Analysis of Singh et al. (2002) Prostate Cancer Data

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

Load "sda" package and Singh et al. (2002) data set

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

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

Singh et al. (2002) gene expression data for prostate cancer:

```
data(singh2002)

Xtrain = singh2002$x
Ytrain = singh2002$y

dim(Xtrain)      # 102 6033
```

```
## [1]  102 6033
```

```
length(Ytrain)   # 102
```

```
## [1] 102
```

```
levels(Ytrain)
```

```
## [1] "cancer"  "healthy"
```

Feature ranking with t-scores

Corresponds to assuming a diagonal covariance matrix (DDA).

Compute ranking:

```
ranking.DDA = sda.ranking(Xtrain, Ytrain, diagonal=TRUE)
```

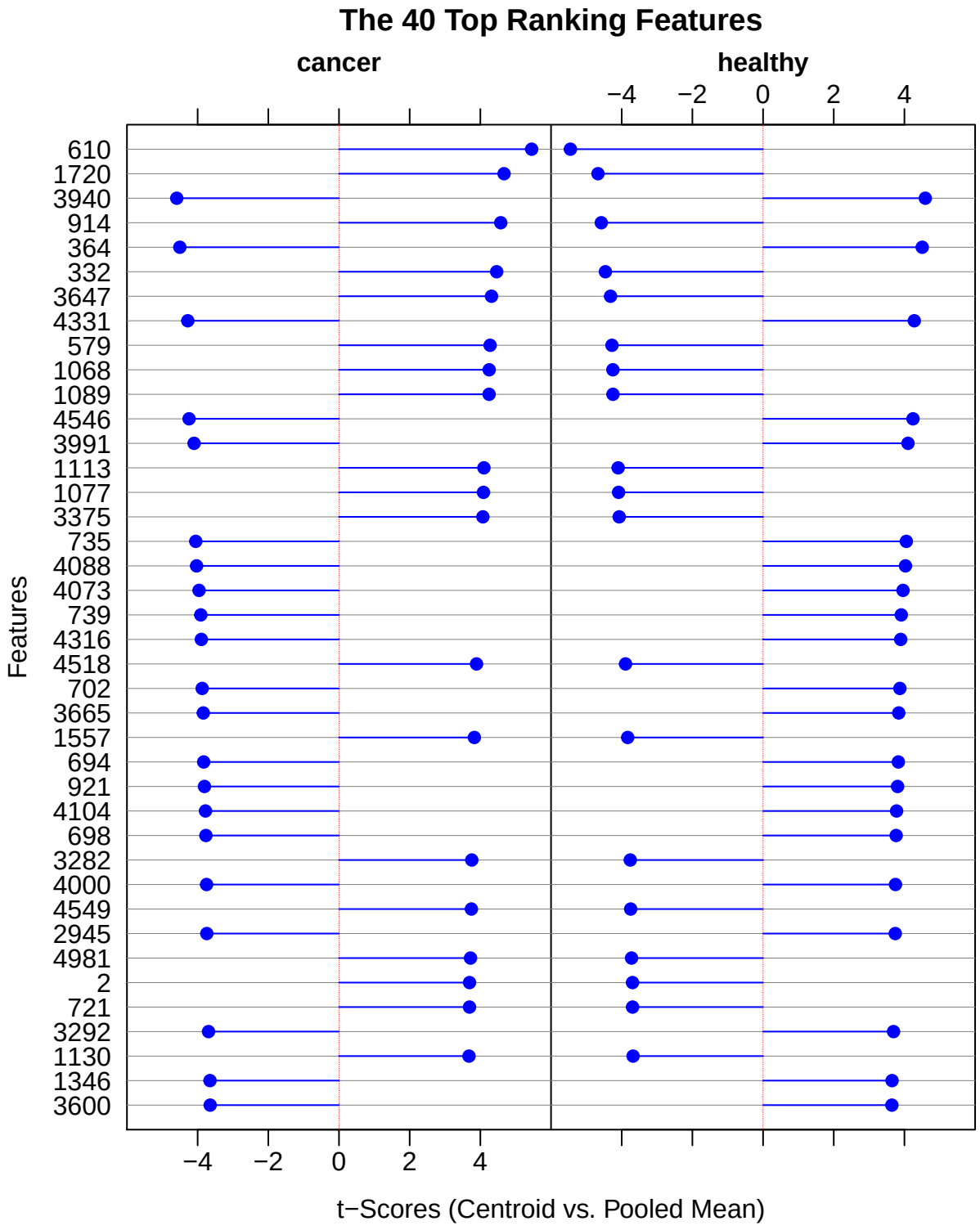
```
## Computing t-scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 6033
## Number of observations: 102
## Number of classes: 2
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 1
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.205
##
##
## Computing false discovery rates and higher cricitism scores for each feature
```

```
ranking.DDA[1:10,]
```

```
##        idx    score  t.cancer t.healthy       lfdr        HC
##  [1,]  610 29.72230  5.451816 -5.451816 0.02654395 0.9987232
##  [2,] 1720 21.81849  4.671027 -4.671027 0.02654395 1.3752835
##  [3,] 3940 21.07592 -4.590851  4.590851 0.02654395 1.6870685
##  [4,]  914 20.94853  4.576957 -4.576957 0.02654395 1.9588904
##  [5,]  364 20.28183 -4.503535  4.503535 0.02654395 2.1857747
##  [6,]  332 19.90548  4.461556 -4.461556 0.03293651 2.3948242
##  [7,] 3647 18.63023  4.316275 -4.316275 0.03293651 2.5525263
##  [8,] 4331 18.31253 -4.279314  4.279314 0.03293651 2.7272030
##  [9,]  579 18.28490  4.276085 -4.276085 0.03293651 2.9037418
## [10,] 1068 18.06190  4.249930 -4.249930 0.03293651 3.0609904
```

