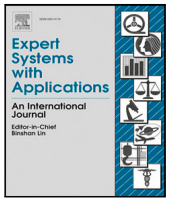
Expert Systems With Applications 233 (2023) 120981

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



# A hybrid machine learning framework for forecasting house price

Choujun Zhan [a,b], Yonglin Liu [b], Zeqiong Wu [b], Mingbo Zhao [b,c,*], Tommy W.S. Chow [c]

[a] School of Computing, South China Normal University, Guangzhou, Guangdong, China
[b] School of Electrical and Computing Engineering, Nanfang College, Guangzhou, Guangdong, China
[c] Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China

ARTICLE INFO

*Keywords:*
Machine learning
House prices
Stacking
Bagging
Transformer

ABSTRACT

House price prediction is one of the most important factors affecting national real estate policies. However, developing an accurate housing price prediction model is a significant challenge for the real estate market. This study presents a framework of house price prediction models that address this issue by improving forecasting performance, explicitly demonstrating the novelty in the Hybrid Bayesian Optimization (HBO) models combined with Stacking (HBOS), Bagging (HBOB), and Transformer (HBOT) techniques. These hybrid models employ Bayesian Optimization for hyperparameter tuning, leading to superior prediction accuracy and stability. Additionally, the proposed framework can assess a statistical and accurate assessment of the predictive performance of house price forecasting models in different scenarios. Furthermore, we constructed a multi-source dataset containing 1,898,175 transactions of the Hong Kong real estate market covering a period from January 2, 1996, to May 13, 2021. This dataset, another major contribution to the field, enables comprehensive model testing and could be a valuable resource for future research. Then, the proposed hybrid models are compared with 18 benchmark models. Thirteen evaluation metrics are used to evaluate the predictive performance, while the non-parametric testing, including Friedman, Iman–Davenport, and Nemenyi post-hoc tests methods, are adopted to assess the significance of differences in the predictive performance of each model. The experimental results show that the HBOS models are superior to the other benchmark models for application in the house price prediction problem. The HBOS-CatBoost model showed superior performance in terms of RMSE compared to both the HBOB-XGBoost and HBOT-ConvLSTM models, with relative RMSE reductions of 5.11% and 25.56%, respectively. The main contributions of this work are the creation of a rich multi-source dataset, the proposal of novel hybrid models for improved house price prediction, and a comprehensive performance evaluation framework. These findings offer a significant step forward in the housing price prediction field.

## 1. Introduction

For many individuals and families, residential housing is one of the most important living resources and properties. While for cities, houses not only provide living space for people in society but also serve as an attractive market for investors, especially in international metropolis (Chen, Ong, Zheng, & Hsu, 2017). The development of the real estate market can highly influence the economic activities of a country or even the entire world. For instance, the US subprime mortgage crisis in 2007 triggered a severe global financial crisis, which caused huge asset losses to financial markets and organizations. This crisis brought about the shrinkage of bank credit and the depreciation of the real estate market, resulting in slowing down economic development all over the world (Hodson & Quaglia, 2009). Hence, the healthy development of a real estate market is one of the most important factors strongly associated with the sustainable development of the economy. The real estate market is not just an important issue for individuals and businesses but also for socioeconomic stability. Therefore, house price prediction has attracted significant attention from various fields, including economy, politics, computer science and etc. (Iacoviello & Minetti, 2008; Park & Bae, 2015; Selim, 2009; Tsai, 2013). However, the main challenge is that the real estate prices prediction is a nonlinear time series forecasting issue for complex systems, which is influenced by various factors, including economic factors (such as income ratio, housing vacancy rate, income inequality and etc.), house attributes (such as house age, the number of transactions, living area, parking spaces, number of rooms and geographic location) and crisis events (such as pandemic and economic crisis). Part of the studies focuses on determining factors related to house prices. Researchers investigate

* Corresponding author at: Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China.
*E-mail addresses:* zchoujun2-c@my.cityu.edu.hk (C. Zhan), nfsysuliuyonglin@gmail.com (Y. Liu), wuzeqiong@gmail.com (Z. Wu), mzhao4-c@my.cityu.edu.hk (M. Zhao), eetchow@cityu.edu.hk (T.W.S. Chow).

https://doi.org/10.1016/j.eswa.2023.120981
Received 23 December 2022; Received in revised form 9 June 2023; Accepted 9 July 2023
Available online 14 July 2023

the relationship between house prices and various factors, including monetary policy (Tsai, 2013), labor market (Fischer, Huber, Pfarrhofer, & Staufer-Steinnocher, 2021), investment and consumption (Benetton, Bracke, Cocco, & Garbarino, 2022), and etc. In this work, we aim to develop a state-of-art framework for house price prediction and conduct an extensive experimental comparison for assessing a statistical and accurate assessment of the predictive performance of house price forecasting models on different datasets.

Accurately predicting house prices can greatly assist real estate market analyses and place-based decisions. Hence, researchers have developed various housing price prediction models, roughly classified into three categories: (1) Classical linear regression model. The Auto Regressive Integrated Moving Average (ARIMA) is a time series prediction statistical modeling model that is frequently used as a baseline model for house price forecasting tasks. Research has demonstrated that in terms of forecasting, deep learning and machine learning models generally outperform ARIMA models (Abidoye, Chan, Abidoye, & Oshodi, 2019; Temur, Akgün, & Temur, 2019); (2) Hedonic price models. The hedonic price model can be used to determine the relationship between attributes and house prices (Goodman, 1978). Then, the hedonic pricing model has be applied to predict house prices (Harding, Knight, & Sirmans, 2003; Maurer, Pitzer, & Sebastian, 2004); (3) Machine learning models. Due to the inability of hedonic price models to capture non-linear relationships between features (Limsombunchai, 2004), machine learning (ML) methods are employed to discover potential non-linear relationships in time series. Researchers employ machine learning techniques to forecast property prices, which always show a better predictive performance (Abidoye & Chan, 2018; Rahman, Maimun, Razali, & Ismail, 2019). These techniques include artificial neural networks (ANN) (Abidoye et al., 2019), ensemble learning models (Kang et al., 2020; Park & Bae, 2015), and deep learning models (Ge, 2019; Law, Paige, & Russell, 2019).

Studies have discovered that hybrid models can combine the strengths of different models to obtain higher prediction accuracy (Polley & Van Der Laan, 2010). Different ensemble approaches, including Bagging, Boosting and Stacking, have been applied in the research. Bagging and Boosting focus on reducing variance and bias, respectively, while Stacking achieves the two goals of reducing variance and bias by finding the optimal way to combine algorithms (Large, Lines, & Bagnall, 2019). In addition, hyperparameters are essential for machine learning models, as they have a significant impact on the effectiveness of machine learning models and directly influence predictive performance. However, it is difficult to optimize hyperparameters using traditional methods such as grid search or random search (Bischl et al., 2021). Therefore, designing an effective strategy to determine the optimal hyperparameters of the predictive model is a crucial step in enhancing the predictive power During the past decade, various techniques for hyperparameter optimization have been proposed, including Bayesian optimization algorithms (Wang, Wang, & Peng, 2021), genetic algorithms (Zhou, Huang, Wang, & Qiu, 2021), differential evolutionary algorithms (Wang, Zeng, & Chen, 2015) and particle swarm optimization algorithms (Zhang, Zhang, & Zhang, 2020). Therefore, our work will aim to employ hyperparameter optimization techniques and ensemble approaches to develop hybrid house price prediction models with high predictive performance.

Recently, various methods have been proposed for dealing with house price prediction problems. However, it is well known that the performance of a predictive model is heavily dependent on the available datasets. For evaluating the predictive performance of a model, researchers commonly adopt one or several evaluation metrics. However, one method may have a different evaluation as we adopt different evaluation metrics. The adoption of evaluation metrics has an impact on the evaluation of model performance. Improper or insufficient usage of evaluation metrics may lead to a biased assessment of the overall performance of a model. Hence, it is important to conduct statistical tests on the result (Demšar, 2006). Our work will also focus on deriving a statistical and accurate assessment of the predictive performance of house price forecasting models in various scenarios.

In conclusion, this research proposes a novel framework for predicting housing prices, consisting of three different hybrid models based on the stacking, bagging, and transformer approaches. Meanwhile, the hyperparameters of the models are globally optimized by the bayesian optimization algorithm to improve the forecasting accuracy of the models. Finally, we construct a framework for comparing the predictive performance of the 27 house price forecasting models through the Friedman, Iman–Davenport and Nemenyi post-hoc tests methods.

Our contributions are three-fold:

- We constructed a multi-source dataset containing 1,898,175 housing transactions across three regions of Hong Kong (Kowloon, Hong Kong Island, and the New Territories), which includes 17 housing properties and 10 economic factors and covers the period from 1996 to 2021. This dataset provides a valuable resource for researchers and practitioners interested in studying the property market in Hong Kong, which represents a significant contribution to the field. Furthermore, our work identifies the relationship between house prices and various influencing factors. For instance, we noted that monetary policies heavily influence house prices in affluent districts, while their impact is less pronounced in less wealthy areas.
- We found that the constraints of data (short time span, limited vector length, and high similarity among price trends across regions) often lead to reduced prediction accuracy. In response to the unique challenges posed by the limited length and high similarity of house price vectors, we innovatively leveraged a framework of hybrid models for house price prediction, namely, we proposed three types of hybrid models—Hybrid Bayesian Optimization with Stacking-based (HBOS), Bagging-based (HBOB), and Transformer-based (HBOT) models. The framework of house price prediction models integrates diverse machine learning methods and utilizes a Bayesian Optimization algorithm for hyperparameter optimization, thus addressing the data limitations and enhancing prediction accuracy and stability. We propose these hybrid models to provide more robust and reliable methods for analyzing and predicting house prices in real estate markets. This framework contributes a novel perspective for improving house price predictions, particularly in contexts with similar data constraints.
- We proposed a comparative framework for assessing a statistical and accurate assessment of the predictive performance of house price forecasting models on different datasets. This framework contains three non-parametric tests, i.e., Friedman, Iman–Davenport and Nemenyi post-hoc tests, for evaluating and ranking the predictive performance of 27 house price forecasting models on three Hong Kong house price datasets by 13 evaluation metrics. Our analysis demonstrated the effectiveness of the proposed hybrid models in predicting house prices, thereby demonstrating the practical implications of our work.

This research contributes to the field by elucidating the relationships between housing prices and various factors, providing new hybrid models to overcome data constraints, and offering a robust comparative framework for evaluating predictive performance. These advancements provide valuable insights and tools for scholars and practitioners aiming to better understand and predict housing market trends.

The remainder of this paper is organized as follows. Section 2 reviews relevant research about house price forecasting. The data description and analysis are presented in Section 3. Section 4 introduces the details of our proposed house price forecasting framework. In Section 5, the evaluation measures, statistical evaluation and detailed experimental results are described, while Section 6 presents the conclusion of the entire study and the direction of future research.

## 2. Literature review

In this section, we discuss the existing work on the housing price prediction problem. Here, we mainly focus on works about hedonic price models and machine learning models.

*Classical hedonic price models:* Early house price prediction models mainly focused on hedonic price regression, which identified price determinants based on internal attributes of the good being sold as well as external factors influencing it (Rosen, 1974). For instance, Yang proved that the hedonic price models could be used to estimate house prices in Beijing, and further emphasized the strong influence of building quality on house prices (Yang, 2001). Maurer et al. adopted the hedonic model for analyzing 84,686 transactions in the real estate market in Paris (France), which considers the heterogeneous character of real estate (Maurer et al., 2004). Recently, Yang et al. applied hedonic pricing models to investigate the correlation between Urban rail transit accessibility and property price (Yang et al., 2020). The results indicated that hedonic indices could describe the real estate market characteristics on the basis of transaction prices. In summary, classical hedonic price models have been widely used in real estate research to identify price determinants based on internal attributes and external factors. At the same time, a few studies attempted to utilize the hedonic price model for forecasting house prices.

*Machine learning models:* With the development of machine learning techniques, researchers try to adopt machine learning models for predicting house prices. In the early stage of research on developing house price prediction models, The ANN models are wildly adopted (Lam, Yu, & Lam, 2008; Limsombunchai, 2004; Selim, 2009; Wilson, Paris, Ware, & Jenkins, 2002). Wilson et al. trained ANN models to predict the future trend of the real estate market based on the national housing transaction data (Wilson et al., 2002). Additionally, researchers compared the predictive performance of the classical hedonic regression model and the ANN model. Studies indicate that the hedonic price models show poor predictive ability in the out-of-sample scenario (Limsombunchai, 2004) and ANN has a better predictive performance for forecasting house prices (Selim, 2009). Then, a hybrid model by combining the ANN model and the entropy model was proposed, which also performed well (Lam et al., 2008). Recently, deep learning models have attracted extensive attention from scholars. Law et al. developed a deep neural network model to automatically extract visual features from images to estimate housing prices in London, England (Law et al., 2019). Similarly, Ge constructed a data set of community housing prices in New York and Beijing, and also proposed a hybrid model based on CNN and LSTM to capture the spatial and temporal patterns of community house prices, then estimate the house price (Ge, 2019).

Ensemble learning models have also been widely adopted to establish forecasting models. Researchers adopted the classical C4.5, RIPPER, Naïve Bayesian, and AdaBoost model for forecasting house prices (Adetunji et al., 2022; Park & Bae, 2015). Experimental results indicate that ensemble learning models also have a accurate forecasting performance. Scholars also try to combine ensemble learning models and deep learning models to develop hybrid models to enhance the forecasting accuracy. For instance, Kang et al. tried to extract high-level visual features from street view images and house photos by a deep learning model. Then, a data fusion framework based on Multiple Linear Regression (MLR) and Gradient Boosting Machine (GBM) models are proposed for predicting house price appreciation potentials (Kang et al., 2020).

In conclusion, hedonic price models and machine learning techniques were primarily used in the research to address the problem of predicting house prices. However, hedonic price models have difficulty in capturing non-linear relationships between features (Limsombunchai, 2004). Additionally, hybrid models were employed in a small portion of the reviewed literature, and there is a lack of work to perform a comprehensive comparison between various classical house price prediction models. To fill this gap, we propose a framework that includes different hybrid models and conduct extensive experimental comparisons. Through the proposed framework and experimental comparisons, this work aims to contribute to the field of house price prediction by providing a more effective and comprehensive approach to addressing this important problem.

