House Price Prediction Based on House Features

Yitong Liu
Sociology, Brown University
GitHub: nicoleliuu/ds1030final

1. Introduction

Machine learning algorithms are increasingly used in the real estate industry to help potential homeowners make informed decisions about their ideal houses. This project uses Melbourn Housing Snapshot, sourced from Kaggle, to predict house sale prices in Melbourne, Australia, based on household features. [1]

The database has 13,580 data points, each including housing prices in Australian dollars and up to 15 features around the house's architectural features and location. To protect privacy, I exclude features that concern exact addresses or human names.

I could not find peer-reviewed research using this database. However, a few open-source projects use this database to predict house sale prices, yielding R2 scores of roughly 0.8. [2][3]

2. Exploratory Data Analysis

I use EDA techniques to better understand the dataset's target variables and features. As Figures 1 and 2 show, the target variable - house prices - is a right-skewed distribution. The average house price is 1.075 million Australian dollars. Most houses are priced within 2 million dollars, with the 95% percentile equaling 2.29 million dollars and the 99% percentile equaling 3.34 million dollars.

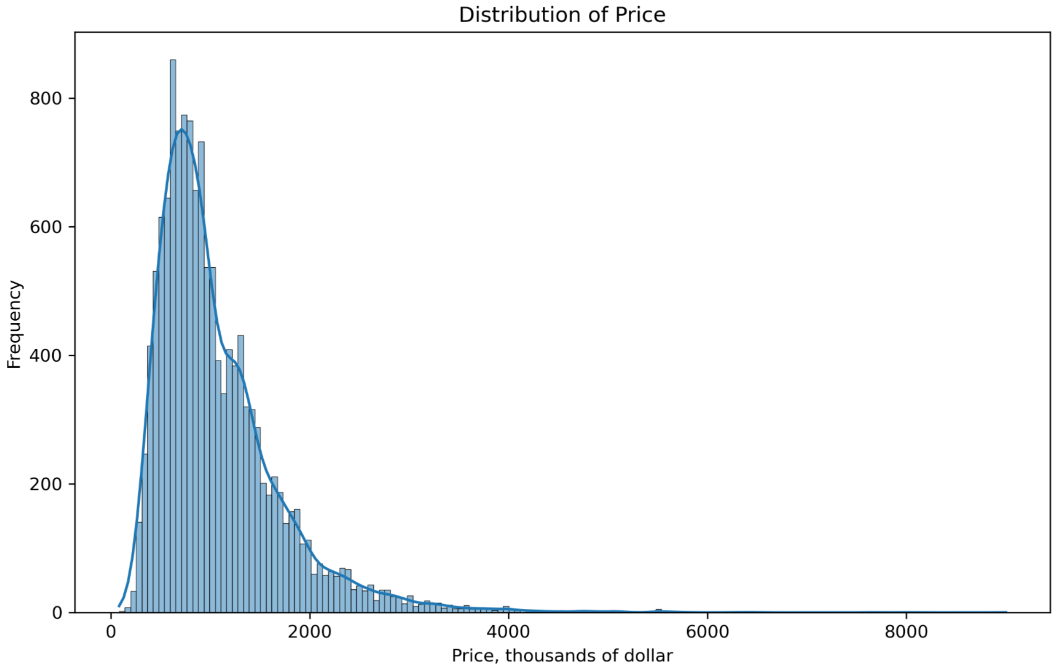

Fig.1 Distribution of house sale prices, which is right-skewed distributed.

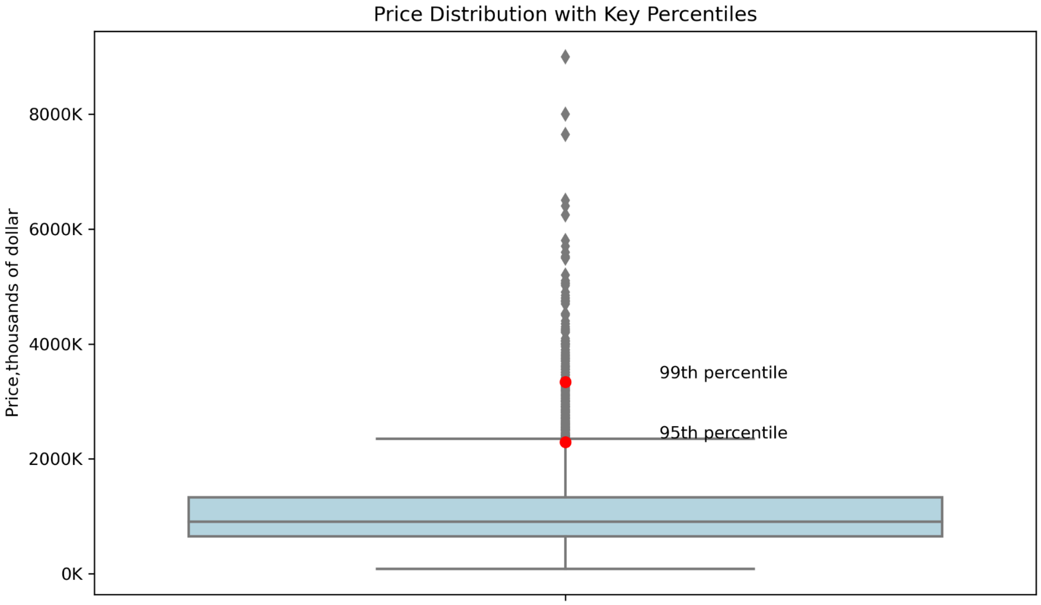

Fig.2 Price distribution box plot showing 99% percent houses within 4 million dollars.

