Housing Sale Price Prediction Using Machine Learning

Khanh Tran – T00711160
Department of Mathematics and Statistics and Department of Computing Science
Thompson Rivers University
Kamloops, Canada
trank22@mytru.ca

Abstract—The background literature review shows that advances in machine learning algorithms can be used to build accurate and reliable predictive models for housing sale prices. In this study, I develop housing sale price prediction models using several regression machine learning algorithms, including Linear, Ridge, Lasso, Elastic Net, Principal Component, and Random Forest regressions. This study uses the Ames Housing dataset, which includes information about homes in Ames, Iowa, and has 79 explanatory variables and one target variable, sale price. I perform data cleaning and preprocessing, feature engineering using techniques such as log transformation and feature selection, and evaluate the performance of the different regression models. The results show that the Lasso regression model performs the best in predicting housing sale prices, with an R-squared value of 0.904 and an RMSE of 0.129. This study provides valuable insights into the factors that drive housing prices and demonstrates the potential of machine learning techniques in predicting housing values.

Keywords—Housing Price, Machine Learning, Linear Regression, Ridge Regression, Lasso Regression, Elastic Net Regression, Principal Component Regression, Random Forest Regression

1. Introduction

Predicting precise house selling prices is a challenging task since the real estate market is a sophisticated and dynamic industry. However, thanks to advancements in machine learning algorithms and the accessibility of data, in recent years, it is possible now to build accurate and reliable predictive models for housing sale prices.

In this project, I am going to develop accurate and reliable predictive models for housing sale prices using several regression machine learning algorithms. To achieve this goal, I will implement several popular regression algorithms in the field of machine learning that I have learned in the DASC5420 – Theoretical Machine Learning course. These algorithms include Linear, Ridge, Lasso, Elastic Net, Principal Component, and Random Forest regressions. Some of these selected regression algorithms are expected to address the challenge that the Ames Housing dataset has a large number of independent variables [1]. I will then compare their performance in predicting housing sale prices and recommend the most effective algorithms.

Accurately forecasting housing sales has various advantages, including helping purchasers in planning budgets properly, supporting sellers in determining appropriate prices, and assisting financial institutions in making informed financing decisions. In addition, by predicting home sale prices, researchers acquire a better knowledge of housing market patterns and gain insights into the factors that drive housing prices.

2. Background

House price prediction has been an increasingly popular topic in recent years, owing to the significant implications for the property market. Several researchers have proposed and built numerous house price prediction models [4, 5, 6, 7]. These studies concentrate on different aspects of house price prediction, such as data processing, feature engineering, and regression models. Chenchen et al. [4] proposed a house price prediction algorithm in Ames, Iowa, which ranked 35th on the Kaggle.com leaderboard. The authors used data processing, feature engineering, and combination forecasting techniques to achieve good performance with minimal over-fitting. Kiran et al. [5] aimed to predict house prices for non-householders using various parameters and advanced machine learning regression techniques. They found that the Catboost Regression Algorithm outperformed other algorithms in terms of efficiency and minimum error. Sifie et al. [6] proposed a hybrid Lasso and Gradient boosting regression model to predict individual house prices of the Ames Housing dataset. The proposed method achieved promising performance, ranking in the top 1% of all competing teams and individuals in the Kaggle competition. On the other hand, Yiyang

[7] focused on micro factors such as lot area and pool area. The author implemented machine learning algorithms such as random forest and support vector machine to predict asset pricing and report good results.

The selected papers demonstrate that data processing and feature engineering are critical in predicting house prices accurately. The papers also show that advanced machine learning techniques such as Lasso and Gradient boosting regression models, and machine learning algorithms such as random forest and support vector machine, can improve the accuracy of house price prediction. The datasets utilized, the parameters evaluated, and the models offered vary among articles, indicating that there is no one-size-fits-all approach to housing price prediction. However, the findings reported in these publications are encouraging, suggesting that machine learning techniques may be used to estimate housing values. More study is needed, however, to determine the best mix of methodologies and models for distinct datasets and characteristics. Therefore, in this project, I would like to focus on feature engineering using techniques such as log transformation, and feature selection to improve and compare the performance of several regression models on this dataset to find the most accurate and reliable predictive models.

