Predictive Modeling and Analysis of Housing Prices

Puja Saha
University of Guelph
Engineering Department
psaha03@uoguelph.ca

Parya Abadeh
University of Guelph
Engineering Department
pabadeh@uoguelph.ca

Om Bhosale
University of Guelph
Engineering Department
obhosale@uoguelph.ca

.

Sanchi Sanchi
University of Guelph
Engineering Department
ssanchi@uoguelph.ca

***Abstract*— One of the fundamental needs for individuals worldwide is housing and accommodation. However, the challenge of acquiring a home has become prevalent due to financial constraints in various countries. This study addresses the difficulties faced by individuals globally in finding and purchasing a house. Our objective is to enhance predictive accuracy for house prices by employing various regression techniques, including Random Forest, KNN, Linear Regression, Lasso Regression, Gradient Boosting Regression, Support Vector Regression and Ridge Regression. This research aims to provide valuable insights that can benefit prospective buyers and real estate advisors in comparing properties or making informed decisions on new acquisitions. Additionally, the study considers pertinent factors influencing housing costs, such as physical conditions, concept, and location.**

Keywords—regression, house prediction, home, price.

I. INTRODUCTION

This report explores the latest research predictions and trends in conjunction with forecasting economic dynamics. The primary focus of the project, "Forecasting on House Price," is to optimize house price predictions by employing suitable algorithms and determining the most effective approach with a minimal error rate. The significance of this endeavor lies in its relevance to a widespread and impactful concern – the buying and selling of homes. As analysts in the realm of house prices, this investigation offers valuable insights into the housing market, empowering individuals to make well-informed decisions.

The analysis presented in this paper draws extensively from datasets sourced from Kaggle, a renowned platform recognized for its accurate and comprehensive datasets. Our objective is to achieve the most accurate house price predictions, leveraging techniques such as Linear Regression (LR), Ridge Regression (RR), Lasso Regression (LR), Gradient Boosting Regression (GBR), Random Forest Regression (RFR), Support Vector Regression (SVR) , and K Nearest Neighbors (KNN). The performance of each algorithm was evaluated based on scores, Mean Square Error (MSE) and R-Squared. The implementation of these techniques is facilitated through Python programming, and the corresponding code is accessible on GitHub at https://github.com/pariyaab/House_Prices_Regression_Models.

Figure 1 visually represents the data flow and processing intricacies involved in employing diverse regression techniques, encapsulating the essence of our methodology in predicting house prices with precision.

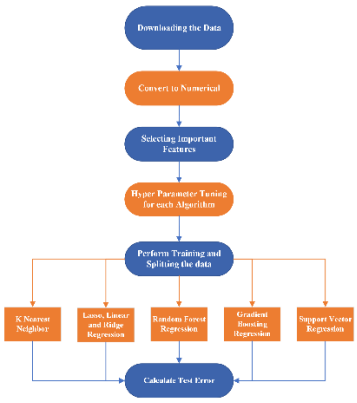

Fig1 - Data Flow Diagram

II. RELATED WORKS

In this study [1] , the authors conducted a comparative analysis of various regression techniques for house price prediction. The objective of the research was to forecast house prices based on financial provisions and aspirations of non-house holders. The study utilized different regression techniques including Multiple Linear Regression, Ridge Regression, LASSO Regression, Elastic Net Regression, Ada Boosting Regression, and Gradient Boosting Regression. The authors collected house sales data from the King County dataset. The performance of each algorithm was evaluated based on scores, Mean Square Error (MSE), and Root Mean Square Error (RMSE). The results showed that the Gradient Boosting Regression algorithm had the highest accuracy for house price predictions. The study aimed to assist sellers in estimating the selling cost of a house accurately and help people predict the right time to buy a house. The research can

be cited as a valuable reference for predicting house prices using regression techniques.

According to [5], the crucial variables in predicting house prices are square footage, number of bathrooms, and number of bedrooms. Their study indicates a 2.6% increase in house value for every 100-square-foot expansion, and a 10.4% to 13.7% rise in price with an additional bedroom or bathroom. In a broader context, previous studies have extensively used 19 attributes to assess house prices.

