Non-parametric Statistics: Notes 10
Smoothing methods and robust model fitting.

Gregory Matthews [1]

[1] Department of Mathematics and Statistics
Loyola University Chicago

Fall 2014

Outline

Cross validation
- Internal validation
- External Validation

- Two of the big ideas that we have learned so far are bootstrapping and CART.
- Tell me what these are.
- Ok. So what happens if we combine them?
- Random Forests!

The algorithm

- Randomly draw a bootstrap sample from the original data.
- Build a tree BUT only consider a subset of candidate predictors at each potential split.
- Repeat this process B times.

Out-of-bag (OOB) error

- Since each tree is built on only a subset of the data, there is implicitly a hold out sample each time.
- For each tree, make a prediction for the data that was not sampled.
- Average these predictions across trees.

Prediction

- For regression: Our prediction is the average prediction across all trees.
- For classification: Our prediction is the most often predicted class (i.e. Majority rules)

Variable importance

http://stat.ethz.ch/education/semesters/ss2012/ams/slides/v10.2.pdf

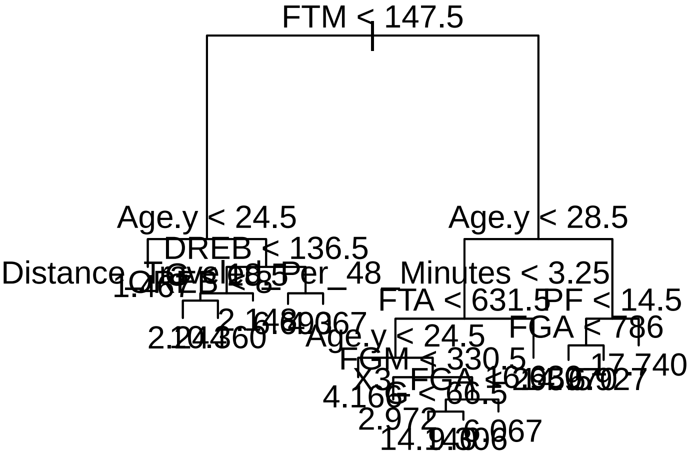
Prediction

http://stat.ethz.ch/education/semesters/ss2012/ams/slides/v10.2.pdf

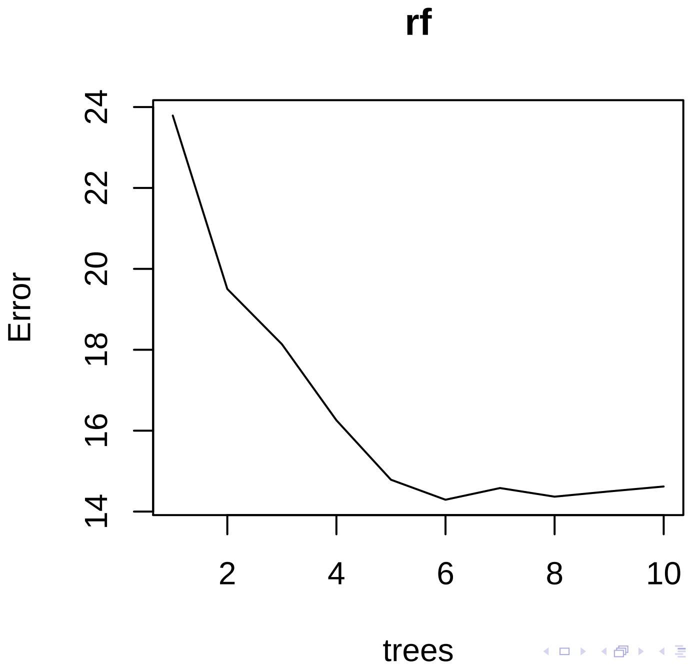
- Trees:
  - Pro: Yield Insight into decision rules
  - Pro: Rather Fast
  - Pro: Easy to tune parameters
  - Con: Tend to have high variance
- Random Forests:
  - Pro: Smaller prediction variance
  - Pro: Easy to tune parameters
  - Con: Relatively slower
  - Con: "Black Box"y.

```r
#Krishna Narsu (@knarsu3) gave me this data set
nba<-read.csv("/Users/gregorymatthews/Dropbox/LoyolaTeachi
#Make salary numeric
nba$Salary<-as.numeric(gsub("[$,]","",nba$Salary))

## Warning:  NAs introduced by coercion

#Remove obsrevations missing salary
nba<-nba[!is.na(nba$Salary),]
nba<-nba[!nba$Player=="Kobe Bryant",]
```

