Capstone 1 Milestone Report - AirBnB Price Prediction

Alexandra Michel

--Problem Statement--

AirBnb is a website that allows people to search and reserve short-term and apartment rentals, as well as become a host on the platform by opening up their own space to potential guests. All prices are set by the hosts, and guests/renters can narrow their search by different criteria to find a place to stay within their budget while still meeting their needs. As a host looking to earn supplemental income through AirBnB, it is important to know what the market price for your space is so that you can meet the demand of more consumers and ultimately have more guests book a stay with you.

There are third-party companies (Like this one, and this one) out there who provide AirBnB pricing for a fee; probably implementing similar techniques on data that is free to access like the data set I will use for this project. AirBnB also provides hosts with some advice when it comes to pricing, but who knows what attributes they are using to determine those prices, or how successful they are to attract guests (as hosts don't have to go with the suggested price, this would be hard to test!). AirBnB could be interested in this kind of exploration as a way to generate more revenue through their hosts and enhance their prediction models. The average consumer with a spare room or living space is the ultimate consumer of the information this model would provide, as it would guide them to set a list price for that space. Within the real estate industry, this would have a big impact for real estate agents and their clients (homebuyers and investors) looking to use a measure like this as part of the bigger picture in describing and analyzing the potential return on investment of a home purchase.

--Description of the dataset, how I obtained, cleaned, and wrangled it--

I obtained the data set from InsideAirBnB.com and loaded the .csv file for the Los Angeles, CA listings into a Pandas dataframe. Each row in the dataframe is an available listing on the AirBnB site in Los Angeles, CA, and it has features related to listing criteria (bedrooms, bathrooms, beds, number of guests), as well as reviews, location, amenities, min/max number of nights. The dataframe had many free-text features which I dropped for this initial attempt, but I would include them in the future as I learn to implement NLP techniques. I started out by also dropping a handful of columns which have mostly NaN values. There were also columns in the dataframe

that were closely related to each other (having to do with the total listings by that same host) which I dropped and just left the `host_listings_count` column to account for that information. After these manipulations, I was already down from 106 columns to 60, reducing the dimension significantly.

There were also sparse categories within `property_type` which I grouped together into `House`, `Apartment`, and `Other`. I was going to create a group called `Hotel` to include `Hotel`, `Boutique Hotel`, and `Bed and Breakfast`, but there would have only been 468 listings out of the total ~$38,000 listings so I decided it wasn't significant enough to keep.

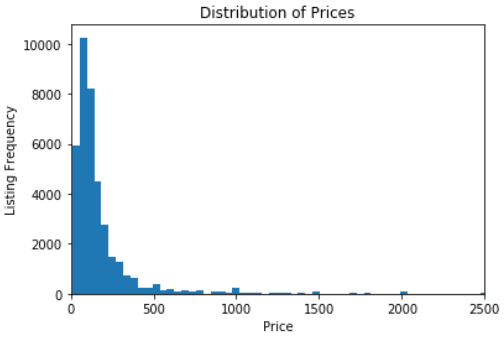
The dollar-value columns `price`, `security_deposit`, `cleaning_fee` and the column `extra_people` needed to be changed from `string` to `int` values, stripping the dollar signs where needed and converting the `NaN` values to zeros.

Once I had changed the `price` column to an `int`, I began exploring the basic range of price values in the data. I came across a few listings with $0 for the nightly price which I thought was odd, and I also found that the highest nightly price was $22,000. I found the specific listings, and by using the listing URL was able to validate that, in fact, there are two luxury retreat listings at $22,000/night. However, the $0/night listings were mistakes. I know in general I would really just drop those entries, but because there were only 5 of them, I verified the prices on the website and manually inputted them into the dataframe.

--Initial findings from exploratory analysis--

Question: What is the distribution of prices like?

Keeping in mind that I am working to predict the nightly price of an AirBnB listing, I decided to take a look at what the distribution of prices looked like to get an idea of where the majority of listings fell, and to see if we had any major outliers. It turns out, the prices range all the way up to $22,000/night, but the amount of listings above $2,500/night is less than 1% of the entire dataset. I figured it would be best to inspect the distribution for listings below that price to give a better sense of the majority of prices.



