Model selection using regression trees

Marek Chadim

10-04-2024

Exercise 1: Within-time period prediction

Data Prep

```
d <- read.csv("growthdata92_02.csv")
d <- d[, 3:ncol(d)]
set.seed(42)
train <- sample(1:nrow(d), nrow(d) * .8)
d.test <- d[-train, "growth"]
```

Regression Trees

```
library(tree)
tree.d <- tree(growth ~ ., d, subset = train)
cv.d <- cv.tree(tree.d)
size <- cv.d$size[which.min(cv.d$dev)]
prune.d <- prune.tree(tree.d, best = size)
yhat <- predict(prune.d, newdata = d[-train, ])
rmse_tree <- sqrt(mean((yhat - d.test)^2))
rmse_tree
```

```
## [1] 0.02953695
```

Bagging and Random Forests

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```
mtry_values <- seq(1, ncol(d) - 1, by = 1)
oob_error_values <- numeric(length(mtry_values))

bag.d <- randomForest(growth ~ ., data = d, mtry = length(mtry_values), subset = train, importance = TRU
yhat.bag <- predict(bag.d, newdata = d[-train, ])
rmse_bag <- sqrt(mean((yhat.bag - d.test)^2))
rmse_bag
```

```
## [1] 0.03008512
```

```r
for (i in 1:length(mtry_values)) {
  rf.d <- randomForest(growth ~ ., data = d, mtry = mtry_values[i], subset = train, importance = TRUE)
  oob_error_values[i] <- rf.d$mse[rf.d$ntree]
}
best_mtry <- mtry_values[which.min(oob_error_values)]
rf.d <- randomForest(growth ~ ., data = d, mtry = best_mtry, subset = train, importance = TRUE)
yhat.rf <- predict(rf.d, newdata = d[-train, ])
rmse_rf <- sqrt(mean((yhat.rf - d.test)^2))
rmse_rf
```

```
## [1] 0.0289186
```

Boosting

```r
library(gbm)
```

```
## Loaded gbm 2.2.2

## This version of gbm is no longer under development. Consider transitioning to gbm3, https://github.co
```

```r
boost.d <- gbm(growth ~ ., data = d[train, ], distribution = "laplace", n.trees = 10000, interaction.dep
yhat.boost <- predict(boost.d, newdata = d[-train, ], n.trees = 10000)
rmse_boost <- sqrt(mean((yhat.boost - d.test)^2))
rmse_boost
```

```
## [1] 0.02339858
```

Bayesian Additive Regression Trees

```r
library(BART)
```

```
## Loading required package: nlme

## Loading required package: survival
```

```r
x <- d[, -2]
y <- d[, "growth"]
xtrain <- x[train, ]
ytrain <- y[train]
xtest <- x[-train, ]
ytest <- y[-train]
bartfit <- gbart(xtrain, ytrain, x.test = xtest)
```

```
## *****Calling gbart: type=1
## *****Data:
## data:n,p,np: 89, 27, 23
## y1,yn: 0.053894, 0.018602
## x1,x[n*p]: 0.900275, 0.337561
## xp1,xp[np*p]: -0.862508, -0.980124
## *****Number of Trees: 200
## *****Number of Cut Points: 88 ... 77
## *****burn,nd,thin: 100,1000,1
## *****Prior:beta,alpha,tau,nu,lambda,offset: 2,0.95,0.00343254,3,7.40032e-05,0.0187056
## *****sigma: 0.019491
## *****w (weights): 1.000000 ... 1.000000
```

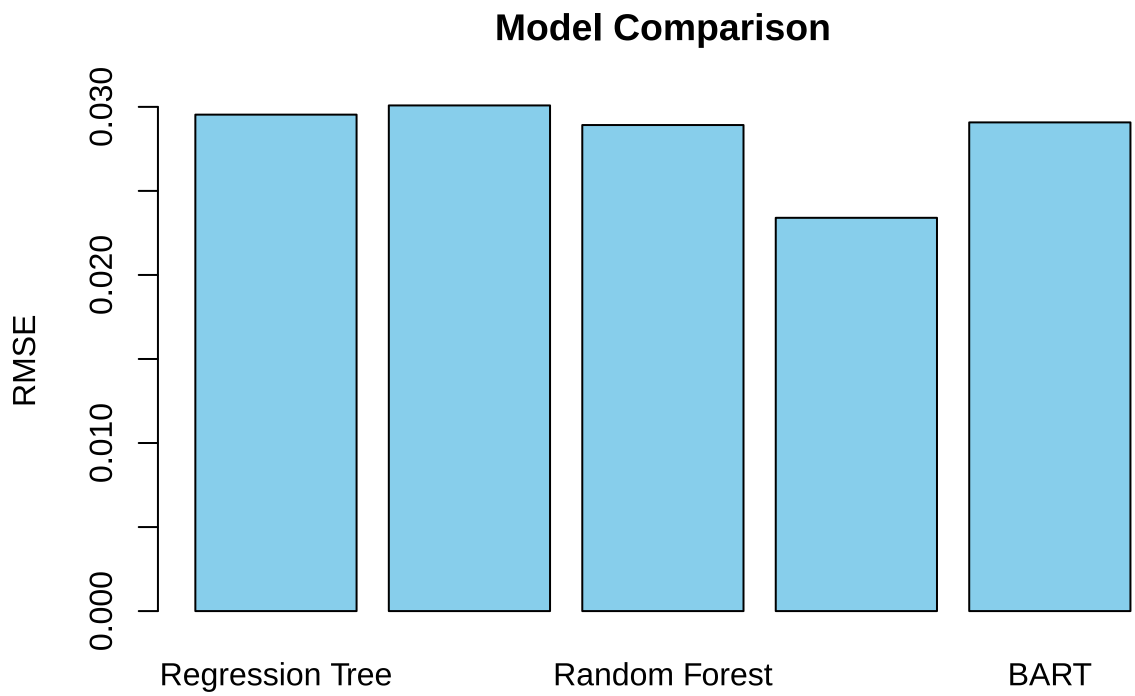
```
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,27,0
## *****printevery: 100
##
## MCMC
## done 0 (out of 1100)
## done 100 (out of 1100)
## done 200 (out of 1100)
## done 300 (out of 1100)
## done 400 (out of 1100)
## done 500 (out of 1100)
## done 600 (out of 1100)
## done 700 (out of 1100)
## done 800 (out of 1100)
## done 900 (out of 1100)
## done 1000 (out of 1100)
## time: 2s
## trcnt,tecnt: 1000,1000
```