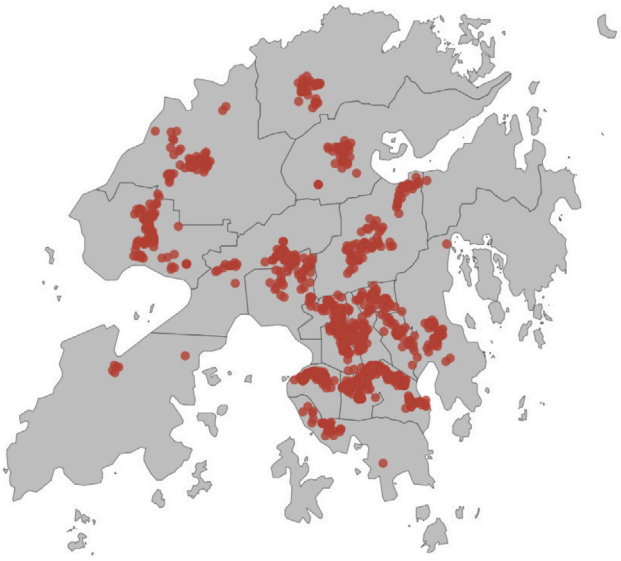


**Fig. 1.** Distribution map of each estate. Red dots indicate the location of each estate.

## 3. Data description and analysis

Hong Kong is a metropolis located in the eastern Pearl River Delta in South China, which has one of the most active real estate markets in the world. In this section, we will introduce the data about the Hong Kong real estate in detail, which includes the data collection, processing and analysis.

### 3.1. Data collection

The house attribute and economic factor data were sourced from the Hong Kong real estate market (https://www.28hse.com/) and the Hong Kong Monetary Authority website (https://www.hkma.gov.hk/), respectively.

#### 3.1.1. House attributes

The data of the housing information is updated daily by the Hong Kong real estate market website, which consists of 1,898,175 records from January 2, 1996, to May 13, 2021. That is, a 26-year period dataset includes 95 districts and 3662 estates in Hong Kong. Here, the distribution map of each estate is shown in Fig. 1.

The housing attributes describe the basic characteristics of the housing unit, including the location of an estate (the longitude $x_{lon}$ and the latitude $x_{lat}$), the transaction date $x_{td}$, the district where the house belongs $x_d$, the estate $x_e$, the change of price $x_c$ (the ratio of profit or loss when sale second-hand housing units), the gross area (in squared feet) $x_{ga}$, the usable area (in squared feet) $x_{ua}$, the unit price of usable area $x_{upua}$, the total transactions of a unit $x_{ttu}$, the occupancy date $x_{od}$, the school district $x_s$, the annual total transaction $x_{at}$, the contract form of agreement $x_{cf1}$, the contract form of assignment $x_{cf2}$, the contract form of speculation $x_{cf3}$, the unit price of gross area $x_p^j$. and total price $x_{tp}$. Table 1 summarizes these 17 house attributes, while Table 2 summarizes the 10 economic factors.

**Table 1**
Description of house attributes.

| Variable | Description |
|---|---|
| $x_{td}$ | Transaction date |
| $x_d$ | District |
| $x_e$ | Estate |
| $x_c$ | Change of price |
| $x_{ga}$ | Gross area |
| $x_{ua}$ | Usable area |
| $x_{upua}$ | Unit price of usable area |
| $x_{cf1}$ | Estate contract form of agreement |
| $x_{cf2}$ | Estate contract form of assignment |
| $x_{cf3}$ | Estate contract form of speculation |
| $x_{ttu}$ | Total transaction for the unit |
| $x_{od}$ | Occupancy date |
| $x_s$ | School district |
| $x_{lon}$ | The longitude of the estate |
| $x_{lat}$ | The latitude of the estate |
| $x_p^j$ | Unit price of gross area |
| $x_{at}$ | Annual total transaction |

**Table 2**
Description of economic factors.

| Variable | Description |
|---|---|
| $m_1$ | Money supply $m_1 = m_{1,1} + m_{1,2}$ |
| $m_{1,1}$ | Legal tender notes and coins in hands of public |
| $m_{1,2}$ | Demand deposits with licensed banks |
| $m_2$ | Money supply $m_2 = m_1 + m_{2,1} + m_{2,2} + m_{2,3}$ |
| $m_{2,1}$ | Savings deposits with licensed banks |
| $m_{2,2}$ | Time deposits with licensed banks |
| $m_{2,3}$ | NCDs issued by licensed banks and held by public |
| $m_3$ | Monely supply $m_3 = m_2 + m_{3,1} + m_{3,2}$ |
| $m_{3,1}$ | Deposits with RLBs and DTCs |
| $m_{3,2}$ | NCDs issued by RLBs and DTCs and held by public |

#### 3.1.2. Economic factors

The economic factors include 10 Hong Kong money supply indicators, including legal tender notes and coins in the hands of the public $m_{1,1}$, demand deposits with licensed banks $m_{1,2}$, money supply $m_1$, savings deposits with licensed banks $m_{2,1}$, time deposits with licensed banks $m_{2,2}$, NCDs issued by licensed banks and held by public $m_{2,3}$, money supply $m_2$, deposits with RLBs and DTCs $m_{3,1}$, NCDs issued by RLBs and DTCs and held by public $m_{3,2}$, money supply $m_3$. In Hong Kong, the money supply is a combination of $m_{1,1}$ and $m_{1,2}$; money supply $m_2$ is a combination of money supply $m_1$, $m_{2,1}$, $m_{2,2}$ and $m_{2,3}$; money supply $m_3$ is a combination of money supply $m_2$, $m_{3,1}$ and $m_{3,2}$.

### 3.2. Data processing and correlation analysis

First, we use Box Plots (Kwak & Kim, 2017) to remove extreme outliers of variables in the data of Kowloon, Hong Kong Island, and New Territories. Then, we adopt a Multiple Imputation (MI) (Rubin, 1978) method to impute the missing values in the data. Additionally, the variable of contract form is a categorical feature, and there are three categories: agreement, assignment and speculation. We have to convert the categorical variable of contract form into continuous variables for further analysis. Then, the categorical variables were processed through the Weight of Evidence (WOE) method (Wod, 1985), which allow us to capture the non-linear and non-monotonic relationship between the contract form variable and the target variable. Finally, the categorical variable of contract form was converted into three continuous variables ($x_{cf1}$, $x_{cf2}$, $x_{cf3}$).

Pearson correlation analysis is adopted to derive the correlation between various factors (including house attributes and economic factors) and the house price in the three regions of Hong Kong. Experimental results indicate that there is a strong correlation between the money supply and the unit price of gross area $x_p^j$ on Hong Kong Island. The correlation coefficients between Legal tender notes and coins in the hands of the public $m_{1,1}$, Demand deposits with licensed banks $m_{1,2}$, money supply $m_1$, Savings deposits with licensed banks $m_{2,1}$, Time deposits with licensed banks $m_{2,2}$, NCDs issued by licensed banks and held by public $m_{2,3}$, $m_2$, $m_3$ and the target variable $x_p^j$ are 0.83, 0.81, 0.82, 0.82, 0.78, 0.55, 0.83, 0.83, respectively. Obviously, the money supply has the greatest impact on the house price in Hong Kong Island, followed by Kowloon, and New Territories. Note that the Hong Kong Island is the richest district in Hong Kong, followed by Kowloon and New Territories. The results show that the influence of factors differs by districts. Furthermore, the correlation between monetary policy and housing prices in poor districts is weak.

## 4. Methodology

In this section, we first define the problem description for house price forecasting. Then, we propose a framework for house price prediction based on machine learning and Bayesian optimization. In this framework, we investigate the performance enhancement of three different type of hybrid models (HBOS, HBOB and HBOT) for house price forecasting, with a total of 9 hybrid models. In addition, the framework applies 18 benchmark models for comparison. 13 evaluation measures are adopted to evaluate the predictive performance of the 27 models in the framework. Furthermore, the model results are analyzed by non-parametric statistical tests Friedman, Iman–Davenport and Nemenyi post-hoc tests to achieve an evaluation of model performance.

In this study, the goal is to predict the house prices of each estate in the next year based on historical house attributes and economic factors. Without loss of generality, we set the unit price of the gross area as the house prices of an estate. Then, the house prices prediction can be treated as a supervised machine learning task and the house prices prediction problem can be defined as follows:

- Given a time series containing $N$ annual data points $X(t) = \{x_{t-N+1}, \dots, x_{t-1}, x_t\}$, where $x_t$ contains the house attributes $h_t$ and economic factors $m_t$, namely, $x_t = \{h_t, m_t\}$:

$$h_t = \{x_{td}, x_d, x_e, x_c, x_{ga}, x_{ua}, x_{upua}, x_{cf1}, x_{cf2}, x_{cf3}, x_{ttu}, x_{od}, x_s, x_{lon}, x_{lat}, x_{at}, x_p^j\},$$
$$m_t = \{m_1, m_{1,1}, m_{1,2}, m_2, m_{2,1}, m_{2,2}, m_{2,3}, m_3, m_{3,1}, m_{3,2}\}.$$

  As stated above, $h_i \in \mathbb{R}^{17\times1}$ and $m_i \in \mathbb{R}^{10\times1}$. Hence, $X(t) \in \mathbb{R}^{N\times27}$. These features are normalized as follows:

$$X' = \frac{X - \bar{X}}{\sigma}, \tag{1}$$

  where $X$ is the original feature vector, namely $X = \{h_t, m_t\}$. $\bar{X}$ represents the average value of the feature vector, and $\sigma$ is the standard deviation of the feature vector.
- In order to predict the house prices for the next year with historical data of $k$ years, the input of the supervision model is $X(t-k), X(t-k+1), \dots, X(t-1)$. The goal of the house prices prediction model is to learn the non-linear mapping of the target value $y(t+n)$:

$$y(t+n) = F(X(t-k), X(t-k+1), \dots, X(t-1)), \tag{2}$$

  where $F(\cdot)$ is the non-linear mapping function, which stands for the predictive model.

### 4.1. An overview of the proposed framework

The framework of our proposed prediction models is shown in Fig. 2, which contains three types of hybrid models (HBOS, HBOB, and HBOT) and a total of nine hybrid models. To combine different machine learning methods, the hybrid models HBOS and HBOB employ stacking and bagging strategies, respectively. Stacking is a mechanism that attempts to find the optimal combination of a set of prediction algorithms (Van der Laan, Polley, & Hubbard, 2007), while

Bagging is a machine learning method that uses bootstrap aggregation predictor integration and is extremely sensitive to small changes in the training data. As a result, they have the potential to improve prediction performance (Shirzadi et al., 2018). Transformer is the existing mainstream deep learning network (Vaswani et al., 2017), which adds a self-attention mechanism to learn sequential information and can better capture the global input–output dependencies. The self-attention mechanism enables faster training and a more direct flow of information throughout the sequence, resulting in a more direct gradient flow (Zeyer, Bahar, Irie, Schlüter, & Ney, 2019). Then, a hybrid model (HBOT), consisting of the Transformer encoder with different decoders, is proposed. Furthermore, when comparing house price prediction models, we train 9 hybrid models and 18 benchmark models on each of Hong Kong's three regional datasets, which compensates for the shortcomings of most existing studies in which training only a single type of model can lead to biased results. In addition, we use the BO method to optimize the hyperparameters of these 27 models in order to improve model performance. In addition, we use the BO method to optimize the hyperparameters of the models in order to improve model performance. Additionally, 18 benchmark models are adopted for comparing with the proposed hybrid models.

There has been a great deal of research about the problem of house price forecasting. However, most of the relevant studies on the performance analysis of house price forecasting algorithms are limited to identifying the optimal performance under certain evaluation metrics, and rarely provide a comprehensive comparative analysis of the performance of various algorithms. Hence, it is difficult to analyze the overall performance of the methods and more fair statistical analysis conclusions are required. We propose a model comparison framework for measuring the predictive performance to obtain comparative performance of different forecasting methods. Firstly, the framework introduces 13 evaluation metrics to evaluate the predictive results of each models. Then, the predictive performance of each model is ranked by nonparameteric statistical tests.

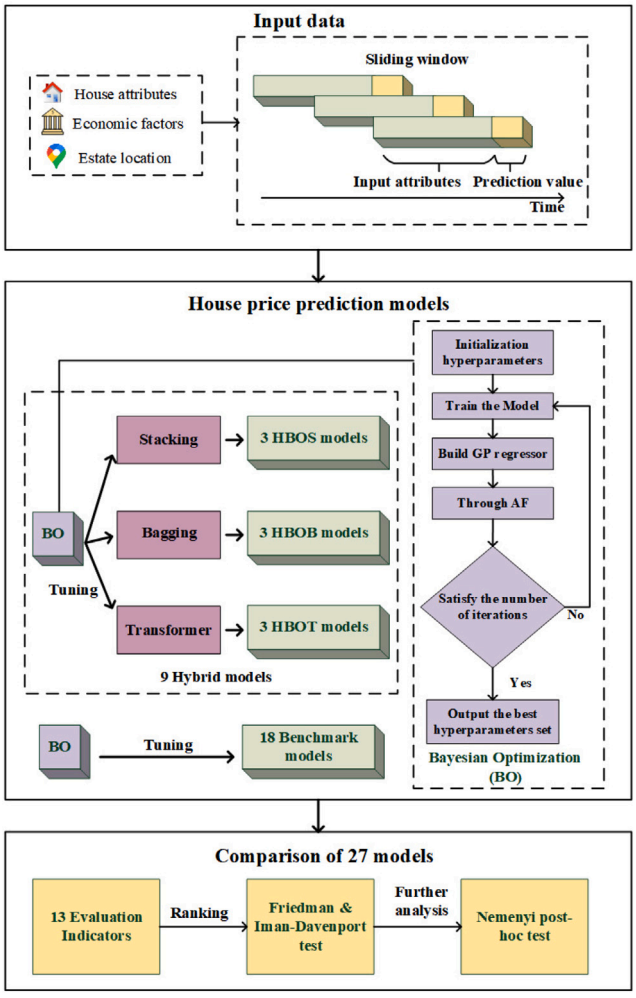



**Fig. 2.** The framework of the proposed house price prediction. The time series dataset is preprocessed using a sliding window approach. Subsequently, the research employs a total of 27 house price prediction models, including 9 proposed hybrid models and 18 benchmark models. Finally, statistical methods are applied to rank and analyze the predictive performance of these models.