The group-structure feature, suburb, also reveals essential information about Melbourne's real estate market. As shown in Figure 3 and 4, Kooyong, Canterbury, and Middle Park are the top three most expensive neighborhoods with average house sale prices beyond 3 million

Australian dollars. Unexpectedly, there is no overlap between the top ten most expensive neighborhoods and the top ten neighborhoods with the highest number of transactions. This suggests that the trending neighborhoods in Melbourne are those with lower-ended, more affordable houses.

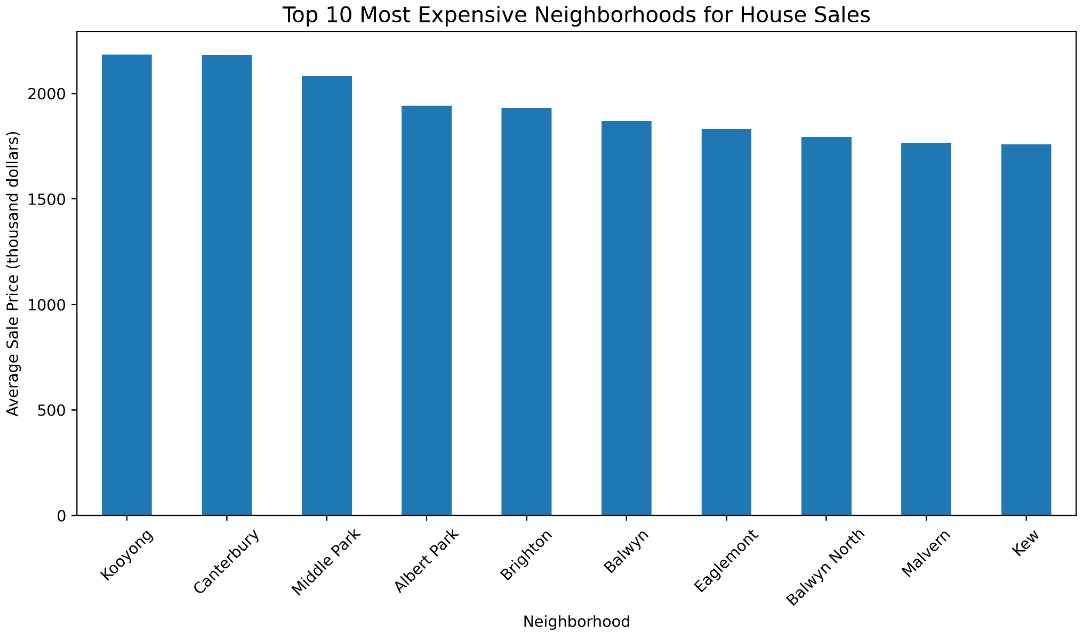

Fig 3. Bar plot of average house sale price by neighborhood, which displays the top 10 most expensive neighborhoods in Melbourne.

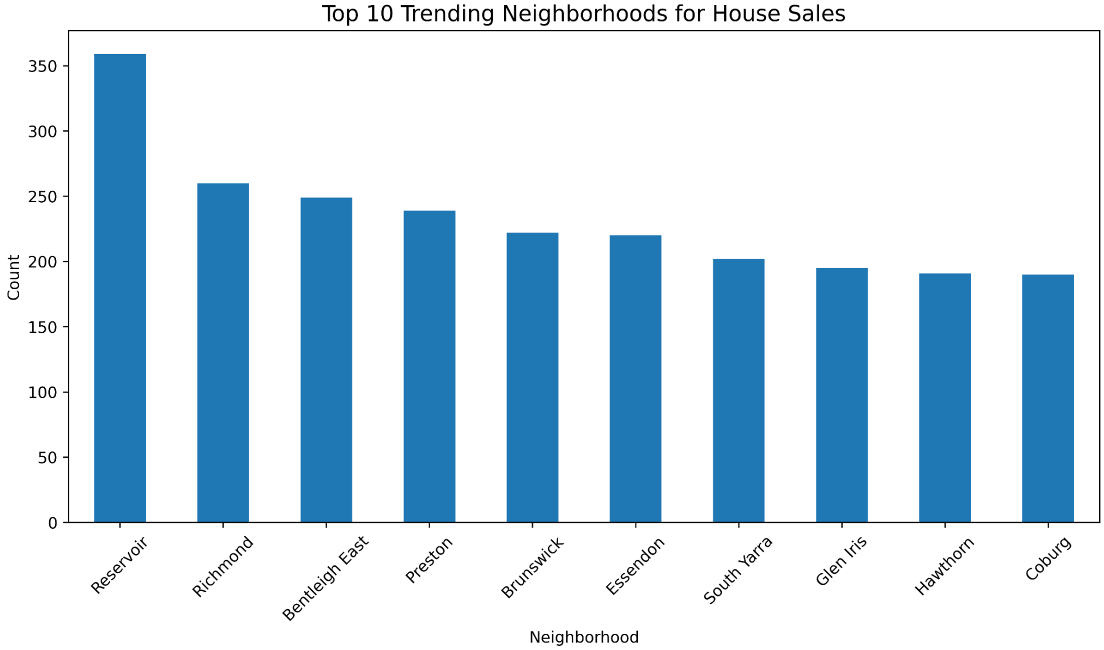

Fig 4. Bar plot of house transaction number by neighborhood, which lists 10 neighborhoods with the highest number of house sales.

