Using Regression to Predict Prices of AirBnB's in NYC

Lindsay Kim | Sooyeon Kim | Nick Pacia
ek532@cornell.edu | sk2696@cornell.edu | snp43@cornell.edu

December 12, 2022

Abstract

In this research project, we sought to use existing algorithms to accurately predict the prices of AirBnB rentals in New York City with various data about the listing. We used several linear regression models, coupled with preprocessing strategies such as bag-of-words models for text data, and one-hot encoding for categorical data. We tuned our model iteratively to improve its performance on our training and development data, primarily using the R^2 score and the symmetric mean absolute percentage error (SMAPE) to assess our model's performance. Overall, our model had disappointing performance and we mostly attribute this to a difficulty in predicting prices more generally.

1 Introduction

AirBnB is a global two-sided market platform that allows users to arrange and offer their housing so that others can rent the ones that satisfy their desired conditions for a certain period. Since there are diverse options for users to input in AirBnB, there are many factors that the guests can choose according to their needs. Hosts can choose room types, amenities, minimum/maximum nights, etc and guests can see their available options, price, and reviews. These days, people seek AirBnB more frequently as an alternative temporary housing option instead of expensive hotels. Especially, New York City is one of the most densely and diversely populated cities in the world. Every year, many tourists visit New York City to spend their vacation and use the AirBnB platform to arrange their accommodations. Therefore, we conduct an exploratory data analysis using the application of machine learning to help tourists to choose a good AirBnB in New York City.

2 Background

For our project, we have decided to use a data set we found from Kaggle containing AirBnB listing data in New York City [1]. The columns included in this data set are as follows:

- An index
- The name of the listing (text data)
- The name of the host
- A unique host ID
- The borough the listing is located in
- The neighborhood that the listing is located in (e.g. Soho, Long Island City)
- The latitude and longitude of the listing
- The room type (e.g. apartment, private room)
- The minimum number of nights a guest can book the listing
- The number of reviews the listing has
- The date of the most recent review
- The reviews per month
- The total number of listings the host has
- The number of days per year the listing is available

We sought this data set with the goal of using data about a listing to predict the price of AirBnB listings.

There are some previous work done related to our project on predicting the price of AirBnB. One of the projects also tried to predict the price of AirBnB rentals in Boston, MA [2]. The project answered the following questions from the analysis: 1) What features affect the price? and 2) the popular description words in different price groups [2]. To find the features that affect the price, the author took top 30 features using the RFE model [2]. The author concluded that every additional bedroom, bathroom, and guests will cost extra money [2]. Additionally, neighborhood areas, superhosts' listings, property types, renting the entire room, and strict cancellation policy also affected the price of AirBnB [2]. Moreover, to identify the popular description words, the author used Wordcloud and successfully found out that the more expensive listings contain more information about the comfort and location more frequently [2].

3 Method

3.1 Data Preprocessing

Prior to any preprocessing, we felt it fit to eliminate some columns completely due to irrelevance to our objective or redundancy. Namely, we removed the columns with the host's name, unique host ID, and the date of the most recent review, as we felt these did not pertain to our final objective and are mostly independent of the price of the listing.

Since there was no separate training and testing set, we used scikit-learn's `train_test_split` to split the data into training and testing sets, then further split the training set into a training and development set. The overall split was 70/15/15 for training, development, and test sets, respectively.

Now that the data was split, we began our preprocessing. We thought it would be interesting to convert the name of the listing into a bag-of-words model and assess its impact on the final price of the listing. Using regular expressions, we cleaned the listing names by stripping punctuation, removing stop words and setting all the characters to lowercase. To create our bag-of-words model, we used scikit-learn's CountVectorizer. Since there were nearly 50,000 rows in our data frame, we thought it would be wise to utilize the `min_df` argument in CountVectorizer, for that reason we set it to 15.

For the initial scale of numerical data, we standardized the data by using StandardScaler. Since variables that are measured at different scales do not contribute equally to the fit/ learning function of the model and could end up creating a bias, we chose to scale the data using StandardScaler for all columns with numeric data (latitude, longitude, minimum nights, etc.).

Finally, we chose to use one-hot encoding to encode the borough and room-type columns. We felt that these were great candidates for one-hot encoding because they are made of categorical data and all values fit into one of several categories. For this, we used scikit-learn's OneHotEncoder.

