Austin Airbnb Price Prediction

1. Overview

Airbnb is a home-sharing platform that allows property owners ('hosts') to put their spare room, apartment or home ('listings') online, so that guests can pay to stay in them in thousands of cities worldwide. Hosts are expected to set their own prices for their listings. Currently Airbnb and other sites provide some general guidance, but with this project we are trying to improve our base price services which help hosts price their properties using a wide range of data points.

1.1 Problem Statement:

Each year, Austin plays host to a large number of events such as ACL Festival and SXSW. We would expect that properties near downtown have higher asking prices based on the convenience of location to some of these events. Airbnb pricing is important to get right, particularly during these events where there is lots of competition and even small changes in prices can make a big difference. Price too high and no one will book. Price too low and you'll be missing out on a lot of potential income.

1.2 The client and why do they care about this problem:

The client in this case would be Airbnb and they care about this problem because by suggesting the right rental price for each property, they can help their hosts to be profitable while at the same time helping their guests to get fair rental prices.

Instead of breaking down statistics by ZIP code, we want to generalize the sections of Austin for ease of discussion and analysis so we decided to break the city into five regions: Central (downtown and nearby areas), East, West, North, and South.

This project aims to predict the base price for properties in Austin by using machine learning.

2. Data

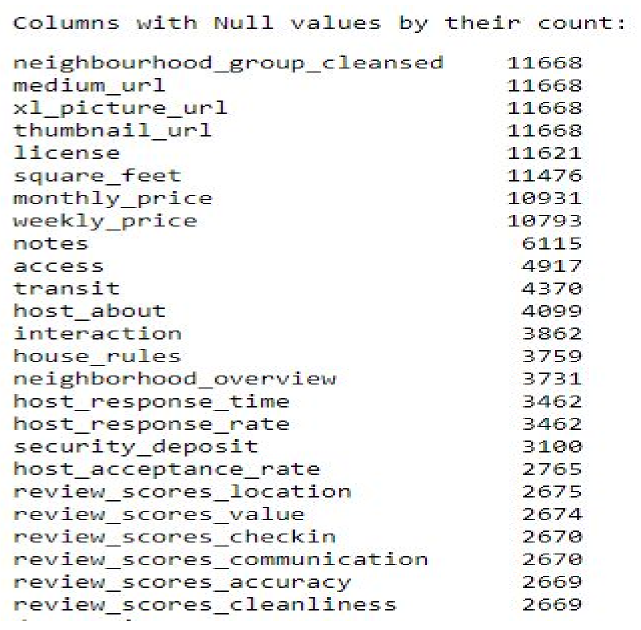
Conveniently, insideairbnb has collected rental data for properties all over the world, “sourced from publicly available information from the Airbnb site.” Before we started our investigation of the data, our expectations were that the size of the property, location, and review ratings should be more correlated to the rental price than other features.

2.1 Dataset Description:

The dataset that we are using in this project is listings.csv file which was compiled on March 17th, 2020 and was downloaded from http://insideairbnb.com/get-the-data.html for the city of Austin. The dataset contains 11,668 rows and 106 columns including the id column being used as the index.

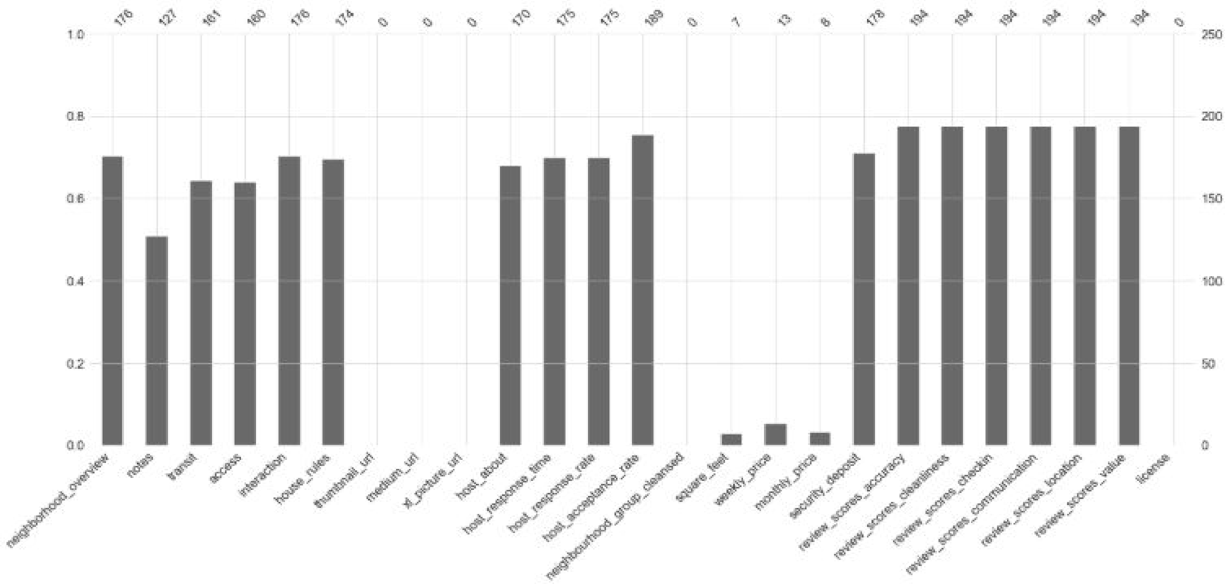
2.2 Data Cleaning and Visualization:

First, we look at the top 25 columns with missing values to see if they add any value to our model or if we can drop them.



```
Columns with Null values by their count:

neighbourhood_group_cleansed    11668
medium_url                      11668
xl_picture_url                  11668
thumbnail_url                   11668
license                         11621
square_feet                     11476
monthly_price                   10931
weekly_price                    10793
notes                            6115
access                           4917
transit                          4370
host_about                       4099
interaction                      3862
house_rules                      3759
neighborhood_overview            3731
host_response_time               3462
host_response_rate               3462
security_deposit                 3100
host_acceptance_rate             2765
review_scores_location           2675
review_scores_value              2674
review_scores_checkin            2670
review_scores_communication      2670
review_scores_accuracy           2669
review_scores_cleanliness        2669
```



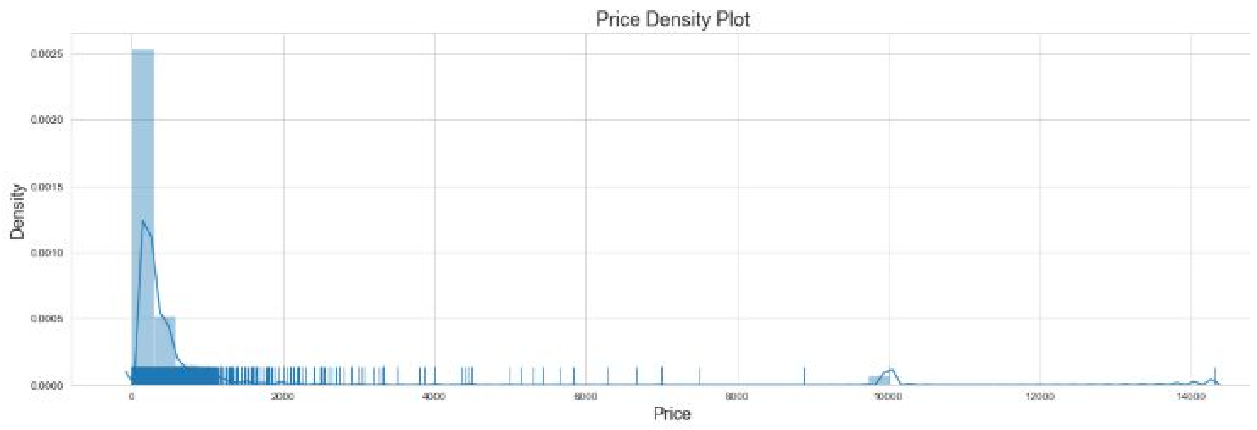
Features like 'square_feet', 'monthly_price' or 'weekly_price' have a lot of NULL values but we can use bedroom, bathroom, and accommodates data as a replacement for space size.

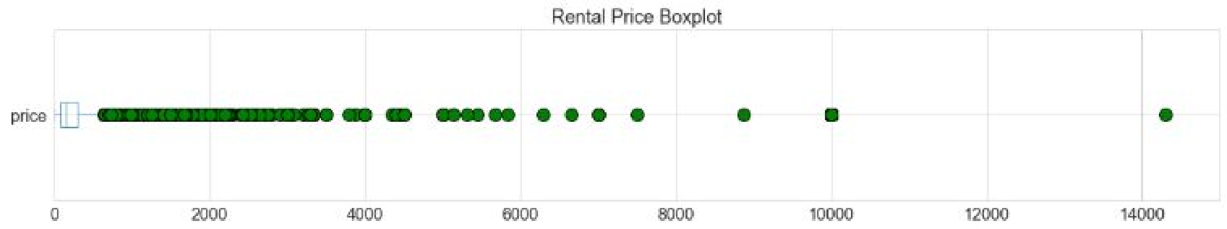
id	price	monthly_price	weekly_price
2265	$225.00	NaN	NaN
5245	$100.00	NaN	NaN
5456	$95.00	NaN	NaN
5769	$40.00	NaN	$160.00
6413	$99.00	$1,900.00	$700.00

Also use 'price' for our prediction instead of 'monthly_price' or 'weekly_price'. We can drop all of these columns since we have other columns that we can use as replacements.