Probing the relationships between several house features also reveals much interesting information. For example, figure 5 shows that once moving away from the city center, house prices are lower on average and with lesser variation.

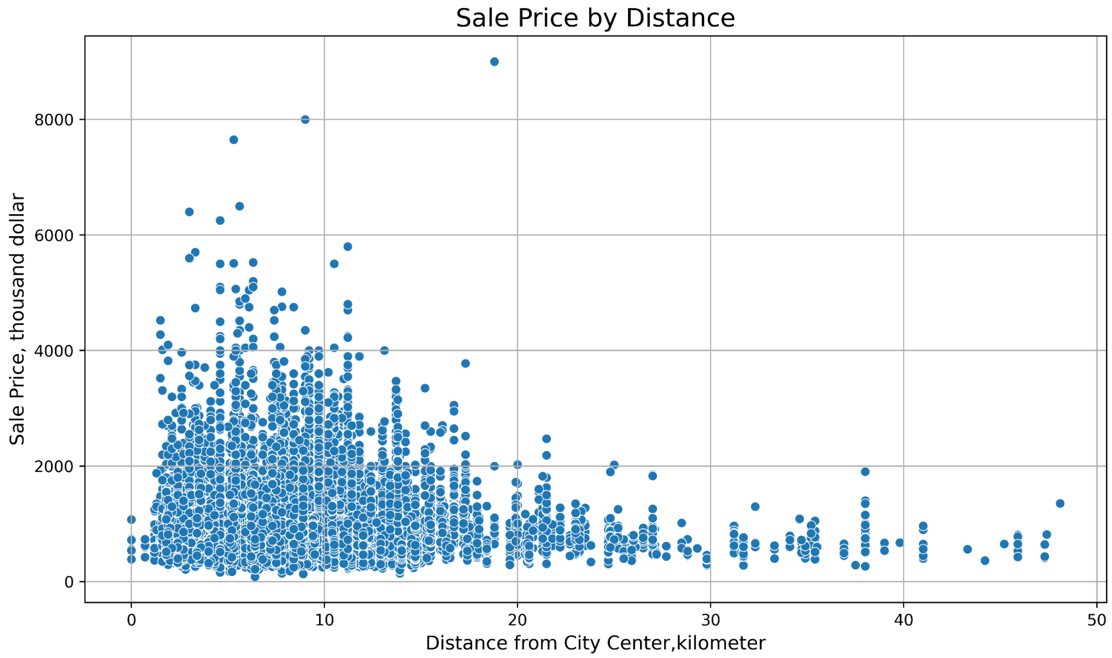

Fig 5. Scatter plot showing the relationship between distance to city center and house sale price.

Lastly, I calculate the correlation among features using Pearson Correlation for numerical features, Cramér's V for categorical Features, and ANOVA for categorical-numerical feature combinations. Numerical features display low correlation. However, I find strong correlations (with Cramér's V higher than 0.95) among some categorical features: suburb, post code, council area, and region name. This is expected since they all describe house location from the respective angles of geographical division, urban planning, and administrative division. With this in mind, I keep these features for preprocessing and machine learning model fitting, which results in several convergence warnings and poor model performance. Given this, I drop the highly correlated variables and only keep three categorical features for machine learning: suburb, house type, and sale method.