```r
ord <- order(bartfit$varcount.mean, decreasing = T)
yhat.bart <- bartfit$yhat.test.mean
rmse_bart <- sqrt(mean((ytest - yhat.bart)^2))
rmse_bart
```

```
## [1] 0.02907511
```

Model Comparison

```r
models <- c("Regression Tree", "Bagging", "Random Forest", "Boosting", "BART")
rmse_values <- c(rmse_tree, rmse_bag, rmse_rf, rmse_boost, rmse_bart)
rmse_comparison <- data.frame(Model = models, RMSE = rmse_values)
barplot(rmse_comparison$RMSE, names.arg = rmse_comparison$Model, col = "skyblue", main = "Model Compari
```



Exercise 2: Out-of-sample prediction

```
d_92_02 <- read.csv("growthdata92_02.csv")
d_92_02 <- d_92_02[, 3:ncol(d_92_02)]
d_02_11 <- read.csv("growthdata02_11.csv")
d_02_11 <- d_02_11[, 3:ncol(d_02_11)]
x_train <- model.matrix(growth ~ ., d_92_02)[, -1]
y_train <- d_92_02$growth
x_test <- model.matrix(growth ~ ., d_02_11)[, -1]
y_test <- d_02_11$growth
```

Regression Tree

```
tree.d <- tree(growth ~ ., data = d_92_02)
cv.d <- cv.tree(tree.d)
size <- cv.d$size[which.min(cv.d$dev)]
prune.d <- prune.tree(tree.d, best = size)
yhat_tree <- predict(prune.d, newdata = d_02_11)
rmse_tree <- sqrt(mean((yhat_tree - y_test)^2))
rmse_tree
```

```
## [1] 0.02684352
```

Bagging

```
bag.d <- randomForest(growth ~ ., data = d_92_02, mtry = length(mtry_values), importance = TRUE)
yhat_bag <- predict(bag.d, newdata = d_02_11)
rmse_bag <- sqrt(mean((yhat_bag - y_test)^2))
rmse_bag
```

```
## [1] 0.02878643
```

Random Forest

```
oob_error_values2 <- numeric(length(mtry_values))
for (i in 1:length(mtry_values)) {
  rf.d <- randomForest(growth ~ ., data = d_92_02, mtry = mtry_values[i], subset = train, importance = 
  oob_error_values2[i] <- rf.d$mse[rf.d$ntree]
}
best_mtry2 <- mtry_values[which.min(oob_error_values2)]
rf.d <- randomForest(growth ~ ., data = d_92_02, mtry = best_mtry2, subset = train, importance = TRUE)
yhat_rf <- predict(rf.d, newdata = d_02_11)
rmse_rf <- sqrt(mean((yhat_rf - y_test)^2))
rmse_rf
```

```
## [1] 0.03074198
```

Boosting

```
boost.d <- gbm(growth ~ ., data = d_92_02, distribution = "gaussian", n.trees = 10000, shrinkage = 0.00
yhat_boost <- predict(boost.d, newdata = d_02_11, n.trees = 10000)
rmse_boost <- sqrt(mean((yhat_boost - y_test)^2))
rmse_boost
```

```
## [1] 0.03431188
```

BART

```
x <- d_92_02[, -2]
y <- d_92_02[, "growth"]
bartfit <- gbart(x, y, x.test = d_02_11[, -2])
```

