In Class Assignment 6

Lydia Strebe

March 27, 2019

Predicting Heart Disease with Trees

In this document we will use different tree models to predict whether or not a patient has heart disease.

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 3.5.2
```

```
library(MASS)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.2
```

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

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.5.2
```

```
## Loaded gbm 2.1.5
```

```
Heart_all=read.csv("C:/Users/lydia/Documents/Heart.csv",header=TRUE)

#Remove the rows with missing data
Heart=na.omit(Heart_all)

set.seed(1)

#split the data randomly into a training set of size 200 and a test set of size 97
test_rows=sample(1:nrow(Heart),size=97,replace=FALSE)
Heart_train=Heart[-test_rows,]
Heart_test=Heart[test_rows,]
```

Single Tree

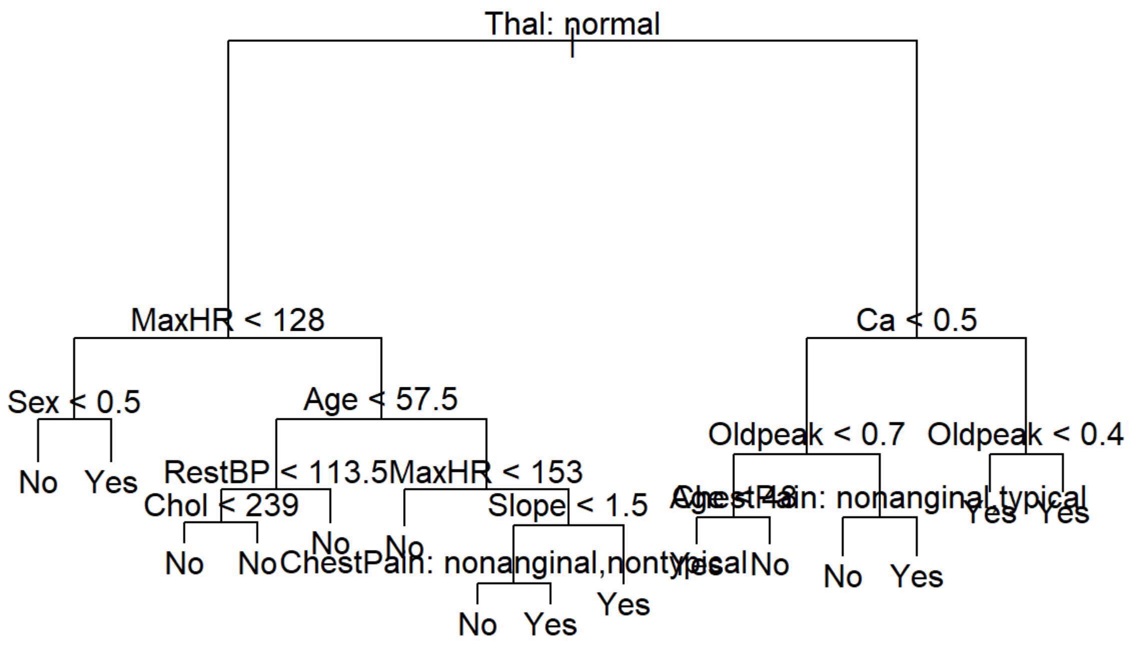
We first build a single tree with all the training data.

```
#build a single tree
tree.heart=tree(AHD~.-X,Heart_train)

#summary info
summary(tree.heart)
```

```
##
## Classification tree:
## tree(formula = AHD ~ . - X, data = Heart_train)
## Variables actually used in tree construction:
##  [1] "Thal"      "MaxHR"     "Sex"       "Age"       "RestBP"
##  [6] "Chol"      "Slope"     "ChestPain" "Ca"        "Oldpeak"
## Number of terminal nodes:  15
## Residual mean deviance:  0.461 = 85.29 / 185
## Misclassification error rate: 0.095 = 19 / 200
```

```
plot(tree.heart)
text(tree.heart,pretty=0)
```



We then run our test data through the tree and compare our prediction to the actual results.

```
#prediction
tree.pred=predict(tree.heart,Heart_test,type="class")
table(tree.pred,Heart_test$AHD)
```

```
##
## tree.pred No Yes
##       No  39  14
##       Yes 12  32
```

The proportion of patients correctly classified in the test set is calcuated below.

```
#correct classification
(39+32)/97
```

```
## [1] 0.7319588
```

