Untitled

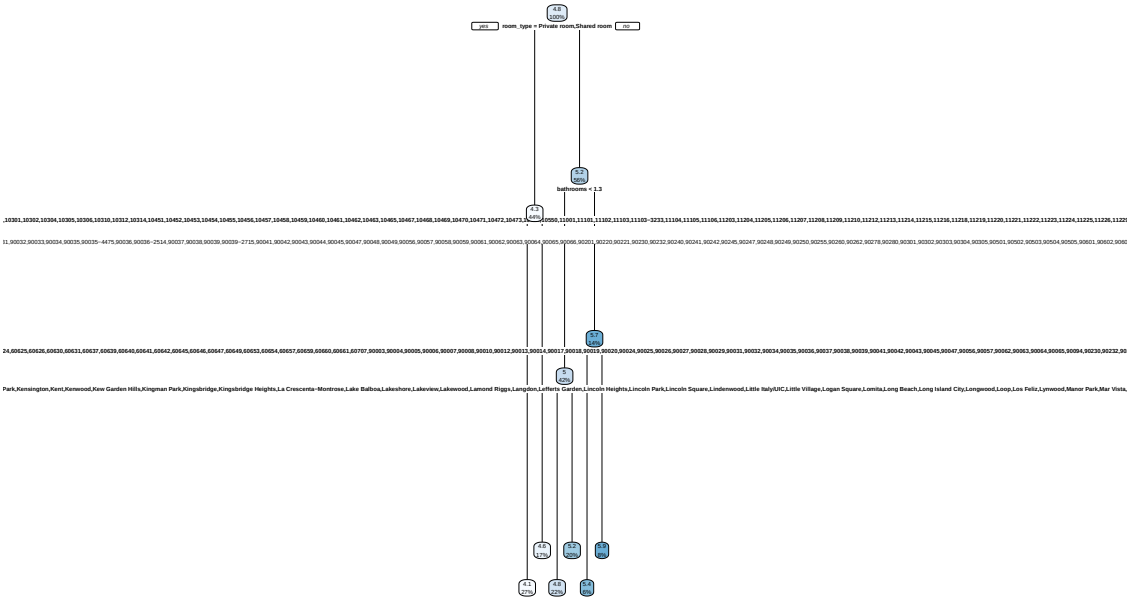
Mengdi Hao

2024-05-19

CART Model

Decision Tree

The graph below illustrates the decision tree structure using all the features in the dataset. The original decision tree has an RMSE of 0.4855 and an R^2 of 0.5469. The close values of in-sample (IS) R^2 and out-of-sample (OOS) R^2 suggest that the decision tree model does not suffer from over-fitting. Both IS and OOS R^2 values are slightly over 50%. Since R^2 indicates the proportion of the variance in the dependent variable that is predictable from the independent variables, an R^2 of 0.55 means that approximately 55% of the variance in the `log_price` can be explained by the model. While this indicates that the model captures a significant portion of the variance, it also suggests that nearly half of the variance is unexplained, indicating room for improvement.



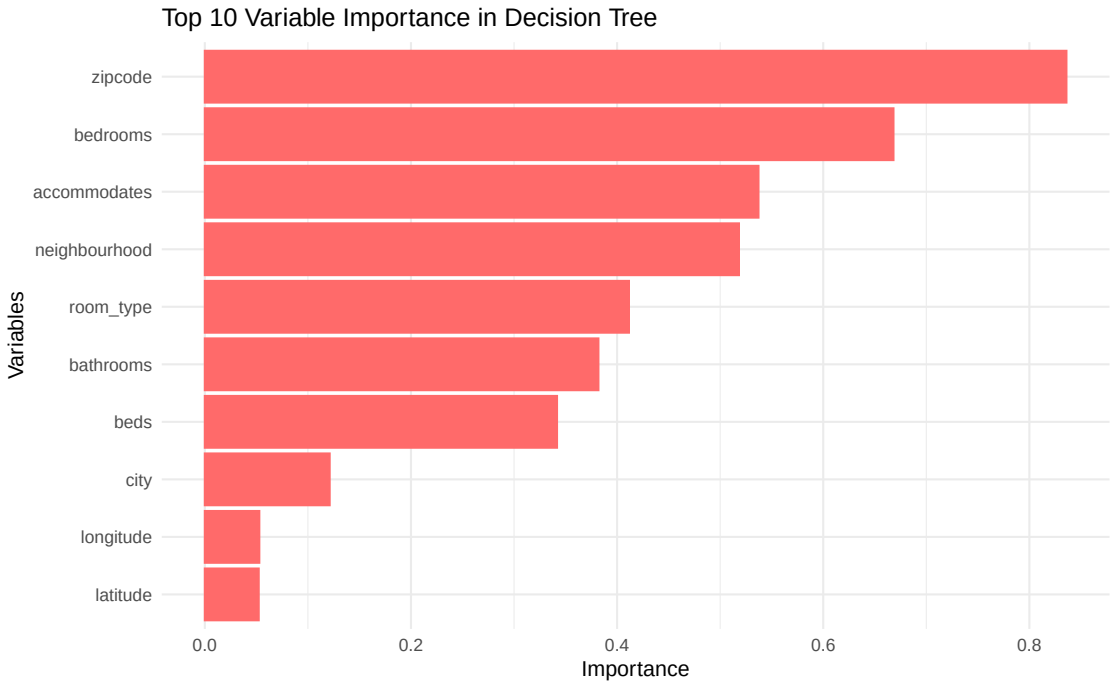

```
## [1] "Decision Tree In-sample R^2: 0.556758365170533"
```

```
## [1] "Decision Tree Out-of-sample R^2: 0.546862383999424"
```

```
## [1] "Decision Tree RMSE: 0.485505403772968"
```

