House price prediction using machine learning: A case study in Seattle, U.S.

Jiapei Liao

Faculty of Science and Technology, University of Macau, Macau, 999078, China

Corresponding author: dc12784@umac.mo

Abstract. In recent years, predicting housing prices has become a prominent research topic. The motivation behind this research is the lack of precise analysis and comprehensive comparison of different machine learning models for Seattle housing prices. To address this issue, this paper utilizes a dataset obtained from Kaggle, consisting of Seattle housing prices. This study focuses on analyzing and comparing the performance of different machine learning models in predicting housing prices in Seattle. After preprocessing the data, eight different machine learning models are applied to predict housing prices in Seattle. A comprehensive comparison of these models is conducted to analyze their differences. The experimental results show that the differences in performance among the models are not substantial. However, StackingAveragedModels emerges as the top-performing model, with an RMSLE of 0.2328 and an R2 of 0.7771. These findings contribute to a better understanding of the predictive capabilities of different machine learning models for Seattle housing prices.

Keywords: Machine Learning, Seattle, Housing Price Forecast, StackingAveragedModels.

1. Introduction

House price prediction has become a popular research field, particularly with the advancement of artificial intelligence and the widespread use of machine learning algorithms [1]. Accurate house price prediction holds significant importance for real estate professionals, potential homebuyers, as well as government policies and urban planning, among other stakeholders [2]. It can greatly assist individuals in making informed decisions. Consequently, there is a need for a high-precision real estate price prediction model. House prices are influenced by various factors, such as geographical location, house size, and land area. By leveraging machine learning algorithms, reliable predictions can be obtained.

Homeownership is highly valued by Americans, as owning a home is seen as both a tool for economic mobility and a symbol of social status. Having a home can deepen one's sense of security and provide the flexibility to move to more desirable neighborhoods. Therefore, for American families, considering housing options is of great significance.

This paper utilizes Seattle, a prominent seaport city on the west coast of the United States, as a prime example to forecast its housing prices. Seattle serves as the vibrant seat of King County, Washington, and stands as the most populous city in the state of Washington and the entire Pacific Northwest region of North America.

However, Seattle's notorious traffic congestion ranks among the worst in the United States [3]. In recent years, the city's real estate market has been booming, with both property prices and rents experiencing significant increases. The high cost of housing has resulted in a growing homelessness issue, making Seattle the second-largest city in terms of homelessness, just behind Los Angeles. The study aims to employ machine learning techniques in constructing a promising house price prediction model for Seattle.

A related research report on Kaggle trained and predicted the same dataset, yielding favorable results. However, it did not conduct a comparative analysis of multiple algorithm models. Despite the maturity of machine learning algorithms, there has not been a sufficiently in-depth analysis of the variations among different models. Therefore, this paper selects eight commonly used machine learning models to conduct an in-depth study of the Seattle housing dataset. A comprehensive comparison and analysis of these eight models were conducted to determine the best-performing model.

2. Method

In this study, the training and test sets were sourced from the Kaggle platform. The data indicators were subjected to an analytical examination, followed by preprocessing of the input data. Subsequently, an ensemble of machine learning models, including linear regression (Lasso), ElasticNet, KernelRidge (KRR), Random Forest (RF), GBoost, and XGBoost regression (XGB), were constructed and trained. Additionally, two ensemble approaches, namely AveragingModels and StackingAveragedModels, were employed. The resulting models were subsequently utilized to generate outcomes for further analysis.

2.1. Exploratory data analysis

According to previous research, the dataset's quality significantly impacts research outcomes. In this study, we utilized the Seattle Housing dataset from Kaggle [4], which consists of real housing price data for homes sold in Seattle, Washington. The variable targeted for prediction is the house price, while the independent variables consist of beds, baths, size, total land area, and zip code. The Seattle Housing Dataset comprises data about 2515 single family houses sold in Seattle, Washing, USA between August and December 2022. Each sample in the dataset consists of 8 features (shown in Table 1).

Table 1. Description of features in the Dataset

Features	Description
beds	The number of bedrooms within the house
baths	The count of bathrooms available in the property. Please note that a value of 0.5 indicates the presence of a half-bath, which includes a sink and toilet but lacks a bathtub or shower
size	The overall floor area of the property
size_units	The units in which the previous measurement is expressed
lot_size	The complete land area on which the property is situated. The ownership of the lot resides with the homeowner
lot_size_units	Units of the previous measurement
zip_code	A postal code utilized in the United States
price	The selling price of the property in US dollars

First, the normal distribution plot of the "price" variable and the quantile-quantile plot are drawn. (Figure 1). By examining the quantile-quantile plot, we can determine whether the "price" variable follows a normal distribution.