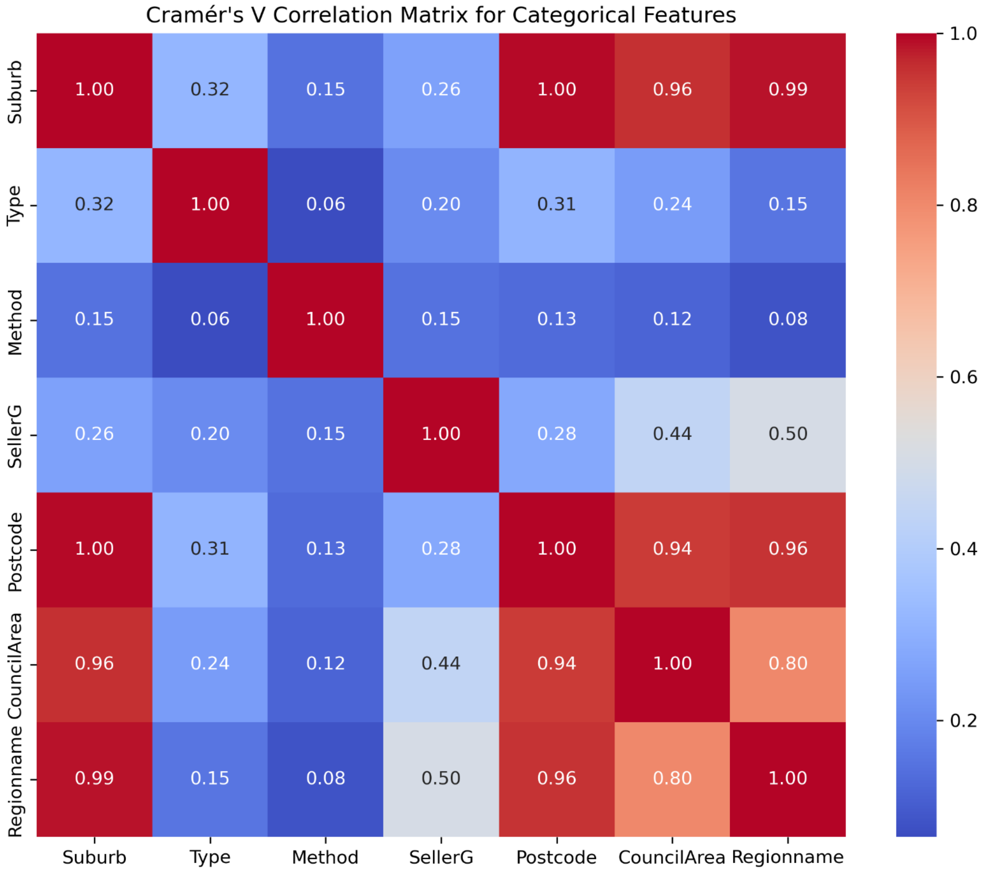
Cramér's V Correlation Matrix for Categorical Features

Fig 6. Cramér's V Matrix, which measures correlation among categorical features.

Methods

For preprocessing, I use the one-hot encoder to transform categorical variables and address the issue of missing data. I apply the standard scaler on numerical variables. I tried simple imputation, multivariate imputer, and XGBoost for numerical features with missing values. Judging by overall model performance scores, multivariate imputation performs relatively better than simple imputation at the cost of making models computationally more intensive. XGBoost balances complexity with applicability the best. After going through the preprocessing pipeline, there are 205 columns in the database.

For splitting strategies, because the data is group structured, preventing data leakage is crucial. As the machine learning flowchart shows (figure 6), I first use group shuffle split to split data between other and test. Then, I use the group k-fold method to split the other data into four folds of data for training and validation. The ratio between train, validation, and test is 6:2:2.

In Grid Search CV, I use the train data to fit the selected machine learning model, then use validation data to compare model performance across different parameter combinations. Eventually, the Grid Search CV process produces a set of optimal parameters for the machine learning model. I use test data to evaluate each machine-learning model's performance under the optimal combination of parameters and compare test scores across machine learning models to find the best machine learning model for house price prediction.

Given that this is a regression problem and the size of the target variable is large, I choose root mean square error (RMSE) as the evaluation strategy. The baseline performance is 639,287.2 dollars.

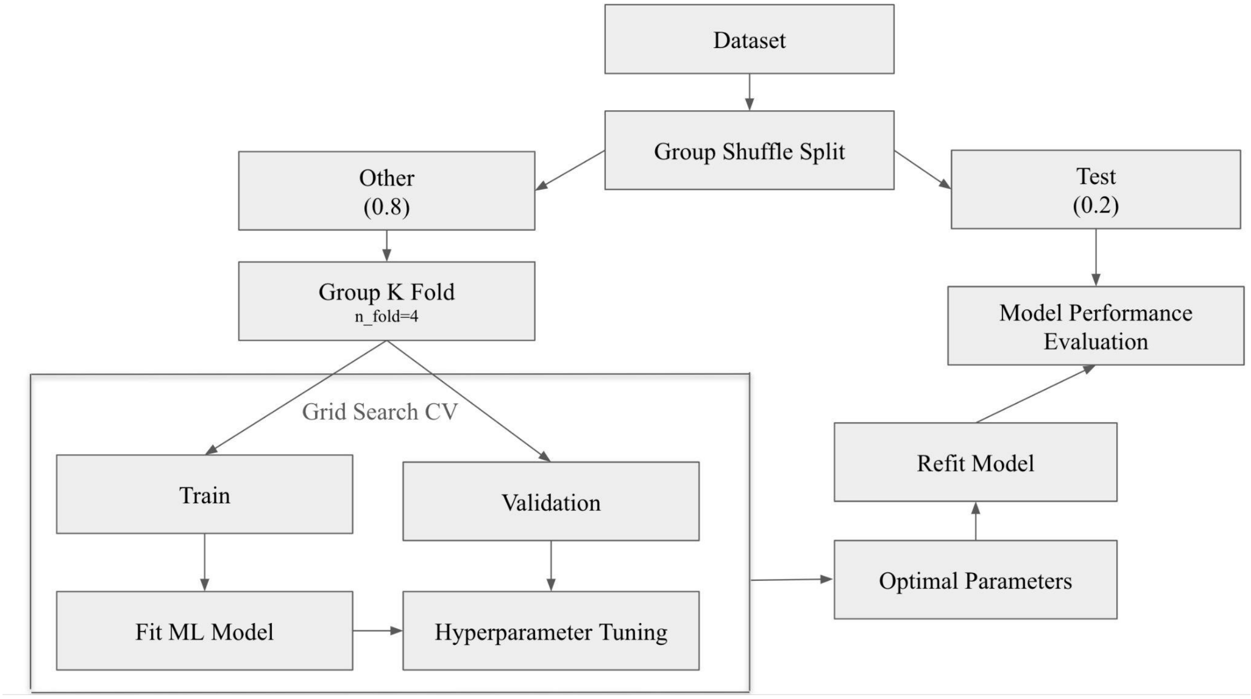

Fig 6. Flow chart of the model development and cross validation pipeline.

