Building prediction models on house prices in Budapest

Anton Shestakov

January 2026

Data

Detailed data pre-processing and cleaning: https://github.com/Anton21a/DA3_Machine_Learning_2026

Preparation

Before starting there is a disclaimer that all following modifications and other data shenanigans were applied both for 2025q1 and 2025q2 periods for Budapest as well as for Prague (which is later used for cross-city validation). There are insignificant distinctions in data structure between cities that eventually led to a bit different way of data wrangling and extracting variables. However, the structure of data for one city within periods is mostly similar.

In this part, the data preparation for Budapest is presented. The rest of the data pre-processing (including cleaning) for Prague is on GitHub which url I placed above.

The first step is devoted to defining and encoding key categories inside property type and room type variables. After checking the number of observations in aggregated dataset, it was decided to keep 3 the most popular categories for the former, and 2 categories - for the latter. Table 1 demonstrates results for property types in Budapest for 2025q1.

Original ‘property_type‘	**Recoded Category**	**Observations**
Entire rental unit	Rental Unit	6268
Entire condo	Condo	3232
Private room in rental unit	Private Room	375

Table 1: Encoded values of property type for Prague in 2025q1

Another technical side relates to type variables transformation from numerical to categorical, from character to factor and etc. The code also generates new numerical variables, including a logarithmic transformed price, log and squared transforms of accommodates, beds, and number of reviews. In addition, a new bandwidth for price variable is set to exclude extreme values.

Some variables contain a high ratio of missing values, so they are excluded from the data. The low percentage of missingness in numeric variables is filled by imputation the medians. For such variables as number of bathrooms or minimum nights (before values imputation) additional categorical variable is created. It includes a separate category 'NA' to account for artificial filling the data. Table 2 shows descriptive statistics for some numeric variables for Budapest in 2025q1:

Variable	Mean	SD	P25	Median	P75
`ln_price`	9.878	0.53	9.541	9.810	10.156
`ln_accommodates`	1.262	0.51	0.693	1.386	1.609
`beds_n`	2.154	1.68	1.000	2.000	3.000
`bathrooms_n`	1.232	0.38	1.000	1.000	1.500
`p_host_response_rate`	90.65	23.5	100.0	100.0	100.0
`kitchen`	0.935	0.247	1.000	1.000	1.000
`number_of_reviews_n`	84.58	119.0	7.000	37.00	112.0

Table 2: Descriptive statistics of some numeric variables

Cleaning

In this part of the chapter, data was cleaned by dropping broken lines, adjusting values writing in cells, and wrangling text data in the variable amenities.

Variables price and rates (whether response or acceptance) involves sign writings in form of ($) and (%), respectively. These signs were removed. Particular attention has been paid to variable amenities with a long list of different things. After preliminary cleaning there has been around 1900 binary columns (obviously some of them overlapping with tiny distinctions, i.e uppercase or lowercase). Eventually, it was decided to preserve only top 20 the most frequent amenities for their following embedding in the models. The list of amenities is placed in Table 3 below.

Top 20 amenities	
Wi-Fi	Hair dryer
Hot water	Dishes and silverware
Kitchen	Iron
Cooking basics	Hangers
Refrigerator	Bed linens
Essentials	Microwave
Hot water kettle	Smoke alarm
Dedicated workspace	Self check-in
Carbon monoxide alarm	Lockbox
Room-darkening shades	Wine glasses

Table 3: List of common amenities in Airbnb listings

Ultimately, the work has the following datasets:

- **Budapest 2025 Q1**: 9,875 observations and 92 variables.
- **Budapest 2025 Q2**: 9,262 observations and 92 variables.
- **Prague 2025 Q2**: 7,321 observations and 92 variables.

