18) Regression and Classification Trees

Vitor Kamada

March 2019

Reference

Tables, Graphics, and Figures from:

1) Hastie et al. (2017): Ch 9.2

2) James et al. (2017): Ch 8.1, 8.3.1, and 8.3.2

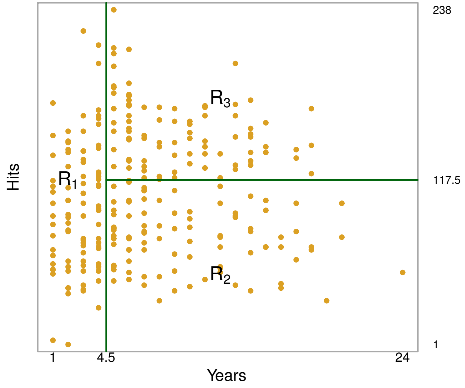
Hitters Data

$$e^{5.11} \cong \$165K, \; e^{6} \cong \$402K, \; e^{6.74} \cong \$845K$$

Three-Region Partition

$$R_3 = \{X | Years \geq 4.5, Hits \geq 117.5\}$$

$$R_1 = \{X | Years < 4.5\}$$



$$R_2 = \{X | Years \geq 4.5, Hits < 117.5\}$$

OLS vs Trees (Impurity Measure)

$$f(x) = \beta_0 + \sum_{j=1}^{p} \beta_j x_j$$

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$$

$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2$$

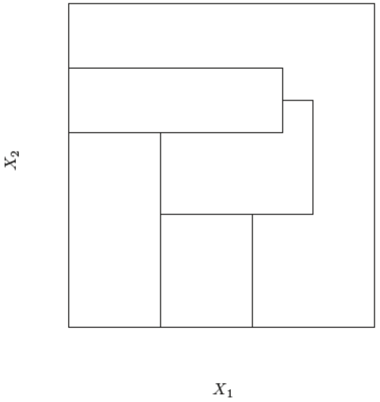
Top-down Greedy (Recursive Binary Splitting)

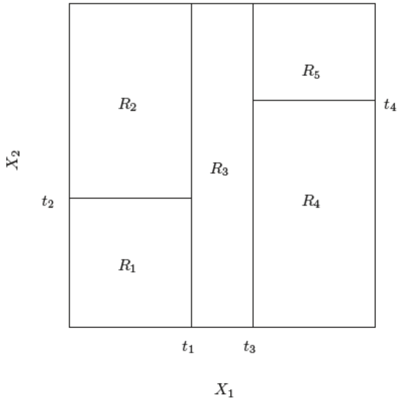
$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

$$R_1(j,s) = \{X | X_j < s\}$$
$$R_2(j,s) = \{X | X_j \geq s\}$$

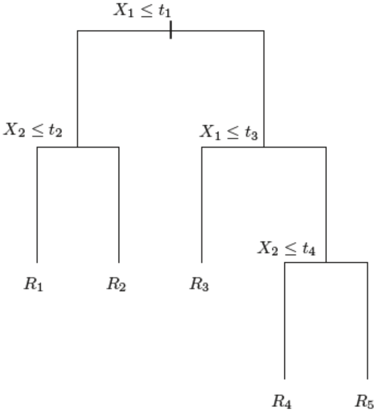
$$\sum_{i: x_i \epsilon R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \epsilon R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

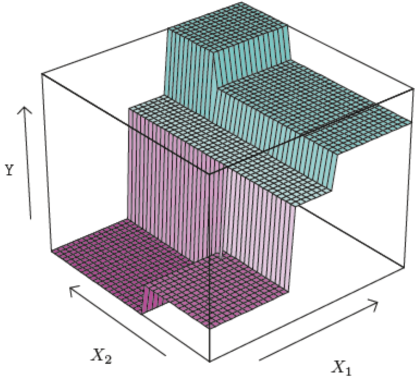
No Recursive Binary Splitting vs Recursive Binary Splitting





Tree and Perspective Plot



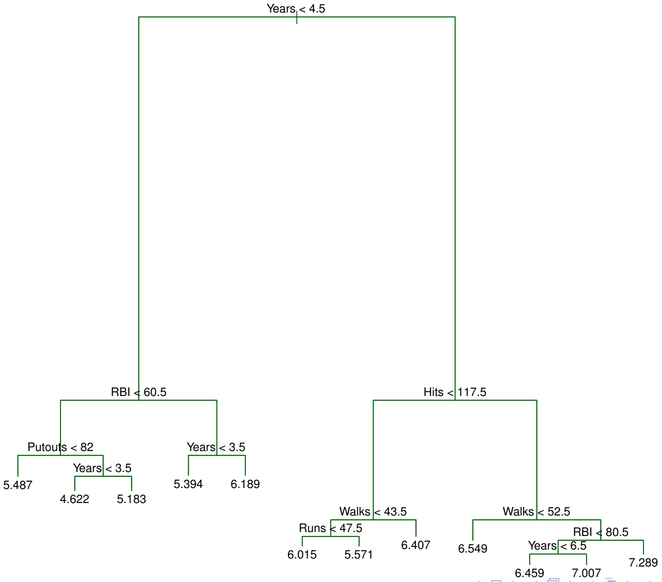


Cost Complexity Pruning (Weakest Link Pruning)

