doc-analysis-final

Zhaoliang Zhou

4/21/2022

```
library(glmnet)
library(coefplot)
library(ROCR)
library(gbm)
library(e1071)
library(nnet)
library(pls)
library("ridge")
library("sva")
library("car")
library("preprocessCore")
library("ROCR")
```

Data processing

```
setwd("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final pro
```

```
source("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final pr



source("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final pr

source("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final pr

source("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final pr
```

```
load(file="C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final

load(file="C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mining/final
```

```
###################################################
### code chunk number 5: docetaxelBreastCancer.Snw:42-46
###################################################
sensDoce <- read.csv("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data m
as.is=TRUE)
doceic50s <- sensDoce$"IC.50"
names(doceic50s) <- sensDoce$"Cell.Line.Name"
```

```
###################################################
### code chunk number 6: docetaxelBreastCancer.Snw:50-58
###################################################
```

```
pData <- read.delim("C:/Users/leonz/Desktop/UMN/Spring 2022/PubH 7475 - Statistical Learnign and Data mi
```

```
pDataUnique <- pData[pData$Source.Name %in% names(which(table(pData$Source.Name)
== 1)), ]

rownames(pDataUnique) <- pDataUnique$Source.Name
commonCellLines <- rownames(pDataUnique)[rownames(pDataUnique) %in% names(doceic50s)]
pDataUniqueOrd <- pDataUnique[commonCellLines, ]
doceic50sOrd <- doceic50s[commonCellLines]
trainDataOrd <- gdsc_brainarray_syms[, pDataUniqueOrd$"Array.Data.File"]
```

ridge (paper)

```
predictedSensitivity_ridge1 <- calcPhenotype_ridge(doceVivoNorm_syms, trainDataOrd, doceic50sOrd,
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used

## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  8239  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data

##
## Fitting Ridge Regression model... Done
##
## Calculating predicted phenotype...

## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used

## Done
```

```
t.test(predictedSensitivity_ridge1[1:10], predictedSensitivity_ridge1[11:24], alternative="less")
```

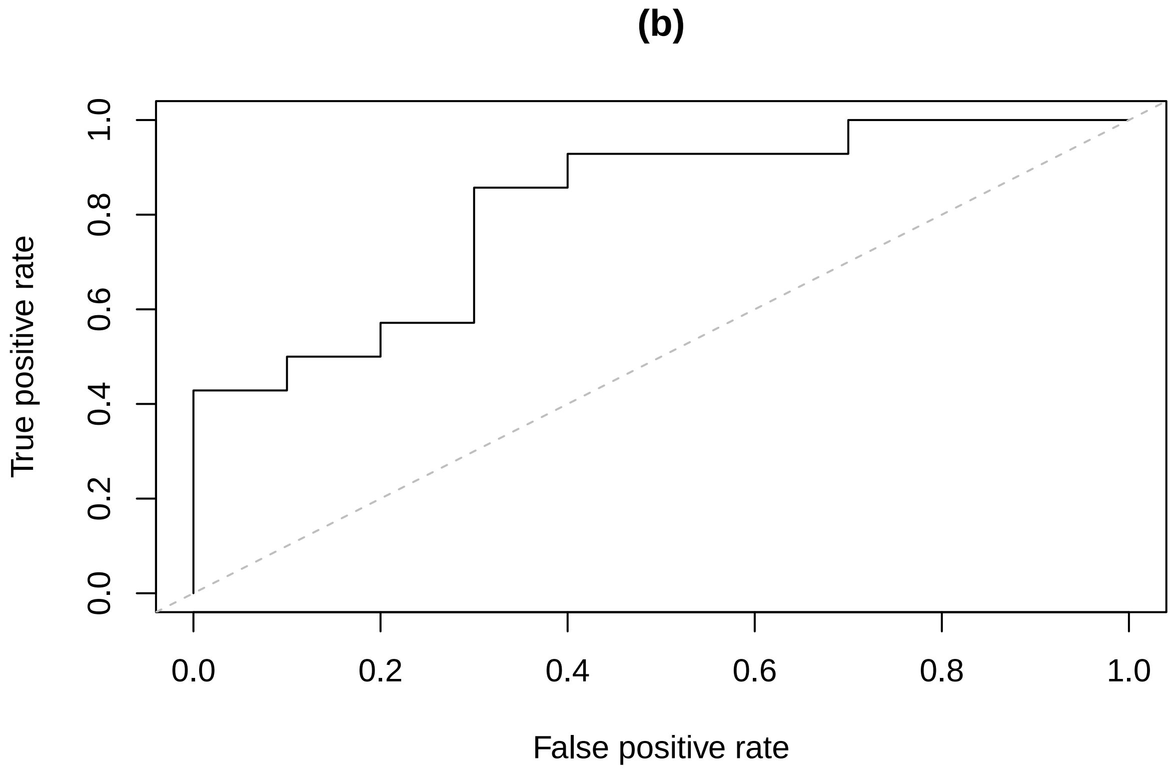
```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_ridge1[1:10] and predictedSensitivity_ridge1[11:24]
## t = -2.8835, df = 21.772, p-value = 0.004342
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
```

```
##        -Inf -0.1775691
## sample estimates:
## mean of x mean of y
## -5.585158 -5.145871
```

```
pred_ridge1 <- prediction(predictedSensitivity_ridge1, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_ridge1 <- performance(pred_ridge1, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_ridge1, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.814285714285714"
```

```
plot(perf_ridge1, main="(b)")
abline(0, 1, col="grey", lty=2)
```



ridge (glmnet)

```
set.seed(99)
predictedSensitivity_ridge2 <- calcPhenotype_ridge_glmnet(doceVivoNorm_syms, trainDataOrd, doceic50sOrd
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used
```

```
## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  8239  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data

##
## Fitting glmnet ridge Regression model... Done
##
## Calculating predicted phenotype...
## best lambda min is: 11.13241

## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used

## Done
```

