Homework 3

Shiyu Wang

February 23, 2017

Setting up

```
# Read data
data= read.table("http://www-stat.stanford.edu/~tibs/ElemStatLearn/datasets/SAheart.data",
                 sep=",",head=T,row.names=1)
```

Question 1

Part A

```
set.seed(1)
sub <- sample(nrow(data), floor(nrow(data)/2))
train<-data[sub,]
valid<-data[-sub,]
head(train)
```

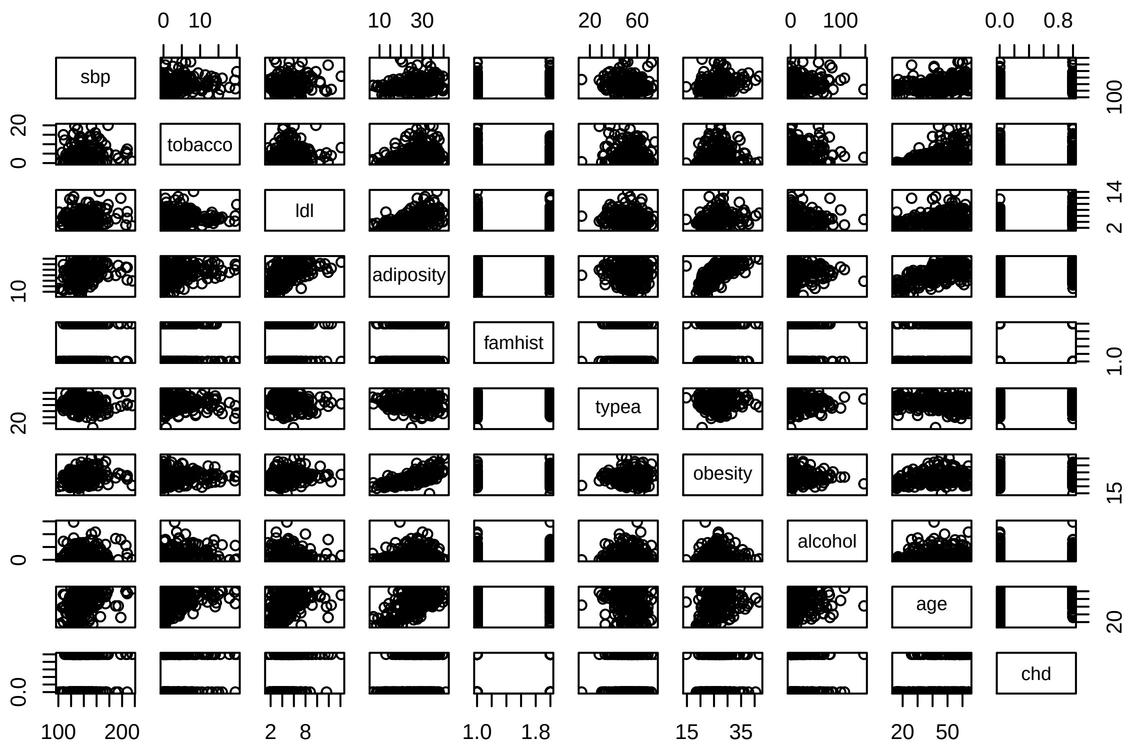
```
##     sbp tobacco  ldl adiposity famhist typea obesity alcohol age chd
## 123 120    0.00 5.01     26.13  Absent    64   26.21   12.24  33   0
## 172 118    0.75 2.58     20.25  Absent    59   24.46    0.00  32   0
## 265 136    5.00 4.19     23.99 Present    68   27.80   25.86  35   0
## 418 158   16.00 5.56     29.35  Absent    36   25.92   58.32  60   0
## 93  143    0.46 2.40     22.87  Absent    62   29.17   15.43  29   0
## 412 178   20.00 9.78     33.55  Absent    37   27.29    2.88  62   1
```

Part B

```
# numeric categorical column
train$famhist = as.numeric(train$famhist)
valid$famhist = as.numeric(valid$famhist)
## One-variable summary
summary(train)
```

```
##       sbp           tobacco            ldl           adiposity
##  Min.   :101.0   Min.   : 0.000   Min.   : 1.550   Min.   : 6.74
##  1st Qu.:126.0   1st Qu.: 0.150   1st Qu.: 3.505   1st Qu.:21.45
##  Median :136.0   Median : 2.550   Median : 4.640   Median :27.58
##  Mean   :139.2   Mean   : 3.852   Mean   : 4.940   Mean   :26.57
##  3rd Qu.:149.0   3rd Qu.: 5.995   3rd Qu.: 5.905   3rd Qu.:32.52
##  Max.   :216.0   Max.   :20.000   Max.   :14.160   Max.   :41.05
##     famhist          typea          obesity         alcohol
##  Min.   :1.000   Min.   :13.0   Min.   :14.70   Min.   :  0.00
##  1st Qu.:1.000   1st Qu.:48.0   1st Qu.:23.65   1st Qu.:  0.00
##  Median :1.000   Median :53.0   Median :26.09   Median :  7.61
##  Mean   :1.442   Mean   :53.1   Mean   :26.31   Mean   : 17.01
##  3rd Qu.:2.000   3rd Qu.:60.0   3rd Qu.:28.64   3rd Qu.: 24.68
##  Max.   :2.000   Max.   :75.0   Max.   :41.76   Max.   :147.19
##       age             chd
##  Min.   :15.00   Min.   :0.0000
##  1st Qu.:33.00   1st Qu.:0.0000
##  Median :45.00   Median :0.0000
##  Mean   :43.65   Mean   :0.3853
##  3rd Qu.:55.00   3rd Qu.:1.0000
##  Max.   :64.00   Max.   :1.0000
```

```
# Two-variable summary
pairs(train)
```



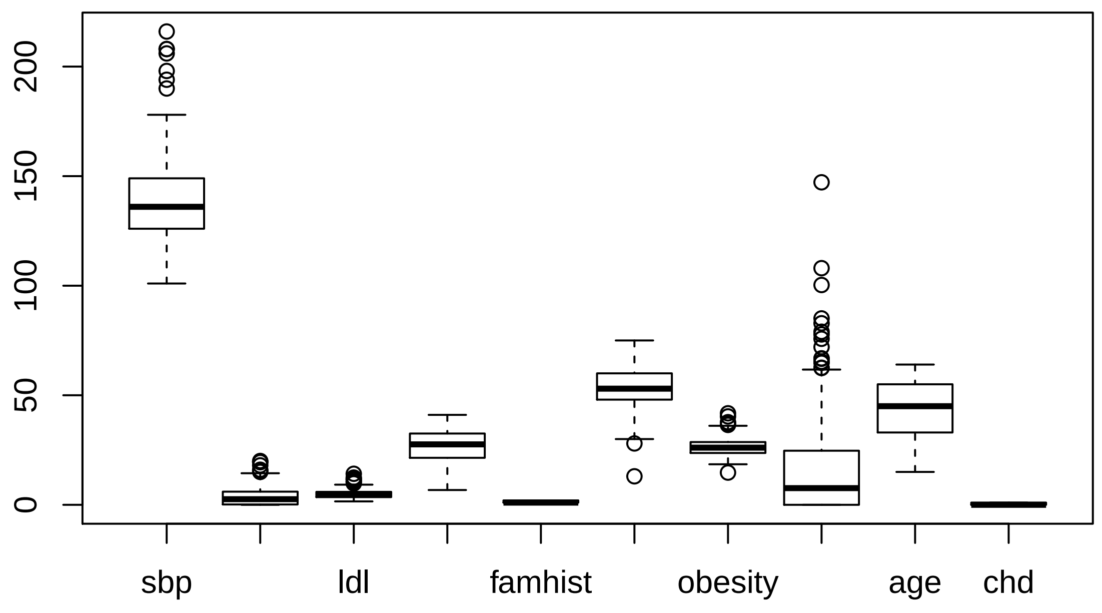
```
# Correlation coefficients ###
round(cor(train),3)
```

```
##              sbp tobacco    ldl adiposity famhist  typea obesity alcohol
## sbp        1.000   0.127  0.170     0.259   0.121 -0.142   0.217   0.142
## tobacco    0.127   1.000  0.169     0.286   0.047 -0.090   0.070   0.223
## ldl        0.170   0.169  1.000     0.489   0.134 -0.079   0.294  -0.040
## adiposity  0.259   0.286  0.489     1.000   0.162 -0.082   0.710   0.052
## famhist    0.121   0.047  0.134     0.162   1.000  0.061   0.127   0.059
## typea     -0.142  -0.090 -0.079    -0.082   0.061  1.000   0.054   0.023
## obesity    0.217   0.070  0.294     0.710   0.127  0.054   1.000   0.021
## alcohol    0.142   0.223 -0.040     0.052   0.059  0.023   0.021   1.000
## age        0.335   0.459  0.358     0.666   0.196 -0.155   0.298   0.089
## chd        0.113   0.274  0.343     0.261   0.245  0.051   0.064   0.030
##                age   chd
## sbp          0.335 0.113
## tobacco      0.459 0.274
## ldl          0.358 0.343
## adiposity    0.666 0.261
## famhist      0.196 0.245
## typea       -0.155 0.051
## obesity      0.298 0.064
## alcohol      0.089 0.030
## age          1.000 0.379
## chd          0.379 1.000
```

```
# Missing values ###
any(is.na(train))
```

