vars where type=1;

quit;

%end;

data vars; set vars (where=(type=1)); run;

proc sql noprint;

select count(\*) into : ck from vars;

quit;

%put number of ordinal variables = &ck;

%\*\* If there are ordinal variables to process \*\*;

%if &ck > 0 %then %do;

%\*\* Get missing counts to check variable validity \*\*;

proc means data=&insdn noprint ;

var &ordvar;

output out=nmiss (drop=\_type\_ \_freq\_) nmiss= ;

run;

proc transpose data=nmiss out=nmiss name=name; run;

%\*\* Combine \*\*;

proc sql;

create table vars2 as

select a.\*, b.col1 as miss\_cnt

from vars a

left join nmiss b on a.name = b.name;

quit;

data vars2; set vars2;

length new\_var $32.;

length sign $3.;

length relationship $6.;

length max\_cnt unique\_cnt 8.;

run;

%if %upcase(&transformationO) = Y %then %let maxlen=29;

%else %let maxlen=32;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set vars2 (where=(length(strip(name)) + length(strip("&prefix")) <= &maxlen)) end=eof;

if eof then call symput("VARCNT",\_N\_);

call symput("IV"|| trim(left(put(\_N\_,4.))) ,name);

call symput("miss"|| trim(left(put(\_N\_,4.))) ,miss\_cnt);

run;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%put processing ordinal variable &&iv&\_I\_;

%if %sysevalf((&nobs\*&missrate)<&&miss&\_I\_) %then %put missing count is &&miss&\_I\_ which is too high;

%else %do;

%put missing count is &&miss&\_I\_ ;

%pord(&insdn,&&iv&\_I\_,&&miss&\_I\_);

%end;

%end;

%\*\* Create permanent variable summary file \*\*;

data out.CE2\_ordinal\_vars; set vars2; run;

%\*\* Write out list of variables to keep to code file \*\*;

%let rck = 0;

data \_NULL\_; set out.CE2\_ordinal\_vars (where=(missing(new\_var)=0)) end=eof;

m = mod(\_N\_,7);

length v $256.;

retain v ;

if m = 1 then v = new\_var;

else v = strip(v) || " " || strip(new\_var);

FILE "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt" mod;

if \_N\_=1 then do;

PUT ' ' '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_O =';

end;

if m = 0 or eof then PUT v '0d'x;

if eof then PUT ';';

if eof then call symputx("rck" ,\_N\_);

run;

%put number of recoded ordinal variables = &rck;

%\* Capture old and new variable name;

data tmp (keep=orig\_var orig\_label variable);

length variable orig\_var $32.;

length orig\_label $256.;

set out.CE2\_ordinal\_vars (where=(missing(variable)=0) rename=(name=orig\_var label=orig\_label new\_var=variable));

run;

data keep\_vars; set keep\_vars tmp; run;

proc datasets nolist; delete vars vars2 tmp; quit; run;

%end;

%\*\* If there are no ordinal variables to process \*\*;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_O = ;' '0d'x;

run;

%end;

%mend ord\_cntl;

\*\* Continuous variable processing macro \*\*;

%macro pcont(insdn,var,nmiss);

%\* check if dep\_var has the constant value in nonmissing part;

proc sql noprint;

select min(&dep\_var)=max(&dep\_var) into : cons\_dep from &insdn

where missing(&var)=0;

quit;

%if &cons\_dep = 1 %then %put &var has constant dependent value in nonmissing part and will be excluded;

%else %do;

%pnum(&insdn,&var,&nmiss,C);

%\*\* Profiling \*\*;

%if %upcase(&profiling) = Y %then %do;

%prof2(&insdn,&var);

%prof3(C);

%end;

%end;

%mend pcont;

\*\* Continuous processing control macro \*\*;

%macro cont\_cntl(insdn);

%\*\* Initialize code file \*\*;

data \_null\_;

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_CONTINUOUS\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

run;

%\*\* Get variables to process \*\*;

proc contents data=&insdn (keep=&contvar)

out=vars (keep=varnum name type length label)

noprint;

run;

proc sql noprint;

select max(type) into : ck\_type from vars;

quit;

%if &ck\_type = 2 %then %do;

proc sql noprint; select name into : bad separated by ',' from vars where type = 2; quit;

%put The following character variables were dropped from the continuous variable list:;

%put &bad;

proc sql noprint;

select name into : contvar separated by ' ' from vars where type=1;

quit;

%end;

data vars; set vars (where=(type=1)); run;

proc sql noprint;

select count(\*) into : ck from vars;

quit;

%put number of continuous variables = &ck;

%\*\* If there are continuous variables to process \*\*;

%\*\* If there are continuous variables to process \*\*;

%if &ck > 0 %then %do;

%\*\* Get statistics to check variable validity \*\*;

proc means data=&insdn noprint ;

var &contvar;

output out=nmiss nmiss= ;

run;

proc transpose data=nmiss out=nmiss name=name; run;

%\*\* Combine \*\*;

proc sql;

create table vars2 as

select a.\*, b.col1 as miss\_cnt

from vars a

left join nmiss b on a.name = b.name;

quit;

%\*\* Add new variables to summary file \*\*;

data vars2; set vars2;

length P&p\_lo P&p\_hi 8.;

length new\_var $32.;

length sign $3.;

length relationship $6.;

run;

proc datasets nolist; delete nmiss; quit; run;

%if %upcase(&transformationC) = Y %then %let maxlen=29;

%else %let maxlen=32;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set vars2 (where=(length(strip(name)) + length(strip("&prefix")) <= &maxlen)) end=eof;

if eof then call symputx("VARCNT",\_N\_);

call symputx("IV"|| trim(left(put(\_N\_,4.))) ,name);

call symputx("miss"|| trim(left(put(\_N\_,4.))) ,miss\_cnt);

run;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%put processing continuous variable &&iv&\_I\_;

%if %sysevalf((&nobs\*&missrate)<&&miss&\_I\_) %then %put missing count is &&miss&\_I\_ which is too high;

%else %do;

%put missing count is &&miss&\_I\_ ;

proc means data=&insdn noprint ;

var &&iv&\_I\_;

output out=st P&p\_lo=P\_lo P&p\_hi=P\_hi ;

run;

data \_null\_; set st;

call symputx("lo&\_I\_" ,P\_lo);

call symputx("hi&\_I\_",P\_hi);

run;

data vars2; set vars2;

if name = "&&iv&\_I\_" then do;

P&p\_lo = &&lo&\_I\_;

P&p\_hi = &&hi&\_I\_;

end;

run;

%if %sysevalf(&&lo&\_I\_ = &&hi&\_I\_) %then %put variable has constant value and will be excluded;

%else %do;

%pcont(&insdn,&&iv&\_I\_,&&miss&\_I\_);

%end;

%end;

%end;

%\*\* Create permanent variable summary file \*\*;

data out.CE2\_continuous\_vars; set vars2; run;

%\*\* Write out list of variables to keep to code file \*\*;

%let rck = 0;

data \_NULL\_; set out.CE2\_continuous\_vars (where=(missing(new\_var)=0)) end=eof;

m = mod(\_N\_,7);

length v $256.;

retain v ;

if m = 1 then v = new\_var;

else v = strip(v) || " " || strip(new\_var);

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" mod;

if \_N\_=1 then do;

PUT ' ' '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_C =';

end;

if m = 0 or eof then PUT v '0d'x;

if eof then PUT ';';

if eof then call symputx("rck" ,\_N\_);

run;

%if &rck = 0 %then %do;

data \_null\_;

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_C = ;' '0d'x;

run;

%end;

%put number of recoded continuous variables = &rck;

%\* Capture old and new variable name;

data tmp (keep=orig\_var orig\_label variable);

length variable orig\_var $32.;

length orig\_label $256.;

set out.CE2\_continuous\_vars (where=(missing(variable)=0) rename=(name=orig\_var label=orig\_label new\_var=variable));

run;

data keep\_vars; set keep\_vars tmp; run;

proc datasets nolist; delete vars vars2 tmp; quit; run;

%end;

%\*\* If there are no continuous variables to process \*\*;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" mod;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_C = ;' '0d'x;

run;

%end;

%mend cont\_cntl;

\*\* Master EDA and Recode macro \*\*;

%macro CE\_EDA\_Recode(insdn=out.CE1\_Resampled);

%\*\* Extract working portion of file \*\*;

data workfile; set &insdn (where=(mod\_val\_test ^= 3)); run;

%\* Get global numbers;

%global nobs overall\_avg minbinn;

proc sql noprint;

select count(\*) into : nobs from workfile;

select mean(&dep\_var) into : overall\_avg from workfile;

quit;

data \_null\_;

call symputx("minbinn",max(&minbinnc,(&nobs\*&minbinnp))); \*\* minimum bin size for nominal;

run;

%\*\* Initialize profiling dataset \*\*;

%if %upcase(&profiling) = Y %then %do;

data out.CE2\_profile; set \_NULL\_;

length type $10.;

length variable $32.;

length label $256.;

length category $256.;

format count comma8.;

format percent percent8.2;

%if %upcase(&binary\_dv) = Y %then %do;

format Average\_DV percent8.2;

%end;

%else %do;

format Average\_DV 12.2;

%end;

format index 8.0;

length star $8.;

run;

%end;

%\*\* Initialize variable name lookup dataset \*\*;

data keep\_vars; set \_NULL\_;

length orig\_var $32.;

length orig\_label $256.;

length variable $32.;

run;

%if %symexist(binvar) %then %do;

%bin\_cntl(workfile);

%end;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Binary\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_BINARY\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_B = ;' '0d'x;

run;

%end;

%if %symexist(nomvar) %then %do;

%nom\_cntl(workfile);

%end;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Nominal\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_NOMINAL\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_N = ;' '0d'x;

run;

%end;

%if %symexist(ordvar) %then %do;

%ord\_cntl(workfile);

%end;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_ORDINAL\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_O = ;' '0d'x;

run;

%end;

%if %symexist(contvar) %then %do;

%cont\_cntl(workfile);

%end;

%else %do;

data \_null\_;

FILE "&Path\_output.CE2\_Continuous\_Var\_Recode.txt" LRECL=256;

PUT "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x/

"\*\*\*\* CE2\_CONTINUOUS\_VAR\_RECODE.TXT \*\*\*\*;" '0d'x/

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;" '0d'x;

PUT ' ' '0d'x;

PUT '%LET KEEPLIST\_C = ;' '0d'x;

run;

%end;

%\*\* Create dataset with recoded variables \*\*;

data out.CE2\_Recoded; set &insdn;

%inc "&Path\_output.CE2\_Binary\_Var\_Recode.txt";

%inc "&Path\_output.CE2\_Nominal\_Var\_Recode.txt";

%inc "&Path\_output.CE2\_Ordinal\_Var\_Recode.txt";

%inc "&Path\_output.CE2\_Continuous\_Var\_Recode.txt";

keep &keep\_list &KEEPLIST\_B &KEEPLIST\_N &KEEPLIST\_O &KEEPLIST\_C;

run;

%\*\* Finalize variable name lookup dataset \*\*;

proc contents data=out.CE2\_Recoded out=vars (keep=name label) noprint; run;

proc sql;

create table out.CE2\_vars as

select a.variable, b.label, a.orig\_var, a.orig\_label

from keep\_vars a, vars b

where a.variable = b.name

order by a.variable;

quit;

%\*\* Create EDA report in Excel;

ods listing close;

ods Tagsets.ExcelxP body="&Path\_output.CE2\_EDA\_report.xls" style=sasweb;

%if %sysfunc(exist(out.CE2\_binary\_vars)) %then %do;

proc sql noprint;

select max(length(name)) into: l1 from out.CE2\_binary\_vars;

select max(length(label)) into: l2 from out.CE2\_binary\_vars;

select max(length(new\_var)) into: l3 from out.CE2\_binary\_vars;

quit;

ods tagsets.excelxp options(sheet\_name="Binary Variables" absolute\_column\_width="&l1,8,8,&l2,&l3,8,8"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_binary\_vars noobs;

var name type length label new\_var;

var miss\_cnt ones\_cnt / style={tagattr='format:#,##0'};

run;

%end;

%if %sysfunc(exist(out.CE2\_nominal\_vars)) %then %do;

proc sql noprint;

select max(length(name)) into: l1 from out.CE2\_nominal\_vars;

select max(length(label)) into: l2 from out.CE2\_nominal\_vars;

select max(length(new\_vars)) into: l3 from out.CE2\_nominal\_vars;

quit;

ods tagsets.excelxp options(sheet\_name="Nominal Variables" absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_nominal\_vars noobs;

var name type length label new\_vars;

var miss\_cnt unique\_cnt max\_cnt / style={tagattr='format:#,##0'};

run;

%end;

%if %sysfunc(exist(out.CE2\_ordinal\_vars)) %then %do;

proc sql noprint;

select max(length(name)) into: l1 from out.CE2\_ordinal\_vars;

select max(length(label)) into: l2 from out.CE2\_ordinal\_vars;

select max(length(new\_var)) into: l3 from out.CE2\_ordinal\_vars;

quit;

ods tagsets.excelxp options(sheet\_name="Ordinal Variables"

