Capstone Project 2

Introduction:

For my second Capstone project, I chose to work with an Airbnb dataset for New York City, with the goal of developing a model that could predict listing price using the available variables. In addition to seeking to assess the drivers behind listing prices, my analysis and exploratory work could provide additional insight into the dataset. This analysis could also provide insight that could be used to inform about other city's Airbnb data.

Data:

Information on the columns is as follows:

- Id - listing ID
- Name - name of the listing
- Host_id - host ID
- Host_name - name of the host
- Neighbourhood_group - location, categorical
    - Manhattan
    - Brooklyn
    - Queens
    - Staten Island
    - Bronx
- Neighbourhood - area, categorical
    - 218 neighbourhoods
- Latitude - latitude coordinates
- Longitude - longitude coordinates
- Room_type - listing space type, categorical
    - Private room
    - Entire home / apartment
    - Shared room
- Price - price in dollars
- Minimum_nights - amount of nights minimum
- Number_of_reviews - number of reviews
- Last_review - latest review (date)
- Reviews_per_month - number of reviews per month
- Calculated_host_listings_count - amount of listing per host
- Availability_365 - number of days when listing is available for booking

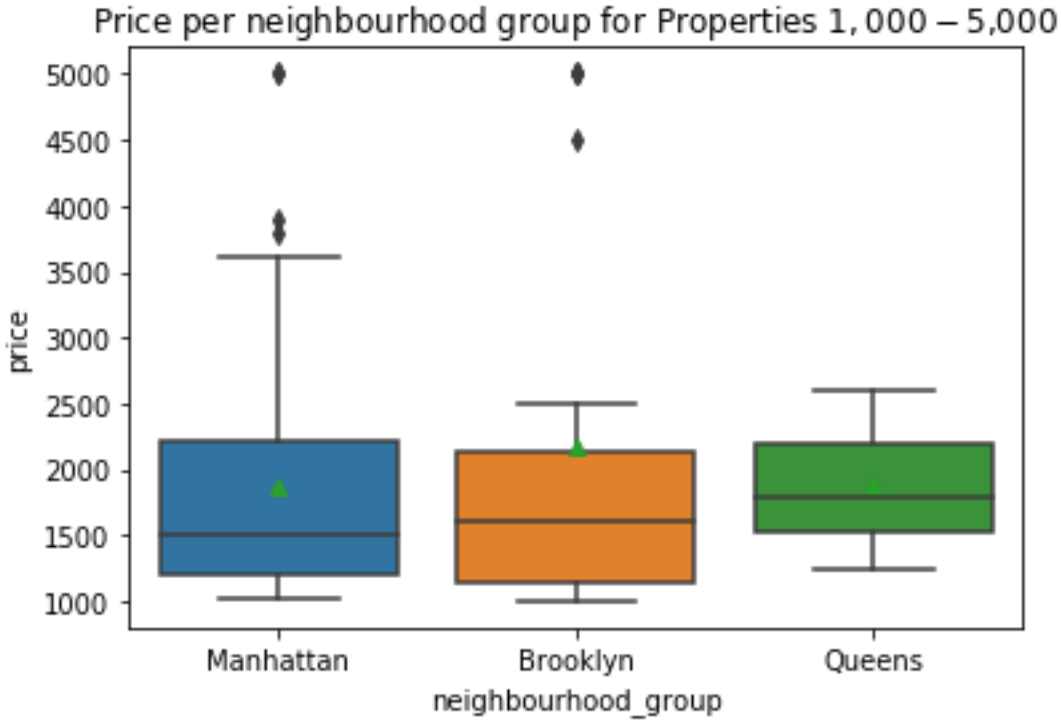
Exploratory and Visualizations:

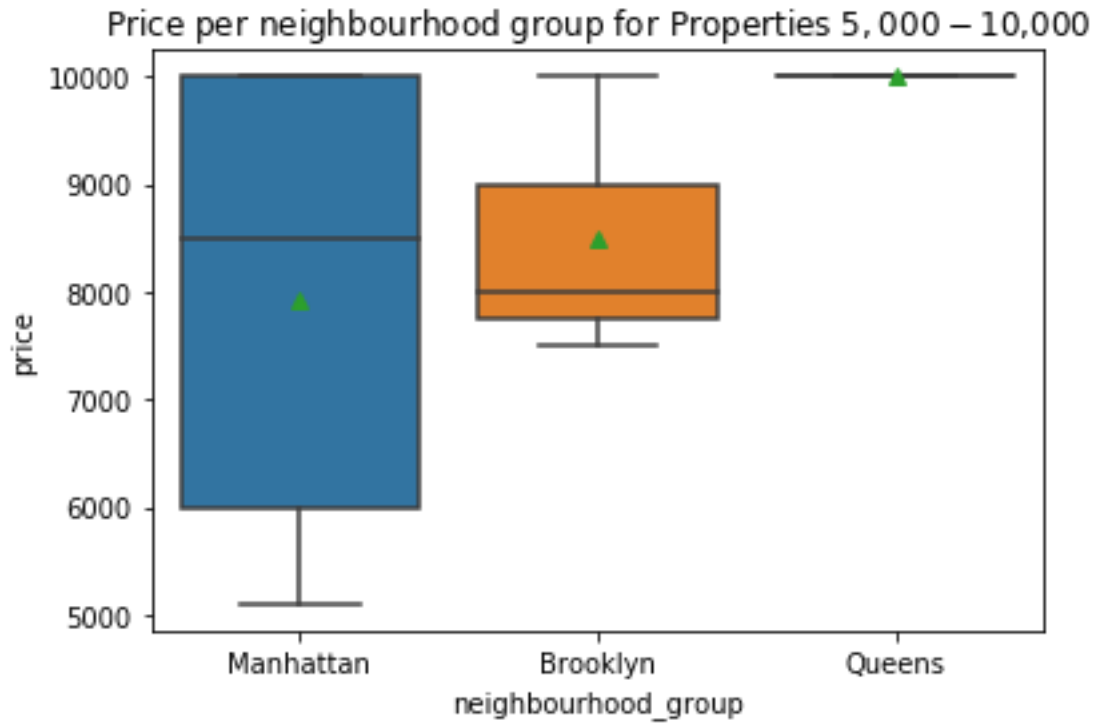
Price by neighbourhood group broken into 5 different price categories:


Price per neighbourhood group for Properties under $100
price
300
250
200
150
100
50
0
Brooklyn
Manhattan
Queens
Staten Island
Bronx
neighbourhood_group

