Research on the House Price Forecast Based on machine learning algorithm

Weinan Weng[1, *]

[1]Faculty of Science & Engineer, University of Liverpool, Liverpool, L69 3BX, United Kingdom

*Corresponding author: sgwweng@liverpool.ac.uk

Abstract. House price experiences some fluctuations every year, due to some potential factors such as location, area, facilities and so on. Housing price prediction is a significant topic of real estate, and it is beneficial for buyers to make strategy decisions about house dealing. There are many research on house price forecast, yet the current research cannot comprehensively compare and analyze the popular house price prediction approach. Constructing a model begins with pre-processing data to fill null values or remove data outliers and the categorical attribute can be shifted into required attributes by using one hot encoder methodology. This paper used the following five algorithms decision tree, random forest regression, Adaptive Boosting (AdaBoost), Gradient Boosting Decision Tree (GBDT), and extreme gradient boosting (XGBoost) this paper utilized to predict house prices and compared according to the root mean squared error. This paper found GBDT and XGBoost have more accurate prediction results compared with other algorithms. Besides, this paper found which features most affect the price of a house. In real-world applications, machine learning based housing price prediction models are utilized by banks and financial institutions to obtain better house price assessment, risk analysis and lending decisions.

Keywords: House Price; Forecast; Machine Learning; Regression; Hyperparameter Optimization.

1. Introduction

House is a daily necessity for people, and its price may rely on various factors involving location, facilities, features as well as demand and supply of real estate. As a result, housing price forecasting has become one of the significant research points in these years. By predicting future home price trends, not only can home buyers plan their investments in advance to save time and money, but real estate companies can also develop marketing strategy ahead to maximize their profits[1]. Nowadays, machine learning is a trend technology and is booming in the market, where machine learning algorithms can be used to forecast house sale prices purely based on previous data. From the previous literature, Phan (2018) uses a support vector machine model to predict house prices in Australia[2]. In addition, machine learning has become a price prediction method. Shah and Vatsal (2007) predicted the stock prices of Google and Yahoo using machine learning models such as Adaboost, SVM[3].

This paper predicts the housing price by machine learning models such as random forest, decision tree, AdaBoost, GDBT as well as XGBoost. In this paper, Bayesian optimization is used to help machine learning models find their most appropriate parameters, making the models better at predicting house prices. The predictive house price model can be used to derive which characteristics of a house will most influence the price of a house. This study can guide people to accurately calculate the price of their homes.

2. Data preprocessing and Performance evaluation

2.1 Data preprocessing

In this paper, dataset was extracted from Kaggle, involving 1460 data with 80 features and 1 label. The description and data type for each feature can be seen in the **Table 1**. The first phase is to calculate the correlation coefficient between features and the lable of dataset in order to explore the relevance in **Figure 1**.

Table 1. The description and data type for each feature

Feature	Description	Data type	Feature	Description	Data type
MSSubClass	Identifies the type of dwelling involved in the sale.	Int64	Foundation	Type of foundation	object
MSZoning	Identifies the general zoning classification of the sale.	object	BsmtQual	Evaluates the height of the basement	object
LotFrontage	Linear feet of street connected to property	Int64	BsmtCond	Evaluates the general condition of the basement	object
Street	Type of road access to property	object	BsmtExposure	Refers to walkout or garden level walls	object
LotFrontage	Linear feet of street connected to property	Int64	BsmtFinType1	Rating of basement finished area	object
LotArea	Lot size in square feet	Int64	BsmtFinSF1	Type 1 finished square feet	object
Alley	Type of alley access to property	object	BsmtFinType2	Rating of basement finished area (if multiple types)	object
LotShape	General shape of property	object	BsmtFinSF2	Type 2 finished square feet	object
LandContour	Flatness of the property	object	BsmtUnfSF	Unfinished square feet of basement area	object
Utilities	Type of utilities available	object	TotalBsmtSF	Total square feet of basement area	object
LotConfig	Lot configuration	object	Heating	Type of heating	Object
LandSlop	Slope of property	object	HeatingQC	Heating quality	object
Neighborhood	Physical locations within	object	CentralAir	and condition	object
			Electrical	Central air	object
Condition1	Ames city limits	object	1stFlrSF	conditioning	int64
Condition2	Proximity to various	object	2ndFlrSF	Electrical system	int64
			LowQualFinSF	First Floor square	object
BldgType	conditions	object		feet	
HouseStyle	Proximity to	object	GrLivArea	Second floor	object
OverallQual	various	int64		square feet	
	conditions (if		GarageCars	Low quality	int64
OverallCond	more than one is	int64	BsmtFullBath	finished square	object
YearBuilt	present)	object	BsmtHalfBath	feet (all floors)	object
YearRemodAdd	Type of dwelling	object	FullBath	Above grade	object
	Style of dwelling		HalfBath	(ground) living	object
RoofStyle	Rates the overall	object	Bedroom	area square feet	object
RoofMatl	material and	object		Size of garage in	
Exterior1st		object		car capacity	