%if %upcase(&stdmethodC) ^= NO %then %do;

%if %upcase(&binary\_dv)=Y %then %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,8,8,9,8,8,8,8" %end;

%else %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,8,8,9,8,8,8" %end;

%end;

%else %do;

%if %upcase(&binary\_dv)=Y %then %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,9,8,8,8,8" %end;

%else %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,9,8,8,8" %end;

%end;

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_ordinal\_vars noobs;

var name type length label new\_var relationship;

var miss\_cnt unique\_cnt max\_cnt / style={tagattr='format:#,##0'};

var var\_lb var\_ub var\_median / style={tagattr='format:0.00'};

%if %upcase(&stdmethodO) ^= NO %then %do; var Location Scale / style={tagattr='format:0.00'}; %end;

var miss\_impute / style={tagattr='format:0.0000'};

%if %upcase(&Binary\_dv)= Y %then %do;

var Concordant / style={tagattr='format:0.00'};

var CValue Chisq PValue / style={tagattr='format:0.0000'};

%end;

%else %do;

var RSquare FValue PValue / style={tagattr='format:0.0000'};

%end;

run;

%end;

%if %sysfunc(exist(out.CE2\_continuous\_vars)) %then %do;

proc sql noprint;

select max(length(name)) into: l1 from out.CE2\_continuous\_vars;

select max(length(label)) into: l2 from out.CE2\_continuous\_vars;

select max(length(new\_var)) into: l3 from out.CE2\_continuous\_vars;

quit;

ods tagsets.excelxp options(sheet\_name="Continuous Variables"

%if %upcase(&stdmethodC) ^= NO %then %do;

%if %upcase(&binary\_dv)=Y %then %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,8,8,9,8,8,8,8" %end;

%else %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,8,8,9,8,8,8" %end;

%end;

%else %do;

%if %upcase(&binary\_dv)=Y %then %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,9,8,8,8,8" %end;

%else %do;

absolute\_column\_width="&l1,8,8,&l2,&l3,8,8,8,8,8,8,8,9,8,8,8" %end;

%end;

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_continuous\_vars noobs;

var name type length label new\_var relationship;

var miss\_cnt / style={tagattr='format:#,##0'};

var P&p\_lo P&p\_hi var\_lb var\_ub var\_median / style={tagattr='format:0.00'};

%if %upcase(&stdmethodC) ^= NO %then %do; var Location Scale / style={tagattr='format:0.00'}; %end;

var miss\_impute / style={tagattr='format:0.0000'};

%if %upcase(&Binary\_dv)= Y %then %do;

var Concordant / style={tagattr='format:0.00'};

var CValue Chisq PValue / style={tagattr='format:0.0000'};

%end;

%else %do;

var RSquare FValue PValue / style={tagattr='format:0.0000'};

%end;

run;

%end;

ods Tagsets.ExcelxP close;

ods listing;

proc corr data=out.CE2\_Recoded noprint outp=corr;

var &dep\_var;

with &PREFIX.:;

run;

proc sql;

create table out.CE2\_corr as

select a.name as variable, a.label, b.&dep\_var as correlation

from vars a, corr b

where a.name = b.\_name\_

order by abs(b.&dep\_var) desc;

quit;

proc sql noprint;

select max(length(variable)) into: l1 from out.CE2\_corr;

select max(length(label)) into: l2 from out.CE2\_corr;

quit;

ods listing close;

ods Tagsets.ExcelxP body="&Path\_output.CE2\_Correlation\_report.xls" style=sasweb;

ods tagsets.excelxp options(sheet\_name="Correlations" absolute\_column\_width="&l1,&l2,8" Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_corr noobs;

var variable label;

var correlation / style={tagattr='format:0.0000'};

run;

ods Tagsets.ExcelxP close;

ods listing;

%if %upcase(&profiling) = Y %then %do;

proc sql noprint;

select max(length(variable)) into: l1 from out.CE2\_profile;

select max(length(label)) into: l2 from out.CE2\_profile;

select max(length(category)) into: l3 from out.CE2\_profile;

quit;

%\*\* Create Profile report in Excel;

%\* Order file;

proc sql;

create table ord as

select variable, max(abs(index))-min(abs(index)) as diff

from out.CE2\_profile

group by variable

order by diff desc;

quit;

data ord; set ord;

sord = \_n\_;

run;

data prof; set out.CE2\_profile; s2 = \_n\_; run;

proc sql;

create table prof2 as

select a.\*

from prof a, ord b

where a.variable = b.variable

order by b.sord, a.s2;

quit;

data out.CE2\_profile; set prof2 (drop= s2); run;

proc datasets nolist; delete prof prof2; quit; run;

ods listing close;

ods Tagsets.ExcelxP body="&Path\_output.CE2\_Profile\_report.xls" style=sasweb;

ods tagsets.excelxp options(sheet\_name="Binary Variables" absolute\_column\_width="&l1,&l2,&l3,8,8,9,8,6"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_profile noobs;

where type = "Binary";

var variable label category;

var count / style={tagattr='format:#,##0'};

var percent Average\_DV index star;

run;

ods tagsets.excelxp options(sheet\_name="Nominal Variables" absolute\_column\_width="&l1,&l2,&l3,8,8,9,8,6"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_profile noobs;

where type = "Nominal";

var variable label category;

var count / style={tagattr='format:#,##0'};

var percent Average\_DV index star;

run;

ods tagsets.excelxp options(sheet\_name="Ordinal Variables" absolute\_column\_width="&l1,&l2,&l3,8,8,9,8,6"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_profile noobs;

where type = "Ordinal";

var variable label category;

var count / style={tagattr='format:#,##0'};

var percent Average\_DV index star;

run;

ods tagsets.excelxp options(sheet\_name="Continuous Variables" absolute\_column\_width="&l1,&l2,&l3,8,8,9,8,6"

Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE2\_profile noobs;

where type = "Continuous";

var variable label category;

var count / style={tagattr='format:#,##0'};

var percent Average\_DV index star;

run;

ods Tagsets.ExcelxP close;

ods listing;

%end;

proc datasets nolist; delete workfile corr vars keep\_vars; quit; run;

%MEND;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* 3.Variable reduction and ranking \*\*\*;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\* Master variable reduction macro \*\*;

%macro CE\_Var\_Redu(insdn=out.CE2\_Recoded);

%\* Check if the user has specified a fast run;

%if %symexist(fast\_opt) %then %do;

%if %upcase(&fast\_opt) = Y %then %do;

%let sources = 1;

%let univ\_reg = N;

%let correlation = N;

%let principal = N;

%let cluster = N;

%if %upcase(&binary\_dv) = Y %then %do;

%let regression = N;

%let logistic = Y;

%end;

%else %do;

%let regression = Y;

%let logistic = N;

%end;

%let information = N;

%let ind\_correlation = N;

%let ind\_dv\_corr=N;

%end;

%end;

%\*\* Create working file;

proc sql noprint;

select count(\*) into : nobs from &insdn where mod\_val\_test^=3;

quit;

%put observation count is &nobs;

data workfile; set &insdn (where=(mod\_val\_test^=3));

%if &nobs > &samplesize %then %do; if uniform(131071)<=1.0\*&samplesize/&nobs; %end;

run;

%\*\* Get independent variables;

proc contents data=workfile (drop=&keep\_list)

out=vars (keep=name label) noprint;

run;

proc sql noprint;

select count(\*) into : varnum from vars;

quit;

%put Number of variables is &varnum;

%\*\* Reorder variables to avoid bias when working in groups;

data vars; set vars; rannum = ranuni(274923); run;

proc sort data = vars; by rannum; run;

%\*\* Using Univariate Regression method \*\*;

%if %upcase(&univ\_reg)=Y %then %do;

%do i=1 %to &varnum;

data \_null\_; set vars;

if \_n\_=&i;

call symputx('curvar',name);

run;

%Univ\_Reg(workfile,&curvar);

%if &i = 1 %then %do;

data outsdnuniv; length variable $32.; set univ\_tmp; run;

%end;

%else %do;

data outsdnuniv; set outsdnuniv univ\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete univ\_tmp; quit; run;

%\* Finalize file;

data outsdnuniv; set outsdnuniv;

if PValue <= &maxpuni then univ\_flag = 1; else univ\_flag = 0;

run;

proc sort data = outsdnuniv; by variable; run;

%end;

%\*\* Using correlation method \*\*;

%if %upcase(&correlation)=Y %then %do;

proc corr data=workfile noprint outp=outsdncorr;

var &dep\_var;

with &PREFIX.:;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

%\* Finalize file;

data outsdncorr (drop=\_type\_);

length variable $32.;

set outsdncorr (where=(\_type\_='CORR') rename=(\_name\_=variable &dep\_var=corr));

if abs(corr) >= &corrcut then corr\_flag=1; else corr\_flag=0;

run;

proc sort data= outsdncorr; by variable; run;

%end;

%\*\* Using factor/principal analysis method \*\*;

%if %upcase(&principal)=Y %then %do;

%if &varnum <= &nprin %then %do;

%let nprin = %eval(&varnum-1);

%end;

%\*\* Split variables into manageable groups;

%let group=%sysfunc(max(%eval(&varnum/(&nprin\*5)),1));

data tmp; set vars; rank=ceil(\_n\_/(&varnum/&group)); run;

%\*\* Process variables by group;

%do i = 1 %to &group;

proc sql noprint; select name into : regvlist separated by ' '

from tmp where rank = &i;

quit;

%single\_group\_prin(&insdn,prin\_tmp,&regvlist);

%if &i = 1 %then %do;

data outsdnprin; length variable $32.; set prin\_tmp; run;

%end;

%else %do;

data outsdnprin; set outsdnprin prin\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete prin\_tmp tmp; quit; run;

%\* Finalize file;

%if &group > 1 %then %do;

proc sql noprint; select variable into : regvlist separated by ' '

from outsdnprin where Factor >= &minprin;

quit;

%single\_group\_prin(&insdn,outsdnprin,&regvlist);

%end;

data outsdnprin; set outsdnprin;

if Factor >= &minprin then prin\_flag=1; else prin\_flag=0;

run;

proc sort data = outsdnprin; by variable; run;

%end;

%\*\* Using Clustering method \*\*;

%if %upcase(&cluster)=Y %then %do;

%if &varnum <= &maxc %then %do;

%let maxc = %eval(&varnum-1);

%end;

%\*\* Split variables into manageable groups;

%let group=%sysfunc(max(%eval(&varnum/(&maxc\*5)),1));

data tmp; set vars; rank=ceil(\_n\_/(&varnum/&group)); run;

%\*\* Process variables by group;

%do i = 1 %to &group;

proc sql noprint; select name into : regvlist separated by ' '

from tmp where rank = &i;

quit;

%single\_group\_clus(&insdn,clus\_tmp,&regvlist);

%if &i = 1 %then %do;

data outsdnclus; length variable $32.; set clus\_tmp; run;

%end;

%else %do;

data outsdnclus; set outsdnclus clus\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete clus\_tmp tmp; quit; run;

%\* Finalize file;

%if &group > 1 %then %do;

proc sql noprint; select variable into : regvlist separated by ' '

from outsdnclus where RSquareRatio <= &maxratio;

quit;

%single\_group\_clus(&insdn,outsdnclus,&regvlist);

%end;

data outsdnclus; set outsdnclus;

if RSquareRatio <= &maxratio then clus\_flag=1; else clus\_flag=0;

run;

proc sort data = outsdnclus; by variable; run;

%end;

%\*\* Using regression method \*\*;

%if %upcase(&regression)=Y %then %do;

%\*\* Split variables into manageable groups;

%let group=%sysfunc(ceil(&varnum/100));

data tmp; set vars; rank=ceil(\_n\_/(&varnum/&group)); run;

%\*\* Process variables by group;

%do i = 1 %to &group;

proc sql noprint; select name into : regvlist separated by ' '

from tmp where rank = &i;

quit;

%single\_group\_reg(&insdn,reg\_tmp,&regvlist);

%if &i = 1 %then %do;

data outsdnreg; length variable $32.; set reg\_tmp; run;

%end;

%else %do;

data outsdnreg; set outsdnreg reg\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete reg\_tmp tmp; quit; run;

%\* Finalize file;

%if &group > 1 %then %do;

proc sql noprint; select variable into : regvlist separated by ' '

from outsdnreg;

quit;

%single\_group\_reg(&insdn,outsdnreg,&regvlist);

%end;

proc sort data = outsdnreg; by variable; run;

%end;

%\*\* Using logistic method \*\*;

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do;

%\*\* Split variables into manageable groups;

%let group=%sysfunc(ceil(&varnum/100));

data tmp; set vars; rank=ceil(\_n\_/(&varnum/&group)); run;

%\*\* Process variables by group;

%do i = 1 %to &group;

proc sql noprint; select name into : regvlist separated by ' '

from tmp where rank = &i;

quit;

