Predicting Apartment Rental Prices: A Machine Learning Approach

Rahul Kumar Sahoo
Politecnico di Torino
s316411@studenti.polito.it

Shiva Pourfeilieh
Politecnico di Torino
s329207@studenti.polito.it

June 23, 2025

Abstract

This project tackles the challenge of predicting apartment rental prices using machine learning. Relying on a large and diverse dataset of 99,487 apartment listings collected from U.S. cities, we developed a regression pipeline that estimates monthly rent prices by combining structured features—such as square footage, location, and number of rooms—with unstructured information, including textual descriptions and amenities. The dataset is divided into a development set (79,589 records with known prices) and an evaluation set (19,898 records without prices). Our approach involves a series of preprocessing steps, feature engineering techniques, and model selection procedures aimed at reducing the prediction error, measured using the Mean Absolute Error (MAE). The final Random Forest model achieved an MAE of 179.44 and R^2 score of 0.778, demonstrating strong generalization performance in real estate price forecasting.

1 Problem Overview

Accurate rental price prediction plays a significant role in the real estate sector, supporting more informed decisions for tenants, landlords, investors, and urban policymakers. This project addresses the task of estimating monthly apartment rent using machine learning techniques applied to a real-world dataset of housing advertisements across the United States.

The dataset contains **99,487 apartment** listings, divided into two parts: **a development set of 79,589 entries**, where the rent price is available and serves as the target variable, and **an evaluation set of 19,898 entries**, where the price is withheld. Each listing includes a diverse set of features—both structured (e.g., number of bedrooms and bathrooms, square footage, geographic coordinates) and unstructured (e.g., textual descriptions, amenities).

This task is formulated as a supervised regression problem. The aim is to build a model that minimizes prediction error, measured using the **Mean Absolute Error (MAE)** as per competition guidelines. Addressing this challenge requires a robust preprocessing pipeline, including the treatment of missing values, encoding of categorical features, and the extraction of relevant information from text data such as listing titles and descriptions.

1.1 Dataset Description

The dataset consists of real estate listings collected from various U.S. platforms. Each entry includes structured fields such as geographic coordinates, apartment size, number of rooms, and publication time. It also includes textual fields like the title and full description of the listing, as well as a list of available amenities.

This combination of structured and unstructured data provides a rich basis for modeling rental prices. While numerical and categorical features offer measurable indicators of apartment characteristics, the textual and semantic content adds contextual value that may influence price.

2 Proposed Approach

To predict rental prices accurately, we designed a complete regression pipeline composed of four main phases: preprocessing, model selection, hyperparameter tuning, and evaluation. Each stage contributes to improving the model's ability to generalize well on unseen data.

2.1 Preprocessing

As a first step, we inspected the dataset using pandas [2] and NumPy [4] to understand the structure and data types of each column, as well as the presence of missing values. This initial assessment helped us distinguish between features with negligible missing data and those with more significant gaps. For example, we found substantial missing values in fields such as amenities, pets_allowed, and address, which required dedicated handling.... During our analysis, we identified a small group of records that were missing not only geographical coordinates (latitude, longitude), but also higher-level location fields such as cityname and state. These listings had no structured location information at all, making them problematic for location-based modeling. To recover this information, we extracted a list of all unique city names available in the dataset and searched the title and body fields for any mention of these known cities.

We addressed missing values in the bedrooms and bathrooms columns by scanning the combined title and body fields for expressions such as ”2BR,” ”three bedrooms,” or ”one bath.” For remaining records, we used a rule-based imputation strategy based on apartment size (square_feet), grouping listings into ranges and computing average room counts for each range using NumPy [4] array operations for efficient computation.