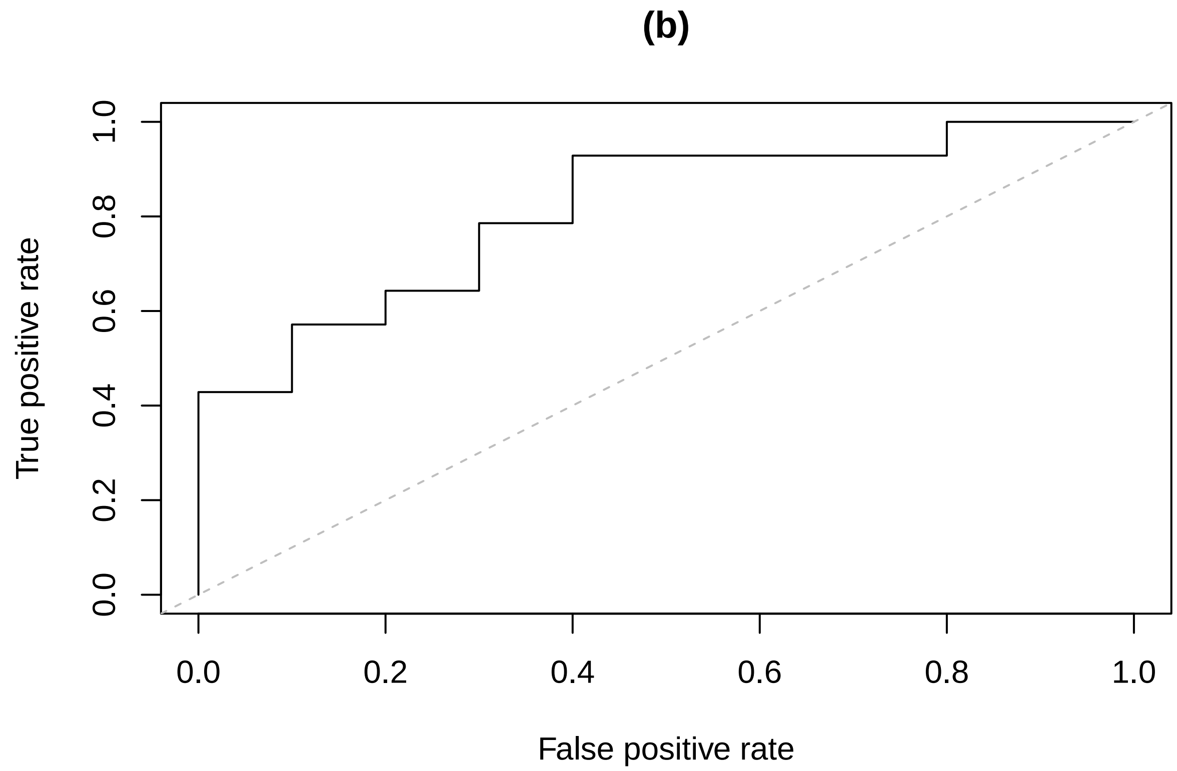
```
t.test(predictedSensitivity_ridge2[1:10], predictedSensitivity_ridge2[11:24], alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_ridge2[1:10] and predictedSensitivity_ridge2[11:24]
## t = -2.8061, df = 21.81, p-value = 0.005173
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf -0.1465096
## sample estimates:
## mean of x mean of y
## -5.552185 -5.174418
```

```
pred_ridge2 <- prediction(predictedSensitivity_ridge2, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))

perf_ridge2 <- performance(pred_ridge2, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_ridge2, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.814285714285714"
```

```
plot(perf_ridge2, main="(b)")
abline(0, 1, col="grey", lty=2)
```



lasso

```
set.seed(99)
predictedSensitivity_lasso <- calcPhenotype_lasso(doceVivoNorm_syms, trainDataOrd, doceic50sOrd,
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used

## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  8239  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data
```

```
##
## Fitting glmnet lasso Regression model... Done
##
## Calculating predicted phenotype...
## best lambda min is: 0.02236761

## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used

## Done
```

```
predictedSensitivity_lasso$preds
```

```
##                    s0
## GSM4903.CEL -6.449850
## GSM4907.CEL -5.973250
## GSM4908.CEL -5.948351
## GSM4914.CEL -4.727162
## GSM4915.CEL -5.806864
## GSM4917.CEL -5.360147
## GSM4919.CEL -6.313203
## GSM4920.CEL -5.402329
## GSM4921.CEL -6.056590
## GSM4923.CEL -5.521219
## GSM4901.CEL -4.589387
## GSM4902.CEL -4.854067
## GSM4904.CEL -5.353856
## GSM4905.CEL -4.045032
## GSM4906.CEL -5.814619
## GSM4909.CEL -3.356327
## GSM4910.CEL -4.906997
## GSM4911.CEL -5.327233
## GSM4912.CEL -5.211067
## GSM4913.CEL -5.462324
## GSM4916.CEL -5.842630
## GSM4918.CEL -5.011206
## GSM4922.CEL -5.218598
## GSM4924.CEL -4.958060
```

