

# 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", "ORGANUM", "BLIL6", "BLLPLAT", "BEST")]
training.active.best <- training.active[, "BEST"]

training.control.data <- training.control[, c("Health", "PRAPACHE", "AGE", "BLGCS", "ORGANUM", "BLIL6", "BLLPLAT", "BEST")]
training.control.best <- training.control[, "BEST"]

testing.data.data <- testing.data[, c("Health", "PRAPACHE", "AGE", "BLGCS", "ORGANUM", "BLIL6", "BLLPLAT", "BEST")]

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

#testing.data
```

### Demo

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
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(nodesizes))
```

```
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", "BLIL12", "BLIL18", "BLIL24", "BLIL30", "BLIL36", "BLIL42", "BLIL48", "BLIL54", "BLIL60", "BLIL66", "BLIL72", "BLIL78", "BLIL84", "BLIL90", "BLIL96", "BLIL102", "BLIL108", "BLIL114", "BLIL120", "BLIL126", "BLIL132", "BLIL138", "BLIL144", "BLIL150", "BLIL156", "BLIL162", "BLIL168", "BLIL174", "BLIL180", "BLIL186", "BLIL192", "BLIL198", "BLIL204", "BLIL210", "BLIL216", "BLIL222", "BLIL228", "BLIL234", "BLIL240", "BLIL246", "BLIL252", "BLIL258", "BLIL264", "BLIL270", "BLIL276", "BLIL282", "BLIL288", "BLIL294", "BLIL300", "BLIL306", "BLIL312", "BLIL318", "BLIL324", "BLIL330", "BLIL336", "BLIL342", "BLIL348", "BLIL354", "BLIL360", "BLIL366", "BLIL372", "BLIL378", "BLIL384", "BLIL390", "BLIL396", "BLIL402", "BLIL408", "BLIL414", "BLIL420", "BLIL426", "BLIL432", "BLIL438", "BLIL444", "BLIL450", "BLIL456", "BLIL462", "BLIL468", "BLIL474", "BLIL480", "BLIL486", "BLIL492", "BLIL498", "BLIL504", "BLIL510", "BLIL516", "BLIL522", "BLIL528", "BLIL534", "BLIL540", "BLIL546", "BLIL552", "BLIL558", "BLIL564", "BLIL570", "BLIL576", "BLIL582", "BLIL588", "BLIL594", "BLIL600", "BLIL606", "BLIL612", "BLIL618", "BLIL624", "BLIL630", "BLIL636", "BLIL642", "BLIL648", "BLIL654", "BLIL660", "BLIL666", "BLIL672", "BLIL678", "BLIL684", "BLIL690", "BLIL696", "BLIL702", "BLIL708", "BLIL714", "BLIL720", "BLIL726", "BLIL732", "BLIL738", "BLIL744", "BLIL750", "BLIL756", "BLIL762", "BLIL768", "BLIL774", "BLIL780", "BLIL786", "BLIL792", "BLIL798", "BLIL804", "BLIL810", "BLIL816", "BLIL822", "BLIL828", "BLIL834", "BLIL840", "BLIL846", "BLIL852", "BLIL858", "BLIL864", "BLIL870", "BLIL876", "BLIL882", "BLIL888", "BLIL