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

```
# Outliers
boxplot(train)
```



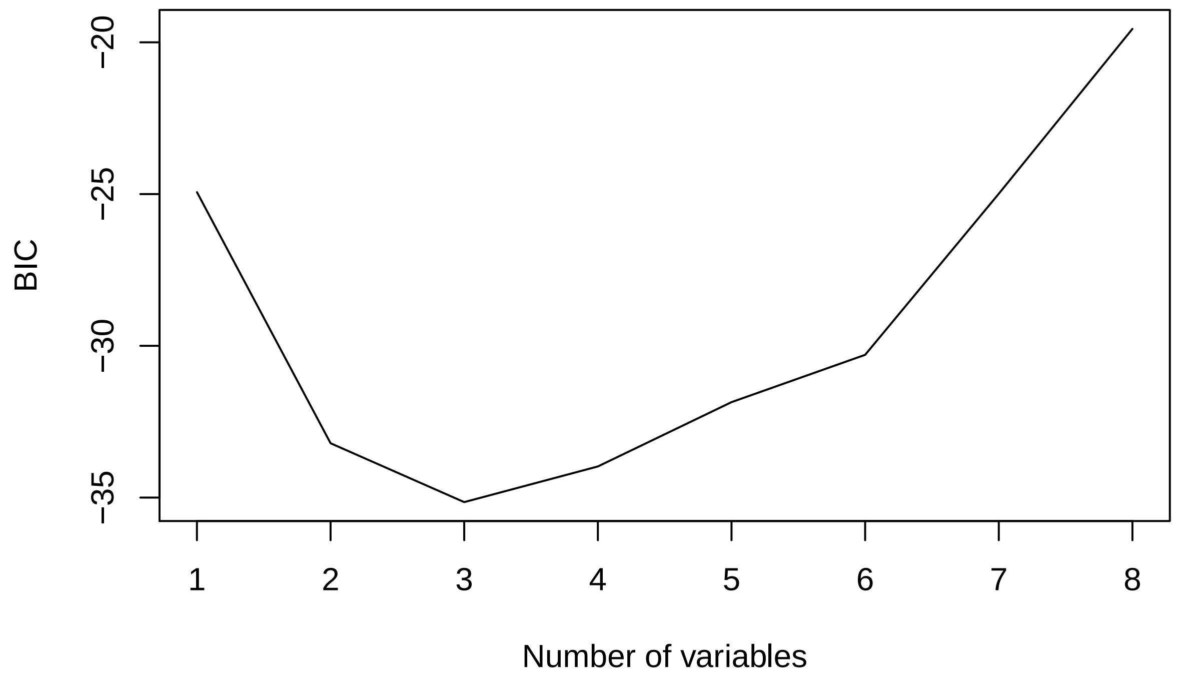
Question 2

```
library(leaps)
#performs all subset selection (best subset selection)
regfit.full <- regsubsets(chd~., data = train)
reg.summary <- summary(regfit.full)
```

```r
reg.summary
```

```
## Subset selection object
## Call: regsubsets.formula(chd ~ ., data = train)
## 9 Variables  (and intercept)
##           Forced in Forced out
## sbp           FALSE      FALSE
## tobacco       FALSE      FALSE
## ldl           FALSE      FALSE
## adiposity     FALSE      FALSE
## famhist       FALSE      FALSE
## typea         FALSE      FALSE
## obesity       FALSE      FALSE
## alcohol       FALSE      FALSE
## age           FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          sbp tobacco ldl adiposity famhist typea obesity alcohol age
## 1  ( 1 ) " " " "     " " " "       " "     " "   " "     " "     "*"
## 2  ( 1 ) " " " "     "*" " "       " "     " "   " "     " "     "*"
## 3  ( 1 ) " " " "     "*" " "       "*"     " "   " "     " "     "*"
## 4  ( 1 ) " " "*"     "*" " "       "*"     " "   " "     " "     "*"
## 5  ( 1 ) " " " "     "*" " "       "*"     "*"   "*"     " "     "*"
## 6  ( 1 ) " " "*"     "*" " "       "*"     "*"   "*"     " "     "*"
## 7  ( 1 ) " " "*"     "*" " "       "*"     "*"   "*"     "*"     "*"
## 8  ( 1 ) "*" "*"     "*" " "       "*"     "*"   "*"     "*"     "*"
```

```r
#Perform variable selection using all subsets selection BIC criteria.
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
```



```r
#Getting number of parameters
which.min(reg.summary$bic)
```

```
## [1] 3
```

```r
# Based on the random train data set, the
```

```r
# Apply logistic regression
sa.heart.fit = glm(chd ~ age + ldl + famhist, family = binomial, data=train)
summary(sa.heart.fit)
```

```
##
## Call:
## glm(formula = chd ~ age + ldl + famhist, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5725  -0.8927  -0.4520   0.9533   2.2370
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.38137    0.82334  -6.536 6.32e-11 ***
## age          0.05354    0.01340   3.996 6.44e-05 ***
## ldl          0.25547    0.07883   3.241  0.00119 **
## famhist      0.79597    0.30742   2.589  0.00962 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 307.96  on 230  degrees of freedom
## Residual deviance: 251.64  on 227  degrees of freedom
## AIC: 259.64
##
## Number of Fisher Scoring iterations: 4
```

```
coef(sa.heart.fit)
```

```
## (Intercept)         age         ldl     famhist
## -5.38137237  0.05354403  0.25546871  0.79596898
```

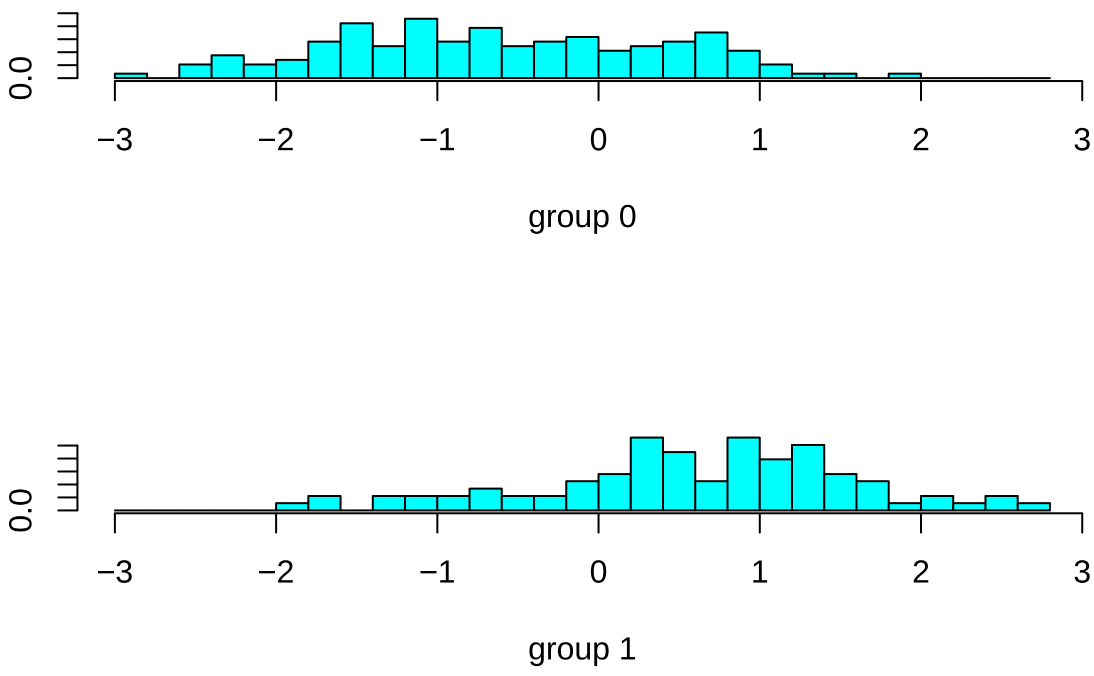
```
# The logistic model is:
# log(chd/(1-chd)) = -5.38137237 + 0.05354403 x age + 0.25546871 x ldl + 0.79596898 x famhist + error
```