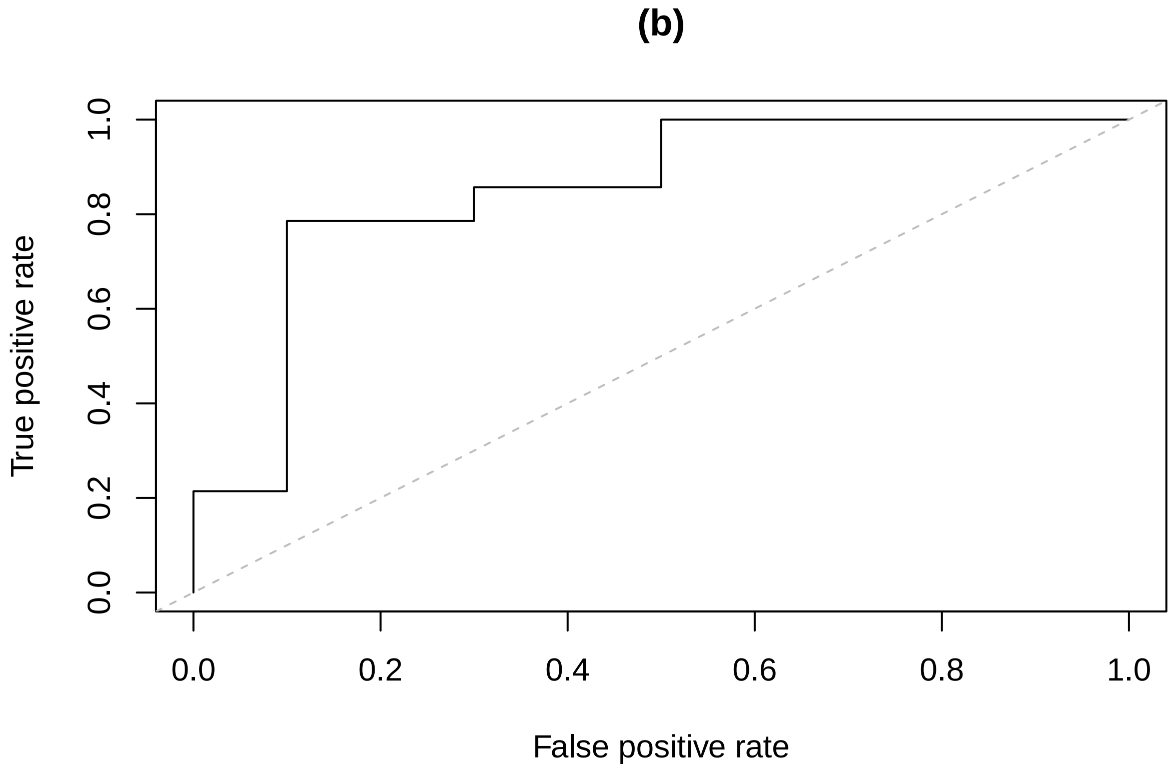
```
t.test(predictedSensitivity_lasso$preds[1:10], predictedSensitivity_lasso$preds[11:24], alternative="les
```

```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_lasso$preds[1:10] and predictedSensitivity_lasso$preds[11:24]
## t = -3.1604, df = 21.809, p-value = 0.002284
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.3466203
## sample estimates:
## mean of x mean of y
## -5.755896 -4.996529
```

```
pred_lasso <- prediction(predictedSensitivity_lasso$preds, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_lasso <- performance(pred_lasso, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_lasso, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.85"
```

```
plot(perf_lasso, main="(b)")
abline(0, 1, col="grey", lty=2)
```



```
# print top 5 non-zero coeffs
head(predictedSensitivity_lasso$coeffs)
```

```
##      features        coefs
## 1 (Intercept)  1.482967121
## 2       KRT17 -0.010212719
## 3       DFNA5 -0.011152406
## 4       DUSP6  0.004313232
## 5       GSTP1 -0.005576963
## 6      SLC1A3 -0.001985691
```

```
cat("the number of non-zero coefficients is:", length(predictedSensitivity_lasso$coeffs$coefs))
```

```
## the number of non-zero coefficients is: 104
```

```
predictedSensitivity_lasso$coeffs$features
```

```
##   [1] "(Intercept)"  "KRT17"        "DFNA5"        "DUSP6"        "GSTP1"
##   [6] "SLC1A3"       "INHBA"        "GFPT2"        "PHACTR1"      "DEFB1"
##  [11] "MYO1B"        "NOV"          "SRPX2"        "MAL"          "MPDZ"
##  [16] "RORA"         "SERPINH1"     "VSNL1"        "COL16A1"      "TNNC1"
##  [21] "SYCP2"        "CPM"          "ARHGEF10"     "ABCB1"        "TTC39A"
##  [26] "MAOB"         "SMAD7"        "FOXO1"        "LYST"         "ME3"
##  [31] "GLI3"         "GALC"         "GLS"          "CEACAM1"      "BID"
##  [36] "ACTN4"        "COL9A2"       "SHC2"         "CNTN1"        "CRH"
##  [41] "RTN2"         "PKP4"         "AARS"         "RPP40"        "KIF3B"
##  [46] "PKM"          "TRAF1"        "CFDP1"        "PTPRR"        "DEAF1"
##  [51] "RUNDC3B"      "TMEM41B"      "SPG11"        "KCNB1"        "STARD3"
##  [56] "SP140"        "CMAHP"        "ATP8A1"       "PTGER4"       "UBR4"
##  [61] "CAMTA1"       "CDS2"         "LOC100506963" "CALM1"        "KBTBD2"
##  [66] "SERPINB13"    "BCL2L1"       "HOXA2"        "BCAT2"        "TNNI2"
##  [71] "STOML2"       "SND1-IT1"     "CLOCK"        "LUZP1"        "ST8SIA1"
##  [76] "MBD1"         "PHB2"         "SKI"          "NIT1"         "PDHB"
##  [81] "TAC1"         "MYBPC1"       "ARID4A"       "STX16"        "GOLGB1"
##  [86] "PDIA2"        "MAML1"        "HSPB3"        "RABL3"        "PPP1R7"
##  [91] "ITGAX"        "YTHDF3"       "WDR82"        "IGSF1"        "CD226"
##  [96] "PLAA"         "REN"          "TTTY15"       "IL11"         "TOX4"
## [101] "NPR1"         "FOSL2"        "KXD1"         "AMPD3"
```

elestic net

```
set.seed(99)
predictedSensitivity_enet <- calcPhenotype_enet(doceVivoNorm_syms, trainDataOrd, doceic50sOrd,
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used
```

```
## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used
```

```
## 
##  8239  gene identifiers overlap between the supplied expression matrices...
## 
```

```
## Found2batches
```

```
## Adjusting for0covariate(s) or covariate level(s)
```

```
## Standardizing Data across genes
```

```
## Fitting L/S model and finding priors
```

```
## Finding parametric adjustments
```

```
## Adjusting the Data
```

```
## 
## Fitting glmnet enet Regression model... Done
## 
## Calculating predicted phenotype...
## best lambda min is: 0.04473521
```

```
## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used
```

```
## Done
```

```
t.test(predictedSensitivity_enet[1:10], predictedSensitivity_enet[11:24], alternative="less")
```

