A Comparative Analysis of Machine Learning Models for House Price Prediction

Anasah Wawem Christopher

January 12, 2026

1 Introduction

House price prediction is an essential task in real estate analytics, where accurate prediction supports buyers and sellers, as well as policy decisions. Traditional statistical methods for this objective includes OLS regression, which provides baseline predictive capability, but sometimes does not work that well with nonlinear and complex data patterns usually seen in house prices. In the recent literature, machine learning approaches are usually applied to improve predictive accuracy, especially those using ensemble-based methods.

(Madhuri et al., 2019) demonstrated that Gradient Boosting Regression outperformed traditional linear models in the King County dataset and also concluded that advanced algorithms of machine learning perform better than traditional regression in predicting property prices. (Choy & Ho, 2023) also established that Random Forest outperformed OLS in predicting Hong Kong home prices, especially following feature engineering. Similarly, (Zhang, 2023) reported that XGBoost achieved higher accuracy than Multiple Linear Regression, which is also confirmed by (Jiang, 2025) that Random Forest and XGBoost handled large-scale nonlinear real estate data more effectively than linear models. (Weng, 2022) further demonstrated that, when paired with appropriate preprocessing and feature selection, boosting techniques like GBDT and XGBoost yield the lowest error metrics.

These findings led to a comparative analysis of five models: OLS Regression, Random Forest Regression, Gradient Boosting Regression, XGBoost, and LightGBM to assess their efficacy in predicting house prices. Despite the adaptability of machine learning models, the challenge lies in selecting the best algorithm and processing data effectively to reveal hidden patterns. The study aims to identify the model that provides the most reliable and accurate price forecasts while recognizing complex relationships within housing data.

2 Methodology

This study employed five machine learning algorithms to model and predict the target variable: Ordinary Least Squares regression (OLS), Random Forest Regression, Gradient Boosting Regression, XGBoost, and LightGBM. These models represent various learning approaches, facilitating a comprehensive comparison of linear, ensemble, and boosting-based methodologies.

2.1 Ordinary Least Squares (OLS) Regression

The baseline model was Ordinary Least Squares (OLS) Regression because of its simple nature of use and interpretability. OLS estimates the parameter vector β by minimizing the sum of squared residuals:

$$\hat{\beta} = (X^\top X)^{-1} X^\top y \tag{1}$$

The model prediction is given by:

$$\hat{y} = X\hat{\beta} \tag{2}$$

The objective function minimized is:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

The assumptions of this model are linearity, independence of errors, homoscedasticity, and absence of multicollinearity, making it an excellent place to start before implementing more advanced models.

2.2 Random Forest Regression

This ensemble learning technique reduces variance and avoids overfitting by building several decision trees and averaging their predictions. Given T trees, the final prediction is:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} f_t(x) \tag{4}$$

Under this strategy, the model is able to capture nonlinear relationships and feature interactions since each tree is formed using a randomly chosen subset of features and a bootstrap sample of the training data.

2.3 Gradient Boosting Regression

Gradient Boosting Regression builds decision trees sequentially, each of which seeks to correct the residual errors of the model that came before it. Model updates are produced iteratively as:

$$F_m(x) = F_{m-1}(x) + \nu h_m(x) \tag{5}$$

where ν is the learning rate and $h_m(x)$ is the m-th tree fitted to the negative gradient of the loss function.

Complex nonlinear patterns can be predicted using this method, but overfitting must be avoided by properly adjusting it.

2.4 XGBoost (Extreme Gradient Boosting)

XGBoost is an enhanced form of gradient boosting that features parallelization, regularization, and effective tree construction. Its objective function is:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{6}$$

The regularization term is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{7}$$

where T is the number of leaves and w_j are the weights of the leaf. All of the boosted trees are combined in the final prediction:

$$\hat{y} = \sum_{k=1}^{K} f_k(x) \tag{8}$$

This tree-based machine learning algorithm is widely used for its accuracy, speed, and ability to handle missing values.

