Enhancing Housing Price Prediction in Real Estate through Advanced Machine Learning Techniques

Authors:

- **Omair Ansari**
  MSc in Data Science, Thompson Rivers University
  TRU ID: T00751056
- **Jiayi Zhang**
  MSc in Data Science, Thompson Rivers University
  TRU ID: T00752757
- **Jiahui Yan**
  MSc in Data Science, Thompson Rivers University
  TRU ID: T00752485
- **Sree Aryan Sathyamurthy Parimala Priyadharshini**
  MSc in Data Science, Thompson Rivers University
  TRU ID: T00751318

Affiliation: Faculty of Science, Thompson Rivers University, Kamloops, Canada

Date: 9th December 2024

Abstract

For real estate, which acts as the blood of economic activity as well as wealth generation, housing price prediction becomes very important. Though traditional methods of valuation, such as heuristics and linear regression models, do not seem to be able to capture the complexities and the dynamism of the housing market, resulting in inaccuracies and inefficiencies, this study studies the advanced machine learning techniques to develop the performance of the outcomes for prediction of price in real estate.

Using more than 487,000 entries for Canadian housing data, various models of Random Forest, XGBoost, LightGBM, CatBoost, and Neural Networks were employed and compared. The Pre-processing of feature encoding, data normalization, and missing handled data made the model robust to training. Ensemble models showed superiority with Random Forest giving highest accuracy ($R^2 = 0.988$) while CatBoost suited best for processing categorical data. The other major features for housing prices with bathrooms, square footage, and walkability score would directly lead to giving effectual insights to the stakeholders.

Limitations on the existing methodologies include the absence of temporal information, lack of adequate regional customization, and issues regarding interpretability. Future directions may

involve combining the above with macroeconomic variables, model development regionally specific, and hybrid ensemble copying effort having an accuracy efficiency. Thus, the findings hold true to the transformation brought by machine learning applications into real estate, enhanced by predictive precision as well as practical applications in very diverse market scenarios.

1 Introduction

Real estate is among the most vital fields for economic activity and wealth building in the world. The housing markets affect not only people looking to buy or sell houses but also have wider implications for national economies. This makes accurate property price prediction very important in this sector; key stakeholders like homebuyers, investors, developers, and policymakers make informed strategic decisions. A good price estimation system will, therefore, make property transactions smooth, reduce the risk of overvaluation or undervaluation, and make the market more transparent.

Traditional approaches to real estate valuation rely quite a lot on heuristic methods, market appraisals, and simple regression-based models. While these have worked reasonably well, they can never truly capture the complexity and dynamics of housing markets. Adding factors such as changes in demand and supply, macroeconomic indicators, changing preferences of buyers, and geographical disparities brings variability that traditional models can be adapted to easily. Also, most of the valuation procedures are tiresome and, as a rule, exposed to human bias—thus, inconsistencies in property prices keep happening.

Machine learning has brought a paradigm shift to many different industries, including real estate. ML algorithms excel at processing big volumes of raw data and finding patterns that are difficult to identify using classical methods of analysis. Machine learning models can expose hidden connections between specific property info and market trends in ways not intuitive to humans, such as through trending neighborhood factors to expose unseen influences on price.

So-called neighborhood characteristics, such as distance to schools and public transport, are among the variables that a machine learning model receives along with more conventional metrics like square footage and age of building.

However, there are a few challenges to leveraging machine learning in real estate. One of the big problems the industry faces is data fragmentation: information is scattered across various sources, from property listings and municipal records to census data. Besides, housing markets are so diverse from region to region that model adaptation to local conditions becomes a necessity. These are big challenges; however, the upside in machine learning applied to price prediction in real estate could be huge, offering much more precision and reliability than anything that had been possible with the stakeholders heretofore.

1.1 Aims

The goal of this project is to apply several machine learning algorithms in order to improve prediction accuracy and efficiency of real estate prices. This addresses a number (listed below) of main research questions and objectives:

Developing Predictive Models:

Implementing different machine learning algorithms like linear regression, decision tree, random forest, or even a neural network to predict real estate prices. Through comparing these models we hope to be able to find the best methods that capture these complex relational structures that exist in real estate data.

Identifying Key Predictors:

Real estate prices are affected by many things — property attributes such as square footage, number of bedrooms and its age; location-specific factors such as distance to schools or public transport; and broader market trends such as inflation and interest rates, addresses that property value is broadly dependent on a selection of different predictors and the aim here is to review and order these predictors to understand the most important aspects of these properties

Addressing Regional Variability

Given the diversity in housing markets across cities and regions, the research will look into ways of adapting models to take differences in regions into consideration. This involves tuning the feature selection and algorithm parameters to better capture local market dynamics. Demonstrating Real-World Applications: Going beyond theoretical development, this project tries to prove the practical usefulness of machine learning models in real estate. Analysing a large dataset, this study will extract actionable insights for stakeholders and show how large-scale adoption of data-driven pricing strategies is possible. Benchmarking Against Traditional Methods The project will benchmark the advantages of machine learning in making predictions by comparing them to traditional statistical methods and appraisals. Such benchmarking will point out areas where machine learning shows clear improvements and scenarios where traditional methods may still be relevant. In achieving these objectives, the project would address a little something about making a real contribution within real estate in addressing century-old difficulties of property valuation with the help of advanced analytical techniques. The results will provide much-needed energy to other fields and professions with highly complex data environments and multivariate datasets.

2. Literature Review

2.1 Related Work/Previous Literature

In recent years, there have been growing applications of machine learning (ML) to real estate price prediction, as researchers and practitioners realize the potential of ML to model complex relationships in property data. Many studies have researched applying different types of ML algorithms like linear regression, decision trees, support vector machine (SVM), and neural networks to property price prediction using historical data. These studies seek to build better statistical methods than where we are with traditional statistics, such as their reliance on linear assumptions and inability to handle large, multidimensional datasets.

To illustrate, regression models were traditionally the baseline method for price predictions because of their simplicity and interpretability. Yet, for the analysis of relationship between property features and market dynamics, regression methods are still more likely to underperform as per the studies. For instance, algorithms like decision trees and gradient boosting models achieved higher accuracy statistics simply because machine learning models unveiled patterns in data not easily visible with traditional models.

Ensemble approaches (e.g., Random Forest (RF), Gradient Boosted Trees (GBT), and XGBoost) have been demonstrated to perform well to predict the real-estate price, according to recent studies. Stacking, bagging, and boosting are used to combine and improve the predictions of several weak learners, improving model performance by quite some margin. Deep learning models have also become popular in the field, as it is explicitly designed to take advantage of unstructured data, like images of homes, or textual descriptions of various properties.

Remaining Challenges:

Almost all studies have a major limitation, that is, an incompleteness in handling categorical data, such as property types or zoning information, that become very critical in predictive modeling. While many individual ML models are available, there is little systematic comparison of various ensemble methods to identify the best ones under different conditions.