As illustrated in Figure 1, it is evident that the "price" variable deviates from a normal distribution. Therefore, in order to address this non-normality, a logarithmic transformation has been applied to the data. The transformation involves taking the natural logarithm of the values, using the formula log(1+x). This will generate a new set of transformed data. Subsequently, we can draw the normal distribution plot and the quantile-quantile plot (Figure 2) for the transformed data. In the quantile-quantile plot, we would expect the red line and the blue points to closely coincide, indicating conformity with the normal distribution. By comparing the quantile-quantile plot in Figure 1 with that in Figure 2, it can be observed that after the logarithmic transformation, the red line aligns more closely with the blue points. This indicates that the data is more in line with a normal distribution.

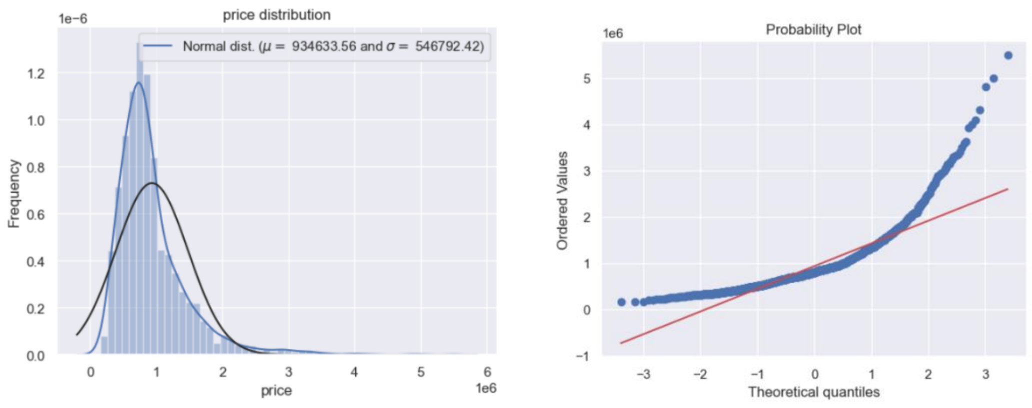


Figure 1. Normal Distribution Plot and Quantile-Quantile Plot of the 'Price' Variable

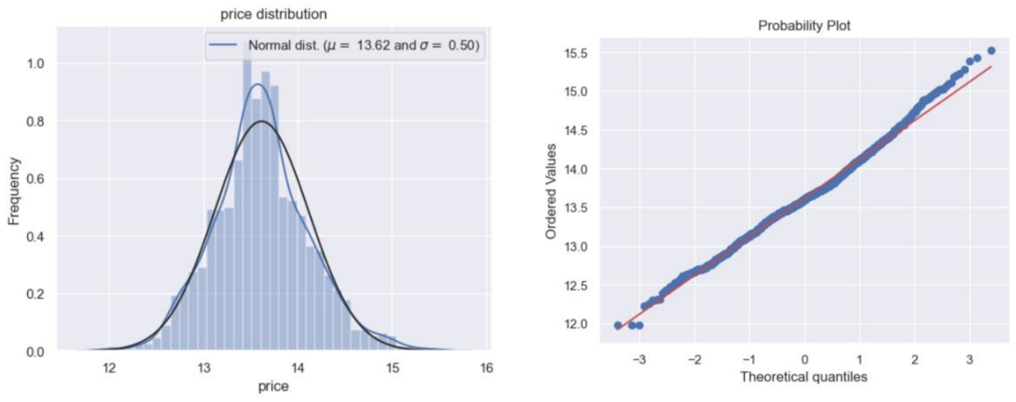
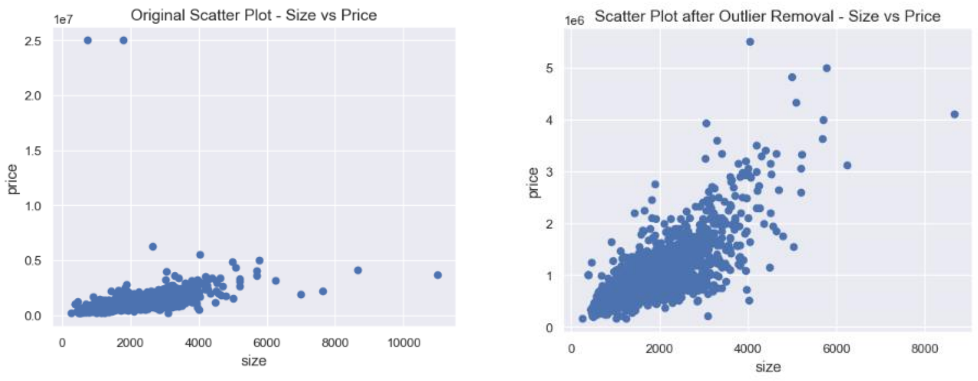


