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. XXX

Academic Year 2020/2021

Acknowledgments

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Contents

List of Figures

List of Tables

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Keywords:

Chapter 1

Introduction

Pricing a rental property on Airbnb is a challenging task for the owner as it determines the number of customers for the place

1.1 Motivation and Goals

In recent years

1.2 Literature Review

Parts of the existing literature on property pricing focus on non-shared property purchase or rental price prediction

Chapter 2

Data Set

2.1 Data Acquisition

The dataset used for this study comes from the Inside Airbnb website *Inside Airbnb* [10]. 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... A GeoJSON file of New York borough boundaries was also downloaded from the same site.

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.

2.2 Data Pre-Processing

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 three key methods that illustrate ways in which data may be "cleaned," or better organized, or scrubbed of potentially incorrect, incomplete, or duplicated data.

2.2.1 Data Filtering

Following Wang and Nicolau [18], 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, Law, and Gu [19] which confirms that online review ratings are associated with hotel-room sales, indicating that reviews suggest real transactions.

2.2.2 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.4) .
- 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.

2.2.3 Dealing with Missing Value

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 2.1.

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 2.1: Columns with majority of null values

Column	Number of missing value
host_acceptance_rate	50599
jurisdiction_names	50583
license	50577
square_feet	50213
monthly_price	45683
weekly_price	44945

2.2.4 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`).

2.2.5 Binning Predictors

We want to separate 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'

2.2.6 Data Transformation

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

2.2.6.1 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 σ is its standard deviation.

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

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

2.2.7 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.

Pandas Library provides the `get_dummies()` method to perform One-Hot enconding

Chapter 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 [17] 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

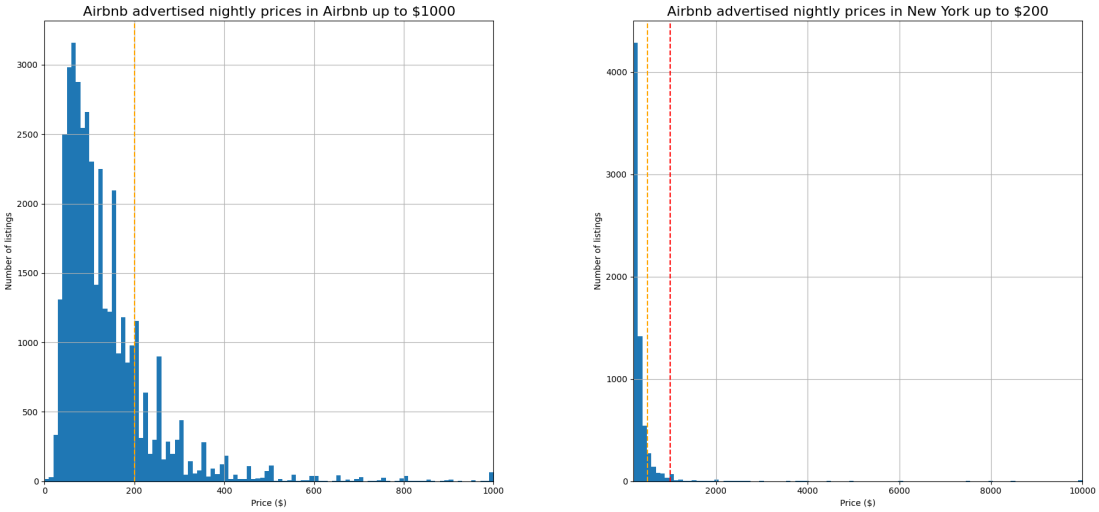
3.1 Numerical Features

3.1.1 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 3.1a and Figure 3.1b 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: Price Distribution

(a) Airbnb advertised price up to $1000 **(b)** Airbnb advertised price up to $200



3.1.2 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 own many listings. About 55% of listings are from hosts with one listing, and 45% are from multi-listing hosts.

3.1.3 Number of people accommodated, bathrooms, bedrooms and beds

Figure 3.2 reveals that the most common listing type accommodates two people in one bed in one bedroom with one bathroom.

Figure 3.2: Histogram Plot of Accomodate, Bathrooms, Bedrooms, and Beds

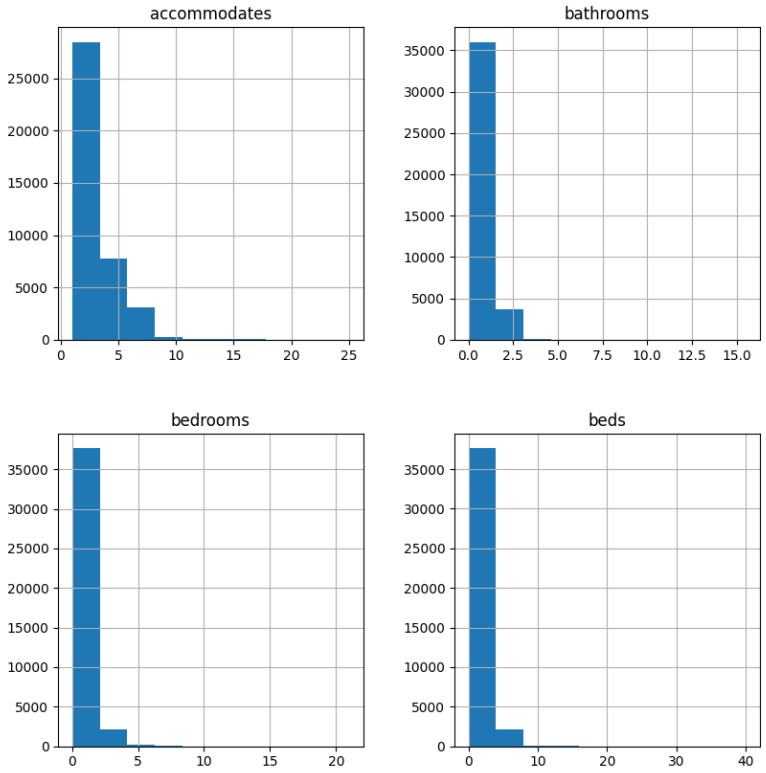
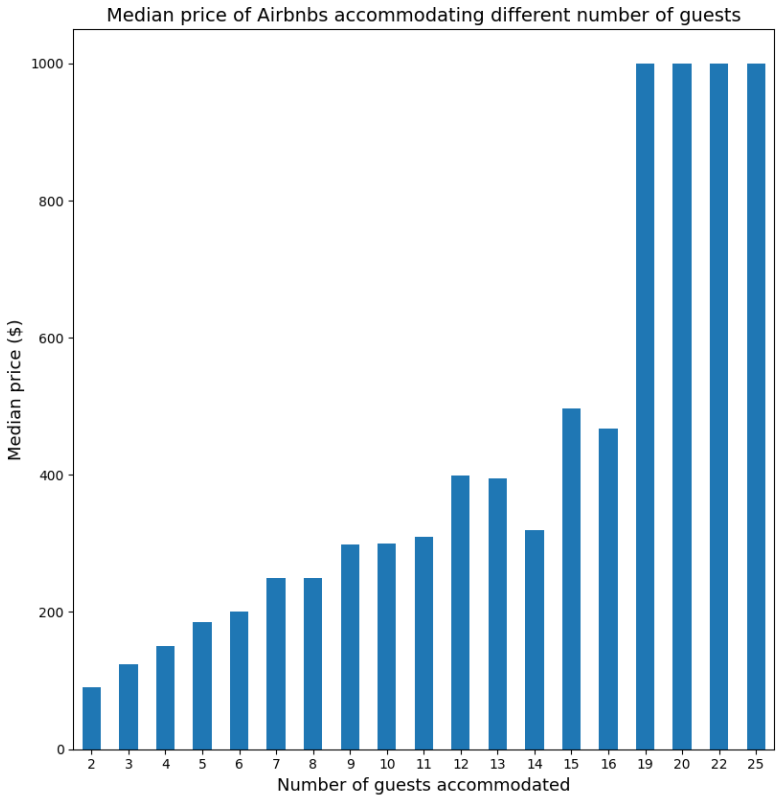


Figure 3.3 shows that the more people a listing accommodates, the higher the price they can charge their customers.