```
## *****Calling gbart: type=1
## *****Data:
## data:n,p,np: 112, 27, 112
## y1,yn: 0.067975, -0.003520
## x1,x[n*p]: -0.862508, 0.567224
## xp1,xp[np*p]: -0.149882, 0.224655
## *****Number of Trees: 200
## *****Number of Cut Points: 100 ... 94
## *****burn,nd,thin: 100,1000,1
## *****Prior:beta,alpha,tau,nu,lambda,offset: 2,0.95,0.00343254,3,8.86306e-05,0.0202264
## *****sigma: 0.021331
## *****w (weights): 1.000000 ... 1.000000
```

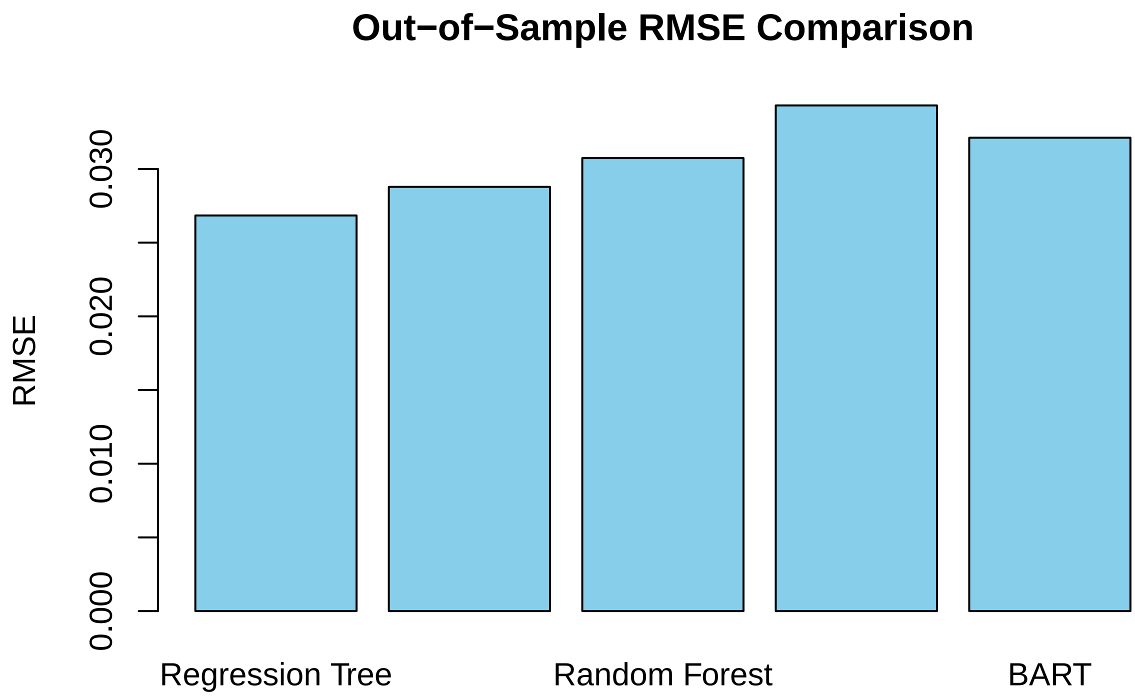
```
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,27,0
## *****printevery: 100
##
## MCMC
## done 0 (out of 1100)
## done 100 (out of 1100)
## done 200 (out of 1100)
## done 300 (out of 1100)
## done 400 (out of 1100)
## done 500 (out of 1100)
## done 600 (out of 1100)
## done 700 (out of 1100)
## done 800 (out of 1100)
## done 900 (out of 1100)
## done 1000 (out of 1100)
## time: 2s
## trcnt,tecnt: 1000,1000
```

```r
yhat_bart <- bartfit$yhat.test.mean
rmse_bart <- sqrt(mean((y_test - yhat_bart)^2))
rmse_bart
```

```
## [1] 0.03211992
```

Model Comparison

```r
models <- c("Regression Tree", "Bagging", "Random Forest", "Boosting", "BART")
rmse_values <- c(rmse_tree, rmse_bag, rmse_rf, rmse_boost, rmse_bart)
rmse_comparison <- data.frame(Model = models, RMSE = rmse_values)
barplot(rmse_comparison$RMSE, names.arg = rmse_comparison$Model, col = "skyblue", main = "Out-of-Sample
```



Exercise 3: Testing for Changing Data Generating Process

```
d2 <- read.csv("growthdata02_11.csv")
d2 <- d2[, 3:ncol(d2)]
train <- sample(1:nrow(d2), nrow(d2) * .8)
d2.test <- d2[-train, "growth"]
```

Regression Tree

```
tree.d <- tree(growth ~ ., d2, subset = train)
cv.d <- cv.tree(tree.d)
size <- cv.d$size[which.min(cv.d$dev)]
prune.d2 <- prune.tree(tree.d, best = size)
yhat <- predict(prune.d2, newdata = d2[-train, ])
rmse_tree2 <- sqrt(mean((yhat - d2.test)^2))
rmse_tree2
```

```
## [1] 0.02752428
```

Bagging and Random Forests

```
bag.d <- randomForest(growth ~ ., data = d2, mtry = length(mtry_values), subset = train, importance = TR
yhat.bag <- predict(bag.d, newdata = d2[-train, ])
rmse_bag2 <- sqrt(mean((yhat.bag - d2.test)^2))
rmse_bag2
```

```
## [1] 0.02003076
```

```
oob_error_values3 <- numeric(length(mtry_values))
for (i in 1:length(mtry_values)) {
  rf.d <- randomForest(growth ~ ., data = d2, mtry = mtry_values[i], subset = train, importance = TRUE)
  oob_error_values3[i] <- rf.d$mse[rf.d$ntree]
}
best_mtry3 <- mtry_values[which.min(oob_error_values3)]
rf.d2 <- randomForest(growth ~ ., data = d2, mtry = best_mtry3, subset = train, importance = TRUE)
yhat.rf <- predict(rf.d2, newdata = d2[-train, ])
rmse_rf2 <- sqrt(mean((yhat.rf - d2.test)^2))
rmse_rf2
```

```
## [1] 0.01957409
```

Boosting

```
boost.d2 <- gbm(growth ~ ., data = d2[train, ], distribution = "laplace", n.trees = 10000, interaction.d
yhat.boost <- predict(boost.d2, newdata = d2[-train, ], n.trees = 10000)
rmse_boost2 <- sqrt(mean((yhat.boost - d2.test)^2))
rmse_boost2
```

