Feature Selection Bias in Classification of High Dimensional Data

John H Maindonald

March 10, 2025

Abstract

This vignette reproduces modified versions the calculations and the figures that appear in Section 12.3 of: Maindonald, J.H. and Braun, W.J. *Data Analysis and Graphics Using R.* 2^{nd} edition 2007; 3^{rd} edition 2010, Cambridge University Press. In order to reduce execution time, there are fewer folds for cross-validation and fewer simulations.

1 What groups are of interest?

The data frame `golubInfo` has details of the classifying factors for the 72 columns of the data set `golub`. The 72 observations are classified into one of the three cancer types ALL B-type (coded `allB`), ALL T-type (coded `allT`) and AML (coded `aml`). The two classifications that will be investigated are (1) according to tissue type and sex, given by the factor `tissue.mf`, and (2) according to cancer type (ALL B-type, ALL T-type, AML), given by the factor `cancer`.

```
library(hddplot)
data(golubInfo)
with(golubInfo, table(cancer, tissue.mf))

      tissue.mf
cancer BM:NA BM:f BM:m PB:NA PB:f PB:m
  allB     4   19   10     2    1    2
  allT     0    0    8     0    0    1
  aml     16    2    3     1    1    2
```

For the classification according to tissue type and sex (`tissue.mf`), restriction to the `allB` leukemia type and to patients whose sex is known gives a relatively homogeneous set of data. We will define a factor `tissue.mfB` that classifies the `allB` subset of the data for which the sex of the patient is known, and for which at least two samples are available. The single `allB` observation that is `PB:f` will be omitted.

The following calculations separate out the allB subset (GolubB) of the data, and derive the factor tissue.mfB whose levels are BM:f, BM:m and PB:m:

```
attach(golubInfo)
## Identify allB samples for that are BM:f or BM:m or PB:m
subsetB <- cancer=="allB" & tissue.mf%in%c("BM:f","BM:m","PB:m")
## Form vector that identifies these as BM:f or BM:m or PB:m
tissue.mfB <- tissue.mf[subsetB, drop=TRUE]
## Separate off the relevant columns of the matrix Golub
data(Golub)          # NB: variables (rows) by cases (columns)
GolubB <- Golub[,subsetB]
detach(golubInfo)
```

Distributions across features for a selection of observations

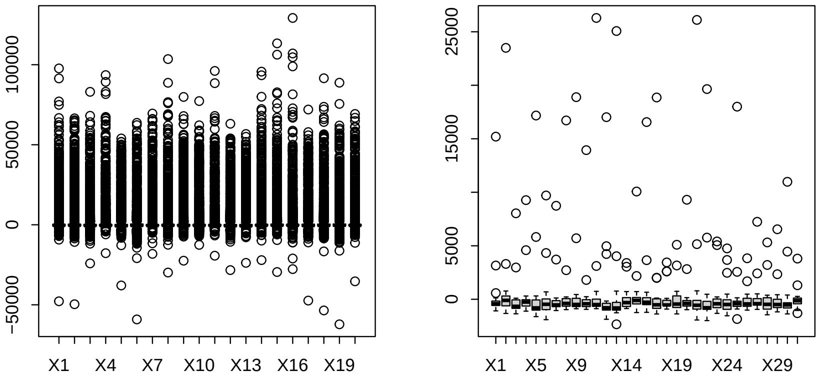

Figure 1: Boxplots of distributions across features for a selection of observations

```
## Display distributions for the first 20 observations
boxplot(data.frame(GolubB[, 1:20]))  # First 20 columns (observations)
## Random selection of 20 rows (features)
boxplot(data.frame(GolubB[sample(1:7129, 20), ]))
```

2 Linear Discriminant Analysis, following variable selection

Flawed graphs

Panel A in Figure 2 shows a convincing separation between groups, based on the use of linear discriminant analysis with the 15 features that best separate the groups. Panel B, which uses random normal data to repeat the calculations for Panel A, highlights the flaws in the methodology.

