DATA 1030 Project Final Report

Predicting Airbnb Listing Prices in New York City Using Regression Models

Zhe (Peter) Huang

Brown University

Data Science Initiative

Github Repository

Introduction

Airbnb brings us a new way of traveling and exploring the world other than staying in "boring" hotels. However, unlike hotels, the pricing for Airbnb is more customized and, therefore, more variant. When we book a place on Airbnb, we always ask ourselves: is this place worth the price? In other words, customers want to understand the pattern of pricing and what characteristics impact the pricing decision the most. Thus, it may be worthwhile to create a regression model to predict Airbnb listing prices based on their features, so we can predict the price of future listings as accurately as possible and get a reference to the price for a particular place to stay.

The data set is named "New York City Airbnb Open Data", and the original data set can be found here on Kaggle. This data set contains information on 48,895 observations of Airbnb listings in New York City in 2019 with 16 variables, in which 5 of them are categorical variables, and the rest are numeric variables, including one date variable. There are some missing values. The target variable is “price”, which is the price per day in dollars. The information was collected by Airbnb, Inc. The structure of this data set can be found in Appendix A.

Exploratory Data Analysis (EDA)

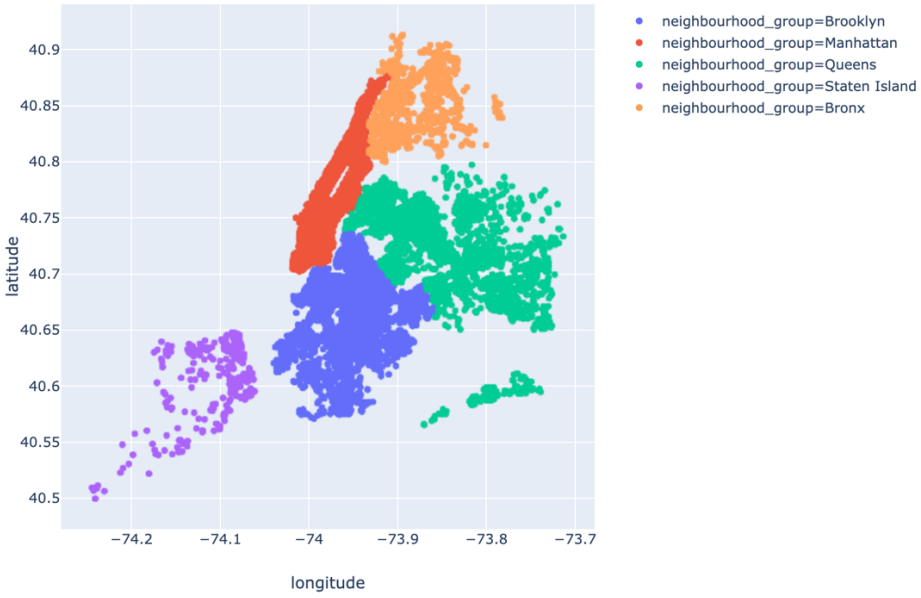

Figure 1: The Scatterplot of Latitude vs. Longitude

In Figure 1, we can see that the majority of listings are on the right side of the plot, and most listings are gathered together, but not spread out with a clear pattern. The listings in Manhattan are the most gathered, and the listings in Staten Island are scattered but more gathered by the Hudson River.

