Daniel Lupercio
STAT 724
Professor Iordan Slavov
December 20, 2021

Analysis of NYC AirBnb Listings

1. Introduction

Airbnb is an online marketplace that connects people who want to rent out their homes/property, with people who are looking for room or apartment accommodations in a listings area [1]. People who are renting out their apartment are considered hosts. For the hosts, listing their apartment is a way to earn passive income. For the guests, there are a plethora of inexpensive accommodations throughout a city. Guests are liable for extensive damage(s) done to the property. The risk for the guests can be that the listing does not live up to the standard as advertised.

The data collected for this project can be found on insidearbnb.com. This “Inside Airbnb” project was brought to the public by Murray Cox and John Morris [2]. The creators of this website wanted to shed light on the issue of housing inequality throughout New York City. They allege that Airbnb listings are contributing to the gentrification of low-income neighborhoods. Which is not only an issue in NYC, but in other major cities such as San Francisco.

The data used in this project are from NYC listings in the last decade. The dataset originally begins with 61 variables, describing aspects of the listings posted on the Airbnb website. This paper aims to assess how well different machine models can perform when predicting the price of a listing. Described below are five fields found in the data frame:

Field	Type	Description
host_listing_count	Text	The number of listings the host has (per Airbnb calculations)
host_identity_verified	Boolean	Has the host's identity been verified by Airbnb. True or false values
neighbourhood_cleansed	Text	In the NYC data, these are the five boroughs
room_type	Text	All homes are grouped into the following four room types: Entire place, Private room, Shared room, Hotel room
bedrooms	Integer	The number of bedrooms in that listing

Table 1: Five fields provided by the Inside Airbnb Data Dictionary [3].

1.1 Data Cleaning

Forty-six fields were kept from the original dataset provided on the inside-Airbnb website. From these 46 fields, several dummy variables were created. For example, the "room_type" field was transformed into three distinct dummy variables. The "host_response_rate" and "host_acceptance_rate" fields had to be stripped of their "%" symbol. These values were then converted to floats and divided by 100. Missing values in this data frame also had to be imputed with the pandas "fillna" function, using "method = 'ffill.'" Scikit-learn does not allow for the data frames of models to contain NaN values, so the data frame was then imputed with mean values. With the values of the fields varying too much from each other, it was decided to scale this data frame. The MinMaxScaler() python function was used.

1.2 Exploratory Visualizations

Figure 1-3: Histogram of types of rooms from listings found throughout Manhattan, Brooklyn, and Queens.

2. Regression methods

2.1 Train Test Split

After the data set has been cleaned and prepped for machine learning algorithms, we will work with a total of 15,991 listings. Each listing has 42 fields of information. The dataset is then split into a training and testing portion. We will begin by exploring machine learning regression methods with the 'price' of the listing being the response variable.

Using the scikit-learn train_test_split() function, with a test_size = 0.3. We are returned an X_train data frame shape of 11,193 rows, with 41 fields. We are returned an X_test data frame shape of 4,798 rows, with the same number of fields. We can say something similar for the y_train and y_test data frames, except with the 'price' as the only field.

2.2 Multiple Linear Regression

Sklearn's LinearRegression() model is used to fit a model where the 'price' of Airbnb listings from the y_train data frame, are regressed onto the fields in the X_train data frame. Sklearn's recursive feature elimination (RFE) function is used to provide additional analysis to this model.

We begin with the estimator for the RFE() function, this being the linear model initiated with our training data frames. The arbitrary number of features to select initially was 20. The metric used to evaluate this model was the correlation of coefficient (R^2). We are returned $R^2 = 0.282$, when finding the correlation between the y_train observations, and the predictions of this linear model. This is not as optimal as I hoped, telling me that there is little correlation with 'price' and the predictors.

These results encouraged me to increase the number of features to train the RFE() model. The number of features increased to 25. Following the same steps, we are returned an R^2 value = 0.292. Note that this is not much of an increase from the initial model with 20 features.

2.2.1 Implement K-Fold Cross Validation

With the results of the correlation of coefficient being low, I began to implement K-fold cv with my training dataset. With k = 10, each fold is then used once as a validation while the 9 remaining folds form the training set. I continue to train a multiple linear regression model with the training sets. I make use of the parameters in the KFold() function, with "shuffle = True," and "random_state = 100."

fold	1	2	3	4	5	6	7	8	9	10
R^2	0.259	0.306	0.065	0.258	0.272	0.304	0.324	0.332	0.202	0.212
MSE	0.0011	0.0004	0.0002	0.0002	0.0002	0.0002	0.0001	0.0004	0.0002	0.0005

Table 2: Correlation of coefficient and mean squared error scores for each fold.