Figure 3.3: Median Price of Airbnb Listing Accommodating Different Number Of Guests



3.2 Categorical features

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

3.2.1 Neighbourhood

Manhattan and Brooklyn have the most Airbnb properties, followed by Queens (Figure 3.5)

Figure 3.4: Number of Airbnb listings in each New York borough

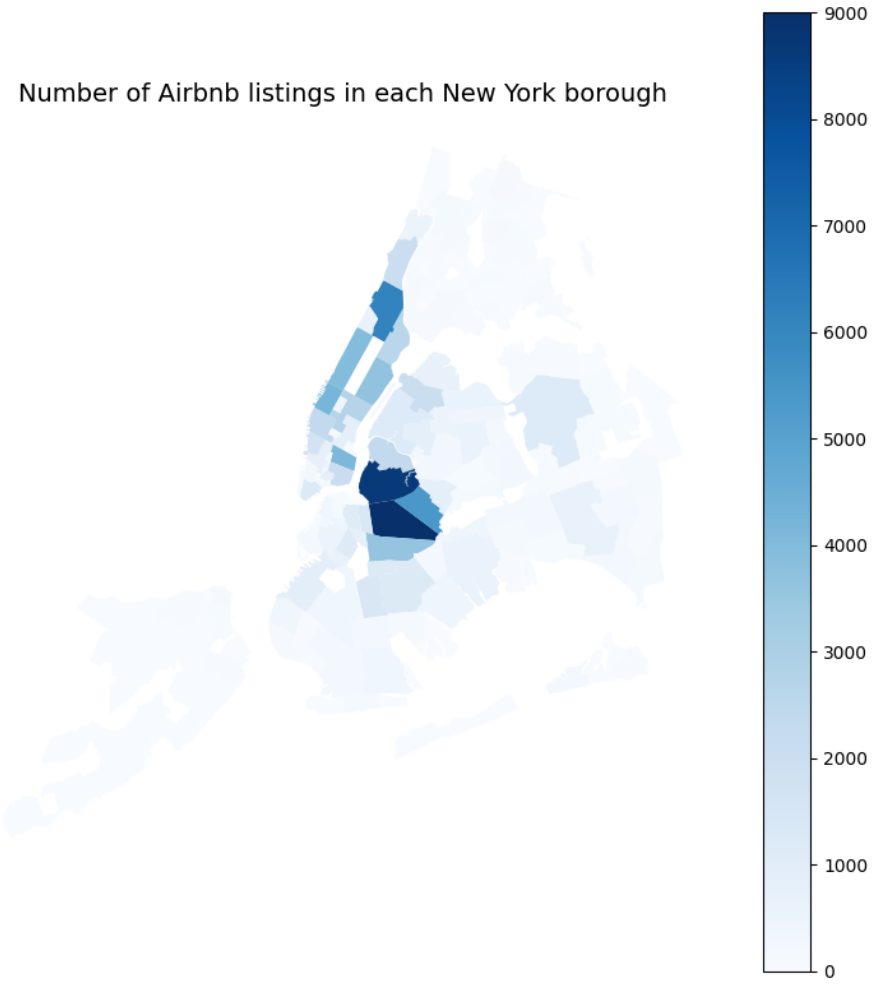


Figure 3.5: Borough Listings

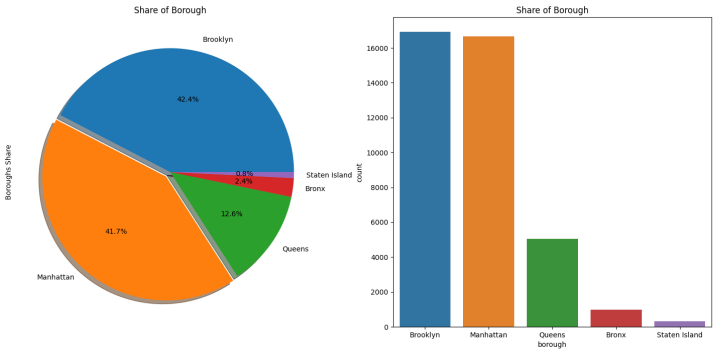
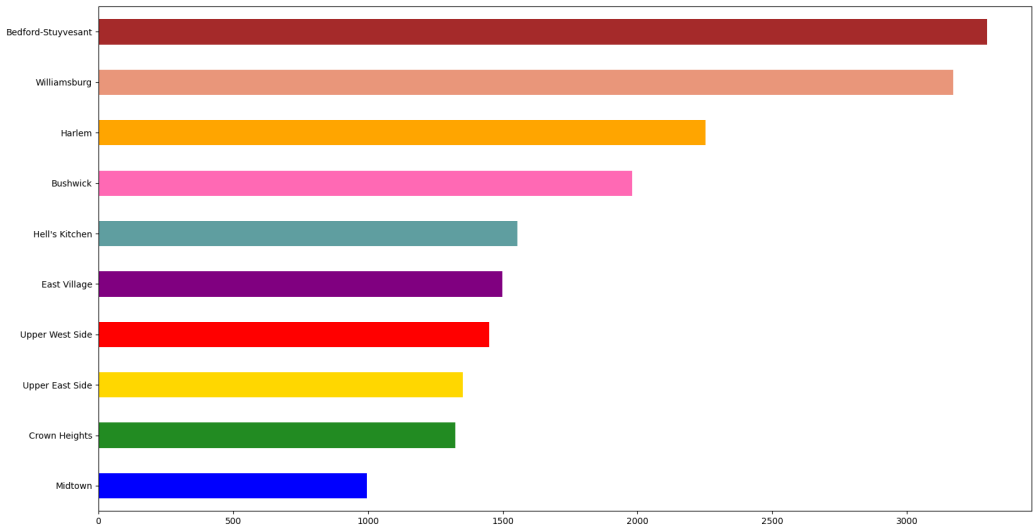


Figure below reveal top ten neighbourhood with most listings:

Figure 3.6: Top 10 Neighbourhood with Most Listings



As shown in Figure 3.7 and Figure 3.8 , Unsurprisingly, Manhattan is the most expensive borough - this is a famously expensive area to live, with some of the world's highest house prices. The second most expensive area is in Brooklyn, followed by Queens. Staten Island is the least expensive area to rent Airbnb accommodation.

Figure 3.7: Median Price of Airbnb listings in each New York borough

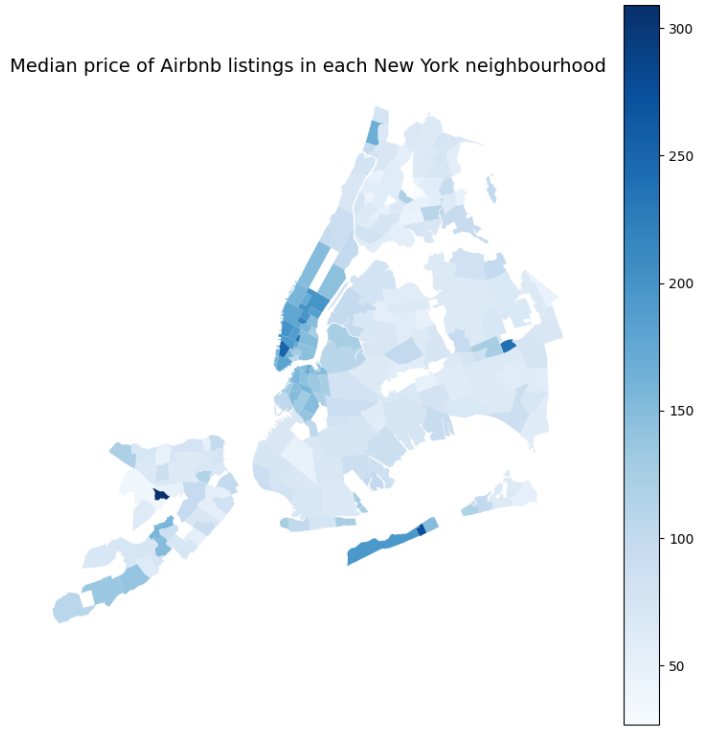
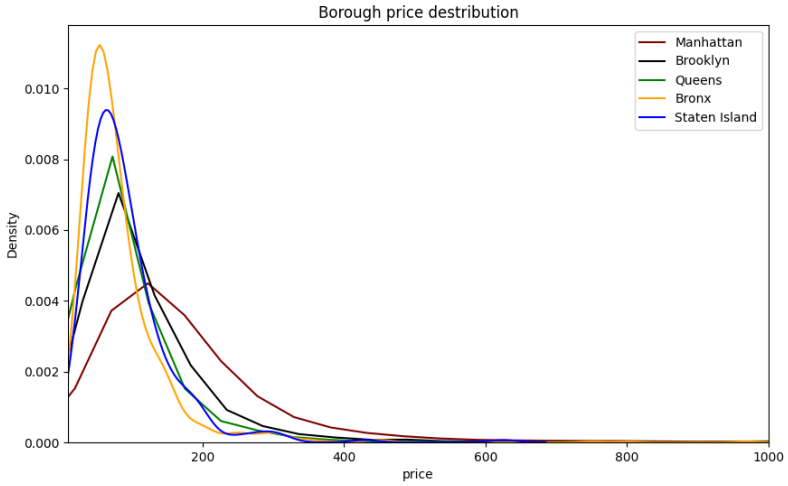


Figure 3.8: Borough Price Distribution



3.2.2 Property and room types

As shown in Figure 3.9, about 80% properties are apartments. The remainder are houses or more uncommon property types (e.g. 'bed and breakfast' or 'yurt').

