Classificaiton Tree

```r
library(MASS)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```r
library(tidymodels)
```

```
## -- Attaching packages --------------------------------- tidymodels 0.0.2 --
```

```
## v broom     0.5.1     v recipes   0.1.4
## v dials     0.0.2     v rsample   0.0.4
## v infer     0.4.0     v yardstick 0.0.3
## v parsnip   0.0.1
```

```
## -- Conflicts ------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x dplyr::select()   masks MASS::select()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```r
library(tree)
library(kernlab)  # for the data spam
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:scales':
##
##     alpha
```

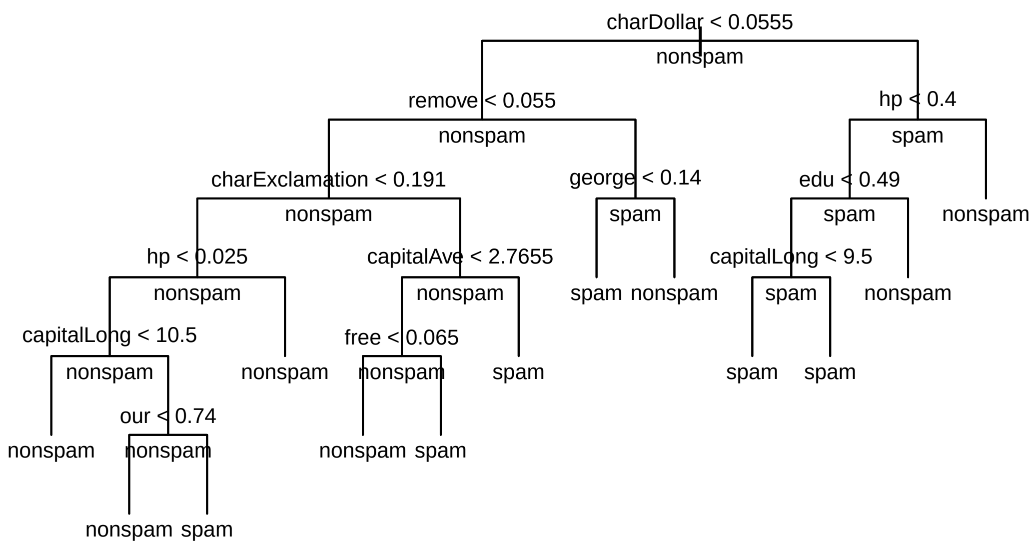
```
## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```r
data(spam)
tree_fit <- tree(type~., spam)
summary(tree_fit)
```

```
##
## Classification tree:
## tree(formula = type ~ ., data = spam)
## Variables actually used in tree construction:
##  [1] "charDollar"      "remove"          "charExclamation"
##  [4] "hp"              "capitalLong"     "our"
##  [7] "capitalAve"      "free"            "george"
## [10] "edu"
## Number of terminal nodes:  13
## Residual mean deviance:  0.4879 = 2238 / 4588
## Misclassification error rate: 0.08259 = 380 / 4601
```

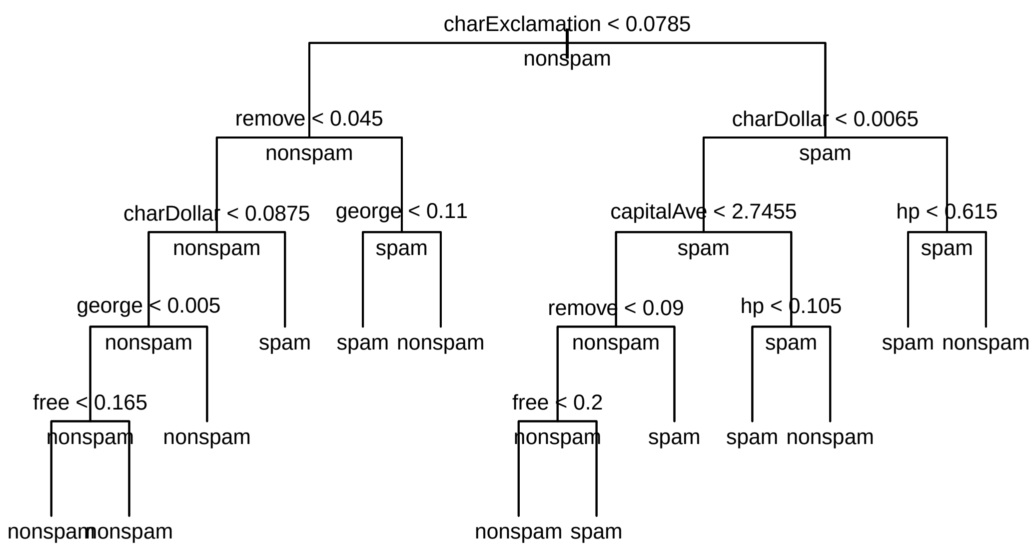
```r
plot(tree_fit, type = "uniform")
text(tree_fit, pretty = 1, all = TRUE, cex = 0.7)
```



```r
set.seed(2)
rs <- initial_split(spam, prop = 0.7)
spam_trainset <- as_tibble(training(rs))
spam_testset <- as_tibble(testing(rs))
train_fit <- tree(type ~ ., spam_trainset)
summary(train_fit)
```

```
##
## Classification tree:
## tree(formula = type ~ ., data = spam_trainset)
## Variables actually used in tree construction:
## [1] "charExclamation" "remove"          "charDollar"      "george"
## [5] "free"            "capitalAve"      "hp"
## Number of terminal nodes:  13
## Residual mean deviance:  0.5278 = 1693 / 3208
## Misclassification error rate: 0.09593 = 309 / 3221
```

```r
plot(train_fit, type = "uniform")
text(train_fit, pretty = 1, all = TRUE, cex = 0.7)
```