Recent research emphasizes locational and structural attributes. Notably, [7] underscores location as a key price predictor, with attributes like hospital access, schools, campuses, and leisure parks influencing house prices. Structural attributes, including the number of bedrooms, bathrooms, and total square footage, play a significant role, aligning with [8]'s findings on attributes impacting home selling rates. Neighborhood attributes, such as low crime rates and pleasant scenery, are crucial determinants of house prices, according to [6]. Despite limited focus, economic attributes like individual income and construction expenses also exert a major impact on house prices, as acknowledged by [5] and [6]. The evaluation of these attributes is pivotal, aligning with the study's primary research question. Subsequently, data mining methodologies, particularly predictive models like support vector regression and artificial neural networks, are employed to estimate house prices. Notably, [8] identifies XGBoost as the optimal model, providing the lowest RMSE value among various classifiers, addressing the study's second research question.

In 2017, with support vector regression used by a group of researchers to predict real estate prices and found its efficiency was quite impressive in this task. [10]. But with different loss functions and several methods such as linear regression, NN regression, random forest etc. But eventually found that gradient boosting showed the least amount of error for the task of housing price prediction. [11]

In a study conducted by Adetunji et al. (2022) [9], the authors explored the application of Random Forest machine learning technique for house price prediction using the UCI Machine Learning Repository Boston housing dataset. The dataset comprised 506 entries with 14 features related to various aspects of homes in different suburbs of Boston. The study aimed to predict price variance, considering it as a classification issue.

The authors highlighted the limitations of using the House Price Index (HPI) alone for predicting individual housing prices, emphasizing the need for additional information beyond HPI due to the diverse factors influencing house prices, such as location, city, and population. They specifically employed the Random Forest algorithm for its ability to handle complex problems through ensemble learning. The methodology involved data collection, exploration, and preprocessing stages, including the normalization of numerical values and encoding of categorical values. The Random Forest regression model was then developed, and the performance was evaluated using metrics such as Mean Absolute Error (MAE), R^2 (Coefficient of Determination), and Root Mean Square Error (RMSE).

In our related work, we adopted a similar approach, leveraging the Random Forest Regressor as a machine learning technique for predicting housing prices. Like Adetunji et al., we conducted a comprehensive exploration of hyperparameters and model performance, ensuring the robustness of our predictive model. Our study aligns with the findings of Adetunji et al., contributing to the body of knowledge on the application of Random Forest in the domain of housing price prediction.

III. Methodology

Before In this section, we aim to discuss the methodologies that have been implemented, shedding light on their underlying mechanisms and providing a comprehensive understanding of their application.

A. K Nearest Neighbours (KNN)

The K Nearest Neighbors (KNN) algorithm is versatile, serving purposes in both classification and regression tasks. Its fundamental mechanism involves modeling each data point in the space. When new data is introduced, the algorithm explores the distances between the new data point and all other existing data points. Subsequently, it selects the k nearest neighbors based on these calculated distances. For instance, if k is set to 10, the algorithm identifies the 10 closest neighbors, considering their distances.

In the context of a classification task, the algorithm determines the most frequent class among these neighbors. In contrast, for a regression task, it calculates the average value of each data point within the selected neighbors. This approach provides a robust foundation for decision-making in various analytical scenarios.

In this study, our initial step involves preprocessing the data, followed by its division into training, testing, and validation sets (specific details will be elaborated in the experimental results section). The optimization process for determining the optimal k-value is conducted, and it is found to be 44 (the methodology for this determination will be thoroughly discussed in the experimental results section). We designate "SalePrice" as the target variable for prediction, utilizing the average value derived from each set of 44 nearest neighbors. The outcomes of these computations are visually presented in Figure 1. The evaluation metrics are mean squared error and R-squared, with their formulas and calculation methods as follows:

Mean Squared Error (MSE):

Mean Squared Error is a measure of the average squared difference between predicted values and actual values. It is calculated using the formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y'_i)^2 \qquad (1)$$

where n is the number of data points, y_i is the actual value, and y'_i is the predicted value.

R-Squared (R2):

R-squared, also known as the coefficient of determination, represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s). It is calculated by the formula:

$$R^2 = 1 - \frac{SS_{res}}{SS_{totall}} \qquad (2)$$

Where SS_{res} is the sum of squared residual and SS_{tot} is the total sum of squared.

For our project, we first split our data into train and test by only 80 and 20 ration and then train our model by best k and test on test data. Finally, our scores for these two metrics are as follows:

Mean Squared Error ≅ 0.003078 and R-squared ≅ 0.39378.

Which are the acceptable results for our model and show its ability to predict the house prices.

B. Linear Regression, Lasso Regression, Ridge Regression

Linear Regression is a supervised machine learning algorithm used for predicting a continuous outcome variable (also called the dependent variable) based on one or more predictor variables (also called independent variables). The relationship between the variables is assumed to be linear, meaning that a change in the value of the predictor variable is associated with a constant change in the outcome variable. The simplicity and interpretability of linear regression make it a powerful tool for understanding and making predictions based on data.