2.2 Contributions of This Study

This study, thus, relates the discourse at present to research by providing these gaps as attachments:

Comparative Study of Ensemble Models:

Comparative study of the various ensemble techniques like Random Forest, Gradient Boosted Trees, and XGBoost-based implementation and evaluation on real estate data, this study has made yet another comprehensive comparison in prediction performance for different model options. This will help for an informed choice of the most applicable model for a particular case.

High-level Processing of Categorical Variables:

This study employs target encoding and embedding representations to model easily categorical features into predictive models to increase model performance and also efficiency.

Accuracy and Interpretation Balance:

Except for attaining the high level of accuracy stated above, this study joins the research aimed at being interpretable models. Individual predictors' contribution will be made explainable by means of techniques, such as those of Shapley Additive explanations, hence defying to-transparent interpretations for industry stakeholders.

Model Adaptation Based on Regional Contexts:

This study is based on a dataset from various Canadian cities for the development of a fitted localized model to fit its region variation. The findings may indicate the extent to which predictive models can be made relevant for specific market conditions.

3. Dataset and Preprocessing

3.1 Dataset Description

The dataset used to generate models contains 487,202 entries and 20 columns. Here's a summary of the key features:

Table 1. Key Features of Dataset 1

Feature	Description	Datatype
Rentfaster_id	Unique Identifier for each Listing	Integer
City	City of the listing	Object
Province	Province of the listing	Object
Latitude Longitude	Geographic Location of the Coordinates	Float
Price	Monthly Rental Price	Float
Beds, Baths	Number of Bedrooms and Bathrooms	Object
Sq_feet	Square foot of the property	Object
Bike Score, Walk Score	Accessibility rating for Biking and Walking	Float

The dataset used to analyze historical trends of housing prices contains 14,280 entries and 4 columns. Here's a summary of the key features:

Table 2. Key Features of Dataset 2

Feature	Description	Datatype
Date	Monthly date	Date
Composite_Benchmark_SA	Benchmark Price of House	Integer
Townhouse_Benchmark_SA	Benchmark Price of Townhouse	Integer
Apartment_Benchmark_SA	Benchmark Price of Apartment	Integer

3.2 Data Analysis

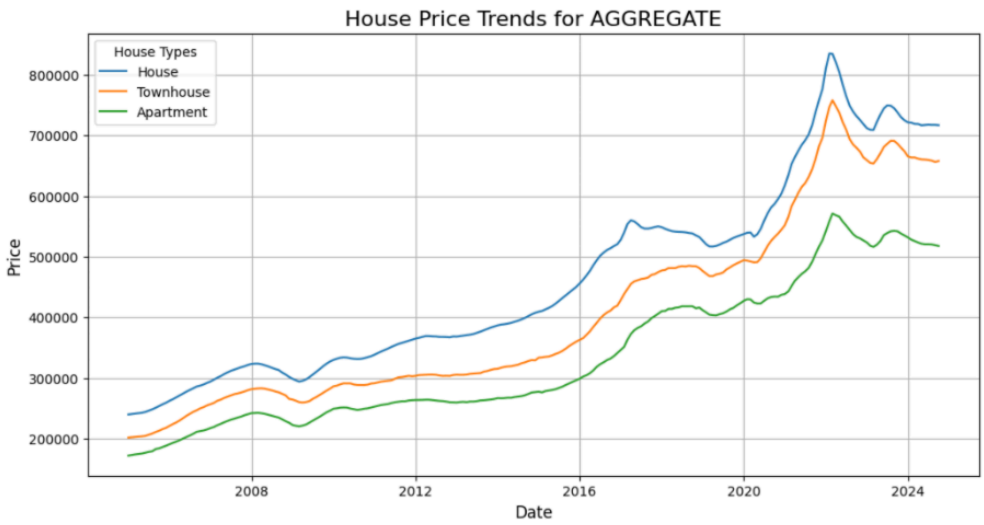

Fig 1. House Price Trend for Aggregate

We analyzed historical monthly data of housing prices for three types of homes—detached houses, townhouses, and apartments—from January 2005 to October 2024 in 10 provinces and 50 cities across Canada. The overall trend shows consistent growth in housing prices across all regions as shown in Figure 1. Detached houses are the most expensive, followed by townhouses, with apartments having the lowest prices.

A slight drop in housing prices occurred in 2009, attributed to the global financial crisis. This period saw reduced consumer confidence, tighter credit conditions, and a decrease in housing demand, resulting in a temporary decline in prices. From 2017 to 2018, there was a moderate increase in housing prices, driven by low interest rates and a strong economy. This combination boosted housing demand, especially in urban centers. A significant price surge occurred after 2020, largely due to the COVID-19 pandemic and inflation. The pandemic led to higher demand for detached homes as remote work became more prevalent, with people seeking larger living spaces. Inflation, which increased construction costs and homebuyer demand, also contributed to the rise in prices. After 2022, housing prices began to decline, and the trend has since stabilized, showing little fluctuation.

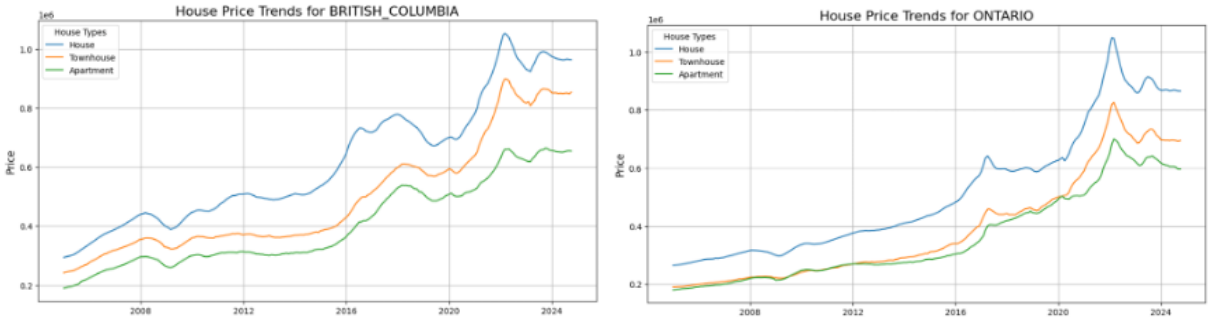

Fig 2. House Price Trend for British Columbia and Ontario

At the provincial level, Ontario and British Columbia consistently had housing prices above the national average as shown in Figure 2. Their trends closely mirrored the national data, driven by major economic hubs like Toronto and Vancouver. Both cities saw strong housing demand, which pushed prices higher.