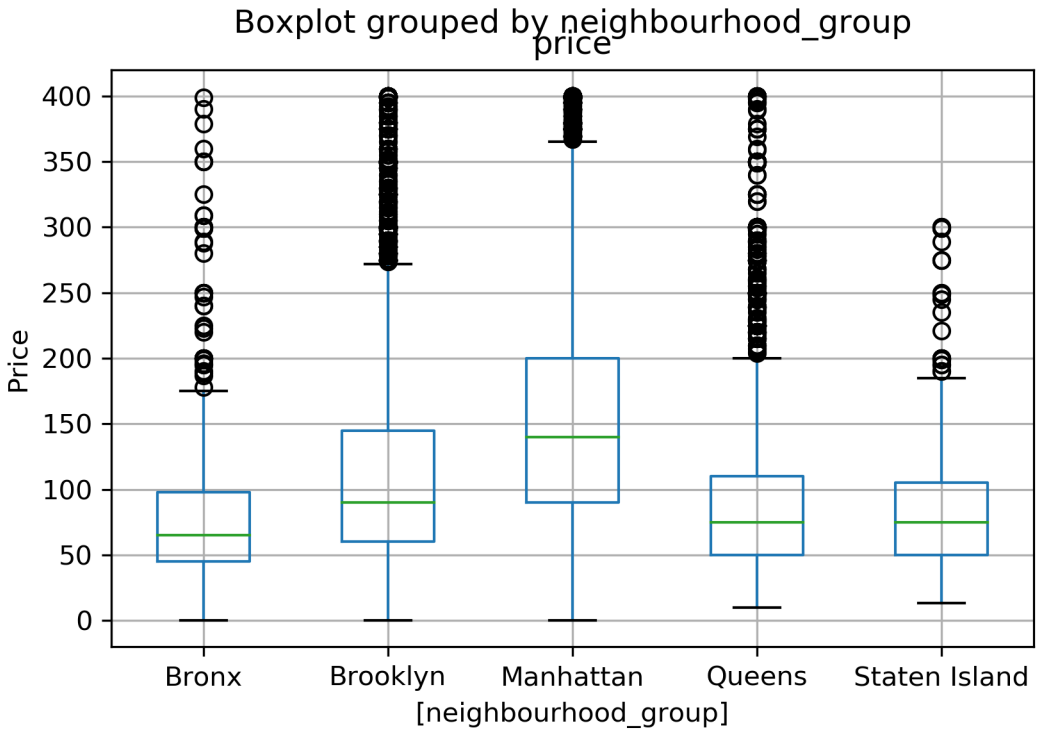

Figure 2: The Boxplot of Price vs. Region

In Figure 2, we can see the listing price per day in Manhattan is the highest, and the price in Brooklyn is the second highest, which intuitively makes sense. In addition, there are lots of extreme values that I need to keep in mind.

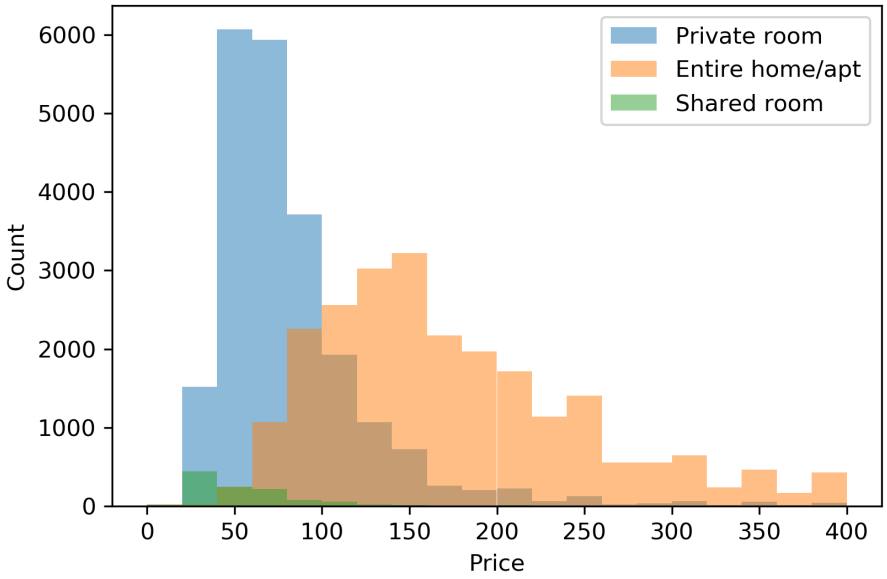

Figure 3: The Histogram of Price vs. Room Type

In Figure 3, we can see that there are relatively not many shared rooms, and they have the lowest price. The price of private rooms is right-skewed, and the entire home or apartment has the highest price.

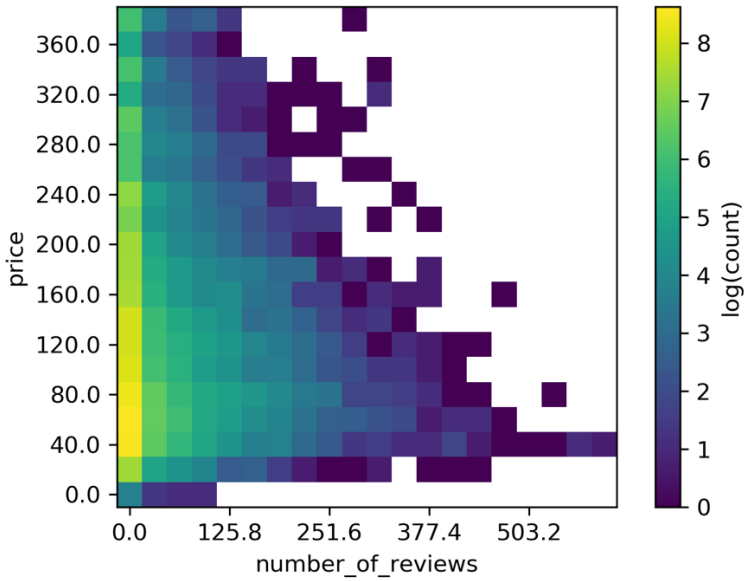

Figure 4: The Heat Map of Price vs. the Number of Reviews

In Figure 4, surprisingly, lots of listings have only a few reviews or no reviews at all. Notice that the count here is the logarithm of the actual count.

