Homework 2

Kent Codding

2025-02-28

For this homework we will be using the curatedOvarianData package. Consider as candidate biomarkers all possible gene expression measurements in the TCGA study. Then, use debulking as our binary phenotype outcome variable, with optimal debulking as a positive case. Furthermore, only use data from patients 1 through 400 for this assignment. Then, complete the following questions:

```
library(knitr)
```

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

```
library(curatedOvarianData)
```

```
## Loading required package: Biobase

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, saveRDS, setdiff,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
library(pROC)
```

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

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

##
## Attaching package: 'pROC'

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

```
data(TCGA_eset)
# Refer to Section 1 material if the above two lines have errors such as "there is no package called 'c
XX = as.matrix(cbind(exprs(TCGA_eset)))
YY = 1 * as.vector(pData(TCGA_eset)[,"debulking"]=="suboptimal")
XX = XX[,!is.na(YY)]
YY = YY[!is.na(YY)]
```

```
library(purrr)
```

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

```
gen_data = function(n, effect_a, effect_b) {
    p = rbeta(1, effect_a, effect_b)
    data = list(rbinom(n, 1, p))
    list(p=p, data=data)
}

do_bayes = function(x, prior_a, prior_b) {
    n_succ = sum(x$data[[1]])
    n_fail = length(x$data[[1]]) - n_succ
    x$post_a = prior_a + n_succ   #Beta binomial conjugacy
    x$post_b = prior_b + n_fail
    x
}

do_freq = function(x) {
    x$freq_est = mean(x$data[[1]])
    x
}

calc_se = function(x) {
    x$post_mean = x$post_a / (x$post_a + x$post_b)
    x$bayes_se = (x$p - x$post_mean)^2
    x$freq_se = (x$p - x$freq_est)^2
    x
}
set.seed(117)
tmp1 <- map(1:10,rnorm,n=10)
tmp1[[1]]
```

```
##  [1]  1.5763630  1.0886244  1.6925449 -2.0432846  0.8584076 -0.4142345
##  [7]  1.0824264  0.8063087  1.7030709  2.2114279
```

1)

Compute the three metrics (AUC, Fold Change, and Negative Log P-value) for biomarker discovery covered in class for each of the biomarkers. Use the CompScores function included in the starter code. Interpret what each metric is telling us.

```
library(ROCR)
```

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

```r
subs = 1:100 # subset of first 400 patients
XX = XX[,subs]
YY = YY[subs]

# Helper function to quickly calculate the AUC without worrying about the ROC curve
fast_auc = function(x, y) {
    # x <- XX[1,]
    # y <- YY
    rank_value = rank(x, ties.method="average")
    n_pos = sum(y == 1)
    rank_sum = sum(rank_value[y == 1])
    u_value = rank_sum - (n_pos * (n_pos + 1)) / 2
    auc = u_value / (n_pos * (length(x) - n_pos))
    if (auc >= 0.50) auc else 1.0 - auc
}

# Helper function for fast t-test
fast_t_test = function(x, y) {
    nx = length(x); mx = mean(x); vx = mean((x - mx)^2)/(nx-1);
    ny = length(y); my = mean(y); vy = mean((y - my)^2)/(ny-1);
    stderr <- sqrt(vx + vy)
    df <- stderr^4/(vx^2/(nx - 1) + vy^2/(ny -  1))
    tstat = (mx - my)/stderr
    list(p_value = 2 * pt(-abs(tstat), df),
         pooled_var = stderr^2)
}

# Calculate summary statistics for each gene in the expression matrix
CompScores = function(XX, YY) {
    NGenes = nrow(XX)
    # create empty vectors to store results in
    AUC = numeric(NGenes)
    nlpvalueT = numeric(NGenes)
    Var = numeric(NGenes)

    #set.seed(5118) # For replication purposes
    for (gg in 1:NGenes) { #gg is index
        # AUC
        AUC[gg] = fast_auc(XX[gg, ], YY)
        # Tests difference in means for XX under Y=1 and XX under Y=0
        # T-test negative log p-value
        t_test = fast_t_test(XX[gg, YY==1], XX[gg, YY==0])
        nlpvalueT[gg] = -log(t_test$p_value)
        Var[gg] = t_test$pooled_var
    }

    # Difference in means (aka fold change as data are on log scale)
    FoldChange = rowMeans(XX[, YY==1]) - rowMeans(XX[, YY==0])

    scores = data.frame(AUC=AUC, nlpvalueT=nlpvalueT,
                        FoldChange=FoldChange, Var=Var)

    return(scores)
}

Scores = CompScores(XX, YY)
```

