Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

Evaluating Machine Learning Models in Housing Price Forecasting

Kailin Wang*

Lehigh University, 27 Memorial Drive West, Bethlehem, PA 18015 USA

*Corresponding author: kaw226@lehigh.edu

Abstract. This study aims to compare the performance of different machine learning algorithms in predicting housing prices and to construct an experimental process based on data preprocessing, feature selection, and comparative analysis of multiple regression models. Initially, a Random Forest will filter out the ten most significant variables. This variable will be utilized to build Multivariable Linear Regression (MLR). To find the best subset, lasso, best subset, and 5-fold Cross-Validation methods will be applied to find the final model. At the same time, this study incorporates training and testing using models such as K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Bagging, Random Forest (RF), Boosting, and Bayesian Additive Regression Trees (BART). By comparing the error performance of the model on the training set and the test set, the mean squared error (MSE) will be calculated to evaluate the strengths and weaknesses of each model in housing price prediction. The goal is to find the modeling method and variable combination that best suits the data set.

Keywords: Housing Price Prediction; Machine Learning; Feature Selection.

1. Introduction

Predicting house prices is meaningful in real-world applications, especially in areas and cities experiencing rapid urbanization and growth in the real estate sector. Accurately forecasting housing prices is conducive to having a better negotiation between buyers and sellers which contributes to formulating a fair price. Also, it assists the government to assess real estate market trends and make more effective decisions. However, traditional calculation models often have significant errors when compared to the actual selling price. In this case, there is a need for advanced machine learning algorithms to improve the model accuracy and provide better interpretations [1]. For example, Quang Truong et al. used five models, including Random Forest, Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LightGBM), Hybrid Regression, and Stacked Generalization Regression, and concluded that although the stacked regression model has a more complex structure, by comparing root mean squared logarithmic error (RMSLE), the Stacked Generalization Regression method has the highest accuracy [2]. Similarly, Danh Phan used a series of algorithms, including Linear Regression, Polynomial Regression, Regression Trees, SVM, and Neural Networks to predict Melbourne house prices. Finally, the optimized SVM was determined to be the best performance based on mean square error [3].

Other studies have focused on traditional regression methods. Qingqi Zhang used the Boston housing dataset, applied MLR with Spearman correlation-based feature selection and found that MLR can predict general price trends, although it lacks accuracy in more complex situations [4]. In contrast, Nadia Putri Ariyanti et al. compared the KNN and MLR model in predicting housing prices in Jabodetabek, Indonesia and concluded the multiple linear regression model is significantly better than the KNN model in these indicators [5].

Ensemble learning approaches have also demonstrated strong performance in house price prediction. Abigail Bola Adetunji et al. trained a Random Forest model on the UCI Boston dataset and achieved a tight error margin of ±5%, proving its effectiveness in capturing relationships between multiple variables [6]. In a related study, Anders Hjort et al. introduced a locally interpretable boosting method, training the model using real house price data and introducing interaction constraints to constrain the

tree structure. Compared to traditional black-box ensemble models, the experimental results suggest that boosting provides more transparent insights into variable interactions [7].

Bayesian approaches have also gained attention for their flexibility and robustness. Gu Jirong et al. evaluated several models, including SVM, RF, GBM, and BART, and found that the BART model performs well in prediction accuracy and outperforms other comparison models [8]. Additionally, Chipman et al. proposed BART as a Bayesian additive model that regularizes each tree and performs variable selection via posterior sampling. Experimental results showed that BART is more stable and has a lower mean square error (RMSE) in the prediction tasks of 42 real data sets compared to methods such as Random Forests, Boosting, Neural Networks, and Lasso [9].

These diverse studies show that the performance of each method usually depends on data characteristics, feature structure, and the trade-off between accuracy and interpretability. Therefore, this study aims to conduct a comprehensive comparative analysis of several machine learning algorithms (including MLR, LASSO, best subset regression, KNN, SVM, Bagging, RF, Boosting, and BART). By evaluating model performance based on MSE on both training and testing sets to determine the best modeling method and variable combination to achieve accurate and interpretable housing price prediction.

2. Methodology

2.1. Data Preprocessiong

The dataset for this article comes from a Kaggle competition [10], and the dataset contains 1460 data with 81 variables. Variables that had more than 50% missing values were removed, and the mode method was used to replace the missing value. Since the dataset contains 81 variables, using all of them in modeling could lead to high computational cost and overfitting. Therefore, Random Forest can be applied in feature selection to identify the top 10 most important variables that contribute the most to the prediction of house prices; they are "OverallQual," "GrLivArea," "GarageCars," "TotalBsmtSF," "ExterQual," "YearBuilt," "GarageArea," "first floor," "second floor," and "BsmtQual". Categorical variables were converted into numerical format through category encoding to ensure machine learning algorithms ran smoothly. The dataset was split into a training set and a test set. 80% of the data was randomly divided into train data and the rest as test data.