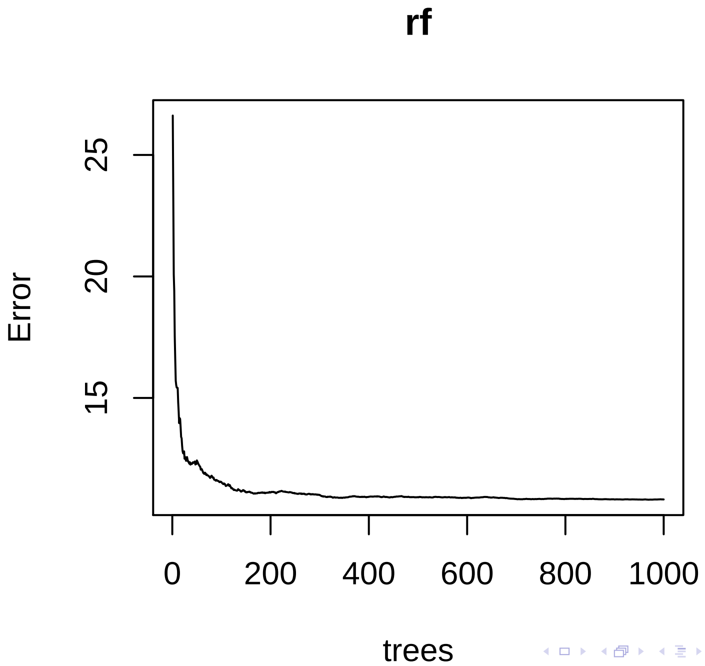
```
library(randomForest)

## Warning:  package 'randomForest' was built under R
version 3.1.1
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug
fixes.

nba<-nba[complete.cases(nba),]
nba$Salary<-round(nba$Salary/1000000,2)
    form<-formula(Salary~MINUTES + FGM +
    FGA + X3_FGM + X3_FGA  + FTM + FTA  + OREB +
    DREB + REB + AST + TOV + STL + BLK + PFoul +
    PTS + Plus.Minus + Age.y + G + MP +
    PG + SG + SF + PF + C + Distance_Traveled_Per_48_Minute
```


FTM < 147.5
Age.y < 24.5
Age.y < 28.5
DREB < 136.5
Minutes < 3.25
FTA < 631.5
PF < 14.5
FGA < 786
Age.y < 24.5
FGM < 330.5
17.740
4.166
2.972
6.067

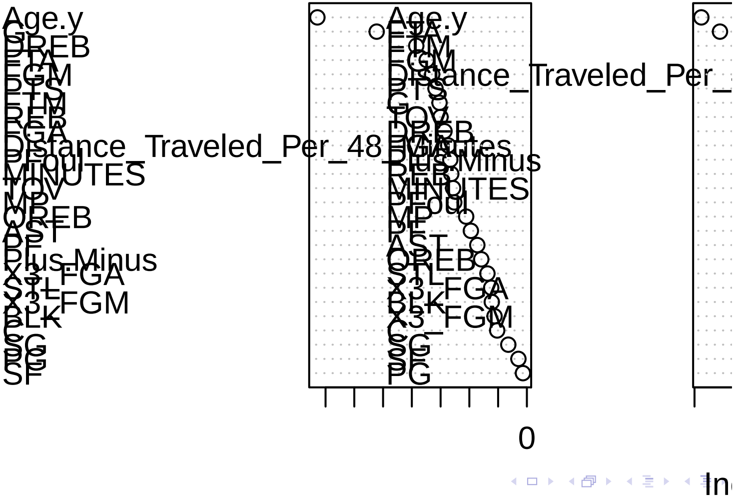
```
#predict and plot methods exist
(rf<-randomForest(form,data=nba,ntree=10))

##
## Call:
##  randomForest(formula = form, data = nba, ntree = 10)
##                Type of random forest: regression
##                      Number of trees: 10
## No. of variables tried at each split: 8
##
##           Mean of squared residuals: 14.62018
##                     % Var explained: 34.41
```