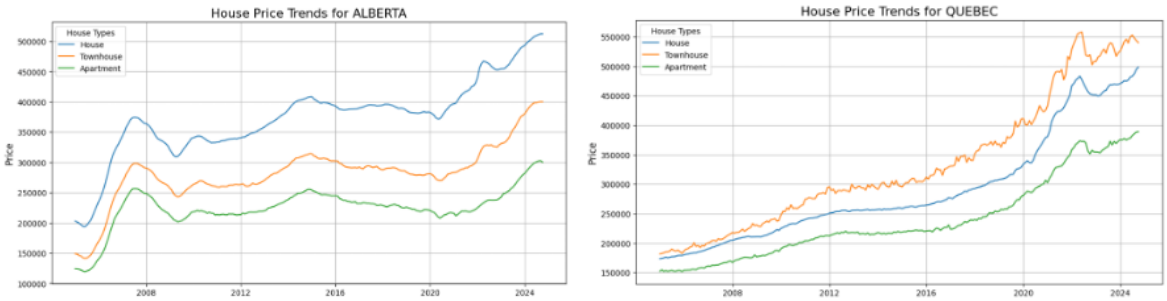

Fig 3. House Price Trend for Alberta and Quebec

In contrast, Alberta and Quebec showed different trends as shown in Figure 3. Despite lower peak prices (about half those in Ontario and British Columbia), both provinces experienced continuous price growth. Alberta's market benefitted from a rebound in oil prices and population growth, while Quebec's housing demand, particularly in Montreal, was driven by economic expansion and an influx of new residents. These trends reflect regional differences, illustrating how local economic factors shaped the housing market.

3.3 Preprocessing Steps

Dropping Unrelated Columns:

Columns like address and link are mostly text-heavy and cannot be much significant in any analysis or predictive tasks. Hence, removal from here.

Dealing with Missing Data:

- Columns smoking, cats, and dogs contain lots of missing data.

- Imputing with a mode or one option “unknown" value.
- Dropping the row or columns if missing in significant amounts or not applicable.

Encoding Categorical Variables:

City, province, and furnished are transformed numerically through one-hot encoding (for machine learning) or label encoding.

Standardization of Numeric Features:

Price, Walk Score, and Bike Score differ significantly in scale, and these can all be standardized, such as by z-score normalization so that during analysis they contribute to the process equally.

Cleanup Data Related Challenges:

Mixed Data Types:

- One of the columns, say beds, has different forms of writing. For example, '2 Beds' means that 2 should be transformed.
- 'Studio' or 'no Beds', for example, would mean 0 or mapped accordingly.

Work on Outlier:

The price column has outliers that may cause distortion of results. For instance: Most of the rental prices fall into a more realistic range, while few pull sky-high.

These outliers could be either capped at a maximum value or removed altogether depending on the use case.

Feature Engineering for Text Columns:

Some text-heavy columns (i.e., e.g.: lease term and furnishing) would require much more attention for powerful information extraction.

Key Features:

Geographic: latitude and longitude, spatial analysis by location (e.g. cluster properties based on location).

Property Characteristics: Type (type), size (beds, baths, Sq_feet), and lease terms (lease term).

Price Data: Provides different rental costs on different properties, making it an effective market price research tool.

Walkability and Accessibility:

1) Walk Score: Measures how walkable the area around a property is.
2) Bike Score: Evaluates how bike-friendly the area is.

4. Model Selection

Before building the models, the dataset was meticulously processed to enhance the models' ability to learn underlying patterns efficiently. Specifically, numerical features were standardized using z-scores, which rescale the data to have a mean of zero and a standard deviation of one, thereby mitigating the effects of feature scale disparities. Categorical variables were transformed through label encoding, assigning each unique category a distinct integer value. This encoding was essential for models that require numerical input but do not inherently handle categorical data. After preprocessing, the dataset comprised 20 features.The dataset was then split into training and testing subsets with a ratio of 4:1 using stratified sampling via the scikit-learn package [5]. Stratified sampling ensures that the distribution of the target variable is preserved across both subsets, enhancing the generalizability of the model evaluations.

The evaluation metrics employed in this study are Mean Absolute Error (MAE), Mean Squared Error (MSE), and the coefficient of determination (R^2). These metrics assess the average error, penalize larger errors, and quantify the proportion of variance explained by the models, respectively.

4.1 Random Forest Regressor

Random Forest is an ensemble learning method that combines the predictions of multiple decision trees to improve overall accuracy and reduce overfitting. It is recognized for its robustness and versatility in handling both numerical and categorical data.

Algorithm Overview:

Random Forest operates through the following steps, as outlined by S. Raschka and V. Mirjalili in “Python Machine Learning” [11]:

1. **Bootstrap Sampling:** Draw a random bootstrap sample of size nn (randomly select nn samples from the training set with replacement).
2. **Tree Construction:** Grow a decision tree from the bootstrap sample. At each node:
    - **Feature Selection:** Randomly select dd features without replacement.
    - **Best Split:** Split the node using the feature that provides the best split according to the objective function (e.g., maximizing information gain).
3. **Iteration:** Repeat steps 1-2 kk times to build kk decision trees.
4. **Aggregation:** Aggregate the predictions from each tree by computing their mean to assign the final regression output.

Parameter Settings:

- n_estimators = 100: Number of trees in the forest.
- random_state = 42: Ensures reproducibility of results.
- max_depth = 20: Restricts the maximum depth of each tree to prevent overfitting.
- min_samples_split = 10: A node must have at least 10 samples before it can be split.

Performance Metrics:

- **MAE:** 28.38
- **MSE:** 9,651.76
- **R^2:** 0.988

Model Performance:

The Random Forest Regressor exhibited exceptional performance on the training data, achieving the lowest MAE among all models. Its ability to handle high-dimensional data and mitigate overfitting through ensemble averaging makes it a powerful tool for regression tasks. Figure 1 illustrates the correlation between the predicted and actual housing prices, showcasing the model's high precision.

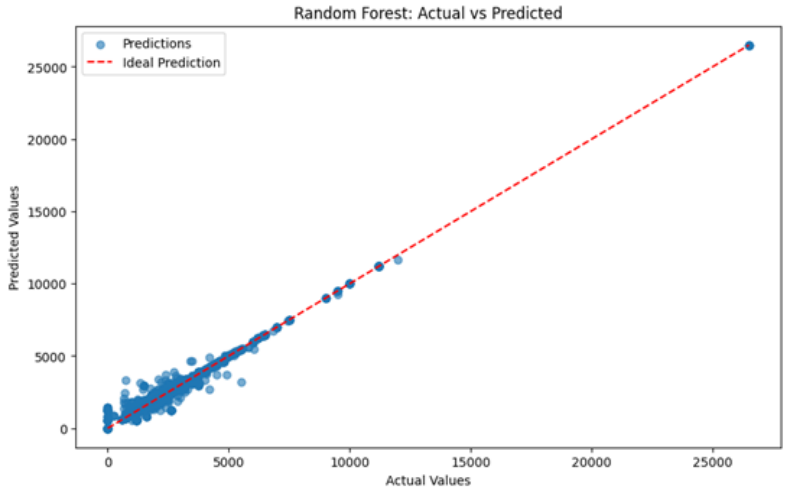

Fig.4. Random Forest for Training Data

4.2 Extreme Gradient Boosting (XGBoost)

XGBoost is a scalable and efficient gradient boosting framework renowned for its performance and speed. It builds trees sequentially, with each new tree aiming to correct the errors of the preceding ensemble, thereby incrementally improving the model's accuracy.