2.2. Exploratory Data Analysis

Before modeling, Exploratory Data Analysis (EDA) was conducted to show the linear relationship among the variables. As shown in Figure 1, a scatter matrix can visualize variable correlations and identify potential multicollinearity. The analysis revealed that several pairs have a strong correlation. They are (GarageCars, GarageArea), (TotalBsmtSF, first floor), and (OverallQual, SalePrice). These strong associations suggest the presence of multicollinearity, which may impact the performance and interpretability of regression models.

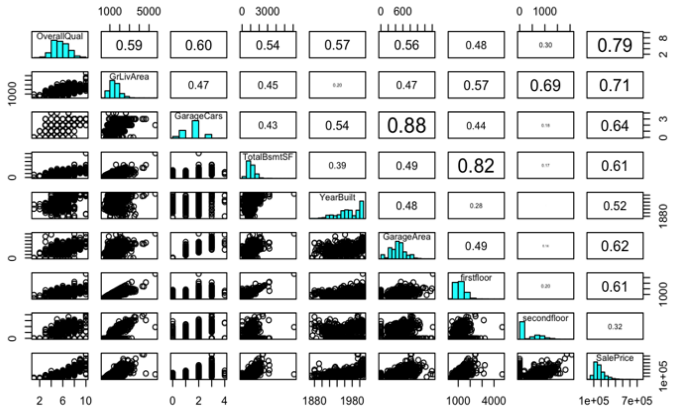

Fig. 1 Scatter matrix showing relationships among selected variables (Picture credit: Original)

2.3. Model Selection

For each model, hyperparameters were tuned using cross-validation when applicable, and model performance was evaluated by using MSE, defined as the mean of the squared differences between predicted and actual values on the training set and testing set.

2.3.1. Multiple Linear Regression

First, a model was fitted using all variables. The initial model showed good explanatory power, with an adjusted R-squared of 0.7865 and a significant F statistic. However, diagnostic plots and statistical tests indicate that the model violates the assumption due to BeNon-normality and heteroscedasticity in residuals, which was confirmed by Shapiro-Wilk and Breusch-Pagan tests. To address these issues, a log transformation was applied to the target variable, SalePrice. Although it improves residual distribution slightly, heteroscedasticity stays. Then, multicollinearity was tested using Variance Inflation Factor (VIF), revealing severe multicollinearity between some variables (with VIF values exceeding 100). Thus, Best Subset and LASSO are used. The analysis suggests reducing the set of predictors, with LASSO recommending using the full model. To avoid overfitting, cross-validation was performed for Lasso regression to determine the optimal regularization parameter λ. As shown in Figure 2, the 5-fold cross-validation curve plots the MSE against log(λ). By choosing λ=λ.min to get the minimum error. The result of 5-fold cross-validation showed that the final model should use the full model selected by Lasso. Finally, ten variables were modeled and adjusted R-squared = 0.7679, and residuals are a normal distribution.

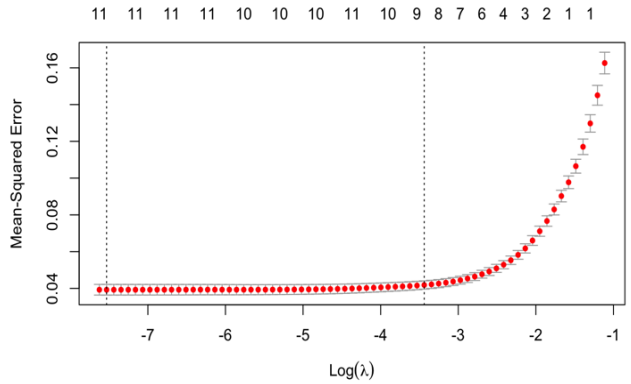

Fig. 2 Lasso Regression 5-Fold Cross-Validation Curve for Optimal Lambda Selection (Picture credit: Original)

2.3.2. K-Nearest Neighbors

For the KNN model, category variables were removed first, and then Leave-One-Out Cross-Validation(LOOCV) was used to determine the optimal value of k, as shown in Figure 3, as k increases, the LOOCV error generally decreases and stabilizes around k equal 5 to 10. Then, the `knnreg()` function was applied to model the train data with k=5. The training error rate was calculated using MSE. The result of the training error rate by KNN is 0.00932068.