OLS

Predictions with OLS include staggered running of multiple specifications with different sets of predictors. To ensure more accurate evaluation of models' quality, 5-fold cross-validation is used throughout the analysis. The following models were built to train the data with further validation based on the control data.

$$\text{Model 1:} \quad \ln(\text{price}) = \beta_0 + \beta_1 \cdot \text{property\_type} + \varepsilon$$

$$\begin{aligned}\text{Model 2:} \quad \ln(\text{price}) = {} & \beta_0 + \beta_1 \cdot \text{property\_type} + \beta_2 \cdot \text{f\_room\_type} + \beta_3 \cdot \ln(\text{accommodates}) \\ & + \beta_4 \cdot \ln(\text{accommodates})^2 + \beta_5 \cdot \ln(\text{number\_of\_reviews}) + \beta_6 \cdot \text{f\_bathroom} \\ & + \beta_7 \cdot \text{f\_minimum\_nights} + \varepsilon\end{aligned}$$

$$\begin{aligned}\text{Model 3:} \quad \ln(\text{price}) = {} & \beta_0 + \beta_1 \cdot \text{property\_type} + \beta_2 \cdot \text{f\_room\_type} + \beta_3 \cdot \ln(\text{accommodates}) \\ & + \beta_4 \cdot \ln(\text{accommodates})^2 + \beta_5 \cdot \ln(\text{number\_of\_reviews}) + \beta_6 \cdot \text{f\_bathroom} \\ & + \beta_7 \cdot \text{beds\_n} + \beta_8 \cdot \text{f\_minimum\_nights} + \beta_9 \cdot \text{refrigerator} + \beta_{10} \cdot \text{microwave} \\ & + \beta_{11} \cdot \text{wifi} + \beta_{12} \cdot \text{smoke.alarm} + \beta_{13} \cdot \text{heating} + \beta_{14} \cdot \text{hot.water} + \beta_{15} \cdot \text{essentials} \\ & + \beta_{16} \cdot \text{dining.table} + \beta_{17} \cdot \text{bed.linens} + \varepsilon\end{aligned}$$

Expectedly, the third model performs the best value of RMSE across folds for 2025q1 period. The first model demonstrates a clear deviation due to the lack of predictors. Its RMSE is fluctuated around the values of 0.47 across all folds, while specifications with predictors show improved RMSE results with mean value of 0.40 or below [Figure 1].