Table 2 returns two metrics for each of the k = 10 folds. We see that the highest R^2 is provided by fold #7, while also returning the lowest MSE value. All the MSE values can be more accurate by increasing the number of significant digits. However, these low MSE values leads me to believe that these models are over fitted.

2.2.2 Implementing Grid Search CV

I was not convinced yet with the optimal choice of predictors needed to get a good enough R^2 value. That is why I relied on implementing the grid search cv method. Using the initial RFE() method, along with the K-fold method, with k = 10. We fit 10 folds for each of the 41 candidates, totaling 410 fits. The R^2 is used once again to evaluate the model.

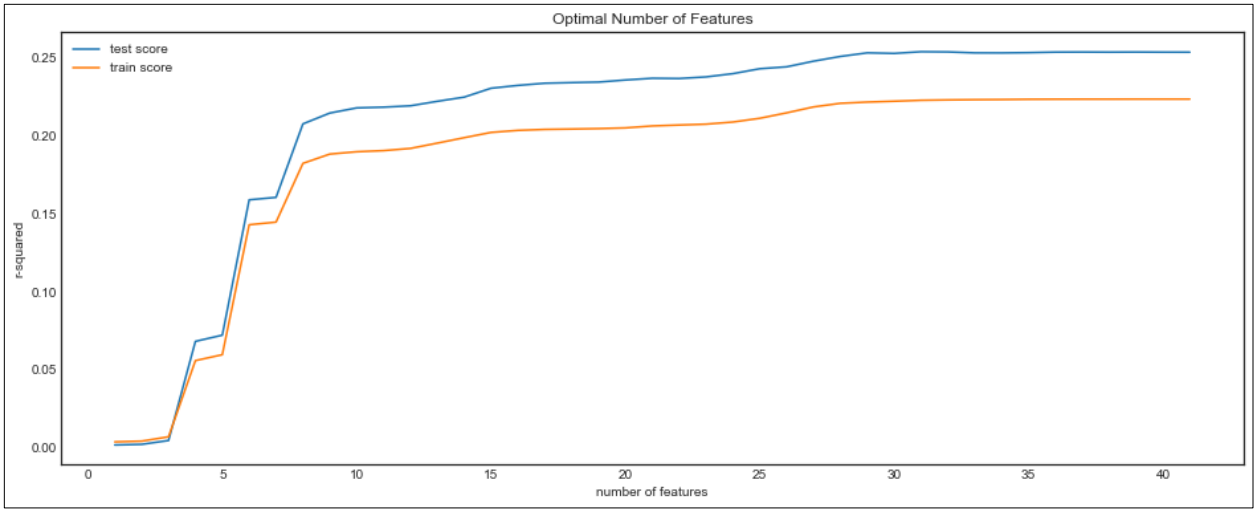

Figure 4: R^2 when compared to the number of features.

Converting the grid search cv results to a data frame allows me to plot figure 4. We see that both the train and test R^2 score stabilize after 26 features. Using the RFE() method one last time, and implying that 26 is the optimal number of features to select, a linear model is fit. We are returned $R^2 = 0.299$. With a low R^2 once again, linear regression is a complicated method to use to predict the 'prices' of listings. Other methods will be considered.

2.3 Ridge Regression

Ridge regression gives us the opportunity to conduct linear least squares, with L2 regularization. The goal is to minimize the object function: $\left||X\beta - y|\right|_2^2 + \alpha\left||\beta|\right|_2^2$. This model solves a regression model where the loss function is the linear least squares function and regularization is given by the L2-norm [4].

