Capstone Final Report

Introduction:

AirBnB manages a short term rental platform, where people can list their apartment or home and allow others to rent a room or even the entire property for days at a time. In mid-2017, according to one article, they were on pace to serve over 100 million customers for the year, representing an estimated 25% of leisure travellers and 23% of business travellers. AirBnB, like other members of the relatively new and controversial sharing economy, has managed to tap into an inefficient market - rooms or entire homes that are unused and could be earning the property-owner (and sometimes, probably illegally, the renter) some extra income.

For those listing a property on AirBnB (referred to as "hosts"), one of the most difficult decisions they need to make is how much money to charge for their listing. If the price is set too high, they may be skipped over by customers looking for a better deal. If they charge too little, they could be leaving a great deal of income unearned. AirBnB provides a predictive pricing tool to hosts, but the tool has received complaints for not being accurate enough. Some hosts even believe AirBnB may be artificially decreasing prices to keep new customers coming into the platform. A small number of third party tools exist that attempt to solve the pricing issue, but these do not take desired utilization (the rate a location is booked) into account. My tool will seek give potential hosts the ability to see what different pricing outcomes will have on their utilization rate, and see the optimal price for the number of days they want to be booked.

Problem Statement:

I plan to develop two models. A regression model to predict the price of a listing, and a second ensemble model to predict overall utilization, which will use the predicted price as an input. The first goal will be to provide a more accurate prediction of what price hosts should list their property at, based on a variety of factors gleaned from multiple datasets. Potentially relevant variables from AirBnBs own data can include property amenities, the number of reviews, the average rating, the description of the property, the number of guests allowed, the time of year, and more. There are already a handful of third-party pricing tools available, so I will seek to differentiate my model by joining AirBnB data to other sources to provide a complete picture of neighborhood popularity, or how "up and coming" it is, under the assumption that more popular areas will be able to set higher prices. This will be done by joining data from Yelp's academic dataset.

The second goal for my model will be to predict utilization rate - defined as the percentage of days the property is booked in a given year. This model will be based on the recommended price, or a price the host provides, so they can perform a "what-if" analysis. This could be a harder problem to solve because the AirBnB dataset includes the number of days a location was booked, but not the number of days it was made available by the host, so the true availability needs to be estimated based on certain assumptions, particularly around number of reviews a listing has (more on this later).

Client Description:

The client for this project is any host placing a listing on AirBnB that wants a second opinion on the pricing recommendation AirBnB has provided. Hosts will be able to use this tool to determine the optimal price to set their listing at, depending on how many days they would like their listing to be booked and how much money they would like to earn.

Though not the primary client, an alternate client would be potential AirBnB customers that want to check if they are being charged a fair price for a listing before they book it.

Data Description:

The primary data to be used in this project comes from Inside AirBnB, a website that scrapes and maintains data from all listings on AirBnB and makes them available by city. It contains detailed data on each listing in the city, and consists of 96 variables. These include details on the listing itself, such as neighborhood name, latitude and longitude (masked to within 500ft of the actual location), type of property, number of rooms and beds, and many more.

I also plan to make use of Yelp's Academic dataset to determine which neighborhoods have the most popular restaurants and businesses. This dataset contains information on businesses and restaurants in approximately 10 cities and is provided as a series of JSON files. I will make use of the Business file, because it contains number of business near each AirBnB listing and their average rating. These data points will be fed into my predictive models as features.

Approach:

I plan to follow the CRISP-DM methodology for this project, which stands for Cross-Industry Standard Process for Data Mining. This is made up of six phases, which are iterative.

- Business Understanding - this involves framing the question and learning as much as possible about the business and how the data to be used is collected. I have already framed the question, but I plan to do some research and learn how others have approached similar pricing problems.
- Data Understanding (EDA) - this phase involves performing exploratory data analysis, generating summary statistics and data visualizations, to look for interesting trends or patterns in the data.
- Data Preparation - data transformations are made in this phase. Here I will remove unnecessary fields and create derived variables, for example taking the listing amenities and separating them into multiple binary flag fields. I will also join the AirBnB dataset to the Yelp dataset, using either Neighborhood Name or Latitude/Longitude to join on.
- Modeling - the actual model-building is done in this phase. To predict pricing, which is a continuous numeric target, I plan to start with a simple linear regression, then move on to more complex models such as random forests and compare the results. To predict utilization I will calculate the difference between the predicted price and the actual price, to see whether a listing being over- or undervalued impacts how often it is booked.

- Evaluation - after building models against a training dataset I will run them against a test subset of data, which was removed prior to the modeling phase. I will need to do this twice, first to test the price predictions, then again to test the utilization prediction.
- Deployment - deploying the model. This step will not be covered as this is a learning project not meant for production, but in the real world the end result would be some type of interactive tool where end users can input facts about their own listing and view a price prediction related to it.

Deliverables:

The deliverables for this project will include:

- The project report writeup
- A presentation and associated slide-deck
- Jupyter Notebooks containing all code used for data exploration, transformation, model-building, and evaluation
- If time allows - an interactive dashboard where users can enter relevant data points and view a prediction for their listing.

Data Cleaning/Wrangling Steps (Data Preparation)

Code for these steps is contained in the following Jupyter Notebook scripts, available on my GitHub page:

Data Wrangling Script:
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/Capstone_DataWrangling_Script.ipynb
Data Merge Script:
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/Capstone_Data_Merge_Script.ipynb
Statistical Inference Script (Utilization Rate calculated here):
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/Capstone_Statistical_Inference.ipynb

Cleaning the AirBnB Listings Dataset

Initial Inspection and Removing Redundant Fields
The initial Toronto dataset contained 17,542 records and 96 columns. My first step was to remove fields that do not seem useful for various reasons. For example I dropped all columns related to reviews. My model is meant to provide new hosts (people that list their property on AirBnB are referred to as hosts) with a tool for deciding the optimal price to set their listing at. Therefore basing the model on reviews would not help, as new listings do not have reviews yet. The exception being reviews_per_month. This field can be used to estimate how heavily utilized a listing is, which is going to be the second stage of my model's predictions.

I also checked for and removed columns where all records contained a single value. For example last_scraped, which is the date the data was scraped by Inside AirBnB, and is_business_travel_ready, which consisted entirely of the value “f”. Altogether dropping unnecessary variables, and those with a single value, brought me from 96 to 42 variables.

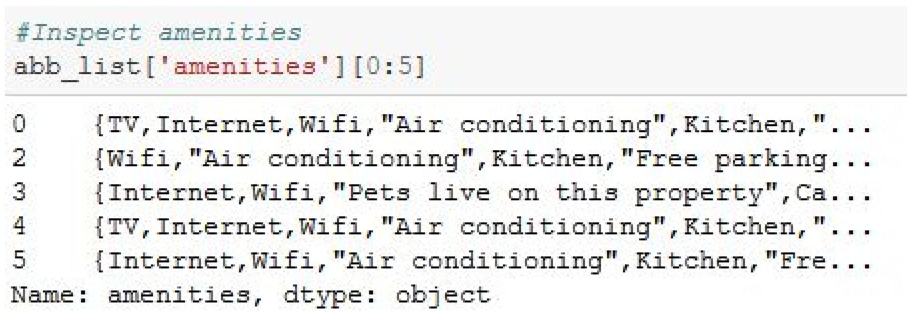
Null Values
My next step was to look for null values. 15 columns had missing data, ranging from a few records (host_is_superhost) to one column that was 100% missing (neighbourhood_group_cleansed). I dropped the 100% missing column entirely. A few variables were between 86-98% missing, so I deleted these columns and replaced them with a binary flag variable which indicates if there was a missing value. The flag variable can be fed into a predictive model.

