David Aaron, Connor Lydon, Angela Zhang
Professor Hersh
MGSC 310-2
12 / 18 / 2020

AirBnb Price Prediction

Many property owners are often stumped when they need to enter in a price for their property when listing it on Airbnb. It's difficult to decide how much they should charge per night for the property. Some owners list their property at a similar price to neighbors and others list their property with the desired profit. Since every property is unique, the price cannot be determined by just comparing it to your neighbors arbitrarily. There needs to be a way to offer these property owners a suggested price to help them maximize their profit through the characteristics of their house. Our group wants to build a pricing tool to help property owners decide the price that they should list their property based on the features of their house and other pieces of data on the house. Another audience for our pricing tool is real estate developers that want to invest in property to list them on Airbnb. We will be able to inform them of the most important characteristics that the most profitable properties have and the most profitable locations in Hawaii.

We chose to analyze an Airbnb dataset for listings in the state of Hawaii; the features included in the dataset describe various attributes about each Airbnb listing. The main variables we plan on using to predict the room price for a given listing include: neighborhood, latitude, longitude, minimum number of nights, room type, number of reviews, reviews per month, calculated host listings, and the availability of the listing out of 365 days. Using this dataset, we aim to predict the nightly rate of a given listing using three different models and conclude on the best model to use for the dataset. We will be analyzing a linear model, an elastic net model, a lasso model, and a random forest, which determines if a property is luxury, to predict the nightly rate of a given listing. The

ability to predict the nightly rate of an Airbnb listing helps people realize the expected price of a property.

Cleaning:

Exact details of how it was cleaned and where the data came from can be found in the cleaning rmd file. This data was very dirty. The goal with cleaning this dataset, or any dataset for that matter, is to remove outliers and unusable values while still maintaining a substantive and representative set to build predictive models from. Because it wasn't precleaned and it was just scraped there are a lot of imperfections. These unusual attributes included blank columns, listing/user identifier numbers, inaccurate date-time data, and URLs. There were some other columns that consisted of vectors of characters. One was amenities, which a user could select certain attributes as well as add others. Another column was property type, where the user could enter a type or pick from a set. You can't do sentiment on this and it isn't discrete. One property was a lighthouse with a boat! This would make it impossible to analyze because the components of the vectors are not discrete. Even if they were discrete, the sparse matrices would be huge and inefficient to work with. Since this data was scraped there were a lot of columns that had a majority of them blank. The biggest set of them related to review scores. None of the data came with documentation, so if the data is null it might be suspected that the property has no reviews. After the diagnosis we removed columns that had more than 10% NAs, which includes the review columns. We could have also removed the rows for the review related variables which were NA and do an analysis that way, but it would be a large chunk of data. We also did a sentiment score which was surprisingly useful in the final analysis.

The last important piece of cleaning that we did is that we removed really high prices. We arbitrarily set this to $1000. We did this because a mansion in the country can't be valued the same

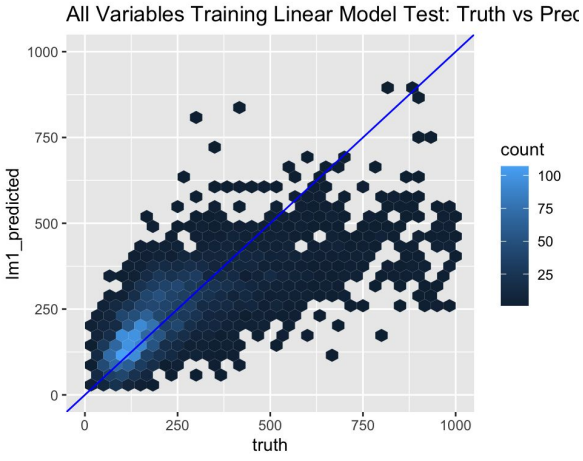
as a studio in the city, just as wall art at target can't be valued the same as art from a famous artist like Banksy. To also help the linear models work with the exponential pattern seen in price we made a log price variable. Just looking at our data, a price of $1000 is still an outlier. If we didn't do this we'd see extreme seed to seed variance. For example, there was a property that was $25k a night and skewed the data a lot. Even though the prices we removed were over $1000, a little over 600 rows were removed. This improved our models from around 0.1, being highly variable seed to seed, to r^2 to around 0.4 - 0.5 r^2.