Key Features:

- **Regularization:** Incorporates L1 and L2 regularization to prevent overfitting.
- **Handling Missing Data:** Natively manages missing values, enhancing flexibility.
- **Parallel Processing:** Utilizes parallel computation to expedite training.

Parameter Settings:

- n_estimators = 200: Number of boosting rounds.
- learning_rate = 0.1: Step size for updating model weights.
- max_depth = 6: Maximum depth of each tree.
- min_child_weight = 2: Minimum sum of instance weight needed in a child.
- subsample = 1: Subsample ratio of the training instances.
- colsample_bytree = 0.8: Subsample ratio of columns when constructing each tree.
- reg_lambda = 0.45: L2 regularization term.
- reg_alpha = 0: L1 regularization term.
- gamma = 0.5: Minimum loss reduction required to make a further partition.

Performance Metrics:

- **MAE:** 116.99
- **MSE:** 32,744.09
- **R^2:** 0.960

Model Performance:

XGBoost demonstrated high accuracy with a robust R value, indicating its effectiveness in explaining the variance in housing prices. The model's scalability and efficient handling of large datasets make it suitable for complex regression tasks. Figure 5 depicts the distribution of XGBoost's predictions against actual values, highlighting its predictive reliability.

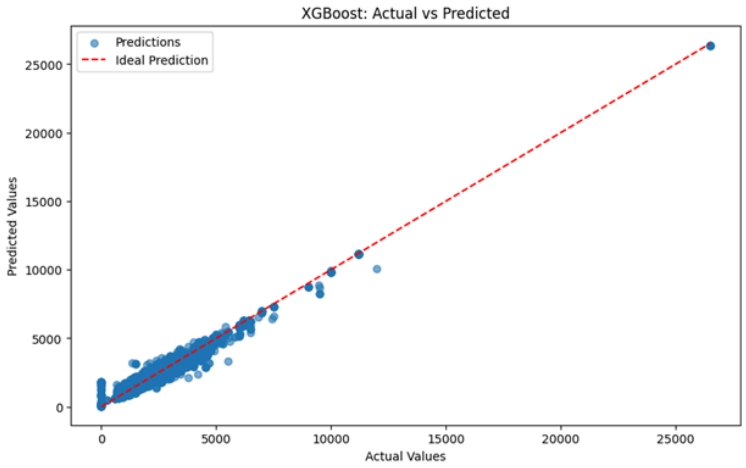

Fig.5 XGBoost for Training Data

4.3 Light Gradient Boosting Machine (LightGBM)

LightGBM is an advanced gradient boosting framework optimized for speed and memory efficiency. It is particularly effective with large-scale datasets and offers several enhancements over traditional boosting algorithms.

Key Features:

- **Histogram-Based Algorithms:** Reduces memory usage and accelerates training by binning continuous feature values.
- **Leaf-Wise Tree Growth:** Grows trees leaf-wise rather than level-wise, allowing for deeper and more complex trees that capture intricate data patterns.
- **Parallel Training and GPU Acceleration:** Supports parallel and GPU-based training, significantly speeding up the learning process.
- **Native Categorical Handling:** Processes categorical variables directly without the need for explicit encoding.

Parameter Settings:

- learning_rate = 0.15: Step size for weight updates.
- n_estimators = 64: Number of boosting iterations.
- min_child_weight = 2: Minimum sum of instance weight needed in a child.
- num_leaves = 36: Maximum number of leaves per tree.
- colsample_bytree = 0.8: Subsample ratio of columns when constructing each tree.
- reg_lambda = 0.40: L2 regularization term.
- random_state = 42: Ensures reproducibility.

Performance Metrics:

- **MAE:** 129.98
- **MSE:** 41,424.17
- **R^2:** 0.949

Model Performance:

LightGBM achieved decent accuracy with an R^2 value close to 0.95, indicating its capability to model the underlying data effectively. While its performance was not as strong as some other models, LightGBM's superior training speed and memory efficiency make it the fastest among all models evaluated in this study. Figure 7 illustrates the relationship between LightGBM's predicted and actual housing prices, demonstrating its performance.

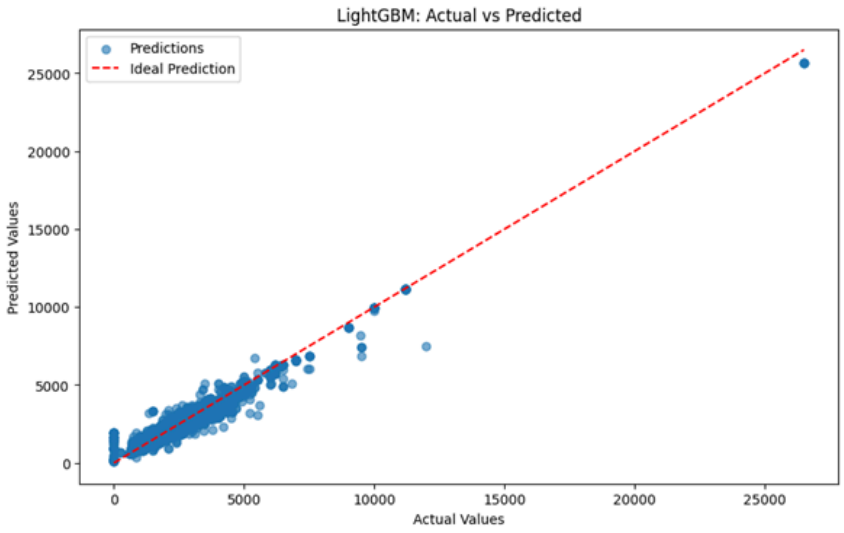

Fig.7. LightGBM for Training Data

4.4 CatBoost Regressor

CatBoost is a gradient boosting algorithm specifically optimized for handling categorical data. It introduces unique methodologies to address overfitting and leverage the strengths of categorical feature processing, making it highly effective for datasets with significant categorical variables.

Key Features:

- **Categorical Variable Handling:** Processes categorical features directly without the need for manual encoding, utilizing target and ordered encoding techniques.
- **Ordered Boosting:** Prevents data leakage by ensuring that the model only uses past data for training during boosting.
- **GPU Acceleration:** Enhances training efficiency, especially beneficial for large datasets.
- **Robust Handling of Overfitting:** Minimizes overfitting through its ordered boosting and regularization techniques.