For the remaining variables with nulls, I either replaced missing values with 0 (for example security_deposit, where I assumed missing meant it was not charged), “Unknown” (for text fields like neighborhood), or the most common value or mode (for example bathrooms, and host_is_superhost). I wrote a function to handle these missing values programmatically. The function takes a dataframe, a list of columns, and arguments “how” (drop or impute), “imp_num_method” (defaults to mean, this determines the statistic to replace numeric variable nulls with), and “imp_with” (defaults to most_common, this determines what to replace nulls with in categorical variables).

Outliers
I used pandas .describe() method with the percentiles argument to view the mean, median, standard deviation, min/max, and the 1st, 10th, 50th, 90th, and 99th percentiles. I used these to spot outliers in the data, by looking for wide gaps between the min and 1% (or 10%), the median and mean, or the max and 99% (or 90%). Wide gaps between these numbers, usually more than 2-3 standard deviations, are a good indicator that outliers exist.

I generally handled outliers by trimming them to the within 3 standard deviations of the mean. For a few variables like price, which had a relatively small number of records above $2000 (around 200 out of 17K in the dataset) I decided to drop all records above the 99th percentile. The mega-expensive properties were not indicative of the majority of listings and I felt they would confuse my model. I also created a function for handling outliers programmatically, with arguments to trim or drop depending on need.

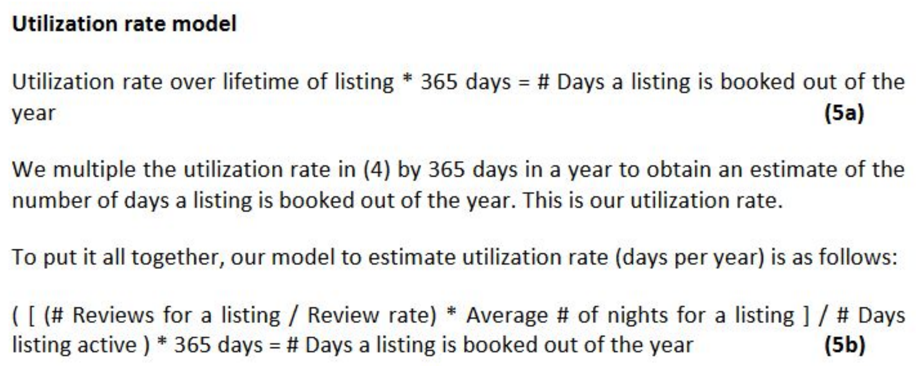
Unpacking Amenities
The Amenities variable represented a special challenge. It’s values are actually a list containing all the amenities a property has, for example “TV”, “Wifi”, “Air Conditioning”, etc. Here is an example of the first 5 records in the dataset:


```
#Inspect amenities
abb_list['amenities'][0:5]
```
```
0    {TV,Internet,Wifi,"Air conditioning",Kitchen,"...
2    {Wifi,"Air conditioning",Kitchen,"Free parking...
3    {Internet,Wifi,"Pets live on this property",Ca...
4    {TV,Internet,Wifi,"Air conditioning",Kitchen,"...
5    {Internet,Wifi,"Air conditioning",Kitchen,"Fre...
Name: amenities, dtype: object
```

There are 185 amenities total. I knew I wanted to unpack these into binary flag variables for each amenity, indicating whether the listing has that amenity (True) or not (False). To accomplish this I wrote a function that loops through each listing and checks if each amenity exists for it. If it does, a new column is created in the format “has_x”, where x is the name of the amenity.

Estimating Utilization

My secondary goal is to predict the utilization rate of bookings, so next I needed to derive that utilization rate. The process is described on pages 49-51 in the "Analysis of the impact of short-term rentals on housing" report from San Francisco's Budget and Legislative Office. The document is available here: https://sfbos.org/sites/default/files/FileCenter/Documents/52601-BLA.ShortTermRentals.051315.pdf
I recommend reading the pages few cited above to understand the full process of calculating utilization rate, but this section summarizes the model I used:



Utilization rate model

Utilization rate over lifetime of listing * 365 days = # Days a listing is booked out of the year **(5a)**

We multiple the utilization rate in (4) by 365 days in a year to obtain an estimate of the number of days a listing is booked out of the year. This is our utilization rate.

To put it all together, our model to estimate utilization rate (days per year) is as follows:

([(# Reviews for a listing / Review rate) * Average # of nights for a listing] / # Days listing active) * 365 days = # Days a listing is booked out of the year **(5b)**

The one change I made is to skip the final step of multiplying by 365 days to get # of days booked per year. I prefer to estimate the percentage of days booked, so I leave this last step out. 30 listings went above 100%, so I trimmed these down to 100%.

Note on the accuracy of the Utilization Rate Model:

It is important at this stage to point out that the utilization rate model is dependent on an assumption that may not hold up under statistical analysis. This is the use of a constant Review Rate, or the percentage of people that book a listing and then leave a review.

This number is estimated at 72% by AirBnB, and 30.5% by the City of New York (for my project I decided to use AirBnB’s 72% rate because it results in fewer listings going over 100%). This number is then divided into the total # of reviews a listing has received to estimate how many times the listing has been booked. Multiple sources use this method, including the City of San Francisco (link above and page 36 of http://commissions.sfplanning.org/cpcpackets/2014-001033PCA.pdf) and Inside AirBnB (http://insideairbnb.com/about.html#disclaimers).

However, using a static number instead of an actual, known distribution means it is likely that this method underestimates actual bookings for listings with few reviews, and overestimates actual bookings for listings with many reviews. I have seen this in the fact that some listings went over 100%. **Though the use of a static review rate is a faulty assumption, I will use it because there is no way to get the true number of bookings a listing has received.**

With these changes completed, the final dataset contained 16,939 records and 233 variables.

Cleaning the Yelp Business Dataset

Initial Inspection and Removing Redundant Fields
This dataset required much less cleaning than the AirBnB data. The initial dataset had 188,593 records and 15 columns. The raw file has over 100 cities in it, but 6 make up the vast majority of records. The city of Toronto and surrounding suburbs in the state of Ontario, which I have decided to analyze with this project, makes up 32,393 records. I saved those records and deleted the rest.I also removed closed businesses (where is_open = 0), bringing the dataset to 26,031 records.

Next I dropped columns I do not anticipate using in my modeling. My goal with this file is to create a variable for each AirBnB listing that shows the number of businesses and average star rating within a certain radius, not yet defined but likely .1 or .25 mile. The Yelp dataset has Neighborhood and Lat/Long, both of which I can use to match with the listing data. Therefore I do not need fields like address, city, name, etc. I was able to drop 9 variables, bringing the total in the dataset from 15 to 7.

Limiting to just Restaurants and Bars
I used the Categories variable to limit the dataset to only the categories “Food”, “Restaurant”, “Nightlife”, and “Bar”. I did this because my thesis is that having restaurants and bars in close proximity will increase the price of a listing. The Categories variable was a nested list similar to Amenities in the AirBnB data, so I used a similar process. Except rather than unpack the variable, I simply checked if those words were in the value and kept only those records where the result was True.

Null Values
Checking for Nulls initially returned no missing values, but visually inspecting the data I noticed blanks in the neighborhood column. I used the .value_counts() method to determine this is actually the most common neighborhood, around 20% of records are missing neighborhood. I plan to use this variable to limit the number of lat/long matches required when merging with the AirBnB data (to improve performance). I don’t want to delete these missing records though, as they represent too much data to lose. For now I replaced the “” with “Unknown”. No other values had missing data.

