Housing Price Prediction Models

Siddhanta Phuyal, Yassine El Maazouzi
DePauw University
Greencastle, IN 46135, U.S.A.

December 12, 2022

Abstract:

This paper aims to analyze the pre-processing techniques and algorithms that have been used to predict housing prices given some explanatory variables. We use the Ames Housing Dataset complied by Dean De Cock, available via Kaggle competition, to pre-process the data, engineer some new variables, build a model, and predict the price of houses. We determine the success of the project by obtaining the score from Kaggle after uploading the predicted housing prices to the website. Kaggle uses a scoring mechanism where a lower score means a better score. We were able to achieve a score of 0.138.

1 Data Description

The Ames Housing Dataset provides 79 different explanatory variables. We found 36 of the explanatory variables contain numerical values and 43 of them contain categorical values. Kaggle provides us with two CSV files: train.csv and test.csv. The train.csv file contains a dataset that would be used to train our model and the test.csv would be used to test the model by sending the predictions to Kaggle. The training dataset contains 1460 rows of data, and the testing dataset contains 1459 rows of data. Kaggle also provides a description file to describe the attributes and their values.

After carefully reading the description file, we found that even though some variables contain numerical values, the values represent categories. For example, for the attribute, MSSubClass, the value 90 means the house is a duplex and the value 190 means the house is a 2-family conversion. Similarly, for the attribute, OverallQual, the value 10 translates to 'Very Excellent' and the value 1 translates to 'Very Poor'.

We also found that for most of the categorical attributes, 'NA' translates to the absence of that feature in the house rather than missing or unknown. For example, for the attribute, BsmtQual, the value, NA, translates to 'No Basement'. Similarly, for the attribute, GarageFinish, the value, NA, translates to 'No Garage'. Python treats these values as a missing values even though these values are not missing and they are still providing some information that could be used in our model.

It is also important to note that there are only 1460 rows of data in the training set for 79 different explanatory variables. Our goal is to build a model that generalizes well, and including all the explanatory variables might confuse the model. We start pre-processing the data keeping these things in our mind.

2 Experiment

2.1 Pre-Processing

First, I use the provided helper functions to find the name of attributes that are numeric and have missing values in the training dataset and testing dataset. In training dataset, 'LotFrontage', 'MasVnrArea', and 'GarageYrBlt' attributes are missing some values. In testing dataset, 'LotFrontage', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'BsmtFullBath', 'BsmtHalfBath', 'GarageYrBlt', 'GarageCars', and 'GarageArea' attributes are missing some values.

For 'LotFrontage' and 'MasVnrArea' attributes, I replace all the missing values in training set and testing set with the corresponding mean values from the training set.

For Garage related attributes, first I find all rows in the training set and testing set that have 'NA' values for 'GarageFinish' attribute. These 'NA' value translates to 'No Garage'. If there is no garage, the

Non-Numeric Attributes	**Related Non-Numeric Attributes**	**NA' translates to:**
Alley		No Alley
BsmtQual	BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2	No Basement
GarageType	GarageFinish, GarageQual, GarageCond	No Garage
PoolQC		No Pool
Fence		No Fence
MiscFeature		None

Table 1: Showing the non-numeric attributes whose NA values mean absence of that feature.

values for 'GarageCars' and 'GarageArea' attributes should be zero and the value for 'GarageYrBlt' will be equal to the year the house is sold. The idea is to transform this variable into a variable that provides the age of garage. Then, the corresponding value would be zero and hence it will have no effect on price of the house.

For Basement related attributes, I use the same procedure to first find the houses that have no basement and then replace all the basement numeric attributes with zero values.

Then, I again check if any numeric attributes are missing values to find none of the numeric attributes are missing values.

After processing the numeric attributes, we begin processing the non-numeric attributes. First, we find the non-numeric attributes that contain missing values for training set and testing set. As we have discussed above, there are many non-numeric attributes that contain 'NA' values which translates to absence of those features in that house. So, we must recode all of those 'NA' values into something else. With the help of data description file, we construct table 1.

