

Paper 222-2010

ROC Hard? No, ROC Made Easy!

Kriss Harris, GlaxoSmithKline, UK

SUMMARY

ROC, Specificity, Sensitivity, Receiver Operating Curves, Graph Template Language, GTL, SGE, SGRENDER, SGPLOT.

ABSTRACT

A ROC (Receiver Operator Characteristic) curve shows how well two groups are separated by plotting the Sensitivity by $1 - \text{Specificity}$. When used alone, the cut off used to balance the desired Sensitivity and Specificity is not shown. Furthermore the ROC curve does not display the interesting counts and percentages of the True Positives, True Negatives, False Positives, and False Negatives.

The macro presented in this paper uses Graph Template Language (GTL) and the Statistical Graphics Engine (SGE) to quickly and easily create an informative graph that can be edited and which displays the Sensitivity and Specificity of your binary classifier. The macro also displays scatter and box plots of the desired variable by the binary classifiers to identify unusual results. Options also display a confusion matrix and area under the curve (AUC) results. The confusion matrix is calculated from the default Sensitivity and Specificity intersection, but the cut off can be made explicitly. The AUC statistic shows how well the binary classifier discriminates. This macro is used with SAS® 9.2 Phase 2.

INTRODUCTION

The ROC curve was first developed by electrical engineers and radar engineers during World War II for detecting enemy objects in battle fields, also known as the signal detection theory, and was soon introduced in psychology to account for perceptual detection of signals

In signal detection theory, a ROC curve, is a graphical plot of the sensitivity vs. $(1 - \text{specificity})$ for a binary classifier system as its discrimination threshold is varied. Sensitivity is the ability to correctly identify the cases who have the condition and Specificity is the ability to correctly identify the cases who do not have the condition.

ROC analysis has been used in medicine, radiology, and other areas for many decades, and it has been introduced relatively recently in other areas like machine learning and data mining.

THE DATA

The data used for the %ROC_CUTOFF macro was randomly generating using SAS. 40 random normal variables with a Mean of 3 and Standard Deviation of 1 were simulated, and another 40 random normal variables with a Mean of 10 and a Standard Deviation of 5 were simulated. In this example patients in $\text{cht} = 1$ are referred to normal patients and patients in $\text{cht} = 2$ are referred to as abnormal.

```
/* Data */  
  
data blood_for_roc;  
CALL STREAMINIT(400);  
do i = 1 to 40;  
Sputum_Eosinophils = RAND('NORMAL',3, 1);  
cht = 1;  
output;  
end;  
do i = 41 to 80;  
Sputum_Eosinophils = RAND('NORMAL',10, 5);  
cht = 2;  
output;  
end;  
run;
```

THE PLOTS AND MACRO CALL

The code below illustrates how the macro call works for the %ROC_CUTOFF macro. The DATASET parameter is the name of the SAS dataset which you're interested in, the OUTCOME parameter is the name of your Categorical Variable, The OUTCOME_LEV parameter is that level of the outcome that defines the result, and should be a numerical value. In this example the outcome level is either 1 or 2. The XVAR parameter are the continuous predictors and the XVAR_LABEL parameter is the label that you wish to assign to your continuous predictor variable. The AUC_TABLE parameter and the CONFUSION parameter gives you the option to display the (Area Under the Curve) AUC Table or the Confusion Matrix on the plot. The CUTOFF parameter gives you the option to specify the particular cut-off you would like to investigate, this should be left blank or a numerical value.

```

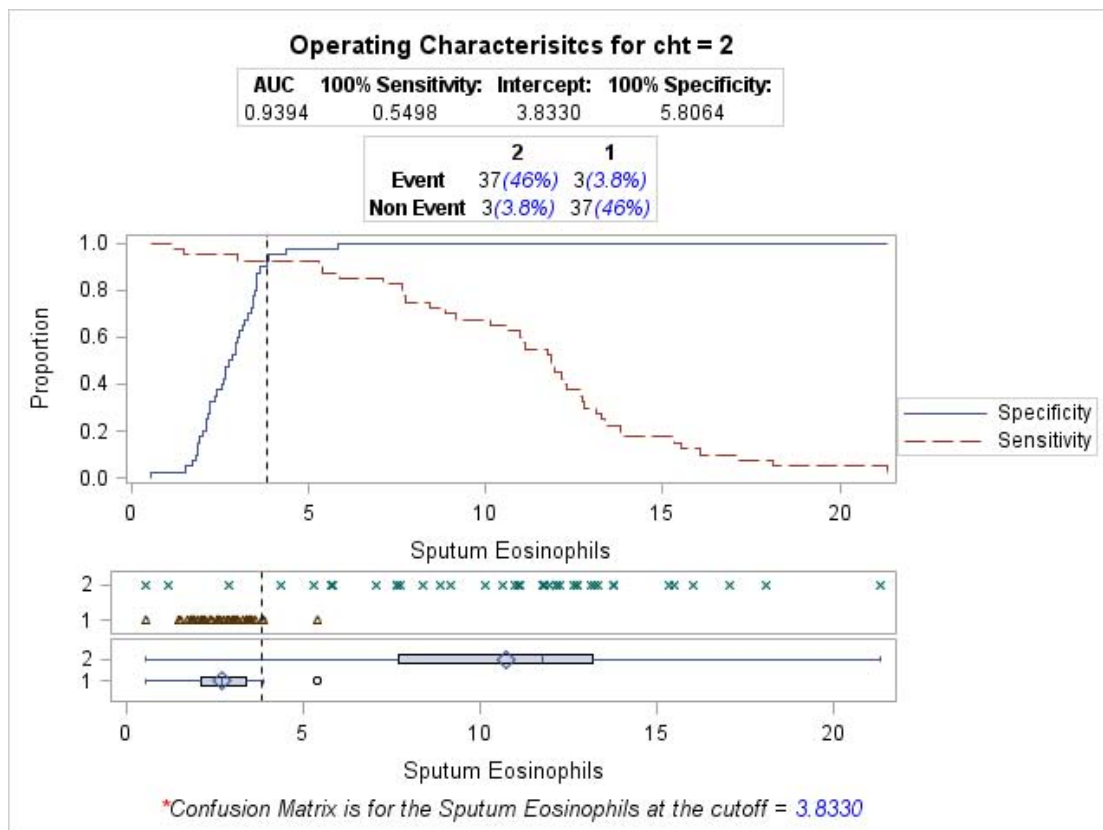
/* Macro */

%macro roc_cutoff(DATASET = blood_for_roc ,
                 OUTCOME = cht ,
                 OUTCOME_LEV = 2 /* Number */ ,
                 XVAR = Sputum_Eosinophils,
                 XVAR_LABEL = Sputum Eosinophils,
                 AUC_TABLE = Y,
                 CONFUSION = N,
                 CUTOFF =
);