3. Data

This study uses the Ames Housing dataset, which is available on the Kaggle website [1]. The dataset includes details about homes in Ames, Iowa, including variables like overall quality, lot size, the number of beds, and the year of construction. The dataset has a total of 1460 observations, with 79 explanatory variables and one target variable, sale price [1].

The summary statistics of the numerical variables revealed a wide range of values and potential outliers in some of the variables. The correlation matrix showed that the overall quality of the house, living area, and garage size were highly correlated with the sale price. Additionally, a scatterplot matrix revealed some interesting relationships between these variables and sale price (Figure 1). The scatterplot matrix showed that as the overall quality of the house, living area, and garage size increased, so did the sale price.

The distribution of the response variable, sale price, was right skewed, showing that most residences were sold for around or less than $200,000, while some were sold for significantly higher prices. The boxplots of the sale price by overall quality, neighborhood, and year built showed some significant differences in the median sale prices across these variables (Figure 1). Overall, this data exploratory analysis provided valuable insights into the dataset and identified potential variables that could be used to predict the sale price of residential homes in Ames, Iowa.

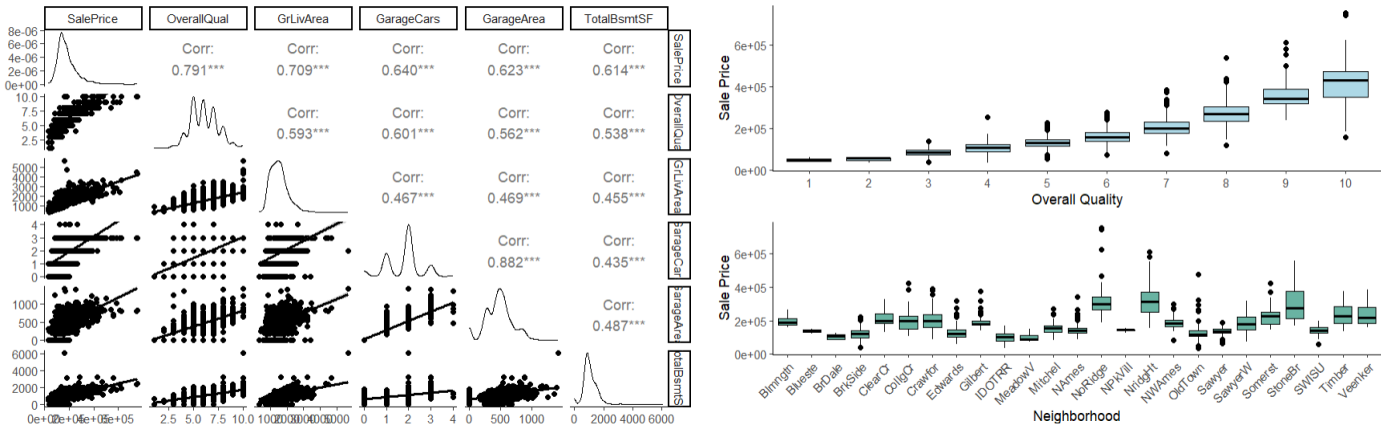

Figure 1. Scatterplot matrix (left) and boxplots (right) illustrate relationships between housing sale prices and several predictor variables.

4. Method

This methodology section consists of seven main sub-items, a data cleaning and processing item, and six regression model items. All data processing and model building codes were implemented in R and can be found by this GitHub link.

4.1 Data Cleaning and Processing

The Ames Housing dataset required extensive cleaning and preprocessing before it could be used for further analysis. The dataset contained 1460 observations and 81 variables; some had a high percentage of missing values, and some were unnecessary for subsequent analysis. Therefore, data cleaning and preprocessing steps aimed to impute missing values, transform, and create new variables, remove unnecessary variables, and ensure data consistency.