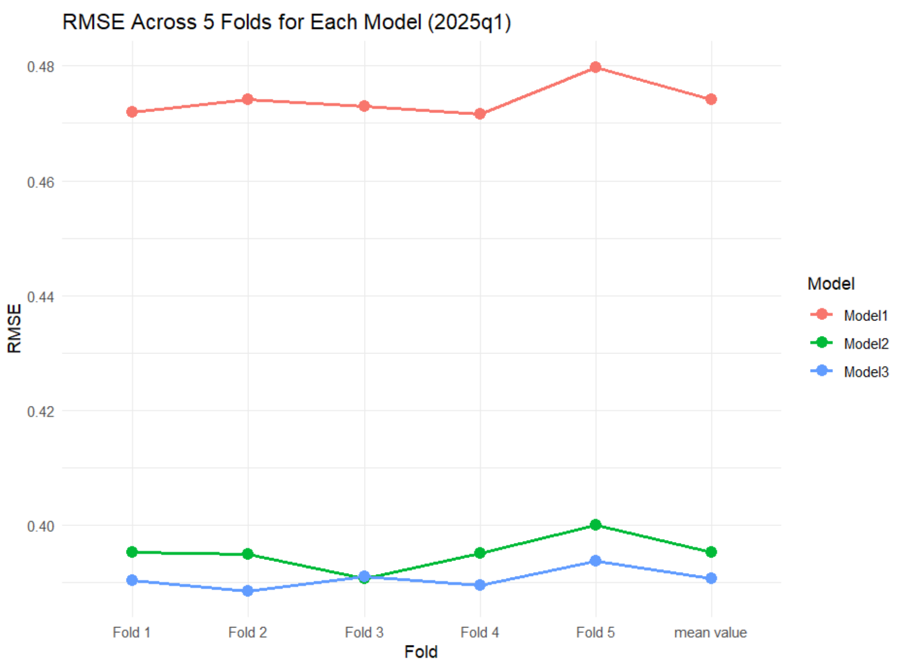

Figure 1: RMSE across folds in OLS

LASSO

The comparison between OLS models and LASSO regression across 5-fold cross-validation 2025q1 demonstrates the superior predictive performance of the latter method. The distinctive feature of this model compared to OLS comes down to embedding interactions between type of property or room and other categorical predictors. LASSO achieves lower RMSE values on average than OLS [Figure 2]. This improvement is notable for 4 out of 5 folds in each year, but especially in 3 and 4 folds in 2025 where the performance gap between LASSO and the best OLS specification widens.

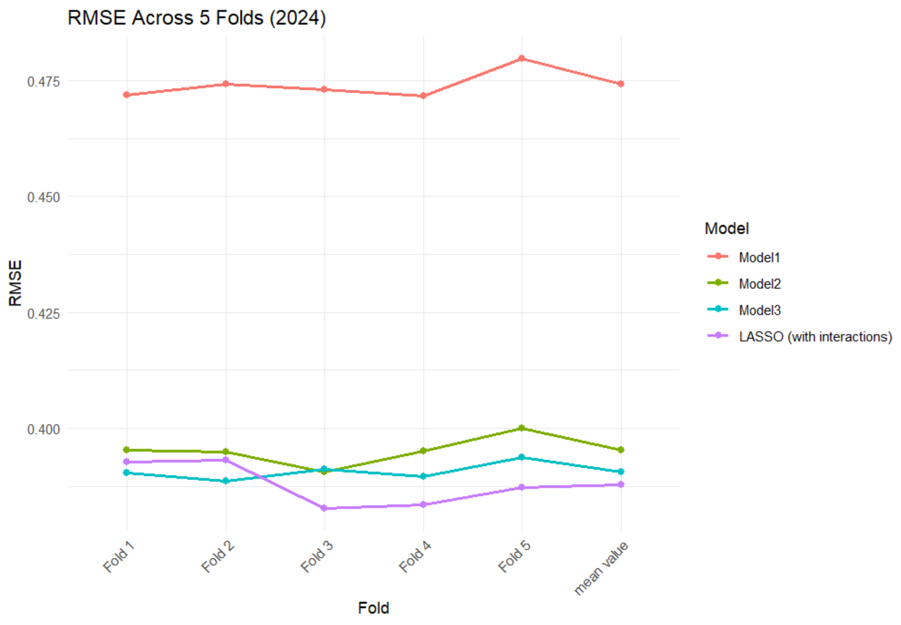

Figure 2: RMSE across folds in OLS and LASSO

Random Forest, GBMboosting, and XGBoosting

In the course of the analysis, additional methods with 5-cv were used for more precise predictions. In particular, the code involves running random forest algorithm and boosting algorithm with GBM and XGB methods. The random forest regression model was implemented using the ranger engine via the 'caret' package. To optimize model performance, a grid search was conducted over key hyperparameters, including the number of variables randomly selected at each split (mtry), which was tested across values 3, 5, 7, and 9. In total, the model used 500 trees and fixed parameters such as a min.node.size of 5 and the variance split rule.

The complementary step of the analysis was extracting the top 10 most important predictors based on impurity reduction. This approach allows for both model tuning and interpreting with further contribution to a more robust understanding of predictive performance [Figure 3].

