Tree Based Methods: Classification Tree

Section 8.1

Cathy Poliak, Ph.D.
cpoliak@central.uh.edu

Department of Mathematics
University of Houston

Tree Based Models

So far we have covered such (relatively) basic models as:

- Linear Regression
- Logistic Regression

and such resampling techniques as

- Cross-Validation
- Bootstrap

we are ready for another model class:

- **Tree-based models.**

Tree-based models can be applied to **both** regression and classification problems.

Classification Trees

- Used to predict a qualitative (categorical) response.
- Recall regression trees - predicted response for an observation is given by the mean response of the training observations that belong to the same terminal node.
- Classification tree - predicts that each observation belongs to the **most commonly occurring class** of training observations in the region to which it belongs.
- Interpretation of classification tree - class prediction corresponding to a particular terminal node regions and the class proportions among the training observations that fall into that region.

Growing a Classification Tree

- Task of growing a classification tree is similar to the regression trees, we use recursive binary splitting to grow a classification tree.

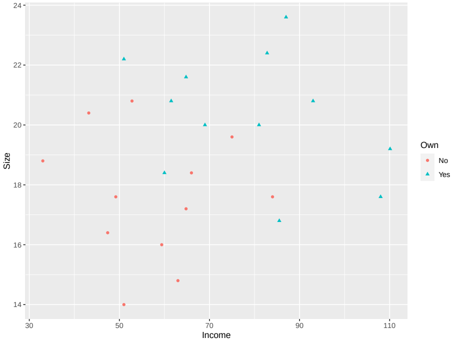
  $RSS = \sum (y_i - \bar{y}_{R_i})^2$

- Recall criterion for regression tree is **RSS**. Since the response is categorical we cannot calculate the residual standard error. Thus we use other criterion to grow a classification tree.
  - Classification error rate
  - Gini index
  - Entropy

Example

From *Applied Multivariate Statistical Analysis* by Johnson and Wichern:

A riding-mower manufacturer would like to find a way of classifying families in a city into those that are likely to purchase a riding mower and those who are not likely to buy one. A pilot random sample of 12 owners and 12 non-owners in the city is undertaken. The data are plotted below. The independent variables here are Income (x1) and Lot Size (x2). The categorical y variable has two classes: owners and non-owners.



Classification Error Rate

- A natural alternative to RSS
- The fraction of the training observations in that region that do not belong to the most common class:

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

  Where $\hat{p}_{mk}$ represents the portion of training observations in the mth region that are from the kth class.

- This is not sufficiently sensitive for tree-growing. This only finds the best split at that immediate place.

Gini Index

- A measure of total variance across the K classes.

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

- This is measure of node purity - a small value indicates that a node contains predominantly observations from a single class.
- Measures the degree or probability of a particular variable being wrongly classified when it is randomly chosen.
- For more information and more examples see: Gini Example.

Entropy

Given by

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

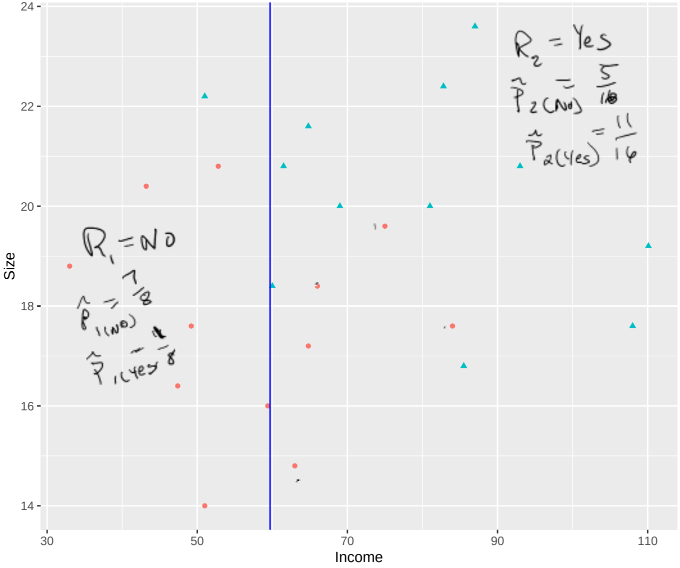
- Since $0 \leq \hat{p}_{mk} \leq 1$, then $0 \leq -\hat{p}_{mk} \log \hat{p}_{mk}$
- The entropy will take on a value near zero if the $\hat{p}_{mk}$ are all near zero or one.
- The entropy will be a small value if the *m*th node is pure.
- Also called **Information gain** measurement.
- This gives us the maximum information about a class.
- Harder to compute because of the *log*, thus Gini index is preferred.