Parameter Settings:

- iterations = 1000: Number of boosting iterations.
- learning_rate = 0.1: Learning rate for gradient updates.
- depth = 6: Maximum depth of each tree.
- random_state = 42: Ensures reproducibility.
- verbose = 100: Frequency of progress updates during training.

Performance Metrics:

- **MAE:** 79.37

- **MSE:** 17,973.28
- **R^2:** 0.978

Model Performance:

CatBoost achieved excellent performance with a high R^2 value, nearly matching the Random Forest Regressor. Its ability to handle categorical variables natively streamlined the preprocessing workflow and preserved the integrity of categorical relationships. Figure 8 showcases CatBoost's predictions on the test dataset, highlighting its alignment with actual housing prices.

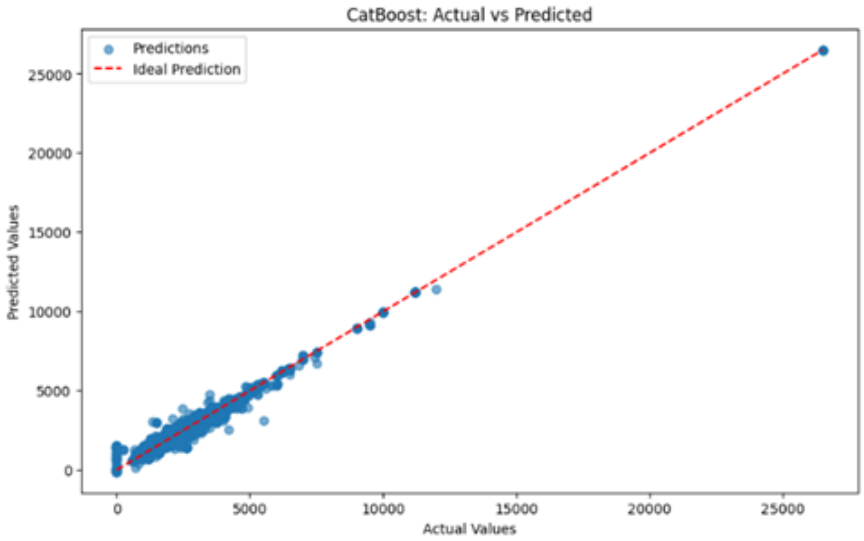

Fig.8. CatBoost for Training Data

4.5 Gradient Boosting Regressor

The Gradient Boosting Regressor is a foundational ensemble technique that builds models sequentially, with each new model aiming to correct the errors of its predecessors. This method is renowned for its flexibility and strong predictive performance across various applications.

Key Features:

- **Sequential Model Building:** Constructs trees one at a time, with each tree focusing on minimizing the residual errors of the ensemble thus far.
- **Customizable Loss Functions:** Supports a wide range of loss functions, enabling tailored approaches to specific regression problems.
- **Regularization Techniques:** Incorporates shrinkage (learning rate adjustment) and subsampling to control overfitting and enhance generalization.

- **Flexible Tree Structures:** Allows customization of tree depth and other hyperparameters to balance bias and variance effectively.

Parameter Settings:

- n_estimators = 100: Number of boosting iterations.
- learning_rate = 0.1: Learning rate for weight updates.
- max_depth = 6: Maximum depth of each tree.
- random_state = 42: Ensures reproducibility.

Performance Metrics:

- **MAE:** 103.54
- **MSE:** 29,409.59
- R^2**:** 0.955

Model Performance:

The Gradient Boosting Regressor delivered solid performance with a R^2 value of 0.955, indicating a strong capacity to explain the variance in the target variable. While effective, its MAE and MSE were higher compared to the Random Forest and CatBoost models, suggesting that it may benefit from further hyperparameter tuning or additional regularization to achieve lower error margins. Figure 9 illustrates the correlation between the Gradient Boosting Regressor's predictions and actual housing prices, demonstrating its predictive capability.

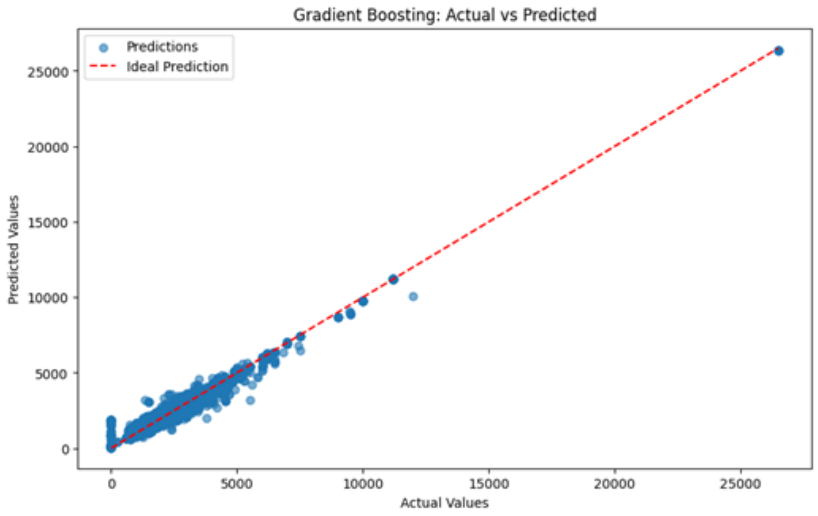

Fig.9 Gradient Boosting for Training Data

4.6 Neural Network

Neural Networks are a class of machine learning models inspired by the human brain's network of neurons. They are particularly adept at modelling complex, nonlinear relationships in data through layers of interconnected nodes (neurons), each applying transformations to the input data.

Key Features:

- **Dense Layers:** Fully connected layers that enable the network to learn intricate feature interactions by combining inputs in a weighted manner.
- **Activation Functions:** Nonlinear activation functions like Rectified Linear Unit (ReLU) introduce nonlinearity, allowing the network to model complex relationships.
- **Dropout Layers:** Implement dropout regularization by randomly deactivating a subset of neurons during training to prevent overfitting.
- **Backpropagation and Optimization:** Utilizes backpropagation algorithms to adjust weights based on the gradient of the loss function, optimizing the model's performance iteratively.

Architecture:

- **Input Layer:** Receives the preprocessed features from the dataset.
- **Hidden Layers:** Comprises three layers with 128, 64, and 32 neurons respectively, performing successive transformations on the input data to extract higher-level features.
- **Dropout:** Applied after each hidden layer to regularize the model by preventing co-adaptation of neurons.
- **Activation Functions:** ReLU activation is used in hidden layers to introduce nonlinearity.
- **Output Layer:** Consists of a single neuron that outputs the regression prediction, providing a continuous value based on the learned representations.

Parameter Settings:

- **Optimizer:** Adam optimizer for efficient gradient descent.
- **Loss Function:** Mean Squared Error (MSE) for regression tasks.
- **Batch Size:** 32 for training iterations.
- **Epochs:** 50 to allow sufficient training iterations.
- **Dropout Rate:** 0.5 to effectively prevent overfitting.