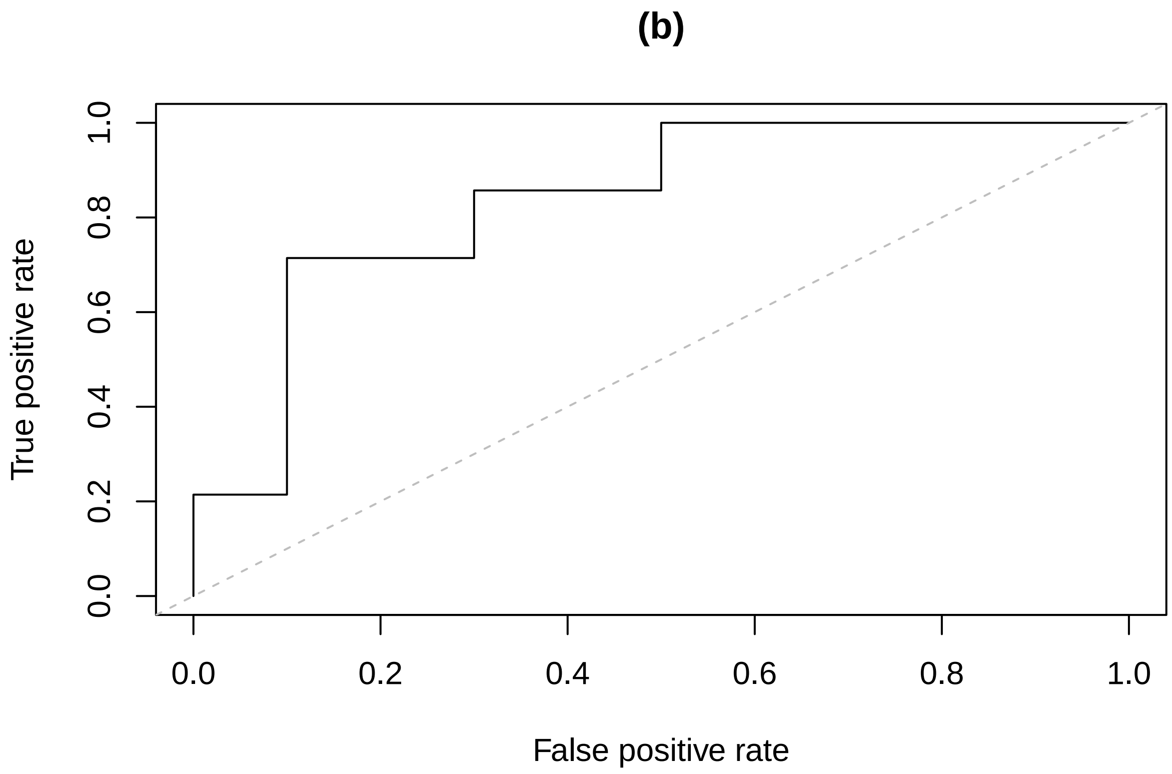
```
## 
```

```
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_enet[1:10] and predictedSensitivity_enet[11:24]
## t = -3.0131, df = 21.669, p-value = 0.003234
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.2883759
## sample estimates:
## mean of x mean of y
## -5.709701 -5.038639
```

```
pred_enet <- prediction(predictedSensitivity_enet, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_enet <- performance(pred_enet, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_enet, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.835714285714286"
```

```
plot(perf_enet, main="(b)")
abline(0, 1, col="grey", lty=2)
```



SVM

```
set.seed(99)
predictedSensitivity_svm <- calcPhenotype_svm(doceVivoNorm_syms, trainDataOrd, doceic50sOrd,
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used

## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  8239  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data

##
## Fitting svm Regression model... Done
##
## Calculating predicted phenotype... best gamma 0.000152952 best cost 1

## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used

## Done
```

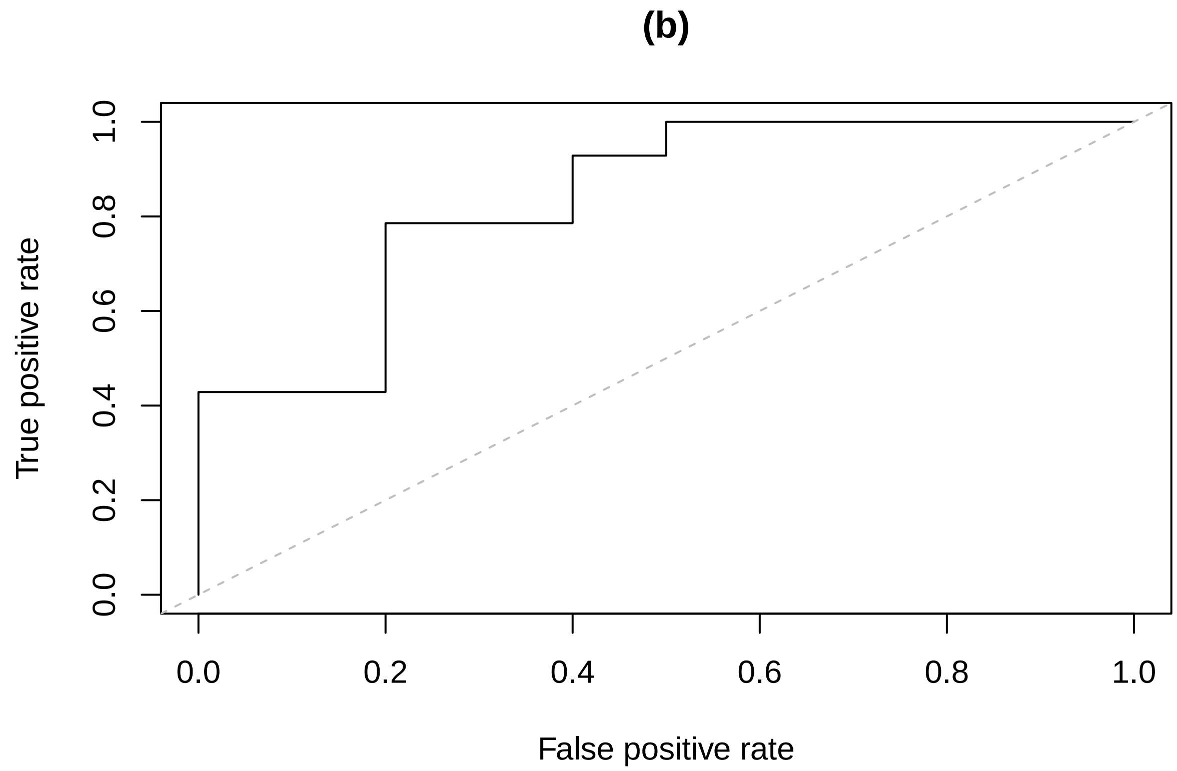
```
t.test(predictedSensitivity_svm[1:10], predictedSensitivity_svm[11:24], alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_svm[1:10] and predictedSensitivity_svm[11:24]
## t = -3.1038, df = 21.577, p-value = 0.00263
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.4683334
## sample estimates:
## mean of x mean of y
## -5.879156 -4.829734
```

```
pred_svm <- prediction(predictedSensitivity_svm, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_svm <- performance(pred_svm, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_svm, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.835714285714286"
```

```
plot(perf_svm, main="(b)")
abline(0, 1, col="grey", lty=2)
```