The cleaning and preprocessing step started by loading the data into R and checking variable names and types. Missing values were then imputed using various methods, such as mean imputation, mode imputation, and constant imputation, depending on the variable and the nature of the missingness. Data scaling and standardizing were applied where the algorithms required. Existing variables were combined to generate new ones, such as the total square feet variable, which was formed by adding the total basement, first-floor, and second-floor square feet variables.

The cleaning and preprocessing step also transformed some variables into factors to treat them as categorical variables in the subsequent analysis. For example, the MSSubClass variable, which represented the type of dwelling, was transformed into a factor to treat it as a categorical variable. Unnecessary variables were removed from the dataset, such as the ID, Street variables, and variables with a high percentage of missing values. In addition, since house price data is a very right-skewed distribution, taking the log of the sale price compresses the range of the data and makes the data more normally distributed, which can lead to better model performance (Figure 2).

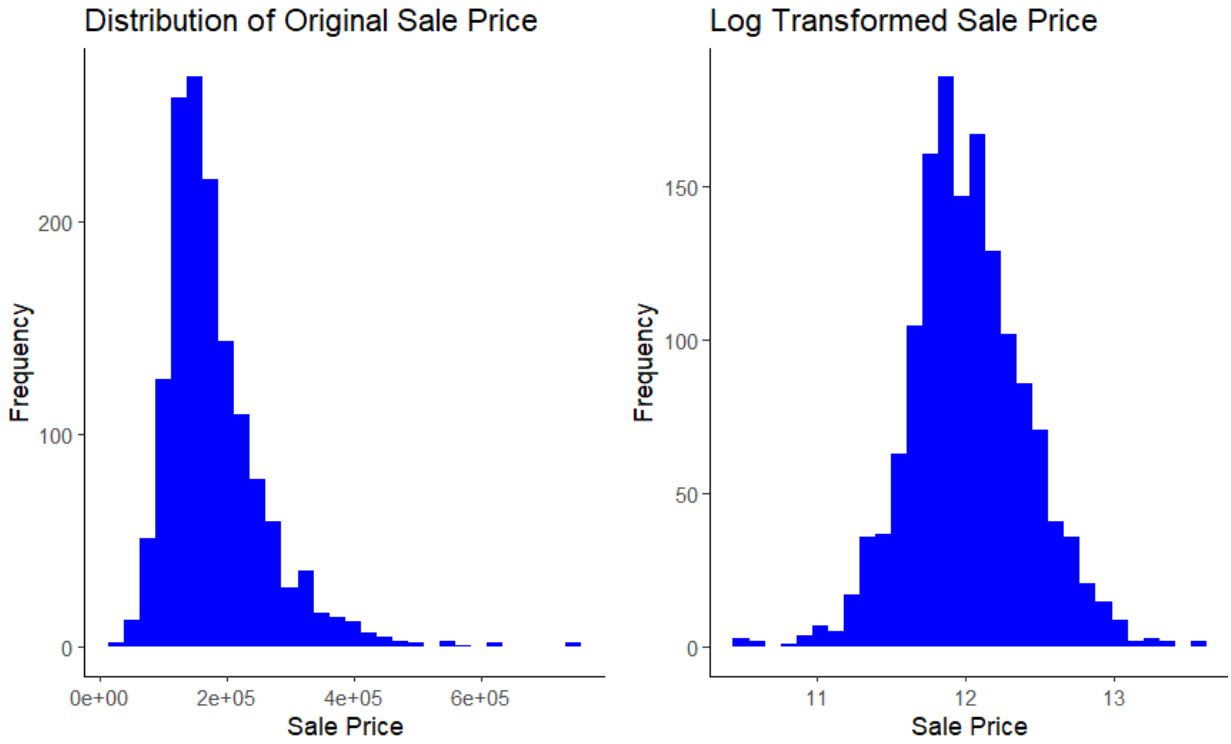

Figure 2. Distribution of original and log-transformed the sale price of housing.

