STA 160: Airbnb Midterm Project

Geospatial and Ethnic Analysis: NY Airbnb Prices

Ricardo Simpao, Falak Shah

Student ID: 914767659 Email: rlsimpao@ucdavis.edu
Student ID: 914663151 Email: fdshah@ucdavis.edu

Dr. Fushing Hsieh, Instructor
STA 160
University of California, Davis
May 3, 2021

STA 160 Project - Airbnb

1 Abstract

The sharing economy describes a system of sharing and renting out private resources, such as books, cars, apartment rooms, and even housing, where individuals often arrange their transactions through apps and websites. A major problem for the online systems arises from concerns for safety in asymmetric information situations, where the host and the tenant do not know the characteristics of the other. To address these concerns, companies, like Airbnb, mandate hosts to provide information on themselves and their listings through descriptions and pictures. However, the approach may unwittingly engender unconscious discrimination based on looks, ethnicity, and other characteristics. Our paper finds hosts with English names charge around $2.15 more for their listings compared to hosts with non-English names, controlling for listing features. Additionally, our paper finds factors such as neighborhood group, room type, minimum required nights, number of reviews, the number of listings by the given host, and location-dependent variables are also significant factors of a listing's price.

2 Introduction

Uber, Lyft, and Airbnb represent a few examples of the dominant companies in the sharing economy for the transportation and accommodation sector. Most notably, Airbnb provides a world-wide network of accommodations, facilitating tourist trips and temporary shelters otherwise not available to the public. In order to quell concerns of safety of entering an unknown host's accommodations, Airbnb encourages hosts to provide substantial information surrounding their listings as well their own personal characteristics via written descriptions and images. Through this approach of providing information, however, potential tenants may experience an unconscious bias in selecting which listings to choose. Our paper will delve into visual analysis to explore some geographical relationships with price as well as model building through multiple linear regression, LASSO, and random forest and classification through Logisitic Regression to gain insights on what influences the listing's price and price level.

3 Related Works

Our paper builds on the ideas presented in "Digital Discrimination: The Case of Airbnb.com," a working paper by Benjamin Edelman and Michael Luca in 2014. Their study explores the claim of whether consumers discriminate on their chosen listing based on the perceived ethnicity of the host. Their paper uses a data set that contains comprehensive information on hosts and Airbnb listings for New York in 2012, such as host pictures, social media presence, listing pictures, listing scores, and other amenities. To derive the ethnicity of the hosts, they hired Amazon Mechanical Turk workers to manually identify the host of ethnicity given their posted pictures, and they later simplify their analysis to "Black" and "Non-Black" hosts. Their study finds that non-Black hosts charge approximately 12% more than Black hosts, controlling for host and listing attributes.

Our study also follows another paper "Key Factors Affecting the Price of Airbnb Listings: A Geographically Weighted Approach" by Zhang, Chen, Han, and Yang, which discusses the various factors that influence the listing prices. Their study uses 794 observations of Airbnb listings from Metro Nashville, Tennessee. They leverage their data by building a general linear model and a geographically weighted regression model to identify the significance of their factors. The most notable feature of their work comes from their analysis of the listing's distances to the highways and Metro Nashville's convention center. They find that prices lower with increased distance from the convention center, a major economic hub.

4 Methodology

4.1 Dataset

Our paper uses pricing and location data of 2019 Airbnb listings in New York City. The data contains roughly 49,000 observations of listings with information on price, geographical location, last reviews, reviews per month, availability, and room type. For the purposes of our analysis, we will categorize the prices into four ranges: affordability, moderately priced, expensive, and very expensive. Inspired by the work on geographical factors influencing Airbnb listings, we also utilize data with geographical information on the nearest New York subways and major routes as further amenities to the given listings. Finally, our paper pursues a variation of the study by Edelman and Luca, by calculating the predicted ethnicity of the hosts derived from their first name.

4.2 Multiple Linear Regression

Multiple linear regression models the relationships between a response variable and independent explanatory variables by minimizing the squared error terms of the residuals and predicted values. The regression follows this form:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

We use the multiple linear regression to generate a preliminary model for our other regression models.

4.3 Logistic Regression

Binary logistic regression models the probability a given listing falls within a certain price range or availability class. The regression follows this form:

$$l = log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

4.4 LASSO

LASSO regression is an extension of linear regression that adds penalties to the loss function during training to have less complex models with smaller coefficients. In other words, it uses regularization to prevent overfitting of the data. LASSO uses shrinkage and shrinks the less important predictors towards 0. Given the nature of our data set, (more on this in the implementation section) we use LASSO to predict price and compare its performance to linear regression and Random Forest.