(b) Property Type Bar Chart

(a) Property Type Pie Chart

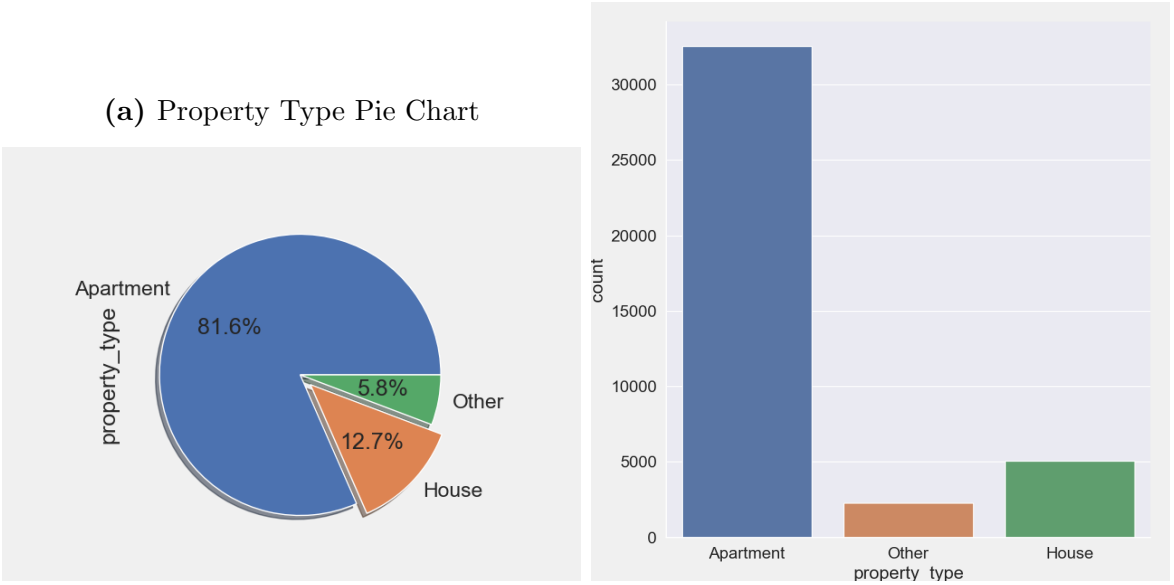
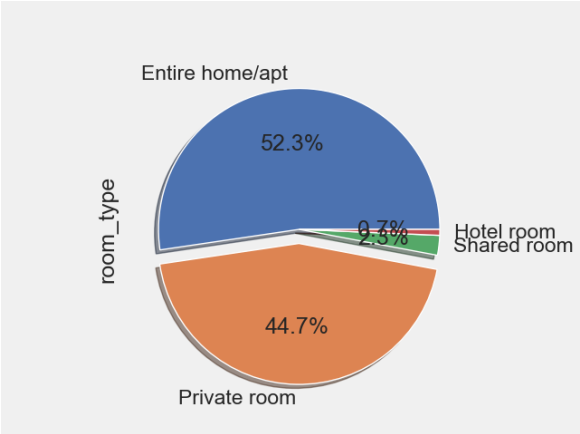


Figure 3.9: Property Type

Figure 3.10 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).

(a) Room Type Pie Chart



(b) Room Type Bar Chart

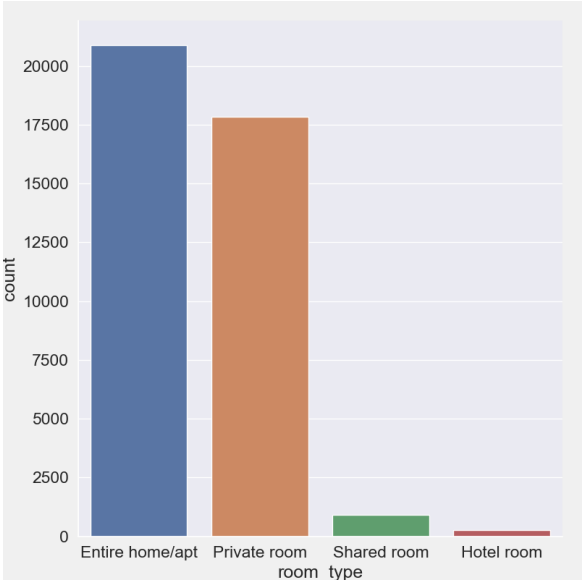
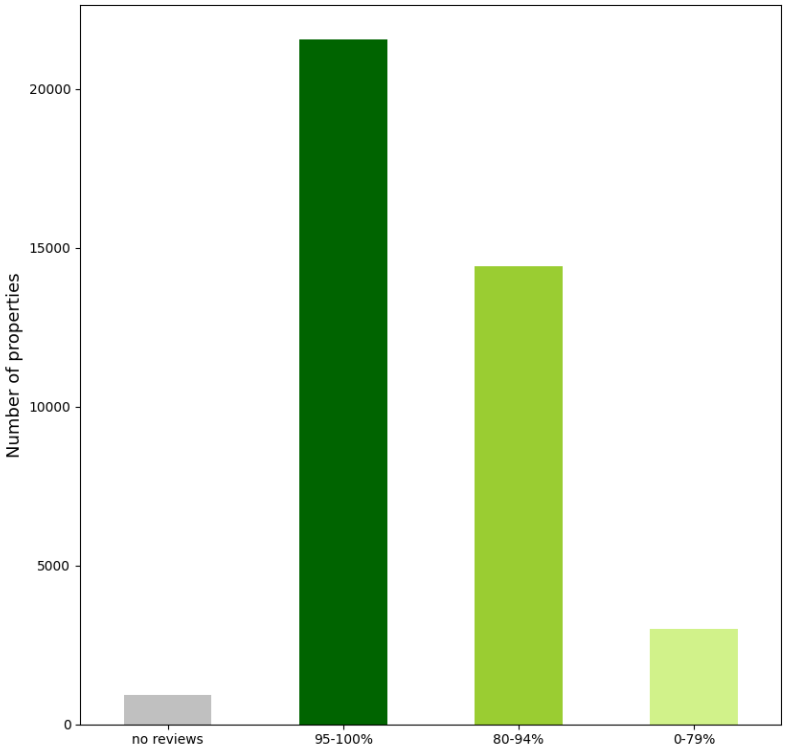


Figure 3.10: Room Type

3.2.3 Reviews

From Figure 3.11, 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.

Figure 3.11: Overall Listing Rating



3.2.4 First and Last Review

As can be seen from the Figure 3.12a, 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.

The bar plot 3.12b 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.