ANN

subset the trainDataOrd to those selected by LASSO

```
# lasso selected variables
lasso_select_vars <- c(predictedSensitivity_lasso$coeffs$features)

# remove intercept
lasso_select_vars <-  lasso_select_vars[-1]
lasso_select_vars
```

```
##  [1] "KRT17"        "DFNA5"        "DUSP6"        "GSTP1"        "SLC1A3"
##  [6] "INHBA"        "GFPT2"        "PHACTR1"      "DEFB1"        "MYO1B"
## [11] "NOV"          "SRPX2"        "MAL"          "MPDZ"         "RORA"
## [16] "SERPINH1"     "VSNL1"        "COL16A1"      "TNNC1"        "SYCP2"
## [21] "CPM"          "ARHGEF10"     "ABCB1"        "TTC39A"       "MAOB"
## [26] "SMAD7"        "FOXO1"        "LYST"         "ME3"          "GLI3"
## [31] "GALC"         "GLS"          "CEACAM1"      "BID"          "ACTN4"
## [36] "COL9A2"       "SHC2"         "CNTN1"        "CRH"          "RTN2"
## [41] "PKP4"         "AARS"         "RPP40"        "KIF3B"        "PKM"
## [46] "TRAF1"        "CFDP1"        "PTPRR"        "DEAF1"        "RUNDC3B"
## [51] "TMEM41B"      "SPG11"        "KCNB1"        "STARD3"       "SP140"
## [56] "CMAHP"        "ATP8A1"       "PTGER4"       "UBR4"         "CAMTA1"
## [61] "CDS2"         "LOC100506963" "CALM1"        "KBTBD2"       "SERPINB13"
## [66] "BCL2L1"       "HOXA2"        "BCAT2"        "TNNI2"        "STOML2"
## [71] "SND1-IT1"     "CLOCK"        "LUZP1"        "ST8SIA1"      "MBD1"
##  [76] "PHB2"         "SKI"          "NIT1"         "PDHB"         "TAC1"
##  [81] "MYBPC1"       "ARID4A"       "STX16"        "GOLGB1"       "PDIA2"
##  [86] "MAML1"        "HSPB3"        "RABL3"        "PPP1R7"       "ITGAX"
##  [91] "YTHDF3"       "WDR82"        "IGSF1"        "CD226"        "PLAA"
##  [96] "REN"          "TTTY15"       "IL11"         "TOX4"         "NPR1"
## [101] "FOSL2"        "KXD1"         "AMPD3"
```

```
length(lasso_select_vars)
```

```
## [1] 103
```

```
trainDataOrd_subset <- trainDataOrd[rownames(trainDataOrd) %in% lasso_select_vars, ]  # Extract rows fr
dim(trainDataOrd_subset)                                                # Print data frame subset
```

```
## [1] 103 482
```

```
set.seed(99)
predictedSensitivity_ann <- calcPhenotype_ann(doceVivoNorm_syms, trainDataOrd_subset, doceic50sOrd,
selection=1)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used

## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  103  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data

##
## Fitting ANN Regression model... # weights:  85
## initial  value 2773.042457
## iter  10 value 57.027060
## iter  20 value 29.331598
## iter  30 value 27.053713
## iter  40 value 25.886597
## iter  50 value 23.683072
## iter  60 value 22.294595
## iter  70 value 21.794830
## iter  80 value 21.456706
## iter  90 value 21.324820
## iter 100 value 21.237957
## iter 110 value 21.173157
## iter 120 value 21.104221
## iter 130 value 21.075449
## iter 140 value 21.056351
## iter 150 value 21.031580
## iter 160 value 21.015922
## iter 170 value 21.012346
## iter 180 value 21.008744
## iter 190 value 21.008698
## final  value 21.008681
## converged
## Done
##
## Calculating predicted phenotype... /nbest size is: 1
```

```
## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used
```

```
## Done
```

```
predictedSensitivity_ann
```

```
##                   [,1]
## GSM4903.CEL -7.670590
## GSM4907.CEL -6.247696
## GSM4908.CEL -5.937449
## GSM4914.CEL -4.939728
## GSM4915.CEL -6.618077
## GSM4917.CEL -5.050923
## GSM4919.CEL -7.109106
## GSM4920.CEL -5.590867
## GSM4921.CEL -6.544139
## GSM4923.CEL -5.425384
## GSM4901.CEL -3.106459
## GSM4902.CEL -3.389602
## GSM4904.CEL -5.445045
## GSM4905.CEL -2.424021
## GSM4906.CEL -5.379297
## GSM4909.CEL -2.105535
## GSM4910.CEL -5.650974
## GSM4911.CEL -6.198831
## GSM4912.CEL -5.141844
## GSM4913.CEL -5.140597
## GSM4916.CEL -5.977018
## GSM4918.CEL -3.810158
## GSM4922.CEL -5.177073
## GSM4924.CEL -4.870464
```