Lasso Regression (Least Absolute Shrinkage and Selection Operator) is a linear regression algorithm with regularization. Similar to linear regression, lasso regression aims to predict a continuous outcome variable based on one or more predictor variables. However, lasso regression includes an additional regularization term to prevent overfitting and perform feature selection.

Ridge Regression, also known as Tikhonov regularization or L2 regularization, is a linear regression algorithm with a regularization term. Similar to ordinary linear regression, Ridge Regression includes a regularization term to prevent overfitting and stabilize the model, especially when dealing with multicollinearity (high correlation among predictor variables).
Here is the formula for calculating house prices:

$$Hourse\ Price = \beta_0 + \beta_1 \times features + \epsilon \quad (6)$$

Where β0 is the intercept (the base price of the house when one of the features is 0). β1 is the slope (the change in house price for a one-unit increase in features). ϵ is the error term, representing unobserved factors affecting the house price other than given features.

The linear regression model undergoes training with a dataset comprising both features and corresponding house prices. During this process, the model learns optimal values for coefficients (β_0 and β_1) that minimize the disparity between predicted and actual house prices in the training data. Once trained, the model transitions to predicting house prices for new, unseen data, leveraging the acquired coefficients. In the realm of model assessment, metrics like mean squared error (MSE) and R-squared come into play. MSE quantifies the average squared difference between predicted and actual house prices, while R-squared offers insight into how effectively the model elucidates variance in house prices. To further refine accuracy, we explore Ridge Regression as an augmentation to linear regression. Particularly prominent in house prediction models and similar regression tasks, Ridge Regression introduces regularization to the linear regression framework. This regularization term, an L2 penalty, is integrated into the loss function, effectively curbing overfitting, especially in datasets marked by multicollinearity (high feature correlation). The regularization mechanism penalizes large coefficients, promoting the model to maintain smaller coefficients, contributing to a more robust and generalizable predictive performance.

The objective function of Ridge Regression combines the least squares loss and the regularization term, creating a balanced approach to linear regression enhancement.

$$\min\left[\sum_{i=1}^{m}\left(y_i - \hat{y_i}\right)^2 + \alpha\sum_{j=1}^{n}{\beta_j}^2\right] \quad (7)$$

Where the first term is the standard least squares loss, which measures the squared difference between predicted value and actual value. The second term is L2 penalty term, which penalized large values of coefficient. The strength of regularization is controlled by hyperparameter α.

The Ridge regression model is represented as:

$$Hourse\ Price = \beta_0 + \beta_1 \times Feature_1 + \beta_2 \times Feature_2 + \cdots \beta_n \times Feature_n + \epsilon \quad (8)$$

The objective is to find the values of β0, β1, …. βn that minimized the Ridge Regression objective function. For further analysis, we used another extended version of linear Regression that is Lasso Regression. Similar to Ridge Regression, Lasso Regression introduces a regularization term to the standard linear regression objective function, helping to prevent overfitting and encourage sparse coefficient estimates (some coefficients become exactly zero).

$$\min\left[\sum_{i=1}^{m}\left(y_i - y_i\right)^2 + \alpha\sum_{j=1}^{n}|\beta_j|\right] \quad (8)$$

Where the first term is the standard least squares loss, which measures the squared difference between predicted value and actual value. The second term is L1 penalty term, which is the sum of the absolute values of the coefficient.

C. Gradient Boosting Regression

One effective ensemble learning method for estimating home prices is the Gradient Boosting Regressor (GBR). Using a sequential modeling technique, weak learners—often decision trees—are progressively constructed to fix the mistakes of earlier models.With each iteration, the Gradient Boosting algorithm attempts to decrease the total residual errors by fitting new models iteratively, improving the prediction power. In GBR model, the prediction at iteration m can be shown as follows:

$$F_m(x) = F_{m-1}(x) + \gamma_m\, h_m(x) \quad (9)$$

Where $F_{m-1}(x) =$ Prediction in previous iteration and $\gamma_m =$ Learning rate and $h_m(x) =$ Prediction of the weak learner.