Figure 2. Normal Distribution Plot and Quantile-Quantile Plot of Logarithmic Transformed Data"

To analyze the relationship between each variable and the "price" variable, scatter plots were generated. Firstly, in Figure 3, a scatter plot of "size" and "price" was created using the scatter function. Upon examination, outliers were identified and subsequently removed, resulting in a revised scatter plot.

Next, a scatter plot of "lot_size" and "price" was generated using the scatter function. Similarly, outliers were detected and eliminated, leading to an updated scatter plot.



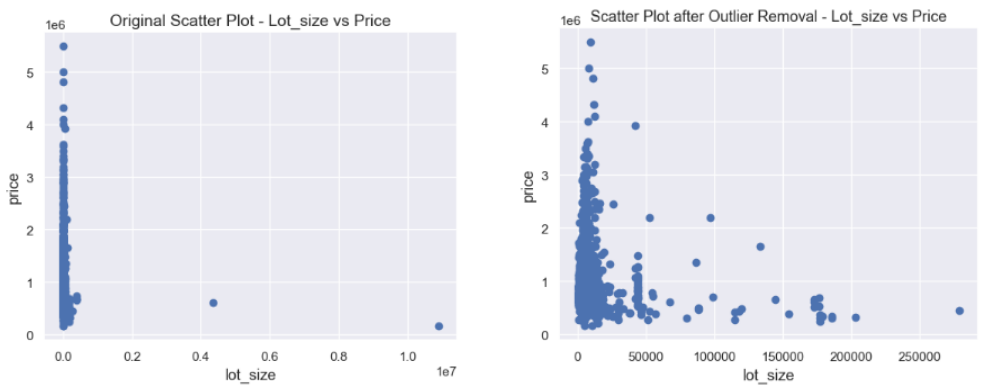


Figure 3. Scatter Plots Analyzing Relationships with the 'Price' Variable

After removing outliers, a heat map of the data was generated (Figure 4). The heat map illustrates the correlation between different variables in the dataset at this stage. It indicates that "beds", "baths" and "size" exhibit a moderate correlation with the "price" variable. This finding implies that if a multiple linear regression model is employed, the issue of multicollinearity needs to be taken into account. Multicollinearity refers to a high correlation between predictor variables, which can cause problems in interpreting the coefficients and stability of the model. To address this, appropriate techniques such as feature selection, dimensionality reduction, or regularization methods may be employed.

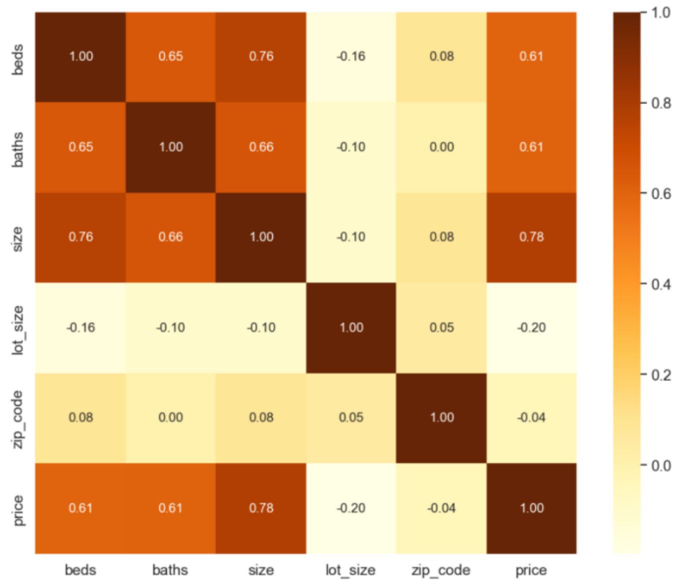


Figure 4. Heatmap illustrating the correlation between variables after logarithmic transformation

