Data Mining: House Price Prediction

Syed Mustafa, Yousaf Khan
Department of Computer Science, DePauw University
Greencastle, IN 46135, U.S.A.
Yousafkhan_2022@depauw.edu
Smustafa_2021@depauw.edu

Abstract

As the demand for housing has risen in the midst of the covid-19 pandemic, it has become increasingly important to accurately price the value of houses. This paper uses Linear Regression and Gradient Boosting Regressor algorithms to predict the Sales Price of houses, given a data set of attributes.
Keywords: Linear Regression, Gradient Boosting Regressor, Correlation

1 Data Description

The data set is comprised of both numeric and non-numeric attributes. These attributes include physical features of the house such as year built, size, number of rooms, number of floors etc that can be used to determine Sales price of houses. The attributes also include other external features such as neighborhood, closeness to highways, slope of property, among others. Some attributes prove as better predictors for Sales Price than others. Some attributes were also interconnected and help us understand the same house feature eg height of basement, condition of basement, total square feet of basement.

2 Experiment

2.1 Pre-Processing

First, we used the 'getNumericAttrs' helper function to identify all of the numeric attributes. These attributes include OverAllQual, YearBuilt, MasVnrArea, GrLivArea. We ran a correlation test between all the numeric variables and Sales Price. We only wanted to keep values that had a high linear relationship with Sales Price: this is signified by a greater magnitude correlation figure. Any feature with a correlation lying in the range -0.2 < correl < +0.3 were thus dropped.

The remaining numeric attributes were standardized to prevent different units of measurement from adversely affecting our prediction and model.

The dataset also consisted of missing values for certain attributes. We used a fillna() for each of these instances, replacing them with appropriate substitute values - we used backward fill in most cases. However, for features with majority missing values, backward fill leads to an inaccurate substitution since many missing values would end up being the same. Thus, for attributes with majority missing values, we classified the values ourselves.

For the 'BsmtCond' variable, we created a pie chart to assess the frequency of values present. We noticed that the entirety of the attribute was dominated by 'TA' and 'Gd' values: this can be seen in the figure below:

Thus, we classified the values keeping TA and Gd distinct, and grouping together the rest. Doing so, we were able to attain a high correlation between the attribute and 'SalePrice'. Thus, we included this as part of our predictors.

Next, we identified that 'BsmtFinType1' and 'BsmtFinType2' are two attributes that measure the same physical quality: rating of basement finished area. Thus, we wanted to analyze if one or the other gives us a better prediction for 'SalePrice'. We also noticed that both attributes have significant missing values, thus we decided to fill them

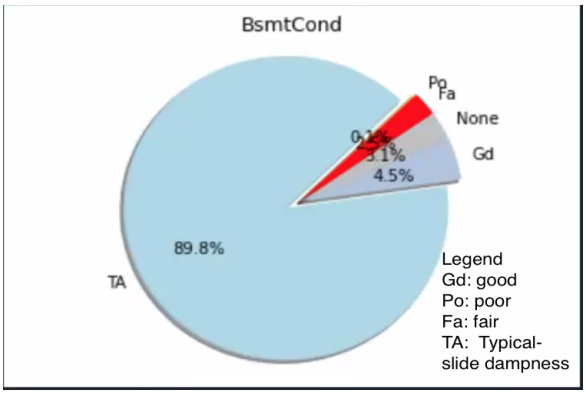

Figure 1: Pie chart showing distribution of the Basement Condition Attribute subfeatures

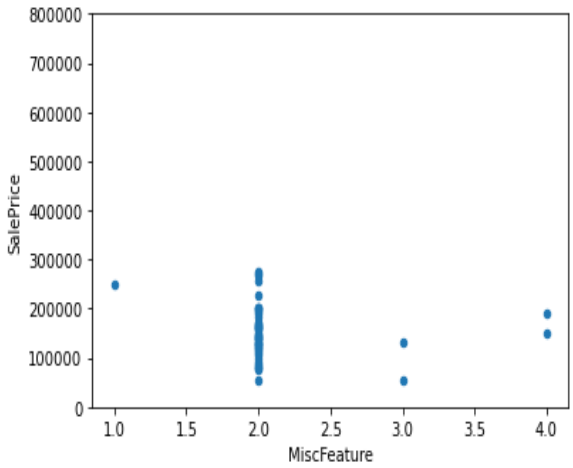

Figure 2: Scatter Plot showing the relationship between 'MiscFeature' and 'SalePrice' attribute

Table 1: Classification of 'MiscFeature' attribute

Value	Classification
TenC	1
Shed	2
Othr	3
Gar2	4
Elev	5

with the mode. We classified both and checked correlation to see if both are relevant to our prediction.

