Analysis of SRBCT Data

Requires "sda" in version 1.3.2 (January 2014) or later

Load "sda" package and create SRBCT data set

```
library("sda")
```

```
## Loading required package: entropy
## Loading required package: corpcor
## Loading required package: fdrtool
```

Load data set from Khan et al. (2001):

```
data(khan2001)
```

Create data set containing only the SRBCT samples:

```
del.idx = which( khan2001$y == "non-SRBCT" )
srbct.x = khan2001$x[-del.idx,]
srbct.y = factor(khan2001$y[-del.idx])
dim(srbct.x)
```

```
## [1]   83 2308
```

Four subtypes of cancer:

```
levels(srbct.y)
```

```
## [1] "BL"  "EWS" "NB"  "RMS"
```

Divide into training and test data

```
Xtrain = srbct.x[1:63,]
Ytrain = srbct.y[1:63]
Xtest = srbct.x[64:83,]
Ytest = srbct.y[64:83]
```

Diagonal Discriminant Analysis (DDA)

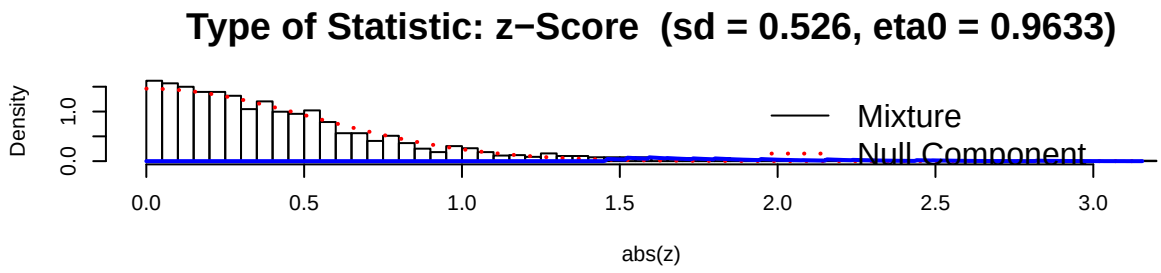
In DDA correlation among predictors is assumed to be zero, i.e. a diagonal covariance matrix is used.

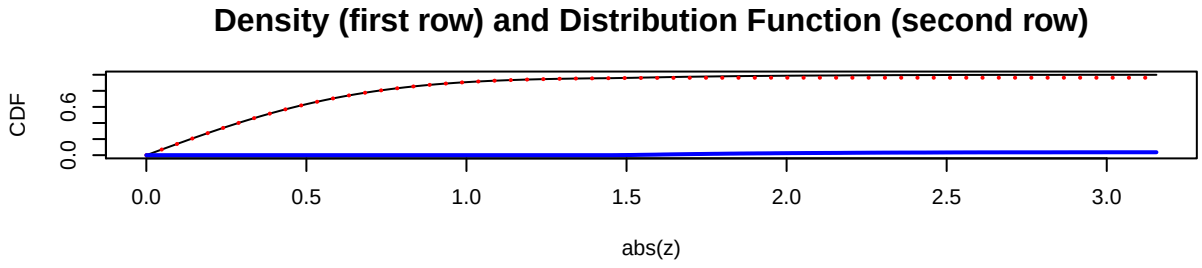
Step 1 - feature ranking

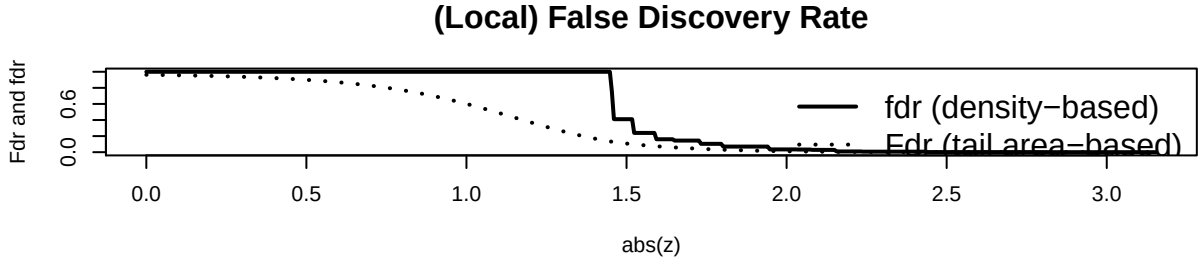
As there are more than two groups in the response there are three different ways to obtain a summary test statistic to rank genes: a) ranking by averaged squared t-scores across the four groups

```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, diagonal=TRUE, ranking.score="avg")
```

```
## Computing t-scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
##
##
## Computing false discovery rates and higher cricitism scores for each feature
```







```
sum( ra[, "lfdr"]< 0.80)  # 97 genes included in classifier (by FNDR control)
```

```
## [1] 97
```