Question 3

```
library(MASS)
lda.fit = lda(chd ~ ., data=train)
lda.fit
```

```
## Call:
## lda(chd ~ ., data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6147186 0.3852814
##
## Group means:
##        sbp  tobacco      ldl adiposity  famhist    typea  obesity  alcohol
## 0 137.4366 2.890000 4.368592  25.01965 1.345070 52.71831 26.11035 16.44761
## 1 141.8989 5.387079 5.851573  29.04404 1.595506 53.70787 26.63730 17.89685
##        age
## 0 39.47887
## 1 50.30337
##
## Coefficients of linear discriminants:
##                     LD1
## sbp       -0.0007009372
## tobacco    0.0676367524
## ldl        0.2725124656
## adiposity  0.0025873212
## famhist    0.7608116762
## typea      0.0284787815
## obesity   -0.0709984841
## alcohol   -0.0020146982
## age        0.0409978361
```

```
plot(lda.fit)
```



Question 4

Part A

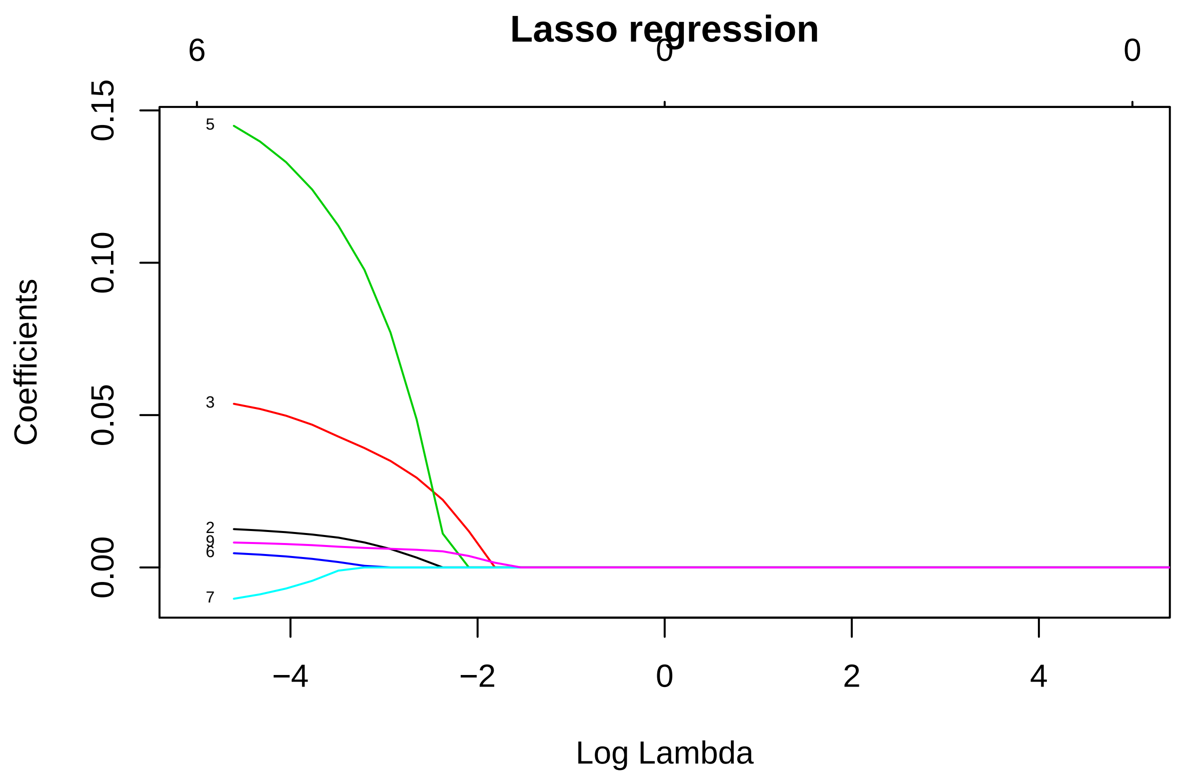
```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

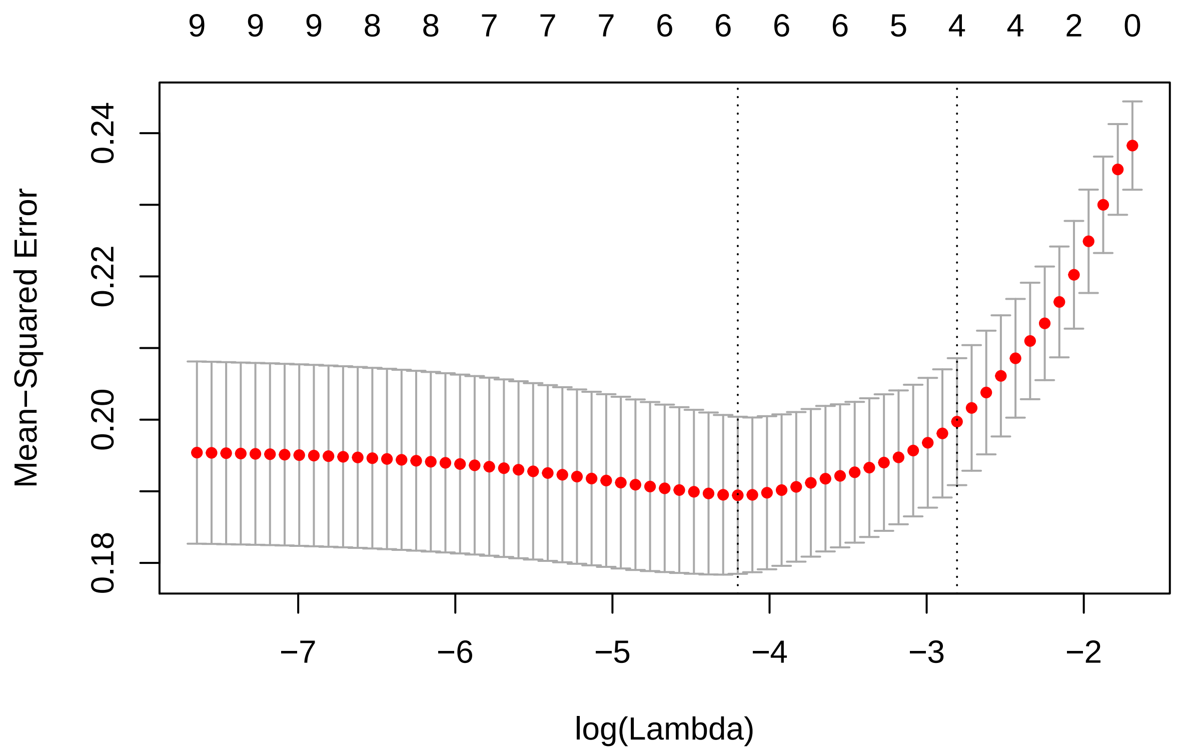
```
## Loaded glmnet 2.0-5
```

```
x <- model.matrix(chd~.,train)[,-1]
y <- train$chd
grid = 10^seq(10,-2, length = 100)
lasso.mod <- glmnet(x,y, alpha = 1, lambda = grid)
plot(lasso.mod, main = "Lasso regression", label = TRUE, xvar = "lambda", xlim = c(-5,5))
```



Part B

```
cv.out <- cv.glmnet(x,y,alpha = 1)
plot(cv.out)
```



### Part C

```
# find lambda
bestlam.lasso <- cv.out$lambda.min
bestlam.lasso
```

```
## [1] 0.01496105
```

```
#best log(lambda)
log(bestlam.lasso)
```

```
## [1] -4.202305
```

```
lasso.mode <- glmnet(x, y, alpha=1, lambda = bestlam.lasso)
predict(lasso.mode, s = bestlam.lasso, type = "coefficients")[1:10,]
```

```
##  (Intercept)          sbp      tobacco          ldl    adiposity
## -0.451303256  0.000000000  0.011903375  0.051104631  0.000000000
##      famhist        typea      obesity      alcohol          age
##  0.137024080  0.003962402 -0.008046859  0.000000000  0.007837167
```

Part D

```
lasso.fit = glm(chd ~ tobacco + ldl + famhist + typea + obesity + age, family = binomial, data=train)
summary(lasso.fit)
```