Conversly, the misclassification error is

```
#misclassification
(14+12)/97
```

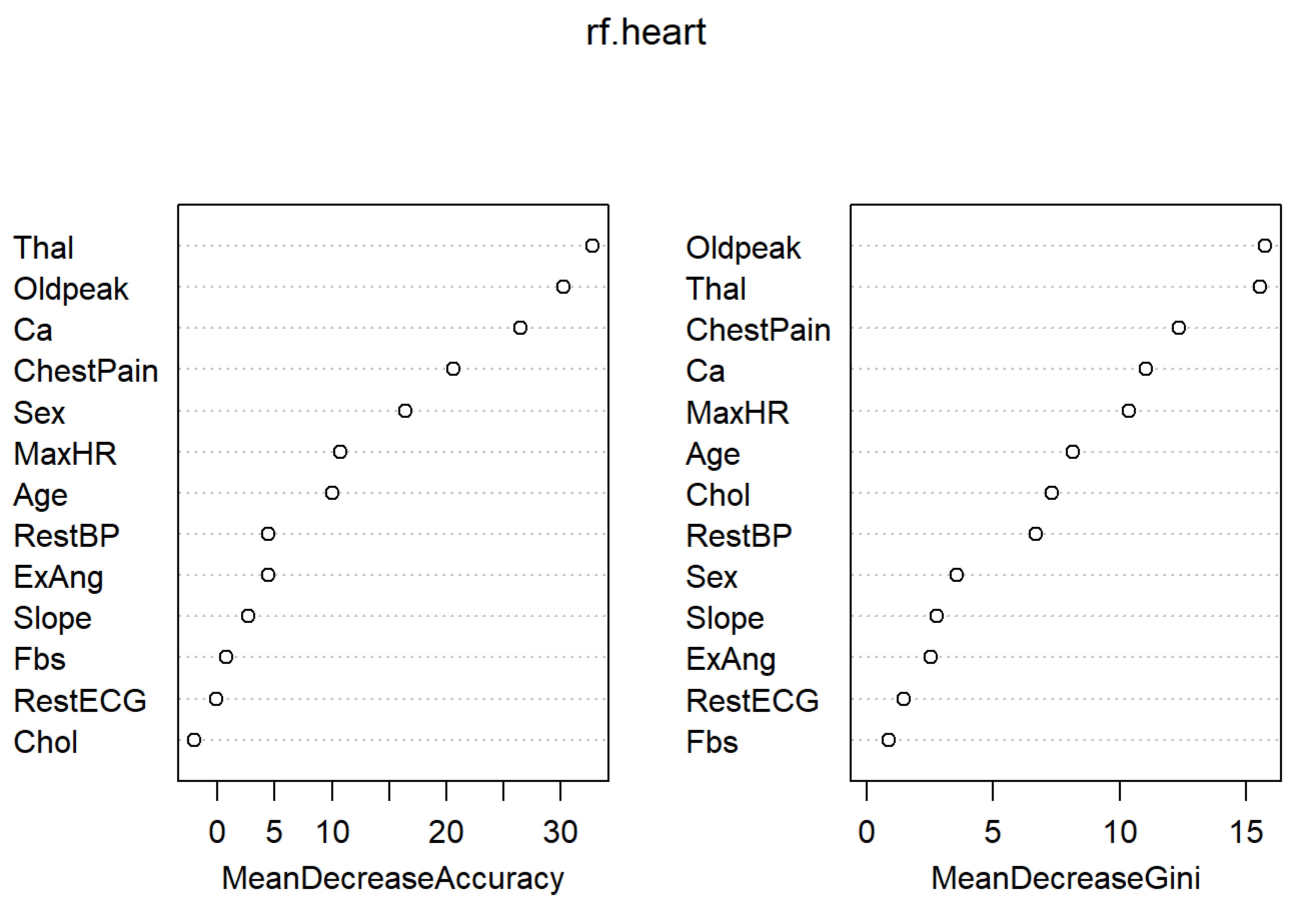
```
## [1] 0.2680412
```

Random Forest

Next we use Random Forest to make a model. Per conventional wisdom, we split 4 times because it is roughly equal to the square root of 13, the number of regressors we are using.

```
#Random Forest with 1000 trees
rf.heart=randomForest(AHD~.-X,data=Heart_train,ntree=1000,mtry=4,importance=TRUE)

#summary info
varImpPlot(rf.heart)
```



We then test our prediction.

```
#prediction
ypred.rf=predict(rf.heart,Heart_test)
table(ypred.rf,Heart_test$AHD)
```

```
##
## ypred.rf No Yes
##      No  40  12
##      Yes 11  34
```