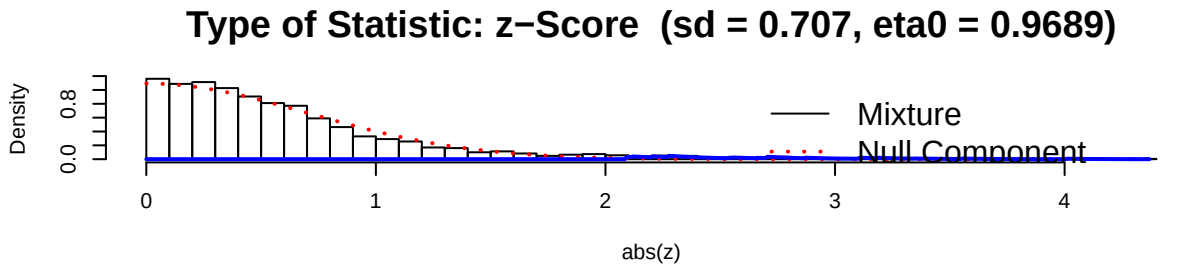
```
which.max( ra[, "HC"] )  #  145 genes according to HC criterion
```

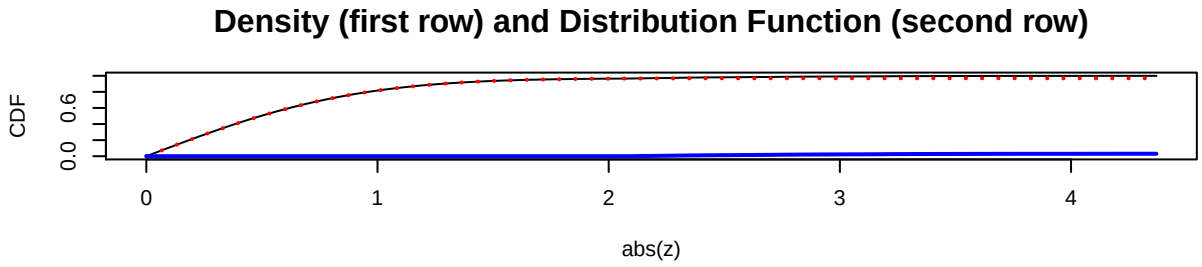
```
## 200814
##    145
```

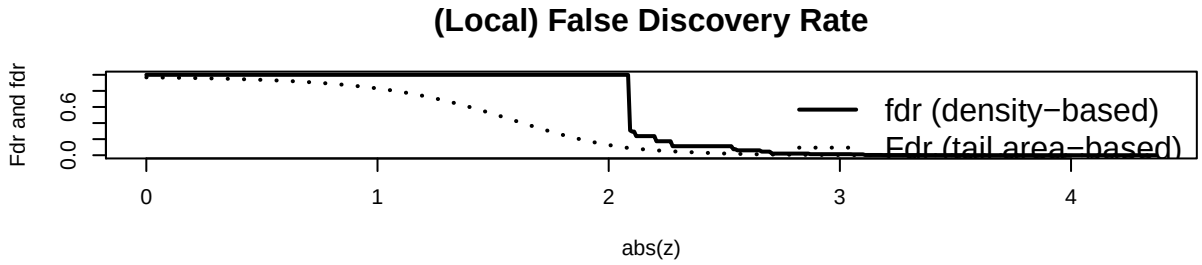
b) ranking by maximum of squared t-scores across the four groups

```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, diagonal=TRUE, ranking.score="max")
```

```
## Computing t-scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
##
##
## Computing false discovery rates and higher cricitism scores for each feature
```








```
sum( ra[, "lfdr"]< 0.80)  # 78 genes included in classifier (by FNDR control)
```

```
## [1] 78
```

```
which.max( ra[, "HC"] )  #  121 genes according to HC criterion
```