Interpretation for the first gene in the dataset: A1CF

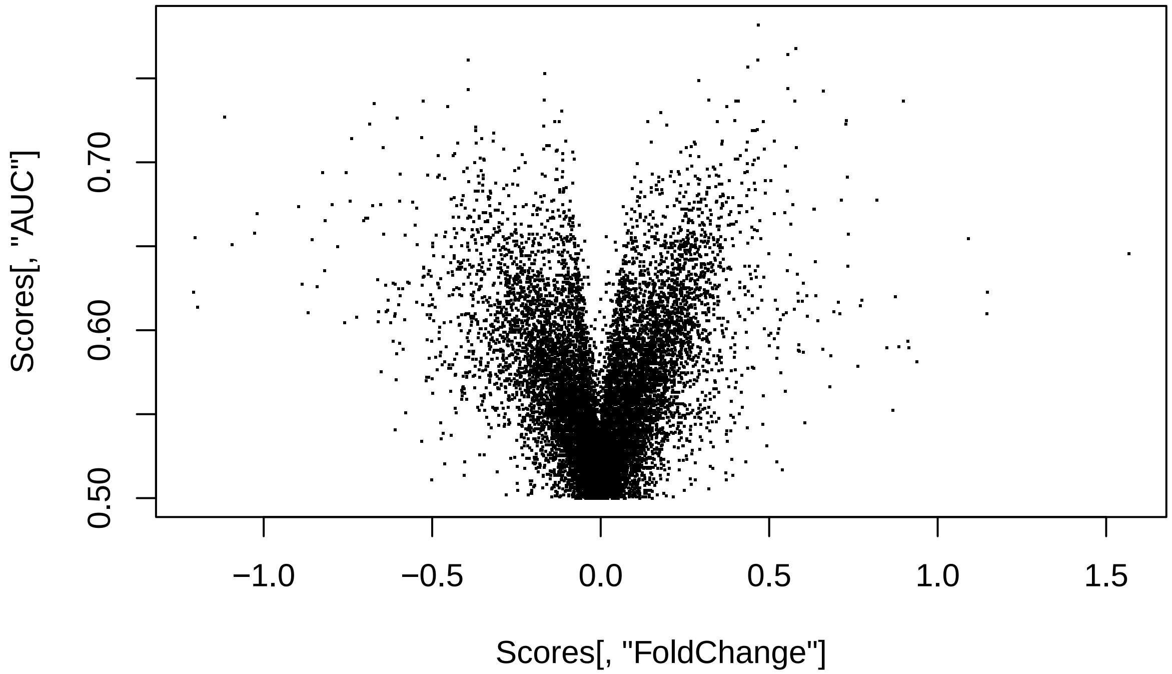
```
## 0.6077236 : the area under the Receiving Operator Characteristic curve,
##  which represents the ratio of true positives to false positives across thresholds k
##

## 2.769626 : the negative log of the probability that the difference between suboptimal versus
##  optimal group means of gene expression is due to random chance
##

## -0.07604677 : the mean increase in log gene expression of A1CF between the optimal debulking group
##  and suboptimal debulking group
##
```

2)

Using visualizations, compare and briefly summarize the three metrics two at a time. Then for each plot, describe a scenario or reason in which a biomarker may have a larger value for the one metric and a smaller value for the other metric. I.e., (under what scenarios might a biomarker to have a large NLP-value but a smaller fold change, absolute value wise? Or what might cause a biomarker to have a comparatively smaller NLP-value but a larger fold change?) You only need to consider one such comparison out of the two possible for each of the 3 plots.

