DS-6030 Homework Module 7

Tom Lever

07/08/2023

DS 6030 | Spring 2023 | University of Virginia

8. In the lab, a classification tree was applied to the Carseats data set after converting Sales into a qualitative response variable.

   Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

   (a) Split the data set into a training set and a test set.

```
set.seed(1)
library(ISLR2)
library(TomLeversRPackage)
training_and_testing_data <- split_data_set_into_training_and_testing_data(
    Carseats,
    proportion_of_training_data = 0.9
)
training_data <- training_and_testing_data$training_data
testing_data <- training_and_testing_data$testing_data
head(training_data, n = 2)
```

```
#     Sales CompPrice Income Advertising Population Price ShelveLoc Age Education
# 324 10.36       107    105          18        428   103    Medium  34        12
# 167  6.71       119     67          17        151   137    Medium  55        11
#     Urban  US
# 324   Yes Yes
# 167   Yes Yes
```

```
head(testing_data, n = 2)
```

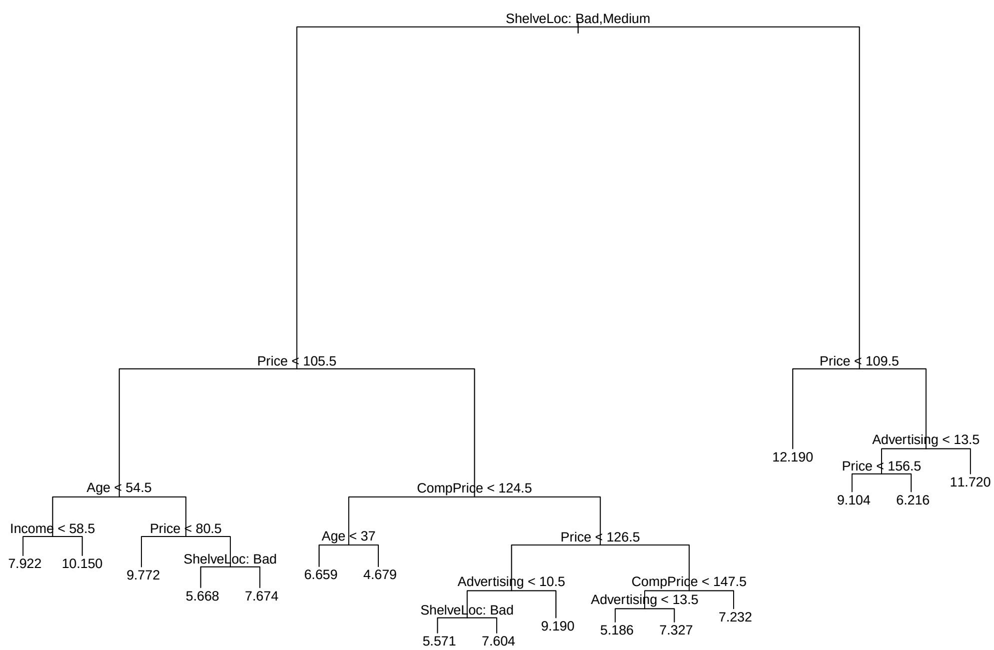
```
#     Sales CompPrice Income Advertising Population Price ShelveLoc Age Education
# 8   11.85       136     81          15        425   120      Good  67        10
# 123  6.88       119    100           5         45   108    Medium  75        10
#     Urban  US
# 8     Yes Yes
# 123   Yes Yes
```

   (b) Fit a regression tree to the training set. Plot the tree, and interpret the results. What test MSE do you obtain?

```
library(tree)
full_tree <- tree(Sales ~ ., data = training_data)
summary(full_tree)
```

```
#
# Regression tree:
# tree(formula = Sales ~ ., data = training_data)
# Variables actually used in tree construction:
# [1] "ShelveLoc"   "Price"       "Age"         "Income"      "CompPrice"
# [6] "Advertising"
# Number of terminal nodes:  17
# Residual mean deviance:  2.653 = 910.1 / 343
# Distribution of residuals:
#     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
# -5.18600 -1.09000  0.05305  0.00000  1.08300  4.63100
```

```r
plot(full_tree)
text(full_tree, pretty = 0)
```




```r
vector_of_predicted_sales <- predict(full_tree, newdata = testing_data)
vector_of_actual_sales <- testing_data$Sales
calculate_mean_squared_error(vector_of_predicted_sales, vector_of_actual_sales)
```