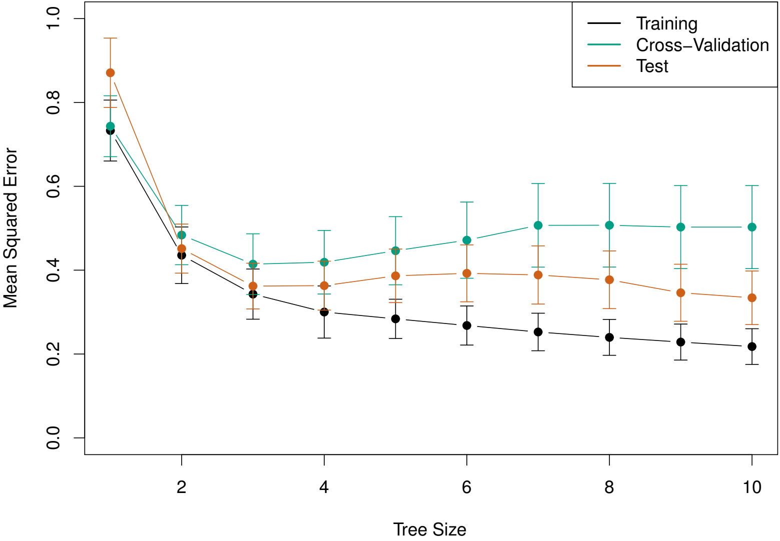
$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

$|T| = \#$ of terminal nodes of the tree

Unpruned Tree (Top-down Greedy Splitting)



Six-Fold Cross-Validation for Pruning Tree



Training Error Rate, Gini Index, and Entropy

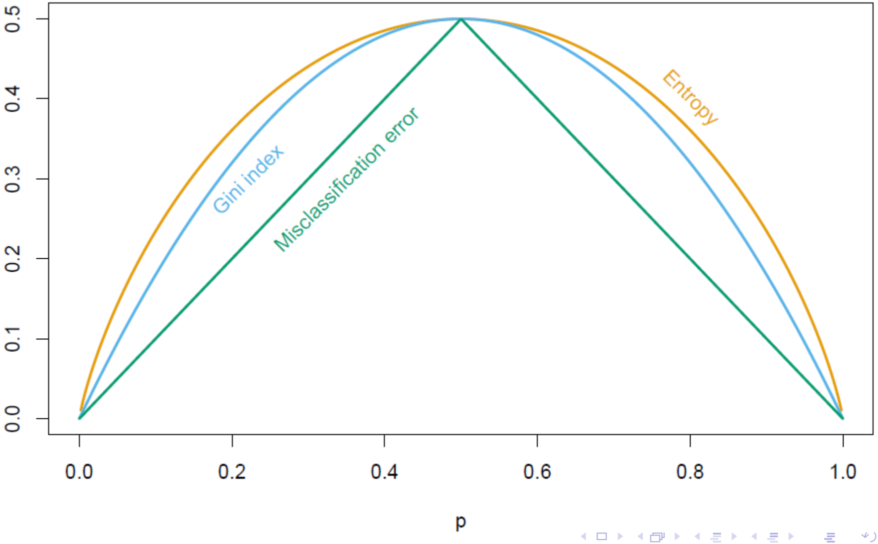
$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

$$E = 1 - \max_k (\hat{p}_{mk})$$

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

$$D = - \sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

Gini Index and Entropy are more sensitive to changes in the node probabilities



Heart Data Set

303 patients

AHD: Yes for heart disease based on an angiographic test

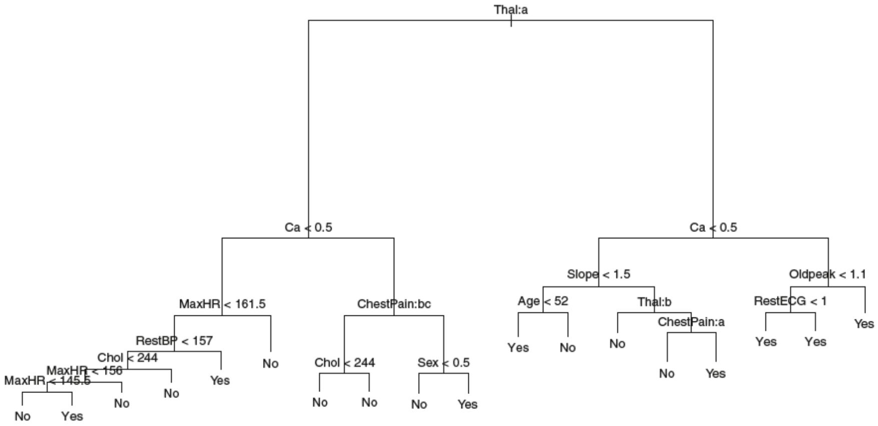
Thal: Thallium stress test, nuclear imaging shows how blood flows into heart