894", "BLIL900", "BLIL906", "BLIL912", "BLIL918", "BLIL924", "BLIL930", "BLIL936", "BLIL942", "BLIL948", "BLIL954", "BLIL960", "BLIL966", "BLIL972", "BLIL978", "BLIL984", "BLIL990", "BLIL996", "BLIL1002", "BLIL1008", "BLIL1014", "BLIL1020", "BLIL1026", "BLIL1032", "BLIL1038", "BLIL1044", "BLIL1050", "BLIL1056", "BLIL1062", "BLIL1068", "BLIL1074", "BLIL1080", "BLIL1086", "BLIL1092", "BLIL1098", "BLIL1104", "BLIL1110", "BLIL1116", "BLIL1122", "BLIL1128", "BLIL1134", "BLIL1140", "BLIL1146", "BLIL1152", "BLIL1158", "BLIL1164", "BLIL1170", "BLIL1176", "BLIL1182", "BLIL1188", "BLIL1194", "BLIL1200", "BLIL1206", "BLIL1212", "BLIL1218", "BLIL1224", "BLIL1230", "BLIL1236", "BLIL1242", "BLIL1248", "BLIL1254", "BLIL1260", "BLIL1266", "BLIL1272", "BLIL1278", "BLIL1284", "BLIL1290", "BLIL1296", "BLIL1302", "BLIL1308", "BLIL1314", "BLIL1320", "BLIL1326", "BLIL1332", "BLIL1338", "BLIL1344", "BLIL1350", "BLIL1356", "BLIL1362", "BLIL1368", "BLIL1374", "BLIL1380", "BLIL1386", "BLIL1392", "BLIL1398", "BLIL1404", "BLIL1410", "BLIL1416", "BLIL1422", "BLIL1428", "BLIL1434", "BLIL1440", "BLIL1446", "BLIL1452", "BLIL1458", "BLIL1464", "BLIL1470", "BLIL1476", "BLIL1482", "BLIL1488", "BLIL1494", "BLIL1500", "BLIL1506", "BLIL1512", "BLIL1518", "BLIL1524", "BLIL1530", "BLIL1536", "BLIL1542", "BLIL1548", "BLIL1554", "BLIL1560", "BLIL1566", "BLIL1572", "BLIL1578", "BLIL1584", "BLIL1590", "BLIL1596", "BLIL1602", "BLIL1608", "BLIL1614", "BLIL1620", "BLIL1626", "BLIL1632", "BLIL1638", "BLIL1644", "BLIL1650", "BLIL1656", "BLIL1662", "BLIL1668", "BLIL1674", "BLIL1680", "BLIL1686", "BLIL1692", "BLIL1698", "BLIL1704", "BLIL1710", "BLIL1716", "BLIL1722", "BLIL1728", "BLIL1734", "BLIL1740", "BLIL1746", "BLIL1752", "BLIL1758", "BLIL1764", "BLIL1770", "BLIL1776", "BLIL1782", "BLIL1788", "BLIL1794", "BLIL1800", "BLIL1806", "BLIL1812", "BLIL1818", "BLIL1824", "BLIL1830", "BLIL1836", "BLIL1842", "BLIL1848", "BLIL1854", "BLIL1860", "BLIL1866", "BLIL1872", "BLIL1878", "BLIL1884", "BLIL1890", "BLIL1896", "BLIL1902", "BLIL1908", "BLIL1914", "BLIL1920", "BLIL1926", "BLIL1932", "BLIL1938", "BLIL1944", "BLIL1950", "BLIL1956", "BLIL1962", "BLIL1968", "BLIL1974", "BLIL1980", "BLIL1986", "BLIL1992", "BLIL1998", "BLIL2004", "BLIL2010", "BLIL2016", "BLIL2022", "BLIL2028", "BLIL2034", "BLIL2040", "BLIL2046", "BLIL2052", "BLIL2058", "BLIL2064", "BLIL2070", "BLIL2076", "BLIL2082", "BLIL2088", "BLIL2094", "BLIL2100", "BLIL2106", "BLIL2112", "BLIL2118", "BLIL2124", "BLIL2130", "BLIL2136", "BLIL2142", "BLIL2148", "BLIL2154", "BLIL2160", "BLIL2166", "BLIL2172", "BLIL2178", "BLIL2184", "BLIL2190", "BLIL2196", "BLIL2202", "BLIL2208", "BLIL2214", "BLIL2220", "BLIL2226", "BLIL2232", "BLIL2238", "BLIL2244", "BLIL2250", "BLIL2256", "BLIL2262", "BLIL2268", "BLIL2274", "BLIL2280", "BLIL2286", "BLIL2292", "BLIL2298", "BLIL2304", "BLIL2310", "BLIL2316", "BLIL2322", "BLIL2328", "BLIL2334", "BLIL2340", "BLIL2346", "BLIL2352", "BLIL2358", "BLIL2364", "BLIL2370", "BLIL2376", "BLIL2382", "BLIL2388", "BLIL2394", "BLIL2400", "BLIL2406", "BLIL2412", "BLIL2418", "BLIL2424", "BLIL2430", "BLIL2436", "BLIL2442", "BLIL2448", "BLIL2454", "BLIL2460", "BLIL2466", "BLIL2472", "BLIL2478", "BLIL2484", "BLIL2490", "BLIL2496", "BLIL2502", "BLIL2508", "BLIL2514", "BLIL2520", "BLIL2526", "BLIL2532", "BLIL2538", "BLIL2544", "BLIL2550", "BLIL2556", "BLIL2562", "BLIL2568", "BLIL2574", "BLIL2580", "BLIL2586", "BLIL2592", "BLIL2598", "BLIL2604", "BLIL2610", "BLIL2616", "BLIL2622", "BLIL2628", "BLIL2634", "BLIL2640", "BLIL2646", "BLIL2652", "BLIL2658", "BLIL2664", "BLIL2670", "BLIL2676", "BLIL2682", "BLIL2688", "BLIL2694", "BLIL2700", "BLIL2706", "BLIL2712", "BLIL2718", "BLIL2724", "BLIL2730", "BLIL2736", "BLIL2742", "BLIL2748", "BLIL2754", "BLIL2760", "BLIL2766", "BLIL2772", "BLIL2778", "BLIL2784", "BLIL2790", "BLIL2796", "BLIL2802", "BLIL2808", "BLIL2814", "BLIL2820", "BLIL2826", "BLIL2832", "BLIL2838", "BLIL2844", "BLIL2850", "BLIL2856", "BLIL2862", "BLIL2868", "BLIL2874", "BLIL2880", "BLIL2886", "BLIL2892", "BLIL2898", "BLIL2904", "BLIL2910", "BLIL2916", "BLIL2922", "BLIL2928", "BLIL2934", "BLIL2940", "BLIL2946", "BLIL2952", "BLIL2958", "BLIL2964", "BLIL2970", "BLIL2976", "BLIL2982", "BLIL2988", "BLIL2994", "BLIL3000", "BLIL3006", "BLIL3012", "BLIL3018", "BLIL3024", "BLIL3030", "BLIL3036", "BLIL3042", "BLIL3048", "BLIL3054", "BLIL3060", "BLIL3066", "BLIL3072", "BLIL3078", "BLIL3084", "BLIL3090",
```

```

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], nodesize=nodesizes[j])
    control.forest <- randomForest(Health~., data=training.control.data, mtry=mtrys[i], nodesize=nodesizes[j])

    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