```

The macro call above produces the plot below.

Figure 1 GENERATED PLOT WITH OPTIONAL TABLES



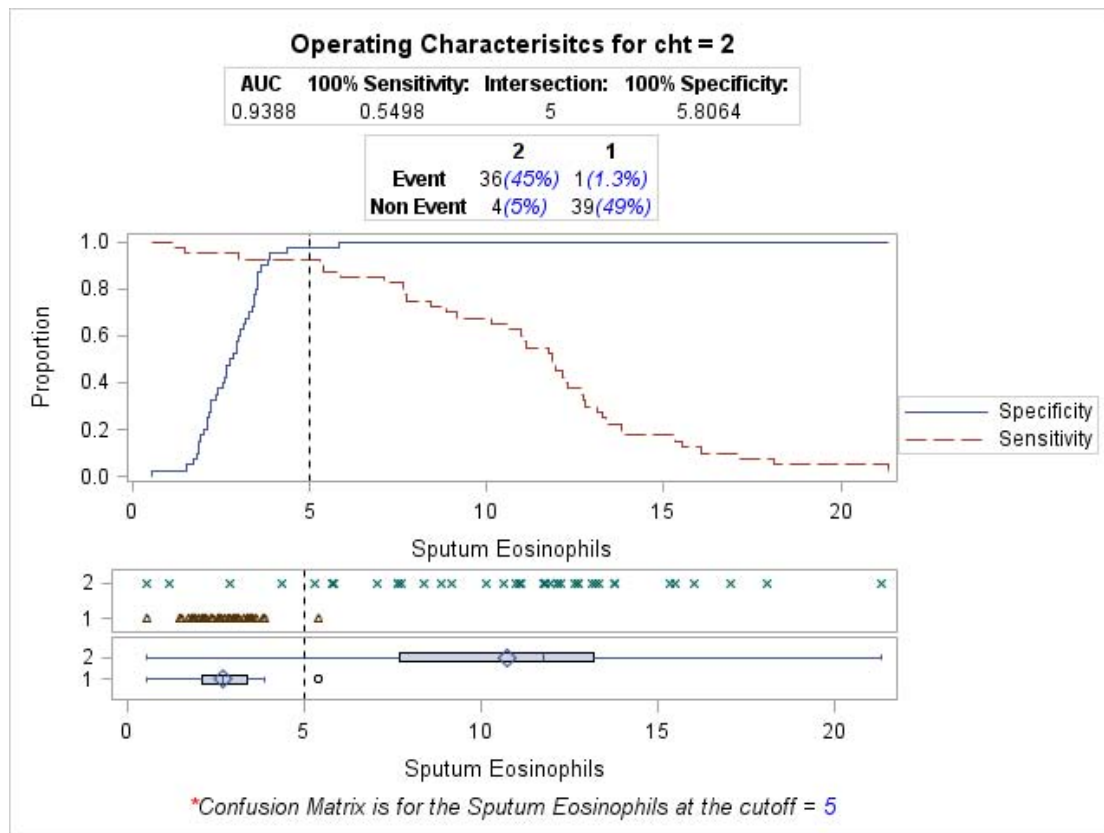
The figure above shows an example of the plot that the %ROC_CUTOFF macro produces. The AUC table and the Confusion Matrix is optional. The AUC result of 94% shows that the cohort's discriminate between the Sputum Eosinophils very well. The number of 0.5498 at 100% Sensitivity means that all of Cohort 2 Sputum Eosinophils are above this number, therefore to achieve 100% Sensitivity for Cohort 2, a cut-off of 0.55 should be used. Conversely value of 5.8064 means that all of Cohort 1 Sputum Eosinophils are below this number therefore to achieve 100% Specificity a value of 5.80 should be used.

