Tree Based Methods: Classification Trees

Rebecca C. Steorts, Duke University

STA 325, Chapter 8 ISL

Agenda

- What are tree based methods?
- Regression trees
- Classification trees

Regression versus Classification trees

Classification trees are similar to regression trees, except that it is used to predict a **qualitative response** rather than a **quantitative one**.

For a regression tree, the prediction response for an observation is given by the mean response of the training observations that belong to the same terminal node.

In contract, 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.

Interpretation of classification trees

We are usually interterested in

- Interpreting the results of the classification tree.
- The class prediction corresponding to a particular terminal node region and
- the class proportions among the training observations that fall into that region.

How to grow a classification tree?

- Growing a classification tree is very similar to that of a regression tree.
- We use a recursive binary split to grow the tree, however in the classification setting, the RSS cannot be used as a criterion for making splits.
- A natural alternative to the RSS is the classification error rate.

Classification error rate

The classification error rate is simply the fraction of the training observations in that region that do not belong to the most common class:

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

where $\hat{p}_{mk}$ represents the proportion of training observations in the region m that are from class k.

It turns out that the classification error is not sufficiently sensitive for tree-growing and two other measures are preferable (Gini-index and cross-entropy).

The Gini index

Recall that $\hat{p}_{mk}$ represents the proportion of training observations in the region m that are from class k.

The Gini index is defined by

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{2}$$

which is a measure of the total variance across the K classes.

The Gini index

The Gini index takes on a small value if all of the $\hat{p}_{mk}$ are close to 0 or 1.

Because of this, the Gini index is called a measure of node purity.

For example, a small value of $\hat{p}_{mk}$ indicates indicates that a node contains predominantly observations from a single class.

The cross entropy

An alternative to the Gini index is the cross entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk}) \tag{3}$$

Since

$$0 \leq \hat{p}_{mk} \leq 1$$

this implies that

$$0 \leq -\hat{p}_{mk} \log(\hat{p}_{mk}).$$

Exercise: The cross entropy will take on a value near 0 if the $\hat{p}_{mk}$'s are all near 0 or 1.

Gini index, cross entropy, and classification error rate

Like the Gini index, the cross-entropy will take on a small value if node m is pure.

In fact, it turns out that the Gini index and the cross-entropy are quite similar numerically.

When building a classification tree, either the Gini index or the cross- entropy are typically used to evaluate the quality of a particular split, since these two approaches are more sensitive to node purity than is the classification error rate.

Any of these three approaches might be used when pruning the tree, but the classification error rate is preferable if prediction accuracy of the final pruned tree is the goal.

Deviance

For classification trees, the deviance is given by the summary() function and can be calculated via

$$-2 \sum_m \sum_k n_{mk} \log(\hat{p_{mk}}) \tag{4}$$

where n_{mk} is the number of observations in the mth terminal node that belongs to class k.

A small deviance indicates a tree that provides a good fit to the (training) data.

The residual mean deviance reported is simply the deviance divided by $n - |T_o|$.

Application to Carseats Dataset

We use regression and classification trees to analyze the Carseats data set.

In the Carseats data, Sales is a continuous variable, and so we begin by recording it as a binary variable.

Task 1

1. First use the ifelse() function to create a variable, called High, which takes on a value of Yes if the Sales variable exceeds 8, and takes on a value of No otherwise. This creates a binary variable.
2. Next, use a data.frame() to merge High with the rest of the Carseats data.
3. Finally, use the tree() function to fit a classification tree in order to predict high using all variables but Sales.

Solution Task 1 (a,b,c)

```r
library(ISLR)
library(tree)
attach(Carseats)
# creating a binary variable
High <- ifelse(Sales <= 8, "No", "Yes")
# merge High with the rest of the Carseats
Carseats <- data.frame(Carseats, High)
# fit a regression tree
tree.carseats <- tree(High ~. - Sales, Carseats)
```

Task 2

How does the tree fit? Plot your tree and explain your explaination.

Solution Task 2

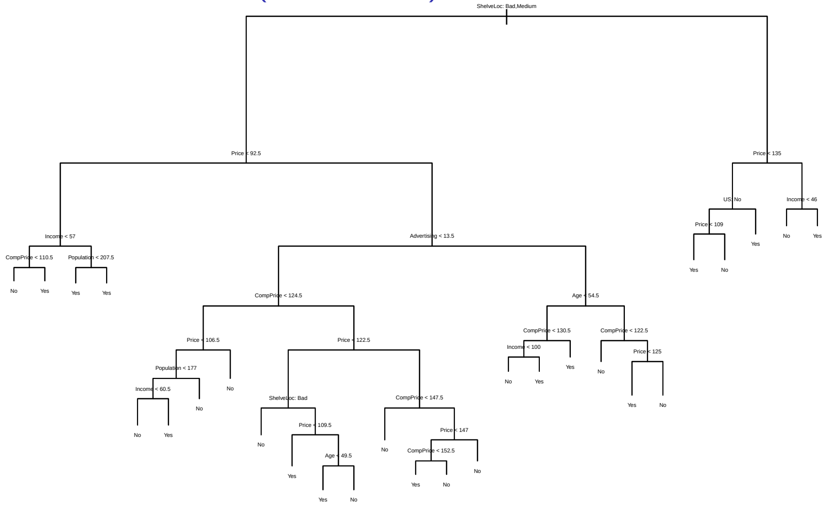
In order to answer Task 2, we will look at the summary of our tree and also plot the tree. In general, we have just built the tree, so remember that we will need to prune it back!

Solution Task 2 (continued)

```r
summary(tree.carseats)
```

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

Solution to Task 2 (Continued)



The most important indicator of Sales appears to be shelving location, since the first branch differentiates Good locations from Bad and Medium locations.

Task 3

1. What is the training error?
2. What is the residual mean error or deviance?
3. What must we do to properly evaluate the performance of a classification tree on the data? (Hint: use the predict function and be sure to explain any results that you obtain). Note: in this step break the data into training and testing data. Don't prune the data yet!

Solution to Task 3

Task 4

Given Task 3, now prune the tree to see if you get improved results. Note: use the argument FUN=prune.misclass 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 the deviance.

Solution to Task 4