Homework 4

Author: Dong Bin

email: bindong2@illinois.edu

Question 1

Data preparation

```
rm(list=ls())

set.seed(1)

raw.data <-read.csv("Sepsis.csv")

n <- nrow(raw.data)

ntest <- round(n*0.3)

test.id = sample(1:n, ntest)


training.data <- raw.data[-test.id, ]

testing.data <- raw.data[test.id, ]


training.active <- subset(training.data, THERAPY==1)

training.control <- subset(training.data, THERAPY==0)

training.active.data <- training.active[, c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLPl
training.active.best <- training.active[, "BEST"]

training.control.data <- training.control[,c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLl
training.control.best <- training.control[, "BEST"]

testing.data.data <- testing.data[, c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLPLAT","l

testing.data.best <- testing.data[, c("BEST")]

#testing.data
```

Demo

```
library(randomForest)
```

```
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
```

```
active.forest <- randomForest(Health~., data=training.active.data, mtry=3, nodesize=100)
control.forest <- randomForest(Health~., data=training.control.data, mtry=3, nodesize=100)

predictions<-predict(active.forest, testing.data.data)

result <- sum((testing.data.best - predictions)^2)/length(predictions)



GetPrediction <- function(prediction.active, prediction.control)
{
  diff <- prediction.active - prediction.control
  result <- sapply(diff, function(x){
    if(x>0){return <- 1}
    else {return <- 0}
      })
  return <- result
}

mtrys <- c(1, 3, 6, 9)

nodesizes <- c(10, 100, 500, 1000)

errors<- matrix(rep(0, times=length(mtrys)*length(nodesizes)), nrow=length(mtrys), ncol=length(nodesize
```

Train random forest

```
for(iter in c(1:100))
{
  cat(iter, " ")
  if(iter%%10==0)
  {
    cat("\n")
  }


  set.seed(iter)

  test.id = sample(1:n, ntest)

  training.data <- raw.data[-test.id, ]
  testing.data <- raw.data[test.id, ]

  training.active <- subset(training.data, THERAPY==1)
  training.control <- subset(training.data, THERAPY==0)

  training.active.data <- training.active[, c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BL
  training.active.best <- training.active[, "BEST"]

  training.control.data <- training.control[,c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BL
  training.control.best <- training.control[, "BEST"]

  testing.data.data <- testing.data[, c("Health", "PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLPLAT"

  testing.data.best <- testing.data[, c("BEST")]

  for(i in 1:length(mtrys))
  {
    for(j in 1:length(nodesizes))
    {
      active.forest <- randomForest(Health~., data=training.active.data, mtry=mtrys[i], nodesiz
      control.forest <- randomForest(Health~., data=training.control.data, mtry=mtrys[i], nodes

      predictions.active <- predict(active.forest, testing.data.data)

      predictions.control <- predict(control.forest, testing.data.data)
      predictions <- GetPrediction(predictions.active, predictions.control)

      error <- sum((testing.data.best - predictions)^2)/length(predictions)

      errors[i,j]= errors[i,j]+error
    }
  }
}
```

```
## 1  2  3  4  5  6  7  8  9  10
## 11  12  13  14  15  16  17  18  19  20
## 21  22  23  24  25  26  27  28  29  30
## 31  32  33  34  35  36  37  38  39  40
## 41  42  43  44  45  46  47  48  49  50
## 51  52  53  54  55  56  57  58  59  60
## 61  62  63  64  65  66  67  68  69  70
## 71  72  73  74  75  76  77  78  79  80
## 81  82  83  84  85  86  87  88  89  90
## 91  92  93  94  95  96  97  98  99  100
```