(a) Time Since First Review **(b)** Time Since Last Review

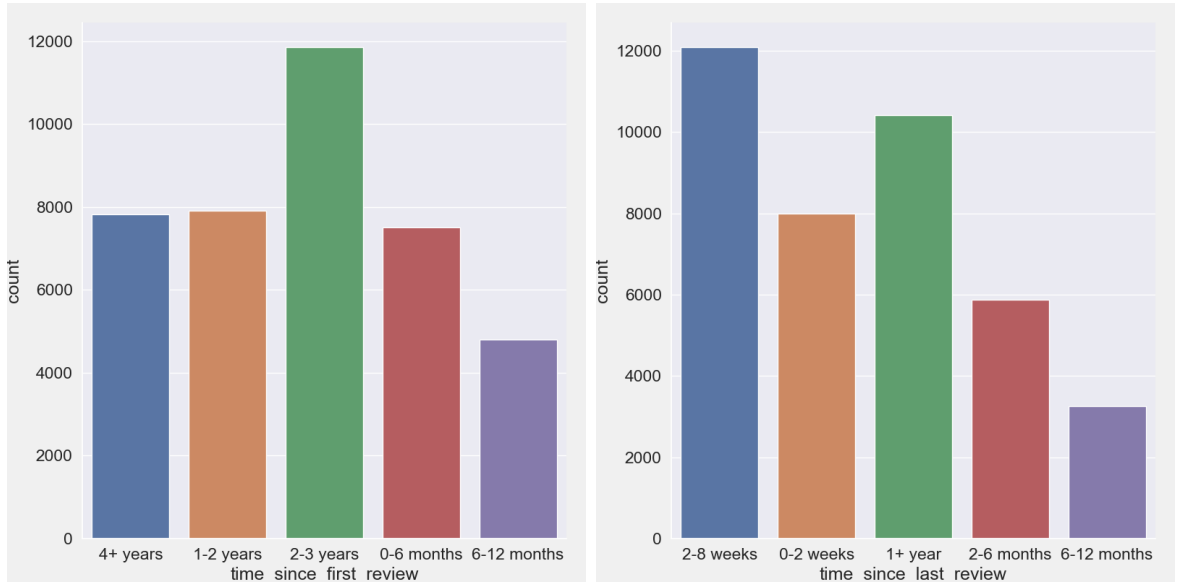


Figure 3.12: First and Last Review

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).

3.3.1 Superhosts

Figure 3.13 shows that about 23% of hosts are "superhosts". As expect that being a superhost (a mark of quality, requiring various conditions to be met) improve the median price of Airbnb listing.

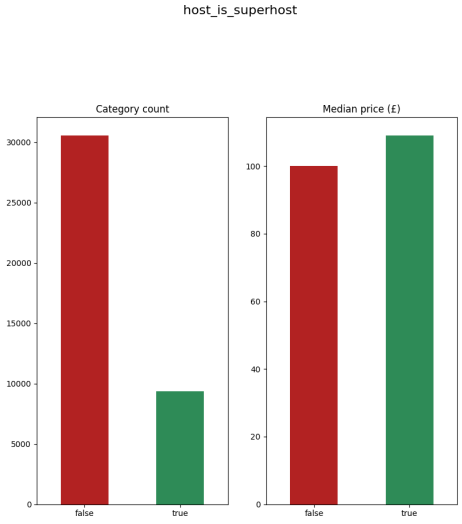


Figure 3.13: host_is_superhost

3.3.2 Host verification

In Figure 3.14, about 49% of hosts are verified. As with superhost, verifying host's profile (e.g. by providing ID and verifying your phone number and email address) can lead to a price premium. The reason for this may have something to do with the fact that providing host's verification increases the trustworthiness of the host (Ert, Fleischer, and Magen [4]).

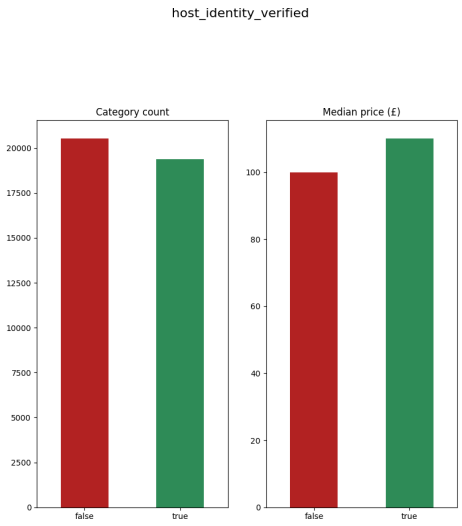


Figure 3.14: host_identity_verified

3.3.3 Instant booking

As shown in figure below, about 40% of properties are instant bookable. However, the added convenience does not seem to have any effect on the median price per night. This negative link can be explained by both emotional (Wang and Nicolau [18]) and economic (Benitez 2018)

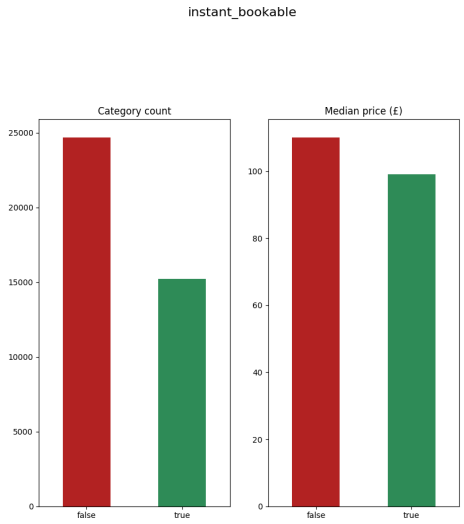


Figure 3.15: instant_bookable

3.3.4 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), Air conditioner.(See Figure A.1)

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

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

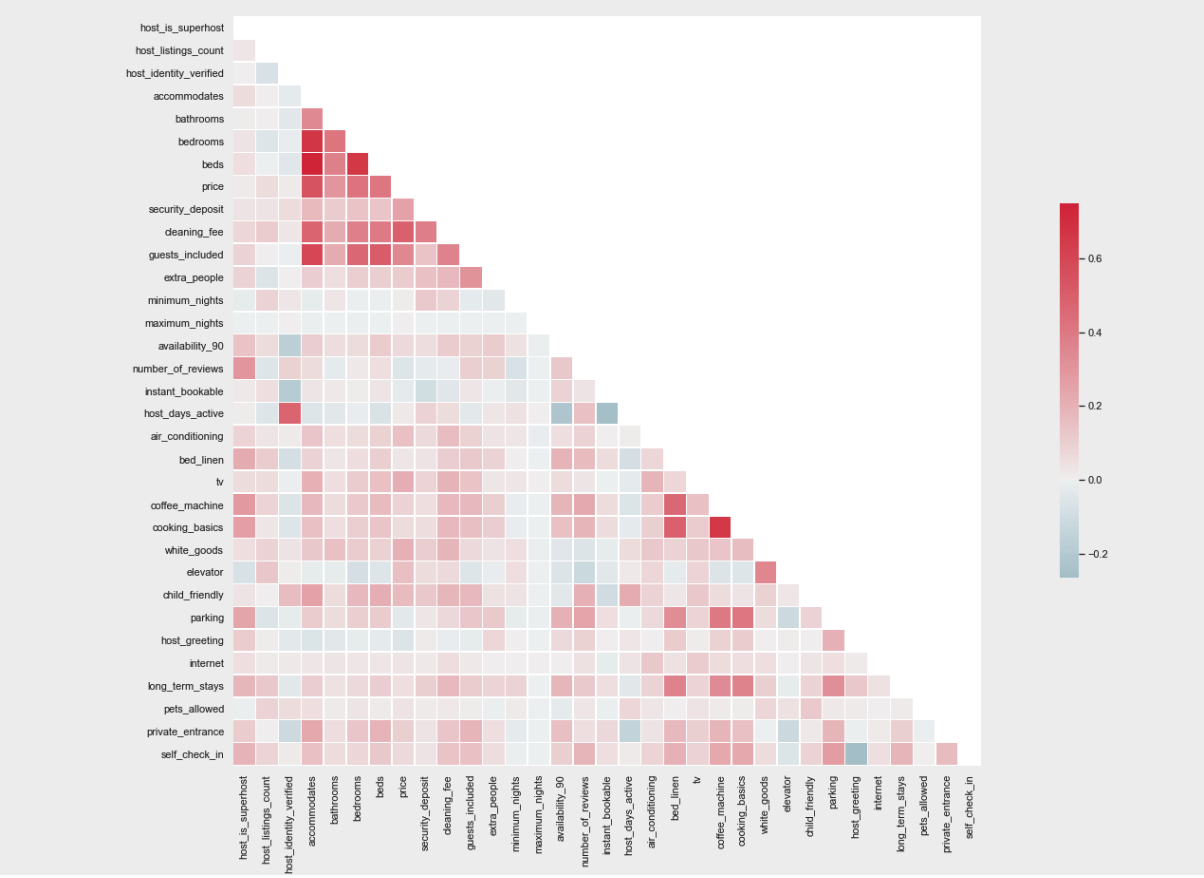
3.4 Multivariate Exploration

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. [2]).