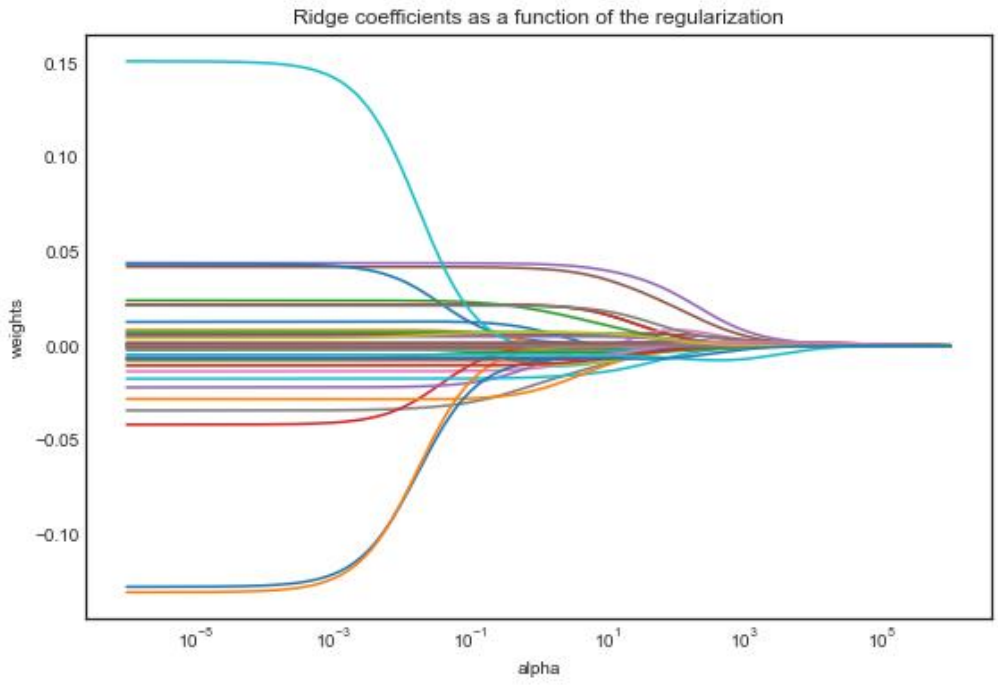

Figure 5: Weights plotted against alphas levels of this Ridge regression.

When alpha is very large, the regularization effect dominates the squared loss function, and the coefficients tend to zero. At the end of the path, as alpha tends toward zero and the solution tends towards the ordinary least squares [5]. Figure 5 shows us a plot of the ridge coefficients as a function of the regularization. For $\alpha = 10^{-5}$, there are about three coefficients that are approximately weighted either 0.15 or -0.15. These three coefficients differ a lot from most of the coefficients, that are bounded between weights = (-0.05, 0.05). Ridge estimators with $\alpha > 10^{3}$ return an MSE score of 0.000. This indicates that our models are overfitted. This attempt at predicting the 'prices' of Airbnb listings did not go as planned. I am unaware of which predictors return the best model.

2.4 Random Forests

A random forest is a meta estimator that fits several decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-

fitting. The sub-sample size is controlled with the max_samples parameter if 'bootstrap=True' (default), otherwise the whole dataset is used to build each tree [6]. When building these decision trees, each time a split in a tree is considered, a random sample of *m* predictors is chosen as split candidates from the full set of *p* predictors [7].

The first model used with this estimator includes all 41 predictors of the X_train data frame. This model returns a MSE = 0.000 and R^2 = 0.373, when rounded to three significant digits.

A second model is initiated, this time with 15 predictors from the training data frame. This model returns MSE = 0.000 and R^2 = 0.421, when rounded to three significant digits. We can visualize the variable importance. Which is computed using the mean decrease in Gini index and expressed relative to the maximum [6].