%single\_group\_log(&insdn,log\_tmp,&regvlist);

%if &i = 1 %then %do;

data outsdnlog; length variable $32.; set log\_tmp; run;

%end;

%else %do;

data outsdnlog; set outsdnlog log\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete log\_tmp tmp; quit; run;

%\* Finalize file;

%if &group > 1 %then %do;

proc sql noprint; select variable into : regvlist separated by ' '

from outsdnlog;

quit;

%single\_group\_log(&insdn,outsdnlog,&regvlist);

%end;

proc sort data = outsdnlog; by variable; run;

%end;

%\*\* Using information value method \*\*;

%if %upcase(&information)=Y %then %do;

%do i=1 %to &varnum;

data \_null\_; set vars;

if \_n\_=&i;

call symputx('curvar',name);

run;

%Info\_Val\_Var(workfile,&curvar,&varnum);

%if &i = 1 %then %do;

data outsdninfv; length variable $32.; set infv\_tmp; run;

%end;

%else %do;

data outsdninfv; set outsdninfv infv\_tmp; run;

%end;

%end;

proc datasets library=work nolist; delete infv\_tmp; quit; run;

%\* Finalize file;

data outsdninfv; set outsdninfv;

if infv >= &infvcut then infv\_flag = 1; else infv\_flag = 0;

run;

proc sort data = outsdninfv; by variable; run;

%end;

%\* Get basic metrics for variables;

proc means data=&insdn noprint ;

var &prefix: ;

output out=cnt n= ;

output out=min min= ;

output out=max max= ;

output out=mean mean= ;

run;

data stats;

set cnt (in=a) min (in=b) max (in=c) mean (in=d);

length var $8;

if a then var = "Count";

else if b then var = "Minimum";

else if c then var = "Maximum";

else if d then var = "Mean";

drop \_freq\_ \_type\_;

run;

proc transpose data=stats out=stats name=Variable label=Label; id var; run;

proc sort data=stats; by variable; run;

data stats; length variable $32.; set stats; run;

%\*combine results from all methods together;

data out.CE3\_Var\_Redu;

merge stats

%if %upcase(&univ\_reg)=Y %then %do; outsdnuniv(in=tuniv) %end;

%if %upcase(&correlation)=Y %then %do; outsdncorr(in=tcorr) %end;

%if %upcase(&principal)=Y %then %do; outsdnprin(in=tprin) %end;

%if %upcase(&cluster)=Y %then %do; outsdnclus(in=tclus) %end;

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; outsdnlog(in=tlog) %end;

%if %upcase(&regression)=Y %then %do; outsdnreg(in=treg) %end;

%if %upcase(&information)=Y %then %do; outsdninfv(in=tinfv) %end;

;

by variable;

%if %upcase(&univ\_reg)=Y %then %do; if tuniv and univ\_flag=1 then univsource=1; else univsource=0; drop univ\_flag; %end;

%if %upcase(&correlation)=Y %then %do; if tcorr and corr\_flag=1 then corrsource=1; else corrsource=0; drop corr\_flag; %end;

%if %upcase(&principal)=Y %then %do; if tprin and prin\_flag=1 then prinsource=1; else prinsource=0; drop prin\_flag; %end;

%if %upcase(&cluster)=Y %then %do; if tclus and clus\_flag=1then clussource=1; else clussource=0; drop clus\_flag; %end;

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; if tlog then logsource=1; else logsource=0; %end;

%if %upcase(&regression)=Y %then %do; if treg then regsource=1; else regsource=0; %end;

%if %upcase(&information)=Y %then %do; if tinfv and infv\_flag=1 then infvsource=1; else infvsource=0; drop infv\_flag; %end;

num\_sources=sum(of

%if %upcase(&univ\_reg)=Y %then %do; univsource %end;

%if %upcase(&correlation)=Y %then %do; corrsource %end;

%if %upcase(&principal)=Y %then %do; prinsource %end;

%if %upcase(&cluster)=Y %then %do; clussource %end;

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; logsource %end;

%if %upcase(&regression)=Y %then %do; regsource %end;

%if %upcase(&information)=Y %then %do; infvsource %end;

);

run;

proc sort data=out.CE3\_Var\_Redu; by descending num\_sources %if %upcase(&information)=Y %then %do; descending infv %end; ;run;

%\*\* Exclude highly correlated variables \*\*;

%if %upcase(&ind\_correlation)=Y %then %do;

%Maxx\_Corr(workfile);

%end;

%\*\* Exclude variables highly correlated to dependent variable \*\*;

%if %upcase(&ind\_dv\_corr)=Y %then %do;

proc sql;

create table tmp as

select a.\*, b.correlation as dv\_corr,

case when b.correlation > &max\_dv\_corr then 'Y' end as drop\_dv\_corr

from out.CE3\_Var\_Redu a

left join out.CE2\_corr b on a.variable = b.variable;

quit;

data out.CE3\_Var\_Redu; set tmp; run;

proc datasets library=work nolist; delete tmp; quit; run;

%end;

%\*\* Create report in Excel;

ods listing close;

ods Tagsets.ExcelxP body="&Path\_output.CE3\_Var\_Redu Results.xls" style=sasweb;

ods tagsets.excelxp options(sheet\_name="Variables");

proc print data=out.CE3\_Var\_Redu (drop=

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; LogStep %end;

%if %upcase(&regression)=Y %then %do; RegStep %end;

%if %upcase(&cluster)=Y %then %do; Cluster %end;

) noobs;

run;

ods Tagsets.ExcelxP close;

ods listing;

%\*Output list of top variables;

data selected; set out.CE3\_Var\_Redu;

%if %upcase(&ind\_correlation)=Y %then %do; if missing(drop\_corr); %end;

%if %upcase(&ind\_dv\_corr)=Y %then %do; if missing(drop\_dv\_corr); %end;

if num\_sources>=&sources

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; or logsource=1 %end;

%if %upcase(&regression)=Y %then %do; or regsource=1 %end; ;

run;

%let rck = 0;

data \_null\_; set selected end=eof;

m = mod(\_N\_,7);

length v $256.;

retain v ;

if m = 1 then v = variable;

else v = strip(v) || " " || strip(variable);

FILE "&Path\_output.CE3\_Varlist\_redu.txt" lrecl=256;

if \_N\_=1 then PUT '%let varlist\_redu =';

if m = 0 or eof then PUT v '0d'x;

if eof then PUT ';';

if eof then call symputx("rck" ,\_N\_);

run;

%put number of selected variables = &rck;

proc datasets nolist; delete stats cnt min max mean selected workfile vars

%if %upcase(&univ\_reg)=Y %then %do; outsdnuniv %end;

%if %upcase(&correlation)=Y %then %do; outsdncorr %end;

%if %upcase(&principal)=Y %then %do; outsdnprin %end;

%if %upcase(&cluster)=Y %then %do; outsdnclus %end;

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; outsdnlog %end;

%if %upcase(&regression)=Y %then %do; outsdnreg %end;

%if %upcase(&information)=Y %then %do; outsdninfv %end;

; quit; run;

%mend;

\*\* Univ\_Reg: using univariate regression to do variable reduction \*\*;

%macro Univ\_Reg(insdn,var);

data univ\_tmp; set \_NULL\_; run;

%PUT \*\*\*Univ\_Reg STEP, CURRENT VARIABLE: &var\*\*\*;

%if %upcase(&Binary\_dv) = Y %then %do;

%\* Run univariate logistic regression;

ods listing close;

ods output parameterestimates=parm association=fitstat1;

proc logistic data=&insdn desc namelen=32;

model &dep\_var=&var;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

ods listing;

%if %sysfunc(exist(fitstat1)) %then %do;

data fitstat1 (keep=CC\_RSQ); set fitstat1(keep=cvalue1 obs=1);

CC\_RSQ = input(cvalue1,best8.);

run;

%end;

%else %do;

data fitstat1; CC\_RSQ = .; output; run;

%end;

data parm (keep=ProbChiSq sign rename=(ProbChiSq=PValue));

length sign $6. ;

set parm (firstobs=2 obs=2 keep=variable Estimate ProbChiSq);

if Estimate>0 then sign='(+)'; else sign='(-)';

run;

%end;

%else %do;

%\* Run univariate simple regression;

ods listing close;

ods output SelParmEst=parm SelectionSummary=fitstat1;

proc reg data=&insdn;

model &dep\_var=&var /selection=forward MAXSTEP=1 slentry=0.999;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

ods listing;

%if %sysfunc(exist(fitstat1)) %then %do;

data fitstat1 (rename=(ModelRsquare=CC\_RSQ)); set fitstat1(keep=ModelRsquare obs=1); run;

%end;

%else %do;

data fitstat1; CC\_RSQ = .; output; run;

%end;

data parm (keep=ProbF sign rename=(ProbF=PValue));

length sign $6. ;

set parm (firstobs=2 obs=2 keep=variable Estimate ProbF);

if Estimate>0 then sign='(+)'; else sign='(-)';

run;

%end;

%\* Combine;

data univ\_tmp;

length variable $32.;

merge parm fitstat1;

variable="&var";

run;

%\* Clean up;

proc datasets library=work nolist; delete fitstat1 parm; quit; run;

%mend;

\*\* Single\_Group\_Prin: using principal components to do variable reduction \*\*;

%macro Single\_Group\_Prin(insdn,outsdn,vlist);

%local j;

data &outsdn; set \_NULL\_; run;

%\* Run factor / principal components analysis;

proc factor data=&insdn out=tmpdataa2 method=prin priors=one nfact=&nprin noprint;

var &vlist;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

%\* Get correlations;

proc corr data=tmpdataa2 out=tmpdataa3 noprint;

var Factor: ;

with &vlist;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

%\* Rearrange dataset;

data tmpdataa4 (drop=\_type\_ \_name\_); set tmpdataa3 (where=(\_type\_='CORR'));

length variable $32;

variable=\_name\_;

%do j = 1 %to &nprin;

factor=abs(factor&j);

output;

%end;

run;

%\* Get best correlation for each variable;

proc sort data=tmpdataa4 (keep=variable factor); by variable descending factor; run;

data &outsdn; set tmpdataa4;

by variable;

if first.variable;

run;

proc datasets library=work nolist; delete tmpdataa: ; quit; run;

%mend;

\*\* Single\_Group\_Clus: using clustering to do variable reduction \*\*;

%macro Single\_Group\_Clus(insdn,outsdn,vlist);

data &outsdn; set \_NULL\_; run;

%\* Run cluster procedure;

ods listing close;

ods output rsquare=&outsdn;

proc varclus data=&insdn (keep=&vlist &weight) minc=&maxc maxc=&maxc short;

var &vlist;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

ods listing;

%\* Clean up file;

data &outsdn (keep=variable cluster RSquareRatio);

length variable $32.;

set &outsdn;

retain clustertemp;

if cluster ne ' ' then clustertemp=cluster;

else cluster=clustertemp;

run;

%mend;

\*\* Single\_Group\_Reg: using linear regression to do variable reduction \*\*;

%macro Single\_Group\_Reg(insdn,outsdn,vlist);

data &outsdn; set \_NULL\_; run;

ods listing close;

ods output SelectionSummary=&outsdn;

proc reg data=&insdn;

model &dep\_var = &vlist /selection=forward slentry=&alphareg;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

ods listing;

data &outsdn (keep=variable RegStep RegPValue);

length variable $32.;

set &outsdn (rename=(VarEntered = variable step=RegStep ProbF = RegPValue));

run;

%mend;

\*\* Single\_Group\_Log: using logistic regression to do variable reduction \*\*;

%macro Single\_Group\_Log(insdn,outsdn,vlist);

data &outsdn; set \_NULL\_; run;

ods listing close;

ods output ModelBuildingSummary=&outsdn;

proc logistic data=&insdn desc namelen=32;

model &dep\_var = &vlist /selection=forward slentry=&alphalog;

%if %upcase(&redu\_weight) = Y %then %do; weight &weight; %end;

run;

ods listing;

data &outsdn (keep=variable LogStep LogPValue);

length variable $32.;

set &outsdn (rename=(EffectEntered = variable step=LogStep ProbChiSq = LogPValue));

run;

%mend;

\*\* Info\_Val\_Var: using information value to do variable reduction \*\*;

%macro Info\_Val\_Var(insdn,var,set\_size);

data infv\_tmp; set \_NULL\_; run;

%\* Check number of unique values;

proc sql noprint;

select count(distinct(&var)) into : unq from &insdn;

quit;

%if %eval(&unq>&decile) %then %do;

%if %upcase(&redu\_weight) = Y %then %do;

%\* Create bins if there is weighting;

proc sql noprint;

select sum(&weight) into : cumwgt from &insdn;

quit;

proc sort data=&insdn (keep=&dep\_var &var &weight) out=tmp; by &var; run;

data tmp (drop=rank); set tmp;

retain rank;

rank + &weight;

bin = (floor(rank\*&decile/(&cumwgt+1)));

run;