We recode all the 'NA' values for non-numeric attributes in the left column and their related non-numeric attributes in the middle column of the table 1 into the corresponding values from the right column in the table 1. For example, If BsmtQual has 'NA' value, it means that the house does not have any basement. Thus, the NA values from BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2 attributes is replaced by 'No Basement'.

We found that if there is no fireplace in the house, the related non-numeric attribute, FireplaceQu, is missing. So, for such cases, we replace the missing values for this attribute by 'No Fireplace'.

The remaining missing values for the non-numeric attributes are replaced by the mode of their respective attributes from the training set. For example, if there are missing values in 'MasVnrType' attribute, the missing values are replaced by the mode of 'MasVnrType' from the training set.

The missing values from the testing set are processed in the same way as the training set. If the mean values from the training set are used to replace the missing values in training set, the same means are used to replace the missing values from the testing set. This confirms uniformity in the processing of dataset.

Then, we use 'One Hot Encoding' method to convert the categorical input variables into dummy variables. The dummy variables are the duplicate variables created for each values within the given attributes. Those variables have only two values: 0 and 1. Presence of the value is represented by 1 and the absence is represented by 0. For example, for sex variable, there are two values: Male and Female. In this case, two dummy variables would be created: sex_male and sex_female. For males, sex_male would have value of 1 and sex_female would have value of 0. In our project, we use get_dummies() method from pandas modules to create dummy variables for the categorical attributes. The method not only creates the dummy variables, but also populates them with the right values for us. We also create dummy variables for 'MSSubClass' and 'OverallQual' because as we explained in the data description section, even though these variables have numerical values, they represent categories.

Finally, we transform some of the old variables into new variables that would produce better results. First, we create 'ageOfHouse' and 'ageAfRemod' which are the variables that represent the age of the house, and the age of the house after remodeling at the time the house was sold respectively. We create these attributes for both training set and testing set. We fill in the 'ageOfHouse' attribute with the difference between the values from 'YrSold' and 'YearBuilt'. Similarly, we fill in the 'ageAfRemod' with the difference between the values from 'YrSold' and 'YearRemodAdd'. Second, we create another attribute called ‘TotFinBsmtSF’ which represents the total surface area of the finished basement. We fill in this attribute with the sum of the total surface areas for different types of finished basement. Third, we create a attribute that represents the total number of bathrooms in the basement: ‘BsmtBath’. The values for this attribute would be equal to the sum of total number of full bathrooms in the basement and half of the total number of half bathrooms in the basement. Similarly, we create an attribute that represents the total number of bathrooms in the house above ground level.

2.2 Algorithms and Hyper-parameterization:

We use Linear Regression model available via ‘sklearn’ module as a foundational model to check what pre-processing techniques would improve the accuracy of the model. Linear Regression Model, a basic and yet very powerful model, attempts to fit a linear equation that would capture the relationship between the input variables and the single output variable. First, the model uses all explanatory variables to predict a value for the output variable given the values for the explanatory variables. Then, the model finds a line of best-fit by minimizing the sum of squares of the deviations of the true output values from the respective predicted output values. Thus, the Linear Regression model tries to find a generalized relationship between the explanatory variables and the output variable.

While the Linear Regression model works best in the cases where there is a linear relationship between the independent and dependent variables, the model might slack when the relation is parabolic or exponential. For example, Mincer (1974) showed that an additional one year of schooling would increase the income of the individual by a multiplicative factor rather than by a constant amount while analyzing the relation between education and income. In such cases, if proper variable transformations are not done, the results from linear regression models can be misleading because the linear regression model assumes a linear relationship between them. Similarly, the presence of outliers can increase the effect of extrapolation on the linear regression models. Extrapolation can be described as estimating the same trend will continue for the new dataset even though the new dataset has been extracted from a completely different scenario. Consider this example where we use the children’ data as a training set to fit the relationship between height and weight. Then we use the same model to predict the height of adults given their weights. The model will fail to correctly predict in such cases because of extrapolation.

Other than linear regression models, we have attempted to use Bayesian Ridge Regressor, a Ridge Regressor, and a Gradient Boosting Regressor models to analyze the pre-processing techniques.