After checking the descriptive statistics on the dataframe, we noticed that our target variable ‘price’ as well as 'cleaning_fee', 'security_deposit' and 'extra_people' costs are not included as part of the report. We did more investigation and noticed that they were an object data type. We then chained .str.replace().astype() methods to remove the ‘$’ and ‘,’ characters from these columns and also changed the data type to float.

Next, we decided to add a column called ‘Total_price’ which includes the sum of 'price', 'cleaning_fee', 'security_deposit' and 'extra_people' costs. After changing the data type, we looked at a few different plots like boxplot and distplot on the ‘price’ column and noticed that its distribution is not normal.



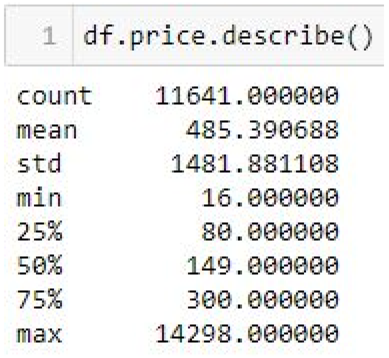


As we can see on the figures above, there are some outliers above $8,000 that we need to look at. There also seems to be a few prices below $15 that we need to check but the good news is that there are no null values in the ‘price’ column.

We start looking at properties where the ‘price’ is set to 0 and we notice that these are just a few incorrect values and we fill them by getting the median price based on the ‘accommodates’ rate for each of these properties.

We go up to the next level price which is set to $1 and we notice that all of these properties are for Hurricane Harvey Refugees and they haven't been updated since 2017 so we can drop them.

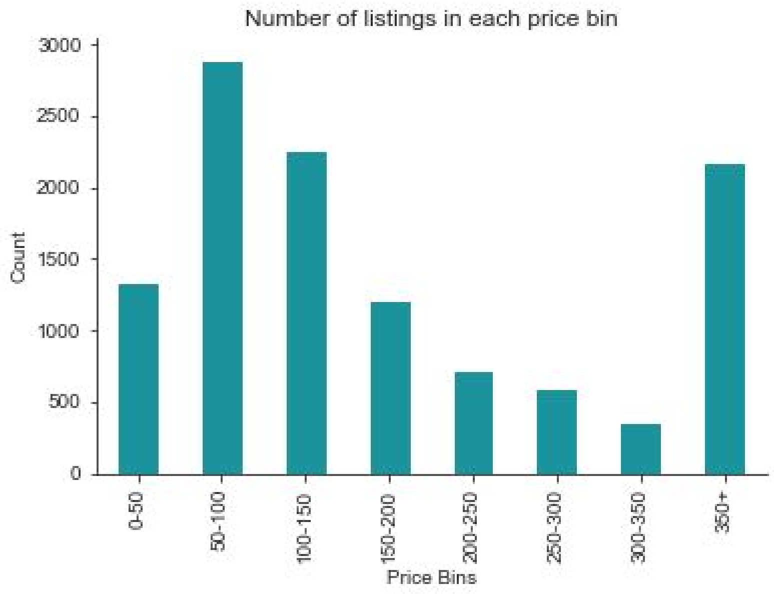
The next few levels for prices up to $16 can be dropped as these are camp site rentals and they do not add any value to solving our problem.



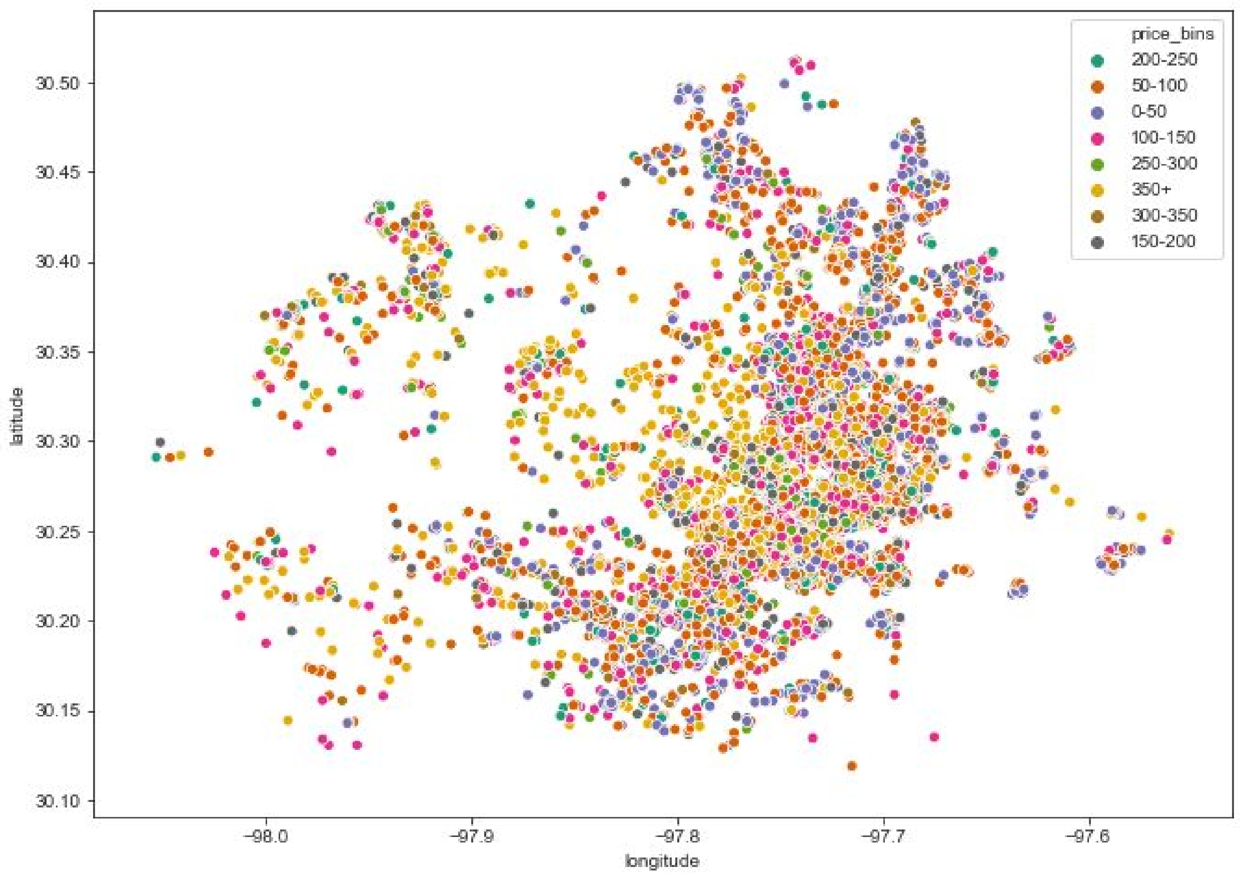
```
df.price.describe()
```

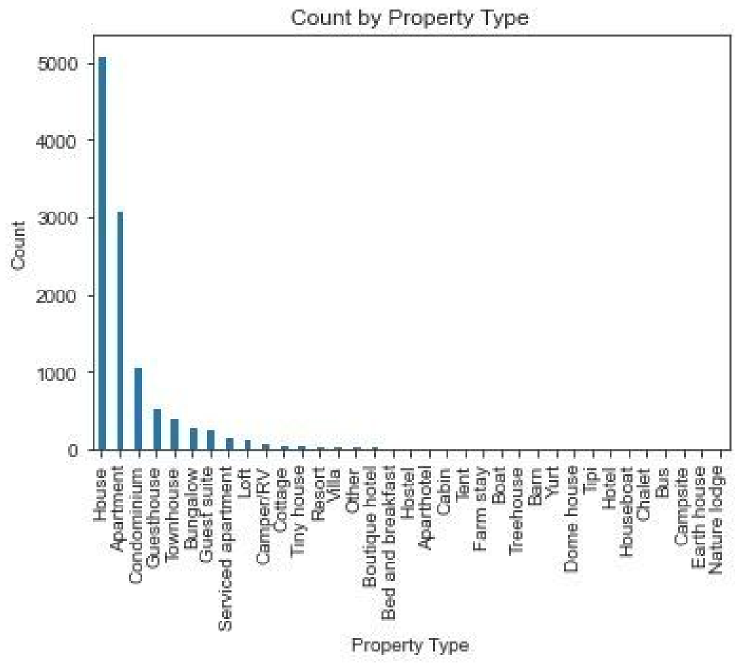
```
count    11641.000000
mean       485.390688
std       1481.881108
min         16.000000
25%         80.000000
50%        149.000000
75%        300.000000
max      14298.000000
```

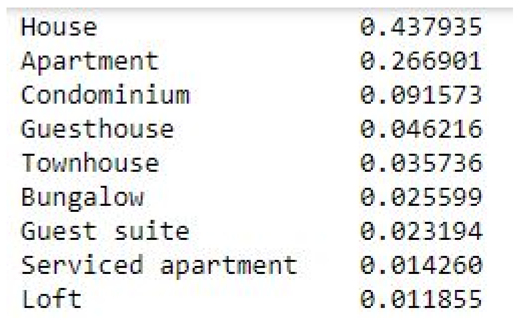
We ran descriptive statistics on the price column and now the minimum is set to $16 and the maximum is $14,298!