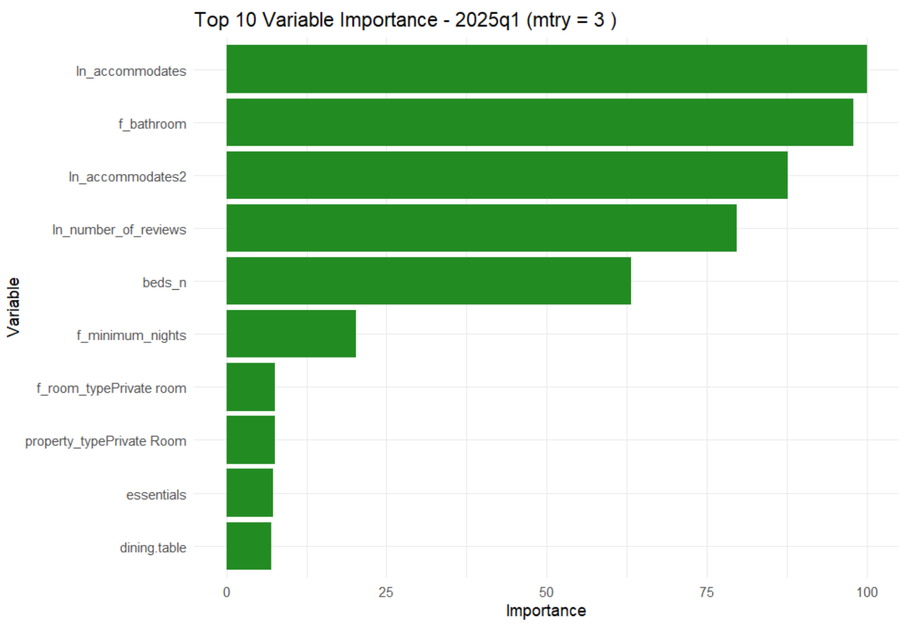

Figure 3: Variable Importance Plot for top 10

The variable importance plots for Budapest in 2025q1 shows that ln_accommodates, f_bathroom, ln_accommodates2, and ln_number_of_reviews take top positions demonstrating the highest predictive impact for the given data and model design.

Analogically with xgbBoosting, a grid search was performed over key hyperparameters including learning rate in the range between 0.03 and 1, maximum tree depth (range of 4 and 6), number of boosting rounds (between 200 and 300), and regularization terms. Overall, hyperparameters indicate that the algorithm tries many options without inferring results based on overly simple analyses. However, along with the dataset size, it doesn't over-complicate the prediction by using too complex values of hyperparameters. At the same extent, there top predictors which are shown on the horizontal bar graph [Figure 4].

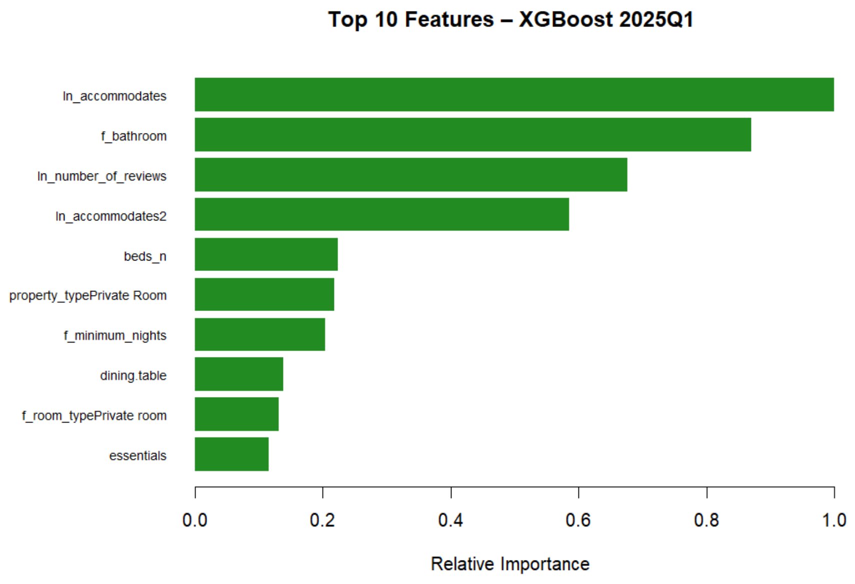

Figure 4: Variable Importance Plot for top 10

The same idea of optimal hyperparameters usage was applied for boosting algo with GBM method. Following the running of all algorithms, a horse race graph is generated to visually compare their predictive accuracy based on RMSE values within 5-fold cross validation.