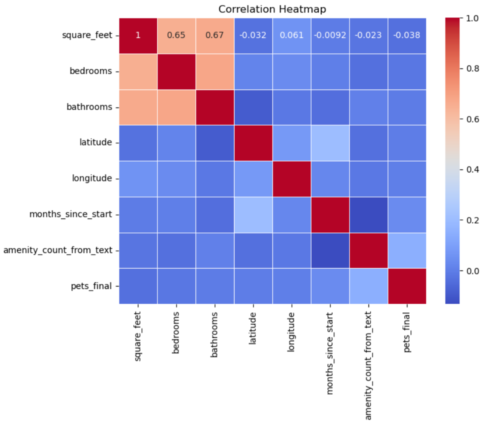

Figure 1: Correlation heatmap showing linear relationships among numerical features. Square footage and number of rooms show notable positive correlation with price.

2.2 Model Selection

In order to identify the most suitable model for predicting rental prices, we compared three different regression techniques using scikit-learn [1]: Linear Regression, Random Forest Regressor (RFR) [3], and XGBoost Regressor. Each of these models has distinct characteristics and assumptions, and our goal was to balance interpretability, generalization, and predictive performance.

Linear Regression produced a low R^2 score of 0.213 and a high MAE of 497.56, indicating that the data's underlying patterns are highly non-linear. Random Forest Regressor [1][3] significantly outperformed other models, achieving an R^2 score of 0.778 and an MAE of 179.44. XGBoost produced an R^2 score of 0.722 and MAE of 238.44, better than Linear Regression but still below Random Forest performance.

Table 1: Comparison of regression models on validation set

Model	**R^2 Score**	**MAE**
Linear Regression	0.213	497.56
XGBoost	0.722	238.44
Random Forest	**0.778**	**179.44**

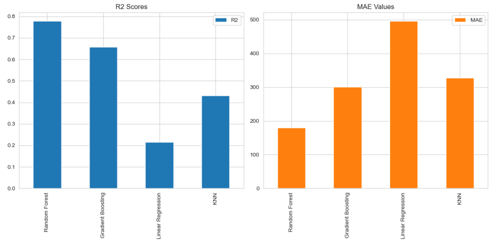

Figure 2: Comparison of model performance based on R^2 score and MAE. Random Forest shows the best overall performance.

2.3 Hyperparameter Tuning

After selecting Random Forest [3] as our base model, we conducted extensive hyperparameter optimization using grid search cross-validation implemented with scikit-learn [1]. The key parameters tuned included:

- `n_estimators`: Number of trees in the forest
- `max_depth`: Maximum depth of individual trees
- `min_samples_split`: Minimum samples required to split a node
- `min_samples_leaf`: Minimum samples required at leaf nodes
- `max_features`: Number of features considered for best split

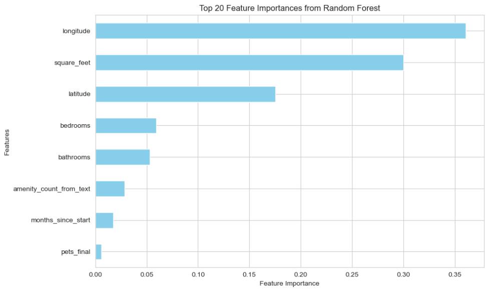

Figure 3: Feature importance plot from the Random Forest model. Square footage, latitude, and number of bathrooms are among the most influential variables.

2.4 Evaluation

Once the final model was selected and tuned, we evaluated its performance using the development set through cross-validation implemented with scikit-learn [1]. The main evaluation metric, as specified in the project requirements, was Mean Absolute Error (MAE). In addition, we monitored the R^2 score to measure how well the model captures variance in the target variable.

The best-performing model (Random Forest Regressor with optimized hyperparameters) achieved a strong balance between low error and high explanatory power, with an MAE of approximately 179.44 and an R^2 score of 0.778. These results indicate that the model is capable of making relatively accurate predictions across a wide range of listings.... To ensure the reliability of the evaluation process, we used 5-fold cross-validation, which helped reduce the risk of overfitting to a particular subset of data. Moreover, we verified that the error distribution was stable across folds, confirming the robustness of our pipeline.

3 Results

The final model—a Random Forest Regressor [1][3] tuned through extensive grid search—demonstrated strong predictive performance on the validation data. It achieved a Mean Absolute Error (MAE) of approximately 179.44 and an R^2 score of 0.778, substantially outperforming both Linear Regression and XGBoost in our experiments.