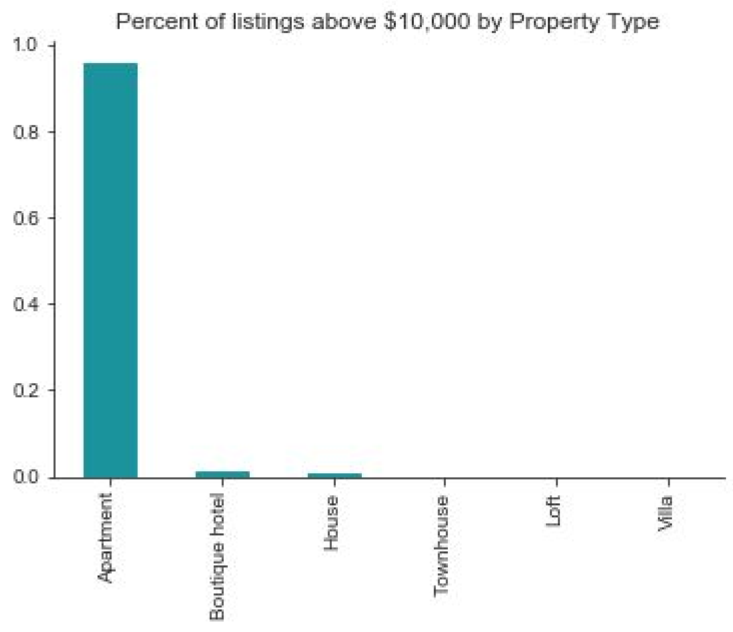
In order to do some more visualization on the ‘price’ column, we break it into 8 bins with each bin incrementing by $50. Then we use this new ‘price_bins’ column as the value for hue in the scatterplot and by doing this, we can see that most of the properties above the 350 price range are located on the West side of Austin.





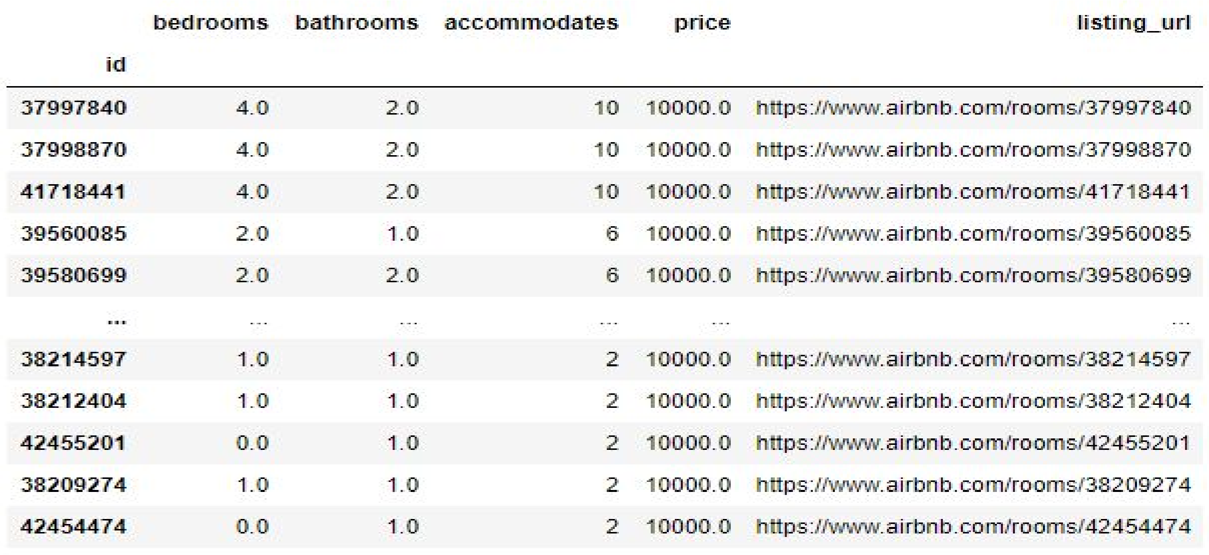

```
House                 0.437935
Apartment             0.266901
Condominium           0.091573
Guesthouse            0.046216
Townhouse             0.035736
Bungalow              0.025599
Guest suite           0.023194
Serviced apartment    0.014260
Loft                  0.011855
```

Grouping properties by every unique type shows us that 44% of the properties are Houses, 27% are Apartments, 9% are Condominiums, 5% are Guesthouses and 3% areTownhouse. The rest of the property types hold lower than 3% counts and we can group them together later on to help our model with only keeping the top 90% values and the rest be set to ‘Other’.



```
Apartment         0.960159
Boutique hotel    0.015936
House             0.011952
Townhouse         0.003984
Loft              0.003984
Villa             0.003984
```

We look at the unique property types above $10,000 per night to see if we can find which property types the outliers are related to. By looking at the chart above, we see that 96% of these prices are for Apartment property type. We need to look at random Apartment listings using the listing_url to confirm if the prices are actually above $10,000 for these listings.

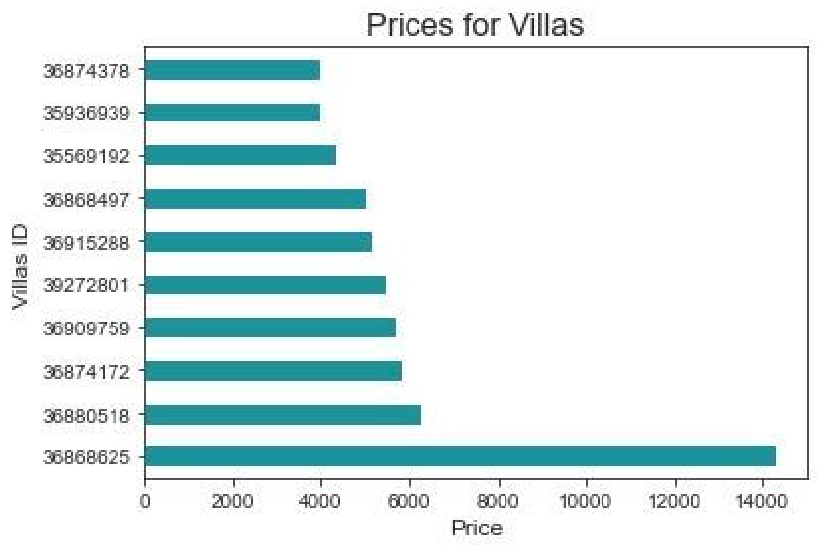


	bedrooms	bathrooms	accommodates	price	listing_url
id					
37997840	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/37997840
37998870	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/37998870
41718441	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/41718441
39560085	2.0	1.0	6	10000.0	https://www.airbnb.com/rooms/39560085
39580699	2.0	2.0	6	10000.0	https://www.airbnb.com/rooms/39580699
...	...	...	...	...	...
38214597	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38214597
38212404	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38212404
42455201	0.0	1.0	2	10000.0	https://www.airbnb.com/rooms/42455201
38209274	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38209274
42454474	0.0	1.0	2	10000.0	https://www.airbnb.com/rooms/42454474

Next, we check the price of apartments that are more than $10,000 per night. We manually checked some of these listings randomly and only 1 of them has a price above $10,000/night.

We will replace these Apartment prices with the median price associated with the accommodates rate for each apartment.

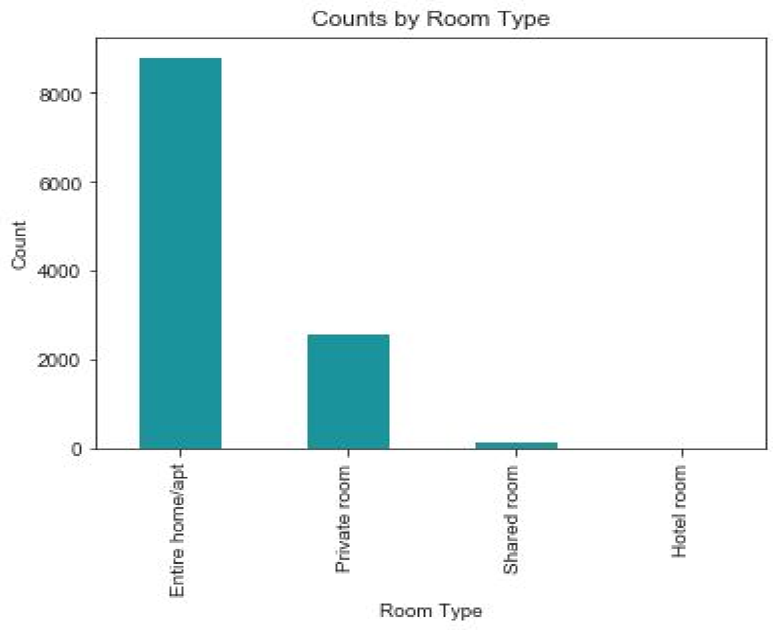
The max price is $14,298 and we couldn't find this price in Apartments. The next property type that we should check is Villa. These properties are larger houses, having an estate and consisting of farm and residential buildings arranged around a courtyard and it would make sense if they have higher rental prices.



```
id
36868625    14298.0
36880518     6291.0
36874172     5832.0
36909759     5669.0
39272801     5443.0
```