I fit four machine learning models: XGBoost, Random Forest, Elastic-Net Regression, and Lasso Regression. Graph 1 displays the parameters I tuned. To measure the uncertainties of my evaluation metric, I calculate model performance scores under four random states (random state=38, 42, 55, 64).

Model	Parameters 1	Parameters 2
XGBoost	learning rate: [0.03, 0.05]	max depth: [1, 3, 10, 30, 100]
Lasso Regression	alpha: [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 500, 1000, 1500, 2000]	
Elastic-Net Regression	alpha: [0.1, 1, 10, 100, 500, 1000, 1500, 2000]	L1 ratio: [0.01,0.05,0.1, 0.3, 0.5, 0.8]
Random Forest	n estimators: [10, 100, 500, 1000, 1500]	max depth: [50, 100, 150, 500, 1000, 1500, 2000]

Graph 1. Model Selection and Parameter Tuning

Results

As Figure 7 shows, all four machine learning models outperformed the baseline score (RMSE=639,287.2). Among the models, the XGBoost model has the best performance score with RMSE = 409,118.2. The Random Forest model has the lowest standard deviation (SD=22261.4), but the XGBoost model's standard deviation is also relatively small (SD=14523.8). Given these considerations, I choose XG Boost as my best performance model for further interpretation.

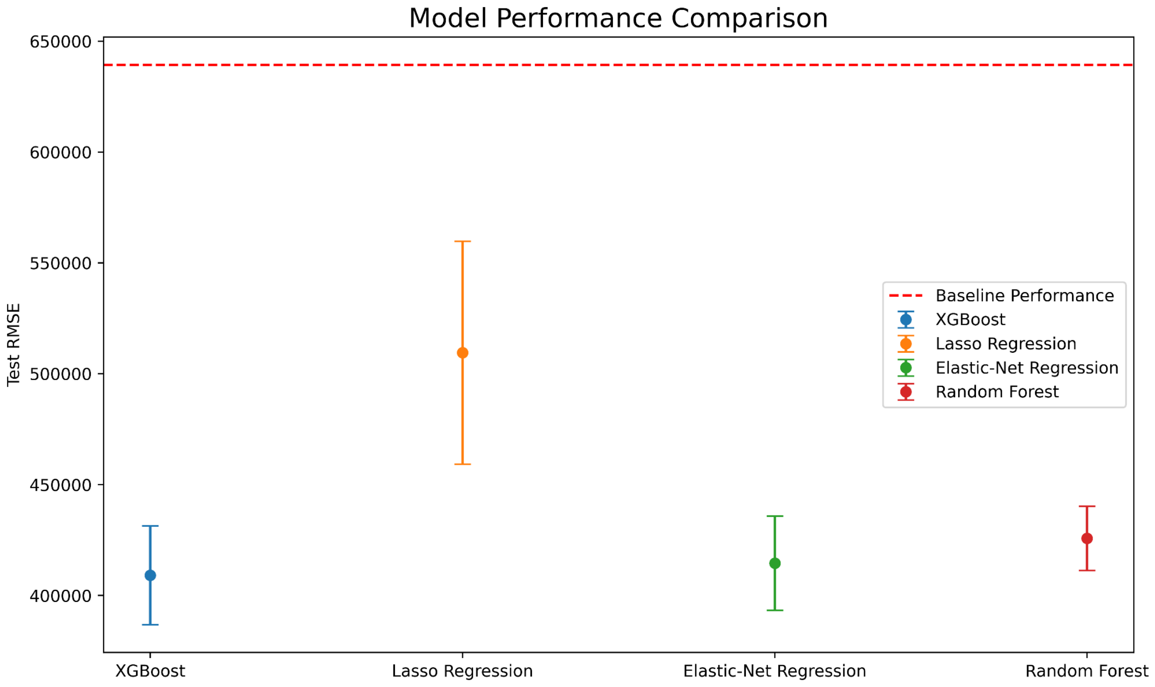

Fig 7. Error bar of machine learning model performance. The X-axis displays model names and the Y-axis shows RMSE scores.

Comparing the XGBoost model's predictions with the true values, I find the model performs fairly well for most data points, as shown in Figure 8. However, it is worth noting that the model's performance decreases when house sale prices go beyond 3 million dollars. That means the model make better predictions for lower-end and middle-end houses than high-end houses.

