FIN 510 Big Data Analytics in Finance

Lab 14: Regression Trees

Due on 10/16/2021

Predicting Boston Housing Prices

The file BostonHousing.csv contains information collected by the US Bureau of the Census concerning housing in Boston, Massachusetts. The dataset includes information on 506 census housing tracts in the Boston area. The dataset contains 13 predictors, and the response is the median housing price (MEDV). The following table describes each of the predictors and the response.

DESCRIPTION OF VARIABLES FOR BOSTON HOUSING EXAMPLE	
CRIM	Per capita crime rate by town
ZN	Proportion of residential land zoned for lots over 25,000 ft^2
INDUS	Proportion of nonretail business acres per town
CHAS	Charles River dummy variable (=1 if tract bounds river; =0 otherwise)
NOX	Nitric oxide concentration (parts per 10 million)
RM	Average number of rooms per dwelling
AGE	Proportion of owner-occupied units built prior to 1940
DIS	Weighted distances to five Boston employment centers
RAD	Index of accessibility to radial highways
TAX	Full-value property tax rate per $10,000
PTRATIO	Pupil/teacher ratio by town
LSTAT	Percentage lower status of the population
MEDEV	Median value of owner-occupied homes in $1000s

0) Load the packages

Use library() to load rpart and rpart.plot.

1) Create a data frame

Load the data with read.csv(), remove the variable named CAT..MEDV (column 14), and save the result in a data frame named housing.df.

Use head() and names() to return the first six rows and column names.

2) Data partition

Partition the data into training (60%) and test (40%) sets: use set.seed(1) to set the random seed and sample() to take a sample of row numbers for the training set. Save a sample of row numbers, the training set, and the testing set as train.index, train.df and test.df, respectively.

Hint: dim(housing.df)[1] returns the length of the rows in the data frame, 0.6* dim(housing.df)[1] specifies the number of rows to select for the training set, and c(1:dim(housing.df)[1]) represents row numbers.

3) Fit a shallow regression tree

To predict the median housing price, fit a shallow regression tree with all the predictors in the training set using rpart() with method="anova". Set the smallest value of the complexity parameter to 0.5 (cp=0.5). Save the regression tree as rt.shallow.

Plot the shallow tree using prp(). Set type to 1 to label all nodes and set extra to 1 to display the number of observations that fall in the node.

According to the shallow regression tree, use predict() with type = "vector“ to compute predicted housing prices for records in the test set. Save the predicted prices as rt.shallow.pred and return the first six values using head().

Evaluate the model performance by computing the mean squared error (MSE) in the test set.

4) Fit a deeper regression tree

To predict the median housing price, fit a deeper regression tree with all the predictors in the training set using rpart() with method="anova". Set the smallest value of the complexity parameter to 0.01 (cp=0.01). Save the regression tree as rt.deep.

Plot the deeper tree using prp(). Set type to 1 to label all nodes and set extra to 1 to display the number of observations that fall in the node.

According to the deeper regression tree, use predict() with type = "vector“ to compute predicted housing prices for records in the test set. Save the predicted prices as rt.deep.pred and return the first six values using head().

Evaluate the model performance by computing the mean squared error (MSE) in the test set.

5) Prune the regression tree

Use set.seed(1) to set the random seed such that the result in question 6 can be reproduced.

To identify the complexity parameter value at which the lowest cross-validated prediction error is achieved, use rpart() to fit a regression tree with 5-fold cross validation on the training set.

Hint: fit a regression tree with all the predictors in the training set using rpart() with method="anova". Set the smallest value of the complexity parameter to 0.001 (cp=0.001). Use xval=5 to specify 5-fold cross validation. Save the regression tree as cv.rt.

Use cv.rt$cptable to display various complexity parameter values and their cross-validated errors.

Return the cp value that corresponds to the lowest cross-validated error (xerror).

Hint: which.min(cv.rt$cptable[,"xerror"]) returns the index of the row that contains the minimum cross-validated error. cv.rt$cptable[which.min(cv.rt$cptable[,"xerror"]),"CP"] subsets the cp table by the row index and the column name.

6) Identify the best-pruned regression tree

Find the pruned tree using prune(). The first parameter in the function is cv.rt and the second parameter is the cp value that yields the lowest cross-validated error computed in question 5. Save the pruned tree as rt.pruned.

Plot the pruned tree using prp(). Set type to 1 to label all nodes and set extra to 1 to display the number of observations that fall in the node.

According to the pruned regression tree, use predict() with type = "vector“ to compute predicted housing prices for records in the test set. Save the predicted prices as rt.pruned.pred and return the first six values using head().

Evaluate the model performance by computing the mean squared error (MSE) in the test set.