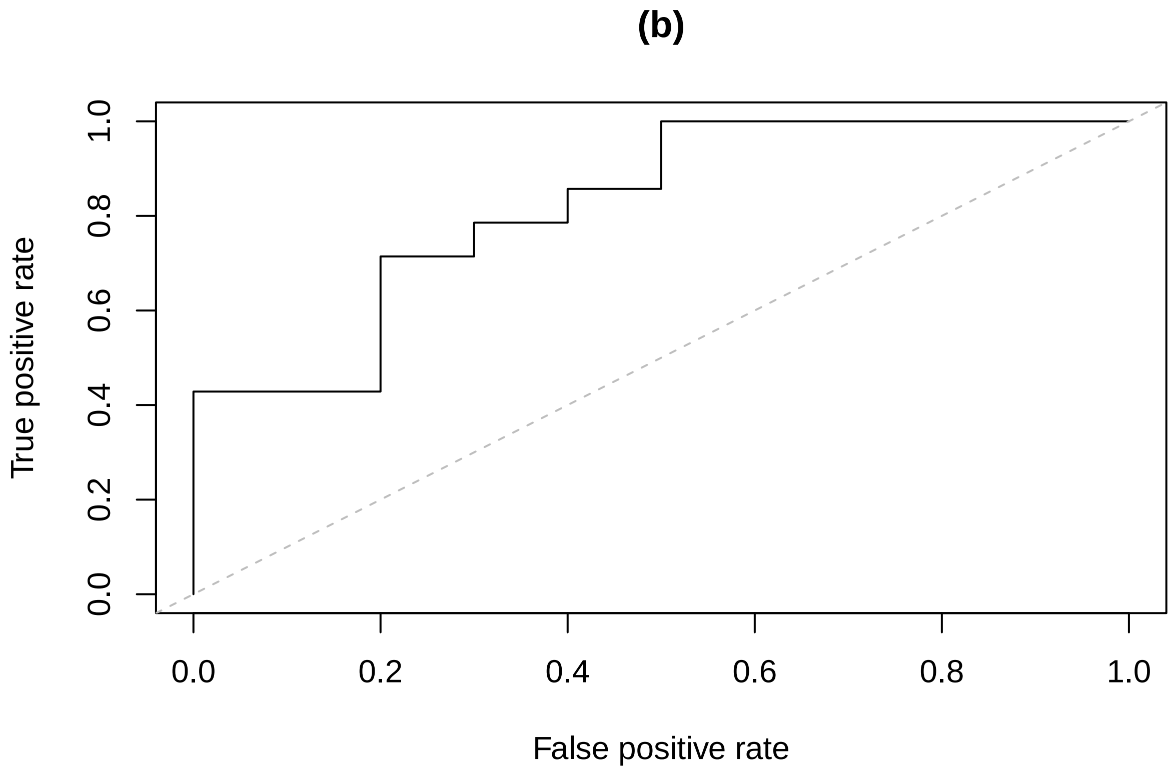
```
t.test(predictedSensitivity_ann[1:10], predictedSensitivity_ann[11:24], alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_ann[1:10] and predictedSensitivity_ann[11:24]
## t = -3.4213, df = 21.947, p-value = 0.001225
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.7744916
## sample estimates:
## mean of x mean of y
## -6.113396 -4.558351
```

```
pred_ann <- prediction(predictedSensitivity_ann, c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_ann <- performance(pred_ann, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_ann, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.821428571428571"
```

```
plot(perf_ann, main="(b)")
abline(0, 1, col="grey", lty=2)
```