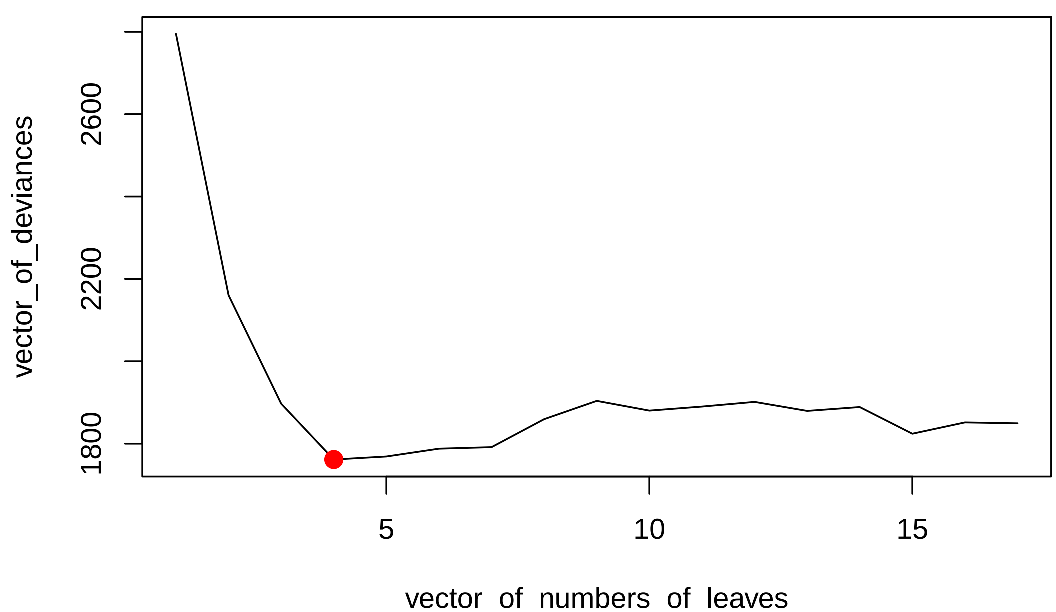
```
# [1] 4.896065
```

When shelf location is good and price is less than 109.5 monetary units, our tree predicts that 12.190 thousand child car seats will be sold at each location in each time period.

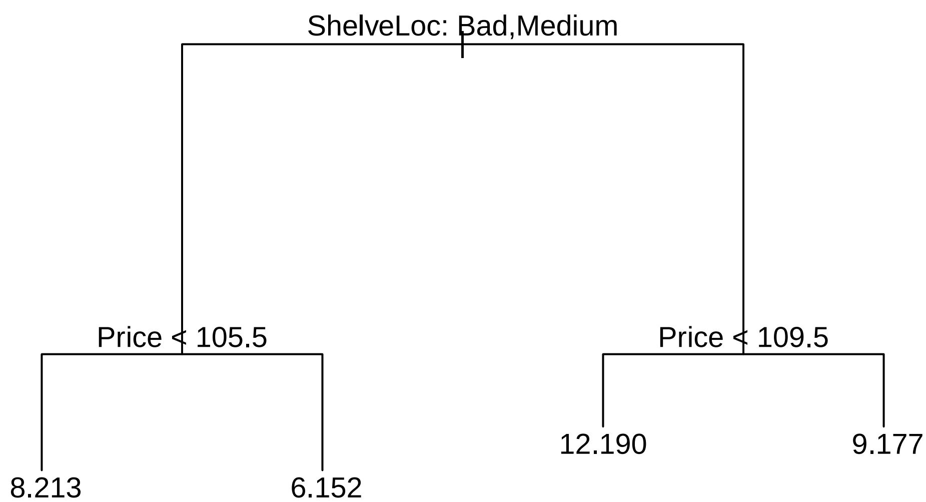
The test Mean Squared Error of our tree when predicting sales is 4.896 $thousand^2$.