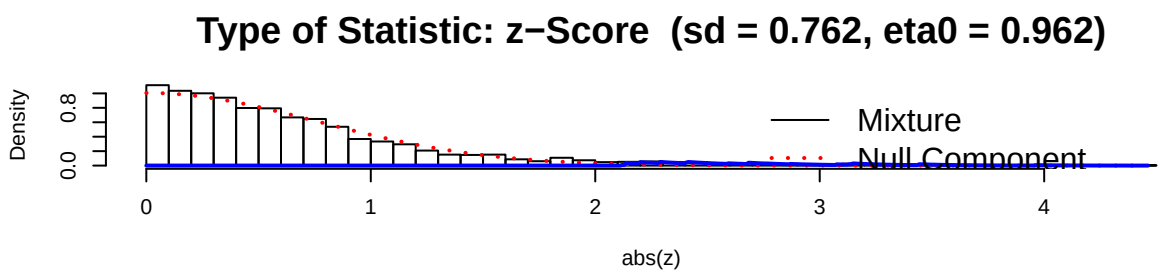
```
## 80338
##   121
```

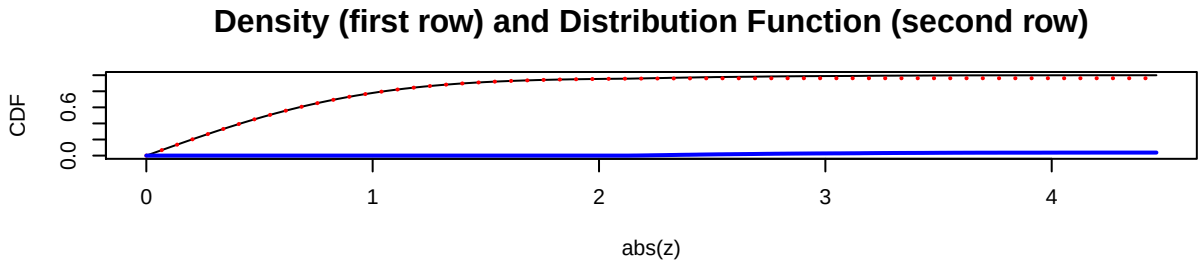
c) ranking by mutual information (weighted sum of squared t-scores)

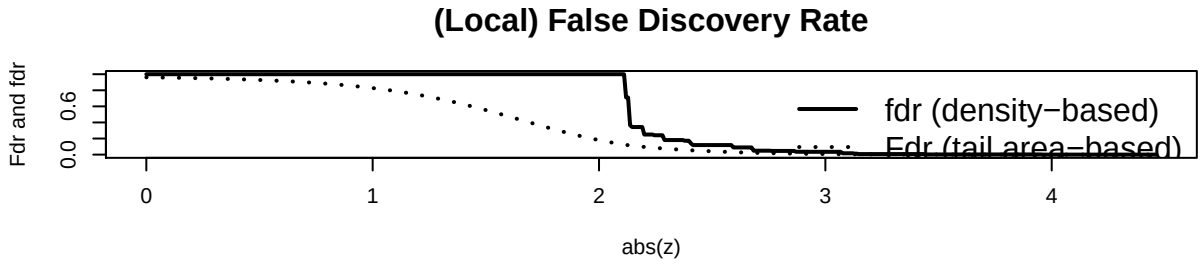
```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, diagonal=TRUE, ranking.score="entropy")
```

```
## Computing t-scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
```

```
##
##
## Computing false discovery rates and higher cricitism scores for each feature
```








```
sum( ra[, "lfdr"]< 0.80)  # 99 genes included in classifier (by FNDR control)
```

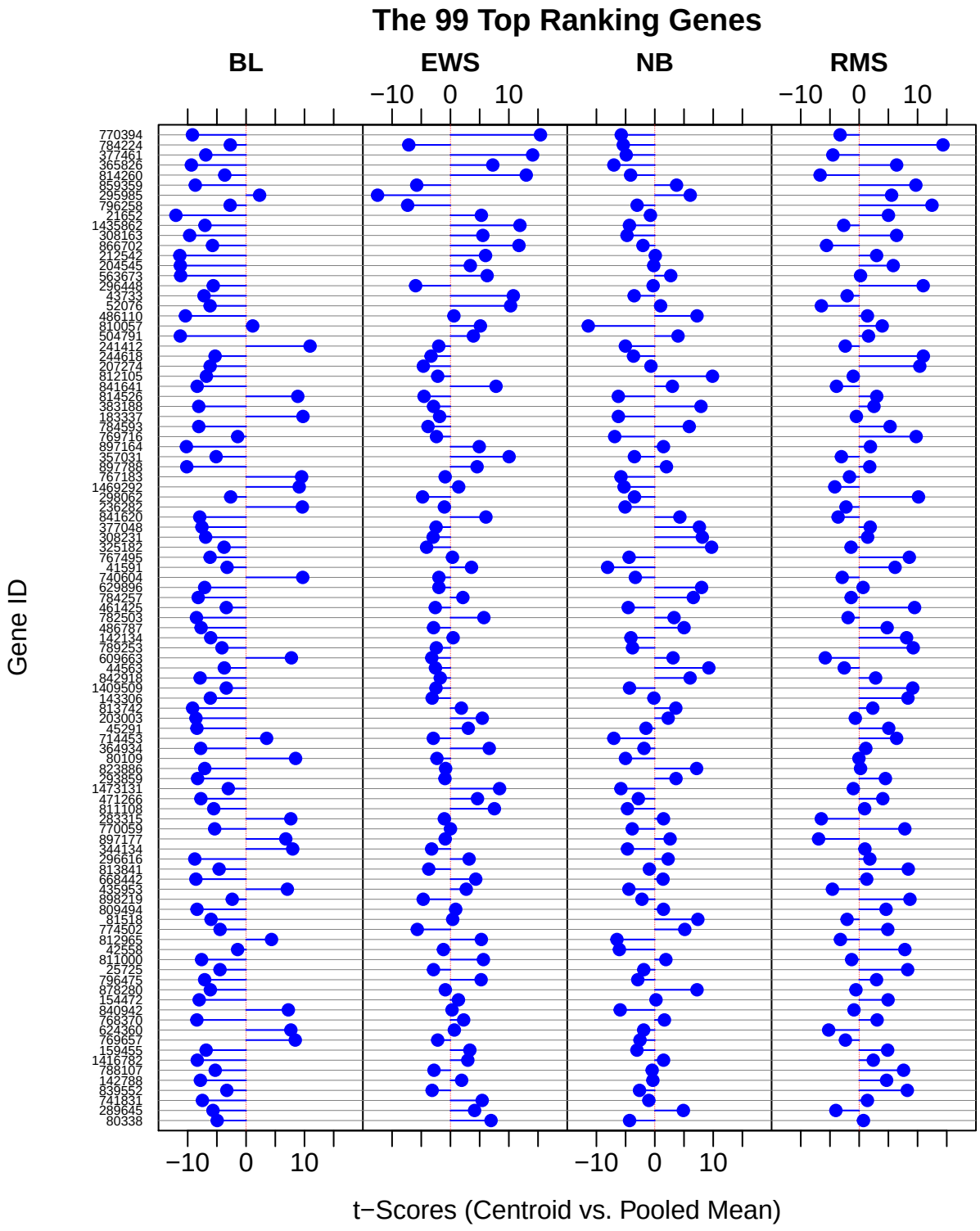
```
## [1] 99
```

```
which.max( ra[, "HC"] )  #  158 genes according to HC criterion
```

```
## 220096
##    158
```