The mathematical equation for LASSO:

$$\sum_{i=1}^{n}(y_i - \sum_j x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

λ = Shrinkage parameter (which can be tuned by cross validation)

4.5 Random Forest

Random Forest, used in regression and classification, is a supervised machine learning algorithm that uses ensemble learning for regression. Ensemble learning is a method that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. Due to its ability to work effectively with large data sets and handle many input variables, we use it to predict price and compare its performance to Linear and LASSO regression.

4.6 One-Versus-Rest Classification

Our paper addresses the problem of multiple classification of price ranges through the one-versus-rest approach. Essentially, we use the logistic regression model to find the probability a certain listing classifies as "affordable" or "not affordable." Then, we repeat the process for the other classes, determining a probability for each class given the listing characteristics. In the end, we label the comment based on which class has the highest probability.

4.7 K-Fold Cross Validation

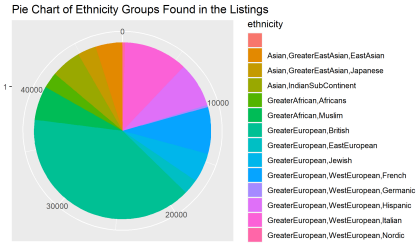
We use k-fold cross validation to assess the quality of our classification models, specifically we use five-fold cross validation. K-fold cross validation describes a process of repeatedly dividing our data into training and test sets, model fitting, and evaluating the MSE. In one iteration of the procedure, the first step divides the data into k partitions, using one of the k partitions as the validation set. From there, we can fit a model on the remaining k-1 partitions and test our model on the validation data. We achieve an error score from the first iteration, using the the training and predicted labels. We then repeat the same procedure for a different partition out of the k partitions and obtain another error score. At the end of k iterations, we obtain k errors scores and average them to obtain an overall evaluation metric for how well our chosen model predicted the validation data.

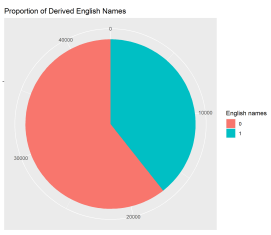
5 Implementation Details

5.1 Preprocessing

To prepare our data, we first note that several NA values exist in our 'reviews per month' variable and replace them with zeroes. We also delete the fourteen rows wherein the price of the listing is $0. Additionally, on observing the distribution of minimum_nights (Fig 5.2.2(a)), 48870 out of the 48884 listings have minimum nights less than or equal to 365. Among them, 48137 listings (98.5% of our data) have minimum nights less than or equal to 30 (Fig 5.2.2(b)). Hence, we filter the data to only keep those rows.

For our demographic analysis, we utilized the names of the hosts to predict the host ethnicity. We use the Python module "ethnicolor," which uses U.S. census data, Florida voter registration data, and Wikipedia data to train their model to classify ethnicity given only the first and last name. Our paper uses their model trained on Wikipedia data to classify names into either one of fourteen ethnic groups. To facilitate the classification process, we cleaned the names with unneeded characters. Additionally, for simplicity, we assume that the first name carries as much predictive power as last names, and in listings with multiple names, we used the first given name. To facilitate our analysis, we filtered out empty and non-name listings as well as listings that had ambiguous results in the ethnicity classification. To address the imbalance of ethnic groups, we simplify the classifications as either "English" or "non-English," classifying "GreaterEuropean,British" as "English" and everything else as "non-English."