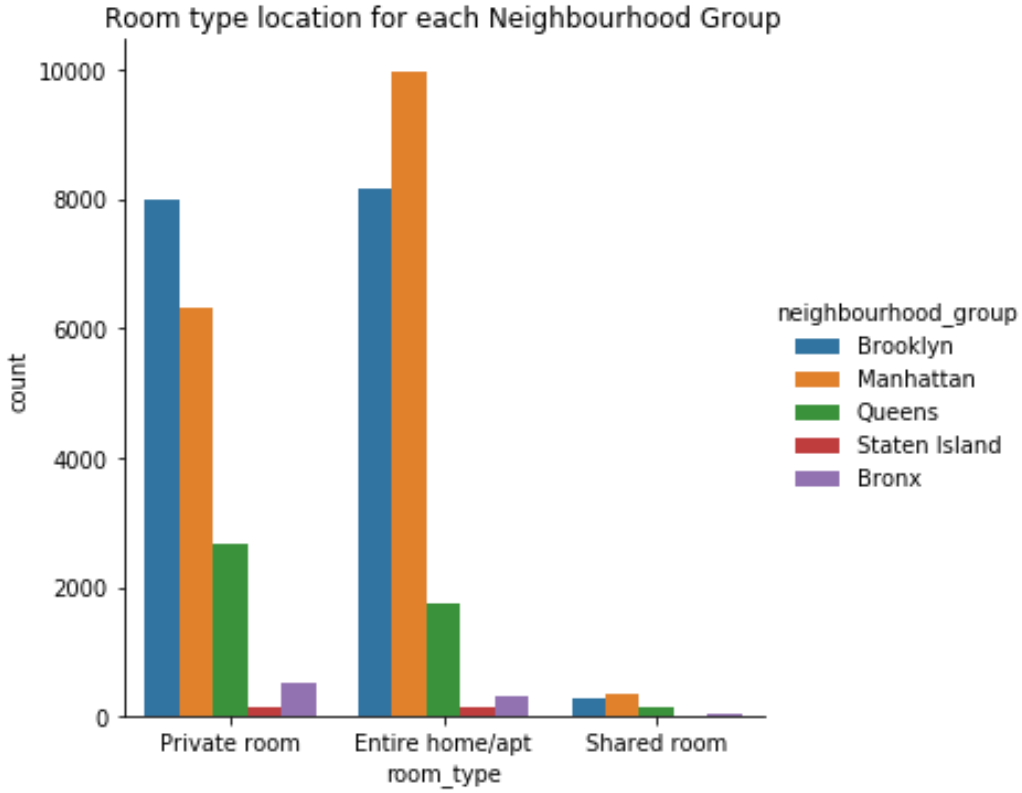

Price per neighbourhood group for Properties 100 − 500
price
500
450
400
350
300
250
200
150
100
Brooklyn
Manhattan
Queens
Bronx
Staten Island
neighbourhood_group


Price per neighbourhood group for Properties 500 − 1,000
1000
900
800
700
600
500
price
Brooklyn
Manhattan
Staten Island
Queens
Bronx
neighbourhood_group

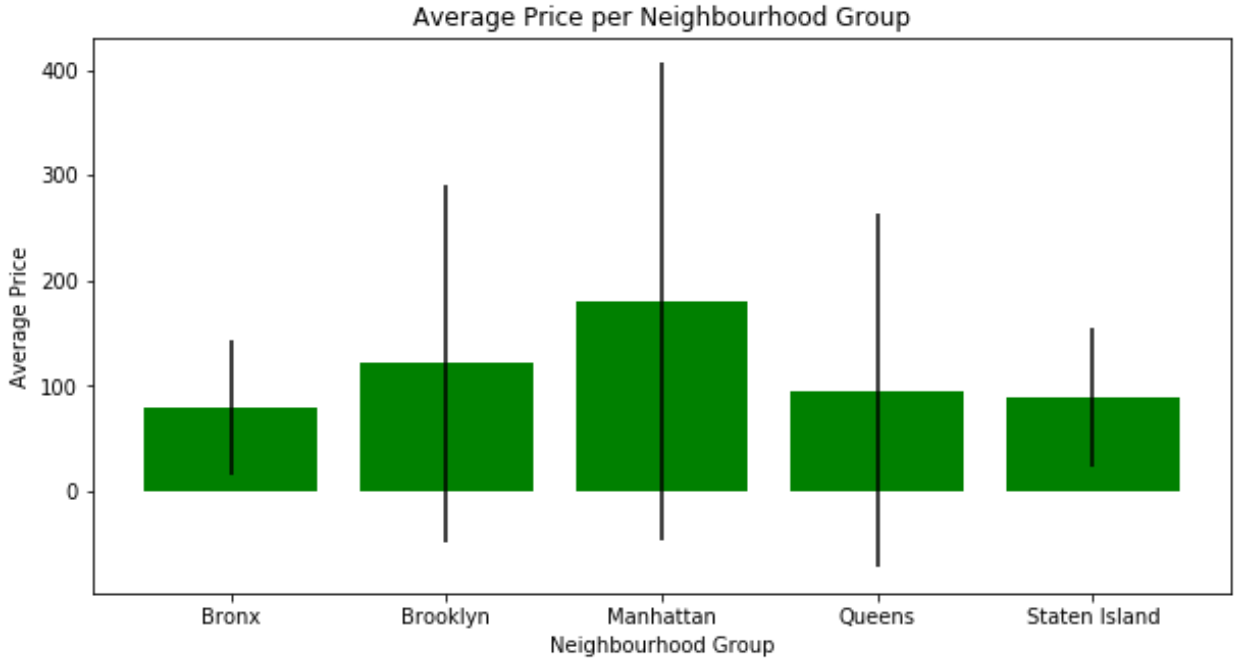

Price per neighbourhood group for Properties 1,000 − 5,000
5000
4500
4000
3500
3000
2500
2000
1500
1000
price
Manhattan
Brooklyn
Queens
neighbourhood_group