Exterior2nd	finish of the house	object	Kitchen	Basement full bathrooms	object
			KitchenQual		object
MasVnrType	Rates the overall condition of the house	object	TotRmsAbvGrd	Basement half bathrooms	object
MasVnrArea:		object	Functional	Full bathrooms above grade	object
ExterQual	Original construction date	object		Half baths above grade	object
ExterCond	Remodel date (same as construction date if no remodeling or additions)	object	Fireplaces		object
			FireplaceQu	Bedrooms above grade (does NOT include basement bedrooms)	object
GarageYrBlt		int64	GarageType		object
GarageFinish		object	GarageArea		int64
GarageCond		object	GarageQual		object
WoodDeckSF	Type of roof	object	PavedDrive	Kitchens above grade	object
OpenPorchSF	Roof material	object	Fence		object
EnclosedPorch	Exterior covering on house	object	MiscFeature	Kitchen quality	object
3SsnPorch		object		Total rooms above grade (does not include bathrooms)	
ScreenPorch	Exterior covering on house (if more than one material)	object	MiscVal		object
PoolArea		object	MoSold		object
PoolQC		object	YrSold	Home functionality (Assume typical unless deductions are warranted)	object
SaleType:	Masonry veneer type	object	SaleCondition		object
	Masonry veneer area in square feet			Number of fireplaces	
	Evaluates the quality of the material on the exterior			Fireplace quality	
				Garage location	
	Evaluates the present condition of the material on the exterior			Size of garage in square feet	
				Garage quality	
	Year garage was built			Paved driveway	
				Fence quality	
	Interior finish of the garage			Miscellaneous feature not covered in other categories	
	Garage condition			$Value of miscellaneous feature	
	Wood deck area in square feet				
	Open porch area in square feet			Month Sold (MM)	
	Enclosed porch area in square feet			Year Sold (YYYY)	
				Condition of sale	
	Three season porch area in square feet				
	Screen porch area in square feet				

	Pool area in square feet Pool quality Type of sale

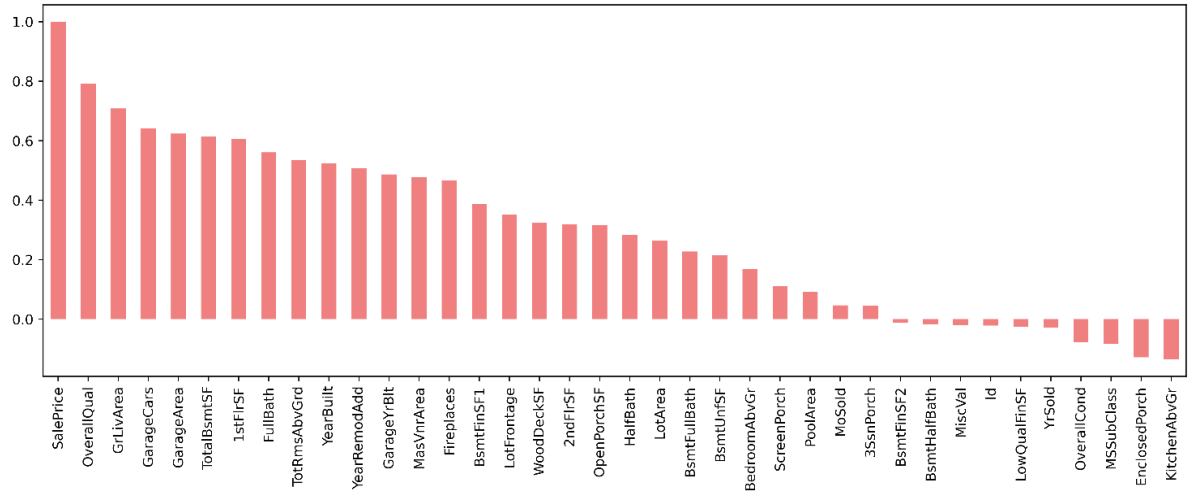

Fig. 1 The correlation coefficient between features and one label

The ten features in **Fig. 1** containing 1stFlrSF, OverallQual, TotRmsAbvGrd, GrLivArea, GarageCars, GarageArea, FullBath, TotalBsmtSF, YearBuilt as well as YearRemodAdd were then selected based on the condition of highly and medium related (correlation coefficients less than 0.8 and greater than 0.6), to remove outliers. The correlation degree corresponding to the absolute value is from Pearson correlation[4] which can be seen in **Table 2**.

Table 2. The correlation degree

Absolute value range of P	Correlation degree
0.0-0.3	Extremely weak or irrelevant
0.3-0.5	Low correlation
0.5-0.8	Medium related
0.8-1	Highly related
1	Totally related

Since the correlation coefficient were computed is positive, some samples with the large feature value but small label value were removed. Based on **Fig. 2**, sale price 200,000 was relatively low to remove the outliers for the following steps.