2.5 LightGBM

LightGBM is a gradient boosting framework that has been improved for scalability and computing efficiency. It creates trees leaf-wise using feature binning based on histograms. The benefit of splitting for a node is computed as follows:

$$\text{Gain} = \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right) - \gamma \tag{9}$$

where G_L and G_R are the gradients and H_L and H_R the Hessians of the left and right child nodes. This algorithm is particularly efficient for huge datasets due to its quick training and ability to model complicated structures.

3 Experiments

3.1 Dataset Overview, Cleaning, and Preparation

The data set used in this study is housing data from Kaggle.com made up of 21,613 houses built between 1900 and 2015 with 20 features that are about structure, location, grade, and year. There are no missing values found in the dataset. All the variables in the data set are numeric. The maximum house price is $7,700,000, the minimum price is $75,000 and the average price is $540,088.6. To check for duplicate IDs, the year, month, and day were taken out of the "date". However, the "condition" variable verifies that the residences were renovated and resold at a different price, as indicated by the duplicated ID. There were outliers present in the data set. The data was split into 80% for training and 20% for testing.

3.2 Feature Exploration

Figure 1 shows the correlation plot of the features used in the building of the models. There is a strong positive correlation among sqft_living, bathrooms, grade, sqft_above, and sqft_living 15 and sqft_lot.

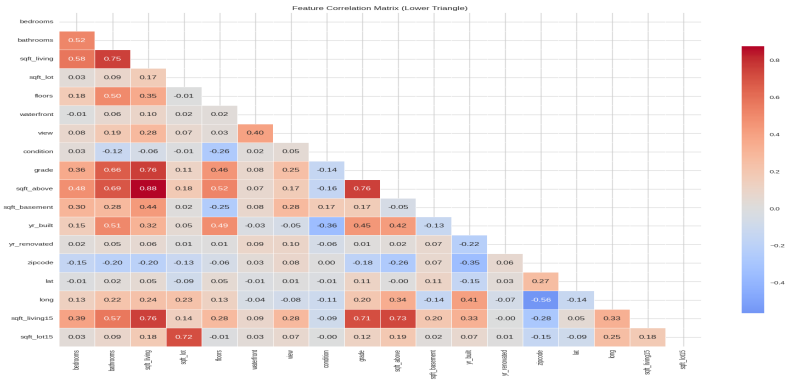
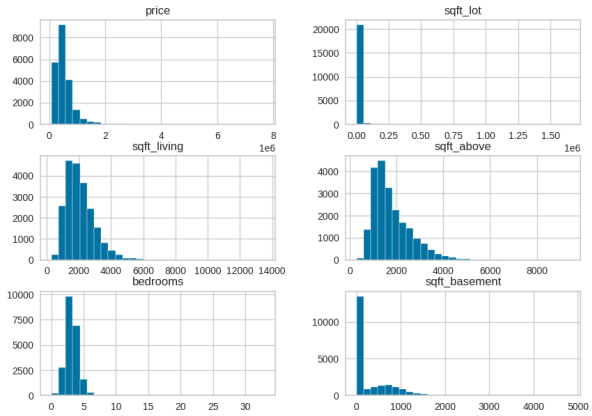
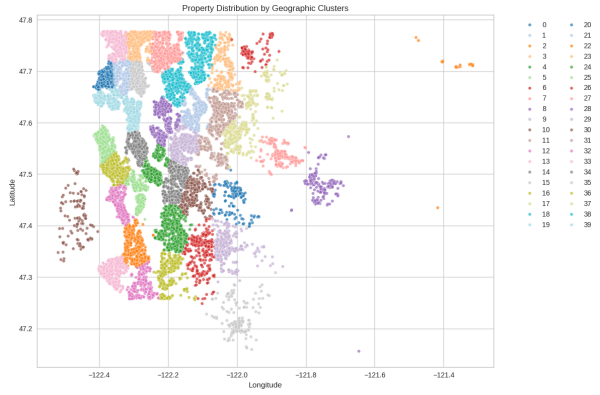


Figure 1: Correlation of Features

Figure 2a shows the histogram plot of some variables. There is skewness in the data with most properties being at the lower end of the scale for price and all features (dense clusters in the bottom-left of most plots). This is typical for real estate data. The skewness in the data can be perfectly handled by tree-based regression.
Figure 2b shows how properties (houses) are grouped into clear geographic clusters based on their latitude and longitude using K-means. Each cluster represents a unique neighborhood-like area with comparable spatial features. Although smaller or isolated clusters represent more remote locales, dense clusters show densely occupied dwelling areas. All things considered, this clustering successfully divides the housing market into significant location-based groups, which is a useful representation for building models.