While multicollinearity does not hurt the model's predictive power(Kutner et al. [13]), 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.

Figure 3.16: Correlation Matrix



As shown in Figure 3.16 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(4.4), which reduce the error variance of estimators.

3.5 Time Series Analysis

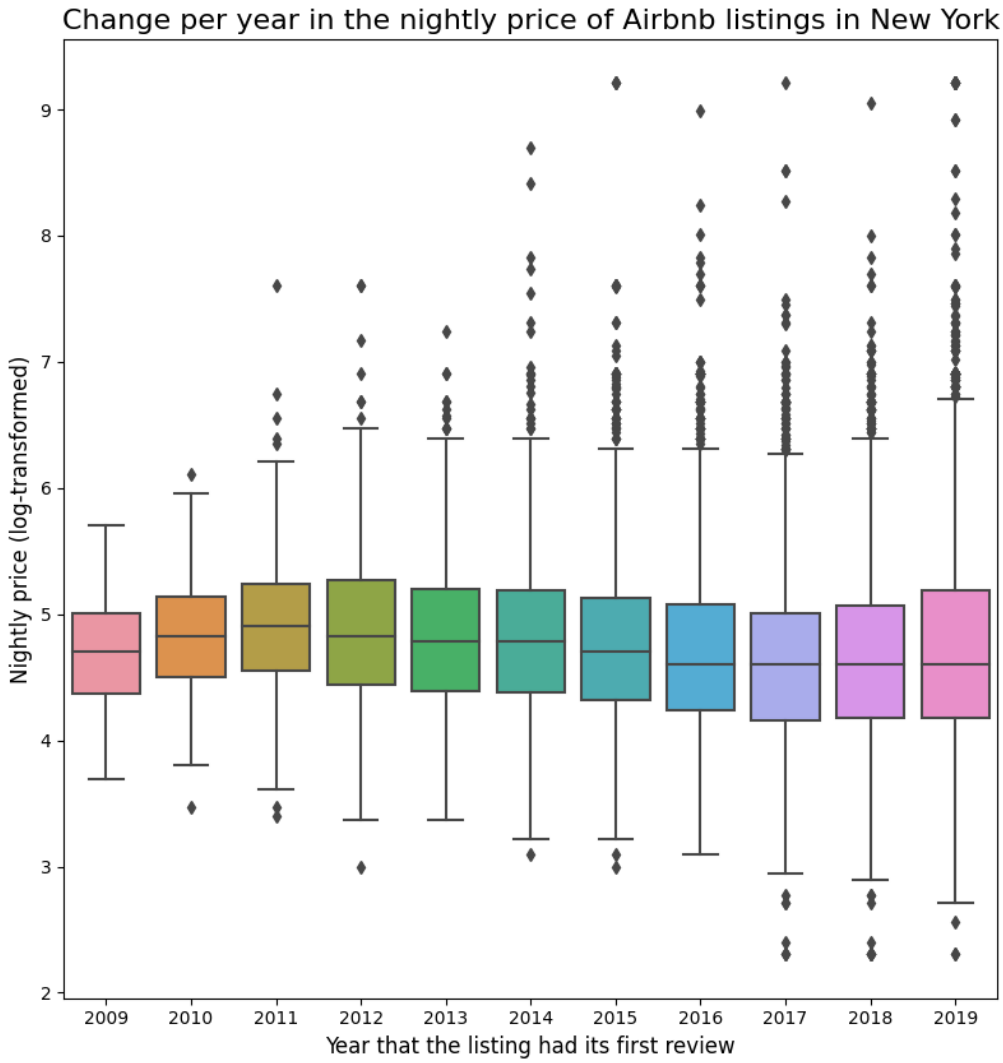
We now also plotted time series charts to check for trends and seasonality. First, we examine how long hosts have been listing properties on Airbnb in New York.Of the Airbnb hosts that are active on the site, the first joined on 22 August 2008, and the most recent joined on 03 December 2019. From 2011 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 (3.17).

Strong evidence of seasonality was found in Figure 3.17. 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 3.17: Number of hosts joining Airbnb each month

In terms of how prices changed over time, Figure 3.18 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 3.18: Change per year in the nightly price of Airbnb listings in New York



Chapter 4

Data Modelling Methods

In this chapter, we discuss the concept of regression and related methods and the role they play in predicting future housing prices. Regression analysis is a statistical technique used to determine the contributing effect of a set of explanatory variables on the response variable.

4.1 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.

4.1.1 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:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{4.1}$$

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.

4.1.2 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)

4.2 The Problem of Overfitting

New technologies have changed how we collected data in various fields as diverse as finance, commerce, and medicine in recent years. It is not unusual to obtain an almost unlimited number of feature measurements. We might expect it will always be better to use as many features as possible in our model. In other words, we suppose that the best model is the most flexible one. However, this is not always the case. Fitting a model with a large number of features can lead to overfitting the data, which means that the model follows the errors or noise too closely. This type of model will usually have poor accuracy when predicting a new sample.

To prevent overfitting, in this study we use less flexible fitting methods such as penalized regression models as explained in 4.4

4.3 Linear Regression

Linear regression is a straightforward approach for supervised learning. In particular, linear regression is a useful tool for predicting a quantitative response. Linear regression has been around for a long time and is the topic of numerous works of literature. Linear regression can represent more than one variable as input, but all the variables are linear correlated. This is in line with the Hedonic Price Theory, which asserts that a product's price can be regarded as a function of the measurable,utility-affecting attributes or characteristics of the product [15]

According to hedonic pricing theory, an Airbnb accommodation listing is a bundle

of factors that impact the overall product's quality and provide consumers with value and satisfaction. Accordingly, a listing's price can be linked to the presence or absence of specific items; it is a price proposal reflecting the host's assumptions about implicit marginal prices of particular listing characteristics.

The aim of ordinary least squares linear regression is to find the plane that minimizes the sum-of-squared errors between the observed and predicted response.

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{4.2}$$

where y_i is the outcome and $\hat{y}_i$ is the model prediction of that sample's outcome. It can be proven mathematically that the optimal plane is

$$(X^T X)^{-1} X^T y \tag{4.3}$$

where $\mathbf{X}$ is the matrix of predictors, and y is the response vector. Equation 4.3 is also known as $\hat{\beta}$ ("beta-hat") in statistical texts and is a vector that comprises the parameter estimates or coefficients for each predictor. This quantity (4.3) is easy to compute, and the coefficients are directly interpretable

Given some minimal premises about the distribution of the residuals [1], it has conclusively been shown that the parameter estimates that minimize SSE are the ones that have the least bias of all possible parameter estimates (Graybill [7]). Therefore, these estimates minimize the bias element of the bias-variance trade-off.

In linear regression, overfitting is more likely to be a serious concern [8]. When 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 prediction performance.

4.4 Penalized Regression Models

Penalized regression models shrink the size of multiple linear regression coefficients, thus introducing bias, while attempting to improve the prediction.

[1] the error terms ϵ are independent, uncorrelated and normally distributed with mean of zero and constant variance σ^2 (a.k.a. homoskedasticity)

4.4.1 Ridge Regression

When the model overfits the data or issues with collinearity, the linear regression parameter estimates may become inflated. We want to find a method to control the magnitude of these estimates to reduce the SSE. One possible solution is to use Ridge Regression (Hoerl and Kennard [9]) to control (or regularize) the parameter estimates by adding a penalty to the SSE if the estimates become large.

Ridge regression 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 $\hat{\beta}^R$ are the values that minimize:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{4.4}$$

In Equation 4.6 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 β_1,...,β_p are close to zero, and so it has the effect of shrinking penalty the estimates of β_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 λ.

4.4.2 Lasso Regression

One drawback of ridge regression that it does not shrink any of the parameter estimates towards 0. Therefore, the model does not conduct feature selection.