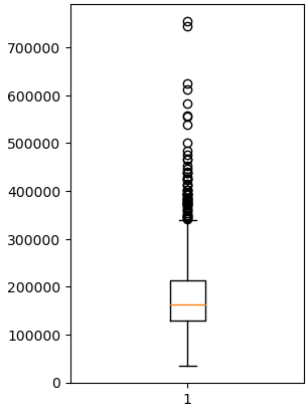

Fig. 2 The box diagram of housing sales price

The outliers which are in the low right corner need to be deleted can be found in **Fig. 3**.

After computing, 19 features have some missing data as seen in **Table 3.**

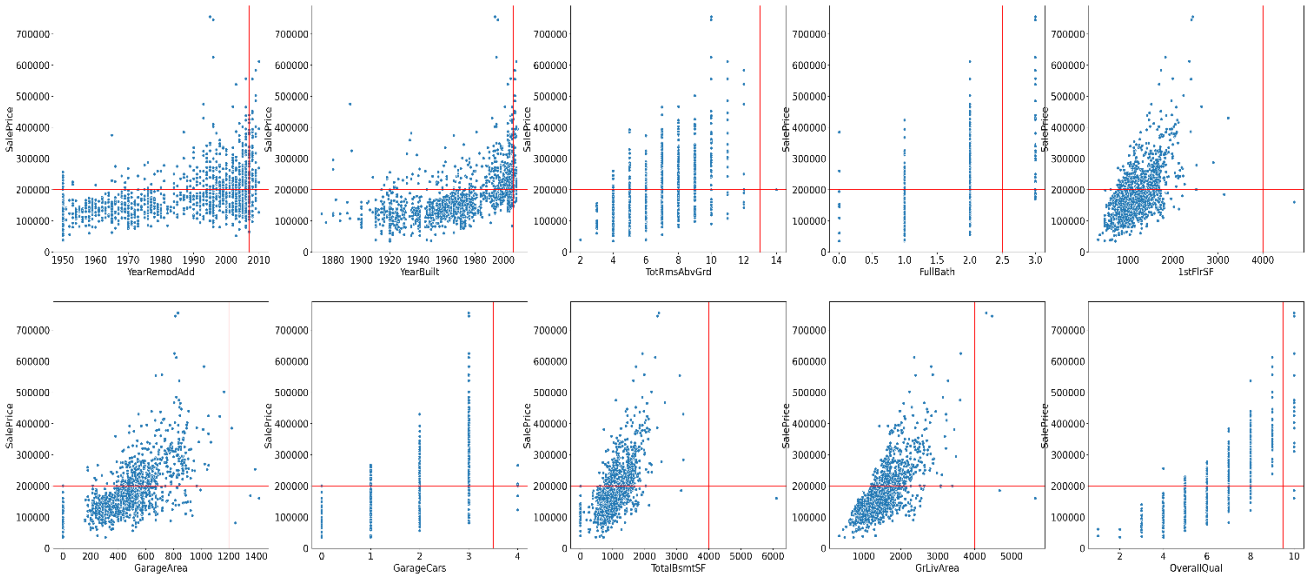

Fig. 3 The scatter plots with two lines

Table 3. The number of missing data for each feature

Feature	**Number of missing data**	**Feature**	**Number of missing data**
PoolQC	1452	GarageQual	81
MiscFeature	1404	BsmtFinType2	38
Alley	1367	BsmtExposure	38
Fence	1177	BsmtQual	37
FireplaceQu	690	BsmtCond	37
LotFrontage	259	BsmtFinType1	37
GarageYrBlt	81	MasVnrArea	8
GarageCond	81	MasVnrType	8
GarageType	81	Electrical	1
GarageFinish	81		

And the next step is to deal with missing data, which could be divided into five processes:

Remove features which have large number of missing data.

In the table, the first five features have more than 500 data , these features need to be deleted.

Remove the feature Id.

Get rid of the sample with missing value corresponding to the feature ‘Electrical ‘

Fill the missing data with ‘None‘ or ‘0‘ for the following features

GarageType, GarageYrBlt, GarageCond, GarageType, GarageFinish, GarageQual, BsmtFinType2, BsmtExposure, BsmtQual, BsmtCond, BsmtFinType1, MasVnrArea and MasVnrType were filled with ‘None‘ if the data type is object and ‘0‘ if the data type is int.

Fill ‘LotFrontage‘ by Groupby function in Pandas

Since the feature ‘Neighborthood‘ have the biggest number of categories, it was selected to fill the missing values of ‘LotFrontage‘. The feature ‘Neighborthood‘ have 25 various categories in **Fig. 4**.

