Università degli Studi di Trento

Department of Economics and Management

Master of Science in Economics

Master Thesis

# Using Machine Learning for Airbnb Price Prediction

Author:
Duc Tuong Vu

Supervisor:
Prof. Marco Bee

Academic Year 2020/2021

# Acknowledgments

I would first like to pay special regards to my thesis advisor Professor. Marco Bee of the Department of Economics and Management at the University of Trento. The door to Professor Bee's office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my work but steered me in the right direction whenever he thought I needed it.

Some special words of gratitude go to my friends who have always been a significant source of support when things get a bit discouraging: Federico, Nick, Ivan, Thai, Giang, Thuy, Hong. Thank you guys for always be there for me.

Nobody has more important to me in the pursuit of this project than the members of my family. I would like to thank my parents for their great love and understanding. To my brother Lieu, you are always in my heart, and I hope you get well soon. Most importantly, I wish to thank my brother Ly for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

In recent years, the so-called sharing economy has spread all over the world. People share access to resources such as their homes, cars, and even personal time with other people in a peer-to-peer manner. One of the triumphant stories of the sharing economy in the travel and hospitality industry is Airbnb. Airbnb is a company that pioneers in deploying this business model to let individuals' share' extra space in their homes with those who need temporary accommodation via the online marketplace. In 2017, Airbnb generated approximately \$93 million in profit out of \$2.6 billion in revenue (Zaleski, 2018). As of 2020, there are 2.9 million hosts on Airbnb, with over 7 million accommodations in 100,000 cities worldwide (Airbnb 2020).

An accurate prediction of the rental price helps maximize the Airbnb host's profits and develop their business. So far, however, little attention has been paid to empirical Airbnb rental price prediction. This indicates a need to have an Airbnb rental price prediction model to address the research gap.

New York City (NYC) is one of Airbnb's most popular destinations with over 50,000 listings and ranked fourth in the most popular cities for booking experiences (Airbnb, 2020). With plenty of listing and booking activities, New York City serves as an excellent example for the study of Airbnb pricing.

## 1.2 Objectives

This study systematically reviews the data for Airbnb in New York City, aiming:

1. To employ machine learning techniques to build a model to predict Airbnb listing price in New York City.

2. To identify which features of an Airbnb listing are most important in predicting the price

## 1.3 Research Methodologies

This thesis contains four main research methodology:

1. Relative to collecting data method, we scrape the information about Airbnb listing from the InsideAirbnb website.
2. The data preprocessing step includes cleaning, data filtering, normalization, transformation, etc. The product of this step is the final training set ready for model fitting.
3. Exploratory analysis: we use visual methods to perform initial investigations on data to discover patterns and trends.
4. Model fitting: We applied several machine learning algorithms to make Airbnb rental price prediction. The comparisons between the prediction performance of those models are presented in chapter 3.

## 1.4 Thesis Outline

The overall structure of the study takes the form of four chapters. A short description of the chapter is listed as follows:

- Chapter 2 begins by reviewing the past literature on the pricing in the sharing economy based accommodation rentals. Then, we will briefly introduce machine learning concepts and considerations when applying machine learning to predictive modeling. Furthermore, we describe the machine learning algorithms used to solve the research question.
- Chapter 3 - Data Analysis is concerned with the methodology used for this study and the results we have obtained throughout our analysis.
- Chapter 4 - Conclusions and Future Works: In this chapter, we will answer the research question based on the findings obtained in chapter 3. Moreover, we will state some possible future adjustments to improve our results in this thesis.

# Chapter 2

# Background

## 2.1 Related Works

As discussed in Section  1.2, our primary focus is to answer the prediction question. However, besides the literature on using machine learning techniques for price prediction, we first briefly review the related works on the inference side.

### 2.1.1 Inference

While some research has been carried out on hotel price determinants, few empirical investigations into the price determinants of the non-hotel accommodation offer have been conducted. Furthermore, most researchers investigating the price determinant of sharing economy based accommodation have utilized the hedonic price model.

Monty and Skidmore (2003) assessed the price determinants of bed and breakfast amenities and confirmed the positive effects of a hot tub, a private bath, and a larger room on room price. In an analysis of the impact of private tourist accommodation facilities on prices, Portolan (2013) found that the presence of free parking places and sea view could be associated with a higher room rate. Both of the above studies recognized that location has a vital influence on the price.

In recent years, there has been an increasing amount of literature on the price determinants of Airbnb. Several studies (Gutt and Herrmann (2015); Ikkala and Lampinen (2014) ) have shown that hosts who offer accommodation to rent on Airbnb.com usually charge higher prices if their accommodation has received high star ratings. Using a dataset from New York City, J. Li et al. (2016) showed that properties managed by professional hosts earn more in daily revenue, have higher occupancy rates, and are less likely to exit the market than properties owned by nonprofessional hosts. Kakar et al.

(2016) measures the impact of information on hosts' racial background on Airbnb listing prices in San Francisco and found that Hispanic hosts and Asian hosts, on average, have a lower list price relative to their white counterparts.

Wang and Nicolau (2017) analyzed the Airbnb data from 33 countries and concluded that 24 out of 25 variables within five categories (host attributes, site, and property attributes, amenities and services, rental rules, and the number of online reviews and ratings) are good predictors of price. In a similar study, Cai et al. (2019) examines the impacts of five groups of explanatory variables on Airbnb price in Hong Kong.

### 2.1.2 Prediction

To date, there is little published research on applying machine learning techniques to predict the price of Airbnb listing. Tang and Sangani (2015) work on the task of price prediction for San Francisco Airbnb listings by turning the regression problem into a binary classification problem. Y. Li et al. (2016) has attempted to create a price prediction model for Airbnb in different cities by using the clustering method with the distance of the property to the city landmarks as the clustering feature. In more recent work, Kalehbasti et al. (2019) try to develop a reliable price prediction model using machine learning, deep learning, and natural language processing techniques.

This study aims to contribute to this growing area of research by utilizes a holistic approach. In particular, the contribution is as followed:

- We rely on "domain knowledge" in the feature engineering process (as demonstrated in Chapter 3).
- We make use of visualization techniques to gain insights from data.
- We introduce regularization techniques to improve the disadvantage of the traditional ordinary least square model.
- We present the Boosting modeling strategy, a machine learning approach that performs well when there are many predictors as a baseline model.

## 2.2 Machine Learning

Machine learning refers to creating and using models that are learned from data. Typically, our goal will be to use existing data to develop models that we can use to predict various outcomes for new data. Examples of machine learning applications can be found every where. Credit card companies use it to track fraud. The movie recommendation system is used by Netflix to suggest movies to subscribers. Insurance companies use it to predict the risks of potential auto, health, and life policyholder.

Most machine learning problems fall into one of two categories: supervised or unsupervised. The details of machine learning outline are shown in 2.1.

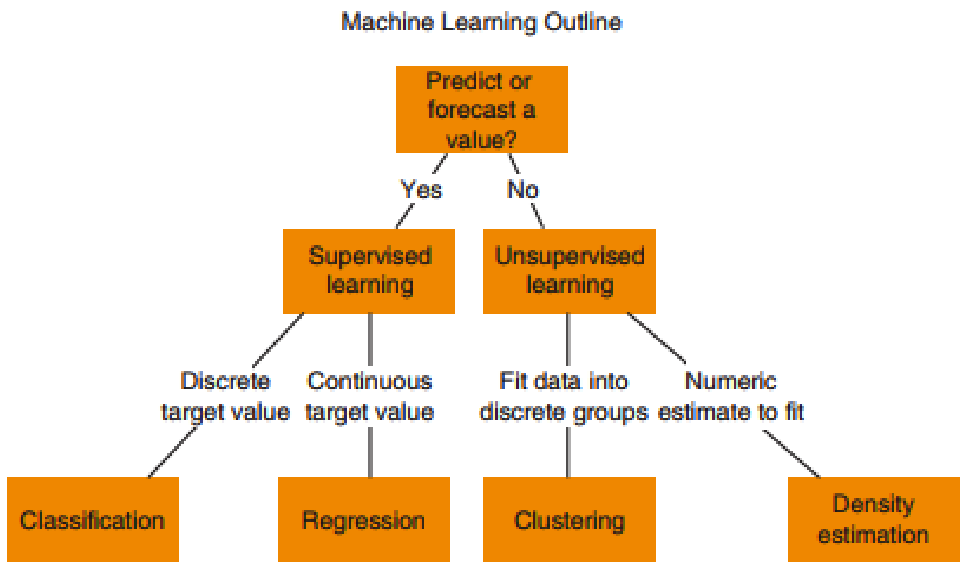


**Figure 2.1:** Machine Learning Outline

Source: Shah (2020)

### 2.2.1 Supervised Learning

If our goal is to predict a specific outcome by learning how various predictors and the response variables relate, we are dealing with **supervised learning**. If the response variable is continuous, the problem becomes that of *regression*. If, on the other hand, the response variable is discrete, this becomes a *classification problem*. For instance, predicting the age of a person, predicting house prices are regression problems, while predicting the gender of a person, predicting whether a credit card transaction is fraudulent are classification problems.

### 2.2.2 Unsupervised Learning

Unlike supervised learning, **Unsupervised learning** is a type of machine learning that looks for hidden patterns in a data set with no output labels and minimal human supervision. One of the most common unsupervised learning task is clustering, where we group similar things together. For example, we can group potential customers into different groups based on their characteristics, such as income, age, gender, and shopping habits; we can group news articles into different themes. if we are trying to explain the data by estimating underlying processes that may be responsible for how the data is distributed, this becomes a density estimation problem.

## 2.3 Problem Formalization

The problem of predicting the rental price can boil down to approximate a target function `f` for the output variable rental price (Y) based on a set of predictors such as `bathrooms`, `accomodates`... We can describe the relationship between `price` (`Y`) and its predictors $X = (X_1, X_2, ..., X_p)$ as followed:

$$Y = f(X) + \epsilon \tag{2.1}$$

The price of a listing can be predicted by:

$$\hat{Y} = \hat{f}(X) \tag{2.2}$$

## 2.4 Quantitative Measures of Performance

Selecting the best method among many different statistical learning approaches can be one of the practitioners' most daunting tasks. One particular approach may work best on a particular data set, but some other methods may perform better on a similar but different data set. Therefore, it is critical to select which method produces the best results for any given data set. To assess a particular model's predictive performance on a given data set, we need some way to measure how well its predictions match the observed data. In this study, we use two standard criteria: mean squared error and coefficient of determination.

### 2.4.1 Mean Squared Error

When an outcome is a number (regression problem), the most commonly used method for characterizing a model's predictive capabilities is to use the mean squared error (MSE), which is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{2.3}$$

where $\hat{f}(x_i)$ is the prediction that $\hat{f}$ gives for the $i^{th}$ observation. If the predicted responses are very close to the actual responses, the MSE will be small and vice versa.

In general, we do not care how well the method works on the training data. Instead, we are interested in the accuracy of the test MSE predictions we obtain when applying our method to previously unseen test data. Therefore, we want to choose the approach that gives the lowest test MSE instead of the lowest training MSE.

### 2.4.2 Coefficient of Determination

We also use the coefficient of determination ($R^2$) to measure the proportion of the information in the data explained by the model. For example, an $R^2$ value of 0.8 means that the model can explain 80 percent of the outcome's variation. An $R^2$ of 1 indicates that the regression predictions perfectly fit the data. It should be noted that $R^2$ is a measure of correlation, not accuracy. $R^2$ is calculated as the correlation coefficient between the observed and predicted values and squares it:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

where $SS_{res}$ is the sum of squares of residuals and $SS_{tot}$ is the total sum of squares (proportion to the variance of the data)

## 2.5 Variance-Bias Tradeoff

From James et al., 2013, the expected test MSE, for a given value $x_0$, can be broken down into three parts as followed:

$$E(y_0 - \hat{f}(x_0))^2 = \text{Var}(\epsilon) + [\text{Bias}(\hat{f}(x_0)]^2 + \text{Var}(\hat{f}(x_0)) \tag{2.4}$$

The first term is the irreducible error, which cannot be reduced regardless of what algorithm we use. It is a measure of the amount of noise in our data. No matter how good we make our model, our data will have a certain amount of noise or irreducible errors that can not be removed.

The second term is the model's squared bias, reflecting how close the target function (`f`) can get to the real relationship between the predictors and the outcome. A model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to increased errors in training and test data. An example of a high bias model is linear algorithms. While having a high bias makes them fast to learn and more interpretable, they are generally less flexible. As a result, they usually have a poor predictive performance on complex problems that fail to satisfy the simplifying assumptions of the linear regression algorithm.

Variance is how much the value of target function (f) will vary if we use different training data. In contrast to high bias algorithms, models with high variance focus too much on the training data and do not generalize the data it has not seen before. Consequently, such models may perform very well on training data but have high test prediction errors. It is generally true that more complex models can have very high variance, which leads to *overfitting*, which essentially means they follow the errors or noise too closely. Choosing an overly simple model might lead to *underfitting*, which means it will not learn the data's underlying structure, so its prediction is bound to be inaccurate, even on the training data.

As can be seen in equation 2.4, minimizing the test MSE means reducing the combination of bias and variance. Ideally, we want to have a model that has low bias and low variance. Unfortunately, it is typically impossible to do both simultaneously. If our model is too simple and has very few parameters, it may have high bias and low variance. On the other hand, if our model is too complicated, we start focusing too much on each data point in our training set, it will have high variance and low bias.

There is no escaping the relationship between bias and variance in machine learning. Therefore, this study's strategy is to try various machine learning algorithms with different variance-bias tradeoff levels to decide which is the best model to achieve a low bias and low variance model. In other words, we seek to find a sweet spot in between the variance bias spectrum that will yield the best generalization performance. These algorithms will be described in the next section.

## 2.6 Models and Algorithms

### 2.6.1 Linear Regression

Linear regression is the simplest and earliest predictive method. Multiple regression analysis is the foundation of Hedonic price modelling which has been widely used in real estate, tourism, and hotels to explore the relationship between a product's price and its characteristics (Rosen, 1974).

Hedonic pricing theory states that a product's price can be regarded as a function of the product's measurable, utility-affecting attributes or characteristics. Accordingly, We can decompose Airbnb accommodation into features that impact the overall product's quality and provide consumers with value and satisfaction. These may include the number of bedrooms, the number of capacities, number of reviews it receives, and amenities. We can specify the hedonic price function of Airbnb listings as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \epsilon \tag{2.5}$$

where $X_j$ represents the $j^{th}$ predictor and $\beta_j$ quantifies the association between that variable and the response

Given estimates $\hat{\beta}_0$,$\hat{\beta}_1$,...,$\hat{\beta}_p$ , we can make predictions using the formula:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + ... + \hat{\beta}_p X_p \tag{2.6}$$

The parameters are estimated using the least-squares approach in multiple linear regression. We choose $\beta_0$, $\beta_1$, . . . , $\beta_p$ to minimize the sum of squared residuals.

$$\begin{aligned} \text{RSS} &= \sum_{i=1}^{n} (y - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - ... - \hat{\beta}_p x_{ip})^2 \end{aligned} \tag{2.7}$$

Given some minimal premises about the distribution of the residuals [1], the hedonic regression parameter estimates that minimize SSE are the ones that have the least bias of all possible parameter estimates (Graybill (1976) ). Therefore, these estimates minimize the bias element of the bias-variance trade-off.

While having a distinct advantage of highly interpretable, the hedonic regression model suffers from overfitting (Harrell Jr, 2015). In a situation in which there is little or no theory available to guide on selecting the predictors, we want to include in our model as many features as possible. However, we run the risk of including irrelevant features that actually have no relation to the dependent variable being predicted, thus might not improve the predictive power on future data.

### 2.6.2 Penalized Regression Models

Overfitting frequently occurs in the regression setting (ibid.). While a model with many explanatory variables that have no relation to the response might increase the model's performance on the training data, it might not generalize well on the validation dataset.

To lessen the effect of overfitting, we use regularization techniques, which essentially constrain or regularize the coefficient estimate towards zero. Consider the case of the

[1]The error terms are independent, uncorrelated and normally distributed with a mean of zero and constant variance (a.k.a. homoskedasticity)

linear model with two parameters $\theta_0$ and $\theta_1$. This gives the learning algorithm two degrees of freedom to accommodate the model to the training data. If we forced $\theta_1 = 0$, the algorithm would have only one degree of freedom and would find it more difficult to fit the data properly. If we allow the algorithm to shrink the $\theta_1$ to a small amount, the learning algorithm degrees of freedom will be between one and two. It will produce a simpler model than one with two degrees of freedom, but more complex than one with just one. Either way, we can achieve a simpler model that can generalize well to unseen data. Therefore, applying regularization can significantly reduce the model's variance. In this section, we present the two best-known methods for regularization, ridge regression, and the lasso.

### Ridge Regression

Ridge Regression (Hoerl and Kennard (1970)) regularizes the parameter estimates by adding a penalty to the MSE. The algorithm is very similar to least squares, except that the coefficients ridge are estimated by minimizing a slightly different quantity. In particular, the ridge regression coefficient estimates are the values that minimize:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{2.8}$$

In Equation 2.10 we made a trade-off between two different criteria. As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small. But, the second term, $\lambda\sum_{j=1}^{p}\beta_j^2$ signifies that a second-order penalty (i.e., the square) is being used on the parameter estimates. This shrinkage penalty is small when $\beta_1$,...,$\beta_p$ are close to zero, and so it has the effect of shrinking penalty the estimates of $\beta_j$ towards zero.

When $\lambda = 0$, the penalty term has no effect, and ridge regression is the same as least squares estimates. However, as $\lambda \to \infty$, the influence of the shrinkage penalty increases, and the ridge regression will shrink the estimates towards zero. In contrast to least squares, which produces only one set of coefficient estimates, ridge regression will generate a different set of coefficient estimates, $\hat{\beta}_\lambda^R$, for each value of $\lambda$.

### Lasso Regression

One drawback of ridge regression that it does not set any of the parameter estimates equal to 0. Therefore, the model does not conduct feature selection. Least Absolute Shrinkage and Selection Operator (Lasso) (R. Tibshirani, 1996) model is a modern alternative to ridge regression that overcomes this disadvantage. The name comes from its functionality that it does not only shrink coefficients towards zero, but it also performs feature selection. The lasso coefficients, $\hat{\beta}^L$ , minimize the quantity:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p} \mid \beta_j \mid = RSS + \lambda\sum_{j=1}^{p} \mid \beta_j \mid \tag{2.9}$$

Notice that the lasso and ridge regression have similar formulations. The only distinction is in the penalty term. In particular, the $\beta_j^2$ term in the ridge regression penalty (6.5) has been replaced by $\mid \beta_j \mid$ in the lasso penalty The implication of this modification is that penalizing the absolute values has the effect of setting some of the coefficient estimates to be exactly equal to zero for some value of $\lambda$. Thus the lasso yields models that simultaneously use regularization to improve the model and to conduct feature selection.

We can formulate the lasso and ridge regression coefficient estimates in 2.9 and 2.10 as solving following problems:

$$\min_{\beta} \quad \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \mid \beta_j \mid \leq s \tag{2.10}$$

and

$$\min_{\beta} \quad \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s \tag{2.11}$$

In the figure 2.2, we illustrate why the lasso performs feature selection, while ridge regression does not.To simplify the problem, we consider only two parameters $\beta_1$ and $\beta_2$. The elliptic centered around $\hat{\beta}$ are contours of the sum of squares error with the OLS estimator in the center. All of the points on a given ellipse has the same value of RSS. The diamond and circle represent the lasso and ridge regression constraints in 2.9 and 2.10.

The lasso and ridge regression coefficient estimates are the first point at which an ellipse touches the constraint set. Note that the ridge regression has a circular constraint with no sharp points. This intersection will not generally occur on an axis. So the ridge regression coefficient estimates will be exclusively non-zero. However, the lasso constraint has corners at each of the axes, and so the ellipse will often intersect the constraint region at an axis. When this occurs, one of the coefficients will equal zero.In this way, the lasso performs *feature selection.*

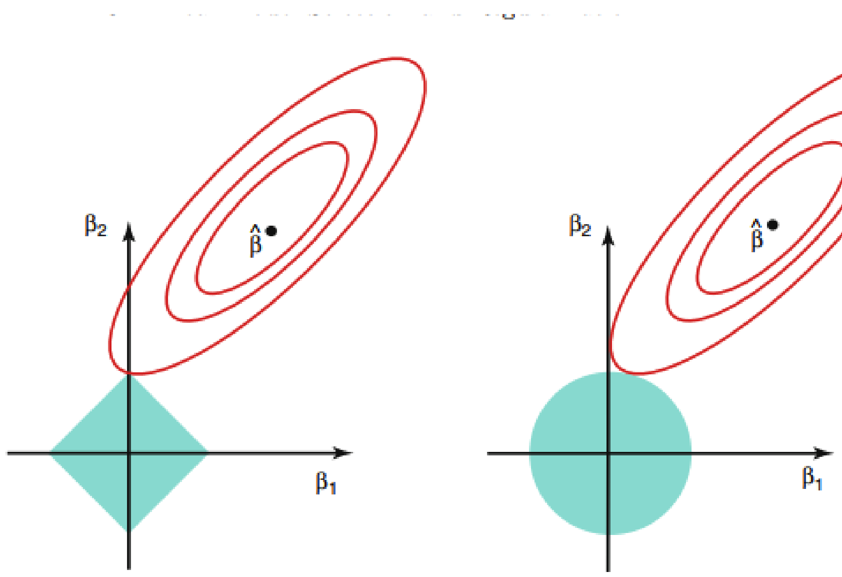


**Figure 2.2:** Illustration of two dimensional case of estimation for the lasso (left) and the ridge regression (right)

Source: James et al. (2013)

### Selecting the Tuning Parameter

Choosing an optimized value of tuning parameter $\lambda$ is critical when implementing the ridge and the lasso regression. We do not want to choose $\lambda$ too small due to the low restriction. On the other hand, when $\lambda$ is very large, the restriction is more substantial than it is desired. We handle the problem of the optimal value of by using a cross-validation technique. Efron and R. J. Tibshirani (1994) introduce the algorithm, which describes the procedure of cross-validation.

