Natura non facit saltus (Nature does not make jumps) – Gottfried Wilhelm Leibniz

Eamonn O'Brien

31 July, 2016

1 'Cutpoints made up by boys in back rooms'

Clinical laboratory procedures are optimised to efficiently process patient samples in order to generate high quality patient information with low sample attrition. Yet the same is often not true in the analysis of laboratory data. Many machine learning techniques that are applied in this setting use improper accuracy scoring rules to dichotomise the output of laboratory processes, effectively throwing away precious patient samples. Optimisation of improper accuracy scoring rules will result in the wrong answer. Many clinicians too prefer 'classifying' patients, dichotomising continuous data and therefore patients, as 'high'/'low', 'positive'/'negative' etc, despite the loss of information and the detrimental impact to the patient. Information loss is greatest with only two groups, but this approach is most common. Cutpoints simply do not exist for continuous data as the cutpoint has to be a function of all other relevant covariates. Dichotomisation of clinical data is a truly disastrous practice, reducing statistical power, concealling non-linear relationships, is a colossal waste of time, money and resources; and yet it persists. The statistican should not be the provider of the utility function, rather it is a job of the statistician to use all information as efficiently as possible and to deliver probabilities. A lack of statistical knowledge at all levels, including senior management is a major contributing factor to this issue.

The following is an example of one problem that does not exist, yet is created by dichotomisation. An arbitrary cutpoint for an assay is used forcing a binary decision such that patients with results either side of the cutpoint are managed differently, perhaps 'positive' results (less than the cutpoint) are recruited on to a trial. A concern is raised regarding the 'misclassification' of patients that result from measurement error.

The quote above is attributed to Frank Harrell Jr.

1.1 Create continuous assay results for a number of reference patients

```r
set.seed(35252)
x <- rnorm(120, -1.81, 1.03)
mosaic::favstats(x)
```

```
      min        Q1    median        Q3       max      mean       sd   n missing
 -4.661345 -2.450643 -1.740088 -1.004394 0.4020143 -1.796988 1.062829 120       0
```

1.2 Convert patient responses to the probability of crossing the cutpoint due to normally distributed measurement error

```
threshold <- quantile(x, 0.25)[[1]]   # diff. patient management decisions made dependent on cutpoint
noise <- 0.3                          # test method measurement error
n <- length(x)

# convert the assay results to probabilities of flipping over the cutpoint
probx <- ifelse( x<threshold, pnorm(x, threshold, sd = noise),
                            1-pnorm(x, threshold, sd = noise))
```

1.3 Number of samples with 25% or greater chance of changing call

```
  binom::binom.confint( length(probx[probx>=.25]) , n, methods="wilson")
```

```
  method  x   n      mean      lower     upper
1 wilson 14 120 0.1166667 0.07078118 0.1863335
```

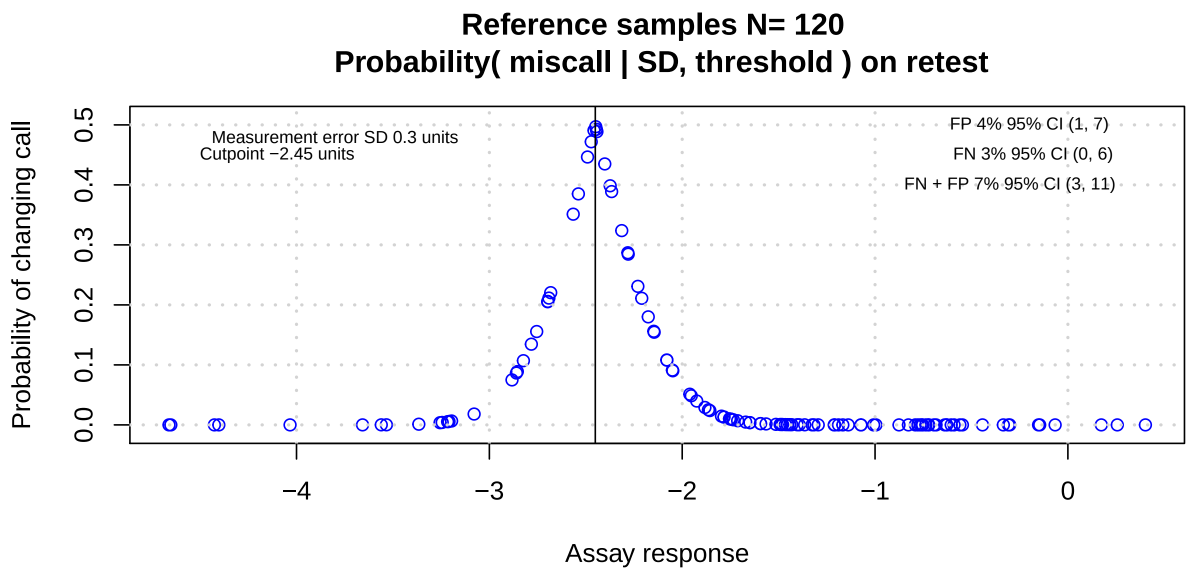
1.4 Probability distribution for different probabilities

```
miscall <- function(p) {

  Ex <- sum(p)             # expectation of the sum of the reference samples
  var <- sum(p*(1-p))      # because the samples are independent this is the variance
  print(c(Ex, Ex + c(-1, +1)*qnorm(.975)*var^0.5))  # the distribution of sums is ~N(Ex, var)

}

# counts and probabilites
tot <- miscall(probx) / n                # FN + FP
FN <-  miscall(probx[x<threshold])  / n  # FN
FP <-  miscall(probx[x>threshold])  / n  # FP
```