The misclassification error is

```
(12+11)/97
```

```
## [1] 0.2371134
```

Which means that the proportion of patients correctly classified is

```
(40+34)/97
```

```
## [1] 0.7628866
```

We don’t need to choose the number of trees precisely (there is no danger of overfitting), but we want to make sure that our Random Forest has enough trees, so we compare our results above to Random Forest models with 500 and 1500 trees respectively.

```
#Random Forest with default 500 trees
rf.heart2=randomForest(AHD~.-X,data=Heart_train,mtry=4,importance=TRUE)
#prediction
ypred.rf2=predict(rf.heart2,Heart_test)
table(ypred.rf2,Heart_test$AHD)
```

```
##
## ypred.rf2 No Yes
##       No  42  15
##       Yes  9  31
```

```
#misclassification
(15+9)/97
```

```
## [1] 0.2474227
```

```
#correct classification
(42+31)/97
```

```
## [1] 0.7525773
```

```
#Random Forest with 1500 trees
rf.heart3=randomForest(AHD~.-X,data=Heart_train,ntree=1500,mtry=4,importance=TRUE)
#prediction
ypred.rf3=predict(rf.heart3,Heart_test)
table(ypred.rf3,Heart_test$AHD)
```

```
##
## ypred.rf3 No Yes
##       No  40  12
##       Yes 11  34
```

```
#misclassification
(12+11)/97
```

```
## [1] 0.2371134
```

```
#correct classification
(40+34)/97
```

```
## [1] 0.7628866
```

The results are about the same.

AdaBoost

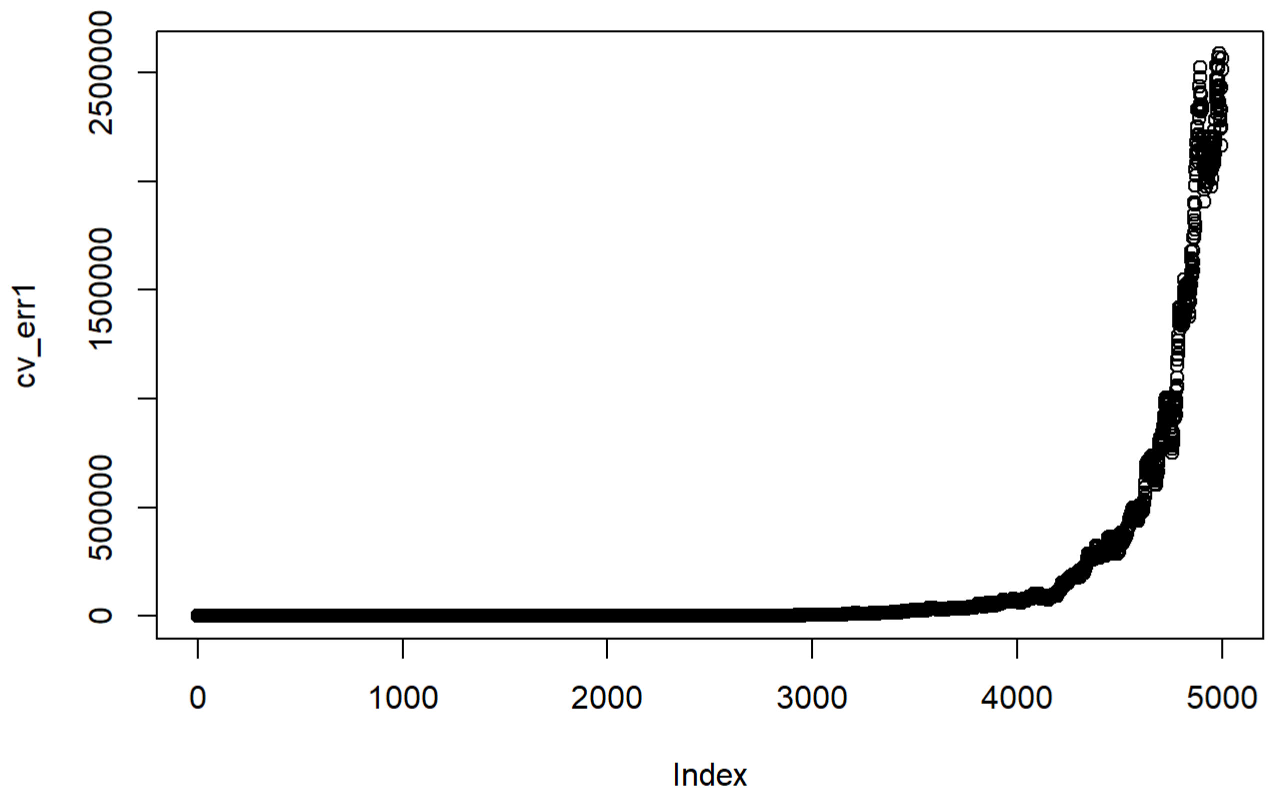
Finally, we use AdaBoost. For AdaBoost it is possible to underfit or overfit the data, so we use 10-fold cross validation to find the best number of trees at depths of 1, 2, and 3.

```
#preparing the data
AHD01=ifelse(Heart$AHD=="No",0,1)
Heart2=data.frame(Heart,AHD01)
extra_columns=c(1,15,17)
Heart_train2=Heart2[-test_rows,-extra_columns]
Heart_test2=Heart2[test_rows,-extra_columns]

set.seed(1)

#tuning hyperparameters

#depth of 1
boost.heart1=gbm(AHD01~.,data=Heart_train2,distribution="adaboost",n.trees=5000,interaction.dept
h=1,cv.folds=10)
cv_err1=boost.heart1$cv.error
plot(cv_err1)
```