```

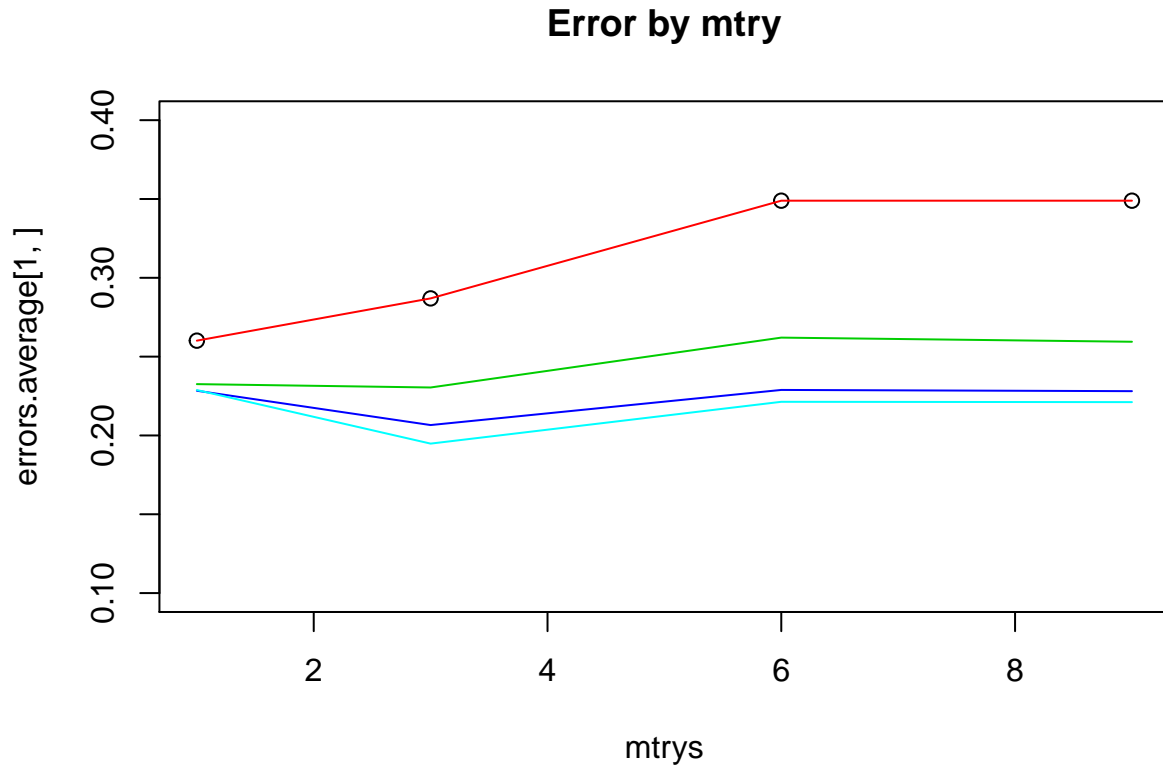
```

##           [,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

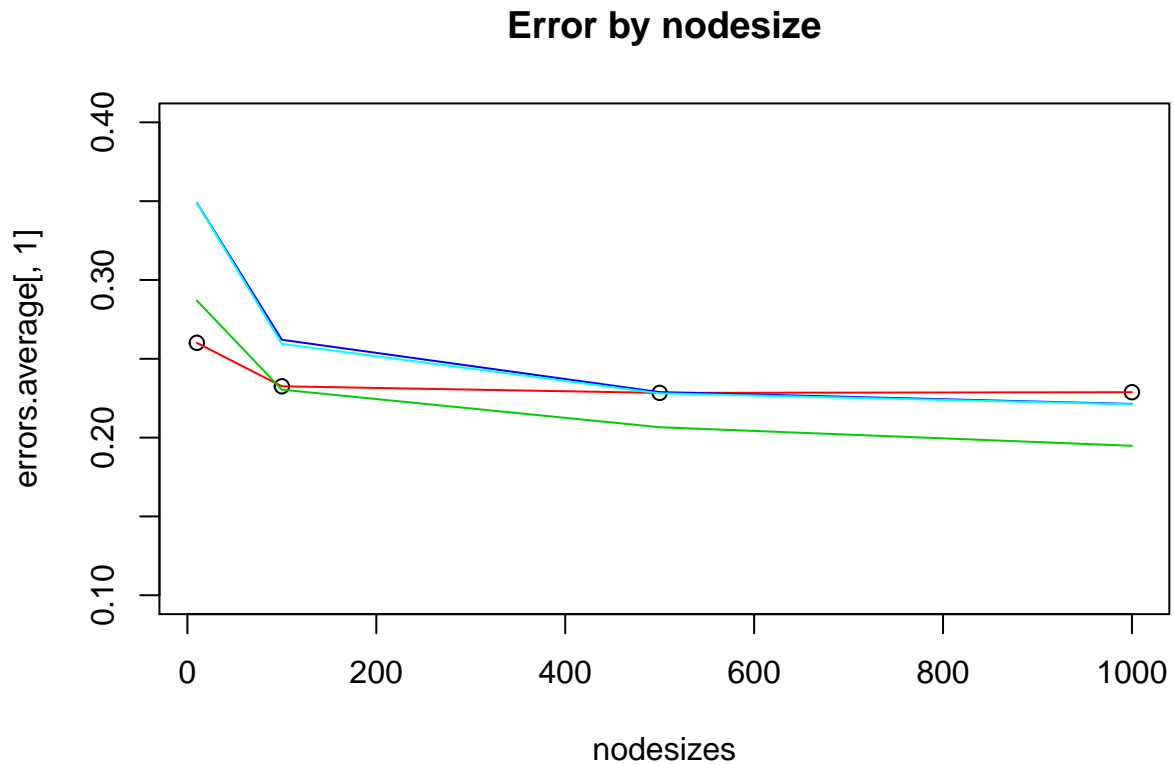
```

