154Lab12

Jiyoon Clover Jeong

11/18/2017

```
attach(Carseats)
High <- ifelse(Sales <= 8, "No", "Yes")
carseats <- data.frame(Carseats, High)


tree_carseats <- tree(High ~ .-Sales , data=carseats)

tree_carseats
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##   1) root 400 541.500 No ( 0.59000 0.41000 )
##     2) ShelveLoc: Bad,Medium 315 390.600 No ( 0.68889 0.31111 )
##       4) Price < 92.5 46  56.530 Yes ( 0.30435 0.69565 )
##         8) Income < 57 10  12.220 No ( 0.70000 0.30000 )
##          16) CompPrice < 110.5 5   0.000 No ( 1.00000 0.00000 ) *
##          17) CompPrice > 110.5 5   6.730 Yes ( 0.40000 0.60000 ) *
##         9) Income > 57 36  35.470 Yes ( 0.19444 0.80556 )
##          18) Population < 207.5 16  21.170 Yes ( 0.37500 0.62500 ) *
##          19) Population > 207.5 20   7.941 Yes ( 0.05000 0.95000 ) *
##       5) Price > 92.5 269 299.800 No ( 0.75465 0.24535 )
##        10) Advertising < 13.5 224 213.200 No ( 0.81696 0.18304 )
##          20) CompPrice < 124.5 96  44.890 No ( 0.93750 0.06250 )
##            40) Price < 106.5 38  33.150 No ( 0.84211 0.15789 )
##              80) Population < 177 12  16.300 No ( 0.58333 0.41667 )
##               160) Income < 60.5 6   0.000 No ( 1.00000 0.00000 ) *
##               161) Income > 60.5 6   5.407 Yes ( 0.16667 0.83333 ) *
##              81) Population > 177 26   8.477 No ( 0.96154 0.03846 ) *
##            41) Price > 106.5 58   0.000 No ( 1.00000 0.00000 ) *
##          21) CompPrice > 124.5 128 150.200 No ( 0.72656 0.27344 )
##            42) Price < 122.5 51  70.680 Yes ( 0.49020 0.50980 )
##              84) ShelveLoc: Bad 11   6.702 No ( 0.90909 0.09091 ) *
##              85) ShelveLoc: Medium 40  52.930 Yes ( 0.37500 0.62500 )
##               170) Price < 109.5 16   7.481 Yes ( 0.06250 0.93750 ) *
##               171) Price > 109.5 24  