Dominique Desert

Mr. Paolucci

Data Science

6 June 2025

Predicting Airbnb Prices with Machine Learning

Throughout the years, the popularity of Airbnb has grown significantly, and more and more people are using it as a side hustle and are investing in properties that they can flip on Airbnb. But, with no prior knowledge of real estate, it is hard to determine the proper price to list the locations on Airbnb. Therefore, I wanted to solve the issue using machine learning models.

Project structure

There are multiple components to the project: the models, the frontend, and the data cleaning and preprocessing. The data was gathered from InsideAirbnb, where I was able to get the listings and reviews.

First, the core files used to perform analysis and predictions on the dataset are ML.py, NN.py, and stacked_model.py. The train_models.py creates the training pipeline, while the app.py is the web component of the project.

The data preprocessing and analysis files helped prepare the code for model training and analysis. The data_grouping.py file organizes the listings into categories based on performance, while calculate_averages.py helps with creating a default JSON file that would contain the values that should be used for missing data, and the statistical scripts, such as stats.py and statswithplots.py, help with giving insight into the dataset.

Data Preprocessing and Cleaning

In order to use the raw price data, the data needed to be properly cleaned. First, the commas and currency symbols needed to be removed so that the data could take the numerical values as integers and floats instead of strings. Then log transformation was used to handle the right-skewed distribution of prices: this was crucial for improving the model performance and normalizing the target variable, transforming the data to make it more suitable for machine learning models. Feature engineering was very important; the pipeline needed to categorize the amenities offered from the listings into essential, luxury, and outdoor categories, creating features that would be able to grasp the quality and type of amenities offered. Location features are generated by clustering the listings based on their coordinates, while the reviews were used to help create quality indicators. Then, label encoding, converting categorical variables to numerical values, was used for room types, property types, and neighborhoods. Missing values were handled using a combination of median values and default values to ensure predictions could be made even with missing data.

Models

ML.py contained the machine learning model that used Random Forest for prediction, NN.py used a Neural Network to predict the prices, and stacked_model.py used a combination of Random Forest and XGBoost for price prediction.

ML.py had the best performance, using 200 decision trees and a depth of 15. It had an R^2 of 87.65% and an MAE of $17.30. The model was successful due to its ability to work with non-linear relationships and outliers.

NN.py used deep learning with multiple layers for prediction. The network had 50 neurons and two hidden layers with 100 and 50 neurons, with an output of a single neuron. The model used LeakyReLU, batch normalization, and dropout layers; these were used to prevent overfitting in the model. Even though it wasn’t as accurate compared to Random Forest, it was able to work with the complex patterns and insights well.

Lastly, stacked_model.py was a combination of both base models, achieving an R² of 0.8613. It had more stable predictions by using the strengths offered by both models.

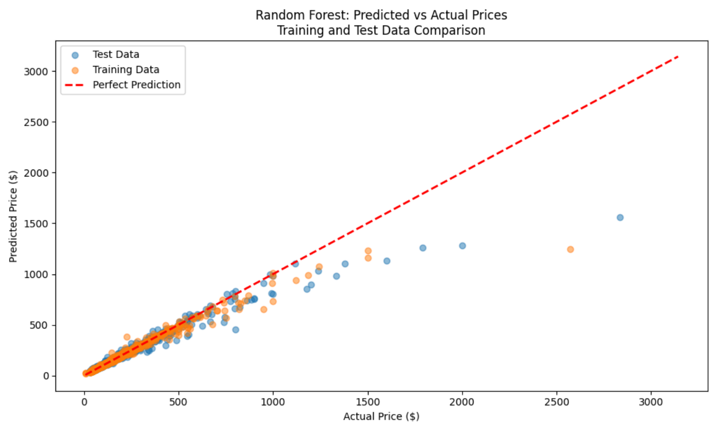
Web Application

The front end was built using Flask. It has a main page, a prediction page, a model information page, and lastly, a page for all the listings. The backend uses RESTful API endpoints for price prediction and data retrieval.

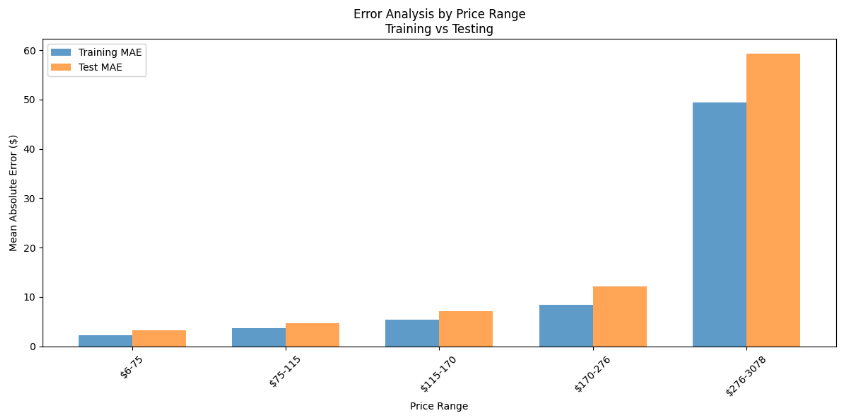
Results:

The system performed well using MAE and RMSE to evaluate the results. The Random Forest model achieved the best performance with an R² of 0.8764 (87.65% accuracy) and MAE of \$17.30. The Neural Network follows with an R² of 0.8442 (84.42%) and an MAE of \$19.42. Lastly, the Stacked Model had an R² of 0.8613 (86.13% accuracy) and an MAE of \$18.20.