here we pick the top 99 genes of option c)

```
plot(ra, top=99, main="The 99 Top Ranking Genes", ylab="Gene ID")
```



Select these 99 variables:

```
idx = ra[1:99,"idx"]
Xtrain2 = Xtrain[,idx]
Xtest2 = Xtest[,idx]
```

Step 2 - training the classifier

Learn DDA predictor:

```
sda.fit = sda(Xtrain2, Ytrain, diagonal=TRUE)
```

```
## Number of variables: 99
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.1951
```

Step 3 - prediction

Predict class labels from test data and compare with known labels:

```
dim(Xtest2)
```

```
## [1] 20 99
```

```
predict(sda.fit, Xtest2)
```

```
## Prediction uses 99 features.

## $class
##  [1] NB  RMS NB  EWS RMS BL  EWS RMS EWS EWS RMS RMS BL  RMS NB  NB  NB
## [18] NB  BL  EWS
## Levels: BL EWS NB RMS
##
## $posterior
##         BL       EWS NB       RMS
## TEST-8   0 0.0000000  1 0.0000000
## TEST-10  0 0.0000000  0 1.0000000
## TEST-1   0 0.0000000  1 0.0000000
## TEST-2   0 1.0000000  0 0.0000000
## TEST-4   0 0.0000000  0 1.0000000
## TEST-7   1 0.0000000  0 0.0000000
## TEST-12  0 1.0000000  0 0.0000000
## TEST-24  0 0.0000000  0 1.0000000
## TEST-6   0 1.0000000  0 0.0000000
## TEST-21  0 1.0000000  0 0.0000000
## TEST-20  0 0.0009358  0 0.9990642
## TEST-17  0 0.0000000  0 1.0000000
## TEST-18  1 0.0000000  0 0.0000000
## TEST-22  0 0.0000000  0 1.0000000
## TEST-16  0 0.0000000  1 0.0000000
## TEST-23  0 0.0000000  1 0.0000000
## TEST-14  0 0.0000000  1 0.0000000
## TEST-25  0 0.0000000  1 0.0000000
## TEST-15  1 0.0000000  0 0.0000000
## TEST-19  0 1.0000000  0 0.0000000
```

```
ynew = predict(sda.fit, Xtest2)$class
```

```
## Prediction uses 99 features.
```

Number of missclassified test samples:

```
sum(ynew != Ytest)
```

```
## [1] 1
```

Linear Discriminant Analysis (LDA)

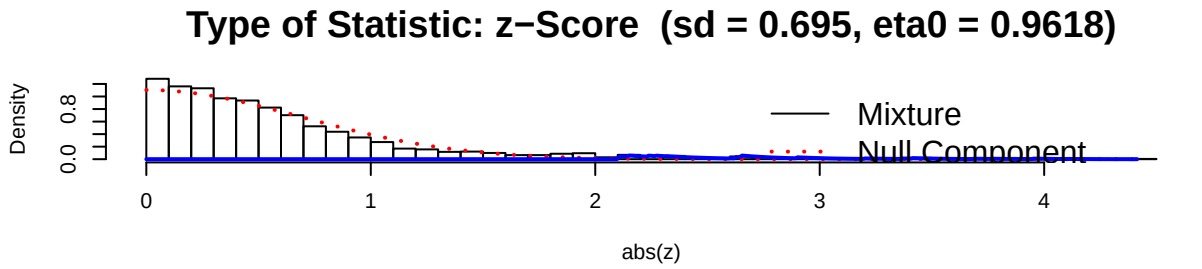
In LDA correlation among predictors is taken into account.