What Method to Use

- When building a classification tree, either the Gini index or the entropy are typically used to evaluate the quality of a particular split.
- Any of the three approaches might be used when **pruning** the tree.
- The classification error rate is preferable if prediction accuracy of the final pruned tree is the goal.
- The `tree` function uses the entropy ~~classification error~~ rate as the default but can also use the Gini Index.

Split Using Gini Index

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



Own
- No
- Yes

$$G_1 = \left(\tfrac{7}{8}\right)\left(\tfrac{1}{8}\right) + \left(\tfrac{1}{8}\right)\left(\tfrac{7}{8}\right)$$
$$= 0.21875$$

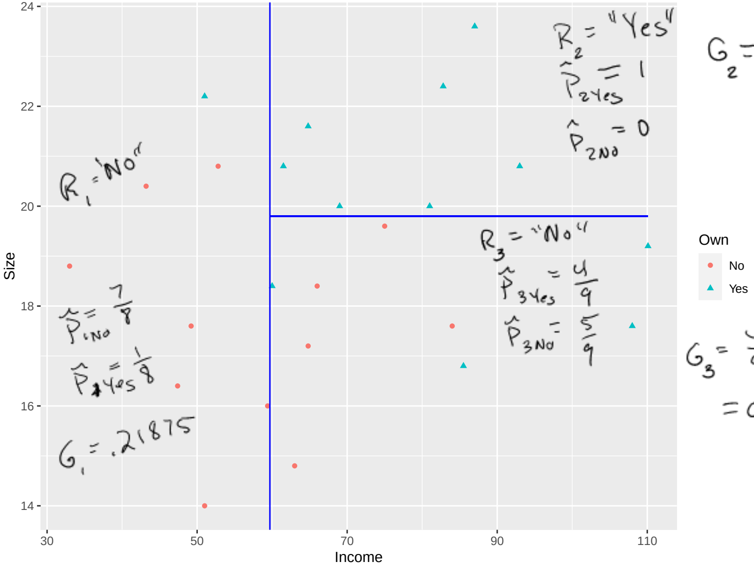
$$G_2 = \left(\tfrac{5}{16}\right)\left(\tfrac{11}{16}\right) + \left(\tfrac{11}{16}\right)\left(\tfrac{5}{16}\right)$$
$$= 0.4297$$

R_1 3-classes A, B, C $n = 10$ $A = 3$ $B = 2$ $C = 5$

$\hat{p}_{1A} = \frac{3}{10}$ $\hat{p}_{1B} = \frac{2}{10}$ $\hat{p}_{1C} = \frac{5}{10}$

$G_1 = \frac{3}{10}\left(\frac{7}{10}\right) + \frac{2}{10}\left(\frac{8}{10}\right) + \frac{5}{10}\left(\frac{5}{10}\right)$

Second Split



$G_2 = 0$

$G_3 = \frac{4}{9}\left(\frac{5}{9}\right) + \frac{5}{9}\left(\frac{4}{9}\right)$
$= 0.4938$

Classification Trees in R

```
library(tree)
mower$Own = as.factor(mower$Own)
tree.mower = tree(Own ~ Income + Size,data = mower)
```

Need to make sure that the categorical variable is of a factor.