Least Absolute Shrinkage and Selection Operator (Lasso) [16] 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:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p} \mid \beta_j \mid = RSS + \lambda\sum_{j=1}^{p} \mid \beta_j \mid \tag{4.5}$$

Notice that the lasso and ridge regression have similar formulations. The only distinction is in the penalty term. In particular, the β_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 λ. 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 4.5 and 4.6 as solving following problems:

$$\min_{\beta} \quad \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \mid \beta_j \mid \leq s \tag{4.6}$$

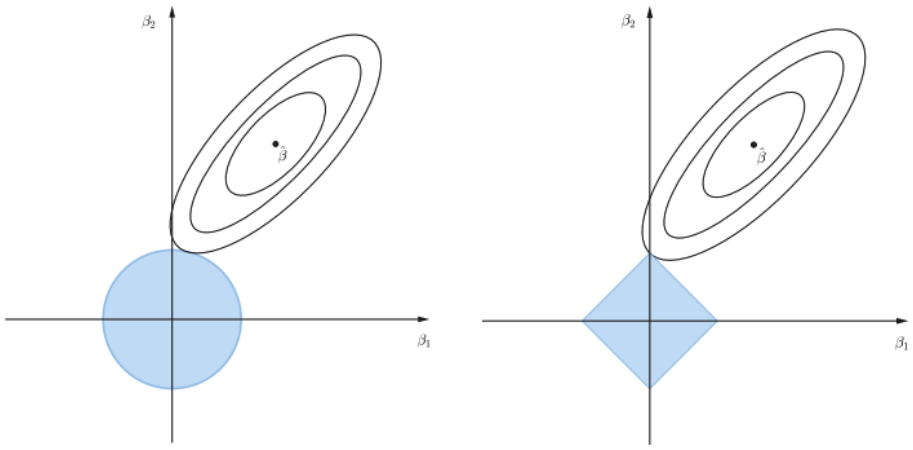
and

$$\min_{\beta} \quad \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s \tag{4.7}$$

In the figure 4.1, we illustrate why the lasso performs feature selection, while ridge regression does not.To simplify the problem, we consider only two parameters β_1 and β_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 4.5 and 4.6.

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 4.1: Illustration of two dimensional case of estimation for the lasso (right) and the ridge regression



4.4.3 Selecting the Tuning Parameter

Choosing an optimized value of tuning parameter λ is critical when implementing the ridge and the lasso regression. We do not want to choose λ too small due to the low restriction. On the other hand, when λ 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 Tibshirani [3] 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 λ, compute the average error over all folds

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

We choose a grid of λ values and calculate the cross-validation error for each value

of λ. We then select the tuning parameter value for which the cross-validation error is smallest.

4.5 XGboost

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.

Using this approach, we have to face the bias-variance tradeoff. On the one hand, we want our model to have enough degrees of freedom to resolve the underlying complexity of the data we are working with. On the other hand, we also want it to have not too many degrees of freedom to avoid high variance and be more robust.

Boosting, on the other hand, takes a more iterative approach. It is an ensemble technique in which several models are combined to achieve a better one.

4.5.1 Ensemble Methods

Imagine we want to buy a new laptop. We are unlikely to walk into a store and buy a laptop. You search the product on the internet, read reviews, compare models, prices. We ask for opinions from our family and friends. In brief, we investigate, seek the *wisdom of the crowd* before making an informed decision.
Likewise, if we combine multiple learning algorithms, we will often obtain better predictions than the best individual algorithm. We call a group of predictors an *ensemble*, and the technique is called Ensemble Learning. The Ensemble Learning algorithm is called an Ensemble method.

In practice, two ensemble models have proven to be effective on a wide range of datasets for classification and regression, both of which use decision trees as their building blocks: random forests and gradient boosted decision trees.

4.5.2 Gradient Boosting

While there are several boosting methods available, the two most popular are Adaptive Boosting [5] and Gradient Boosting [6]. In this study, we focus on Gradient boosting

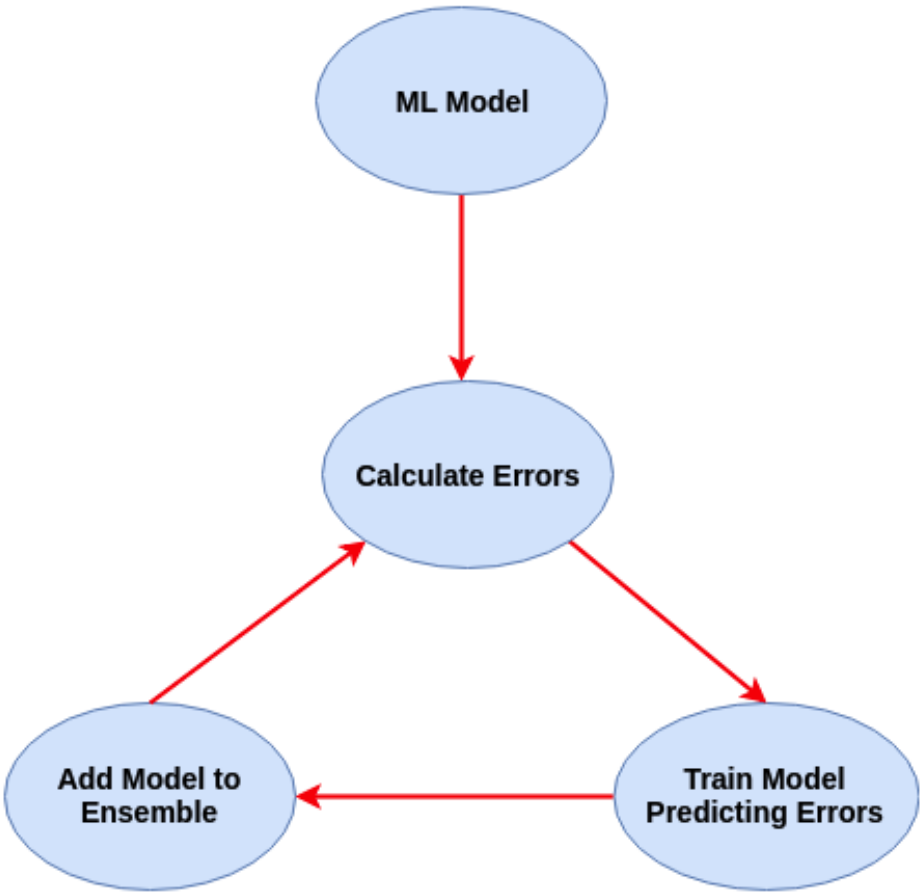
trees and its high-performance implementation, XGBoost. The Gradient boosting trees model uses a regression tree as the base learner. Kuhn, Johnson, et al. [12] suggests three reasons why trees make an excellent base learner for boosting:

1. By limiting their depth, they can be weak learners.
2. We can add together separate trees easily.
3. Trees can be created quickly.

The basic gradient boosting regression principles are: given mean squared error (a loss function) and regression trees (weak learners). The goal of Gradient Boosting Trees is to find an additive model that minimizes the loss function.

We illustrate this approach in Figure 4.2

Figure 4.2: Gradient Boosting Approach



Kuhn, Johnson, et al. [12] 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 sample
3 **for** $k \leftarrow 0$ **to** K **do**
4 Compute the residual, the difference between observed value and the *current* predicted value, for each sample
5 Fit a regression tree of depth, D, using the residuals as the response
6 Predict each sample using regression fit in the previous step
7 Update the predicted value of each sample by adding the previous iteration's predicted value to the predicted value generated in the previous step
8 **end**

4.5.3 XGBoost

Extreme Gradient Boosting (XGBoost) is a variant of the gradient tree boosting. Gradient boosting and XGBoost follows the same principle. The key differences between them lie in implementation details. Specifically, features of XGBoost which make it stand out of from the rest of other gradient boosting algorithms, are

- 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.

Since its introduction in 2014, this algorithm has been credited with winning numerous Kaggle competitions and being the driving force under the hood for several cutting-edge industry applications. Futhermore, 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 Chen and Guestrin [1] Nielsen [14].

While XGBoost is suitable for predicting, it does so at the expense of its interpretability. As shown in Figure 4.3, compared to restrictive models such as Lasso or Least Squares, Boosting is a highly flexible (complex) approach that is harder to interpret.