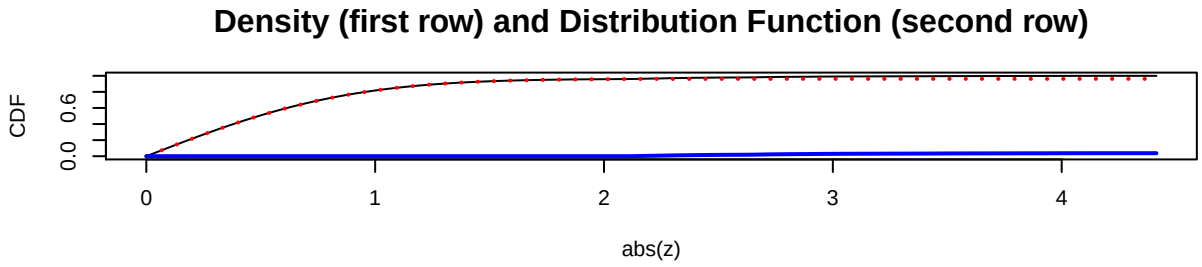
Step 1 - feature ranking

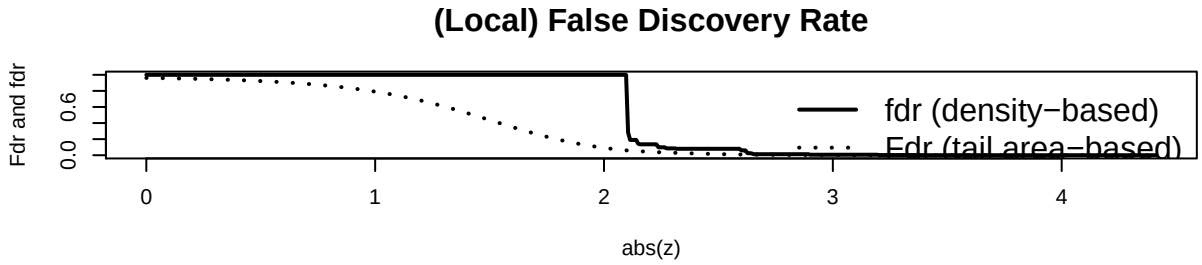
As there are more than two groups in the response there are three different ways to obtain a summary test statistic to rank genes: a) ranking by averaged squared cat-scores across the four groups

```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, ranking.score="avg")
```

```
## Computing cat scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
##
## Computing the square root of the inverse pooled correlation matrix
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.3279
##
## Computing false discovery rates and higher cricitism scores for each feature
```








```
sum( ra[, "lfdr"]< 0.80)  # 93 genes included in classifier (by FNDR control)
```

```
## [1] 93
```

```
which.max( ra[, "HC"] )  #  143 genes according to HC criterion
```

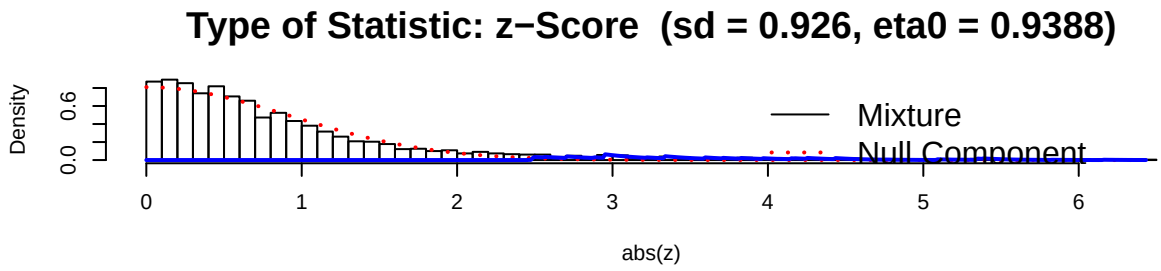
```
## 380620
##    143
```

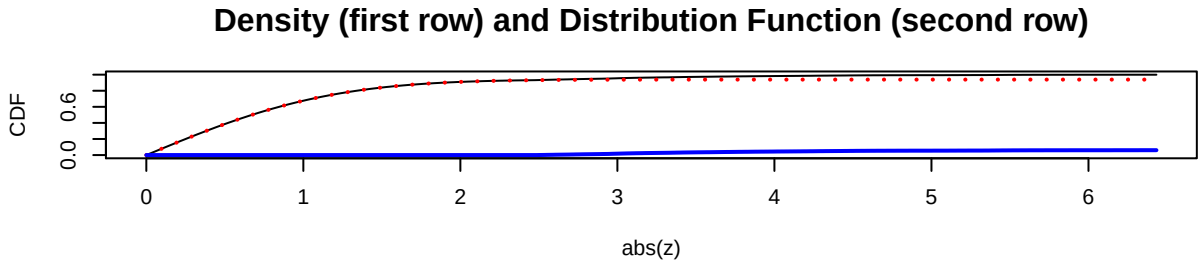
b) ranking by maximum of squared cat-scores across the four groups