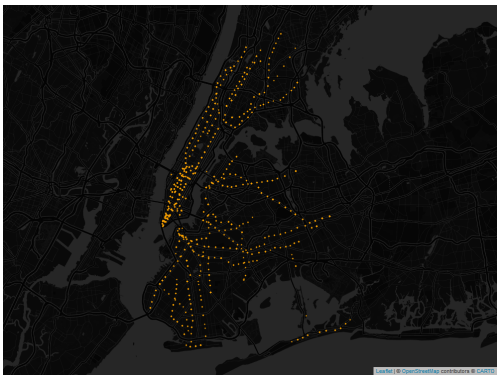




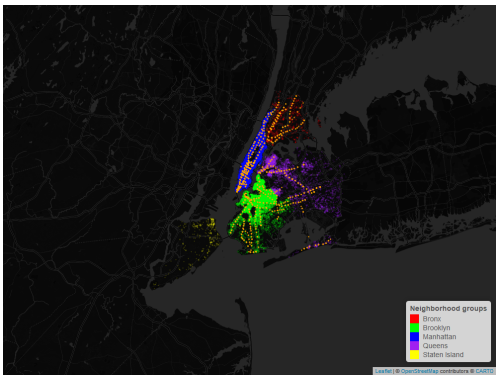
For our geographical analysis, we used data containing the geographical locations of New York subway stations. We reason that the farther the listing is from the nearest subway station, the listing price lowers to compensate. In transportation economics theory, a similar phenomenon occurs in housing prices and their distance towards the inner city, where housing prices tend to rise from rural to urban areas of the city. To obtain the distances between listings and subway stations, we utilized an R library called "geosphere" and used their function distGeo to calculate a distance matrix between locations given their latitude and longitude. As the locations are relatively close, there are minute differences in the calculation of their other given functions, so we opt to use the distGeo function. After calculating the distance matrix, we found the nearest subway station and then calculated the distance in miles.

5.2 Exploratory Visualization

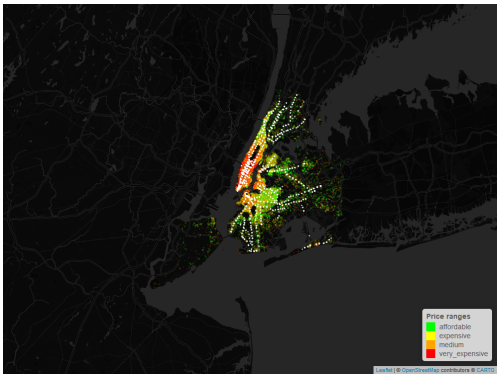
5.2.1 Geographical Analysis



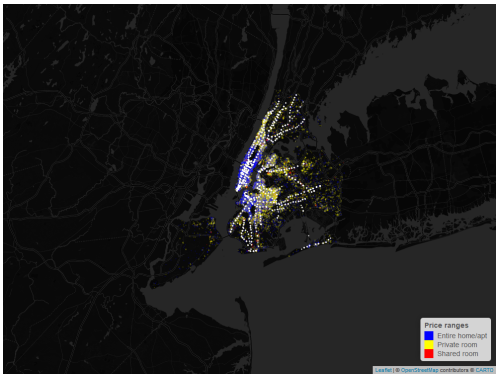
(a) Subway Stations in New York



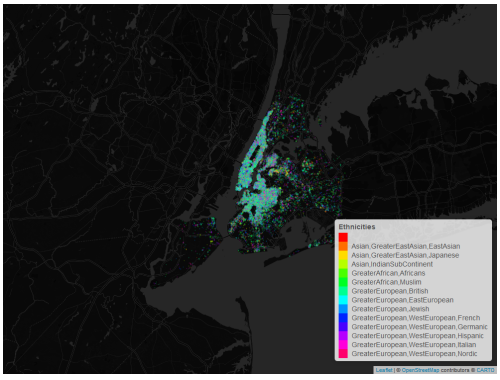
(b) Listings by Neighborhood Group



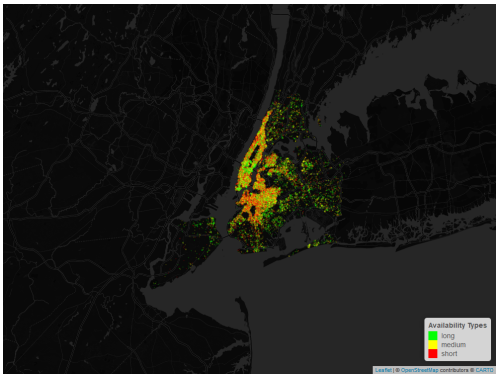
(c) Price with Subway Station Locations



(d) Room Type with Subway Station Locations



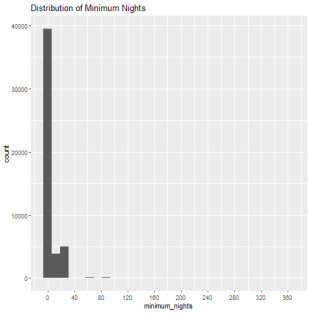
(e) Ethnic Groups Concentrations in New York



