

# CS771A Assignment 1: Decision Trees

Saurav Kumar (12641)

January 19, 2014

```
library(rpart)
library(rpart.plot)
set.seed(10)
rawData = read.csv(file="data", header=F, sep=",")
originalData = rawData[sample(nrow(rawData)),]
colnames(originalData) = c("PregnantCount", "Glucose", "BP", "Triceps",
                           "Insulin", "BMI", "DPF", "Age", "Class")

N = nrow(originalData)
K = 5
foldWidth = floor(N/K)
Accuracy = 0
for (i in (1:K))
{
  data = originalData
  start = as.integer((i-1)*foldWidth)+1
  end = as.integer(i*foldWidth)
  if(i==K)
  {
    end = N
  }
  testData = data[c(start:end),]
  learnData = data[c(-start:-end),]

  nonZerosCount = colSums(learnData!=0)
  meanVals = colSums(learnData)/nonZerosCount
  learnData$Glucose[learnData$Glucose==0] = meanVals["Glucose"]
  learnData$BP[learnData$BP==0] = meanVals["BP"]
  learnData$Triceps[learnData$Triceps==0] = meanVals["Triceps"]
  learnData$Insulin[learnData$Insulin==0] = meanVals["Insulin"]
  learnData$BMI[learnData$BMI==0] = meanVals["BMI"]

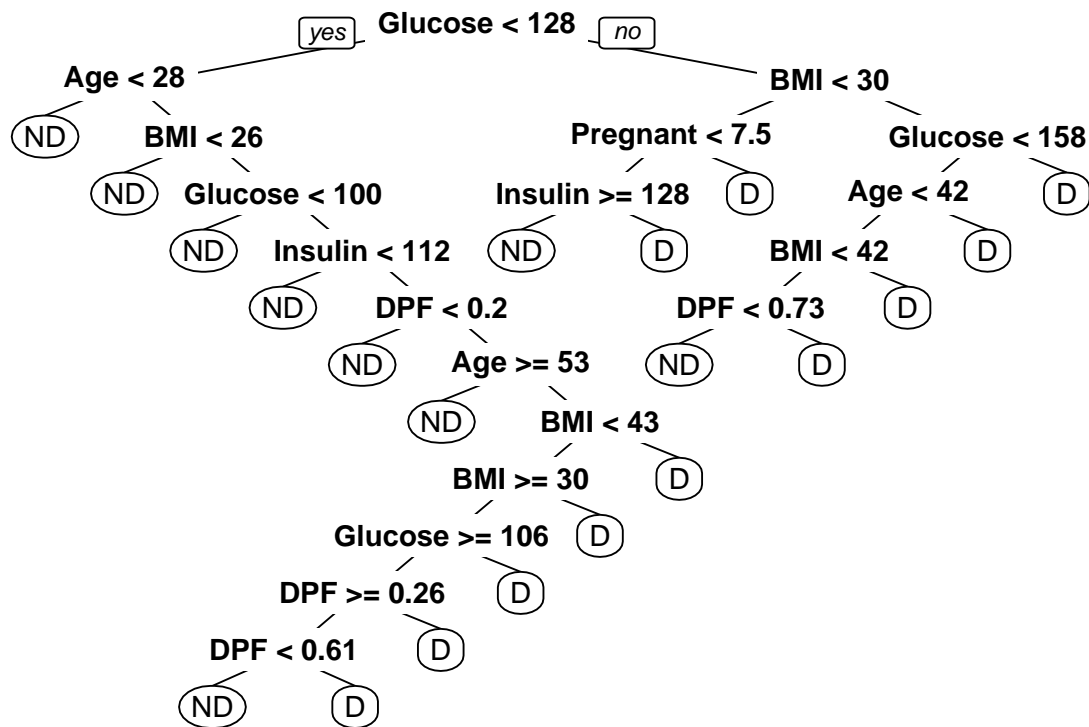
  testData$Glucose[testData$Glucose==0] = NA