To estimate house values, GBR sequentially combines a number of weak learners—in this case, decision trees. The max_depth parameter defines the maximum depth of each individual decision tree in the ensemble, whereas the n_estimators parameter establishes the number of decision trees in the ensemble when housing price estimate using gradient boosting is applied. These parameters regulate the ensemble model's complexity, which affects how well it can represent the complicated relationships seen in the housing dataset while reducing overfitting. GridSearchCV was used to explore these regularization parameters. By iteratively fixing these mistakes, the model learns from the mistakes made in earlier iterations and progressively increases the precision of price forecasts. GBR accurately estimates housing prices by capturing intricate correlations between several factors and the target variable through this procedure.

D. Support Vector

Regression techniques such as Support Vector Regression (SVR) works by identifying the hyperplane that best fits the data within a given margin of tolerance (epsilon), and it is derived from Support Vector Machines (SVM). While permitting some variations from the predicted values, SVR seeks to reduce prediction errors. The C parameter of SVR controls how much of the hyperplane is flat and how many instances are permitted inside the margin.
SVR was used in conjunction with GridSearchCV to explore kernel types ('linear' and 'rbf'), regularization parameter C, and gamma values in hyperparameter tuning. Based on minimizing prediction errors, the best SVR model was chosen. With the help of kernel functions, the trained SVR model was able to accommodate non-linear interactions while providing accurate estimations and efficiently capture the intricate correlations between housing attributes and prices by fitting an ideal hyperplane. SVR's promise as a reliable method for estimating home prices was demonstrated by the project.

E. E) Random Forest Regression

Employing the robust ensemble learning technique of Random Forest Regression, our study focuses on enhancing the accuracy of regression models for predicting house prices. This algorithm constructs multiple decision trees during training, leveraging randomized subsets of data and features at each split to curtail overfitting. The aggregated predictions from diverse trees form the final output, ensuring model generalization and effectiveness in complex prediction tasks.

In our research, Random Forest Regression is applied to house price prediction using the scikit-learn library in Python. Hyperparameter optimization involves an exhaustive search to enhance model performance and generalization. To ensure reliability, cross-validation assesses the model's performance across different training data subsets. The final model, characterized by optimal hyperparameters, is evaluated on a separate test set using metrics like Mean Squared Error (MSE) and R-squared (R2). This methodology strikes a balance between complexity and performance, capturing intricate data relationships while mitigating overfitting. The results contribute valuable insights into predictive modeling for housing prices, offering a versatile tool for real-world applications. The application of the Random Forest Regression algorithm is validated through evaluation metrics, specifically MSE and R-Squared.

IV. Experimental Results

In this section, our aim is to elucidate the various steps, including the introduction of the dataset, the preprocessing methodology, and the hyperparameter tuning approach. These steps are essential for facilitating a comprehensive comparison of results..

A. Dataset

We obtained our dataset from the Kaggle website, and it is publicly available at the following link: https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data. The dataset encompasses various features related to houses, with "SalePrice" as our target variable. A selection of these features, along with their descriptions, is provided in Table 1.

MSSubClass	The Building Class	
LotFrontage	Linear feet of street connected to property	
LotArea	Lot size in square feet	
Street	Type of road access	
LotShape	General shape of property	
LandContour	Flatness of the property	
MSZoning	The general zoning classification	
Utilities	Type of utilities available	

Table1- some of the features in our dataset.

Street	Type of road access to property
Grvl	Gravel
Pave	Paved

Table2 – A description of "Street" feature.

B. Preprocessing Data

As evident in Table 2, some of our features are categorical, and since regression algorithms require numerical data, we must convert categorical features into numerical ones. One of the most effective methods at this stage is one-hot encoding. This technique involves transforming categorical features into numeric ones by adding columns equal to the number of categories to the dataset. For instance, considering the "street" feature with only two categories, we introduce two columns, each corresponding to a type of street. We assign a value of 1 in the respective column if the house's street category is 1, and repeat this process for all data points. Following this step, our original dataset, initially with dimensions (2919, 81), expands to (2919, 2262).

Subsequently, we proceed to the feature selection step to identify the most informative features among the multitude. This process will be expounded upon in the next paragraph.

In the feature selection phase, we adopt the Mutual Information Gain method based on the principles outlined in [2]. Mutual Information (MI), synonymous with Information Gain (IG), quantifies the dependence or shared information between two random variables [3]. It is rooted in Shannon's definition of entropy, expressed as:

$$H(X) := -\sum_{x} p(x)\log\big(p(x)\big) \qquad (3)$$

This definition captures the uncertainty inherent in the random variable X. In the context of feature selection, our goal is to maximize the mutual information, signifying relevance, between a given feature and the target variable.

