Tree Based Methods

******Consider the Hitters dataset from the ISLR package in R. It consists of 20 variables measured on 263 major league baseball players (after removing those with missing data). Salary is the response variable and the remaining 19 are predictors. All data will be taken as training data. For all the models below, use leave-one-out cross-validation (LOOCV) to compute the estimated test MSE.***

a) Tree was fitted to Hitters data. Test MSE using LOOCV is reported in Table 1.

- Variables actually used in tree construction: `CAtBat`, `CHits`, `AtBat`, `CRuns`, `Hits`, `Walks`, `CRBI`.
- Number of terminal nodes: 9
- Residual mean deviance: 0.1694 = 43.03 / 254.

$$
\begin{aligned}
R_1 &= \{X|CAtBat < 1452, CHits < 182, AtBat < 147\} \\
R_2 &= \{X|CAtBat < 1452, CHits < 182, AtBat \geq 147, CRuns < 58.5\} \\
R_3 &= \{X|CAtBat < 1452, CHits < 182, AtBat \geq 147, CRuns \geq 58.5\} \\
R_4 &= \{X|CAtBat < 1452, CHits \geq 182\} \\
R_5 &= \{X|CAtBat \geq 1452, Hits < 117.5, Walks < 43.5\} \\
R_6 &= \{X|CAtBat \geq 1452, Hits < 117.5, Walks \geq 43.5\} \\
R_7 &= \{X|CAtBat \geq 1452, Hits \geq 117.5, CRBI < 273\} \\
R_8 &= \{X|CAtBat \geq 1452, Hits \geq 117.5, CRBI \geq 273, Walks < 60.5\} \\
R_9 &= \{X|CAtBat \geq 1452, Hits \geq 117.5, CRBI \geq 273, Walks \geq 60.5\}
\end{aligned}
$$

```r
library(ISLR)
Hitters.n <- na.omit(Hitters)
Hitters.n$Salary<- log(Hitters.n$Salary)
```

```r
# Grow a tree using the training set
library(tree)
tree.Hitters <- tree(Salary ~ ., Hitters.n)
tree.Hitters
```

```
node), split, n, deviance, yval
      * denotes terminal node

 1) root 263 207.200 5.927
   2) CAtBat < 1452 103  36.220 5.093
     4) CHits < 182 56  18.360 4.771
       8) AtBat < 147 5   5.899 5.961 *
       9) AtBat > 147 51   4.691 4.655
        18) CRuns < 58.5 28   1.019 4.462 *
        19) CRuns > 58.5 23   1.357 4.890 *
     5) CHits > 182 47   5.165 5.476 *
   3) CAtBat > 1452 160  53.080 6.464
     6) Hits < 117.5 70  17.610 6.154
      12) Walks < 43.5 51  12.700 6.041 *
      13) Walks > 43.5 19   2.493 6.459 *
     7) Hits > 117.5 90  23.490 6.706
      14) CRBI < 273 20   4.418 6.208 *
      15) CRBI > 273 70  12.700 6.848
        30) Walks < 60.5 40   4.709 6.677 *
        31) Walks > 60.5 30   5.275 7.075 *
```

```
summary(tree.Hitters)
```

```
Regression tree:
tree(formula = Salary ~ ., data = Hitters.n)
Variables actually used in tree construction:
[1] "CAtBat" "CHits"  "AtBat"  "CRuns"  "Hits"   "Walks"  "CRBI"
Number of terminal nodes:  9
Residual mean deviance:  0.1694 = 43.03 / 254
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.7080 -0.2213  0.0353  0.0000  0.2303  1.7020
```

```
par(mar = c(3.8, 3.8,1,1))
plot(tree.Hitters)
text(tree.Hitters, pretty = 0, cex=0.7)
```