(c) Use cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?

```r
object_of_types_prune_and_tree_sequence <- cv.tree(full_tree)
vector_of_numbers_of_leaves <- object_of_types_prune_and_tree_sequence$size
vector_of_deviances <- object_of_types_prune_and_tree_sequence$dev
plot(vector_of_numbers_of_leaves, vector_of_deviances, type = "l")
index_of_minimum_deviance <- which.min(vector_of_deviances)
optimal_number_of_leaves <-
    vector_of_numbers_of_leaves[index_of_minimum_deviance]
minimum_deviance <- min(vector_of_deviances)
points(
    optimal_number_of_leaves,
    minimum_deviance,
    col = "red",
    cex = 2,
    pch = 20
)
```




```r
pruned_tree <- prune.tree(full_tree, best = optimal_number_of_leaves)
plot(pruned_tree)
text(pruned_tree, pretty = 0)
```




```
vector_of_predicted_sales <- predict(pruned_tree, newdata = testing_data)
calculate_mean_squared_error(vector_of_predicted_sales, vector_of_actual_sales)
```

```
# [1] 6.785063
```

The test Mean Squared Error for the pruned tree is greater and less desirable than the Mean Squared Error for the full tree.

(d) Use the bagging approach in order to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important.

Per *An Introduction to Statistical Learning* (Second Edition), bagging "is simply a special case of a random forest with [the number of variables randomly sampled as candidates at each split] $m = p$ the number of predictors."

```
library(randomForest)
```

```
# randomForest 4.7-1.1
```

```
# Type rfNews() to see new features/changes/bug fixes.
```

```
index_of_column_Sales <-
    get_index_of_column_of_data_frame(training_data, "Sales")
data_frame_of_predictors <- training_data[, -index_of_column_Sales]
data_frame_of_sales <- training_data[, index_of_column_Sales]
number_of_predictors <- ncol(data_frame_of_predictors)
BAg <- randomForest(
    x = data_frame_of_predictors,
    y = data_frame_of_sales,
    mtry = number_of_predictors,
    importance = TRUE
```

```
)
vector_of_predicted_sales <- predict(BAg, newdata = testing_data)
calculate_mean_squared_error(vector_of_predicted_sales, vector_of_actual_sales)
```

```
# [1] 2.912954
```

```
importance(BAg)
```

```
#               %IncMSE IncNodePurity
# CompPrice   38.593171     289.70157
# Income      14.313945     170.07098
# Advertising 25.822124     210.16247
# Population  -2.211183      81.20849
# Price       79.323197     766.34347
# ShelveLoc   81.267672     818.13437
# Age         25.716498     270.77786
# Education    2.885808      77.40764
# Urban       -1.691179      11.68266
# US           2.270115      13.65068
```

The test Mean Squared Error for our bootstrap aggregation (BAg) is 2.974, which is 0.607 of the MSE for our full tree and 0.438 of the MSE for our pruned tree.

According to In a random forest, is larger %IncMSE better or worse?, "%IncMSE is the most robust and informative measure. IT is the increase in mse of predictions(estimated with out-of-bag-CV) as a result of variable j being permuted(values randomly shuffled)... the higher the number, the more important."

is highest for *Price* followed by *ShelveLoc*; *Price* and *ShelveLoc* are the two most important variables.

(e) Use random forests to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important. Describe the effect of m, the number of variables considered at each split, on the error rate obtained.

(f) Now analyze the data using BART, and report your results. (skip this exercise)

9. This problem involves the OJ data set which is part of the ISLR package.

(a) Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.

(b) Fit a tree to the training data, with Purchase as the response and the other variables as predictors. Use the `summary()` function to produce summary statistics about the tree, and describe the results obtained. What is the training error rate? How many terminal nodes does the tree have?

(c) Type in the name of the tree object in order to get a detailed text output. Pick one of the terminal nodes, and interpret the information displayed.

(d) Create a plot of the tree, and interpret the results.

(e) Predict the response on the test data, and produce a confusion matrix comparing the test labels to the predicted test labels. What is the test error rate?

(f) Apply the `cv.tree()` function to the training set in order to determine the optimal tree size.

(g) Produce a plot with tree size on the x-axis and cross-validated classification error rate on the y-axis.

(h) Which tree size corresponds to the lowest cross-validated classification error rate?

(i) Produce a pruned tree corresponding to the optimal tree size obtained using cross-validation. If cross-validation does not lead to selection of a pruned tree, then create a pruned tree with five terminal nodes.

(j) Compare the training error rates between the pruned and unpruned trees. Which is higher?

(k) Compare the test error rates between the pruned and unpruned trees. Which is higher?