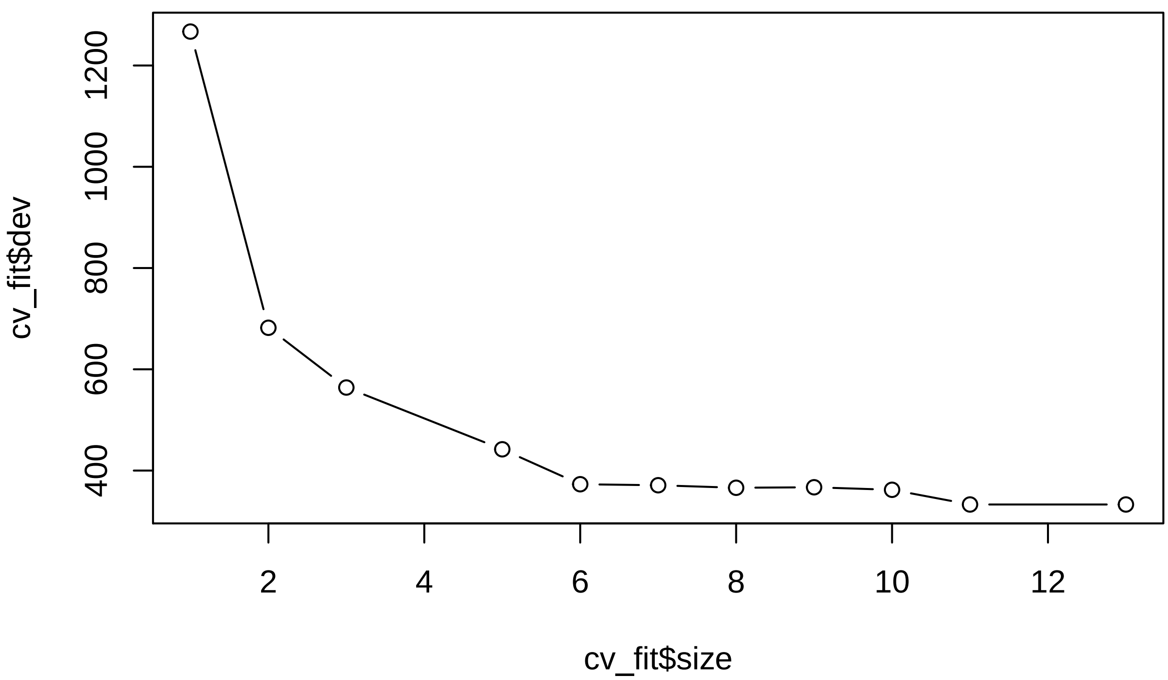



```r
spam_testset %>% modelr::add_predictions(train_fit, type = "class") %>%
    accuracy(type, pred)
```