## 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())
library(randomForest)
set.seed(1)

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

data.cols <- c("Health", "PRAPACHE", "AGE", "BLGCS", "ORGANUM", "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)

result <- predictions==raw.data["BEST"]
accuracy <- sum(result==TRUE)/length(result)

accuracy

```

```
## [1] 0.8595745
```

Accuracy is about 86%

### Train decision tree

```

library(rpart)
library(rpart.plot)

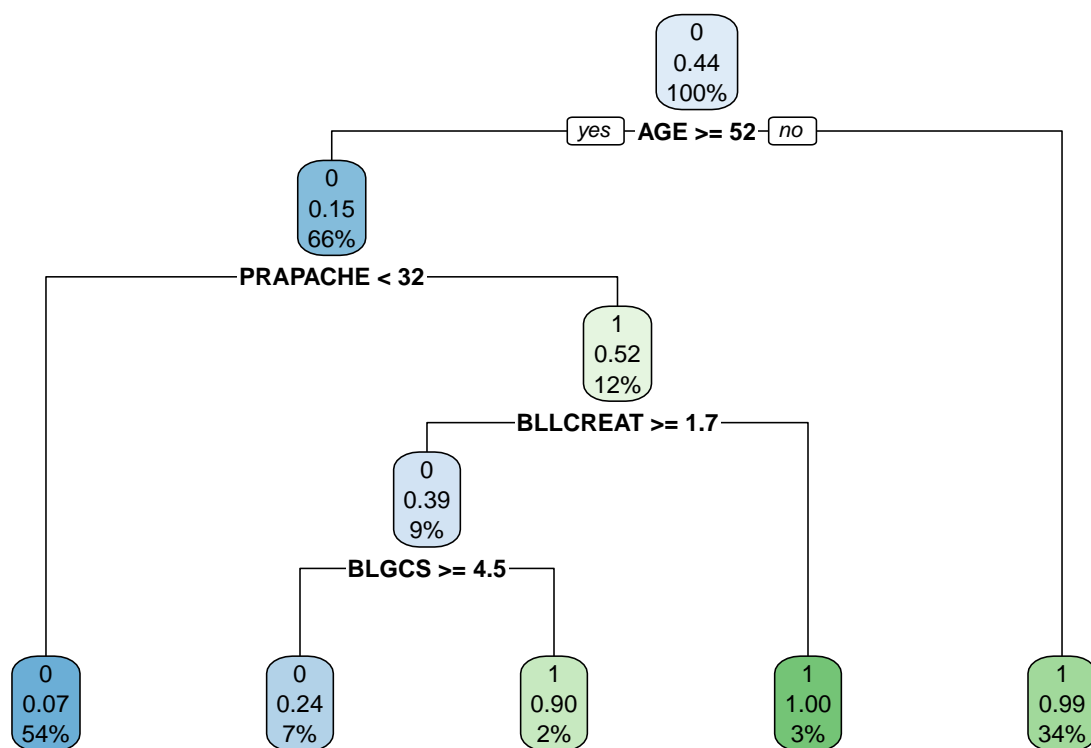
data.cols <- c("PRAPACHE", "AGE", "BLGCS", "ORGANUM", "BLIL6", "BLLPLAT", "BLLBILI", "BLLCREAT", "TIMFIRST", "B")

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.