3.2 Exploratory Data Analysis

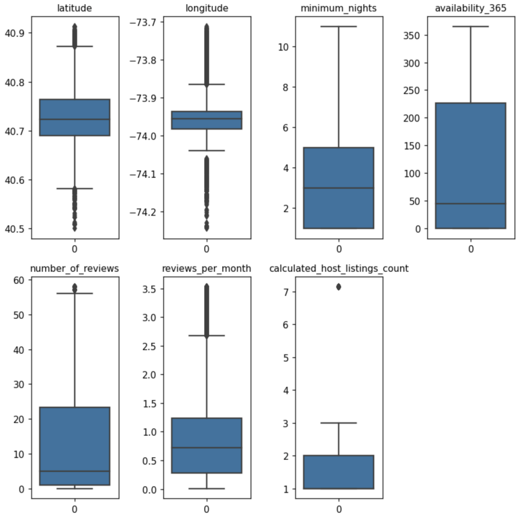

Figure 1: Boxplots for numerical data

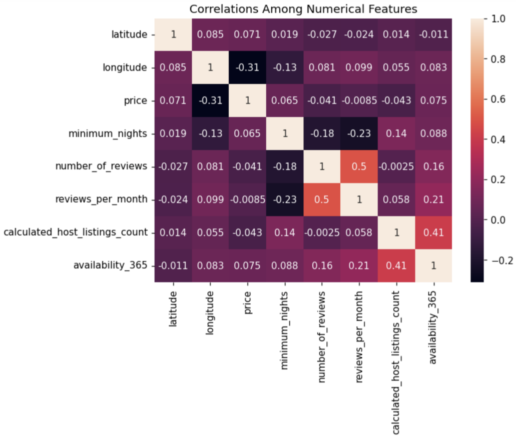

Figure 2: Correlations among numerical data

Since we were interested in detecting outliers in our data, we graphed box plots for all numerical features. See figure 1.

We saw that there were a lot of outliers in our data. We define outliers in our numerical data as follows:

$$\begin{gathered} x \text{ is an outlier if} \\ x < Q_1 - 1.5 \times \text{IQR or} \\ x > Q_3 + 1.5 \times \text{IQR} \end{gathered}$$

After detecting outliers by using the above rule, we imputed them with a median value.

Then, we plotted a heat map of Pearson correlation coefficients among the numerical features in order to find features that are highly correlated. See Figure 2. From the heatmap, we could see that there are two pairs of highly correlated features: *availability 365* and *calculated host listings count* with $\rho = 0.41$ and *number of reviews* and *reviews per month* with $\rho = 0.5$. Because highly correlated features could result in a high variance, we must exclude one of the two features. We decided to remove the one that is less correlated to our target variable, price, than the other. The features *reviews per month* and *calculated host listings count* were dropped.

3.3 Linear Regression

Our baseline model to predict the price of Airbnb listings was an ordinary least squares (OLS) linear regression model. A linear model follows the form:

$$y = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + ... + \theta_d \cdot x_d$$

The input variables for a linear model could be represented using a matrix X, which is defined as follows:

$$X = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_d^{(2)} \\ \vdots & & & \\ x_1^{(n)} & x_2^{(n)} & \dots & x_d^{(n)} \end{bmatrix} = \begin{bmatrix} - & (x^{(1)})^\top & - \\ - & (x^{(2)})^\top & - \\ & \vdots & \\ - & (x^{(n)})^\top & - \end{bmatrix}$$

The OLS linear regression model uses means square error as its objective function. The optimal θ that minimizes our objective function could be found by the normal equation:

$$\theta^* = (X^\top X)^{-1} X^\top y.$$

3.4 Kernel Ridge Regression

Kernel Ridge Regression (KRR) combines Ridge Regression and Classification with the kernel trick. The kernel trick means that we can use complex non-linear features within these algorithms with little additional computational cost. The form of the model learned by Kernel Ridge is identical to Support Vector Regression. However, different loss function, squared error loss combined with l2 regularization, is used for KRR.

We can compute a prediction $\phi(x')^\top \theta$ for x' only using their dot products as below:

$$\phi(x')^\top \theta = \sum_{i=1}^{n} \alpha_i \phi(x')^\top \phi(x^{(i)}).$$

The most important point from Kernel Ridge Regression is that we can get the value of θ by using only the dot products.

