STAT 380 – Decision Trees Part 2 (Lecture 16)

NOTE: Advantages and disadvantages of CART (Classification And Regression Trees):

- Pros
  - Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression!
  - Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches seen in previous chapters.
  - Trees can be displayed graphically and are easily interpreted even by a non-expert (especially if the trees are small).
  - Trees can easily handle categorical predictors without the need to create dummy variables.
- Cons
  - Trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.
  - Trees can be very non-robust. In other words, a small change in the data can cause a large change in the final estimated tree.

Classification Trees

IDEA: A classification tree is very similar to a regression tree, except that it is used to predict a categorical response rather than a quantitative one. For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs.

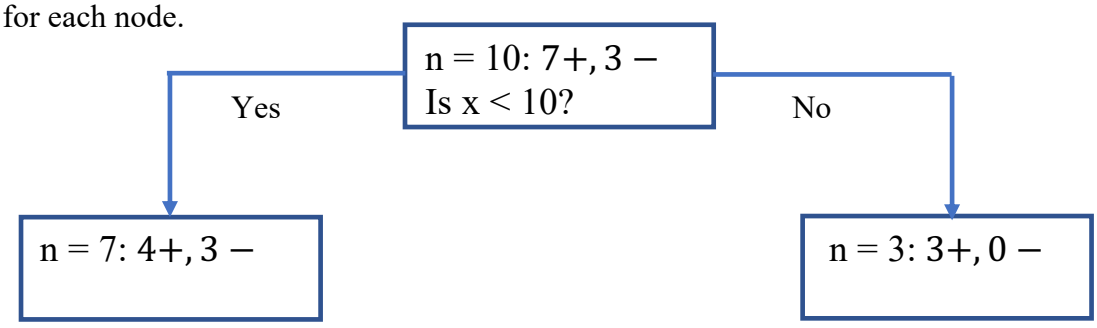
NOTE: The task of growing a classification tree is quite similar to the task of growing a regression tree. Unfortunately, in the classification setting, RSS cannot be used as a criterion for making the binary splits.

Definition: Let $\hat{p}_{mk}$ represent the proportion of training observations in the m^{th} region that are from the k^{th} class. The Gini Index is defined as

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

which is a measure of total variance across the K classes. It is not hard to see that the Gini index takes on a small value if all of the $\hat{p}_{mk}{}'s$ are close to zero or one. For this reason, the Gini index is referred to as a measure of node purity—a small value indicates that a node contains predominantly observations from a single class.

EXAMPLE 1: Suppose we are trying to predict a categorical variable that takes on the values Positive (+) and Negative (-) using a single quantitative input variable (x). Given the following tree, calculate the Gini impurity for each node.



EXAMPLE 1: (Continued)

In each case, we have $K = 2$ because there are 2 possible classes (+ or –). For the left node (Yes, $x < 10$): We have

$$G = \frac{4}{7} * \left(1 - \frac{4}{7}\right) + \frac{3}{7} * \left(1 - \frac{3}{7}\right) = \frac{24}{49} \approx 0.4898$$

For the right node (No, $x < 10$): We have:

$$G = \frac{3}{3} * \left(1 - \frac{3}{3}\right) + \frac{0}{3} * \left(1 - \frac{0}{3}\right) = 0$$

Notice that when a node is pure (all observations are in the same class), the Gini impurity is 0. Finally, we can calculate the overall impurity of the tree by calculating a weighted average of the terminal node impurities:

$$\frac{7}{10} * \left(\frac{24}{49}\right) + \frac{3}{10} * (0) = \frac{24}{70} \approx 0.3429$$

Definition: An alternative to the Gini Index is the entropy. The entropy is given by

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

which takes on a small value if all of the $\hat{p}_{mk}'s$ are close to zero or one. Therefore, like the Gini index, the entropy will take on a small value if the mth node is pure.

EXAMPLE 2: Use L12_titanic3.csv. The dataset contains the following information:

Variable Name	Variable Meaning	Notes
Survived	Whether the passenger survived	Values include 'Yes' and 'No'
PClass	Purchased ticket class	1 = First class 2 = Second class 3 = Third class
Sex	Passenger's sex	Values include 'male' and 'female'
Age	Passenger's age	
Siblings	Number of siblings or spouses aboard	
Parch	Number of parents or children aboard	
Fare	Passenger's fare	

Build a classification tree for predicting whether the person survived based on the remaining variables in the dataset. Use an 80/20 training/testing split based on a seed of 123. Build the tree for the train, create a plot depicting the tree, and obtain predictions for the test. Report the confusion matrix for the testing data and the overall accuracy of the method. NOTE: Be sure you know how to find both predicted class probabilities and predicted classes.