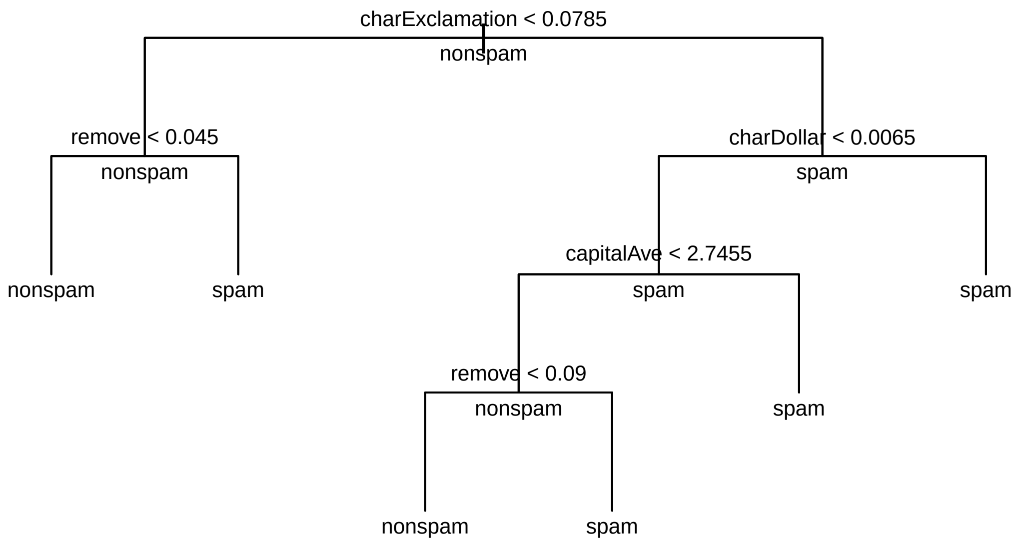
```
## # A tibble: 1 x 3
```

```
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.905
```

```r
set.seed(2)
cv_fit <- cv.tree(train_fit, method = "misclass")
plot(cv_fit$size, cv_fit$dev, type = "b")
```




```r
prune_fit <- prune.misclass(train_fit, best = 6)
plot(prune_fit, type = "uniform")
text(prune_fit, all = TRUE, cex = 0.7)
```




```
spam_testset %>% modelr::add_predictions(prune_fit, type = "class") %>%
    accuracy(type, pred)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.887
```

Regression Tree

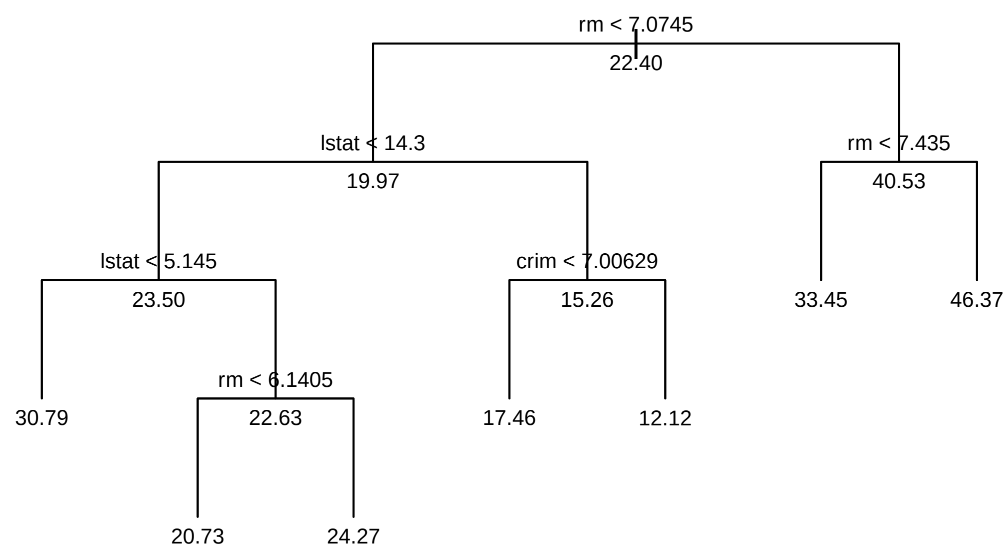
```
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
##   lstat medv
## 1  4.98 24.0
## 2  9.14 21.6
```

```
## 3  4.03 34.7
## 4  2.94 33.4
## 5  5.33 36.2
## 6  5.21 28.7
```

```r
set.seed(2)
rs <- initial_split(Boston, prop = 0.7)
boston_trainset <- as_tibble(training(rs))
boston_testset <- as_tibble(testing(rs))
```

```r
tree_boston <- tree(medv~., boston_trainset)
plot(tree_boston, type = "uniform")
text(tree_boston, all = TRUE, cex = 0.7)
```