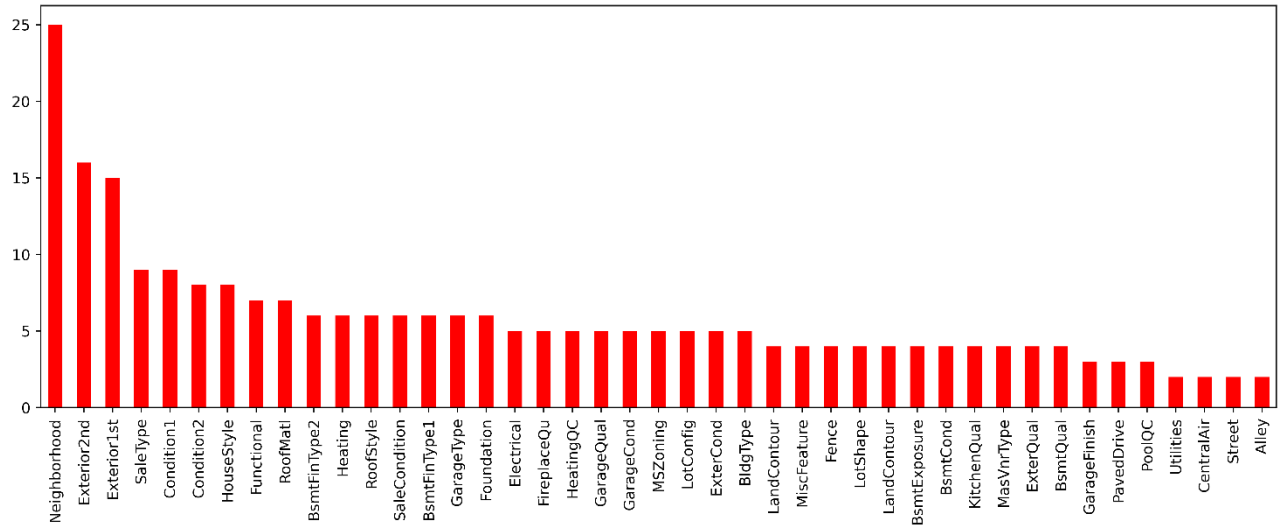

Fig. 4 The categories of feature 'Neighborhood'

All data are divided into 25 classes according to the classification of 'Neighborhood', then the mean value is calculated by ignoring the missing values of 'LotFrontage' in each class, and finally the mean value is filled into the missing values of 'LotFrontage' in which it is located.

Table 5. The mean value for each category in feature 'Neighborhood'

The category of the feature 'Neighborhood'	Mean value	The category of the feature 'Neighborhood'	Mean value
Blmngtn	47.142857	NPkVill	32.285714
Blueste	24.000000	NWAmes	80.883721
BrDale	21.562500	NoRidge	91.878788
BrkSide	57.509804	NridgHt	81.881579
ClearCr	83.461538	OldTown	62.788991
CollgCr	71.682540	SWISU	59.272727
Crawfor	71.804878	Sawyer	74.437500
Edwards	64.875000	SawyerW	71.500000
Gilbert	79.877551	Somerst	64.666667
IDOTRR	60.757576	StoneBr	62.700000
MeadowV	27.800000	Timber	80.379310
Mitchel	69.742857	Veenker	59.714286
NAmes	76.497297		

After the missing data were solved, 1448 samples and 74 features were left to do one hot encoder, and then the number of features is 241.

2.2 Performance evaluation

2.2.1 RMSE

Before constructing the model, evaluation function called Root Mean Squared Error (RMSE) is utilized to evaluate the performance of various models. RMSE can reduce the error compared with Mean Squared Error (MSE) when the data have large quantity. The formula is as follow:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (1)$$

where n denotes the total number of samples, y_i stands for real value and $\hat{y}_i$ represents predicted value.

In this paper the difference between test RMSE and train RMSE will be considered as the degree of model overfitting.

2.2.2 Five-fold Cross-validation

Five-fold cross-validation (five-fold CV) is a strategy all data in our dataset is used to train our model. The basic theory is that the data is divided into five equal sections and a section of each trial is chosen for testing and the others for training. The average of the five trials is obtained. As shown below, the first experiment uses the first copy as the test set and the others as the train set [5]. Using a cross-validation approach can help us make fuller use of the data set and obtain more accurate RMSE.

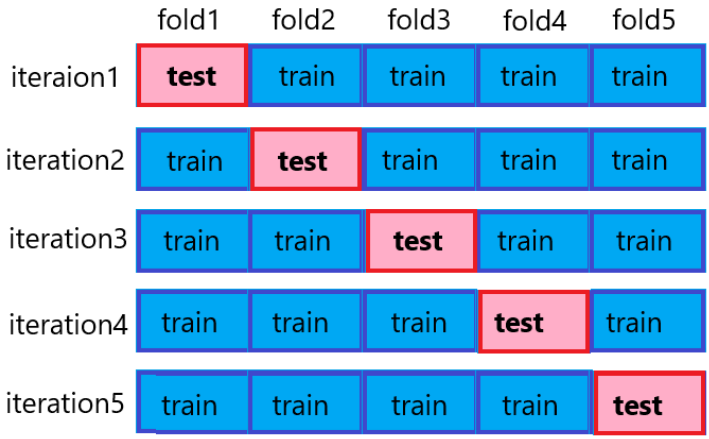

Fig. 5 The description of five-fold cross-validation

2.2.3 Bayesian Optimization

Bayesian optimization methods are the current state of the art in the field of hyperparametric optimization and can be considered as the most advanced optimization framework currently available, which can be applied to major areas of AutoML, neural network architecture search NAS and advanced areas. Bayesian optimization is an approach that utilizes Bayes' theorem to help guide the finding the extreme value(maximum or minimum) of the objective function, which implies in every iteration, the previously information (prior knowledge) is applied for the next optimization[6]. Bayesian optimization is used to find an acceptable maximum by estimating the black box function without knowing what the objective function (black box function) is. Compared with grid search, the advantage is that there are fewer iterations (saving time) and the granularity can be to small, the disadvantage is that it is difficult to find the global optimal solution[7].