### 4.2. Bayesian Optimization (BO)

Hyperparameters play a crucial role in the performance of machine learning algorithms, but manual selection through trial and error is inefficient, subject to bias, errors, and computational irreproducibility (Bischl et al., 2021). Therefore, in order to increase the model's robustness and accuracy, we optimize the model hyperparameters selection process using the BO approach. The basic idea of BO is to assess the posterior distribution of an unknown function through the acquisition function, and then select the best hyperparameters based on the Acquisition Function (AF). In this work, the Gaussian process is used as a prior function, while the acquisition function is based on Expected Improvement (EI). The optimization goal of BO is to maximize the objective function:

$$f(p) = \arg\max_{p \in \mathbb{P}_{\mathbb{BO}}} \left(1 - \frac{\sum(y^i - \hat{y}^i(p))^2}{\sum(y^i - \overline{y})^2}\right) \tag{3}$$

where $p$ represents a set of hyperparameters in the model, such the learning rate, the maximum tree depth, the minimum child weight and etc., while $\mathbb{P}_{\mathbb{BO}}$ stands for candidate hyperparameter sets. $y^i$ is the true house price of the $i$th sample, $\hat{y}^i(p)$ means the predicted house price of the $i$th sample with the hyperparameters $p$, and $\overline{y}$ is the average house price.

The BO process of model is represented as follows:

1. Initialize the hyperparameter set of the model $\mathbb{P}_{\mathbb{BO}} = \{p_1, p_2, \ldots, p_i\}$, input the hyperparameter set into the model for training, and obtain the results of the model in the evaluation index $Y = \{y_1, y_2, \ldots, y_i\}$. Then, the dataset of BO $D_{BO} = \{(p_1, y_1), (p_2, y_2), \ldots, (p_i, y_i)\}$ is obtained.
2. The data set $D_{BO}$ is modeled by Gaussian Process (GP), and the $GP$ is expressed as:

$$P(\mathbb{P}_{\mathbb{BO}}) \sim \mathcal{N}(\mu(p_i), k(p_i, p_i')), \tag{4}$$

where $\mu(p_i)$ is the mean function and $k(p_i, p_i')$ is the covariance function.
3. The next set of hyperparameter $p_{i+1}$ is selected using the collection function AF. Here, we choose the EI function $a_{EI}$, whose mathematical expression is as the Eq. (5):

$$a_{EI}(p_i) = \mu(p_i) \cdot k(p_i)\varphi(k(p_i)) + \mathcal{N}(k; 0, 1), \tag{5}$$

where $\varphi$ is the normal cumulative distribution function. The EI function uses the hyperparameter with the largest expected value as the next group of parameters $p_{i+1}$, as shown in the Eq. (6):

$$p_{i+1} = \arg\max_{p_i \in \mathbb{P}_{\mathbb{BO}}} a_{EI}(p_i). \tag{6}$$

Then, update the data set as shown in the Eq. (7):

$$D_{BO} = \{(p_1, y_1), (p_2, y_2), \ldots, (p_i, y_i), (p_{i+1}, y_{i+1})\}. \tag{7}$$

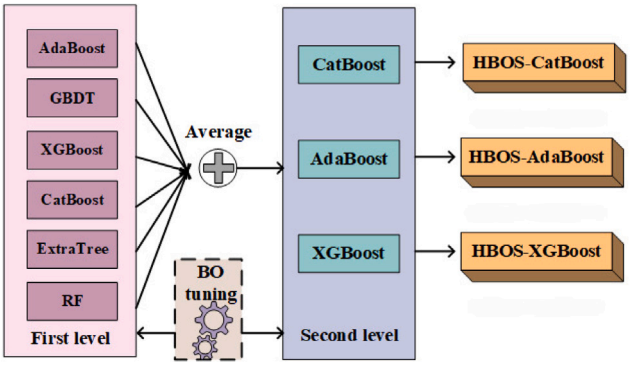



**Fig. 3.** The overall process of the HBOS model. HBOS models include a stacking-based ensemble model and BO technology. Six first-level learners are used, and XGBoost, CatBoost, and AdaBoost algorithms are applied as the meta-learner algorithm in the second level. Three hybrid models are constructed: HBOS-XGBoost, HBOS-CatBoost, and HBOS-AdaBoost.

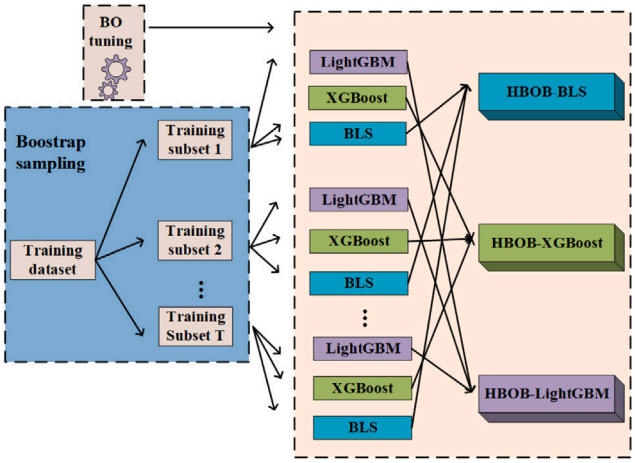



**Fig. 4.** The overall process of the HBOB model. HBOB models employ the BO technique with the Bagging strategy. BLS, XGBoost, and LightGBM are used as regressors for the Bagging process, resulting in three hybrid models: HBOB-BLS, HBOB-XGBoost, and HBOB-LightGBM.

4. Repeat step-2 to step-3 for achieving the optimal hyperparameter set $p_{best}$.

The optimization procedure is summarized in Algorithm 1.

**Algorithm 1** Bayesian Optimization

**Input** : Hyper parameter sets $\mathbb{P}_{\mathbb{BO}} = \{p_1, p_2, ..., p_i\}$ ;
Target optimizate model $M$ ;
Gaussian Process: $GP$;
Acquisition Function: $a_{EI}$;
Number of iterations: $e$.
**Output:** The best Parameter $p_{best}$
Step1: Initialization
**for** $i = 0$ to $\mathbb{P}_{\mathbb{BO}}$ **do**
Calculate the value of output result based on Training Model $M$
**end for**
Get Bayesian Optimization Dataset:
$D_{BO} = \{(p_1, y_1), (p_2, y_2), ..., (p_i, y_i)\}$
**for** $i \leftarrow D_{BO}$ to $e$ **do**
Step2: Build GP Regressor
Build GP $P(\mathbb{P}_{\mathbb{BO}})$ based on Eq. (4)
Step3: Extract next parameter set $p_{i+1}$ through AF $a_{EI}$
$p_{i+1} = \arg\max_{p_i \in \mathbb{P}_{\mathbb{BO}}} a_{EI}(p_i)$ based on Eq. (6)
$y_{i+1} = f(p_{i+1})$
Update $D_{BO} \leftarrow D_{BO} \cup (p_{i+1}, y_{i+1})$
**end for**
**return** $p_{best}$

### 4.3. Hybrid BO with Stacking-based (HBOS) method

The HBOS models compose of a stacking-based ensemble machine learning model and BO technology, as shown in Fig. 3. We combine multiple base learners into a two-layer integrated model based on a Stacking strategy, with each learner using Bayesian optimization to configure the corresponding hyperparameters automatically. There are two levels in the Stacking-based framework. In the first level, we use six first-level learners, including AdaBoost, GBDT, XGBoost, CatBoost, ExtraTree and RF. For the second level, we apply XGBoost, CatBoost and AdaBoost algorithms as the meta learner algorithm, respectively. In addition, we compare the effects of different meta learners on the prediction effect. Therefore, there are three hybrid BO with stacking-based models: HBOS-XGBoost, HBOS-CatBoost and HBOS-AdaBoost.

Assuming that the input sample size is $m$, the training data set is $D = \{x^i, y^i\}_{i=1}^m$, the learner algorithm of the first level is $\zeta$, and the meta-learner algorithm is $\epsilon$. In the first step, the first-level learner algorithm $\zeta$ is trained using the training dataset $D$, and the output $h_f$ is then obtained for the first level. In the second step, for each sample $x^i$ in the training data set $D$, the new data set generated by the first learner is $D'$. In the third step, the new dataset $D'$ is trained based on the meta-learner algorithm $\epsilon$ to get the output of the meta-learner $h'_f$ in the second layer. So the output of the final model is $H' = h'_f(h_1(x^i), h_2(x^i), \ldots, h_F(x^i))$.

The process of developing HBOS models includes the following parts:

- Initialization: Initialize the hyperparameters of the HBOS models by the BO, including first and second-level learners. We first optimize the model hyperparameters of the first-level learner, followed by the meta learner. Therefore, the model hyperparameters set in the first level are as follows:

$$p_i^{1st} = \{p_i^{1st}(AdaBoost), p_i^{1st}(GBDT), p_i^{1st}(XGBoost), p_i^{1st}(CatBoost), p_i^{1st}(ExtraTree)\}. \quad (8)$$

  The model hyperparameter set in second level is as follows:

$$p_i^{2nd} = \{p_i^{2nd}(XGBoost), p_i^{2nd}(CatBoost), p_i^{2nd}(AdaBoost)\}. \quad (9)$$

- Trian the First-level Learner: We input the training data set $D = \{x^i, y^i\}_{i=1}^m$ with sample size $m$, and adopt the hyperparameters of first-level learner $p_i^{1st}$ for the AdaBoost, GBDT, XGBoost, CatBoost, ExtraTree and RF models for training, respectively. Iterate $e$ times until the optimal hyperparameters $p_{best}^{1st}$ for the AdaBoost, GBDT, XGBoost, CatBoost, ExtraTree and RF models are found using BO method.
- Generate the new dataset: Using the first-level prediction results from the AdaBoost, GBDT, XGBoost, CatBoost, ExtraTree, and RF models as a new dataset $D' = \{x_F^i, y^i\}_{i=1}^m$, where $F$ is the number of models in the first level and $m$ is the sample size. Moreover, $m$ is the same as the sample size of input dataset $D$. Therefore, the predicted value $y^i$ in the input data set $D$ remains unchanged, and $x_F^i$ stands for the average of the prediction results from the $F$ models.

- Trian the Meta Learner: In this step, we train the XGBoost, CatBoost and AdaBoost models in the meta learner with the hyperparameters of second level learners $p_i^{2nd}$ using new data set $D' = \{x_F^i, y^i\}_{i=1}^m$. At last, looping $e$ iterations until the optimal hyperparameters $p_{best}^{2nd}$ of the XGBoost, CatBoost and AdaBoost models are found.

The HBOS is summarized in Algorithm 2.

**Algorithm 2** The proposed HBOS framework

**Input** : Training Dataset $D = \{x^i, y^i\}_{i=1}^m$;
First-level learner algorithm $\zeta$;
Meta-learner algorithm $\epsilon$.
**Output:** The Stacking result $H^{'}$

Step1: Initialize hyperparameters of the first level $p_i^{1st}$ and the second level $p_i^{2nd}$ models based on Algorithm 1.
Step2: Train the first-level learner $h_f$ based $D$
**for** $f = 1$ to F **do**
$h_f = \zeta_f(D)$ using Algorithm 1 to obtain the optimal hyperparameters $p_{best}^{1st}$.
**end for**
Step3: Generate the new Dataset $D^{'}$ from $D$
**for** $i$ =1 to m **do**
$z_f = h_f(D)$
**end for**
$D^{'} = \{x^{i'}, y^i\}_{i=1}^m$,where $x^{i'} = z_1(x^i), z_2(x^i), ..., z_F(x^i)$
Step4: Train the meta-learner $h_f^{'}$ based $D^{'}$ using Algorithm 1 to obtain the optimal hyperparameters $p_{best}^{2nd}$.
$h_f^{'} = \epsilon(D^{'})$
**return** $H^{'} = h_f^{'}(h_1(x^i), h_2(x^i), ..., h_F(x^i))$

### 4.4. Hybrid BO with Bagging-based (HBOB) method

The HBOB models employ the BO technique with the Bagging strategy, as shown in Fig. 4. Here, we select BLS, XGBoost and LightGBM as regressors for the Bagging process, namely HBOB-BLS, HBOB-XGBoost and HBOB-LightGBM.

Given the training data set is $D = \{x^i, y^i\}_{i=1}^m$ with the sample size of $m$. We use the bootstrap method to randomly sample the training set $T$ times to obtain a sub-sample set $D_t$. Repeat the above process to generate $T$ sub-training sets $D_t = \{D_1, D_2, \dots, D_T\}$, where each $D_t$ has a sample size of $m$. Then, $T$ training subsets $D_t$ are applied to train $T$ base learners $h_t$. Finally, $T$ regressors are aggregated to obtain the predicted values of the integrated model $H$ by averaging, as shown in Eq. (10):

$$H = \frac{1}{T}\sum_{i=1}^{T} h_t. \tag{10}$$

The process of establishing the HBOB models includes the following parts:

- Initialization: We firstly adopt the BO method to initialize the hyperparameters set of the BLS, XGBoost and LightGBM models, which is $p_i(BLS)$, $p_i(XGBoost)$ and $p_i(LightGBM)$, respectively.
- Generate the sub-training datasets: We generate $T$ sub-training sets $D_t = \{D_1, D_2, \dots, D_T\}$ from the training set $D$ using Bootstrap sampling $T$ times.
- Train the sub-model: $T$ sub-training datasets $D_t$ establish $T$ sub-models by utilizing the $p_i(BLS)$, $p_i(XGBoost)$ and $p_i(LightGBM)$, respectively. Then, we combine each sub-model by averaging method Eq. (10). Note that $H$ is the output result of the HBOB models. Looping over $e$ iterations, we finally get the best hyperparameters $p_{best}$ in the HBOB models.