ChestPain: angina, atypical angina, non-anginal pain, and asymptomatic

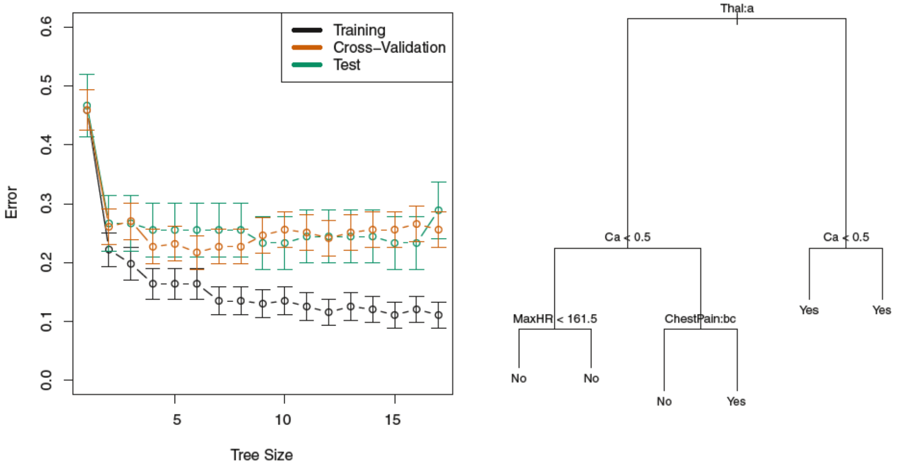
RestECG: Electrocardiograms

Heart Data: Unpruned Tree

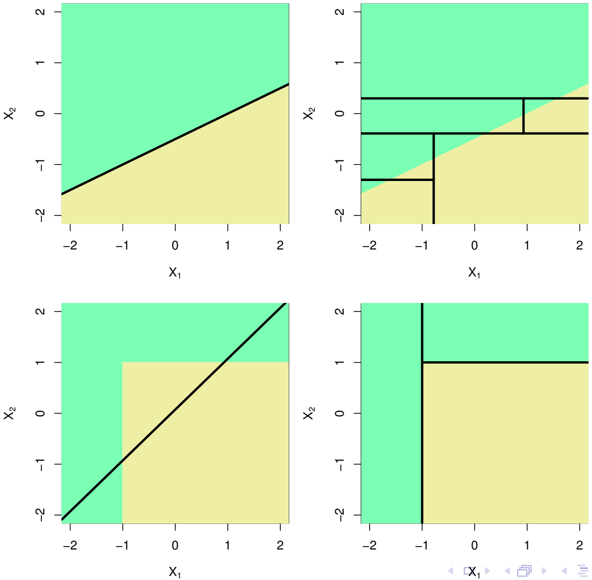
Normal < |Thal:a| < Fixed or Reversible Defects



Pruned Tree (Minimal Cross-Validation Error)



Linear vs Non-linear True Decision Boundary



library(ISLR); library(stargazer); stargazer(Carseats)

Statistic	N	Mean	St. Dev.	Min	Max
Sales	400	7.496	2.824	0.000	16.270
CompPrice	400	124.975	15.335	77	175
Income	400	68.657	27.986	21	120
Advertising	400	6.635	6.650	0	29
Population	400	264.840	147.376	10	509
Price	400	115.795	23.677	24	191
Age	400	53.322	16.200	25	80
Education	400	13.900	2.621	10	18

library(tree); High=ifelse(Sales<=8,"No","Yes")

