FIN 510 Big Data Analytics in Finance

Lab 10: Variable Selection

Due on 10/02/2021

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 12 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 package

Use library() to load leaps.

1) Create a data frame

Load the data, 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) Fit a multiple linear regression model

Using housing.df, fit a multiple linear regression model to the median house price (MEDV) as a function of CRIM, CHAS, and RM. Save the result as lm.

Use summary() to print the summary table of lm.

3) Predict for a new sample

Use data.frame(CHAS=, CRIM=, RM=) to create a data frame named new and store the information regarding a tract in the Boston area that does not bound to the Charles river (CHAS=0), has a crime rate of 0.1, and where the average number of rooms per house is 6.

According to the regression model that was estimated in question 2 (lm), what is the predicted median house price for this new tract?

Hint: use predict(lm, new) to make a prediction for the tract in the data frame named new according to model lm.

4) 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.

5) Exhaustive search

To return best models of all sizes, use regsubsets() to apply an exhaustive search on training data with the 12 predictors. Specify parameters nbest=1, nvmax=12, and method= "exhaustive" in the function. Save the result as search.

Use summary(search) to return an object with elements and save the object as sum. Use sum$which and sum$adjr2 to return a logical matrix indicating which elements are in each model and adjusted r-squared of each model.

Hint: dim(train.df)[2] returns the length of the columns in the data frame, which is 13. Since there are 12 predictors, the maximum size of subsets to examine, nvmax, is 12. Column 13 is the response variable, MEDV.

6) Model with the highest adjusted R-squared

Use order() with decreasing=T to sort the adjusted r-squared of the 12 best models (sum$adjr2) in a descending order. The first number in the output is 10, which means that the model with 10 predictors (10th model) has the highest adjusted r-squared.

Find the names of the predictors in the 10th model, which has the highest adjusted r-squared, and save these names of predictors as selected.vars.

Hint: sum$which[10,] returns the a logical vector indicating whether a predictor is in the 10th model or not, and names(train.df)[sum$which[10,]] returns the names of predictors in the 10th model.

Fit a linear regression model to the median house price (MEDV) as a function of the predictors identified in the 10th model (selected.vars) using the training set. Save the result as lm.search.

Hint: Specify parameter data = train.df[,selected.vars] in lm() and use . after ~ to include all the columns in train.df[,selected.vars] as predictors.

According to the regression model estimated in question 6 (lm.search), calculate the predicted median house prices using predict() and the mean squared error for tracts in the test set.

7) Backward elimination

Using the training set, fit a linear regression model to the median house price (MEDV) as a function of all 12 predictors. Save the full model as lm.full.

Use step() to run backward elimination on the training set, and specify direction= "backward" in the function. Save the selected model from backward elimination as lm.step.backward.

Use summary() to print the summary table of lm.step.backward.

According to the best model from backward elimination (lm.step.backward), calculate the predicted median house prices using predict() and the mean squared error for tracts in the test set.

8) Forward selection

Using the training set, fit a linear regression model to the median house price (MEDV) as a function of no variables. Save the intercept only model as lm.null.

Use step() to run forward selection on the training set, and specify scope and direction parameters in the function. Save the selected model from forward selection as lm.step.forward.

Use summary() to print the summary table of lm.step.forward.

According to the best model from forward selection (lm.step.forward), calculate the predicted median house prices using predict() and the mean squared error for tracts in the test set.

9) Stepwise regression

Use step() to run stepwise regression on the training set, and specify direction= "both" in the function. Save the selected model from stepwise regression as lm.step.both.

Use summary() to print the summary table of lm.step.both.

According to the best model from stepwise regression (lm.step.both), calculate the predicted median house prices using predict() and the mean squared error for tracts in the test set.