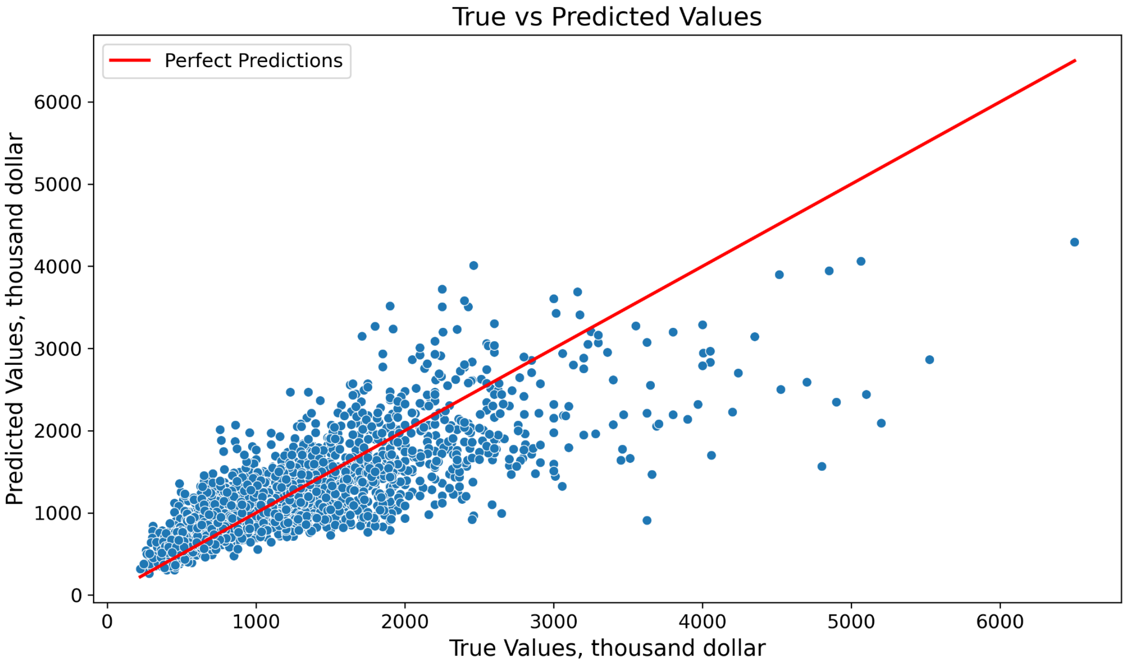

Fig 8. Scatter plot of true and predicted house sale prices. The X-Axis is the true value in thousand dollars, and the Y-Axis is the XGBoost model's predicted value, in thousand dollars.

To understand features' global importance, I try several global importance features, and display three metrics for illustration (figure 9 -11). Across metrics, land size, distance to city center, and number of rooms are the consistently important features.

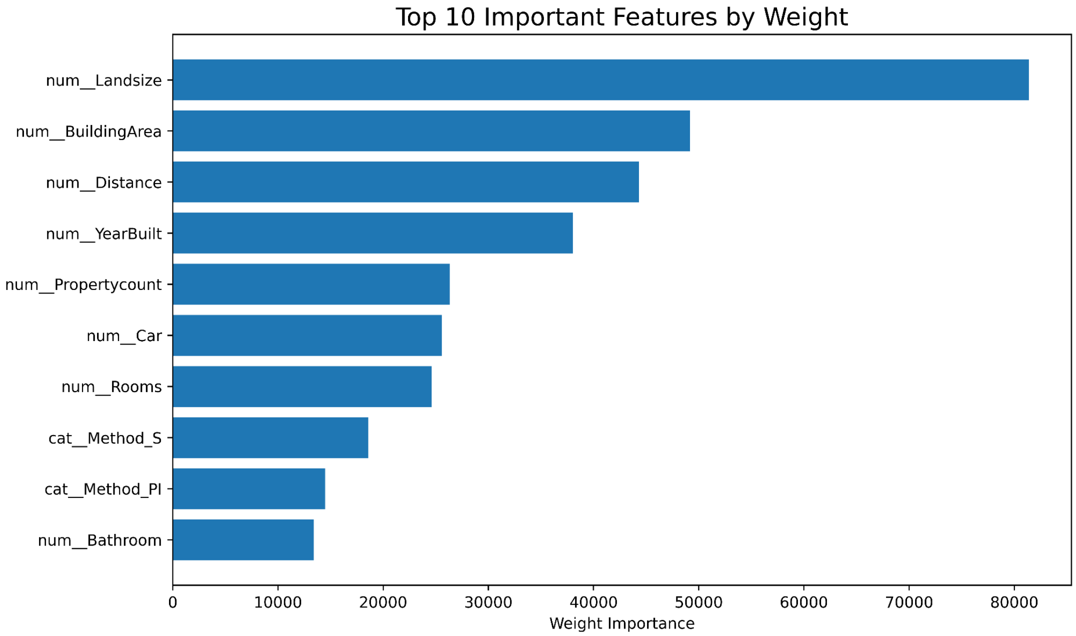

Fig 9. Top 10 important features by weight importance.