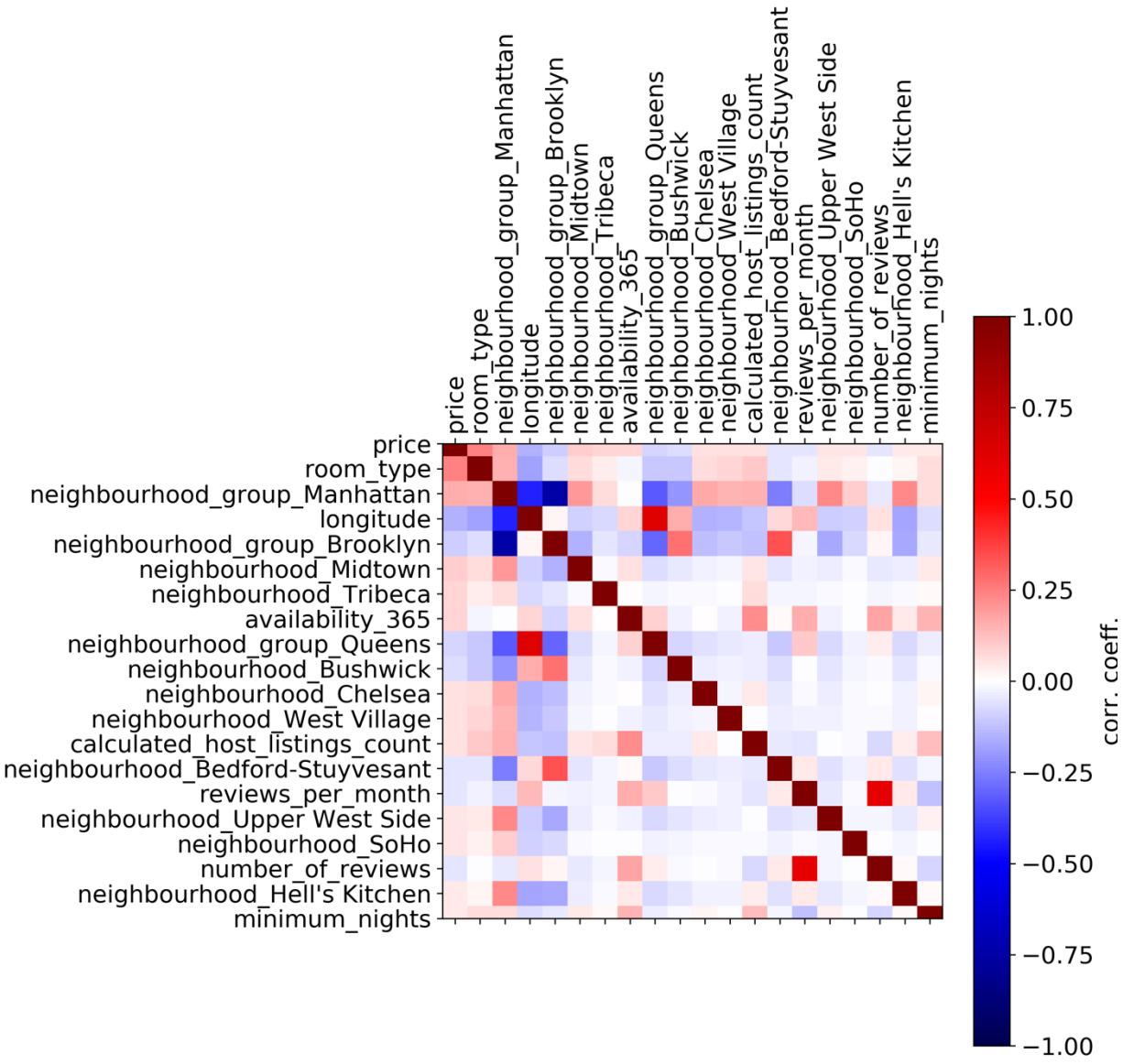

Figure 5: The Correlation Matrix with the Top 20 Correlated Features with the Target Variable

In Figure 5, we can see that there are not many strong interactions between the features. The three features that have the strongest correlation with the target variable “price” is room type, whether it is in Manhattan, and longitude.