This cleaning and preprocessing step were critical to ensuring that the subsequent analysis was based on a clean and consistent dataset. The imputation of missing values, creation of new variables, transformation of variables, and removal of unnecessary variables improved the quality of the dataset and reduced the likelihood of bias in the subsequent analysis.

4.2 Multiple Linear Regression

The first method applied in this dataset is multiple linear regression, which is a statistical modelling approach that uses two or more input variables to predict a continuous numerical variable. The method works by determining the coefficients of a linear equation that best reflects the relationship between the

input and output variables and minimizing the mean squared error between anticipated and actual values. By minimizing the following object equation, the optimal coefficients can be estimated [3].

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\}$$

In this project, multiple linear regression was implemented in R using the train function from the caret package. The train function is used to train a linear regression model using the lm technique and the training data from the training dataset. For model selection, the trainControl parameter is configured to perform 10-fold cross-validation. The tuneLength parameter is set to 10, although, there are no explicit hyperparameters to tune for the lm method in linear regression, the train function will still tune the parameters of the model during cross-validation to optimize performance.

The predict function is used to compute predictions on the testing dataset once the model has been trained. Finally, two metrics, root mean squared error (RMSE) and R-squared (R^2) were calculated to assess the linear regression model's performance.

4.3 Ridge Regression

Ridge regression is a regularization technique used to prevent overfitting in linear regression models [2, 3]. The method works by adding a penalty term to the ordinary least squares regression objective function, which is proportional to the square of the magnitude of the coefficients. This results in the regression coefficients being shrunken towards zero, reducing the variance of the model at the expense of increasing the bias [2, 3]. The Ridge regression coefficients are estimated by optimizing the following equation [3].

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s$$

The Ridge regression using the glmnet package in R. First, the model.matrix() function is used to create a matrix of predictor variables x and a vector of response variable y from the training dataset. Then, the glmnet function is used to fit a ridge regression model with alpha equal to 0 and lambda is set to use a range of lambda values to be used for tuning the model defined by the lambda variable. The alpha parameter determines the type of regularization, with 0 corresponding to ridge regression and 1 corresponding to lasso regression. The lambda parameter controls the strength of the regularization, with larger values resulting in more shrinkage. Next, cross-validation is used to find the optimal value of lambda. The cv.glmnet() function is used to perform cross-validation with 10 folds. The resulting object cv contains information about the optimal value of lambda and the cross-validated mean squared error. After finding the optimal value of lambda, a new ridge regression model is fit using the glmnet function with alpha = 0 and lambda set to the value of lambda.min from the cross-validation. Similar to multiple regression, the predict() function is used to make predictions on the testing dataset. RMSE and R^2 metrics are also used to evaluate Ridge model performance.

4.4 Lasso Regression

Lasso regression is another type of linear regression that is used for feature selection and regularization. It is similar to Ridge regression, but instead of using the L2 penalty, it uses the L1 penalty [2, 3]. The L1 penalty helps to reduce the coefficients of less important predictors to zero, effectively eliminating them from the model. This method can be useful in cases where there are many predictors and only a few are actually important in predicting the response variable [2, 3]. The following equation is a mathematical expression of the Lasso objective function [3].

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

The first step is to fit a Lasso regression model using the glmnet() function with alpha equal to 1 which specifies Lasso regression. The range of lambda values to be used for tuning the model is defined by lambda variable. The 10-fold cross-validation is then performed on the model using the cv.glmnet() function, which selects the optimal lambda value. The lambda.min value is then used to fit a new Lasso regression model using the glmnet function with alpha = 1. Predictions are then made on the test data using the predict() function and the resulting RMSE and R-squared values are calculated using the caret package.