Outliers
There were no outliers in the fields I kept for analysis.

Merging the AirBnB and Yelp Datasets

My next step was to merge the cleaned AirBnB and Yelp datasets together. First I joined the datasets on Neighborhood, after spending extensive time cleaning Neighborhood names to have as many matches as possible. The match on Neighborhood was important because it limited the size of the dataset after merging. Next I wrote a function to compute the distance between each listing and business based on latitude and longitude. I then created two sets of three columns each: the number of businesses, the number of reviews for those businesses, and the average star rating of those businesses, within .1 mile and .5 mile each (so 6 new columns were created).

Initial Findings (Exploratory Data Analysis)

Here are my preliminary findings after doing visual and statistical exploratory data analysis. The code is split into two Jupyter Notebooks, one for the data visualizations, the other for statistical inference testing.

Data Visualization Script:
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/Capstone_Data_Story_Script.ipynb
Statistical Inference Script:
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/Capstone_Statistical_Inference.ipynb

Distribution of Prediction Variables

Price has a skewed distribution with a long tail going to the right. It is not normally distributed. Because of this I created a log version of price which appears to more closely follow a normal distribution. I will make predictions on log price, then convert back to raw form.

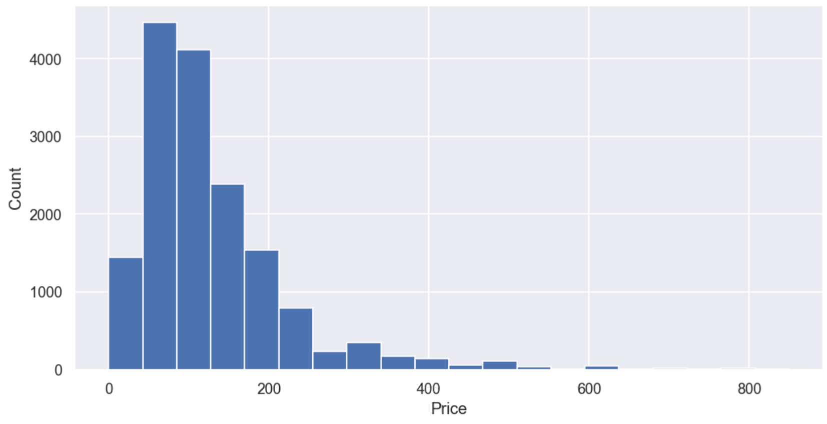


Figure 1 - Distribution of Price

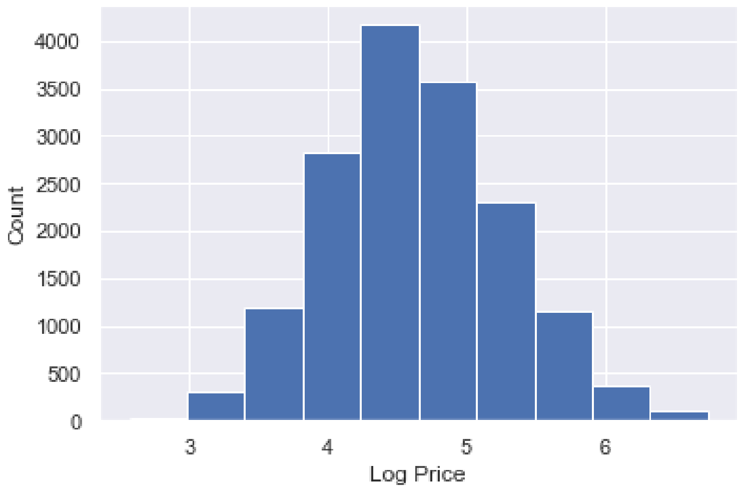


Figure 2 - Distribution of Log Price

Utilization follows a similarly skewed distribution.

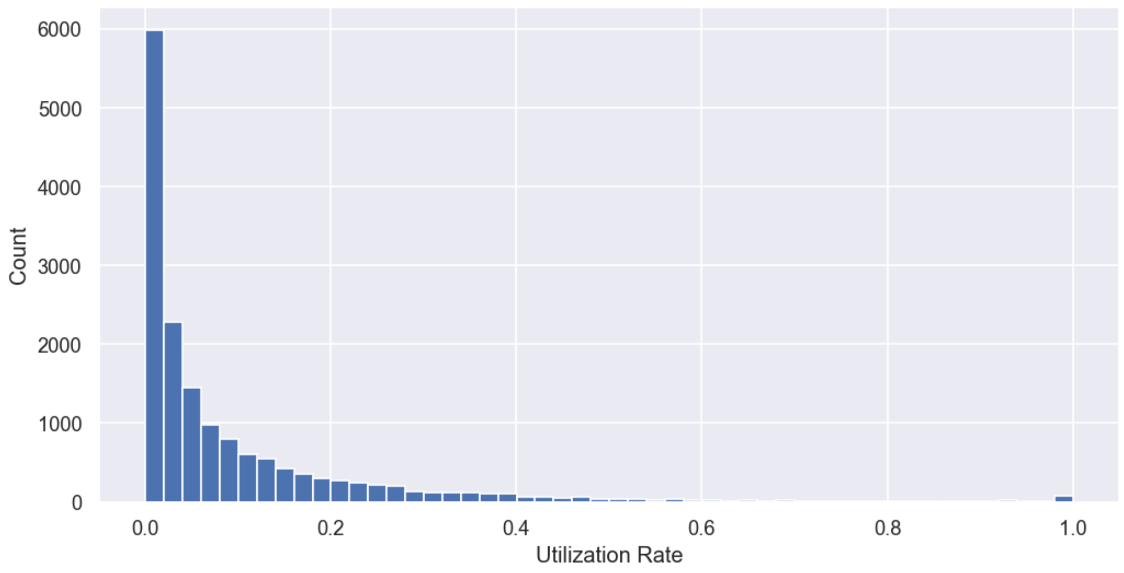


Figure 3 - Distribution of Utilization Rate

A correlation test between Price and Utilization shows a correlation close to 0 (-0.03) with a statistically significant p-value at 2.949e-6.

Facts about Listing

Beds, Bedrooms, Bathrooms, and Accommodates

Figure 4 appears to show a strong relationship between Price and Beds, Bedrooms, Bathrooms, and Accommodates (the number of people that can stay in a listing). As each of these variables goes up, the price goes up as well.

Statistical inference confirms this relationship. Calculating the correlation coefficient between Price and Accommodates, Beds, and Bedrooms shows a coefficient of .59, .47, and .43 respectively. The correlation with Bathrooms is weaker but still shows practical significance at .3. All four correlations have a p-value extremely close to 0, meaning they are statistically significant as well.

The correlation with Utilization is not nearly as strong. It hovers around 0 for all four variables.