```
##
## Call:
```

```
## glm(formula = chd ~ tobacco + ldl + famhist + typea + obesity +
##     age, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8657  -0.8132  -0.4169   0.8768   2.3527
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.95245    1.55916  -3.818 0.000135 ***
## tobacco      0.07001    0.03680   1.903 0.057081 .
## ldl          0.31295    0.08703   3.596 0.000323 ***
## famhist      0.84793    0.31932   2.655 0.007921 **
## typea        0.03701    0.01760   2.103 0.035475 *
## obesity     -0.07713    0.04281  -1.802 0.071589 .
## age          0.05300    0.01504   3.524 0.000425 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 307.96  on 230  degrees of freedom
## Residual deviance: 240.58  on 224  degrees of freedom
## AIC: 254.58
##
## Number of Fisher Scoring iterations: 4
```

```
coef(lasso.fit)
```

```
## (Intercept)     tobacco         ldl     famhist       typea     obesity
## -5.95245042  0.07001089  0.31295446  0.84792663  0.03700809 -0.07713433
##         age
##  0.05300280
```

Question 5

Part A

```
library(pamr)
```

```
## Loading required package: cluster
```

```
## Loading required package: survival
```

```
# Reformat the dataset for parm
pamrTrain <- list(x=t(as.matrix(train[,-10])), y=train[,10])
pamrValid <- list(x=t(as.matrix(valid[,-10])), y=valid[,10])

# Fit the classifier on the entire training set
fit.pamr <- pamr.train(pamrTrain)
```

```
## 123456789101112131415161718192021222324252627282930
```

```
fit.pamr
```

```
## Call:
```

```
## pamr.train(data = pamrTrain)
##    threshold nonzero errors
## 1  0.000     9       79
## 2  0.137     9       76
## 3  0.273     8       78
## 4  0.410     6       81
## 5  0.547     5       84
## 6  0.684     5       83
## 7  0.820     5       86
## 8  0.957     5       87
## 9  1.094     5       89
## 10 1.231     4       90
## 11 1.367     3       89
## 12 1.504     3       89
## 13 1.641     2       89
## 14 1.778     2       89
## 15 1.914     2       89
## 16 2.051     1       89
## 17 2.188     1       89
## 18 2.325     1       89
## 19 2.461     1       89
## 20 2.598     1       89
## 21 2.735     1       89
## 22 2.872     1       89
## 23 3.008     1       89
## 24 3.145     1       89
## 25 3.282     1       89
## 26 3.419     1       89
## 27 3.555     1       89
## 28 3.692     1       89
## 29 3.829     1       89
## 30 3.966     0       89
```

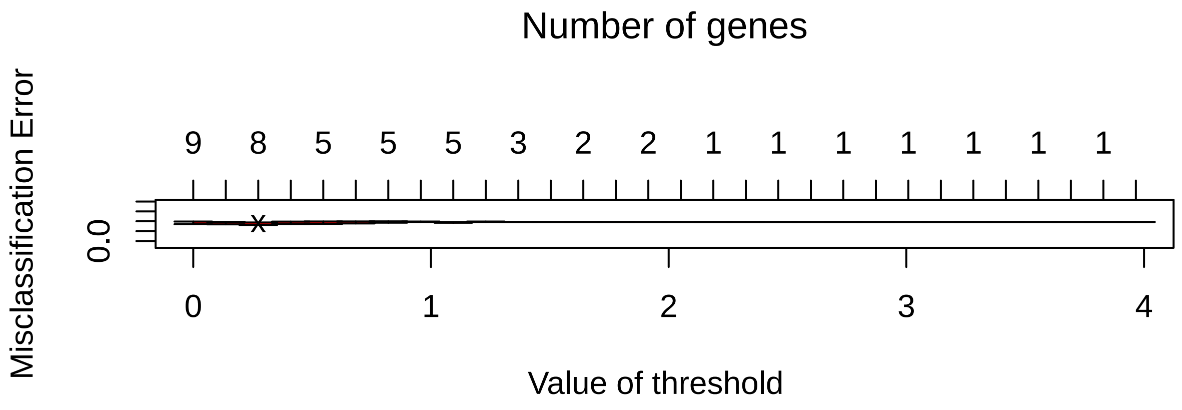
```
# Use cross-validation to select the best regularization parameter
fit.cv.pamr <- pamr.cv(fit.pamr, pamrTrain)
```

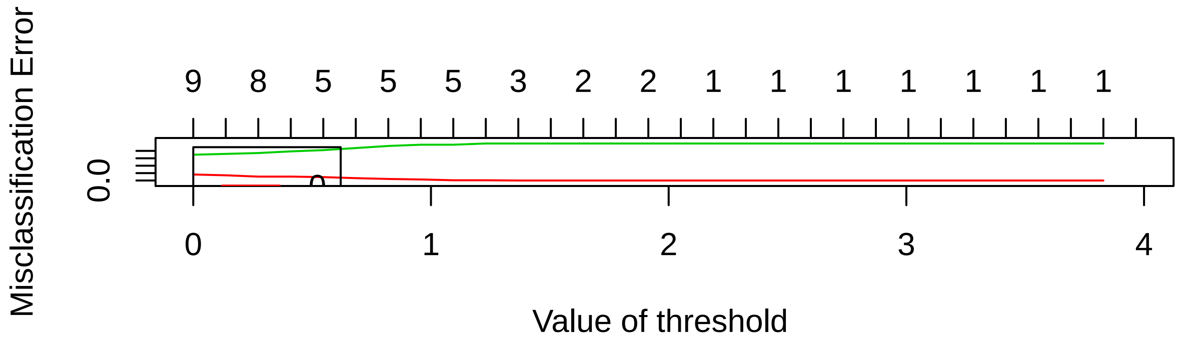
```
## 12Fold 1 :123456789101112131415161718192021222324252627282930
## Fold 2 :123456789101112131415161718192021222324252627282930
## Fold 3 :123456789101112131415161718192021222324252627282930
## Fold 4 :123456789101112131415161718192021222324252627282930
## Fold 5 :123456789101112131415161718192021222324252627282930
## Fold 6 :123456789101112131415161718192021222324252627282930
## Fold 7 :123456789101112131415161718192021222324252627282930
## Fold 8 :123456789101112131415161718192021222324252627282930
## Fold 9 :123456789101112131415161718192021222324252627282930
## Fold 10 :123456789101112131415161718192021222324252627282930
```

```
fit.cv.pamr
```

```
## Call:
## pamr.cv(fit = fit.pamr, data = pamrTrain)
##    threshold nonzero errors
## 1  0.000     9       85
## 2  0.137     9       84
## 3  0.273     8       81
## 4  0.410     6       85
## 5  0.547     5       86
## 6  0.684     5       87
## 7  0.820     5       89
## 8  0.957     5       90
## 9  1.094     5       87
## 10 1.231     4       90
## 11 1.367     3       89
## 12 1.504     3       89
## 13 1.641     2       89
## 14 1.778     2       89
## 15 1.914     2       89
## 16 2.051     1       89
## 17 2.188     1       89
## 18 2.325     1       89
## 19 2.461     1       89
## 20 2.598     1       89
## 21 2.735     1       89
## 22 2.872     1       89
## 23 3.008     1       89
## 24 3.145     1       89
## 25 3.282     1       89
## 26 3.419     1       89
## 27 3.555     1       89
## 28 3.692     1       89
## 29 3.829     1       89
## 30 3.966     0       89
```

```
# Manually select the threshold depending on the plots and on the confusion matrix
pamr.plotcv(fit.cv.pamr)
```





Part B

```
#Let's compare thresholds=1 and 2 to illustrate the effect of shrinkage
pamr.confusion(fit.cv.pamr, threshold=0.137)
```

```
##     0  1 Class Error rate
## 0 127 15        0.1056338
## 1  66 23        0.7415730
## Overall error rate= 0.349
```

```
pamr.confusion(fit.cv.pamr, threshold=0.273)
```

```
##     0  1 Class Error rate
## 0 127 15        0.1056338
## 1  66 23        0.7415730
## Overall error rate= 0.349
```

```
#Get the best threshhold
thresh <- max(fit.cv.pamr$threshold[fit.cv.pamr$error==min(fit.cv.pamr$error)])
thresh
```