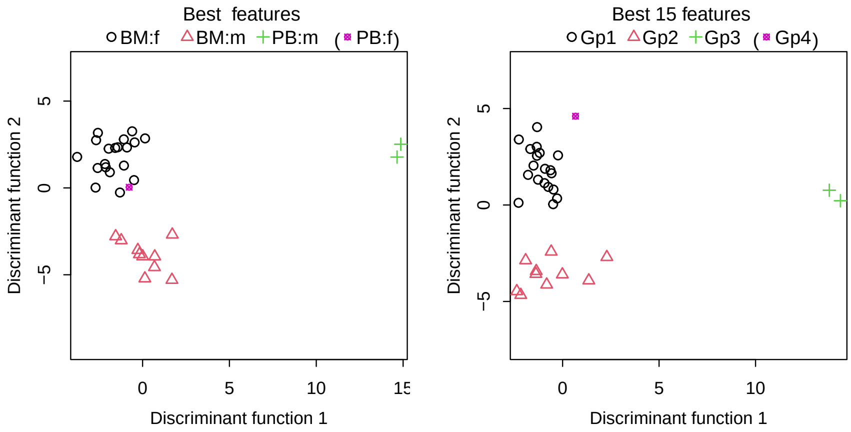

Figure 2: Panel B, with random normal data, illustrates the potential for getting spurious results with the methodology used for Panel A.

```r
## Uses orderFeatures() (hddplot); see below
ord15 <- orderFeatures(GolubB, cl=tissue.mfB)[1:15]
## Panel A
dfB.ord <- data.frame(t(GolubB[ord15, ]))
## Calculations for the left panel
## Transpose to observations by features
dfB15 <- data.frame(t(GolubB[ord15, ]))
library(MASS)
dfB15.lda <-  lda(dfB15, grouping=tissue.mfB)
scores <- predict(dfB15.lda, dimen=2)$x
## Scores for the single PB:f observation
df.PBf <- with(golubInfo,
  data.frame(t(Golub[ord15, tissue.mf=="PB:f" & cancer=="allB",
                     drop=FALSE])))
scores.PBf <- predict(dfB15.lda, newdata=df.PBf, dimen=2)$x
## For comparison: simulated scores
```

```
simscores <- simulateScores(nrow=7129, cl=rep(1:3, c(19,10,2)),
                            cl.other=4, nfeatures=15, seed=41)
  # Returns list elements: scores, cl, scores.other & cl.other
```

```
opar <- par(mar=c(4,4,2.6,.1))
## Warning! The plot that now follows may be misleading!
## Use scoreplot(), from the hddplot package
scoreplot(list(scores=scores, cl=tissue.mfB, other=scores.PBf,
               cl.other="PB:f"))
## Panel B: Repeat plot, now with random normal data
scoreplot(simscores)
par(opar)
```

3 Distributional extremes

Calculated F-statistics (Figure 3) will be compared with the permutation distribution and with the theoretical F-distribution, but limiting attention to just the first two classes. Code is:

```
## In the following, B is too small for the simulation to
## give a good indication of behavior in the extreme tail.
## Code should be re-run with B >> 1000.
library(multtest, quietly=TRUE)
cl.mfB <- unname(unclass(tissue.mfB)-1)
  ## NB: Use unnamed integer vector to identify classes,
  ## here with values running from 0 to 2
GolubB.maxT <- mt.maxT(GolubB, cl.mfB, test="f", B=1000)
```

```
## Compare calculated F-statistics with permutation distribution
qqthin(qf(1-GolubB.maxT$rawp, 2, 28), GolubB.maxT$teststat,
       print.thinning.details = FALSE)
## Compare calculated F-statistics with theoretical F-distribution
qqthin(qf(ppoints(7129), 2, 28), GolubB.maxT$teststat,
       print.thinning.details = FALSE)
  # The theoretical F-distribution gives estimates of quantiles
  # that are too small
## NB also the comparison between the permutation distribution
## and the theoretical F:
qqthin(qf(ppoints(7129), 2, 28), qf(1-GolubB.maxT$rawp, 2, 28),
       print.thinning.details = FALSE)
  # qqthin() is a version of qqplot() that thins out points where
  # overlap is substantial, thus giving smaller graphics files.
```