The intersection of the Sensitivity and Specificity values are usually considered as a good cut-off to use. In this case it's 3.8 and the Confusion Matrix is based on that value. The Confusion Matrix shows that 37 results in Cohort 2 (abnormal patients) have Sputum Eosinophils higher than the intersection, and 3 results are lower than the intersection. In Cohort 1 (normal patients) 37 results are lower than the intersection while 3 are higher. This breakdown of numbers is very useful. The breakdown of numbers can be changed by entering a specific cut off value in the **cutoff** macro parameter.

The main plot easily shows the Sensitivity and Specificity of the binary classifier at each Sputum Eosinophil cut off. The vertical reference line is the Sensitivity and Specificity intersection. Again this can be changed by entering a specific cut off value in the cutoff macro parameter.

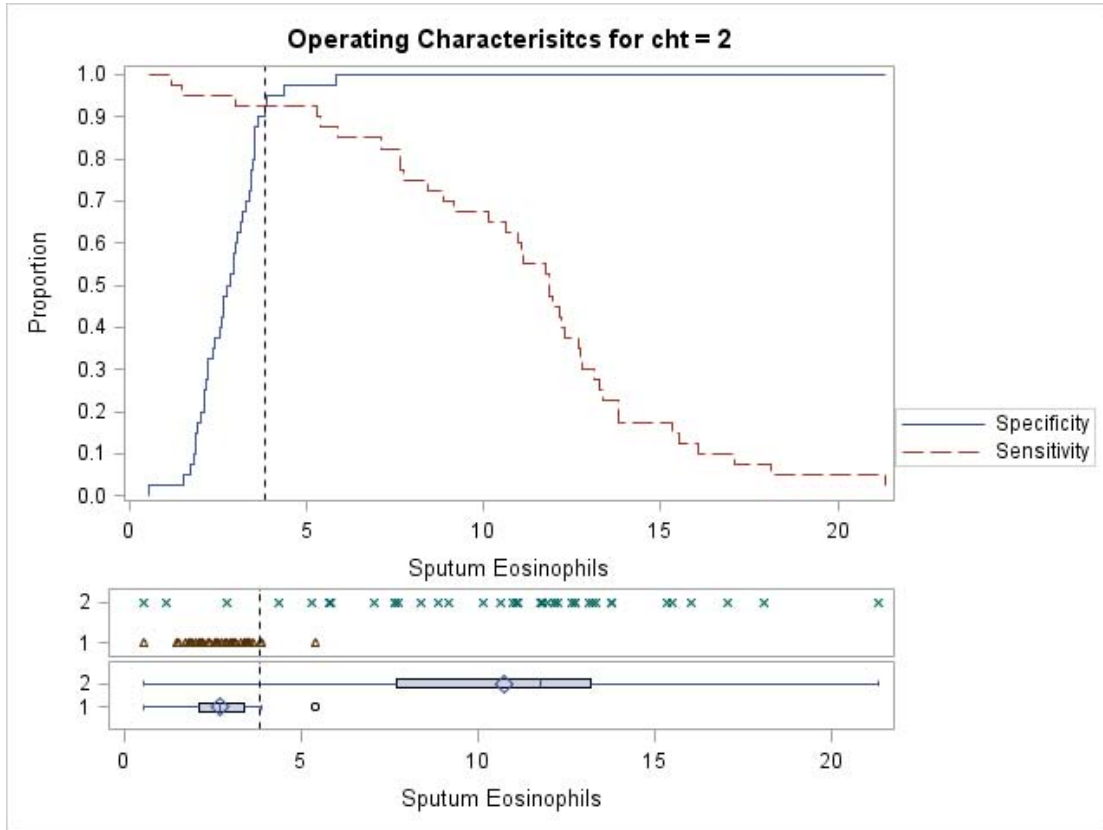
The final 2 plots show a scatterplot and box plot of the Sputum Eosinophils by the Cohorts, which is useful for investigating the raw data.

Figure 2 SPECIFYING CUT-OFF LIMIT



The figure above shows an example of investigating the sensitivity and specificity at a specific cut-off. This can be done by simply entering a numerical value in the cutoff macro parameter. Using a cut-off of 5 the confusion matrix and the Specificity and Sensitivity plot show that the binary classifier predicts the normal patients better than the default cut-off, but predicts the abnormal patients less better than the default cut-off of 3.83.