Training/Testing Split

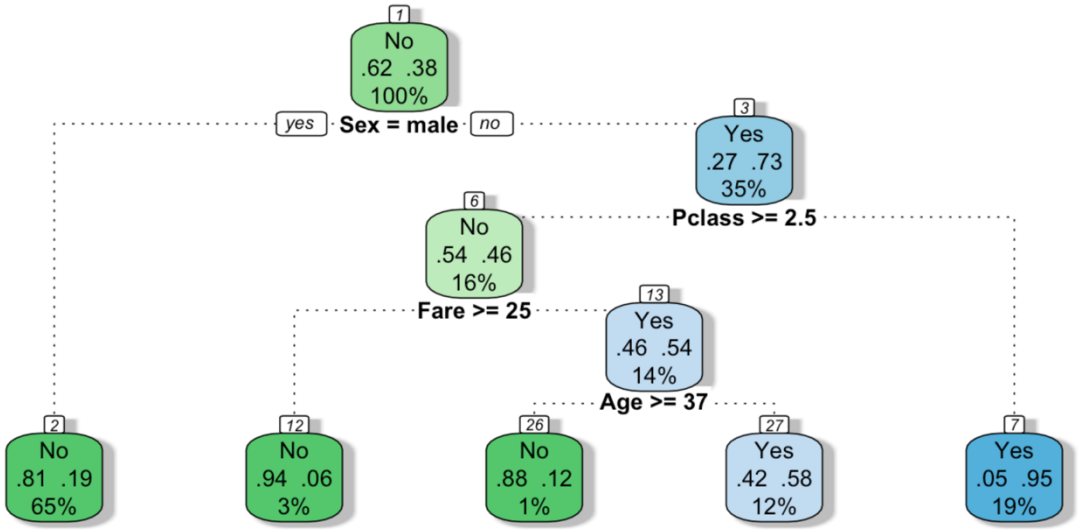
```
set.seed(123)
train_ind <- sample(1:nrow(Titanic), floor(0.8 * nrow(Titanic)))
set.seed(NULL)

Train <- Titanic[train_ind, ]
Test <- Titanic[-train_ind, ]
```

Build Tree on Train

```
regTree <- rpart(Survived ~ ., method = "class", data = Train)
#Note if Survived was 0/1 indicator, convert Survived to a factor to ensure Classification
#Recall: method = "anova" was used for a regression tree
summary(regTree)
```

Create Plot

```
#requires rattle package
fancyRpartPlot(regTree, cex = .8)
```



Obtain Predicted Probabilities

```
#Predicted Probabilities
pred_prob <- predict(regTree, newdata = Test, type = "prob")
#Returns a matrix with 1 row per obs in Test, and 1 column per response class

head(pred_prob)
```

```
##            No       Yes
## 1  0.80827887 0.1917211
## 3  0.42045455 0.5795455
## 7  0.80827887 0.1917211
## 12 0.05147059 0.9485294
## 15 0.42045455 0.5795455
## 18 0.80827887 0.1917211
```

Obtain Predicted classes

```
#Predicted Classes
pred_surv <- predict(regTree, newdata = Test, type = "class")
head(pred_surv)
```

```
##   1   3   7  12  15  18
##  No Yes  No Yes Yes  No
## Levels: No Yes
```

NOTE: To obtain predictions for a regression tree, you can omit the `type =` specification.

Obtain Confusion matrix and calculate accuracy

```
#Confusion Matrix
table(pred_surv, Test$Survived)
```

```
##
## pred_surv No Yes
##       No  99  22
##       Yes  7  50
```

```
#Calculate accuracy
mean(pred_surv == Test$Survived)
```

```
## [1] 0.8370787
```

NOTE: For a binary response, we could use the predicted probabilities to generate an ROC curve and/or find AUC.

NOTE: For a regression tree, we could use the predicted responses to calculate MSE (or RMSE) for the test set.

IDEA: We can obtain an overall summary of the importance of each predictor using the variable importance. The variable importance indicates the improvement (in RSS for regression trees or Gini Index in classification trees) achieved by splitting on a variable. The higher the variable importance, the more important the predictor variable.

NOTE: The summary(rpart.object) function produces a scaled version of the variable importances so that the variable importances sum to 1.

EXAMPLE 3: For the L12_titanic3.csv data, which three variables were the most important predictors?

```
summary(regTree)
```

```
## Call:
## rpart(formula = Survived ~ ., data = Train, method = "class")
##   n= 709
##
##           CP nsplit rel error    xerror       xstd
## 1 0.42222222      0 1.0000000 1.0000000 0.04788806
## 2 0.02962963      1 0.5777778 0.5777778 0.04085432
## 3 0.02222222      3 0.5185185 0.5888889 0.04113325
## 4 0.01000000      4 0.4962963 0.5555556 0.04027770
##
## Variable importance
##      Sex     Fare   Pclass      Age    Parch Siblings
##       50       16       15        7        7        5
##
## Node number 1: 709 observations,    complexity param=0.4222222
##   predicted class=No   expected loss=0.3808181  P(node) =1
##     class counts:   439   270
##    probabilities: 0.619 0.381
##   left son=2 (459 obs) right son=3 (250 obs)
##   Primary splits:
##       Sex      splits as  RL,           improve=93.093170, (0 missing)
##       Pclass   < 2.5      to the right, improve=40.918900, (0 missing)
##       Fare     < 10.48125 to the left,  improve=38.624580, (0 missing)
##       Parch    < 0.5      to the left,  improve= 7.465376, (0 missing)
##       Siblings < 0.5      to the left,  improve= 6.858455, (0 missing)
##   Surrogate splits:
##       Parch    < 0.5      to the left   agree=0.681, adj=0.096, (0 split)
```