Feature Importance

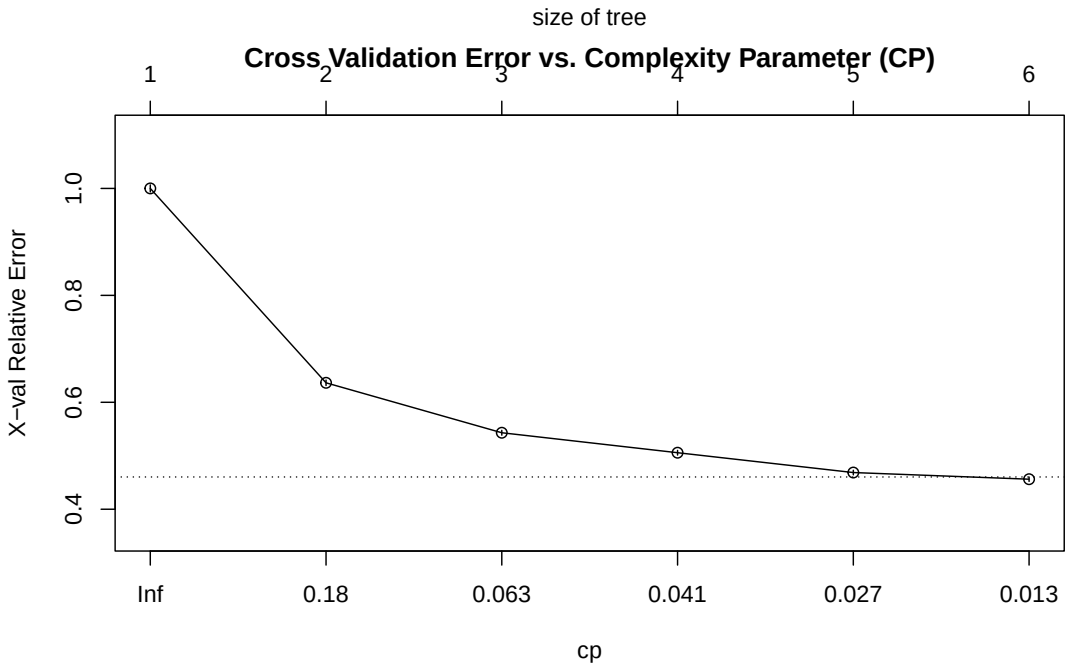
From the feature importance plot, it is evident that `zipcode`, `bedrooms`, `accommodates`, `neighbourhood`, `room_type`, `bathrooms`, `beds`, `city`, `longitude`, and `latitude` are the most significant splitting criteria. An importance score of over 0.8 for `zipcode` suggests that geographical location plays a crucial role in predicting `log_price`. The other variables are also relevant but to a lesser extent compared to `zipcode`. This indicates that Airbnb house prices are primarily determined by location and house conditions, with `zipcode` being the most influential factor.