32.600 No ( 0.58333 0.41667 )
##                 342) Age < 49.5 13  16.050 Yes ( 0.30769 0.69231 ) *
##                 343) Age > 49.5 11   6.702 No ( 0.90909 0.09091 ) *
##            43) Price > 122.5 77  55.540 No ( 0.88312 0.11688 )
##              86) CompPrice < 147.5 58  17.400 No ( 0.96552 0.03448 ) *
##              87) CompPrice > 147.5 19  25.010 No ( 0.63158 0.36842 )
##               174) Price < 147 12  16.300 Yes ( 0.41667 0.58333 )
##                 348) CompPrice < 152.5 7   5.742 Yes ( 0.14286 0.85714 ) *
##                 349) CompPrice > 152.5 5   5.004 No ( 0.80000 0.20000 ) *
##               175) Price > 147 7   0.000 No ( 1.00000 0.00000 ) *
##        11) Advertising > 13.5 45  61.830 Yes ( 0.44444 0.55556 )
##          22) Age < 54.5 25  25.020 Yes ( 0.20000 0.80000 )
```

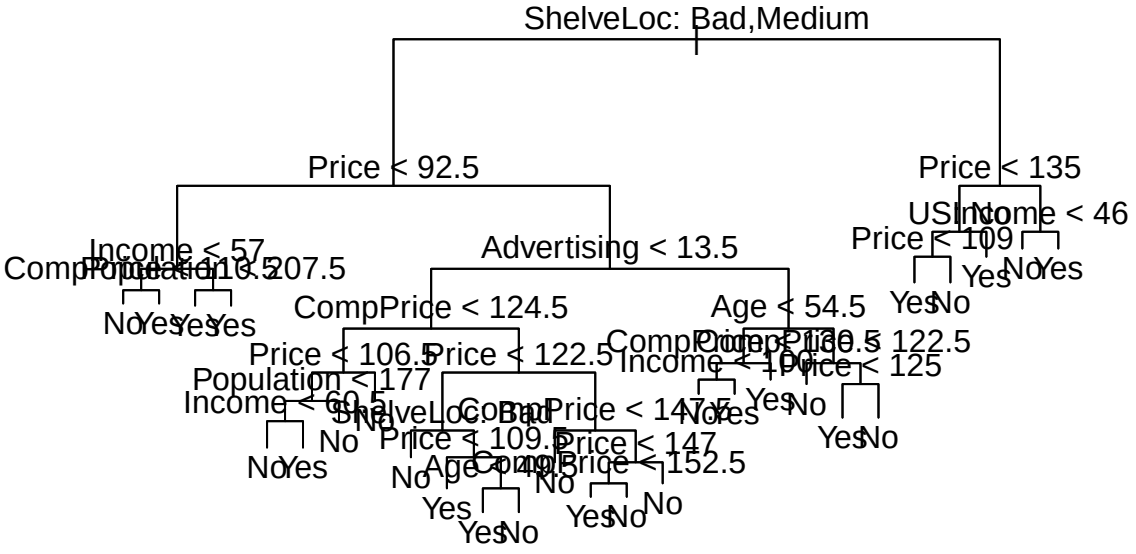
```
##           44) CompPrice < 130.5 14  18.250 Yes ( 0.35714 0.64286 )
##             88) Income < 100 9  12.370 No ( 0.55556 0.44444 ) *
##             89) Income > 100 5   0.000 Yes ( 0.00000 1.00000 ) *
##           45) CompPrice > 130.5 11   0.000 Yes ( 0.00000 1.00000 ) *
##         23) Age > 54.5 20  22.490 No ( 0.75000 0.25000 )
##           46) CompPrice < 122.5 10   0.000 No ( 1.00000 0.00000 ) *
##           47) CompPrice > 122.5 10  13.860 No ( 0.50000 0.50000 )
##             94) Price < 125 5   0.000 Yes ( 0.00000 1.00000 ) *
##             95) Price > 125 5   0.000 No ( 1.00000 0.00000 ) *
##     3) ShelveLoc: Good 85  90.330 Yes ( 0.22353 0.77647 )
##       6) Price < 135 68  49.260 Yes ( 0.11765 0.88235 )
##        12) US: No 17  22.070 Yes ( 0.35294 0.64706 )
##          24) Price < 109 8   0.000 Yes ( 0.00000 1.00000 ) *
##          25) Price > 109 9  11.460 No ( 0.66667 0.33333 ) *
##        13) US: Yes 51  16.880 Yes ( 0.03922 0.96078 ) *
##       7) Price > 135 17  22.070 No ( 0.64706 0.35294 )
##        14) Income < 46 6   0.000 No ( 1.00000 0.00000 ) *
##        15) Income > 46 11  15.160 Yes ( 0.45455 0.54545 ) *
```