$-2\sum_{k} n_i \log(p_i)$

```
> tree.mower.gini
node), split, n, deviance, yval, (yprob)
     * denotes terminal node

1) root 24 33.270 No ( 0.5000 0.5000 )
  2) Income < 59.7 8  6.028 No ( 0.8750 0.1250 ) *
  3) Income > 59.7 16 19.870 Yes ( 0.3125 0.6875 )
    6) Size < 19.8 9 12.370 No ( 0.5556 0.4444 ) *
    7) Size > 19.8 7  0.000 Yes ( 0.0000 1.0000 ) *
```

Results

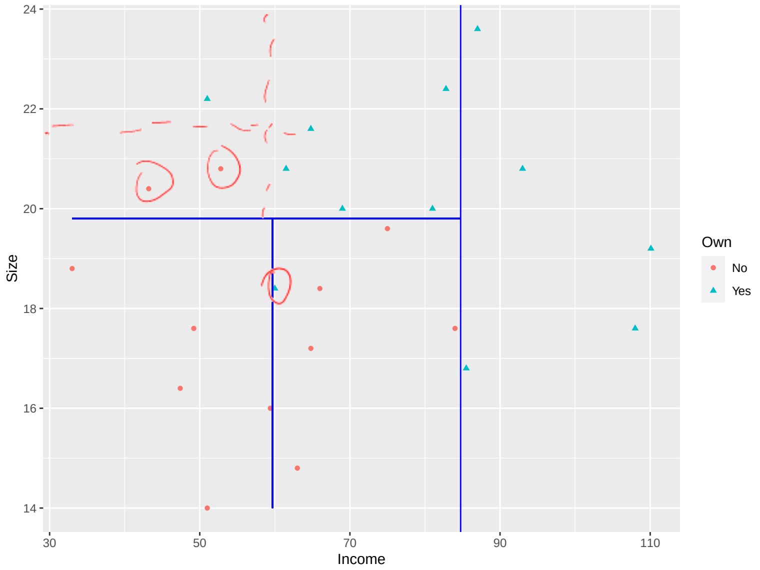
The calculation for the *deviance* is $-2\sum_{i=1}^{k} n_i log(p_i)$.

```
> tree.mower
node), split, n, deviance, yval, (yprob)
* denotes terminal node

1) root 24 33.270 No ( 0.50000 0.50000 )
  2) Income < 84.75 19 25.010 No ( 0.63158 0.36842 )
    4) Size < 19.8 11  6.702 No ( 0.90909 0.09091 )
      8) Income < 59.7 5  0.000 No ( 1.00000 0.00000 ) *
      9) Income > 59.7 6  5.407 No ( 0.83333 0.16667 ) *
    5) Size > 19.8 8  8.997 Yes ( 0.25000 0.75000 ) *
  3) Income > 84.75 5  0.000 Yes ( 0.00000 1.00000 ) *
```

$-2(12\log(.5) + 12\log(.5))$

$-2(12\log(0.63158) + 7\log(0.36842))$



24
22
20
18
16
14
Size
30
50
70
90
110
Income
Own
No
Yes

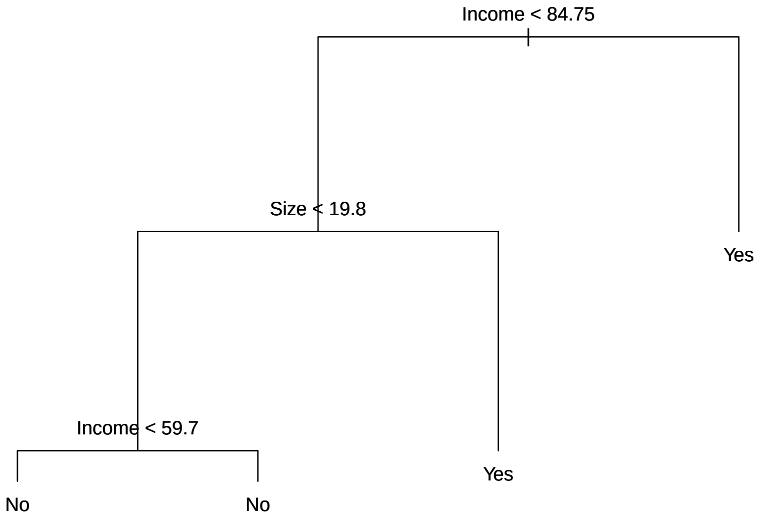
Summary In R

```
> summary(tree.mower)

Classification tree:
tree(formula = Own ~ Income + Size, data = mower)
Number of terminal nodes:  4
Residual mean deviance:  0.7202 = 14.4 / 20
Misclassification error rate: 0.125 = 3 / 24
```