```
## [1] 0.2734973
```

```
# Refit the classifier on the full dataset, but using the threshold
fit.pamr <- pamr.train(pamrTrain, threshold=0.1367487)
```

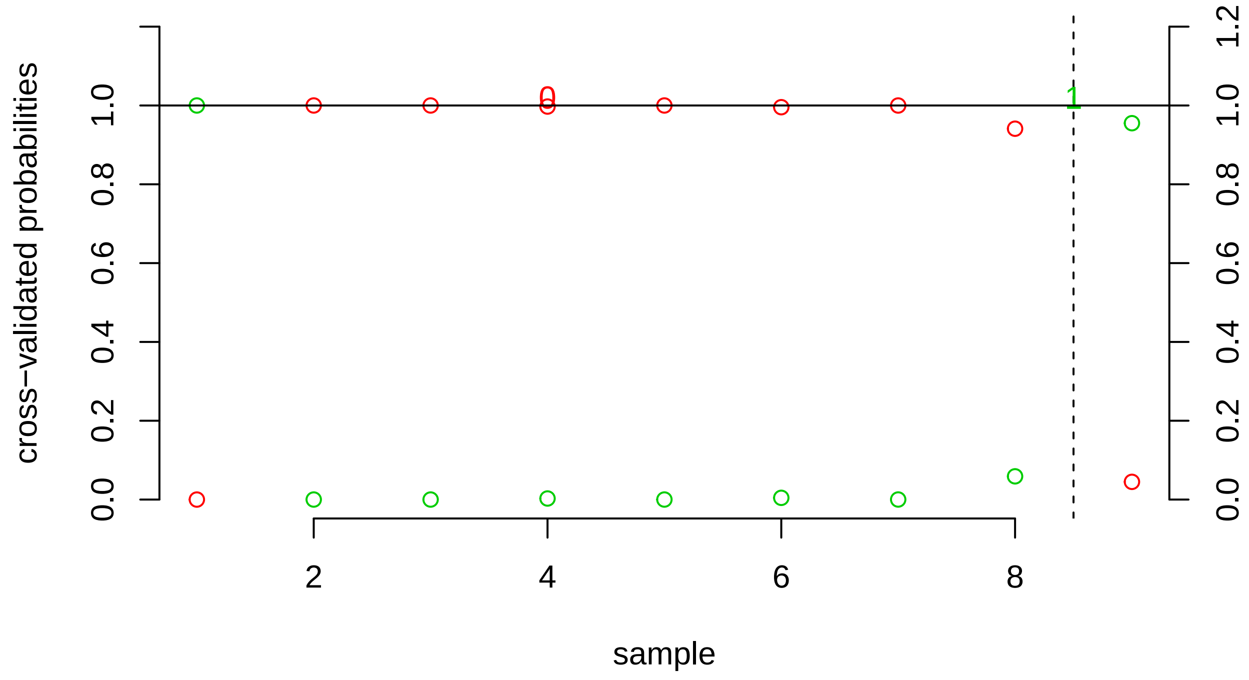
```
## 1
```

Part C

```
set.seed(120)
mydata <- list(x=x,y=y)
mytrain <- pamr.train(mydata)
```

```
## Warning: a class contains only 1 sample123456789101112131415161718192021222324252627282930
```

```
pamr.plotcvprob(mytrain,mydata, threshold=0.1367487)
```



Question 6

Part A

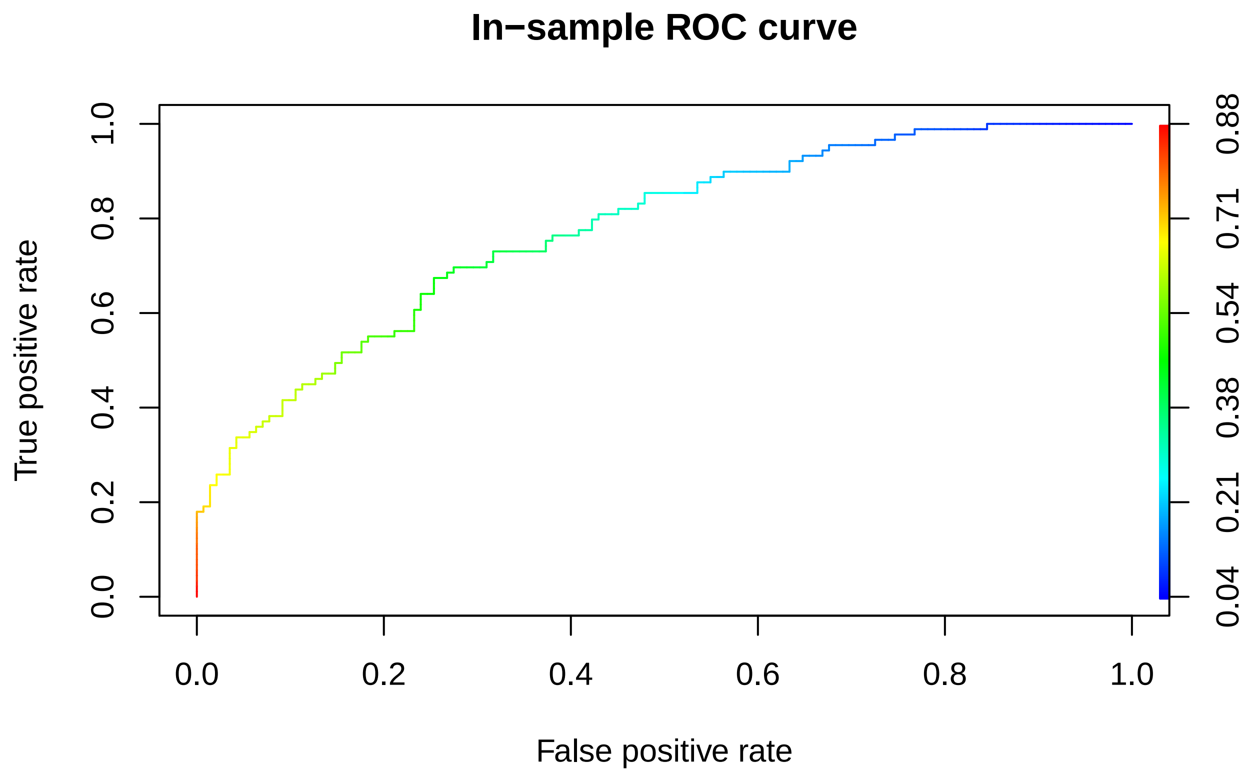
```
library(ROCR)
```

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

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

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

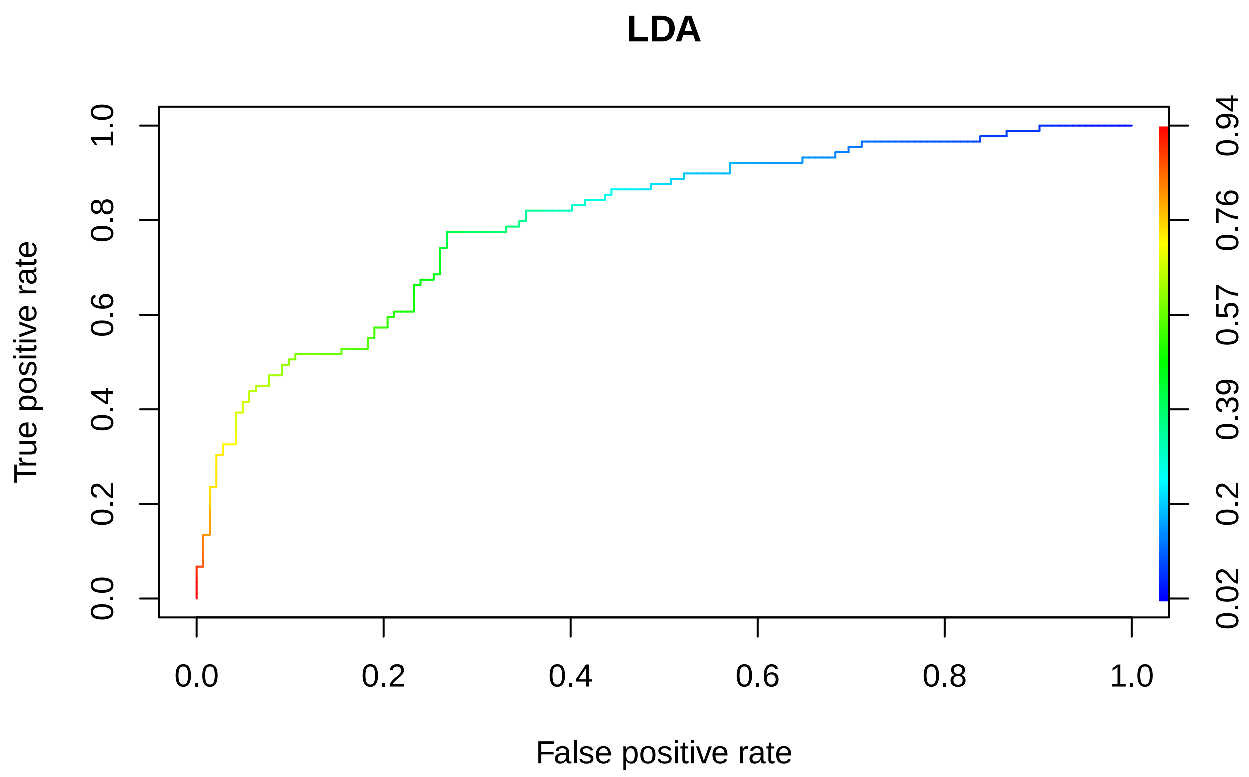
```r
# Logistic
scores <- predict(sa.heart.fit, newdata=train, type="response")