%\* Summarize data to bins;

proc summary data=tmp;

weight &weight;

var &var &dep\_var;

class bin;

output out=tmp2 (drop=\_freq\_) sumwgt=cnt mean(&dep\_var)=mean\_dv max(&dep\_var)=max\_dv

mean(&var)=mean\_var min(&var)=min\_var max(&var)=max\_var /noinherit;

run;

%end;

%else %do;

%\* Create bins if there is no weighting;

proc rank data=&insdn (keep=&dep\_var &var) out=tmp groups=&decile;

var &var;

ranks bin;

run;

%\* Summarize data to bins;

proc summary data=tmp;

var &var &dep\_var;

class bin;

output out=tmp2 (rename=\_freq\_=cnt) mean(&dep\_var)=mean\_dv max(&dep\_var)=max\_dv

mean(&var)=mean\_var min(&var)=min\_var max(&var)=max\_var /noinherit;

run;

%end;

%end;

%else %do;

%\* Summarize to &var value when less than &decile;

%if %upcase(&redu\_weight) = Y %then %do;

proc summary data=&insdn;

weight &weight;

var &dep\_var;

class &var;

output out=tmp2 (drop=\_freq\_ rename=&var=bin) sumwgt=cnt mean(&dep\_var)=mean\_dv max(&dep\_var)=max\_dv /noinherit;

run;

%end;

%else %do;

proc summary data=&insdn;

var &dep\_var;

class &var;

output out=tmp2 (rename=(\_freq\_=cnt &var=bin)) mean(&dep\_var)=mean\_dv max(&dep\_var)=max\_dv /noinherit;

run;

%end;

data tmp2; set tmp2;

mean\_var = bin;

min\_var = bin;

max\_var = bin;

run;

%end;

%\* Create global metrics;

proc sql noprint;

select mean\_dv into : norm from tmp2 where \_type\_ = 0;

select cnt into : ntotal from tmp2 where \_type\_ = 0;

select max(max\_dv) into : maxdv from tmp2 where \_type\_ = 0;

select sum(cnt\*mean\_dv) into: totalresp from tmp2 where \_type\_ = 1;

quit;

%\* Calculate information value, ks and gini statistics;

data tmp3; set tmp2 (where=(\_type\_=1));

lift\_index = (mean\_dv/&norm)\*100;

retain cumresp cumtotal;

if \_n\_ = 1 then do;

cumresp = 0;

cumtotal = 0;

end;

cumresp = cumresp + (mean\_dv\*cnt);

cumpct\_resp = cumresp/&totalresp;

cumtotal = cumtotal + cnt;

cumpct\_freq = cumtotal/&ntotal;

cumavg = cumresp/cumtotal;

cumindex = (cumavg/&norm)\*100;

badrate = mean\_dv/&maxdv;

goodrate = 1 - badrate;

badcnt = badrate \* cnt;

goodcnt = goodrate \* cnt;

badratio = (cnt \* mean\_dv)/&totalresp;

goodratio = (cnt - badcnt)/(&ntotal-&totalresp/&maxdv);

drop cumresp cumtotal;

run;

data infv\_tmp; set tmp3 end=eof;

length variable $32.;

variable = "&var";

retain infv gini ks cumbad cumgood 0;

prev\_dv = lag(cumpct\_resp);

prev\_pct = lag(cumpct\_freq);

if \_n\_ ne 1 then

gini = gini+2\*((cumpct\_freq+prev\_pct)/2-(cumpct\_resp+prev\_dv)/2)\*(cumpct\_freq-prev\_pct);

if badratio+goodratio = 0 then contribution = 0;

else if badratio\*goodratio=0 then contribution=cnt\*1.0/&set\_size\*max(badratio,goodratio);

else contribution = (badratio-goodratio)\*log(Max(badratio/goodratio,0.00001));

infv = infv+(contribution);

cumbad = cumbad + badratio;

cumgood = cumgood + goodratio;

ks = max(ks,abs(cumbad-cumgood));

if eof then output;

keep variable infv gini ks;

run;

proc datasets library=work nolist; delete tmp: ; quit; run;

%mend;

\*\* Maxx\_Corr: eliminate variables that are highly correlated \*\*;

%macro Maxx\_Corr(insdn);

%\* Add field to variable dataset;

data out.CE3\_Var\_Redu; set out.CE3\_Var\_Redu;

length drop\_corr $1.;

run;

%\* Get variables to evaluate;

data tmp; set out.CE3\_Var\_Redu;

if num\_sources>=&sources

%if %upcase(&logistic)=Y and %upcase(&Binary\_dv)=Y %then %do; or logsource=1 %end;

%if %upcase(&regression)=Y %then %do; or regsource=1 %end; ;

run;

proc sql noprint;

select variable into : varlist\_redu separated by ' ' from tmp;

quit;

%\* Correlation table;

proc corr data=&insdn out=tmp noprint;

var &dep\_var &varlist\_redu;

run;

proc sql;

create table corrmatrix (drop=\_type\_) as

select \* from tmp where \_type\_ = 'CORR' and \_name\_ ^= "&dep\_var" order by abs(&dep\_var) desc;

quit;

%\* Cycle through to eliminate variable wiht correlations that are too high;

%let breakdelcorr=0;

%do %while(&breakdelcorr=0);

data corrmatrix (drop=i); set corrmatrix;

varnum = \_n\_;

cnt = 0;

array vars{\*} &prefix.: ;

do i = 1 to dim(vars);

if &maxcorr < abs(vars{i}) then cnt = cnt + 1;

end;

run;

proc sql noprint;

select count(\*) into: vcnt from corrmatrix where cnt>1;

quit;

%if &vcnt = 0 %then %let breakdelcorr=1; %\* No more variables to drop;

%else %do;

%let ndropvar=0;

data \_null\_; set corrmatrix (where=(cnt>1) obs=1) end=eof;

call symputx("keepvar",\_name\_);

if eof then call symputx("ndropvar",1);

run;

data \_null\_; set corrmatrix (where=(abs(&keepvar) > &maxcorr and \_name\_ ^= "&keepvar")) end=lastob;

call symputx('dropvar'||left(\_n\_),\_name\_);

if lastob then call symputx('ndrop',\_n\_);

run;

%if &ndropvar=0 %then %let breakdelcorr=1; %\* No more variables to drop;

%else %do; %\* More variables to drop;

data corrmatrix; set corrmatrix (drop= %do i = 1 %to &ndrop; &&dropvar&i %end; );

%do i = 1 %to &ndrop;

if upcase(\_name\_) = upcase("&&dropvar&i") then delete;

%end;

run;

data out.CE3\_Var\_Redu; set out.CE3\_Var\_Redu;

%do i = 1 %to &ndrop;

if upcase(variable) = upcase("&&dropvar&i") then do;

drop\_corr = 'Y';

end;

%end;

run;

%end;

%end;

%end;

proc datasets nolist; delete tmp corrmatrix; quit; run;

%mend;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* 4.Model selection and tuning \*\*\*;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\* Master macro for binary dependent variable \*\*;

%macro Model\_Val\_Logistic(insdnT, insdnV, regvlist, sle=0.05, sls=.05,

metric\_sdnout=out.CE4\_metric\_sdnout);

%\* Set up independent variable lists;

%if &excludelist = %then %let nexcludes = 0;

%else %let nexcludes = %sysfunc(countw(&excludelist)); /\* get number of variables in exclude list \*/

%if &nexcludes ^= 0 %then %do;

%let regvlist=%StringMinus(&regvlist, &excludelist); /\* Remove excluded variables from variable list \*/

%end;

%if &startlist = %then %let nstarts = 0;

%else %let nstarts = %sysfunc(countw(&startlist)); /\* get number of variables in start list \*/

%if &nstarts ^= 0 %then %do;

%if nexcludes ^= 0 %then %do; /\* Make sure there are no excluded variables in the start list\*/

%let startlist=%StringMinus(&startlist, &excludelist);

%end;

%let nstarts = %sysfunc(countw(&startlist)); /\* re-get number of variables in start list \*/

%if &nstarts ^= 0 %then %do;

%let regvlist=&startlist %StringMinus(&regvlist, &startlist);

%end;

%end;

%if &includelist = %then %let nincludes = 0;

%else %let nincludes = %sysfunc(countw(&includelist)); /\* get number of variables in include list \*/

%if &nincludes ^= 0 %then %do;

%if nexcludes ^= 0 %then %do; /\* Make sure there are no excluded variables in the include list\*/

%let includelist=%StringMinus(&includelist, &excludelist);

%end;

%let nincludes = %sysfunc(countw(&includelist)); /\* re-get number of variables in include list \*/

%if &nincludes ^= 0 %then %do;

%let regvlist=&includelist %StringMinus(&regvlist, &includelist);

%end;

%end;

ods listing close;

%\*\* Build forward model \*\*;

ods output ParameterEstimates=Parm1;

proc logistic data=&insdnT desc namelen=32;

model &dep\_var = &regvlist /selection=forward lackfit rsq details slentry=&sle

%if &nincludes ne 0 %then %do; include=&nincludes %end;

%if &nstarts ne 0 %then %do; start=&nstarts %end;

;

run;

%\*\* Build backward model \*\*;

ods output ParameterEstimates=Parm2;

proc logistic data=&insdnT desc namelen=32;

model &dep\_var = &regvlist /selection=backward lackfit rsq details slstay=&sls

%if &nincludes ne 0 %then %do; include=&nincludes %end;

%if &nstarts ne 0 %then %do; start=&nstarts %end;

;

run;

ods listing;

%\*\* Extract variables \*\*;

proc sql noprint;

select max(step) into: max1 from Parm1;

select max(step) into: max2 from Parm2;

quit;

data parm1; set parm1 (where=(step=&max1)); run;

data parm2; set parm2 (where=(step=&max2)); run;

proc sql noprint;

create table varlisttmp as

select variable

from parm1

union

select variable

from parm2;

select variable into: varlisttmp separated by ' ' from varlisttmp where variable ^= 'Intercept';

select count(\*) into: nvars from varlisttmp where variable ^= 'Intercept';

quit;

%put &varlisttmp;

%ctl\_Stats\_Log(&insdnT, &varlisttmp, &nvars);

data &metric\_sdnout;

length file $4.;

set stats;

file = 'Dev';

run;

%ctl\_Stats\_Log(&insdnV, &varlisttmp, &nvars);

data &metric\_sdnout;

set &metric\_sdnout stats (in=a);

if a then file = 'Val';

run;

%\*\* Clean up;

proc datasets library=work nolist;

delete varlisttmp globalfit full vars tmp1 tmp2 parms vif stats;

run;

quit;

%mend;

\*\* Master macro for continuous dependent variable \*\*;

%macro Model\_Val\_Reg(insdnT, insdnV, regvlist, sle=0.05, sls=.05,

metric\_sdnout=out.CE4\_metric\_sdnout);

%\* Set up independent variable lists;

%if &excludelist = %then %let nexcludes = 0;

%else %let nexcludes = %sysfunc(countw(&excludelist)); /\* get number of variables in exclude list \*/

%if &nexcludes ^= 0 %then %do;

%let regvlist=%StringMinus(&regvlist, &excludelist); /\* Remove excluded variables from variable list \*/

%end;

%if &startlist = %then %let nstarts = 0;

%else %let nstarts = %sysfunc(countw(&startlist)); /\* get number of variables in start list \*/

%if &nstarts ^= 0 %then %do;

%if nexcludes ^= 0 %then %do; /\* Make sure there are no excluded variables in the start list\*/

%let startlist=%StringMinus(&startlist, &excludelist);

%end;

%let nstarts = %sysfunc(countw(&startlist)); /\* re-get number of variables in start list \*/

%if &nstarts ^= 0 %then %do;

%let regvlist=&startlist %StringMinus(&regvlist, &startlist);

%end;

%end;

%if &includelist = %then %let nincludes = 0;

%else %let nincludes = %sysfunc(countw(&includelist)); /\* get number of variables in include list \*/

%if &nincludes ^= 0 %then %do;

%if nexcludes ^= 0 %then %do; /\* Make sure there are no excluded variables in the include list\*/

%let includelist=%StringMinus(&includelist, &excludelist);

%end;

%let nincludes = %sysfunc(countw(&includelist)); /\* re-get number of variables in include list \*/

%if &nincludes ^= 0 %then %do;

%let regvlist=&includelist %StringMinus(&regvlist, &includelist);

%end;

%end;

ods listing close;

ods graphics on;

%\*\* Build forward model \*\*;

ods output SelParmEst=Parm1;

proc reg data=&insdnT plots(maxpoints=none);

model &dep\_var = &regvlist /selection=forward details slentry=&sle

%if &nincludes ne 0 %then %do; include=&nincludes %end;

%if &nstarts ne 0 %then %do; start=&nstarts %end;

;

run;

ods listing;

ods graphics off;

quit;

ods listing close;

ods graphics on;

%\*\* Build backward model \*\*;

ods output SelParmEst=Parm2;

proc reg data=&insdnT plots(maxpoints=none);

model &dep\_var = &regvlist /selection=backward details slstay=&sls

%if &nincludes ne 0 %then %do; include=&nincludes %end;