Decision Trees

```
summary(tree_carseats)
```

```
##
## Classification tree:
## tree(formula = High ~ . - Sales, data = carseats)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Income"      "CompPrice"   "Population"
## [6] "Advertising" "Age"         "US"
## Number of terminal nodes:  27
## Residual mean deviance:  0.4575 = 170.7 / 373
## Misclassification error rate: 0.09 = 36 / 400
```

```
plot(tree_carseats)
text(tree_carseats, pretty=0)
```



```
tree_carseats
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##   1) root 400 541.500 No ( 0.59000 0.41000 )
##     2) ShelveLoc: Bad,Medium 315 390.600 No ( 0.68889 0.31111 )
##       4) Price < 92.5 46  56.530 Yes ( 0.30435 0.69565 )
##         8) Income < 57 10  12.220 No ( 0.70000 0.30000 )
##          16) CompPrice < 110.5 5   0.000 No ( 1.00000 0.00000 ) *
##          17) CompPrice > 110.5 5   6.730 Yes ( 0.40000 0.60000 ) *
##         9) Income > 57 36  35.470 Yes ( 0.19444 0.80556 )
##          18) Population < 207.5 16  21.170 Yes ( 0.37500 0.62500 ) *
##          19) Population > 207.5 20   7.941 Yes ( 0.05000 0.95000 ) *
##       5) Price > 92.5 269 299.800 No ( 0.75465 0.24535 )
##        10) Advertising < 13.5 224 213.200 No ( 0.81696 0.18304 )
##          20) CompPrice < 124.5 96  44.890 No ( 0.93750 0.06250 )
##            40) Price < 106.5 38  33.150 No ( 0.84211 0.15789 )
##              80) Population < 177 12  16.300 No ( 0.58333 0.41667 )
##               160) Income < 60.5 6   0.000 No ( 1.00000 0.00000 ) *
##               161) Income > 60.5 6   5.407 Yes ( 0.16667 0.83333 ) *
##              81) Population > 177 26   8.477 No ( 0.96154 0.03846 ) *
##            41) Price > 106.5 