**Algorithm 1:** K-Fold Cross Validation

```
1 Split the data into K roughly equal-sized parts.
2 For the $k^{th}$ part, fit the model to the other K - 1 parts of the data, and calculate the
   mean squared error of the fitted model when predicting the $k^{th}$ part of the data.
3 Repeat step 2 for k = 1, 2, . . . , K and average the K estimates of mean squared
   error $MSE_1$, $MSE_2$,..., $MSE_k$.
4 For each tuning parameter value $\lambda$, compute the average error over all folds
```

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

We choose a grid of $\lambda$ values and calculate the cross-validation error for each value of $\lambda$. We then select the tuning parameter value for which the cross-validation error is smallest.

### 2.6.3 Boosting

All the approaches reviewed so far suffer from the fact that they use a single model for predicting the response variable. Hence the ability to choose a suitable model is crucial to have any chance to obtain good results. Machine Learning practitioners have to consider many factors such as the quantity of data, the dimensionality of the space, and distribution hypothesis to find a high predictive power model.

Boosting, on the other hand, is an an ensemble technique in which several weak models such as decision trees are combined to achieve a stronger one.The general idea of most boosting methods is to train predictors sequentially, each trying to correct its predecessor.

#### Regression Trees

Regression Tree is a type of decision trees used to predict continuous numeric values (e.g. the price of a house, or a patient's length of stay in a hospital). In a regression tree, each leaf represenet a numeric value.

A regression tree is recursively constructed through a binary partitioning process. We first select the predictor and the cut point s such that splitting the predictor space into the two regions (S1 = $X|X_j < s$ and S2 = $X|X_j > s$) such that the overall sums of squares error are minimized:

$$\text{SSE} = \sum_{i \in S_1}^{n} (y - \bar{y_1})^2 + \sum_{i \in S_2}^{n} (y - \bar{y_2})^2 \tag{2.12}$$

where $\hat{y_1}$ and $\hat{y_1}$ are the mean response for training set within groups S1 and S2, respectively.

We repeat the process, looking for the best predictor and best cutpoint to split the data further to minimize the sums of squares errors within each group. The recursive partitioning of the data continues until a stopping criterion reached for instance, we may continue until no region contains more than 20 observations. We predict the response for a given test data point using the average value of the training observations in the group to which that test observation belongs.

### Gradient Boosting

While having an advantage of being easily be visualized and understood, the main downside of decision trees is that they tend to overfit and provide poor generalization performance. Therefore, in most applications, the ensemble methods such as boosting are usually used in place of a single decision tree.

The underlying idea behind ensemble learning is that by aggregating the predictions of a group of predictors (such as classifier or regressors) we will ofter get better predictons than the best individual predictor (*wisdom of the crowd*). According to ensemble learning theory, weak learners are models that perform not so well by themselves either because they have a high bias or because they have too much variance. Then the idea of ensemble methods is to reduce bias and/or variance of such weak learners by combining several of them together to create a strong leader that achieves better predictive performance.

Boosting refers to any ensemble method for primarily reducing bias, and also variance by combining several weak learners into a strong learner. Being mainly focused at reducing bias, the base learners that are often considered for boosting are models with low variance but high bias such as shallow decision trees with only a few depths. Another reason for using shallow trees in boosting is that it makes the model smaller in terms of memory and makes prediction faster.

While there are many boosting methods available, in this study, we focus on Gradient boosting trees and its high-performance implementation, XGBoost. In gradient boosting trees, regression trees are chosen as the base learners.

Kuhn and Johnson (2013) illustrate the Gradient Boosting for Regression algorithm as followed:

**Algorithm 2:** Simple Gradient Boosting for Regression

```
1 Select tree depth, D, and number of iterations, K
2 Compute the average response, $\bar{y}$, and use this as initial predicted value for each
   observation in the training set.
3 for $k \leftarrow 0$ to $K$ do
4     Compute the residual, the difference between observed value and the current
        predicted value, for each observation.
5     Fit a regression tree of depth, D, using the residuals as the response.
6     Predict each observation using regression fit in the previous step.
7     Update the predicted value of each observation by adding the previous iteration's
        predicted value to the predicted value generated in the previous step.
8 end
```

We can find the the optimal value of tree depth (D) and the number of iterations (K) by using cross-validation techniques.

### XGBoost

XGBoost is an end-to-end tree boosting system developed by T. Chen and Guestrin (2016) based on a gradient boosting framework. XGBoost has been shown to provide state-of-the-art results for diverse problems, including web text classification, customer behavior prediction, motion detection, and malware classification (ibid.).

Nielsen (2016) shows some features of XGBoost that make it stand out of from the rest of other gradient boosting algorithms. That is:

- Clever penalization of trees
- A proportional shrinking of leaf nodes
- Newton Boosting
- Extra randomization parameter
- Implementation on single, distributed systems and out-of-core computation

Besides these superior features, there are other reasons why XGBoost is getting popular in the machine learning community:

- Can solve a wide range of applications such as regression, classification, ranking, and user-defined prediction problems.
- Portability: Runs smoothly on Windows, Linux, and OS X.
- Languages: Supports all major programming languages, including C++, Python, R, Java, Scala, and Julia.
- Cloud Integration: Supports AWS, Azure, and Yarn clusters and works well with Flink, Spark, and other ecosystems.

While XGBoost is suitable for predicting, it does so at the expense of its interpretability. As shown in Figure 2.3, compared to restrictive models such as Lasso or Least Squares, Boosting is a highly flexible (complex) approach that is harder to interpret.

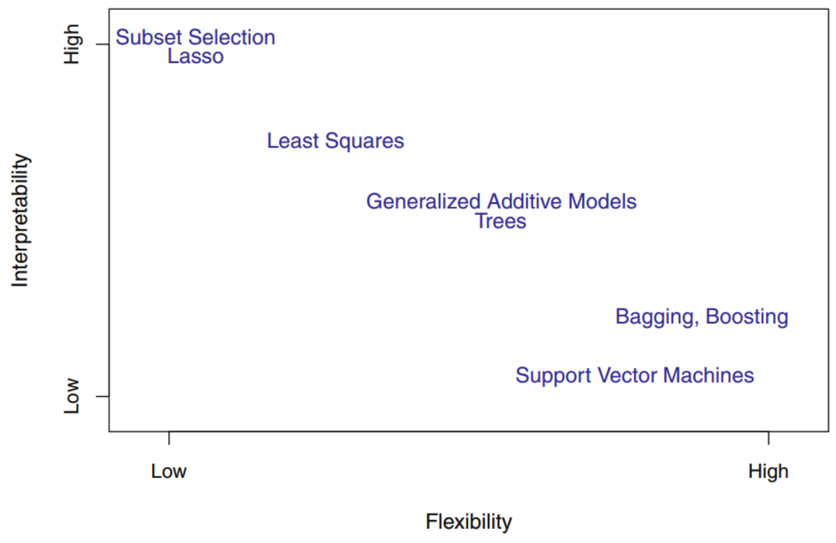


**Figure 2.3:** Interpretability and Flexibility Tradeoff

Source: James et al. (2013)

# Chapter 3

# Data Analysis

## 3.1 Data Acquisition

The dataset used for this study comes from the Inside Airbnb website *Inside Airbnb* (2019). This independent initiative collects Airbnb data for more than 30 major cities around the world that scrapes Airbnb listings, reviews, and calendar data from multiple cities worldwide. The dataset was scraped on December 4, 2019, contains information about the location (latitude and longitude coordinates) of all Airbnb listings in New York City and data about the hostname and ID, room type, price, minimum nights, number of reviews listings per host, and availability...

The raw data is quite untidy and has some weaknesses. The major one is that it only includes the advertised price (sometimes called the 'sticker' price). The sticker price is the overall nightly price advertised to potential renters, rather than the actual average amount paid per night by previous guests. A host can set the advertised prices to any arbitrary amount, and those that do not have much experience with Airbnb will often set these to unreasonable prices, such as too low (e.g., \$0) or very high (e.g., \$10,000) amounts.

These variables are listed and defined in Table A.2.

## 3.2 Data Preprocessing

Real-world data is often dirty; that is, it is in need of being cleaned up before it can be used for the desired purpose such as visualization, model building. This is often called data pre-processing. Since there are several reasons why data could be "dirty," there are just as various techniques to "clean" it. For this analysis, we will take a look at few key methods that illustrate ways in which data may be "cleaned," or better organized, or scrubbed of potentially incorrect, incomplete, or duplicated data.

### 3.2.1 Data Filtering

Following Wang and Nicolau (2017), this study filtered the listings with at least one online customer review to guarantee that our sample listings were "active." This view is supported by Ye et al. (2009) which confirms that online review ratings are associated with hotel-room sales, indicating that reviews suggest real transactions.

### 3.2.2 Removing Predictors

There are three reasons why we should consider removing predictors before modeling. First, fitting a model with fewer predictors reduces computational time and complexity. Second, removing highly correlated features makes the model more parsimonious and interpretable model. Lastly, some models can be crippled by predictors with degenerate distributions. Therefore, eliminating the problematic predictors prior to fitting a model can significantly improve model performance and stability.

- We remove predictors with a single unique value as they are zero variance predictors. These predictors are `has_availability`, `host_has_profile_pic`,`requires_license`, `is_business_travel_ready`, `require_guest_phone_verification`, `require_guest_profile_picture` (See A.18) .
- It is beyond the scope of this study to explore the natural language processing for predictive modeling. Therefore, we will drop free-text columns for now, as will other columns that are not useful for predicting price (e.g., `url`, `hostname`, and other host-related features unrelated to the property).
- We drop feature which adds relatively little information, or are relative unhelpful in differentiating between different listings.

### 3.2.3 Dealing with Missing Values

Missing data is a common issue in many data analysis applications. Data may be missing due to problems with the process of collecting data or equipment malfunction. Some data may get lost due to system or human error while storing or transferring the data.

There are two approaches we take to handle missing data:

1. Filling out missing value : we drop the predictors that contain a majority of null entries as in Table 3.1.

2. Filling in missing value:we perform data imputation, which means filling the missing data with some estimated ones. In particular,

    - Missing values in numeric features such as the number of bathrooms (`bathrooms`) bathrooms, the number of bedrooms (`bedrooms`), the number of beds (`beds`) will be replaced with the median.
    - For monetary features such as the amount required as a security deposit (`security_deposit`), the amount of the cleaning fee (`cleaning_fee`), the price per additional guest (extra_people), having a missing value for them is the same as having the value of $0, so the missing values will be replaced with 0. Similarly, an amenity that has a missing value will be replaced with 0.

**Table 3.1:** Columns with majority of null values

| | count | percentage |
|---|---|---|
| host_acceptance_rate | 50599 | 100.00 |
| jurisdiction_names | 50583 | 99.97 |
| license | 50577 | 99.96 |
| square_feet | 50213 | 99.24 |
| monthly_price | 45683 | 90.28 |
| weekly_price | 44945 | 88.83 |

### 3.2.4 Feature Construction

- We convert DateTime columns such as the date that the host first joined Airbnb (`host_since`) to measure the number of days a host has been on the platform, measured from the date that the data was collected (`host_days_active`).
- We create a new feature that measures the number of days between the first review and the date we collect the data (`time_since_first_review`) from the feature the date of the first review (`first_review`).
- We can compute the number of days between the most recent review and the date the data was scraped (`time_since_last_review`) using the date of the most recent review (`last_review`).

### 3.2.5 Binning Predictors

We want to discretize continuous data into "bins" for analysis in many situations, especially when that data has a wide range. Binning, also known as quantization, is a useful technique for converting continuous numeric features into discrete ones (categories). In our dataset, we peform the following binning:

- The feature that measures proportion of messages that the host replies (host_response_rate) will be bin into four categories '0-49%', '50-89%', '90-99%', '100%',
- The number of days between the first review and the date we collect the data (`time_since_first_review`) will be bin into '0-6 months', '6-12 months', '1-2 years', '2-3 years', '4+ years'
- The number of days between the most recent review and the date the data was scraped (`time_since_last_review`) will be bin into '0-2 weeks','2-8 weeks', '2-6 months', '6-12 months', '1+ year'
- Review scores rating (`review_scores_rating`) will be bin into following categories '0-79/100', '80-94/100', '95-100/100'

### 3.2.6 Data Transformation

#### Feature Scaling

Feature scaling is an essential step in the preprocessing process, as some machine learning algorithms do not perform well when the numerical input attributes have very different scales.

Feature scaling through standardization (or Z-score normalization) is straightforward and common-used in the machine learning community. Standardization means that we rescale the predictors so that they have a zero mean and standard deviation 1. We first compute the mean and standard deviation for the feature and scale it based on:

$$x^{'} = \frac{x - \bar{x}}{\sigma}$$

where $x$ is the original feature vector, $\bar{x}$ = average(x) is the mean of that feature vector, and $\sigma$ is its standard deviation.

In our dataset we use a transformer called StandardScaler from Scikit-Learn for standardization.

#### Transformation to Resolve Skewness

Many models assume a normal distribution, which means data are symmetric about the mean. Unfortunately, our real-life datasets do not always follow the normal distribution. Instead, they are usually skewed, which makes the results of our statistical analyses invalid.

Figure A.19 shows the distribution of numerical features. We notice that most of the predictors have a heavy-tailed distribution. Therefore we transform them by replacing them with their logarithm. Figure A.20 shows the result; some of the distributions become normal. More importantly, the outcome variable price appears much more normally distributed.

### 3.2.7 Handling Categorical Attributes

One of the significant problems with machine learning is that many algorithms cannot work directly with categorical data (non-numerical values). Hence, we need a way to convert categorical data into a numerical form, and our machine learning algorithm can take in that as input. In this study, we use One-Hot Encoding, one of the most widely used encoding techniques. One-hot encoding is processed in 2 steps:

1. First, we split categories into different columns.

2. We then put 0 for others and 1 as an indicator for the appropriate column.

There is a very simple way to encode the data in python pandas library, using the `get_dummies()` function.

## 3.3 Exploratory Data Analysis

Before embarking on developing statistical models and generating predictions, it is essential to understand your data. This is typically done using conventional numerical and graphical methods. Tukey (1977) advocated the practice of exploratory data analysis (EDA) as a critical part of the scientific process.

We present the descriptive statistics of variables in Table A.1

### 3.3.1 Numerical Features

#### Price

The nightly advertised prices range from \$0 to \$10,000. The range is so broad because hosts do not understand how to set Airbnb advertised prices. Figure 3.1 and Figure 3.2 show the distributions of price up to \$1,000 and \$200 respectively. While the price's range is extensive, most of its values concentrate on the range \$10 to \$1000. Hence, for minimal values under \$10, we will increase them to \$10, and values above \$1,000 will be reduced to \$1,000.

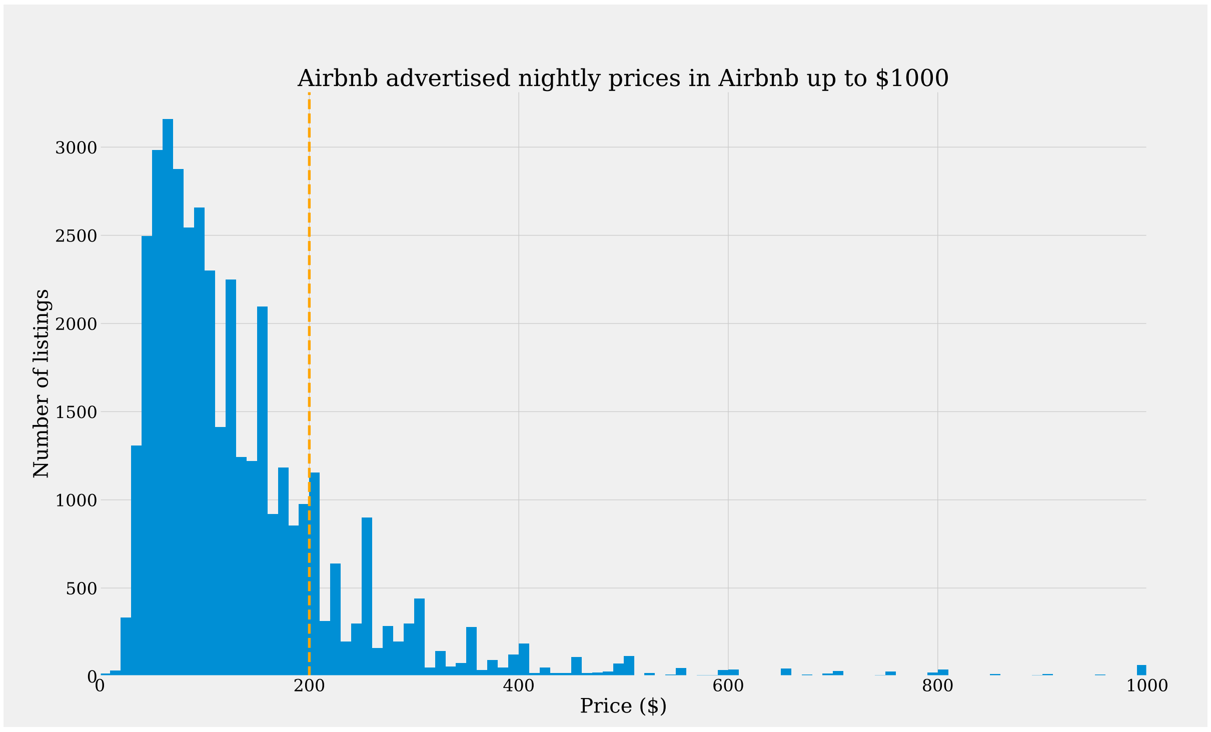


**Figure 3.1:** Airbnb advertised price up to $1000

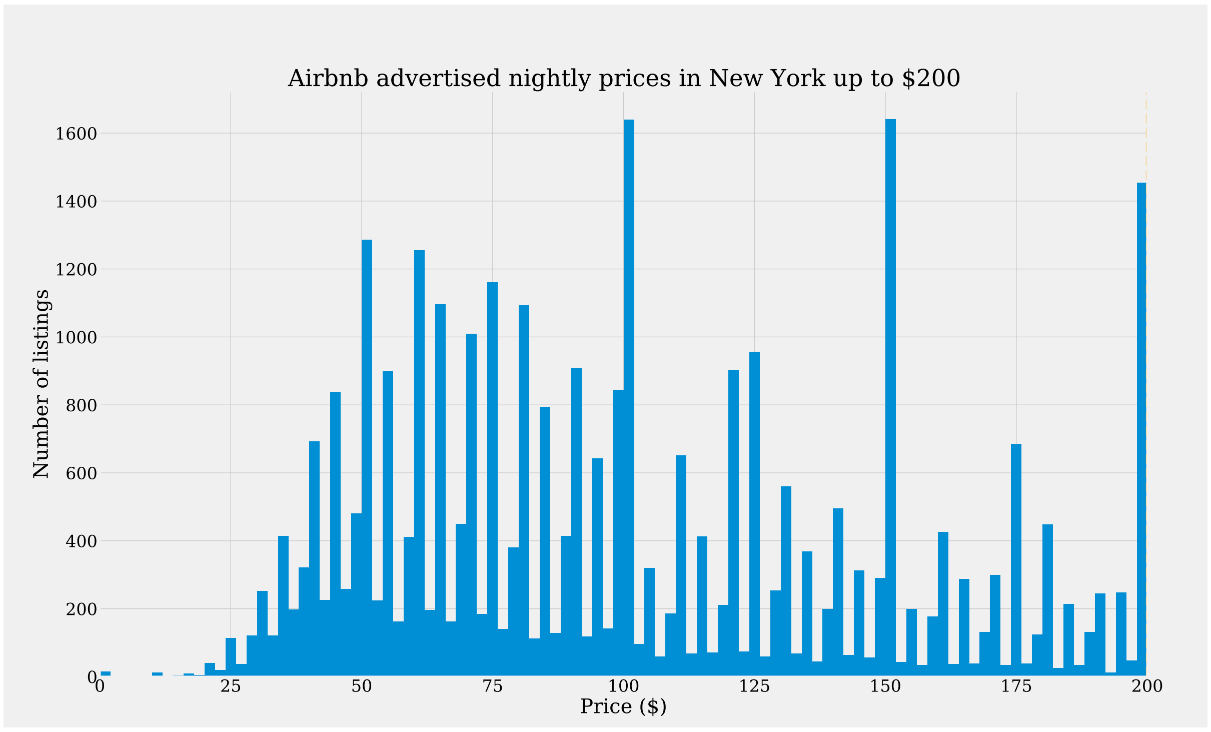


**Figure 3.2:** Airbnb advertised price up to $200

#### Host Listings Count

The median number of listings that the host of each listing has is 1. The mean is higher (8 in total) due to some hosts running many listings. About 55% of listings are from hosts with one listing, and 45% are from multi-listing hosts. This feature has been shown to have a positive effect on the price of Airbnb listings(Y. Chen and Xie, 2017, Ert et al., 2016, Wang and Nicolau, 2017)

#### Number of people accommodated, bathrooms, bedrooms and beds

Figures A.1 to A.4 reveal that the most common listing type accommodates two people in one bed in one bedroom with one bathroom.

The number of bedrooms, number of bathrooms, and number of accommodations appear to have a positive impact on Airbnb rental price (Ert et al., 2016; Y. Chen and Xie, 2017; Wang and Nicolau, 2017; Gibbs et al., 2018). Figures A.5 - A.8 show that the more people a listing accommodates, the more number of bedrooms it has, the more bathrooms it has, the higher the price they can charge their customers. However, we can see that those figures' general trends are similar, which implies that those features may be highly correlated.

### 3.3.2 Categorical features

Our main EDA objective for categorical data is to know the unique values and their corresponding count.