```
## [1] 0.02164913
```

BART

```
x <- d2[, -2]
y <- d2[, "growth"]
xtrain <- x[train, ]
ytrain <- y[train]
xtest <- x[-train, ]
ytest <- y[-train]
bartfit2 <- gbart(xtrain, ytrain, x.test = xtest)
```

```
## *****Calling gbart: type=1
## *****Data:
## data:n,p,np: 89, 27, 23
## y1,yn: -0.031192, 0.034746
## x1,x[n*p]: 1.338654, -0.086136
## xp1,xp[np*p]: -0.534929, 0.224655
## *****Number of Trees: 200
## *****Number of Cut Points: 88 ... 88
## *****burn,nd,thin: 100,1000,1
## *****Prior:beta,alpha,tau,nu,lambda,offset: 2,0.95,0.00228507,3,6.61617e-05,0.03111
## *****sigma: 0.018430
## *****w (weights): 1.000000 ... 1.000000
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,27,0
## *****printevery: 100
```

```
##
## MCMC
## done 0 (out of 1100)
## done 100 (out of 1100)
## done 200 (out of 1100)
## done 300 (out of 1100)
## done 400 (out of 1100)
## done 500 (out of 1100)
## done 600 (out of 1100)
## done 700 (out of 1100)
## done 800 (out of 1100)
## done 900 (out of 1100)
## done 1000 (out of 1100)
## time: 2s
## trcnt,tecnt: 1000,1000
```

```r
ord2 <- order(bartfit2$varcount.mean, decreasing = T)
yhat.bart <- bartfit$yhat.test.mean
rmse_bart2 <- sqrt(mean((ytest - yhat.bart)^2))
```

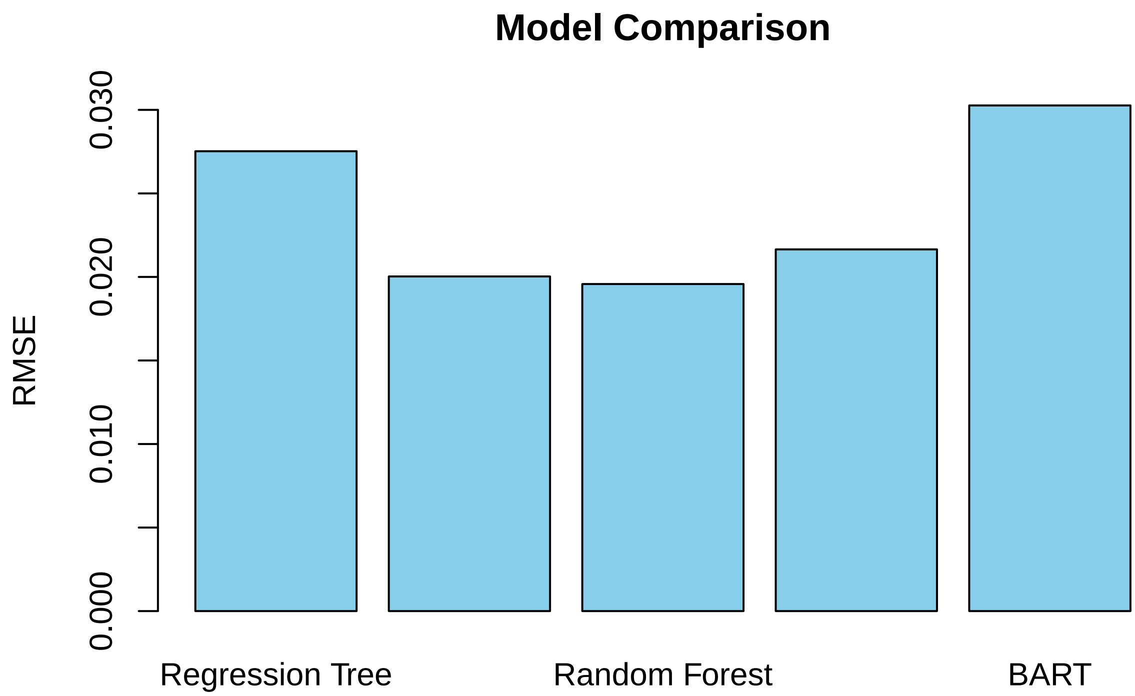
```
## Warning in ytest - yhat.bart: longer object length is not a multiple of shorter
## object length
```