58   0.000 No ( 1.00000 0.00000 ) *
##          21) CompPrice > 124.5 128 150.200 No ( 0.72656 0.27344 )
##            42) Price < 122.5 51  70.680 Yes ( 0.49020 0.50980 )
##              84) ShelveLoc: Bad 11   6.702 No ( 0.90909 0.09091 ) *
##              85) ShelveLoc: Medium 40  52.930 Yes ( 0.37500 0.62500 )
##               170) Price < 109.5 16   7.481 Yes ( 0.06250 0.93750 ) *
##               171) Price > 109.5 24  32.600 No ( 0.58333 0.41667 )
##                 342) Age < 49.5 13  16.050 Yes ( 0.30769 0.69231 ) *
##                 343) Age > 49.5 11   6.702 No ( 0.90909 0.09091 ) *
##            43) Price > 122.5 77  55.540 No ( 0.88312 0.11688 )
##              86) CompPrice < 147.5 58  17.400 No ( 0.96552 0.03448 ) *
##              87) CompPrice > 147.5 19  25.010 No ( 0.63158 0.36842 )
##               174) Price < 147 12  16.300 Yes ( 0.41667 0.58333 )
##                 348) CompPrice < 152.5 7   5.742 Yes ( 0.14286 0.85714 ) *
##                 349) CompPrice > 152.5 5   5.004 No ( 0.80000 0.20000 ) *
##               175) Price > 147 7   0.000 No ( 1.00000 0.00000 ) *
##        11) Advertising > 13.5 45  61.830 Yes ( 0.44444 0.55556 )
##          22) Age < 54.5 25  25.020 Yes ( 0.20000 0.80000 )
##            44) CompPrice < 130.5 14  18.250 Yes ( 0.35714 0.64286 )
##              88) Income < 100 9  12.370 No ( 0.55556 0.44444 ) *
##              89) Income > 100 5   0.000 Yes ( 0.00000 1.00000 ) *
##            45) CompPrice > 130.5 11   0.000 Yes ( 0.00000 1.00000 ) *
##          23) Age > 54.5 20  22.490 No ( 0.75000 0.25000 )
##            46) CompPrice < 122.5 10   0.000 No ( 1.00000 0.00000 ) *
##            47) CompPrice > 122.5 10  13.860 No ( 0.50000 0.50000 )
##              94) Price < 125 5   0.000 Yes ( 0.00000 1.00000 ) *
##              95) Price > 125 5   0.000 No ( 1.00000 0.00000 ) *