3.5 Coefficient of Determination

The Coefficient of Determination, R^2, measures the accuracy of the predictions, relative to constantly predicting the average $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y^{(i)}$. The coefficient denotes as following:

$$R^2 = 1 - \left(\frac{\sum_{i=1}^{n} \left(f(x^{(i)}) - y^{(i)}\right)^2}{\sum_{i=1}^{n} \left(\bar{y} - y^{(i)}\right)^2} \right)$$

When we interpret the value of R^2, R^2 of zero means that f is not better than the average prediction and R^2 of one corresponds to perfect accuracy.

Table 1: Linear regression performance for varying number of features (n)

Metric	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$
${R^2}_{\text{train}}$	0.3841	0.5219	0.5634	0.5681	0.5744	0.5753	0.5758	0.5772	0.5801
${R^2}_{\text{dev}}$	0.4004	0.5151	0.5627	0.5692	0.5757	0.5780	0.5790	0.5795	0.5520
$\text{SMAPE}_{\text{train}}$	0.3310	0.2894	0.2735	0.2718	0.2695	0.2695	0.2692	0.2689	0.2693
$\text{SMAPE}_{\text{dev}}$	0.3002	0.3002	0.2798	0.2784	0.2755	0.2745	0.2743	0.2736	0.2755

Features used for each n:
$n = 1$: (room type)
$n = 2$: (room type, name)
$n = 3$: (room type, name, neighborhood)
$n = 4$: (room type, name, neighborhood, availability 365)
$n = 5$: (room type, name, neighborhood, availability 365, minimum nights)
$n = 6$: (room type, name, neighborhood, availability 365, minimum nights number of reviews)
$n = 7$: (room type, name, neighborhood, availability 365, minimum nights number of reviews, longitude)
$n = 8$: (room type, name, neighborhood, availability 365, minimum nights number of reviews, longitude, borough)
$n = 9$: (room type, name, neighborhood, availability 365, minimum nights number of reviews, longitude, borough, latitude)

3.6 SMAPE

To account for differences in scale, we considered the metric of scaled losses. Therefore, we chose symmetric mean absolute percent error (SMAPE) since this allows either $y^{(i)}$ or $f(x^{(i)})$ to be small (or zero). The formula for SMAPE is as following:

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n} \frac{\left|f(x^{(i)}) - y^{(i)}\right|}{\left(|y^{(i)}| + |f(x^{(i)})|\right)/2}$$

We can interpret the SMAPE value by examining how close the value is to zero. The closer the value is to zero, the less error the model produces.

4 Experimental Analysis

Our very first baseline model that predicts the price of AirBnB listings was a linear regression model. The model was trained on data that contain these features: *name*, *latitude*, *longitude*, *minimum nights*, *number of reviews*, *availability 365*, *borough*, *neighborhood*, and *room type*. As discussed above, we used two performance metrics, the coefficient of determination (R^2) and SMAPE to evaluate our model. We fit scikit-learn's default linear regression model to processed features and achieved the R^2 values of 0.1870 on our training set and 0.2090 on our development set and SMAPE values of 0.4461 and our training set and 0.4536 on the development set. Because the R^2 values were disappointingly low and SMAPE values were unexpectedly high, we performed outlier detection and imputation. After imputing outliers based on the IQR method discussed above, we were able to achieve R^2 values of 0.5801 on our training set and 0.5520 on our development set and SMAPE values of 0.2693 on our training set and 0.2755 on our development set. The baseline model after replacing outliers performed much better than our original model. However, by exmaining the R^2 and SMAPE values, we could see that the new model performed slightly better on the training data set than the development set and thus, it was slightly overfitting.

In order to further improve our model and reduce overfitting, we tried different sets of features in our model. We experimented with all different combinations of features of size n, where $1 < n <$ total number of features. For example, $n = 2$ means that we are working with feature combinations, (*latitude*, *longitude*), (*latitude*, *minimum nights*), (*latitude*, *number of reviews*), (*latitude*, *availability 365*), and so on. For each n, we found the feature combination that resulted in the highest R^2 value on the development set when used in our linear regression model. The table 1 summarizes the result of this experiment. As seen in the table, the R^2 value increases and the SMAPE value decreases as n increases. However, we could observe that as n increases, the difference between the R^2 values of successive n is decreasing. In other words, the R^2 value is converging as n increases. Therefore, we calculated the difference between the R^2 values for development data of successive n and used a threshold of 10^{-3} to detect convergence. In other words, if the difference between successive R^2 values is less than or equal to the threshold, 10^{-3}, we can conclude that it is not worth adding more features to increase the performance slightly. It was found that the R^2 value converges when $n = 6$. This model uses the following features: *name*, *room type*, *neighborhood*, *availability 365*, *minimum nights*, and *number of reviews*.