# compare predicted probabilities to labels, for varying probability cutoffs
pred <- prediction(scores, labels=train$chd )
perf <- performance(pred, "tpr", "fpr")

# plot the ROC curve
plot(perf, colorize=T, main="In-sample ROC curve")
```



```r
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.7757557
```

```r
#LDA
scores <- predict(lda.fit, newdata= train)$posterior[,2]
pred <- prediction( scores, labels= train$chd )
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=T, main="LDA")
```



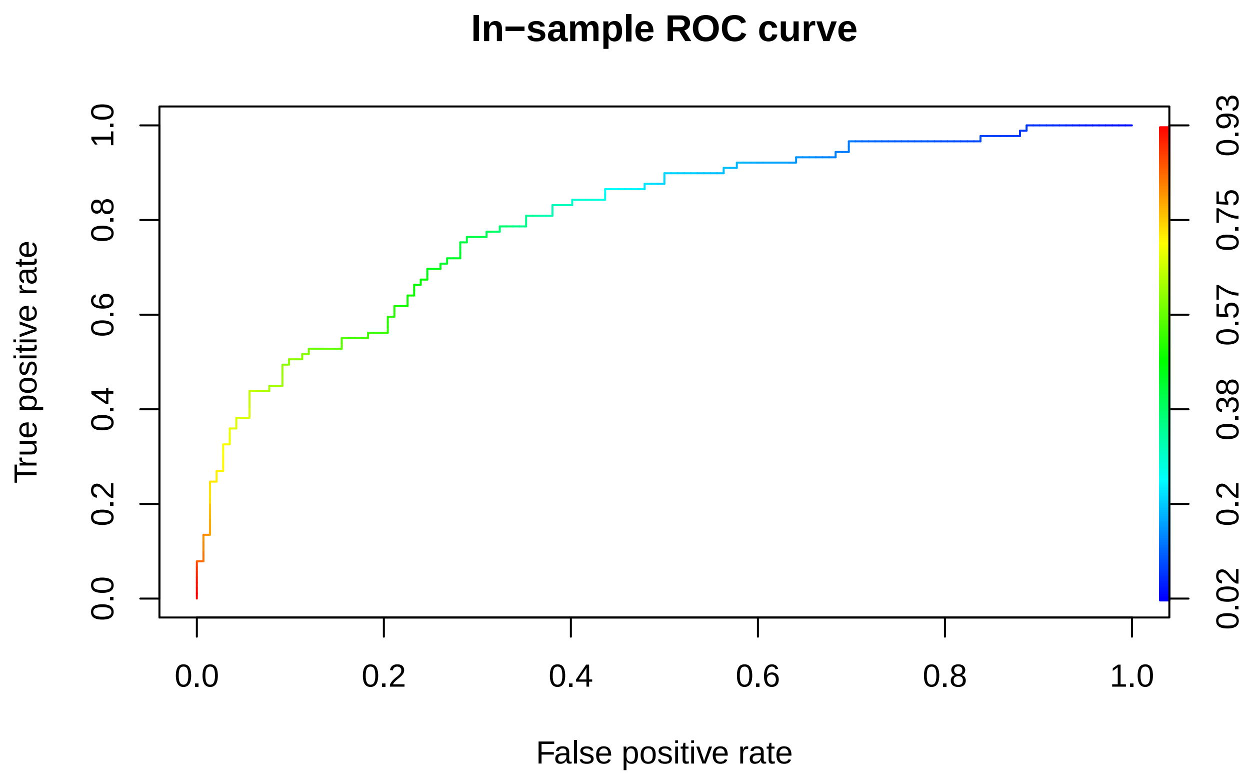
```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.8002057
```

```
#Lasso
summary(lasso.fit)
```

```
##
## Call:
## glm(formula = chd ~ tobacco + ldl + famhist + typea + obesity +
##     age, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8657  -0.8132  -0.4169   0.8768   2.3527
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.95245    1.55916  -3.818 0.000135 ***
## tobacco      0.07001    0.03680   1.903 0.057081 .
## ldl          0.31295    0.08703   3.596 0.000323 ***
## famhist      0.84793    0.31932   2.655 0.007921 **
## typea        0.03701    0.01760   2.103 0.035475 *
## obesity     -0.07713    0.04281  -1.802 0.071589 .
## age          0.05300    0.01504   3.524 0.000425 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 307.96  on 230  degrees of freedom
## Residual deviance: 240.58  on 224  degrees of freedom
## AIC: 254.58
##
## Number of Fisher Scoring iterations: 4
```

```
# calculate predicted probabilities on the same training set
scores <- predict(lasso.fit, newdata=train, type="response")

# compare predicted probabilities to labels, for varying probability cutoffs
pred <- prediction(scores, labels=train$chd )
perf <- performance(pred, "tpr", "fpr")

# plot the ROC curve
plot(perf, colorize=T, main="In-sample ROC curve")
```

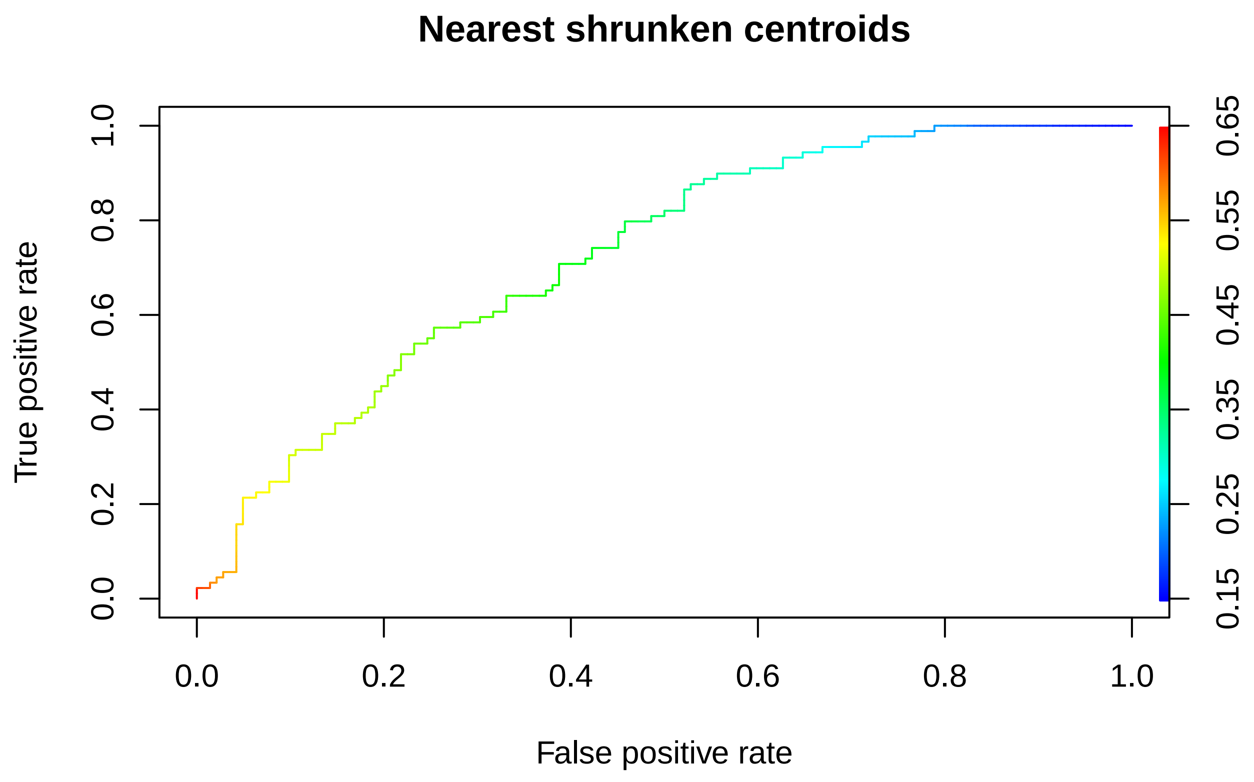


```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.8004431
```

```
# Nearest shrunken centroids
pred.pamr.train <- pamr.predict(fit.pamr, newx=pamrTrain$x, threshold=0.1367487, type="posterior")[,2]
pred <- prediction(predictions=pred.pamr.train, labels= pamrTrain$y)
```