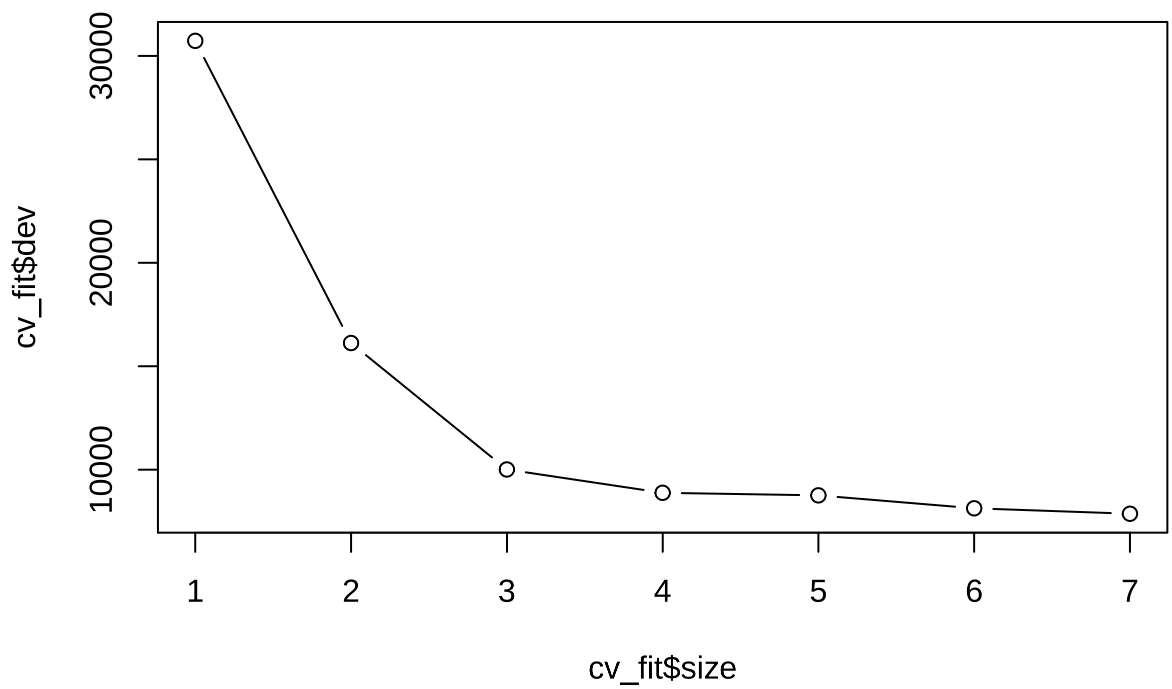

```r
modelr::mse(tree_boston, boston_testset)
```

```
## [1] 28.38208
```

```r
lm_fit <- lm(medv~., boston_trainset)
modelr::mse(lm_fit, boston_testset)
```

```
## [1] 26.59445
```

```r
set.seed(2)
cv_fit <- cv.tree(tree_boston)
plot(cv_fit$size, cv_fit$dev, type = "b")
```




```r
prune_boston <- prune.tree(tree_boston, best = 6)
modelr::mse(prune_boston, boston_testset)
```

```
## [1] 29.03037
```

Mars

```r
library(earth)
```

```
## Loading required package: plotmo
```

```
## Loading required package: Formula
```

```
## Loading required package: plotrix
```

```
##
## Attaching package: 'plotrix'
```