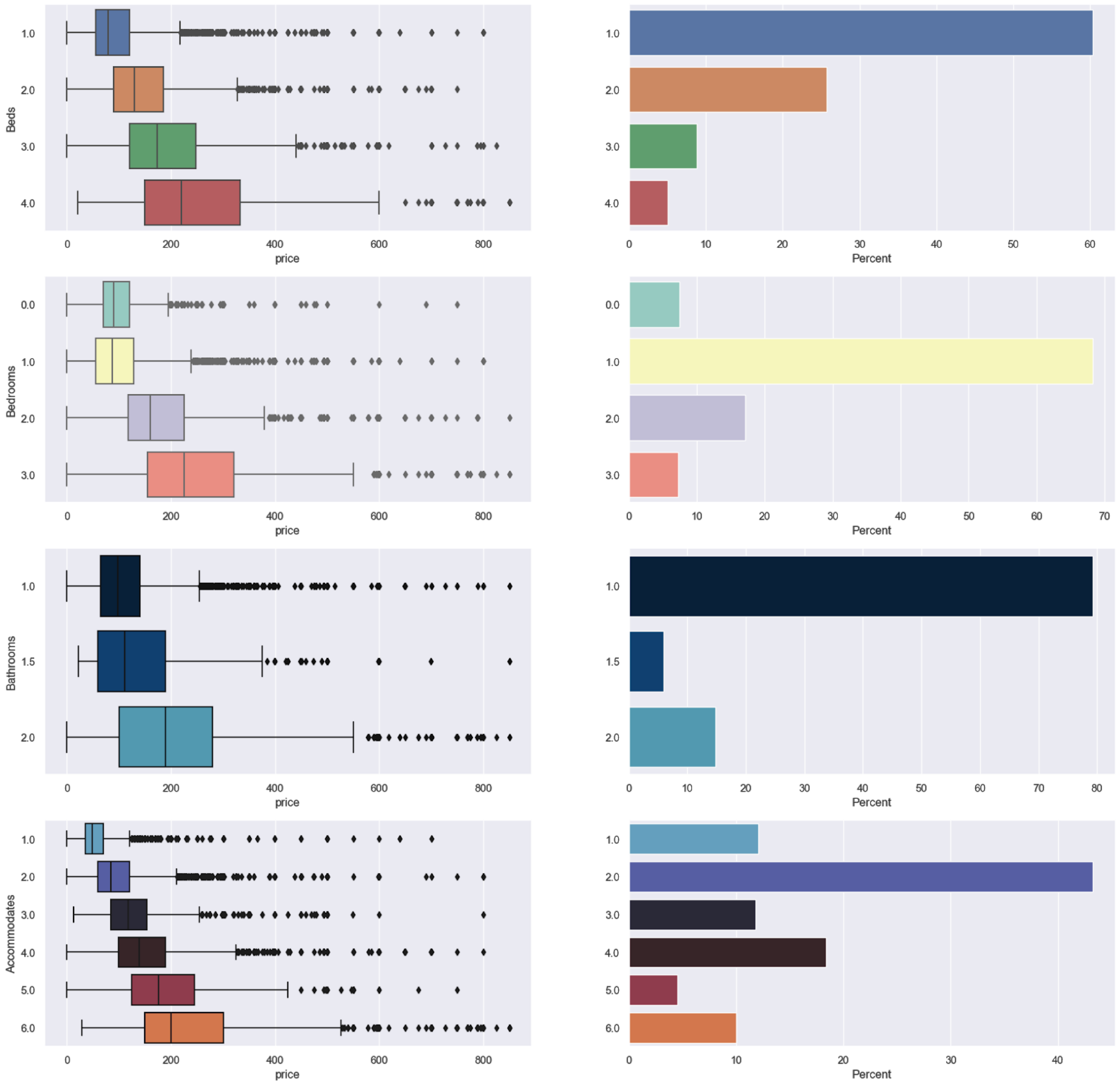


Figure 4 - Relationship Between Price and Beds, Bedrooms, Bathrooms, Accommodates

Property Types

A visualization of Property Type shows some relationship with price. Some types are clearly more valuable than others, such as Condominium, Loft, and Serviced Apartment. But it is difficult to tell the strength of this relationship based on the visual alone.

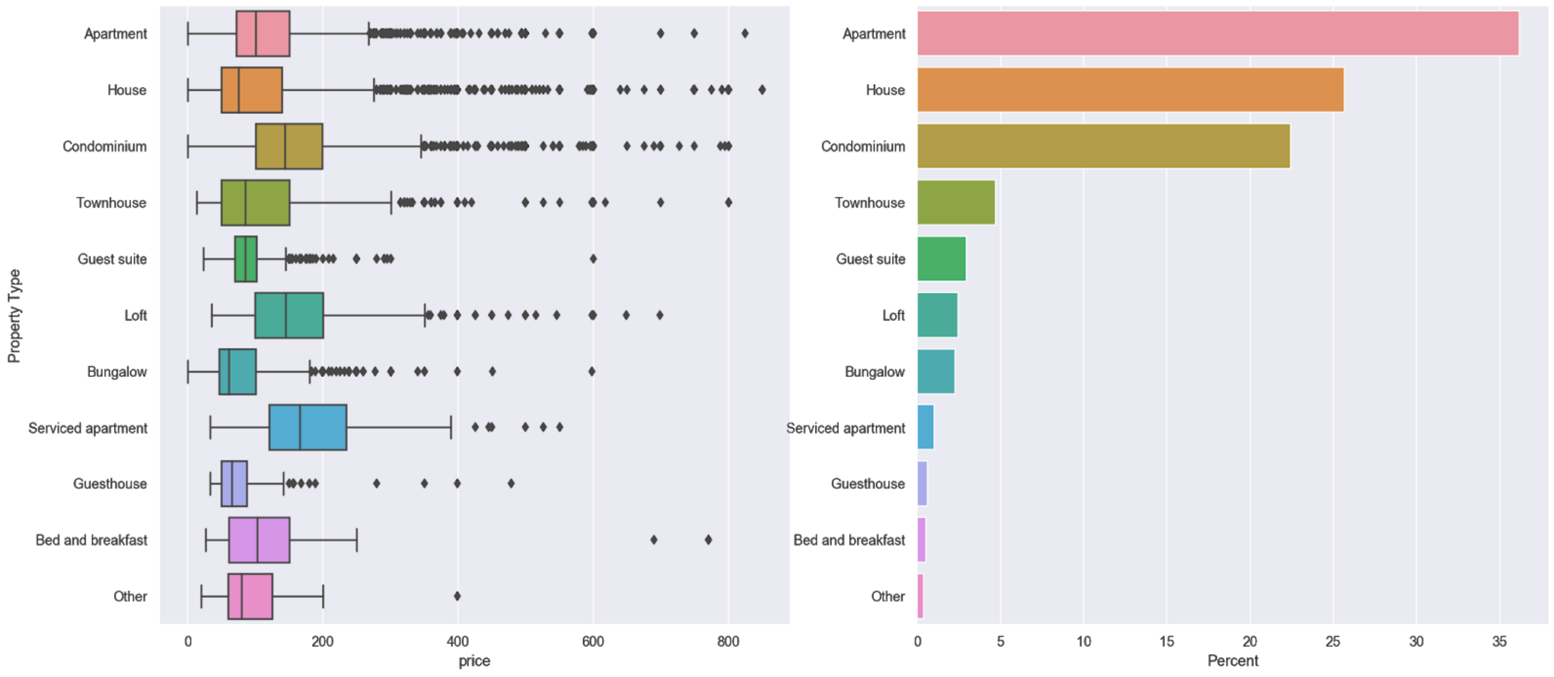


Figure 5 - Relationship Between Property Type and Price

To get a more accurate view of the relationship, I did a series of t-tests comparing the mean price of each property type against all other property types. This generated Table 1.

	prop_type	stat	pvalue	mean_True	mean_False	count_has	mean_diff
8	Serviced apartment	7.575520	0.00000000	183.398773	127.092050	163	56.306723
2	Condominium	24.550107	0.00000000	161.161748	117.978424	3592	43.183325
5	Loft	8.299147	0.00000000	166.892031	126.688556	389	40.203475
10	Bed and breakfast	0.394169	0.69346145	131.674419	127.643561	86	4.030857
3	Townhouse	-2.353052	0.01863205	119.765873	128.056630	756	-8.290757
0	Apartment	-7.826311	0.00000000	119.903215	132.056615	5786	-12.153400
7	Other	-2.225591	0.02605544	112.567708	127.848429	192	-15.280721
1	House	-10.821039	0.00000000	113.967209	132.405851	4117	-18.438642
4	Guest suite	-8.201626	0.00000000	92.602537	128.732432	473	-36.129895
6	Bungalow	-8.030962	0.00000000	88.387363	128.578823	364	-40.191460
9	Guesthouse	-4.300339	0.00001715	86.084211	127.913369	95	-41.829158

Table 1 - Property Type t-test results

To read the table, stat represents the statistical test metric. mean_True and mean_False are the mean price for that property type, and the mean price for all other property types, respectively. Count_has is the count of records for that property type. And mean_diff is the difference in mean price between that property type and all other property types. A positive value means the property type earns more than the others, a negative value means it earns less.