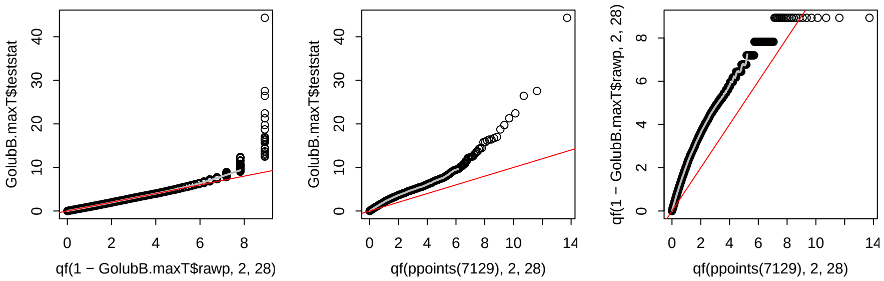

Figure 3: Compare calculated F-statistics with the permutation distribution and with the theoretical F. The theoretical F makes unrealistic normality and independence assumptions.

```
## In the following, B is too small for the simulation to
## give a good indication of behavior in the extreme tail.
## Code should be re-run with B >> 1000.
library(multtest, quietly=TRUE)
cl.mfB <- unname(unclass(tissue.mfB)-1)
  ## NB: Use unnamed integer vector to identify classes,
  ## here with values running from 0 to 2
GolubB.maxT <- mt.maxT(GolubB, cl.mfB, test="f", B=1000)
```

```
## Compare calculated F-statistics with permutation distribution
qqthin(qf(1-GolubB.maxT$rawp, 2, 28), GolubB.maxT$teststat,
       print.thinning.details = FALSE)
## Compare calculated F-statistics with theoretical F-distribution
qqthin(qf(ppoints(7129), 2, 28), GolubB.maxT$teststat,
       print.thinning.details = FALSE)
  # The theoretical F-distribution gives estimates of quantiles
  # that are too small
## NB also the comparison between the permutation distribution
## and the theoretical F:
qqthin(qf(ppoints(7129), 2, 28), qf(1-GolubB.maxT$rawp, 2, 28),
       print.thinning.details = FALSE)
  # qqthin() is a version of qqplot() that thins out points where
  # overlap is substantial, thus giving smaller graphics files.
```

4 Discriminant Analysis – Training/Test

```
##              Selection of features that discriminate
## ss 12.3.3: Accuracies and Scores for test data
Golub.BM <- with(golubInfo, Golub[, BM.PB=="BM"])
cancer.BM <- with(golubInfo, cancer[BM.PB=="BM"])
## Now split each of the cancer.BM categories between two subsets
##  Uses divideUp(), from hddplot
gp.id <- divideUp(cancer.BM, nset=2, seed=29)
  # Set seed to allow exact reproduction of the results below
table(gp.id, cancer.BM)

     cancer.BM
gp.id allB allT aml
    1   17    4  10
    2   16    4  11
```

```
accboth <- accTrainTest(x = Golub.BM, cl = cancer.BM,
                        traintest=gp.id, , print.progress=FALSE)



Training/test split          Best accuracy, less 1SD
I (training) / II (test)     0.89 (14 features)
II (training) / I (test)     0.92 (10 features)

Training/test split          Best accuracy
I (training) / II (test)     0.94 (20 features)
II (training) / I (test)     0.97 (17 features)
```