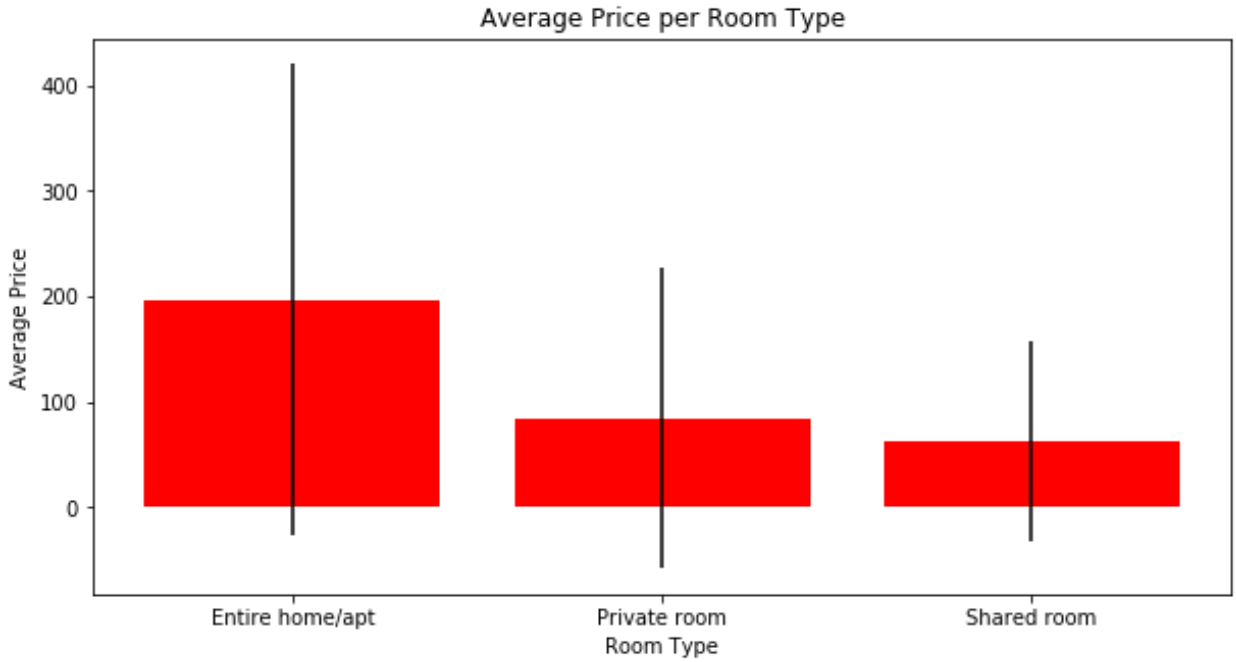


Counts of room type for each neighbourhood group

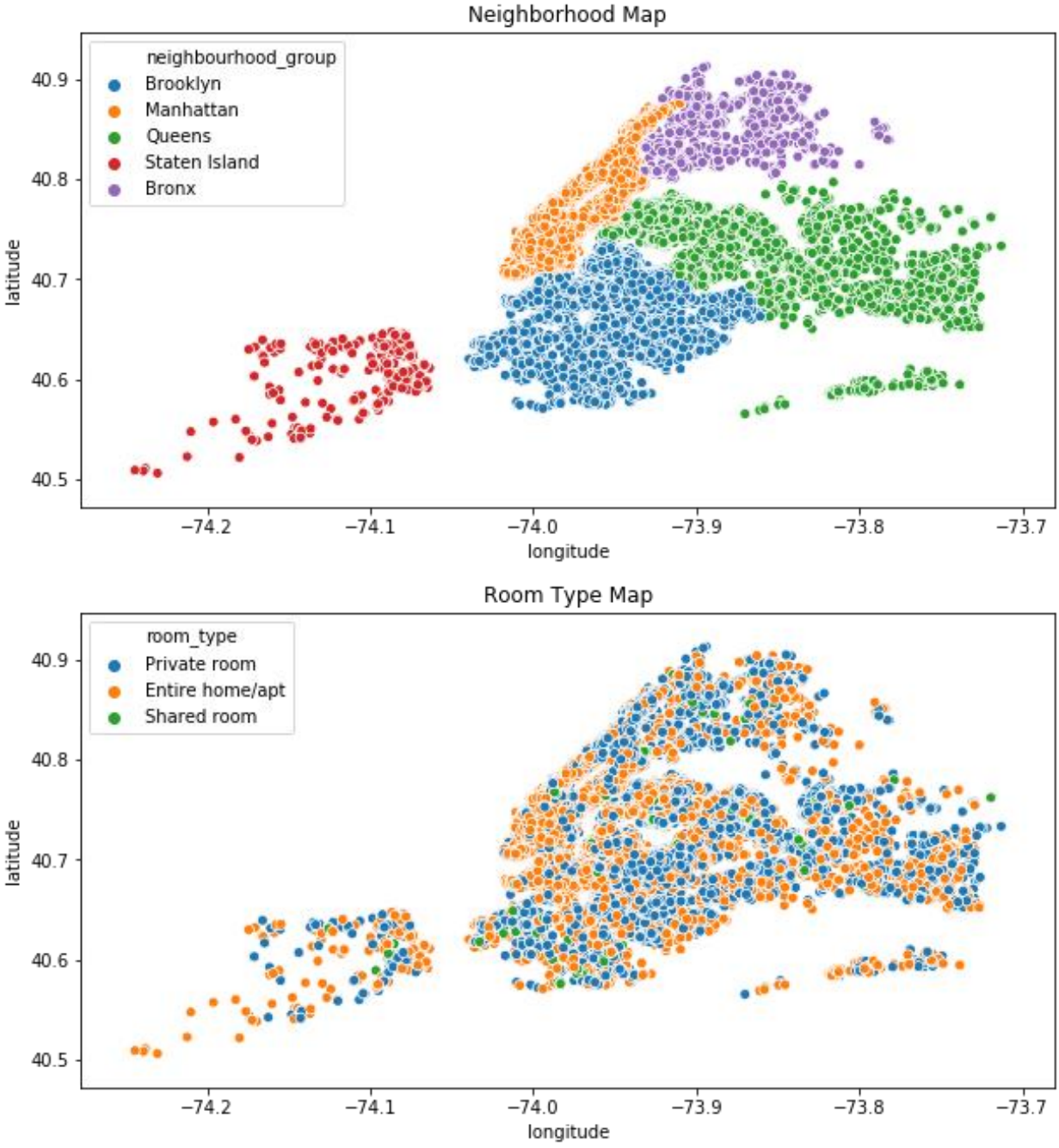


Average plots:





Neighborhood and Room Type Map Plots



Data Cleaning:

There were several steps necessary in order to clean and prepare the data for modeling. Some involved using logic about what makes sense for the data, and others involved transformation in order to get the data in a more usable form. The first cleaning steps were to remove NA values. Next, I removed listings where the price was $0, because this was likely a mistake. I then selected listings where the minimum nights variable was below one month. This decision was made on the assumption that a stay exceeding one month is likely not for a vacation and thus price will probably be more static. These changes impacted the row counts as follows:

- Raw uncleaned dataset: 48,895 rows
- NAs removed: 38,821 rows
- Listings with prices of $0 removed: 38,811 rows
- Listings with minimum nights below 31 nights: 38,511 rows

The next steps taken involved getting certain variables into a form where they could be used in a model. During the exploratory steps I found significant skew in several continuous variables. These variables are: price, minimum_nights, number_of_reviews, reviews_per_month, calculated_host_listings_count, availability_365. In order to improve the distributions of these variables, I added a very small number to all of them (so as to eliminate any cases where there is a 0) and then log transformed them. This resulted in improved distributions.

In order to include all variables in the analysis, I needed to determine dummy variables for the categorical variables: room_type, neighbourhood, and neighbourhood_group.

The last cleaning step was related to our predicted variable: price. Our plots and quantile information showed a large range and thus significant variance in the price. Further exploration revealed the following information:

- There are 19,388 listings below $100 a night.
- There are 18,811 listings between $100 and $500 a night.
- There are 515 listings between $500 and $1,000 a night.
- There are 98 listings between $1000 and $5,000 a night.
- There are 9 listings between $5,000 and $10,000 a night.