##     3) ShelveLoc: Good 85  90.330 Yes ( 0.22353 0.77647 )
##       6) Price < 135 68  49.260 Yes ( 0.11765 0.88235 )
##        12) US: No 17  22.070 Yes ( 0.35294 0.64706 )
##          24) Price < 109 8   0.000 Yes ( 0.00000 1.00000 ) *
##          25) Price > 109 9  11.460 No ( 0.66667 0.33333 ) *
```

```
##        13) US: Yes 51  16.880 Yes ( 0.03922 0.96078 ) *
##       7) Price > 135 17  22.070 No ( 0.64706 0.35294 )
##        14) Income < 46 6   0.000 No ( 1.00000 0.00000 ) *
##        15) Income > 46 11  15.160 Yes ( 0.45455 0.54545 ) *
```

Random Forests

```
set.seed(100)

tree_carseats <- tree(High ~ .-Sales , data=carseats)

a <- createDataPartition(carseats[, 1], p = 0.8)$Resample1
train <- carseats[a, ]
test <- carseats[-a, ]


random <- randomForest(High ~ . - Sales, data = train, importance=TRUE)

random
```

```
##
## Call:
##  randomForest(formula = High ~ . - Sales, data = train, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 18.69%
## Confusion matrix:
##      No Yes class.error
## No  166  22   0.1170213
## Yes  38  95   0.2857143
```

```
summary(random)
```

```
##                 Length Class  Mode
## call               4   -none- call
## type               1   -none- character
## predicted        321   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes            642   matrix numeric
## oob.times        321   -none- numeric
## classes            2   -none- character
## importance        40   -none- numeric
## importanceSD      30   -none- numeric
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y                321   factor numeric
## test               0   -none- NULL
```

```
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
tree.pred <- predict (random , newdata = test ,type ="class")
tree.pred
```

```