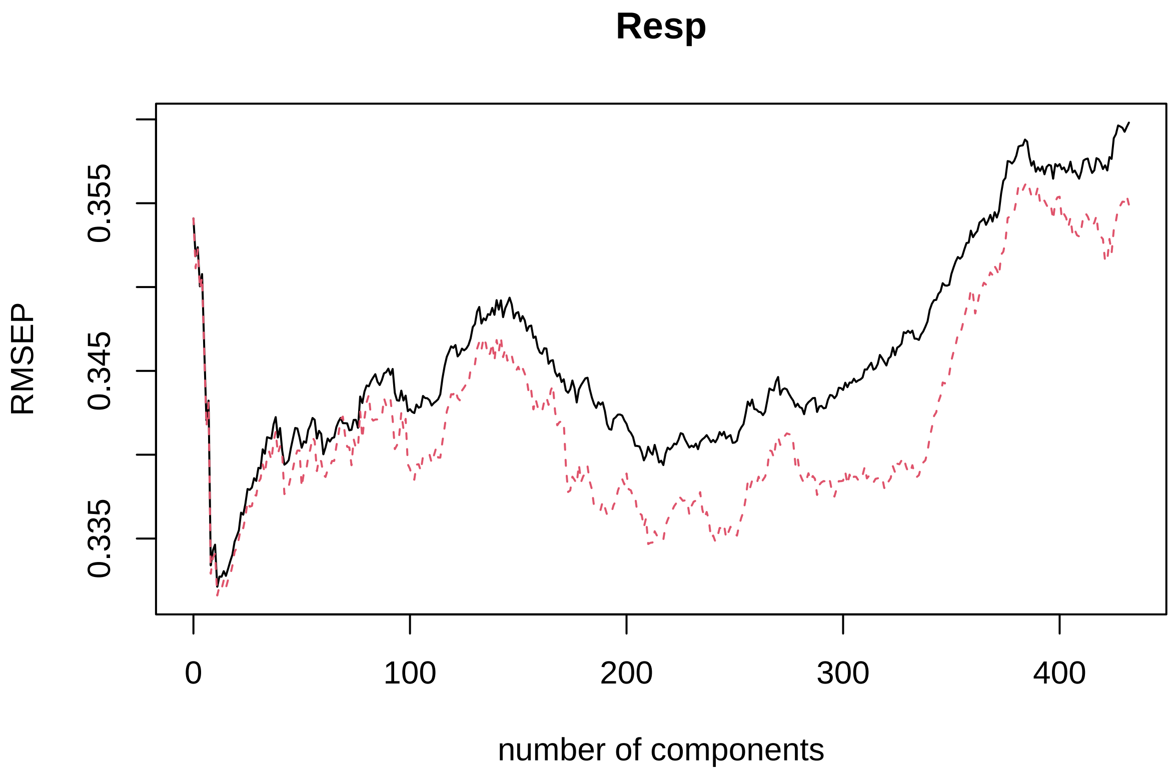


PCR

```
set.seed(99)
predictedSensitivity_pcr <- calcPhenotype_pcr(doceVivoNorm_syms,
                                              trainDataOrd,
                                              doceic50sOrd,
                                              selection=1,
                                              Ncomp=10)
```

```
## Warning in if (class(testExprData) != "matrix") stop("ERROR: \"testExprData\"
## must be a matrix."): the condition has length > 1 and only the first element
## will be used
```

```
## Warning in if (class(trainingExprData) != "matrix") stop("ERROR:
## \"trainingExprData\" must be a matrix."): the condition has length > 1 and only
## the first element will be used

##
##  8239  gene identifiers overlap between the supplied expression matrices...
##

## Found2batches

## Adjusting for0covariate(s) or covariate level(s)

## Standardizing Data across genes

## Fitting L/S model and finding priors

## Finding parametric adjustments

## Adjusting the Data

##
## Fitting PCR Regression model... Done
##
## Calculating predicted phenotype... /n

## Warning in if (class(homData$test) == "numeric") {: the condition has length > 1
## and only the first element will be used
```



```
## Done
```

```
predictedSensitivity_pcr
```

```
## , , 10 comps
##
```

```
##                  Resp
## GSM4903.CEL -5.716404
## GSM4907.CEL -5.761135
## GSM4908.CEL -5.573619
## GSM4914.CEL -5.129185
## GSM4915.CEL -5.401530
## GSM4917.CEL -5.367297
## GSM4919.CEL -5.561711
## GSM4920.CEL -5.326728
## GSM4921.CEL -5.608927
## GSM4923.CEL -5.296237
## GSM4901.CEL -5.079388
## GSM4902.CEL -5.474484
## GSM4904.CEL -5.417295
## GSM4905.CEL -4.924616
## GSM4906.CEL -5.992417
## GSM4909.CEL -4.557812
## GSM4910.CEL -4.898645
## GSM4911.CEL -5.216310
## GSM4912.CEL -5.366141
## GSM4913.CEL -5.309044
## GSM4916.CEL -5.700080
## GSM4918.CEL -5.065483
## GSM4922.CEL -5.143478
## GSM4924.CEL -5.074535
```

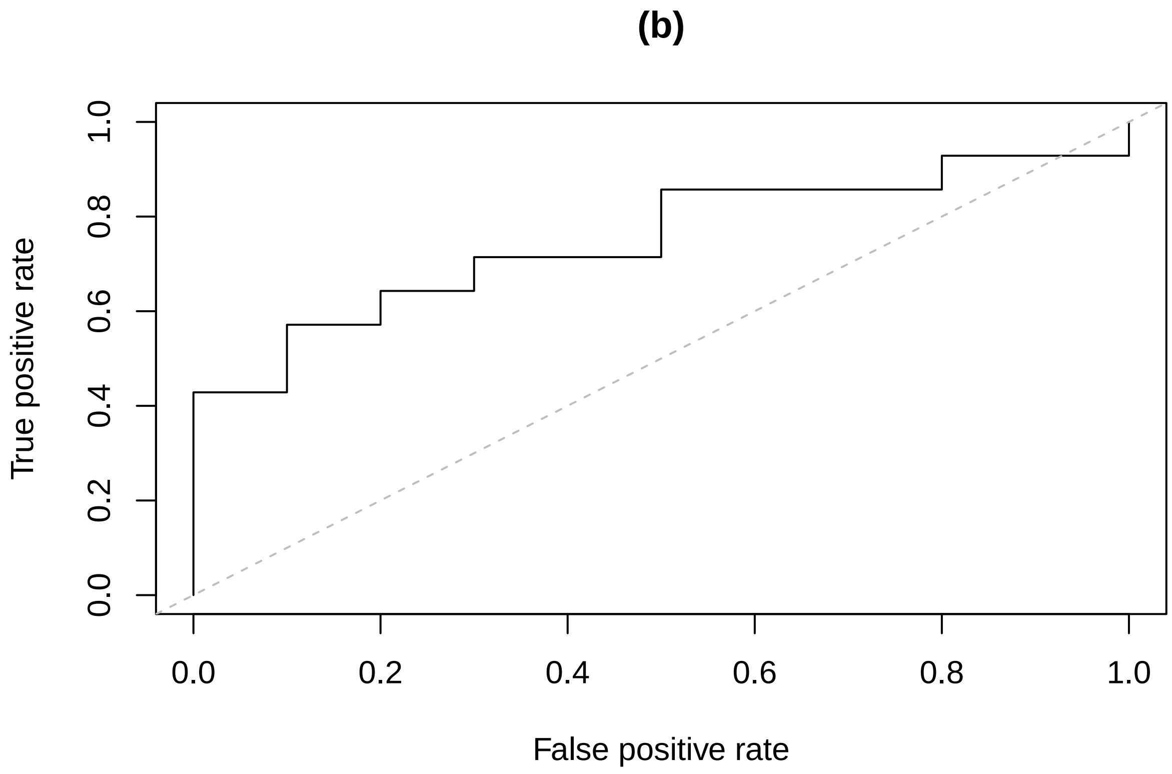
```
t.test(predictedSensitivity_pcr[1:10], predictedSensitivity_pcr[11:24], alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  predictedSensitivity_pcr[1:10] and predictedSensitivity_pcr[11:24]
## t = -2.1282, df = 21.116, p-value = 0.02263
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.04682115
## sample estimates:
## mean of x mean of y
## -5.474277 -5.229980
```

```
pred_pcr <- prediction(predictedSensitivity_pcr[1:24], c(rep("sens", 10), rep("res", 14)),
label.ordering=c("sens", "res"))
perf_pcr <- performance(pred_pcr, measure = "tpr", x.measure = "fpr")
print(paste("AUC:", performance(pred_pcr, measure = "auc")@"y.values"[[1]]))
```

```
## [1] "AUC: 0.75"
```