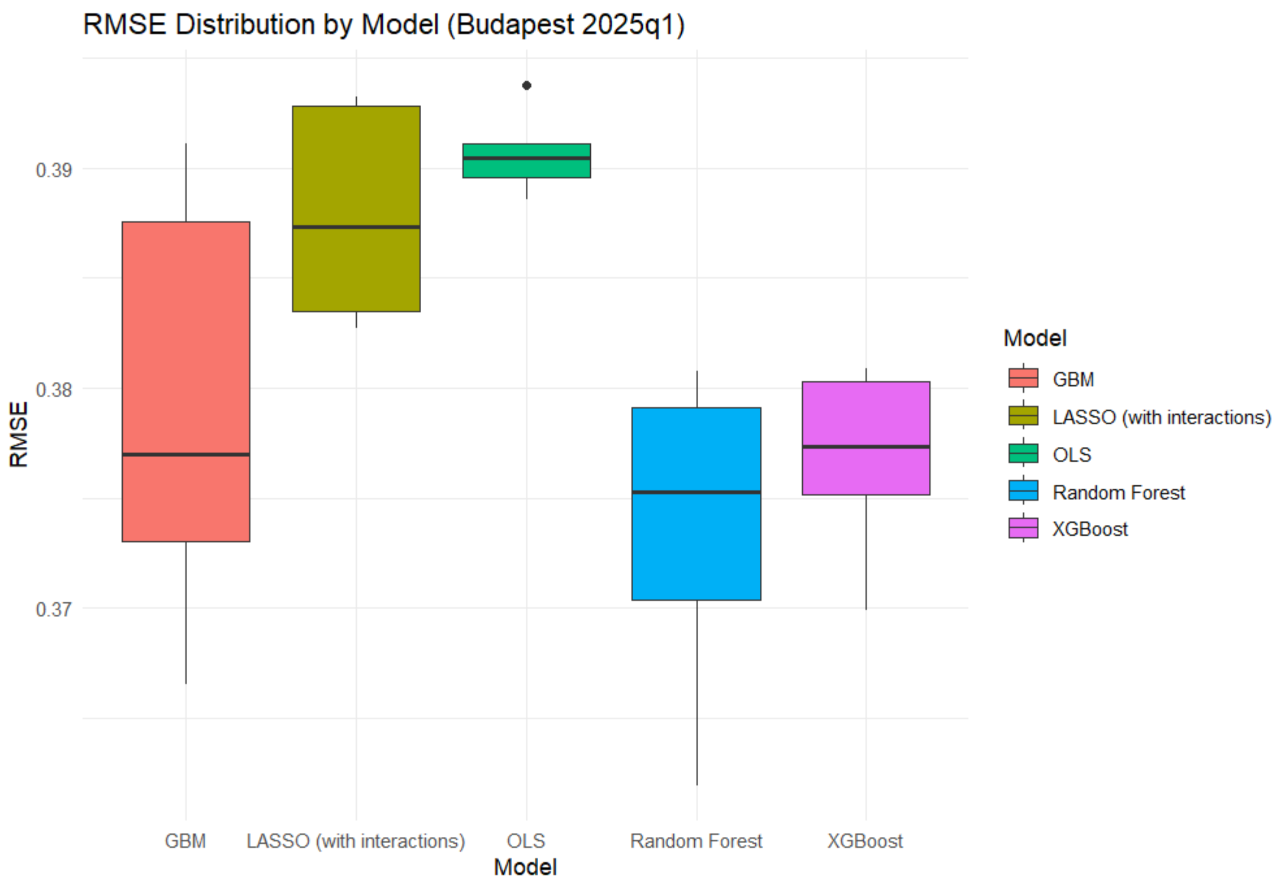

Figure 5:

In the horse graph [Fig. 5] with RMSE values within 5 different methods for prediction on house pricing in Budapest for 2025q1, it can be observed that on average Random Forest outperforms other methods. Even though its results show relatively high variability, it performs better mean result that boosting algo with xgb method. GBM and LASSO also perform competitively, with RMSE values slightly higher than RF and xgb boosting algorithms but more stable than OLS. The latter shows the weakest performance, with the highest RMSE and the widest spread, indicating relatively lower predictive accuracy and greater sensitivity to data variation.

Robustness check using live data for 2025Q2 in Budapest

The horse race graph is based on models trained on Budapest data from 2025Q1 and evaluated using live data from 2025Q2, providing an out-of-sample assessment of predictive performance. It amplifies the inference regarding the ranking of models' strength, highlighting the superior performance of ensemble models, particularly Random Forest and XGBoos [Fig. 6]. The latter two achieves the lowest RMSEs among all models with lower variability across folds compared with OLS and LASSO. GBM also performs strongly, with slightly higher RMSE but tight dispersion, indicating stable performance. In contrast, OLS again shows the highest RMSE, suggesting weaker and less reliable predictions. LASSO performs better than OLS but was outperformed by all tree-based methods. Overall, the results with using live data for the second quarter confirm that advanced boosting and bagging techniques consistently outperform linear models in predictive accuracy for prices on housing in Budapest.