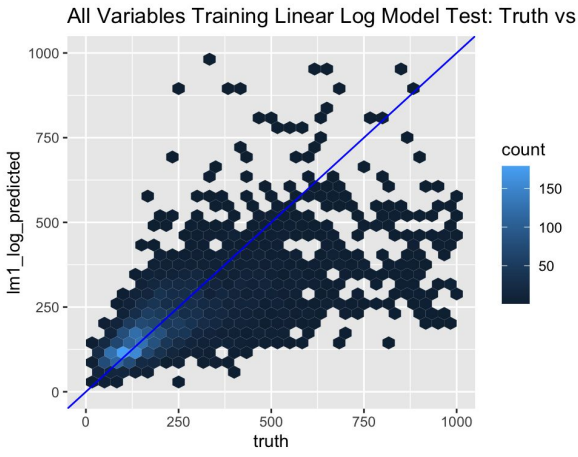
Summary Statistics:

See last page for summary statistics

Model 1: Linear and Logged Linear Model

Starting off, we chose to use all attributes except for price as our dependent variables. Doing so yielded promising results. With a training R-Squared of ≈0.489 and a testing R-Squared of ≈0.491. The error looks really good. Plotting the model predictions versus the actual test values revealed that the linear model could yet be tuned. The model is obviously underpredicting the





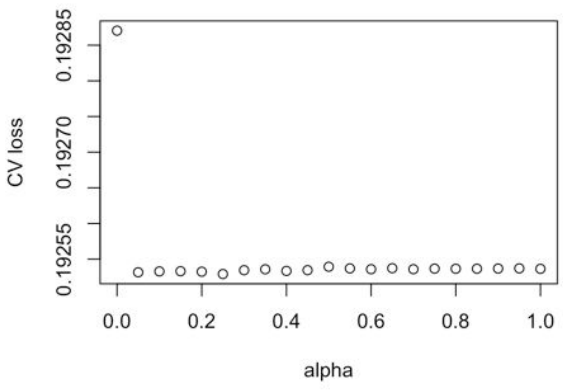
expensive properties. The reason the error rates look so good is likely due to the amount of data centered on the bottom left. Although the current plot shows the mostly-linear relationship between the predictions and actual values, the underpredictions lead us to believe that using the log() of price

would reveal a more representative model. If just looking at predictions for expensive properties, the model is very biased. It doesn't seem like a truly linear model predicts the true pattern of the data.

Turning the target variable (price) into log(price) will help with the predictions of expensive properties. Logging the variable linearizes an exponential pattern. Creating the log(price) model and exponentiating the predictions reveals the opposite results - mean squared error increased for both training and testing data and our R-squared value decreased for both as well. When looking at the truth vs predicted plot, the predictions for expensive properties are better, but the spread in the variables is higher. This is a beneficial trade off because it more accurately portrays the true relationship in the data even though the variance is higher and being overfit.

Model 2: Elastic Net Model & Lasso Model

The elastic net model is a way to regularize the data from overfitting. When we look at the linear log model, it looks a bit overfitted. When first initializing the model we use 20 alpha levels iterating from 0 to 1; this goes from a ridge, minimizing variables, to lasso regression, zeroing variables. The plot to the left shows the error rate as the alpha value changes. The lowest loss was found at 0.05, which is nearly a ridge model. With 10 iterations this isn't found. This is likely that some single variable, even minimized, causes the loss to go higher, so when the alpha goes above 0 it allows for greater reduction of variables. We used lambda.1se to achieve some regularization and it reduced the number of variables from 51 to 41.This elastic net model yielded us slightly better testing error rates than the regular linear log model while retaining nearly the same training error rates.