(a) Histogram Plot of Some Important Variables



(b) Geographic Clusters of Residential Properties Based on Latitude and Longitude

Figure 2: Exploratory Data Analysis of Housing Data

3.3 Performance Metrics

The following metrics were used in the study:

Mean Absolute Error (MAE): is a measurement of the average size of errors between paired observations that is typically used to assess how well a model predicts real values.

Root Mean Absolute Error (RMSE): is a metric with the same units as the data being measured that is used to quantify the difference between predicted and actual values. It is especially sensitive to outliers and huge errors because it is computed by taking the square root of the average of squared errors.

R-Squared (R^2) Score: measures the proportion of variation in a dependent variable that is predictable from the independent variable(s) in a regression model. It provides information about the goodness of fit of a regression line; the value ranges from 0 to 1. A score of 1 represents a perfect fit, while 0 indicates that the model explains no variability.

Mean Absolute Percentage Error (MAPE): Mean Absolute Percentage Error (MAPE) is a metric used to assess prediction accuracy by calculating the average absolute difference between predicted and actual values as a percentage of actual values. It averages the absolute percentage errors for each data point, providing insights into forecast accuracy, especially when comparing scaled models. A lower MAPE indicates better accuracy, though it can be misleading when actual values are extreme or near-zero.

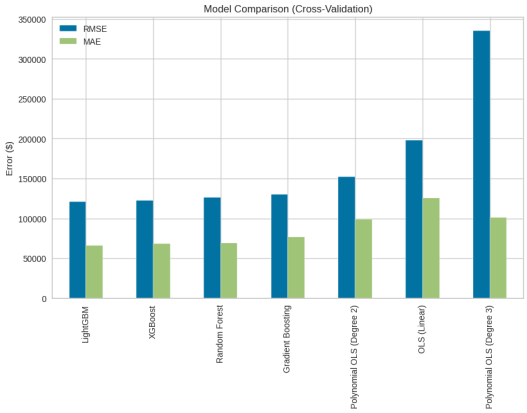
3.4 Cross Validation Performance

Table 1 provides the performance of the models during cross-validation using the four metrics with the train data (80 %) partitioned into 10 folds. LightGBM turns out to be the best and OLS (linear) turns out to be the worst model. The OLS model was increased to a polynomial of degree 3 but led to overfitting. The degree 2 polynomial has better performance than the linear model, hence served as the baseline model.

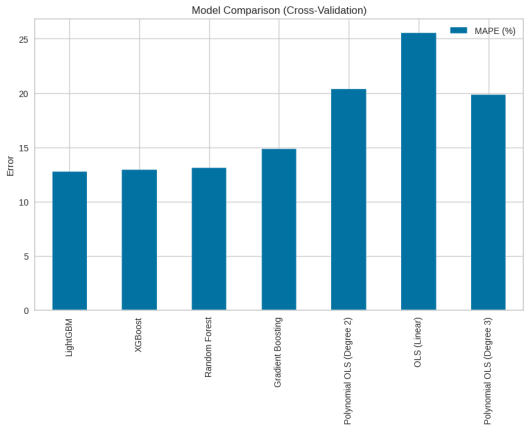
Table 1: Model Performance Comparison of Cross Validation

Model	RMSE	MAE	MAPE (%)	R^2
LightGBM	120 646.271	66 455.466	12.782	0.888
XGBoost	122 631.882	68 069.292	12.931	0.884
Random Forest	126 692.749	69 413.352	13.143	0.876
Gradient Boosting	130 439.824	76 944.412	14.830	0.868
Polynomial OLS (Degree 2)	152 254.124	98 966.163	20.370	0.820
OLS (Linear)	197 812.532	125 237.386	25.572	0.699
Polynomial OLS (Degree 3)	335 638.401	101 040.295	19.854	−0.379

Figure 3 shows the visualization of the cross-validation error in Table 1 above. Figure 3a is a visualization of MAE and RMSE, while Figure 3b visualizes MAPE.