  testData$BP[testData$BP==0] = NA
  testData$Triceps[testData$Triceps==0] = NA
  testData$Insulin[testData$Insulin==0] = NA
  testData$BMI[testData$BMI==0] = NA
  diabStat = factor(learnData$Class, levels=0:1, labels=c('ND','D'))
  ## Any split that does not decrease the overall lack of fit by a factor
  ## of cp is not attempted. By setting appropriate (here, 0.004), we are
  ## controlling the growth of tree by thresholding impurity decrease.
  cfit = rpart(
    diabStat ~ PregnantCount+Glucose+BP+Triceps+Insulin+BMI+DPF+Age,
    data = learnData,
    na.action = na.rpart,
    method = 'class',
    parms = list(split = "gini"),
    control = rpart.control(
```

```

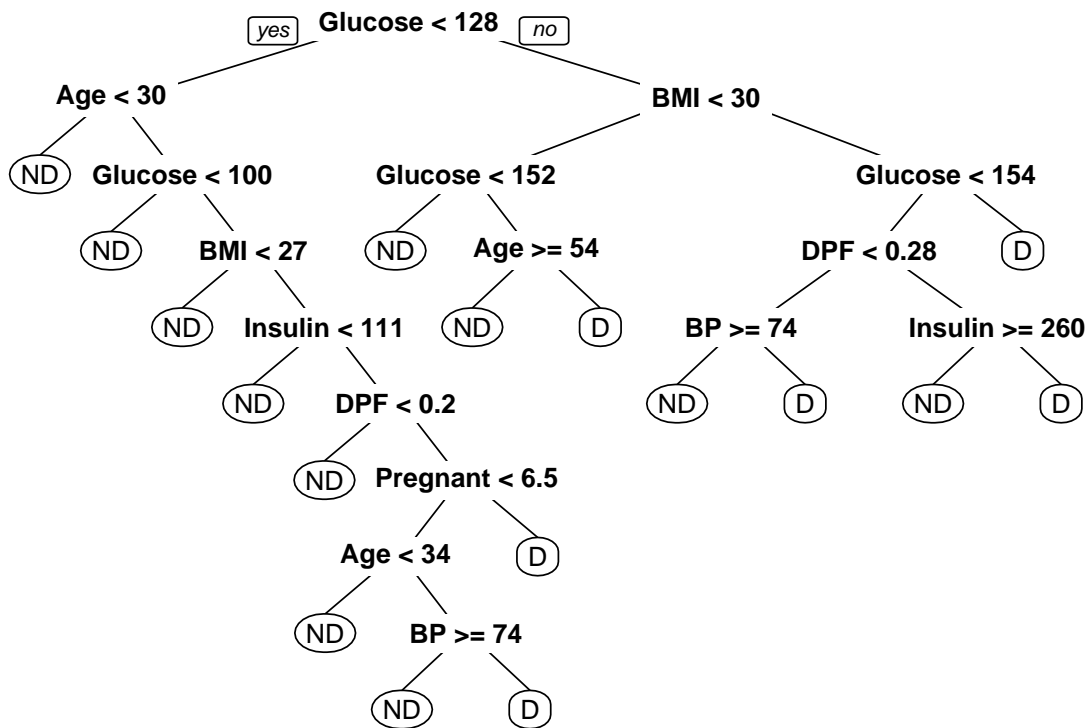
    cp = 0.004,
  )
)
predictedFactor = predict(cfit, testData, type="class")
predictedFrame = as.data.frame.factor(predictedFactor)
predicted = c(predictedFrame[,1]) - 1 # gives 1 for ND, 2 for D
actual = testData$Class
TP = sum(predicted & actual)
TN = nrow(testData) - sum(predicted | actual)
# Accuracy
print((TP+TN)/nrow(testData))
Accuracy = Accuracy + (TP+TN)/nrow(testData)
rpart.plot(cfit)
}