4.5 Elastic Net Regression

Elastic Net regression is a regularization method that combines the Lasso and Ridge regression methods. It adds a penalty term to the objective function of the linear regression model, which is a combination of the L1 (Lasso) and L2 (Ridge) regularization terms [2, 3]. The Elastic Net method aims to overcome the limitations of Lasso and Ridge regression and provide better predictive performance and model interpretability. The Elastic Net regression's objective equation is expressed as follows [3].

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij}\right)^2 + \lambda \left[(1-\alpha) \sum_{j=1}^{p} \beta_j^2 + \alpha \sum_{j=1}^{p} |\beta_j|\right]$$

Elastic Net regression model was first fitted to the training dataset using the glmnet method in the train function of the caret package. The trainControl function is used to specify the type of resampling method to be used for cross-validation, and the tuneLength parameter is used to specify the number of tuning parameter values to be evaluated. The best tuning parameter value is determined using 10-fold cross-validation. The best-tuned model obtained from the train function was used to predict housing sale prices on the testing dataset. RMSE and R-squared metrics will be used to evaluate the elastic net model performance metrics.

4.6 Principal Component Regression

Principle Components Regression (PCR) is a method used for predicting a response variable by analyzing the relationship between the response variable and predictor variables. However, unlike other above regression methods, PCR uses a set of principal components as predictors, which are linear combinations of the original predictors. The following is a mathematical expression to find the principal components [2].

$$\text{Maximize}_{\phi_{11}, \ldots, \phi_{p1}} \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \quad \text{subject to the constraint that } \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

The train() function from the caret package is used to fit the PCR model using the training data. The formula specifies that sale price is the response variable and all other variables in the data frame are predictor variables. The method argument is set to pcr to specify that the PCR method should be used. The preProcess argument specifies that the data should be centered and scaled before fitting the model. The trControl argument specifies that 10-fold cross-validation should be used to evaluate the model. After fitting the model, summary() and plot() are used to print the model summary and plot the model. The predict() function is used to make predictions on the test data.

4.7 Random Forest Regression

Random Forest Regression (RFR) is a machine learning technique used for regression tasks. RFR involves creating multiple decision trees and combining their predictions to produce the final output. In RFR, each

decision tree is built using a subset of the data and a random subset of the predictor variables, which helps to reduce overfitting and improve accuracy.

RFR is implemented by using the caret library in R. The train() function from the caret package is used to fit the RFR model using the training data. The formula SalePrice ~ . specifies that sale price is the response variable and all other variables in the data frame are predictor variables. The method argument is set to "rf" to specify that the RFR method should be used. The trControl argument specifies that 10-fold cross-validation should be used to evaluate the model, and the tuneLength argument specifies the number of trees to grow in the forest. After fitting the model, the predict() function is used to make predictions on the test data, and the resulting predictions are stored in rf.pred. Finally, caret::RMSE() and caret::R2() functions are used to calculate the performance metrics RMSE and R^2 between the predicted values and the actual sale price values in the testing data.

5. Results

Models were evaluated based on two common regression metrics, RMSE and R2. Detailed results are shown in Table 1.

Table 1. RMSE and R^2 of regression models on test data

Model	RMSE	Rsquare
Lasso Regression	0.129	0.904
Elastic Net Regression	0.132	0.902
Ridge Regression	0.140	0.891
Linear Regression	0.142	0.879
Random Forest Regression	0.149	0.875
PCR Regression	0.160	0.853

RMSE is a metric that measures the accuracy of regression models by calculating the square root of the average squared difference between predicted and actual sale prices. A lower value of RMSE indicates a higher accuracy of the predictive model. Among all models, the Lasso regression model has the lowest RMSE score of 0.129, indicating the highest accuracy in predicting housing prices on the test dataset. On the other hand, the PCR regression model has the highest RMSE score of 0.160, which indicates this model has the highest prediction error among all models. The other two regularization regression methods, Elastic Net and Ridge, have the second and third lowest RMSE values of 0.132 and 0.140, respectively. Following these models, Linear regressions and Random Forest have an RMSE value of 0.142 and 0.149, respectively. Overall, all models have relatively low RMSE values, indicating that they perform well in predicting the response variable, housing sale price.