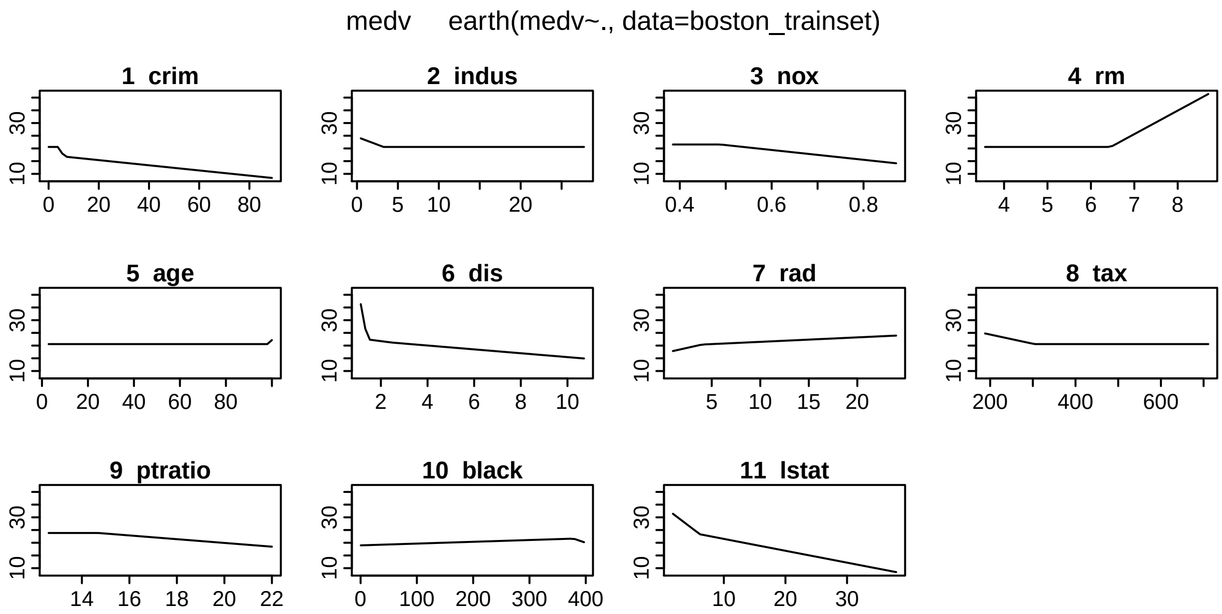
```
## The following object is masked from 'package:scales':
##
##     rescale

## Loading required package: TeachingDemos
```

```r
earth_boston <- earth(medv~., boston_trainset)
modelr::mse(earth_boston, boston_testset)
```

```
## [1] 17.09923
```

```r
plotmo(earth_boston)
```

```
##  plotmo grid:    crim zn indus chas   nox    rm  age    dis rad tax
##                0.29916  0  8.56    0 0.538 6.172 79.2 3.2628   5 330
##  ptratio  black lstat
##     19.1 391.93 11.98
```



Random Forest

```r
library(randomForest)
```

```
## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
# use all predictor
set.seed(2)
rf_boston <- randomForest(medv~., boston_trainset, mtry = 13)
rf_boston
```

```
##
## Call:
##  randomForest(formula = medv ~ ., data = boston_trainset, mtry = 13)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 13
##
##           Mean of squared residuals: 11.01538
##                     % Var explained: 87.19
```

```r
modelr::mse(rf_boston, boston_testset)
```

```
## [1] 15.95017
```

```r
set.seed(2)
rf_boston2 <- randomForest(medv~., boston_trainset, mtry = sqrt(13), importance = TRUE)
```

```r
modelr::mse(rf_boston2, boston_testset)
```

```
## [1] 13.42329
```

```r
importance(rf_boston2)
```

```
##           %IncMSE IncNodePurity
## crim    13.853061    1619.27348
## zn       4.997058     251.79563
```

```
## indus    10.745623    1657.79227
## chas      2.563871      30.92239
## nox      19.155831    1597.83667
## rm       34.153964    9626.22680
## age      12.928410     743.57990
## dis      16.147502    1519.31858
## rad       7.106276     288.56183
## tax      13.464302    1079.32248
## ptratio 16.180430    2502.94520
## black     9.313268     523.41593
## lstat    27.543197    8686.23240
```

```
varImpPlot(rf_boston2)
```

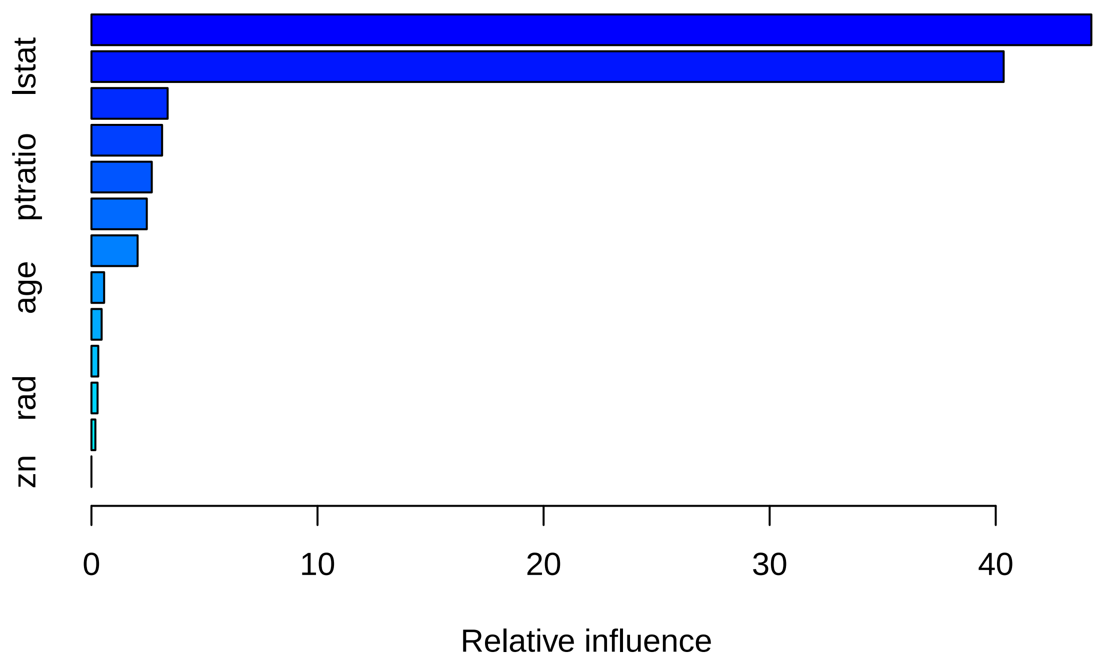
Gradient Boosting

```
set.seed(1)
library(gbm)
```

```
## Loaded gbm 2.1.5
```