Complexity Parameter (CP) Plot

The CP plot displays the cross-validation error versus the complexity parameter (cp). The horizontal line represents the minimum cross-validation error plus one standard error (1-SE rule). The plot indicates that the error stabilizes and does not significantly decrease after a certain point of complexity.

After pruning the decision tree using cross-validation, the RMSE and R^2 values remain unchanged, indicating that the decision tree does not undergo any modifications. The identical R^2 and RMSE values before and after pruning suggest that the tree was already optimally pruned at the chosen CP value. This implies that further pruning would not simplify the model without losing predictive power. The decision tree captures the essential patterns in the data with the selected variables, and additional complexity does not contribute to model improvement.




```
## [1] "Pruned Decision Tree In-sample R^2: 0.556758365170533"
```

```
## [1] "Pruned Decision Tree Out-of-sample R^2: 0.546862383999424"
```

```
## [1] "Pruned Decision Tree RMSE: 0.485505403772968"
```

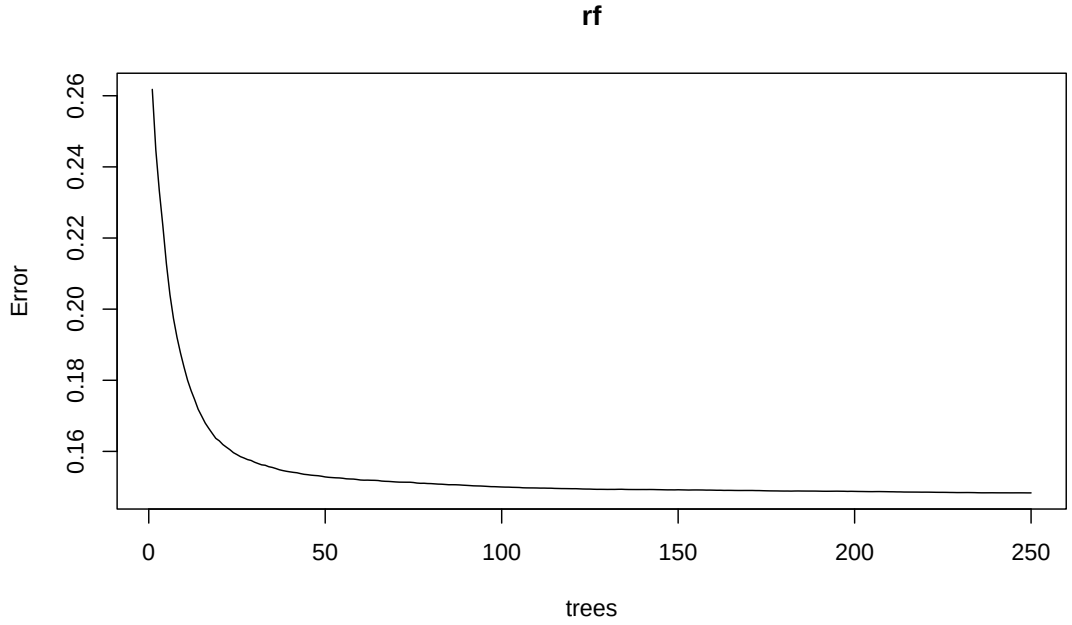
To further enhance the performance of the model, the next step constructs a random forest model comprising 250 single decision trees. This ensemble method aims to improve predictive accuracy and robustness by averaging the results of multiple decision trees. Random forests reduce over-fitting and improve generalization by combining the predictions of many individual trees, each trained on a random subset of the data and features.

Random Forest

Decision Tree vs. Random Forest

Unlike a single decision tree, the results of a random forest cannot be easily visualized in a tree structure, as it usually comprises hundreds of trees. Instead, we can plot a trend line indicating the model error as the number of trees in the model increases. From the plot, we can observe that the model error decreases steadily as the number of trees increases. However, after reaching approximately 50 trees, the reduction in error becomes less significant, indicating diminishing returns from adding more trees.

The in-sample (IS) R^2 of the random forest model is 0.867, and the out-of-sample (OOS) R^2 is 0.713. Both IS and OOS R^2 values are higher than those of the single decision tree. However, the larger difference between IS R^2 and OOS R^2 in the random forest model, with IS R^2 being much higher, suggests that the random forest model might suffer from overfitting. Despite this potential overfitting, a random forest model is still preferred over a single decision tree due to its better out-of-sample performance.