Methods

Data Preprocessing

The columns “id” and “host_id" are removed. For “host_id”, I could have checked if a host has multiple listings, but another feature “calculated_host_listings_count” already did that for me.

Then, when I searched for missing values, I found the features date of the last review and reviews per month have missing values, but it only happens when the number of reviews is 0. For reviews per month, I simply set the missing values to 0 because when the number of reviews is 0, reviews per month is definitely 0. However, it is different for the date of the last review since it contains date strings. I set a reference date of July 9, 2019, which is one day after the latest date of review, and I calculated the days between the dates and the reference date and set the missing values to 0.

For the feature listing name, I calculated the length of the name, and I checked if there is an exclamation mark in the name. For the feature hostname, I checked if the length of the name is greater than 2 because a name with length greater than 2 is more likely to be a real name. Besides, for the target variable “price”, I found 11 observations with price 0, which does not make sense, so I deleted these observations. Since the prices have a wide range with a maximum of $10,000, I used the natural logarithm of prices in model fitting and prediction.

For categorical variables, I used a one-hot encoder on the neighborhood group, which contains Bronx, Brooklyn, Manhattan, Queens, and Staten Island, and I used an ordinal encoder on the room type: 0 for shared room, 1 for private room, and 2 for the entire home or apartment.

For numeric variables, I used a min-max scaler on latitude, longitude, and the number of days available in a year. I used standard scaler on name, minimum nights, the number of reviews, the date of the last review, reviews per month, and host listings count. Notice here the features name and the date of the last review are already numerical.

After all, the preprocessed data set has 48,884 observations and 18 features.

Machine Learning Pipeline

I developed a machine learning pipeline with a k-fold grid search cross-validation. First of all, it splits the data into train and test set with 20% in the test set. Then, the train set is divided into k folds (I used k = 5 for the entire project.) and each fold will be used as the test set in turn in cross-validation. I chose the mean-squared error (MSE) to evaluate the models’ performance because it can be easily converted to the root-mean-square error (RMSE), and RMSE explains the average error made by the model.

I trained 4 models in total. For least absolute shrinkage and selection operator (Lasso), I tuned the parameter α with values 1e-7, 1e-6, 1e-5, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, and 1,000. Cross-validation shows that $\alpha =$ 1e-5 is the best parameter for Lasso. The model has a cross-validation MSE of 0.25 with a standard deviation of 0.01 and a test MSE of 0.25.

For ridge regression, I tuned the parameter α with values 1e-7, 1e-6, 1e-5, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, and 1,000. Cross validation shows that $\alpha = 0.1$ is the best parameter for ridge regression. The model has a cross-validation MSE of 0.25 with a standard deviation of 0.01 and a test MSE of 0.25.

For the elastic net regularization, the model combines Lasso and ridge regression with an L1 ratio (the ratio of Lasso). I tuned the parameter α with values 1e-7, 1e-6, 1e-5, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, and 1,000 and the parameter L1 ratio with values 0, 0.05, 0.1, 0.15, …, 0.95, 1. Cross-validation shows that $\alpha =$ 1e-5 and L1 ratio $= 1$ are the best parameters for the elastic net. Since the L1 ratio here is 1, this model is indeed a Lasso model, and its parameter α is the same as the Lasso model I trained above, so these two models are actually the same model. The model has a cross-validation MSE of 0.25 with a standard deviation of 0.01 and a test MSE of 0.25.

For random forest, I tuned the parameter maximum depth with values 10, 30, 50, …, 290, 310 and the parameter minimum samples split with values 2, 4, 8, 16, 32, 64, 128, 256, and 512. Cross-validation shows that maximum depth $= 30$ and minimum samples split $= 16$ are the best parameters for the random forest. The model has a cross-validation MSE of 0.19 with a standard deviation of 0.01 and a test MSE of 0.2.

Uncertainties

Since the process of data splitting is random, there are some uncertainties due to splitting. By shuffling each feature, I calculated the shuffled test MSE for each feature in the random forest model. Most features have shuffled test MSEs close to the original test MSE, but the feature room type has a significantly high shuffled test MSE of 0.506 with a standard deviation of 0.005. In addition, the feature longitude has a relatively high shuffled test MSE of 0.296.