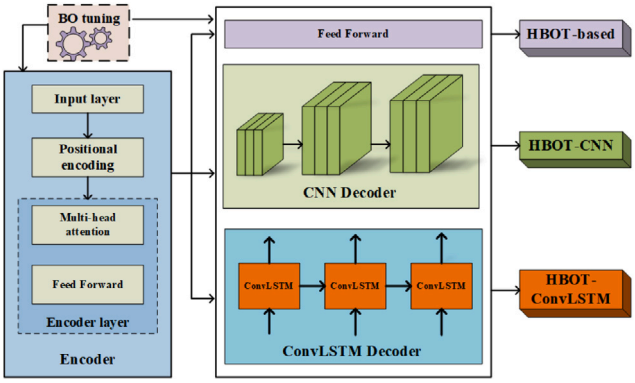



**Fig. 5.** HBOT models consist of three key components: the encoder, the decoder, and the BO module. Specifically, we explore different decoder architectures, including HBOT-based, HBOT-CNN, and HBOT-ConvLSTM, that are combined with the encoder to construct the HBOT models.

The HBOB is summarized in Algorithm 3.

**Algorithm 3** The proposed HBOB framework

**Input** : Training Dataset $D = \{x^i, y^i\}_{i=1}^m$;
Base learning model algorithm $\zeta$;
Epoch : $T$.
**Output:** The Bagging result $H$

Step1: Adopt Algorithm 1 to initialize the hyperparameters set of the HBOB models.
Step2: Generate the sub-training datasets
**for** $i = 1$ to T **do**
Randomly sampling the training dataset $D$ with $T$ times to get sub-sampling dataset $D_t$.
Step3: Train the sub-model
Utilize Algorithm 1 to get the best result $h_t = \zeta(D_t)$
**end for**
**return** $H = \frac{1}{T}\sum_{i=1}^{T} h_t$

### 4.5. Hybrid BO with Transformer (HBOT) method

In this research, we also develop the time series forecasting method based on the Transformer model (Vaswani et al., 2017), called the HBOT model. The overall framework of the HBOT is presented in Fig. 5. The proposed HBOT models have three components, including the encoder, the decoder and the BO. Here, we compose the encoder with different decoders: HBOT-based, HBOT-CNN, and HBOT-ConvLSTM.

The transformer model consists of Encoder and Decoder layers, whereas the proposed HBOT-based model has only an encoder. In addition, the HBOT-CNN and HBOT-ConvLSTM consider the CNN decoder and ConvLSTM decoder, respectively. Finally, the BO is used to adjust the hyperparameters of these models to enhance predictive performance.

An encoder is composed of an input layer, a positional encoding layer, and an encoder layer. The input layer contains a fully connected network, while the position encoding layer with sine and cosine functions. First, the linear transformation of the sine and cosine functions is used to provide the position information of the model, which means that this layer can provide the order information of the time series data. Then, the obtained position information is input to the encoder layer, which includes two sub-layers, a multi-head attention layer and a fully connected forward propagation layer. Lastly, the encoder generates a $d_{model}$-dimensional vector.

Given the input time series $X(t) \in \mathbb{R}^{N\times 27}$, the output of the HBOT model is $\hat{X}(t)$ after passing through the positional encoding layer, as

Eq. (13). Here, we quote the function in Vaswani et al. (2017) to define the positional encoding layer as follows:

$$PE(pos, 2i) = sin(pos/10000^{2i/d_{model}}), \tag{11}$$

$$PE(pos, 2i+1) = cos(pos/10000^{2i/d_{model}}), \tag{12}$$

where $pos$ stands for the position and $i$ represents the dimension of the location feature.

$$\hat{X(t)} = \sigma W^i[PositionEncoding(X(t))], \tag{13}$$

where $\sigma$ is the activation function of the input layer, which can be ReLU, LeakyReLU, Sigmoid and Tanh. The final activation function is optimized by the BO method. $W^i$ represents the weight of the input layer, and $PositionEncoding(X(t))$ is the time series processing of the input layer by the position encoding layer.

Then multi-head self-attention layers take $\hat{X(t)}$ as input, and are computed by following Eq. (14).

$$Q = W^Q \hat{X(t)}, \tag{14}$$

$$K = W^K \hat{X(t)}, \tag{15}$$

$$V = W^V \hat{X(t)}, \tag{16}$$

$$Attention(Q, K, V) = softmax\left(\frac{(QK)^T}{\sqrt{d_k}}\right)V, \tag{17}$$

where $d_k$ represents the number of heads. There are two head attention mechanisms at $dk = 2$. $Q$, $K$ and $V$ are queries, keys, values, while $W^Q$, $W^K$ and $W^V$ are the weights of queries, keys, and values, respectively.

Next, followed by LayerNormalization as Eq. (18), we normalize the hidden layer in the neural network to a standard normal distribution, which can speed up the training speed and accelerate the convergence.

$$Z = Norm(\hat{X(t)} + Attention(Q, K, V)). \tag{18}$$

In the final output layer, two layers of a linear transformation are performed on $Z$, and then matrix $O$ of the output layer is obtained through activation function as $\sigma$ follows:

$$O = \sigma(Linear(Z)) \tag{19}$$

**CNN Decoder:** In addition, the output layer of the HBOT-based model is connected to the CNN decoder to develop the HBOT-CNN model, and the hidden state of the CNN decoding layer is obtained as $\hat{d_{cnn}}(Z)$, as in Eq. (20):

$$\hat{d}_{cnn}(Z) = \sigma(f^{1*3}(MaxPool(Z))), \tag{20}$$

where $\sigma$ is the activation function, $f^{1*3}$ is the convolution operation with $1 * 3$ core size, $MaxPool$ is the maximum pooling layer. Then use the fully connected layer to perform time series regression prediction. Therefore, the output of the HBOT-CNN model $O_{tcnn}$ is as follows:

$$O_{tcnn} = \sigma(Linear(\hat{d_{cnn}}(Z))) \tag{21}$$

**ConvLSTM Decoder:** The HBOT-ConvLSTM model utilizes a ConvLSTM decoder with a hidden state $\hat{d}_{clstm}(Z)$ as shown in Eq. (22).

$$\hat{i}_t = \sigma(W_{\hat{i}} * x_t + W_{\hat{i}} * h_{t-1} + W_{\hat{i}} \circ C_{t-1} + b_{\hat{i}}), \tag{22}$$

$$\hat{o}_t = \sigma(W_{\hat{o}} * x_t + W_{\hat{o}} * h_{t-1} + W_{\hat{o}} \circ C_t + b_{\hat{o}}), \tag{23}$$

$$\hat{f}_t = \sigma(W_{\hat{f}} * x_t + W_{\hat{f}} * h_{t-1} + W_{\hat{f}} \circ C_{t-1} + b_{\hat{f}}), \tag{24}$$

$$\hat{C}_t = \hat{i}_t \circ tanh(W_{\hat{C}} * x_t + W_{\hat{C}} * h_{t-1} + b_{\hat{C}}) + \hat{f}_t \circ \hat{C}_{t-1}, \tag{25}$$

$$d_{clstm} = \hat{o}_t \circ tanh(\hat{C}_t), \tag{26}$$

where $*$ means convolution operation, $\circ$ is Hadamard product, $W_{\hat{i}}$, $W_{\hat{o}}$, $W_{\hat{f}}$, $W_{\hat{C}}$ represent the weights of Input gate, Output gate, Forget gate, and Temporary cell state of the ConvLSTM decoder. $b_{\hat{i}}$, $b_{hato}$, $b_{\hat{f}}$, $b_{\hat{C}}$ represents the corresponding bias, $\sigma$ and $tanh$ are the sigmoid and tanh activation functions, respectively. $d_{clstm}$ represents the output of the ConvLSTM decoder. Then, the ConvLSTM decoder output $d_{clstm}$ is also input to the fully connected layer. Therefore, the output of the HBOT-ConvLSTM model is $O_{tclstm}$, as shown in Eq. (27).

$$O_{tclstm} = \sigma(Linear(\hat{d}_{clstm}(Z))). \tag{27}$$

### 4.6. Model evaluation and statistical tests

In the current related work, most researchers use MSE, RMSE, MAE, $R^2$ to evaluate the model performances. However, the evaluation of predictive performance always depends heavily on the adopted evaluation measures, which may lead to a biased assessment of the overall performance of the model. Here, we try to address the aforementioned issue. First, we evaluate the results using 13 assessment metrics, including MSE, RMSE, MAE, $R^2$, MAD, MAPE, $R^2_{adjusted}$, RMSLE, EVS, ME, MPD, MGD and PL. Then, non-parametric statistical tests, Friedman, Iman–Davenport and Nemenyi post-hoc tests, are then used for a comprehensive analysis.

#### 4.6.1. Evaluation measures

We use 13 evaluation measures to assess the predictive performance of the model. They are as follows defined:

- mean square error (MSE):

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y^i - \hat{y}^i)^2. \tag{28}$$

- Root mean square error (RMSE):

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y^i - \hat{y}^i)^2}. \tag{29}$$

- Mean absolute error(MAE):

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y^i - \hat{y}^i|. \tag{30}$$

- R-Square ($R^2$):

$$R^2 = 1 - \frac{\sum(y^i - \hat{y}^i)^2}{\sum(y^i - \bar{y})^2}. \tag{31}$$

- Median Absolute Deviation (MAD):

$$MAD = \frac{1}{m}\sum_{i=1}^{m} median_m(y^i - \hat{y}^i). \tag{32}$$

- Mean absolute percentage error (MAPE):

$$MAPE = \frac{1}{m}\sum_{i=1}^{m}\frac{|y^i - \hat{y}^i|}{max(\tau, |y^i|)}. \tag{33}$$

- Adjusted R-Square ($R^2_{adjusted}$):

$$R^2_{adjusted} = 1 - \left[\frac{(1-R^2)(m-1)}{m-k-1}\right]. \tag{34}$$

- Root Mean Square Logarithmic Error (RMSLE):

$$RMSLE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(log_e(1+y^i) - log_e(1+\hat{y}^i))^2}. \tag{35}$$

- Explained Variance Score (EVS):

$$EVS = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}}. \tag{36}$$

- Max Error (ME):

$$ME = max(|y^i - \hat{y}^i|). \tag{37}$$

- Mean Poisson Deviance (MPD):

$$MPD = \frac{1}{m}\sum_{i=1}^{m} 2(y^i log(y^i/\hat{y}^i) + \hat{y}^i - y^i). \tag{38}$$

- Mean Gamma Deviance (MGD):

$$MGD = 2(log(\bar{y}^i/y^i) + y^i/\bar{y}^i - 1). \tag{39}$$

- Pinball loss (PL):

$$PL = \frac{1}{m}\sum_{i=1}^{m} \alpha\ max(y^i - \hat{y}^i, 0) + (1-\alpha)max(\hat{y}^i - y^i, 0), \tag{40}$$

where $m$ stands for the number of samples, $y^i$ is the real house price of the $i$th sample, $\hat{y}^i$ is the predicted house price of the $i$th sample, $\bar{y}$ is the average house price, $y$ is the true house price for all samples, $k$ is the number of sample features, $median(\cdot)$ represents the median value, $\tau$ is an arbitrarily small but strictly positive number to avoid undefined results when $y^i$ is zero, $max(\cdot)$ represents the maximum value, and $Var(\cdot)$ represents the variance.

#### 4.6.2. Statistical evaluation

To further analyze whether there are statistically significant differences in the performance of the different methods, we employed the non-parametric statistical test methods, including the Friedman test, Iman–Davenport and Nemenyi post-hoc test as recommended by Demšar (2006). The statistical evaluation process involves three stages: (1) ranking the performance of each algorithm according to different evaluation measures to obtain an average ranking for each algorithm; (2) using Friedman and Iman–Davenport to test whether the algorithms all perform equally; (3) utilizing the Nemenyi post-hoc test to further distinguish between the algorithms. The following is a detailed description of the three stages.

In the first stage, we rank the prediction performance of the 27 models using 13 evaluation measures, where $R_{jw}$ denotes the ranking of the $j$th model in the $w$th evaluation measures. In this case, model with the best performance is ranked 1, followed by the model ranked 2, and so on. In the case of a tie, it assigns an average rank. The mean value of an algorithm's ranking $R_j$ is calculated as:

$$R_j = \frac{1}{W}\sum_{w=1}^{W} R_{jw}. \tag{41}$$

In the second stage, we determine whether there is a significant difference in the performance of these models based on the Friedman and Iman–Davenport tests. The Friedman test is a non-parametric test based on the calculation of $R_j$. The Friedman test is calculated as follows:

$$T_{\chi^2} = \frac{12d}{d(d+1)}\sum_{j=1}^{k}\left(R_j - \frac{k+1}{2}\right)^2, \tag{42}$$

where $k$ is the number of models, $d$ is the number of evaluation measures. The null hypothesis $H_0$ of Friedman and Iman-Davenport tests that all model performances are non-significant. Under the assumption that the original hypothesis $H_0$ retains, the Friedman statistic $T_{\chi^2}$ is roughly distributed as a $\chi^2$ distribution with $k-1$ degrees of freedom. Iman and Davenport stated that the Friedman test formula is conservative. When $d$ becomes larger, $T_{\chi^2}$ will gradually be biased towards chi-square random variables with degrees of freedom $k-1$. Therefore, the modification Eq. (43) based on Friedman test Eq. (42) is proposed:

$$T_F = \frac{(d-1)*T}{d(k-1)-T}. \tag{43}$$

Under the assumption that the original hypothesis H0 retains, the Iman–Davenport statistic $T_F$ is roughly distributed as a $F$ distribution with $(d-1)$ degrees of freedom. Suppose the Friedman and Iman–Davenport test values are greater than the corresponding distribution values. In that case, there is a statistically significant difference in the performance of all algorithms, and we reject the original hypothesis.