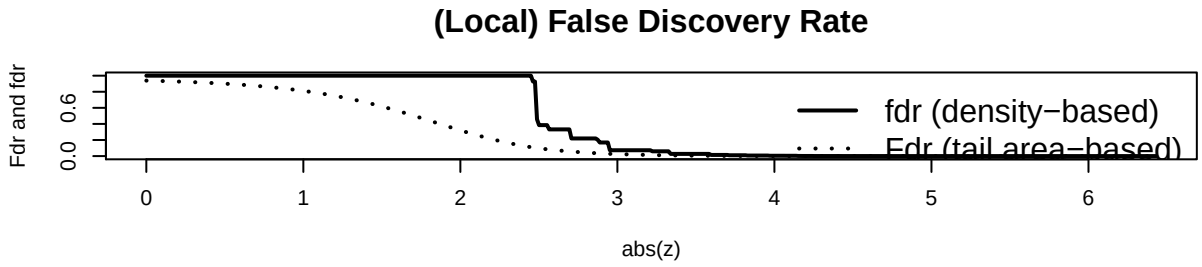
```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, ranking.score="max")
```

```
## Computing cat scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
```

```
##
## Computing the square root of the inverse pooled correlation matrix
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.3279
##
## Computing false discovery rates and higher cricitism scores for each feature
```








```
sum( ra[, "lfdr"]< 0.80)  # 156 genes included in classifier (by FNDR control)
```

```
## [1] 156
```

```
which.max( ra[, "HC"] )  #  194 genes according to HC criterion
```