Random forest is an ensemble learning method that involves bootstrapping. Bootstrapping draws samples randomly with replacement, which creates uncertainty. To measure it, I used the same parameter space to train a random forest model, but with a different seed (random state). The tuning result is that the best maximum depth changes from 30 to 50, and the best minimum samples split stays at 16. Although the parameter is changed, the cross-validation MSE and test MSE are close to the previous random forest model.

Results

The results of the analysis suggest that a random forest model with maximum depth 30 and minimum samples split 16 will give us the best prediction. Before we compare the model to a baseline model, I would like to first explain the math behind the actual RMSE. Since I used the natural logarithm of prices in model fitting and prediction, the residual (e_i) is calculated as

$$e_i = \ln y_i - \ln \hat{y}_i$$

and it can be converted to

$$e_i = \ln \frac{y_i}{\hat{y}_i}$$
$$\exp(e_i) = \frac{y_i}{\hat{y}_i}$$
$$y_i = \hat{y}_i \exp(e_i)$$

so the exponential of the residual is the ratio of the actual price and the predicted price.

The coefficient of determination R^2 of the baseline model is 0, and the R^2 of the random forest is 0.82. The RMSE of the baseline model is 0.70, and $e^{0.70} = 2.01$, so the average ratio of the actual price and the price predicted by the baseline model is about 2. The RMSE of the random forest is 0.44, and $e^{0.44} = 1.56$, so the average ratio of the actual price and the price predicted by the random forest is about 1.56. By comparing the two ratios, we can see that the random forest model clearly explains more variance in the data and reduces the error significantly.

I also calculated the feature importance in the random forest model (see Appendix B). The feature room type has the highest feature importance of 0.47, followed by the features longitude and latitude, with feature importance 0.123 and 0.116, respectively.

Speaking of model performance, although random forest significantly reduces the error, an average ratio of 1.56 between the actual and the predicted price is still large. For example, if the predicted price is $100, the prediction can be off by 56% on average, or $56.

We can learn from the model that room type and location (latitude and longitude) are the key points to determine the listing prices. In real life, if people are traveling to New York City and have all the information needed about a place to stay, they may plug the corresponding values into the model and get a predicted price from the model as a reference, even though the

prediction may not be accurate. Besides, if a host in New York City wants to post his or her house on Airbnb, he or she can also use this model as a reference to set the listing price. Nevertheless, for either travelers or hosts, the predicted price should not be considered as a fair price since this is only the best model I can get from the available data, and the effect of this model is limited.

Outlook

In general, random forest lowers variance without causing too much of increase in bias. However, using random forest could cost more computational resources due to its complexity, and because of its complexity, the random forest is less intuitive compared to decision trees and not as easy to explain. To improve the model's interpretability, I could construct a decision tree algorithm to visualize data and what could happen in the forest. In addition, the random forest is a non-deterministic algorithm, which brings up different outputs in different runs. To ensure a consistent result, a seed must be set before running the algorithm, but the best seed is unknown. To improve this model, I can use cross-validation on different seeds and see which seed returns the smallest test MSE. Other than the models I trained, I could have used k-nearest neighbors (KNN) and gradient boosting. In terms of data, I could collect more information on the floor plan (number of bedrooms and bathrooms), the presence of front yard and back yard, water view, and the star (review) of each listing and host.

References

Dgomonov. “New York City Airbnb Open Data”. Kaggle, 2019, www.kaggle.com/dgomonov/new-york-city-airbnb-open-data

Dgomonov. “Data Exploration on NYC Airbnb”. Kaggle, 13 Aug 2019, www.kaggle.com/dgomonov/data-exploration-on-nyc-airbnb

Saurabh. “Complete Analysis of Airbnb Data - New York City”. Kaggle, 2019, https://www.kaggle.com/scsaurabh/complete-analysis-of-airbnb-data-new-york-city

Appendix

Appendix A – The Structure of the Data Set

Variable	Description	Type
id	listing ID	ID
name	name of the listing	String
host_id	host ID	ID
host_name	name of the host	String
neighbourhood_group	location	String
neighbourhood	area	String
latitude	latitude coordinates	Numeric
longitude	longitude coordinates	Numeric
room_type	listing space type	String
price	price in dollars	Numeric
minimum_nights	amount of nights minimum	Numeric
number_of_reviews	number of reviews	Numeric
last_review	latest review	Date
reviews_per_month	number of reviews per month	Numeric
calculated_host_listings_count	amount of listing per host	Numeric
availability_365	number of days when listing is available for booking	Numeric

Appendix B – Feature Importance in the Random Forest Model