1.5 FN + FP counts and probabilities

```
[1]  8.932572  4.123081 13.742062
```

```
[1] 0.07443810 0.03435901 0.11451718
```

1.6 FN counts and probabilities

```
[1] 3.6803132 0.5857192 6.7749071
```

```
[1] 0.030669277 0.004880993 0.056457560
```

1.7 FP counts and probabilities

```
[1] 5.252258 1.570589 8.933927
```

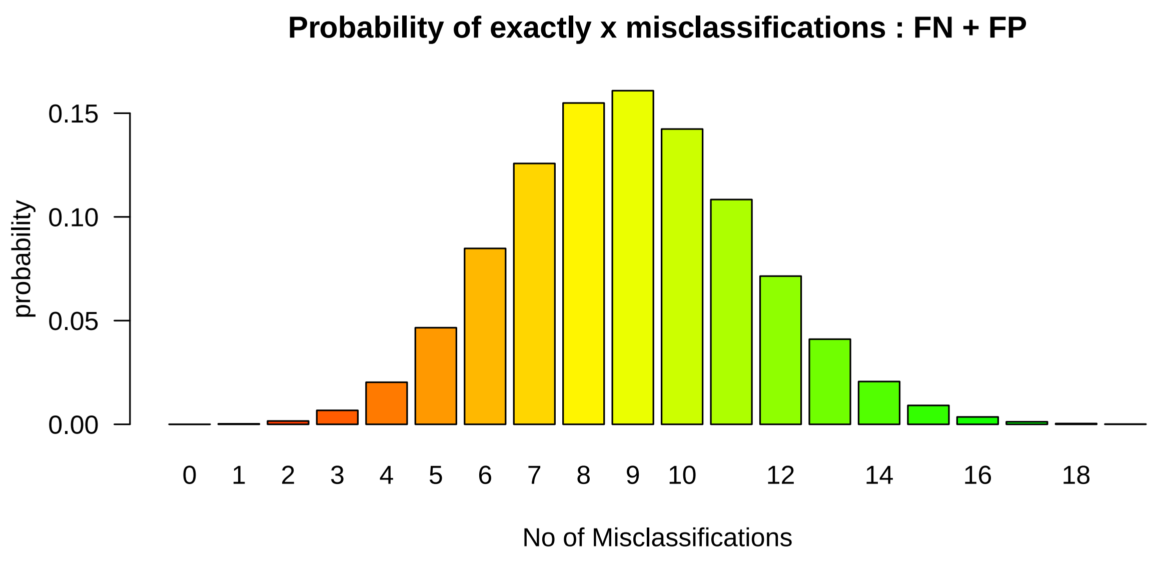
```
[1] 0.04376882 0.01308824 0.07444940
```