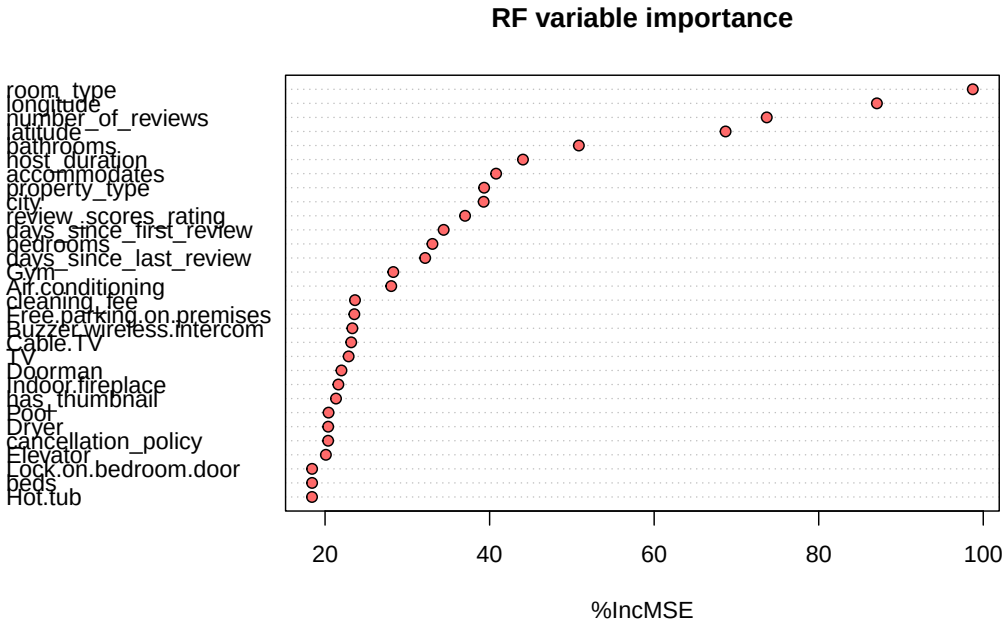



```
## [1] "In-sample R2: 0.861662749823644"
```

```
## [1] "Out-of-sample R2: 0.711574024583154"
```

Feature Importance

From the feature importance plot, we can see that more features are deemed important in the random forest than in the decision tree. The top 10 important features are `room_type`, `longitude`, `number_of_reviews`, `latitude`, `bathrooms`, `host_duration`, `accommodates`, `property_type`, `city`, and `review_scores_rating`. Generally, features that are important in a single decision tree are also important in the random forest. `zipcode` and `neighbourhood`, which are considered very important in the decision tree, do not fit in a random forest model because it cannot support categorical variables that have more than 53 categories. This issue can be fixed using more advanced techniques in XGBoost model.