2.2. Data preprocessing

To obtain the final dataset, the training set and test set were combined. Upon inspecting the data, it was observed that there are missing values present in the dataset. Specifically, both "lot_size_unit" and "lot_size" contain missing values. In this case, the missing values in "lot_size" were filled with the median value of the "lot_size" column. Additionally, the missing values in "lot_size_unit" were represented as "sqft" units.

For variables that are not continuous, they were converted into categorical values. On the other hand, for variables that are not categorical, their skewness was examined. It was noted that four data features still exhibited significant skewness. To address this, a Box-Cox transformation was applied with a lambda (λ) value of 0.15 to transform the data with skewness greater than 0.75.

Finally, all columns in the dataset were individually one-hot encoded, and the results were stored back in the "all_data" dataset. By performing one-hot encoding, the categorical variables such as "zip_code," "size_units," and "lot_size_units" were converted into multiple binary feature columns.

2.3. Model selection and construction

In this study, several classical machine learning algorithms and ensemble learning models were employed to predict housing prices. Firstly, a linear regression model was chosen. To address the issue of multicollinearity, two commonly used linear regression methods, Lasso regression and ElasticNet regression, were applied. These methods incorporate regularization terms during the model training process, thereby reducing the number of features or driving their coefficients towards zero through penalty terms on the model parameters. For nonlinear regression, the KernelRidge algorithm was utilized. As for ensemble learning models, three typical approaches were selected: Random Forest, Gradient Boosting, and XGBoost. To facilitate performance comparison and further analysis, two combined model algorithms were developed. AveragingModels involved averaging the prediction results of multiple models, while StackingAveragedModels utilized the predictions from multiple base models as inputs for metamodels. These ensemble methods aim to enhance prediction accuracy and robustness.

These models are introduced respectively:

- **Lasso:**

Lasso regression, also known as L1 regularization, is a technique that introduces a penalty term into the traditional linear regression model. It aims to provide an interpretable model while effectively managing the risk of overfitting. By adding the penalty term, Lasso encourages sparsity in the model by forcing some coefficients to be exactly zero. This promotes feature selection and helps in obtaining a simpler and more understandable model.

The optimization objective for Lasso can be expressed as:

$$\text{Lasso} = (1/(2 \times n\_samples)) \times \|y - Xw\|^2\_2 + alpha \times \|w\|\_1 \tag{1}$$

The first item measures the squared difference between the true target. values (y) and the predicted values based on the feature matrix (X) and the weight vector (w). alpha is the regularization parameter that controls the strength of the penalty term. In this experiment,the value of alpha is set to 0.0005 for the Lasso regularization.

- **ElasticNet:**

ElasticNet is a regularization technique that combines the L1 regularization of Lasso and the L2 regularization of Ridge regression [5].

Minimizes the objective function:

The optimization objective for ElasticNet can be expressed as:

$$\begin{aligned}\text{ElasticNet} = {} & (1/(2 \times n\_samples)) \times \|y - Xw\|^2\_2 \\ & + alpha \times ((1 - l1\_ratio)) \times \|w\|\_2^2/2 + l1\_ratio \times \|w\|\_1\end{aligned} \tag{2}$$

In this experiment, the value of alpha is set to 0.0005. l1-ratio is set to 0.9, indicating a higher emphasis on L1 (Lasso) regularization. The first term measures the squared difference between the true target values (y) and the predicted values based on the feature matrix (X) and the weight vector (w). The second term represents the regularization term, which consists of two parts. The first part, $((1 - l1\_ratio)) \times \|w\|\_2^2/2$ _1 corresponds to the L2 regularization (Ridge), and the second part, $l1\_ratio \times \|w\|\_1$, corresponds to the L1 regularization (Lasso). The alpha parameter controls the overall strength of the regularization. By adjusting the values of alpha and $l1\_ratio$, ElasticNet allows for flexible regularization and can effectively handle situations where there are both correlated and uncorrelated features.