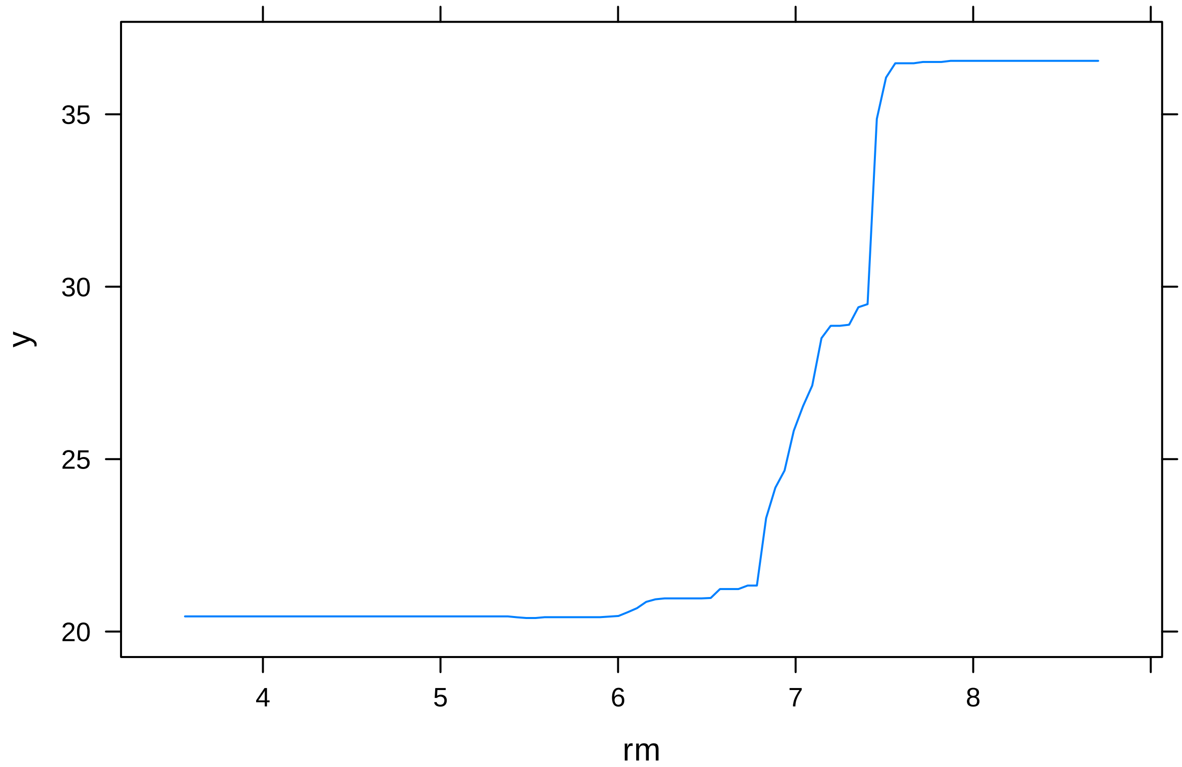
```r
boost_boston <- gbm(medv~., data= boston_trainset, distribution = "gaussian", n.trees = 1000, shrinkage
```

```r
summary(boost_boston)
```