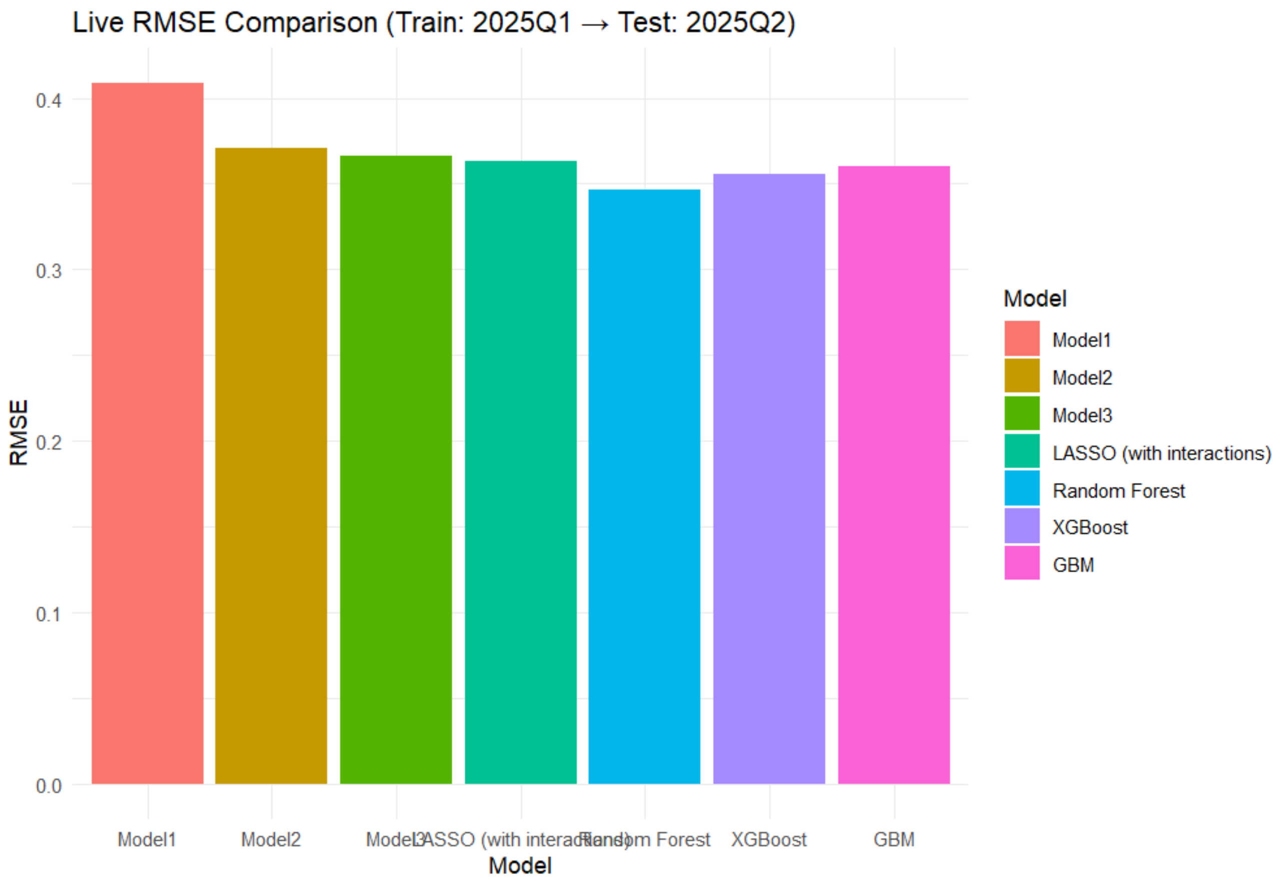

Figure 6:

Robustness check using Prague

As a part of robustness checking, the models were run using price data on housing in Prague for 2025q2. Given that both cities are located almost in neighboring countries and share similar touristic dynamics, the inclusion of Prague contributes to additional models' validity check.

By applying the same modeling procedures to a different but comparable cities, the analysis allows to asses whether the observed patterns of performance keep the same nature beyond the Prague dataset. In particular, an important part is double validation the superiority of tree-based models over simple OLS and LASSO.

In Figure 7, the results are presented for Budapest within all models using live data from Prague dataset for the second quarter 2025. Taking into account that the regression formula hasn't been changed much, it is possible to compare algorithms performances between each other.

As for the period comparison, random forest and XGBboosting algorithms doesn't persist their dominance over other methods within this validation. It is hardly possible to say by looking at the graph whether RF outperforms OLS by mean value of RMSE nor GBM visual comparison with other models. Out of this reason, it was decided to output the table with RMSE results [Table 4]. In turn, it shows that the values of RMSE have significantly increased across all models. The values are within the range of 2.40, while the previous obtained values using live data on Budapest span in interval between 0.35 and 0.40.

On the one hand, this increase is not surprising, given that the models are trained on data from a different city. Despite the close geographical location of Budapest and Prague, the two cities differ in their housing market structures and demand features. These structural differences imply that relationships learned in one market may not transfer seamlessly to another, leading to higher prediction errors in cross-city evaluations.