The Mutual Information (MI) is defined as the relative entropy between the joint distribution and the product distribution:

$$MI(X;T) := \sum_{X}\sum_{T} p(x^j,t)\log\frac{p(x^j,t)}{p(x^j)p(t)} \qquad (4)$$

Here $p(x^j,t)$ represents the joint probability density function of feature x^j and target t, while $p(x^j)\ and\ p(t)$ denote the marginal density functions. The MI is zero or greater than zero for independent or dependent X and Y, respectively. To maximize MI, a greedy step-wise selection algorithm is employed.

The algorithm initializes a subset of features, denoted by matrix S, with one feature. Features are added one by one to this subset, and the index of the selected feature is determined using:

$$j := \arg\max\ MI(Y_{S\ \cup x^j}\ ;t) \qquad (5)$$

This equation is also used for selecting the initial feature. The selected features are assumed to be independent. The addition of new features stops when the highest increase in MI at the previous step is achieved. It is crucial to note that even if a variable appears redundant or uninformative in isolation, it may still contribute valuable information when combined with another variable. This approach effectively reduces the dimensionality of features without significantly compromising performance [4].

For our study, we set a threshold of 0.01 for our features. Subsequently, out of 2262 features, we selected 56 features deemed most informative. Our algorithm is then executed, and the dataset is split, considering only these selected features – effectively constituting our new dataset. In Table 3, you can find some of the essential features. Notably, "Built Year" emerges as a crucial feature significantly influencing house prices.

Feature Name	Score
MSSubClass	0.158
YearBuilt	0.087343
Neighborhood	0.1158657413
MasVnrArea	0.084393

Table 3 – some selected features along with their score.

C. *Experimental Setup*

Following the selection of important and informative features, the subsequent step involves hyperparameter tuning for each algorithm through dataset splitting and cross-validation. Take the K-Nearest Neighbors (KNN) algorithm as an example: initially, we shuffle our data to mitigate bias or indexing issues. Subsequently, we partition our dataset into three sets - train, test, and validation, with an 80-10-10 ratio, respectively. This process is repeated across ten distinct partitions.

For each value of k ranging from 1 to 51, we execute a ten-fold cross-validation. In the first iteration, segments 0-7 are assigned as the training set, 8 as the validation set, and 9 as the test set. In the subsequent iteration, segments 1-8 serve as the training set, 9 as the validation set, and 0 as the test set and the rest of the iterations are the same.

Throughout each iteration, the model is trained on the training set and tested on the validation set to identify the optimal k. This process is repeated ten times, and the errors for each k are averaged. The results are then sorted, and the minimum errors are selected as the best k. Figure 2 visually represents the errors corresponding to different k values, utilizing Mean Squared Error (MSE). Following the determination of the best k, ten additional iterations are conducted. The model is constructed solely with the optimal k, trained on the training set, and tested on the test set. Subsequently, the mean score over ten tests is computed, and the best k is reported, considering the mean error ± standard deviation, as illustrated in Table 4.

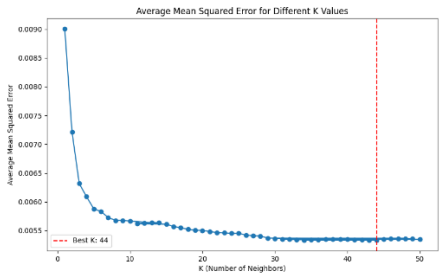

Figure2 – show MSE over different K

Algorithm	Best Parameter	Errors
KNN	K = 44	0.0053±0.002
RFR	'n_estimators': 600, 'min_samples_split': 5, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': 70, 'bootstrap': False	0.0047±0.0013
SVR	C= 0.1 Kernel = Linear	0.006422±0.0015
GBR	n_estimators = 200, Learning Rate = 0.05 Maximum Depth = 4	0.001673±0.0008
RR	α: 0.1	0.003326±0.012
LR	α: 10	0.00553±0.001

Table4 – the error of hyper parameter tuning

D. *Experimental Results*

In this section, we want to have a overview of the different algorithms and compare them together. You can find complete result and errors for each algorithm in Table 5. As it is evident the best algorithm for our dataset is Gradient Boosting Regression with 0.001673 MSE.