```

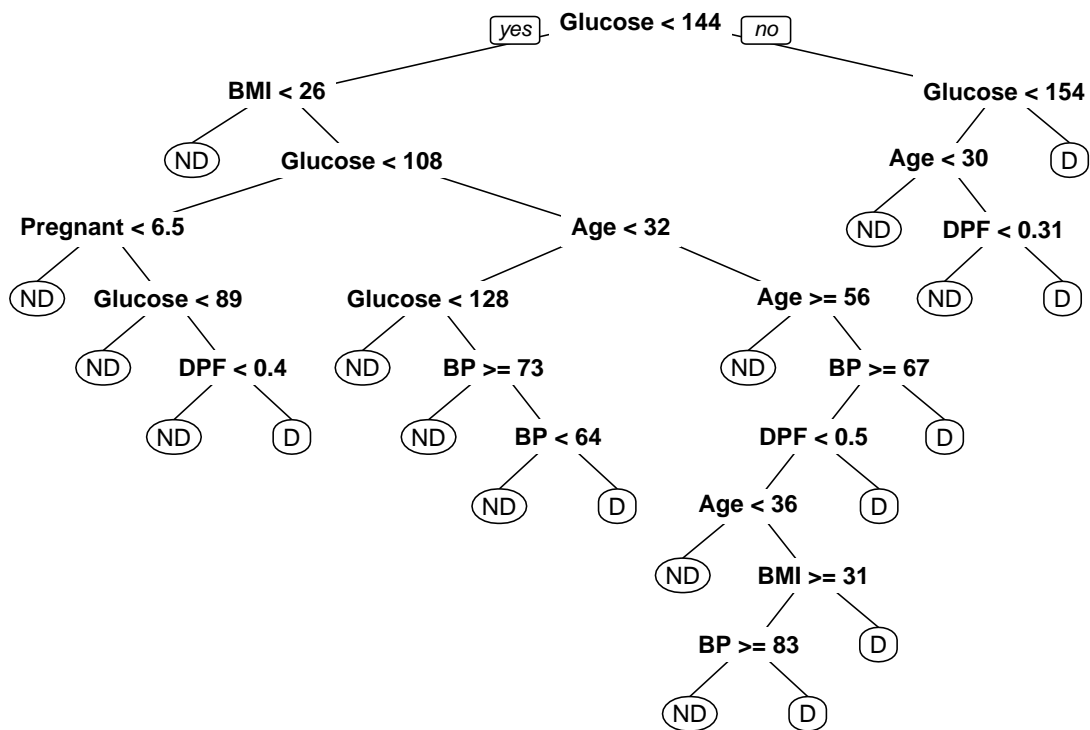
```
## [1] 0.7385621
```



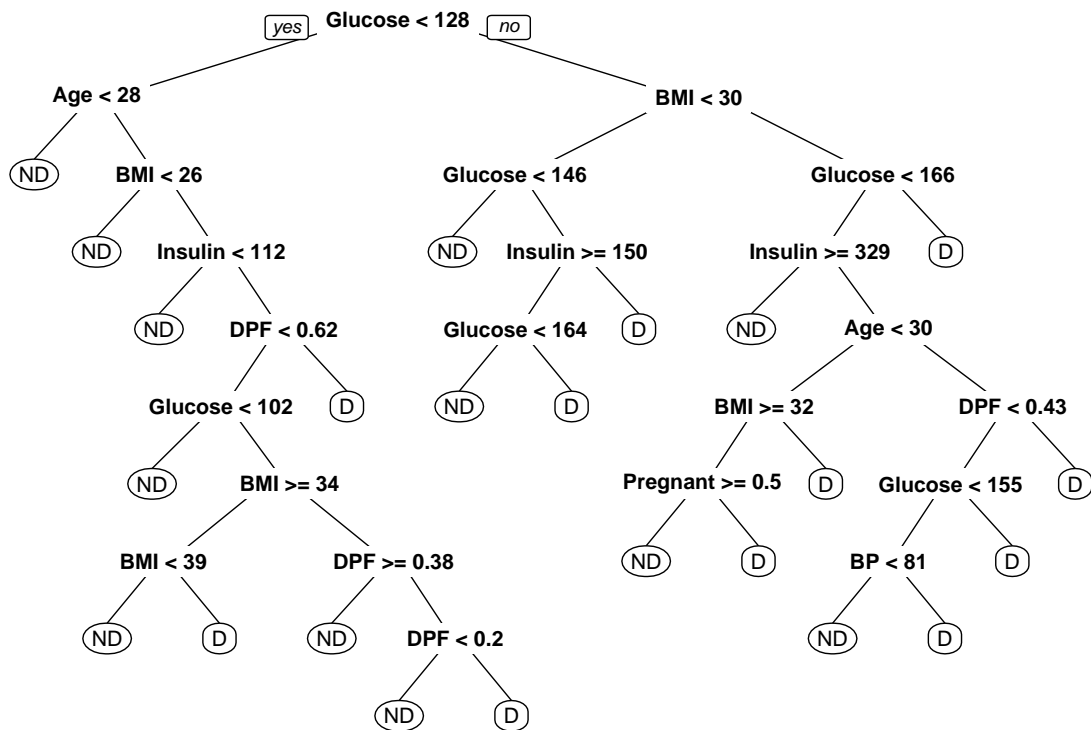
```
## [1] 0.7581699
```