```r
rmse_bart2
```

```
## [1] 0.03026457
```

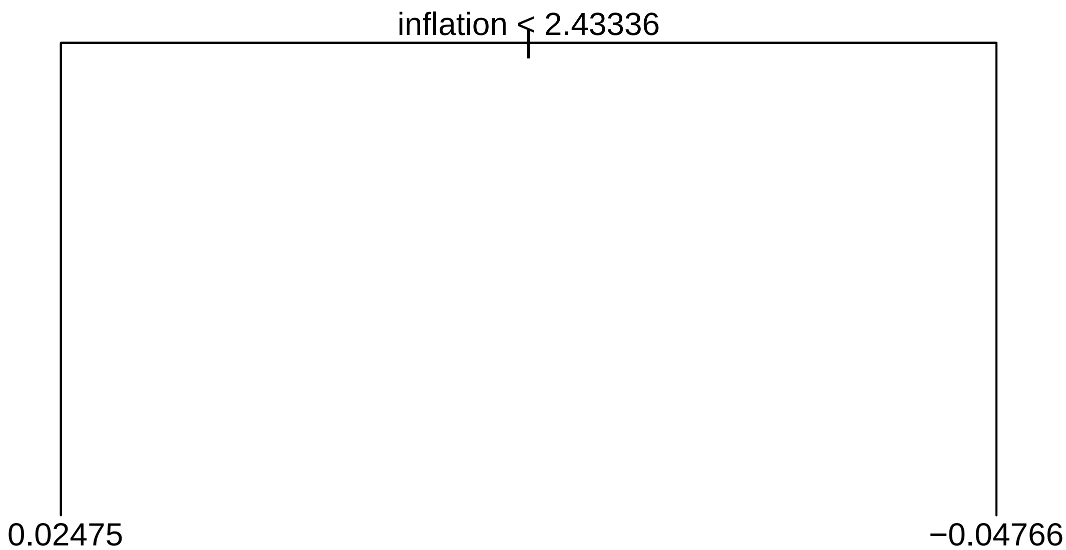
Model Comparison

```r
models <- c("Regression Tree", "Bagging", "Random Forest", "Boosting", "BART")
rmse_values2 <- c(rmse_tree2, rmse_bag2, rmse_rf2, rmse_boost2, rmse_bart2)
rmse_comparison2 <- data.frame(Model = models, RMSE = rmse_values2)
barplot(rmse_comparison2$RMSE, names.arg = rmse_comparison2$Model, col = "skyblue", main = "Model Compa
```

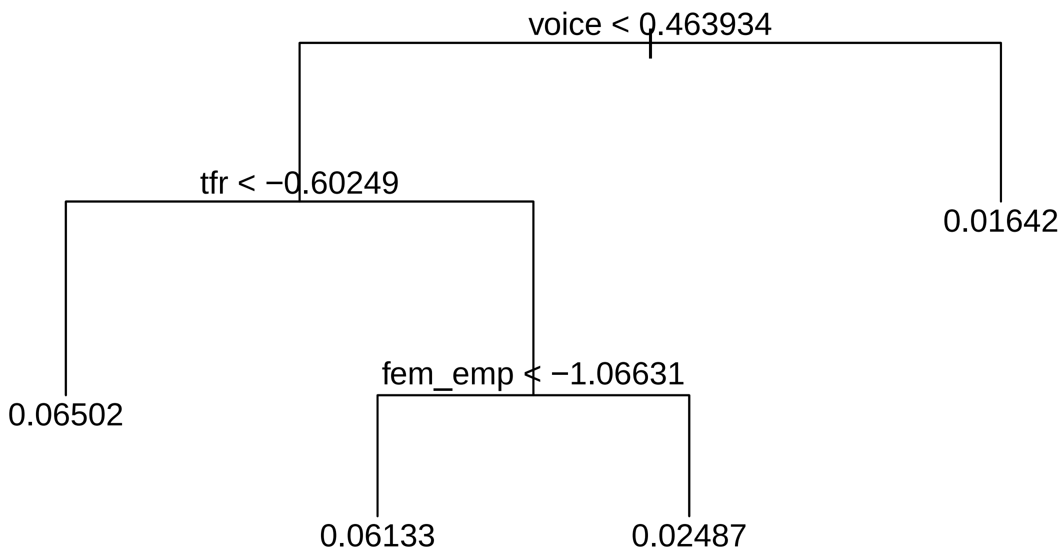


Parameter Stability

```
plot(prune.d)
text(prune.d, pretty = 0)
```




```r
plot(prune.d2)
text(prune.d2, pretty = 0)
```

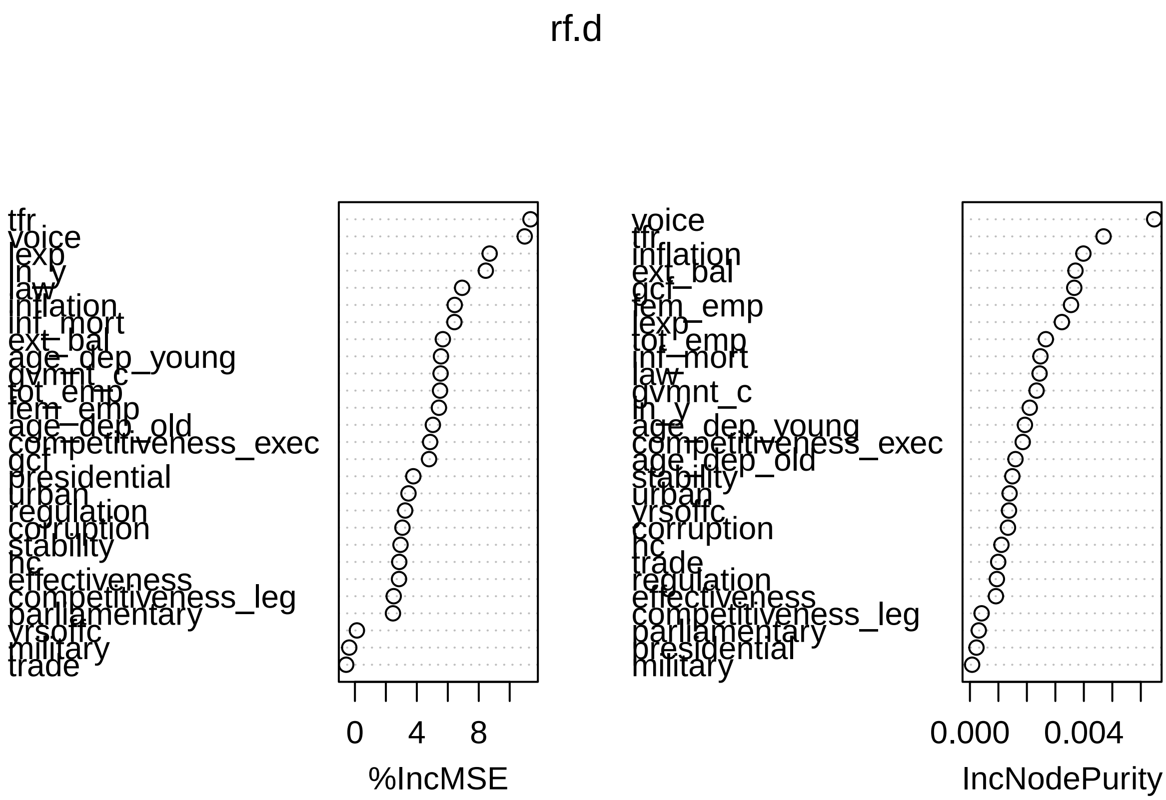



