# Predicting Income for AirBnB Listings

Zihan Ye
May 15, 2020

## 1 Introduction

The goal of this project is to create a model that predicts how much annual income an owner can expect from his/her listing on Airbnb. To do this, we will use datasets from InsideAirbnb.com, which scrapes publicly available information directly from the Airbnb website for major cities around the world, including listing information and the reviews for each listing.

The question we're interested in is: how much money can I (a property owner) make from listing my property on Airbnb? This serves several practical purposes. For example, an owner can use our model when deciding whether he should rent the property to a long-term tenant instead of listing the property on Airbnb, or, if he decides to rent the property to a long-term tenant, how much he should charge for rent to make it worth the opportunity cost.

## 2 Data

### 2.1 Data Description

As noted previously, all the datasets that we'll use from this project are downloaded from InsideAirbnb.com. According to the website, Murray Cox, an independent digital storyteller and community activist, compiled and analyzed the data and built the site for the purpose of letting anyone explore how Airbnb is really being used in cities around the world, and particularly how it's being used to compete with the residential housing market. [1] The data was collected by compiling publicly available information from the Airbnb website, such as listing information (description, price, location, etc.) and the reviews for each listing. The data is verified, cleansed, analyzed and aggregated by InsideAirbnb.com.

The website provides datasets for various cities around the world, but for this project, we will use datasets for New York City, as it is a big city with a large sample size of listings.

### 2.2 Datasets

Our variables of interest are average nightly price and occupancy throughout the year, as we can model income as a function of price and occupancy (more on that in Section 3).

There are 7 datasets available for each city: calendar.csv, listings.csv, listings.csv.gz, neighborhoods.csv, neighborhoods.geojson, reviews.csv, reviews.csv.gz. Each files contains the following information:

- listings.csv.gz: a 51097 x 106 dataset where each observation is a listing. The columns represent features such as neighborhood, reviews per month, host response rate, room type (ex. private room, hotel, entire home, shared room) and price, which is one of our variables of interest. Of the 106 variables, 42 are quantitative and 64 are categorical.
- reviews.csv.gz: a 1254654 x 6 dataset where each observation is a review for a given listing. For each review, we have 6 features: listing id, review id, review date, reviewer id, reviewer name, and the review text (comments).
- listings.csv: a 50599 x 16 dataset that's a subset of the full listings file (listings.csv.gz). Each row corresponds to a listing, but contains only 16 out of the 106 features available to us in the full dataset. As we are using the full listings dataset for our model, this subset of the full file won't be used for our model as it's unnecessary and redundant.
- reviews.csv: a 1255322 x 2 dataset that's a subset of the full reviews file (reviews.csv.gz). Each row corresponds to a review, but only contains the listing id that the review is for and the review date. As we are using the full

[1] http://insideairbnb.com/about.html

reviews dataset for our model, this subset of the full file won't be used for our model as it's unnecessary and redundant.

- calendar.csv: a 18650686 x 7 dataset where each row is the availability for a given listing on a given day for the next 365 days from when the dataset was compiled. For example, the dataset was compiled on 2/12/20, so we have availability information from 2/12/20 to 2/12/21. The 7 columns of this dataset are: listing id, date, available (a binary categorical variable indicating whether or not that listing was available on that day), price, adjusted price, minimum nights for stay, and maximum nights for stay.
- neighborhoods.csv: a 230 x 2 dataset where each observation is a neighborhood. Each row contains the neighborhood name as well as the "neighborhood group" that the neighborhood is in. For example, a row in this dataset would look like (Williamsburg, Brooklyn), where Williamsburg is the neighborhood and Brooklyn is the neighborhood region. As the listings dataset already contains the neighborhood and the neighborhood region for a given listing, we don't need to include this dataset in the model.
- neighborhoods.geojson: a geojson file for the neighborhoods in New York. We will exclude it from the model since the listings dataset already contains the information about the neighborhood that a listing is in.

In conclusion, we will only use 3 datasets for our model: the full listings dataset, the full reviews dataset, and the calendar dataset, since the other four datasets contain redundant information.

## 2.3 Data Quality

The overall data quality is good for the New York Airbnb datasets. The listings dataset has a fair amount missing values. Of the 106 available variables, 31 of them have more than 20% missing values, and this is something that we will address during the data cleaning process. The reviews dataset has about 5% missing values for review text, and the calendar dataset has negligible missing values (<.5%).