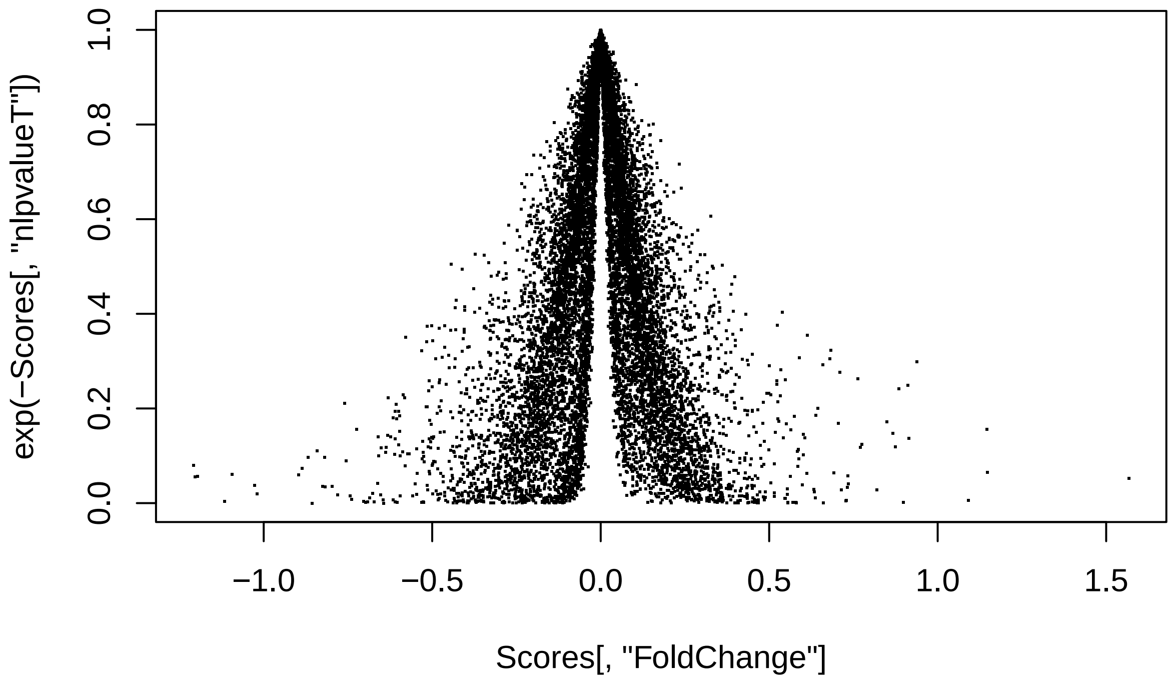
Plot of Fold Change vs. AUC

```r
plot(Scores[,"FoldChange"], Scores[,"AUC"], pch=".")
```