In the third stage, the Nemenyi post-hoc tests method is used as the post-hoc test to further differentiate the algorithms. In this process, the performance of the two models will be significantly different when their average rank difference exceeds a certain critical distance. Nemenyi post-hoc test is defined as follows:

$$CD = q_\alpha\sqrt[2]{\frac{k(k+1)}{6d}} \tag{44}$$

where the critical value $q_\alpha$ is based on the studentized statistical range divided by $\sqrt{2}$. We use Eq. (44) to calculate the CD value of the model, and then compare it with the rank difference between the two models. Moreover, we present the results of the Nemenyi post-hoc test using critical diagrams.

**Table 3**
Friedman and Iman–Davenport tests results for all models in terms of the three datasets.

| Dataset | Method | Test value | Distribution value | Hypothesis |
|---|---|---|---|---|
| Hong Kong Island | Friedman test | 289.67 | 38.89 | Reject |
| | Iman–Davenport | 71.92 | 1.53 | Reject |
| Kowloon | Friedman test | 301.77 | 38.89 | Reject |
| | Iman–Davenport | 99.95 | 1.53 | Reject |
| New Territories | Friedman test | 320.40 | 38.89 | Reject |
| | Iman–Davenport | 218.49 | 1.53 | Reject |

## 5. Experimental results

In this section, we present the results of the experimental evaluation. First, we compare the predictive performance of the various models across the three regional datasets. Then, the statistical differences between the models are analyzed and shown in critical diagrams. Finally, we conclude the assessment.

Tables 4 to 9 show the performance evaluation results, rankings and average ranks of the 27 models considering 13 evaluation metrics in the three datasets, respectively. The following results can be found: (1) For the Kowloon dataset, Tables 4 and 5 show that the HBOS-CatBoost model outperforms and is more stable than HBOS-AdaBoost and CatBoost, which are ranked second and third on average. This is due to the HBOS-CatBoost model ranking first in 10 of 13 evaluation measures, with the MAD and ME metrics ranking seventh and sixth, respectively; (2) For the Hong Kong Island dataset, the HBOS-CatBoost model given the best performance, followed by the HBOS-AdaBoost and AdaBoost models (shown in Tables 6 and 7). HBOS-CatBoost model ranked first under three evaluation measures, and second under all metrics except under the ME measure, where it ranked 12th. (3) For the New Territories dataset, the results of Tables 8 and 9 show that the HBOS-XGBoost model ranks first in terms of overall performance, first in ten evaluation metrics, sixth in the evaluation metric $MAD$, and second and third in the remaining two metrics, MAE and PL, respectively.

Therefore, we can draw two conclusions from the above findings: (1) HBOS-CatBoost performs the best among all models, and ranks first on average in all thirteen evaluation metrics in the Kowloon and Hong Kong Island datasets. This indicates that the HBOS-CatBoost is more stable and performs better overall in the house price dataset. (2)The house price prediction models perform best in the New Territories dataset, followed by Kowloon. Comparing the three top-performing models in the dataset for RMSE, MAE, $R^2$, and MAPE, it was found that the HBOS-XGBoost model had a 52.8% lower RMSE in the New Territories and the HBOS-CatBoost model had a 27.5% lower RMSE in Kowloon compared to the HBOS-CatBoost model in Hong Kong Island.

**Table 4**
Comparison of thirteen evaluation metrics for all models on the Kowloon.

| Evaluation metrics | MSE | RMSE | MAE | $R^2$ | MAD | MAPE | $R^2_{adjusted}$ |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | 664 355.8966 (6) | 815.0803 (6) | 465.6041 (9) | 0.9467 (6) | 243.9131 (11) | 9.3532 (7) | 0.944 (6) |
| HBOS-CatBoost | **622 165.9228 (1)** | **788.775 (1)** | **441.418 (1)** | **0.9501 (1)** | 231.9397 (7) | **8.9177 (1)** | **0.9475 (1)** |
| HBOS-AdaBoost | 644 536.3925 (3) | 802.8302 (3) | 452.8176 (4) | 0.9483 (3) | 237.3209 (8) | 9.23 (2) | 0.9456 (3) |
| HBOB-XGBoost | 705 391.0791 (9) | 839.8756 (9) | 461.6278 (7) | 0.9434 (9) | 228.151 (3) | 9.4054 (8) | 0.9405 (9) |
| HBOB-LightGBM | 714 990.2805 (11) | 845.571 (11) | 470.5801 (10) | 0.9426 (11) | 238.4801 (10) | 9.9709 (13) | 0.9397 (11) |
| HBOB-BLS | 1 010 746.8917 (19) | 1005.3591 (19) | 568.9187 (17) | 0.9189 (19) | 324.6312 (16) | 12.5727 (19) | 0.9147 (19) |
| HBOT-based | 1 262 824.5389 (24) | 1123.7547 (24) | 681.6707 (24) | 0.8986 (24) | 473.7583 (26) | 14.6971 (24) | 0.8935 (24) |
| HBOT-CNN | 1 065 878.1023 (21) | 1032.4137 (21) | 656.909 (21) | 0.9144 (21) | 436.0205 (24) | 14.4076 (23) | 0.9101 (21) |
| HBOT-ConvLSTM | 1 089 054.3618 (23) | 1043.5777 (23) | 670.9359 (22) | 0.9126 (23) | 430.5891 (23) | 13.823 (22) | 0.9081 (23) |
| BPNN | 820 873.4193 (14) | 906.0207 (14) | 514.9126 (14) | 0.9341 (14) | 278.7603 (14) | 10.5785 (14) | 0.9308 (14) |
| CNN | 1 077 482.9004 (22) | 1038.0187 (22) | 673.8135 (23) | 0.9135 (22) | 403.4894 (22) | 13.7708 (21) | 0.9091 (22) |
| GRU | 1 017 326.6585 (20) | 1008.6261 (20) | 614.0086 (20) | 0.9183 (20) | 357.7565 (19) | 12.5354 (18) | 0.9142 (20) |
| LSTM | 892 048.3982 (17) | 944.4831 (17) | 594.6041 (19) | 0.9284 (17) | 358.0533 (20) | 12.3473 (17) | 0.9247 (17) |
| BLS | 973 320.1052 (18) | 986.5699 (18) | 576.5003 (18) | 0.9219 (18) | 346.1583 (18) | 12.9805 (20) | 0.9179 (18) |
| Seq2Seq-GRU | 850 317.3342 (16) | 922.1265 (16) | 557.4739 (15) | 0.9317 (16) | 321.2823 (15) | 11.6455 (15) | 0.9283 (16) |
| Seq2Seq-LSTM | 834 274.0825 (15) | 913.3861 (15) | 561.4802 (16) | 0.933 (15) | 345.7321 (17) | 11.9881 (16) | 0.9296 (15) |
| Deep Transformer | 1 282 791.4901 (26) | 1132.6039 (26) | 732.1369 (25) | 0.897 (26) | 478.4657 (27) | 16.5731 (26) | 0.8918 (26) |
| AdaBoost | 664 060.4512 (5) | 814.899 (5) | 447.8392 (2) | 0.9467 (5) | 245.5 (12) | 9.2956 (5) | 0.944 (5) |
| CatBoost | 642 852.5231 (2) | 801.7808 (2) | 451.8683 (3) | 0.9484 (2) | 227.4194 (2) | 9.2487 (4) | 0.9458 (2) |
| GBDT | 703 889.4351 (8) | 838.9812 (8) | 476.3537 (12) | 0.9435 (8) | 246.0677 (13) | 9.6849 (10) | 0.9406 (8) |
| ExtraTrees | 710 456.7243 (10) | 842.8859 (10) | 461.4137 (6) | 0.943 (10) | 231.6624 (6) | 9.6823 (9) | 0.9401 (10) |
| LightGBM | 653 376.0494 (4) | 808.3168 (4) | 453.4237 (5) | 0.9475 (4) | **223.6908 (1)** | 9.2301 (3) | 0.9449 (4) |
| XGBoost | 682 926.9481 (7) | 826.3939 (7) | 464.5034 (8) | 0.9452 (7) | 231.6521 (5) | 9.3387 (6) | 0.9424 (7) |
| Bagging | 736 642.5584 (12) | 858.2788 (12) | 473.9565 (11) | 0.9409 (12) | 231.2523 (4) | 9.8145 (11) | 0.9379 (12) |
| KNN | 1 272 650.6238 (25) | 1128.1182 (25) | 737.4607 (26) | 0.8978 (25) | 468.936 (25) | 16.1537 (25) | 0.8926 (25) |
| RF | 779 356.699 (13) | 882.8118 (13) | 477.0243 (13) | 0.9374 (13) | 237.4151 (9) | 9.9315 (12) | 0.9343 (13) |
| SVR | 2 673 869.2353 (27) | 1635.197 (27) | 791.557 (27) | 0.7853 (27) | 370.08 (21) | 19.0068 (27) | 0.7744 (27) |

**Table 5**
(Continued).

| Evaluation metrics | RMSLE | EVS | ME | MPD | MGD | PL | Average rank |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | 0.1486 (3) | 0.9467 (6) | 6140.258 (2) | 100.2146 (5) | 0.0198 (3) | 235.1115 (14) | 6.4615 |
| HBOS-CatBoost | **0.1444 (1)** | **0.9502 (1)** | 6567.7788 (6) | **94.5591 (1)** | **0.0186 (1)** | 203.7874 (3) | 2 |
| HBOS-AdaBoost | 0.1475 (2) | 0.9486 (2) | 6230.0 (3) | 98.3686 (2) | 0.0193 (2) | 197.5357 (2) | 3 |
| HBOB-XGBoost | 0.1525 (8) | 0.9434 (9) | 6677.781 (9) | 106.7179 (8) | 0.0206 (8) | 220.7146 (11) | 8.2308 |
| HBOB-LightGBM | 0.1579 (11) | 0.9428 (11) | 7305.0799 (14) | 109.2763 (10) | 0.0217 (11) | 212.6653 (6) | 10.7692 |
| HBOB-BLS | 0.2117 (23) | 0.9189 (19) | 9326.0454 (26) | 172.6283 (22) | 0.0524 (25) | 271.6955 (17) | 20 |
| HBOT-based | 0.2063 (22) | 0.8993 (26) | 9038.271 (25) | 189.722 (24) | 0.0378 (21) | 299.9119 (21) | 23.7692 |
| HBOT-CNN | 0.2016 (21) | 0.9144 (21) | 8497.026 (23) | 175.0189 (23) | 0.038 (22) | 332.0409 (25) | 22.0769 |
| HBOT-ConvLSTM | 0.1955 (19) | 0.9126 (23) | 8167.431 (20) | 170.6424 (19) | 0.0363 (19) | 348.7418 (26) | 21.9231 |
| BPNN | 0.1659 (14) | 0.9346 (14) | 7399.3504 (16) | 124.761 (14) | 0.0247 (14) | 293.4808 (20) | 14.6154 |
| CNN | 0.1956 (20) | 0.9143 (22) | **6130.376 (1)** | 171.0562 (20) | 0.0369 (20) | 382.1961 (27) | 20.3077 |
| GRU | 0.1833 (18) | 0.9185 (20) | 7792.5684 (18) | 153.7862 (18) | 0.0315 (18) | 324.8507 (23) | 19.3846 |
| LSTM | 0.179 (17) | 0.9288 (17) | 8058.8086 (19) | 141.0198 (17) | 0.0302 (17) | 328.5318 (24) | 18.0769 |
| BLS | 0.2184 (25) | 0.9219 (18) | 8882.3831 (24) | 171.5403 (21) | 0.058 (27) | 273.7654 (18) | 20.0769 |
| Seq2Seq-GRU | 0.1743 (15) | 0.9325 (16) | 6618.179 (7) | 132.308 (16) | 0.0279 (15) | 234.5936 (13) | 14.6923 |
| Seq2Seq-LSTM | 0.1751 (16) | 0.9332 (15) | 7184.143 (12) | 131.6294 (15) | 0.028 (16) | 261.7909 (15) | 15.2308 |
| Deep Transformer | 0.222 (26) | 0.9001 (24) | 8200.407 (21) | 206.2878 (26) | 0.0435 (24) | 277.4467 (19) | 24.7692 |
| AdaBoost | 0.1493 (6) | 0.947 (5) | 8343.7 (22) | 100.9864 (6) | 0.0199 (5) | **197.2986 (1)** | 6.4615 |
| CatBoost | 0.1491 (5) | 0.9484 (3) | 6305.3755 (4) | 99.5254 (3) | 0.0199 (6) | 217.218 (8) | 3.5385 |
| GBDT | 0.1541 (9) | 0.9435 (8) | 6530.052 (5) | 109.0001 (9) | 0.0217 (10) | 230.4103 (12) | 9.2308 |
| ExtraTrees | 0.1578 (10) | 0.9432 (10) | 7336.2431 (15) | 109.4342 (11) | 0.0214 (9) | 205.2474 (4) | 9.2308 |
| LightGBM | 0.1487 (4) | 0.9476 (4) | 6824.8394 (11) | 99.9681 (4) | 0.0198 (4) | 211.7768 (5) | 4.3846 |
| XGBoost | 0.1506 (7) | 0.9452 (7) | 6716.715 (10) | 104.6672 (7) | 0.0205 (7) | 218.4712 (10) | 7.3077 |
| Bagging | 0.1582 (12) | 0.941 (12) | 7218.6314 (13) | 112.5276 (12) | 0.0218 (12) | 217.4814 (9) | 11.0769 |
| KNN | 0.2163 (24) | 0.8995 (25) | 7408.6207 (17) | 201.4045 (25) | 0.0412 (23) | 304.5768 (22) | 24 |
| RF | 0.1618 (13) | 0.9376 (13) | 6639.5598 (8) | 119.1201 (13) | 0.0229 (13) | 216.5692 (7) | 11.7692 |
| SVR | 0.272 (27) | 0.7917 (27) | 10 407.6163 (27) | 350.7006 (27) | 0.0562 (26) | 269.0204 (16) | 25.6154 |

The MAE was approximately 54.07% and 30.02% lower, and the MAPE was 35.54% and 16.92% less for the former models. Additionally, the $R^2$ values were approximately 3.7% and 1.54% higher for the former models.