```r
importance(rf.d)
```

```
##                           %IncMSE IncNodePurity
## ln_y                    8.4558393  2.100621e-03
## hc                      2.8644284  1.106251e-03
## gvmnt_c                 5.5354374  2.339867e-03
## gcf                     4.7894480  3.663064e-03
## ext_bal                 5.6811095  3.705258e-03
## trade                  -0.5652375  9.930267e-04
## inflation               6.4484533  3.984379e-03
## fem_emp                 5.4269494  3.550165e-03
## tot_emp                 5.4991678  2.665494e-03
## inf_mort                6.4261347  2.475775e-03
## lexp                    8.7046246  3.228741e-03
## tfr                    11.3443015  4.692842e-03
## age_dep_old             5.0363724  1.598848e-03
## age_dep_young           5.5596024  1.930792e-03
## urban                   3.4608484  1.392704e-03
## yrsoffc                 0.1265864  1.372252e-03
## military               -0.3672169  7.664977e-05
## competitiveness_leg     2.5045331  4.096505e-04
## competitiveness_exec    4.8578881  1.854169e-03
## parliamentary           2.4575030  3.104519e-04
## presidential            3.7720142  2.294127e-04
## voice                  10.9692376  6.471398e-03
## stability               2.9446651  1.489212e-03
```

```
## effectiveness          2.8492161  9.121501e-04
## regulation             3.2446510  9.460018e-04
## law                    6.9268499  2.446016e-03
## corruption             3.0694568  1.335793e-03
```