Figure 3 GENERATED PLOT WITHOUT OPTIONAL TABLES

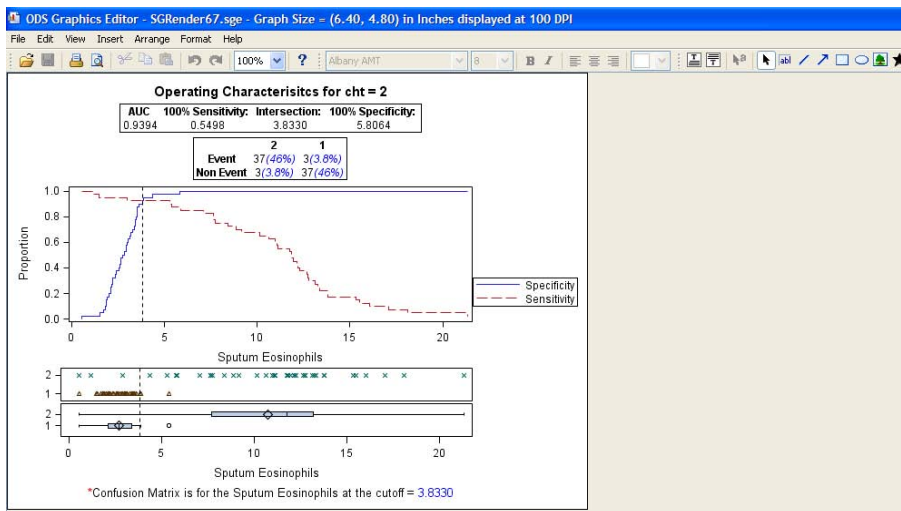


The figure above shows an example of how the plot looks without the AUC or Confusion Matrix Options.

SGE

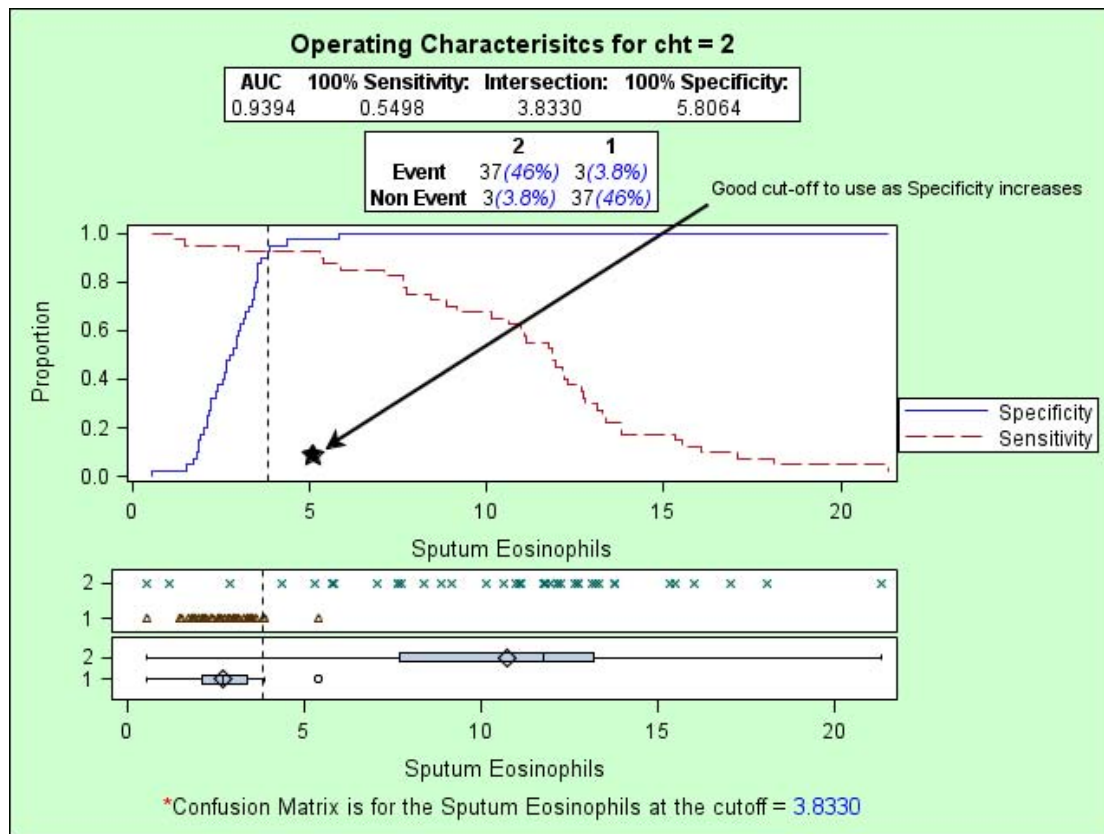
SGE can enable the easy editing of graphs, such as customizing titles and labels, annotating data points, adding text, and changing graph element properties such as fonts, colours, and line styles. Don't worry no datapoints can be edited though, so data integrity is upheld.