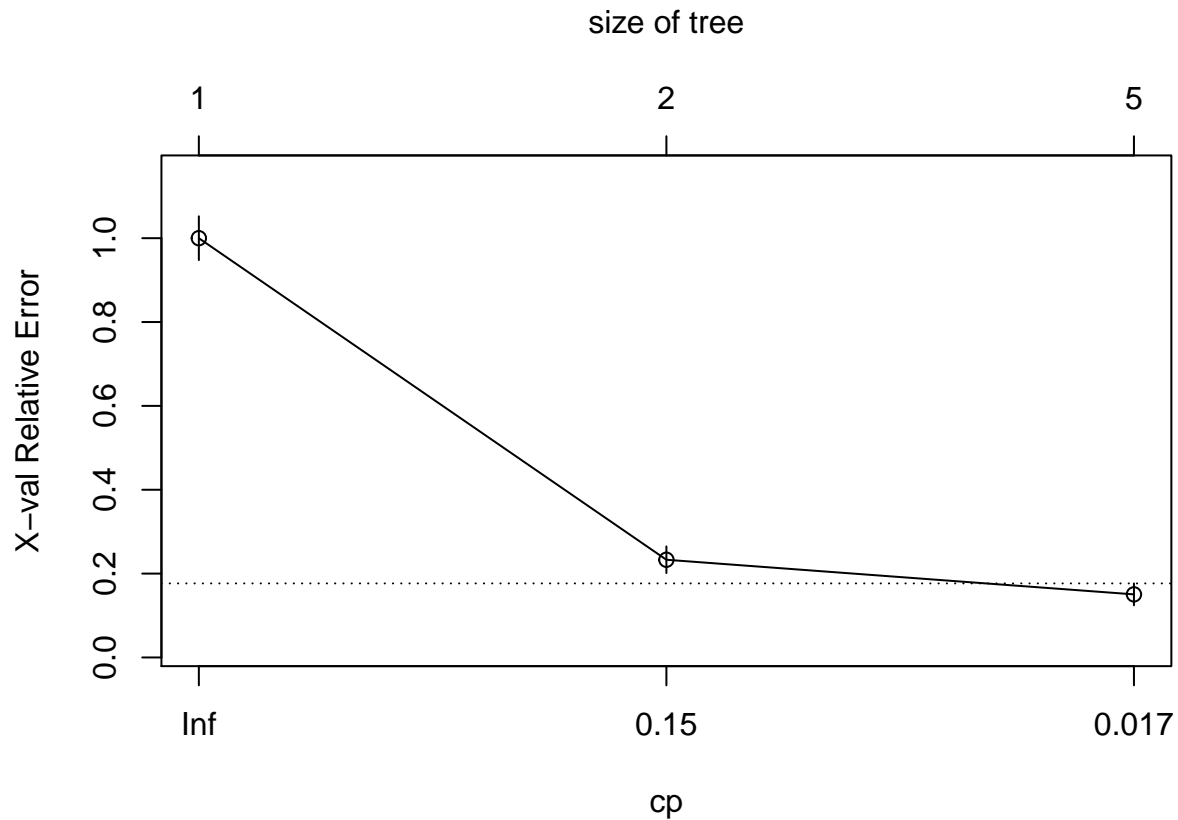
```



```
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) *
```

```
predictions.tree <- predict(rpart.fit, raw.data.data)
```

```
pre <- ifelse(predictions.tree[,2]>0.5,1,0)
result <- pre==raw.data["BEST"]
accuracy <- sum(result==TRUE)/length(result)
```

```
accuracy
```

```
## [1] 0.8340426
```

Accuracy of using Decision tree is about 83%