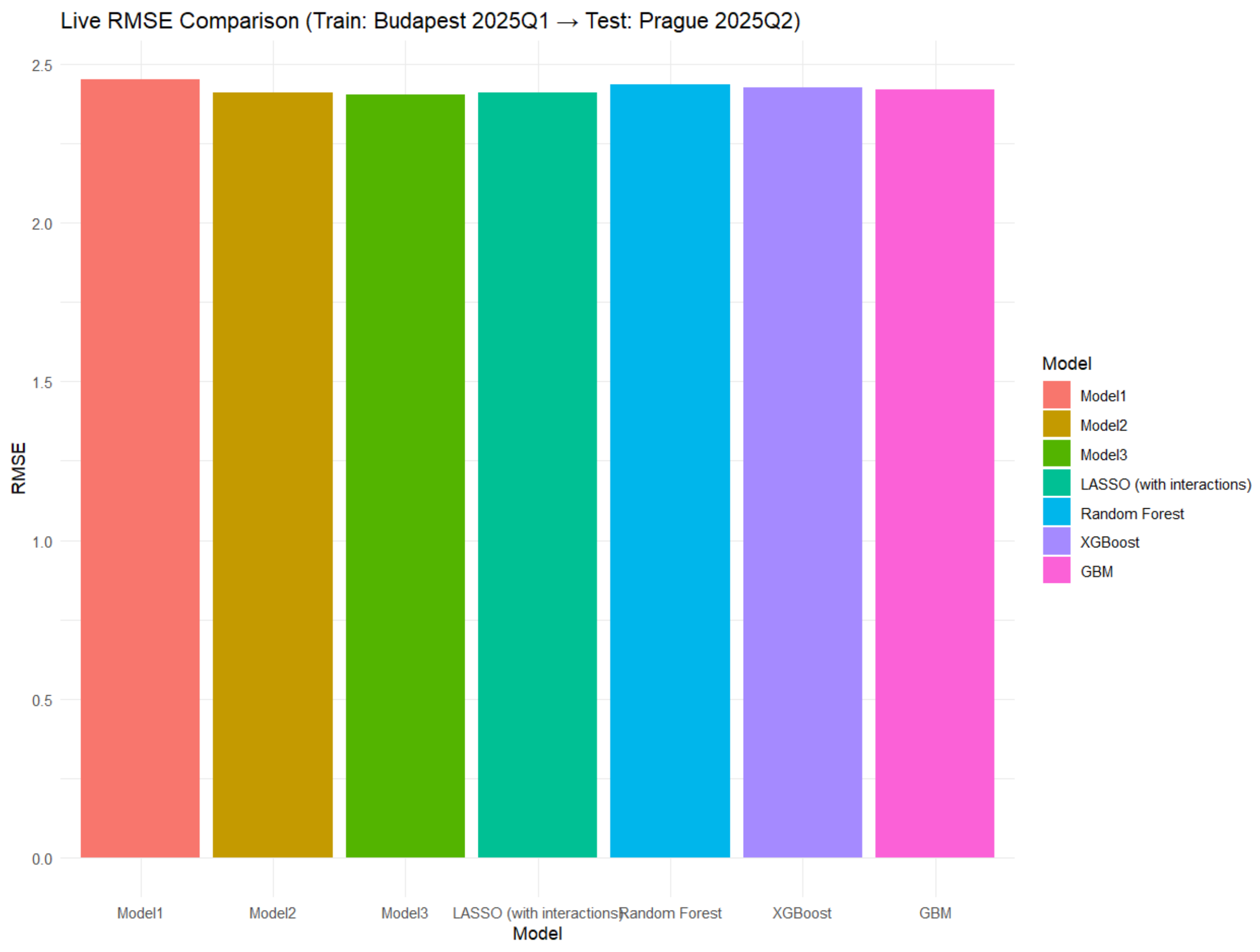

Figure 7:

Model	**RMSE**
Model1	2.451
Model2	2.410
Model3	2.403
LASSO (with interactions)	2.409
Random Forest	2.435
XGBoost	2.425
GBM	2.418

Table 4: RMSE by model (Prague live sample, 2025Q2)

In this setting, all models are trained on the Budapest sample and evaluated on the Prague live data for 2025Q2, which provides a strict test of cross-city generalization. The results indicate that no single model overwhelmingly dominates across specifications. Among the linear approaches, OLS exhibits the weakest performance, giving the highest RMSE, while LASSO with interactions achieves a modest improvement but remains inferior to ensemble methods. Ensemble models demonstrate stronger predictive performance: Random Forest, XGBoost, and GBM all achieve lower RMSE values, with GBM delivering the best overall result in this cross-city evaluation. These findings suggest that flexible, non-linear models generalize better across urban housing markets, while linear models appear limited in capturing structural differences between cities.