Performance Metrics:

- **MAE:** 160.37
- **MSE:** 112,318.34
- **R^2:** 0.826

5. Results

The Neural Network model, despite its flexibility and capacity to model complex relationships, underperformed relative to the tree-based models in this study. The lower R^2 value and higher MAE and MSE indicate that the Neural Network was less effective in capturing the underlying patterns of the dataset. Several factors likely contributed to this outcome:

1. **Dataset Size:** Neural Networks generally require large amounts of data to learn effectively. A limited dataset size may hinder the network's ability to generalize.
2. **Model Complexity:** The architecture, while moderately deep, may not have been optimal for the specific dataset, leading to potential overfitting or underfitting.
3. **Training Challenges:** Sensitivity to hyperparameters such as learning rate, batch size, and the number of epochs can impact performance if not optimally tuned.
4. **Feature Engineering:** Unlike tree-based models, Neural Networks benefit significantly from feature engineering and may require more sophisticated preprocessing to unlock their full potential.

Despite these challenges, Neural Networks offer unparalleled flexibility and can excel in scenarios with ample data and proper tuning. They are particularly well-suited for tasks involving image, text, or sequential data, where their ability to learn hierarchical representations is advantageous. In this regression task, however, the inherent strengths of tree-based models aligned more closely with the data characteristics, resulting in superior performance. Figure 10 visualizes the Neural Network's predictions on the test dataset, highlighting areas where the model deviated from actual values.

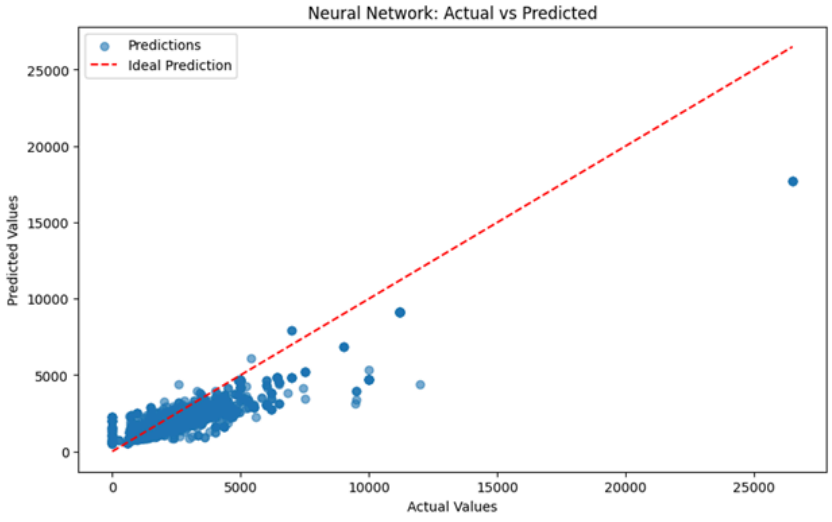

Fig.10 Neural Network for Training Data

5.1 Model Performance

Many iterations of performance tuning were done to find the optimal solution of each model.

All six models were tuned by the function in scikit-learn Python pack to achieve the result in bar graph Fig.11. and the comparison chart table.3.. There are promoting methods that can enhance result performance in next-step research like Hybrid Regression or other ensemble learning methods by combining these models together in proper weights to get better regression. The parameter in each model could be adjusted in further studies.

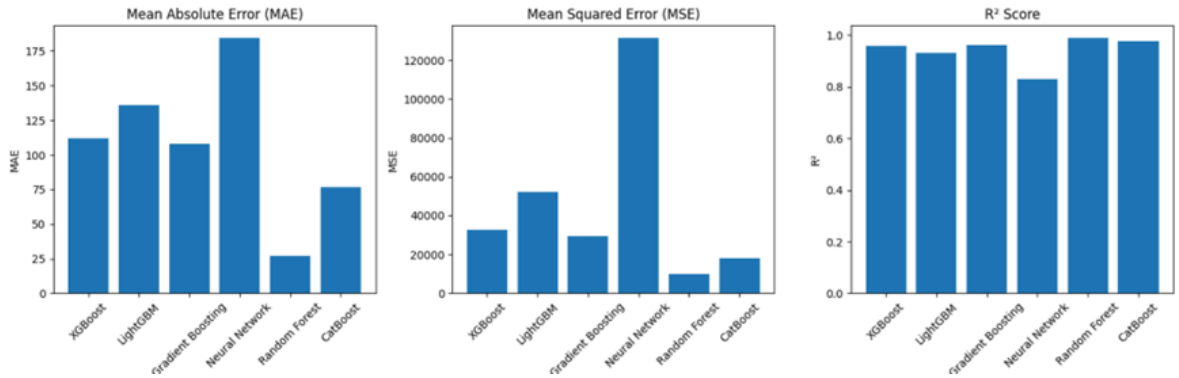

Fig.11. Bar graph visualization of the MAE, MSE,prediction result

Table 3. Prediction result

Model	MAE	MSE	R^2
XGBoost	112.01	32747.82	0.96
LightGBM	135.69	52268.50	0.93
Gradient Boosting	108.00	29557.57	0.96
Neural Network	184.10	131192.58	0.83
Random Forest	27.17	9878.28	0.99
CatBoost	76.74	17882.99	0.98

As listed in Table 1,the performance of another four models is similar,76.74-135.69,17882.99-52268.50,0.93- 0.98 in following MAE,MSE, and R^2 because of their alike model principle. More specifically, CatBoost is the best model in these Gradient Boosting Machine models. One of the reasons is that CatBoost is great at dealing with medium

size datasets by selecting crucial parameters and features as the first step of modeling. It is surprised to discover every model accomplishes the predicted mission with the R^2 is all above 0.83, which is a satisfied result.