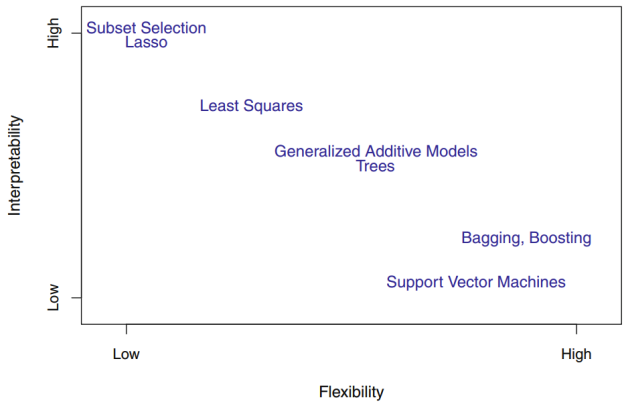


Figure 4.3: Interpretability and Flexibility Tradeoff

Source: James et al. [11]

Chapter 5

Experiments and Results

In this chapter, we discuss the application of the methods from Chapter 4 to the New York AirBnB dataset from Chapter 2

5.1 Training and Test Sets

We split the dataset into 80 listings (80%) and 20 (20%) listings for training and test sets, respectively. First, we built a model on the training data, and then we test its performance on the test set.

Typically, we are not interested in how well the method works on the training data. Instead, our goal is to assess the predictive accuracy we get when applying our method to previously unseen test data. In other words, we want to test the model's generalizability. Normally, the test set is a smaller dataset than the training set, which the model did not see previously. Moreover, The data splitting has to be unbiased. In particular, we have to randomly sample the data to guarantee that the testing and training sets are similar in terms of variation and representative.

We implement the data splitting by using the train_test_split() function from sklearn

5.2 Results

5.2.1 Best Performance Model

A summary of of results is reported in Table 5.1

Table 5.1: 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

5.2.2 Linear Regression

As expected in 4.3, incorporating such a large number of features (309) makes the linear regression model overfit the data. As shown in Table 5.1, 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

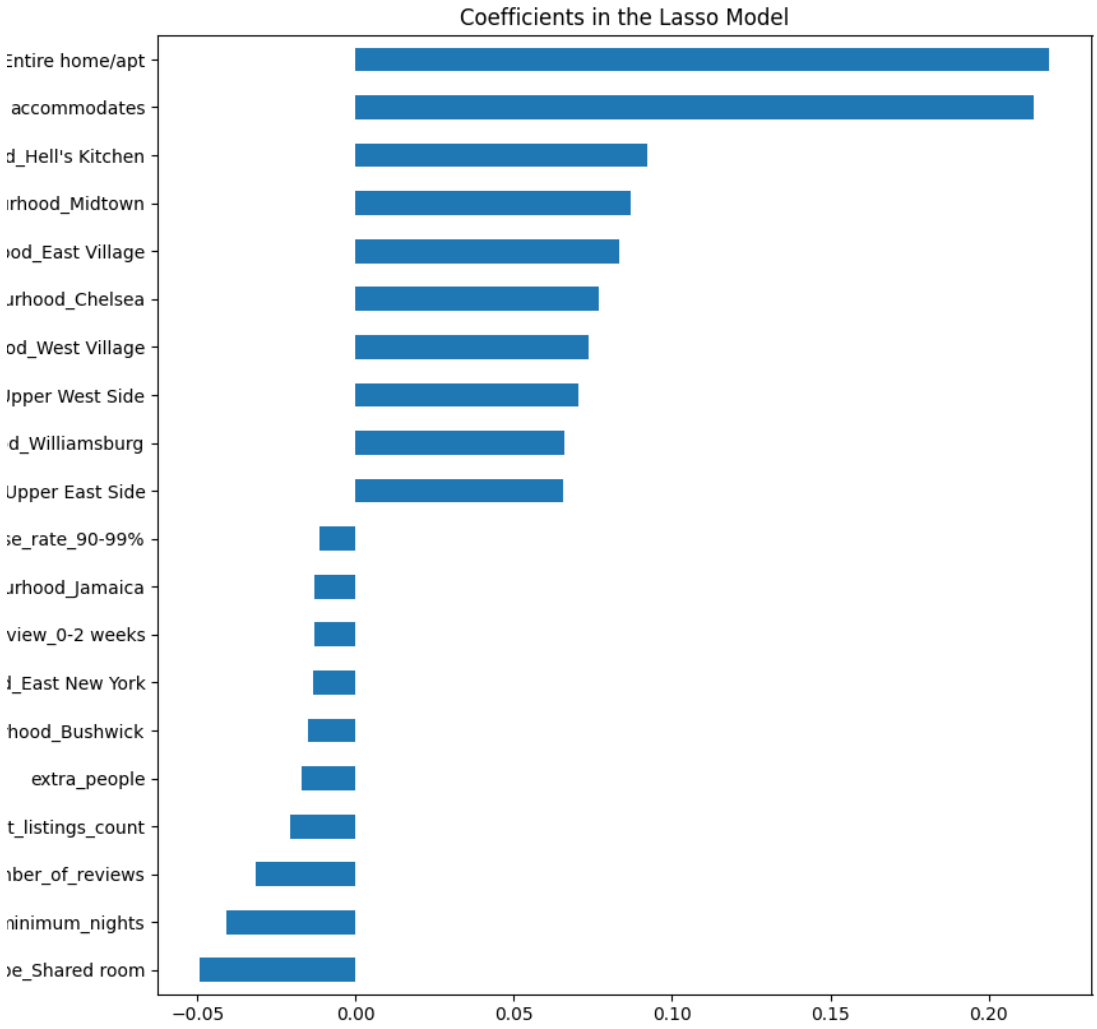
5.2.3 Ridge Regression

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

5.2.4 Lasso Regression

Using cross-validation as described in 5.2.1, we find the optimized penalty value λ 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. The figure 5.1 shows the top 20 important features.

Figure 5.1: Lasso Feature Importance



It is not surprising that the two most important positive features are whether the type of listing is the entire home and how many people the property accommodates. These are the main things a customer would use to search for properties in the first place. Features related to the location are in the top 10. Being in Hell's Kitchen, Midtown, East Village, Chelsea, West Village, Upper West Side, Williamsburg, Upper East Side, SoHo, Lower East Side neighbourhood is associated with an increase in the listing price.

5.2.5 XGboost

As shown by the Table 5.1, 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 5.2: XGBoost Top 20 Feature Weights

	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

Chapter 6

Conclusion

The study shows a comparision between different algorithms applied to predict AirBnb New York data

Appendices

A.1 Tables

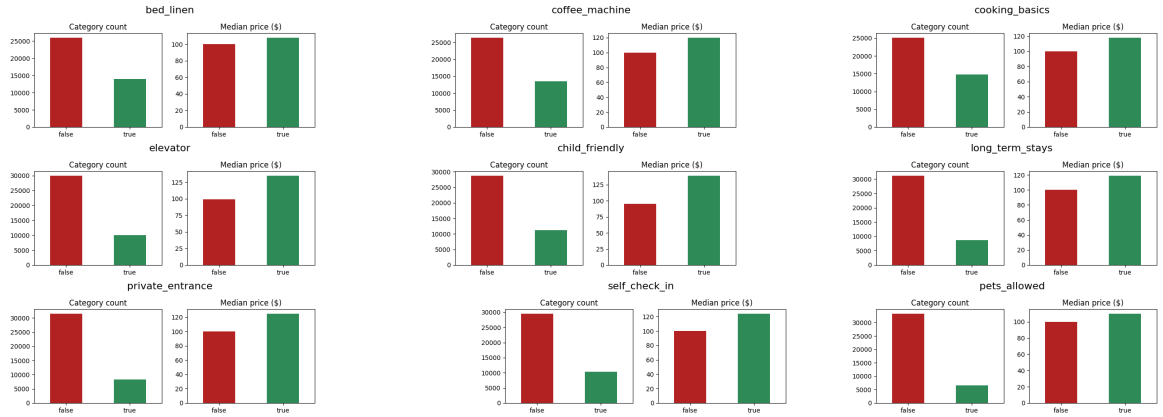


Figure A.1: Amenities Group 1

Figure A.2: Amenities Group 2

Figure A.3: Amenities Group 3

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
pets_allowed	0.165	0.371
private_entrance	0.210	0.407
self_check_in	0.260	0.438

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 be accepted)
cancellation_policy	the type of cancellation policy, e.g. strict or moderate
reviews_per_month	average number of reviews left by guest each month