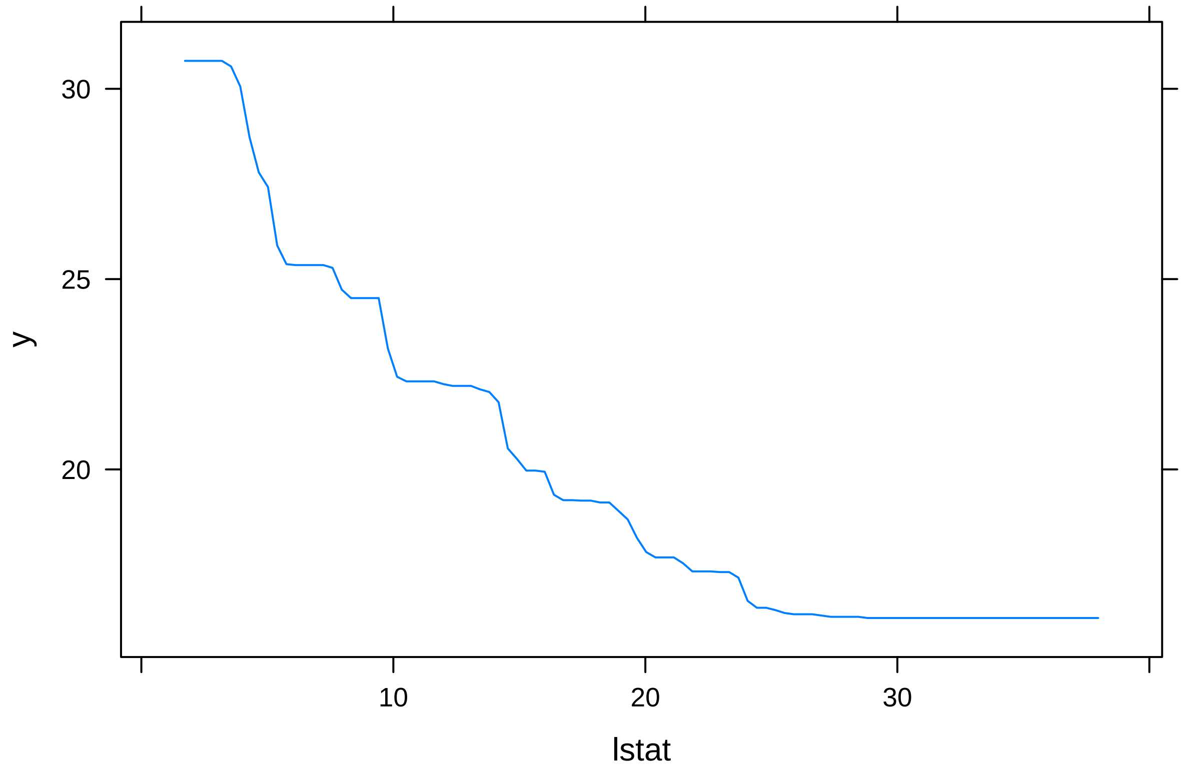



```
##             var    rel.inf
## rm           rm 44.2297637
## lstat     lstat 40.3512019
## crim       crim  3.3717265
## nox         nox  3.1261862
## ptratio ptratio  2.6692392
## dis         dis  2.4483048
## tax         tax  2.0432919
## age         age  0.5613363
## black     black  0.4514922
## indus     indus  0.3020250
## rad         rad  0.2712242
## chas       chas  0.1742080
## zn           zn  0.0000000
```

```r
par(mfrow = c(1, 2))
plot(boost_boston, i.var = "rm")
```




```r
plot(boost_boston, i.var = "lstat")
```




```r
# note: the `modelr::mse` function doesn't work
pred <- predict(boost_boston, boston_testset, n.trees=1000)
mean((boston_testset$medv - pred)^2)
```

```
## [1] 18.68067
```

Extreme gradient boosting

```r
library(xgboost)
```

```
##
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':
##
##     slice
```

```r
m <- model.matrix(medv~., data = boston_trainset)
xgboost_boston <- xgboost(data = m, label = boston_trainset$medv, nrounds = 1000, verbose = 0, objectiv
```