(a) Visualization of Cross-Validation MAE and RMSE



(b) Visualization of Cross-Validation MAPE

Figure 3: Visualization of Cross Validation Errors

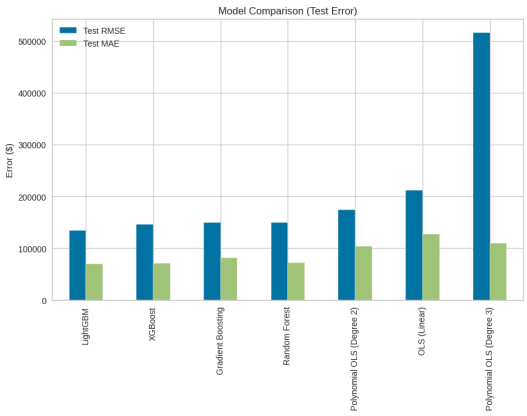
3.5 Held-Out Performance (Test Performance)

Table 2 shows the performance of the models on the test data after rebuilding on the entire 80% and tested on the 20% held out. From the table, LightGBM still stands the best with slightly increased in the error.

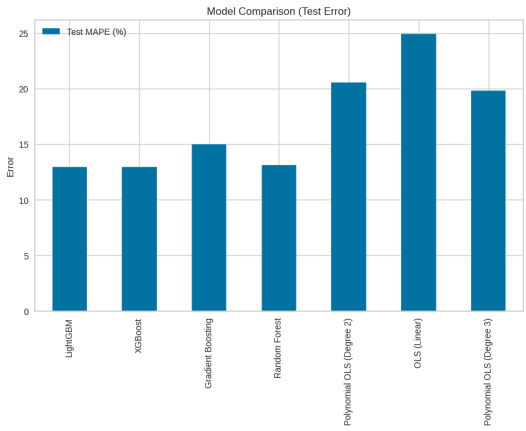
Table 2: Test Set Performance Comparison

Model	Test RMSE	Test MAE	Test MAPE (%)	Test R^2
LightGBM	134 036.165	69 968.626	12.953	0.881
XGBoost	145 976.629	71 070.488	12.954	0.859
Gradient Boosting	149 446.733	81 301.751	15.013	0.852
Random Forest	149 823.651	72 890.952	13.147	0.852
Polynomial OLS (Degree 2)	174 441.196	104 219.056	20.546	0.799
OLS (Linear)	212 539.472	127 493.630	24.954	0.701
Polynomial OLS (Degree 3)	516 639.626	109 450.204	19.852	−0.766

Figure 4 shows the visualization of the test errors in Table 2 above. Figure 4a is a visualization of MAE and RMSE, while Figure 4b visualizes MAPE.



(a) Visualization of Test MAE and RMSE



(b) Visualization of Test MAPE

Figure 4: Visualization of Test Errors

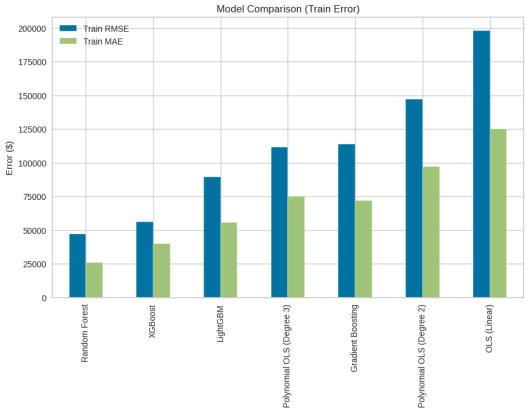
3.6 Train Performance

Table 3 provides the training errors of the various models after building them on the training data and tested on the same data using the metrics. The Random Forest model achieved very small training error (low bias) but large test error (high variance) compared to baseline, hence overfitting. LightGBM has low bias and low variance compared to the baseline error.