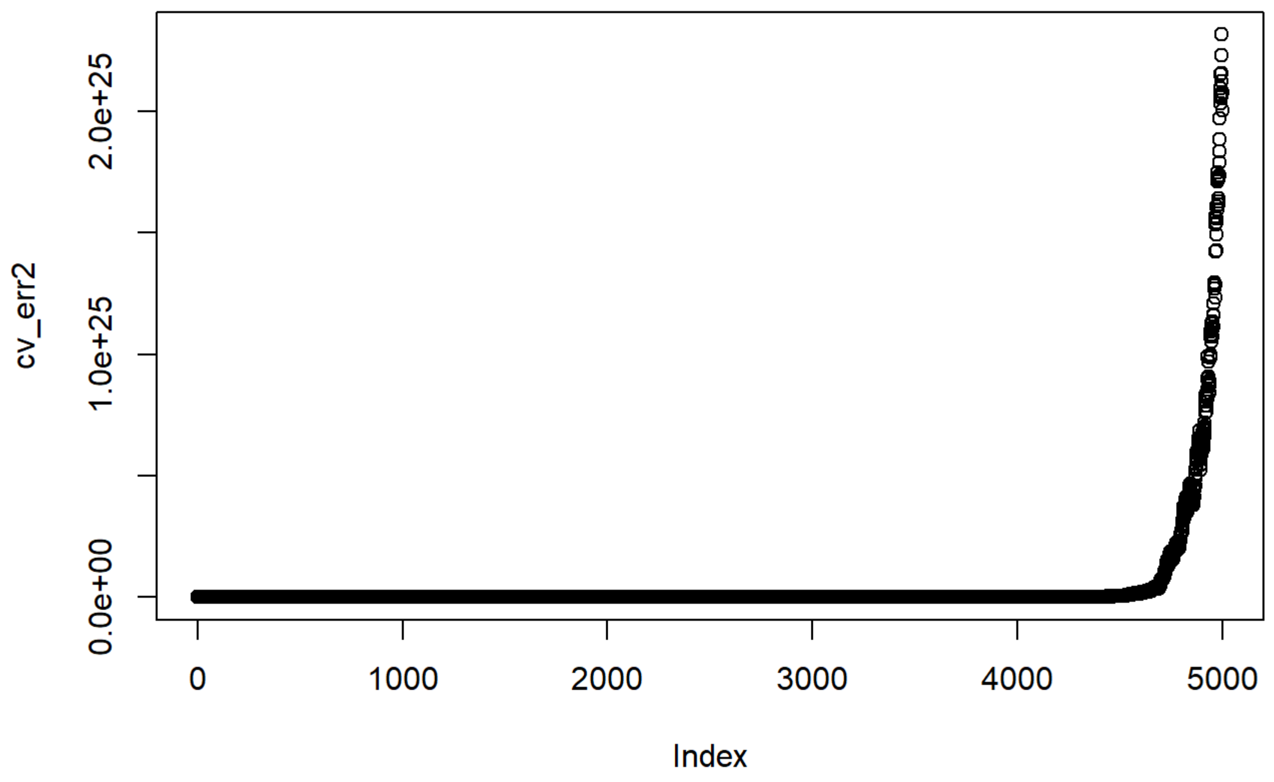
```
#lowest error
min(cv_err1)
```

```
## [1] 0.6260804
```

```
#number of trees with lowest error
which.min(cv_err1)
```

```
## [1] 78
```

```
#depth of 2
boost.heart2=gbm(AHD01~.,data=Heart_train2,distribution="adaboost",n.trees=5000,interaction.dept
h=2,cv.folds=10)
cv_err2=boost.heart2$cv.error
plot(cv_err2)
```