- **KernelRidge:**

Kernel ridge regression (KRR) combines ridge regression, which applies L2-norm regularization to linear least squares, with the use of kernels [6]. This integration allows KRR to model a function within a transformed space determined by the specific kernel. By employing various kernels, KRR can capture complex relationships within the original feature space. In this experiment, the value of alpha is set to 0.6 'polynomial' is chosen as the kernel function.The degree parameter for the polynomial kernel.is set to 2. The constant term in the polynomial kernel function is set to 2.5.

- **Random Forest:**

Random Forest is an algorithm that uses ensemble learning to combine multiple decision trees and perform predictions. It leverages the concept of bagging, where each tree is trained on a randomly selected subset of the training data with replacement. Additionally, Random Forest introduces additional randomness by randomly selecting a subset of features at each split point.

The algorithm's main objective is to reduce overfitting and improve prediction accuracy. By averaging individual tree forecasts, Random Forest decreases variance and provides more robust predictions compared to a single decision tree.

To assess the importance of variable x_j in predicting Y within a Random Forest, the computation involves summing the weighted impurity decrease $p(t)\Delta i(s_t, t)$for all nodes t where x_j is utilized [7]. This calculation is then averaged across all trees φ_m (indexed as m=1,...,M) in the forest:

$$\mathrm{Imp}(X_j) = \frac{1}{M}\sum_{m=1}^{M}\sum_{t\in\varphi_m} 1(j_t = j)[p(t)\Delta i(s_t, t)] \quad (3)$$

- **Gradient Boosting**

Gradient Boosting is an overall learning algorithm which combines several low learners to create a powerful predictive model [8].

The algorithm's primary objective is to minimize a predefined loss function by iteratively fitting new models to the residual errors of the ensemble. Each iteration focuses on the samples that were predicted incorrectly, assigning higher weights to these samples to prioritize their correct classification or prediction. This iterative process steadily enhances the predictive performance of the overall model by diminishing the remaining errors.

The optimization objective in Gradient Boosting can be formulated as follows:

$$F(x) = F_{M-1}(x) + \eta \times f\_m(x) \quad (4)$$

In this experiment, learning rate is set to 0.05, indicating a relatively small learning rate. The number of boosting stages or regression trees is set to 3000. $F(x)$ represents the ensemble model's prediction, $F_{M-1}(x)$ is the prediction from the previous ensemble (M-1) models, η is the learning rate that controls the contribution of each weak model, and $f\_m(x)$ is the weak learner's prediction for the current iteration (m).

- **XGBoost Regression**

XGBoost Regression, an efficient implementation of gradient boosting, has garnered substantial acclaim in both machine learning competitions and practical use cases [9].

Similar to Gradient Boosting, XGBoost Regression builds an ensemble model by sequentially adding weak learners, typically decision trees, to correct the errors made by previous models. However, XGBoost Regression introduces several enhancements, including regularization techniques, parallel processing, and tree pruning, to improve both the accuracy and efficiency of the model [10].

The optimization objective in XGBoost Regression is defined as follows:

$$L(\varphi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \quad (5)$$

In XGBoost Regression, the optimization objective is defined using a customizable differentiable convex loss function$l$that quantifies the difference between the predicted values $\hat{y}_i$ and the target values y_i. The regularization term Ω is introduced to penalize the complexity of the model, discouraging overfitting. In this experiment, the value of the L1 regularization term on the weights is 0.4640, while the L2 regularization term on the weights is set to 0.8571.

Due to the involvement of functions as parameters, XGBoost trains equation in an additive manner, as traditional optimization methods in Euclidean space are not applicable for solving this problem [8].

- **AveragingModels**

AveragingModels is an ensemble learning technique that combines multiple individual models by averaging their predictions. By training multiple models, AveragingModels captures different aspects and perspectives of the data, reducing the risk of relying too heavily on the biases of a single model.

In this paper, AveragingModels is applied by averaging the predictions of three base models: ElasticNet (ENet), Gradient Boosting (GBoost), and Kernel Ridge Regression (KRR). The individual predictions of these base models are combined using simple averaging. Additionally, a meta-model, specifically Lasso, is used to further refine the final prediction. This ensemble approach leverages the diverse strengths of the base models and combines their predictions to achieve improved overall performance.

- **StackingAveragedModels**

StackingAveragedModels is an ensemble learning technique that integrates multiple individual models hierarchically. It aims to leverage the strengths of different models by using their predictions as inputs for a meta-model, which produces the final prediction.