rf
Error
14
16
18
20
22
24
2
4
6
8
10
trees

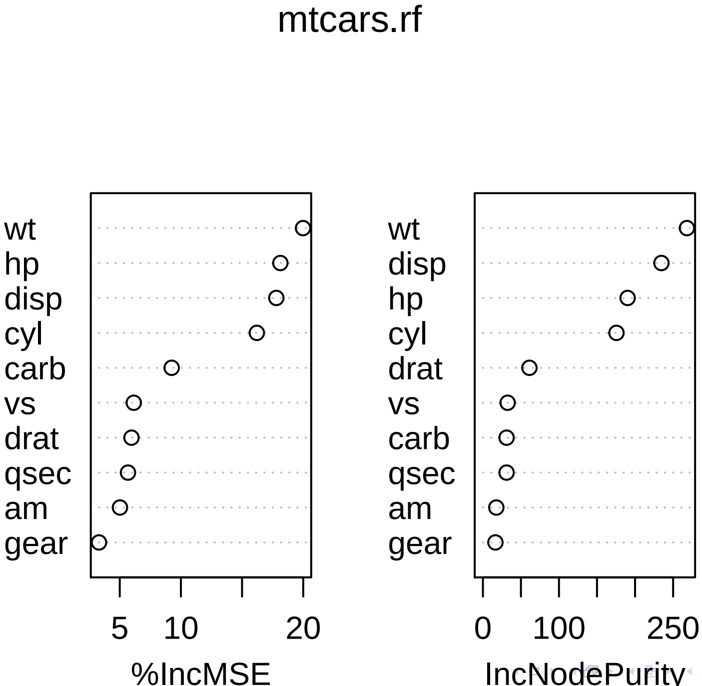
```r
#predict and plot methods exist
(rf<-randomForest(form,data=nba,ntree=1000,importance=TRUE)

##
## Call:
##  randomForest(formula = form, data = nba, ntree = 1000,
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 8
##
##           Mean of squared residuals: 10.82381
##                     % Var explained: 51.45
```


rf
25
20
15
Error
0
200
400
600
800
1000
trees

rf


Age.y
G
DREB
FTA
FGM
PTS
FTM
REB
FGA
Distance_Traveled_Per_48_Minutes
PFoul
MINUTES
TOV
MP
OREB
AST
PF
Plus.Minus
X3_FGA
STL
X3_FGM
BLK
C
SG
PG
SF
Age.y
FTA
FTM
FGM
Distance_Traveled_Per_
PTS
G
TOV
DREB
FGA
Plus.Minus
REB
MINUTES
PFoul
MP
PF
AST
OREB
STL
X3_FGA
BLK
X3_FGM
C
SG
SF
PG
0

```
set.seed(4543)
data(mtcars)
mtcars.rf <- randomForest(mpg ~ ., data=mtcars, ntree=1000,
  keep.forest=FALSE,importance=TRUE)
```


mtcars.rf
wt
hp
disp
cyl
carb
vs
drat
qsec
am
gear
5
10
20
%IncMSE
wt
disp
hp
cyl
drat
vs
carb
qsec
am
gear
0
100
250
IncNodePurity

- We want to answer the question: How will my model generalize to an independent set of data?
- To do this we can use cross validation.