Check Errors

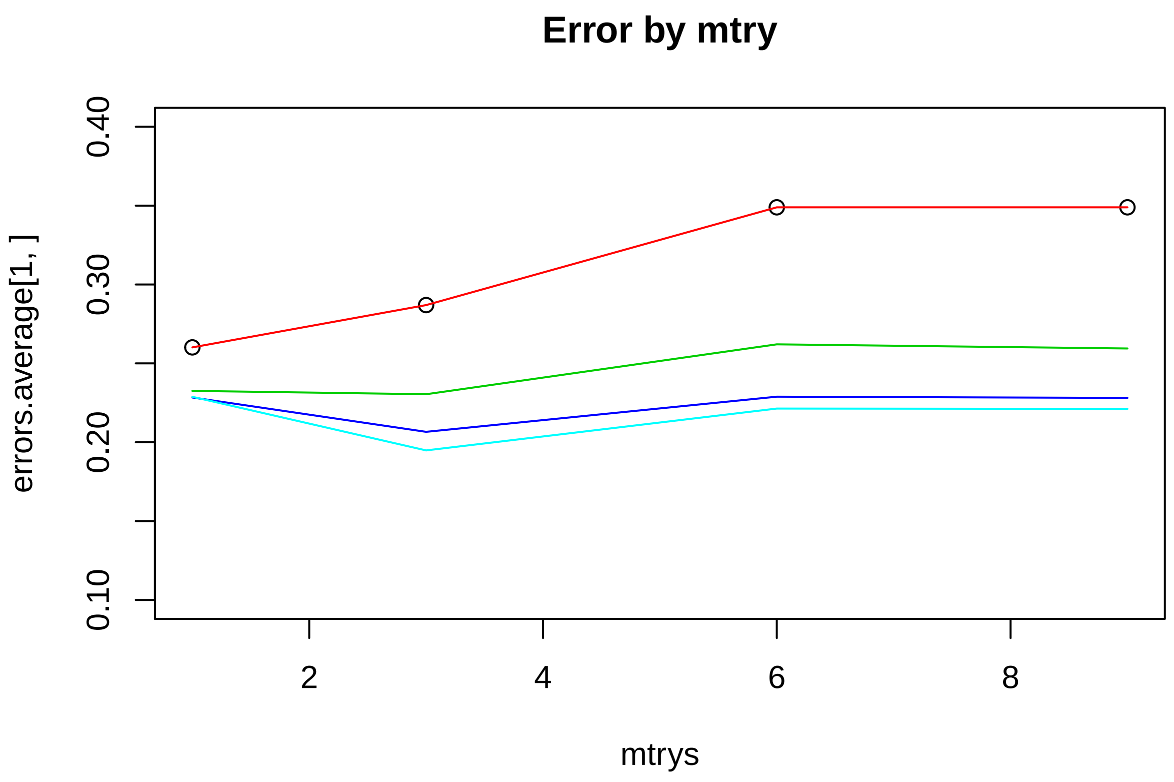
```
errors.average <- errors/100

errors.average
```

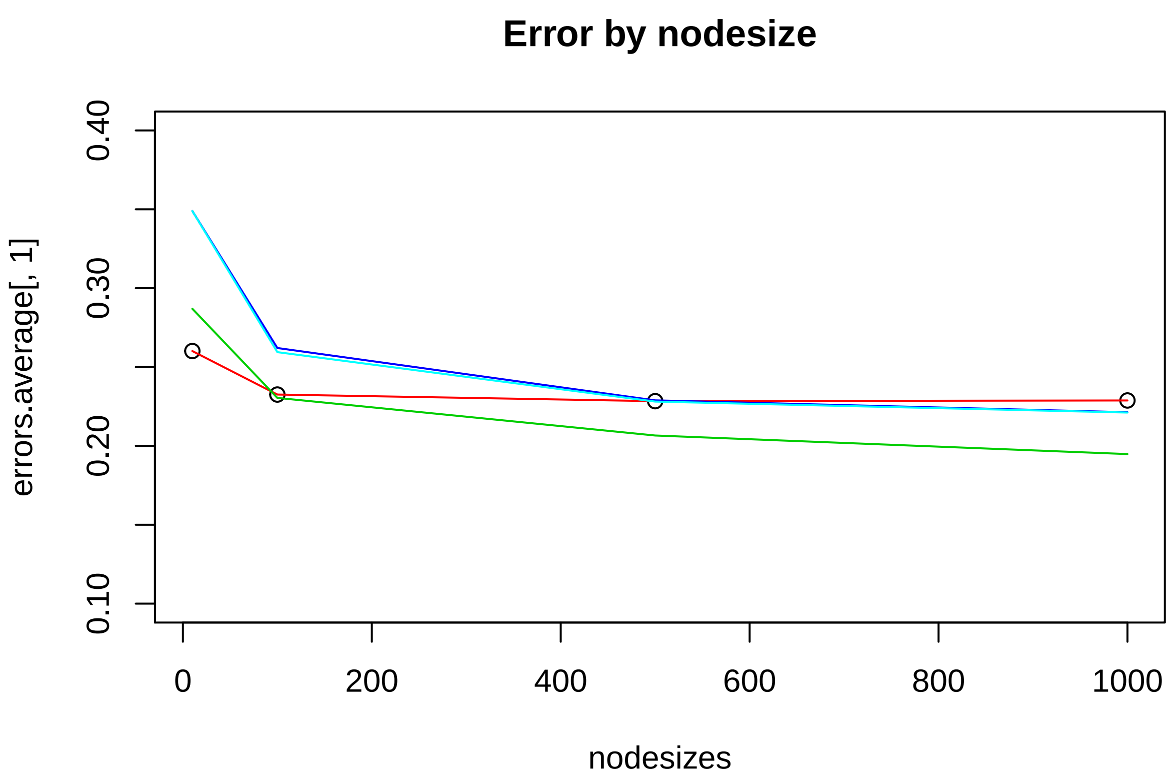
```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.2601418 0.2869504 0.3489362 0.3489362
## [2,] 0.2325532 0.2304255 0.2620567 0.2594326
## [3,] 0.2283688 0.2065957 0.2288652 0.2280851
## [4,] 0.2287943 0.1948227 0.2213475 0.2211348
```