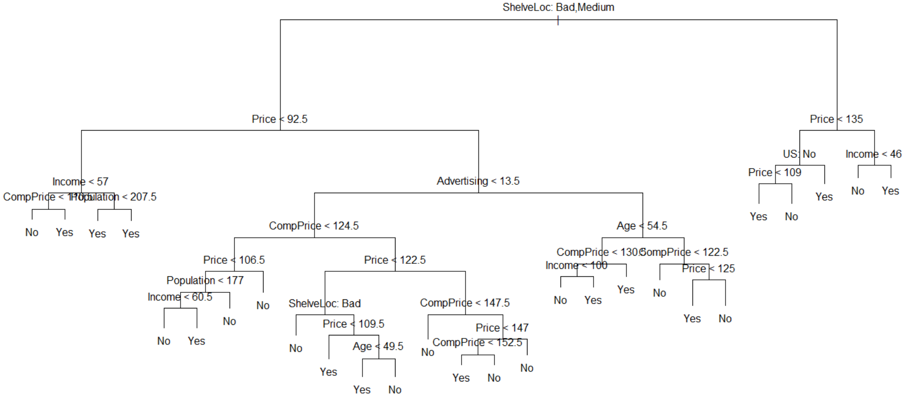
Carseats=data.frame(Carseats,High)

tree.carseats=tree(High~.-Sales,Carseats);
summary(tree.carseats)

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

$$\frac{-2\sum_m\sum_k n_{mk} \log \hat{p}_{mk}}{n-|T_0|} = \frac{170.7}{400-27} = 0.45$$

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 ) *
```

set.seed(2); train=sample(1:nrow(Carseats), 200)

```
Carseats.test=Carseats[-train,]

High.test=High[-train]

tree.carseats=tree(High~.-Sales,Carseats,subset=train)

tree.pred=predict(tree.carseats,Carseats.test,type="class")

table(tree.pred,High.test)
```

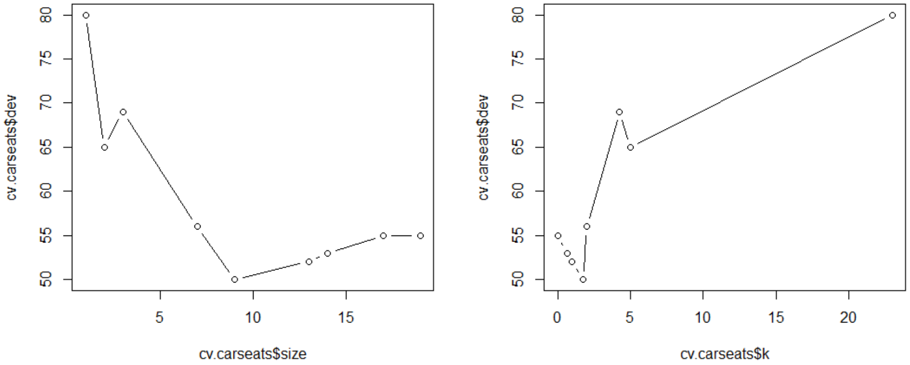
```
         High.test
tree.pred No Yes
      No  86  27
      Yes 30  57
```

$$\frac{86+57}{200} = 71.5\%$$

set.seed(3); cv.carseats=cv.tree(tree.carseats, FUN=prune.misclass); par(mfrow=c(1,2))

plot(cv.carseats$size,cv.carseats$dev,type="b")

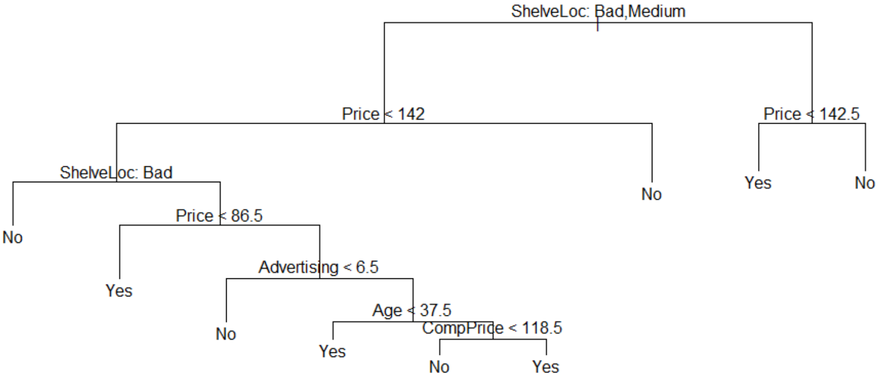
plot(cv.carseats$k,cv.carseats$dev,type="b")



prune.carseats=prune.misclass(tree.carseats,best=9)

plot(prune.carseats)

text(prune.carseats,pretty=0)



tree.pred=predict(prune.carseats, Carseats.test,type="class")

table(tree.pred,High.test)

```
          High.test
tree.pred No Yes
      No  94  24
      Yes 22  60
```

$$\frac{94+60}{200} = 77\%$$

prune.carseats=prune.misclass(tree.carseats,best=15)

tree.pred=predict(prune.carseats, Carseats.test,type="class")

table(tree.pred,High.test)

```
          High.test
tree.pred  No Yes
      No   86  22
      Yes  30  62
```

$$\frac{86+62}{200} = 74\%$$