Figure 4 OPENING GRAPH IN SGE



To open the Figure with SGE, navigate to the SGE file and double click. A similar picture should be produced to the one above.

Figure 4 EDITING THE GRAPH WITH SGE



CONCLUSION

Graphics Template Language and SGE, combined with ROC analysis can produce a highly effective visualisation tool which helps to choose the appropriate cut off to use.

ACKNOWLEDGEMENTS

I want to thank Andrew Miskell for asking for my help to give a Graphics course in SAS 9.2 and hence giving me the opportunity to delve into the Graphical Features of SAS 9.2.

REFERENCES

Jennifer Lambert, Ilya Lipkovich 2008. "A Macro For Getting More Out Of Your ROC Curve." *Proceedings of the 2008 SAS Global Forum*. Eli Lilly and Company, Indianapolis, IN. Available at <http://www2.sas.com/proceedings/forum2008/231-2008.pdf>

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Please contact the author at:

Kriss Harris
 GlaxoSmithKline
 Mail Stop : 1B106003
 Gunnels Wood Road
 Stevenage SG1 2NY
 Phone: 01438 76 4904
 Email: Kriss.5.Harris@gsk.com
 Web: <http://www.krissharris.co.uk>

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.

APPENDIX

```

/*****
AUTHOR : KRISS HARRIS
AUTHOR EMAIL: KRISS.5.HARRIS@GSK.COM
DATE WRITTEN: JUNE 2009

MACRO NAME : ROC_CUT-OFF
PURPOSE : CREATES A PANEL OF GRAPHS WHICH ASSESS THE SENSITIVITY,
          AND SPECIFICITY OF THE CLASSIFIER, AND THE RAW DATA.
SAS VERSION : VERSION 9.2
PARAMETERS :
-----
NAME TYPE DEFAULT DESCRIPTION AND VALID VALUES
-----
DATASET REQUIRED : SOURCE DATASET
OUTCOME REQUIRED : RESPONSE VARIABLE (CATEGORICAL)
OUTCOME_LEV REQUIRED : EVENT OF INTEREST. A NUMERICAL LEVEL OF THE OUTCOME
PARAMETER THAT DEFINES THE EVENT
(E.G., OUTCOME_LEV=1)
XVAR REQUIRED : CONTINUOUS PREDICTOR

XVAR_LABEL OPTIONAL : LABEL OF PREDICTOR IN GRAPHICAL OUTPUT
AUC_TABLE OPTIONAL Y : Y/N. DISPLAY AREA UNDER CURVE
CONFUSION OPTIONAL Y : Y/N. DISPLAY CONFUSION MATRIX
CUTOFF OPTIONAL Y : NUMERICAL VALUE WHICH CALCULATES THE CONFUSION MATRIX
                     BASED ON THE INPUTTED CUTOFF

*****/

/* Data */

data blood_for_roc;
CALL STREAMINIT(400);
do i = 1 to 40;
Sputum_Eosinophils = RAND('NORMAL',3, 1);
cht = 1;
output;
end;
do i = 41 to 80;
Sputum_Eosinophils = RAND('NORMAL',10, 5);
cht = 2;
output;
end;
run;

%macro roc_cutoff(DATASET = blood_for_roc ,
                  OUTCOME = cht ,
                  OUTCOME_LEV = 2 /* Number */ ,
                  XVAR = Sputum_Eosinophils,
                  XVAR_LABEL = Sputum Eosinophils,
                  AUC_TABLE = Y,
                  CONFUSION = N,
                  CUTOFF =
);

%*****;
%* GETTING ROC OUTPUT FROM MODEL *;
%*****;

%LET OUTCOME_LEV = %SYSFUNC (TRANWRD (&OUTCOME_LEV, %STR(%"), ));
%LET OUTCOME_LEV = %SYSFUNC (TRANWRD (&OUTCOME_LEV, %STR(%'), ));

ods output Association = roc_association (WHERE = (UPCASE(LABEL2) = "C"));