Plot t-scores for the top 40 genes:

```
plot(ranking.DDA, top=40)
```



Number of features with local FDR < 0.8 (i.e. features useful for prediction):

```
sum(ranking.DDA[,"lfdr"] < 0.8) # 166
```

```
## [1] 166
```

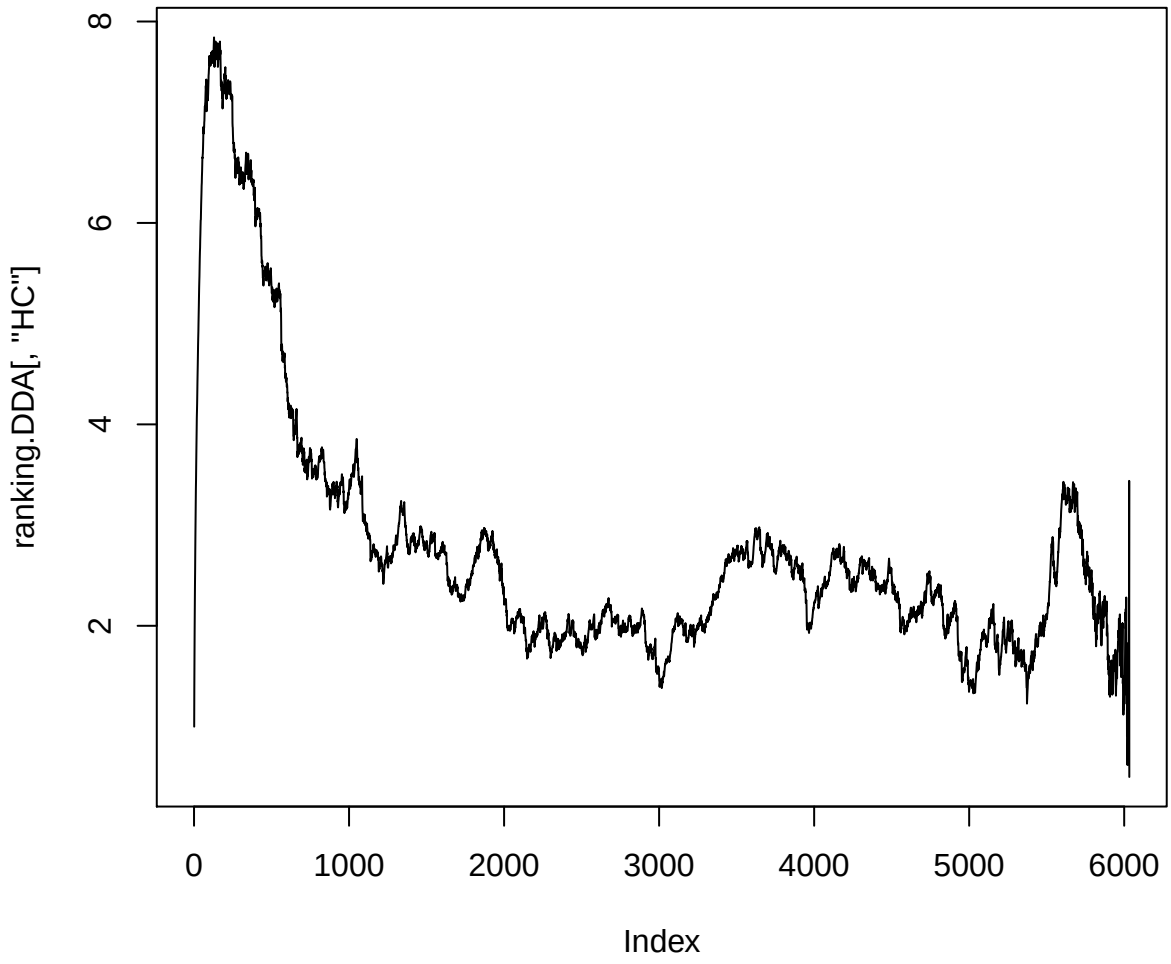
Number of features with local FDR < 0.2 (i.e. significant non-null features):

```
sum(ranking.DDA[,"lfdr"] < 0.2) # 53
```

```
## [1] 53
```