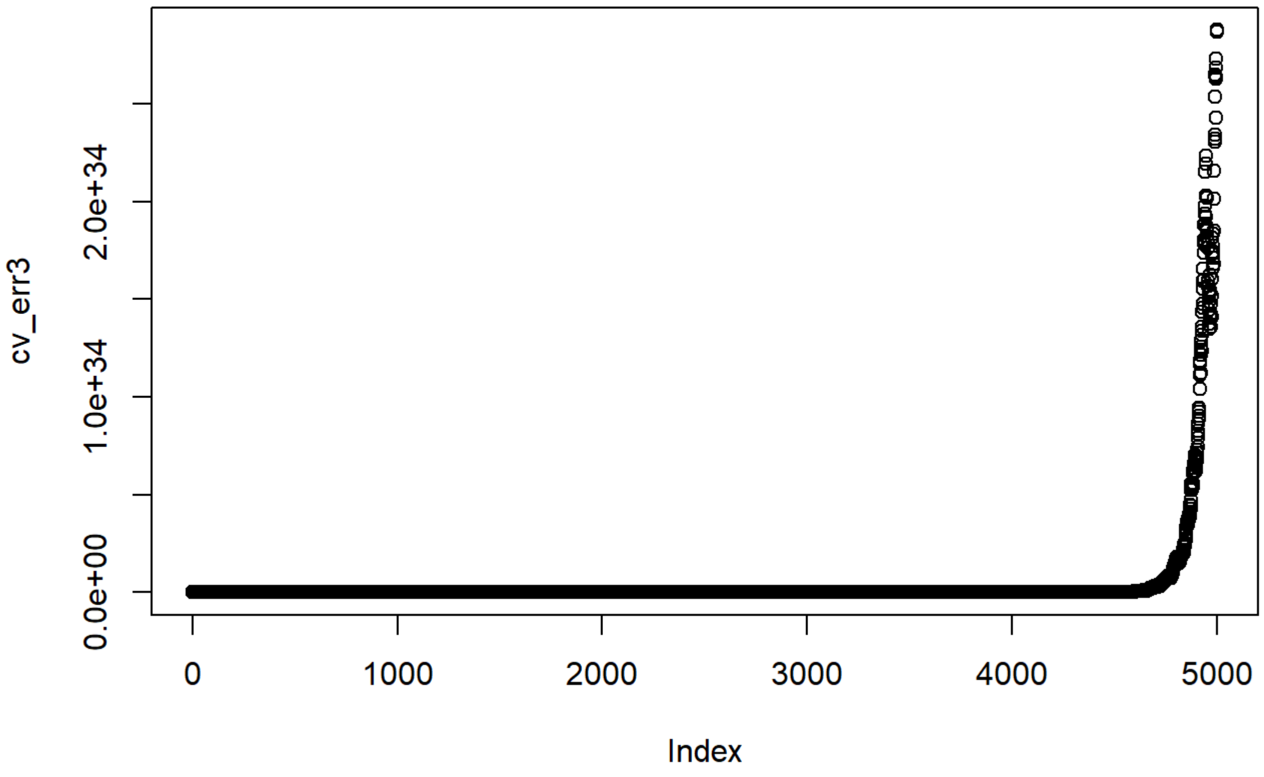
```
#lowest error
min(cv_err2)
```

```
## [1] 0.6611403
```

```
#number of trees with lowest error
which.min(cv_err2)
```

```
## [1] 51
```

```
#depth of 3
boost.heart3=gbm(AHD01~.,data=Heart_train2,distribution="adaboost",n.trees=5000,interaction.dept
h=3,cv.folds=10)
cv_err3=boost.heart3$cv.error
plot(cv_err3)
```



```
#lowest error
min(cv_err3)
```