```

```

proc logistic data=&dataset outest = outest;
model &OUTCOME. (event = "&OUTCOME_LEV.") = &XVAR./EXPB outroc=ROCData;
output out = roctest p = p ;
run;

DATA ROCData;
SET ROCData;
INDEX =1;
RUN;

DATA Outest (KEEP=INTERCEPT &XVAR. INDEX);
SET Outest;
INDEX =1;
RUN;

/* Calculating X value */

DATA _merge;
MERGE ROCData Outest;
BY INDEX;
SPEC = 1- 1MSPEC ;
YI = (_SENSIT_ + SPEC) - 1;
N = _POS_ + _NEG_ + _FALPOS_ + _FALNEG_ ;
TA = ( _POS_ + _NEG_ )/N;
X_VALUE = (LOG(_PROB_/(1-_PROB_))-INTERCEPT)/&XVAR.;
IF (_NEG_+_FALNEG_) NE 0 THEN DO;
NPV = _NEG_/(_NEG_+_FALNEG_);
END;
IF (_POS_+_FALPOS_) NE 0 THEN DO;
PPV = _POS_/(_POS_+_FALPOS_);
END;
IF
(( _POS_+_FALPOS_)*(_POS_+_FALNEG_)*(_NEG_+_FALPOS_)*(_NEG_+_FALNEG_)) NE 0
THEN DO;
MCC = (( _POS_*_NEG_)-(_FALPOS*_FALNEG_))/
SQRT((( _POS_+_FALPOS_)*(_POS_+_FALNEG_)*(_NEG_+_FALPOS_)*(_NEG_+_FALNEG_)));
END;
RUN;

/* Merging in the original data, so that this can be plotted to */
proc sql;
create table merge2 as
select a.*, b.&XVAR. as XVAR1, b.&OUTCOME., b.p,
case when b.&OUTCOME. = &OUTCOME_LEV. then 3
else 4 end as Index1
from _merge as a right join Roctest as b
on a._PROB_ = b.p ;
quit;

data merge3;
set merge2;
string check = anyalpha(&OUTCOME.);
if string_check = 0 then &OUTCOME._char = strip(put(&OUTCOME., best.))
;
else &OUTCOME. = &OUTCOME._char;
run;

%*****;
%* CALCULATING X Value CUT-OFFS for Particular Specificities and
Sensitivities and Vice Versa
%*****;

proc sql;
create table One Hundred PC Sens as
select max(X_VALUE) as max_x format 10.2
from merge
where _SENSIT_ = 1;
quit;

proc sql;
create table sens_spec as
select *, abs(_SENSIT_ - SPEC) as abs_sens_spec, min(calculated abs_sens_spec) as min
from _merge;
quit;

```

```

proc sql;
create table sens_spec_intersection as
select X_VALUE format 10.2
from sens_spec
where abs_sens_spec = min;
quit;

proc sql;
create table One_Hundred_PC_Spec as
select min(X_VALUE) as min_x format 10.2
from _merge
where spec = 1;
quit;

/* Fetching the values */

%let dsid=%sysfunc(open(Roc_association,i));
%let num_AUC=%sysfunc(varnum(&dsid,nValue2));
%let rc=%sysfunc(fetch(&dsid,1));
%let AUC=%sysfunc(getvarn(&dsid,&num_AUC));
%let AUC_format=%sysfunc(putn(&AUC,bestd10.2));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(One_Hundred_PC_Sens,i));
%let num_Sens=%sysfunc(varnum(&dsid,max_x));
%let rc=%sysfunc(fetch(&dsid,1));
%let Sens=%sysfunc(getvarn(&dsid,&num_Sens));
%let Sens_format=%sysfunc(putn(&Sens,bestd10.2));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(One_Hundred_PC_Spec,i));
%let num_Spec=%sysfunc(varnum(&dsid,min_x));
%let rc=%sysfunc(fetch(&dsid,1));
%let Spec=%sysfunc(getvarn(&dsid,&num_Spec));
%let Spec_format=%sysfunc(putn(&Spec,bestd10.2));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(sens_spec_intersection,i));
%let num_section=%sysfunc(varnum(&dsid,X_VALUE));
%let rc=%sysfunc(fetch(&dsid,1));
%let section=%sysfunc(getvarn(&dsid,&num_section));
%let rc=%sysfunc(close(&dsid));

%if &CUTOFF = %NRSTR() %then %do;
    %let section2 = &section;
    %let section_format=%sysfunc(putn(&section2,bestd10.2));
    %let rc=%sysfunc(close(&dsid));
%end;
%else %do;
%let section2 = &CUTOFF;
%let section_format = &CUTOFF;
%end;

%*****;
%* 2*2 Tables *;
%*****;

data freq_input;
set merge3;
if XVAR1 ne .;
if XVAR1 <= &section2 then new_result = 1;
else new_result = 0;
run;

ods output CrossTabFreqs = CrossTabFreqs;
proc freq data = freq_input;
tables new_result * &OUTCOME._char / NOROW NOCOL;
run;

data crossTabFreqs_ref;
set CrossTabFreqs;
if &OUTCOME._char ne "";
if new_result ne .;
run;