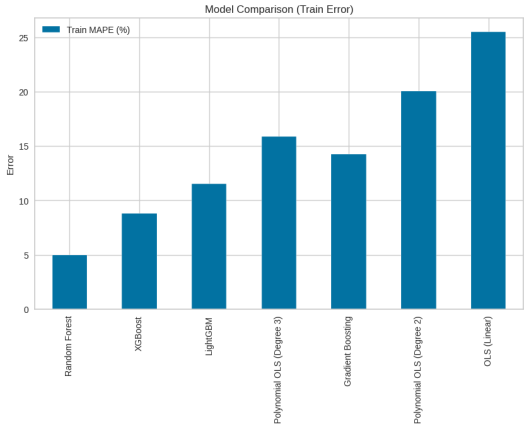
Table 3: Training Set Performance Comparison

Model	**Train RMSE**	**Train MAE**	**Train MAPE (%)**	**Train R^2**
Random Forest	47 204.496	25 870.578	4.957	0.983
XGBoost	56 197.042	39 933.492	8.817	0.976
LightGBM	89 303.509	55 505.277	11.513	0.939
Polynomial OLS (Degree 3)	111 625.628	74 981.698	15.861	0.905
Gradient Boosting	113 900.904	71 977.835	14.280	0.901
Polynomial OLS (Degree 2)	147 256.004	96 994.649	20.074	0.834
OLS (Linear)	198 272.254	125 033.643	25.541	0.699

Figure 5 shows the visualization of the train errors in Table 3 above. Figure 5a is visualization of MAE and RMSE while, Figure 5b visualizes MAPE.



(a) Visualization of Train MAE and RMSE



(b) Visualization of Train MAPE

Figure 5: Visualization of Train Errors

3.7 Hyperparameter Tuning on Top Two

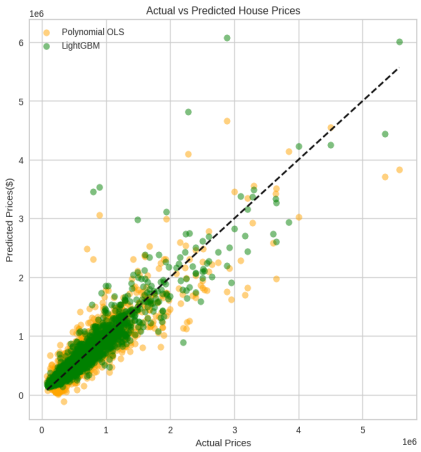
Table 4 presents the train, test, and cross-validation errors of the top two models after hyperparameter tuning. From the table, the cross-validation error for LightGBM and XGBoost has decreased. The train error for LightGBM has decreased, but there is an increment in XGBoost train error. The test error has reduced for both models. Overall, there is an improvement after tuning the parameters. The optimized parameters for LightGBM are: 1000 individual decision trees at the boosting stage, 0.01 learning rate, 8 maximum levels allowed in each individual decision tree, 0.6 of training data randomly sampled without replacement to train each new tree, and 0.6 of features randomly sampled per tree.

Table 4: Model Performance Comparison After Hyperparameter

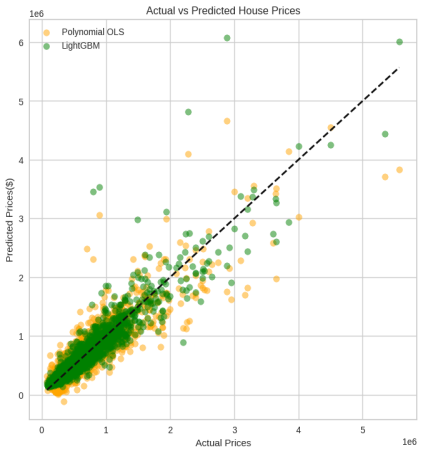
	Train		CV		Test	
Model	**MAPE%**	**R^2**	**MAPE%**	**R^2**	**MAPE%**	**R^2**
LightGBM	8.722	0.971	12.190	0.971	12.262	0.865
XGBoost	9.422	0.971	12.290	0.876	12.416	0.876

3.8 Prediction Plot of Best Model and Baseline Model (Before and After Hyperparameter Tuning)

Figure 6 shows the predictions using the best and baseline models. Before hyperparameter tuning, the baseline model shows significant overfitting, showing large gaps between predicted values indicating the model isn't capturing continuous relationships well. After hyperparameter tuning, the baseline model has much better alignment with the diagonal line compared to pre-tuning and overfitting has also been reduced. The final model (LightGBM) has improved from good to excellent performance with tighter error margins.