#### Neighbourhood

Several numbers of published studies recognize the importance of locational factors in the pricing strategy of Airbnb. A listing close to the city center (ibid.;J. Li et al., 2016; Wang and Nicolau, 2017; Z. Zhang et al., 2017;Gibbs et al., 2018) and coastline (Perez-Sanchez et al., 2018) has a higher room rate. ibid. also found that a listing located within sightseeing, eating, or shopping area gains a price premium.

As shown in Figure 3.3, Manhattan and Brooklyn have the most Airbnb properties. Figure 3.4 plots the distributions of New York boroughs using kernel density estimation. Manhattan and Brooklyn are the most expensive boroughs, which is not surprising because they are famous tourist attractions. In those two boroughs, tourists can find neighborhoods for almost any interest. For example:

- Sightseeing: Midtown is the heart of New York shopping and theater and home to some of its most iconic buildings.

- Nightlife: More clubs are found in “Hell’s Kitchen,”
- Food: In Soho, tourists can experience a host of the highest-rated dining places.
- Theather: There is no more convenient home base than the Theater District, located in 42nd Street to 50th Street west of Sixth Avenue.
- For families: Upper West Side is bordered with parks and playgrounds and boasting both a children’s museum and the famed dinosaurs at the American Museum of Natural. This neighborhood is also considered one of the safest areas of New York City

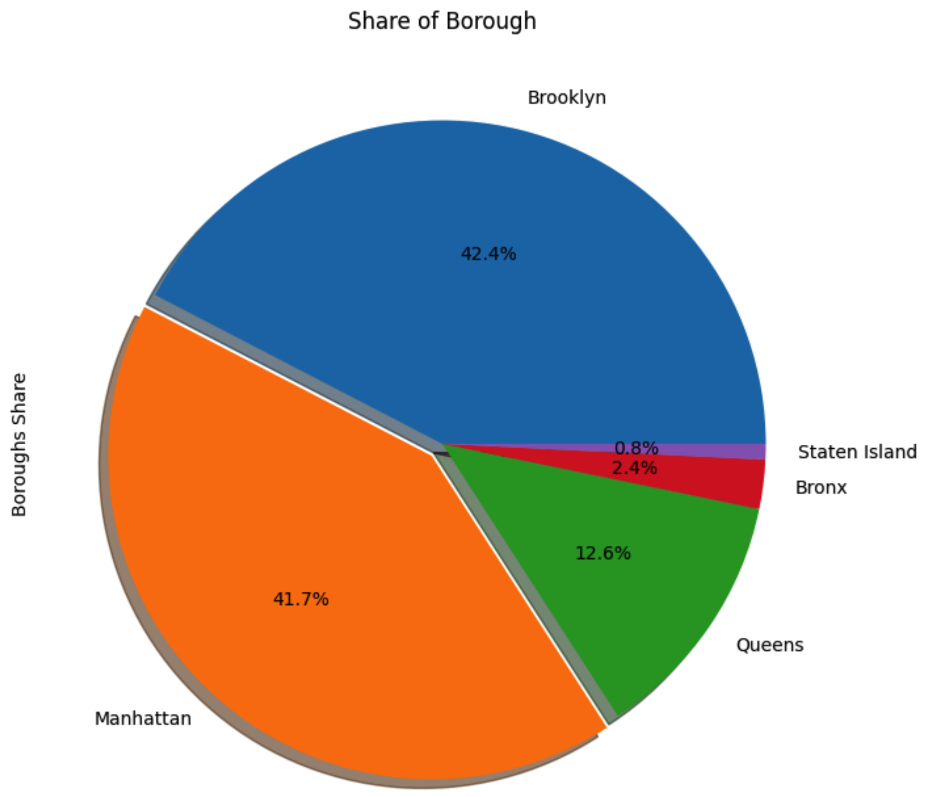


**Figure 3.3:** Borough Listings

### Property and room types

As shown in Figure 3.5, about 80% properties are apartments. The remainder are houses or more uncommon property types (e.g. ’bed and breakfast’ or ’yurt’). The median price of apartment type listing is higher than that of house type listing but lower than uncommon property types (3.6).

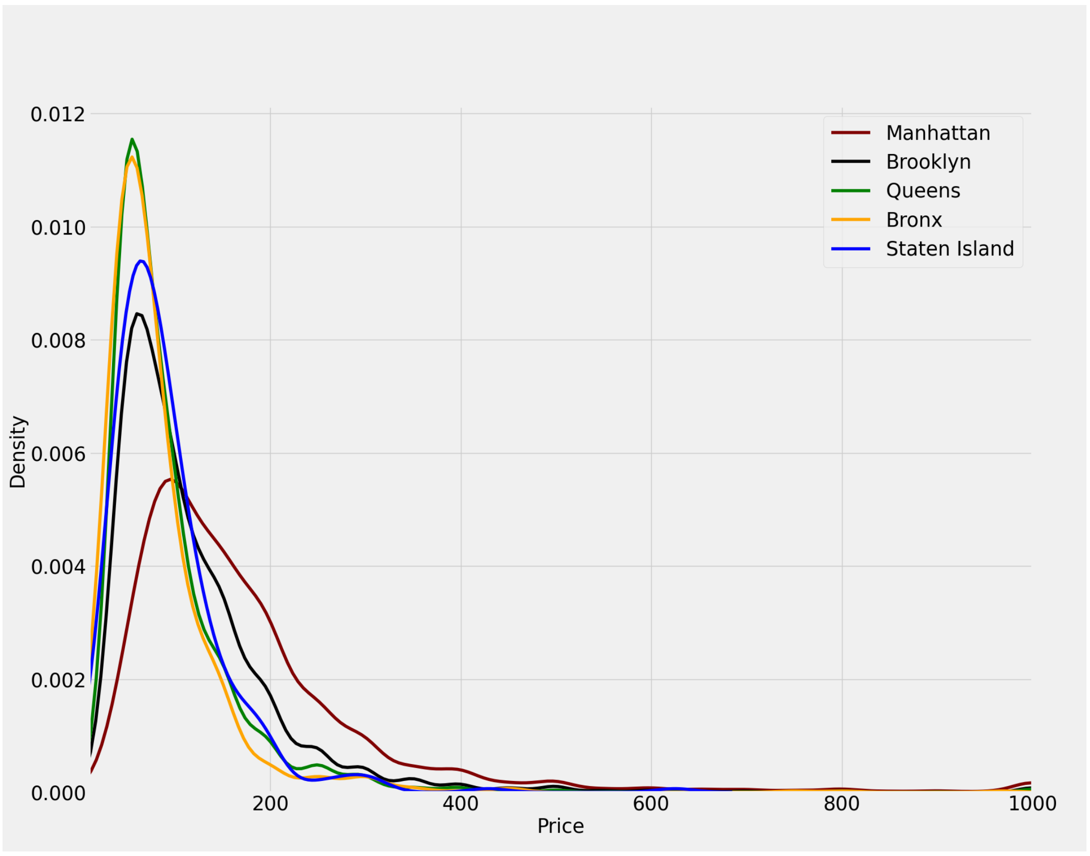
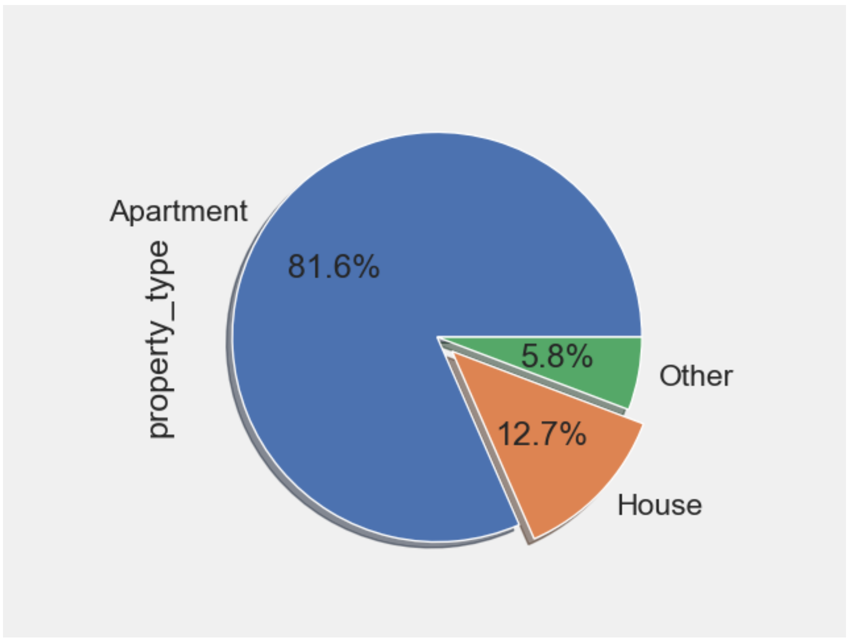


**Figure 3.4:** Kernel Density Estimates of Airbnb Rental Price in New York Boroughs

**Figure 3.5:** Property Type Pie Chart



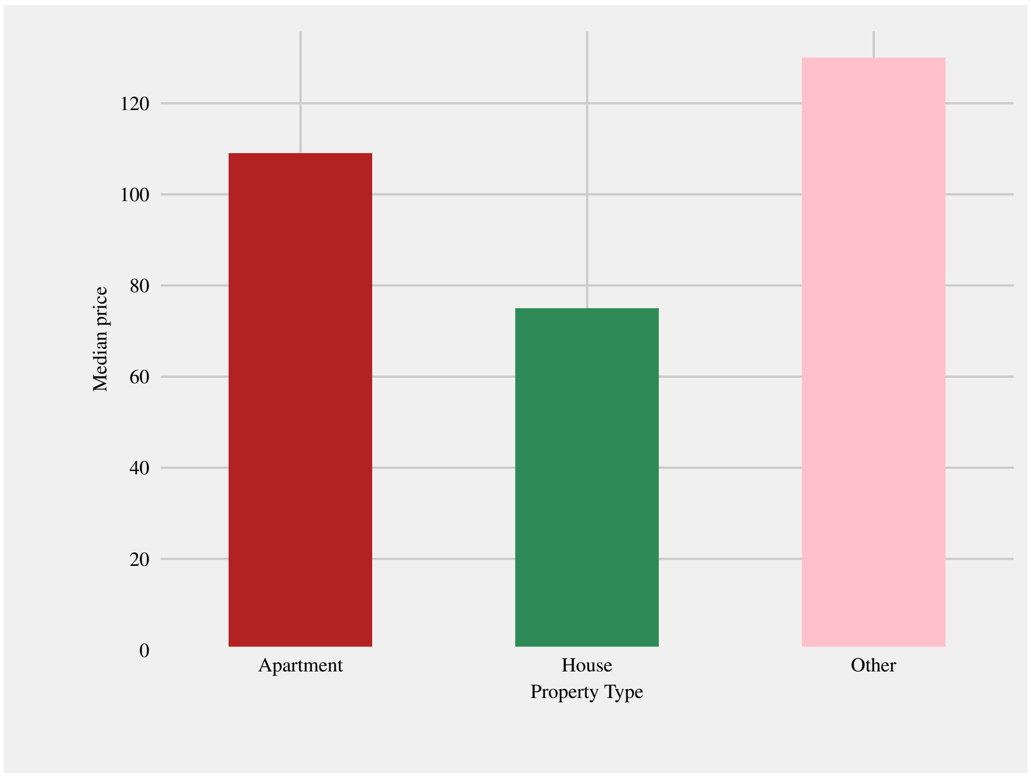


**Figure 3.6:** Median Price By Property Type

Figure 3.7 shows that about 52% of listings are entire homes (i.e. you are renting the entire property on your own). Most of the remainder are private rooms (i.e. you are renting a bedroom and possibly also a bathroom, but there will be other people in the property). Fewer than 3% are shared rooms (i.e. you are sharing a room with either the property owner or other guests).

On the question of whether there's a price difference between different types of rooms, Figure 3.8 reveals that the rental price of the entire home and a private room, and a hotel room is higher than the shared room. This finding is consistent with many recent studies (Cai et al., 2019 ; Benitez-Aurioles, 2018; Y. Chen and Xie, 2017; Gibbs et al., 2018)

### Reviews

From Figure 3.9, we see that, while few listings receive review ratings of 80 or below, most listings with a review have received a 95-100/100 overall, indicating that the customers adore their Airbnbs.

As shown in Figure 3.10, the review score rating has a positive effect on the median rental price. It all makes intuitive sense customers are willing to pay a premium price for a listing with a good reputation. However, the evidence for the relationship between is inconclusive. Many studies (Y. Chen and Xie, 2017; Gibbs et al., 2018; Wang and Nicolau, 2017) have shown that the overall rating score has a positive impact on rental

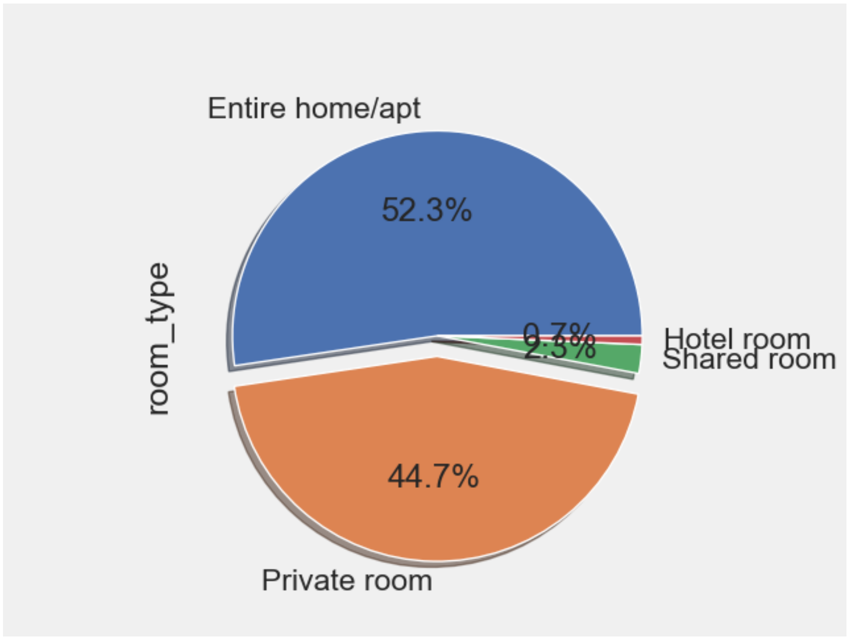


**Figure 3.7:** Room Type Pie Chart

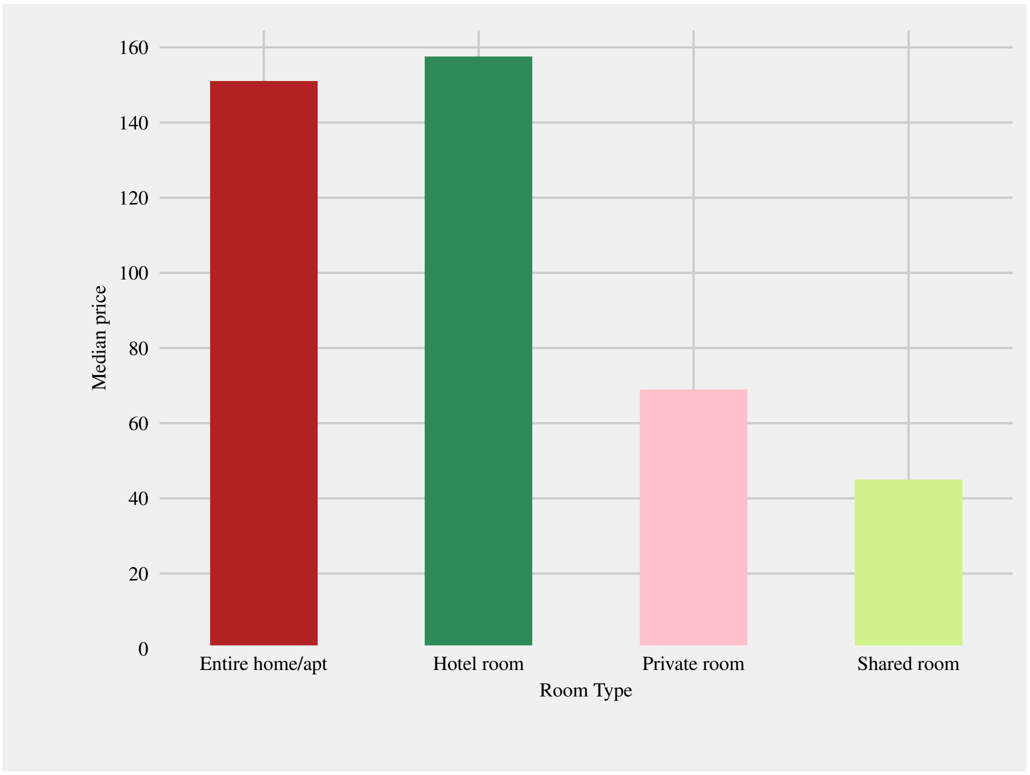


**Figure 3.8:** Median Price By Room Type

price, while others (J. Li et al., 2016; Z. Zhang et al., 2017) suggests the otherwise.

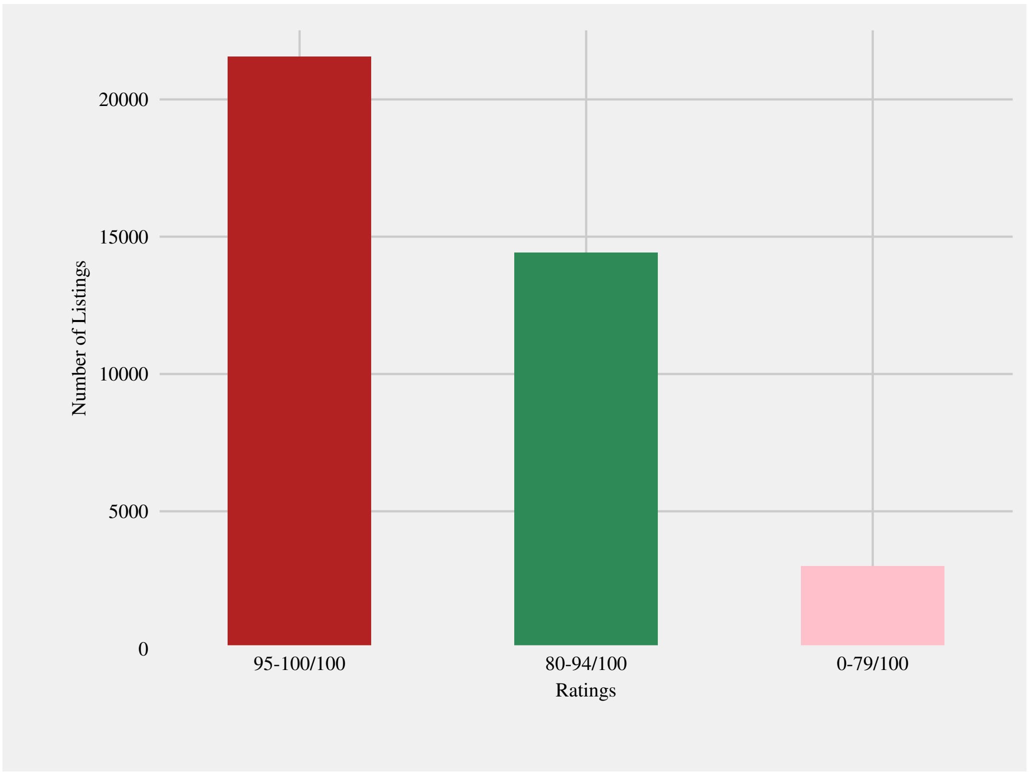


**Figure 3.9:** Overall Listing Rating Distribution

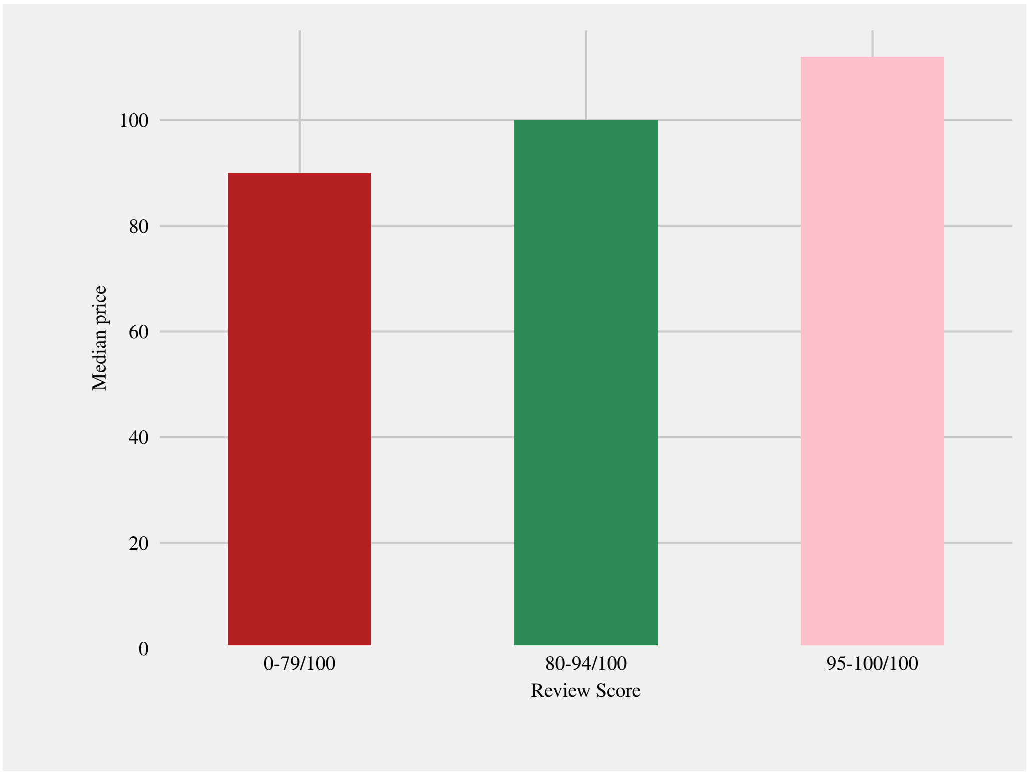


**Figure 3.10:** Median Price By Review Score Rating

### First and Last Review

As can be seen from the Figure 3.11, the most common period in which Airbnb listings had their first review is 2-3 years, which means that many listings on the site have been active for at least a couple of years. However, fewer listings have been on Airbnb for more than four years.