5.2 Feature Importance Analysis

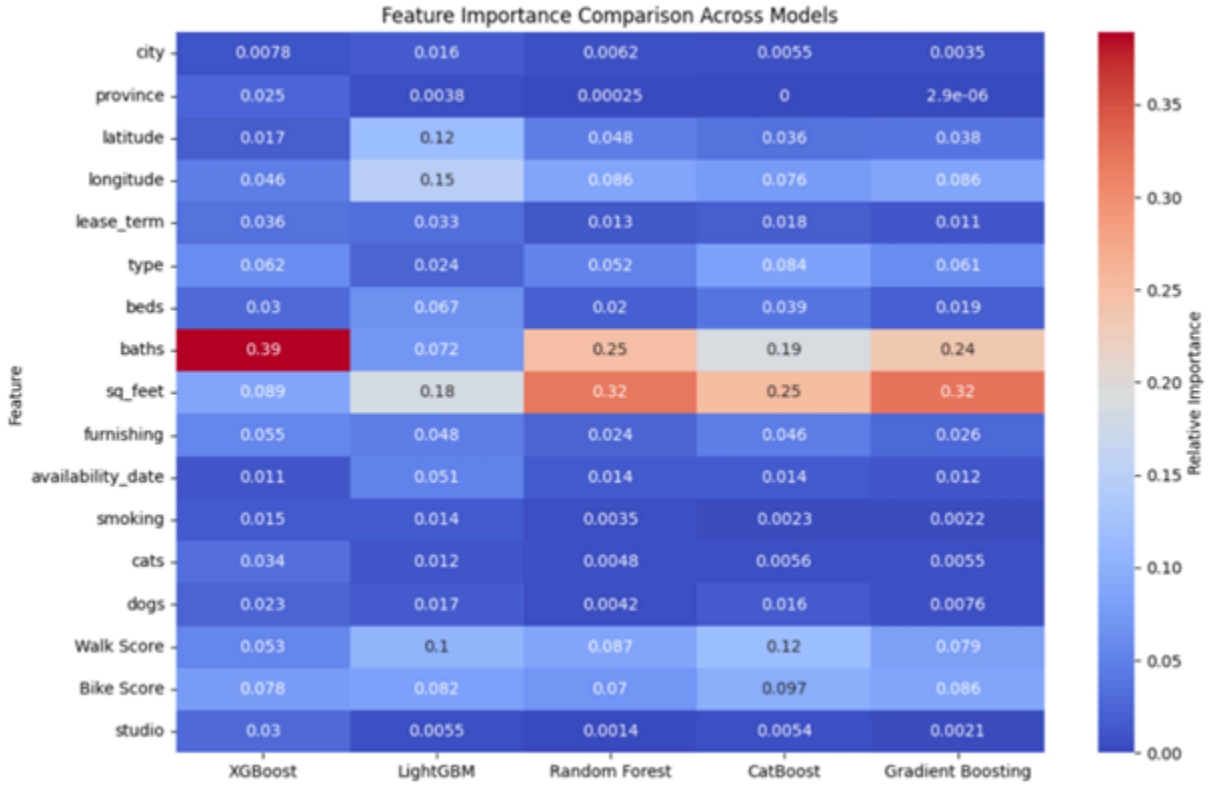

Fig.12 Feature importance comparison across models

Fig.12. presents the feature importance across models except Neural Network. Baths and sq_feet are the top 2 important factors among features. In the most radical situation, baths take up nearly a half of the XGBoost parameter analyzation process in 0.39 importance score. Unlike other models, LightGBM mainly focuses on modeling speed with the sacrifice of precise in feature importance recognizing. So, beside LightGBM, baths and sq_feet significantly stand out among features. The result meets the reality phenomenon perfectly. In general scenarios, the space of the house, housing locations are relatively important, in contrast to cats or dogs, walks or bikes--these irrelevant things when making an evaluation of the house. It is noteworthy that location relevant features: latitude and longitude are not as significant as we considered. One possible reason is that we have not input Canada map and the weight of every city into the model, resulting in errors of predicting.

5.3 Insights

There is no denying that every feature influences the prediction of housing prices in different aspects. However, there are some priorities to the features. Moreover, there are some insights about the modeling worth conclusion.

Features priority: baths/space(sq_feet)>walk/bike score> others

In routine life, when choosing an acceptable price of housing, it is nearly impossible to consider all factors together due to the complexity and time-consuming in analyzing. So, the result above gives us guidance to choose a house more efficiently. The ratio of price divide space amount is the key to compare different housing locations and space.

Modeling choosing: CatBoost and Random Forest(original)

These two models give the best performance in result output. It is justified when combining models together, CatBoost and Random Forest deserve higher weight. There is a need to mention that time complexity must be taken into consideration since both of them contain Random Forest, a high time complexity model.

6. Case Study: Predicting Prices for New Properties

In this case study, we explore how machine learning models, including XGBoost, LightGBM, Gradient Boosting, and Random Forest, can be utilized to predict the housing prices of unseen properties. By applying these models to real estate data from Kamloops, British Columbia, we aim to compare the predictions against actual market prices and analyze the model performance.

Step 1: Preprocessing Unseen Data

The first crucial step in applying machine learning models to predict housing prices is ensuring that the data matches the format the models expect. In this case, we define a function preprocess_unseen_data to preprocess new, unseen property data. The function first converts the new property data (in the form of a dictionary) into a pandas DataFrame. Secondly, we use a LabelEncoder to turn text into categorical data (like city, province, lease_term, etc.) into numbers so that the model can understand them, checked for Missing Columns and scaled the numerical features using the same scaling method that was used during training. After making these changes, the function returns the preprocessed and scaled data, which is now ready to be used by the model for prediction.

Step 2: Model Predictions for New Properties

Once the unseen data is preprocessed, we can apply the trained models to make predictions. For this case study, we use a real-world property in Kamloops, BC, as an example. The data for House1 and House2 is shown below:

Table 4. Features of House1&2

Columns	Features of House1	Features of House2
'city'	'kamloops'	'kamloops'
'province'	'british columbia'	'british columbia'
'latitude'	50.670143906487	50.670143906487
'longitude'	-120.22682388089	-120.22682388089
'lease_term'	'long term'	'long term'
'type'	'apartment'	'apartment'
'beds'	2	1
'baths'	2	1
'sq_feet'	980	522
'furnishing'	'furnished'	'unfurnished'
'availability_date'	'immediate'	'immediate'
'smoking'	'non-smoking'	'non-smoking'
'cats'	1	1
'dogs'	1	1
'Walk Score'	24	89
'Bike Score'	14	68
'studio'	0 # Not a studio apartment	0 # Not a studio apartment

For each property, we run predictions using the Random Forest Model. The predictions from the Random Forest model for these two houses are as follows:

Table 5. Predicted Price and Real Price of House1&2

Random Forest	Predicted price	Real Price	MAE	MSE	R^2
Features of House1	$2,141.53	$2,274.00	132.47	17,551.71	0.9178
Features of House2	$1,633.34	$1,600.00	33.34	1111.36	0.9178

In the case of House1, the predicted price is slightly lower than the real price, with a Mean Absolute Error (MAE) of $132.47, indicating a reasonable prediction. The model's R^2 value of

0.9178 suggests that the Random Forest model accounts for a significant portion of the variance in the housing price.

For House2, the prediction is slightly higher than the real price, but the MAE of $33.34 shows that the model is still providing a reasonably accurate estimate. The R^2 score remains the same at 0.9178, further indicating that the model performs consistently well.

Step 3: Expanding Predictions with Array-Like Data

In the case of **House2**, some features, such as the number of beds and sq_feet, can have multiple possible values. This introduces the need to handle **array-like data**. For example, for House2, the number of bedrooms can be either 1 or 2, and the square footage can be either 522 or 620 according to the rental website. To handle these possibilities, we define a function preprocess_unseen_data_with_arrays that generates all possible combinations of these array-like features. This function preprocesses each combination separately, ensuring that every possible scenario is accounted for.