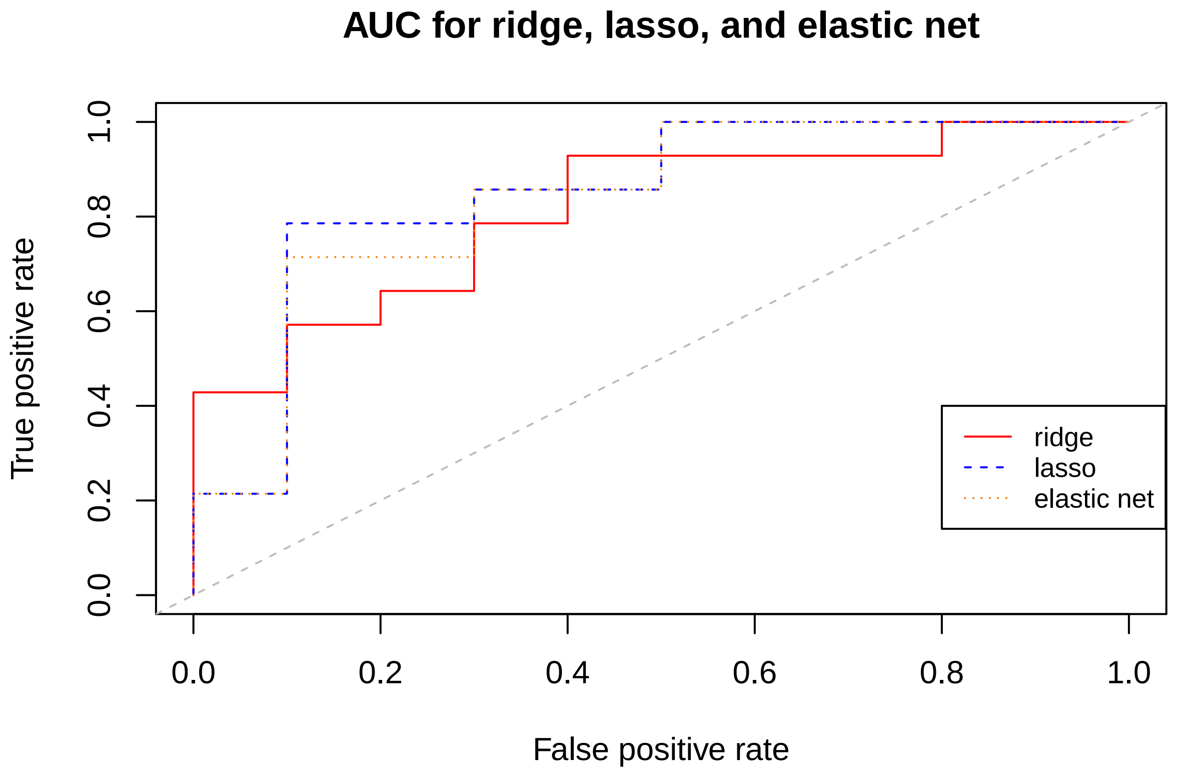
```
plot(perf_pcr, main="(b)")
abline(0, 1, col="grey", lty=2)
```



ROC curves plots

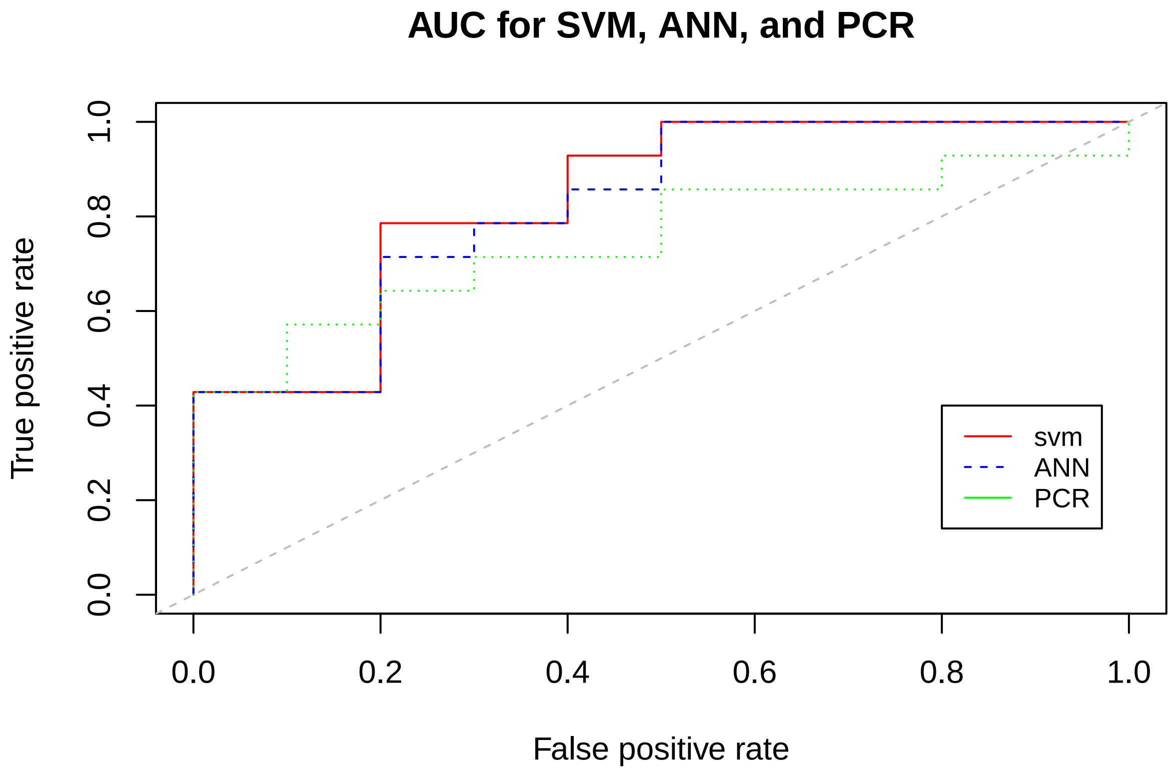
1st set of plots: ridge, lasso, elestic net

```
plot(perf_ridge2 ,main="AUC for ridge, lasso, and elastic net", col="red", lty=1)
lines(perf_lasso@x.values[[1]], perf_lasso@y.values[[1]], col="blue", lty=2)
lines(perf_enet@x.values[[1]], perf_enet@y.values[[1]], col="darkorange1", lty=3)
abline(0, 1, col="grey", lty=2)
legend(0.8, 0.4, legend=c("ridge", "lasso","elastic net"),
       col=c("red", "blue","darkorange1"), lty=1:3, cex=0.8)
```



part 2: svm, ann, pcr

```
plot(perf_svm ,main="AUC for SVM, ANN, and PCR", col="red", lty=1)
lines(perf_ann@x.values[[1]], perf_ann@y.values[[1]], col="blue", lty=2)
lines(perf_pcr@x.values[[1]], perf_pcr@y.values[[1]], col="green", lty=3)
abline(0, 1, col="grey", lty=2)
legend(0.8, 0.4, legend=c("svm", "ANN","PCR"),
       col=c("red", "blue","green"), lty=1:2, cex=0.8)
```


AUC for SVM, ANN, and PCR
True positive rate
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
svm
ANN
PCR