##   2   3   5   8  11  13  19  23  28  30  32  35  48  54  56  59  72  75
## Yes Yes  No Yes  No  No Yes  No  No Yes  No  No  No  No  No  No  No  No
##  79  81  88  89  94 100 105 120 125 126 144 151 153 156 160 162 164 168
##  No Yes Yes  No  No  No  No  No  No Yes  No Yes Yes Yes  No  No  No  No
## 171 179 188 202 211 213 215 219 222 228 232 235 248 249 255 261 265 268
##  No Yes  No  No  No Yes  No Yes  No  No  No Yes  No  No Yes  No  No  No
## 270 277 278 286 290 294 298 301 302 317 319 326 328 331 339 341 352 356
##  No  No  No  No Yes Yes  No  No Yes Yes Yes Yes  No  No  No Yes Yes Yes
## 370 373 377 382 386 390 395
## Yes  No Yes  No  No  No  No
## Levels: No Yes
```

```
table <- table(tree.pred ,test$High)
table
```

```
##
## tree.pred No Yes
##       No  42  10
##       Yes  6  21
```

```
# error rate
(table[1,2] + table[2,1]) / (sum(table))
```

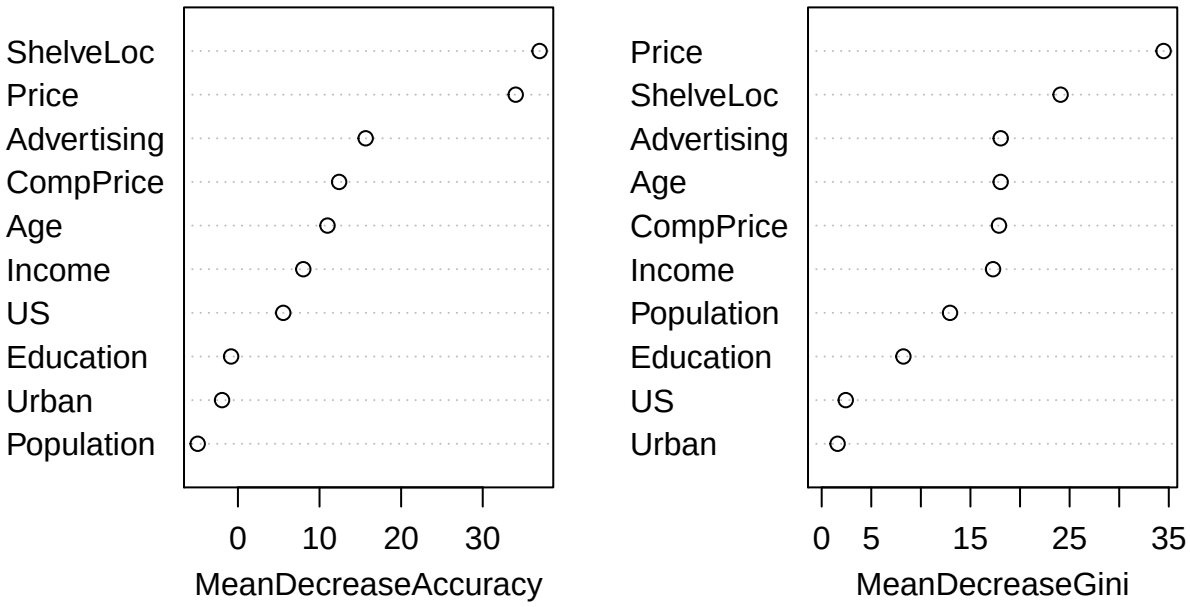
```
## [1] 0.2025316
```

```
#OOB Rate


randomForest::importance(random)
```

```
##                    No       Yes MeanDecreaseAccuracy MeanDecreaseGini
## CompPrice   10.3185504  8.293108           12.4018765        17.876753
## Income       5.1680899  6.192562            8.0110830        17.279201
## Advertising  8.6549309 14.406692           15.6539527        18.052183
## Population  -2.6662645 -4.707655           -4.9324078        12.941434
## Price       26.9617037 24.615768           34.0437151        34.475974
## ShelveLoc   28.5153098 28.321534           36.9699724        24.089643
## Age          6.5143800  9.653838           10.9757012        18.048773
## Education   -0.1232544 -1.115789           -0.8382232         8.239433
## Urban       -0.7577068 -2.159222           -1.9448340         1.603411
## US           2.2326986  4.806352            5.5517784         2.425390
```

```
varImpPlot(random)
```

random



Boosted Trees

```
set.seed(100)

n <- dim(carseats)[1]