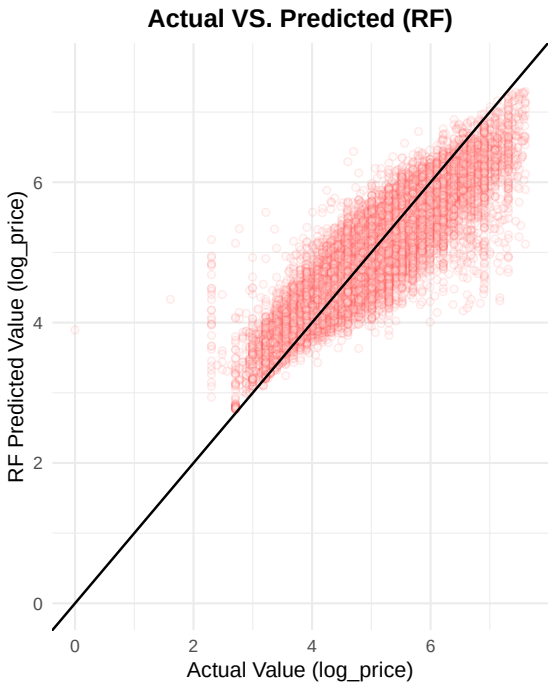


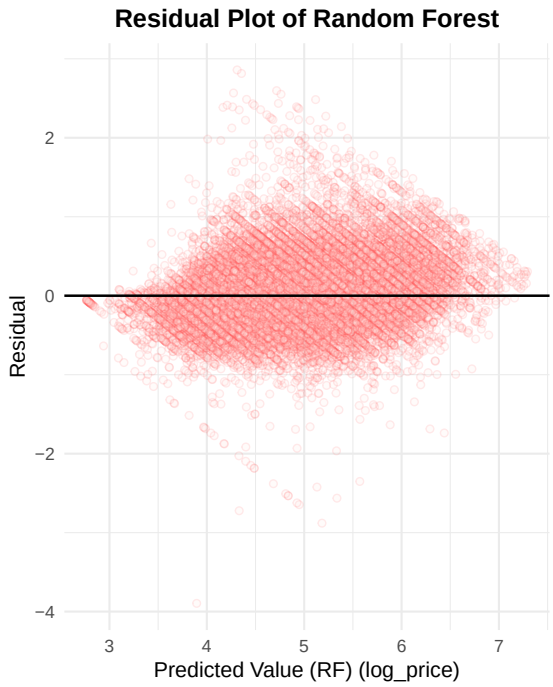
Actual vs. Predicted Values

From the scatter plot of actual vs. predicted values of the random forest, we can see that the points are generally distributed uniformly around the reference line. The residual plot shows that the model tends to make higher errors in predicting prices in the middle range. This could be due to the fact that house prices are easier to predict when they are extremely low or high.

Despite the random forest outperforming a single decision tree, a drawback is that it requires significantly more computational resources in terms of both space and time to construct. To address these issues, the next step involves using a more advanced algorithm that has gained popularity due to its speed and performance.

The next step implements Gradient Boosting, an advanced ensemble learning algorithm. Gradient Boosting builds models sequentially, each new model correcting errors made by previous models. This approach can significantly improve model accuracy and is well-suited for handling complex datasets. By using Gradient Boosting, we aim to achieve better predictive accuracy while maintaining reasonable computational efficiency. This advanced technique will help address the limitations observed in the random forest model and further enhance the performance of our predictive model.





XGBOOST

XGBOOST vs. Random Forest

The in-sample (IS) R^2 using XGBoost is 0.774, and the out-of-sample (OOS) R^2 is 0.722. Compared with the random forest model, XGBoost shows a reduced over-fitting problem, as indicated by the closer IS and OOS R^2 values. Additionally, XGBoost achieves better overall performance, which can be attributed to its advanced boosting mechanism that builds trees sequentially, each one correcting the errors of the previous ones. This allows XGBoost to handle complex patterns in the data more effectively than the random forest.

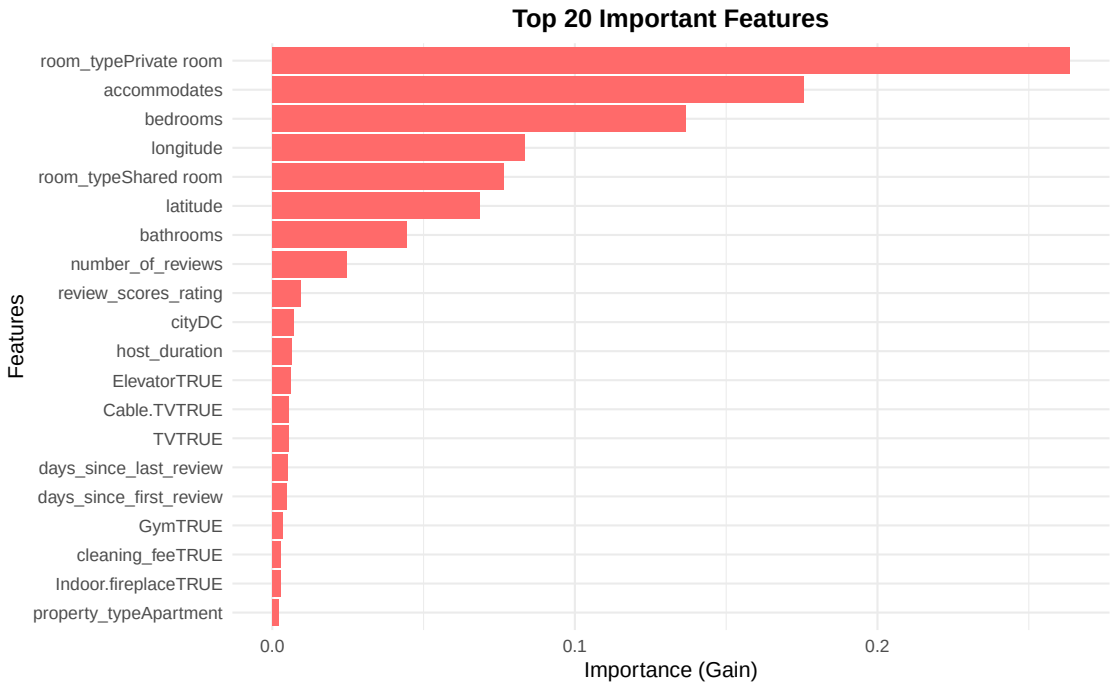
```
## [1] "XGBoost In-sample R2: 0.77399481514639"
```

```
## [1] "XGBoost Out-of-sample R2: 0.721658968384683"
```