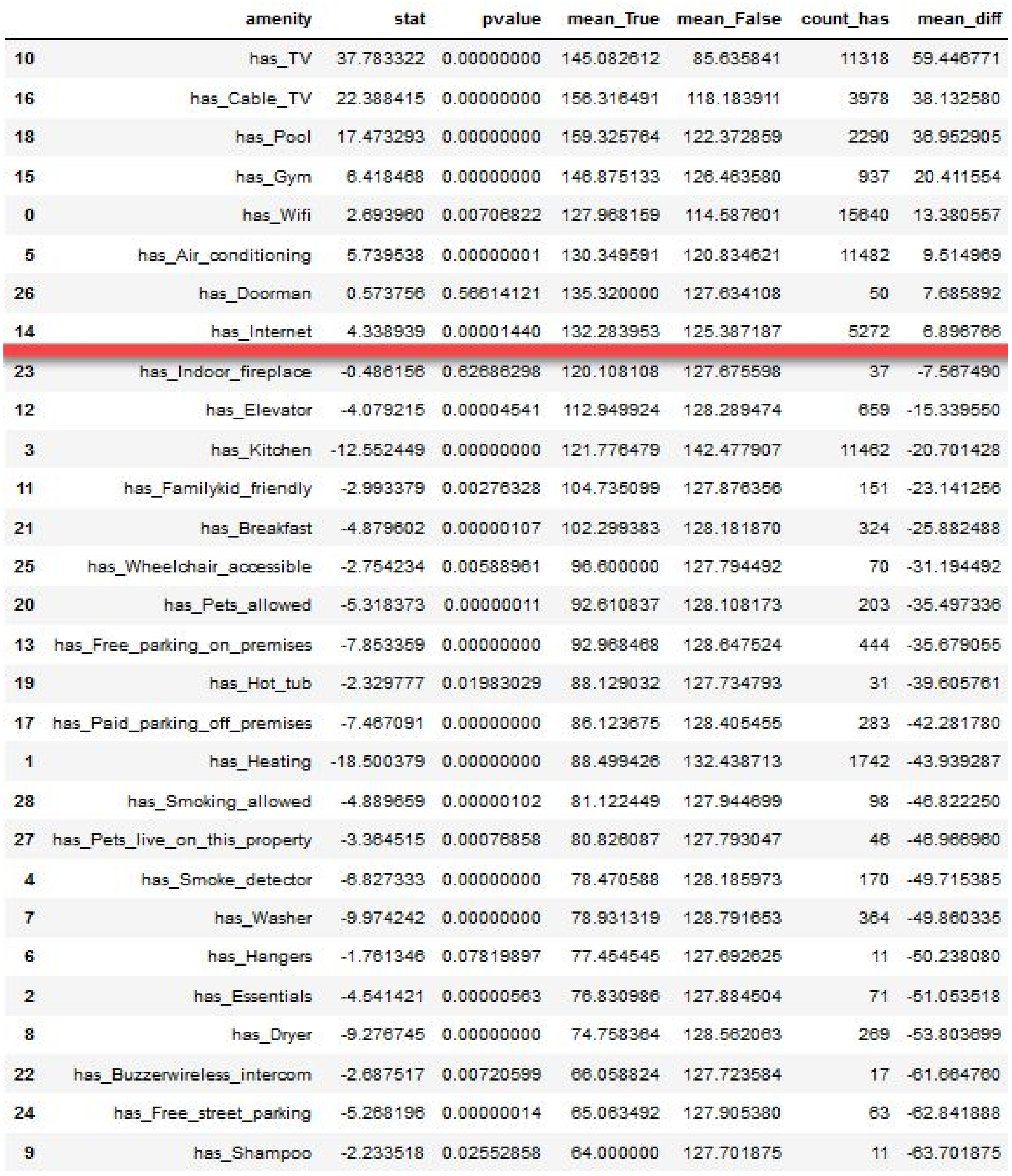
It is clear that Serviced Apartment, Condominium, and Loft are the most valuable property types, being worth $56, $43, and $40 more than the rest respectively. On the flipside, Guest Suites, Bungalows, and Guesthouses are worth less than the others, losing $36, $40, and $41 respectively.

Amenities
I performed a similar set of t-tests on each of the amenities flag variables, after removing any that only 10 listings contain. This resulted in Table 2, visible on the next page.

Interestingly, most amenities resulted in a decrease in price. The most significant positive-impact amenities are having a TV/Cable TV, a Pool, a Gym, and Wifi.

On the negative-impact side, not all of the amenities make sense at first glance. For example having Shampoo as a listed amenity is good for a $63.70 price decrease, though only 11 listings have this item. Having a Washer or Dryer are each good for about a $50 decrease. I would think these would have a positive impact, but the data does not support it. My theory is that advertising things like Shampoo, Essentials, etc. means that the listing does not have much else going for it. Perhaps it is in a less exciting area, or is a small guesthouse for example. Advertising something like free street parking could mean that parking is hard to find, and having a buzzer/wireless intercom might mean it is in an unsafe neighborhood, or just the getting into the listing is cumbersome.

Other negative-impact amenities make more sense. Allowing pets or having pets on the premises, allowing smoking, and having paid parking, all have a negative impact and seem logical.



	amenity	stat	pvalue	mean_True	mean_False	count_has	mean_diff
10	has_TV	37.783322	0.00000000	145.082612	85.635841	11318	59.446771
16	has_Cable_TV	22.388415	0.00000000	156.316491	118.183911	3978	38.132580
18	has_Pool	17.473293	0.00000000	159.325764	122.372859	2290	36.952905
15	has_Gym	6.418468	0.00000000	146.875133	126.463580	937	20.411554
0	has_Wifi	2.693960	0.00706822	127.968159	114.587601	15640	13.380557
5	has_Air_conditioning	5.739538	0.00000001	130.349591	120.834621	11482	9.514969
26	has_Doorman	0.573756	0.56614121	135.320000	127.634108	50	7.685892
14	has_Internet	4.338939	0.00001440	132.283953	125.387187	5272	6.896766
23	has_Indoor_fireplace	-0.486156	0.62686298	120.108108	127.675598	37	-7.567490
12	has_Elevator	-4.079215	0.00004541	112.949924	128.289474	659	-15.339550
3	has_Kitchen	-12.552449	0.00000000	121.776479	142.477907	11462	-20.701428
11	has_Familykid_friendly	-2.993379	0.00276328	104.735099	127.876356	151	-23.141256
21	has_Breakfast	-4.879602	0.00000107	102.299383	128.181870	324	-25.882488
25	has_Wheelchair_accessible	-2.754234	0.00588961	96.600000	127.794492	70	-31.194492
20	has_Pets_allowed	-5.318373	0.00000011	92.610837	128.108173	203	-35.497336
13	has_Free_parking_on_premises	-7.853359	0.00000000	92.968468	128.647524	444	-35.679055
19	has_Hot_tub	-2.329777	0.01983029	88.129032	127.734793	31	-39.605761
17	has_Paid_parking_off_premises	-7.467091	0.00000000	86.123675	128.405455	283	-42.281780
1	has_Heating	-18.500379	0.00000000	88.499426	132.438713	1742	-43.939287
28	has_Smoking_allowed	-4.889659	0.00000102	81.122449	127.944699	98	-46.822250
27	has_Pets_live_on_this_property	-3.364515	0.00076858	80.826087	127.793047	46	-46.966960
4	has_Smoke_detector	-6.827333	0.00000000	78.470588	128.185973	170	-49.715385
7	has_Washer	-9.974242	0.00000000	78.931319	128.791653	364	-49.860335
6	has_Hangers	-1.761346	0.07819897	77.454545	127.692625	11	-50.238080
2	has_Essentials	-4.541421	0.00000563	76.830986	127.884504	71	-51.053518
8	has_Dryer	-9.276745	0.00000000	74.758364	128.562063	269	-53.803699
22	has_Buzzerwireless_intercom	-2.687517	0.00720599	66.058824	127.723584	17	-61.664760
24	has_Free_street_parking	-5.268196	0.00000014	65.063492	127.905380	63	-62.841888
9	has_Shampoo	-2.233518	0.02552858	64.000000	127.701875	11	-63.701875