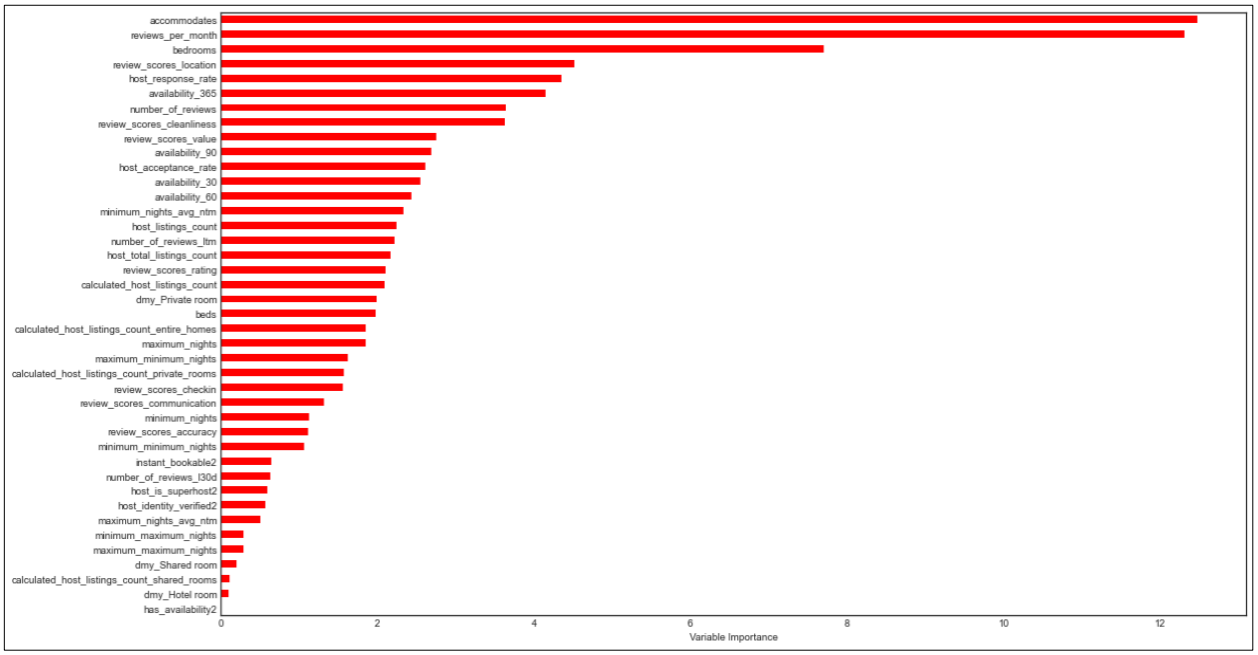

Figure 6: Variable importance of a Random Forest classifier, with 15 predictors.

We see that the variables with a variable importance score greater than 6 are, "accommodates", "reviews_per_month", and "bedrooms."

2.4.1 Boosting

In random forests boosting, trees are grown sequentially. Each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set. Note that in boosting, unlike in bagging, the construction of each tree depends strongly on the trees that have already been grown [8].

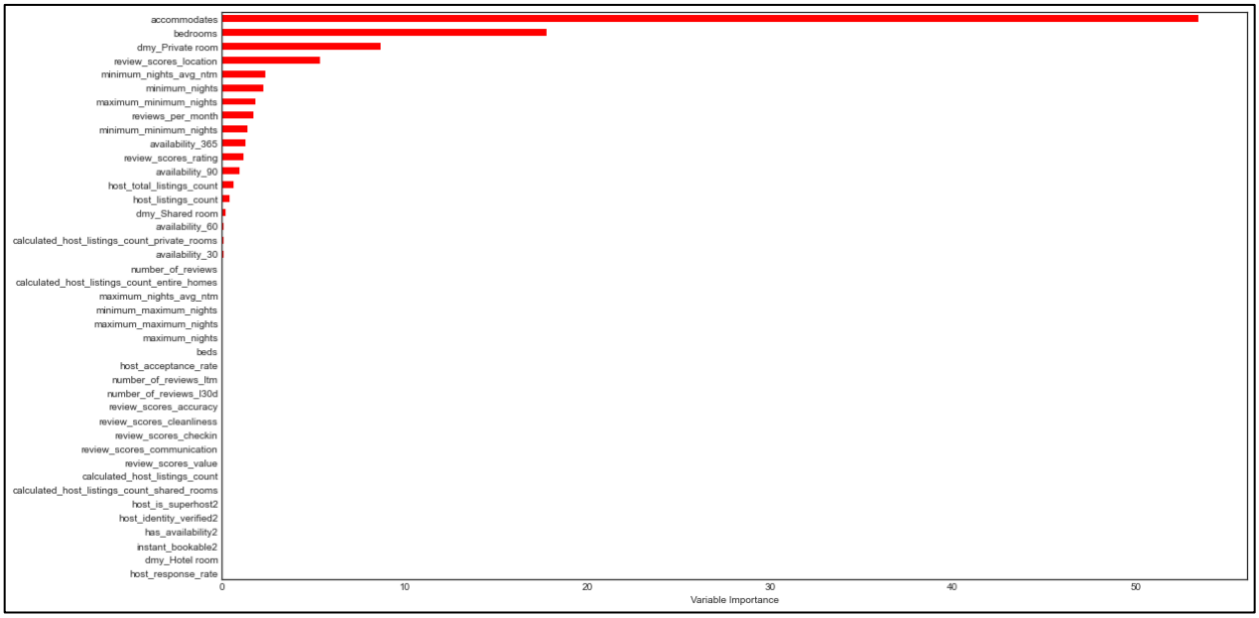

Figure 7: Variable importance of this Random Forest boosting classifier.