From this graph we can see that the majority (94.2%) of the listings fall under $500/night. When I think about an average hotel night or airbnb night stay, this price makes sense to me (an average middle class consumer). I also calculated that the median price is $110.0/night while the mean price is $226.88/night, which is in line with what we see visually on the graph. Because we have a handful of higher priced listings, they are skewing the data so the mean comes out to be higher than most listings. There are still many more listings priced in the $0-$250/night range, which keeps the median price lower.

Question: How have prices changed over time?

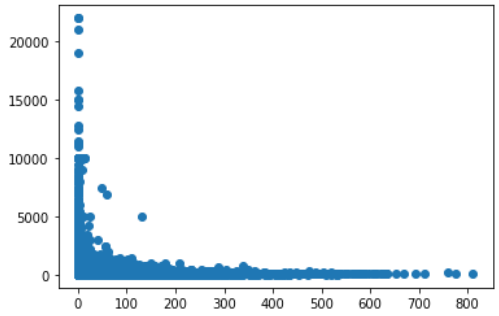
Next I thought it would be interesting to take the time data to see how these listing prices have changed since 2009. As we see in the next figure, the median price has only slightly increased over time, because again, the overwhelming amount of listings at lower prices overpower the fact that the maximum price has increased drastically over time.

Because the maximum prices and amount of listings at those higher price points have increased substantially from when this data started being compiled, it has caused the mean to increase from $93.11 per night to $180.77 per night, almost doubling in just 11 years of data. This begs the question: When did AirBnB start introducing the "Luxury Retreat" listings to the site?

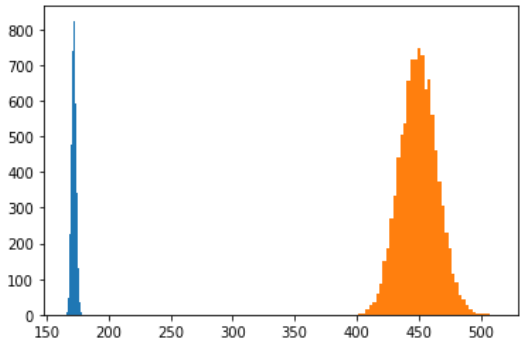
```
Mean nightly price of listings in each year on Airbnb in LA:
first_review
2009-01-01     93.11
2010-01-01    118.73
2011-01-01    122.68
2012-01-01    176.51
2013-01-01    176.62
2014-01-01    161.75
2015-01-01    164.66
2016-01-01    161.53
2017-01-01    163.79
2018-01-01    169.95
2019-01-01    183.05
2020-01-01    180.77
```

When I first looked at this, I also wondered if including the luxury listings or excluding them would lead to different results once I got to modeling this data. It seemed like these higher prices could skew the dataset, but I didn't want to get rid of them prematurely.

Question: What is the distribution of reviews like? Does the absence of reviews affect price significantly?



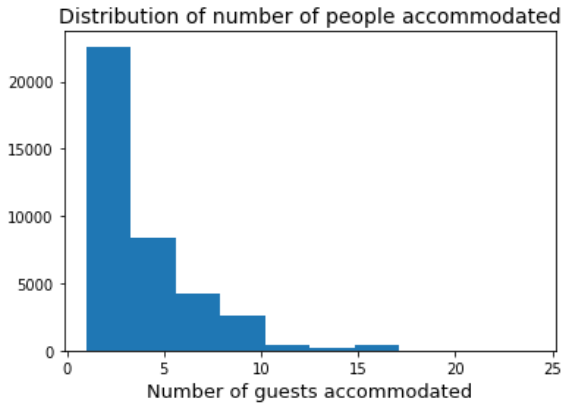
By separating the data into listings with/without reviews, it looked like listings with no reviews had a higher sample mean in general. We can also infer that a listing with no reviews is either a newer listing or a very expensive listing that not many guests have booked because of price. I decided to see if this is significant using bootstrapping:

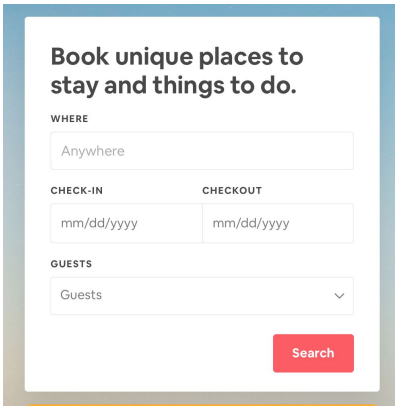


- 98% confidence interval for the sample mean for listings with reviews: [167.81, 176.41]
- 98% confidence interval for the sample mean for listings without reviews [416.57, 485.10]
- This tells us there is a 98% confidence that a listing with no reviews will be listed for $240 more

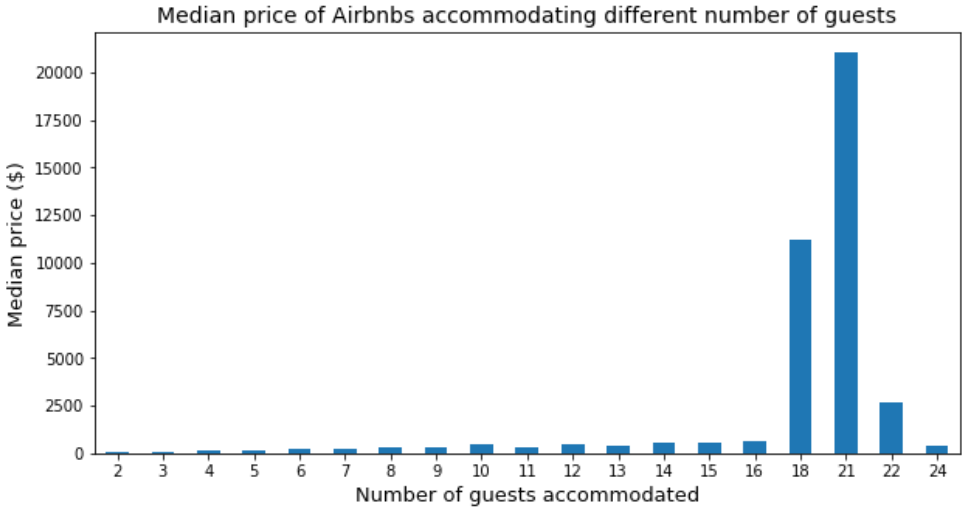
Question: What is the distribution of the number of people accommodated in an LA listing, and how does this affect price?

Before answering this question, I thought about what the user experience is when you go to airbnb.com to search for a listing. What information are you asked for first in order to view listings?

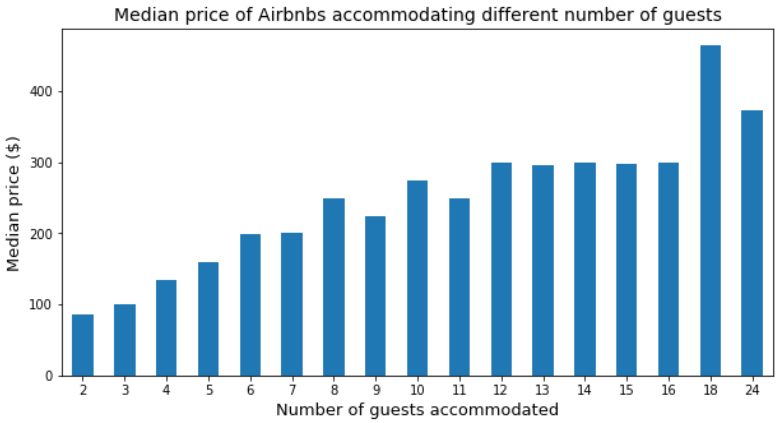




Aside from when and where your stay will take place, you immediately enter the number of guests you are looking to accommodate in the airbnb stay. To me, this indicates that this feature 'accommodates' will be a pretty important feature in our model. Looking at the distribution of this feature, we see that most listings accommodate between 1-5 guests.
I then plotted the median price of listings with different accommodations, but quickly found that again the listings with higher prices were not allowing me to see what was really happening here...