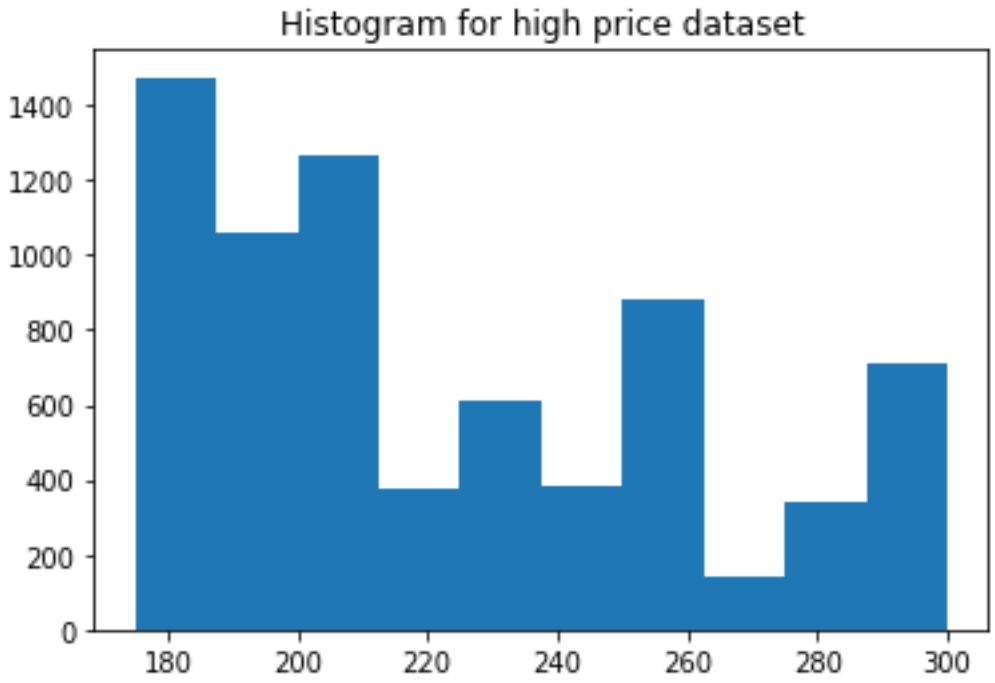
While I did run a regression model on the entire dataset, I found improvement when I broke the data into a high and low price range. I decided to eliminate any listings above $300 since they represent extreme outliers (top 5% of price) which will only confuse the model and weaken performance. The low and high ranges were separated by $175. For the classification models, greater than $175 was high and categorized as a 1, below $175 as a 0. This is because:

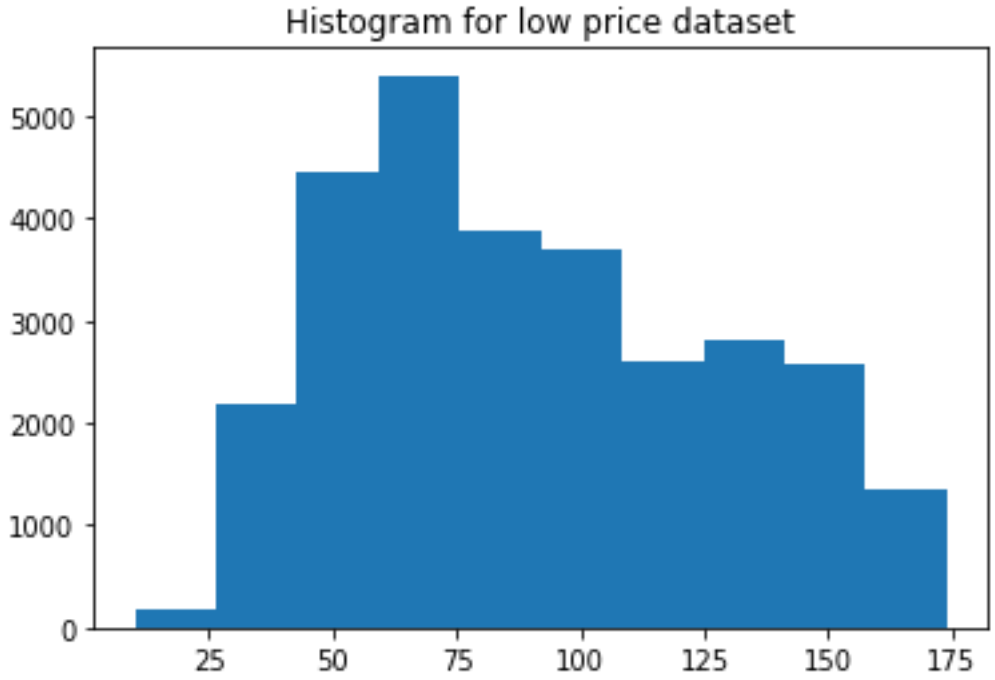
- There are 36,642 listings below $300 a night.
- There are 2,179 listings over $300 a night.
- 94.39% of listings are below $300 a night
- 5.61% of listings are above $300 a night