## 2.4 Important Variables

From these datasets, we have identified a few variables that we believe are critical for our model:

- price: this variable from the listings dataset is one of our variables of interest, and is the response variable for the models that we will fit.
- location variables: in the listings dataset, each listing has information on neighborhood (ex. East Village), neighborhood group (ex. Manhattan), longitude, and latitude. We hypothesize that the location of a listing has a significant impact on the price and occupancy of a listing. For example, a listing in Manhattan is likely to be more expensive than a listing in Staten Island, ceteris paribus.
- host-is-superhost: this variable in the listings dataset is a binary categorical variable indicating whether or not the host is a "superhost." A "superhost" is a prestigious title bestowed on experienced and well-reviewed by Airbnb, and we believe that the host being a superhost will increase both the price of a listing and the occupancy.
- reviews: there are many features that we can create from the reviews dataset that we hypothesize will have a strong influence on price and occupancy. For example, we expect the average review length for a listing to be correlated with price and occupancy. The average review length can speak to the quality of a listing as well as information available to a potential customer when deciding to rent the listing. Another example of an important feature derived from review text is sentiment score: we believe a higher sentiment score (more positive review) will have a positive impact on listing price.

## 2.5 Exploratory Data Analysis

First, we examined the distribution of average nightly price (Figure 1 in Appendix). It appears that most listings tend to be in the $50-$250 price range, with a few outliers. The distribution is right-skewed, with a mean of $139 and a median of $105 after excluding outliers. Excluding the outliers, the most expensive listing is $999.

Next, we examine how price is distributed across different room types. Figure 3 shows the distribution of the four different room types (Entire home/apartment, Private room, Shared room, and Hotel room) and Figure 2 shows the

price distribution by room type. Figure 2 shows that entire homes and hotel rooms tend to be more expensive and have more outliers, with the most expensive listings being over $1000 per night. On the other hand, the prices of private rooms or shared rooms rarely exceed $250 per night. One important note is that the data for listings is dominated by entire homes and private rooms, as hotel rooms and shared rooms combined take up less than 5% of the data. From this, we expect that room type will be an important feature in our model, as the distributions of price between entire homes and private rooms, for example, have different shapes.

# 3 Problem Statement

We model our parameter of interest, income, as a function of average nightly charge and occupancy rate throughout the year:

$$income = \overline{price}_{nightly} \times days_{available} \times occupancy_{rate}$$

$$\overline{price}_{nightly} = f(x_1 + x_2 + ... + x_k)$$

$$occupancy_{rate} = g(reviews, lengthofstay, reviewrate)$$

We will create models for nightly price and occupancy in a year and multiply the outputs of those models with the number of days the listing is available in a given year to predict annual income. We denote the models for price and occupancy as $f(\vec{x})$ and $g(reviews, lengthofstay, reviewrate)$, respectively, where the x's are all the features that we have to work with (p features in total), either given through the datasets or created by us.

To estimate $f(\vec{x})$, we will consider various linear models and machine learning algorithms when choosing the optimal model. We will discuss our choice of model for $f(\vec{x})$ in Section 5.3.

To model occupancy, we adopted the "San Francisco Model" used by the San Francisco Planning Department.[2] One way that occupancy over a period of time was estimated in the report was by finding the number of reviews over a period of time, multiplying by the average length of stay, and then dividing by the percent of guests that leave reviews (i.e. review rate, which is estimated). For example, if a listing got 20 reviews/year on average, and the average length of stay was 3 days, and we estimate that 50% of guests who stay in a listing leave reviews, then we have:

$$occupancy = min(20 \times 3 \div .5, 365)$$

Airbnb stated in 2012 that 72% of guests leave reviews, but we feel this is an overly enthusiastic estimate with little evidence to support the claim. According to InsideAirbnb, "using a review rate of 30.5% is more fact based", so we will estimate annual income conservatively by assuming only 30.5% of guests leave reviews. To get the occupancy rate over a year, we will divide the days of occupancy by 365.

$$occupancy_{rate} = \frac{min(reviews_{annual} \times \overline{length}_{stay} \div .305, 365)}{365}$$

# 4 Data Preprocessing

## 4.1 Data Cleaning

As noted earlier, we have significant missing data in the listings dataset. If a feature has NA's for more than 60% of the observations, then we excluded the feature from our model, as we don't expect that feature to contribute much information with so many missing values. For features with less than 60% missing values, we took the following approach to deal with NA's, based on the data type:

- Quantitative: replace NA by the mean of the column. Examples include "host response rate" and "cleaning fee".
- Categorical: convert all NA's into a new category called "Unknown". Examples include "host is superhost" and "neighborhood".