Optimal number of features according to Higher Criticism:

```
plot(ranking.DDA[,"HC"], type="l")
```




```
which.max( ranking.DDA[1:1000,"HC"] ) #129
```

```
## [1] 129
```

Feature ranking with correlation-adjusted t-scores (CAT scores)

Corresponds to assuming a full covariance matrix (LDA).

Compute ranking:

```
ranking.LDA = sda.ranking(Xtrain, Ytrain, diagonal=FALSE)
```

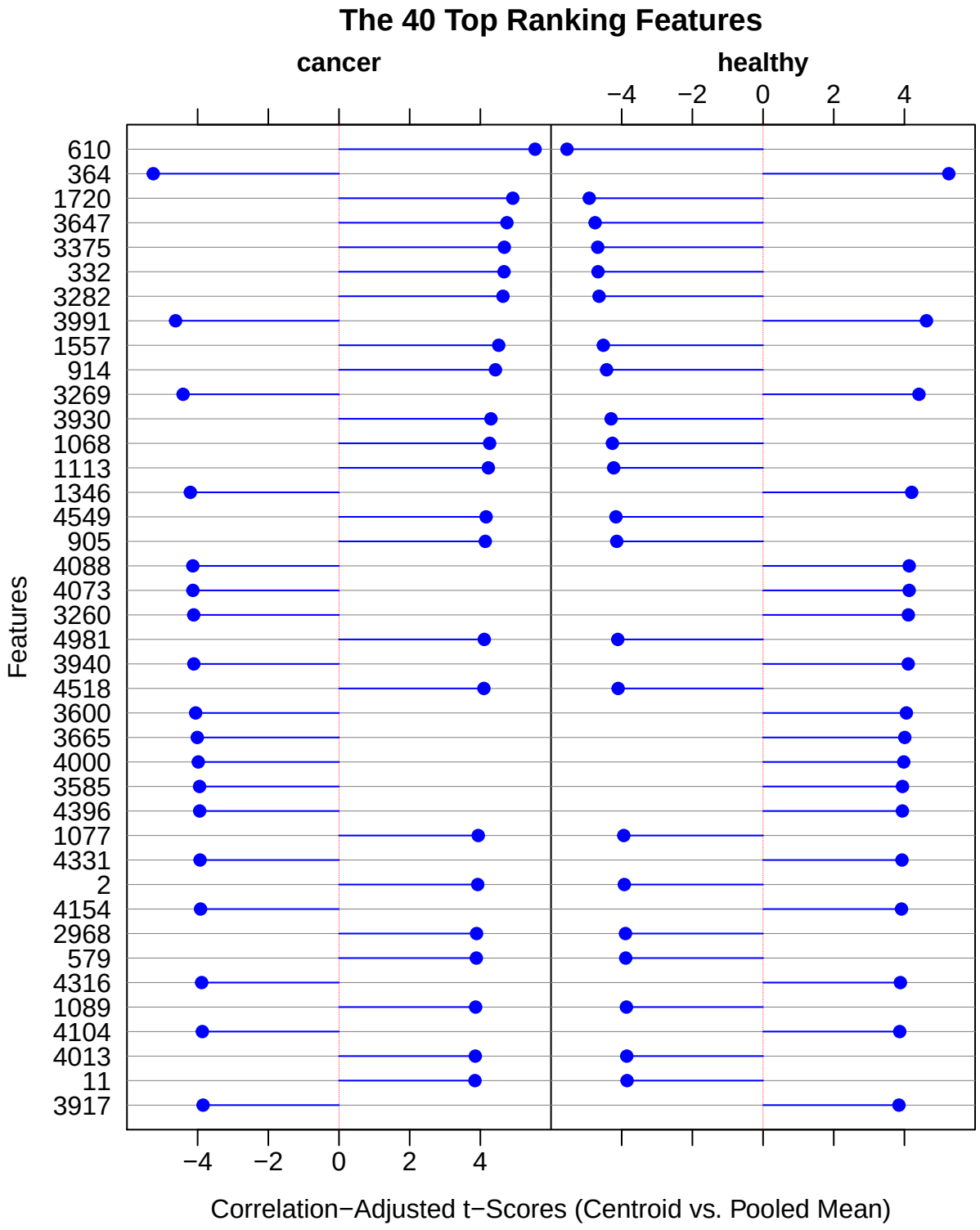
```
## Computing cat scores (centroid vs. pooled mean) for feature ranking
##
## Number of variables: 6033
## Number of observations: 102
## Number of classes: 2
##
## Estimating optimal shrinkage intensity lambda.freq (frequencies): 1
## Estimating variances (pooled across classes)
## Estimating optimal shrinkage intensity lambda.var (variance vector): 0.205
##
## Computing the square root of the inverse pooled correlation matrix
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.8924
##
## Computing false discovery rates and higher cricitism scores for each feature
```

```
ranking.LDA[1:10,]
```

```
##        idx    score cat.cancer cat.healthy        lfdr        HC
##  [1,]  610 30.79055   5.548923   -5.548923 0.001534840 0.9996617
##  [2,]  364 27.61803  -5.255285    5.255285 0.006636026 1.4130468
##  [3,] 1720 24.18723   4.918052   -4.918052 0.006636026 1.7263343
##  [4,] 3647 22.58188   4.752040   -4.752040 0.006636026 1.9889369
##  [5,] 3375 21.88957   4.678629   -4.678629 0.006636026 2.2222410
##  [6,]  332 21.81020   4.670139   -4.670139 0.006636026 2.4367013
##  [7,] 3282 21.53894   4.641007   -4.641007 0.006636026 2.6324612
##  [8,] 3991 21.37747  -4.623578    4.623578 0.022021468 2.8152840
##  [9,] 1557 20.42711   4.519636   -4.519636 0.022021468 2.9795916
## [10,]  914 19.60042   4.427237   -4.427237 0.022021468 3.1325509
```