In StackingAveragedModels, multiple base models, such as ElasticNet (ENet), Gradient Boosting (GBoost), and Kernel Ridge Regression (KRR), are trained on the same dataset or different subsets of the data. Each base model generates predictions for the target variable. The following inputs need to be provided for the implementation of StackingAveragedModels.: base_models: A list containing multiple base models that will be used to generate predictions. meta_model: The meta-model used to combine the predictions from the base models.n_folds: The number of folds for cross-validation, with a default value of 5.

3. Result

3.1. Experimental Details

The experiment encompassed the importation of the Kaggle dataset into the Jupyter platform, followed by the execution of a sequence of Python code operations to obtain outcomes for multiple machine learning models. The experiment was conducted employing an NVDIA GeForce RTX 3060 Laptop GPU and an AMD Ryzen 7 5800H CPU with Radeon Graphics.

The training parameters were determined using cross-validation, a common technique in machine learning, to ensure better performance and generalization of the house price prediction models. For this experiment, a cross-validation fold of 5 was utilized (n_folds = 5), indicating that the dataset was divided into five subsets for training and validation purposes. The choice of this value aimed to strike a balance between model complexity and computational efficiency.

3.2. Evaluation metrics

In evaluating these models, the chosen parameters for assessment are RMSLE (Root Mean Squared Logarithmic Error) and R^2 (Coefficient of Determination). The logarithmic transformation of the original data renders RMSE (Root Mean Squared Error) and MAPE (Mean Absolute Percentage Error) less informative on the logarithmic scale. Therefore, only RMSLE and R^2 were selected as the evaluation metrics for the models.

The two evaluation methods are introduced:

Root Mean Squared Log Error (RMSLE):

$$\mathrm{RMSLE} = \sqrt{\frac{1}{\mathrm{N}}\sum_{i=1}^{\mathrm{N}}[\log(y_i + 1) - \log(\hat{y}_i + 1)]^2} \tag{6}$$

RMSLE serves as a widely adopted metric for evaluating regression model performance. In this equation, 'N' represents the number of samples, 'y_i' represents the predicted values by the model, and '$\hat{y}_i$' represents the actual target variable values. A lower value of RMSLE indicates better predictive capability of the model

R^2:

R2 (R-squared) serves as a commonly adopted metric to assess the fitting of regression models to the data, representing the proportion of the variance in the dependent variable explained by the model.

$$R^2 = \frac{\sum_{i=1}^{N}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

where $\sum_{i=1}^{N}(y_i - \bar{y})^2$, $\sum_{i=1}^{N}(\hat{y}_i - \bar{y})^2$, $\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ denote the total sum of squares (TSS), the explained sum of squares (ESS) and residual sum of squares (RSS) respectively.

3.3. Model Evaluation

Table 2。 the performance of different models

Model	**RMSLE**	**R^2**
Lasso	0.2592	0.7293
ElasticNet	0.2592	0.7294
KernelRidge	0.2540	0.7383
Random Forest	0.2484	0.7473
Gradient Boosting	0.2343	0.7746
XGBoost Regression	0.2353	0.7765
AveragingModels	0.2434	0.7593
StackingAveragedModels	0.2328	0.7771

Among all the models, due to the application of the Box-Cox transformation on the dataset, the Linear Regression model exhibited overfitting, resulting in significantly large values for R-squared (R2) and RMSLE. To address this issue, these models introduce constraints through regularization techniques to alleviate overfitting (Table 2).

Comparing the results, we can make the following observations:

Based on the evaluation results (), StackingAveragedModels stand out as the top-performing models with the lowest RMSLE values (0.2328) and highest R^2 values (0.7771), indicating better predictive performance in terms of minimizing the logarithmic error. By leveraging the strengths and complementary characteristics of these diverse models, StackingAveragedModels can achieve improved predictive performance.

Lasso, ElasticNet, and KernelRidge demonstrate similar performance, with slightly higher RMSLE values but still within a close range. The relatively underperforming nature of Lasso, ElasticNet, and KernelRidge could be attributed to several factors. One possible reason is that Lasso and ElasticNet rely on L1 regularization, which can lead to feature selection and result in a more simplified model that may not capture all the important relationships in the data. This feature selection process could potentially exclude relevant variables, leading to suboptimal performance. On the other hand, KernelRidge may not be as flexible in capturing complex non-linear relationships present in the data, which can limit its predictive capability compared to other algorithms.As an ensemble model, Random Forest exhibit relatively higher RMSLE values, implying comparatively lower accuracy in predicting house prices. However, they still achieve reasonable results in terms of R-squared scores, indicating a moderate ability to explain the variance in the data. The subpar performance of the model may be attributed to the utilization of default hyperparameter configurations without exploring different parameter combinations to identify the optimal settings.