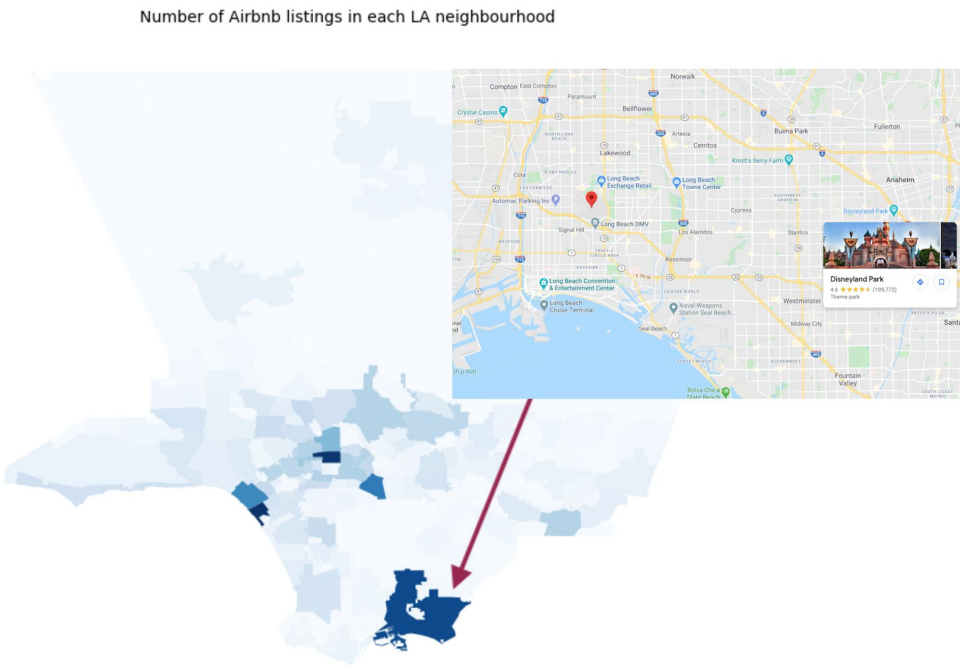


So I removed all listings above $500/night from the plot in order to see that we do have an overall upward trend in price as the number of guests the listing accommodates increases. This is what we would expect, right? If you are looking to book a stay by yourself, you expect it to be less expensive then when you are looking to stay somewhere in a group.



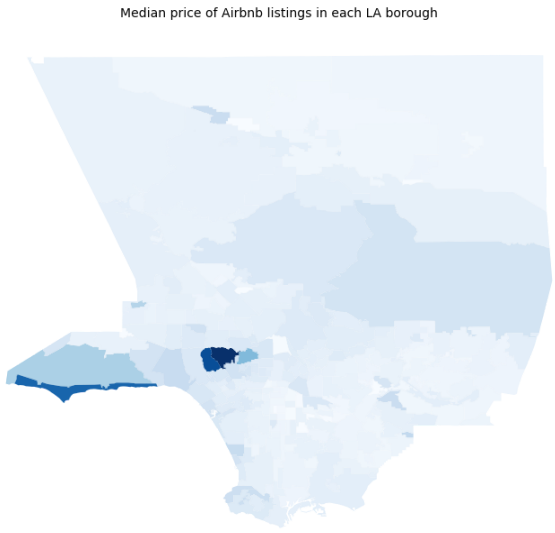
Question: Which areas of LA have the most listings, and which are the most expensive?