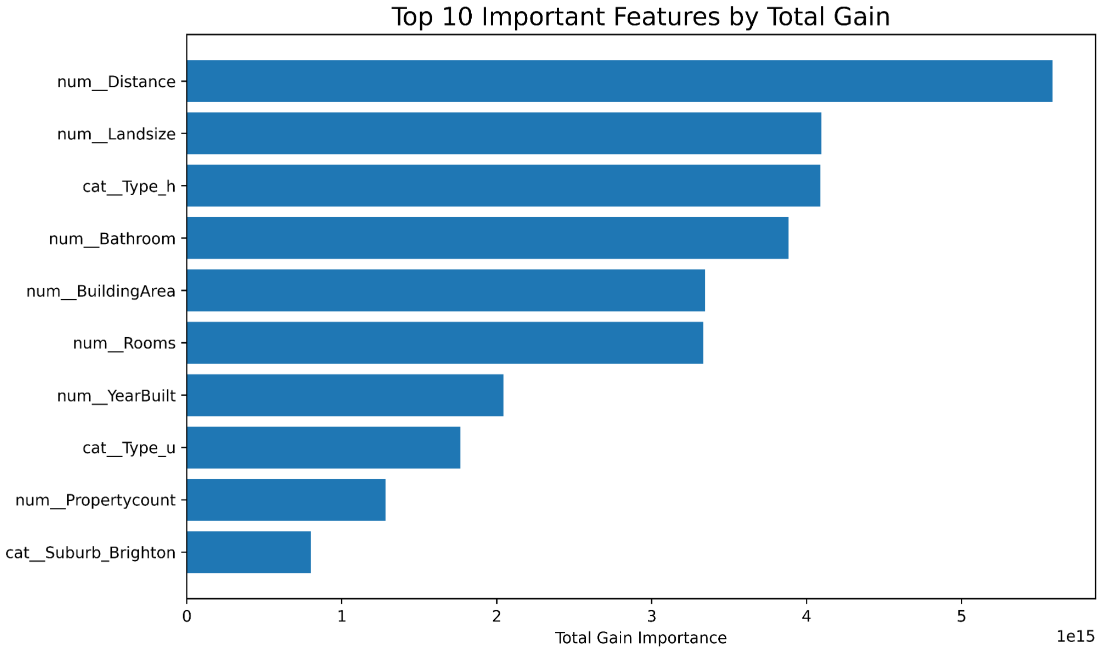

Fig 10. Top 10 important features by total gain importance.

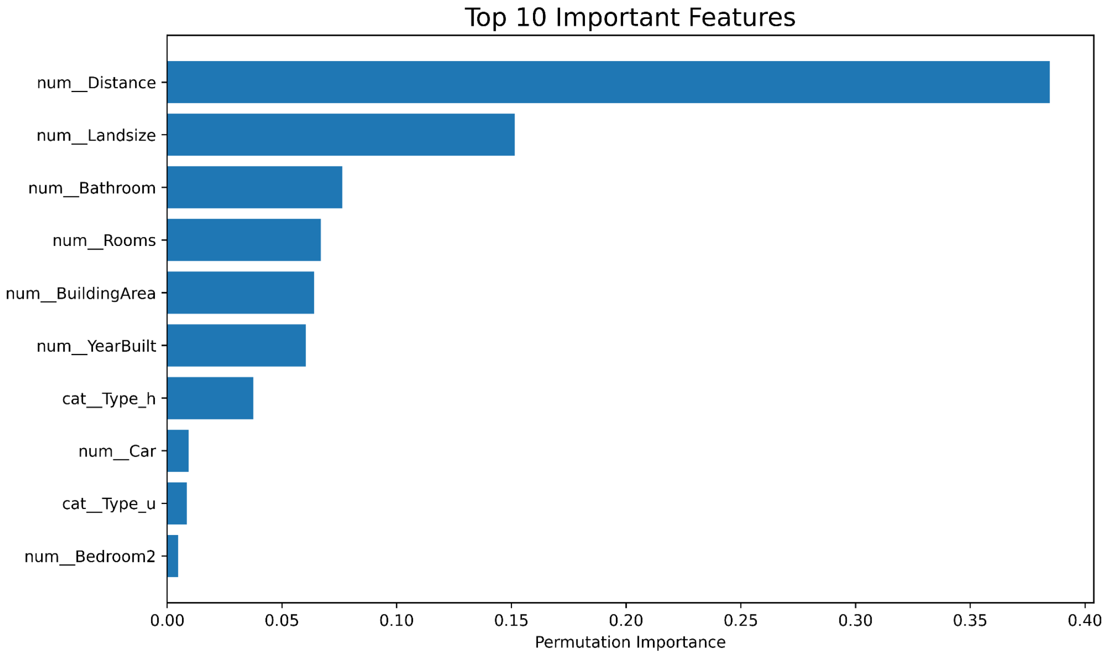

Fig 11. Top 10 important features by permutation importance.

To understand features' local importance, I select three data points (index=3, 100, 200) and illustrate their global feature importance using SHAP force plots, as shown in figure 12-14.