To determine whether there are any significant differences in the performance of the 27 models, we consider adopting non-parametric statistical tests. We employ the $\chi^2$ distribution with $26(=27-1)$ degrees of freedom and the $F$ distribution with $26\times12=312$ degrees of freedom at 0.05 confidence as the critical values for the Friedman and Iman–Davenport tests. Table 3 shows that the values of the Friedman and Iman–Davenport test for all three datasets are greater than the corresponding distribution values. Consequently, the original hypothesis is rejected, indicating the performance of the 27 models is different at the 95% confidence level. On the basis of this, Nemenyi post-hoc tests are conducted to compare 27 models. When the number of models is 27 and the critical value is 3.696, the corresponding critical difference (CD) $CD = 3.696413 * \sqrt[2]{\frac{27(27+1)}{6*13}} \approx 11.51$. The test results are shown in Figs. 6(a) to 6(c). The models that are not significantly different from each other are connected with CD lines. A critical diagram contains an enumeration axis and a calibration axis: the enumeration axis is used

**Table 6**
Comparison of thirteen evaluation metrics for all models on the Hong Kong Island.

| Evaluation metrics | MSE | RMSE | MAE | $R^2$ | MAD | MAPE | $R^2_{adjusted}$ |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | 1 192 991.9642 (3) | 1092.2417 (3) | 643.2743 (5) | 0.9349 (3) | 373.1357 (11) | 10.9835 (5) | 0.9316 (3) |
| HBOS-CatBoost | 1 182 593.483 (2) | 1087.4711 (2) | **630.7392 (1)** | 0.9355 (2) | 350.5499 (2) | **10.7338 (1)** | 0.9322 (2) |
| HBOS-AdaBoost | **1 178 195.8781 (1)** | **1085.4473 (1)** | 638.0088 (3) | **0.9357 (1)** | 350.75 (3) | 10.7431 (2) | **0.9324 (1)** |
| HBOB-XGBoost | 1 237 617.1213 (9) | 1112.4824 (9) | 651.4376 (9) | 0.9325 (9) | 367.556 (8) | 11.1402 (7) | 0.929 (9) |
| HBOB-LightGBM | 1 287 014.8618 (12) | 1134.4668 (12) | 672.691 (13) | 0.9298 (12) | 371.7519 (9) | 11.7794 (12) | 0.9262 (12) |
| HBOB-BLS | 1 801 182.8798 (21) | 1342.0815 (21) | 793.9092 (19) | 0.9017 (21) | 473.1333 (18) | 14.1833 (20) | 0.8967 (21) |
| HBOT-based | 2 176 948.4981 (25) | 1475.4486 (25) | 924.9498 (24) | 0.8812 (25) | 596.1923 (25) | 16.1427 (24) | 0.8752 (25) |
| HBOT-CNN | 1 749 707.5635 (20) | 1322.7651 (20) | 790.932 (18) | 0.9045 (20) | 496.5154 (19) | 13.7946 (19) | 0.8997 (20) |
| HBOT-ConvLSTM | 1 921 457.9291 (22) | 1386.1666 (22) | 891.6633 (22) | 0.8952 (22) | 581.8457 (23) | 15.2422 (22) | 0.8898 (22) |
| BPNN | 1 392 889.1932 (14) | 1180.2073 (14) | 723.9723 (14) | 0.924 (14) | 446.9069 (14) | 13.0698 (15) | 0.9201 (14) |
| CNN | 1 924 279.0964 (23) | 1387.1839 (23) | 894.1959 (23) | 0.895 (23) | 588.4342 (24) | 15.2702 (23) | 0.8897 (23) |
| GRU | 1 733 710.8503 (18) | 1316.7045 (18) | 827.0028 (21) | 0.9054 (18) | 504.4004 (20) | 13.2121 (17) | 0.9006 (18) |
| LSTM | 1 713 760.4734 (17) | 1309.1067 (17) | 816.5843 (20) | 0.9065 (17) | 506.9096 (21) | 13.5145 (18) | 0.9017 (17) |
| BLS | 1 748 432.1607 (19) | 1322.2829 (19) | 788.2492 (17) | 0.9046 (19) | 472.0937 (17) | 14.3011 (21) | 0.8997 (19) |
| Seq2Seq-GRU | 1 533 003.7767 (15) | 1238.1453 (15) | 746.844 (15) | 0.9163 (15) | 447.3463 (15) | 13.0304 (14) | 0.9121 (15) |
| Seq2Seq-LSTM | 1 588 763.0419 (16) | 1260.4614 (16) | 772.3913 (16) | 0.9133 (16) | 457.8387 (16) | 13.0973 (16) | 0.9089 (16) |
| Deep Transformer | 2 106 971.9904 (24) | 1451.5412 (24) | 976.7903 (25) | 0.885 (24) | 661.421 (26) | 16.9839 (25) | 0.8792 (24) |
| AdaBoost | 1 202 220.223 (4) | 1096.458 (4) | 638.2942 (4) | 0.9344 (4) | 366.2672 (7) | 10.8221 (4) | 0.9311 (4) |
| CatBoost | 1 211 945.2767 (6) | 1100.8839 (6) | 647.4757 (6) | 0.9339 (6) | 372.4007 (10) | 11.0802 (6) | 0.9305 (6) |
| GBDT | 1 268 261.6428 (11) | 1126.1712 (11) | 671.3188 (12) | 0.9308 (11) | 398.532 (13) | 11.8362 (13) | 0.9273 (11) |
| ExtraTrees | 1 217 234.2691 (8) | 1103.2834 (8) | 636.2851 (2) | 0.9336 (8) | **349.5958 (1)** | 10.7738 (3) | 0.9302 (8) |
| LightGBM | 1 213 672.9351 (7) | 1101.6683 (7) | 651.2353 (8) | 0.9338 (7) | 356.8874 (5) | 11.1439 (8) | 0.9304 (7) |
| XGBoost | 1 202 955.3716 (5) | 1096.7932 (5) | 655.8585 (11) | 0.9344 (5) | 375.512 (12) | 11.2584 (11) | 0.931 (5) |
| Bagging | 1 247 826.1821 (10) | 1117.0614 (10) | 655.3869 (10) | 0.9319 (10) | 361.9319 (6) | 11.243 (10) | 0.9285 (10) |
| KNN | 2 703 699.7128 (26) | 1644.2931 (26) | 1112.8515 (27) | 0.8525 (26) | 723.7524 (27) | 20.1821 (26) | 0.845 (26) |
| RF | 1 292 930.4107 (13) | 1137.071 (13) | 648.6462 (7) | 0.9294 (13) | 355.7462 (4) | 11.1835 (9) | 0.9259 (13) |
| SVR | 3 637 185.9496 (27) | 1907.1408 (27) | 1039.1191 (26) | 0.8015 (27) | 578.2003 (22) | 20.6795 (27) | 0.7914 (27) |

**Table 7**
(Continued).

| Evaluation metrics | RMSLE | EVS | ME | MPD | MGD | PL | Average rank |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | 0.1695 (5) | 0.9349 (3) | 10 148.47 (13) | 146.5004 (3) | 0.0244 (5) | 326.404 (4) | 5.0769 |
| HBOS-CatBoost | 0.1677 (2) | 0.9355 (2) | 10 146.4547 (12) | 144.7431 (2) | **0.024 (1)** | 320.1256 (2) | 2.5385 |
| HBOS-AdaBoost | **0.1663 (1)** | **0.9357 (1)** | 10 494.6667 (19) | **144.4854 (1)** | 0.0241 (2) | **312.7603 (1)** | 2.8462 |
| HBOB-XGBoost | 0.172 (9) | 0.9325 (9) | 9998.7705 (8) | 151.0048 (9) | 0.025 (8) | 335.8706 (10) | 8.6923 |
| HBOB-LightGBM | 0.1774 (12) | 0.9298 (12) | 10 206.4262 (14) | 157.6604 (11) | 0.0266 (12) | 347.8224 (13) | 12 |
| HBOB-BLS | 0.2811 (26) | 0.9017 (21) | 12 155.2747 (24) | 261.4043 (23) | 5.7396 (27) | 403.8173 (17) | 21.4615 |
| HBOT-based | 0.2218 (22) | 0.8816 (25) | 11 743.696 (23) | 263.1732 (24) | 0.0428 (22) | 426.7626 (21) | 23.8462 |
| HBOT-CNN | 0.2021 (19) | 0.9048 (19) | 10 552.402 (20) | 216.838 (19) | 0.0368 (19) | 424.8051 (20) | 19.3846 |
| HBOT-ConvLSTM | 0.2129 (21) | 0.8954 (22) | 10 479.156 (18) | 239.848 (21) | 0.0416 (21) | 415.0927 (19) | 21.3077 |
| BPNN | 0.1921 (14) | 0.9246 (14) | 11 255.2871 (21) | 176.9467 (14) | 0.0313 (14) | 408.0268 (18) | 14.9231 |
| CNN | 0.2111 (20) | 0.8951 (23) | **9610.3398 (1)** | 238.4997 (20) | 0.0404 (20) | 470.265 (25) | 20.8462 |
| GRU | 0.1924 (15) | 0.9055 (18) | 12 955.6123 (27) | 213.1628 (18) | 0.036 (18) | 432.4532 (23) | 19.1538 |
| LSTM | 0.1958 (18) | 0.9072 (17) | 10 292.4609 (16) | 210.6144 (17) | 0.035 (17) | 458.2766 (24) | 18.1538 |
| BLS | 0.281 (25) | 0.9046 (20) | 12 243.2215 (25) | 258.0003 (22) | 5.7393 (26) | 401.3755 (16) | 20.3846 |
| Seq2Seq-GRU | 0.1929 (16) | 0.9164 (15) | 11 533.89 (22) | 192.1425 (15) | 0.0332 (15) | 365.4446 (14) | 15.4615 |
| Seq2Seq-LSTM | 0.193 (17) | 0.9133 (16) | 9966.887 (6) | 198.3711 (16) | 0.034 (16) | 387.9918 (15) | 15.2308 |
| Deep Transformer | 0.2251 (23) | 0.8853 (24) | 9754.541 (3) | 263.559 (25) | 0.0446 (23) | 521.1858 (26) | 22.7692 |
| AdaBoost | 0.1689 (4) | 0.9344 (4) | 10 024.5 (10) | 147.0359 (4) | 0.0244 (4) | 321.8986 (3) | 4.6154 |
| CatBoost | 0.1708 (7) | 0.9339 (6) | 9987.378 (7) | 148.5872 (6) | 0.0248 (7) | 330.4649 (8) | 6.6923 |
| GBDT | 0.1788 (13) | 0.9308 (11) | 10 283.9965 (15) | 158.6454 (13) | 0.027 (13) | 338.0292 (12) | 12.2308 |
| ExtraTrees | 0.1689 (3) | 0.9336 (8) | 10 021.3885 (9) | 147.3496 (5) | 0.0241 (3) | 327.6695 (5) | 5.4615 |
| LightGBM | 0.1707 (6) | 0.9338 (7) | 10 057.0623 (11) | 148.7912 (7) | 0.0248 (6) | 329.9184 (6) | 7.0769 |
| XGBoost | 0.1713 (8) | 0.9344 (5) | 9663.82 (2) | 149.0525 (8) | 0.0251 (9) | 337.1689 (11) | 7.4615 |
| Bagging | 0.174 (10) | 0.9319 (10) | 9837.954 (4) | 153.6731 (10) | 0.0255 (10) | 335.7805 (9) | 9.1538 |
| KNN | 0.2574 (24) | 0.8526 (26) | 9845.0961 (5) | 341.3468 (26) | 0.0572 (24) | 537.8047 (27) | 24.3077 |
| RF | 0.1755 (11) | 0.9295 (13) | 10 457.4111 (17) | 158.5376 (12) | 0.0261 (11) | 329.928 (7) | 11 |
| SVR | 0.29 (27) | 0.8038 (27) | 12 885.8253 (26) | 415.6979 (27) | 0.0753 (25) | 428.5654 (22) | 25.9231 |

to show the average rank value line of the model, while the calibration axis is employed to show the magnitude of the CD values. The method names are shown along the enumeration axis with the highest ranked ones on the top right of the diagram.

Fig. 6(a) shows the following results: (1) in terms of the Kowloon dataset, there is no statistically significant difference in performance among HBOS-CatBoost, HBOS-AdaBoost, CatBoost, LightGBM, AdaBoost, HBOS-XGBoost, XGBoost, HBOB-XGBoost, ExtraTrees, GBDT, HBOB-LightGBM, Bagging and RF, while HBOS-CatBoost performs the best among these models; (2) there is no statistically significant difference in performance among XGBoost, HBOB-XGBoost, ExtraTrees, GBDT, HBOB-LightGBM, Bagging, RF, BPNN, Seq2Seq-GRU, Seq2Seq-LSTM and LSTM; (3) there is no statistically significant difference in performance among Bagging, RF, BPNN, Seq2Seq-GRU, Seq2Seq-LSTM, LSTM, GRU, HBOB-BLS, CNN, HBOT-ConvLSTM and HBOT-CNN; (4) there is no statistically significant difference in performance among LSTM, GRU, HBOB-BLS, CNN, HBOT-ConvLSTM, HBOT-CNN, HBOT-based, KNN, Deep Transformer and SVR.