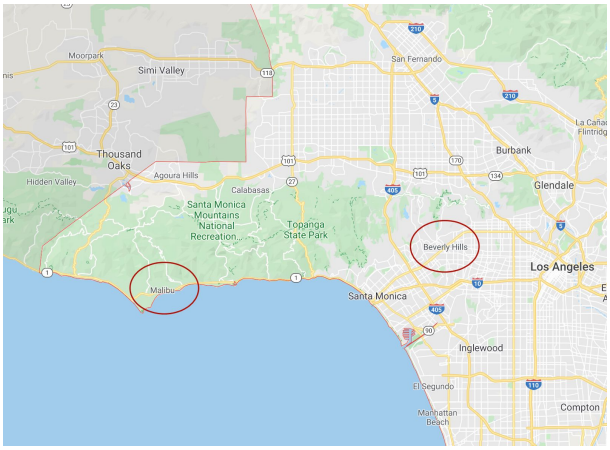
Before implementing the GeoPandas code to highlight these geographically, I made guesses that the most listings would be around Santa Monica or Hollywood, and near airports like LAX and Long Beach, and the most expensive would be the luxury listings in Malibu and Beverly Hills. I will say, I was pretty on target, but I forgot about one thing…



Disneyland! One of the LA area's biggest tourist attractions! I don't know if this is why, but I would suspect the happiest place on earth has something to do with this. (Long Beach Airport is also in this area, but that's not as exciting...)

As for the most expensive listings, we are right on point with what is expected. We can see the Malibu coastline and Beverly Hills highlighted while the rest of LA is looking a bit pale...

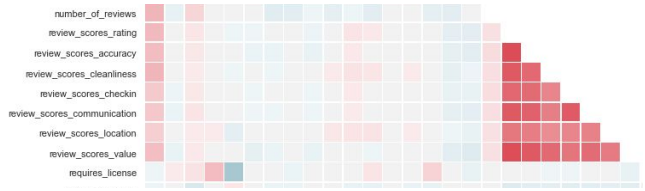


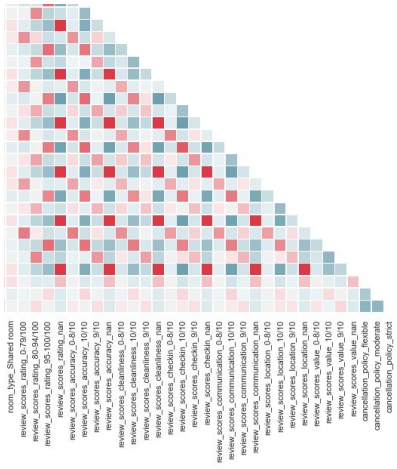


--In-Depth Analysis--

For my regression models, I decided to attempt this with a Random Forest Regression and also with XGBoost to see how they both did. I ended up focusing mostly on the Random Forest outcome, as it had the best initial accuracy and I wanted to focus on tuning one model for this project.

Before I started training and testing these models, I checked for collinearity between the remaining columns to make sure there weren't too many dependencies across columns. By inspecting the dark red and dark blue squares on the heatmap (only a portion shown here) I was able to identify some key columns to drop because their information was essentially encoded by another attribute. This is what the review_scores attributes looked like before I split them into bins, and after:





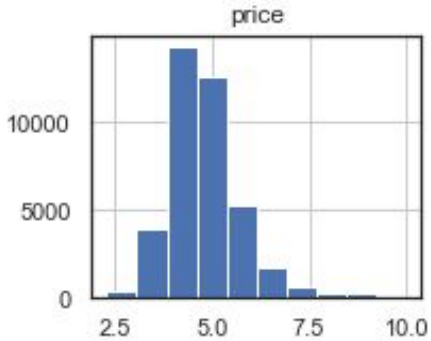
After splitting them up, I found that the attributes with the most collinearity were coming from the one-hot encoded NaN columns, so I dropped those to help the model without losing all of the review_scores information.