%if &nstarts ne 0 %then %do; start=&nstarts %end;

;

run;

ods listing;

ods graphics off;

quit;

%\*\* Extract variables \*\*;

proc sql noprint;

select max(step) into: max1 from Parm1;

select max(step) into: max2 from Parm2;

quit;

data parm1; set parm1 (where=(step=&max1)); run;

data parm2; set parm2 (where=(step=&max2)); run;

proc sql noprint;

create table varlisttmp as

select variable

from parm1

union

select variable

from parm2;

select variable into: varlisttmp separated by ' ' from varlisttmp where variable ^= 'Intercept';

select count(\*) into: nvars from varlisttmp where variable ^= 'Intercept';

quit;

%put &varlisttmp;

%\*\* Get dev model statistics \*\*;

%ctl\_Stats\_Reg(&insdnT, &varlisttmp, &nvars );

data &metric\_sdnout;

length file $4.;

set stats;

file = 'Dev';

run;

%\*\* Get val model statistics \*\*;

%ctl\_Stats\_Reg(&insdnV, &varlisttmp, &nvars);

data &metric\_sdnout;

set &metric\_sdnout stats (in=a);

if a then file = 'Val';

run;

%\* Clean up;

proc datasets library=work nolist;

delete varlisttmp globalfit full vars tmp1 tmp2 vif stats;

run;

quit;

%mend;

\*\* StringMinus substracts one string from another \*\*;

%macro StringMinus(string1, string2);

%local count word StringOut;

%let count=1;

%let word=%scan(&string1,&count,%str( ));

%do %while(&word ne);

%let count=%eval(&count+1);

%if %sysfunc(indexw(%upcase(&string2), %upcase(&word)))=0 %then %do;

%let stringout=&stringout &word;

%end;

%let word=%scan(&string1,&count,%str( ));

%end;

&StringOut

%mend;

\*\* Remember\_nobs\_ngoods\_nbads gets record counts \*\*;

%macro Remember\_nobs\_ngoods\_nbads(insdn,bads=1,goods=0);

%global vnobs;

%global vngoods;

%global vnbads;

%if &weight= %then %do;

proc sql noprint;

select count(\*) into: vnobs from &insdn;

select count(\*) into: vnbads from &insdn where &dep\_var in (&bads);

select count(\*) into: vngoods from &insdn where &dep\_var in (&goods);

quit;

%end;

%else %do;

proc sql noprint;

select sum(&weight) into: vnobs from &insdn;

select sum(&weight) into: vnbads from &insdn where &dep\_var in (&bads);

select sum(&weight) into: vngoods from &insdn where &dep\_var in (&goods);

quit;

%end;

%mend;

\*\* Control program for binary stats \*\*;

%macro ctl\_Stats\_Log(insdn, varlisttmp, nvars);

%\*\* Get counts for validation file \*\*;

%Remember\_nobs\_ngoods\_nbads(&insdn);

%\*\* Get full model statistics \*\*;

%Stats\_Log(&insdn, &varlisttmp, Full Model, int=N, vif=Y);

data full; set globalfit; run;

%\*\* Get model statistics without intercept \*\*;

%Stats\_Log(&insdn, &varlisttmp, Intercept, int=Y, vif=N);

data vars; set globalfit; run;

%\*\* Get model statistics without specific variables \*\*;

%do I = 1 %to &nvars;

%let var = %scan(&varlisttmp,&I);

%let tmplist=%StringMinus(&varlisttmp, &var); /\* Remove single variable \*/

%Stats\_Log(&insdn, &tmplist, &var, int=N, vif=N);

data vars; set vars globalfit; run;

%end;

%\*\* Create macro variables for full models metrics \*\*;

data \_null\_; set full;

call symputx('AIC',AIC);

call symputx('SC',SC);

call symputx('LogL2',LogL2);

call symputx('Rsquare',Rsquare);

call symputx('SomersD',SomersD);

call symputx('Gamma',Gamma);

call symputx('TauA',TauA);

call symputx('c',c);

call symputx('Concord',Concord);

call symputx('Discon',Discon);

call symputx('LackFit',LackFit);

call symputx('Lift',Lift\_index);

call symputx('infv',infv);

call symputx('ks',ks);

run;

proc sql noprint;

select sum(abs(standardizedest)) into: cum from parms where variable ^= 'Intercept';

quit;

%\*\* Subtract full model metrics from sub-model metrics;

%\*\* This shows the impact of each variable;

data tmp1; set vars;

AIC = AIC - &AIC;

SC = SC - &SC;

LogL2 = LogL2 - &LogL2;

Rsquare = Rsquare - &Rsquare;

SomersD = SomersD - &SomersD;

Gamma = Gamma - &Gamma;

TauA = TauA - &TauA;

c = c - &c;

Concord = Concord - &Concord;

Discon = Discon - &Discon;

LackFit = LackFit - &LackFit;

lift\_Index = Lift\_Index - &Lift;

infv = infv - &infv;

ks = ks - &ks;

run;

%\*\* Combine datasets \*\*;

data tmp2; set full (in=a) tmp1;

if a then sord = 1;

else if variable = 'Intercept' then sord = 2;

else sord = 3;

run;

proc sql;

create table stats as

select a.Variable,

b.Label,

b.Estimate,

b.StandardizedEst as StdEst,

b.StdErr,

b.WaldChiSq,

b.ProbChiSq,

c.VarianceInflation as VIF,

abs(b.StandardizedEst)/&cum as RelImp,

a.AIC,

a.SC,

a.LogL2,

a.Rsquare,

a.SomersD,

a.Gamma,

a.TauA,

a.c,

a.Concord,

a.Discon,

a.LackFit,

a.Lift\_Index,

a.infv,

a.ks,

a.sord

from tmp2 a

left join parms b on a.variable = b.variable

left join vif c on a.variable = c.variable

order by a.sord, RelImp desc;

quit;

%\*\* Clean up;

proc datasets library=work nolist;

delete globalfit full vars tmp1 tmp2 parms vif;

run;

quit;

%mend;

\*\* Stats\_Log gets statistics for binary models \*\*;

%macro Stats\_Log(insdn, varlist, var, int=N, vif=N ,bads=1,goods=0);

%\* Extract standard model statistics;

ods listing close;

ods output FitStatistics= FitStatistics;

ods output RSquare= RSquare;

ods output Association= Association;

ods output LackFitChiSq= LackFitChiSq;

%if &vif=Y %then %do;

ods output ParameterEstimates=Parms;

%end;

proc logistic data=&insdn desc namelen=32;

%if &weight ne %then %do; weight &weight; %end;

model &dep\_var = &varlist / stb rsq lackfit parmlabel

%if &int=Y %then %do; noint %end;

;

output out=scoretmp p=pred;

run;

ods listing;

data globalfit (drop=Criterion label1 label2);

length variable $32.;

variable = "&var";

merge

%if &int=N %then %do;

FitStatistics (where=(Criterion='AIC') rename=(InterceptAndCovariates=AIC) keep=Criterion InterceptAndCovariates)

FitStatistics (where=(Criterion='SC') rename=(InterceptAndCovariates=SC) keep=Criterion InterceptAndCovariates)

FitStatistics (where=(Criterion='-2 Log L') rename=(InterceptAndCovariates=LogL2) keep=Criterion InterceptAndCovariates)

%end;

%else %do;

FitStatistics (where=(Criterion='AIC') rename=(WithCovariates=AIC) keep=Criterion WithCovariates)

FitStatistics (where=(Criterion='SC') rename=(WithCovariates=SC) keep=Criterion WithCovariates)

FitStatistics (where=(Criterion='-2 Log L') rename=(WithCovariates=LogL2) keep=Criterion WithCovariates)

%end;

RSquare (rename=(nValue1=Rsquare) keep=nValue1)

Association (where=(label2="Somers' D") rename=(nvalue2=SomersD) keep=label2 nvalue2)

Association (where=(label2='Gamma') rename=(nvalue2=Gamma) keep=label2 nvalue2)

Association (where=(label2='Tau-a') rename=(nvalue2=TauA) keep=label2 nvalue2)

Association (where=(label2='c') rename=(nvalue2=c) keep=label2 nvalue2)

Association (where=(label1='Percent Concordant') rename=(nvalue1=Concord) keep=label1 nvalue1)

Association (where=(label1='Percent Discordant') rename=(nvalue1=Discon) keep=label1 nvalue1)

LackFitChiSq (rename=(ProbChiSq=LackFit) keep=ProbChiSq);

label AIC=" ";

label SC=" ";

label LogL2=" ";

label LackFit=" ";

run;

%\* Assign groups;

data tmp0; set scoretmp (keep=&dep\_var pred &weight);

length group $4.;

bad = (&dep\_var = &bads);

good = (&dep\_var = &goods);

if &dep\_var = &bads then group = 'bad';

else if &dep\_var = &goods then group = 'good';

%if &weight ^= %then %do; xxx=round(&weight); %end;

run;

%\* Create deciles based on predicted value;

%if &weight = %then %do;

proc rank data=tmp0 out=tmp1 groups=10;

var pred;

ranks rank;

run;

%end;

%else %do;

proc sort data=tmp0 out=tmp1; by pred; run;

data tmp1; set tmp1;

retain cum;

cum + &weight;

rank = (floor(cum\*10/(&vnobs+1)));

run;

%end;

%\* Summarize file by rank;

proc summary data=tmp1;

var &dep\_var pred good bad;

class rank;

%if &weight ne %then %do; weight &weight; %end;

%if &weight ne %then %do; output out=tmp2 (drop= \_freq\_) sumwgt=count %end;

%else %do; output out=tmp2 (rename= \_freq\_=count) %end;

mean(&dep\_var)=mean\_dv mean(pred)=mean\_pred sum(bad)=badcnt sum(good)=goodcnt;

run;

data \_null\_; set tmp2 (where=(\_type\_=0));

call symputx('Norm',mean\_dv);

run;

%\* Perform Kolmogorov-Smirnov test;

proc npar1way data=tmp1 edf noprint;

class &dep\_var;

var pred;

%if &weight ne %then %do; freq xxx; %end;

output edf out=kolsmir;

run;

proc sql noprint; select round(\_d\_,.000001) into: kstmp from kolsmir; quit;

%\* Create custom statistics;

data tmp3 (keep=Lift\_Index infv ks);

set tmp2 (where=(\_type\_=1)) end=eof;

Lift\_Index = mean\_dv / &norm \* 100;

retain infv cumbad cumgood;

badratio=badcnt\*(1.0/&vnbads);

goodratio=goodcnt\*(1.0/&vngoods);

if \_n\_=1 then do;

infv=0.0;

cumgood=0.0;

cumbad=0.0;

end;

%\* Information\_value;

if badratio+goodratio>=0.000000001 then do;

if min(badratio,goodratio)<=0.000000001 then do;

infv=infv+count\*max(badratio,goodratio)/&vnobs;

end;

else do;

infv=infv+(badratio-goodratio)\*log(badratio/goodratio);

end;

end;

cumbad = cumbad + badratio;

cumgood = cumgood + goodratio;

%\* K-S value;

ks=&kstmp;

if eof;

run;

data globalfit; merge globalfit tmp3; run;

%\* Variance inflation factor;

%if &vif=Y %then %do;

data scoretmp; set scoretmp;

weight=pred\*(1-pred);

run;

ods listing close;

ods graphics on;

ods output ParameterEstimates=VIF;

proc reg data=scoretmp plots=none;

weight weight;

model &dep\_var = &varlist / vif;

run;

ods graphics off;

ods listing;

%end;

%\* Clean up;

proc datasets library=work nolist;

delete FitStatistics RSquare Association LackFitChiSq

scoretmp tmp0 tmp1 tmp2 tmp3 kolsmir;

run;

quit;

%mend;

\*\* Control program for continuous stats \*\*;

%macro ctl\_Stats\_Reg(insdn, varlisttmp, nvars);

%global vnobs;

%if &weight= %then %do;

proc sql noprint;

select count(\*) into: vnobs from &insdn;

quit;

%end;

%else %do;

proc sql noprint;

select sum(&weight) into: vnobs from &insdn;

quit;

%end;

%\*\* Get full model statistics \*\*;

%Stats\_Reg(&insdnV, &varlisttmp, Full Model, int=N, vif=Y);

data full; set globalfit; run;

%\*\* Get model statistics without intercept \*\*;

%Stats\_Reg(&insdnV, &varlisttmp, Intercept, int=Y, vif=N);

data vars; set globalfit; run;

%\*\* Get model statistics without specific variables \*\*;

%do I = 1 %to &nvars;

%let var = %scan(&varlisttmp,&I);

%let tmplist=%StringMinus(&varlisttmp, &var); /\* Remove single variable \*/

%Stats\_Reg(&insdnV, &tmplist, &var, int=N, vif=N);

data vars; set vars globalfit; run;

%end;