The x-axis scale in figure 7 increased when compared to the model without boosting. In this current model, a total of 20 predictors have a measure of variable of importance. The most prominent variables are “accommodates,” “bedrooms,” “dmy_private_room.” These have a score greater than 5. The MSE of this model is again, 0.000. However, we are returned a smaller R^2, with $R^2 = 0.229$. Although this correlation of coefficient is still not the best, we have a clearer sense of which predictors can help predict the price of a listing.

3. Classification Methods

3.1 Train Test Split

Classification methods prohibits the use of a continuous response variable. For the remaining methods, we will explore classification methods with "instant_booklable2" as the response variable. Originally, this was a boolean variable, with responses of true or false. According to the data dictionary, this variable is defined as; "Whether the guest can automatically book the listing without the host requiring to accept their booking request" [3]. "instant_bookable2" is now a numerical class variable, with 1 representing true and 0 representing false. In using this new response variable, the goal of this section is to use classification methods to assess which fields can help predict if a listing is instantly bookable.

The scaled data frame was once again used in a train-test-split, with a test size of 0.3. We are returned an equivalent number of rows/listings for the testing and training data frames.

3.2 Discriminant Analysis

3.2.1 Linear Discriminant Analysis

Out of the handful of dimensional reduction classifiers learned in lecture, discriminant analysis was the classifier with interpretable results. We begin with LDA, which can be used to perform supervised dimensionality reduction, by projecting the input data to a linear subspace consisting of the directions which maximize the separation between classes [9]. With the use of scikit-learn's function, an LDA model is initiated, with the "solver = 'svd'" parameter used on the training data frames.

The default number of classes in this model is two. The "priors_" attribute of this model estimates the proportion of the training observations that belong to the k^{th} class. In this model, $\widehat{\pi_1} = 0.724$ and $\widehat{\pi_2} = 0.276$.

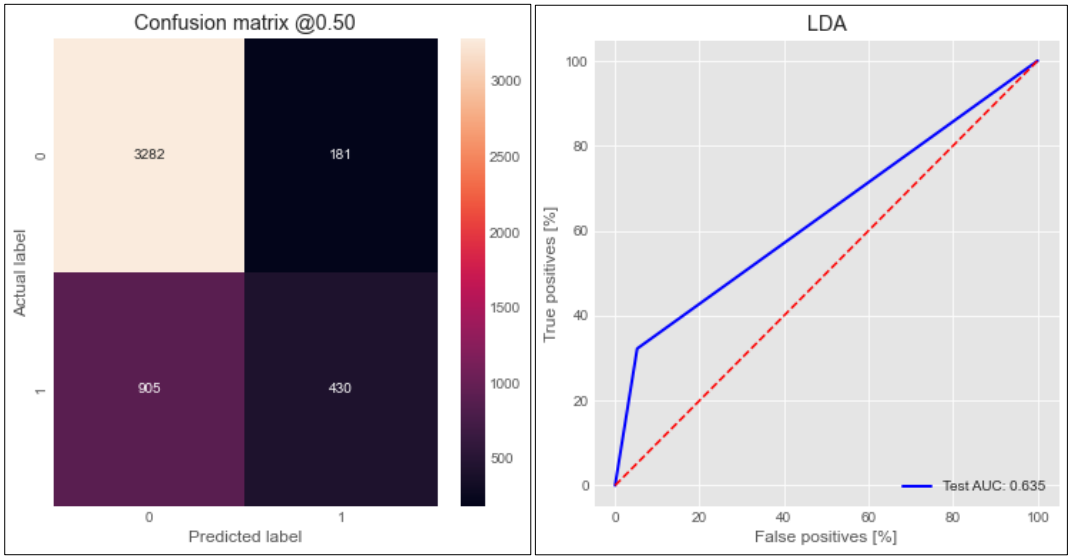

Figure 8 & 9: Confusion matrix and ROC curve for this LDA classifier.

From figure 8, we can calculate the accuracy and precision of this model. This model can accurately classify 77.4% of the listings whether they are instantly bookable. This model can precisely classify 78% of the listings that are not instant bookable and can precisely classify 70% of the listings that are instantly bookable. The AUC value from figure 9 is not optimal, with the ROC curve having a steep slope towards the red-dashed line.