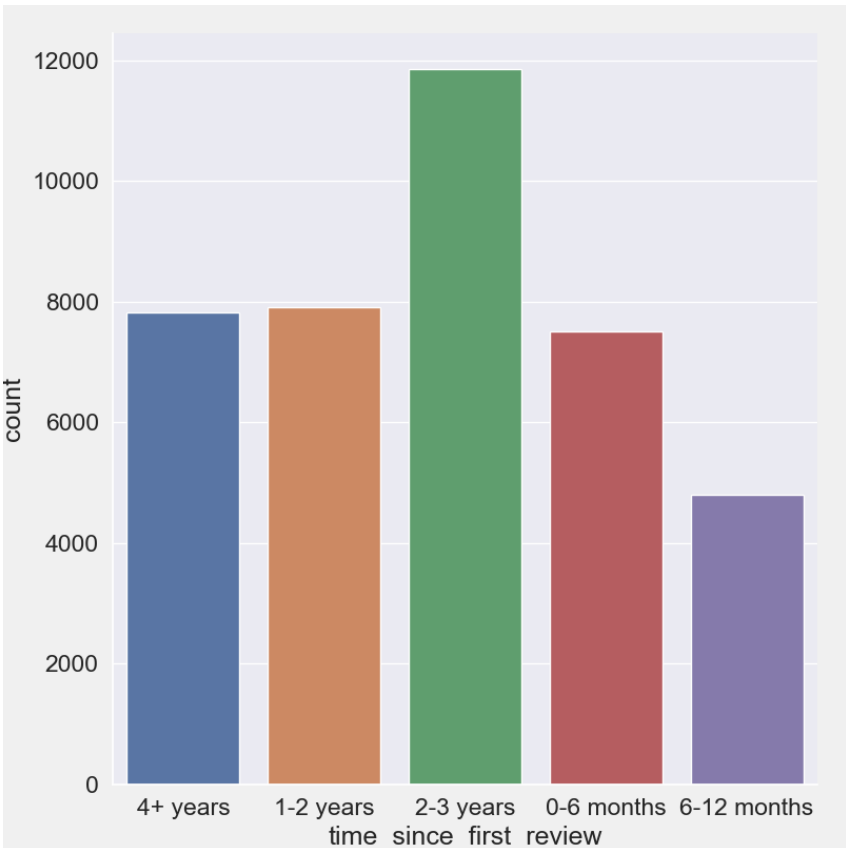


**Figure 3.11:** Time Since First Review

We expect that people would pay a higher price for listing lives in Airbnb for a long time than listings with a recent history with Airbnb. While the rental price is higher for listings had their reviews for four years or more, there's no difference in median price for other categories (Figure 3.12)

The bar plot 3.13 reveals that the most common period since a listing received its last review is 2-8 weeks, which means that many listings have been reviewed relatively recently. What stands out in the figure is that over 10,000 listings have not had a review for more than a year, which means they exist on the site, but they do not have their calendars open and are not available to reserve.

Time since the last review may have been an essential factor in how people decide to rent an Airbnb listing. People may avoid booking accommodation that has not been reviewed for a long time. Therefore, we expect the time since the last review has a negative

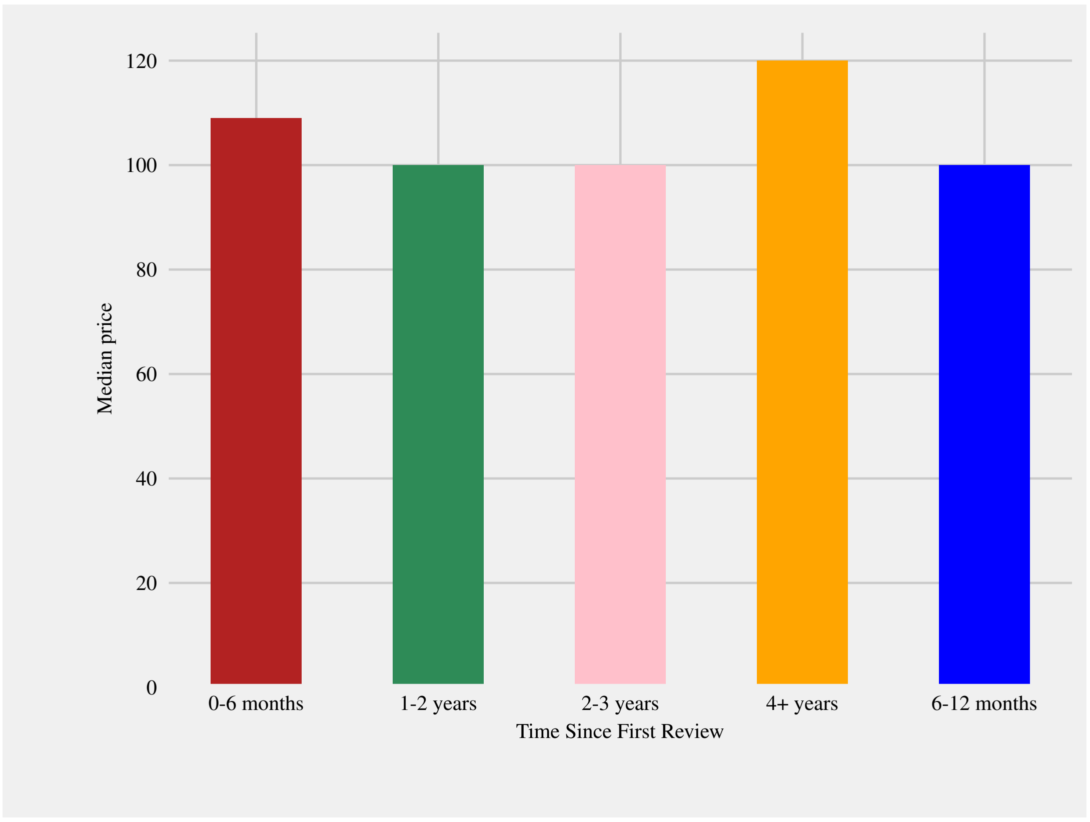


**Figure 3.12:** Median Price By Time Since First Review

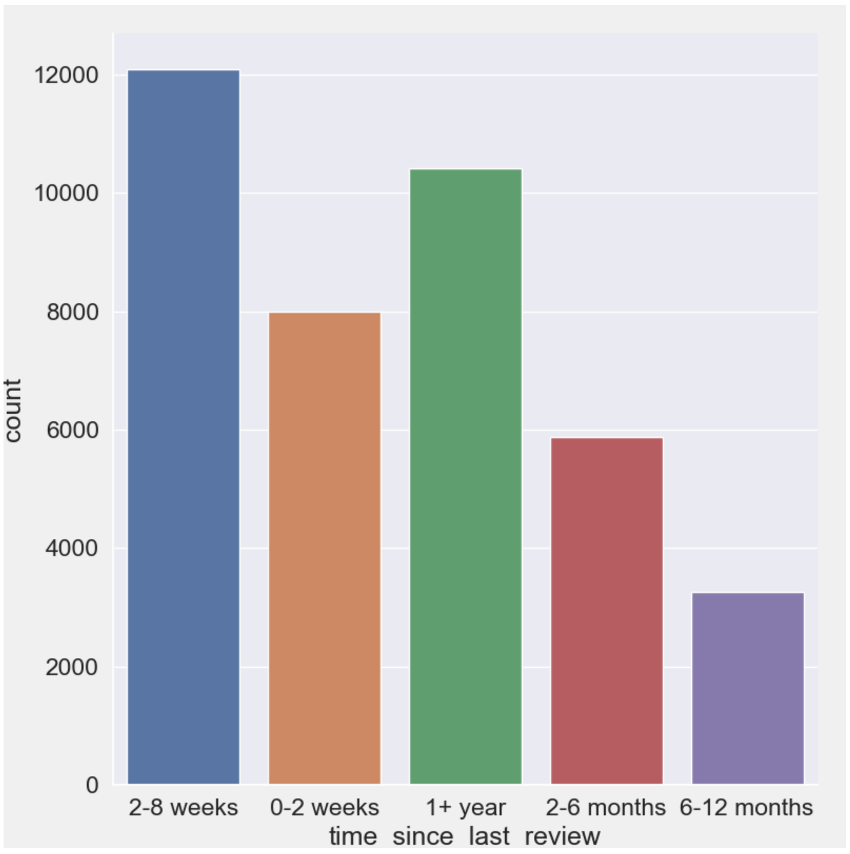


**Figure 3.13:** Time Since Last Review

effect on the rental price. Contrary to our expectation, Figure 3.14 showed no significant difference between different categories of the number of days since the last review.

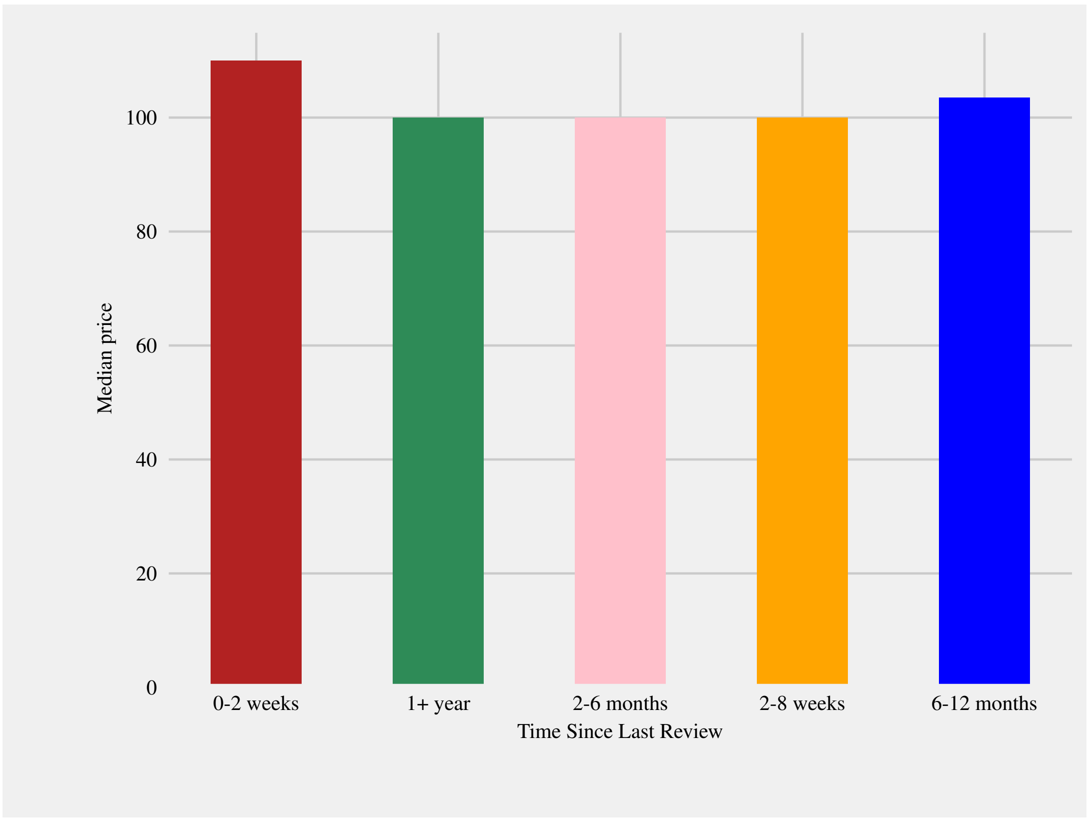


**Figure 3.14:** Median Price By Time Since Last Review

### 3.3.3 Boolean features

Many features (e.g. for amenities) can be true or false. This section compares the proportions of these features that are true or false (to explore the data and also to ascertain whether the feature is worth retaining), and the median price of each category (to explore the relationship between the category and price).

#### Superhosts

Figure 3.15 shows that about 23% of hosts have a superhost badge. Hosts with superhost status usually charge higher prices. A possible explanation for this might be that people are willing to pay a premium price because they consider superhost status a mark of quality. This also accords with earlier studies(Gibbs et al., 2018, Kakar et al., 2016;Wang and Nicolau, 2017,Cai et al., 2019).

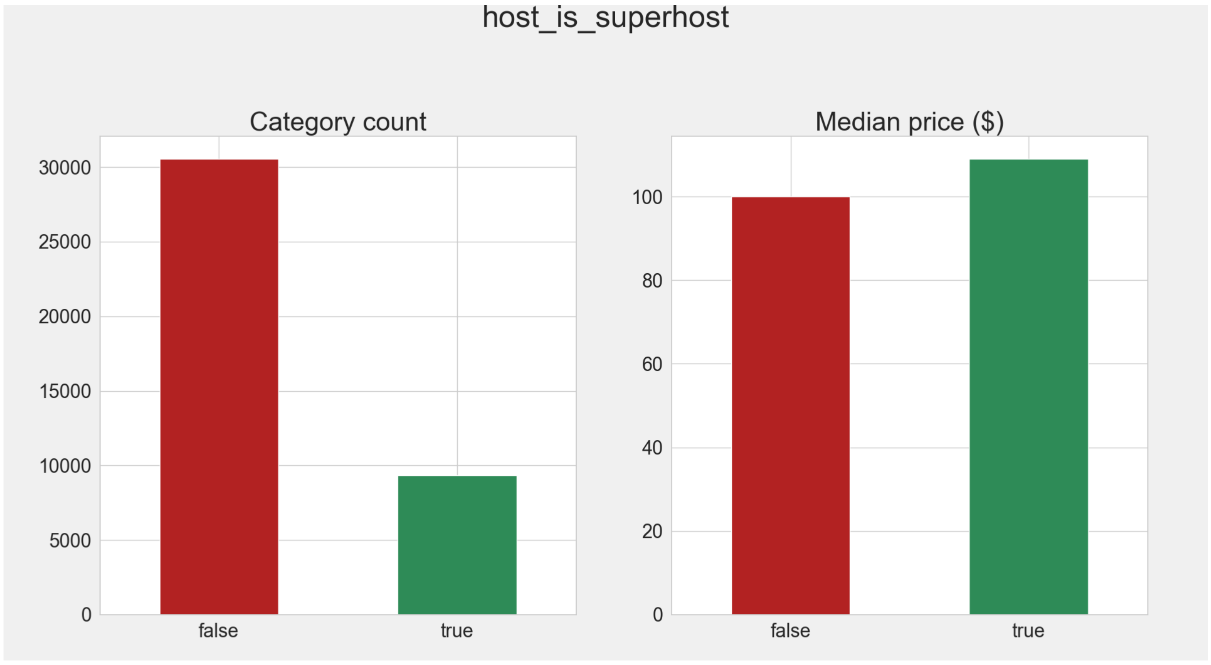


**Figure 3.15:** Count Plot and Median Price By Whether A Host Is a Superhost

### Host verification

In Figure 3.16, about 49% of hosts are verified. Consistent with the literature (Y. Chen and Xie, 2017; Wang and Nicolau, 2017), the figure showed that hosts with verified profiles gain a price premium. The relationship may be explained by the fact that verified profiles (e.g., by providing ID and verifying your phone number and email address) can increase their trustworthiness and, therefore, can charge a higher rental price.

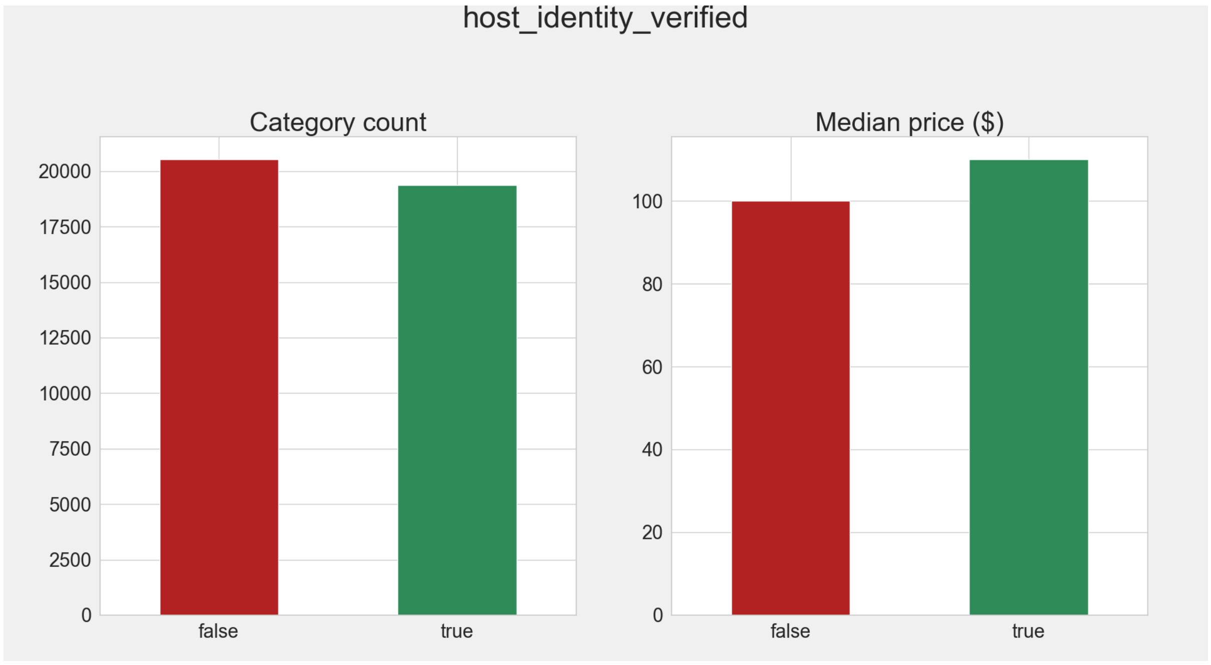


**Figure 3.16:** Count Plot and Median Price By Whether A Host Is Verified

### Instant booking

As shown in figure below, about 40% of properties are instant bookable and hosts that allow for immediate booking without confirmation have lower prices than those who do not. This finding seems counterintuitive, as we would expect higher willingness-to-pay from potential guests for the added convinience of intant booking. This negative link can be explained by both emotional (Wang and Nicolau (ibid.)) and economic (Benitez-Aurioles (2018)).

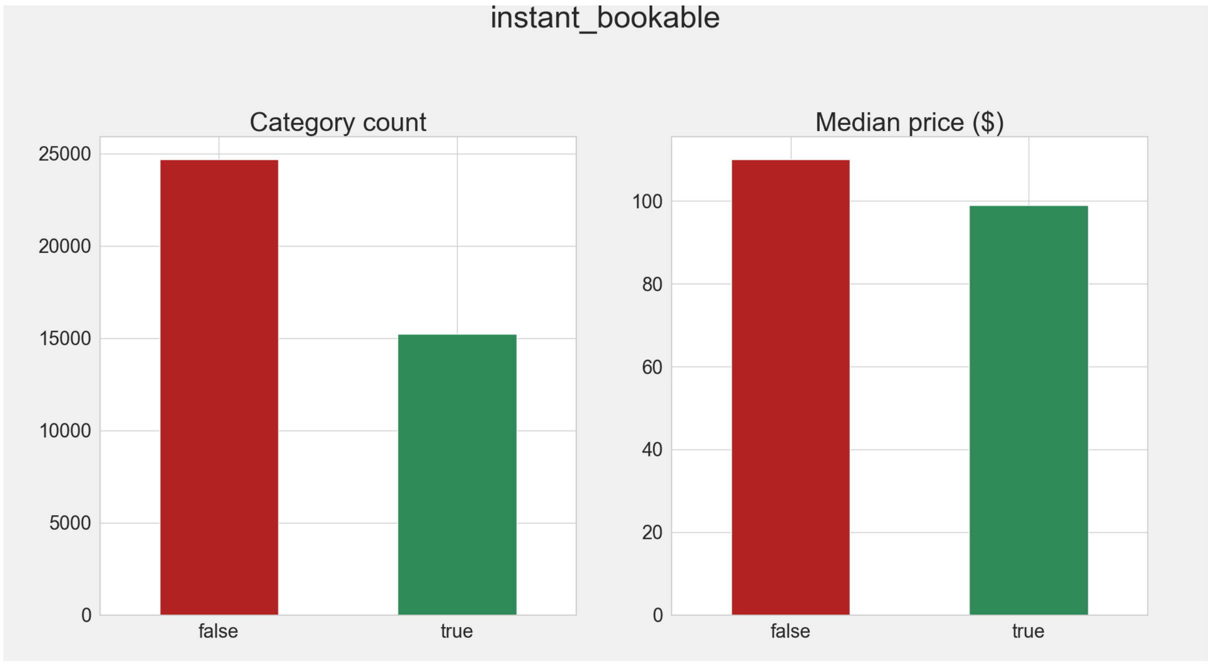


**Figure 3.17:** instant_bookable

### Amenities

Our goal is to identify which amenities are common and which increase the price of an Airbnb listing. We plot the count plot and each amenity's median price to explore the relationship between the amenity and price. Amenities then can be split into three groups:

1. The first group contains uncommon amenity, but listings with it have a higher median price: Bed linen, Coffee machine, Basic cooking equipment, Elevator, Child friendly, Long term stays allowed, Private entrance, Self check-in, Pets allowed, Washer,dryer and/or dishwasher (white goods) (See Figures A.9 to A.13)

2. The second group includes common amenities and listings with it have a higher median price: TV, Internet, Air conditioner (See Figures A.14 and A.15).