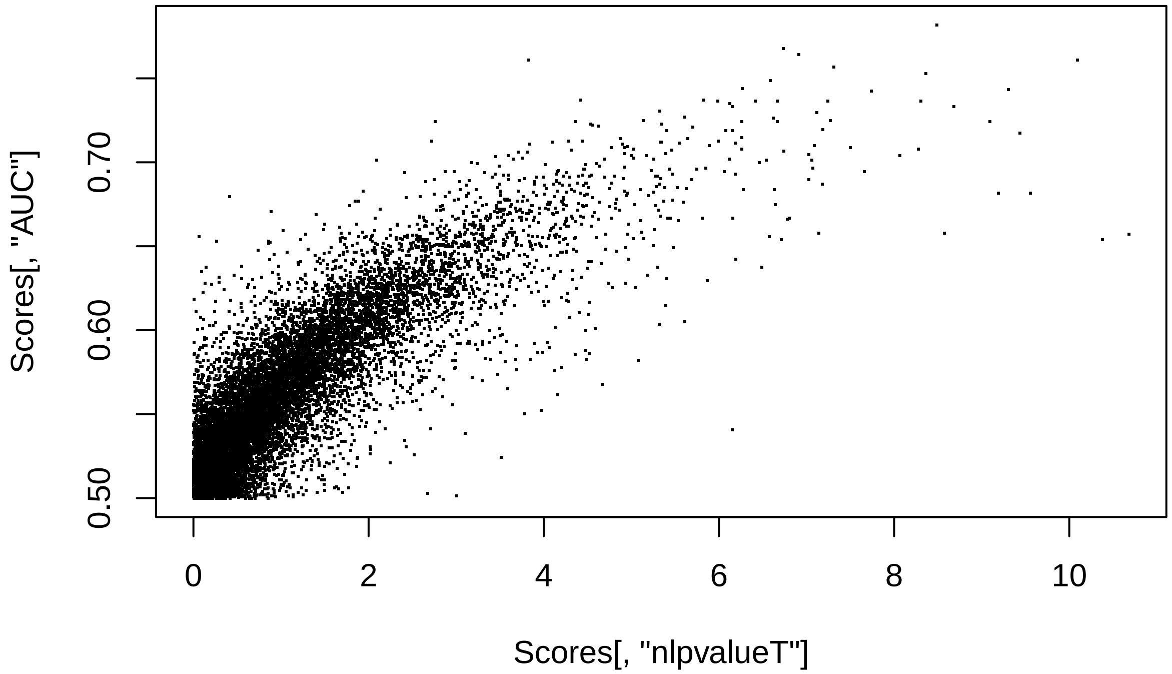
An area with small fold change and high AUC may have extremely small variance within suboptimal versus optimal debulking groups, providing for good discrimination even with a small difference in expression values.

```
plot(Scores[,"FoldChange"], exp(-Scores[,"nlpvalueT"]), pch=".")
```



For this plot, a point with high fold change and low nlp values could represent a biomarker with a large mean difference between suboptimal and optimal debulking groups and a large within-group variance. In this dataset, all biomarkers have the same sample size, but a smaller sample size could also hypothetically cause a high fold change and low nlp value.

```
plot(Scores[,"nlpvalueT"], Scores[,"AUC"], pch=".")
```