Feature Importance

The most important features in splitting are generally consistent between XGBoost and the random forest. `room_type` is identified as the most important feature in both models. Other significant features include `accommodates`, `bedrooms`, `longitude`, `room_typeShared room`, `latitude`, `bathrooms`, `number_of_reviews`, `review_scores_rating`, and `city`.

In XGBoost, the importance score is derived from the feature's contribution to the model's predictive accuracy. It measures the gain, which is the improvement in accuracy brought by a feature to the branches it is involved in. Higher gain indicates more important features. For instance, `room_type` has the highest gain, meaning it significantly improves the model's predictions when used for splitting.

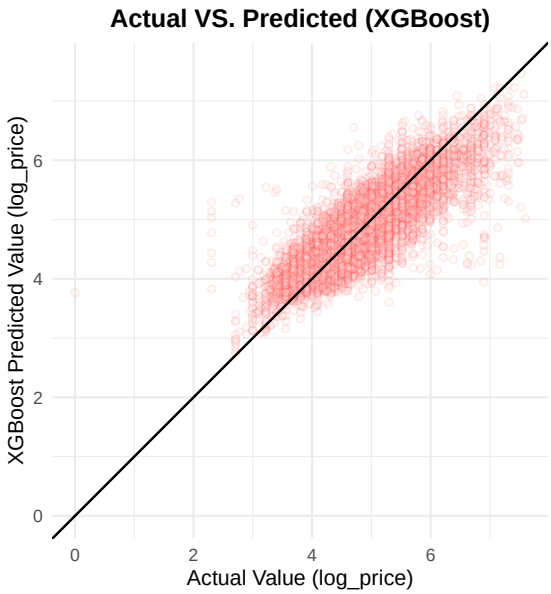


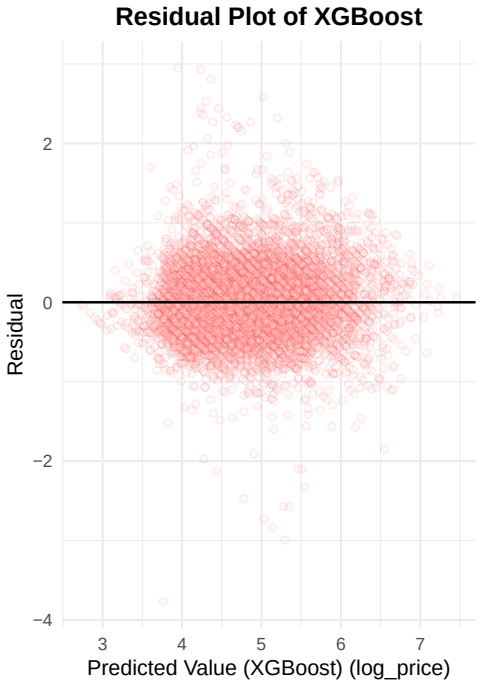
Actual vs. Predicted Values

From the actual vs. predicted values scatter plot and the residual plot, it is evident that the issues observed in the random forest model, such as higher errors in the middle range prices, are mitigated to some extent with XGBoost. The scatter plot shows a more uniform distribution around the reference line, indicating better alignment between actual and predicted values.

The residual plot for XGBoost also demonstrates a more uniform distribution of errors, with residuals centered more closely around zero. This indicates that the model's predictions are more accurate across different price ranges. Additionally, the residual range has shrunk compared to the random forest model, suggesting that XGBoost produces fewer extreme prediction errors.

Overall, XGBoost provides a significant improvement in predictive performance and error distribution, making it a more reliable model for predicting Airbnb house prices.


Actual VS. Predicted (XGBoost)
XGBoost Predicted Value (log_price)
Actual Value (log_price)
0
2
4
6


Residual Plot of XGBoost
Residual
Predicted Value (XGBoost) (log_price)
2
0
−2
−4
3
4
5
6
7