Code for plotting the figures is:

```
opar <- par(mar=c(4,4,3.1,.1))
## Use function plotTrainTest() from hddplot
plotTrainTest(x=Golub.BM, nfeatures=c(14,10), cl=cancer.BM, traintest=gp.id)
par(opar)
```

Now compare the choice of features between I/II and II/I:

```
rbind(accboth$sub1.2[1:20],accboth$sub2.1[1:20])

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
[1,] 4050 2794 6510 6696 4342 5542 4357 5543 1207  4584  6236  1429
[2,] 6606 4342 6510 3594 4050 6236 1694 1207 1268  4847  5542  2061
     [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20]
[1,]  6575  2833  4750  2335  1704  4882  6225  3544
[2,]  5543  4055  4375  1144   379  6696  4196   229
```

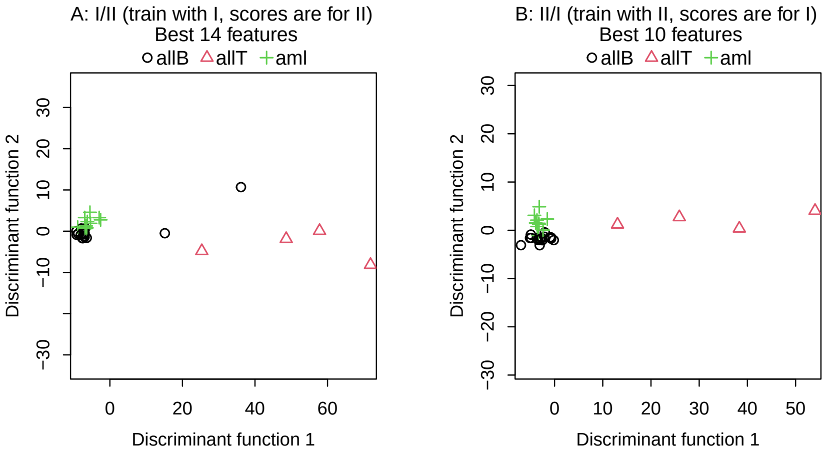

Figure 4: Panel A plots scores for the set II data, using set I for training (the I/II split), as described in the text. Panel B plots the scores for the set I data when the roles of the two sets were reversed, i.e., the split was II/I.

```
match(accboth$sub1.2[1:20],accboth$sub2.1[1:20])

 [1]  5 NA  3 18  2 11 NA 13  8 NA  6 NA NA NA NA NA NA NA NA NA
```