```



```

data TP;
set CrossTabFreqs ref;
if new_result = 0 and &OUTCOME._char = &OUTCOME_LEV.;
run;

data FP;
set CrossTabFreqs ref;
if new_result = 0 and &OUTCOME._char ne &OUTCOME_LEV.;
run;

data FN;
set CrossTabFreqs ref;
if new_result = 1 and &OUTCOME._char = &OUTCOME_LEV.;
run;

data TN;
set CrossTabFreqs_ref;
if new_result = 1 and &OUTCOME._char ne &OUTCOME_LEV.;
run;

/* Fetching the values */

%let dsid=%sysfunc(open(TP,i));
%let num_TP=%sysfunc(varnum(&dsid,Frequency));
%let rc=%sysfunc(fetch(&dsid,1));
%let TP=%sysfunc(getvarn(&dsid,&num_TP));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(FP,i));
%let num_FP=%sysfunc(varnum(&dsid,Frequency));
%let rc=%sysfunc(fetch(&dsid,1));
%let FP=%sysfunc(getvarn(&dsid,&num_FP));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(FN,i));
%let num_FN=%sysfunc(varnum(&dsid,Frequency));
%let rc=%sysfunc(fetch(&dsid,1));
%let FN=%sysfunc(getvarn(&dsid,&num_FN));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(TN,i));
%let num_TN=%sysfunc(varnum(&dsid,Frequency));
%let rc=%sysfunc(fetch(&dsid,1));
%let TN=%sysfunc(getvarn(&dsid,&num_TN));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(TP,i));
%let num_TP_per=%sysfunc(varnum(&dsid,Percent));
%let rc=%sysfunc(fetch(&dsid,1));
%let TP_per=%sysfunc(getvarn(&dsid,&num_TP_per));
%let TP_format=%sysfunc(putn(&TP_per,best3.0));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(FP,i));
%let num_FP_per=%sysfunc(varnum(&dsid,Percent));
%let rc=%sysfunc(fetch(&dsid,1));
%let FP_per=%sysfunc(getvarn(&dsid,&num_FP_per));
%let FP_format=%sysfunc(putn(&FP_per,best3.0));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(FN,i));
%let num_FN_per=%sysfunc(varnum(&dsid,Percent));
%let rc=%sysfunc(fetch(&dsid,1));
%let FN_per=%sysfunc(getvarn(&dsid,&num_FN_per));
%let FN_format=%sysfunc(putn(&FN_per,best3.0));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(TN,i));
%let num_TN_per=%sysfunc(varnum(&dsid,Percent));
%let rc=%sysfunc(fetch(&dsid,1));
%let TN_per=%sysfunc(getvarn(&dsid,&num_TN_per));
%let TN_format=%sysfunc(putn(&TN_per,best3.0));
%let rc=%sysfunc(close(&dsid));

/* Selecting distinct OUTCOME values */

```

```

proc sql;
create table distinct_values as
select distinct &OUTCOME.
from &dataset;
quit;

data LEVEL NOTLEVEL;
  set distinct_values;
  if &OUTCOME. = UPCASE(&OUTCOME_LEV.) then output LEVEL;
  else OUTPUT NOTLEVEL;
run;

/* Turning it to character if it's not */

data level_char;
set level;
string_check = anyalpha(&OUTCOME.);
if string_check = 0 then &OUTCOME._char2 = strip(put(&OUTCOME., best.))
;
else &OUTCOME. = &OUTCOME._char2;
run;

data notlevel_char;
set notlevel;
string check = anyalpha(&OUTCOME.);
if string_check = 0 then &OUTCOME._char2 = strip(put(&OUTCOME., best.))
;
else &OUTCOME. = &OUTCOME._char2;
run;

/* Fetching the distinct levels */

%let dsid=%sysfunc(open(level_char,i));
%let num_level=%sysfunc(varnum(&dsid,&OUTCOME._char2));
%let rc=%sysfunc(fetch(&dsid,1));
%let level=%sysfunc(getvarc(&dsid,&num_level));
%let rc=%sysfunc(close(&dsid));

%let dsid=%sysfunc(open(notlevel_char,i));
%let num_notlevel=%sysfunc(varnum(&dsid,&OUTCOME._char2));
%let rc=%sysfunc(fetch(&dsid,1));
%let notlevel=%sysfunc(getvarc(&dsid,&num_notlevel));
%let rc=%sysfunc(close(&dsid));

/* Plot */

ods html style = statistical;

proc template;
define statgraph sgplot;
mvar OUTCOME_TITLE OUTCOME_LEV OUTCOME AUC Sens Section2 Specif level notlevel confusion
auc_table
XVAR_LABEL TP FP FN TN TP_per FP_per FN_per TN_per Sensitivity Specificity;
dynamic XVAR1 YVAR1;
begingraph;
EntryTitle "Operating Characterisitcs for " OUTCOME_TITLE " = " OUTCOME_LEV /;

if (confusion = "Y")
EntryFootnote textattrs=(color = red )"*" textattrs=(color = blue )"Confusion Matrix is for the "
XVAR_LABEL " at the cutoff = " textattrs=(color = blue ) Section2 ;
*EntryFootnote "At cutoff, Sensitivity = " textattrs=(color = blue ) Sensitivity " and
Specificity = " textattrs=(color = blue ) Specificity ;
endif;

layout lattice / columns=1 rows=2 rowgutter=2px columngutter=2px
rowweights=(.7 .3 );
columnaxes;
columnaxis / label=XVAR_LABEL;
endcolumnaxes;

sidebar / align=right;

layout gridded / border = false;
layout overlay;