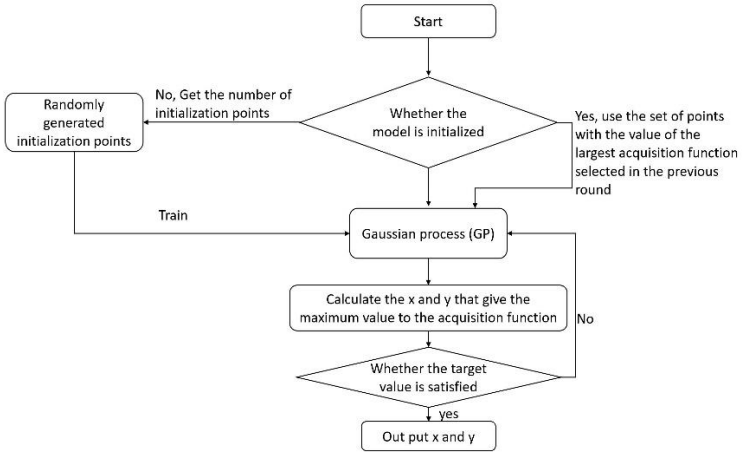

Fig. 6 Bayesian optimization process

3. Forecast results based on 5 methods of Machine Learning

3.1 Decision Tree

A decision tree is a tree-like structure resembling a flowchart, where each node inside the tree denotes a test of a feature, the branches of the genus indicate each test outcome of the feature, and the leaf nodes of the tree stand for a classified outcome [8]. Additionally, a regression tree is a partition of the feature space into cells, each of which has a particular output. Considering the test data, the corresponding output value can be derived by grouping it into a cell according to the features.

3.2 Random Forest

Random forest is an ensemble machine learning model consists of predictions from a number of decision trees to produce more precise outcome. The random forest regressor observes the features of the attributes and trains the model by analyzing the given features. The random forest regulator analyzes data from graphs, combinations of attributes, labels involving features and according to the system.

For each tree, a series of instances that are replaced by a random sample from the train set corresponding to a random vector $\emptyset_k$, constitutes a particular tree. As all sequences will not be identical exactly, the decision trees built from these sequences will also be marginally different [9]. The forecast of the k-th tree for input X can be expressed by formula (2):

$$h_k(X) = h(X, \emptyset_k), k \in \{1,2, \dots, K\} \tag{2}$$

where K denotes the number of trees. As the tree is split, features are randomly chosen for each tree to avoid correlation between features. A node S is partitioned into two subsets, that is, S1 and S2, by choosing a threshold c which minimizes the difference in sum of squared errors [10].

$$SSE = (\sum_{i \in S_1}(v_i - \frac{1}{|S_1|}\sum_{i \in S_1} v_i)^2 + \sum_{i \in S_2}(v_i - \frac{1}{|S_2|}\sum_{i \in S_2} v_i)^2) \tag{3}$$

Then the median or mean output of any subtree could be calculated. In the end, the final prediction is the average of the mean output of each tree, as described in the formula below.

$$h(X) = \frac{1}{K}\sum_{i=1}^{K} h_k(X) \tag{4}$$

After computing the RMSE with five-fold Cross-validation, the line chart of train set and test set with respect to random forest and decision tree is in **Fig. 6**

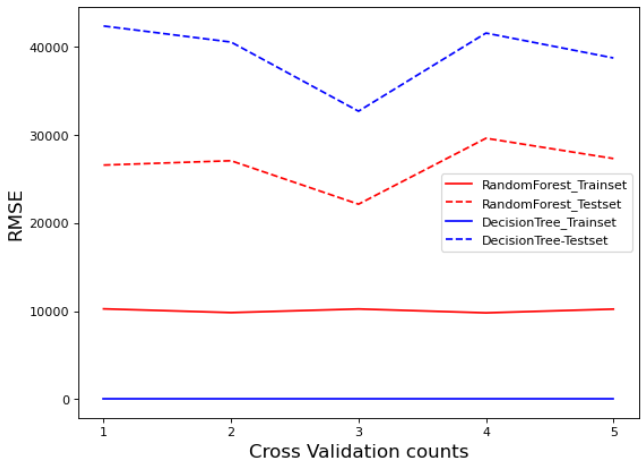

Fig. 6 The RMSE of train set and test set

From the figure above, both the random forest and the decision tree are in an overfitting state, although the random forest is less overfitted and the decision tree is more overfitted. The results of both algorithms are better on the training set, the decision tree can learn the training set perfectly and reach the RMSE=0, while the RMSE of the random forest on the training set hovers around 10000.