3. The third group comprises uncommon amenities, and listings with it have a lower median price: free car parking (presumably because these are less likely to be central properties), greeted by host. (See Figure A.16)

It is somewhat surprising that free car parking does not have a significant influence on the rental price. This may be because the more comfortable way and quickest way to travel around New York City is by the subway, so people would not consider car parking a critical factor when deciding to rent an Airbnb listing. The reason why the host greeting does not associate with a higher price is not apparent.

### 3.3.4 Multivariate Exploration

The goal of this section is to investigate the relationships between pairs of our features. A primary concern when analyzing the relationship between various variables is multicollinearity, a phenomenon in which two or more independent variables substantially correlate(Cohen et al. (2013)).

While multicollinearity does not hurt the model's predictive power (Kutner et al. (2005)), collinearity can pose problems in the regression context. In particular, it can be challenging to separate the individual effects of collinear independent variables on the outcome variable.

A straightforward way to detect collinearity is to construct a correlation matrix among predictors. An element of this matrix that is large in absolute value indicates a pair of highly correlated variables and are indicative of collinearity issues.

As shown in Figure A.17 shows the correlation matrix, areas of multicollinearity are:

- Beds, bedrooms, guests included, and the number of people that property accommodates are highly correlated.
- There are strong negative correlations between houses and apartments and between private rooms and entire homes.

Providing a full remedy for the multicollinearity issue is beyond the scope of this thesis. However, we employ two simple approaches as followed:

1. The first approach is to drop problematic variables from the regression.
2. The second method is to employ regularization techniques, which combat collinearity by using biased models (as demonstrated in section 2.6.2), which reduce the error variance of estimators.

### 3.3.5 Time Series Analysis

Figure 3.18 shows the classical additive decomposition of the number of (currently live) hosts joining the site. This is accomplished by using `seasonal_decompose()` function from the python library `statsmodels`. It decomposes a time series into the overall trend, seasonality (a significant factor in the tourism industry), and the unexplained residuals.

Regarding trend estimation, from 2011 onwards, the number of listings started growing considerably. However, growth in the number of new hosts (of those currently listed on the site) has decreased since mid-2014.

Strong evidence of seasonality was also found. The number of hosts joining Airbnb peaked in the summer because people put properties online to utilize the increased number of tourists in the summer holidays.

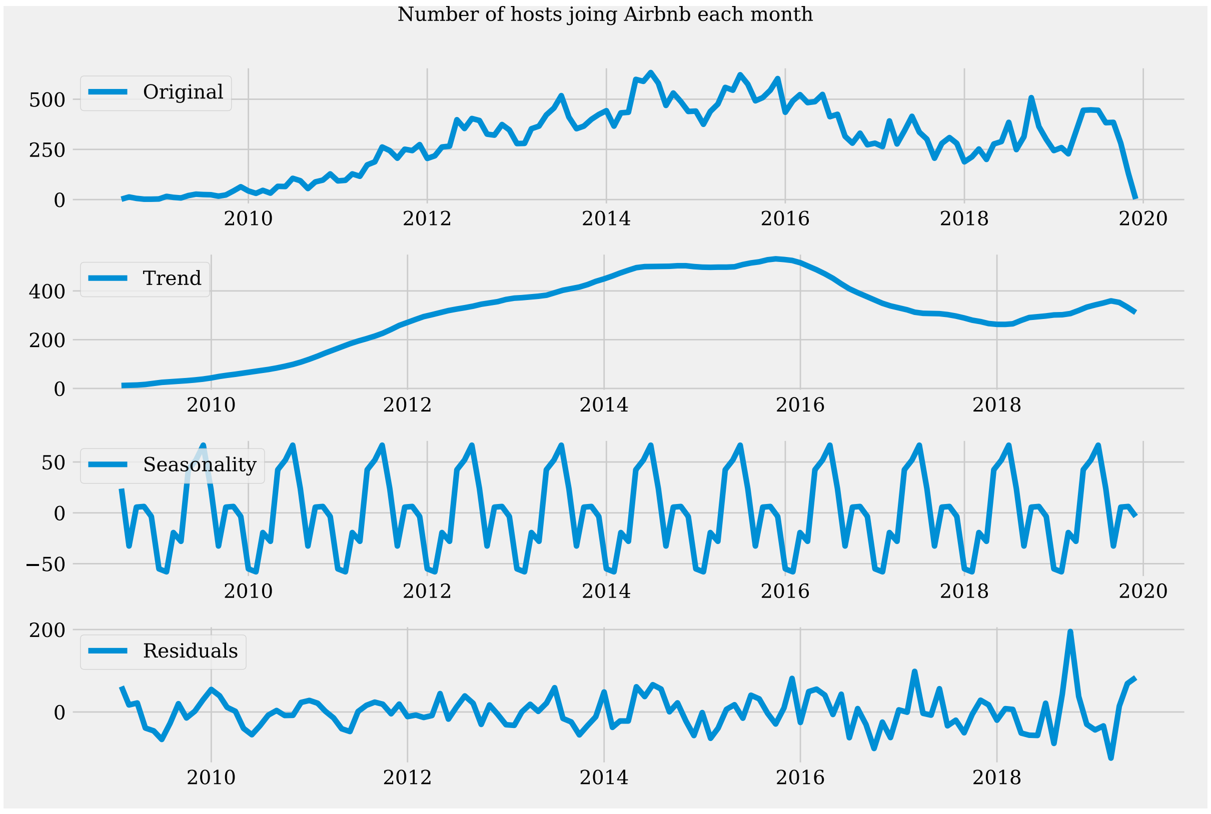


**Figure 3.18:** Number of hosts joining Airbnb each month

In terms of how prices changed over time, Figure 3.19 shows that the average price per night for Airbnb listings in New York has increased slightly over the last ten years. Particularly, the top-tier property prices have increased, resulting in a more substantial increase in the mean price than the median.

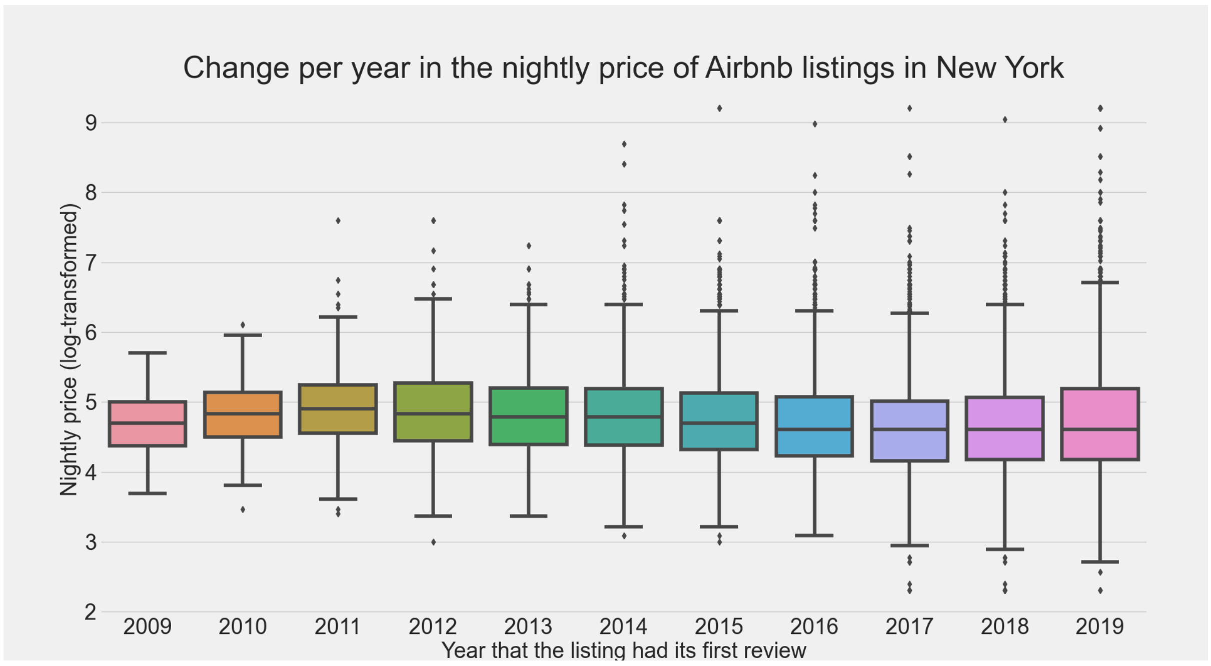


**Figure 3.19:** Change per year in the nightly price of Airbnb listings in New York

## 3.4 Modelling

### 3.4.1 Training and Test Sets

The train-test split procedure is used to estimate machine learning algorithms' performance when making predictions on data not used to train the model. We randomly split the dataset into 80% and 20% of listings for training and test sets, respectively. The training set is used to fit the model, and the test set is used to evaluate the fit machine learning model. The objective is to estimate the machine learning model's performance on new data: data not used to train the model.

The Python scikit-learn library provides an implementation of the train-test-split procedure via the `train_test_split()` function.

### 3.4.2 Findings

A summary of of results is reported in Table 3.2

As expected in 2.6.1 incorporating such a large number of features (309) makes the linear regression model overfit the data. As shown in Table 3.2, while training MSE of the linear regression model is quite good, the model performs poorly on the test set both in terms of in terms of $MSE$ and $R^2$.

By performing k-fold cross-validation with ten folds, we can find the tuning parameter's

**Table 3.2:** Results

| ML Algorithm | Training MSE | Test MSE | Training $R^2$ | Test $R^2$ |
|---|---|---|---|---|
| Linear Regresion | 0.1291 | 8.5E21 | 0.7019 | -1.9E22 |
| Ridge Regression | 0.1291 | 0.138 | 0.7019 | 0.6857 |
| Lasso Regression | 0.1351 | 0.1441 | 0.688 | 0.6718 |
| XGboost | 0.0798 | 0.1173 | 0.8157 | 0.7328 |

value that results in the smallest cross-validation error is 115. The test's MSE is associated with this value of is 0.138. The result represents a considerable improvement of ridge regression over the test MSE we got using least square regression.

Using cross-validation , we find the optimized penalty value $\lambda$ for lasso is 0.005. The associated test MSE is 0.1441, which is slightly higher than the test set MSE of the ridge regression. Recall from that the advantage of using lasso regression over ridge regression is that lasso performs feature selection. Lasso picked 153 variables while eliminated the other 125 features.

As shown by the Table 3.2, XGboost consistently outperforms these competing approaches in both mean squared error and R-squared. With this model, the features explain approximately 73% of the target variable's variance and have smaller MSE than the other regression model.

**Table 3.3:** XGBoost Top 20 Important Features

| Feature | Weight |
|---|---|
| room_type_Entire home/apt | 0.336396 |
| bathrooms | 0.032001 |
| neighbourhood_Midtown | 0.025008 |
| neighbourhood_Hell's Kitchen | 0.018545 |
| neighbourhood_East Village | 0.015763 |
| property_type_Other | 0.015168 |
| neighbourhood_Bedford-Stuyvesant | 0.014314 |
| neighbourhood_West Village | 0.014031 |
| neighbourhood_Chelsea | 0.013612 |
| neighbourhood_Lower East Side | 0.011874 |
| neighbourhood_Bushwick | 0.011854 |
| neighbourhood_Upper West Side | 0.011682 |
| neighbourhood_Washington Heights | 0.011659 |
| neighbourhood_SoHo | 0.011582 |
| room_type_Shared room | 0.011304 |
| neighbourhood_Greenwich Village | 0.010347 |
| room_type_Hotel room | 0.009697 |
| neighbourhood_Theater District | 0.008575 |
| neighbourhood_Williamsburg | 0.008490 |
| neighbourhood_Crown Heights | 0.007979 |

Another important finding from 3.3 was the role of location features play in predicting price. In particular, location features located in Manhattan and Brooklyn borough are in the top 20 important variables to predict the price. This is consistent with what we observed in 3.3.2.

# Chapter 4

# Conclusion and Future Works

This study's primary purpose was to evaluate the predictive performance of various machine learning models to real data on Airbnb listings in the city of New York. The methods used in this study consisted of simple and multiple linear regression, ridge regression, lasso regression, and XGBoost. The models were compared and assessed using mean square error and coefficient of determination criteria.

The research has shown that gradient boosting with all features (XGBoost) performs the best among all models. Ridge regression has the second-best performance. While Lasso's performance is not as good as Ridge Regression and XGBoost, Lasso performs feature selection. With Lasso, we were able to find a sparse model, which includes only relevant variables. Linear Regression performs the worst proved that the abundance of features leads to high variance and weak performance (overfitting).

The second aim of this study was to identify which characteristics of an Airbnb listing were most important in predicting the price. We found that the most critical feature is whether the type of listing is an entire home or not. The second most important feature is the number of bathrooms. Also, location features play an essential role in predicting price. These findings help develop a reliable price prediction model to aid the Airbnb hosts in maximizing their earnings.

In future work, we would like to:

1. Experiment with the data with Neural Network.
2. Find a way to include listing's photo quality as a predictor.
3. Incorporate customer reviews feature through sentiment analysis.

# Appendices

## A.1 Tables

**Table A.1:** Summary Statistics

| | mean | std |
|---|---:|---:|
| host_is_superhost | 0.234 | 0.424 |
| host_listings_count | 7.775 | 54.391 |
| host_identity_verified | 0.486 | 0.500 |
| accommodates | 2.906 | 1.911 |
| bathrooms | 1.140 | 0.421 |
| bedrooms | 1.183 | 0.750 |
| beds | 1.570 | 1.156 |
| price | 138.085 | 118.185 |
| security_deposit | 172.822 | 406.817 |
| cleaning_fee | 54.161 | 54.671 |
| guests_included | 1.584 | 1.210 |
| extra_people | 15.986 | 25.143 |
| minimum_nights | 6.151 | 19.285 |
| maximum_nights | 55397.541 | 10750846.675 |
| availability_90 | 32.857 | 33.807 |
| number_of_reviews | 31.090 | 51.014 |
| instant_bookable | 0.382 | 0.486 |
| host_days_active | 1654.499 | 885.855 |
| air_conditioning | 0.867 | 0.339 |
| bed_linen | 0.349 | 0.477 |
| tv | 0.685 | 0.465 |
| coffee_machine | 0.339 | 0.473 |
| cooking_basics | 0.370 | 0.483 |
| white_goods | 0.447 | 0.497 |
| elevator | 0.250 | 0.433 |
| child_friendly | 0.280 | 0.449 |
| parking | 0.469 | 0.499 |
| host_greeting | 0.171 | 0.377 |
| internet | 0.984 | 0.126 |
| long_term_stays | 0.218 | 0.413 |

**Table A.2:** The variable list

| Variables | Definition |
|---|---|
| experience_offerd | recommended category of travel type, e.g. business |
| host_since | date that the host first joined Airbnb |
| host_response_time | average amount of time the host takes to reply to messages |
| host_response_rate | proportion of messages that the host replies to |
| host_is_superhost | whether or not the host is a superhost, which is a mark of mark of quality for the top-rated and most experienced hosts, and can increase your search ranking on Airbnb |
| host_listings_count | how many listings the host has in total |
| host_identity_verified | whether or not the host has been verified with id |
| neighbourhood_cleansed | NewYork borough the property is in |
| property_type | type of property, e.g. house or flat |
| room_type | type of listing, e.g. entire home, private room or shared room |
| accommodates | how many people the property accommodates |
| bathrooms | number of bathrooms |
| bedrooms | number of bedrooms |
| beds | number of beds |
| bed_type | type of bed, e.g. real bed or sofa-bed |
| amenities | list of amenities |
| price | nightly advertised price (the target variable) |
| security_deposit | the amount required as a security deposit |
| cleaning_fee | the amount of the cleaning fee (a fixed amount paid per booking) |
| guests_included | the number of guests included in the booking fee |
| extra_people | the price per additional guest above the guests_included price |
| minimum_nights | the minimum length of stay |
| maximum_nights | the maximum length of stay |
| calendar_updated | when the host last updated the calendar |
| availability_30 | how many nights are available to be booked in the next 30 days |
| availability_60 | how many nights are available to be booked in the next 60 days |
| availability_90 | how many nights are available to be booked in the next 90 days |
| availability_365 | how many nights are available to be booked in the next 365 days |
| number_of_reviews | the number of reviews left for the property |
| number_of_reviews_ltm | the number of reviews left for the property in the last twelve months |
| first_review | the date of the first review |
| last_review | the date of the most recent review |
| review_scores_rating | guests can score properties overall from 1 to 5 stars |
| review_scores_accuracy | accuracy score of a property's description from 1 to 5 stars |
| review_scores_cleanliness | guests can score a property's cleanliness from 1 to 5 stars |
| review_scores_checkin | guests can score their check-in from 1 to 5 stars |
| review_scores_communication | guests can score a host's communication from 1 to 5 stars |
| review_scores_location | guests can score a property's location from 1 to 5 stars |
| review_scores_value | guests can score a booking's value for money from 1 to 5 stars |
| instant_bookable | whether or not the property can be instant booked (i.e. booked straight away, without having to message the host first and wait to |