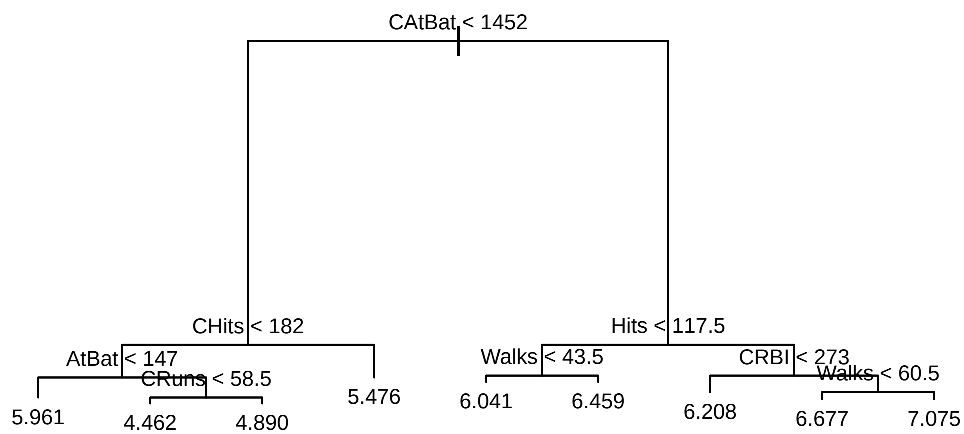


Figure 1: *Tree for Hitters Data*

```
# LOOCV for calculating MSE
loocv.tree<-function(i)
{
  test<-Hitters.n[i,]
  training<-Hitters.n[-i,]
  model<-tree(Salary ~ ., training)
  yhat <- predict(model, newdata = test)
  err<-(yhat - test[, "Salary"])^2
  return(err)
}
K<-nrow(Hitters.n)
RSS1a <- sapply(1:K, FUN = loocv.tree)
MSE1a <- mean(RSS1a)
```

b) After pruning tree we will get lowest dev value(69.13) for tree with size 9 and selected it as the best pruned tree. Therefore there is no difference between the best pruned and un_pruned tree. Test MSE using LOOCV is reported in Table 1. The most important predictors are `CAtBat`, `CHits`, `AtBat`, `CRuns`, `Hits`, `Walks`, `CRBI` as best tree only used these predictors when construction. Out of these variables, 'CAtbat' seems to be the most important as the first split is based on this variable. Test MSE using LOOCV is reported in Table 1.

```
# Pruning tree using LOOCV
set.seed(1)
cv.Hitters <- cv.tree(tree.Hitters, K=nrow(Hitters.n))
cv.Hitters
```

```
$size
[1] 9 8 7 6 5 4 3 2 1

$dev
[1]  69.13337  74.40305  73.31008  73.03341  82.88160  81.10150 107.16969
[8] 105.40736 236.50768

$k
[1]       -Inf   2.314754   2.423858   2.713047   6.377474   7.769090  11.970263
[8]  12.695982 117.857612

$method
[1] "deviance"

attr(,"class")
[1] "prune"         "tree.sequence"
```

```r
which.min(cv.Hitters$size)
```

```
[1] 9
```

c) Bagging approach was carried out to analyze Hitters data. For large B, OOB $\approx$ LOOCV. Therefore OOB error given in the `randomForest()` was taken as LOOCV test error rate. According to total decrease in node impurities `CAtBat` is the most important variable followed by `CRuns` and `CHits`. Test MSE using LOOCV is reported in Table 1.

```r
# part c)

# Bagging
library(randomForest)

set.seed(1)
bag.Hitters <- randomForest(Salary ~ ., data = Hitters.n, mtry = 19, ntree = 1000, importance = TRUE)
bag.Hitters
```

```

Call:
 randomForest(formula = Salary ~ ., data = Hitters.n, mtry = 19,      ntree = 1000, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 1000
No. of variables tried at each split: 19

          Mean of squared residuals: 0.1878328
                    % Var explained: 76.15
```

```r
# Estimate test error rate
yhat.bag <- predict(bag.Hitters)
mean((yhat.bag - Hitters.n$Salary)^2)
```

```
[1] 0.1878328
```

```r
importance(bag.Hitters)
```

```
              %IncMSE IncNodePurity
AtBat     16.39414728     8.7059839
Hits      13.26397793     7.9582602
HmRun      5.45135948     2.0137460
Runs      11.41975969     3.7690205
RBI        8.12540880     5.4747549
Walks     13.12242529     7.1851765
Years     12.84310430     2.2975585
CAtBat    38.52093312    78.9240659
```

```
CHits     18.87884188    27.0610915
CHmRun     9.36908046     3.7118579
CRuns     20.10035380    33.1447878
CRBI      22.19861429    10.8588577
CWalks     9.24963357     4.9603432
League    -2.75627343     0.2117655
Division  -0.83250913     0.2643655
PutOuts    3.54841386     3.9008481
Assists    0.06143034     1.5939945
Errors     0.90940738     1.6873440
NewLeague -0.22530028     0.4041698
```

```r
par(mar = c(3.8, 3.8,4,1))
varImpPlot(bag.Hitters,main="",cex=0.7)
```