carseats$High <- as.numeric(carseats$High) -1

head(carseats)
```

```
##   Sales CompPrice Income Advertising Population Price ShelveLoc Age
## 1  9.50       138     73          11        276   120       Bad  42
## 2 11.22       111     48          16        260    83      Good  65
## 3 10.06       113     35          10        269    80    Medium  59
## 4  7.40       117    100           4        466    97    Medium  55
## 5  4.15       141     64           3        340   128       Bad  38
## 6 10.81       124    113          13        501    72       Bad  78
##   Education Urban  US High
## 1        17   Yes Yes    1
## 2        10   Yes Yes    1
## 3        12   Yes Yes    1
## 4        14   Yes Yes    0
## 5        13   Yes  No    0
## 6        16    No Yes    1
```

```
train <- carseats[a, ]
test <- carseats[-a, ]
```

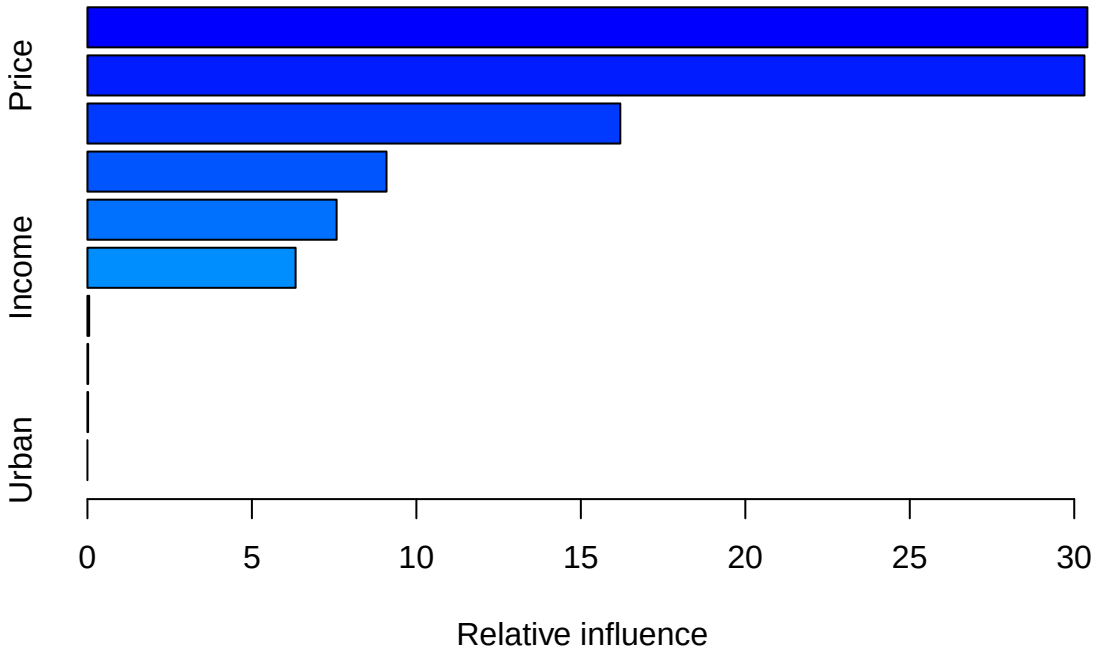
```
boost <- gbm(High ~ . - Sales, train, distribution="bernoulli",n.trees=5000)

yhat.boost <- predict(boost, newdata = test,n.trees =5000, type = "response")

predicted <- ifelse(yhat.boost >= 0.5, 1, 0)

mean(( predicted -test$High)^2)
```

```
## [1] 0.1772152
```

```
boost
```

```
## gbm(formula = High ~ . - Sales, distribution = "bernoulli", data = train,
##     n.trees = 5000)
## A gradient boosted model with bernoulli loss function.
## 5000 iterations were performed.
## There were 10 predictors of which 9 had non-zero influence.
```

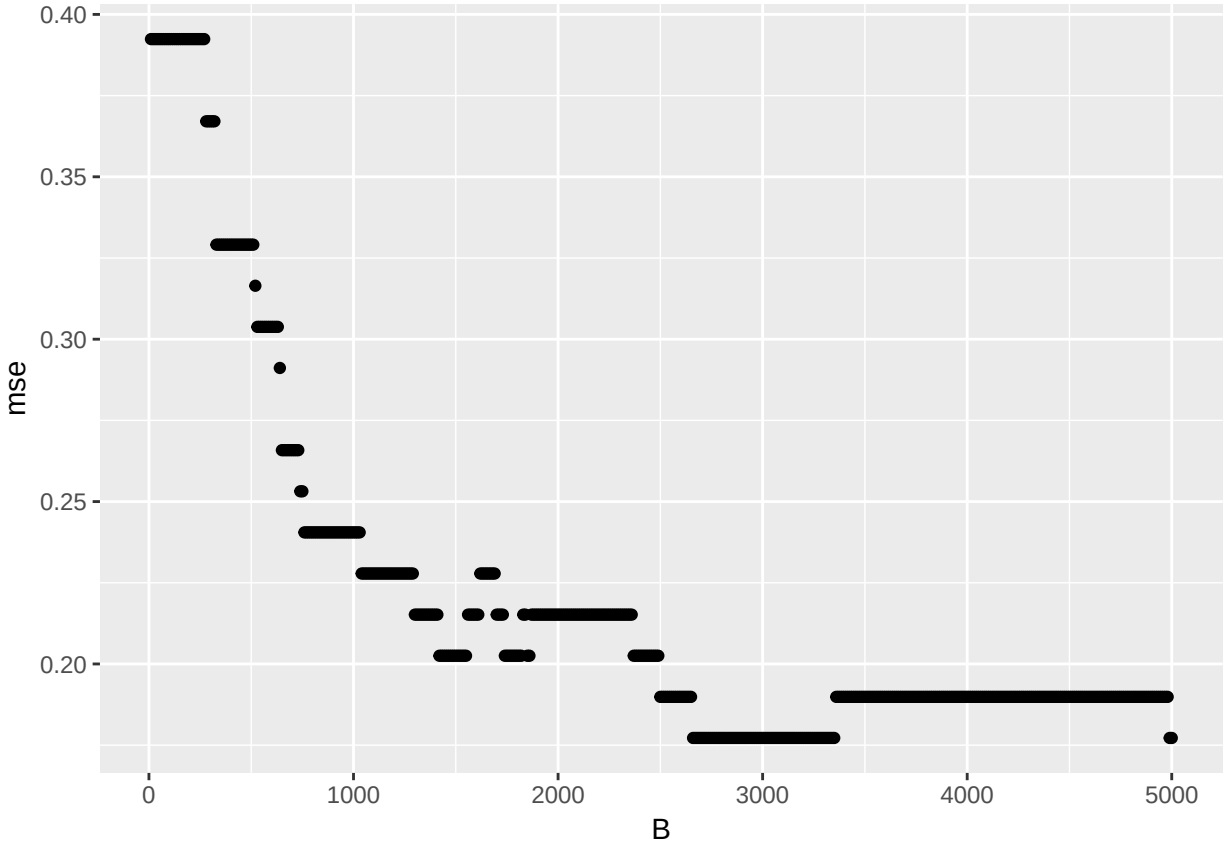
```
summary(boost)
```




```
##                    var     rel.inf
## ShelveLoc     ShelveLoc 30.39934546
## Price             Price 30.30979982
## Advertising Advertising 16.19979133
## Age                 Age  9.09227949
## CompPrice     CompPrice  7.57545195
## Income           Income  6.32826165
## Population   Population  0.05262820
## Education     Education  0.02191546
## US                   US  0.02052664
## Urban             Urban  0.00000000
```