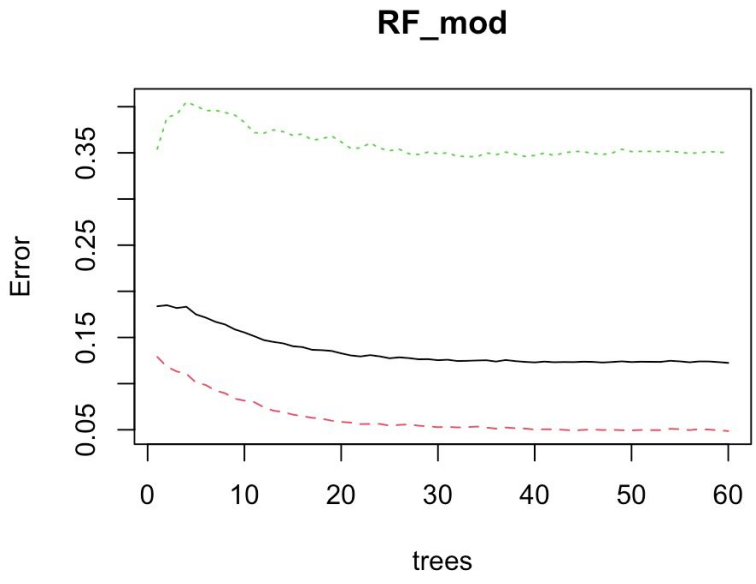


The second ‘elastic net’ model we did was a lasso model. Overall looking at the ‘minlossplot’ the error rates from 0.05 and 1 were very similar, slightly higher at alpha of 1. When we made our model for lasso the variables go from 51 to 36 at lambda 1se. Not much more than the elastic net model, but it is still a greater reduction in variables. The error rates are slightly better than the linear log model. Though the training error is worse than the linear log training, the testing does a better job. We also tried using lambda.min, just to get some variable reduction and the testing error is higher than the model at lambda 1se.

Both these models seem to reduce the variance that is seen in a regular linear_log model. Though both these models are similar the lasso is superior because it has better test error rates and has less variables to interpret.

Model 3: Random Forest

To build the random forest model, the dependent variable of price needed to be binary. Our group decided that we would create a new variable for price that determines if a listing is luxury or not. Based on the last quartile in the summary statistics the price for luxury listings would be $285. Luxury would mean that the property is more expensive than 75% of the listings and non luxury would mean that the property is less expensive than 25% of the listings. The variables price and log price is taken out of the random forest model because they would cause collinearity with the luxury listings variable as they are highly correlated with each other.



We converted the variable luxury to factor variables to set the variable to categorical instead of numeric.

Our group generated a random forest model to determine the number of trees needed. The model includes the dependent variable luxury against all independent variables and we used listings train as our data. We experimented with 200 trees to start and we noticed that the error levelled off at around 60 trees. Finally, we ran the model again at 60 trees and set mtry to the square root of the number of variables (40).

Next we determined which variables would affect whether the listing is luxury or non luxury the most and increase the error if they were removed. We plotted the variable importance plot showing the mean decrease accuracy on the x axis and independent variables on the y axis. The most important variable based on the random forest model is the number of bathrooms. When the variable bathrooms is removed from the random forest model, the error would increase by 25.09%. The second most important variable is ‘accommodates; and removing the variable from the model would lead to an increase in error by 18.74%. The top five most important variables in order would be the number of bathrooms, number of accommodations, neighbourhood group, number of reviews and yearly availability.

Our group sought to make predictions based on the random forest model to determine how well the model performs. We generated confusion matrices for predictions against the truth. The random forest mode’s specificity is 87.72% for testing data. The random forest model is able to predict accurately the non luxury listings 87.72% of the time for testing data. The random forest model’s sensitivity is 88.73% for testing data. This means that the model is able to accurately predict luxury property 88.73% of the time for testing data. This model performs very well in predicting non luxury and luxury listings.

Future changes:

Conclusion:

After familiarizing ourselves with the Airbnb dataset, cleaning the dataset and making three different models, our team came to the conclusion that a linear model yielded the most predictive power out of all of our models for lower priced listings while our random forest model performed well with predicting luxury properties. Although the linear model displays a higher predictive power overall, this model is biased, it doesn't predict for the true pattern in the data. The size of our testing and training data the model's accuracy is limited when moving past a price of 250. Even though these models are usable, because of the inconsistent nature of our data, we could tune our models. Working with the data itself, rather than the models, by doing things such as k-fold cross validation and further cleaning are ultimately the most promising measures to be taken in the future. For example, manipulating the days that are available and making a supermetric from that.

