# Online Appendix 4.A: Operating points from counts table

After opening the software.Rproj file in the software directory corresponding to this chapter, open the file mainOpPtsFromCountsTable.R, the listing for which follows:

### Online Appendix 4.A.1: code listing

# mainOpPtsFromCountsTable.R

rm( list = ls())

library(ggplot2)

source("plotROC.R")

options(digits = 6)

Ktr = array(dim = c(2,5))

# Table 4.2.1

Ktr[1,] <- c(30,19,8,2,1)

Ktr[2,] <- c(5,6,5,12,22)

R <- length(Ktr[1,]) -1

FPF <- array(0,dim = R)

TPF <- array(0,dim = R)

for (r in (R+1):2) {

FPF[(R+2)-r] <-

sum(Ktr[1, r:(R+1)])/sum(Ktr[1,])

TPF[(R+2)-r] <-

sum(Ktr[2, r:(R+1)])/sum(Ktr[2,])

}

cat("FPF =", "\n")

cat(FPF, "\n")

cat("TPF = ", "\n")

cat(TPF, "\n")

mu <- qnorm(.5)+qnorm(.9)

sigma <- 1

Az <- pnorm(mu/sqrt(2))

zeta <- seq(- mu - 2,+ mu + 2,0.01)

fpf <- array(dim = length(zeta))

tpf <- array(dim = length(zeta))

for (i in 1:length(zeta)) {

fpf[i] <- pnorm(-zeta[i])

tpf[i] <- pnorm((mu -zeta[i])/sigma)

}

curveData <- data.frame(FPF = c(1, fpf, 0), TPF = c(1, tpf, 0))

pointsData <- data.frame(FPF = FPF, TPF = TPF)

rocPlot1 <- ggplot(mapping = aes(x = FPF, y = TPF)) +

geom\_line(data = curveData, size = 2) +

geom\_point(data = pointsData, size = 5) +

xlab("FPF")+ ylab("TPF" ) +

theme(axis.title.y = element\_text(size = 25,face="bold"),

axis.title.x = element\_text(size = 30,face="bold")) +

scale\_x\_continuous(expand = c(0, 0)) +

scale\_y\_continuous(expand = c(0, 0))

print(rocPlot1)

cat("uppermost point based estimate of mu = ", mu, "\n")

cat("corresponding estimate of Az = ", Az, "\n")

mu <- 2.17;sigma <- 1.65

Az <- pnorm(mu/sqrt(1+sigma^2))

zeta <- seq(- mu - 2,+ mu + 2,0.01)

fpf <- array(dim = length(zeta))

tpf <- array(dim = length(zeta))

for (i in 1:length(zeta)) {

fpf[i] <- pnorm(-zeta[i])

tpf[i] <- pnorm((mu -zeta[i])/sigma)

}

zeta <- seq(- mu - 2,+ mu + 2,0.01)

fpf <- array(dim = length(zeta))

tpf <- array(dim = length(zeta))

for (i in 1:length(zeta)) {

fpf[i] <- pnorm(-zeta[i])

tpf[i] <- pnorm((mu -zeta[i])/sigma)

}

curveData <- data.frame(FPF = c(1, fpf, 0), TPF = c(1, tpf, 0))

pointsData <- data.frame(FPF = FPF, TPF = TPF)

rocPlot2 <- ggplot(mapping = aes(x = FPF, y = TPF)) +

geom\_line(data = curveData, size = 2) +

geom\_point(data = pointsData, size = 5) +

xlab("FPF") + ylab("TPF" ) +

theme(axis.title.y = element\_text(size = 25,face="bold"),

axis.title.x = element\_text(size = 30,face="bold")) +

scale\_x\_continuous(expand = c(0, 0)) +

scale\_y\_continuous(expand = c(0, 0))

print(rocPlot2)

cat("binormal estimate of Az = ", Az, "\n")