%\*\* Create macro variables for full models metrics \*\*;

data \_null\_; set full;

call symputx('Rsquare',Rsquare);

call symputx('AdjRsq',AdjRsq);

call symputx('RMSE',RMSE);

call symputx('CoeffVar',CoeffVar);

call symputx('AIC',AIC);

call symputx('SBC',SBC);

call symputx('PC',PC);

call symputx('JP',JP);

call symputx('Lift\_Index',Lift\_Index);

call symputx('gini',gini);

call symputx('infv',infv);

call symputx('ks',ks);

run;

proc sql noprint;

select sum(abs(standardizedest)) into: cum from vif where variable ^= 'Intercept';

quit;

%\*\* Subtract full model metrics from sub-model metrics;

%\*\* This shows the impact of each variable;

data tmp1; set vars;

Rsquare = Rsquare - &Rsquare;

AdjRsq = AdjRsq - &AdjRsq;

RMSE = RMSE - &RMSE;

CoeffVar = CoeffVar - &CoeffVar;

AIC = AIC - &AIC;

SBC = SBC - &SBC;

PC = PC - &PC;

JP = JP - &JP;

Lift\_Index = Lift\_Index - &Lift\_Index;

gini = gini - &gini;

infv = infv - &infv;

ks = ks - &ks;

run;

%\*\* Combine datasets \*\*;

data tmp2; set full (in=a) tmp1;

if a then sord = 1;

else if variable = 'Intercept' then sord = 2;

else sord = 3;

run;

proc sql;

create table stats as

select a.Variable,

b.Label,

b.Estimate,

b.StandardizedEst as StdEst,

b.StdErr,

b.tValue,

b.Probt,

b.VarianceInflation as VIF,

abs(b.StandardizedEst)/&cum as RelImp,

a.Rsquare,

a.AdjRsq,

a.RMSE,

a.CoeffVar,

a.AIC,

a.SBC,

a.PC,

a.JP,

a.Lift\_Index,

a.gini,

a.infv,

a.ks,

a.sord

from tmp2 a

left join vif b on a.variable = b.variable

order by a.sord, RelImp desc;

quit;

%\*\* Clean up;

proc datasets library=work nolist;

delete globalfit full vars tmp1 tmp2 vif;

run;

quit;

%mend;

\*\* Stats\_Reg gets statistics for continuous models \*\*;

%macro Stats\_Reg(insdn, varlist, var, int=N, vif=N);

%\* Extract standard model statistics;

ods listing close;

ods graphics on;

ods output FitStatistics=FitStatistics;

%if &vif=Y %then %do;

options label;

ods output ParameterEstimates=VIF;

%end;

proc reg data=&insdn outest=outest plots=none;

%if &weight ne %then %do; weight &weight; %end;

model &dep\_var = &varlist / stb aic sbc jp pc

%if &vif=Y %then %do; vif %end;

%if &int=Y %then %do; noint %end;

;

output out=scoretmp p=pred;

run;

ods graphics off;

ods listing;

quit;

data globalfit (drop=label1 label2);

length variable $32.;

variable = "&var";

merge FitStatistics (where=(label2="R-Square") rename=(nvalue2=Rsquare) keep=label2 nvalue2)

FitStatistics (where=(label2="Adj R-Sq") rename=(nvalue2=AdjRsq) keep=label2 nvalue2)

FitStatistics (where=(label1="Root MSE") rename=(nvalue1=RMSE) keep=label1 nvalue1)

FitStatistics (where=(label1="Coeff Var") rename=(nvalue1=CoeffVar) keep=label1 nvalue1)

outest (keep=\_AIC\_ \_SBC\_ \_PC\_ \_JP\_ rename=(\_AIC\_=AIC \_SBC\_=SBC \_PC\_=PC \_JP\_=JP));

run;

%\* Create deciles based on predicted value;

%if &weight = %then %do;

proc rank data=scoretmp (keep=&dep\_var pred &weight) out=tmp1 groups=10;

var pred;

ranks rank;

run;

%end;

%else %do;

proc sort data=scoretmp (keep=&dep\_var pred &weight) out=tmp1; by pred; run;

data tmp1; set tmp1;

retain cum;

cum + &weight;

rank = (floor(cum\*10/(&vnobs+1)));

run;

%end;

%\* Summarize file by rank;

proc summary data=tmp1;

var &dep\_var pred;

class rank;

%if &weight ne %then %do; weight &weight; %end;

%if &weight ne %then %do; output out=tmp2 (drop= \_freq\_) sumwgt=count %end;

%else %do; output out=tmp2 (rename= \_freq\_=count) %end;

mean(&dep\_var)=mean\_dv mean(pred)=mean\_pred;

run;

%\* Create custom statistics;

proc sql noprint;

select mean\_dv into: norm from tmp2 (where=(\_type\_=0));

select count into: ntotal from tmp2 (where=(\_type\_=0));

select sum(count\*mean\_dv) into: totalresp from tmp2 (where=(\_type\_=1));

select sum(count\*(count\*mean\_dv/&totalresp)) into: totalbad from tmp2 (where=(\_type\_=1));

quit;

data tmp3 (keep=lift\_index gini infv ks);

set tmp2 (where=(\_type\_=1)) end=eof;

Lift\_Index = mean\_dv / &norm \* 100;

retain cumresp cumtotal;

if \_n\_ = 1 then do;

cumresp = 0;

cumtotal = 0;

end;

cumresp = cumresp + mean\_dv \* count;

cumpct\_resp = cumresp / &totalresp;

cumtotal = cumtotal + count;

cumpct\_freq = cumtotal / &ntotal;

cumavg = cumresp / cumtotal;

cumindex = cumavg / &norm \* 100;

badrate = count \* mean\_dv / &totalresp;

goodrate = (1 - badrate) \* count / (&ntotal - &totalbad);

retain gini infv ks cumbad cumgood 0;

prev\_dv = lag(cumpct\_resp);

prev\_pct = lag(cumpct\_freq);

if \_n\_ ne 1 then gini = gini+2\*((cumpct\_freq+prev\_pct)/2-(cumpct\_resp+prev\_dv)/2)\*(cumpct\_freq-prev\_pct);

infv = infv+(badrate-goodrate)\*log(badrate/goodrate);

cumbad = cumbad+badrate;

cumgood = cumgood+goodrate;

ks = max(ks,abs(cumbad-cumgood));

if eof;

run;

data globalfit; merge globalfit tmp3; run;

%\* Clean up;

proc datasets library=work nolist;

delete FitStatistics outest scoretmp tmp1 tmp2 tmp3;

run;

quit;

%mend;

\*\* Variable Tuning and Selection Based on model metric output \*\*;

%macro Vars\_tune(dt1=, dt2=, out\_txt=&path\_output.CE4\_Varlist\_Final.txt);

%\* Check criteria and reset if necessary;

%if %upcase(&binary\_dv) = Y %then %do;

%let c=AIC SC LOGL2 RSQUARE SOMERSD GAMMA TAUA C CONCORD DISCON LACKFIT LIFT\_INDEX INFV KS;

%if %sysfunc(indexw(&c,%upcase(&criteria)))=0 %then %do;

%let criteria = c;

%put Criteria changed to c;

%end;

%end;

%else %do;

%let c=AIC SBC JP RSQUARE ADJRSQ RMSE COEFFVAR PC LIFT\_INDEX GINI INFV KS;

%if %sysfunc(indexw(&c,%upcase(&criteria)))=0 %then %do;

%let criteria = AdjRsq;

%put Criteria changed to AdjRsq;

%end;

%end;

%let neg=RSQUARE SOMERSD GAMMA TAUA C CONCORD INFV TVR KS DIVERG ADJRSQ LIFT\_INDEX GINI;

%if &threshold ^= 0 and %index(&neg,%upcase(&criteria))>0 %then %do;

%let threshold = %sysevalf(-1\*&threshold);

%end;

proc sql;

create table tempfinal as

select variable

from &dt1

where sord = 3 and &criteria <= &threshold

and RelImp >= &MinImp

and file = 'Val'

&SQL\_join

select variable

from &dt2

where sord = 3 and &criteria <= &threshold

and RelImp >= &MinImp

and file = 'Val';

quit;

data \_NULL\_;

set tempfinal end=eof;

FILE "&out\_txt" LRECL=256;

if \_N\_=1 then do;

PUT ' ';

PUT "%"@;

PUT "LET Varlist\_Final=";

end;

PUT variable @;

if eof then PUT ';';

run;

proc sort data=tempfinal; by variable;

data out.ce4\_variables;

merge out.ce4\_variables tempfinal (in=a);

by variable;

if a then final = 'Y';

run;

proc datasets library=work nolist;

delete tempfinal;

quit;

run;

%mend;

\*\* Create Graph report \*\*;

%macro Grafing (inds,varlist,cats);

%\*\* Get variables and unique counts \*\*;

proc sql;

create table cnts as

select

%let i=1;

%let v=%scan(&varlist, &i,' ');

%do %while (&v^=);

count(distinct(&v)) as &v,

%let i=%eval(&i+1);

%let v=%scan(&varlist, &i,' ');

%end;

"dummy" as dummy

from &inds;

quit;

proc transpose data=cnts (drop=dummy) out=cnts (rename=col1=uniq) name=variable; run;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set cnts end=eof;

if eof then call symputx("VARCNT",\_N\_);

call symputx("IV"|| trim(left(put(\_N\_,4.))) ,variable);

call symputx("uniq"|| trim(left(put(\_N\_,4.))) ,uniq);

run;

goptions reset=all device=pdf display gunit=pct border ftext= htitle=8 htext=3;

ods listing close;

ods pdf file="&path\_output.CE4\_Graphs.pdf" style=sasweb startpage=no;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%if &\_I\_ ^= 1 %then %do; ods pdf startpage=now; %end;

%if &&uniq&\_I\_ <= &cats %then %do;

proc summary data=&inds nway missing;

var &dep\_var;

class &&iv&\_I\_;

output out=graf (drop=\_type\_ rename=\_freq\_=count) mean=mean;

run;

proc print data=graf noobs;

title "Variable: &&iv&\_I\_";

var &&iv&\_I\_ count mean;

format count comma8.;

format mean 8.4;

run;

proc sgplot data=graf;

vbar &&iv&\_I\_ / response=mean nostatlabel /\*nooutline\*/ fillattrs=(color="lightblue") /\*transparency=.5\*/;

vline &&iv&\_I\_ / response=count nostatlabel y2axis lineattrs=(color="darkblue" thickness=2);

label count="# of Customers";

label mean="Mean &dep\_var";

label &&iv&\_I\_="&&iv&\_I\_";

keylegend / location = outside

position = top

noborder

title = "&&iv&\_I\_";

format count comma8.;

run;

%end;

%else %do;

proc means data=&inds p1 p99 NOPRINT;

var &&&iv&\_I\_;

output out=tmp p1=var\_p1 p99=var\_p99 / noinherit;

run;

data \_NULL\_;

set tmp;

range = (var\_p99 - var\_p1)/&cats;

call symputx('cut\_lo',var\_p1);

call symputx('cut\_hi',var\_p99);

call symputx('range',range);

run;

data tmp;

set &inds (keep=&dep\_var &&iv&\_I\_);

if &&iv&\_I\_ < &cut\_lo then bin = 1;

else if &&iv&\_I\_ >= &cut\_hi then bin = input("&cats",best10.);

else do;

do k = 1 to &cats;

if &&iv&\_I\_ >= &cut\_lo+(k-1)\*&range and &&iv&\_I\_< &cut\_lo+k\*&range then bin=k;

end;

end;

run;

proc summary data=tmp nway missing;

var &&iv&\_I\_ &dep\_var;

class bin;

output out=graf (drop=\_type\_ rename=\_freq\_=count) min(&&iv&\_I\_)=lo max(&&iv&\_I\_)=hi mean(&dep\_var)=mean;

run;

proc print data=graf noobs;

title "Variable: &&iv&\_I\_";

var bin count lo hi mean;

format count comma8.;

format lo hi mean 8.4;

run;

proc sgplot data=graf;

vbar lo / response=mean nostatlabel fillattrs=(color="lightblue");

vline lo / response=count nostatlabel y2axis lineattrs=(color="darkblue" thickness=2);

label count="# of Customers";

label mean="Mean &dep\_var";

label lo="&&iv&\_I\_";

keylegend / location = outside

position = top

noborder

title = "&&iv&\_I\_";

format count comma8.;

run;

proc datasets nolist;

delete tmp;

run;

%end;

%end;

ods pdf close;

ods listing;

proc datasets nolist;

delete cnts graf;

run;

%mend;

\*\* Master macro for Model selection and tuning \*\*;

%macro CE\_Model\_Val(insdn, varlist);

data mod val;

set &insdn (where=(mod\_val\_test ^= 3));

keep &Dep\_var mod\_val\_test &varlist &weight;

if mod\_val\_test=1 then output mod;

else if mod\_val\_test=2 then output val;

run;

%\* Start variable table;

data out.ce4\_variables;

length variable $32;

%Let I = 1;

%Let var = %scan(&varlist,&I);

%Do %while(&var ne );

variable="&var";

output;

%Let I = %eval(&I + 1);

%Let var = %scan(&varlist,&I);