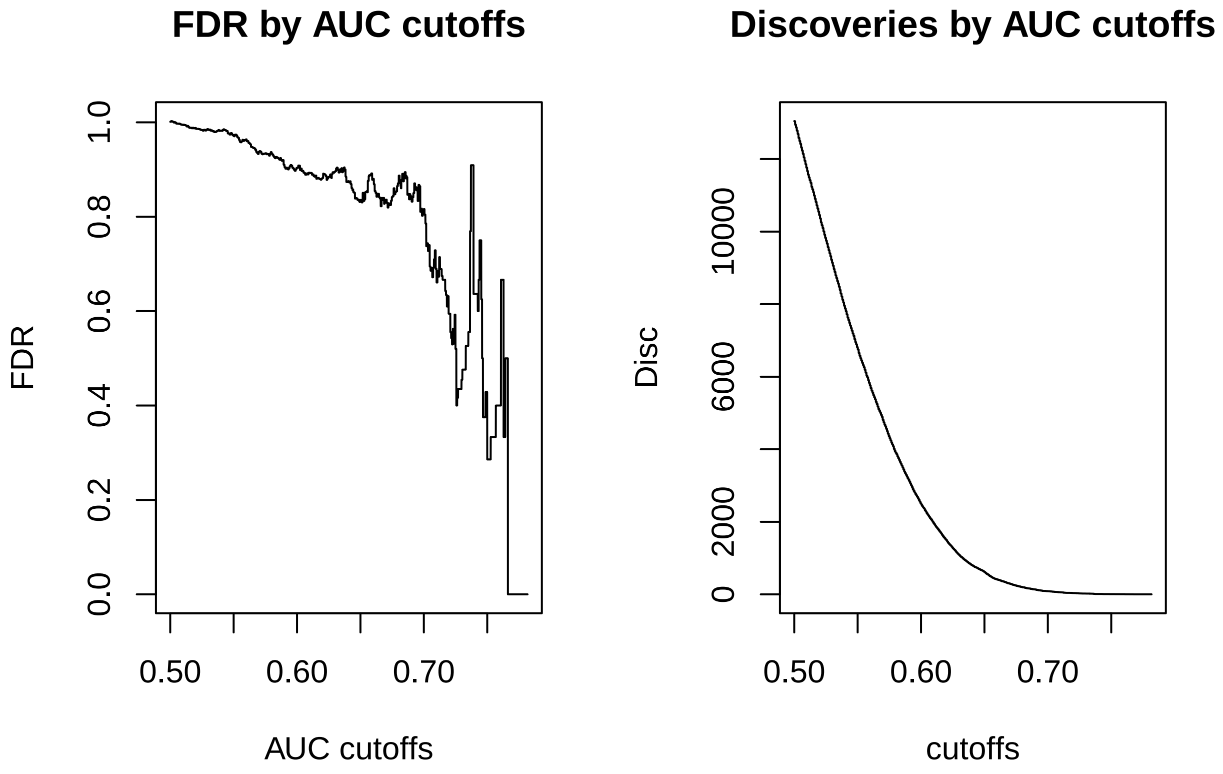


For a point with high negative log p value and low AUC, high variance and fold change could contribute to this since fold change is the numerator of $t^{\star}$. Additionally, outliers and heavy tails like those that we saw in the positive arm of CHD8 in homework 2 can allow statistical significance by skewing a fold change difference without improving the ranking performance, which is what AUC measures.

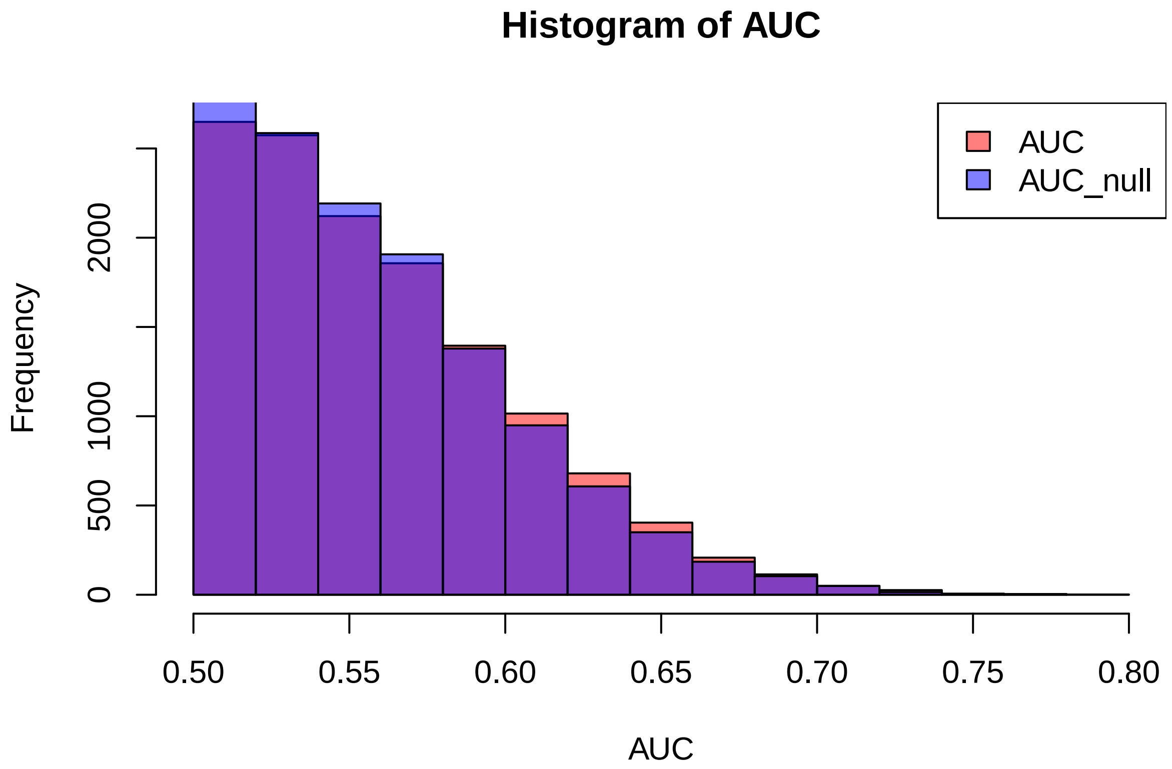
3)