The results of both random forest and decision tree with default values are not very satisfactory, this paper used Bayesian optimization to adjust the parameters. Where, max depth = 20; max features = 75; min impurity decrease = 60, n estimators = 60, min samples leaf=1, min samples split=2, min weight fraction leaf = 0.0, criterion = 'mse',.

Then the parameter above is used to figure out RMSE, the result of model in **Table 6.**

Table 6. RMSE result before and after optimizing with five-fold CV

	Train score	Test score	Run time	Overfit
Default parameter	10007.94	26562.94	3s	16555
After Bayesian optimization	10204.91	25587.55	1.4s	15382.64

When the model uses parameters that are Bayesian optimized, it can be found that the error on the test set declines and the error of the train set rises. The overfitting reduced and the generalization ability is enhanced. The speed of the model runs has also improved significantly.

After that, the parameter above is used to figure out RMSE, this paper obtain the following importance:

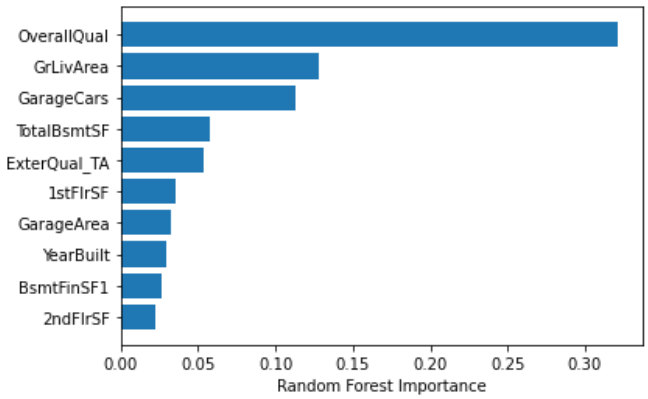

Fig. 7 Random Forest Importance

According to **Fig. 7**, feature 'OveallQual' lead the way more than 0.30 among the importance and the second one is feature 'GrLivArea'. And feature 'GarageCars' have a similar pattern with the second one.

3.3 AdaBoost

The construction process of AdaBoost is very simple: first, a decision tree is built on the full sample, and based on the prediction results and loss function values of the decision tree, the sample weights of the incorrectly predicted samples are increased in the dataset, and the weighted dataset is used to train the next decision tree[11]. This process is equivalent to intentionally weighting the "hard to classify correctly" samples and decreasing the weight of the "easy to classify correctly" samples, while directing the attention of the subsequent weak evaluators to the hard to classify correctly samples. In this process, the result of the previous decision tree influences the next decision tree by affecting the sample weights, i.e., the data distribution, and the whole process is adaptive. When all the weak evaluators are built, the output of the integration algorithm is equal to the weighted average of all the weak evaluator outputs, and the weights used for weighting are also calculated adaptively during the tree building process.

This paper alerted the parameter based on Bayesian optimization as, estimators=300; learning rate=1.0; loss='linear'.

After that, this paper compared the RMSE of train set and test set in this two cases in **Table 7.**

Table 7. RMSE result before and after optimizing with five-fold CV

	Train score	Test score	Run time	Overfit
Default parameter	26008.01	31387.95	0.6s	5379.94
After Bayesian optimization	25769.47	30925.35	1.5s	5155.88

AdaBoost has not changed much after Bayesian optimization, and it can be said that AdaBoost is extremely lacking in tuning space and has a serious lack of learning ability.

The value after optimizing is better than before for the test set.

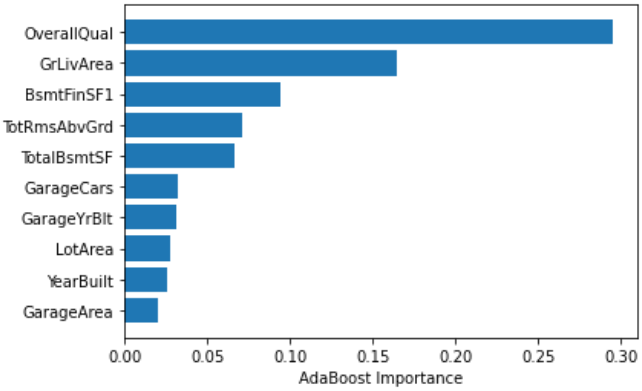

Fig. 8 AdaBoost Importance

The AdaBoost importance can be calculated by Python as seen in **Fig. 8**. To sum up, feature 'OverallQual' stand the first position which implies the rating the overall materials and finishes of the house is the most significant factor in housing price. Moreover, feature 'GrLivArea' shows nearly half of the importance of feature 'OverallQual'.

3.4 Extreme Gradient Boosting (GBDT)

Extreme Gradient Boosting (GBDT) can robustly handle different sorts of data, containing discrete and continuous values. It not only finds the higher-order relations between features automatically, but also has high prediction precision and excellent interpretability with less parameter adjustment time. Given that processing textual data is not as effective as dealing with digital data, it is essential to change textual data to digital data whenever it is possible [12]. The weak evaluator output type of GBDT is no longer the same as GBDT output type. For AdaBoost and Random Forest algorithms, the weak evaluator is also a regressor when the integrated algorithm is performing a regression task, and a classifier when the integrated algorithm is performing a classification task. However, for GBDT, the weak evaluator must be a regressor regardless of the regression, classification, sorting task being performed by GBDT.