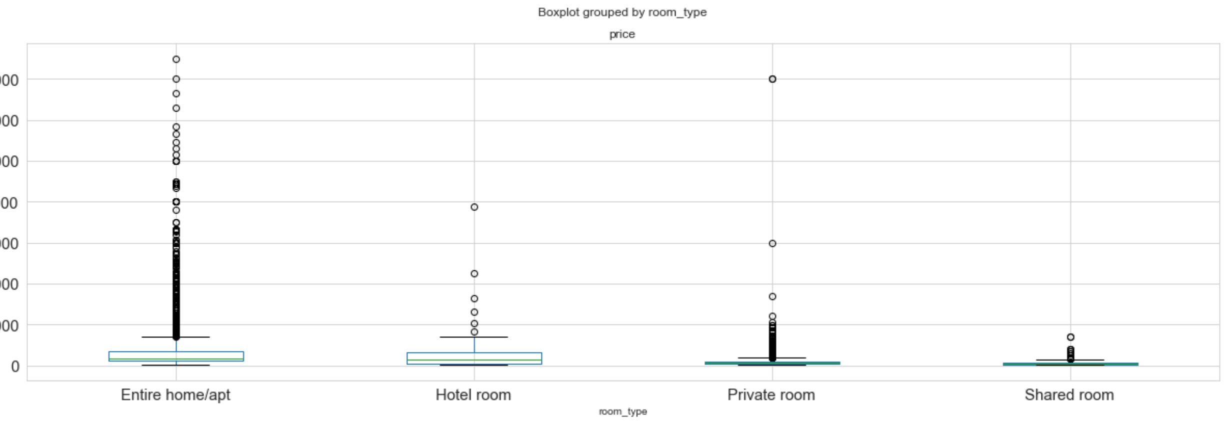
We see that our $14,298 per night property is listed under Villa property type. The median price for Villas is $1,344 and we could use that value to replace these outliers but since Villas are not in the top 90% property types, they will be included in the 'Other' property type at the end.

Using a mask for all the properties with prices above $8,000, prices are set to NaN values and then filled with the median price for each property type.

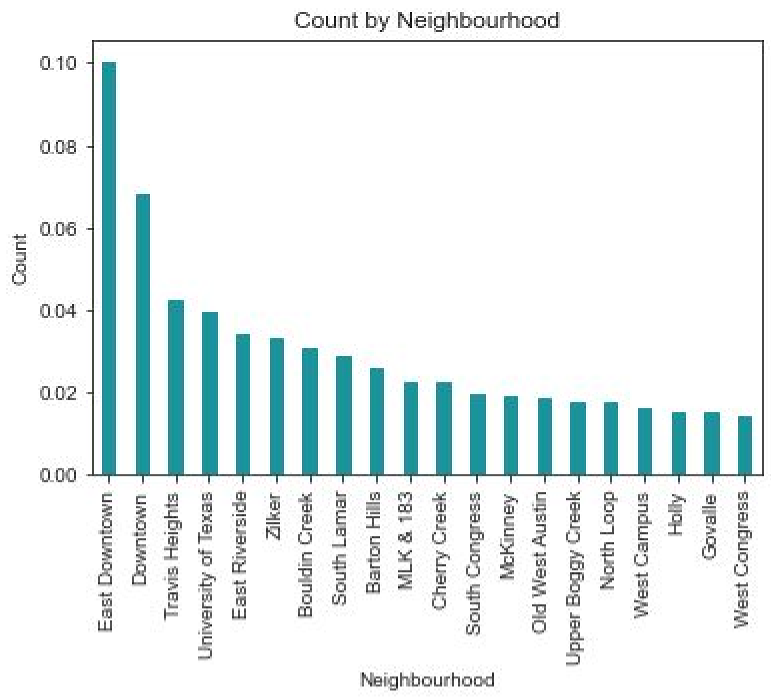


```
Entire home/apt    0.758698
Private room       0.223349
Shared room        0.013573
Hotel room         0.004381
```

Looking at the room types, we see that 76% of them are either the entire home or the entire apartment.

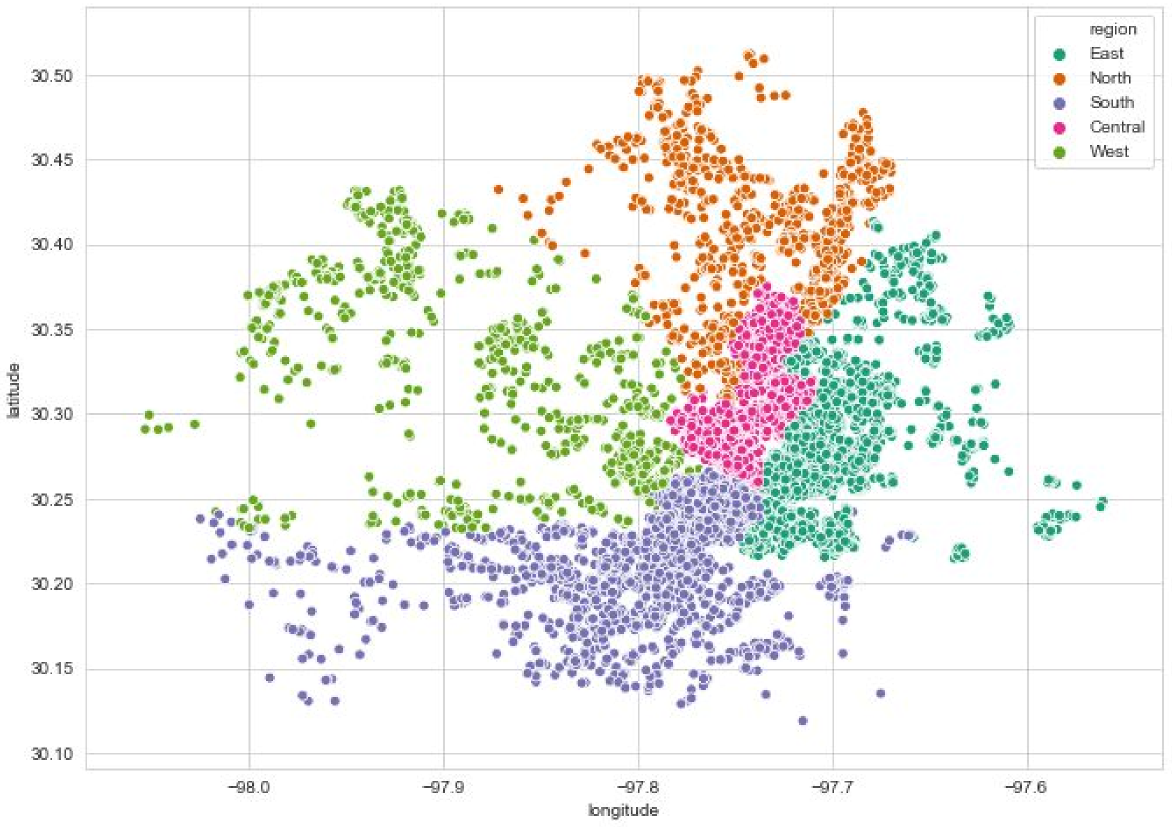


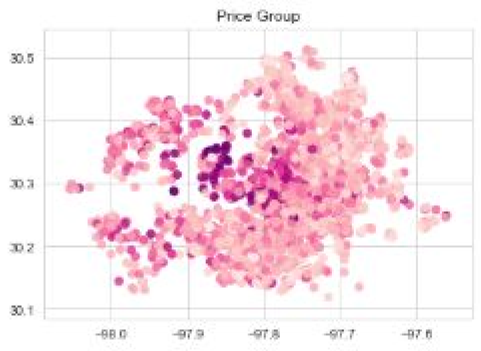
Using boxplot above, shows us that most of the higher rental prices are for Entire home/apt but there are a few high rental prices for Hotel room and Private room types that needs to be checked.

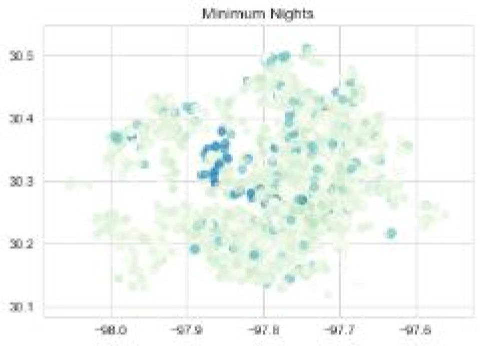


Looking at different neighbourhoods, we can see that 10% of the properties are located in the East Downtown neighbourhood, 7% in Downtown, 4% in Travis Heights and University of Texas each.