Sum of deviance for the terminal Nodes

$N - |T_\alpha| = 24 - 4$


Income < 84.75
Size < 19.8
Yes
Income < 59.7
Yes
No
No

Example 2

- We will use the *Heart* data. See BlackBoard to get the data.
- This data contains info on patients with chest pains, and we'd like to classify if a patient has a heart disease (*AHD*) depending on multiple factors.
- Import the data into `R` type and run the following:

```
set.seed(100)
train = sample(1:nrow(Heart),nrow(Heart)/2+0.5)
Heart$AHD = as.factor(Heart$AHD)
Heart$ChestPain = as.factor(Heart$ChestPain)
Heart$Thal = as.factor(Heart$Thal)
tree.heart = tree(AHD ~ . -X, Heart,subset =  train)
```

Lab Questions

1. Type and run summary(tree.heart). What are the number of terminal nodes?

   a) 9 b) 10 c) 11 d) 13

2. What is the training error rate?

   a) 38.5% b) 10.96% c) 14% d) 51.6%

```
Classification tree:
tree(formula = AHD ~ ., data = Heart, subset = train)
Variables actually used in tree construction:
[1] "Ca"       "Thal"     "Chol"     "RestBP"    "ChestPain" "Oldpeak"
[7] "Slope"    "Sex"      "Age"
Number of terminal nodes:  11
Residual mean deviance:  0.4439 = 59.92 / 135
Misclassification error rate: 0.1096 = 16 / 146
```

3. Type and run the following in R
```
plot(tree.heart)
text(tree.heart)
```
Is a person predicted to have heart disease if the Ca > 0.5, slope < 1.5 and sex $= 0$.

a) Yes
b) No

Test Error Rate

In order to properly evaluate the performance of a classification tree on these data, we must estimate the test error rather than simply computing the training error. Type and run the following in R

```
Heart.test = Heart[-train,]
tree.pred = predict(tree.heart,Heart.test,type = "clas
table(tree.pred,Hear.test$AHD)
```

4. What is the test error rate?

   a) 18.34%
   b) 81.66%
   c) 21%
   d) 17.5%

```
> table(tree.pred,Heart.test$AHD)

tree.pred No Yes
      No  60  18
      Yes 13  60
```

$$\frac{19+13}{60+18+13+60}$$

Pruning

- Next, we consider whether pruning the tree might lead to improved results.
- We use the argument `FUN=prune.misclass` in the `cv.tree()` function in order to indicate that we want the classification error rate to guide the cross-validation and pruning process, rather than the default for the `cv.tree()` function, which is deviance.
- The `cv.tree()` function reports the number of terminal nodes of each tree considered (size) as well as the corresponding error rate and the value of the cost-complexity parameter used (k, which corresponds to α).

Type and run the following in R

```
set.seed(3)
cv.heart = cv.tree(tree.heart,FUN = prune.misclass)
par(mfrow = c(1,2))
plot(cv.heart$size,cv.heart$dev,type = "b")
plot(cv.heart$k,cv.heart$dev,type = "b")
```

5. How many nodes do we really desire?

a) 14
b) 10
c) 4
d) 2

Pruned Tree

Type and run the following:

```
prune.heart = prune.misclass(tree.heart,best = 6)
tree.pred = predict(prune.heart,Heart.test,type = "class")
table(tree.pred,Heart.test$AHD)
```

6. What is the test error rate?

a) 19.2%
b) 73.5%
c) 31.75%
d) 26.5%

```
> table(tree.pred,Heart.test$AHD)

tree.pred No Yes
     No  63  30
     Yes 10  48
```

Trees vs Linear Models

Linear regression assumes a model of the form

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j,$$

Regression trees assume a model of the form

$$f(X) = \sum_{m=1}^{M} c_m \mathbf{1}_{(X \in R_m)}$$

Which is better?