Table 2 - Amenities t-test results (red line indicates boundary between positive and negative Amenities)

Yelp Metrics
Statistical tests reveal some interesting relationships between the Yelp metrics (number of businesses, number of reviews for those businesses, and average rating for those businesses, within .1 and .5 miles of each listing) and price.

First, simply having a bar or restaurant within .1 mile of a listing is worth a $35 price increase, and is statistically significant with a p-value of 5.26e-118. 6,832 out of 16,011 (42.7%) of listings have this characteristic. Having a bar or restaurant within .5 mile of a listing is good for a $31 price increase, and has a similarly low p-value indicating statistical significance. 14,219 out of 16,011 (88.8%) of listings have a business within .5 mile.

I also tested correlations between these metrics and price/utilization. The metrics within .1 mile do not show a strong correlation with price or utilization (none go above .09 correlation coefficient for price, or .02 for utilization).

These is a better correlation between price and the Yelp metrics within .5 mile. Number of businesses and number of reviews each has a .23 correlation coefficient against price, and this is statistically significant with p-values of 4.62e-17 and 8.26e-177 respectively. Average rating does not appear to be correlated with price. None of the metrics has a correlation above .05 with utilization.

Modeling

Code for this stage can be found on my GitHub page here:
https://github.com/jkarpen/Springboard_Projects/blob/master/Capstone/Scripts/5%20-%20Capstone_Modeling_Script.ipynb

Before modeling I completed a few more data cleanup steps. First I duplicated the dataset, creating one version meant to predict Price, and another version meant to predict Utilization Rate. Next I dropped the amenity and “not_NA” flag variables if their t-tests revealed a lack of significance against the respective outcome variable (using the t-tests described earlier in the EDA section).

Next I used one-hot encoding to convert my categorical variables into dummy variables, split my datasets into X and Y subsets, and split them further into train and test subsets to ensure clean model-building.

I used SciKit-Learn for all models, because I wanted to take advantage of the power of that package for generating pipelines that automate model-building, grid-searching multiple parameter values, and cross-validation. All scaling was done with MinMaxScaler because most of my numeric variables are right-skewed. GridSearchCV was used to automate parameter tuning, with 5-fold cross-validation.

The first model built was a simple Ordinary Least Squares model against Price, feeding in all variables. This would serve as my baseline model. The result had an R^2 of 0.50 on the test set. Swapping in Log Price instead of Price as the outcome variable showed an instant improvement to 0.63 so I went with Log Price for all models going forward.

Next I ran the following models to predict Log Price:
- LASSO, using GridSearchCV to find the optimal value for Alpha.

- Ridge Regression using just the 15 explanatory variables left over by LASSO (coefficients not reduced to 0). GridSearchCV to find optimal value for Alpha.
- Ridge Regression using all variables. GridSearchCV to find optimal value for Alpha.
- Random Forest with just the LASSO variables. GridSearchCV used to find optimal values for whether to use bootstrap samples, number of estimators (trees), max tree depth, min samples at each leaf, and min samples to split.
- Random Forest with all variables. GridSearchCV used to find optimal values for number of estimators (trees), max tree depth, min samples at each leaf, and min samples to split.
- Ridge Regression with LASSO variables, plus other variables ranked highly in importance by the Random Forest model.

Of these, the highest-scoring model was the Random Forest with all variables included. This scored 0.66 in test R^2. Unfortunately this model showed definite signs of overfitting, with R^2 being 0.95 on the training set. No other model scored above 0.66 in training so this was a definite outlier.

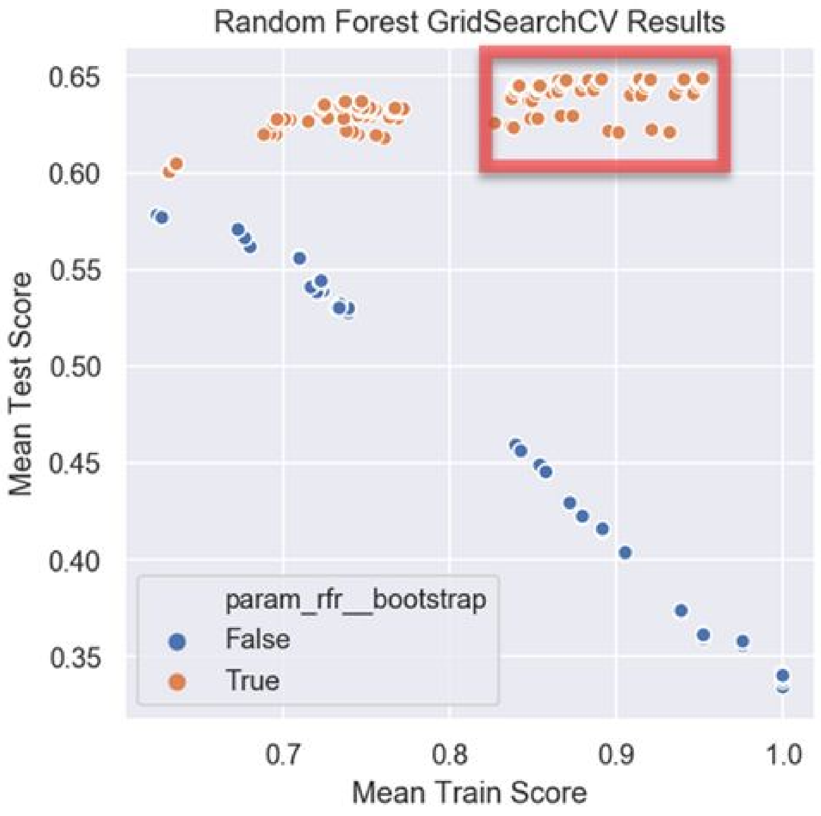
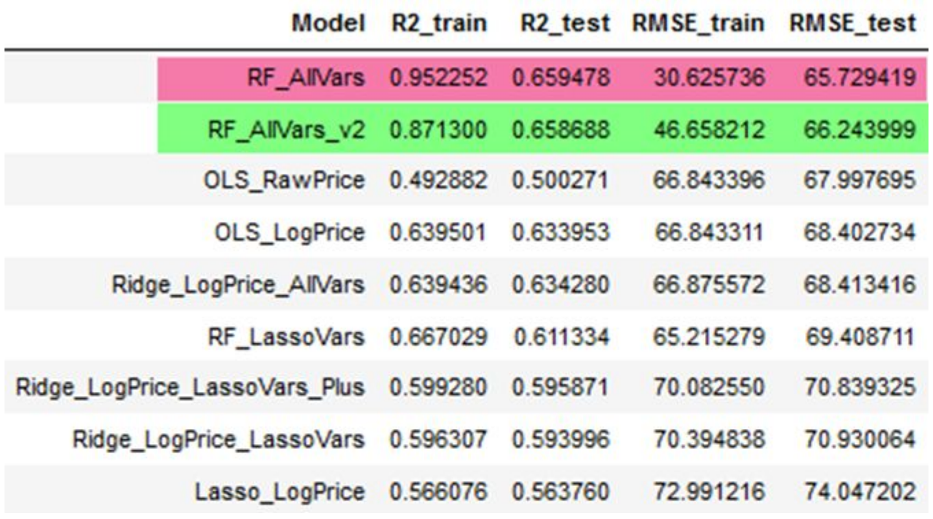