[2] The Executive Summary of Amendments Relating to Short-Term Rentals, page 8

- Text: if the column contains text data (i.e. descriptions of the listing, neighborhood, etc), we created a new binary variable indicating whether or not the observation had an NA for that column. Examples include ”neighborhood overview” and ”house rules”.

For the reviews dataset, we dropped the observations with no reviews (about 5%) and filtered out all non-English comments to facilitate sentiment analysis.

For the calendar dataset, we aggregated the dataset by listing_id. The most important column in the aggregated dataset is ”annual_availability”, which is the number of days available in the next 365 days, an important component of our model for annual income.

## 4.2 Feature Engineering

For the listings dataset, we modified/created certain features as follows:

1. The number of bedrooms for a listing is changed to 1 if the room type is private or shared.
2. The starting date of being a host is converted to the number of years of being a host.
3. The ”amenities” and ”host_verifications” features, which are text features, are converted into quantitative features representing the number of amenities in the listing and the number of different ways the host was verified, respectively.
4. For several text columns (ex. neighborhood_overview, host_about, transit), we generated features representing the length (in characters) of the original feature as a proxy for how detailed the descriptions are.

For the reviews dataset, we created four features based on the review text, in the hopes that these features will give us insights into the nature of the reviews and reflect certain qualities that may be helpful to our model:

1. nwords: the number of words on the comment
2. punc-prop: the proportion of punctuations used in a comment
3. >2 excl: whether or not there are more than 2 exclamation points in a row in a comment
4. propC: the proportion of capitalized letters in a comment.

Additionally, we generated sentiment scores for the review text using the vaderSentiment package, which analyzes a piece of text by matching the words to a pre-built lexicon and generating scores for each word based on the intensity on both polarities (positive/negative). The resulting score takes a number between -1 and 1, where negative values indicate negative sentiment and positive values indicate positive sentiment. Since this relies on a highly generalizable and human-validated lexicon, the bias we potentially introduce using this method is minimal.

## 4.3 Finishing Touches

After joining the cleaned listings, reviews, and aggregated calendar datasets (by listing id), we did the following to prepare our dataset for model fitting:

1. Remove features that don’t intuitively impact price or occupancy (ex. listing id, host name).
2. Remove redundant features: for example, ”street”, ”neighbourhood”, ”neighborhood_cleansed” all have values like ”Brooklyn”, so we only kept ”neighborhood_cleansed” and dropped the other two features.
3. Encode binary categorical features as 0 or 1.
4. Create dummy variables for categorical features with multiple classes. For categorical features with high cardinality, we used likelihood encoding, which essentially encodes each class of a categorical feature with the mean target value within that class, given some regularization. This allows us to create labels that are directly correlated with the target while avoiding creating too many dummy variables.

# 5 Model Fitting

## 5.1 Who Will Our Model Work For?

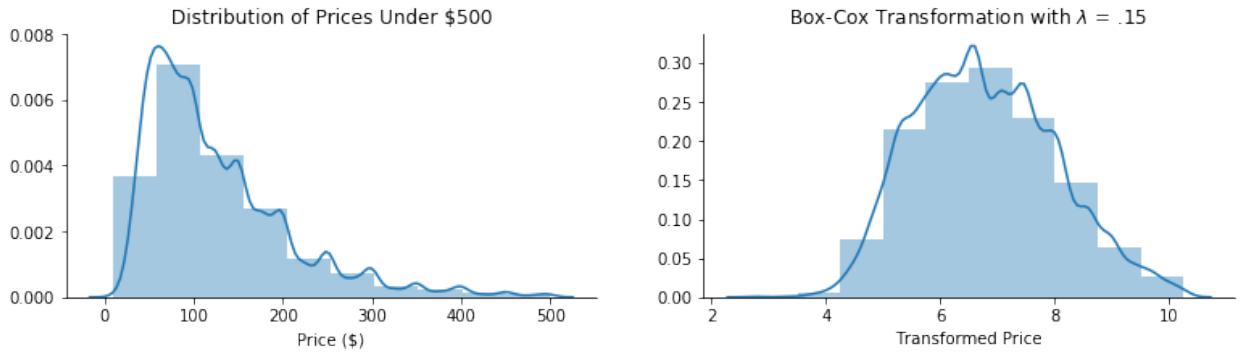
In order to make a reasonably accurate model for price, we will remove some outliers. Since a vast majority of the listings in the dataset are under 500 dollars, we will remove all observations over 500 dollars. This means that our model will not predict well for very expensive listings, which we will accept because those listings only account for about 1.7% of the data. Furthermore, our final goal is to predict annual income, and we imagine occupancy becomes very unstable for a listing that's particularly expensive.