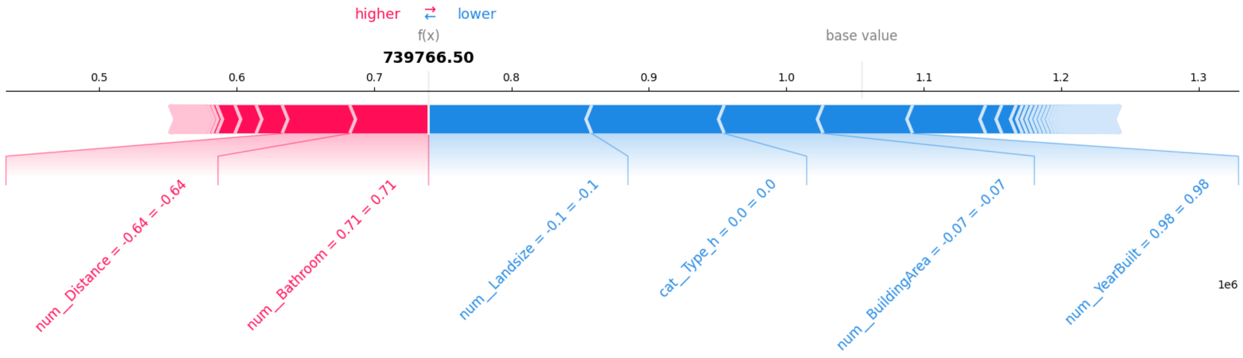

Fig 12. SHAP force plot for data point 3. Land size, house type, building area, and number of bathroom are among the most important features that contribute to this house's price prediction.

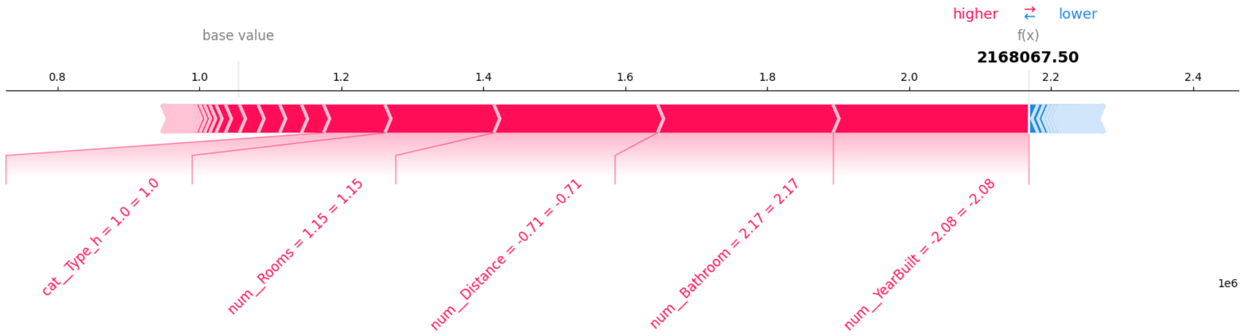

Fig 13. SHAP force plot for data point 100. Year built, number of bathroom, distance are among the most important features that contribute to this house's price prediction.

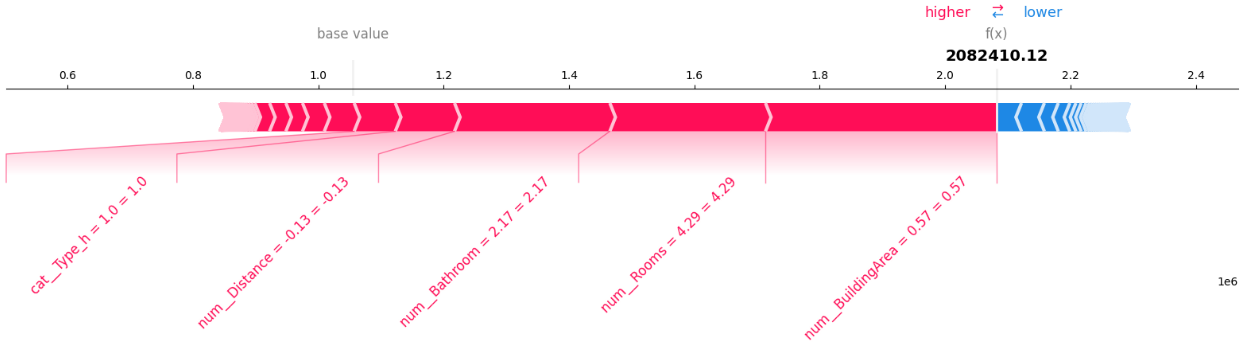

Fig 14. SHAP force plot for data point 200. Building area, number of rooms and bathrooms, and distance are among the most important features that contribute to this house's price prediction.

Outcomes

Throughout this project, I form a deeper understanding of how to predict regression problems using real-life datasets under the challenges of group-structured data and missing value. Context-wise, I also know more about the landscape and particularities of Melbourne's real estate market. If I have more time to work on this project, I will follow the following list of items to improve it:

- Collect more data on houses beyond three million dollar sale price and use Stratified Group KFold in data splitting to increase models' prediction for the higher-end houses;
- Tune more parameters with caution of overfitting;
- Try more machine learning models, such as K-Nearest Neighbors.

Referencing

[1] Melbourne Housing Snapshot:
https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot

[2]Project using Melbourne Housing Snapshot:
https://www.kaggle.com/code/vikasukani/melbourne-house-price-predic-using-ml-model#HYPER-PARAMS-TUNNING

[3]Project using Melbourne Housing Snapshot:
https://www.kaggle.com/code/atasaygin/melbourne-housing-randomforestregression-and-eda#The-Model-Scores

Github Repository:

https://github.com/nicoleliuu/ds1030final