- A common way to do this is to partition your data sets into two parts:
    - A training data set: Used to build the model.
    - A validation (or testing or holdout) data set: Used to validate the model.

```
library(tree)
(tree1_overfit<-tree(form,data=nba,control=tree.control(nob
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##      1) root 425 9.474e+03  4.29700
##        2) FTM < 147.5 314 2.853e+03  2.81400
##          4) Age.y < 24.5 122 2.031e+02  1.46700
##            8) PFoul < 90 62 3.340e+01  0.93840
##             16) Age.y < 20.5 6 9.784e+00  2.12000
##               32) C < 0.5 4 1.263e-01  1.23800
##                 64) BLK < 3 2 7.200e-03  1.41000 *
##                 65) BLK > 3 2 5.000e-05  1.06500 *
##               33) C > 0.5 2 3.121e-01  3.88500
##                 66) FGA < 84.5 1 0.000e+00  3.49000 *
##                 67) FGA > 84.5 1 0.000e+00  4.28000 *
##             17) Age.y > 20.5 56 1.434e+01  0.81180
##               34) G < 24.5 21 2.899e+00  0.56760
```

```
summary(tree1_overfit)

##
## Regression tree:
## tree(formula = form, data = nba, control = tree.control(
##     mincut = 1, minsize = 2, mindev = 1e-06))
## Variables actually used in tree construction:
##  [1] "FTM"                                "Age.y"
##  [3] "PFoul"                              "C"
##  [5] "BLK"                                "FGA"
##  [7] "G"                                  "PF"
##  [9] "X3_FGM"                             "Plus.Minus"
## [11] "SG"                                 "MINUTES"
## [13] "REB"                                "FGM"
## [15] "X3_FGA"                             "SF"
## [17] "OREB"                               "AST"
## [19] "STL"                                "DREB"
## [21] "TOV"                                "Distance_Travel
## [23] "PTS"                                "FTA"
```

```r
sum((predict(tree1_overfit)-nba$Salary)^2)

## [1] 0.4297545
```

```r
#Split the data into two parts
set.seed(1234)
ind<-sample(1:dim(nba)[1],383,replace=FALSE)
#383 observations
nbaTrain<-nba[ind,]
#42 observations
nbaPred<-nba[-ind,]
```

```r
#Notice I am using a different data set now!
(tree1_overfit<-tree(form,data=nbaTrain,control=
tree.control(nobs=383,mincut=1,minsize=2,
mindev=0.000001)))
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##         1) root 383 8.646e+03  4.3650
##           2) FTM < 147.5 280 2.546e+03  2.8440
##             4) Age.y < 25.5 137 6.692e+02  1.7680
##               8) Distance_Traveled_Per_48_Minutes < 3.05
##               9) Distance_Traveled_Per_48_Minutes > 3.05
##                18) PG < 92 125 1.720e+02  1.5250
##                  36) OREB < 104 110 1.324e+02  1.3610
##                    72) Age.y < 20.5 10 1.837e+01  2.5840
##                     144) C < 0.5 7 3.119e+00  1.8210
##                       288) X3_FGA < 169.5 5 5.416e-01  1.
##                         576) PG < 2 3 7.280e-02  1.6900
```

```
summary(tree1_overfit)

##
## Regression tree:
## tree(formula = form, data = nbaTrain, control = tree.con
##     mincut = 1, minsize = 2, mindev = 1e-06))
## Variables actually used in tree construction:
##  [1] "FTM"                               "Age.y"
##  [3] "Distance_Traveled_Per_48_Minutes" "PG"
##  [5] "OREB"                              "C"
##  [7] "X3_FGA"                            "MINUTES"
##  [9] "FGA"                               "FTA"
## [11] "AST"                               "PF"
## [13] "Plus.Minus"                        "X3_FGM"
## [15] "TOV"                               "G"
## [17] "BLK"                               "SG"
## [19] "STL"                               "DREB"
## [21] "FGM"                               "PFoul"
## [23] "SF"                                "REB"
```

```r
sum((predict(tree1_overfit)-nbaTrain$Salary)^2)

## [1] 0.2839522
```

```
sum((predict(tree1_overfit,nbaPred)-nbaPred$Salary)^2)

## [1] 521.9888
```

```r
#Notice I am using a different data set now!
#Now don't over fit
(tree1<-tree(form,data=nbaTrain,control=tree.control
(nobs=383,mincut=2,minsize=4,mindev=0.01)))
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 383 8646.000  4.365
##    2) FTM < 147.5 280 2546.000  2.844
##      4) Age.y < 25.5 137  669.200  1.768 *
##      5) Age.y > 25.5 143 1566.000  3.875
##       10) DREB < 136.5 74  380.900  2.295 *
##       11) DREB > 136.5 69  802.100  5.571 *
##    3) FTM > 147.5 103 3691.000  8.500
##      6) Distance_Traveled_Per_48_Minutes < 3.15 30  984.
##       12) Age.y < 27.5 12  308.300  9.297
##         24) Plus.Minus < 315 7   35.130  5.474 *
##         25) Plus.Minus > 315 5   27.750 14.650 *
```

```
summary(tree1)

##
## Regression tree:
## tree(formula = form, data = nbaTrain, control = tree.cor
##     mincut = 2, minsize = 4, mindev = 0.01))
## Variables actually used in tree construction:
## [1] "FTM"                                  "Age.y"
## [3] "DREB"                                 "Distance_Travele
## [5] "Plus.Minus"                           "X3_FGM"
## [7] "G"                                    "STL"
## [9] "FGM"
## Number of terminal nodes:  12
## Residual mean deviance:  7.39 = 2742 / 371
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -5.3310 -1.4150 -0.6384  0.0000  1.0620 15.8600
```