## A.2 Figures

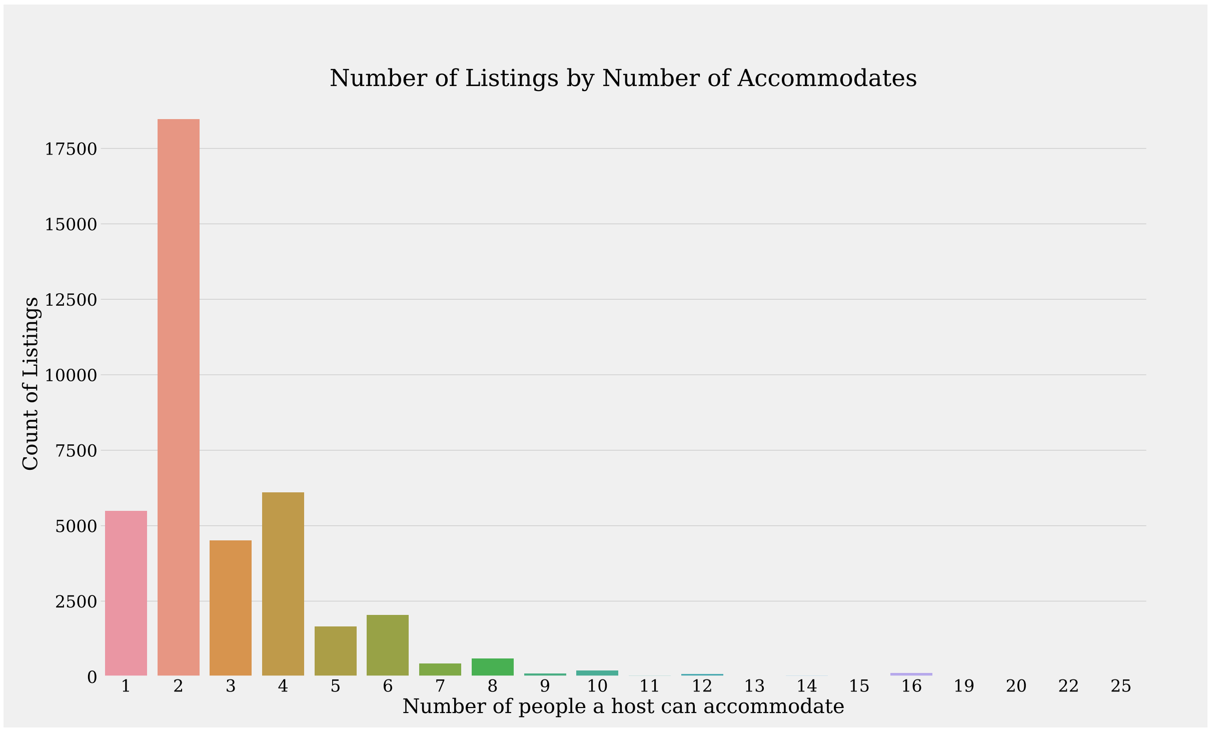


**Figure A.1:** Number of Listings by Accomodates

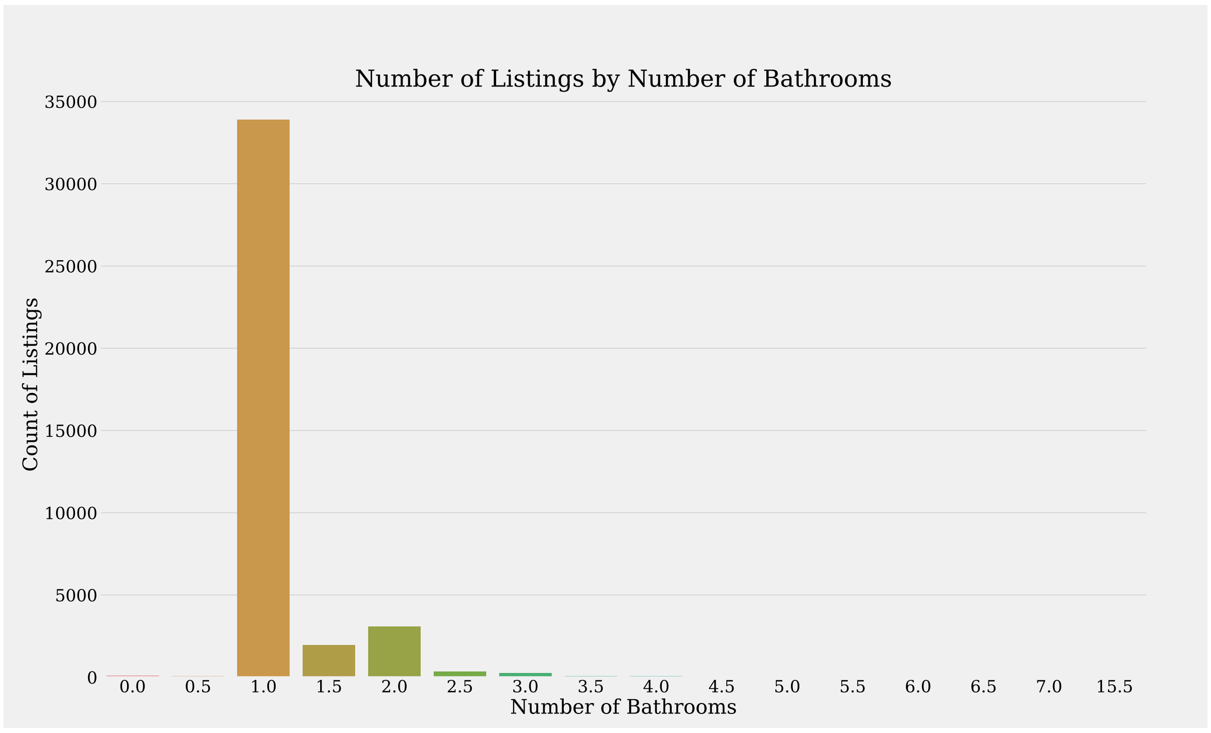


**Figure A.2:** Number of Listings by Number of Bathrooms

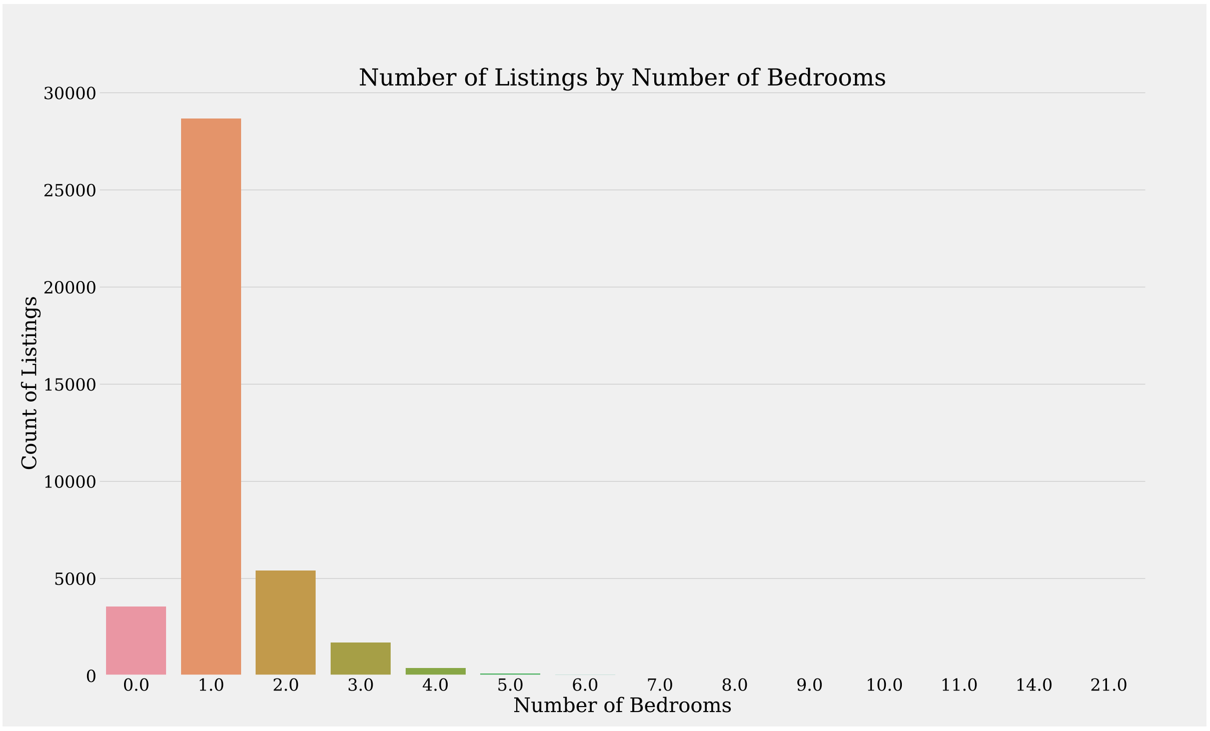


**Figure A.3:** Number of Listings by Number of Bedrooms

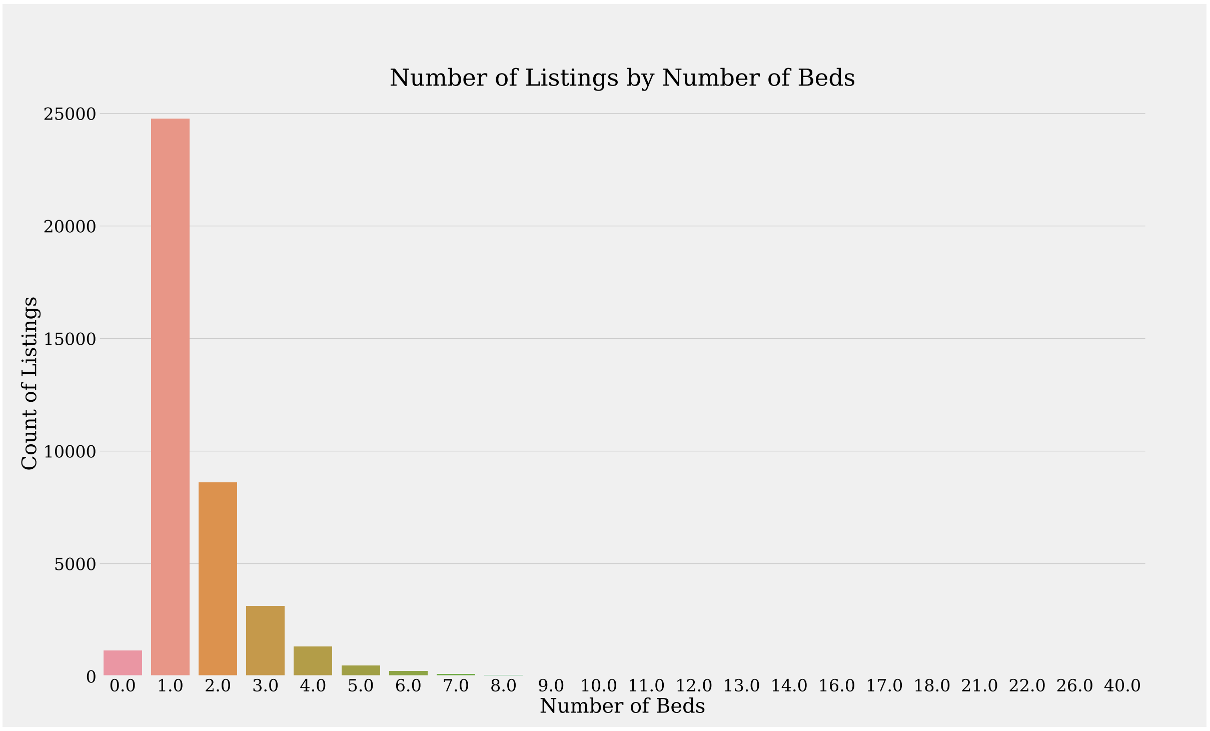


**Figure A.4:** Number of Listings by Number of Beds

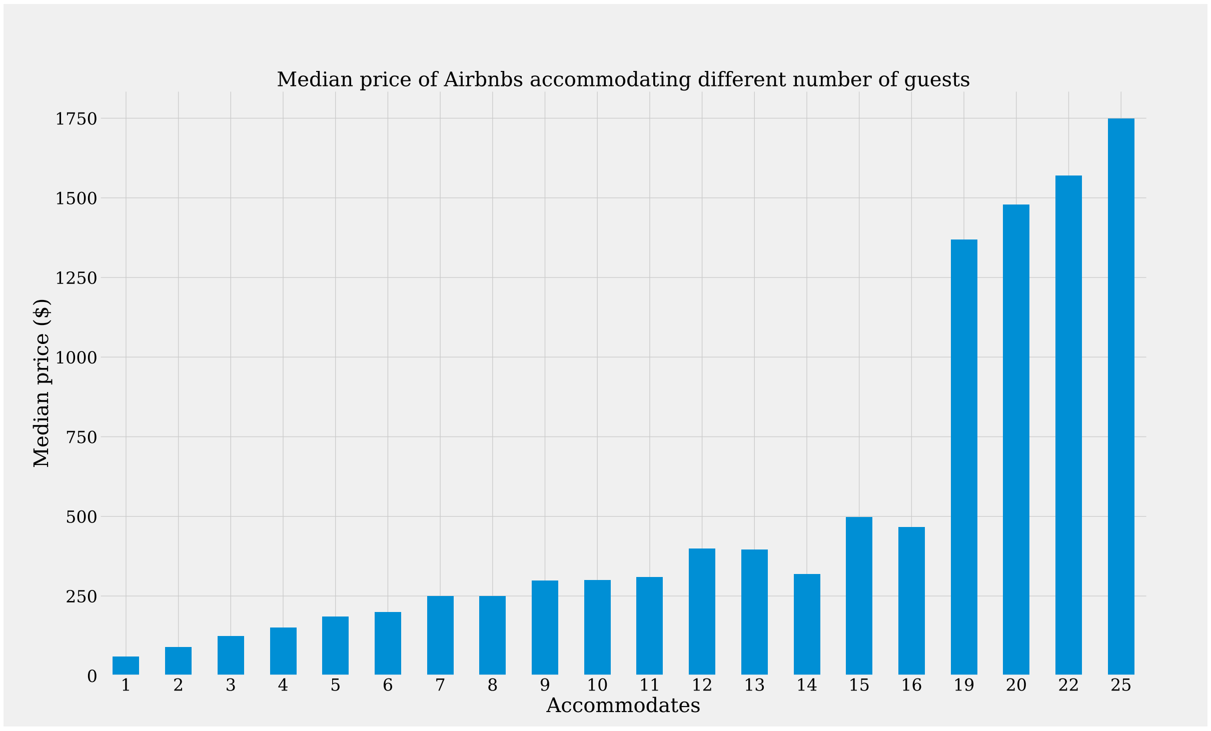


**Figure A.5:** Median Price By Accommodates

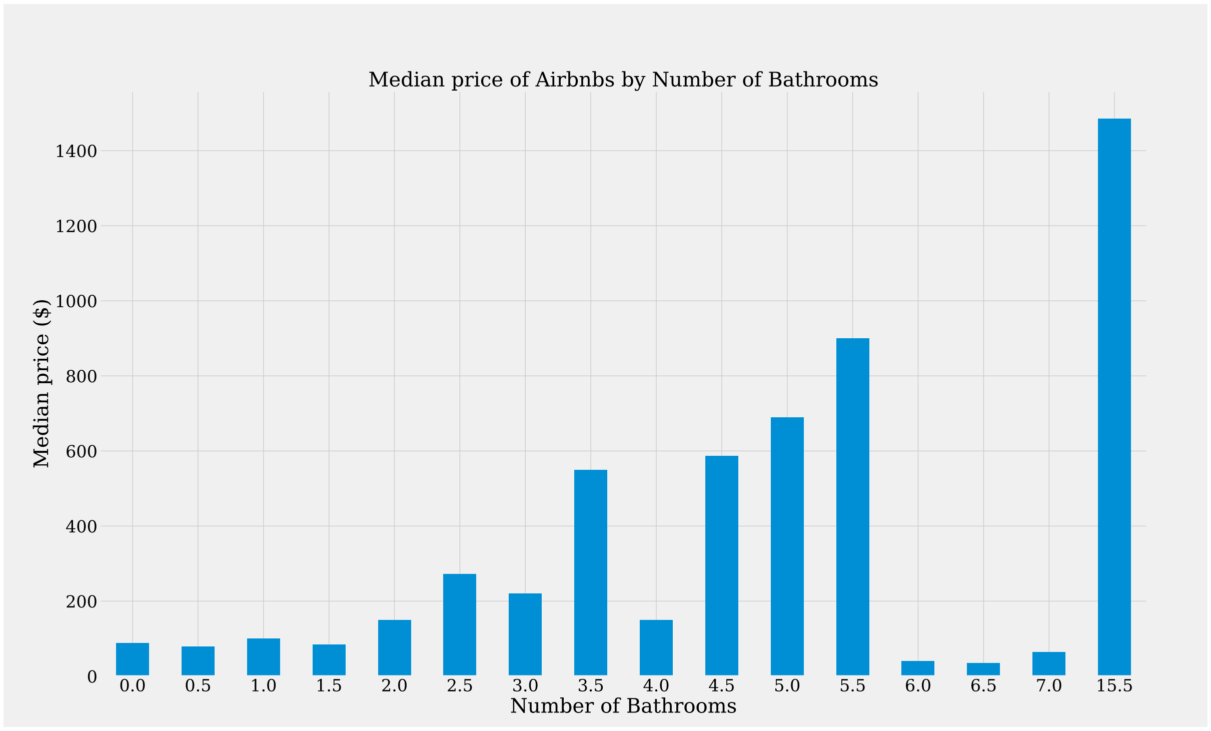


**Figure A.6:** Median Price By Number of Bathrooms

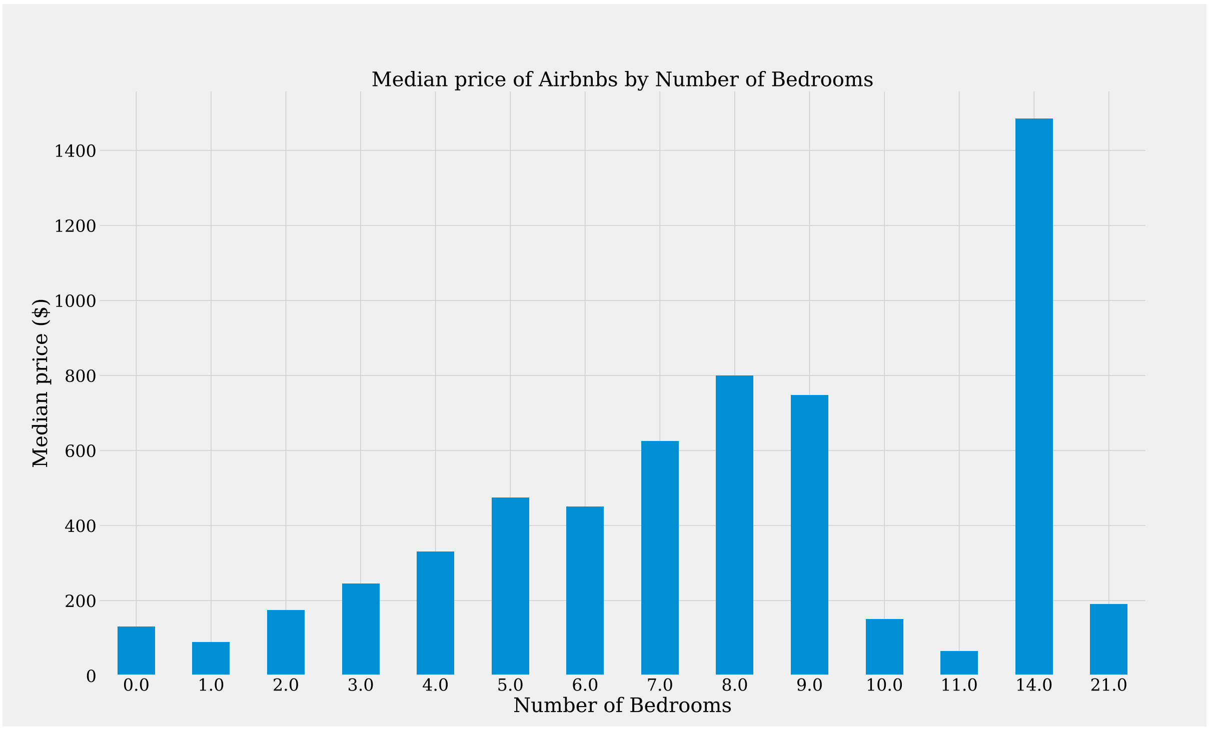


**Figure A.7:** Median Price By Number of Bedrooms

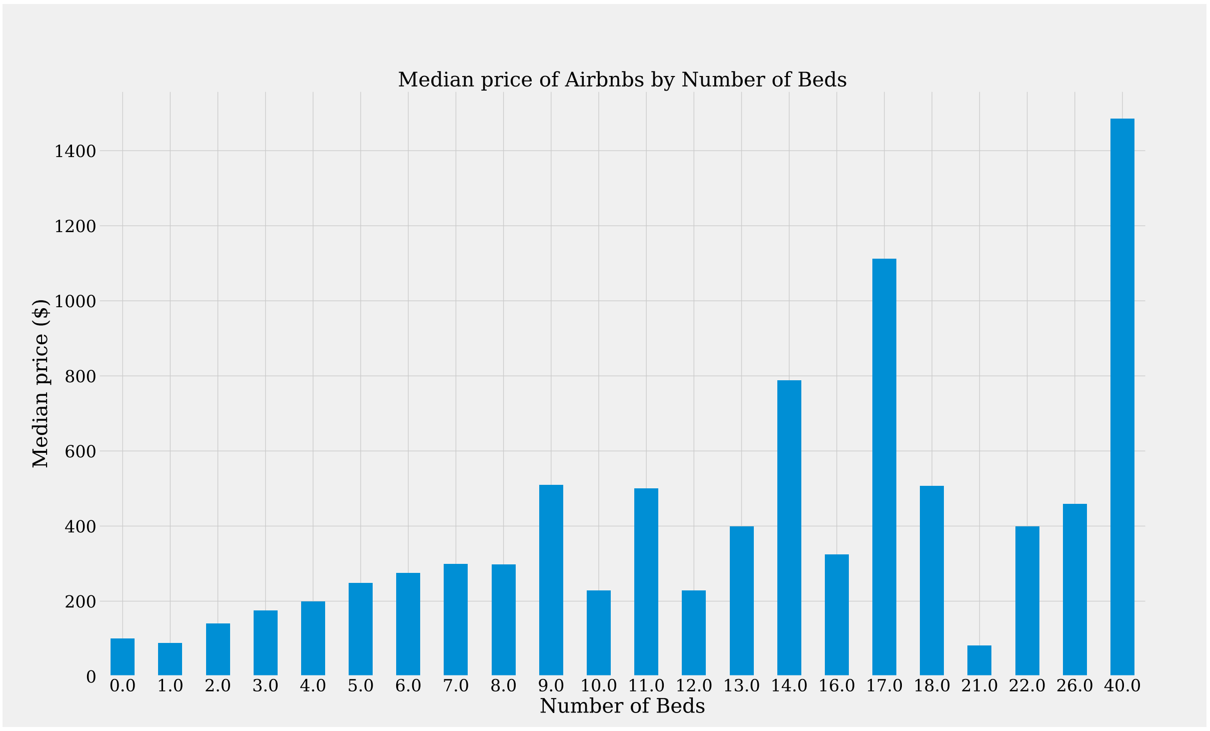


**Figure A.8:** Median Price By Number of Beds

**Figure A.9:** Elevator and Bed Linen

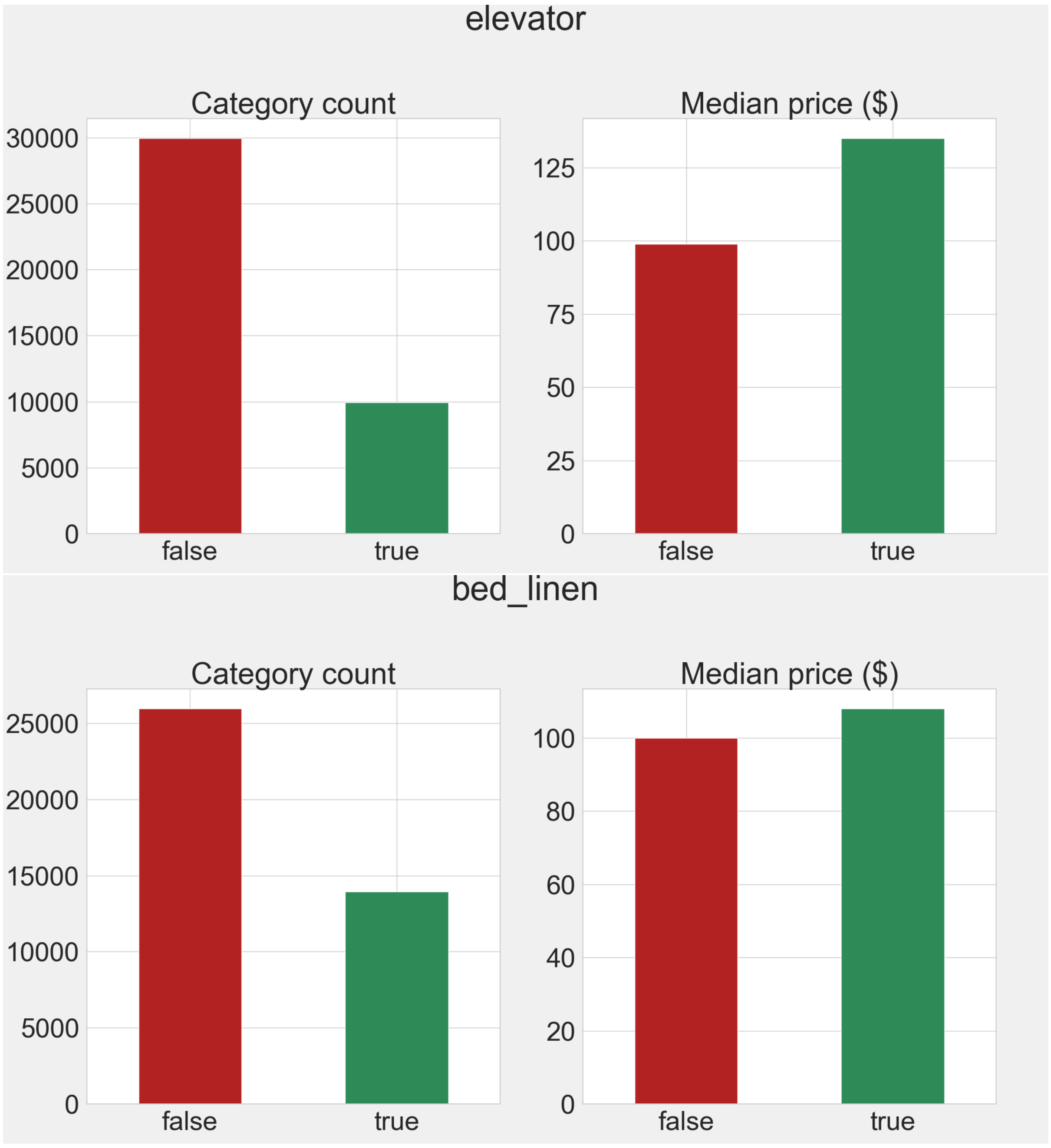


**Figure A.10:** White Goods and Pets Allowed

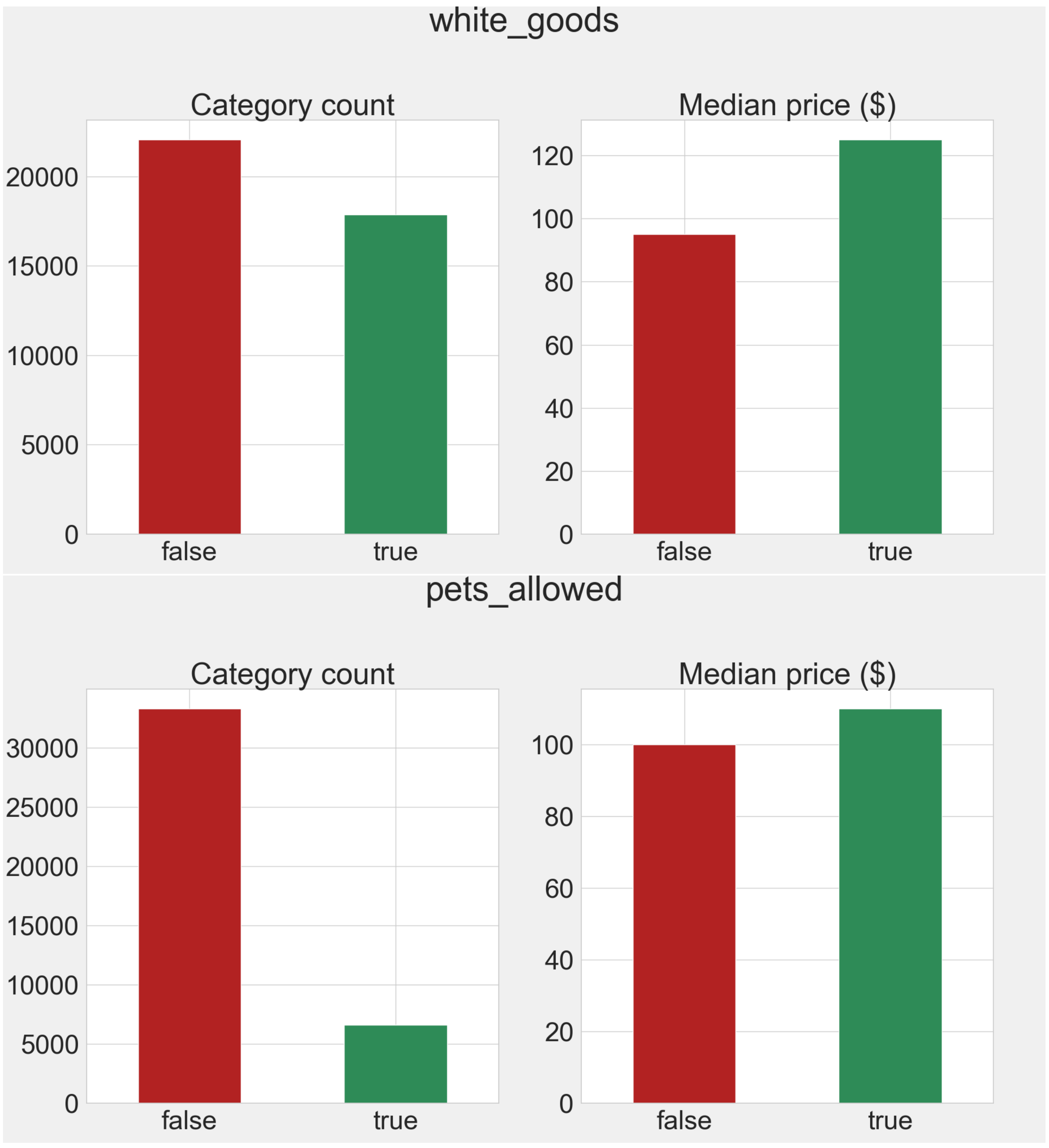


**Figure A.11:** Self Check In and Coffee Machine

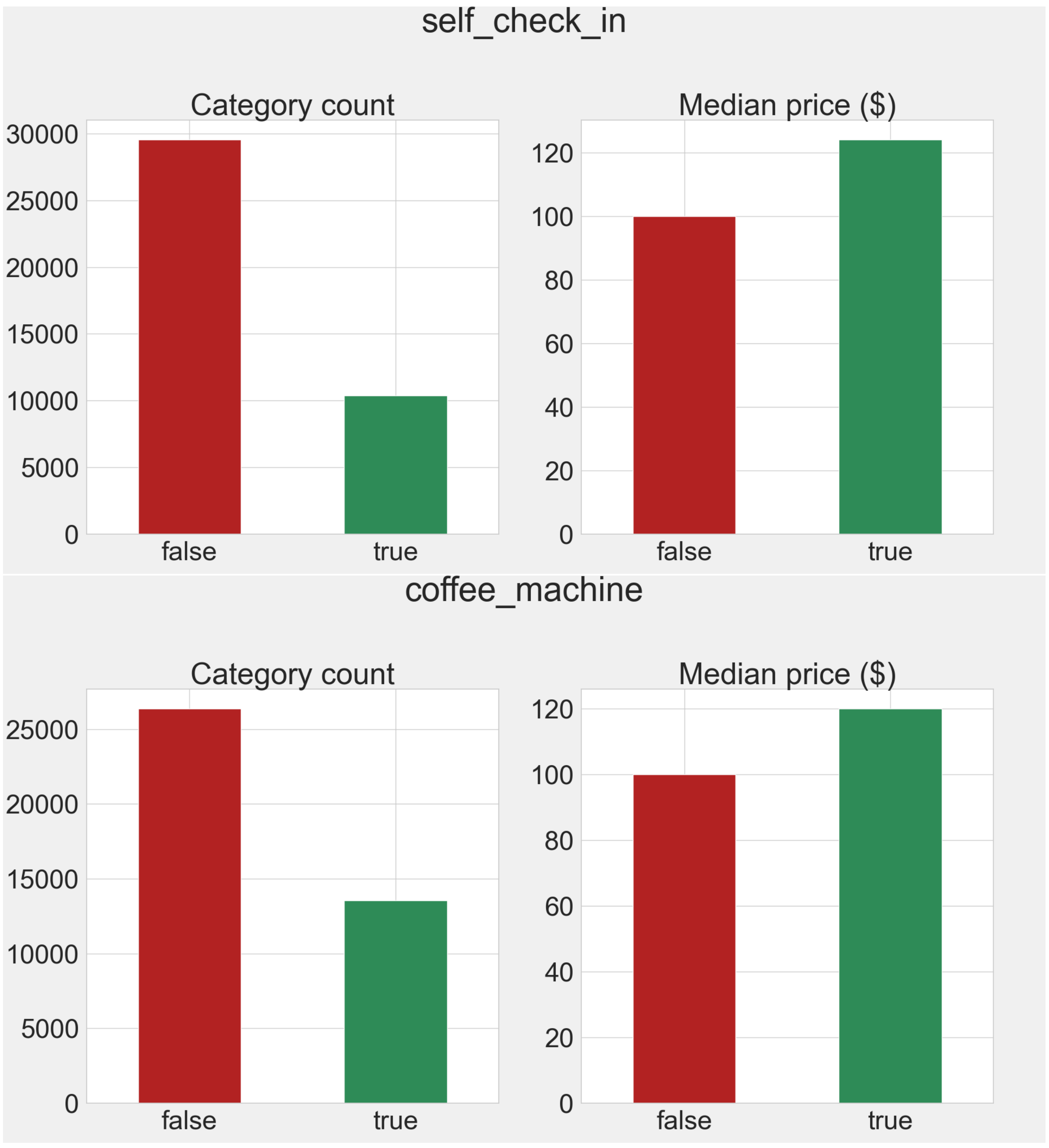


**Figure A.12:** Long Term Stays and Child Friendly

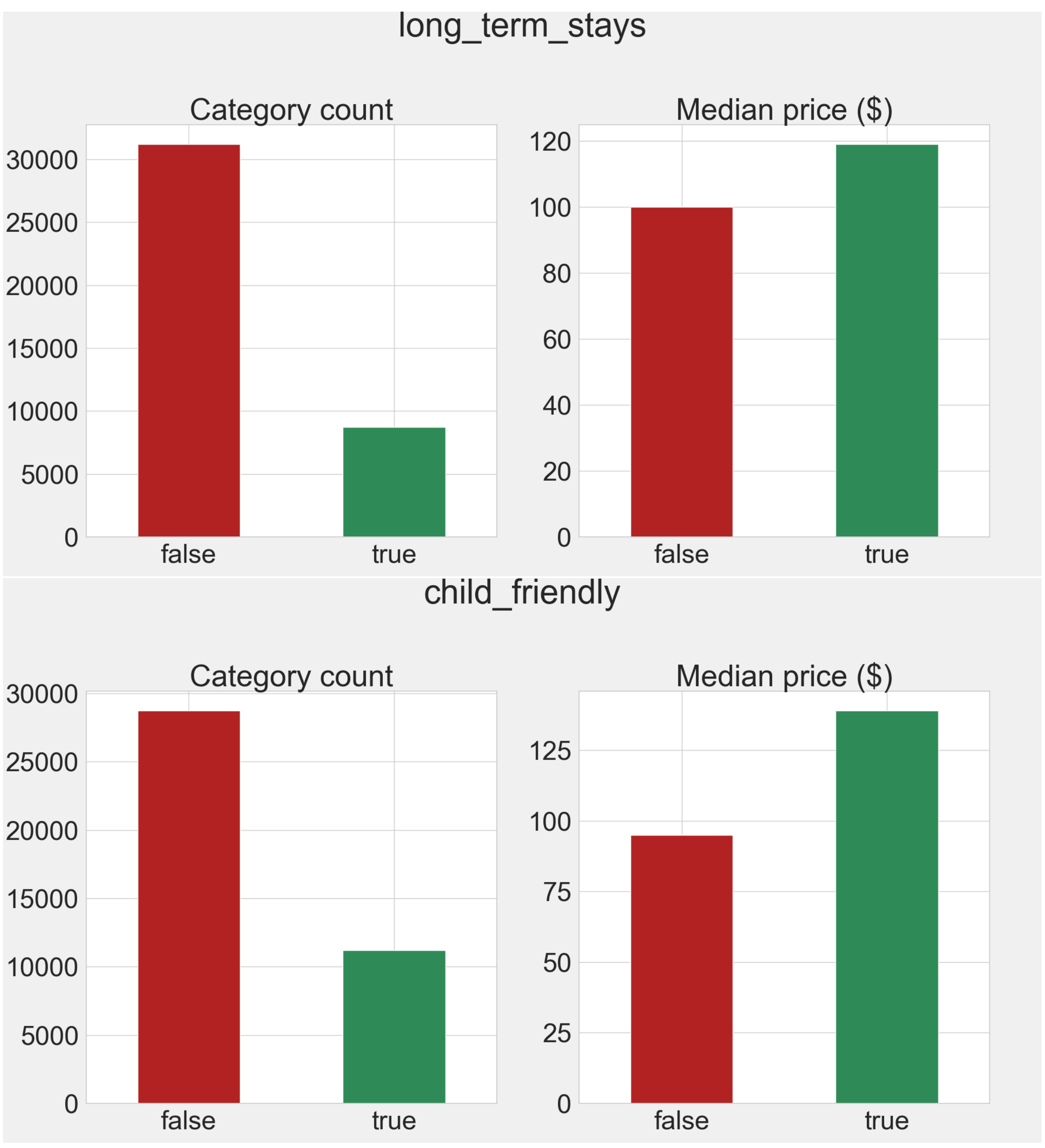


**Figure A.13:** Private Entrance and Cooking Basics

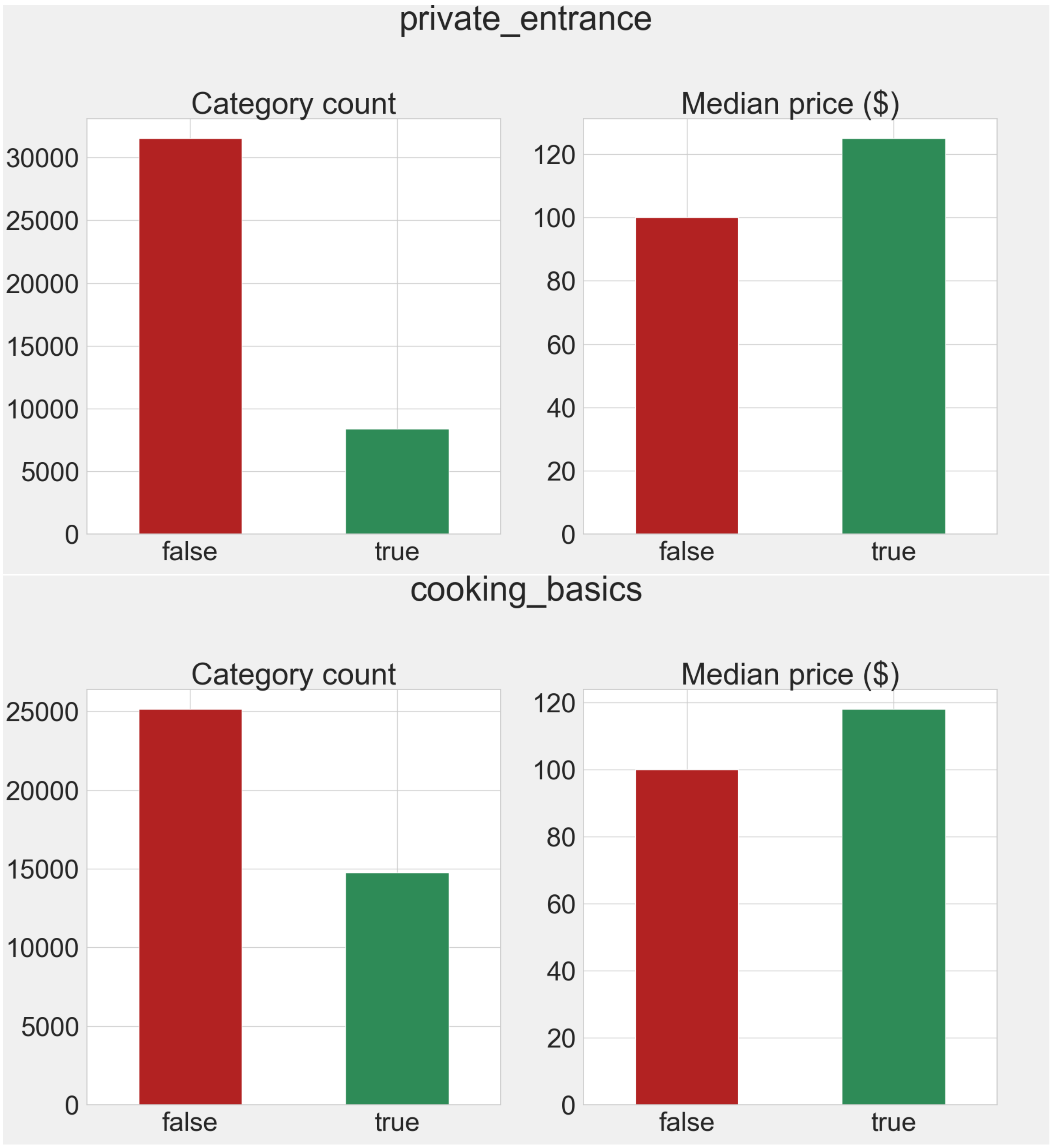


**Figure A.14:** TV and Internet

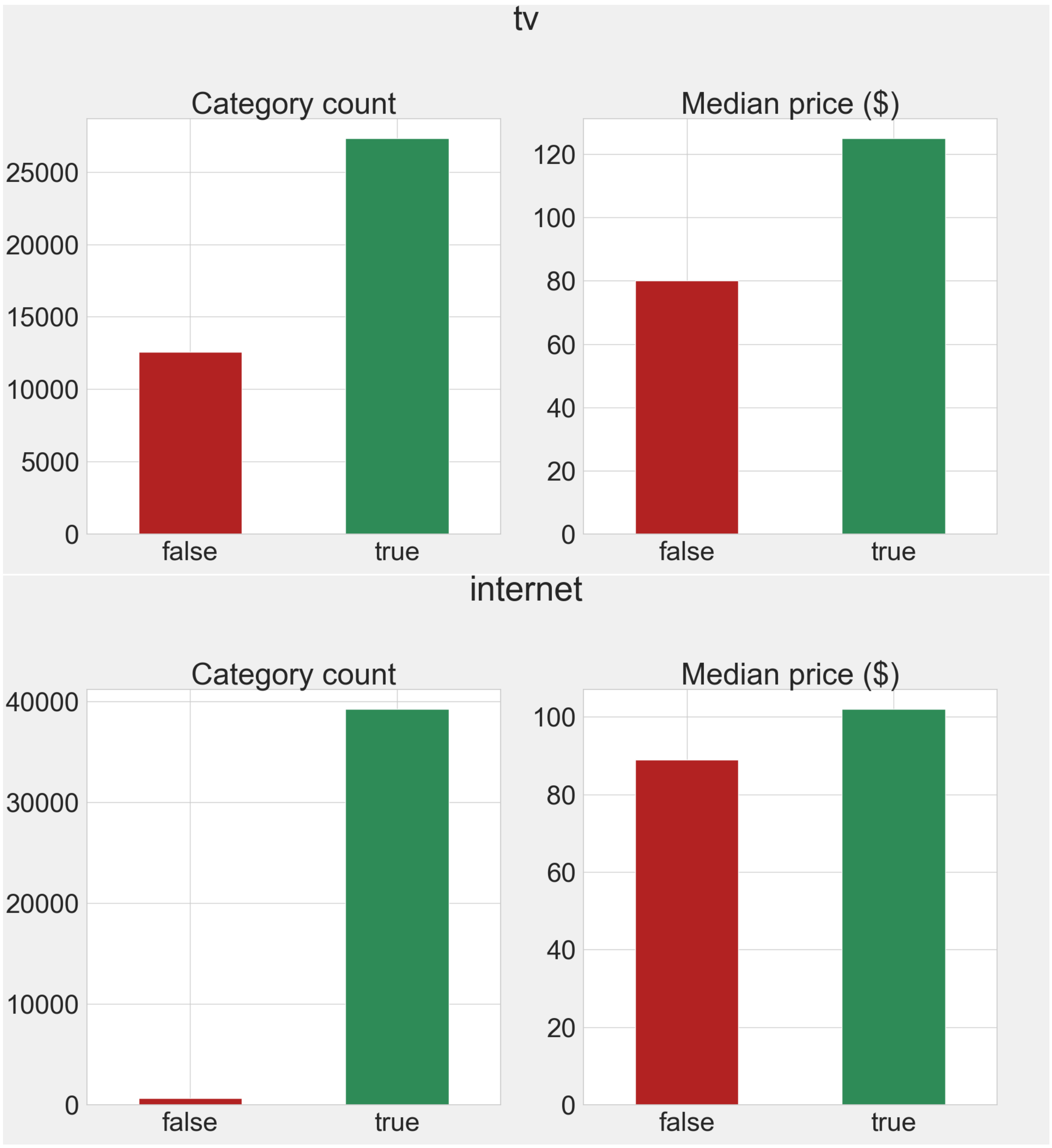


**Figure A.15:** Air Conditioner

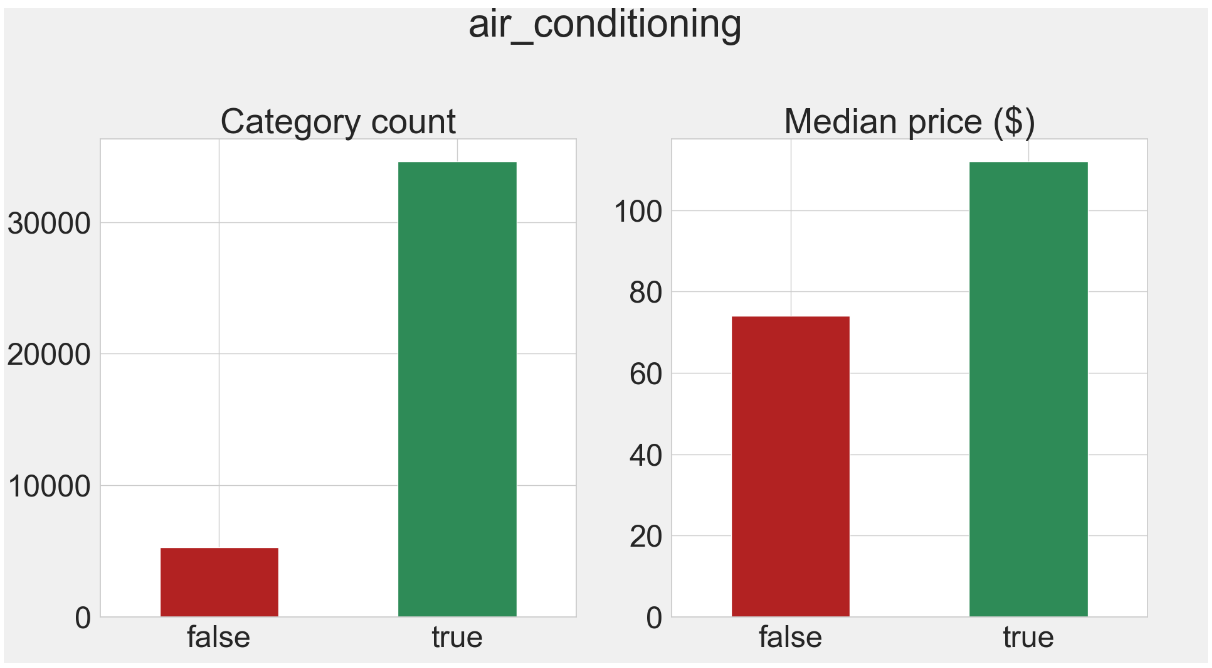
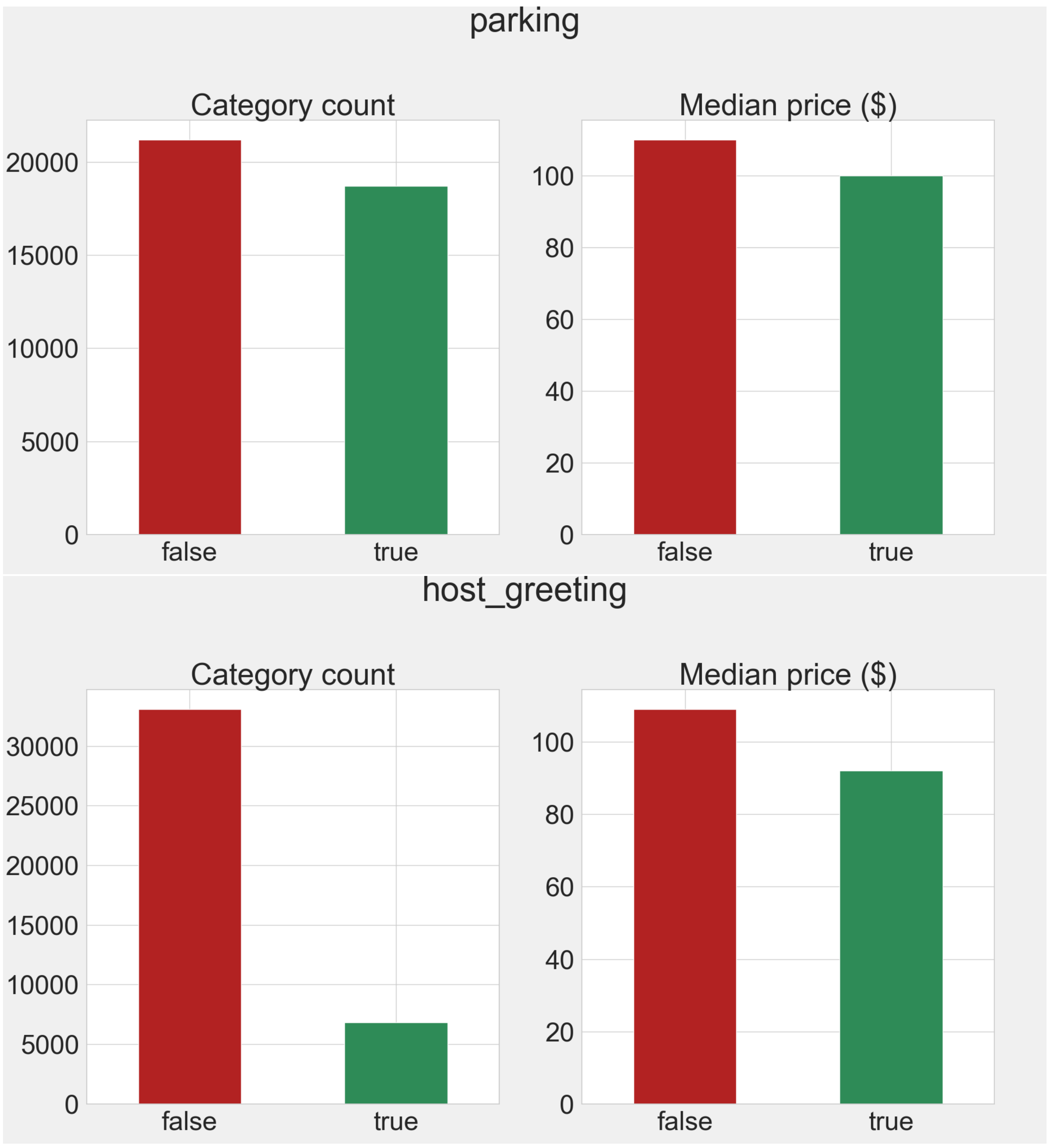


**Figure A.16:** Parking and Host Greeting



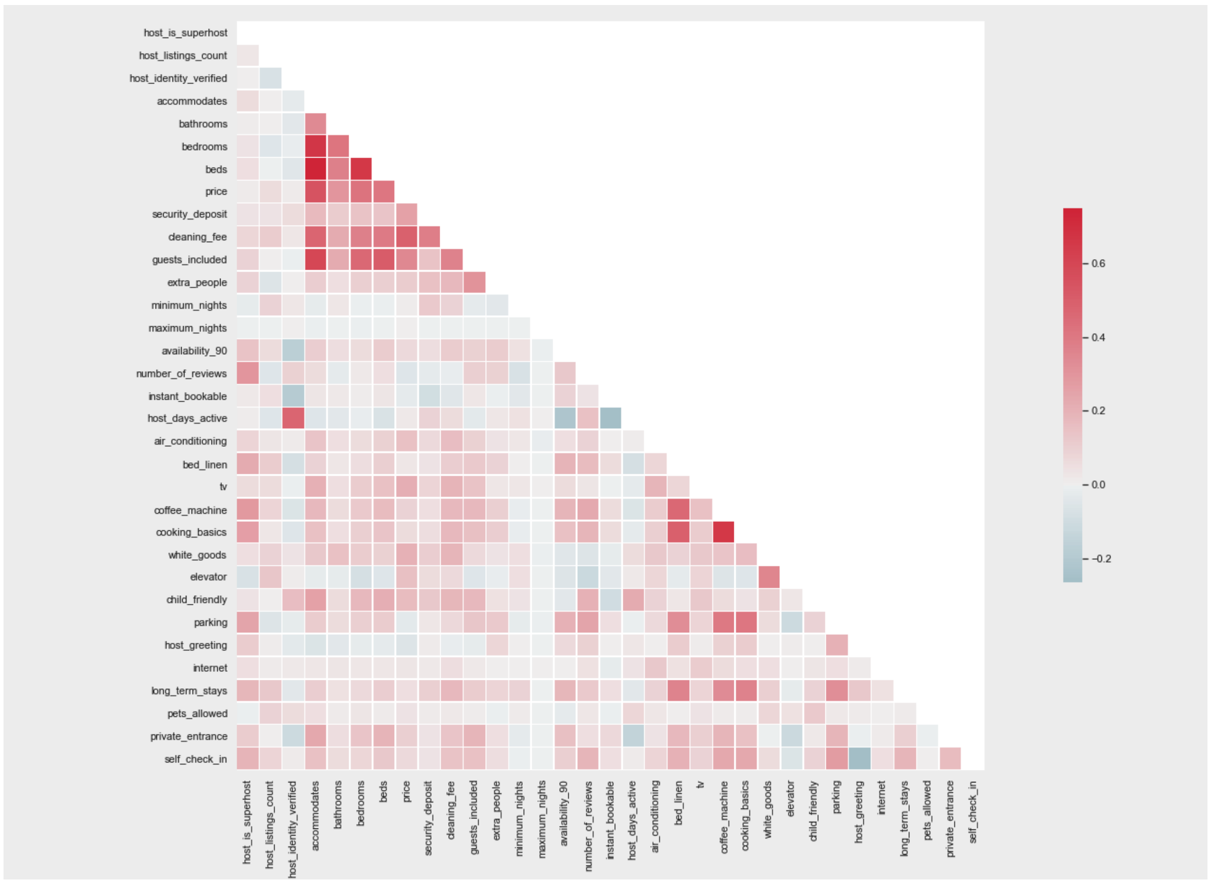


**Figure A.17:** Correlation Matrix

**Figure A.18:** Histogram of Feature Distribution

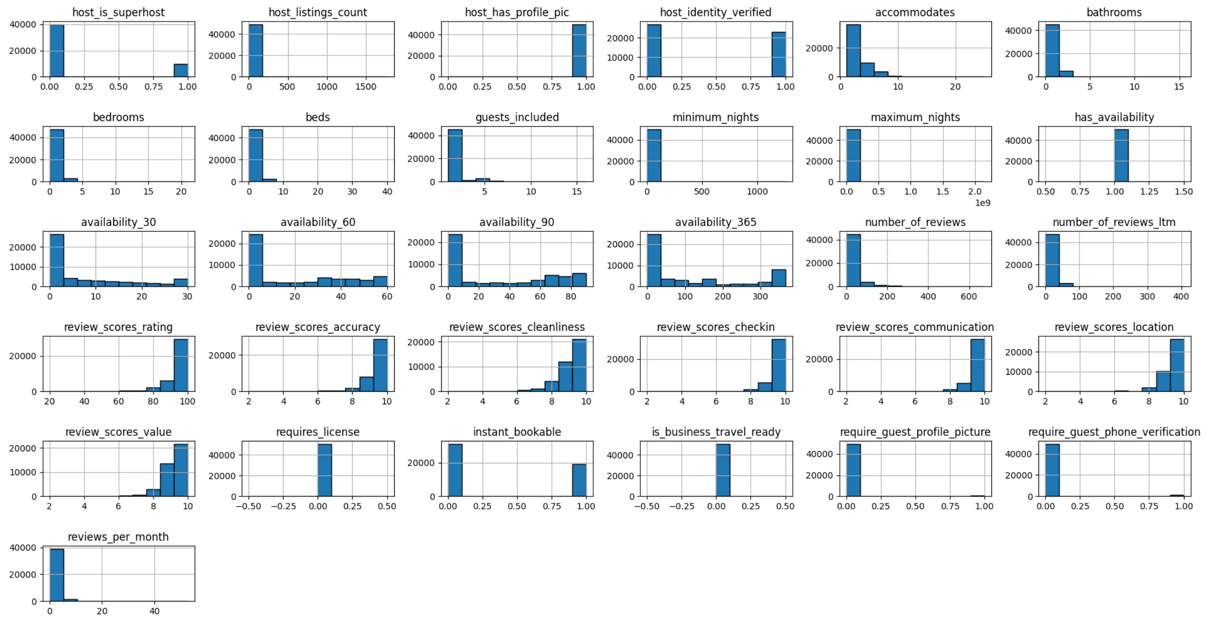


**Figure A.19:** Histogram of Numerical Feature Distribution Before Log Transform

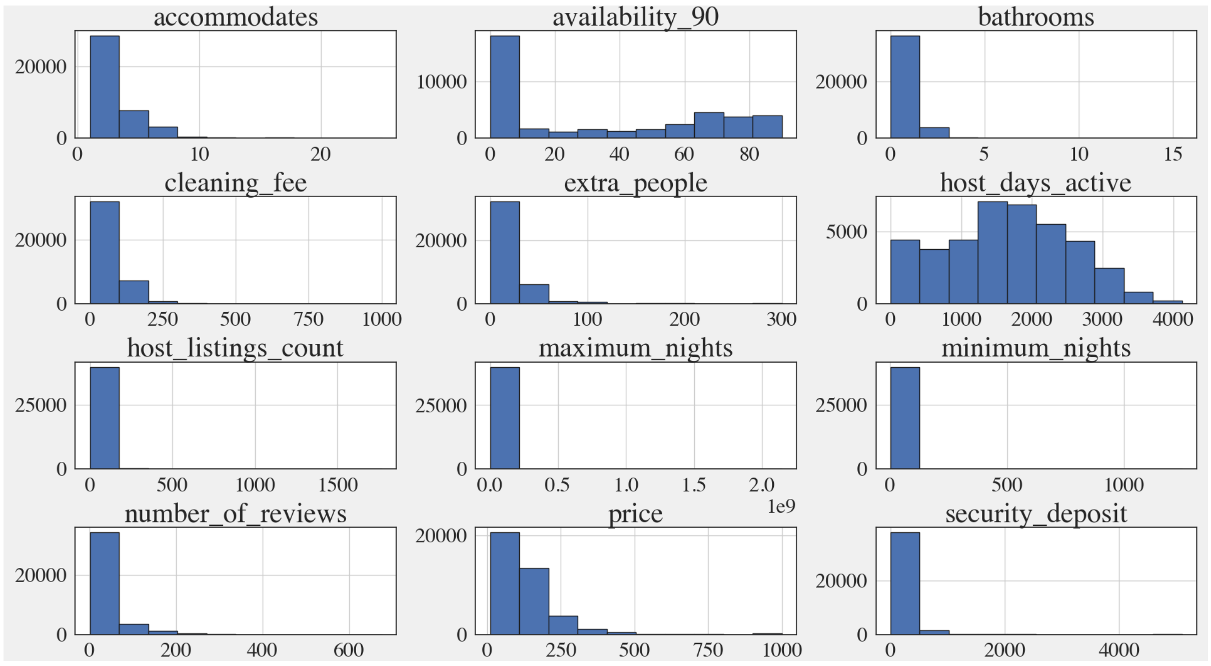
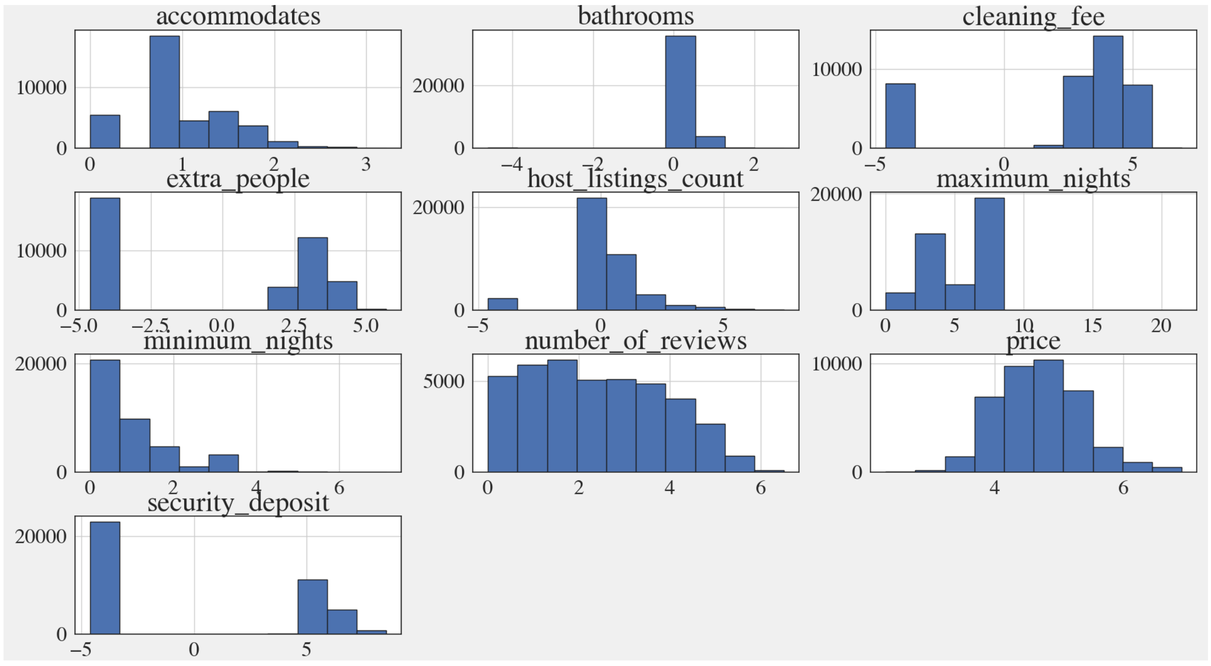


**Figure A.20:** Histogram of Numerical Feature Distribution After Log Transform



# Bibliography

Zaleski, Olivia (2018). *Inside Airbnb's Battle to Stay Private*. `https://www.bloomberg.com/news/articles/2018-02-06/inside-airbnb-s-battle-to-stay-private`.

Airbnb (2020). *Airbnb About Us*. `https://news.airbnb.com/about-us/`.

Monty, Ben and Mark Skidmore (2003). "Hedonic pricing and willingness to pay for bed and breakfast amenities in Southeast Wisconsin". In: *Journal of Travel Research* Vol. 42, No. 2, pp. 195–199.