Further separation by the $175 breakdown shows:

- There are 29,112 listings below $175
- There are 7,236 listings at or above $175

The distributions are as follows:





Linear Model:

For the linear model, I used minimum_nights, number_of_reviews, reviews_per_month, calculated_host_listings_count, availability_365, room_type, neighbourhood_group and neighbourhood as the predictors for price. The continuous variables were log transformed to create a more normal distribution.

Although there seemed to be some correlation between number_of_reviews and reviews_per_month based on the correlation heat plot and the calculation of variance inflation factors, removal of one of these variables actually lowered the model's performance so I decided to keep all of these variables.

When this regression was run on the entire dataset, the adjusted R_2 was: 0.62. The other error metrics were as follows:

- Mean Absolute Error: 0.12
- Mean Squared Error: 0.02
- Root Mean Squared Error: 0.15
- Mean Average Percent Error: 0.06

When split into test and training sets, the model had these following metrics:

- Mean Absolute Error: 0.12
- Mean Squared Error: 0.02
- Root Mean Squared Error: 0.15

- Price mean: 116.84
- Price std: 64.34
- RMSE: 44.15
- R2 score train: 0.61
- R2 score test: 0.62

Overall, eliminating the top 5% of listings allowed for a model which accounted for a reasonable amount of variance. Additionally, we were able to maintain this on the test set which indicates our predictive power was maintained. Furthermore, our error metrics remained low, indicating overall, good model performance.

<u>Results for Linear Model looking at the lower price point:</u>

The adjusted R_2 was: 0.55. The other error metrics were as follows:

- Mean Absolute Error: 0.1
- Mean Squared Error: 0.02
- Root Mean Squared Error: 0.13
- Mean Average Percent Error: 0.05

When the model was split into test and training sets:

- Price mean: 90.82
- Price std: 37.47
- RMSE: 26.17
- R2 score train: 0.52
- R2 score test: 0.52

Our adjusted r_2 remained the same in the training and testing sets which indicates performance was maintained in our test set. Additionally, our RMSE was lower compared to the regression run on the full dataset which indicates that our predictions were better when we split the datasets up.

We also have a low Mean Average Percent Error which indicates that the model has accurate forecasting (it is also lower than the value we have for the model run on the full dataset).

Results for Linear Model looking at the higher price point:

The adjusted R_2 was: 0.05. The other error metrics were as follows:

- Mean Absolute Error: 0.06
- Mean Squared Error: 0.0
- Root Mean Squared Error: 0.07
- Mean Average Percent Error: 0.03

When the model was split into test and training sets:

- Price mean: 221.54
- Price std: 38.23
- RMSE: 2391075014304.85
- R2 score train: 0.05
- R2 score test: -1.111333111036452e+27

Modeling the higher price dataset showed a few challenges. One, the overall r_2 was lower, but more noticably, the RMSE and the test r_2 took on extreme values, this was due to the alleged presence of an NaN or Inf value. Upon close inspection there were no infinity or NaN values in the original dataframe, or unexpected / strange values for any of the predictor variables. I took many efforts to understand what was going on in the dataset to see where these extreme prediction or actual values could be coming from, and ultimately, after seeking advice from two mentors, I decided to proceed with a random forest multiple regression as it is typically a more robust model to these values.

The random forest multiple regressions results were:

- Price mean: 221.54
- Price std: 38.23
- RMSE: 37.53
- R2 score train: 0.17
- R2 score test: 0.03

Overall, this model is much better as the r_2 took on improved values and the RMSE was a realistic number. The test dataset did not fair as well as the training set, however this could be explained by the less normal distribution of the higher price dataset and the lower size of the dataset to sample from. In general, the higher price point had an overall poorer performance. This is likely because, even after transforming the data, the distribution was not completely normal. While perhaps this data could be transformed or segmented further, ultimately this also will impact the integrity of the data. The optimal case is more data for this range of prices, or more powerful and informative predictors.