2 Convolution of probabilities

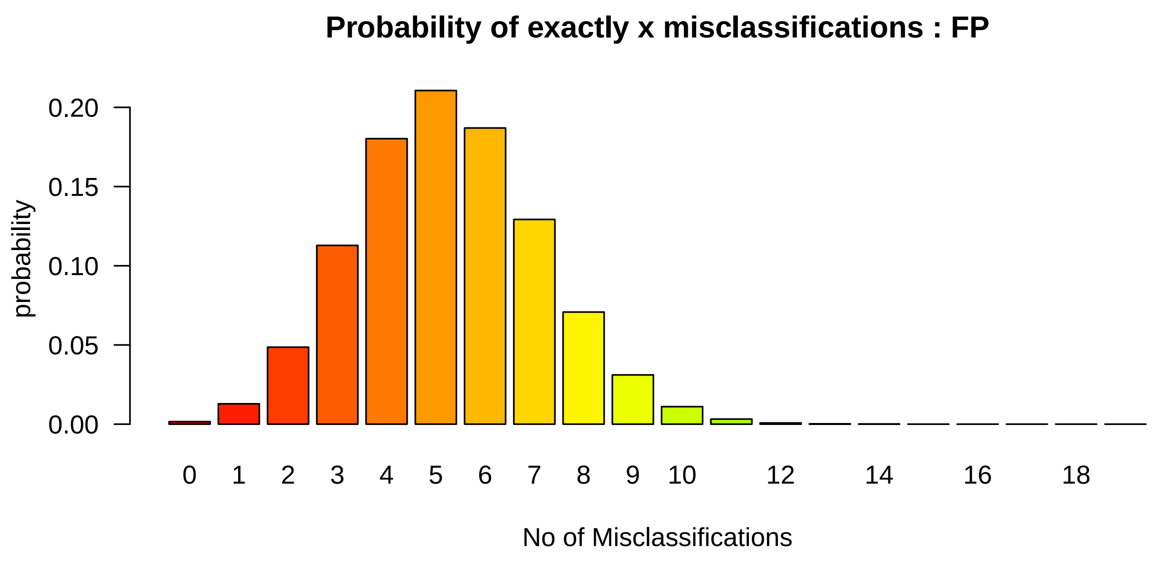
```
convolve.binomial <- function(p) {
  # p is a vector of probabilities of Bernoulli distributions.
  # The convolution of these distributions is returned as a vector
  # `z` where z[i] is the probability of i-1, i=1, 2, ..., length(p)+1.
  n <- length(p) + 1
  z <- c(1, rep(0, n-1))
  sapply(p, function(q) {z <<- (1-q)*z + q*(c(0, z[-n])); q})
  z
}


convolve <- function (p, user.text) {
  r<-convolve.binomial(p)
  r[1:20]
  ##cumulative
  x1<-r[1:20]
  names(x1) <- c(0:19)
  barplot(x1, las=1, main =paste("Probability of exactly x misclassifications :",user.text, sep=" "),
          xlab="No of Misclassifications", ylab = "probability", col=rainbow(50))
}
```

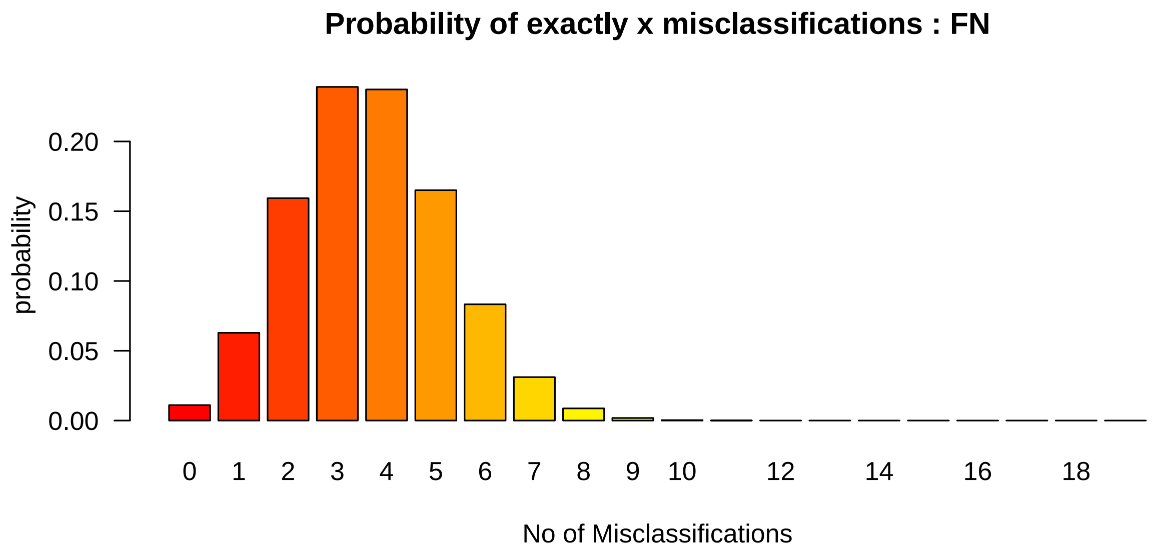
```r
convolve(probx, user.text = "FN + FP")
```



```r
convolve(probx[x > threshold], user.text = "FP")
```



```r
convolve(probx[x < threshold], user.text = "FN")
```



Probability of exactly x misclassifications : FN
probability
0.00
0.05
0.10
0.15
0.20
0
1
2
3
4
5
6
7
8
9
10
12
14
16
18
No of Misclassifications