3.2.2 Quadratic Discriminant Analysis

The results of the QDA classifier did not outperform the LDA classifier. The accuracy of this model decreased to approximately 73%. The AUC value of this classifier approached 0.5 at a much faster rate than the LDA classifier.

4. Conclusion

Using various regression and classification machine learning methods for this set of NYC Airbnb listings, we were able to fit models using ‘price’ or ‘instant_bookable2’ as the response variable.

The metrics used to evaluate the regression models were the MSE and R^2. Multiple linear regression was the first method considered. Initially, the RFE method was used to identify the number of predictors needed to find the optimal model. The first model returned an $R^2 = 0.282$. I then began to use K-fold cross validation, to find the R^2 values of 10 folds of the training data frame. Fold #7 returned $R^2 = 0.324$ and MSE = 0.0001. Grid search cv allowed me to hypothesize that the optimal number of predictors needed for this method is 26. A final model was fit with 26 predictors and returned $R^2 = 0.299$.

Ridge regression was then used on the training data frame. These results were not comprehensible. We do know that for alpha levels greater 10^3, the weights coefficients tend to zero. Ridge estimators also return an MSE score of zero for the same alpha levels.

The random forests method is the final regression method used in this paper. A model with all 41 predictors returned a MSE = 0.000 and $R^2 = 0.373$. I then decided to fit a model with only 15 predictors, with the expectation of increasing the MSE score. This model returned MSE = 0.000 and $R^2 = 0.421$.

When visualizing this model, “accommodates”, “reviews_per_month”, and “bedrooms” are the most important predictors according to the gini index. These fields tend to dictate the high or low cost of a listing. If a listing can accommodate more people per stay, with more bedrooms, the price of the listing tends to increase. A listing with more reviews per month gives validation to the host and gives more confidence to any potential guest.

A boosting random forest model was created, and a total 20 predictors have a measure of variable of importance. The most import variables are “accommodates,” “bedrooms,” “dmy_private_room.” This model returns an MSE = 0.000 and $R^2 = 0.229$.

Linear discriminant analysis was finally used. There is a change in the response variable, with ‘instant_bookable2’ being used. The initial LDA model can accurately classify 77.4% of the listings whether they are instantly bookable. This model can precisely classify 78% of the listings that are not instant bookable and can precisely classify 70% of the listings that are instantly bookable. The test AUC = 0.635.

Quadratic discriminant analysis is the final model discussed in this paper. Unfortunately, the results of the model underperformed. The accuracy of the QDA model decreased to approximately 73%. The AUC value of this classifier approached 0.535.

The model with the most interpretable results is random forests. The boosting method returned results that were more interesting and comprehensible. The difficult methods turned out to be the classification methods. We can argue that the use of the ‘insant_bookable2’ response variable complicated the interpretation of the results. The many predictors in the data frame did not make it easy to provide the best plots for a handfull of other classifying methods.

These models need to be developed further. It is beneficial to have access to this data and can be beneficial to know the many attributes of each individual listing. Unfortunately, most of the fields in the data frame did not provide reasonable results to the model. A MSE score of zero indicates that a model is potentially over-fitted. Which is the problem I ran into with most of the models discussed in this paper.

There is also a spatial aspect that can help predict the price of a listing or can help classify if a listing is instantly bookable. We can infer that Airbnb listings posted in Manhattan or within a 5-mile vicinity of Manhattan can increase the price of a listing.

Overall, this data set was interesting to work with. The use of a dashboard can help readers visualize where these listings are located. A dashboard can also help visualize the neighborhoods that do not offer many Airbnb listings.

5. References

[1] https://www.investopedia.com/articles/personal-finance/032814/pros-and-cons-using-airbnb.asp

[2] http://insideairbnb.com/behind.html

[3] https://docs.google.com/spreadsheets/d/1iWCNJcSutYqpULSQHlNyGInUvHg2BoUGoNRIGa6Szc4/edit#gid=982310896

[4] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html#sklearn.linear_model.Ridge

[5] https://scikit-learn.org/stable/auto_examples/linear_model/plot_ridge_path.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html?highlight=randomforestclassifier#sklearn.ensemble.RandomForestClassifier

[7] ISLR textbook, version 2, chapter 8.2.2 ‘Random Forests.’

[8] ISLR textbook, version 2, chapter 8.2.3 ‘Boosting.’

[9] https://scikit-learn.org/stable/modules/lda_qda.html#lda-qda