Classification:

The last step taken in the analysis was to create a binary category: 1, 0 for whether the listing is expensive or not (> $175, 0 or <= $175 , 1).

I wanted to try a classification model to see if there was another way of predicting listing price based on an expensive or inexpensive category. I felt the best way to model this was by using a logistic regression model which is good for modeling binary outcomes.

After splitting the data into train and test sets, I got the following results:

The classifier was 84% accurate.

The classification report output was:

	precision	recall	f1-score	support
0.0	0.59	0.37	0.45	1,971
1.0	0.87	0.94	0.91	8,934
accuracy			0.84	10,905
macro avg	0.73	0.66	0.68	10,905
weighted avg	0.82	0.84	0.82	10,905

Below I will break this down into the specific metrics we want to consider and what this means for the model:

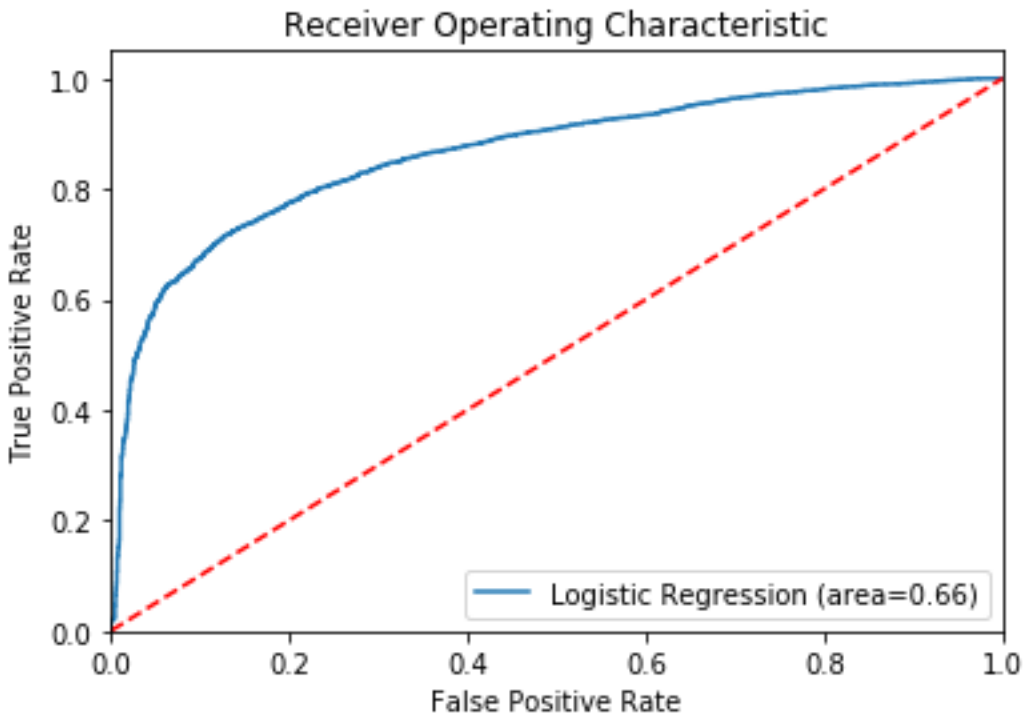
1. Precision score: 0.87
    a. Defined as the ratio of true positives to the sum of true and false positives. 87% of our predictions were correct.
2. Recall score: 0.94

. Defined as the ratio of true positives to the sum of true positives and false negatives. This model caught 94% of the positive cases.

3. F1 score: 0.91:

. Although this metric is best for comparing classification models, as it gives us the percent of correct positive predictions, I am including it because at 91%, our baseline is quite high for correct positive predictions.

Another metric evaluated was ROC AUC. Our AUC was 0.66, which indicates that the model is better than random chance, although perhaps with better predictors this could be improved.



Next steps:

Next steps for this analysis include: gathering more “high priced” listings to have a better distribution, or at least more data for that group of data. Extraction of more features potentially through natural language processing of the listing description, or Airbnb could provide more data for the listings such as the rating for the host. For the logistic regression parameter tuning and perhaps recursive feature elimination could have helped in improving the predictive accuracy of the model, although it already is performing fairly well. To combat uneven sample sizes upsampling of the minority class could potentially help.