```
varImpPlot(rf.d)
```

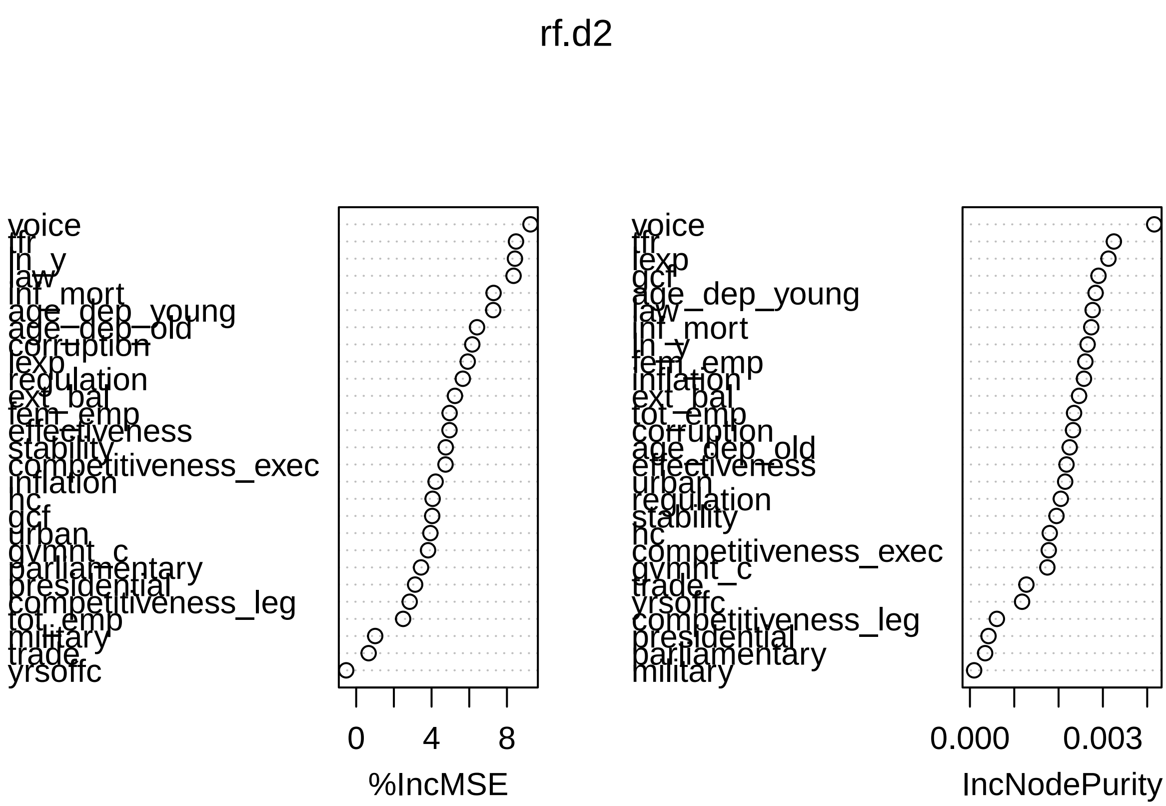



```
importance(rf.d2)
```

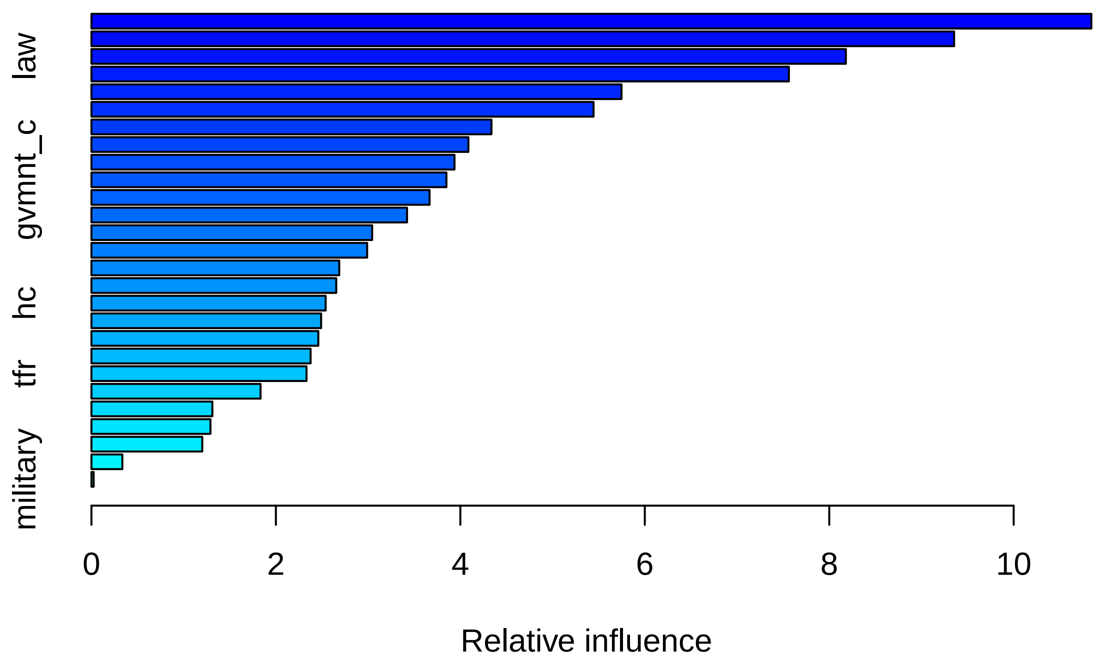
```
##                         %IncMSE IncNodePurity
## ln_y                  8.4246448  2.655050e-03
## hc                    4.0563259  1.801109e-03
## gvmnt_c               3.8132574  1.747286e-03
## gcf                   4.0338185  2.900605e-03
## ext_bal               5.2407855  2.462926e-03
## trade                 0.6601547  1.273943e-03
## inflation             4.2138012  2.572808e-03
## fem_emp               4.9570016  2.604177e-03
## tot_emp               2.4910958  2.349638e-03
## inf_mort              7.2903010  2.737261e-03
## lexp                  5.9186671  3.125207e-03
## tfr                   8.4812057  3.247342e-03
## age_dep_old           6.4135764  2.251270e-03
## age_dep_young         7.2707481  2.835706e-03
## urban                 3.9372214  2.148623e-03
## yrsoffc              -0.5310705  1.176748e-03
## military              1.0081637  9.519483e-05
```

```
## competitiveness_leg   2.8378682  6.080995e-04
## competitiveness_exec  4.7401901  1.778914e-03
## parliamentary         3.4408977  3.428260e-04
## presidential          3.1277734  4.189600e-04
## voice                 9.2501020  4.158268e-03
## stability             4.7539832  1.952549e-03
## effectiveness         4.9518257  2.179002e-03
## regulation            5.6582880  2.050795e-03
## law                   8.3511731  2.768870e-03
## corruption            6.1601159  2.326354e-03
```