Furthermore, we noted earlier that a vast majority of the listings belong are "entire homes/apartments" or "private rooms", so we imagine our model will be able to predict those two room types best.

Combining these two facts, we believe that our model will work best for non-luxury listings in New York (the kinds of listings that don't cost $500+ per night) that are "entire homes/apartments" and "private rooms", which are the room types that dominate our dataset.

## 5.2 Transforming Price

We saw earlier that the distribution of listing prices is highly right skewed even after removing the outliers. Therefore, we used a Box-Cox transformation with $\lambda = .15$ to make the distribution of the response variable more normal, which should help improve model performance. Here is the result of transforming price:




Note that since our model output will now be on a different scale, we need to transform the output back to the original scale by inverting the Box-Cox transformation.

## 5.3 Model Performance

To get a model that best predicts price, we fit the following models on our data: OLS, LASSO, Random Forest, and Gradient Boosting Machine. We chose to fit the first 2 models because of interpretability: the OLS coefficients would tell us how much each feature impacts price and the LASSO coefficients would give us a sense of feature importance. We chose to fit the latter two models because while they aren't as interpretable, they tend to do better for prediction.

Since this is a prediction problem, we will choose the best performing model in terms of test RMSE and use that for our annual income, but we will also look at the other models to get a sense of feature importance. Where applicable, We did a grid search with 5-fold cross validation to select the optimal hyperparameters for a given model. Here are the results:

| Model | OLS | LASSO | Random Forest | GBM |
|---|---|---|---|---|
| Test RMSE | 47.73 | 56.84 | 45.02 | 43.31 |

Since Gradient Boosting gave us the best test RMSE, we will use the GBM model to predict price.

## 5.4 Feature Importance

While we chose the GBM as our final model, we believe that the other models can still provide valuable insight into the features in our dataset. To better understand feature importance, we looked at the 10 most important features selected by the LASSO and random forest models.

For LASSO regression, we estimate $\beta$ as the set of regression coefficients determined by:

$$\hat{\beta}^{Lasso} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta x_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

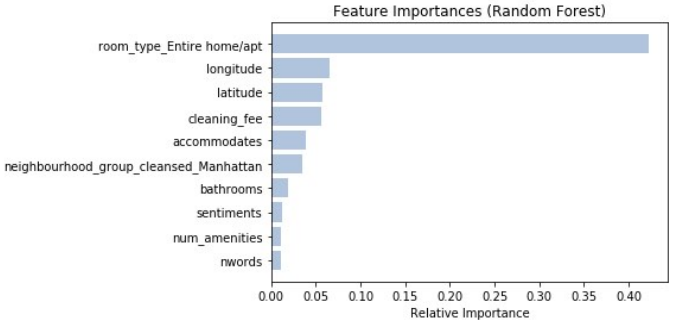
The penalty term naturally drives coefficients to 0, so given a large enough regularization term $\lambda$, we end up with k features, which we can interpret as the "most important" features to our model, since their contribution to the minimizing the squared error loss (first term in the loss function) outweigh their cost (second term in the loss function).

In order to make inference with the LASSO coefficients, we checked the assumptions of the model. The model assumes linearity between the outcome and the features, homoskedasticity, and that the errors have mean 0. We verified that these assumptions were reasonable after fitting the model. Figure 4 shows the residual plot, which is evenly spread out and has no pattern. Since we have such a large sample, the normality assumption for the errors is not needed for asymptotic inference.