After using Bayesian optimization, the parameter was adjusted as follows: n estimators=400, random state=1106, subsample=0.7, max depth =2.

then the results can be figured out by Python which can be found in **Table 8.**

Table 8. RMSE result before and after optimizing with five-fold CV

	Train score	Test score	Run time	Overfit
Default parameter	13321.79	23350.94	2s	10029.15
After Bayesian optimization	14259.93	22916.56	0.3s	8656.63

The prediction error of the GBDT model is small when using the default parameters compared to random forest and Adaboost. The prediction accuracy and running time of the model are further improved when using Bayesian optimization. Then GBDT importance can be seen below, the factor that most influence the price of housing is feature 'OverallQual', almost 0.5. Followed by feature 'GrLivArea' which is nearly one fourth of the former one. The others seem to have little relation with house price since the importance is rather low, as seen in **Fig. 9.**

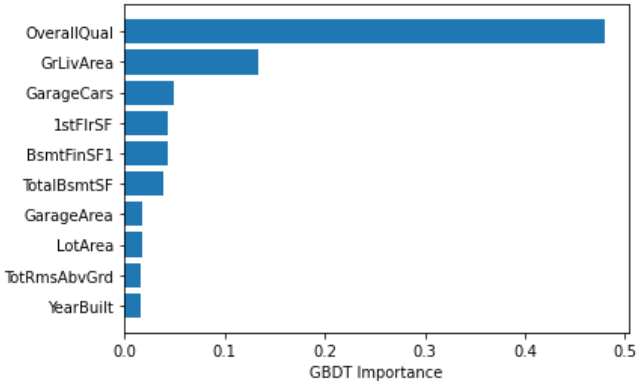

Fig. 9 GBDT Importance

3.5 XGBoost

XGBoost regression is the abbreviation of extreme gradient boosting regression and is one of the best supervised learning algorithms, because of its scalability in all scenarios and faster speed than other machine learning algorithms. XGBoost adds a structural risk term to the loss function to form a new objective function. This change makes XGBoost different from other boosting algorithms in that XGBoost is trained towards minimizing the objective function, rather than minimizing the loss function. This has the ability to effectively prevent model overfitting.

In this section, the XGBRegressor from the xgboost open source package is used [13]. After several adjustments of the XGBoost model by Bayesian optimization, this paper set the parameters as follows:

n estimators = 200, random state=1106, booster ="dart", colsample bynode = 0.35, colsample bytree = 0.65 , eta =0.7, gamma = 1600000000, max depth = 8, min child weight = 7 , objective="reg:squarederror", rate drop = 0.06.

RMSE before and after optimizing based on Bayesian theorem are demonstrated in **Table 9**. Both the train set and test set experienced a slight drop after changing the parameters.

Table 9. RMSE result before and after optimizing with five-fold CV

	Train score	Test score	Run time	Overfit
Default parameter	1166.64	26316.82	4s	25150.18
After Bayesian optimization	11587.18	22629.39	21.4s	11042.21

From **Fig. 10**, XGBoost has a serious tendency to overfit its performance with the default parameters. When using Bayesian optimization XGBoost although obtained the smallest RMSE compared to the model used in the previous section but also required the longest runtime.

The XGBoost importance computed by Python can be seen below, and the most essential factor of housing price is feature 'ExterQual_TA' which means evaluating the quality of the material on the exterior affect sales price most.

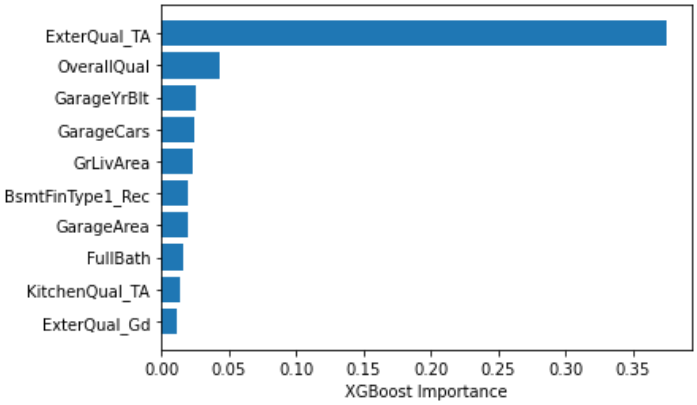

Fig. 10 XGBoost Importance

4. Discussion

From the prediction results, the AdaBoost algorithm performs poorly before and after tuning the parameters with a prediction error of more than 30,000. This proves that the AdaBoost algorithm is not suitable for house price datasets with many features. The forecast error of the random forest model is smaller than that of AdaBoost but has a serious tendency of overfitting. Therefore, the random forest algorithm also shall not apply to for predict the house price data in this paper. Although the XGBoost algorithm obtains the smallest prediction error, this model requires more computing time due to the complexity of the parameters. GBDT model has a larger prediction error than XGBoost but requires much less computing time than XGBoost. XGBoost is a good algorithm for the data set in this paper if only pursuing model accuracy, but if considering the time cost. GBDT is the most cost-effective algorithm.