```r
varImpPlot(rf.d2)
```



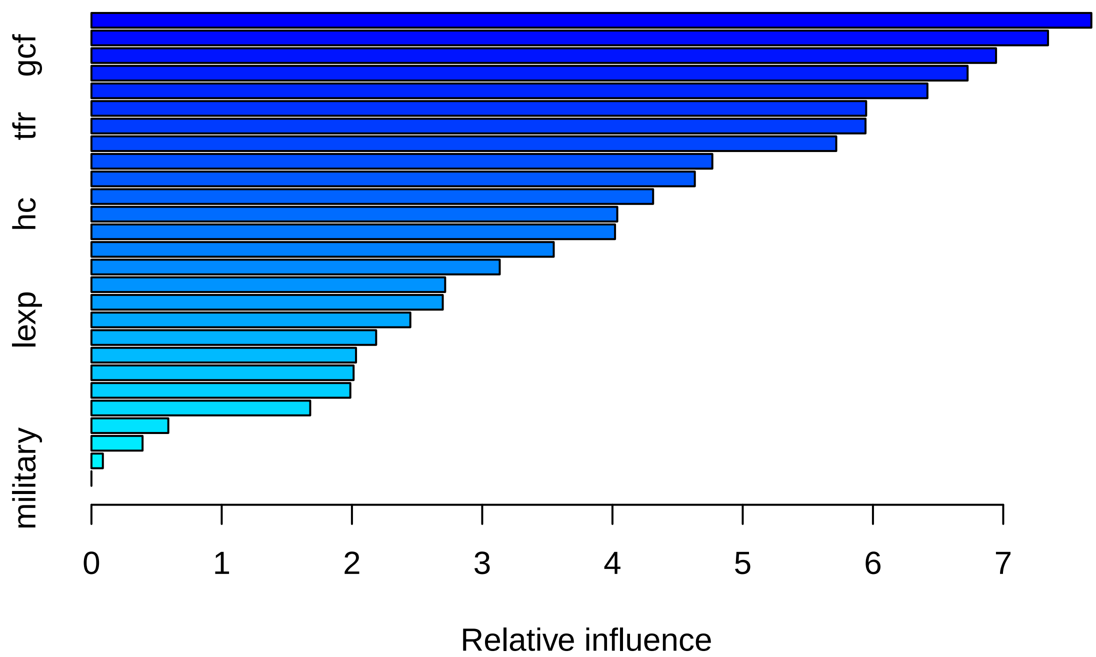

```r
summary(boost.d)
```




```
##                                         var     rel.inf
## inflation                         inflation 10.84216247
## fem_emp                             fem_emp  9.35388494
## law                                     law  8.18021558
## ext_bal                             ext_bal  7.56223215
## corruption                       corruption  5.74747161
## voice                                 voice  5.44455244
## presidential                   presidential  4.33740142
## ln_y                                   ln_y  4.08751579
## regulation                       regulation  3.93667404
## gvmnt_c                             gvmnt_c  3.84909063
## effectiveness                 effectiveness  3.66644592
## tot_emp                             tot_emp  3.42199897
## stability                         stability  3.04396396
## gcf                                     gcf  2.98961085
## trade                                 trade  2.68700651
## age_dep_old                     age_dep_old  2.65423153
## hc                                       hc  2.53950971
## urban                                 urban  2.49046106
## yrsoffc                             yrsoffc  2.46004945
## lexp                                   lexp  2.37660852
## tfr                                     tfr  2.33218823
## inf_mort                           inf_mort  1.83376364
## competitiveness_exec competitiveness_exec  1.31132467
## competitiveness_leg   competitiveness_leg  1.29052724
## age_dep_young                 age_dep_young  1.20171128
```

```
## parliamentary                 parliamentary  0.33522029
## military                           military  0.02417709
```

```r
summary(boost.d2)
```




```
##                                            var    rel.inf
## tot_emp                                tot_emp 7.67666082
## ext_bal                                ext_bal 7.34401484
## gcf                                        gcf 6.94501954
## inflation                            inflation 6.72645849
## ln_y                                      ln_y 6.41816193
## voice                                    voice 5.94765249
## tfr                                        tfr 5.94287212
## urban                                    urban 5.71833054
## gvmnt_c                                gvmnt_c 4.76683438
## trade                                    trade 4.63244792
## yrsoffc                                yrsoffc 4.31211981
## hc                                          hc 4.03698167
## fem_emp                                fem_emp 4.01984903
## stability                            stability 3.54908756
## age_dep_old                        age_dep_old 3.13463290
## age_dep_young                    age_dep_young 2.71562642
## law                                        law 2.69724552
## lexp                                      lexp 2.44830549
## corruption                          corruption 2.18587597
## regulation                          regulation 2.03130745
## effectiveness                    effectiveness 2.01270230
```

```
## inf_mort                             inf_mort 1.98770583
## competitiveness_exec competitiveness_exec 1.67965949
## parliamentary                   parliamentary 0.59020040
## presidential                     presidential 0.39237103
## competitiveness_leg   competitiveness_leg 0.08787609
## military                             military 0.00000000
```

```
bartfit$varcount.mean[ord]
```

```
##         presidential        parliamentary              fem_emp
##               10.786                9.827                7.760
##        age_dep_young              ext_bal                 ln_y
##                8.567                3.382                9.249
##             gvmnt_c competitiveness_exec            inflation
##                6.602                8.253                7.453
##                   hc                urban             military
##                7.653                8.833                9.101
##        effectiveness            stability           regulation
##                7.828                7.757                8.549
##                trade                  law                  tfr
##                4.881                9.151                8.528
##             inf_mort                voice                  gcf
##                7.178                9.076                3.585
##          age_dep_old              tot_emp              yrsoffc
##                6.771                8.212                7.889
##           corruption  competitiveness_leg                 lexp
##                8.631                7.513                6.226
```

```
bartfit2$varcount.mean[ord2]
```

```
##              ext_bal                  gcf         presidential
##                9.691                9.170                9.004
##           regulation              fem_emp                voice
##                8.890                8.614                8.569
##                urban        effectiveness        parliamentary
##                8.484                8.413                8.354
##             military                 lexp              tot_emp
##                8.351                8.276                8.203
##             inf_mort                trade                  tfr
##                8.152                8.103                8.040
##        age_dep_young            inflation                   hc
##                8.034                7.857                7.851
##                  law          age_dep_old                 ln_y
##                7.824                7.785                7.764
##           corruption              gvmnt_c              yrsoffc
##                7.666                7.257                7.243
## competitiveness_exec            stability  competitiveness_leg
##                7.130                7.127                5.976
```