Bayesian Ridge Regression is robust to the problem of insufficient data or poorly distributed data because it fits a linear regression using probability distribution rather than point estimates. Ridge Regression is the extension of linear regression where some regularization are done to penalize the large coefficients. When the relevant independent variables are omitted from the regression models, there is a high chance of upward biases in the estimation of coefficients. In such cases, these regression models decrease the biases by penalizing the large coefficients. Gradient Boosting Regressor is a regression technique where many decision trees are orchestrated together to minimize the loss function. In our project, we have used the squared errors, which is the default option, as the loss function.

3 Results:

We have experimented various pre-processing techniques combined with various models. We summarize the experiments as follows:

1. First, after filling in the missing values and creating the dummy variables for the categorical attributes, we use all the explanatory variables as predictors in our model.

2. We obtain a heatmap showing the correlations among various numerical variables and we drop the variables that have no or very low correlation with the selling price of the house. We engineer new variables as described in the data description section and drop the corresponding old attributes. If more than one attributes are describing the similar feature of the house, we only include one of such attributes. For example, ‘OverallCond’ is describing the overall condition of the house and ‘OverallQual’ is describing the overall quality of the materials used in the house. We only include one of them in the model. If an attribute has same values in more than 80% of the rows in the training set, we drop the attribute from the model. Thus, in this experiment, we only select attributes that fulfill the above requirements.

Predictors	**Models**	**Average R-squared score**
All Original Explanatory Variables	Linear Regression	65.98%
	Bayesian Ridge	75.91%
	Ridge	82.34%
	Gradient Boosting	87.95%
Selected Explanatory Variables	Linear Regression	85.53%
	Bayesian Ridge	76.26%
	Ridge	85.73%
	Gradient Boosting	87.59%
Standardizing All Selected Variables	Linear Regression	85.53%
	Bayesian Ridge	85.17%
	Ridge	85.73%
	Gradient Boosting	87.72%
Taking log of Sale Price	Linear Regression	87.03%
	Bayesian Ridge	86.83%
	Ridge	87.15%
	Gradient Boosting	87.90%

Table 2: Average R-squared values from Regression Models.

3. This builds on top of the second experiment. We standardize all the selected variables in the second experiment including the dependent variable.

4. This builds on top of the second experiment. We modify the dependent variable, which is 'SalePrice', by replacing the values with the natural log of those values.

In this project, we use the scorer R2 ("R Squared") to determine the goodness of the model. The R-squared value represents how much variations in the dependent variable is explained by the independent variables in our model. We summarize the results from each of these experiments in Table 2.

4 Analysis:

When all the original explanatory variables are used in the regression, we find that the average R-squared values for the linear regression model, Ridge regressor, and Gradient Boosting regressor are 65.98%, 82.34%, and 87.95% respectively. We believe that the average R-squared for the linear regression model is significantly lower than Ridge regression because the coefficients in the linear regression model are impacted by the upward biases. The Ridge regression model penalizes the higher coefficients and thus decreases the upward biases in the coefficients. Several attributes were highly correlated, and they were providing similar information. For example, the number of cars in the garage and garage area are highly correlated and they provide information on the size of the garage. Similarly, the overall condition of the house and overall quality of materials used in the house are highly correlated. They provide similar information. When a person sees a house in good condition, it is likely, the person will also give a higher score for the overall quality. However, including both predictors in the model would overfit the model. According to IBM, "Overfitting is a concept in data science, which occurs when a statistical model fits exactly against its training data. When this happens, the algorithm unfortunately cannot perform accurately against unseen data, defeating its purpose." Following Ockham's razor rule, we attempt to generalize our model by only including one predictor among the highly correlated predictors giving similar information about the house. For example, instead of using both overall condition and overall quality, we use one of them as the predictor.