Figure 6 - Random Forest Model GridSearchCV Results, Mean Test Score vs. Mean Train Score

Figure 6 shows a scatterplot of all of the test and train results , with Mean Test Score on the y-axis and Mean Train Score on the x-axis. The “best estimator” found by GridSearchCV is in the top right quadrant, but that cluster of orange dots has a train score very close to 1.0, indicating very strong overfitting has occurred. I decided to look at the details of the models enclosed in the red box to see if I could find one with almost the exact same test score, but a less inflated train score. These models showed that dropping the maximum tree depth, increasing minimum samples for a split to 10 (the “best estimator” chose 2) would return a model with almost identical performance in test but much less overfitting. Full results shown in Table 3 below. RF_AllVars is the original Random Forest model. RF_AllVars_v2 is the model with tweaked parameters to prevent overfitting.



Model	R2_train	R2_test	RMSE_train	RMSE_test
RF_AllVars	0.952252	0.659478	30.625736	65.729419
RF_AllVars_v2	0.871300	0.658688	46.658212	66.243999
OLS_RawPrice	0.492882	0.500271	66.843396	67.997695
OLS_LogPrice	0.639501	0.633953	66.843311	68.402734
Ridge_LogPrice_AllVars	0.639436	0.634280	66.875572	68.413416
RF_LassoVars	0.667029	0.611334	65.215279	69.408711
Ridge_LogPrice_LassoVars_Plus	0.599280	0.595871	70.082550	70.839325
Ridge_LogPrice_LassoVars	0.596307	0.593996	70.394838	70.930064
Lasso_LogPrice	0.566076	0.563760	72.991216	74.047202

Table 3 - Model scoring results

Though RF_AllVars_V2 was the best performing model overall, I should point out that it only scored .02 higher in terms of R^2 on the test set when compared with simple Ordinary Least Squares. And for a simpler model that performed almost equivalently, the model RF_LassoVars is a Random Forest with just the 15 predictor variables identified by running LASSO on the dataset. This model performed only .03 worse on testing R^2 but would be much easier to explain.

Next I used the predicted price from my best performing model to try to predict the Utilization Rate variable. This was not nearly as successful. The price delta appears to have a relationship to Utilization Rate as shown in Figure 7 - the highest-utilization listings are those that have a price delta close to 0. Overpriced listings (those with a positive price delta) have much lower utilization rates.

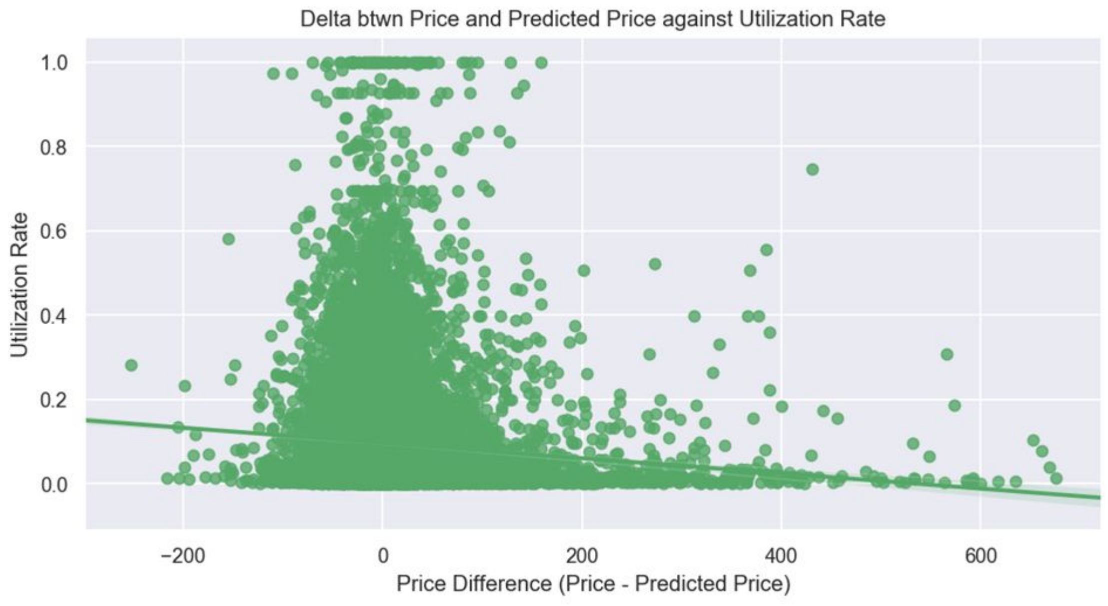


Figure 7 - Price Difference (Price minus Predicted Price) against Utilization Rate

This relationship did not lead to a strong model, however. R^2 was close to 0 when running OLS with just Price Delta as the predictor variable. Adding more variables lead to an R^2 of 0.12 in training. This is a good start and could be useful, but probably needs more work to turn into a production model.

Recommendations

The most important features identified by the final Price prediction Random Forest model are shown in Figure 8. Also included are Tables 4 and 5, which show the coefficients and their exponentiated values converted to show percent change. The following recommendations come from reviewing this information.

1. Bigger listings can charge a higher price. Having more beds, bedrooms, bathrooms, and accommodating more guests allows a host to charge a higher premium to stay at their listing.
2. It is very important to let guests have the entire place to themselves. Room types Private Room and Shared Room lead to significant price reductions when compared with listings that give guests the run of the place.
3. Simply charging a security deposit or cleaning fee leads to a price reduction. Higher fees, however, tend to correspond with more expensive properties. This could mean that less valuable listings that charge these fees are forced to drop their price to compensate, but more popular properties are able to get away with charging a fee and charging a higher nightly rate.
4. Certain neighborhoods are worth more than others. The Entertainment District, Yorkville, Rosedale, for example, are able to charge significantly more than the baseline. Others, like Scarborough, are forced to charge less.
5. Hosts should be careful about who they respond to and how quickly they reply. Hosts that respond quickly tend to have slightly less valuable listings than those that wait a few days to respond. Responses within an hour, a few hours, and under a day performed at least 7% worse than those that wait a few days to respond.