Portolan, Ana (2013). "Impact of the attributes of private tourist accommodation facilities onto prices: A hedonic price approach". In: *European Journal of Tourism Research* Vol. 6, No. 1, p. 74.

Gutt, Dominik and Philipp Herrmann (2015). "Sharing means caring? Hosts' price reaction to rating visibility". In: *ECIS 2015 Research-in-Progress Papers*, p. 54.

Ikkala, Tapio and Airi Lampinen (2014). "Defining the price of hospitality: networked hospitality exchange via Airbnb". In: *Proceedings of the companion publication of the 17th ACM conference on Computer supported cooperative work & social computing*, pp. 173–176.

Li, Jun, Antonio Moreno, and Dennis Zhang (2016). "Pros vs joes: Agent pricing behavior in the sharing economy". In: *Ross School of Business Paper* No. 1298.

Kakar, Venoo, Julisa Franco, Joel Voelz, and Julia Wu (2016). "Effects of host race information on Airbnb listing prices in San Francisco". In:

Wang, Dan and Juan L Nicolau (2017). "Price determinants of sharing economy based accommodation rental: A study of listings from 33 cities on Airbnb. com". In: *International Journal of Hospitality Management* Vol. 62, pp. 120–131.

Cai, Yuan, Yongbo Zhou, Noel Scott, and Jianyu Ma (2019). "Price determinants of Airbnb listings: evidence from Hong Kong". In: *Tourism Analysis* Vol. 24, No. 2, pp. 227–242.

Tang, Emily and Kunal Sangani (2015). *Neighborhood and price prediction for San Francisco Airbnb listings*. `http://cs229.stanford.edu/proj2015/236_report.pdf`. [Online].

Li, Yang, Quan Pan, Tao Yang, and Lantian Guo (2016). "Reasonable price recommendation on Airbnb using Multi-Scale clustering". In: *2016 35th Chinese Control Conference (CCC)*. IEEE, pp. 7038–7041.

Kalehbasti, Pouya Rezazadeh, Liubov Nikolenko, and Hoormazd Rezaei (2019). "Airbnb price prediction using machine learning and sentiment analysis". In: *arXiv preprint arXiv:1907.12665*.

Shah, Chirag (2020). *A Hands-On Introduction to Data Science*. Cambridge University Press.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani (2013). *An introduction to statistical learning*. Vol. 112. Springer.

Rosen, Sherwin (1974). "Hedonic prices and implicit markets: product differentiation in pure competition". In: *Journal of political economy* Vol. 82, No. 1, pp. 34–55.

Graybill, Franklin A (1976). *Theory and application of the linear model*. Vol. 183. Duxbury press North Scituate, MA.

Harrell Jr, Frank E (2015). *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer.