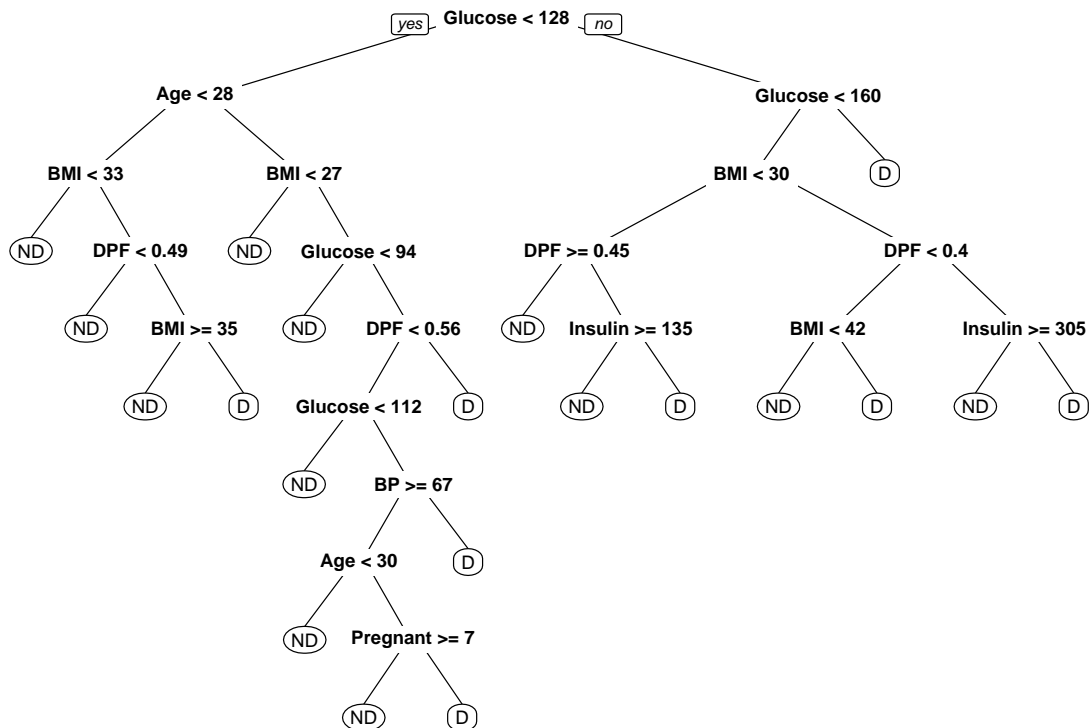
## [1] 0.7581699



## [1] 0.7647059



## [1] 0.7820513



# Mean Accuracy  
print(Accuracy/K)

## [1] 0.7603318