With relatively low RMSLE values and higher R-squared scores, Gradient Boosting and XGBoost Regression exhibit a satisfactory level of predictive capability. As a combination model, AveragingModels performs slightly worse than the top-performing models, with a higher RMSLE value. The main reason is that these model averages the results of KRR (KernelRidge) and ElasticNet models, which individually do not yield very optimal results. However, compared to the results achieved by KRR and ElasticNet as standalone models, AveragingModels optimizes the outcomes.

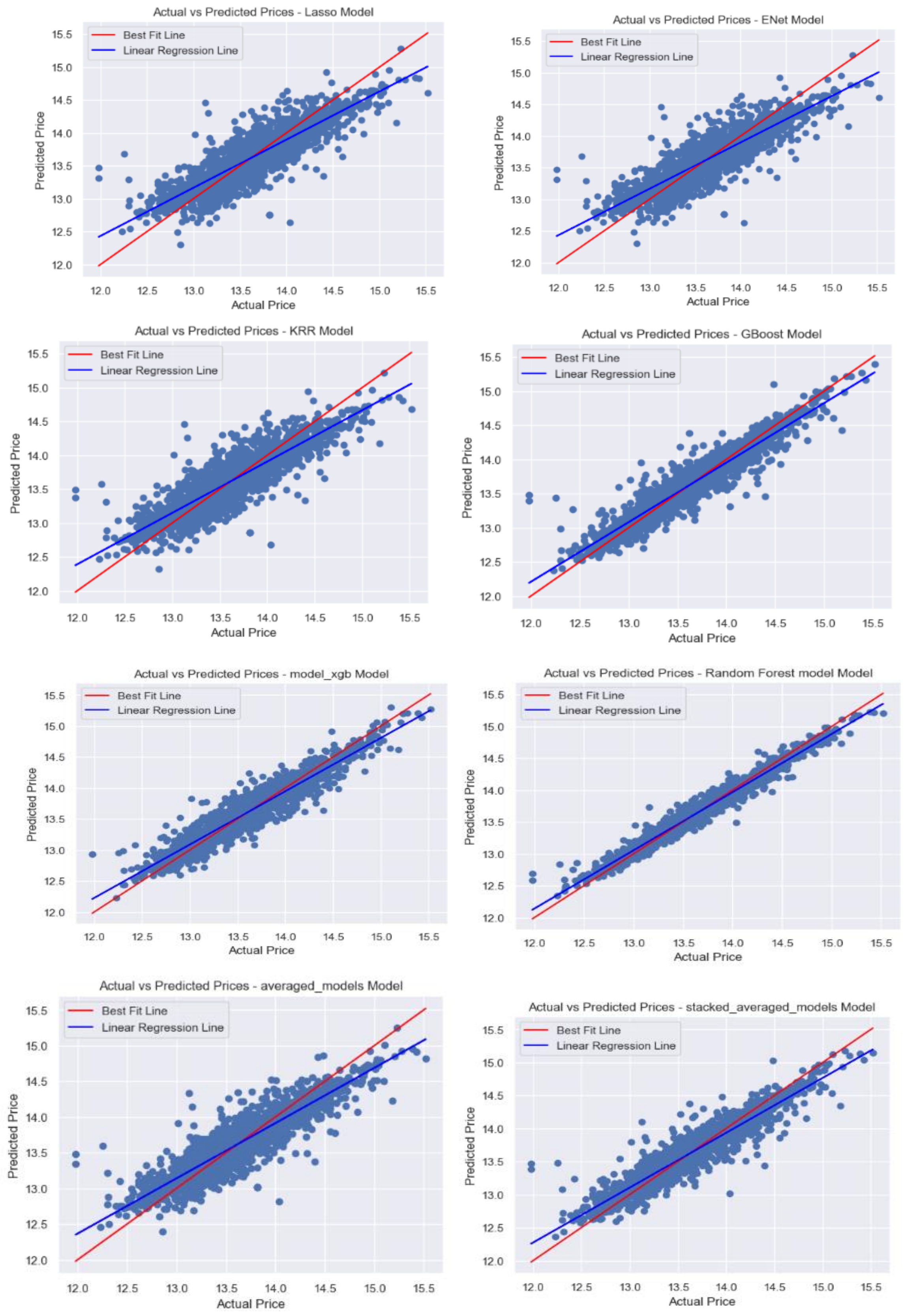


Figure 5. Actual Price vs. Predicted Price for All Models

3.4. Visual Analysis of Model Performance

To further analyze the performance of each model, a visual examination of the relationship between the actual and predicted values was conducted. In Figure 5, each point represents a record from the test set. The x-coordinate of the point represents the actual home value, while the y-coordinate represents the corresponding predicted home price. The red line in the plot represents the best fit line, indicating accurate predictions when the points overlap closely. The blue lines in each chart depict the linear regression lines that show the overall trend of the results, reflecting the relationship between the predicted and true values. This visual analysis allows for a comprehensive understanding of how well the models capture the actual variation in home prices. By examining the alignment of the points with the best fit line and the overall trend portrayed by the blue lines, insights can be gained regarding the accuracy and consistency of the predictions made by each model.

4. Conclusion

In this study, the aim was to predict housing prices in Seattle by employing eight different machine learning methods: Lasso, ElasticNet, KernelRidge, Random Forest, Gradient Boosting, XGBoost Regression, AveragingModels, and StackingAveragedModels. The performance of these models was compared using RMSLE and R2 as evaluation metrics, with a focus on analyzing the differences between the algorithms.

Among the evaluated models, Among the evaluated models, StackingAveragedModels, which combines the predictions of ENet, GBoost, and KRR through weighted averaging, exhibited the highest performance, achieving an RMSLE of 0.2328 and an R2 of 0.7771. Both of these metrics represent the best performance among all the experimental models. StackingAveragedModels can effectively leverage the advantages of each individual basic model, thereby improving the overall predictive capability. This