So, which model is best overall? Logged lasso. The logged lasso model is the best model because it helps reduce the bias in the outliers and helps reduce the variance when it is logged. The lasso model is holistically a better predictor because it has better predictions through all of the data overall. Even though the price is logged, the model still underpredicted at higher prices, a potential solution to this could be to log it with a higher base. Another way to deal with this problem is to bootstrap data equally to all of the price ranges across all the data. If you want to predict with prices less than 250 use the regular linear model. The bias in this model is minimal when used in prices less than 250.

Summary statistics.

```
host_response_time host_response_rate host_acceptance_rate host_is_superhost host_listings_count host_has_profile_pic host_identity_verified
Length:19750       Mode:logical       Min.   :  0          Mode :logical     Min.   :   0.00     Mode :logical        Mode :logical
Class :character   NA's:19750         1st Qu.: 89          FALSE:12047       1st Qu.:   2.00     FALSE:52             FALSE:3989
Mode  :character                      Median : 99          TRUE :7703        Median :  10.00     TRUE :19698          TRUE :15761
                                      Mean   : 90                            Mean   :  72.88
                                      3rd Qu.:100                            3rd Qu.:  89.00
                                      Max.   :100                            Max.   :2143.00

neighbourhood_group_cleansed  room_type          accommodates    bathrooms       bedrooms          beds            price       minimum_nights
Length:19750                  Length:19750       Min.   : 0.0   Min.   :0.000   Min.   : 1.00   Min.   : 0.000   Min.   : 10.0   Min.   :  1.00
Class :character              Class :character   1st Qu.: 3.0   1st Qu.:1.000   1st Qu.: 1.00   1st Qu.: 1.000   1st Qu.:120.0   1st Qu.:  1.00
Mode  :character              Mode  :character   Median : 4.0   Median :1.000   Median : 1.00   Median : 2.000   Median :179.0   Median :  3.00
                                                 Mean   : 4.5   Mean   :1.552   Mean   : 1.71   Mean   : 2.331   Mean   :230.5   Mean   :  5.63
                                                 3rd Qu.: 6.0   3rd Qu.:2.000   3rd Qu.: 2.00   3rd Qu.: 3.000   3rd Qu.:285.0   3rd Qu.:  5.00
                                                 Max.   :16.0   Max.   :9.000   Max.   :11.00   Max.   :17.000   Max.   :999.0   Max.   :200.00
                                                                                NA's   :2802
 host_response_time host_response_rate host_acceptance_rate host_is_superhost host_listings_count host_has_profile_pic host_identity_verified
 Length:19750       Mode:logical       Min.   :  0          Mode :logical     Min.   :   0.00     Mode :logical        Mode :logical
 Class :character   NA's:19750         1st Qu.: 89          FALSE:12047       1st Qu.:   2.00     FALSE:52             FALSE:3989
 Mode  :character                      Median : 99          TRUE :7703        Median :  10.00     TRUE :19698          TRUE :15761
                                       Mean   : 90                            Mean   :  72.88
                                       3rd Qu.:100                            3rd Qu.:  89.00
                                       Max.   :100                            Max.   :2143.00

maximum_nights_avg_ntm has_availability availability_30 availability_60 availability_90 availability_365 number_of_reviews number_of_reviews_ltm
Min.   :   1.0         Mode :logical    Min.   : 0.00   Min.   : 0.00   Min.   : 0.0    Min.   :  0.0    Min.   :  0.00    Min.   :  0.000
1st Qu.:  90.0         FALSE:1          1st Qu.: 0.00   1st Qu.: 1.00   1st Qu.: 3.0    1st Qu.: 48.0    1st Qu.:  1.00    1st Qu.:  0.000
Median :1125.0         TRUE :19749      Median :11.00   Median :27.00   Median :42.0    Median :238.0    Median :  7.00    Median :  2.000