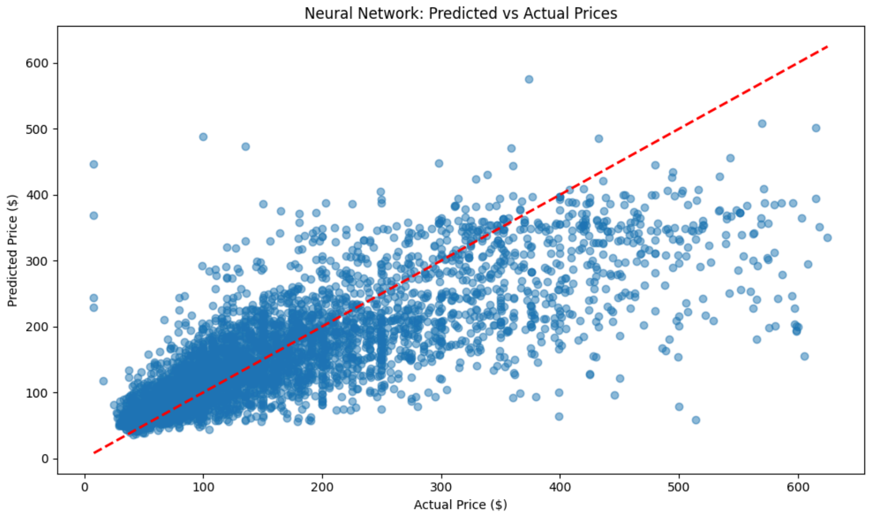
Graph Analysis:



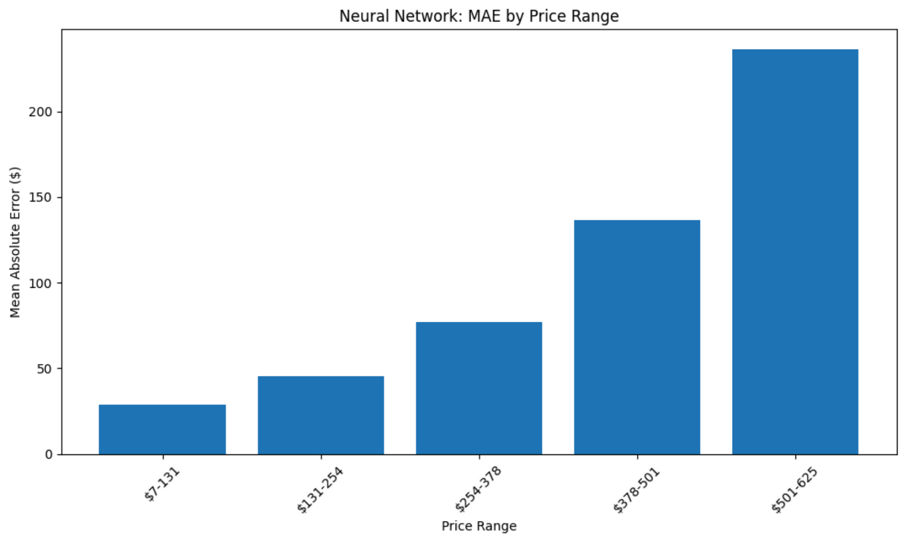
There is a really strong correlation between predicted and actual prices. It shows close clustering between the points around the diagonal line, displaying high prediction accuracy.