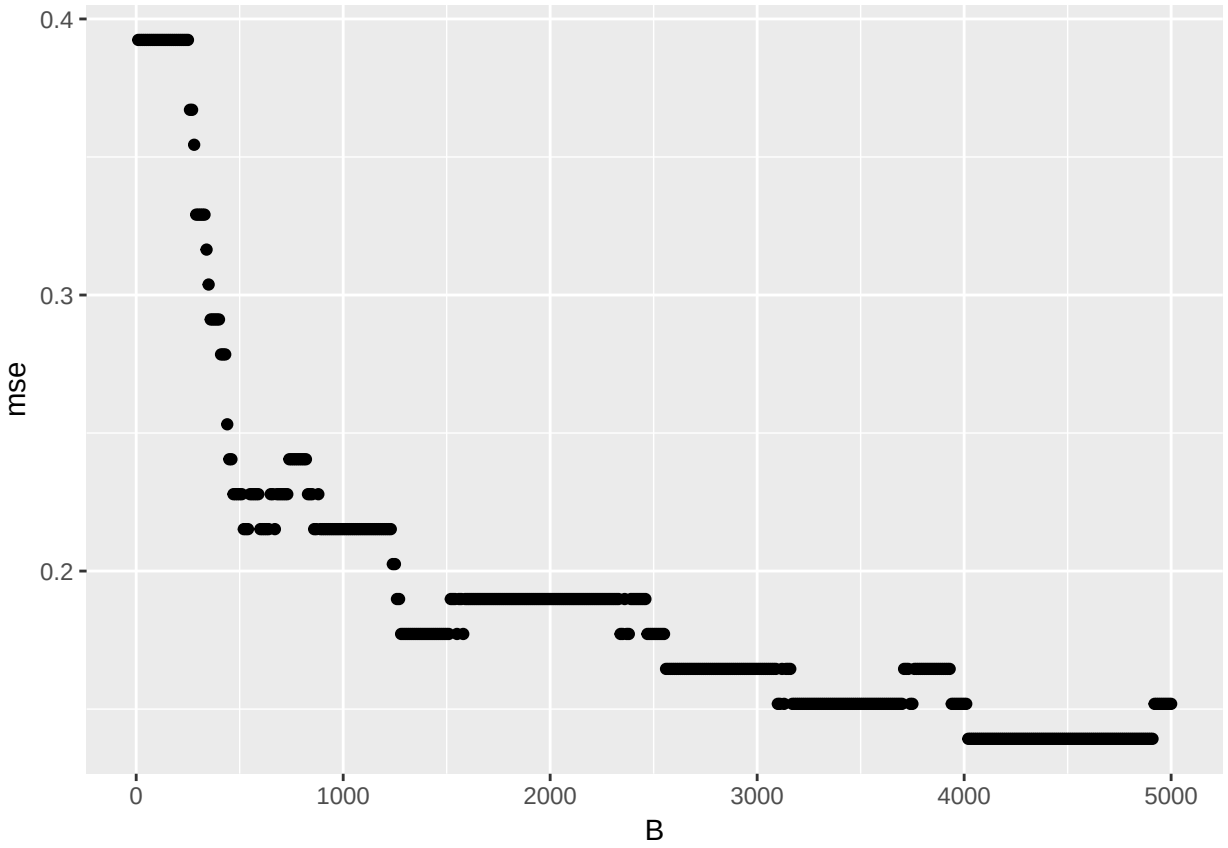
```
MSE <- c()
```

```
for(i in 1:500){
  yhat.boost <- predict(boost, newdata = test,n.trees =i * 10, type = "response")
  predicted <- ifelse(yhat.boost >= 0.5, 1, 0)
  MSE[i] <- mean(( predicted -test$High)^2)

}

df <- data.frame(mse = MSE, B = seq(10,5000,10))

ggplot(df, aes(x = B, y = mse)) + geom_point()
```

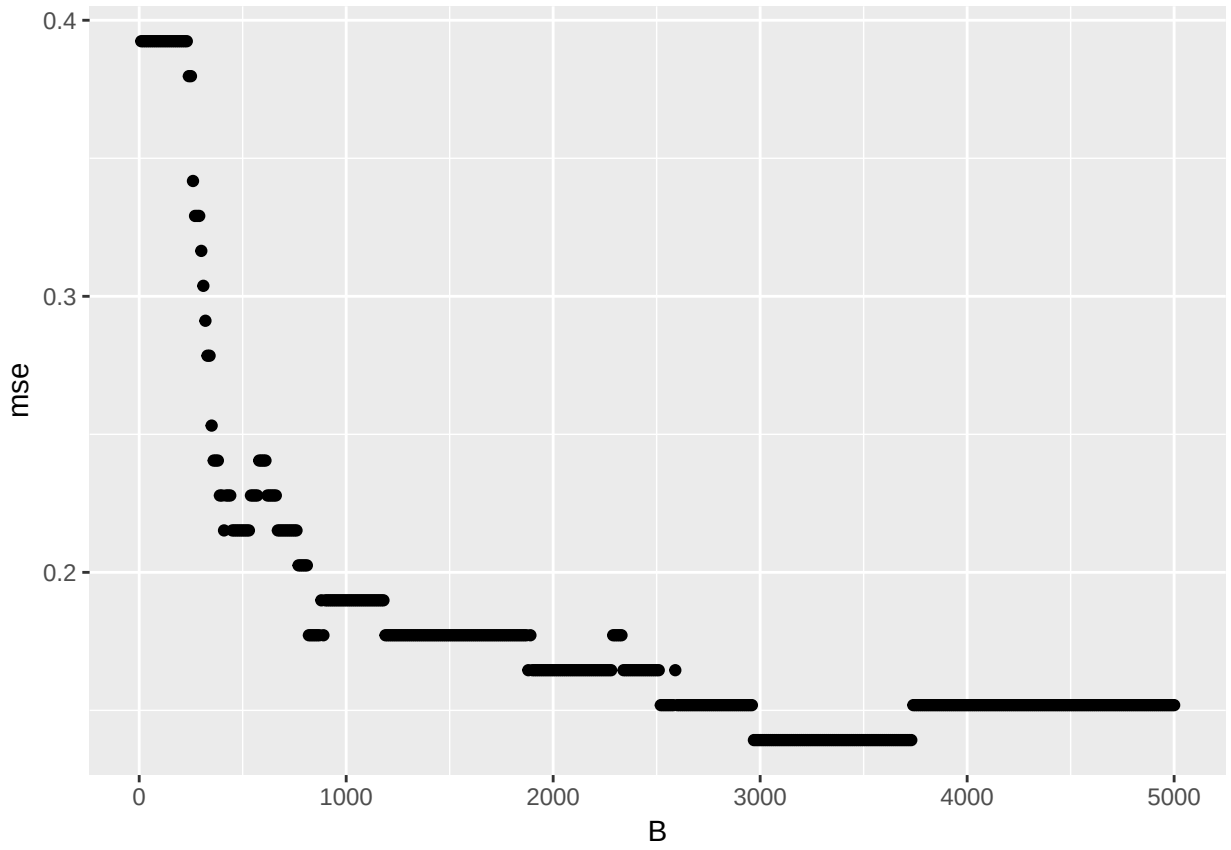



```
MSE <- c()



# d = 2

boost <- gbm(High ~ . - Sales, train, distribution="bernoulli",
             n.trees=5000, interaction.depth = 2)

for(i in 1:500){
  yhat.boost <- predict(boost, newdata = test,n.trees =i * 10, type = "response")
  predicted <- ifelse(yhat.boost >= 0.5, 1, 0)
  MSE[i] <- mean(( predicted -test$High)^2)

}
```

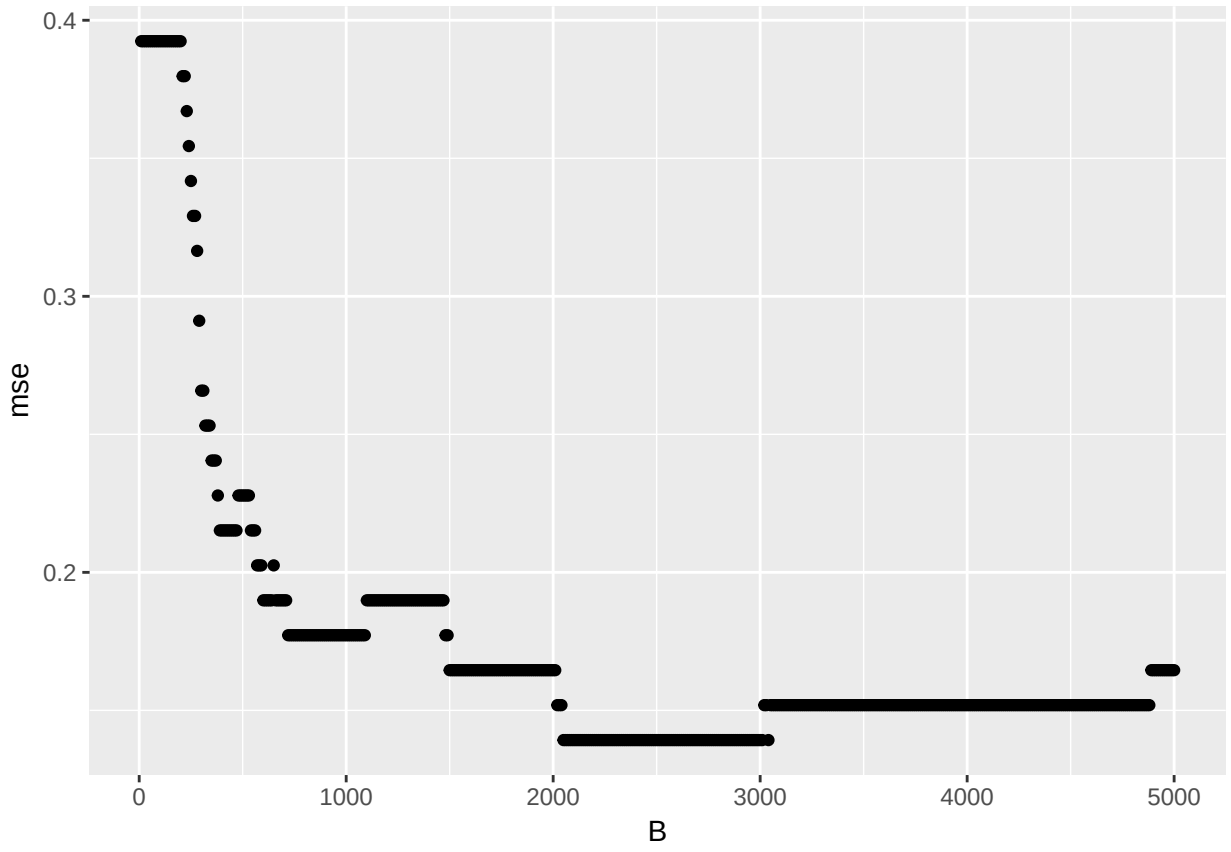
```
df <- data.frame(mse = MSE, B = seq(10,5000,10))

ggplot(df, aes(x = B, y = mse)) + geom_point()
```




```
# d = 3

boost <- gbm(High ~ . - Sales, train, distribution="bernoulli",
             n.trees=5000, interaction.depth = 3)

for(i in 1:500){
  yhat.boost <- predict(boost, newdata = test,n.trees =i * 10, type = "response")
  predicted <- ifelse(yhat.boost >= 0.5, 1, 0)
  MSE[i] <- mean(( predicted -test$High)^2)

}

df <- data.frame(mse = MSE, B = seq(10,5000,10))

ggplot(df, aes(x = B, y = mse)) + geom_point()
```




```
# d = 4

boost <- gbm(High ~ . - Sales, train, distribution="bernoulli",
             n.trees=5000, interaction.depth = 4)

for(i in 1:500){
  yhat.boost <- predict(boost, newdata = test,n.trees =i * 10, type = "response")
  predicted <- ifelse(yhat.boost >= 0.5, 1, 0)
  MSE[i] <- mean(( predicted -test$High)^2)

}

df <- data.frame(mse = MSE, B = seq(10,5000,10))

ggplot(df, aes(x = B, y = mse)) + geom_point()
```


0.4
0.3
0.2
mse
0
1000
2000
3000
4000
5000
B