4.1 Cross-validation based optimum choice of features

With the number of selected features varying from 1 to 25, three different accuracy measures will be compared, for classification of the B-cell data. The plots highlight the serious bias in measures that are to an extent internal to the training data.

```
##  Cross-validation to determine the optimum number of features
## Accuracy measure will be: tissue.mfB.cv£acc.cv
tissue.mfB.cv <- cvdisc(GolubB, cl=tissue.mfB, nfeatures=1:23,
                        nfold=c(5,1), print.progress=FALSE)



Accuracy           Best accuracy, less 1SD   Best accuracy
(Cross-validation) 0.77 (3 features)         0.84 (7 features)

  # 5-fold CV (x1)
## Defective measures will be in acc.resub (resubstitution)
## and acc.sel1 (select features prior to cross-validation)
tissue.mfB.badcv <- defectiveCVdisc(GolubB, cl=tissue.mfB,
                                   foldids=tissue.mfB.cv$folds,
                                   nfeatures=1:23, nfold=c(5,1),
                                   print.progress=FALSE)
## NB: Warning messages have been omitted
```

```
## Calculations for random normal data:
set.seed(43)
rGolubB <- matrix(rnorm(prod(dim(GolubB))), nrow=dim(GolubB)[1])
rtissue.mfB.cv <- cvdisc(rGolubB, cl=tissue.mfB, nfeatures=1:23,
                         nfold=c(5,1), print.progress=FALSE)

[1] "Input rows (features) are not named. Names"
[1] "1:7129 will be assigned."


Accuracy           Best accuracy, less 1SD   Best accuracy
(Cross-validation) 0.46 (1 features)         0.55 (1 features)

rtissue.mfB.badcv <- defectiveCVdisc(rGolubB, cl=tissue.mfB,
                                   nfeatures=1:23, nfold=c(5,1),
                                   foldids=rtissue.mfB.cv$folds,
                                   print.progress=FALSE)

[1] "Input rows (features) are not named. Names"
[1] "1:7129 will be assigned."
```

```
## This function will be used for the plots
plot.acc <- function(cv=cv1, badcv=badcv1, nseq=NULL, badnseq=NULL,
                     title="", ylab="Predictive accuracy",
                     add.legend=TRUE){
  maxg <- min(c(length(badcv$acc.resub), length(cv$acc.cv)))
  if(is.null(nseq))nseq <- 1:maxg
  plot(nseq, badcv$acc.resub[1:maxg], ylim=c(0,1), type="n",
       yaxs="i", xlab="Number of features selected", ylab=ylab)
  par(xpd=T)
  points(nseq, badcv$acc.resub[1:maxg], col=2, type="b", lty=2,
         pch=0, cex=0.8)
  par(xpd=FALSE)
  points(nseq, badcv$acc.sel1[1:maxg], col="gray40", pch=3, cex=0.8)
  lines(lowess(nseq, badcv$acc.sel1[1:maxg], f=.325, iter=0),
        col="gray40", lty=2)
  points(nseq, cv$acc.cv[1:maxg], col="blue", pch=1, cex=0.8)
  lines(lowess(nseq, cv$acc.cv[1:maxg], f=.325, iter=0), col="blue",
        lwd=2)
  xy <- par()$usr[c(1,3)]
  if(add.legend)
    legend(xy[1], xy[2], xjust=0, yjust=0,
           legend=c("Training set 'accuracy'",
             "Defective cross-validation",
             "Cross-validation - select at each fold"),
           lty=c(1,2,1), lwd=c(1,1,2), pch=c(0,3,1),
           col=c("red","gray40","blue"), cex=0.875)
  mtext(side=3,line=0.35, title, adj=0)
}
```

```
plot.acc(tissue.mfB.cv, tissue.mfB.badcv,
         title="A: Golub data (as for Figure 12.9)")
plot.acc(rtissue.mfB.cv, rtissue.mfB.badcv, ylab="",
         title="B: Random data", add.legend=FALSE)
```

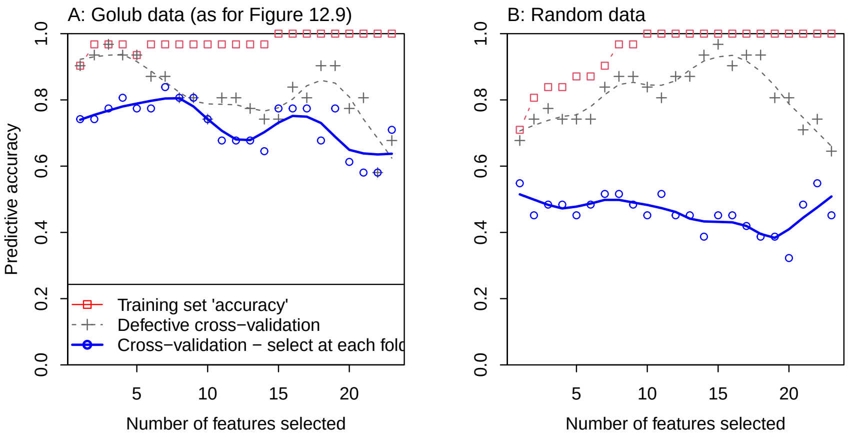

Figure 5: Comparison of different accuracy measures, in the development of a discriminant rule for the classification, into the categories BM:f, BM:m and PB:m, of the B-cell ALL data for which gender is known.