is achieved by taking a weighted average of the prediction results from multiple basic models. By doing so, StackingAveragedModels combines the strengths of each model and enhances the overall prediction performance. This innovative ensemble model, which combines the predictions of multiple base models, was found to effectively reduce errors and yield superior results in predicting housing prices in Seattle.

This finding highlights the effectiveness of StackingAveragedModels in accurately predicting housing prices. This model demonstrates strong performance based on the selected evaluation metrics, which further supports their potential application in real-world scenarios. In the future, advancements in housing price prediction may benefit from applying deep learning techniques, which have the potential to provide more accurate forecasts. Therefore, further research and exploration into integrating deep learning methods into housing price prediction are crucial directions for future studies.

References

[1] Truong Q, Nguyen M, Dang H, et al. Housing price prediction via improved machine learning techniques. Procedia Computer Science, 2020, **174:** 433-442.
[2] Adetunji A B, Akande O N, Ajala F A, et al. House price prediction using random forest machine learning technique. Procedia Computer Science, 2022, **199**: 806-813.
[3] Cervero R. America's suburban centers: the land use-transportation link. Routledge, 2018.
[4] Cortunhas, S., “House Price Prediction Sattle” Kaggle Inc, (2023). https://www. kaggle.com/datasets/samuelcortinhas/house-price-prediction-seattle
[5] Huang Y, Schell C, Huber T B, et al. Traction force microscopy with optimized regularization and automated Bayesian parameter selection for comparing cells. Scientific reports, 2019, **9(1)**: 539.
[6] Suganthan P N. On non-iterative learning algorithms with closed-form solution. Applied Soft Computing, 2018, **70**: 1078-1082.
[7] Wu X, Yang B. Ensemble Learning Based Models for House Price Prediction, Case Study: Miami, US, 2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE). IEEE, 2022: 449-458.

[8] Sahin E K. Assessing the predictive capability of ensemble tree methods for landslide susceptibility mapping using XGBoost, gradient boosting machine, and random forest. SN Applied Sciences, 2020, **2(7):** 1308.
[9] Chen T, Guestrin C. Xgboost: A scalable tree boosting system, Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016: 785-794.
[10] Bhattacharya S, Maddikunta P K R, Kaluri R, et al. A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU. Electronics, 2020, **9(2)**: 219.