Predicting Iowa House Prices

Project Description

Code

Qingchuan Lyu

November, 2019

This Kaggle project predicted house prices in Ames, Iowa with 79 features (2006-2010). The training set had 1460 observations and the test set had 1459 observations. To clean data, I corrected two variables in the test set where the year sold was less than the year built. Then, I drew scatter plots to catch potential outliers, drew distribution plots to study the scale of labels, and performed a log transform as it's right-skewed. Missing values existed in the training dataset and the test dataset. I fill in the missing values according to the summary statistics (i.e., median, mean or mode) of features in the training set, or None/0 as indicated by the data description.

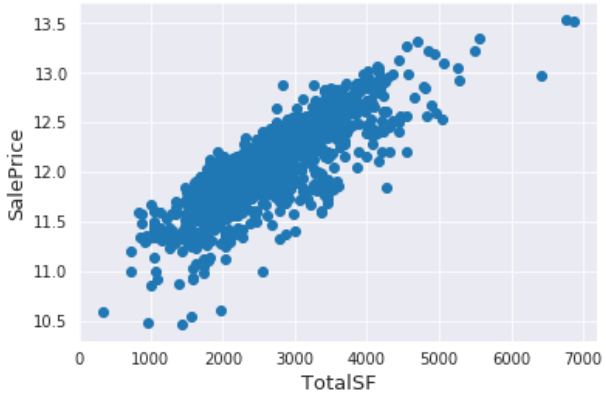

Figure 1: Scatter Plot--x axis total areas, y axis sale prices.

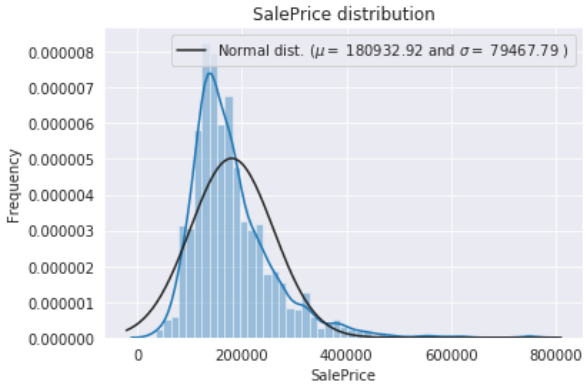

Figure 2: Distribution of "SalePrice" compared to Normal distribution.

Then, I engineered categorical features and numerical features. A few categorical variables were coded as numerical in the original training dataset. I transferred them back to be categorical. I label encoded categorical variables for regression purposes. Since the orders generated by label-encoders didn't make sense for a few variables, such as the names of neighborhoods, I applied one-hot encoders to label-encoded categorical variables (e.g., [0, 1, 0, 0, …, 0]). However, one-hot encoders also caused the elimination of the natural orders of categorical variables, such as "excellent," "good" and "poor." If I had more time, a better approach would be using label encoders on ordered variables and one-hot encoders on unordered variables.

As for numerical variables, I applied box-cox transformation to skewed features so that they were more normally distributed. With 220 features, I performed PCA and selected 143 principal components with 95% of variance retained from the original training set.

I used linear regression as the starting point of modeling. As the size of the training set was fairly small, I created 10 cross-validation folds. The mean squared error was 0.2776 on the

test set. To reduce overfitting, I used Elastic Net, the combination of Ridge and Lasso, and obtained MSE less than Ridge or Lasso. Random Forest performed better than a single decision tree, however, its accuracy is a little worse than Elastic Net (0.1876 > 0.1193). The XGB family performed the worst with an error of 8.512. Accordingly, I assigned the most weights on Elastic Net and the least weight on XGBoost to build a weighted average meta model. The final MSE is better than the MSE obtained from any of three models used, as shown below.

Models	Linear Reg	Elastic Net	Random Forest	XGBoost	Meta Model
Score: MSE	.2776	.1193	.1883	8.512	.11549(test dataset)
Additional information	Largest error	Better than Lasso or Ridge	Better than a decision tree	Long runtime	.8*Elastic Net +.15*RDF +.05*XGB