As mentioned earlier, the regression coefficients from LASSO gives us a sense of feature importance. The random forest model also allows us to get a sense of feature importance via Gini impurity. Here are the top 10 most important features, according to the two models:

| | Feature | Coefficient |
|---|---|---|
| 0 | room_type_Entire home/apt | 0.792326 |
| 1 | accommodates | 0.319836 |
| 2 | neighbourhood_group_cleansed_Manhattan | 0.315307 |
| 3 | cleaning_fee | 0.198945 |
| 4 | review_scores_location | 0.064935 |
| 5 | bedrooms_new | 0.025061 |
| 6 | guests_included | 0.021806 |
| 7 | num_amenities | 0.015814 |
| 8 | calculated_host_listings_count_shared_rooms | -0.036201 |
| 9 | longitude | -0.184456 |

(a) LASSO




(b) Random Forest

According to the LASSO model, the most important feature for predicting price is whether or not the listing is an entire home/apartment, which makes sense since those tend to be more expensive than shared rooms or private rooms. Other important features include whether or not the listing is in Manhattan, how many people the listing can accommodate, and number of bedrooms.

With the impurity-based feature importance, the most important feature by far is still whether or not the listing is an entire home/apartment. The results are similar for the most part, but one notable difference is that the sentiment scores and the number of words in a review were very important features according to the impurity-based feature importance, which means reviews can have a significant impact on price.

## 5.5 Modeling Annual Income

To recap, we are modeling annual income as follows:

$$income = \overline{price}_{nightly} \times days_{Available} \times OccupancyRate$$

Since we now have a model for average nightly price, we obtained the other two components in the following ways:

1. Days Available: the calendar dataset contains information on availability for the next 365 calendar days.
2. Occupancy Rate: we modeled occupancy rate as described in Section 3, using a review rate of 30.5%. The listings dataset contains a feature called number_of_reviews_ltm, which is the number of reviews written in the last 12 months, and a feature called 'minimum_nights_avg_ntm', which is the average minimum nights one needs to rent the listing for over the next 12 months, and we believe this is a good conservative estimate for average length of stay.

With that, we have all the components we need to model annual income. Here is what the output of our model looks like:

| | Listing ID | Annual Income |
|---|---|---|
| 0 | 2595 | 18716.753220 |
| 1 | 3831 | 9553.835681 |
| 2 | 5099 | 0.000000 |
| 3 | 5121 | 6677.003798 |
| 4 | 5178 | 15175.262589 |
| 5 | 5203 | 0.000000 |
| 6 | 5238 | 375.908251 |
| 7 | 5441 | 788.277703 |
| 8 | 5803 | 16509.616297 |
| 9 | 6021 | 12822.678313 |

Note that the annual income estimates are highly dependent on availability. For example, some listings with no availability for the upcoming year will yield an annual income of $0, which is reasonable since a listing with no availability should not generate any income. As another example, listing 5441 from the output above has a predicted annual income of $788.28, which is reasonable given it's only available for rent for 10 days of the year.

# 6 Discussion

Through our analysis, we were able to get a sense of the most important features in the dataset as well as a model that predicts price relatively well. Whether or not the listing is an entire home/apartment is by far the most important feature in our dataset. The number of people the listing can accommodate, the cleaning fee, the number of bedrooms and bathrooms, and location-related features (whether or not the listing is in Manhattan, longitude and latitude) are also important features. We were happy to see that some of the features that we generated, such as the sentiment scores and a feature indicating the average number of words in a review, were deemed as important features, too.

That being said, it's important to note the limitations of our model. Again, we believe our model works best for non-luxury listings (i.e. listings that cost $0-$500 per night) in New York City, since the listings that we used for modeling price had those characteristics. We imagine the pricing dynamics will vary across different cities around the world, so to predict annual income for listings in other cities, we would need to build separate models using datasets for listings in the city of interest. Our best performing model for price resulted in an RMSE of about $43. There is definitely room for improvement in terms of performance (ex. incorporating seasonality), but overall we are happy with the results.

One thing to keep in mind is that our estimates for annual income are conservative in nature. For example, when modeling occupancy, we used a review rate of 30.5% (whereas Airbnb stated in 2012 that 72% of guests leave reviews) and we used the minimum required nights of stay as a proxy for average length of stay, which means the annual income from a listing can be much higher than estimated. However, as the motivation behind this model is to give property owners a tool to help decide what to do with the listing (ex. rent it to a long-term tenant instead, how much to price the monthly rent, etc), we feel it's better to be conservative than overly enthusiastic.