Then, focusing on the AUC metric only, estimate and plot the False Discovery Rate (FDR) for an appropriately chosen range of cutoff values. Make sure your cutoff increments are sufficiently small to create enough detail. Discuss what cutoff or cutoffs for the AUC you would choose and justify your reasoning. (Hint: Consider trade-offs between the total number of discoveries and the FDR).

```r
set.seed(118)
par(mfrow = c(1,2))
YYNull = YY[sample(1:length(YY))]
scores_null = CompScores(XX, YYNull)

# focus on only AUC
AUC = Scores$AUC
AUC_null = scores_null$AUC

cutoffs <- seq(0.5, max(Scores$AUC), by = 0.00005)
FDR <- rep(0, length(cutoffs))
Disc <- rep(0, length(cutoffs))
tet <- rep(0, length(cutoffs))
for(i in 1:length(cutoffs)) {
    cutoff <- cutoffs[i]
    AUCvalueDisc <- 1*(AUC > cutoff)
    AUCvalueNull <- 1*(AUC_null > cutoff)
    Disc[i] <- sum(AUCvalueDisc)
    FDR[i] <- ifelse(sum(AUCvalueDisc) > 0 , sum(AUCvalueNull) / Disc[i], 0)
}

plot(cutoffs, FDR, type = "l",
     main = 'FDR by AUC cutoffs',
     xlab = 'AUC cutoffs')
plot(cutoffs, Disc, type = "l",
     main = 'Discoveries by AUC cutoffs')
```



```
hist(AUC, col = rgb(1, 0, 0, 0.5))
hist(AUC_null, col = rgb(0, 0, 1, 0.5), add = T)
legend("topright", legend = c("AUC", "AUC_null"),
       fill = c(rgb(1, 0, 0, 0.5), rgb(0, 0, 1, 0.5)))
```



In order to find the balance between Type I and Type II errors, it is important to choose a value that simultaneously limits the FDR without losing too many discoveries to Type II error. I would choose a cutoff of roughly 0.75. Unfortunately, we would lose many discoveries to Type II error in this case. But, with that being said, the above distribution of permuted versus actual AUCs suggest that, based on AUC alone, most of the variation from the AUC mean of 0.5 is actually noise and not a true biological difference between suboptimal versus optimal debulking groups. Thus, I think it is appropriate to hold a higher cutoff value to ensure that we reduce the FDR rate as much as possible, which starts at 1, representing equal ratio of false to actual discoveries that is exhibited in the overlaid histogram, and approaches 0 as the AUC approaches its max value.

```
## Although barely visible in the above histogram, the maximum value of AUC is:  0.7818428
##  and there are,  7 true discovery values above my chosen cutoff.
```

Lastly, even though the cutoff could be set to the maximum value of `AUC_null` to completely reduce the FDR to zero, that could be *overfitting* on this single permuted dataset, which typically is practiced by performing multiple permutations and taking the average.