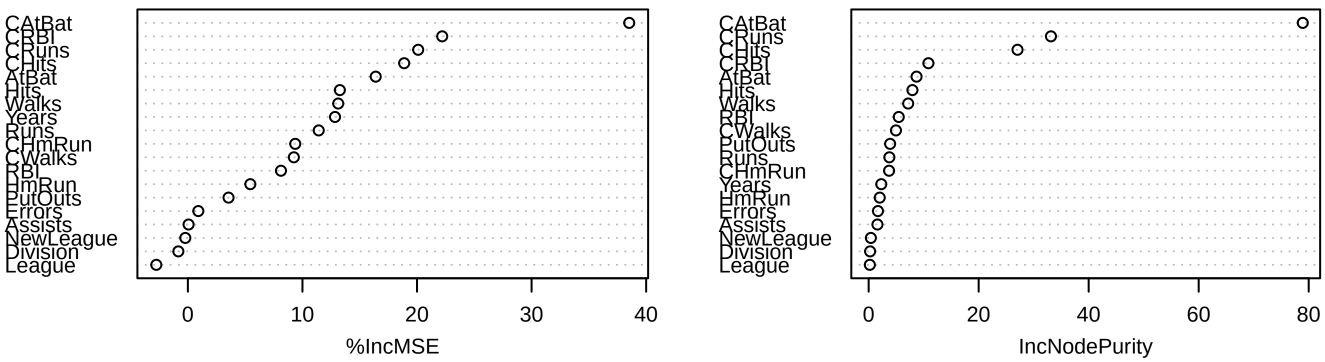


Figure 2: *VarImp plot for bagging approach*

d) Random Forest approach was carried out to analyze Hitters data. For large B, OOB $\approx$ LOOCV. Therefore OOB error given in the `randomForest()` was taken as LOOCV test error rate. According to total decrease in node impurities `CAtBat` is the most important variable followed by `CHits`, `CRuns`, `CWalks` and `CRBI`. Test MSE using LOOCV is reported in Table 1.

```r
# part d)

# Random Forest
set.seed(1)
rf.Hitters <- randomForest(Salary ~ ., data = Hitters.n, mtry = 19/3, ntree = 1000, importance = TRUE)
rf.Hitters
```

```

Call:
 randomForest(formula = Salary ~ ., data = Hitters.n, mtry = 19/3,      ntree = 1000, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 1000
No. of variables tried at each split: 6

          Mean of squared residuals: 0.1802431
                    % Var explained: 77.12
```

```r
# Estimate test error rate
yhat.rf <- predict(rf.Hitters)
mean((yhat.rf - Hitters.n$Salary)^2)
```

```
[1] 0.1802431
```

```r
importance(rf.Hitters)
```

```
             %IncMSE IncNodePurity
```

```
AtBat     15.1500281     7.8591806
Hits      11.7947053     8.0957356
HmRun      6.4331611     2.5796590
Runs      10.6810291     4.7913514
RBI        8.4328343     6.4830584
Walks     12.7771088     5.8701933
Years     15.6597897     7.0560141
CAtBat    23.8343731    40.2152018
CHits     23.0914542    35.2244118
CHmRun    12.1281171     7.4727405
CRuns     21.1777716    32.2252499
CRBI      19.2569818    18.3389603
CWalks    15.8078619    19.5160848
League     0.4445704     0.2855193
Division   0.3628432     0.2823429
PutOuts    2.6649520     3.3595163
Assists    1.2528416     1.6740922
Errors     2.0927205     1.6752935
NewLeague -0.4131073     0.3637911
```

```r
par(mar = c(3.8, 3.8,4,1))
varImpPlot(rf.Hitters,main="",cex=0.7)
```