R-squared is another metric used to evaluate the performance of regression models. It measures the proportion of the variability of the target variable, housing sale prices that is explained by the predictor variables. In contrast to the RSME metric, the higher R-squared, the better the model performance. The results show that the Lasso regression has the highest R-squared of 0.904. This means that 90.4% of the variance in sale prices of housing on the test dataset can be explained by input independent variables. It is followed by Elastic Net regression and Ridge regression models, which have an R-squared of 0.902 and 0.891, respectively. Following Ridge regression is Linear regression model, which has an R-squared of 0.879. This number is slightly higher than the R-squared of Random Forest regression, 0.875. While, PCR regression model has the lowest R-squared of 0.853 on test data.

In summary, the results show that Lasso regression model outperforms other models in terms of both RMSE and R-squared metrics on this dataset. The regularization regression methods, Elastic Net and Ridge, also perform well in terms of RMSE and R-squared. The Linear regression and Random Forest regression models have slightly lower performance metrics than the regularization regression methods. The PCR regression model has the lowest performance metrics among all models. Overall, the models

have relatively low RMSE values and high R-squared values, indicating that they perform well in predicting the response variable, housing sale price, and can be used for further analysis.

The Lasso, Elastic Net, and Linear regressions showed that some variables, such as overall quality, and ground living area, total basement square feet, garage, year built had statistical significance and predictive power for sale price.

6. Discussions & Conclusions

This study uses the Ames Housing dataset to explore the potential of machine learning algorithms in predicting housing sale prices. The study utilized several regression models and found that the Lasso regression model outperformed others, with an R-squared value of 0.904 and an RMSE of 0.129. This result shows that regularization regression models in general and Lasso regression, in particular, has performed well on the Ames Housing dataset with many predictors. The study also highlights the significance of accurate housing sale price forecasting, which can assist buyers and sellers in planning budgets and financial institutions in making informed financing decisions. Moreover, this study reveals that data preprocessing and feature engineering are crucial in predicting house prices accurately.

However, it is important to highlight that the study's conclusions are confined to the dataset utilized in this study. There are external influences on the housing sale prices such as demand, economic shifts, and government policy have not been considered in this research. To increase the accuracy of house price prediction, I am planning to implement other advanced machine learning techniques such as Neural Networks and Deep Learning.

In conclusion, this study provides valuable insights into the factors that drive housing prices and highlights the potential of machine learning techniques in predicting housing values. Further research is needed to apply more complex algorithms like Deep Learning or determine the best mix of methodologies and models for distinct datasets and characteristics. Additionally, future studies should consider external factors to increase the accuracy of housing sale price forecasting.

References

1. Kaggle (2023). House Prices Dataset. Retrieved 2023, March 09 from http://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data
2. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, *Second Edition. Springer*. https://doi.org/10.1007/978-0-387-84858-7
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R, *Second Edition. Springer*.
4. Chenchen Fan, Zechen Cui, Xiaofeng Zhong (2018). House Prices Prediction with machine Learning Algorithms. *Proceedings of the 2018 10th International Conference on Machine Learning and Computing, pp. 6-10*. https://dl.acm.org/doi/pdf/10.1145/3195106.3195133
5. G Kiran, D. Malathi, Neeraja and Syed (2021). Prediction of House Price Using Machine Learning Algorithms, *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 2021, pp. 1268-1271. https://ieeexplore.ieee.org/abstract/document/9452820
6. Sifie, Zengxiang,Zheng, Xulie and Rick (2017). A hybrid regression technique for house prices prediction, *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Singapore, 2017, pp. 319-323. https://ieeexplore.ieee.org/abstract/document/8289904
7. Yiyang (2019). Residential Asset Pricing Prediction using Machine Learning, *2019 International Conference on Economic Management and Model Engineering (ICEMME)*, Malacca, Malaysia, 2019, pp. 193-198. https://ieeexplore.ieee.org/abstract/document/8988299