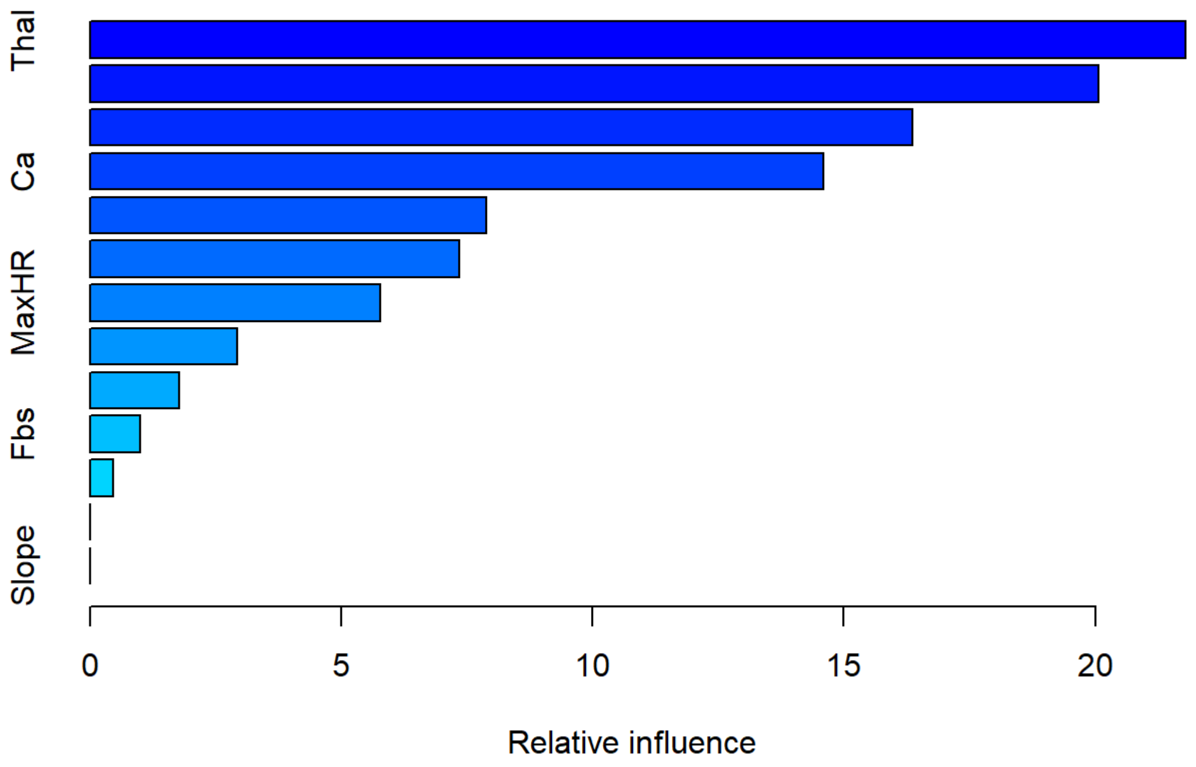
```
## [1] 0.6537091
```

```
#number of trees with lowest error
which.min(cv_err3)
```

```
## [1] 38
```

Overall, our lowest error was at a depth of 1 with 78 trees.

```
#summary info of best model
boost.heart=gbm(AHD01~.,data=Heart_train2,distribution="adaboost",n.trees=78,interaction.depth=1
)
summary(boost.heart)
```



```
##                var     rel.inf
## Thal          Thal 21.7946955
## Oldpeak    Oldpeak 20.0713899
## ChestPain ChestPain 16.3594392
## Ca              Ca 14.5934020
## Chol          Chol  7.8926725
## Sex            Sex  7.3451470
## MaxHR        MaxHR  5.7765290
## RestBP      RestBP  2.9338850
## Age            Age  1.7783026
## Fbs            Fbs  1.0001830
## RestECG    RestECG  0.4543542
## ExAng        ExAng  0.0000000
## Slope        Slope  0.0000000
```

As before, we run our test data through the model and see how our prediction compares to the actual data.

```
#prediction
ypred.ada = predict(boost.heart,Heart_test2,type="link",n.trees=78,interaction.depth=1)
table(ypred.ada>0,Heart_test2$AHD01)
```

```
##
##          0  1
##   FALSE 39 12
##   TRUE  12 34
```

```
#correct classification
(39+34)/97
```

```
## [1] 0.7525773
```

```
#misclassification
(12+12)/97
```

```
## [1] 0.2474227
```

Conclusion

For this iteration, our best model turned out to be Random Forest using 4 splits and 1000 trees with a misclassification error of 0.2371134.