Similarly, we transform multiple attributes into a single attribute to decrease the number of predictors in the model. For example, instead of including both number of full bathrooms and number of half bathrooms in the model, we create a new variable that captures the number of total bathrooms in the house. We also drop the predictors for which the 80% values are same across the rows. Given that a lot of the values are same, the regression model would not be able to correctly predict the effect of an

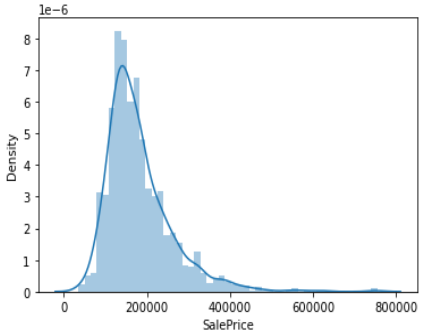

Figure 1: Distribution of Sale Price before log transformation

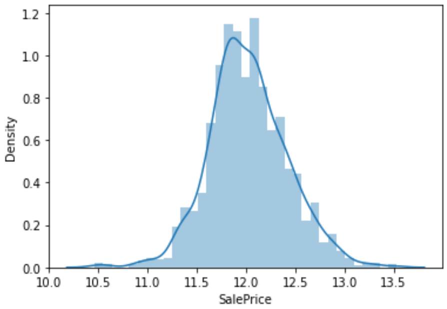

Figure 2: Distribution of Sale Price after log transformation

unseen feature when the test data set is used. This would increase the extrapolation and thus, decrease the accuracy of our model. For example, More than 90% of the values in ’Alley’ attribute are NA which after data processing translates to ’No Alley’. If a model is fit using this data, the model might not accurately predict the effect of having a alley on the price of the house. We exclude such variables from our model.

Thus, after dropping such attributes and selecting only relevant attributes for the model, we observe that the average R-squared for the Linear Regression model has increased to 85.53%. However, we observe that the score for Gradient Boosting Regressor has not changed significantly. The reason for this observation is the Gradient Boosting Regressor has built-in algorithms that automatically drop the irrelevant predictors from the data set.

When two or more independent variables are highly correlated among each other, we say that the data have multicollinearity. The presence of multicollinearity leads to biased estimated coefficients for the correlated variables. Standardizing the variables brings the variables in the same scale and negates the affects of multicollinearity. However, the presence of multicollinearity does not affect the accuracy of the predictions. So, we observe that the standardizing the predictors has not changed the R-squared value for the models.

Finally, the Linear Regression model assumes that there exists a linear relationship between the predictor variables and the dependent variable. However, the relationship could be exponential or parabolic. If the relationship is exponential, the dependent variable changes by a multiplicative factor when the predictor variables change by a unit. We transform the Sale Price of the house by taking natural log of selling price. There are two advantages of using log transformation. First, it linearizes the relationship between the independent variables and dependent variables if their relationship is of exponential nature. Second, it reduces or removes the skewness in our original data.

We observe from Figure 1 and Figure 2 that the distribution of sale price after log transformation has reduced any skewness. This should increase the accuracy of the model. The average R-squared value for the linear regression model has increased to 87.03% after log transformation.

5 Conclusion:

We find that 'Overfitting' the model decreases the average R-squared value for the linear regression model. When we drop highly correlated predictors that provide similar information, the accuracy of the model increases. Similarly, we observe that log transformations can reduce skewness in the data and hence, improve the accuracy of the model. The results from Gradient Boosting regressor are very consistent across all experiments. Finally, our results have shown that understanding the data and pre-processing them accordingly improves the accuracy significantly. We have increased the average R-squared value from 65.98% to 87.03% just from pre-processing the data.

6 References:

1. Mincer, Jacob. "Investment in Human Capital and Personal Income Distribution." Journal of Political Economy, vol. 66, no. 4, 1958, pp. 281–302. JSTOR, http://www.jstor.org/stable/1827422. Accessed 18 Nov. 2022.

2. By: IBM Cloud Education. (n.d.). *What is overfitting?* IBM. Retrieved December 11, 2022, from https://www.ibm.com/cloud/learn/overfitting

3. Seaborn - Linear Relationships. Tutorials Point. (n.d.). Retrieved December 11, 2022, from https://www.tutorialspoint.com/seaborn/seaborn_linear_relationships.htm

4. Linear Models. scikit. (n.d.). Retrieved December 11, 2022, from https://scikit-learn.org/stable/modules/linear_model.html#linear-model