Mean   : 719.6                          Mean   :12.85   Mean   :27.06   Mean   :40.5    Mean   :195.8    Mean   : 27.51    Mean   :  5.402
3rd Qu.:1125.0                          3rd Qu.:26.00   3rd Qu.:49.00   3rd Qu.:69.0    3rd Qu.:310.0    3rd Qu.: 33.00    3rd Qu.:  7.000
Max.   :9978.6                          Max.   :30.00   Max.   :60.00   Max.   :90.0    Max.   :365.0    Max.   :739.00    Max.   :151.000

number_of_reviews_l30d review_scores_rating review_scores_accuracy review_scores_cleanliness review_scores_checkin review_scores_communication
Min.   : 0.00000       Min.   : 20.00       Min.   : 2.000         Min.   : 2.000            Min.   : 2.000        Min.   : 2.000
1st Qu.: 0.00000       1st Qu.: 93.00       1st Qu.:10.000         1st Qu.: 9.000            1st Qu.:10.000        1st Qu.:10.000
Median : 0.00000       Median : 97.00       Median :10.000         Median :10.000            Median :10.000        Median :10.000
Mean   : 0.07266       Mean   : 94.89       Mean   : 9.693         Mean   : 9.531            Mean   : 9.789        Mean   : 9.749
3rd Qu.: 0.00000       3rd Qu.:100.00       3rd Qu.:10.000         3rd Qu.:10.000            3rd Qu.:10.000        3rd Qu.:10.000
Max.   :17.00000       Max.   :100.00       Max.   :10.000         Max.   :10.000            Max.   :10.000        Max.   :10.000
                       NA's   :4683         NA's   :4689           NA's   :4689              NA's   :4693          NA's   :4690
review_scores_location review_scores_value instant_bookable calculated_host_listings_count calculated_host_listings_count_entire_homes
Min.   : 2.000         Min.   : 2.000      Mode :logical    Min.   :  1.00                 Min.   :  0
1st Qu.:10.000         1st Qu.: 9.000      FALSE:6898       1st Qu.:  2.00                 1st Qu.:  1
Median :10.000         Median :10.000      TRUE :12852      Median :  9.00                 Median :  7
Mean   : 9.833         Mean   : 9.412                       Mean   : 46.95                 Mean   : 46
3rd Qu.:10.000         3rd Qu.:10.000                       3rd Qu.: 64.00                 3rd Qu.: 64
Max.   :10.000         Max.   :10.000                       Max.   :279.00                 Max.   :279
NA's   :4693           NA's   :4694
calculated_host_listings_count_private_rooms calculated_host_listings_count_shared_rooms reviews_per_month host_since_days   log_price
Min.   : 0.0000                              Min.   :0.00000                             Min.   : 0.010    Min.   :  60    Min.   :2.303
1st Qu.: 0.0000                              1st Qu.:0.00000                             1st Qu.: 0.190    1st Qu.:1252    1st Qu.:4.787
Median : 0.0000                              Median :0.00000                             Median : 0.540    Median :1697    Median :5.187
Mean   : 0.7518                              Mean   :0.01934                             Mean   : 0.968    Mean   :1789    Mean   :5.222
3rd Qu.: 0.0000                              3rd Qu.:0.00000                             3rd Qu.: 1.380    3rd Qu.:2267    3rd Qu.:5.652
Max.   :40.0000                              Max.   :6.00000                             Max.   :13.140    Max.   :4532    Max.   :6.907
                                                                                         NA's   :4568
name_sentiment    description_sentiment neighborhood_overview_sentiment host_about_sentiment
Min.   :-0.5774   Min.   :-0.4914       Min.   :-0.8000                 Min.   :-1.0000
1st Qu.: 0.0000   1st Qu.: 0.1790       1st Qu.: 0.0000                 1st Qu.: 0.0000
Median : 0.1508   Median : 0.2538       Median : 0.0442                 Median : 0.2582
Mean   : 0.1868   Mean   : 0.2711       Mean   : 0.1290                 Mean   : 0.2409
3rd Qu.: 0.3182   3rd Qu.: 0.3411       3rd Qu.: 0.2282                 3rd Qu.: 0.3741
Max.   : 1.4697   Max.   : 1.6730       Max.   : 1.9923                 Max.   : 1.6604
```