3.1 Model Performance Visualization

To assess prediction quality, we visualized the model's output using diagnostic plots that demonstrate the accuracy and reliability of our predictions.

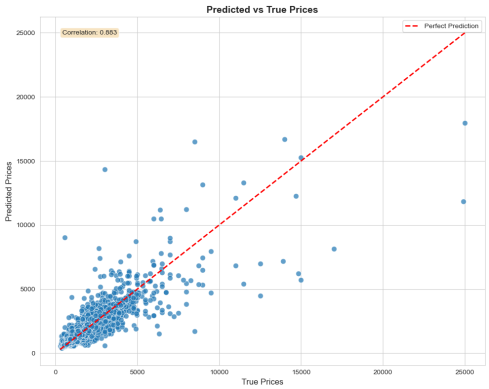

Figure 4: Predicted vs True Rental Prices. Most points lie close to the diagonal line, indicating accurate predictions across a wide price range with strong correlation between predicted and actual values.

As shown in Figure 4, predicted prices closely follow the true values across the majority of listings. The points are generally concentrated along the diagonal line, indicating high prediction accuracy. The strong linear relationship demonstrates that our Random Forest model effectively captures the underlying patterns in rental pricing.

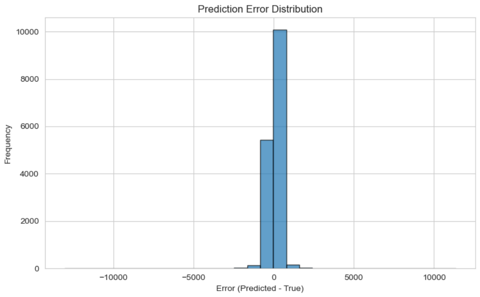

Figure 5: Distribution of prediction errors. The histogram is centered around zero with most errors falling within a narrow range, suggesting low bias and good model calibration.

The error distribution in Figure 5 confirms that most predictions fall within a narrow band around the true value. The histogram is centered around zero, suggesting that the model is both unbiased and consistent. The symmetric distribution of errors indicates that the model does not systematically over or under-predict rental prices.

Figure 5 provides additional insight into the model's performance across different price ranges. The scatter plot shows that prediction errors remain relatively consistent across most price levels, though there is a slight increase in error variance for higher-priced properties, which is common in real estate modeling due to the scarcity of luxury listings in the training data.

3.2 Key Findings

These results validate the effectiveness of our pipeline. Structured features—such as square footage, number of rooms, and geographic coordinates—contributed most to the model's accuracy, as demonstrated in the feature importance analysis implemented using scikit-learn [1]. In contrast, features like amenities, listing source, and posting date did not yield noticeable improvements and were excluded after exploratory analysis.

The final predictions for the evaluation set were generated using the optimized model and demonstrate the practical applicability of machine learning techniques in real estate price prediction. The strong performance metrics and visual diagnostics confirm that our approach successfully balances model complexity with generalization capability.

4 Discussion

Our comprehensive approach to rental price prediction demonstrates the importance of thorough data preprocessing using pandas [2] and NumPy [4], careful feature selection, and systematic model evaluation. The Random Forest algorithm [3] proved particularly well-suited for this heterogeneous dataset, effectively handling the mix of numerical, categorical, and geographic features while maintaining interpretability through feature importance analysis.

The success of our model highlights the value of structured features in real estate prediction, while also showing that not all available features contribute meaningfully to predictive performance. This finding emphasizes the importance of feature selection and the potential pitfalls of including noisy or irrelevant variables in machine learning models.

Future work could explore the integration of external data sources, such as neighborhood demographics, school ratings, or economic indicators, which might provide additional predictive power for rental price estimation.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825-2830, 2011. 2, 3, 4

[2] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, 2010, pp. 56-61. 2, 5

[3] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001. 2, 3, 5

[4] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357-362, 2020.

2, 5