For **House2**, the following combinations of beds and sq_feet are generated:

- **Case 1**: beds = 1, sq_feet = 522
- **Case 2**: beds = 1, sq_feet = 620
- **Case 3**: beds = 2, sq_feet = 522
- **Case 4**: beds = 2, sq_feet = 620

For each of these combinations, the function preprocesses the data and passes it through the models. The predictions from each model for the different cases are shown below:

Table 6. Predicted Price for Various Models

Case	XGBoost Predicted Price	LightGBM Predicted Price	Gradient Boosting Predicted Price	Random Forest Predicted Price
Case1	1698.45	1747.99	1715.81	1611.72
Case2	1766.42	1799.65	1760.4	1778.11
Case3	1914.93	1941.53	1894.47	1989.76
Case4	1974.06	1985.82	1933.3	1987.13

Step 4: Model Comparison

By comparing the predicted prices across different models for each case, we observe the following:

- For Case 1, the predictions from XGBoost and LightGBM are fairly close to each other, but Random Forest predicts a lower price.

- As the number of beds and sq_feet increase (Cases 3 and 4), the predicted prices rise, with Random Forestgenerally predicting slightly higher prices for larger properties.

Therefore, this case study demonstrates the use of machine learning models to predict real estate prices for unseen properties. The preprocessing steps ensure that unseen data is appropriately encoded and scaled to match the models' requirements. By generating predictions for multiple combinations of array-like features, we can capture various scenarios and make more robust predictions. The comparison of model predictions also highlights the strengths and weaknesses of each model, providing valuable insights into how they handle different property characteristics. Through this process, real estate predictions can be improved, allowing for better pricing strategies in the housing market.

7. Limitations and Future Work

7.1 Limitations

1. Lack of Temporal Data:
   The dataset only reflects real rental market price in June 2024, omitting more recent data. This gap between June and December 2024 may lead to reduced accuracy in the model's forecasts, as recent market trends and dynamics are not captured.

2. Unavailability of Macroeconomic Variables:
   The analysis does not incorporate macroeconomic factors such as interest rates, mortgage rate, unemployment rates, or GDP, which are critical in influencing housing markets. Their absence restricts the ability to establish causation between broader economic trends and housing price variations.

3. Impact of External Shocks:
   The dataset reflects trends from the post-pandemic period but does not fully capture the long-term impacts of events like COVID-19. Consequently, the models may lack the ability to accurately project housing prices over extended periods due to the unpredictability of pandemic-related influences.

4. Regional Heterogeneity:
   Differences in regional economic conditions, local policy decisions, and housing supply constraints are not explicitly modeled, which may overlook important drivers of regional price variations.

7.2 Future Work

1. Incorporating Recent Data:
   Extend the dataset to include housing market data up to December 2024 and beyond. Regularly update the dataset to ensure forecasts reflect the latest market trends and dynamics.

2. Integrating Macroeconomic Variables:
   Enhance the analysis by incorporating critical macroeconomic indicators such as interest rates, mortgage rates, unemployment rates, and GDP. This will provide a comprehensive understanding of the interplay between economic factors and housing prices.

3. Scenario-Based Modeling for External Shocks:
   Develop scenario-based models to account for the potential impacts of future external shocks, including pandemics or economic crises. Incorporating historical data from COVID-19 and other global events can improve the robustness of long-term projections.

4. Regional-Level Modeling:
   Create region-specific models that factor in local economic conditions, policy variations, and housing supply dynamics. These tailored models will provide more accurate insights into regional price trends and variations.

5. Time-Series Enhancements:
   As part of future work, advanced time-series techniques such as SARIMA or machine learning-based forecasting models could be explored to capture trends, seasonality, and cyclical behavior in the housing market more effectively. This approach would require detailed historical data for each house to ensure accurate modeling and reliable predictions.

8. Conclusion

The training data showed that the Random Forest model was the best performing in terms of error prediction. However, due to its iterative process of fitting, this is time costly and it is often overfitted. For instance,XGBoost is recognized for its scalability and robust handling of missing values, producing good accuracy at the same time as showing a low-cost computation venue. LightGBM turned out to be the fastest model on account of its technique of a leaf-wise growing algorithm of trees, as well as pretty low consumption of memory.

However,the result presents other models with slightly worse accuracy than these two models, Random Forest and XGBoost. Different prospective promises in their respective way, thus the model application will depend on the accuracy versus computation cost time efficiency trade-off.

The Random Forest will serve in very few applications in most cases as far as it is very accurate. When it comes to huge datasets, XGBoost and LightGBM are the models to be preferred for time efficient real-time forecasts.

Future research could be directed toward these models to understand how they might be applied in a broader landscape, especially in hybrid or ensemble configurations. Some of the interesting future directions could be done in following five parts:

- Hybrid techniques embed a tree-based methodology within the neural network framework.
- High complex models such as Random Forest will be studied to determine some of the other ways they become efficient.
- Develop algorithms that can scale and be interpretable simultaneously in ensemble learning.
- Improve generalization techniques for tree-based methodologies.
- Optimize techniques to reduce time to train complex algorithms.

Overall,the research results describe how data scientists can weigh efficiency against accuracy in predictive modeling and thus capture further highlight points in house price prediction systems in Canada or other countries.

References

[1] Gavrylov, S. (2024). *25,000 Canadian rental housing market (June 2024)* [Data set]. Kaggle. https://www.kaggle.com/datasets/sergiygavrylov/25000-canadian-rental-housing-market-june-2024

[2] Canadian Real Estate Association. (n.d.). *MLS® Home Price Index (HPI) tool*. Canadian Real Estate Association. https://www.crea.ca/housing-market-stats/mls-home-price-index/hpi-tool/

[3] Truong, Q., Nguyen, M., Dang, H., & Mei, B. (2020). Housing price prediction via improved machine learning techniques. *Procedia Computer Science*, *174 (2020) 433–442*. https://doi.org/10.1016/j.procs.2020.06.111

[4] McKinney, W. (2010). Data structures for statistical computing in Python. In *Proceedings of the 9th Python in Science Conference* (pp. 51-56).

[5] Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.

[6] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794).

[7] Abadi, M., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation* (pp. 265-283).

[8] Ke, G., et al. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems* (Vol. 30, pp. 3146-3154).

[9] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90-95.

[10] Apartments.com. (n.d.). *Kamloops Tranquille apartments, Kamloops, BC*. Apartments.com. https://www.apartments.com/kamloops-tranquille-apartments-kamloops-bc/gq98g5z/

[11] Apartments.com. (n.d.). *945 Victoria St W, Kamloops, BC, Unit 101*. Apartments.com. https://www.apartments.com/945-victoria-st-w-kamloops-bc-unit-101/qzlftlx/

[12] OpenAI. (2024). *ChatGPT* (December 10 version) [Large language model]. Retrieved from https://chat.openai.com/