Fig. 6(b) shows the following results: (1) in terms of the Hong Kong Island dataset, there is no statistically significant difference in performance among HBOS-CatBoost, HBOS-AdaBoost, AdaBoost, HBOS-XGBoost, ExtraTrees, CatBoost, LightGBM, XGBoost, HBOB-XGBoost, Bagging, RF, HBOB-LightGBM and GBDT, while HBOS-CatBoost performs the best among these models; (2) there is no statistically significant difference in performance among LightGBM, XGBoost, HBOB-XGBoost, Bagging, RF, HBOB-LightGBM, GBDT, BPNN, Seq2Seq-LSTM,

**Table 8**
Comparison of thirteen evaluation metrics for all models on the New Territories.

| Evaluation metrics | MSE | RMSE | MAE | $R^2$ | MAD | MAPE | $R^2_{adjusted}$ |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | **263 516.7659 (1)** | **513.3388 (1)** | 289.675 (2) | **0.9713 (1)** | 146.317 (6) | **6.9189 (1)** | **0.9698 (1)** |
| HBOS-CatBoost | 269 143.4829 (2) | 518.7904 (2) | **288.9521 (1)** | 0.9707 (2) | 143.3331 (3) | 6.9779 (3) | 0.9692 (2) |
| HBOS-AdaBoost | 273 036.1396 (4) | 522.5286 (4) | 295.1185 (4) | 0.9703 (4) | 154.5671 (10) | 6.979 (4) | 0.9688 (4) |
| HBOB-XGBoost | 310 953.0719 (9) | 557.6317 (9) | 304.9788 (8) | 0.9661 (9) | 145.1456 (4) | 7.4312 (8) | 0.9644 (9) |
| HBOB-LightGBM | 326 896.5802 (12) | 571.7487 (12) | 319.3384 (12) | 0.9644 (12) | 162.0272 (12) | 7.9543 (13) | 0.9626 (12) |
| HBOB-BLS | 444 066.9221 (19) | 666.3835 (19) | 378.6158 (17) | 0.9516 (19) | 217.7526 (17) | 10.0546 (18) | 0.9492 (19) |
| HBOT-based | 554 540.15 (23) | 744.6745 (23) | 507.5688 (24) | 0.9396 (23) | 349.1803 (24) | 12.2767 (23) | 0.9365 (23) |
| HBOT-CNN | 444 991.7288 (20) | 667.077 (20) | 422.1349 (20) | 0.9515 (20) | 277.7305 (22) | 11.1547 (21) | 0.9491 (20) |
| HBOT-ConvLSTM | 560 764.2249 (24) | 748.8419 (24) | 495.9008 (23) | 0.9389 (24) | 349.8744 (25) | 13.2016 (24) | 0.9358 (24) |
| BPNN | 359 971.8898 (14) | 599.9766 (14) | 328.0867 (14) | 0.9608 (14) | 172.4064 (14) | 8.1783 (14) | 0.9588 (14) |
| CNN | 496 194.4311 (22) | 704.4107 (22) | 430.9928 (21) | 0.946 (22) | 253.7474 (20) | 10.7511 (20) | 0.9432 (22) |
| GRU | 420 358.892 (17) | 648.3509 (17) | 391.7857 (19) | 0.9542 (17) | 237.4339 (19) | 10.421 (19) | 0.9519 (17) |
| LSTM | 446 803.1483 (21) | 668.4334 (21) | 432.5658 (22) | 0.9513 (21) | 278.1282 (23) | 11.6081 (22) | 0.9489 (21) |
| BLS | 440 091.381 (18) | 663.3938 (18) | 371.2067 (16) | 0.9521 (18) | 213.9235 (16) | 9.9417 (17) | 0.9496 (18) |
| Seq2Seq-GRU | 390 278.989 (15) | 624.7231 (15) | 357.2047 (15) | 0.9575 (15) | 197.4229 (15) | 8.6219 (15) | 0.9553 (15) |
| Seq2Seq-LSTM | 418 841.3616 (16) | 647.1795 (16) | 381.3772 (18) | 0.9544 (16) | 224.3991 (18) | 9.3998 (16) | 0.9521 (16) |
| Deep Transformer | 607 484.1521 (25) | 779.4127 (25) | 520.8634 (25) | 0.9338 (25) | 353.647 (26) | 13.21 (25) | 0.9305 (25) |
| AdaBoost | 270 876.3538 (3) | 520.4578 (3) | 291.6485 (3) | 0.9705 (3) | 142.375 (2) | 6.9468 (2) | 0.969 (3) |
| CatBoost | 287 312.2919 (5) | 536.0152 (5) | 299.7946 (5) | 0.9687 (5) | 152.4481 (8) | 7.3046 (5) | 0.9671 (5) |
| GBDT | 320 138.246 (11) | 565.8076 (11) | 321.0632 (13) | 0.9651 (11) | 166.2095 (13) | 7.9399 (12) | 0.9634 (11) |
| ExtraTrees | 303 671.0272 (7) | 551.0635 (7) | 302.0548 (6) | 0.9669 (7) | **139.9133 (1)** | 7.3197 (6) | 0.9652 (7) |
| LightGBM | 309 233.369 (8) | 556.0876 (8) | 312.5401 (10) | 0.9663 (8) | 156.7648 (11) | 7.5833 (9) | 0.9646 (8) |
| XGBoost | 317 257.7157 (10) | 563.2563 (10) | 313.8734 (11) | 0.9654 (10) | 154.4315 (9) | 7.6493 (11) | 0.9637 (10) |
| Bagging | 291 973.9411 (6) | 540.3461 (6) | 304.4467 (7) | 0.9682 (6) | 149.1822 (7) | 7.3862 (7) | 0.9666 (6) |
| KNN | 751 233.5703 (26) | 866.7373 (26) | 607.7798 (27) | 0.9182 (26) | 423.9663 (27) | 15.3035 (26) | 0.914 (26) |
| RF | 329 843.3686 (13) | 574.3199 (13) | 310.2968 (9) | 0.9641 (13) | 145.9295 (5) | 7.6123 (10) | 0.9623 (13) |
| SVR | 1 573 945.2375 (27) | 1254.5697 (27) | 581.8636 (26) | 0.8286 (27) | 264.0802 (21) | 17.9854 (27) | 0.8199 (27) |

**Table 9**
(Continued).

| Evaluation metrics | RMSLE | EVS | ME | MPD | MGD | PL | Average rank |
|---|---|---|---|---|---|---|---|
| HBOS-XGBoost | **0.1226 (1)** | **0.9713 (1)** | **4035.7971 (1)** | **53.3834 (1)** | **0.0144 (1)** | 146.6859 (3) | 1.6154 |
| HBOS-CatBoost | 0.1247 (4) | 0.9707 (2) | 4265.089 (2) | 54.6627 (2) | 0.0148 (2) | **144.8107 (1)** | 2.1538 |
| HBOS-AdaBoost | 0.1233 (2) | 0.9703 (4) | 4270.8504 (3) | 55.301 (4) | 0.0151 (4) | 147.2865 (4) | 4.2308 |
| HBOB-XGBoost | 0.1338 (9) | 0.9661 (9) | 4599.6838 (11) | 63.2224 (9) | 0.0169 (9) | 155.3328 (8) | 8.5385 |
| HBOB-LightGBM | 0.1373 (12) | 0.9644 (12) | 4731.2468 (12) | 65.8542 (11) | 0.0175 (11) | 159.7041 (12) | 11.9231 |
| HBOB-BLS | 0.1754 (22) | 0.9516 (19) | 5435.4281 (15) | 98.9701 (21) | 0.0314 (23) | 188.1864 (18) | 18.9231 |
| HBOT-based | 0.1794 (23) | 0.9397 (24) | 6419.7693 (24) | 110.3895 (23) | 0.0329 (24) | 241.3695 (25) | 23.5385 |
| HBOT-CNN | 0.1658 (18) | 0.9515 (20) | 5450.7513 (16) | 91.7589 (18) | 0.0259 (18) | 209.0975 (21) | 19.5385 |
| HBOT-ConvLSTM | 0.1821 (24) | 0.9414 (23) | 6382.9573 (23) | 112.1984 (24) | 0.0307 (22) | 179.5616 (16) | 23.0769 |
| BPNN | 0.1405 (13) | 0.9609 (14) | 5523.7959 (19) | 70.192 (14) | 0.0182 (13) | 174.564 (14) | 14.2308 |
| CNN | 0.1673 (19) | 0.9461 (22) | 5477.8695 (17) | 99.2718 (22) | 0.0272 (20) | 228.7022 (24) | 21 |
| GRU | 0.1614 (17) | 0.9542 (17) | 5803.3109 (21) | 85.5201 (17) | 0.0242 (17) | 202.2506 (20) | 18 |
| LSTM | 0.1703 (20) | 0.9514 (21) | 5294.0882 (14) | 93.2029 (19) | 0.0271 (19) | 225.2135 (23) | 20.5385 |
| BLS | 0.1715 (21) | 0.9521 (18) | 5740.8644 (20) | 95.1382 (20) | 0.0286 (21) | 185.0758 (17) | 18.3077 |
| Seq2Seq-GRU | 0.1421 (15) | 0.9575 (15) | 6655.4153 (25) | 75.0171 (15) | 0.0198 (15) | 177.6916 (15) | 15.7692 |
| Seq2Seq-LSTM | 0.1508 (16) | 0.9544 (16) | 6658.574 (26) | 82.5279 (16) | 0.0221 (16) | 188.4409 (19) | 17.3077 |
| Deep Transformer | 0.1912 (25) | 0.9341 (25) | 5167.718 (13) | 125.8979 (25) | 0.0358 (25) | 283.1143 (26) | 24.2308 |
| AdaBoost | 0.1235 (3) | 0.9705 (3) | 4434.2222 (6) | 55.1836 (3) | 0.0149 (3) | 146.0455 (2) | 3 |
| CatBoost | 0.1301 (6) | 0.9687 (5) | 4396.1521 (5) | 58.8512 (5) | 0.016 (6) | 151.2969 (5) | 5.3846 |
| GBDT | 0.141 (14) | 0.9651 (11) | 4517.9527 (10) | 67.0202 (13) | 0.0194 (14) | 160.6986 (13) | 12.0769 |
| ExtraTrees | 0.1318 (7) | 0.9669 (7) | 4277.3959 (4) | 61.5885 (7) | 0.0163 (7) | 153.4025 (6) | 6.0769 |
| LightGBM | 0.1338 (8) | 0.9663 (8) | 4470.9458 (8) | 62.9706 (8) | 0.0169 (8) | 155.7131 (10) | 8.6154 |
| XGBoost | 0.136 (11) | 0.9654 (10) | 4441.9082 (7) | 65.4187 (10) | 0.0177 (12) | 156.445 (11) | 10.1538 |
| Bagging | 0.1297 (5) | 0.9682 (6) | 4475.9982 (9) | 59.2424 (6) | 0.0158 (5) | 155.6427 (9) | 6.5385 |
| KNN | 0.2005 (26) | 0.9182 (26) | 6103.1184 (22) | 144.1673 (26) | 0.0375 (26) | 303.8765 (27) | 25.9231 |
| RF | 0.136 (10) | 0.9641 (13) | 5486.898 (18) | 66.1708 (12) | 0.0174 (10) | 154.4506 (7) | 11.2308 |
| SVR | 0.2699 (27) | 0.8318 (27) | 8433.6495 (27) | 262.8921 (27) | 0.0526 (27) | 213.0385 (22) | 26.0769 |

Seq2Seq-GRU and LSTM; (3) there is no statistically significant difference in performance among HBOB-LightGBM, GBDT, BPNN, Seq2Seq-LSTM, Seq2Seq-GRU, LSTM, GRU, HBOT-CNN, BLS, CNN, HBOT-ConvLSTM, HBOB-BLS and Deep Transformer; (4) there is no statistically significant difference in performance among LSTM, GRU, HBOT-CNN, BLS, CNN, HBOT-ConvLSTM, HBOB-BLS, Deep Transformer, HBOT-based, KNN and SVR.

Fig. 6(c) shows the following results: (1) in terms of the New Territories dataset, there is no statistically significant difference in performance among HBOS-XGBoost, HBOS-CatBoost, AdaBoost, HBOS-AdaBoost, CatBoost, ExtraTrees, Bagging, HBOB-XGBoost, LightGBM, XGBoost, RF, HBOB-LightGBM and GBDT, while HBOS-XGBoost performs the best among these models; (2) there is no statistically significant difference in performance among Bagging, HBOB-XGBoost, LightGBM, XGBoost, RF, HBOB-LightGBM, GBDT, BPNN, Seq2Seq-GRU and Seq2Seq-LSTM; (3) there is no statistically significant difference in performance among GBDT, BPNN, Seq2Seq-GRU, Seq2Seq-LSTM, GRU, BLS, HBOB-BLS, HBOT-CNN, LSTM, CNN, HBOT-ConvLSTM and HBOT-based; (4) there is no statistically significant difference in performance among Seq2Seq-LSTM, GRU, BLS, HBOB-BLS, HBOT-CNN, LSTM, CNN, HBOT-ConvLSTM, HBOT-based, Deep Transformer, KNN and SVR.

According to the results in the three datasets, we can come to conclusions: (1) the HBOS model has the best performance, followed by HBOB and HBOT; (2) the HBOS models typically outperform the standard CatBoost, AdaBoost, and XGBoost models significantly, while HBOB models rarely outperform the basic BLS, XGB, and LightGBM models; (3) In the HBOT model, the CNN decoder outperforms the ConvLSTM decoder. We have two hypotheses about this: first, for multi-model ensemble strategies for predicting home prices, stacking is a better option than bagging among the model ensemble techniques. Therefore, the HBOB models are not competitive when compared to the HBOS models. Second, the prediction performance is not noticeably improved by the HBOT-ConvLSTM model, which combines CNN, LSTM, and HBOT-based models. Additionally, it demonstrates that the CNN decoder outperforms the ConvLSTM decoder in terms of performance for the HBOT models.

## 6. Conclusion

There is evidence that the issue of predicting housing prices plays a crucial role in the field of economics. Hence, a large number of studies on the application of machine learning to construct house price prediction models, but the current study still has certain limitations. First, different machine learning models have different strengths and weaknesses, and the fact that a model performs well on one metric or in one dataset does not imply that it performs well in all aspects. Second, hyperparameters in machine learning models are one of the important factors affecting the results, and traditional grid search and random search methods have been difficult to optimize the models. Finally, there is a lack of a comprehensive and objective comparative framework for evaluating machine learning models in related house price forecasting studies.