This model had R^2 values of 0.5753 on the training data and 0.5780 on the development data and SMAPE values of 0.2695 on the training data and 0.2745 on the development data. Clearly, the difference in the metrics between the training data and development data is insignificant and thus, the model is no longer overfitting. Moreover, compared to our previous model that used all nine features, the R^2 value on the development data improved by 0.0260 while the SMAPE value on the development data was reduced by 0.0010 for this model. These two metrics demonstrate that this model performed better on the development data than our previous baseline model. In summation, we were able to reduce overfitting while also improving our model performance.

At this point, we believed we had exhausted the extent to which we could improve our model through linear regression, so we decided to try a different regression model. We were interested in trying a more complex regression model than linear regression to lower possible bias that might have come from the simplicity of our linear regression model. We tried a kernel ridge regression model of degree 4 as it is far more expressive than linear regression. However, this made little difference, as the R^2 values were 0.5770 and 0.5800 for training and development, respectively and the SMAPE values were 0.2686 and 0.2731 for training and development, respectively. The R^2 and SMAPE values on the development data were insignificantly higher than 0.5780 and 0.2745, which were R^2 and SMAPE values, respectively, for our previous linear regression model with 5 features. This was disappointing and we were surprised that even a more expressive model could not make a significant difference. In addition, the kernel ridge regression model was significantly more computationally intensive than the linear regression model.

Because the difference in the performance for the two models was insignificant and the kernel ridge regression model was unnecessarily more intensive, we chose our linear regression model with the six features as our final model. On our *test* set, our final R^2 and SMAPE values were 0.5647 and 0.2518, respectively.

5 Discussion

Our final R^2 and SMAPE values on the test data were 0.5647 and 0.2518, respectively. The R^2 value suggests that 56.47% of the variance in the price is explained by the listing name, type of room, neighborhood, availability, the minimum number of nights, and the number of reviews. The SMAPE value suggests that there are 25.18% of error present in our model.

The R^2 and SMAPE values on the test set were slightly lower than 0.5753 and 0.2695, which were R^2 and SMAPE values for the training data, respectively. The difference between the R^2 value for the test data and that of the training data is 0.0106. The difference between the SMAPE value for the test data and that of the training data is 0.0177. This is a negligible difference and thus, we can conclude that our model is neither underfitting nor overfitting, and thus predicts the price of AirBnB listing fairly well.

Although we observed minimal bias and variance, our model fell a bit short of our expectation because only around 50% of the variance in price was explained by the independent variables in our model. We still have some rooms for improvements. For example, we could further improve our model by collecting more data and trying different expressive models other than kernel ridge regression.

6 Conclusion

Our performance fell slightly short of our expectation, and though our model could certainly be improved, we believe our performance is more a testament to the unpredictability of price, rather than any intrinsic issues with the data or our model.

Our research and experimentation in this project have made us aware of efforts by various companies such as Zillow to take on a similar task and attempt to algorithmically predict housing prices and purchase/sell homes based on these predictions. These attempts have been largely unsuccessful and have cost companies such as Zillow hundreds of millions of dollars [3]. The unpredictability of price seems to be an immutable law of economics, as it is proven time and time again how difficult it is to accurately predict prices. Perhaps if we had a (significantly) larger data set we would have achieved greater accuracy, or perhaps if we had the computational resources to use a more complex machine learning model; however, there is always the possibility that the price of AirBnB listings simply cannot be accurately predicted based on the metadata of the listing.

References

[1] Gomonov, Dennis. "New York City Airbnb Open Data" Kaggle. 2019. https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data

[2] Rudovski, Kirill. "Final Project - predict Airbnb price" Kaggle. 2021. https://www.kaggle.com/code/kirillrudovski/final-project-predict-airbnb-price

[3] Parker, Will, and Konrad Putzier. “What Went Wrong with Zillow? A Real-Estate Algorithm Derailed Its Big Bet.” The Wall Street Journal. Dow Jones & Company, November 18, 2021. https://www.wsj.com/articles/zillow-offers-real-estate-algorithm-homes-ibuyer-11637159261