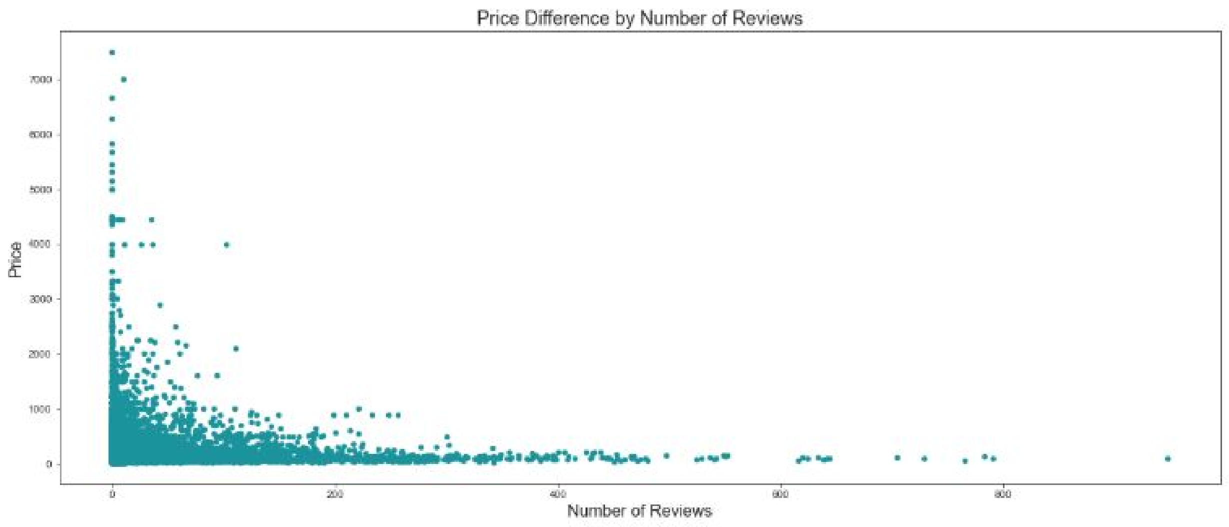
Next, we divide all the zip codes into four different regions. This will help us with some visualizations and analysis.



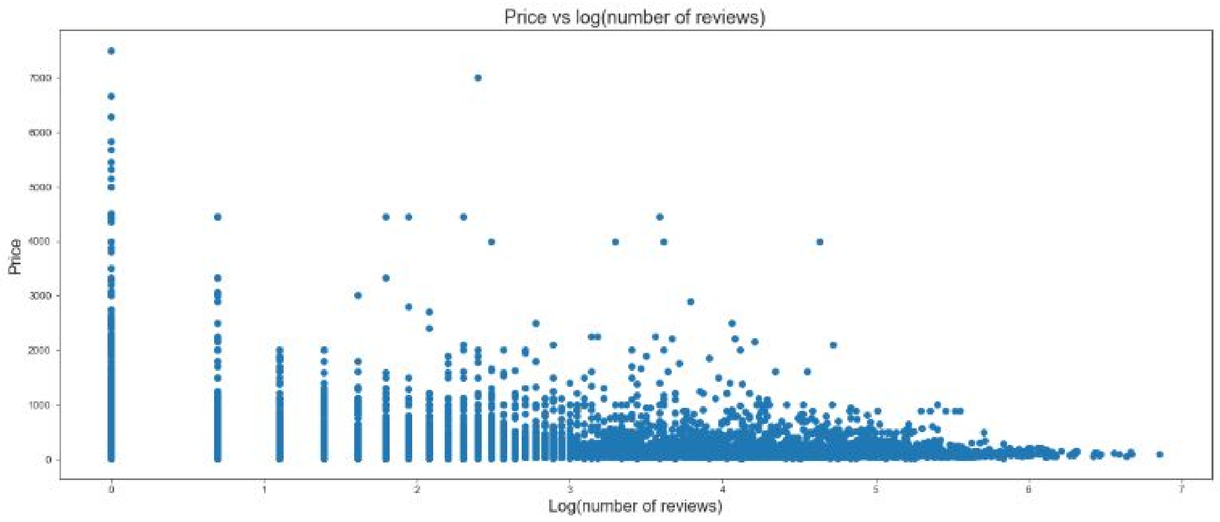




Looking at the scatter plots above, we can see that the most expensive rentals are in the West region. This makes sense because this region has the most count of Villas and properties are bigger as well. They also have the highest minimum nights in their listings.

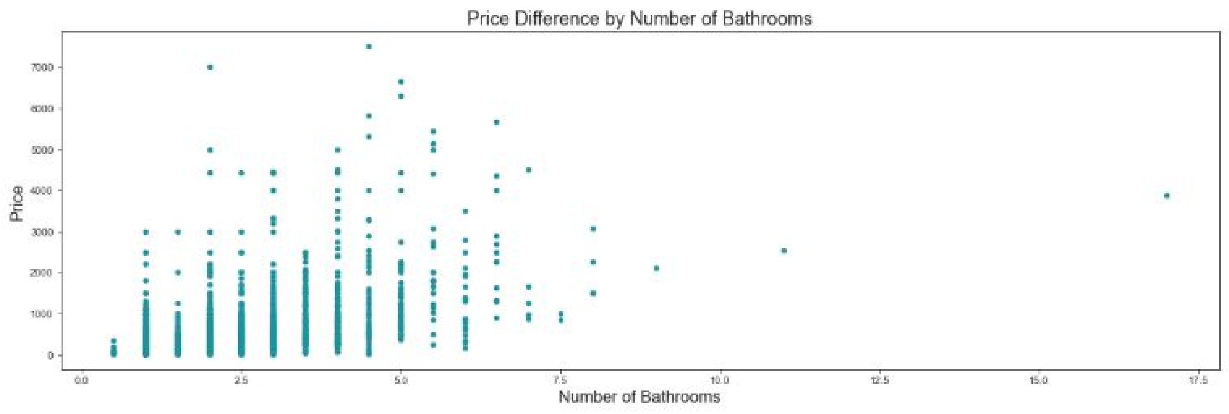


Looking at the number of reviews, we can see that some high rental prices, either don't have a review or the number of reviews are small. These properties might be good candidates for dropping later.

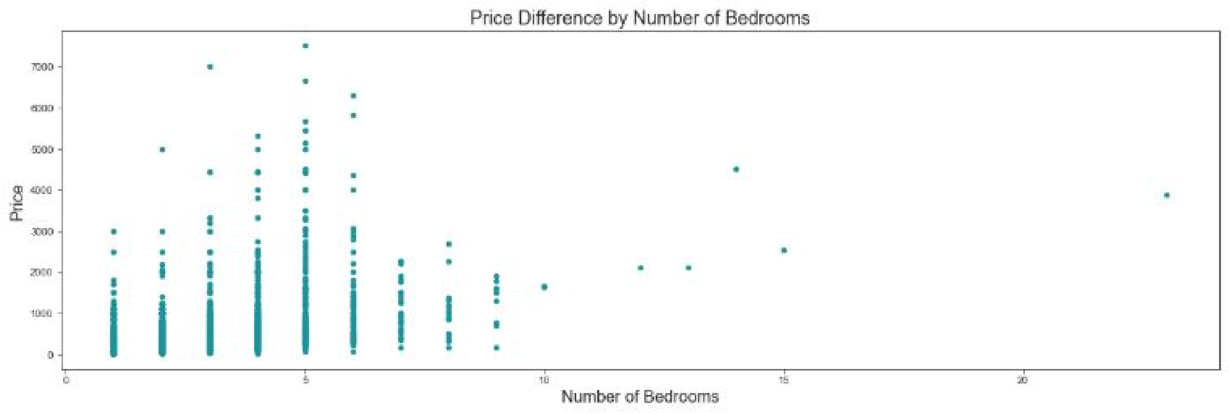


Looking at the log of the number of reviews, we can see in more detail that properties with zero reviews have high prices ranging above $7,000!