%end;

run;

proc sort data=out.ce4\_variables; by variable; run;

%if %upcase(&Binary\_dv) = Y %then %do;

%Model\_Val\_Logistic(mod, val, &varlist,

sle=&sel\_alpha,

sls=&sel\_alpha,

metric\_sdnout=out.CE4\_Model\_Metric\_Mod);

proc sort data=parm1 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

proc sort data=parm2 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

data out.ce4\_variables;

merge out.ce4\_variables parm1 (in=a) parm2 (in=b);

by variable;

if a then Tforward = 'Y';

if b then Tbackward = 'Y';

run;

%Model\_Val\_Logistic(val, mod, &varlist,

sle=&sel\_alpha,

sls=&sel\_alpha,

metric\_sdnout=out.CE4\_Model\_Metric\_Val);

proc sort data=parm1 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

proc sort data=parm2 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

data out.ce4\_variables;

merge out.ce4\_variables parm1 (in=a) parm2 (in=b);

by variable;

if a then Vforward = 'Y';

if b then Vbackward = 'Y';

run;

%\*model selection and tuning;

%Vars\_tune(dt1=out.CE4\_Model\_Metric\_Mod,

dt2=out.CE4\_Model\_Metric\_Val,

out\_txt=&path\_output.CE4\_Varlist\_Final.txt);

%\* Report;

ods listing close;

ods Tagsets.ExcelxP body="&path\_output.CE4\_Model\_report.xls" style=sasweb;

ods tagsets.excelxp options(sheet\_name="Variables");

proc print data=out.ce4\_variables noobs; run;

ods tagsets.excelxp options(sheet\_name="Model" embedded\_titles="yes");

proc print data=out.CE4\_Model\_Metric\_Mod noobs;

title "Diagnostic statistics: Model Portion";

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr WaldChiSq ProbChiSq VIF RelImp / style={tagattr='format:0.0000'};

var AIC SC LogL2 / style={tagattr='format:0.00'};

var Rsquare SomersD Gamma TauA c Concord Discon LackFit / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var infv ks / style={tagattr='format:0.0000'};

run;

title ;

ods tagsets.excelxp options(sheet\_name="Validation" embedded\_titles="yes");

proc print data=out.CE4\_Model\_Metric\_Val noobs;

title "Diagnostic statistics: Validation Portion";

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr WaldChiSq ProbChiSq VIF RelImp / style={tagattr='format:0.0000'};

var AIC SC LogL2 / style={tagattr='format:0.00'};

var Rsquare SomersD Gamma TauA c Concord Discon LackFit / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var infv ks / style={tagattr='format:0.0000'};

run;

title ;

ods Tagsets.ExcelxP close;

ods listing;

%end;

%else %if %upcase(&Binary\_dv) ^= Y %then %do;

%Model\_Val\_Reg(mod, val, &varlist,

sle=&sel\_alpha,

sls=&sel\_alpha,

metric\_sdnout=out.CE4\_Model\_Metric\_Mod);

proc sort data=parm1 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

proc sort data=parm2 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

data out.ce4\_variables;

merge out.ce4\_variables parm1 (in=a) parm2 (in=b);

by variable;

if a then Tforward = 'Y';

if b then Tbackward = 'Y';

run;

%Model\_Val\_Reg(val, mod, &varlist,

sle=&sel\_alpha,

sls=&sel\_alpha,

metric\_sdnout=out.CE4\_Model\_Metric\_Val);

proc sort data=parm1 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

proc sort data=parm2 (keep=variable where=(variable ^= 'Intercept')); by variable; run;

data out.ce4\_variables;

merge out.ce4\_variables parm1 (in=a) parm2 (in=b);

by variable;

if a then Vforward = 'Y';

if b then Vbackward = 'Y';

run;

%\*model selection and tuning;

%Vars\_tune(dt1=out.ce4\_model\_metric\_mod,

dt2=out.ce4\_model\_metric\_val,

out\_txt=&path\_output.CE4\_Varlist\_Final.txt);

%\* Report;

ods listing close;

ods Tagsets.ExcelxP body="&path\_output.CE4\_Model\_report.xls" style=sasweb;

ods tagsets.excelxp options(sheet\_name="Variables");

proc print data=out.ce4\_variables noobs; run;

ods tagsets.excelxp options(sheet\_name="Model" embedded\_titles="yes");

proc print data=out.CE4\_Model\_Metric\_Mod noobs;

title "Diagnostic statistics: Model Portion";

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr tValue Probt VIF RelImp / style={tagattr='format:0.0000'};

var AIC SBC JP / style={tagattr='format:0.00'};

var Rsquare AdjRsq RMSE CoeffVar PC / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var gini infv ks / style={tagattr='format:0.0000'};

run;

title ;

ods tagsets.excelxp options(sheet\_name="Validation" embedded\_titles="yes");

proc print data=out.CE4\_Model\_Metric\_Val noobs;

title "Diagnostic statistics: Validation Portion";

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr tValue Probt VIF RelImp / style={tagattr='format:0.0000'};

var AIC SBC JP / style={tagattr='format:0.00'};

var Rsquare AdjRsq RMSE CoeffVar PC / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var gini infv ks / style={tagattr='format:0.0000'};

run;

title ;

ods Tagsets.ExcelxP close;

ods listing;

run;

title ;

%end;

%\* Do charting?;

%if %upcase(&graph\_plot)=Y %then %do;

%inc "&path\_output.CE4\_Varlist\_Final.txt";

data gdat; set &insdn (where=(mod\_val\_test ^= 3)); run;

%Grafing(gdat,&Varlist\_Final,20);

proc datasets library=work nolist;

delete gdat;

run;

quit;

%end;

proc datasets library=work nolist;

delete mod val parm1 parm2;

run;

quit;

%mend CE\_Model\_Val;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* 5.Final Model build and validation on test sample \*\*\*;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\* Master macro for binary dependent variable \*\*;

%macro Model\_Fin\_Logistic(insdnT, insdnV, regvlist, sle=0.05, sls=.05,

metric\_sdnout=out.CE5\_Model\_Metric\_All,

out\_txt=&path\_output.CE5\_Varlist\_Model.txt, outds=out.CE5\_scored);

%\*\* Build model \*\*;

ods listing close;

ods output ParameterEstimates=Parm1;

ods output ModelBuildingSummary= out.ce5\_SelectionSummary;

proc logistic data=&insdnT outmodel=est desc namelen=32;

%if &weight ne %then %do; weight &weight; %end;

model &dep\_var = &regvlist /selection=&method stb lackfit rsq details

%if &method = stepwise or &method = forward %then %do; slentry=&sle %end;

%if &method = stepwise or &method = backward %then %do; slstay=&sls %end;

;

output out=out1 P=pscore;

run;

ods listing;

%\*\* Extract variables \*\*;

proc sql noprint;

select max(step) into: max1 from Parm1;

quit;

data parm1; set parm1 (where=(step=&max1)); run;

proc sql noprint;

select variable into: varlisttmp separated by ' ' from parm1 where variable ^= 'Intercept';

select count(\*) into: nvars from parm1 where variable ^= 'Intercept';

quit;

%put &varlisttmp;

%\* Score validation dataset;

proc logistic inmodel=est;

score data=&insdnV out=out2;

run;

data &outds; set out1 (drop=\_level\_) out2 (drop=f\_&dep\_var I\_&dep\_var p\_0 rename=(P\_1=pscore)); run;

%ctl\_Stats\_Log(&insdnT, &varlisttmp, &nvars);

data &metric\_sdnout;

length file $4.;

set stats;

file = 'Dev';

run;

%ctl\_Stats\_Log(&insdnV, &varlisttmp, &nvars);

data &metric\_sdnout;

set &metric\_sdnout stats (in=a);

if a then file = 'Val';

run;

%\* Write out final variable list;

data \_NULL\_;

set parm1 (where=(variable ^= 'Intercept')) end=eof;

FILE "&out\_txt" LRECL=256;

if \_N\_=1 then do;

PUT ' ';

PUT "%"@;

PUT "LET Varlist\_Final=";

end;

PUT variable @;

if eof then PUT ';';

run;

%\*\* Clean up;

data out.ce5\_parameterestimates; set parm1 (drop=step); run;

proc datasets library=work nolist;

delete parm1 stats est out1 out2;

run;

quit;

%mend;

\*\* Master macro for continuous dependent variable \*\*;

%macro Model\_Fin\_Reg(insdnT, insdnV, regvlist, sle=0.05, sls=.05,

metric\_sdnout=out.CE5\_Model\_Metric\_All,

out\_txt=&path\_output.CE5\_Varlist\_Model.txt, outds=out.CE5\_scored);

%\*\* Build model \*\*;

ods listing close;

ods graphics on;

ods output SelParmEst=Parm1;

ods output SelectionSummary=out.ce5\_SelectionSummary;

proc reg data=&insdnT outest=est plots(maxpoints=none);

%if &weight ne %then %do; weight &weight; %end;

model &dep\_var = &regvlist /selection=&method stb details

%if &method = stepwise or &method = forward %then %do; slentry=&sle %end;

%if &method = stepwise or &method = backward %then %do; slstay=&sls %end;

;

output out=out1 P=pscore;

run;

ods listing;

ods graphics off;

quit;

%\*\* Extract variables \*\*;

proc sql noprint;

select max(step) into: max1 from Parm1;

quit;

data parm1; set parm1 (where=(step=&max1)); run;

proc sql noprint;

select variable into: varlisttmp separated by ' ' from parm1 where variable ^= 'Intercept';

select count(\*) into: nvars from parm1 where variable ^= 'Intercept';

quit;

%put &varlisttmp;

%\* Score validation dataset;

proc score data=&insdnV score=est out=out2 type=parms;

var &varlisttmp;

run;

data &outds; set out1 out2 (rename=(model1=pscore)); run;

%ctl\_Stats\_Reg(&insdnT, &varlisttmp, &nvars);

data &metric\_sdnout;

length file $4.;

set stats;

file = 'Dev';

run;

%ctl\_Stats\_Reg(&insdnV, &varlisttmp, &nvars);

data &metric\_sdnout;

set &metric\_sdnout stats (in=a);

if a then file = 'Val';

run;

%\* Write out final variable list;

data \_NULL\_;

set parm1 (where=(variable ^= 'Intercept')) end=eof;

FILE "&out\_txt" LRECL=256;

if \_N\_=1 then do;

PUT ' ';

PUT "%"@;

PUT "LET Varlist\_Final=";

end;

PUT variable @;

if eof then PUT ';';

run;

%\*\* Clean up;

data out.ce5\_parameterestimates; set parm1 (drop=step); run;

proc datasets library=work nolist;

delete parm1 stats est out1 out2;

run;

quit;

%mend;

\*\* Build gains table to check performance \*\*;

%macro gains(inputfile = , score = , varlist= , title\_key = );

%\* Create deciles based on predicted value;

%if &weight = %then %do;

proc rank data=&inputfile out=tmp1 descending groups=10;

var &score;

ranks decile;

run;

%end;

%else %do;

proc sql noprint;

select sum(&weight) into: nobs from &inputfile;

quit;

proc sort data=&inputfile out=tmp1; by descending &score; run;

data tmp1; set tmp1;

retain cum;

cum + &weight;

decile = (floor(cum\*10/(&nobs+1)));

run;

%end;

%\* Summarize file by decile;

proc summary data=tmp1;

var &dep\_var &score &varlist ;

class decile;

%if &weight ne %then %do; weight &weight; %end;

%if &weight ne %then %do; output out=tmp2 (drop= \_freq\_) sumwgt=count %end;

%else %do; output out=tmp2 (rename= \_freq\_=count) %end;

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

%if &binary\_dv=Y %then %do; sum(&dep\_var)=numresp %end;

;

run;

data out.ce5\_profiles\_&title\_key;

set tmp2 (drop=minscore maxscore %if &binary\_dv=Y %then %do; numresp %end;);

decile = decile + 1;

run;

%\* Model Performance;

%if &binary\_dv = Y %then %do;

proc sql;

select (numresp/count) into: avg from tmp2 where \_type\_=0;

select numresp into: totresp from tmp2 where \_type\_=0;

quit;

data out.ce5\_gainstable\_&title\_key (drop=\_type\_ cumresp cumcount);

set tmp2 (drop=&varlist where=(\_type\_=1));

decile = decile + 1;

avg\_rate = numresp / count;

lift = (avg\_rate / &avg)\*100;

retain cumresp cumcount 0;

cumresp + numresp;

cumcount + count;

cum\_index = ((cumresp/cumcount) / &avg) \* 100;

resp\_pct = numresp / &totresp;

cum\_resp\_pct = cumresp / &totresp;

run;

%end;

%else %do;

proc sql;

select &dep\_var into: avg from tmp2 where \_type\_=0;

quit;

data out.ce5\_gainstable\_&title\_key (drop=\_type\_);

set tmp2 (drop=&varlist where=(\_type\_=1));

decile = decile + 1;

lift = (&dep\_var / &avg)\*100;

run;

%end;

