Report

After I import the train and test data set, I add a new Sale_Price column to test data set with NA values. Then I combined the two data sets. For the new data set, I checked all the categorical variables and combined those levels with few observations. This will prevent test data set has unseen levels in train data set. Then, I split the temporal whole data into train and test and get rid of the Sale_Price. I keep the PID of test data for future use and delete the PID in both data sets. I drop the variables which have dominating level and drop longitude and latitude. Do the log transform on Sale_Price of train data.

I split the train and test into categorical part and numerical part separately. Because xgboost only accepts numeric input, I did one hot encoding on all the categorical variables. At the same time, all the numeric variables have been centered and scaled with preProcess in caret. Then combine categorical part and numerical part.

The first model is gradient boosting tree model.

I use grid search based on caret to find the best combination of those parameters including eta, gamma, min_child_weight, subsample, max_depth. After I find the best set of parameters. I transformed the train and test data into matrix specifically used for xgboost package.

Xgb.cv is used to find the optimal nrounds. The final parameters are eta = 0.01, gamma = 0, max_depth = 5, min_child_weight = 5, subsample = 0.8, colsample_bytree = 1, nrounds = 2000.

The performance on the ten data sets are:

0.1111137	0.1289860	0.1395812	0.1295883	0.1023156
0.1257647	0.1168142	0.1066100	0.1284553	0.1158369

The second model is still gradient boosting tree model.

This model is very similar to the previous one. Because I found in the previous model. Overfitting is very obvious. Maybe grid search values don't cover the optimal choise. So I decide to make some adjustments to reduce overfitting. The subsample rate is changed to 0.5.

So the final parameters are eta = 0.01, gamma = 0, max_depth = 5, min_child_weight = 5, subsample = 0.5, colsample_bytree = 1, nrounds = 2000.

The final performance on the ten data sets are:

0.1111239	0.1295171	0.1388068	0.1289133	0.1023381
0.1259361	0.1164595	0.1066658	0.1267004	0.1145584

Compare the results from the two models, there are slightly difference in different data sets. And changing the subsample rate doesn't improve the overfitting problem.

For each iteration in one data set, the first model needs 18 seconds. The second model needs 22 seconds.

The computer system: Macbook Pro, 2.3GHz, i5, 8GB memory.