```

```

= 2;
        DiscreteLegend "SERIES" "SERIES1" / location = outside ACROSS = 1 down
        endlayout;
        layout overlay;
        *DiscreteLegend "SERIES3" / location = outside;
        endlayout;

        endlayout;

        endsidebar;

        layout overlay / cycleattrs=true xaxisopts = (Label = XVAR LABEL) yaxisopts=(
Label="Proportion" type=linear linearopts=( tickvaluelist=( 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8
0.9 1 ) viewmin=0 viewmax=1 ) );
        StepPlot X=X_VALUE Y=SPEC / primary=true LegendLabel="Specificity" NAME="SERIES";
        StepPlot X=X_VALUE Y= SENSIT / LegendLabel="Sensitivity" NAME="SERIES1";
        Referenceline x = Section2 / LINEATTRS = (color = black pattern = 2);

        if (confusion = "Y")

            /* 2 by 2 */
            layout gridded / columns=3 rows = 3
            halign = CENTER valign = top border=true
            opaque=true backgroundColor=GraphWalls:color location = outside;

            entry halign=center " " / textattrs=GraphValueText;
            entry halign=center level / textattrs=(WEIGHT=bold) /* Setting the titles to bold */
;
            entry halign=center notlevel / textattrs=(WEIGHT=bold);

            entry halign=center "Event" / textattrs=(WEIGHT=bold) ;
            entry halign=center TP textattrs=(STYLE=ITALIC color = blue ) (" TP_per %") /*
Formatting the percentages */ ;
            entry halign=center FP textattrs=(STYLE=ITALIC color = blue ) (" FP_per %") ;

            entry halign=center "Non Event" / textattrs=(WEIGHT=bold);
            entry halign=center FN textattrs=(STYLE=ITALIC color = blue ) (" FN_per %");
            entry halign=center TN textattrs=(STYLE=ITALIC color = blue ) (" TN_per %");

            endlayout;

            endif;

            if (auc_table = "Y")

                layout gridded / columns=4 rows = 1
                halign = center valign = top border=true
                opaque=true backgroundColor=GraphWalls:color location = outside;
                entry halign=center "AUC " / textattrs=(WEIGHT=bold);
                entry halign=center "100% Sensitivity: " / textattrs=(WEIGHT=bold);
                entry halign=center "Intersection: " / textattrs=(WEIGHT=bold);
                entry halign=center "100% Specificity: " / textattrs=(WEIGHT=bold) ;

                entry halign=center AUC;
                entry halign=center Sens;
                entry halign=center Section2;
                entry halign=center Specif ;

                endlayout;

```

```

endif;

*DiscreteLegend "SERIES" "SERIES1" / location = outside;
endlayout;

    layout lattice / columns=1 rows=2 rowgutter=2px columngutter=2px
                    rowweights=(.5 .5 ) COLUMNNDARANGE = Union;

        columnaxes;
        columnaxis / label=XVAR_LABEL;
    endcolumnaxes;

    layout gridded / AUTOALIGN=NONE columns=3 rows = 3
    halign = right valign = top border=true
    opaque=true backgroundColor=GraphWalls:color location = outside;
    endlayout;

        layout overlay / yaxisopts=( Label=" " );
        Scatterplot x=XVAR1 y = OUTCOME / group = OUTCOME INDEX = Index1 NAME =
"SERIES3";

        Referenceline x = Section2 / LINEATTRS = (color = black pattern = 2);
        endlayout;

        layout overlay / yaxisopts=( Label=" " );
        Boxplot y = XVAR1 x = OUTCOME / ORIENT = horizontal;
        Referenceline x = Section2 / LINEATTRS = (color = black pattern = 2);
        endlayout;

    endlayout;

endlayout;

endgraph;
end;
run;

%let OUTCOME_TITLE = &OUTCOME;
%let OUTCOME_LEV = &OUTCOME_LEV;
%let OUTCOME = &OUTCOME._char;
%let AUC = &AUC_format;
%let Sens = &Sens_format;
%let Section2 = &Section_format;
%let Specif = &Spec_format;
%let level = &level;
%let notlevel = &notlevel;
%let confusion = &confusion;
%let auc_table = &auc_table;
%let XVAR_LABEL = &XVAR_LABEL;
%let TP = &TP;
%let FP = &FP;
%let FN = &FN;
%let TN = &TN;
%let TP_per = &TP_format;
%let FP_per = &FP_format;
%let FN_per = &FN_format;
%let TN_per = &TN_format;

ods listing sge = on;
proc sgrender data=merge3 template=sgplot;
dynamic XVAR1 = "XVAR1" YVAR1 = "YVAR1";
run;
ods listing close;

proc sql;
drop table roc_association, Roctest, ROCData, Outest, _merge, merge2, merge3,
One_Hundred_PC_Sens,
sens_spec, sens_spec_intersection, One_Hundred_PC_Spec, freq_input,
CrossTabFreqs, CrossTabFreqs_ref, TP, FP, FN, TN, distinct_values,
LEVEL, NOTLEVEL, level_char, notlevel_char;
quit;

%mend roc_cutoff;
%roc_cutoff;

```