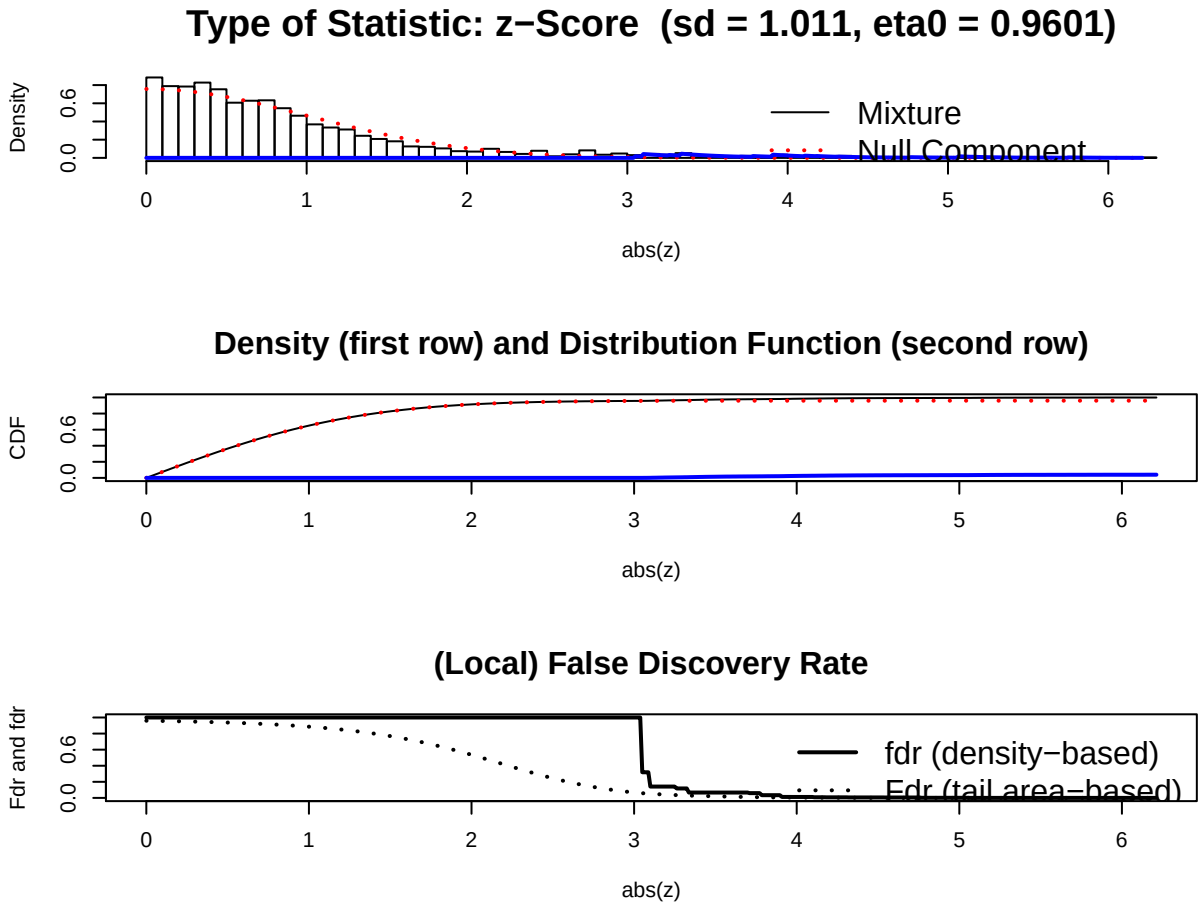
```
## 377048
##    194
```

c) ranking by mutual information (weighted sum of squared cat-scores)

```
ra = sda.ranking(Xtrain, Ytrain, fdr=TRUE, plot.fdr=TRUE, ranking.score="entropy")
```

```
## Computing cat scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 2308
```

```
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.153
##
## Computing the square root of the inverse pooled correlation matrix
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.3279
##
## Computing false discovery rates and higher cricitism scores for each feature
```




```
sum( ra[, "lfdr"]< 0.80)  # 97 genes included in classifier (by FNDR control)
```

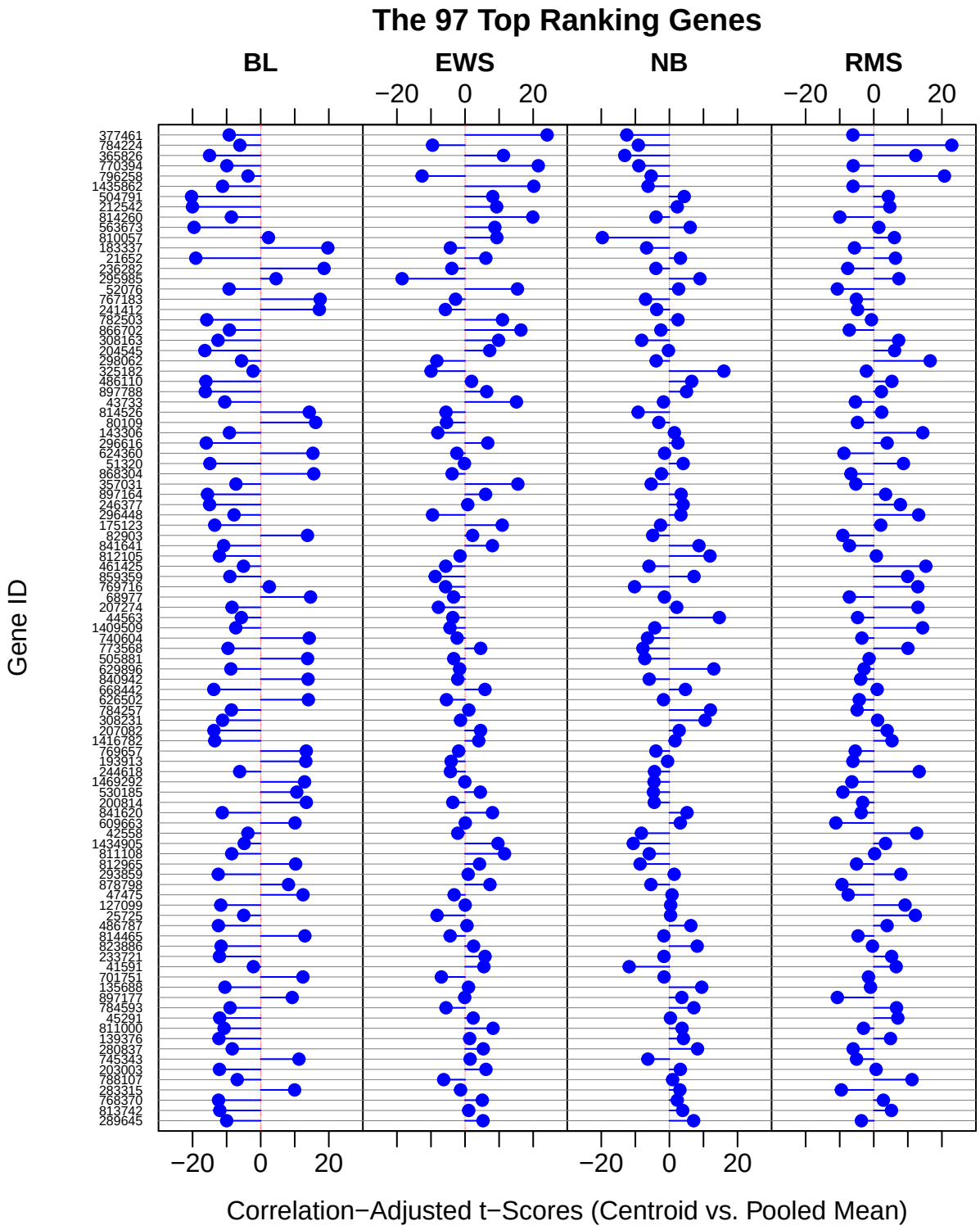
```
## [1] 97
```

```
which.max( ra[, "HC"] )  #  140 genes according to HC criterion
```

```
## 129387
##    140
```

here we pick the top 97 genes of option c)

```
plot(ra, top=97, main="The 97 Top Ranking Genes", ylab="Gene ID")
```