```r
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=T, main="Nearest shrunken centroids")
```

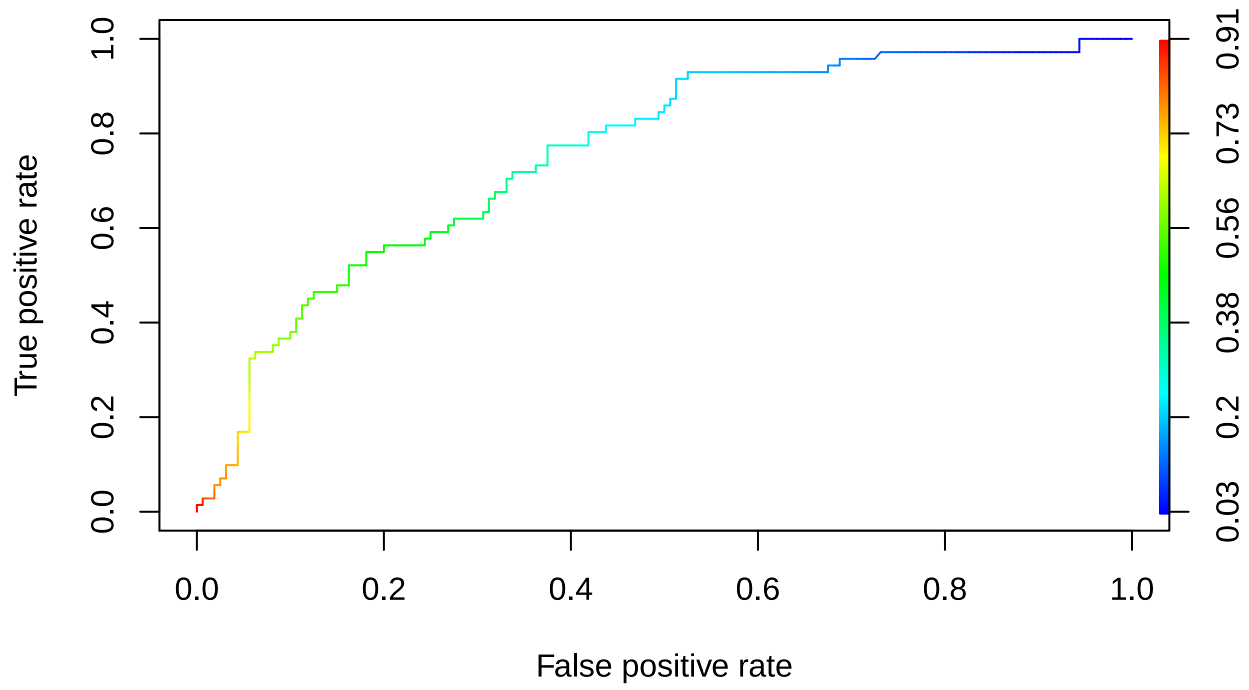


```r
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.724007
```

Part B

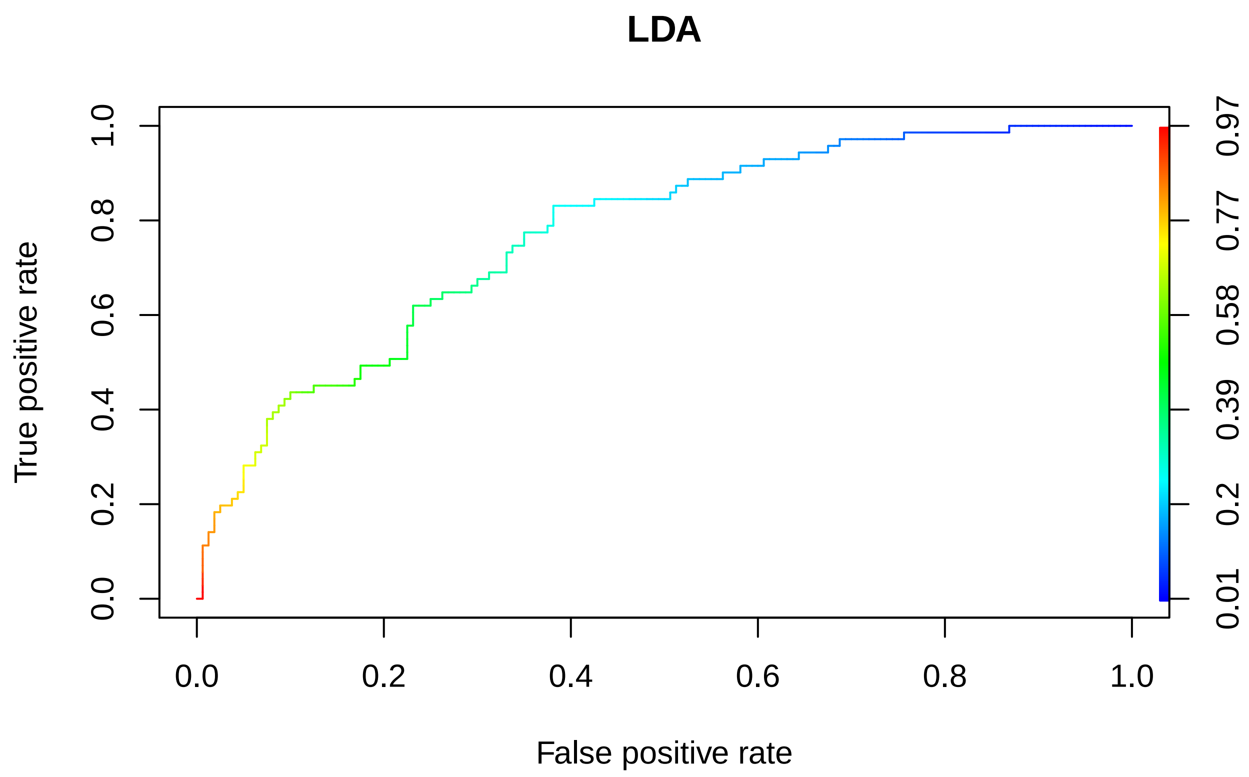
```r
#logistic
# make prediction on the validation dataset
scores <- predict(sa.heart.fit, newdata=valid, type="response")
pred <- prediction(scores, labels=valid$chd )
perf <- performance(pred, "tpr", "fpr")

# overlay the line for the ROC curve
plot(perf, colorize=T)
```



```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.7571743
```

```
# LDA
scores <- predict(lda.fit, newdata= valid)$posterior[,2]
pred <- prediction( scores, labels= valid$chd )
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=T, main="LDA")
```



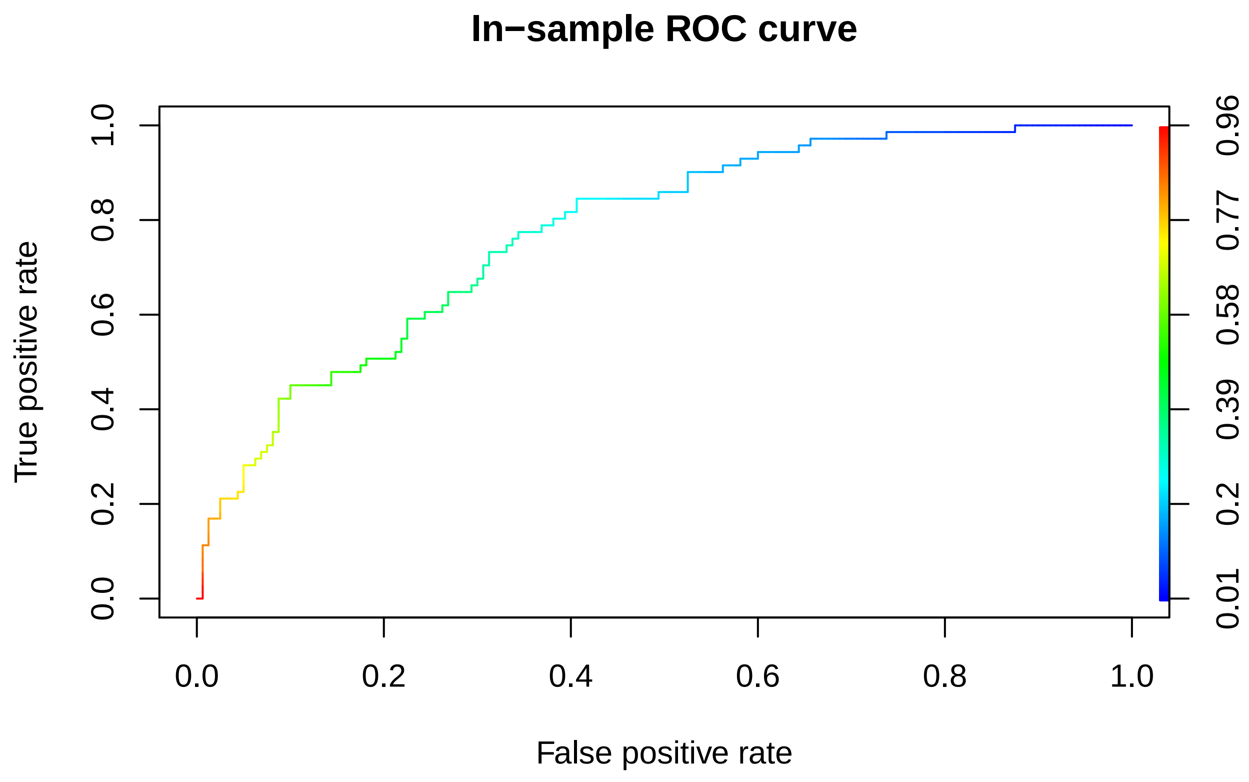
```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.7706866
```