Algorithm	MSE	R-Squared
KNN	0.003078	0.39378
RFR	0.003063	0.396669
SVR	0.004922	0.030513
LR(Linear)	0.278555	0.548566
GBR	0.001673	0.670361
RR	0.005955	0.172927
LR(Lasso)	0.0051022	0.00479

Table5- comparison of different algorithms

CONCLUSION

This paper investigates different models for housing price prediction. Different types of Machine Learning methods including Random Forest, KNN, and Kernel SVR, SVR and two techniques in machine learning including Lasso Regression and Ridge Regression are compared and analyzed for optimal solutions. Even though all those methods achieved desirable results, different models have their own pros and cons. The Random Forest method has the lowest error on the training set but is prone to be overfitting. Its time complexity is high since the dataset must be fit multiple times. The Ridge and Lasso Regression are decent methods when comparing accuracy, but their time complexities are the best, especially Lasso. The Kernel SVR method is simple but performs a lot better than the three previous methods due to the generalization. Finally, the tree regression Generalization Regression method has a complicated architecture, but it is the best choice when accuracy is the top priority. Even though Lasso Regression and Ridge Regression deliver satisfactory results, time complexity must be taken into consideration since both contain Random Forest, a high time complexity model. KNN also has K-fold cross-validation in its mechanism so it has the worst time complexity.

Further research about the following topics should be conducted to further investigate these models, especially the combinations of different models:

- The coupling effect of multiple regression models.
- The “re-learn” ability of machine learning models.
- The combination of Machine Learning and Deep Learning methods.
- The driving factors for the good performance of tree-based models.
- The faster ways to fit complex models.

ACKNOWLEDGMENT

We would like to express our heartfelt gratitude to our instructor, Benaymin Ghojoogh, at the University of Guelph, Canada. His guidance, encouragement, and insightful feedback played a pivotal role in shaping the course of our research.

Our sincere thanks also go to the University of Guelph for providing us with an enriching academic environment and valuable resources that facilitated our study. We extend appreciation to our fellow students and colleagues who contributed to discussions and shared insights that enhanced the quality of our work. Additionally, we want to acknowledge the support from the University of Guelph, which includes any relevant departments, for their role in fostering an environment conducive to research and learning.

REFERENCES

[1] 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.

[2] Ghojogh, B., Samad, M. N., Mashhadi, S. A., Kapoor, T., Ali, W., Karray, F., & Crowley, M. (2019). Feature selection and feature extraction in pattern analysis: A literature review. *arXiv preprint arXiv:1905.02845*.

[3] T. J. Cover TM, Elements of Information Theory. 2nd edn, Wiley-Interscience, New Jersey, 2006, vol. 2.

[4] N. Nicolosiz, “Feature selection methods for text classification,” Department of Computer Science, Rochester Institute of Technology, Tech. Rep., 2008.

[5] A. Jafari and R. Akhavian, “Driving forces for the US residential housing price: a predictive analysis,” Built Environ. Proj. Asset Manag., vol. 9, no. 4, pp. 515–529, 2019, doi: 10.1108/BEPAM-07-2018-0100.

[6] A. Osmadi, E. M. Kamal, H. Hassan, and H. A. Fattah, “Exploring the elements of housing price in Malaysia,” Asian Soc. Sci., vol. 11, no. 24, pp. 26–38, 2015, doi: 10.5539/ass.v11n24p26.

[7] D.-G. Owusu-Manu, D. J. Edwards, K. A. Donkor-Hyiaman, R. O. Asiedu, M. R. Hosseini, and E. Obiri-Yeboah, “Housing attributes and relative house prices in Ghana,” Int. J. Hous. Mark. Anal., vol. 8, no. 2, p. 1998, 2018, doi: 10.1017/CBO9781107415324.004.

[8] G. Ke et al., “LightGBM: A highly efficient gradient boosting decision tree,” Adv. Neural Inf. Process. Syst., vol. 2017-Decem, no. Nips, pp. 3147–3155, 2017.

[9] Binbin Lu, Martin Charlton, Paul Harris & A. Stewart Fotheringham, "Geographically weighted regression with a non-Euclidean distance metric: a case study using hedonic house price data", International Journal of Geographical Information Science, pp. 660-681, Jan 2014.

[10] Li, Da-Ying & Xu, Wei & Zhao, Hong & Chen, Rong-Qiu. (2009). A SVR based forecasting approach for real estate price prediction. Proceedings of the 2009 International Conference on Machine Learning and Cybernetics. 2. 970 - 974. 10.1109/ICMLC.2009.5212389.

[11] Anders Hjort, Johan Pensar, Ida Scheel & Dag Einar Sommervoll (2022) House price prediction with gradient boosted trees under different loss functions, Journal of Property Research, 39:4, 338-364, DOI: 10.1080/09599916.2022.2070525