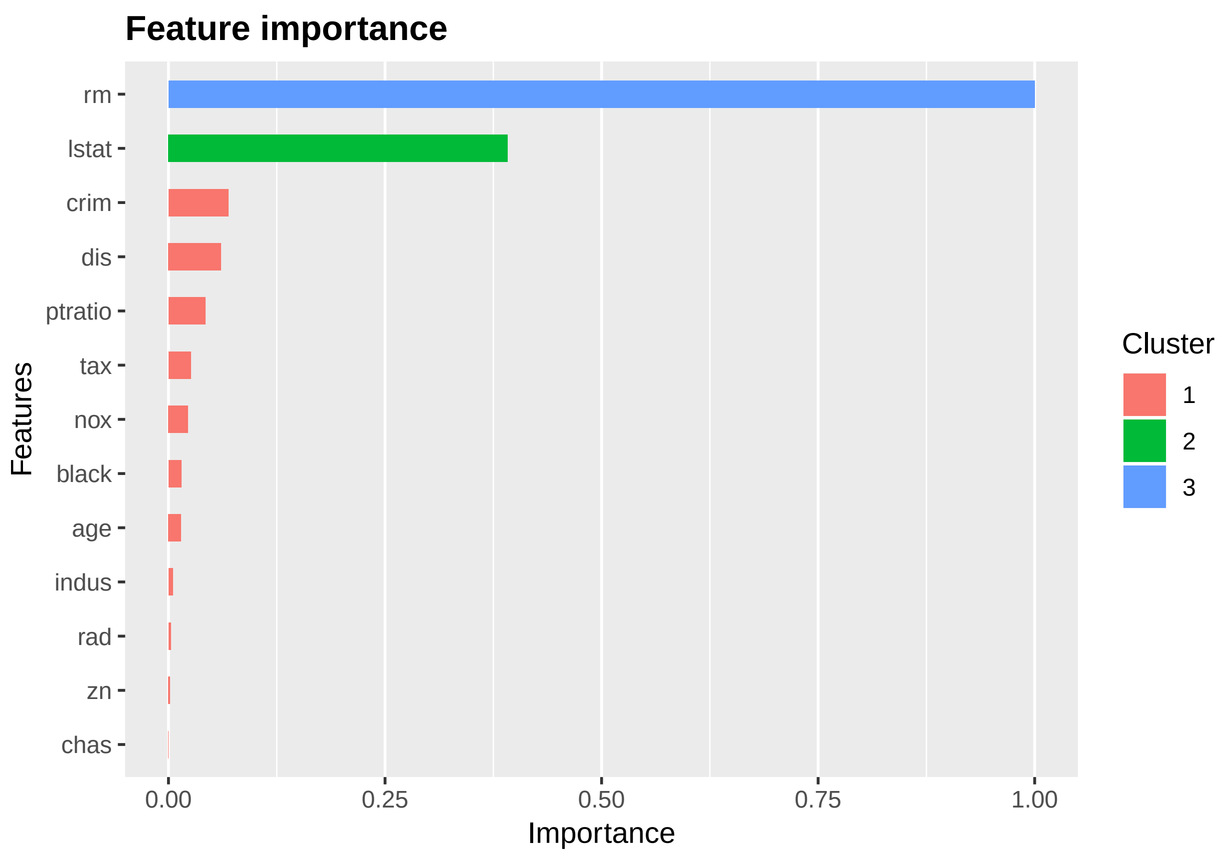
```r
# note: the `modelr::mse` function doesn't work
new_matrix <- model.matrix(medv~., data = boston_testset)
pred <- predict(xgboost_boston, new_matrix)
mean((boston_testset$medv - pred)^2)
```

```
## [1] 13.9866
```

```r
imp <- xgb.importance(model = xgboost_boston)
imp
```

```
##     Feature         Gain       Cover  Frequency
##  1:      rm 0.6053869345 0.198169351 0.16480707
##  2:   lstat 0.2371096034 0.168740867 0.11878196
##  3:    crim 0.0418583923 0.112793667 0.22501162
##  4:     dis 0.0368254919 0.123382525 0.09762901
##  5: ptratio 0.0258649151 0.034596053 0.03091585
##  6:     tax 0.0155595597 0.040550479 0.02440725
##  7:     nox 0.0137273492 0.039047420 0.05509066
##  8:   black 0.0089477988 0.103194001 0.08926081
##  9:     age 0.0087765474 0.130335780 0.11599256
## 10:   indus 0.0030135617 0.026849517 0.03974895
## 11:     rad 0.0016113940 0.011353235 0.01139005
## 12:      zn 0.0011828877 0.007133107 0.01603905
## 13:    chas 0.0001355643 0.003853998 0.01092515
```

```r
xgb.ggplot.importance(imp, rel_to_first = TRUE)
```


Feature importance
rm
lstat
crim
dis
ptratio
tax
nox
black
age
indus
rad
zn
chas
Features
0.00
0.25
0.50
0.75
1.00
Importance
Cluster
1
2
3