%\* Clean up;

proc datasets library=work nolist;

delete tmp1 tmp2;

run;

quit;

%mend;

\*\* Profiling Control \*\*;

%macro Fin\_Profiling (inds,varlist);

%\*\* Initialize profiling dataset \*\*;

data out.CE5\_profile;

set \_NULL\_;

length variable $32.;

length label $256.;

length category $256.;

format count comma8.;

format percent percent8.2;

%if %upcase(&binary\_dv) = Y %then %do;

format Average\_DV percent8.2;

%end;

%else %do;

format Average\_DV 12.2;

%end;

format index 8.0;

length star $8.;

run;

%\*\* Get overall average and count \*\*;

proc sql noprint;

select count(\*) into : nobs from &inds;

select mean(&dep\_var) into : overall\_avg from &inds;

quit;

%\*\* Get variables and unique counts \*\*;

proc sql;

create table cnts as

select

%let i=1;

%let v=%scan(&varlist, &i,' ');

%do %while (&v^=);

count(distinct(&v)) as &v,

%let i=%eval(&i+1);

%let v=%scan(&varlist, &i,' ');

%end;

"dummy" as dummy

from &inds;

quit;

proc transpose data=cnts (drop=dummy) out=cnts (rename=col1=uniq) name=variable; run;

%\*\* Create macro variables to control processing \*\*;

data \_null\_; set cnts end=eof;

if eof then call symputx("VARCNT",\_N\_);

call symputx("IV"|| trim(left(put(\_N\_,4.))) ,variable);

call symputx("uniq"|| trim(left(put(\_N\_,4.))) ,uniq);

run;

%\*\* Loop through for each variable \*\*;

%do \_I\_ = 1 %to &varcnt;

%if &&uniq&\_I\_ <= &fin\_num\_category %then %do;

%fprof1(&inds,&&iv&\_I\_);

%end;

%else %do;

%fprof2(&inds,&&iv&\_I\_);

%end;

%fprof3(&&iv&\_I\_);

%end;

proc datasets nolist;

delete cnts;

run;

%mend;

\*\* Profiling macro 1 \*\*;

%macro fprof1(insdn,var);

proc summary data=&insdn nway missing;

var &dep\_var;

class &var;

output out=prof (drop=\_type\_ rename=\_freq\_=xcount) mean=xmean;

run;

data prof;

set prof;

length xcategory $256.;

xcategory = trim(left(put(&var,best8.)));

run;

%mend;

\*\* Profiling macro 2 \*\*;

%macro fprof2(insdn, var);

%if %upcase(&fin\_equal\_dist) = Y %then %do;

proc means data=&insdn p1 p99 NOPRINT;

var &var;

output out=tmp p1=var\_p1 p99=var\_p99 / noinherit;

run;

data \_NULL\_;

set tmp;

range = (var\_p99 - var\_p1)/&fin\_num\_category;

call symputx('cut\_lo',var\_p1);

call symputx('cut\_hi',var\_p99);

call symputx('range',range);

run;

data tmp;

set &insdn (keep=&dep\_var &var);

if &var < &cut\_lo then bin = 1;

else if &var >= &cut\_hi then bin = input("&fin\_num\_category",best10.);

else do;

do k = 1 to &fin\_num\_category;

if &var>= &cut\_lo+(k-1)\*&range and &var< &cut\_lo+k\*&range then bin=k;

end;

end;

run;

%end;

%else %do;

proc rank data=&insdn (keep=&dep\_var &var) out=tmp ties=High group=&fin\_num\_category;

var &var;

ranks bin;

run;

%end;

proc summary data=tmp nway missing;

var &var &dep\_var;

class bin;

output out=prof (drop=\_type\_ rename=\_freq\_=xcount) min(&var)=lo max(&var)=hi mean(&dep\_var)=xmean;

run;

data prof;

set prof end=eof;

length xcategory $256.;

if \_N\_ = 1 then xcategory = "Low to " || trim(left(put(hi,best8.)));

else if eof then xcategory = trim(left(put(lo,best8.))) || " to High";

else xcategory = trim(left(put(lo,best8.))) || " to " || trim(left(put(hi,best8.)));

run;

proc datasets nolist;

delete tmp;

run;

%mend;

\*\* Profiling macro 3 \*\*;

%macro fprof3(var);

proc sql;

create table prof2 as

select a.variable, c.label, b.xcategory, b.xcount, b.xmean

from cnts a, prof b, out.ce2\_vars c

where a.variable = "&var"

and c.variable = "&var";

quit;

data prof3 (drop=xcount xmean xcategory);

set prof2 end=eof;

length category $256.;

length star $8.;

if \_N\_ = 1 then do;

category = "Overall";

Average\_DV = &overall\_avg;

Count = &nobs;

Percent = 1;

index = 100;

output;

end;

category = xcategory;

count = xcount;

percent = xcount / &nobs;

Average\_DV = xmean;

index = (Average\_DV / &overall\_avg)\*100;

if index >= 110 then star = '\* (+)';

else if index > 100 then star = ' (+)';

else if index <= 90 then star = '\* (-)';

else if index <= 100 then star = ' (-)';

else star = ' (0)';

output;

run;

data out.CE5\_profile;

set out.CE5\_profile prof3;

run;

proc datasets nolist;

delete prof prof2 prof3;

run;

%mend;

\*\* Master macro for Final Model build and validation \*\*;

%macro CE\_Model\_Lift(insdn, varlist);

data mod val;

set &insdn (keep=&Dep\_var mod\_val\_test &varlist &weight);

if mod\_val\_test=3 then output val;

else output mod;

run;

%if %upcase(&Binary\_dv) = Y %then %do;

%Model\_Fin\_Logistic(mod, val, &varlist,

sle=&fin\_alpha,

sls=&fin\_alpha,

metric\_sdnout=out.CE5\_Model\_Metric\_All,

out\_txt=&path\_output.CE5\_Varlist\_Model.txt,

outds=out.CE5\_scored);

%end;

%else %if %upcase(&Binary\_dv) ^= Y %then %do;

%Model\_Fin\_Reg(mod, val, &varlist,

sle=&fin\_alpha,

sls=&fin\_alpha,

metric\_sdnout=out.CE5\_Model\_Metric\_All,

out\_txt=&path\_output.CE5\_Varlist\_Model.txt,

outds=out.CE5\_scored);

%end;

%\* Final variable list;

proc sql noprint;

select variable into: tmplst separated by ' ' from out.ce5\_parameterestimates where variable ^= 'Intercept';

quit;

%\* Performance;

%gains(inputfile = out.CE5\_scored (where=(mod\_val\_test ^= 3)), score = pscore,

varlist=&tmplst, title\_key = train);

%gains(inputfile = out.CE5\_scored (where=(mod\_val\_test = 3)), score = pscore,

varlist=&tmplst, title\_key = test);

%\* Profile Variables;

%Fin\_Profiling(mod,&tmplst);

%\* Report;

ods listing close;

ods Tagsets.ExcelxP body="&Path\_output.CE5\_Model\_report.xls" style=sasweb;

%\* Selection Summary;

ods tagsets.excelxp options(sheet\_name="Selection");

proc print data=out.CE5\_SelectionSummary noobs; run;

%\* Parameters;

ods tagsets.excelxp options(sheet\_name="Parameters");

proc print data=out.CE5\_ParameterEstimates noobs; run;

%\* Statistics;

%if %upcase(&Binary\_dv) = Y %then %do;

ods tagsets.excelxp options(sheet\_name="Statistics");

proc print data=out.CE5\_Model\_Metric\_All noobs;

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr WaldChiSq ProbChiSq VIF RelImp / style={tagattr='format:0.0000'};

var AIC SC LogL2 / style={tagattr='format:0.00'};

var Rsquare SomersD Gamma TauA c Concord Discon LackFit / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var infv ks / style={tagattr='format:0.0000'};

run;

%end;

%else %do;

ods tagsets.excelxp options(sheet\_name="Statistics");

proc print data=out.CE5\_Model\_Metric\_All noobs;

var file Variable Label;

var Estimate / style={tagattr='format:0.000000'};

var StdEst StdErr tValue Probt VIF RelImp / style={tagattr='format:0.0000'};

var AIC SBC JP / style={tagattr='format:0.00'};

var Rsquare AdjRsq RMSE CoeffVar PC / style={tagattr='format:0.0000'};

var Lift\_Index / style={tagattr='format:0.00'};

var gini infv ks / style={tagattr='format:0.0000'};

run;

%end;

%\* Performance;

ods tagsets.excelxp options(sheet\_interval="table");

ods tagsets.excelxp options(sheet\_name="Performance" sheet\_interval="none" embedded\_titles="yes" convert\_percentages="yes");

proc print data=out.CE5\_GainsTable\_Train noobs;

title 'Train';

var decile / style={tagattr='format:0'};

var Count / style={tagattr='format:#,##0'};

var &dep\_var pscore / style={tagattr='format:0.0000'};

var minscore maxscore / style={tagattr='format:0.0000'};

%if %upcase(&Binary\_dv) = Y %then %do;

var avg\_rate / style={tagattr='format:0.00%'};

var lift cum\_index / style={tagattr='format:0'};

var resp\_pct cum\_resp\_pct / style={tagattr='format:0.00%'};

%end;

%else %do;

var lift / style={tagattr='format:0'};

%end;

run;

proc print data=out.CE5\_GainsTable\_Test noobs;

title 'Test';

var decile / style={tagattr='format:0'};

var Count / style={tagattr='format:#,##0.'};

var &dep\_var pscore / style={tagattr='format:0.0000'};

var minscore maxscore / style={tagattr='format:0.0000'};

%if %upcase(&Binary\_dv) = Y %then %do;

var avg\_rate / style={tagattr='format:0.00%'};

var lift cum\_index / style={tagattr='format:0'};

var resp\_pct cum\_resp\_pct / style={tagattr='format:0.00%'};

%end;

%else %do;

var lift / style={tagattr='format:0'};

%end;

run;

title ' ';

%\* Variable Validation;

ods tagsets.excelxp options(sheet\_interval="table");

ods tagsets.excelxp options(sheet\_name="Variable Validation" sheet\_interval="none" embedded\_titles="yes");

proc print data=out.CE5\_Profiles\_Train noobs;

title 'Train';

var decile \_type\_ / style={tagattr='format:0'};

var Count / style={tagattr='format:#,##0'};

var &dep\_var pscore &tmplst / style={tagattr='format:0.0000'};

run;

proc print data=out.CE5\_Profiles\_Test noobs;

title 'Test';

var decile \_type\_ / style={tagattr='format:0'};

var Count / style={tagattr='format:#,##0'};

var &dep\_var pscore &tmplst / style={tagattr='format:0.0000'};

run;

title ' ';

%\* Correlations;

proc corr data = out.ce5\_scored outp=out.CE5\_corr noprint; var &dep\_var &tmplst; run;

data out.CE5\_corr (drop=\_type\_) ; set out.CE5\_corr (rename=(\_name\_=variable) where=(\_type\_ = 'CORR')); run;

ods tagsets.excelxp options(sheet\_interval="table");

ods tagsets.excelxp options(sheet\_name="Correlations");

proc print data=out.CE5\_corr noobs;

var variable;

var &dep\_var &tmplst / style={tagattr='format:0.0000'};

run;

%\* Variable Profiles;

proc sql noprint;

select max(length(variable)) into: l1 from out.CE5\_profile;

select max(length(label)) into: l2 from out.CE5\_profile;

select max(length(category)) into: l3 from out.CE5\_profile;

quit;

ods tagsets.excelxp options(sheet\_name="Profiles" embedded\_titles="yes" convert\_percentages="yes"

absolute\_column\_width="&l1,&l2,&l3,8,8,9,8,6" Frozen\_Headers='Yes' Frozen\_RowHeaders='1');

proc print data=out.CE5\_profile noobs;

var variable label category;

var count / style={tagattr='format:#,##0'};

var percent Average\_DV index star;

run;

ods Tagsets.ExcelxP close;

ods listing;

%\* Write out scoring code;

data \_Null\_;

file "&path\_output.CE5\_Scoring\_equation.txt";

set out.ce5\_parameterestimates (keep=variable estimate) end=eof;

if variable = 'Intercept' then do;

put " " '0d'x;

put "\*\*\*\*\* Model \*\*\*\*\*;" '0d'x;

put " " '0d'x;

%if &binary\_dv = Y %then %do;

put "Logit = 1 \* (" estimate +(-1) ") +" '0d'x;

%end;

%else %do;

put "Score = 1 \* (" estimate +(-1) ") +" '0d'x;

%end;

end;

else do;

if eof then put variable "\* (" estimate +(-1) ") ;" '0d'x;

else put variable "\* (" estimate +(-1) ") +" '0d'x;

end;

if eof then do;

put " " '0d'x;

%if &binary\_dv = Y %then %do;

put "Score=1/(1+exp(-Logit));" '0d'x;

%end;

end;

run;

proc datasets library=work nolist;

delete mod val;

run;

quit;

%mend CE\_Model\_Lift;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;