Select these 97 variables:

```
idx = ra[1:97,"idx"]
Xtrain2 = Xtrain[,idx]
Xtest2 = Xtest[,idx]
```

Step 2 - training the classifier

Learn LDA predictor:

```
sda.fit = sda(Xtrain2, Ytrain)
```

```
## Number of variables: 97
## Number of observations: 63
## Number of classes: 4
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.3156
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.1785
##
##
## Computing inverse correlation matrix (pooled across classes)
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.4407
```

Step 3 - prediction

Predict class labels from test data and compare with known labels:

```
dim(Xtest2)
```

```
## [1] 20 97
```

```
predict(sda.fit, Xtest2)
```

```
## Prediction uses 97 features.

## $class
##  [1] NB  RMS NB  EWS RMS BL  EWS RMS EWS EWS EWS RMS BL  RMS NB  NB  NB
## [18] NB  BL  EWS
## Levels: BL EWS NB RMS
##
## $posterior
##          BL EWS NB RMS
## TEST-8    0   0  1   0
## TEST-10   0   0  0   1
## TEST-1    0   0  1   0
## TEST-2    0   1  0   0
## TEST-4    0   0  0   1
## TEST-7    1   0  0   0
## TEST-12   0   1  0   0
## TEST-24   0   0  0   1
```

```
## TEST-6   0   1  0   0
## TEST-21  0   1  0   0
## TEST-20  0   1  0   0
## TEST-17  0   0  0   1
## TEST-18  1   0  0   0
## TEST-22  0   0  0   1
## TEST-16  0   0  1   0
## TEST-23  0   0  1   0
## TEST-14  0   0  1   0
## TEST-25  0   0  1   0
## TEST-15  1   0  0   0
## TEST-19  0   1  0   0
```

```
ynew = predict(sda.fit, Xtest2)$class
```

```
## Prediction uses 97 features.
```

Number of missclassified test samples:

```
sum(ynew != Ytest)
```

```
## [1] 0
```

Estimate prediction accuracy using crossvalidation

Using crossvalidation we can estimate the prediction error from the training data set alone.

```
library("crossval")
```

Setup prediction function: estimate the accuracy of a predictor with a fixed number of predictors (note this takes into account the uncertainty in estimating the variable ordering).

```
predfun = function(Xtrain, Ytrain, Xtest, Ytest, numVars, diagonal=FALSE,
                   ranking.score="entropy")
{
  # estimate ranking and determine the best numVars variables
  ra = sda.ranking(Xtrain, Ytrain, verbose=FALSE, diagonal=diagonal,
                   fdr=FALSE, ranking.score=ranking.score)
  selVars = ra[,"idx"][1:numVars]

  # fit and predict
  sda.out = sda(Xtrain[, selVars, drop=FALSE], Ytrain, diagonal=diagonal,
                verbose=FALSE)
  ynew = predict(sda.out, Xtest[, selVars, drop=FALSE], verbose=FALSE)$class

  # compute accuracy
  acc = mean(Ytest == ynew)

  return(acc)
}
```

Our setup for crossvalidation:

```
K = 10 # number of folds
B = 20 # number of repetitions
```

Crossvalidation estimate of accuracy for LDA using the top 100 features ranked by CAT scores (combined across groups using "entropy" for overall ranking):

```
set.seed(12345)
cv.lda100 = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=100,
                     diagonal=FALSE, verbose=FALSE)
cv.lda100$stat
```

```
## [1] 1
```

Comparison of LDA / DDA and "entropy" and "max" options

LDA using the top 10 features ranked by CAT scores (combined across groups using "entropy" for overall ranking):

```
set.seed(12345)
cv.lda10 = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=10,
                     diagonal=FALSE, verbose=FALSE)
cv.lda10$stat
```

```
## [1] 0.9909762
```

DDA using the top 10 features ranked by t scores (combined across groups using "entropy" for overall ranking):

```
set.seed(12345)
cv.dda10 = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=10,
                    diagonal=TRUE, verbose=FALSE)
cv.dda10$stat
```

```
## [1] 0.9643869
```

DDA using the top 10 features ranked by t scores, (combined across groups using "max" for overall ranking, as in PAM):

```
set.seed(12345)
cv.dda10b = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=10,
                     diagonal=TRUE, ranking.score="max", verbose=FALSE)
cv.dda10b$stat
```

```
## [1] 0.9585595
```

Conclusions:

1. LDA/CAT score ranking performs petter than DDA/t-score ranking.
2. "entropy" is better as group summary than "max".