Figure A.4: Histogram of Feature Distribution

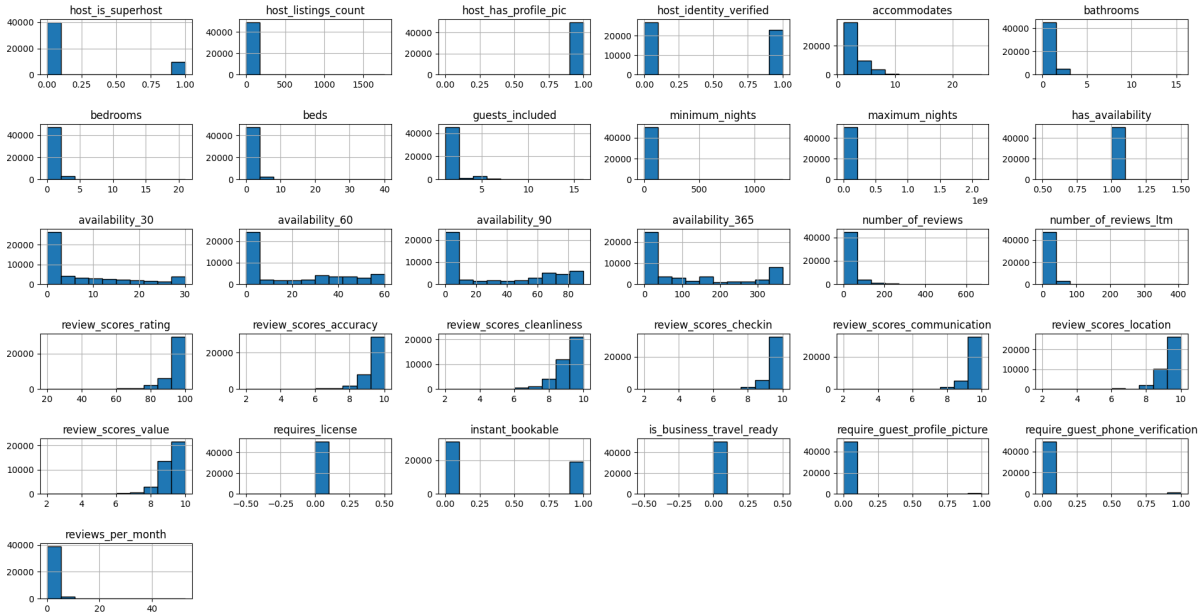


Figure A.5: Histogram of Numerical Feature Distribution Before Log Transform

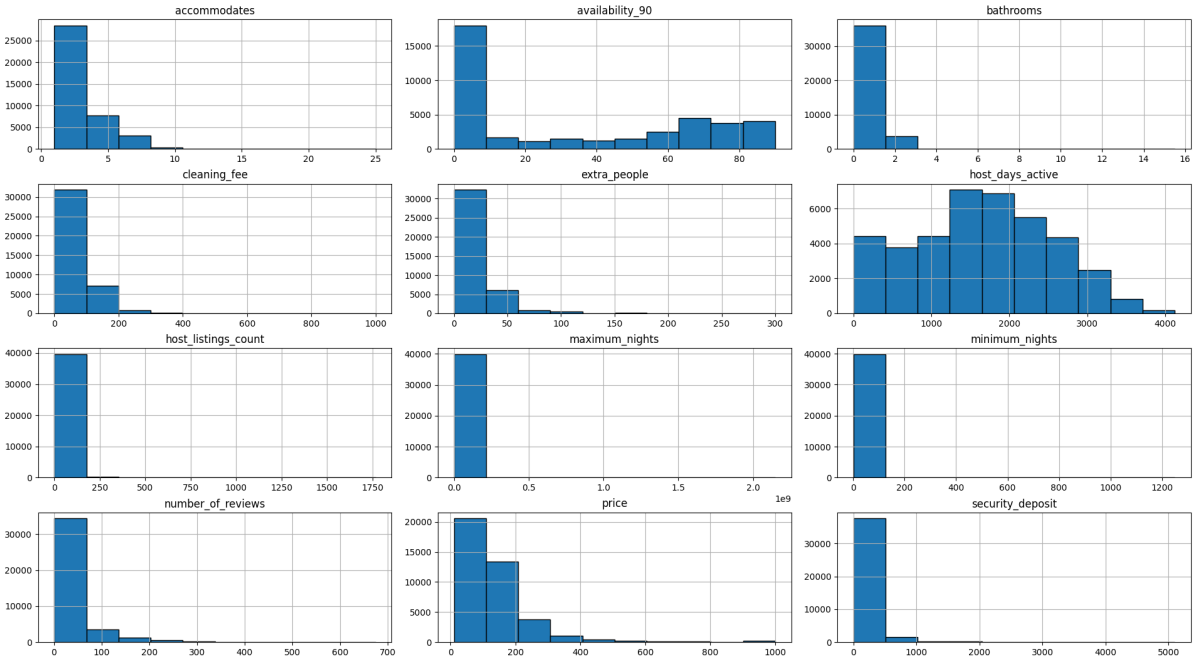
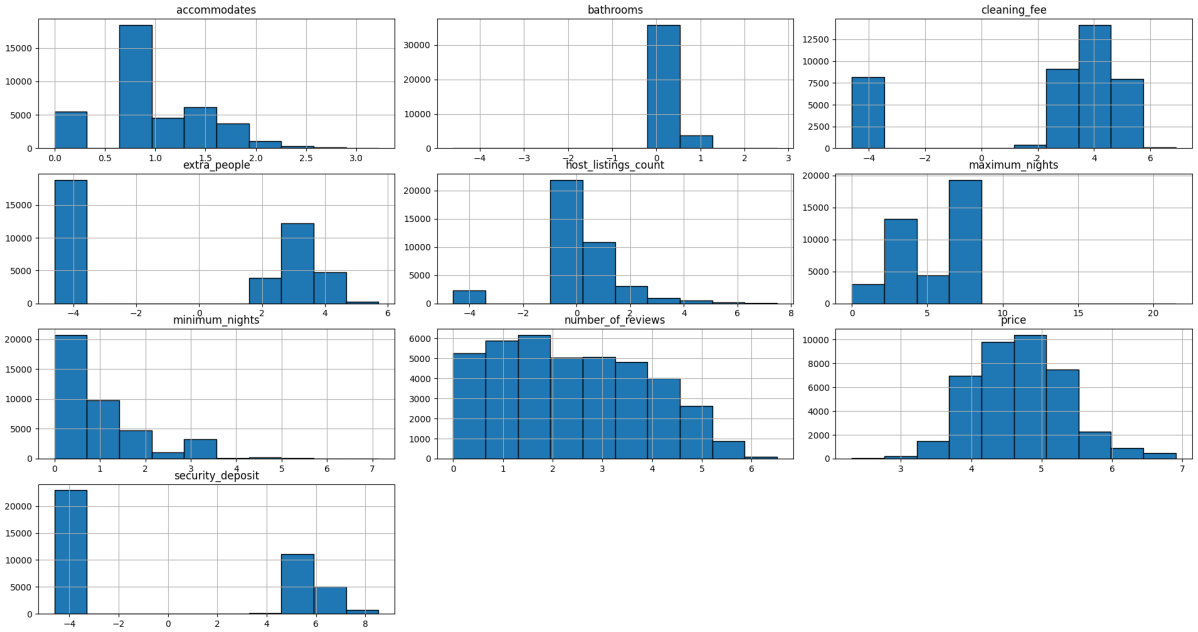


Figure A.6: Histogram of Numerical Feature Distribution After Log Transform



Bibliography

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

[2] Jacob Cohen et al. *Applied multiple regression/correlation analysis for the behavioral sciences*. Routledge, 2013.

[3] Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

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

[5] Yoav Freund and Robert E Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of computer and system sciences* 55.1 (1997), pp. 119–139.

[6] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics* (2001), pp. 1189–1232.

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

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

[9] Arthur E Hoerl and Robert W Kennard. "Ridge regression: Biased estimation for nonorthogonal problems". In: *Technometrics* 12.1 (1970), pp. 55–67.

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

[11] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[12] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*. Vol. 26. Springer, 2013.

[13] Michael H Kutner et al. *Applied linear statistical models*. Vol. 5. McGraw-Hill Irwin New York, 2005.

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

[15] Sherwin Rosen. "Hedonic prices and implicit markets: product differentiation in pure competition". In: *Journal of political economy* 82.1 (1974), pp. 34–55.

[16] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288.

[17] John W Tukey. *Exploratory data analysis*. Vol. 2. Reading, MA, 1977.

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

[19] Qiang Ye, Rob Law, and Bin Gu. "The impact of online user reviews on hotel room sales". In: *International Journal of Hospitality Management* 28.1 (2009), pp. 180–182.