Homework 4

Kent Codding

2025-04-11

For this assignment, please use the full training and testing datasets from the SwitchBox package. The datasets can be called out from the following commands:

```
#BiocManager::install("switchBox")
library(switchBox)
```

```
## Loading required package: pROC
```

```
## Warning: package 'pROC' was built under R version 4.4.2
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
## Loading required package: gplots
```

```
## Warning: package 'gplots' was built under R version 4.4.2
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
data(trainingData)
data(testingData)
```

The first two functions install and load the `switchBox` package. The last two functions, `data(trainingData)` and `data(testingData)`, load the datasets to be used: `matTesting`, `testingGroup`, `matTraining`, and `trainingGroup`.

```
masomenos.train = function(X,   # pxn data frame of predictors
                           Y,   # nx1 vectors of labels or responses
                           P,   # number of predictors
                           training.criterion="AUC",
                           filtering.fraction=.5)
{
  # eliminate unclassified samples
  # X <- matTraining
  # Y <- YY
  YY = Y[!is.na(Y)]
  XX = X[,!is.na(Y)]
```

```
  Nvar = nrow(XX) # Number of genes
  crite = rep(NA,Nvar) # These are to store the criterion for each gene.
  if (training.criterion=="AUC"){
    for (pp in 1:Nvar){
      crite[pp] <- as.numeric( wilcox.test(XX[pp,]~YY)$statistic / (sum(YY==0)*sum(YY==1)) )
      #XX[pp,] gives data values, YY gives levels of the groups to be compared.
    }
  }
  # P = 12
  cutoff = sort(abs(crite- 0.5) + 0.5,decreasing = TRUE)[P]
  # cutoff = sort(abs(crite),decreasing = TRUE)[P]
  # sort the absolute value of the criterion from largest to smallest, then look at the corresponding c
  cutoff
  variables = (1:Nvar)[abs(crite- 0.5) + 0.5 >= cutoff]
  # retrieve the variable name/number of the top P.
  variables
  variables.signs = ( 2 * ( crite > 0.5 ) - 1 ) [variables]
  scores = apply ( XX[ variables, ] * variables.signs, 2, mean )
  # this is known as the risk score, or the mean expression value for the top P genes for each patient.

  if (training.criterion=="AUC"){
    crite.mom = as.numeric( wilcox.test(scores~YY)$statistic / (sum(YY==0)*sum(YY==1)) )
  }

  MoM = list(XX=XX,YY=YY,cutoff=cutoff,
             training.criterion=training.criterion,
             variables = variables,
             variables.signs=variables.signs,
             variables.criterion=crite[variables],
             scores=scores,
             criterion.mom=crite.mom)
  return(MoM)
}

masomenos.test = function(X,    # pxn data frame of predictors
                          Y,    # nx1 vectors of labels or responses
                          MoM.out # output form masomenos.train
){
  # eliminate unclassified samples
  # Y = 1*as.vector(testingGroup=="Good")
  # X = matTesting
  # X = matTesting[genename,]
  # MoM.out = MoM.train
  YY = Y[!is.na(Y)]
  XX = X[,!is.na(Y)]

  Nvar = nrow(XX)
  scores = apply ( XX[ MoM.out$variables, ] * MoM.out$variables.signs, 2, mean )

  if (MoM.out$training.criterion=="AUC"){
    crite.mom = as.numeric( wilcox.test(scores~YY)$statistic / (sum(YY==0)*sum(YY==1)) )
  }
```

```
  MoM = list(XX=XX,YY=YY,
             scores=scores,
             criterion.mom=crite.mom)
  return(MoM)
}
```

Then, complete the following:

1) Use the training set to develop:

```
matTraining[1:10, 1:4]
```

```
##                       Training1.Bad Training2.Bad Training3.Good Training4.Good
## AA555029_RC_Hs.370457  -0.056350595  -0.095194433   -0.037045311   -0.095419142
## AF257175_Hs.15250       0.034705973   0.007342677    0.004681334    0.101843496
## AK000745_Hs.377155     -0.045137256   0.091290570   -0.040696770   -0.209359963
## AKAP2_Hs.516834        -0.155568976   0.052144831    0.110565426    0.016090416
## AL080059_Hs.173094      0.139404525   0.118474636   -0.017371460    0.005628748
## AL137718_Hs.508141     -0.067700161  -0.216129499   -0.033547007   -0.175422053
## ALDH4_Hs.133062        -0.029185248   0.107993349    0.022367094   -0.015099136
## AP2B1_Hs.514819        -0.007791995  -0.219541533   -0.070519054   -0.230893195
## BBC3_Hs.467020          0.035148476   0.029680305    0.083738225   -0.069546449
## CCNE2_Hs.408658         0.057564899   0.054830828    0.078571255   -0.076209651
```

```
head(trainingGroup, 10)
```

```
##  [1] Bad  Bad  Good Good Bad  Bad  Good Good Bad  Bad
## Levels: Bad Good
```

– 1a) A k-tsp classifier with 6 pairs.

```
classifier = SWAP.KTSP.Train(matTraining, trainingGroup, krange=6)
print(classifier)
```

```
## $name
## [1] "6TSPs"
##
## $TSPs
##      [,1]                       [,2]
## [1,] "GNAZ_Hs.555870"           "Contig32185_RC_Hs.159422"
## [2,] "Contig46223_RC_Hs.22917"  "OXCT_Hs.278277"
## [3,] "RFC4_Hs.518475"           "L2DTL_Hs.445885"
## [4,] "Contig40831_RC_Hs.161160" "CFFM4_Hs.250822"
## [5,] "FLJ11354_Hs.523468"       "LOC57110_Hs.36761"
## [6,] "Contig55725_RC_Hs.470654" "IGFBP5_Hs.184339"
##
## $score
## [1] 0.6029423 0.5467924 0.5347600 0.5280755 0.5267389 0.5200542
##
## $labels
## [1] "Bad"  "Good"
```

– 1b) A mas-o-menos classifier using the same genes identified in the top 6 pairs in 1a.

```
gene_names = c(classifier$TSPs)
YY = 1 * as.vector(trainingGroup == "Good")

mom_classifier = masomenos.train(matTraining, YY, P=12, training.criterion="AUC")
```

2) Compare these classifiers in the validation set, using a criterion of your choice (justify).

```
tsp_result = SWAP.GetKTSP.Result(classifier, matTesting, testingGroup)
tsp_result$stats["auc"]
```

```
##       auc
## 0.6905074
```

```
YY_test = 1 * as.vector(testingGroup == "Good")
mom_result = masomenos.test(matTesting, YY_test, mom_classifier)
mom_result$criterion.mom
```

```
## [1] 0.7301146
```

Evaluating the classifiers on the validation set, I chose the Area Under the Receiver Operating Characteristic Curve (AUC) as the performance metric. AUC assesses how well each model distinguishes between 'good' and 'bad' outcomes across all possible decision thresholds, thereby capturing their overall discriminative power. This metric is preferable to fixed-threshold measures like accuracy, sensitivity, or specificity because it summarizes the trade-off between true positive and false positive rates, an especially valuable feature when the classifiers are built on a limited number of gene expression pairs. Thus, based on the AUC, the mas-o-menos classifier performs better on the validation set. Since both algorithms use the same 6 genes in this example, the outcome discrimination improvement is likely due to mas-o-menos averaging the effect of multiple genes while kTSP focuses only on pairwise gene comparisons.

Bibliography

ChatGPT. (2025, April 14). Grammar editing assistance and R code debugging. OpenAI. Retrieved from https://openai.com/chatgpt