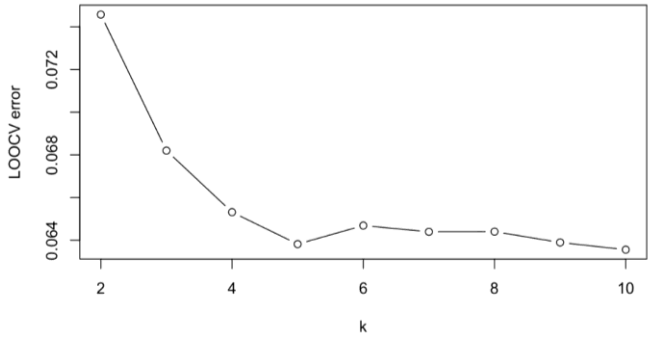

Fig. 3 LOOCV Error vs. K in K-Nearest Neighbors (KNN) Model (Picture credit: Original)

2.3.3. Support Vector Machine

Compare three kernel functions of SVM: Radial Basis Function, Polynomial, and Linear. Using the `svm ()` function for regression modeling, parameter tuning was done by grid search within the preset parameter range through the `tune ()` function where the cost list was set to `exp(seq(log(0.01), log(10), length.out=11))`. After tuning, the model was retrained with the optimal parameters. The analysis indicated that the training error rates were as follows: for the radial kernel, the training error rate was 0.0279815; for the polynomial kernel, the training error rate was 0.0306208; and for the linear kernel, the training error rate was 0.0382253. According to the comparison, the radial kernel function has the lowest training error, so it was selected as the final model.

2.3.4. Bagging

For Bagging model, the 'randomForest ()` function was used with all predictors, specifying `mtry = ncol(data.train)`. This means each tree is built using all available features. The training error calculated by MSE is 0.0067836.

2.3.5. Random Forest

In Random Forest, Cross-validation determines the optimal number of features (mtry) and the result is 8. The model with 500 trees was built by using the `randomForest()` function and the mtry value. The experimental result is that the training error rate by random forest is 0.00682166.

2.3.6. Boosting

In the boosting model, first, some categorical variables (e.g. ExterQual, BsmtQual) are converted into factors. A total of 5000 trees were trained using the `gbm()` function and interaction depth 2, specifying regression as Gaussian distribution. As shown in Figure 4, the Boosting model was used to assess the relative influence of each variable on housing price prediction. The figure visualizes the top predictors ranked by their contribution to the model's performance. firstfloor, YearBuilt, and BsmtQual appear to be the most influential variables, The final resulting training error rate from the boosting model was 0.00854134.

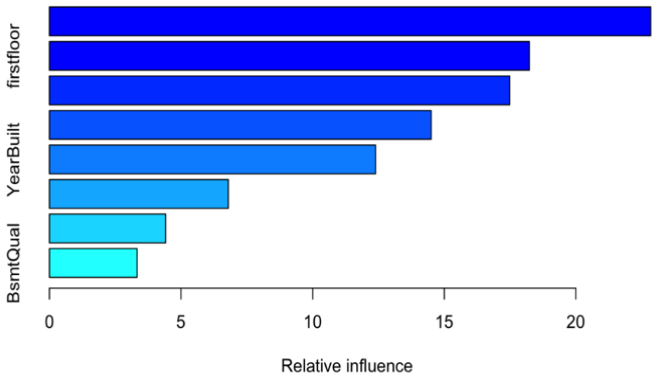

Fig. 4 Variable Importance from Boosting Model (Picture credit: Original)

2.3.7. Bayesian Additive Regression Trees

The BART model specified `type='wBART`, thus, weighted BART variant to account for heteroscedasticity. Additionally, the `print every` parameter was set to 10000, ensuring that the training process was printed every 10,000 iterations. The training error rate by BART is 0.296433.

3. Experiment Result

To evaluate the performance of the models, both training error and testing error were computed using MSE as the evaluation metric. Table 1 summarizes the results across all models, including traditional regression techniques, distance-based algorithms, SVM, and ensemble learning approaches. Each model was trained using the 80% training split and evaluated on the remaining 20% test set. By

comparing training and testing errors, this paper aims to assess the overall accuracy of each model. The models with balanced and relatively low error rates are considered more stable.