# 7 Appendix

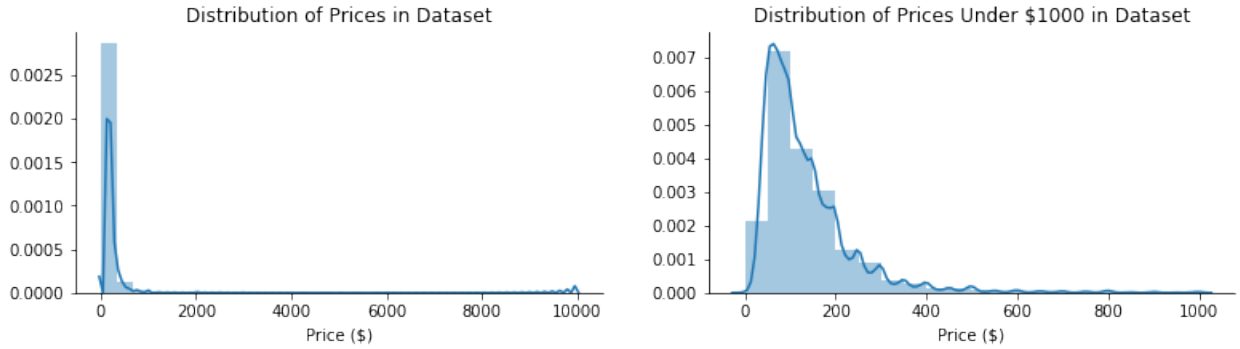



Figure 1: Price Distributions

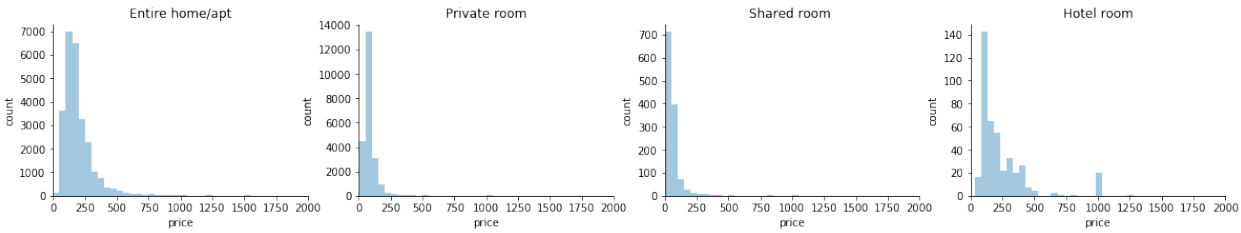
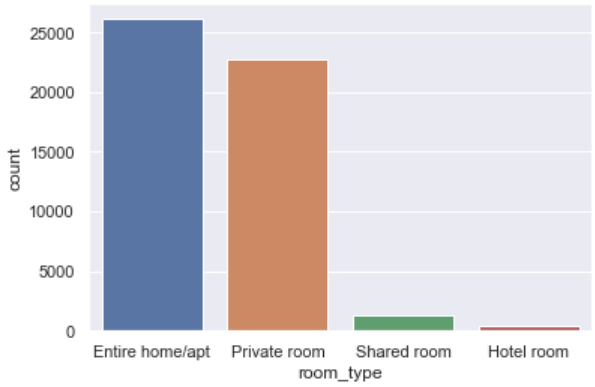
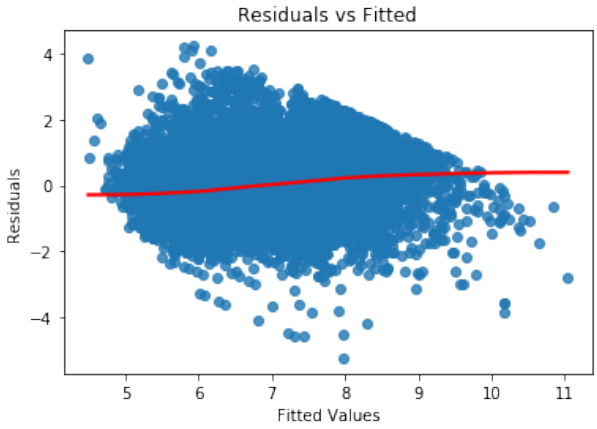



Figure 2: Price Distributions Across Different Room Types




(a) Figure 3: Distribution of Room Types




(b) Figure 4: Residual Plot for LASSO