Other Areas of Collinearity:

- Property info (beds, bedrooms, bathrooms, accommodates) are highly correlated with each other. Dropped ‘beds’ and ‘bedrooms’, as ‘accommodates’ will still hold that information.
- Strong correlation between Private room:Entire Home/Apt, and Apt:House.
  This makes sense because they were the two main categories for their attributes. One of each pair will be dropped.
- The unknown host_response_time and host_response_rate

```
In [91]: #Dropping collinear attributes
to_drop = ['beds',
           'bedrooms',
           'guests_included',
           'host_response_rate_unknown',
           'host_response_rate_0-49%',
           'property_type_Apartment',
           'room_type_Private room',
             'review_scores_rating_nan',
            'review_scores_accuracy_nan',
            'review_scores_cleanliness_nan',
            'review_scores_checkin_nan',
            'review_scores_communication_nan',
            'review_scores_location_nan',
            'review_scores_value_nan'
           ]
transformed_df.drop(to_drop, axis=1, inplace=True)
```

I also took the numerical columns and log transformed the ones that were positively skewed. This transformed the price attribute to appear normally distributed.



I then split the data into training and testing data, and fit both models to the training set in order to then test the accuracy of the model on the test set.

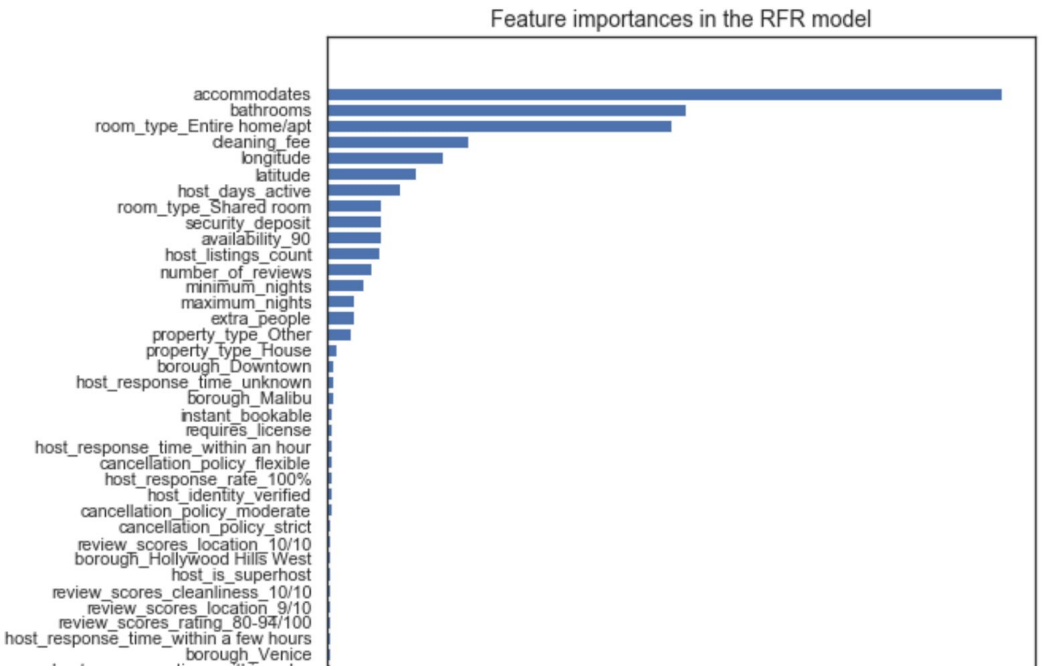
Random Forest

```
In [133]: # Create the random forest model and fit to the training data
          rfr = RandomForestRegressor(n_estimators=200)
          rfr.fit(X_train, y_train)

          #R^2 scores
          print(rfr.score(X_train, y_train))
          print(rfr.score(X_test, y_test))
```

Running this model got me a 97% R^2 score on the training data, but an 80.7% R^2 score on the test data. I was able to run through the feature importances and it looks like most of the neighborhoods have almost zero importance to the model, and 'accommodates' ended up being the most important followed by other property information first. The latitude/longitude is also high up there, so this is basically saying that neighborhoods weren't as predictive as the more precise location. I do see Malibu, Downtown, Venice, and Hollywood Hills West closer to the top here signifying that they are more predictive of price than other neighborhoods.