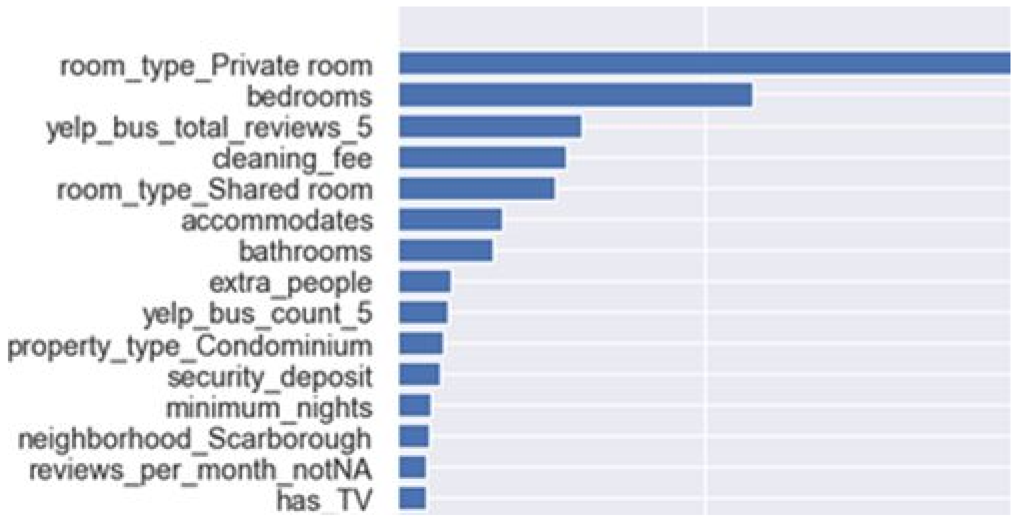
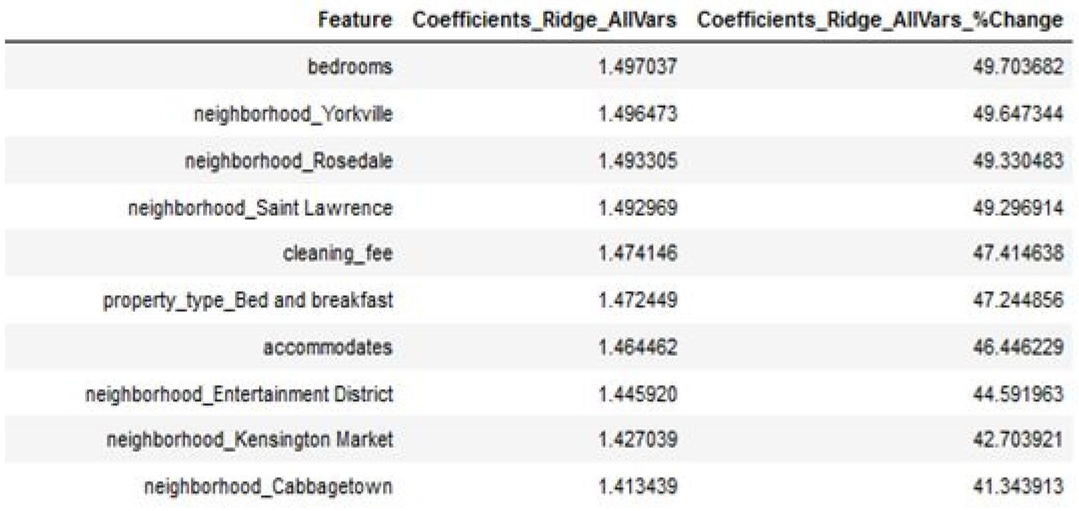
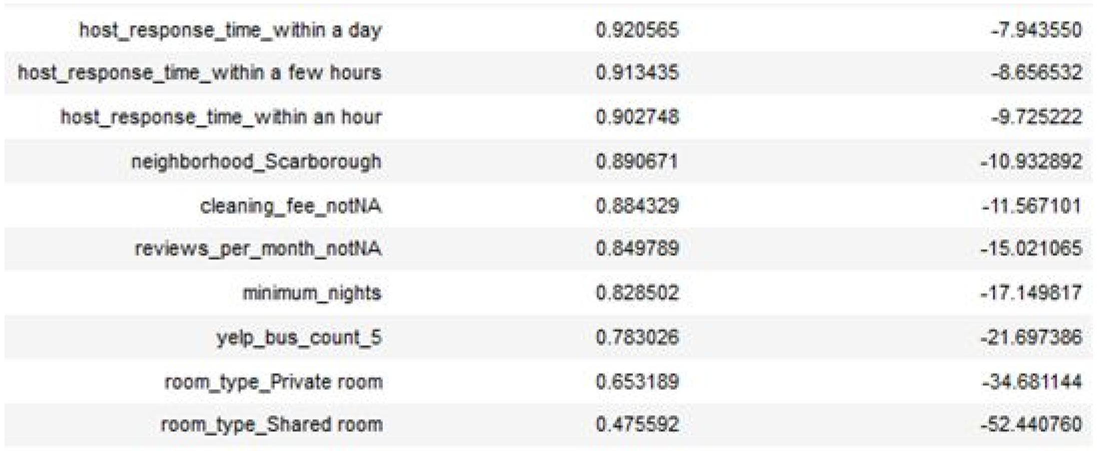


Figure 8 - Random Forest Feature Importances



Feature	Coefficients_Ridge_AllVars	Coefficients_Ridge_AllVars_%Change
bedrooms	1.497037	49.703682
neighborhood_Yorkville	1.496473	49.647344
neighborhood_Rosedale	1.493305	49.330483
neighborhood_Saint Lawrence	1.492969	49.296914
cleaning_fee	1.474146	47.414638
property_type_Bed and breakfast	1.472449	47.244856
accommodates	1.464462	46.446229
neighborhood_Entertainment District	1.445920	44.591963
neighborhood_Kensington Market	1.427039	42.703921
neighborhood_Cabbagetown	1.413439	41.343913

Table 4 - Positive Coefficients (Top 10), Ridge Regression Model



host_response_time_within a day	0.920565	-7.943550
host_response_time_within a few hours	0.913435	-8.656532
host_response_time_within an hour	0.902748	-9.725222
neighborhood_Scarborough	0.890671	-10.932892
cleaning_fee_notNA	0.884329	-11.567101
reviews_per_month_notNA	0.849789	-15.021065
minimum_nights	0.828502	-17.149817
yelp_bus_count_5	0.783026	-21.697386
room_type_Private room	0.653189	-34.681144
room_type_Shared room	0.475592	-52.440760

Table 5 - Negative Coefficients (Bottom 10), Ridge Regression Model

Suggestions for Improvement and Next Steps

I believe there is more improvement to be gained from both the Price and Utilization Rate models - with more time I would have tried using PCA and other feature creation methods, and removing variables that show signs of multi-collinearity.

I would also like to have tried other model types - perhaps a neural network or support vector machine could have shown improvement. For the Utilization Rate model, transforming it to Log of Utilization Rate might have shown an improvement like it did for Price.

Replacing Neighborhoods with a variable showing the average price for each neighborhood might lead to a reduction in overfitting. It would also allow the model to apply to cities other than Toronto (I would also need to remove the Yelp data since it is city-specific). Generalizing the model to lots of cities would allow us to use more of the data on Inside AirBnB, which could help improve the models.

Finally, incorporating the calendar data provided by Inside AirBnB would help with the Utilization Rate model - the calendar data shows the specific dates a listing was available along with the listed price at that date, instead of using an average price. The ability to track seasonal trends might lead to a better model.

In terms of next steps for the usage of these models, I recommend building an interactive dashboard where users can view all the listings in a selected city and view price predictions for them. This dashboard would also let the user enter details about their own home or apartment and receive a prediction about how they should price it in order to maximize utilization from guests. Such a dashboard would be the best way to deploy these models into production.