Doing our correlation test, 'BsmtFinType1' and 'BsmtFinType2' yielded a -0.2687 and 0.0415 correlation figure respectively. The second one is lower than our previously established threshold. Therefore, we decided to drop 'BsmtFinType2'.

The 'MiscFeature' attribute talks about miscellaneous features not covered in any other attribute. The possible values for this attribute included relatively unique features such as having an elevator. Such features will likely not be common in most houses. Therefore, we decided to analyze the frequency of values and their correlation with 'SalePrice'. In order to make the plot, we would need to classify the 'MiscFeatures'. We did this in the manner described in Table 1. After classifying, the scatter plot attained can be seen in Figure 2.

As can be deduced from the plot, there are very few values that are not 'Shed'. Thus, we re-adjusted our classification, and assigned 5 to all of the values other than those that were not available. Doing this pre-processing, we checked the correlation between 'SalePrice' and 'MiscFeature' which was much lower than our threshold. We still included this in our predictors and it dropped the accuracy of our prediction. Consequently, we dropped 'MiscFeature' from our analysis.

We applied a similar classification method to the following non-numerical attributes: Fence, MSZoning, BsmtQual, BsmtCond, BsmtFinType 1 and 2, Condition1 and 2, Alley, HouseStyle, Roofstyle, ExterQual, Heating and QC. We dropped the ones that had too poor of a correlation with SalePrice.

For the feature 'YearBuilt' it made more sense to convert it to a format that is more "readable". To do this, we created a new variable called 'YrsOld' and used the following equation to calculate it:

$$YrsOld = 2021 - YearBuilt$$

We plugged the newly created 'YrsOld' attribute into our list of predictors. Additionally, we applied the same methodology to create a 'GrgYrsOld' attribute using the existing 'GarageYrBlt' attribute.

2.2 Algorithm and Hyperparameterization

2.2.1 Algorithm

In our tests, we have relied on multiple algorithms: Ordinary Least Squares Linear Regression, Gradient Descent Boosting Algorithm, Ridge Regression, Lasso Regression, and Bagging meta-estimator. For our models, we used a consistent CV = 10 for all k-fold cross validations. Please find explanations for these algorithms below:

OLS Linear Regressions uses the ordinary least squares to find a hypothesis function that best predicts unseen values of the dependent variable: in this case, 'SalePrice'. The hypothesis function has an erorr surface, and we want to find the inputs that minimizes the error.

Gradient Descent Boosting is a type of machine learning boosting that essentially minimizes the prediction error by looking at the next best possible model. This is because combining the next best possible model with previous models decreases overall prediction error. To do so, it must re-set the target outcomes for successive models. In order to find the lowest point on the error surface, the target outcome for each case is calculated by measuring the change in overall prediction error by that case's prediction. This process is called gradient boosting because target outcomes are determined by the gradient of the error with regard to the prediction. Boosting works well with weak models.

Ridge regression is very similar to the previously discussed OLS Linear Regression. However, it improves on the former because ridge coefficients have a penalty on their size, as opposed to OLS coefficients.

Lasso Regression is a linear model that estimates sparse coefficients. It reduces the number of features that the solution is dependant on, and thus prefers solutions with non-zero coefficients.

Lastly, Bagging meta-estimator is an ensemble method that builds black-box estimators on randomized subsets of the total dataset. After making its predictions on the individuals, it combines them together to give a final prediction. Bagging is ideal for complex models since they reduce overfitting.

2.2.2 Hyperparameterization

The Ordinary Least Squares Linear Regression does not have any hyperparameters; therefore, we cannot conduct hyperparameterization on this algorithm.

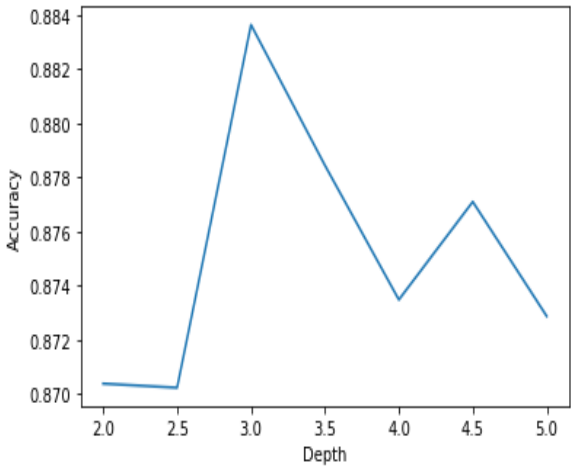

Figure 3: GBR Hyperparameterization

On the other hand, Gradient Descent Boosting Algorithm has many hyperparameters. After a thorough trial and error analysis, we deduced that the 'max_depth' parameter had the most reliable and significant impact on the accuracy of our model. Thus, we defined a special function to tune our model based on this parameter.