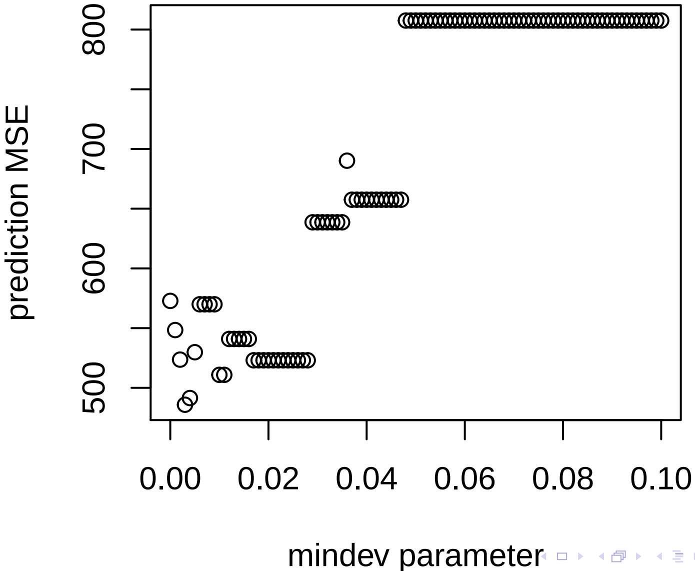
```r
sum((predict(tree1)-nbaTrain$Salary)^2)

## [1] 2741.666
```

```r
#External comparison
sum((predict(tree1,nbaTrain)-nbaTrain$Salary)^2)

## [1] 2741.666

sum((predict(tree1_overfit,nbaTrain)-nbaTrain$Salary)^2)

## [1] 0.2839522
```

```r
#External comparison
sum((predict(tree1,nbaPred)-nbaPred$Salary)^2)

## [1] 510.8855

sum((predict(tree1_overfit,nbaPred)-nbaPred$Salary)^2)

## [1] 521.9888
```

```
treeMSEtest<-function(mindev){
(tree1<-tree(form,data=nbaTrain,control=tree.control
(nobs=383,mincut=4,minsize=8,mindev=mindev)))
out<-sum((predict(tree1,nbaPred)-nbaPred$Salary)^2)
out
}
```

```r
treeMSEtest<-Vectorize(treeMSEtest)
plot(c(100:0)/1000,treeMSEtest(c(100:0)/1000))
```


prediction MSE
500
600
700
800
0.00
0.02
0.04
0.06
0.08
0.10
mindev parameter