After seeing how little this borough attribute contributed to the model, I decided to see if getting rid of it entirely would improve the accuracy of the model. Once I dropped this attribute, which also eliminated complexity of the model, I was able to get a 80.87% R^2 score on the test data. Only a slight improvement, but I got rid of a lot of complexity so it's a win!



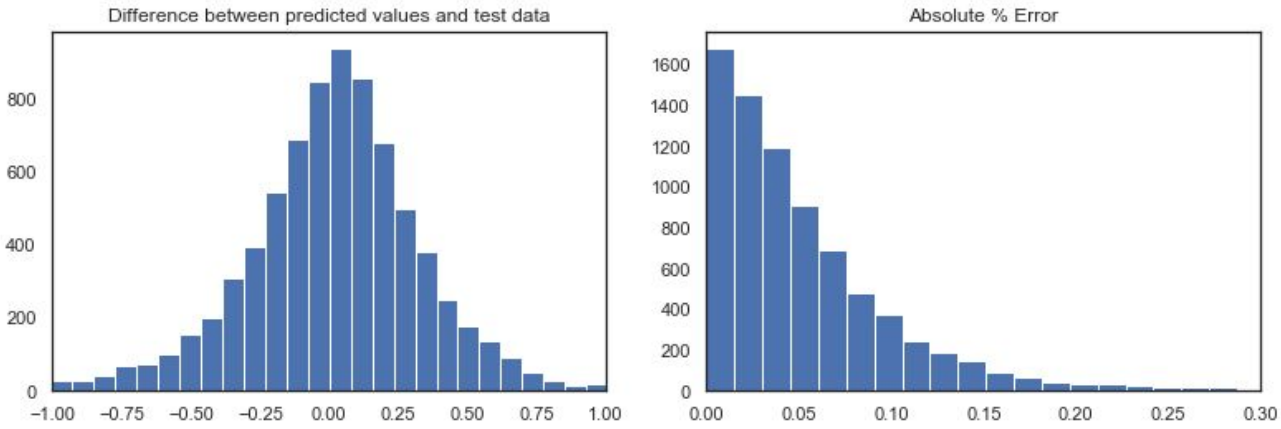
Recommendations and Further Insights

This model could contribute to the AirBnB host platform by providing more insight into the market and how much a host can expect to make by listing their space. This analysis also provides insight into which features of a listing are most important, and what a host can do to earn more for their listing.

Through modeling the data using a RandomForestRegressor, we know that we can predict what the nightly price of an AirBnB listing in LA should be with 80.87% R^2 accuracy, according to other listings and their features. We see from the feature weights that over 70% of the model is based on the 'accommodates', 'bathrooms', 'room_type' and 'latitude'/'longitude' attributes, which are all fixed numbers a host can easily come up with when going to look up whether or not to AirBnB their space.

Because the 'accommodates' feature is so important in determining price, and we saw earlier that the list price increased as the number of guests accommodated increased, we could also recommend that hosts optimize their space to accommodate as many guests (comfortably) as possible to optimize revenue. This could even lead to viewing "space-maximization" home improvement projects with a tangible return on investment, knowing that being able to accommodate more guests will earn the host more per night to offset improvement costs.

Though we have 80.87% accuracy using the R^2 measure, I was also interested to see how many predictions fell within 5% of the actual price, within 10%, and above 15%.



Total # test data	Within 5% error	Within 10% error	More than 15% error

7770	4595	6663	414

This shows us that only about 5% of the predictions came in with more than 15% error. I would recommend investigating which listings fall into this category to see if there are any patterns or trends where the model fails to predict certain listings more than others.
If there is such a pattern, then

1. The model can potentially be enhanced even more, or
2. The hosts of listings which fall into this category can be notified that they may be able to list their space for higher to earn more revenue, or may need to list it lower to gauge competitiveness in the market and then increase as needed.

There are so many directions this project could go in to provide more benefit to the hosts and investors of AirBnB, and I hope I've outlined a few key areas to explore further in this analysis.

--Supplemental Materials--

Github repository

Slide deck