Table 10. Comparison

		Train score	Test score	Run time	Overfit
Random Forest	Default parameter	10007.94	26562.94	3s	16555
	After Bayesian optimization	10204.91	25587.55	1.4s	15382.64
AdaBoost	Default parameter	26008.01	31387.95	0.6s	5379.94
	After Bayesian optimization	25769.47	30925.35	1.5s	5155.88
GBDT	Default parameter	13321.79	23350.94	2s	10029.15
	After Bayesian optimization	14259.93	22916.56	0.3s	8656.63
XGBoost	Default parameter	1166.64	26316.82	4s	25150.18
	After Bayesian optimization	11587.18	22629.39	21.4s	11042.21

5. Conclusion

This paper adopted 5 machine learning algorithms to predict housing prices, including Decision Tree, Random Forest, AdaBoost, GBDT and XGBoost. Compared to previous studies on predicting house prices, this research used a hyperparametric optimization approach which is Bayesian

Optimization to select the most suitable parameters increase the accuracy of the model prediction and mitigate model overfitting.

This study found that GBDT and XGBoost have excellent performance in predicting house prices. Hence, this paper demonstrates that machine learning algorithms can enhance the forecastability of the sale prices of housing and make a considerable contribution to the accurate evaluation of property prices. Besides, according to the model used in this paper it can be summarized that the following five characteristics have the greatest impact on the housing sale price. They rated the overall materials and finishes of the house, the above-ground (ground) living, the size of the car capacity of the garage, the square footage of the first category of finish, and assessed the quality of the exterior materials.

This paper has several implications. In real applications, machine learning algorithm-based housing price prediction models are used by banks and financial institutions for better house price assessment, risk analysis and lending decisions. Possible advantages of utilizing machine learning models contain lowering the expense of property analysis and enabling those in need of loans to obtain them faster. On the other hand, the public can refer to these characteristics to initially determine the price of their house. When people want to buy house, they can also look at these features of the house to determine whether the house matches its price.

The shortcoming of this study is that the temporal characteristics of the house are not specially treated. Because the temporal characteristics will contain a lot of special information, it would be more accurate for house price prediction if this point is taken into account in future studies. In addition, this paper does not use the neural network model, if the neural network model can be added to the house price prediction will get more accurate results. The forecast of house sale prices would be more precise if features could be derived from significant characteristics analyzed by the model.

References

[1] Chen, Yihao, Runtian Xue, and Yu Zhang. "House price prediction based on machine learning and deep learning methods." 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS). IEEE, 2021.

[2] Shah, Vatsal H. "Machine learning techniques for stock prediction." Foundations of Machine Learning| Spring 1.1 (2007): 6-12.

[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*)*, 2018, pp. 35-42, doi: 10.1109/iCMLDE.2018.00017.

[4] Z. Yaping and Z. Changyin, "Gene Feature Selection Method Based on ReliefF and Pearson Correlation," 2021 3rd International Conference on Applied Machine Learning (ICAML), 2021, pp. 15-19, doi: 10.1109/ICAML54311.2021.00011.

[5] Berrar, D. (2018). Cross-validation. Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics, 1–3(January), 542–545.

[6] Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams. "Practical bayesian optimization of machine learning algorithms." Advances in neural information processing systems 25 (2012).

[7] Klein, Aaron, et al. "Fast bayesian optimization of machine learning hyperparameters on large datasets." Artificial intelligence and statistics. PMLR, 2017.

[8] Yoo, K., Yoo, H., Lee, J.M. et al. Classification and Regression Tree Approach for Prediction of Potential Hazards of Urban Airborne Bacteria during Asian Dust Events. Sci Rep 8, 11823 (2018). https://doi.org/10.1038/s41598-018-29796-7

[9] De Aquino Afonso, B. K., Melo, L. C., de Oliveira, W. D. G., Da Silva Sousa, S. B., & Berton, L., (2020). Housing prices prediction with a deep learning and random forest ensemble [Unpublished manuscript]. Anais do Encontro Nacional de Inteligencia Artificial e Computacion.

[10] Winky K.O. Ho, Bo-Sin Tang & Siu Wai Wong (2021) Predicting property prices with machine learning algorithms, Journal of Property Research, 38:1, 48-70, DOI: 10.1080/09599916.2020.1832558

[11] Schapire, Robert E. "Explaining adaboost." Empirical inference. Springer, Berlin, Heidelberg, 2013. 37-52.

[12] D. Yu, Z. Wang and W. Wei, "House Price Prediction Based on a Machine Learning Model," 2021 2nd International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), 2021, pp. 391-395, doi: 10.1109/AINIT54228.2021.00082.

[13] DMLC. xgboost. GitHub. https://github.com/dmlc/xgboost (accessed June 26, 2022).