Table 1. Comparison of Training and Testing Errors Across Regression Models

Error	MLR	KNN	SVM	Bagging	RF	Boosting	Bart
Testing Error	0.0001206456	0.0001932139	9.111517e-06	0.0004351297	0.0004638364	2.578407e-07	5.223437e-06
Training Error	0.00932068	0.00932068	0.0382253	0.0067836	0.00682166	0.00854134	0.296433

According to Table 1, the Boosting model has the lowest testing error (2.5784e-07) among all the evaluated models, indicating that it is highly accurate in predicting actual housing prices. Boosting, as an integrated learning technique, improves model performance by gradually minimizing residual errors. Its excellent performance in this analysis proves its effectiveness in dealing with complex nonlinear relationships in regression tasks. Although the testing errors of SVM and BART are also relatively low, their training errors are relatively high. This indicates that these models may have underfitting problems, which means that they are unstable during the training process. The MLR and KNN models have achieved a balanced error between testing and training errors, indicating that they can be applied to complex data. The table shows that the testing error of the KNN model is lower than that of the MLR, which indicates that KNN is more accurate than MLR in practical application areas. However, both models are easily interpretable, making them more attractive in practical implementation. For the Bagging and Random Forest (RF) models, the training errors are very close—0.00678 and 0.00682, respectively—demonstrating stability. The advantages of these tree-based ensemble methods are resistance to overfitting and stability, which are demonstrated in the experimental results.

In conclusion, while Boosting has the smallest value in testing error values, models such as KNN and MLR have relatively balanced errors and are easy to interpret. These advantages are crucial in practical applications because understanding the model and analyzing its results are as important as the model's predictive ability. In contrast, the BART model, despite its flexibility, produces a high training error in this case and is, therefore, not recommended for use in the final model selection.

4. Conclusion

This study conducted a comprehensive comparison of various machine learning algorithms in the field of house price prediction. Based on training error and test error, the Boosting model showed the best prediction performance and achieved the lowest test error among all evaluated models. Ensemble models such as Bagging and Random Forest also showed stable prediction results with low training errors. MLR and KNN had slightly higher errors but were more interpretable. BART had a higher training error in this experiment, indicating that there may be underfitting. Based on these findings, future work of this study will explore hybrid methods to further improve the prediction performance.

In conclusion, although Boosting showed the highest accuracy in this study, other models such as MLR and KNN also have practical advantages in terms of interpretability and stable performance. Hybrid methods will be a continued focus of future research.

This study not only provides empirical support for understanding the law of housing price changes, but also provides a certain theoretical basis and methodological reference for research and practice in related fields. This paper look forward to the research of advanced algorithms in the future to provide support for building a more scientific and efficient housing price prediction system.

References

[1] Geerts M, Vanden Broucke S, De Weerdt J. A survey of methods and input data types for house price prediction. ISPRS International Journal of Geo-Information, 2023, 12(5): 200.

[2] Q. Truong, M. Nguyen, H. Dang, and B. Mei, "Housing price prediction via improved machine learning techniques," Procedia Computer Science, vol. 174, pp. 433–442, 2020.

[3] T. D. Phan, "Housing Price Prediction Using Machine Learning Algorithms: The Case of Melbourne City, Australia," 2018 International Conference on Machine Learning and Data Engineering (iCMLDE), Sydney, NSW, Australia, 2018, pp. 35-42.

[4] Q. Zhang, "Housing price prediction based on multiple linear regression," Scientific Programming, vol. 2021, Article ID 7678931, 9 pages, 2021.

[5] N. P. Ariyanti, A. Triayudi, and R. T. K. Sari, "Analysis of K-NN Algorithm and Linear Regression to Predict House Prices in Jabodetabek," SaNa: Journal of Blockchain, NFTs and Metaverse Technology, vol. 2, no. 1, pp. 65–71, Feb. 2024.

[6] A. B. Adetunji, O. N. Akande, F. A. Ajala, O. Oyewo, Y. F. Akande, and G. Oluwadara, "House Price Prediction using Random Forest Machine Learning Technique," Procedia Computer Science, vol. 199, pp. 806–813, 2022, doi: 10.1016/j.procs.2022.01.100.

[7] A. Hjort, I. Scheel, D. E. Sommervoll, and J. Pensar, "Locally interpretable tree boosting: An application to house price prediction," Decision Support Systems, vol. 178, Art. no. 114106, 2024, doi: 10.1016/j.dss.2023.114106.

[8] J. Gu, M. Zhu, and L. Jiang, "Housing price forecasting based on genetic algorithm and support vector machine," Expert Systems with Applications, vol. 38, no. 4, pp. 3383–3386, 2011.

[9] H. A. Chipman, E. I. George, and R. E. McCulloch, BART: Bayesian additive regression trees, The Annals of Applied Statistics, vol. 4, no. 1, pp. 266–298, 2010, doi: 10.1214/09-AOAS285.

[10] Montoya A, DataCanary. House Prices – Advanced Regression Techniques. Kaggle, 2016. Available: https://kaggle.com/competitions/house-prices-advanced-regression-techniques