Demo Errors

```
plot(mtrys, errors.average[1,],ylim=c(0.1,0.4), main="Error by mtry")
for(i in 1:length(mtrys))
{
  lines(mtrys, errors.average[i,], col=i+1)
}
```



```
plot(nodesizes, errors.average[,1],ylim=c(0.1,0.4), main="Error by nodesize")
for(i in 1:length(nodesizes))
{
  lines(nodesizes, errors.average[,i], col=i+1)
}
```



Based on the plots above, we can discover that the best prediction model is with parameters:

mtry: 6

nodesizes: 100

the general trend of different parameters are:

- as mtry increase, prediction error decrease.
- the best nodesize is about 100.

Question 2

Get predictions

```
rm(list=ls())

set.seed(1)


raw.data <- read.csv("Sepsis.csv")

data.cols <- c("Health","PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLPLAT","BLLBILI","BLLCREAT","TIM

raw.data.data <- raw.data[, data.cols]
raw.active <- subset(raw.data, THERAPY==1)
raw.active.data <- raw.active[, data.cols]

raw.control <- subset(raw.data, THERAPY==0)
raw.control.data <- raw.control[, data.cols]


GetPrediction <- function(prediction.active, prediction.control)
{
  diff <- prediction.active - prediction.control
  result <- sapply(diff, function(x){
    if(x>0){return <- 1}
    else {return <- 0}
      })
  return <- result
}


active.forest <- randomForest(Health~., data=raw.active.data, mtry=6, nodesize=100)
control.forest <- randomForest(Health~., data=raw.control.data, mtry=6, nodesize=100)

predictions.active <- predict(active.forest, raw.data.data)
predictions.control <- predict(control.forest, raw.data.data)
predictions <- GetPrediction(predictions.active, predictions.control)
```

Train decision tree

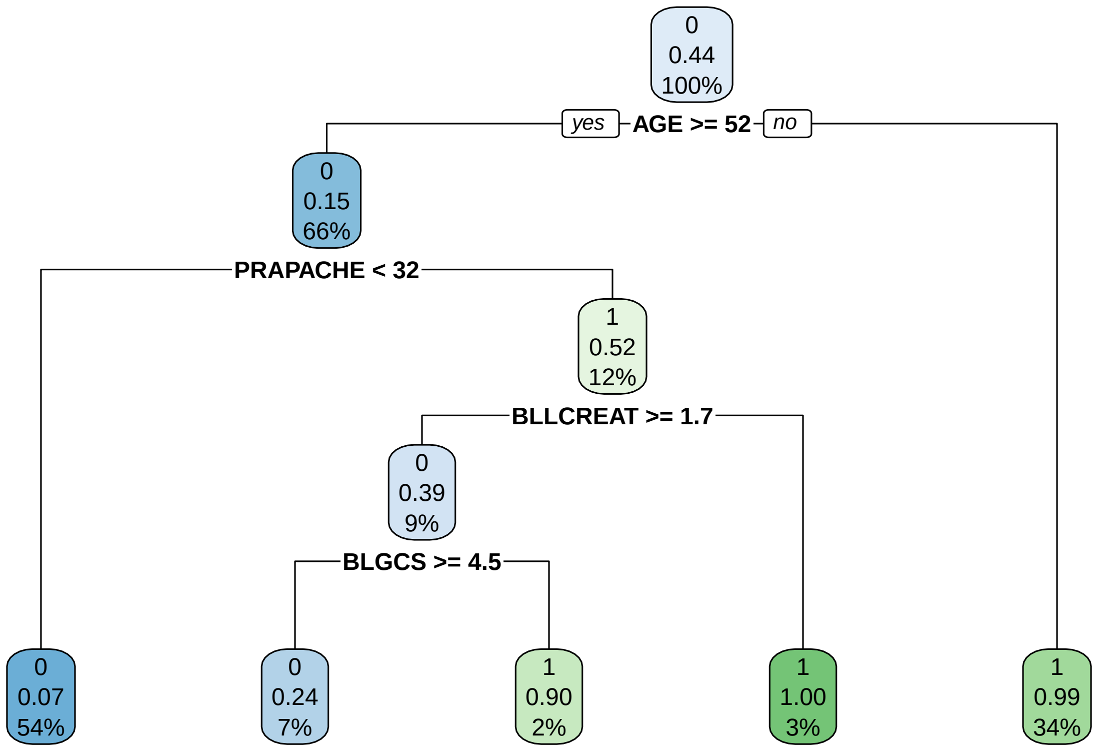
```
library(rpart)
library(rpart.plot)

data.cols <- c("PRAPACHE","AGE","BLGCS","ORGANNUM","BLIL6","BLLPLAT","BLLBILI","BLLCREAT","TIMFIRST","BL

rpart.fit <- rpart(as.factor(predictions)~., data = data.frame(raw.data.data[,data.cols],predictions))


rpart.plot(rpart.fit)
```