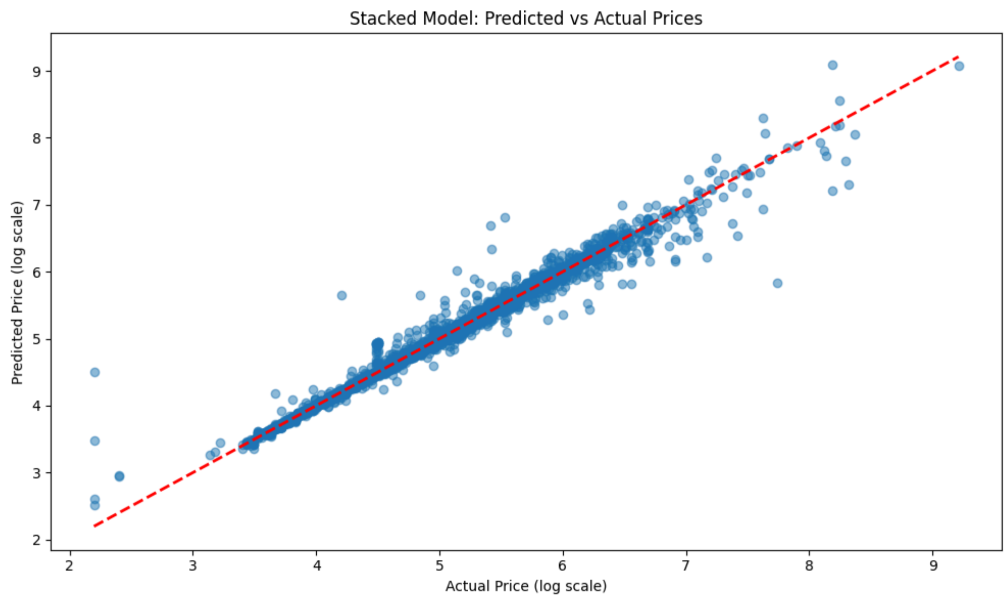
The MAE was consistent across different price ranges, with an increase in error in the higher-priced listings. The consistent error between the models indicates that the model is reliable regardless of the price point. That is why I chose to use this model in the front end for the prediction form.



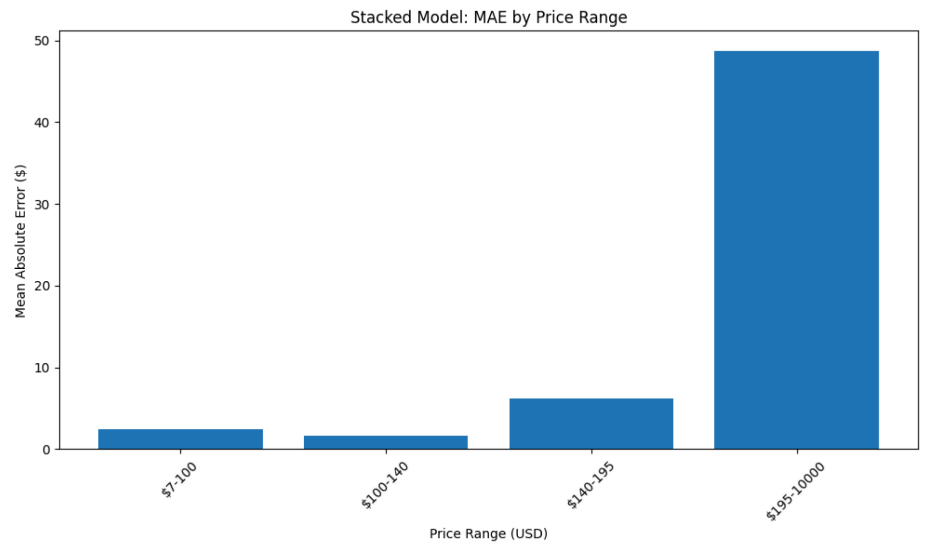
There is a more scattered distribution of predictions compared to the Random Forest model. The model has a general trend of prices and more variance in its predictions, especially as it approaches the higher-priced models.



The graph displays the variance in MAE, with higher errors in both low and high-price segments. This indicates that the model performs well with the mid-range price segments but is less stable with the more extreme values, such as outliers and extremes.



The graph demonstrates a strong linear relationship between the predicted and actual prices. The points were clustered around the 45-degree line, indicating that it has strong performance. The model performed the best with lower- and mid-range priced listings, with scattered predictions at high prices.



The MAE by price range shows that it performed the best for the mid-range properties, with the MAE being the lowest in those sections. The error increases as the price moves to the very high and low extremes, meaning that the accuracy decreases significantly when the property prices are above or below the average listing.

Comparative Analysis

Comparing the three models, the Random Forest model sticks out with the tightest clustering of predictions and consistent error distribution. The Neural Network model, having more variance, demonstrated the ability to capture the complex and nonlinear relationships in the data. The Stacked Model successfully combined the strengths of both models, having a more balanced approach that maintains good accuracy while also being able to adjust to different listing types. The graphs show that the Random Forest is best for general prediction, while the Stacked Model is good for a compromise between accuracy and robustness.

Conclusion

The project demonstrated how machine learning can be used in real-world problems such as Airbnb and real estate. By collecting the data from InsideAirbnb and using preprocessing and cleaning techniques, I was able to create a model that can accurately predict listing prices. Among the 3 models presented, Random Forest performed the best, having strong accuracy and reliability. The Neural Network was able to grasp the complex patterns, and the stacked model played to the strengths of the two base models used for it. The main difficulty that I faced was getting the model to work with the frontend, but after hours of debugging, using ChatGPT and Stack Overflow, it was finally running. This analysis shows that with proper data and modeling strategies, you can improve the decision-making and profitability for Airbnb hosts.

References

Cox, Murray. *Inside Airbnb: Get the Data*. Inside Airbnb, https://insideairbnb.com/get-the-data/.

Accessed 27 May 2025**.**