We tested a range of values and plotted a graph to represent mean accuracy compared to the different depths in this parameter. This graph can be seen in figure 3.

As is visible, setting the hyperparameter to a value of 3.0 yielded the most accurate predictions at 88.364 percent accuracy.

We conducted a similar analysis on Ridge Regression. This time, we modified the alpha value for Ridge Regression. The produced chart can be seen in Figure 4.

Similarly, we conducted a hyperparameterization of the alpha variable on Lasso. The produced chart can be seen in Figure 5.

2.3 Results

The results of our analysis can be found in table 2. The before numbers are attained by applying the same models to the data before our pre-processing and hyperparameterizing.

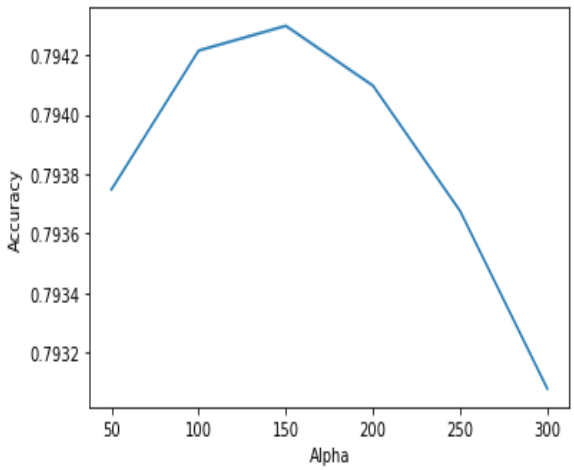

Figure 4: Ridge Regression Hyperparameterization

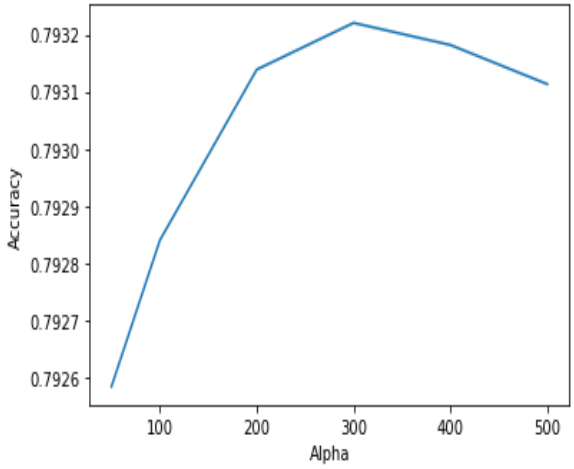

Figure 5: Lasso Hyperparameterization

Table 2: Before and After Analysis

Accuracy	Before	After
Linear Regression	0.564	0.782
GBR	0.584	0.883
Ridge Regression	0.564	0.793
Lasso Regression	0.564	0.792
Bagging	0.247	0.726
Kaggle	0.26	0.164

2.4 Analysis

As can be seen, all of the scores have improved after pre-processing. Additionally, the hyperparameterization has helped us identify the best parameter for GBR. This in turn, helped us improve the accuracy of our predictions. The Gradient Boosting Regression has resulted in the best score and greatest improvement from the original. Gradient Boosting Regression is more accurate than OLS linear regression. We think this is a direct result of the fact that GBR relies on gradient descent, which is more precise than ordinary least square methodology. Due to the same reason, it performs better than many other linear regression algorithms used. In our pre-processing, we chose to drop many attributes from our model's consideration. This indicates that the price of a house is heavily dependant on a few fundamental variables. Consequently, over-analysis of every measurable aspect of a house can mislead pricing estimates. We relied on correlation analysis to identify these fundamental variables, it should be further researched whether better methods exist that can help identify these variables. Additionally, the error in prediction could be attributed to attributes not included in this data set. When looking at new houses, it is likely that consumers consider factors such as noise-level, access to transport e.t.c. Then, it is worth collecting this data, and investigating if the existing error in prediction can be attributed to the lack of such attributes. We were able to use existing attributes to derive new ones that helped with out predictions: it is interesting to see what further derivations we can make with the existing dataset.

3 Conclusion

As shown, the GBR has an 88% accuracy of making the right predictions on unseen problems. It can help us gain insight into how housing prices can fluctuate given variation in different features. Though certain attributes are great predictors of housing price, many attributes that seem important practically do not contribute to a meaningful prediction. To the contrary, they can actually increase error in prediction of price.