Trees vs Linear Models

Linear regression assumes a model of the form

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j,$$

Regression trees assume a model of the form

$$f(X) = \sum_{m=1}^{M} c_m \mathbf{1}_{(X \in R_m)}$$

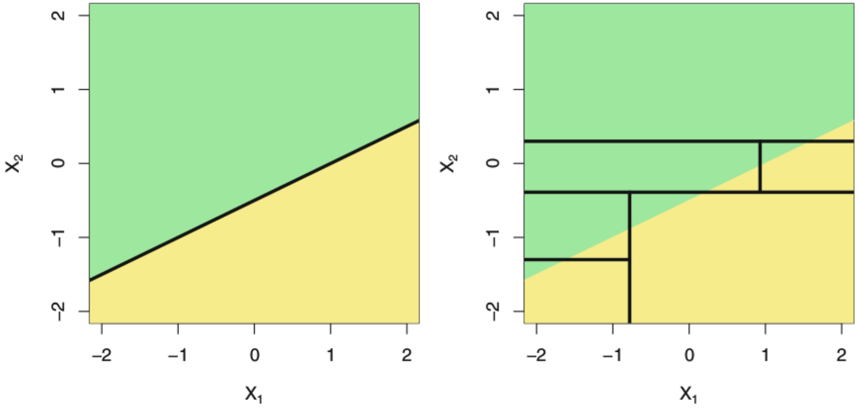
Which is better?

Data Scientist answer: it depends on the problem and data at hand,

- If relationship between predictors $X_1, \ldots, X_p$ and Y is approximately linear $\Longrightarrow$ linear model it is.
- Otherwise, if that relationship is highly non-linear and complex $\Longrightarrow$ decision trees may have an edge.

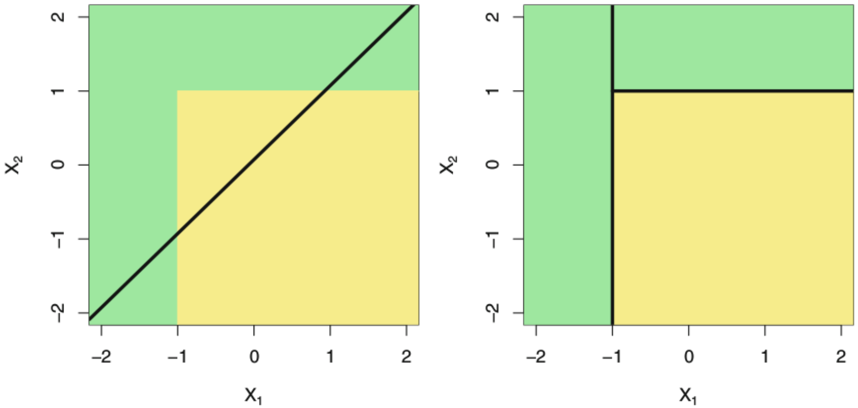
Classification example: Linear preferred over Trees

Example. In the case of a classification problem with a linear boundary - linear approach is equipped to perform better. Below you see results of fitting a linear model (left) and a decision tree (right).



Classification example: Trees preferred over Linear

Example. In classification problem with a more complex, non-linear boundary - decision trees have better chances. Below you see results of fitting a linear model (left) and a decision tree (right).



Decision Trees: Advantages and Disadvantages

Several **advantages** of decision trees as a model:

1. Easy to explain, visualize and interpret.
2. More closely mirror human decision-making than regression and certain other methods.
3. Easily handle both quantitative and qualitative predictors (and responses).

The biggest downside:

1. Trees generally do not have the same level of predictive accuracy as some other regression and classification approaches.

However, by aggregating many decision trees (e.g. bagging, random forests), the predictive performance of trees can be vastly improved.