To address the identified issues in prevailing methodologies for house price prediction, specifically, short time span, limited vector length, and high similarity among price trends across regions, we presented a comprehensive and innovative framework for house price forecasting. First of all, we apply stacking, bagging and transformer with the BO method to develop three types of hybrid models (HBOS, HBOS, HBOT). These three types of hybrid models not only incorporate the advantages of different models, but also allow the model hyperparameters to be tuned with the BO algorithm to obtain the optimal hyperparameters set. Second, we select 13 evaluation measures, including MSE, RMSE, MAE, $R^2$, MAD, MAPE, $R^2_{adjusted}$, RMSLE, EVS, ME, MPD, MGD and PL. Ultimately, we employ Friedman, Iman–Davenport, and Nemenyi post-hoc tests to analyze the experimental results for evaluating the statistical significance of the difference in model performance. While challenges persist due to the high similarity of price trends across regions, the relatively short time span, and the limited length of price vectors in the Hong Kong housing market, our research provides meaningful contributions to the accuracy of house price predictions. Such insights are particularly applicable to scholars and practitioners interested in improving the accuracy of house price predictions. Hence, our study would be a step forward in the field of property market research, by addressing extant limitations and establishing a methodological groundwork for future investigations in house price prediction.

The experimental comparison results reveal that compared with the existing methods, the HBOS models have the best predictive performance. Moreover, the HBOS models perform better than the HBOB models. The HBOT model is better than the Deep Transformer model, but not as effective as the classical deep learning model. We believe that the main reason why HBOT is not as effective as classical deep learning models may be due to insufficient data, as deep learning models typically require a large amount of data to perform well. To improve performance, we can increase the amount of data and consider adding new modules to the model, such as more complex encoder or decoder structures, to enhance the model's learning and expressive capabilities. According to the experimental results, it can be found that compared with the popular Transformer model, the HBOS models are more suitable for forecasting house prices. Our HBOT model is optimized for the house price prediction task and may not perform as well on other datasets. However, we will explore applying the model to

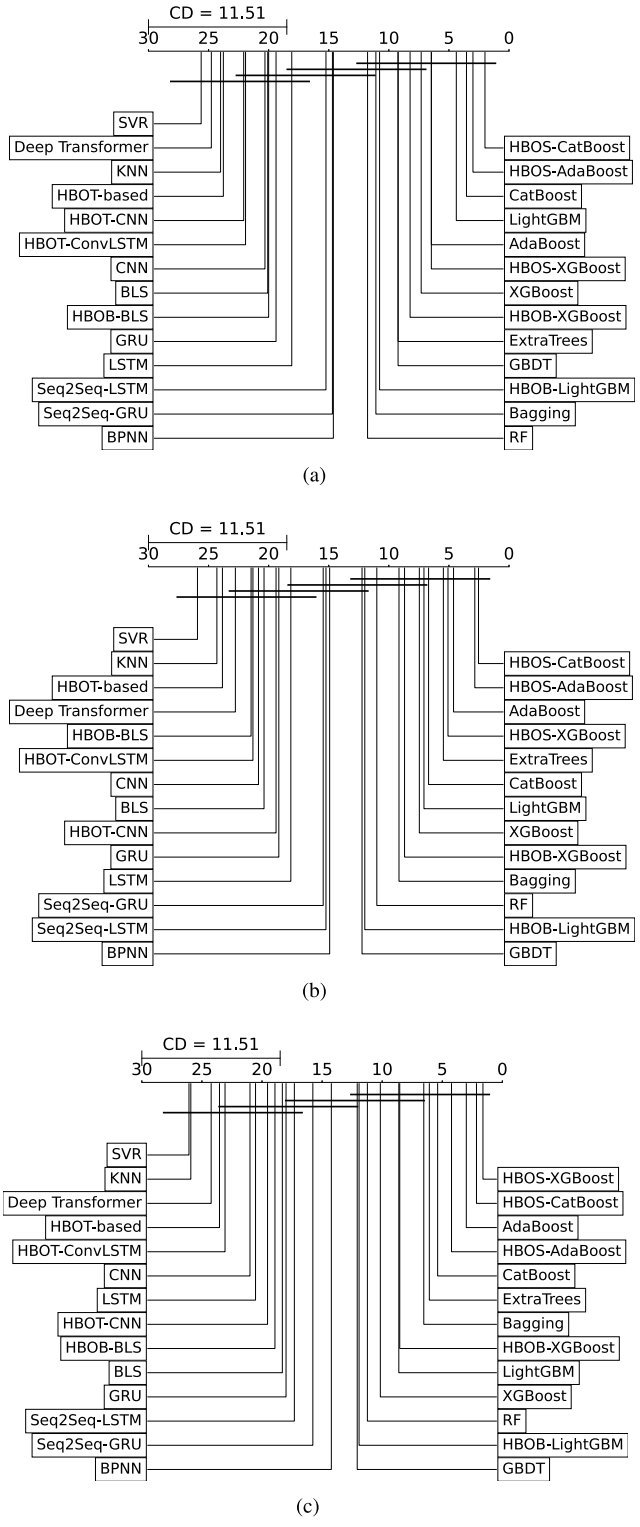



**Fig. 6.** The Nemenyi post-hoc test results for 27 methods in terms of the three datasets: (a) Kowloon; (b) Hong Kong Island; (c) New Territories.

other real estate prediction tasks and datasets in the future to evaluate its performance on different tasks and datasets.

This research establishes the framework for future studies on housing price forecasting. In terms of strengths, our algorithm integrates the strengths of several models to improve prediction accuracy and employs an effective technique to select appropriate model hyperparameters. However, a limitation of our study is the significant computational resources required, which may limit its practical applicability in specific scenarios. To address this limitation, we plan to explore more efficient model architectures, optimization techniques and increase the amount of data in future work. Further work in our study will be threefold. First, we can further integrate a multi-source data fusion framework. Then, a wider range of datasets needs to be used to consider the performance of the model from different perspectives. Lastly, we would consider seeking explanations for what factors would affect the value of the real estate and making recommendations for housing policies.

## CRediT authorship contribution statement

**Choujun Zhan:** Conceptualization, Methodology, Writing – original draft, Funding acquisition. **Yonglin Liu:** Software, Visualization. **Zeqiong Wu:** Data curation, Resources. **Mingbo Zhao:** Validation, Investigation, Funding acquisition, Writing – review & editing. **Tommy W.S. Chow:** Formal analysis, Visualization.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Mingbo Zhao reports was provided by City University of Hong Kong. Mingbo Zhao reports a relationship with City University of Hong Kong that includes: employment. We certify that there is no actual or potential conflict of interest in relation to this article.

## Data availability

Data will be made available on request

## Acknowledgments

This work was supported by, the Natural Science Foundation of Guangdong Province, China (2023A1515011618), Key Scientific Research Platform and Project of Guangdong Provincial Education Department (2022ZDZX1040, 2022ZDZX1039) and Educational Science Planning Topic of Guangdong province (2022GXJK382). This work was also partially supported by the National Natural Science Foundation of China (61971121, 62073272).

## References

Abidoye, R. B., & Chan, A. P. (2018). Improving property valuation accuracy: A comparison of hedonic pricing model and artificial neural network. *Pacific Rim Property Research Journal*, *24*, 71–83.

Abidoye, R. B., Chan, A. P., Abidoye, F. A., & Oshodi, O. S. (2019). Predicting property price index using artificial intelligence techniques: Evidence from hong kong. *International Journal of Housing Markets and Analysis*.

Adetunji, A. B., Akande, O. N., Ajala, F. A., Oyewo, O., Akande, Y. F., & Oluwadara, G. (2022). House price prediction using random forest machine learning technique. *Procedia Computer Science*, *199*, 806–813.

Benetton, M., Bracke, P., Cocco, J. F., & Garbarino, N. (2022). Housing consumption and investment: evidence from shared equity mortgages. *The Review of Financial Studies*, *35*, 3525–3573.

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., et al. (2021). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. In *Wiley interdisciplinary reviews: Data mining and knowledge discovery*. Article e1484.

Chen, J.-H., Ong, C. F., Zheng, L., & Hsu, S.-C. (2017). Forecasting spatial dynamics of the housing market using support vector machine. *International Journal of Strategic Property Management*, *21*, 273–283.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Fischer, M. M., Huber, F., Pfarrhofer, M., & Staufer-Steinnocher, P. (2021). The dynamic impact of monetary policy on regional housing prices in the united states. *Real Estate Economics*, *49*, 1039–1068.

Ge, C. (2019). A lstm and graph cnn combined network for community house price forecasting. In *2019 20th IEEE international conference on mobile data management* (pp. 393–394). IEEE.

Goodman, A. C. (1978). Hedonic prices, price indices and housing markets. *Journal of urban economics*, *5*, 471–484.

Harding, J. P., Knight, J. R., & Sirmans, C. (2003). Estimating bargaining effects in hedonic models: Evidence from the housing market. *Real estate economics*, *31*, 601–622.

Hodson, D., & Quaglia, L. (2009). European perspectives on the global financial crisis: Introduction. *JCMS: Journal of Common Market Studies*, *47*, 939–953.

Iacoviello, M., & Minetti, R. (2008). The credit channel of monetary policy: Evidence from the housing market. *Journal of Macroeconomics*, *30*, 69–96.

Kang, Y., Zhang, F., Peng, W., Gao, S., Rao, J., Duarte, F., et al. (2020). Understanding house price appreciation using multi-source big geo-data and machine learning. *Land Use Policy*, Article 104919.

Kwak, S. K., & Kim, J. H. (2017). Statistical data preparation: management of missing values and outliers. *Korean journal of anesthesiology*, *70*, 407–411.

Van der Laan, M. J., Polley, E. C., & Hubbard, A. E. (2007). Super learner. *Statistical applications in genetics and molecular biology*, *6*.

Lam, K. C., Yu, C., & Lam, K. (2008). An artificial neural network and entropy model for residential property price forecasting in hong kong. *Journal of Property Research*, *25*, 321–342.

Large, J., Lines, J., & Bagnall, A. (2019). A probabilistic classifier ensemble weighting scheme based on cross-validated accuracy estimates. *Data mining and knowledge discovery*, *33*, 1674–1709.

Law, S., Paige, B., & Russell, C. (2019). Take a look around: using street view and satellite images to estimate house prices. *ACM Transactions on Intelligent Systems and Technology*, *10*, 1–19.

Limsombunchai, V. (2004). House price prediction: hedonic price model vs. artificial neural network. In *New Zealand agricultural and resource economics society conference* (pp. 25–26).

Maurer, R., Pitzer, M., & Sebastian, S. (2004). Hedonic price indices for the paris housing market. *Allgemeines Statistisches Archiv*, *88*, 303–326.

Park, B., & Bae, J. K. (2015). Using machine learning algorithms for housing price prediction: The case of fairfax county, virginia housing data. *Expert Systems with Applications*, *42*, 2928–2934.

Polley, E. C., & Van Der Laan, M. J. (2010). *U.C. Berkeley division of biostatistics working paper series*, *Super learner in prediction*: *Working Paper*, (p. 266).

Rahman, S. N. A., Maimun, N. H. A., Razali, M. N. M., & Ismail, S. (2019). The artificial neural network model (ANN) for malaysian housing market analysis. *Planning Malaysia*, 17.

Rosen, S. (1974). Hedonic prices and implicit markets: product differentiation in pure competition. *Journal of political economy*, *82*, 34–55.

Rubin, D. B. (1978). Multiple imputations in sample surveys-a phenomenological bayesian approach to nonresponse. In *Proceedings of the survey research methods section of the American statistical association, Vol. 1* (pp. 20–34). American Statistical Association.

Selim, H. (2009). Determinants of house prices in turkey: hedonic regression versus artificial neural network. *Expert systems with Applications*, *36*, 2843–2852.

Shirzadi, A., Soliamani, K., Habibnejhad, M., Kavian, A., Chapi, K., Shahabi, H., et al. (2018). Novel gis based machine learning algorithms for shallow landslide susceptibility mapping. *Sensors*, *18*(3777).

Temur, A. S., Akgün, M., & Temur, G. (2019). Predicting housing sales in turkey using arima, lstm and hybrid models. *Journal of Business Economics and Management*, *20*, 920–938.

Tsai, I.-C. (2013). The asymmetric impacts of monetary policy on housing prices: A viewpoint of housing price rigidity. *Economic Modelling*, *31*, 405–413.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in neural information processing systems*, *599*, 8–6008.

Wang, Y., Wang, H., & Peng, Z. (2021). Rice diseases detection and classification using attention based neural network and bayesian optimization. *Expert Systems with Applications*, *178*, Article 114770.

Wang, L., Zeng, Y., & Chen, T. (2015). Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. *Expert Systems with Applications*, *42*, 855–863.

Wilson, I. D., Paris, S. D., Ware, J. A., & Jenkins, D. H. (2002). Residential property price time series forecasting with neural networks. In *Applications and innovations in intelligent systems IX: Proceedings of ES2001, the twenty-first SGES international conference on knowledge based systems and applied artificial intelligence, Cambridge, Vol. 2001* (pp. 17–28). Springer.

Wod, I. (1985). Weight of evidence: A brief survey. *Bayesian statistics*, *2*, 249–270.

Yang, Z. (2001). An application of the hedonic price model with uncertain attribute-the case of the people's republic of china. *Property management*, 50–63.

Yang, L., Chen, Y., Xu, N., Zhao, R., Chau, K., & Hong, S. (2020). Place-varying impacts of urban rail transit on property prices in shenzhen, china: Insights for value capture. *Sustainable Cities and Society*, *58*, Article 102140.

Zeyer, A., Bahar, P., Irie, K., Schlüter, R., & Ney, H. (2019). A comparison of transformer and lstm encoder decoder models for asr. In *2019 IEEE automatic speech recognition and understanding workshop* (pp. 8–15). IEEE.

Zhang, G., Zhang, C., & Zhang, W. (2020). Evolutionary echo state network for long-term time series prediction: on the edge of chaos. *Applied Intelligence*, *50*, 893–904.

Zhou, J., Huang, S., Wang, M., & Qiu, Y. (2021). Performance evaluation of hybrid ga–svm and gwo–svm models to predict earthquake-induced liquefaction potential of soil: a multi-dataset investigation. *Engineering with Computers*, 1–19.