Figure 5 compares three different accuracy measures, for the classification of the B-cell data. The training data measure (□) is a severely biased measure. Cross-validation, but with features selected using all the data (+), is less severely biased. An acceptable measure of predictive accuracy (○) requires re-selection of features at each fold of the cross-validation. The right panel shows the performance of each of these measures when the expression values were replaced by random data.

Which features?

```
##                          Which features?
genelist <- matrix(tissue.mfB.cv$genelist[1:3, ,], nrow=3)
tab <- table(genelist, row(genelist))
ord <- order(tab[,1], tab[,2], decreasing=TRUE)
tab[ord,]


genelist         1 2 3
  M58459_at      3 0 1
  L08666_at      1 0 0
  U29195_at      1 0 0
  U91327_at      0 2 0
  U49395_at      0 1 0
  X00437_s_at    0 1 0
  X54870_at      0 1 0
  X62654_rna1_at 0 0 3
  X82494_at      0 0 1
```

5 Cross-validation: bone marrow (BM) samples

```
##         Cross-validation: bone marrow ({BM}) samples only
BMonly.cv <- cvdisc(Golub.BM, cl=cancer.BM, nfeatures=1:25,
                    nfold=c(5,1), print.progress=FALSE)



Accuracy           Best accuracy, less 1SD   Best accuracy
(Cross-validation) 0.87 (12 features)        0.9 (16 features)

tissue.mfB.scores <-
  cvscores(cvlist = tissue.mfB.cv, nfeatures = 3, cl.other = NULL,
           print.progress=FALSE)


1:2:3:4:5

BMonly.scores <- cvscores(cvlist=BMonly.cv, nfeatures=19,
                          cl.other=NULL, print.progress=FALSE)


1:2:3:4:5
```

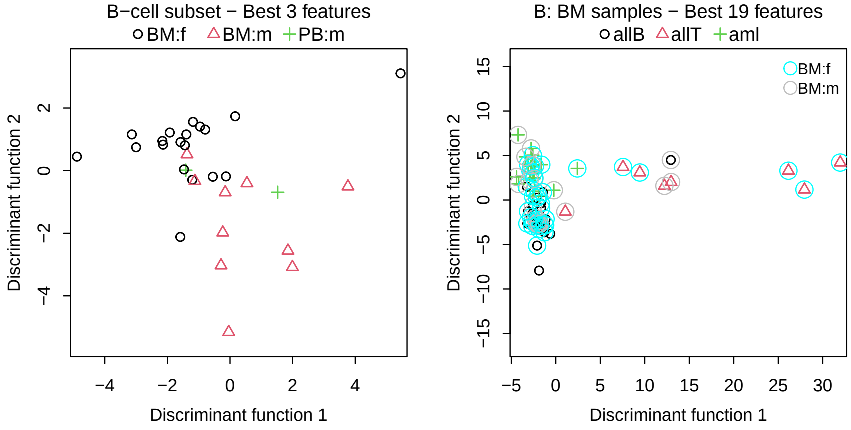

Figure 6: These plots of projections of linear discriminant analysis scores are designed to fairly reflect the performance of a linear discriminant in distinguishing between known groups in the data. The two panels relate to different subsets of the `Golub` data, with different groupings in the two cases. In panel B, for the classification of the 62 bone marrow (`BM`) samples into `allB`, `allT`, and `aml`, points where the sex is known are identified as male or female.

Code is:

```
opar <- par(mar=c(4,4,2.6,.1))
## Panel A: Uses tissue.mfB.acc from above
scoreplot(scorelist = tissue.mfB.scores, cl.circle=NULL,
          prefix="B-cell subset -")
## Panel B; classify bone marrow samples a/c cancer type.
scoreplot(scorelist=BMonly.scores, cl.circle=tissue.mfB,
          circle=tissue.mfB%in%c("BM:f","BM:m"),
          params=list(circle=list(col=c("cyan","gray"))),
          prefix="B: BM samples -")
par(opar)
```