3 Simulation approach (an inefficient approximation to probability theory in this simple example).

```r
callum <- function(mdata=x, threshold,
                   noise, bias,
                   seed=NULL){

  if (!is.null(seed)) set.seed(seed)

      n <- length(mdata)

      # split the sample and see what happens to them
      below <- mdata[mdata < threshold]
      above <- mdata[mdata > threshold]

      # analytical noise
      analytical1 <- rnorm(length(below), 0, noise)
      analytical2 <- rnorm(length(above), 0, noise)

      # add the noise seperately and any bias
      a1 <- analytical1 + below + bias
      a2 <- analytical2 + above + bias

      prop <- length(below)/n

      # less than threshold are considered 'responders' - if you believe that
      (TP <- sum(a1 < threshold)/n)
      (FP <- sum(a2 < threshold)/n)
      (FN <- sum(a1 > threshold)/n)
      (TN <- sum(a2 > threshold)/n)
      FPFN <- FP+FN

        c(TN , FN, FP, FPFN, TP, prop, 1-prop)*100

      }

      res <- replicate(1e5,
                  callum(mdata=x,
                             threshold=threshold, noise=noise, bias=0, seed=NULL))

      multi.fun <- function(x) {
            c(mean = mean(x), ci= quantile(x, c(0.025, 0.975)))
      }

      res1 <- apply(res, 1, multi.fun)
      res1 <- as.data.frame(res1)
      names(res1) <- c("TN", "FN", "FP", "FN+FP", "TP", "Pos", "Neg")



      res1
```

```
               TN        FN       FP     FN+FP       TP Pos Neg
mean     70.62713 3.0662583 4.372875  7.439133 21.93374  25  75
ci.2.5%  67.50000 0.8333333 1.666667  3.333333 19.16667  25  75
ci.97.5% 73.33333 5.8333333 7.500000 11.666667 24.16667  25  75
```

3.1 Print probability theory results again to cross check

3.1.1 The first row below are the expected counts, the second row corresponds to the FN column above

```
miscall(probx[x < threshold]) / n  # FN
```

```
[1] 3.6803132 0.5857192 6.7749071
```

```
[1] 0.030669277 0.004880993 0.056457560
```

3.1.2 The second row below corresponds to the FP column above

```
miscall(probx[x > threshold]) / n  # FP
```

```
[1] 5.252258 1.570589 8.933927
```

```
[1] 0.04376882 0.01308824 0.07444940
```

3.1.3 The second row below corresponds to the FN+FP column above

```
miscall(probx) / n                 # FN + FP
```

```
[1]  8.932572  4.123081 13.742062
```

```
[1] 0.07443810 0.03435901 0.11451718
```

3.2 Reference

http://stats.stackexchange.com/questions/41247/risk-of-extinction-of-schr%c3%b6dingers-cats/41263#41263

http://stats.stackexchange.com/questions/9510/probability-distribution-for-different-probabilities

https://www.youtube.com/watch?v=uULhuuSjBww

https://www.youtube.com/watch?v=twnqtGCCTVE

The function ‘callum’ is named after Professor Callum George Fraser

3.2.0.0.1 Computing Environment

```
R version 3.2.2 (2015-08-14)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 8 x64 (build 9200)

locale:
[1] LC_COLLATE=English_United Kingdom.1252
[2] LC_CTYPE=English_United Kingdom.1252
[3] LC_MONETARY=English_United Kingdom.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United Kingdom.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

other attached packages:
[1] xlsx_0.5.7     xlsxjars_0.6.1 rJava_0.9-8    reshape_0.8.5
[5] knitr_1.13

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.6       magrittr_1.5      MASS_7.3-45
 [4] splines_3.2.2     munsell_0.4.3     lattice_0.20-33
 [7] colorspace_1.2-6  R6_2.1.2          ggdendro_0.1-20
[10] mosaicData_0.14.0 stringr_1.0.0     plyr_1.8.4
[13] dplyr_0.5.0       mosaic_0.14.4     tools_3.2.2
[16] grid_3.2.2        binom_1.1-1       gtable_0.2.0
[19] DBI_0.4-1         htmltools_0.3.5   lazyeval_0.2.0
[22] yaml_2.1.13       digest_0.6.9      assertthat_0.1
[25] tibble_1.1        Matrix_1.2-2      gridExtra_2.2.1
[28] tidyr_0.5.1       ggplot2_2.1.0     formatR_1.4
[31] evaluate_0.9      rmarkdown_1.0     stringi_1.1.1
[34] scales_0.4.0

[1] "~/X/RPUBS"
```

This took 9.26 seconds to execute.