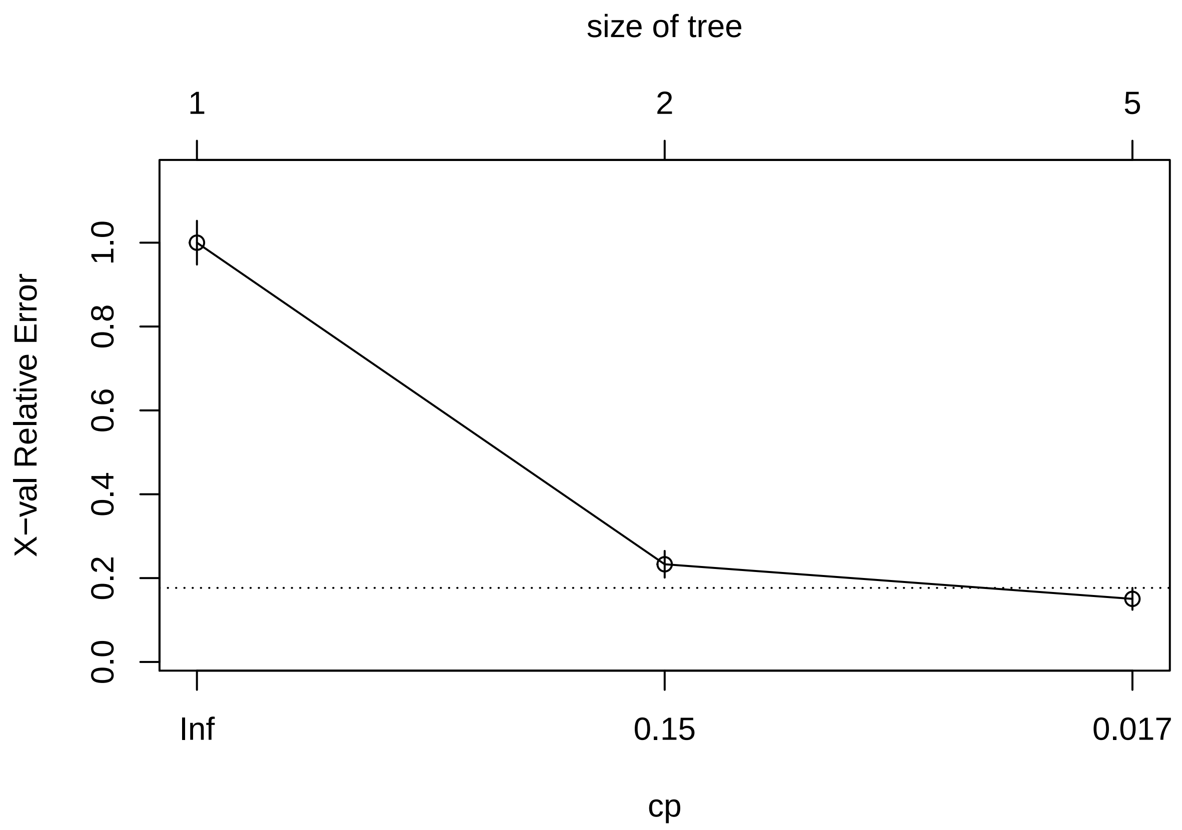
```
## Warning: Bad 'data' field in model 'call' (expected a data.frame or a matrix).
## To silence this warning:
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



```
rpart.fit$cptable
```

```
##           CP nsplit rel error    xerror       xstd
## 1 0.76699029      0 1.0000000 1.0000000 0.05221790
## 2 0.02912621      1 0.2330097 0.2330097 0.03186843
## 3 0.01000000      4 0.1359223 0.1504854 0.02612143
```

```
plotcp(rpart.fit)
```



```
prune(rpart.fit, cp=0.03)
```

```
## n= 470
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 470 206 0 (0.5617021 0.4382979)
##   2) AGE>=51.7395 310  47 0 (0.8483871 0.1516129) *
##   3) AGE< 51.7395 160   1 1 (0.0062500 0.9937500) *
```