Plot cat scores for the top 40 genes:

```
plot(ranking.LDA, top=40)
```



Number of features with local FDR < 0.8 (i.e. features useful for prediction):

```
sum(ranking.LDA[,"lfdr"] < 0.8) # 131
```

```
## [1] 131
```

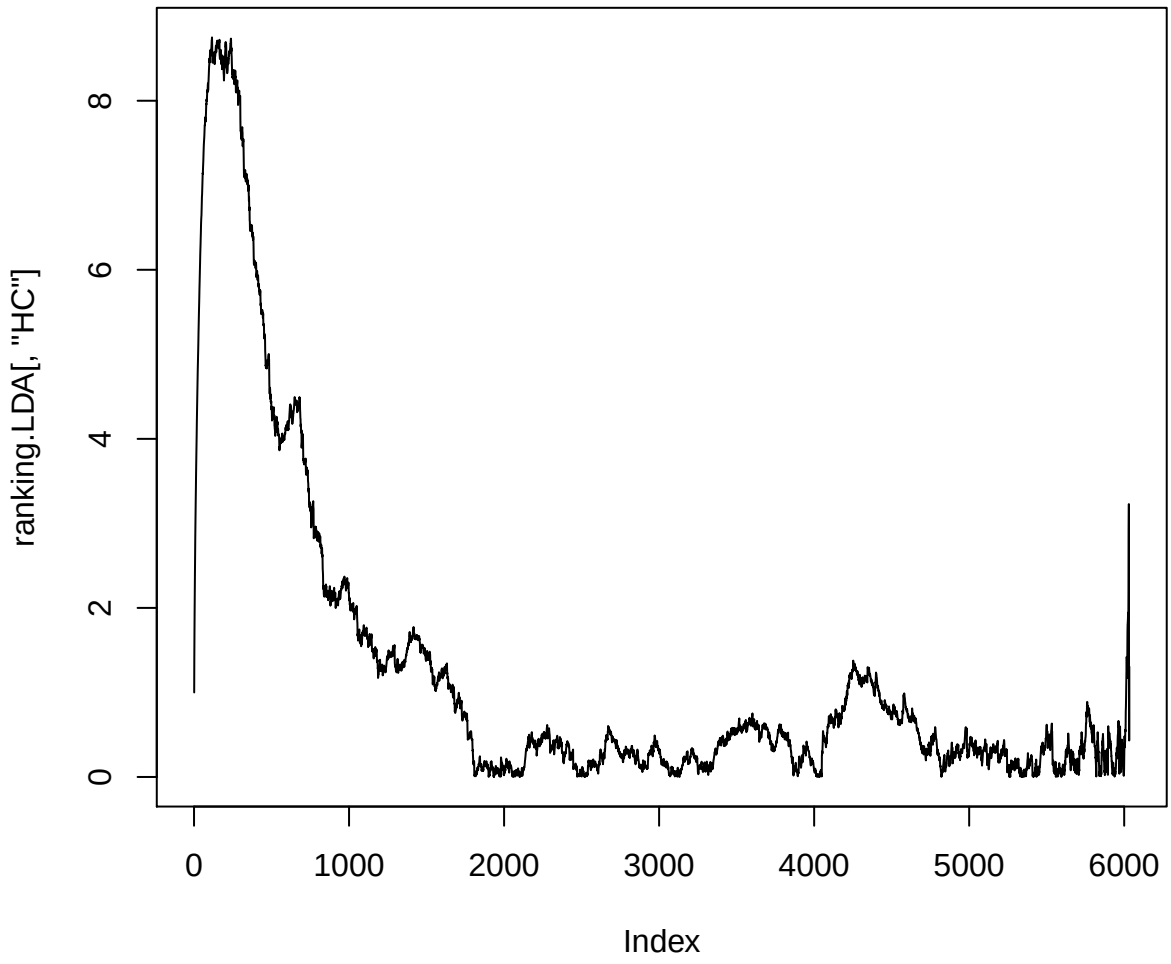
Number of features with local FDR < 0.2 (i.e. significant non-null features):

```r
sum(ranking.LDA[,"lfdr"] < 0.2) # 62
```

```
## [1] 62
```

Optimal number of features according to Higher Criticism:

```r
plot(ranking.LDA[,"HC"], type="l")
```




```r
which.max( ranking.LDA[1:1000,"HC"] ) # 116
```

```
## [1] 116
```

Estimate prediction accuracy using crossvalidation

```
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)
{
  # estimate ranking and determine the best numVars variables
  ra = sda.ranking(Xtrain, Ytrain, verbose=FALSE, diagonal=diagonal, fdr=FALSE)
  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

  # count false and true positives/negatives
  negative = levels(Ytrain)[2] # "healthy"
  cm = confusionMatrix(Ytest, ynew, negative=negative)

  return(cm)
}
```

Our setup for crossvalidation:

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

Estimate accuracy of LDA using the top 120 features ranked by CAT scores:

```
set.seed(12345)
cv.lda120 = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=120, diagonal=FALSE, verbose=FALSE)
cv.lda120$stat
```

```
##    FP    TP    TN    FN
## 0.110 4.725 4.890 0.475
```

```
diagnosticErrors(cv.lda120$stat)
```

```
##       acc      sens      spec       ppv       npv       lor
## 0.9426471 0.9086538 0.9780000 0.9772492 0.9114632 6.0917753
```

Estimate accuracy of DDA using the top 120 features ranked by t scores:

```
set.seed(12345)
cv.dda120 = crossval(predfun, Xtrain, Ytrain, K=K, B=B, numVars=120, diagonal=TRUE, verbose=FALSE)
cv.dda120$stat
```

```
##    FP    TP    TN    FN
## 0.170 4.715 4.830 0.485
```

```
diagnosticErrors(cv.dda120$stat)
```

```
##       acc      sens      spec       ppv       npv       lor
## 0.9357843 0.9067308 0.9660000 0.9651996 0.9087488 5.6211586
```

Same as before but using only the top 10 features:

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

```
##    FP    TP    TN    FN
## 1.370 3.355 3.630 1.845
```

```
diagnosticErrors(cv.dda10$stat)
```

```
##       acc      sens      spec       ppv       npv       lor
## 0.6848039 0.6451923 0.7260000 0.7100529 0.6630137 1.5723944
```