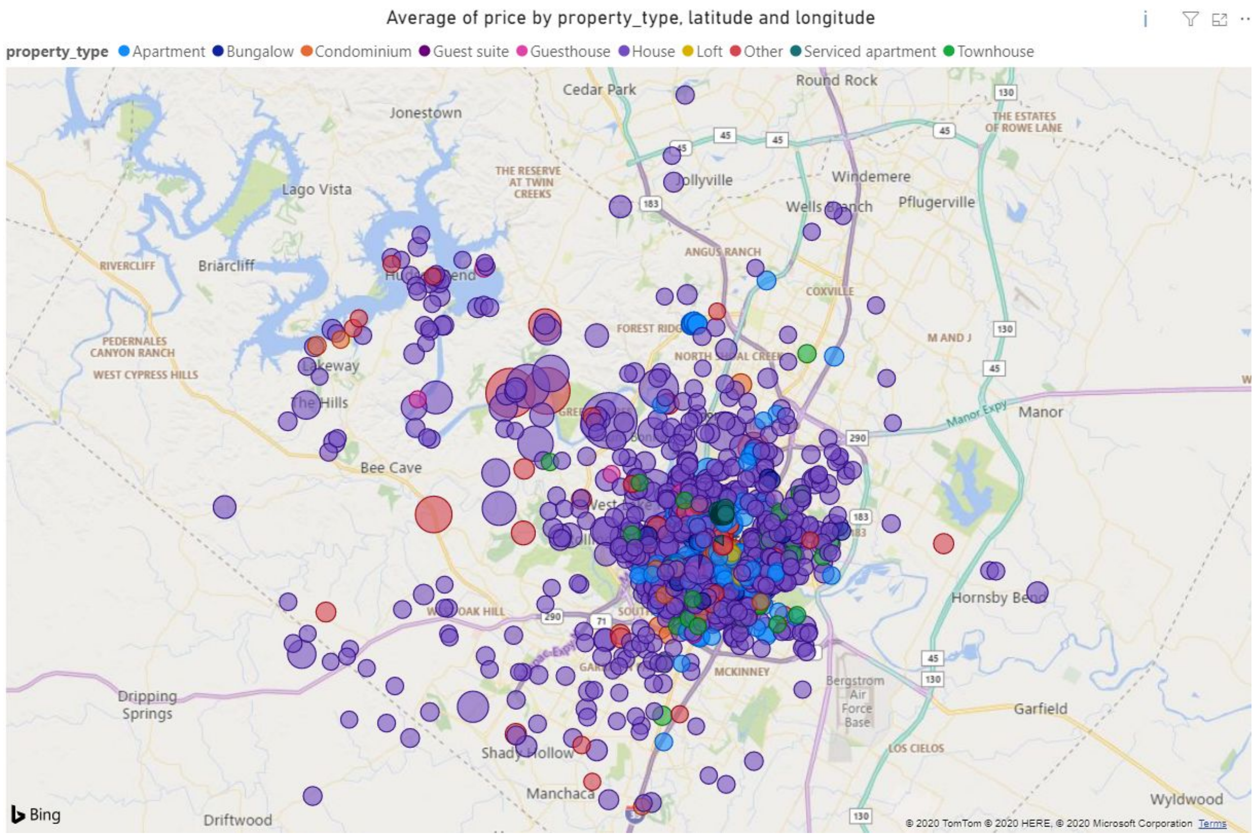
We look at the bathrooms, bedrooms and accommodates columns next. The bathrooms column has 13 missing values as well as 33 values set to zero. The bedrooms column has 15 missing values and 706 values set to 0. We set these 0 values to NaN values and then use the accommodates column to fill them with the median value.



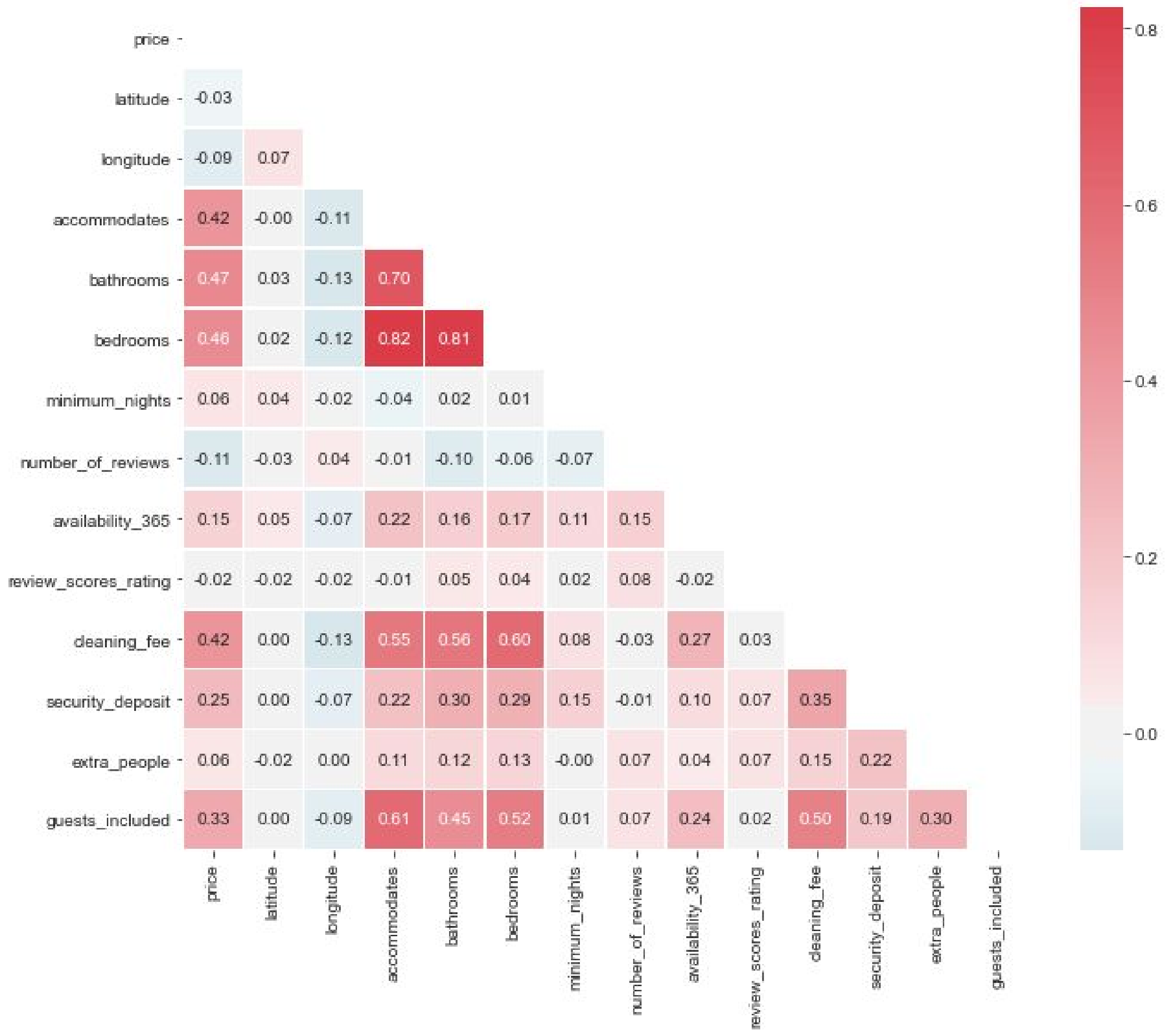
Looking at the chart above, properties with 4 bathrooms or more have higher rental prices!



Looking at the chart above, properties with 4 or more bedrooms have higher rental prices!


Average of price by property_type, latitude and longitude
property_type ● Apartment ● Bungalow ● Condominium ● Guest suite ● Guesthouse ● House ● Loft ● Other ● Serviced apartment ● Townhouse
Cedar Park
Round Rock
Jonestown
Lago Vista
Jollyville
Windemere
Wells Branch
Pflugerville
Briarcliff
Lakeway
The Hills
Bee Cave
Manor
Hornsby Bend
Dripping Springs
Shady Hollow
Manchaca
Garfield
Driftwood
Wyldwood
Bing
© 2020 TomTom © 2020 HERE, © 2020 Microsoft Corporation Terms

Looking at the map above, we can see that most of the expensive rentals are on the west side region. This region has a lot of Villa property types and it makes sense to be expensive.



The correlation matrix above shows that price is correlated with accommodates, bathrooms, bedrooms, cleaning_fee, security_deposit and guests_included. Bedrooms are also correlated with accommodates and bathrooms.

After replacing all the NaN values and dropping some observations, we now have 11,608 observations and 25 features. We save the file to our data folder so it can be used for our Machine Learning steps.

3. Modeling

Now that the data acquired is wrangled properly, we need to divide our dataset into training and test sets and then begin the part where the models will be trained. Since our goal is to predict the price, which is a continuous variable, we have used Regression models for this project.

A list called features_list was used to hold the column names that we would like to use to predict the target variable. Then, df.price.quantile(.95) was also used to subset the data frame to only keep the top 95% of prices due to outliers. The target variable has a few outliers left and this should get rid of them. The features_list was then used to subset the columns that we want to keep on the new data frame.

We have also used the 'One Hot encoding' technique where we have changed the categorical columns to numeric columns, since Machine Learning algorithms can only predict on numeric data. To accomplish this, we used Pandas .get_dummies() function. After transforming our categorical columns, we add StandardScaler() function to scale and transform the numerical columns. Many machine learning algorithms work better when features are on a relatively similar scale and close to normally distributed. Scale generally means to change the range of the values. The shape of the distribution doesn't change. The range is often set at -1 to 1.

In this problem, we decide to scale the quantitative features using StandardScaler. For each value in a feature, StandardScaler standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation. StandardScaler does not meet the strict definition of scale I introduced earlier. StandardScaler results in a distribution with a standard deviation equal to 1. The variance is equal to 1 also, because variance = standard deviation squared. And 1 squared = 1. StandardScaler makes the mean of the distribution 0. About 68% of the values will lie between -1 and 1.

There are a couple of models which we have used for the prediction of the price. We have used Linear Regression as the baseline model to compare with the other models to see how it performs. The other models are:

- Decision Tree Regressor
- Random Forest Regressor which consists of an ensemble of Decision Trees. It uses a method called 'Bagging' for sampling the data and aggregates all the Decision trees into a good performing model. The Random Forest Model without any hyperparameter tuning was performing better than the other models.
- K Neighbors Regressor
- RMSE: It measures the average error performed by the model in predicting the outcome for an observation.
- R^2 score: This corresponds to the squared correlation between the observed outcome values and the predicted values by the model. The higher the R-squared, the better the model.