Hoerl, Arthur E and Robert W Kennard (1970). "Ridge regression: Biased estimation for nonorthogonal problems". In: *Technometrics* Vol. 12, No. 1, pp. 55–67.

Tibshirani, Robert (1996). "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* Vol. 58, No. 1, pp. 267–288.

Efron, Bradley and Robert J Tibshirani (1994). *An introduction to the bootstrap*. CRC press.

Kuhn, Max and Kjell Johnson (2013). *Applied predictive modeling*. Vol. 26. Springer.

Chen, Tianqi and Carlos Guestrin (2016). "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

Nielsen, Didrik (2016). "Tree boosting with xgboost-why does xgboost win" every" machine learning competition?" MA thesis. NTNU.

*Inside Airbnb* (2019). `http://insideairbnb.com/get-the-data.html`. [Online; accessed 04-Dec-2019].

Ye, Qiang, Rob Law, and Bin Gu (2009). "The impact of online user reviews on hotel room sales". In: *International Journal of Hospitality Management* Vol. 28, No. 1, pp. 180–182.

Tukey, John W (1977). *Exploratory data analysis*. Vol. 2. Reading, MA.

Chen, Yong and Karen Xie (2017). "Consumer valuation of Airbnb listings: a hedonic pricing approach". In: *International journal of contemporary hospitality management*.

Ert, Eyal, Aliza Fleischer, and Nathan Magen (2016). "Trust and reputation in the sharing economy: The role of personal photos in Airbnb". In: *Tourism management* Vol. 55, pp. 62–73.

Gibbs, Chris, Daniel Guttentag, Ulrike Gretzel, Lan Yao, and Jym Morton (2018). “Use of dynamic pricing strategies by Airbnb hosts”. In: *International Journal of Contemporary Hospitality Management* Vol. 30, No. 1, pp. 2–20.

Zhang, Zhihua, Rachel JC Chen, Lee D Han, and Lu Yang (2017). “Key factors affecting the price of Airbnb listings: A geographically weighted approach”. In: *Sustainability* Vol. 9, No. 9, p. 1635.

Perez-Sanchez, V Raul, Leticia Serrano-Estrada, Pablo Marti, and Raul-Tomas Mora-Garcia (2018). “The what, where, and why of Airbnb price determinants”. In: *Sustainability* Vol. 10, No. 12, p. 4596.

Benitez-Aurioles, Beatriz (2018). “Why are flexible booking policies priced negatively?” In: *Tourism Management* Vol. 67, pp. 312–325.

Cohen, Jacob, Patricia Cohen, Stephen G West, and Leona S Aiken (2013). *Applied multiple regression/correlation analysis for the behavioral sciences*. Routledge.

Kutner, Michael H, Christopher J Nachtsheim, John Neter, and William Li (2005). *Applied linear statistical models*. Vol. 5. McGraw-Hill Irwin New York.