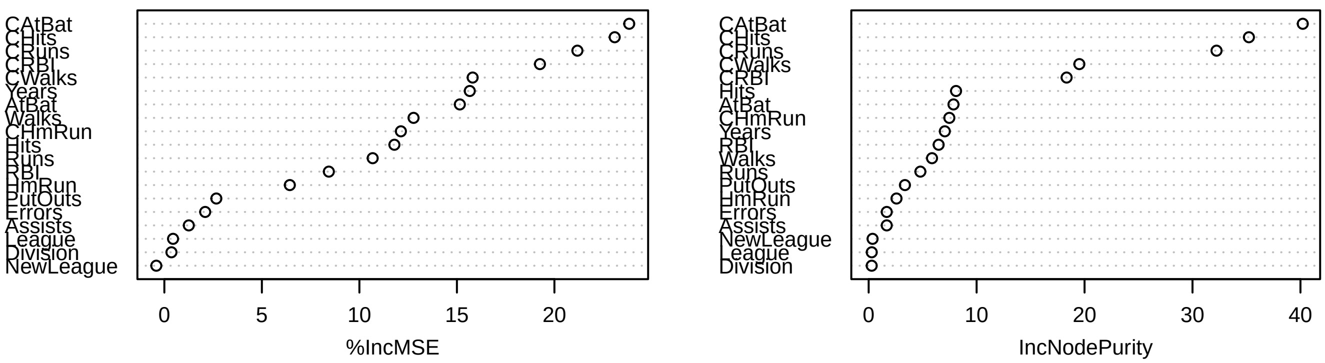


Figure 3: *VarImp plot for random forest approach*

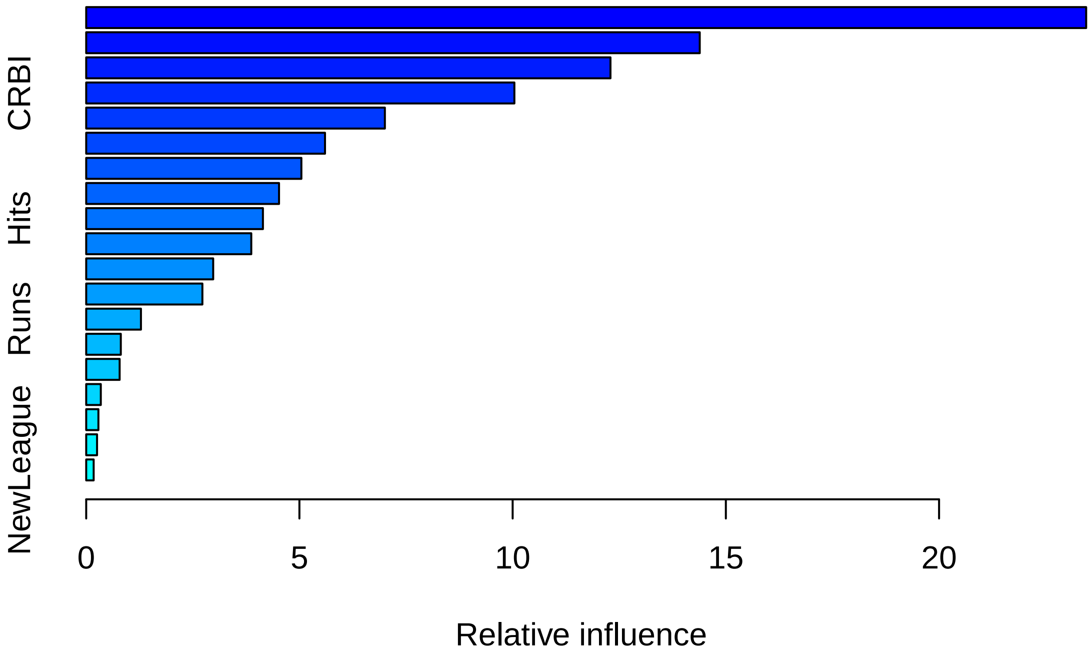
e) Boosting approach was carried out to analyze Hitters data. According to relative influence `CAtBat` is the most important variable followed by `CHits`, `CRuns`,`CRBI` and `CWalks`. Test MSE using LOOCV is reported in Table 1.

```r
library(gbm)
# Fit a boosted regression tree
set.seed(1)
boost.Hitters <- gbm(Salary ~ ., data = Hitters.n, distribution = "gaussian",
    n.trees = 1000, interaction.depth = 1, shrinkage = 0.01, cv.folds = nrow(Hitters.n) )

yhat.boost <- predict(boost.Hitters)
mean((yhat.boost -  Hitters.n$Salary)^2)
```

```
[1] 0.1541302
```

```r
summary(boost.Hitters)
```




```
                var    rel.inf
CAtBat       CAtBat 23.4492712
CHits         CHits 14.3875368
CRuns         CRuns 12.2942197
CRBI           CRBI 10.0409460
CWalks       CWalks  7.0038366
Years         Years  5.5998672
Walks         Walks  5.0460011
CHmRun       CHmRun  4.5218619
Hits           Hits  4.1445894
RBI             RBI  3.8710757
PutOuts     PutOuts  2.9777287
HmRun         HmRun  2.7248893
Runs           Runs  1.2835926
Errors       Errors  0.8118752
AtBat         AtBat  0.7842828
Division   Division  0.3424589
League       League  0.2863447
Assists     Assists  0.2543586
NewLeague NewLeague  0.1752636
```

f) According to test MSE Boosting method seems the best as it has the lowest test MSE. Then random forest and bagging seems best as they have the next lowest MSE respectively. However there is not much difference between these two. Therefore, I would recommend boosting as the best method. According to the previous project I recommended Ridge regression model as it had the lowest MSE of 0.3607. However boosting approach for decision trees seems better than Ridge regression as trees has the lower MSE (0.1541) than Ridge regression model. Therefore, I would recommend boosting as the best method.

	tree	pruned tree	bagging	random forest	boosting
Test MSE	0.2545	0.2545	0.1878	0.1802	0.1541

Table 1: *Summary of test MSE*