The Training and Testing data have been split into 70% and 30% respectively using Machine Learning ‘sci-kit learn’ library. After splitting the data, we used our first model which was the Decision Tree Regressor. We set a few hyperparameters for this model but later on we will use the GridSearchCV to find the best_params_ for this model.

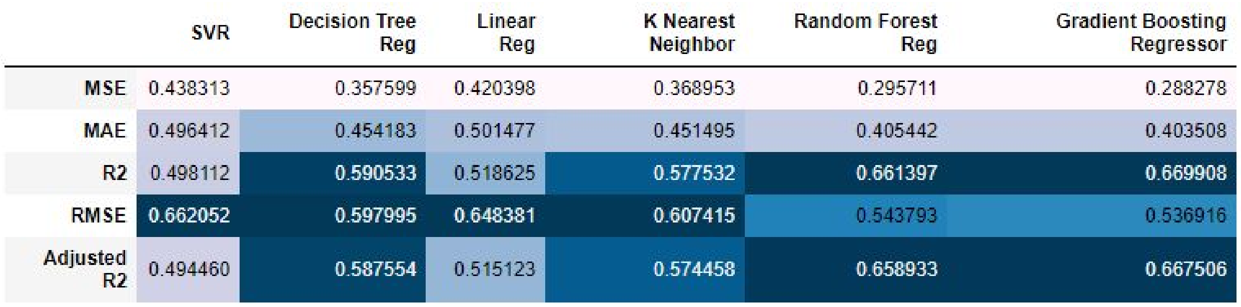
3.1 Model Performance:

The R^2 score for our original Decision Tree model was 0.26. The coefficient R^2 is defined as (1 - u/v), where u is the residual sum of squares ((y_true - y_pred) ** 2).sum() and v is the total sum of squares ((y_true - y_true.mean()) ** 2).sum(). The best possible score is 1.0. Hyperparameter tuning is very important for the performance of the model which we use to train it. After hyperparameter tuning the Decision Tree model we get a R^2 score of 0.59.

Next, we use our Random Forest Regressor model.

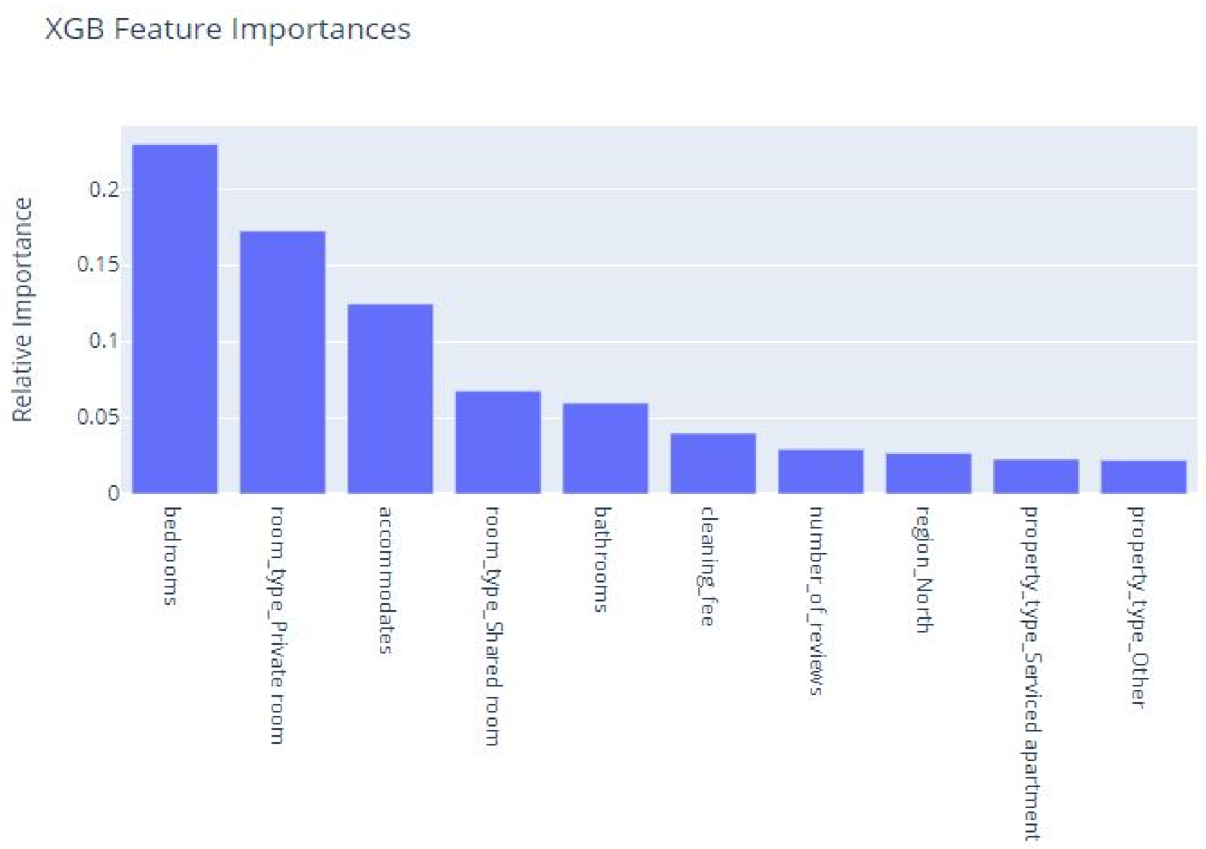
- First, we have implemented a ‘GridSearch Cross Validation’ to select the best hyper-parameter for the Random Forest Regressor.
- Then those parameters were used for the model to train and predict the price. We get an R^2 score of 0.66.

The Random Forest Regressor has been the best model so far but we decided to use the XGBRegressor model as well. After hyperparameter tuning the XGBRegressor model, we get a R^2 score of 0.67.



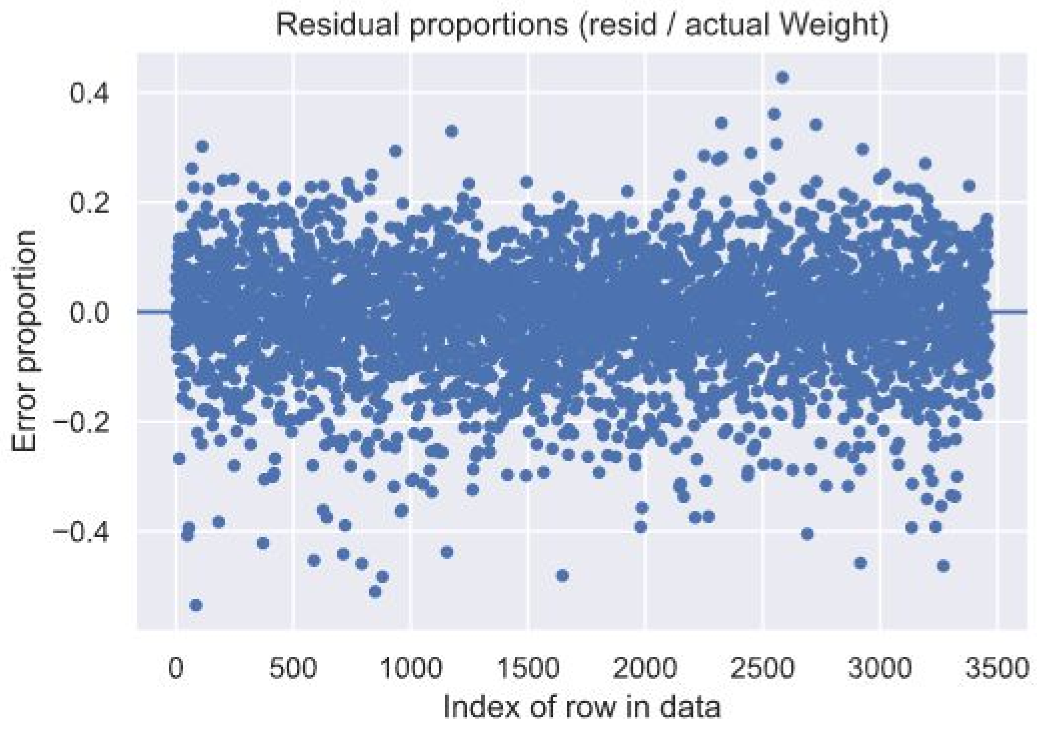
	SVR	Decision Tree Reg	Linear Reg	K Nearest Neighbor	Random Forest Reg	Gradient Boosting Regressor
MSE	0.438313	0.357599	0.420398	0.368953	0.295711	0.288278
MAE	0.496412	0.454183	0.501477	0.451495	0.405442	0.403508
R2	0.498112	0.590533	0.518625	0.577532	0.661397	0.669908
RMSE	0.662052	0.597995	0.648381	0.607415	0.543793	0.536916
Adjusted R2	0.494460	0.587554	0.515123	0.574458	0.658933	0.667506

From the XGBRegressor model, we can have the important features which are crucial for determining the price. Below you can see these features based on their importance level.



Feature Importances from XGBRegressor model shows that the bedrooms, room_type_Private room, accommodates, room_type_Shared room and bathrooms variables appeared to have the highest levels of predictive power. The regions variables that we defined to group the observations into four areas appeared to be of little or even no value to the XGBRegressor model. The security deposit variable has some predictive power in the Random Forest model but no predictive power in the XGBRegressor model.