Lines 7-10 define Ktr, a 2 x 5 array, which contains the count data in book Table 4.1. Line 12 defines R, the number of operating points, 4 in this case (note the minus one; the number of operating points is one less than the number of bins). When a particular value is used in several places it is better to define it as a variable, so if a change is made to the definition of the variable the values, wherever else the number is used, will be automatically updated. Lines 14 - 15 initialize, to zeroes, two arrays FPF and TPF of length 4, each. On line 17, (R+1):2 is the sequence of integers, starting from 5 and ending at 2, i.e., 5, 4, 3, 2 (R knows to "step backwards"). The for-loop beginning on line 17 says, effectively, that the statements in the curly brackets ending at line 22 are to be repeated for the following successive values of r: 5, 4, 3 and 2. Lines 18 - 21 implement the cumulating and copies the 4 pairs of values to the appropriate positions in the array variables FPF and TPF. Since this appears a little complicated, let us work through it and in doing so, illustrate a simple way of understanding code in R by literally running it line-by-line. Insert a break point at line 17 and click on Source. Click on Next; the cursor advances to line 18. Observe in the Environment window that r has the value 5 (the trailing L denotes a long variable which can handle very large integer values; Google "long integers" if you need to understand this further). Now, on line 19, carefully select only Ktr[1, r:(R+1)] and click on Run. You should see the following:

### Online Appendix 4.A.2: code snippet

Browse[2]> Ktr[1, r:(R+1)]

[1] 1

Browse[2]>

This is because r:(R+1) is the sequence 5:5, which is 5 (in R the colon is the sequence operator; the more general form is the seq() function, which takes longer to write but gives you more control). So we are printing Ktr[1,5], which happens to be 1 (see line 9). The sum() function on a single variable argument with value 1 gives 1. On the other hand the denominator of this line gives 60 (the sum of the array defined on the right hand side of line 9); instead of error-prone typing, simply select sum(Ktr[1,] and click on Run. You should see:

### Online Appendix 4.A.3: code snippet

Browse[2]> sum(Ktr[1,])

[1] 60

Browse[2]>

The ratio gives the x-coordinate of the lowest operating point (corresponding to the highest threshold), namely 0.01667, which is assigned to FPF[1].

### Online Appendix 4.A.4: code snippet

Browse[2]> sum(Ktr[1, r:(R+1)])/sum(Ktr[1,])

[1] 0.01667

Browse[2]>

Click on Next. The right hand side of line 21 gives 0.44, which is assigned to TPF[1]. Click on Next again to start executing the 2nd iteration of the for-loop (notice the value of r in the Environment panel). On line 19 select only Ktr[1, r:(R+1)] and click on Run. You should see the following:

### Online Appendix 4.A.5: code snippet

Browse[2]> Ktr[1, r:(R+1)]

[1] 2 1

Browse[2]>

You can see it has picked up one more element from the array defined in line 9, so the sum in the numerator will yield 3, which divided by 60 gives the x-coordinate of the next-higher operating point, i.e., 0.05. Keep clicking on Next and use selecting and Run to confirm that the code is correctly calculating the operating points listed in book Table 4.1. If you keep clicking Next, eventually the code will exit the for-loop. Or you can click Continue to force execution of the remaining code.

# Online Appendix 4.B: The rest of the code

Lines 24 – 27 print the values of the FPF and TPF arrays. Line 29 implements book Equation 3.21 to calculate the mu parameter from a pair of specificity and sensitivity values. Line 30 defines sigma <- 1, the equal variance binormal model. Line 31 implements book Equation 3.30 to convert mu to the area Az under the binormal model fitted ROC curve to a single operating point.

Lines 33 – 50 uses ggplot2 to plot the corresponding ROC curve with superimposed operating points. Some day I will get around to explaining this and update this document, or perhaps a user will supply me with the explanation, and be acknowledged in the credits.

Lines 52 – 53 print the values of mu and Az. Line 55 assigns the unequal variance binormal model fitted parameters; book Chapter 6 to mu and sigma. These are obtained by fitting all 4 operating points, not just one operating point as in the previous plot.

Source the code to get the following output and two plots, corresponding to book Figures 4.1 a and b.

> source(…)

FPF =

0.0166667 0.05 0.183333 0.5

TPF =

0.44 0.68 0.78 0.9

uppermost point based estimate of mu = 1.28155

corresponding estimate of Az = 0.817583

binormal estimate of Az = 0.869645