(a) Prediction with Best and Baseline Models Before Hyperparameter Tuning

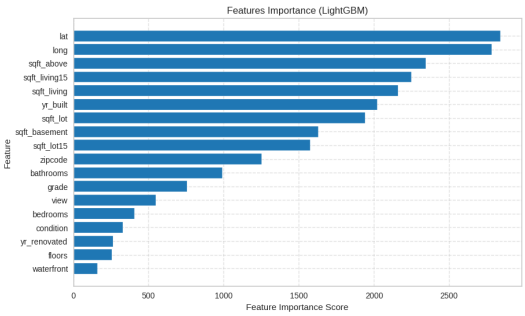


(b) Prediction with Best and Baseline Models After Hyperparameter Tuning

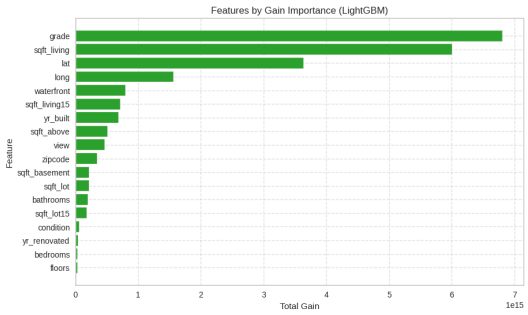
Figure 6: Prediction with Best and Baseline Models

3.9 Feature Importance and Features by Gain of Final Model

Figure 7 shows feature contribution to the best model. Figure 7a shows which features do the model rely on most often to make decisions. From the plot, lat and long are used most frequently, sqft_above and sqft_living15 are also heavily used. The grade drops to the middle of the pack despite being highly influential. Figure 7b shows how much each feature improves the model's accuracy (reduces loss) across all trees. From the plot, grade and sqft_living are by far the most powerful predictors, location features (lat, long), and waterfront are also very influential.



(a) Features Importance (LightGBM)



(b) Features by Gain Importance (LightGBM)

Figure 7: Prediction with Best and Baseline Models

4 Conclusion

This study has shown that machine learning models significantly outperform traditional linear approaches in predicting home prices, particularly when complex and nonlinear housing data are involved. Among all five models tested, LightGBM kept producing the lowest prediction errors across all training, cross-validation, and test sets, hence confirming its robustness and ability to generalize well. XGBoost performed powerfully as well, especially after its hyperparameter tuning, hence further manifesting the effectiveness of boosting techniques in real estate analytics. Tree-based models such as Random Forest and Gradient Boosting captured important nonlinear interactions but showed some overfitting compared to LightGBM. Overall, the results confirm the idea that advanced ensemble and boosting algorithms generate more accurate and reliable house price predictions.

5 Contribution

I played the role of a leader in the team. I led and facilitated our meetings. I searched for the data for the project. I built all the models after the data wrangling by another member, compared them, and plotted all the graphs. I also prepared the presentation slides for the final presentation.

References

Madhuri, C. R., Anuradha, G., & Pujitha, M. V. (2019, March). House price prediction using regression techniques: A comparative study. In *2019 International Conference on Smart Structures and Systems (ICSSS)* (pp. 1–5). IEEE.

Choy, L. H. T., & Ho, W. K. O. (2023). The use of machine learning in real estate research. *Land, 12*(4), 740. https://doi.org/10.3390/land12040740.

Zhang, L. (2023). Housing price prediction using machine learning algorithm. *Journal of World Economy, 2*(3).

Jiang, H. (2025). Machine learning models for predicting second-hand house prices: A comparative study. In *Proceedings of the 2025 International Conference on Big Data, Artificial Intelligence and Digital Economy (BDAIE '25)* (pp. 99–106). Association for Computing Machinery. https://doi.org/10.1145/3767052.3767068.

Weng, W. (2022). Research on the house price forecast based on machine learning algorithm. *BCP Business Management, 32*. Retrieved from https://bcpublication.org.