The overall R^2 score of the six models ranged from a low of approximately .50 (SVR) to a high of .67 (XGBRegressor). In order to have a better understanding of the error in our selected model, we decided to plot the actual prices and the predictions to see how close or far off these values are from each other. Looking at the chart below, we can see that the residual proportions indicate that our predictions are mostly within 20-25% of the correct value.



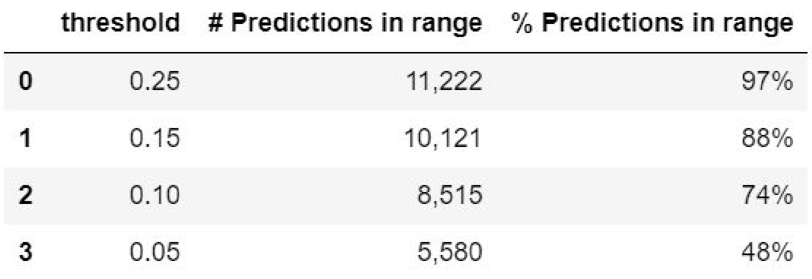
The XGBRegressor model scored highest and it is recommended for use by Airbnb to predict the correct listing prices so that both the hosts and the guests are getting fair deals.

4. Conclusions

Based on the Austin, TX rental data from insideairbnb, property size matters the most for rental pricing, followed by review ratings and location. Price prediction with reasonable accuracy is possible using standard regression techniques, but subjective aesthetics such interior design and marketing quality make it challenging to predict rental price with high accuracy.

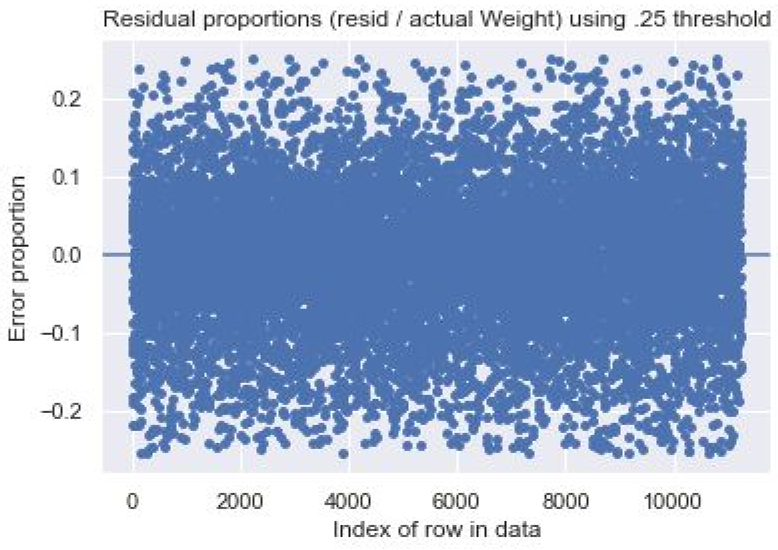
4.1 A Case Study:

Let's assume that we would like to define a threshold (a bucket) that we can use to evaluate our model predictions and see how close or far off the predicted rental price are. The table below shows the prediction results using different thresholds.



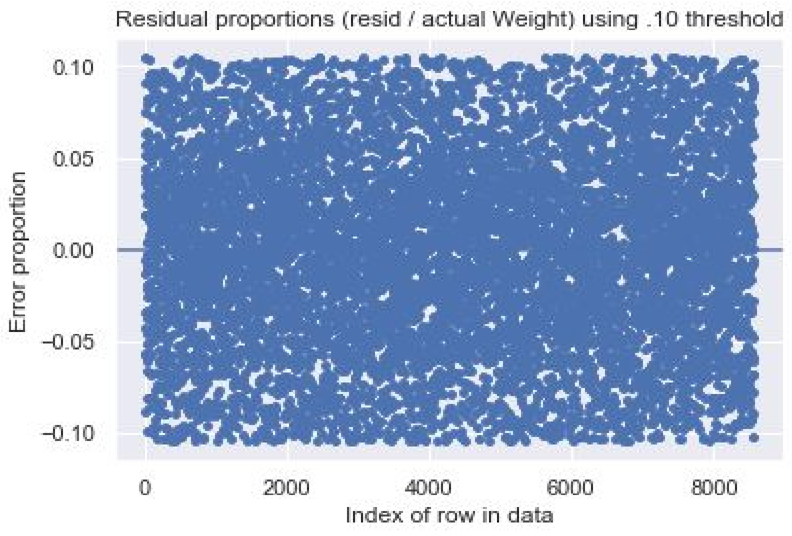
	threshold	# Predictions in range	% Predictions in range
0	0.25	11,222	97%
1	0.15	10,121	88%
2	0.10	8,515	74%
3	0.05	5,580	48%

The first threshold that we try is 0.25 which returns predictions that are in range of 0.25 above or below the actual price. This would return 11,222 acceptable predictions out of 11,541 (97%) total number of listings in that range.



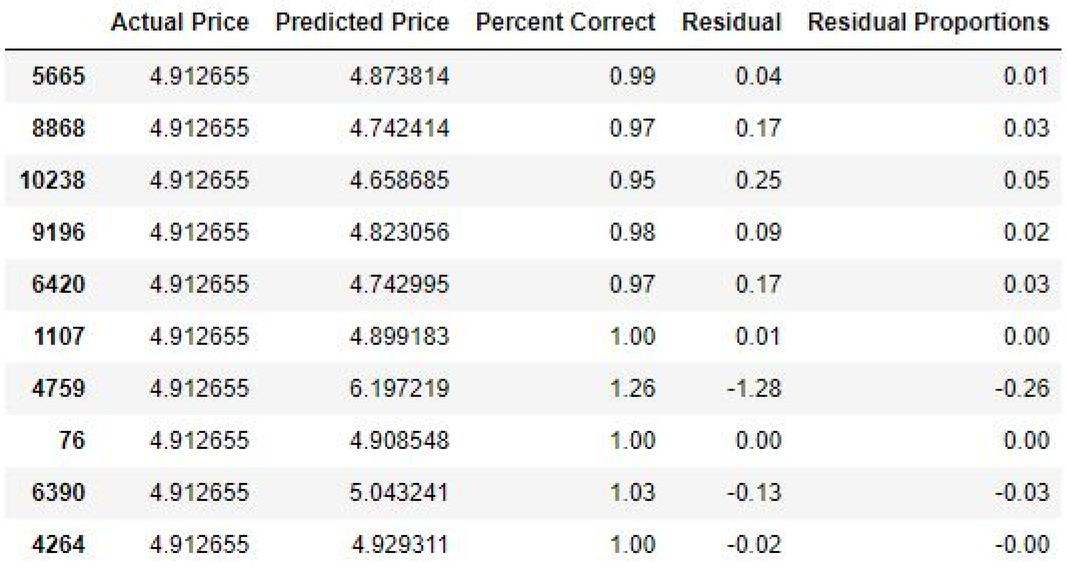
This plot shows the Error proportion in data using the 0.25 threshold

We can lower this threshold depending on how close we would like our predictions to be to the actual prices but this will also lower the number of correctly predicted prices within the accepted range. For example, by lowering the threshold to 0.1, we would only get 8,515 acceptable predictions out of 11,541 (74%) and using 0.05 for threshold, we only get 5,580 rental prices out of 11,541 which means 48% predicted are in range.



This plot shows the Error proportion in data using the 0.1 threshold

Ultimately, Airbnb decides which threshold to use. Looking at the price column statistics, we see that the median is around $136 and if we use this value to filter our predictions dataframe shown below, almost all of the predictions have a Residual Proportions value below .05 threshold. This means that our model can predict almost all of the rental prices around the median!



	Actual Price	Predicted Price	Percent Correct	Residual	Residual Proportions
5665	4.912655	4.873814	0.99	0.04	0.01
8868	4.912655	4.742414	0.97	0.17	0.03
10238	4.912655	4.658685	0.95	0.25	0.05
9196	4.912655	4.823056	0.98	0.09	0.02
6420	4.912655	4.742995	0.97	0.17	0.03
1107	4.912655	4.899183	1.00	0.01	0.00
4759	4.912655	6.197219	1.26	-1.28	-0.26
76	4.912655	4.908548	1.00	0.00	0.00
6390	4.912655	5.043241	1.03	-0.13	-0.03
4264	4.912655	4.929311	1.00	-0.02	-0.00

4.2 Future Work:

- Find a way to incorporate image quality into the model, e.g. by using the output of a convolutional neural network to assess image quality as an input into the pricing model.
- Use better quality/more accurate data which includes the actual average prices paid per night.

- Include a wider geographic area, e.g. other major cities in Texas.
- Augment the model with natural language processing (NLP) of listing descriptions and/or reviews, e.g. for sentiment analysis or looking for keywords.
- In addition to predicting base prices, a sequence model could be created to calculate daily rates using data on seasonality and occupancy, which would allow the creation of actual pricing software.
- Tailor the model more specifically to new listings in order to help hosts set prices for new properties, by removing features that would not be known at the time — e.g. other fees, availability and reviews.