```
#Lasso
summary(lasso.fit)
```

```
##
## Call:
## glm(formula = chd ~ tobacco + ldl + famhist + typea + obesity +
##     age, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8657  -0.8132  -0.4169   0.8768   2.3527
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.95245    1.55916  -3.818 0.000135 ***
## tobacco      0.07001    0.03680   1.903 0.057081 .
## ldl          0.31295    0.08703   3.596 0.000323 ***
## famhist      0.84793    0.31932   2.655 0.007921 **
## typea        0.03701    0.01760   2.103 0.035475 *
## obesity     -0.07713    0.04281  -1.802 0.071589 .
## age          0.05300    0.01504   3.524 0.000425 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 307.96  on 230  degrees of freedom
## Residual deviance: 240.58  on 224  degrees of freedom
## AIC: 254.58
##
## Number of Fisher Scoring iterations: 4
```

```
# calculate predicted probabilities on the same training set
scores <- predict(lasso.fit, newdata=valid, type="response")

# compare predicted probabilities to labels, for varying probability cutoffs
pred <- prediction(scores, labels=valid$chd )
perf <- performance(pred, "tpr", "fpr")

# plot the ROC curve
plot(perf, colorize=T, main="In-sample ROC curve")
```

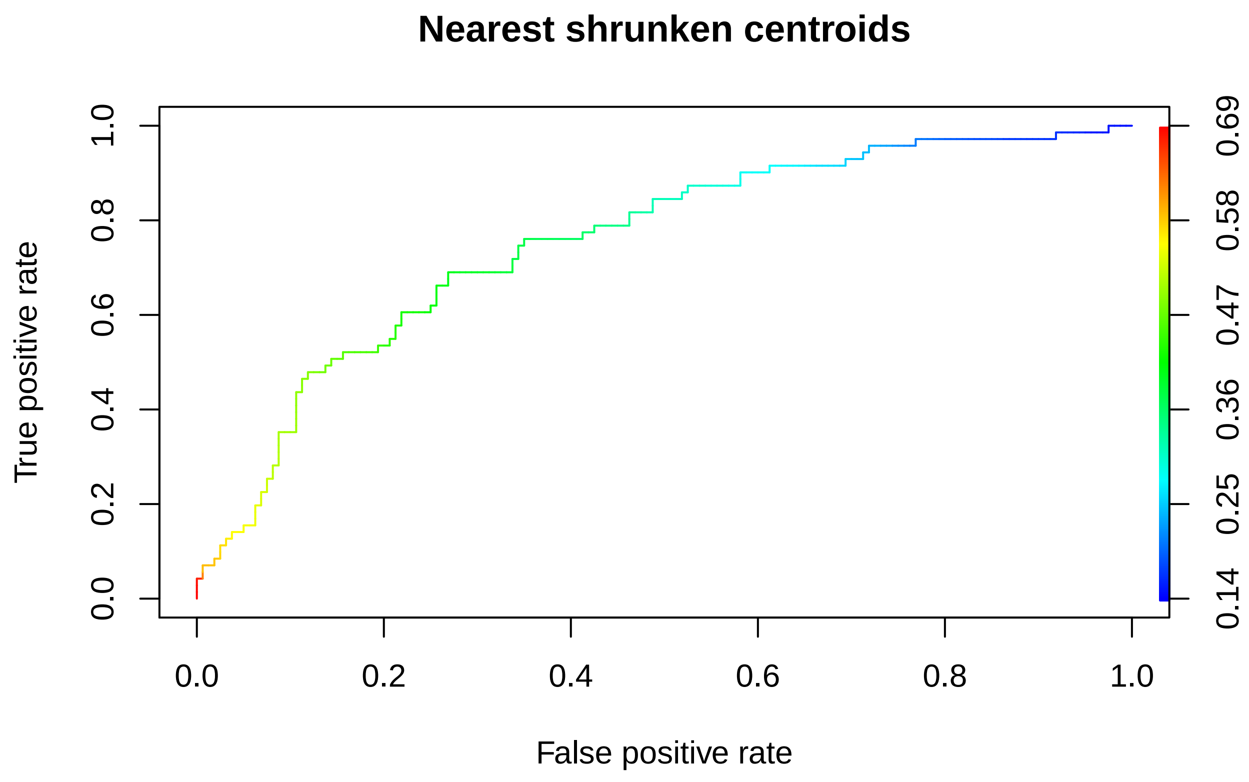


```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.7751761
```

```
# Nearest shrunken centroids
pred.pamr.train <- pamr.predict(fit.pamr, newx=pamrValid$x, threshold=0.1367487, type="posterior")[,2]
pred <- prediction(predictions=pred.pamr.train, labels= pamrValid$y)
```

```r
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=T, main="Nearest shrunken centroids")
```



```r
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)
```

```
## [1] 0.7525528
```

Part C

After evaluating the ROC curve of each classifier, we can find the result between training and validation set are slightly different.

How do the results different between the training and the validation set?

On training set, the the area under the ROC curve:

- Logistic regression: 0.7757557
- LDA: 0.8002057
- Logistic with lasso: 0.8004431
- Nearest shrunken centroids: 0.724007

On valid set, the the area under the ROC curve:

- Logistic regression: 0.7571743
- LDA: 0.7706866
- Logistic with lasso: 0.7751761

- Nearest shrunken centroids: 0.7525528

Which approach(es) perform(s) better on the validation set?

In my experiment, LDA and Logistic with lasso performs a little bit better, but not too much. The best one (Logistic with lasso) and the worst one (Nearest shrunken centroids) only has 0.02 different in AUC.

What is are the reasons for this difference in performance?

They perform similar because this problem is a simple 2 class classification. I believe one of the reasons to cause them slightly different is the random training and validation dataset, they also perform differently on this problem.

- Logistic regression is not the best because we didn't check the correlationship between predictors. And from the pair(train), we can see there are correlation between parameters.
- One drawback of diagonal LDA is that it depends on all of the features. This is the reason it can't perform better in this problem.
- Logistic with Lasso is the best because it shrunk the dataset to filter out the less related predictors with zero coef. If we test different number of predictors, the result possibly become even better.
- Nearest shrunken centroids only depends on a subset of the features, for reasons of accuracy and interpret ability. With an accurate threshold with less err rate, it performs well in this problem.

Which models are more interpretable?

Firstly, LDA is not interpretable because it doesn't apply subset selection. And Logistic with lasso and Nearest shrunk centroids are also both interpretable with small subset. And logistic with best subset selection can be the most interpretable if the selected subset is small. In this case, it is with 3 predictors.

Question 7

- a. GaussI <= LinLog LinLog is the logistic model maximizing the likelihoods
- b. GaussX <= QuadLog Both of them have quadratic features, but QuadLog is the model of this class maximizing likelihoods
- c. LinLog <= QuadLog LinLog with linear features is a subclass of QuadLog with quadratic functions
- d. GaussI <= QuadLog GaussI have higher average log joint probabilities of examples and labels, but it does not necessarily translate to higher likelihoods. So QuadLog with quadratic functions maximizing the log likelihood may perform better.
- e. It is commonly the case that GaussX <= GaussI and that QuadLog <= LinLog