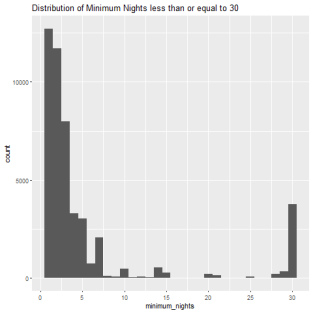
(f) Availability Ranges

From our generated maps, we immediately gain some insights on the potential factors that will influence price. From map (b), we note the large concentration of subways in the Manhattan listings and relatively disperse subways systems in other neighborhood groups, suggesting geographical distances as a potential important factor. From map (c) and (d), we see that the 'very expensive' category of prices and houses tend to concentrate in the Manhattan area as well, where we can intuitively infer that home and apartment listings will tend to have higher prices compared to private and shared rooms. Map (e) displays the concentration of predicted ethnic groups in our listing data. Map (f) displays availability types as approximately uniform in its dispersion.

5.2.2 Variable Analysis



(a) Distribution of Minimum Nights



(b) Distribution of Minimum Nights less than or equal to 30

5.3 Model Fitting and Cross Validation

5.3.1 Multiple Linear Regression

We begin with splitting the pre-processed data set into training and test sets and then randomly sample 70% of the rows as the training set and the remaining 30% as the test set.

For the first model, we fit price using all the available regressors (excluding miles): neighbourhood_group, latitude, longitude, room_type, minimum_nights, number_of_reviews, reviews_per_month, calculated_host_listings_count and availability_365. We then use the fitted model to predict prices for the test data. We observe that the performance of this model is highly below average with $R^2 = 0.1116$ and Residual Mean Squared Error(RMSE) = 201.7488.

Given the performance of the first model, we add the predictor "miles" to, potentially, improve the performance. The model gives an R^2 value of 0.1116, identical to the first model, and the test set RMSE is 201.7454, a slight improvement on the first model.

On observing the rather indifferent performances of the above two models, we inspect our independent variable - "price".

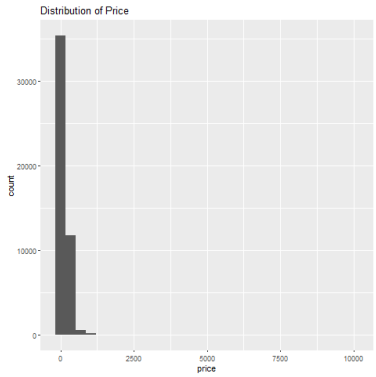
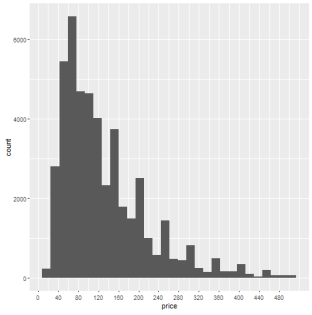
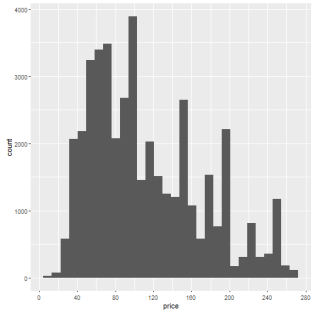


Figure 4: Distribution of Price

As we can visualize from the histogram above (Figure 4), the distribution of price is highly skewed. On further exploration, we find that only 1012 out of the 48137 listings have a price greater than $500. Furthermore, since most of the prices fall within $300 as shown in the histogram below, we filter out the listings with prices greater than $269 - the 90^{th} percentile of price.



(a) Distribution of Price within $500



(b) Distribution of Price within $269

After filtering, the distribution of price looks approximately more normal (figure (b) above). We now fit the remaining models on the listings whose price falls within $269. From our second model, we achieve an R^2 value of 0.5155 and the test set RMSE 41.2377, a significant improvement on our initial models. Furthermore, we observe that the predictors "latitude" and "longitude" distort the model and make the coefficients difficult to interpret. Hence, we drop these two variables and fit the model again. This model gives us an R^2 value of 0.4971 and RMSE = 42.03. We see a minuscule change in removing the predictors, and furthermore, we can now easily interpret our model and its coefficients.

Next, we add the variable "English"- a dummy variable - as one of our predictors. We derive "English" from the predicted ethnicity of the host name in the data set. We encode Host_names, which we predict associates with the British ethnicity, as 1, and we encode all other names as 0. The fit suggests that this variable is significant at the level 0.01. After k-fold fold cross validation (k = 5), the R^2 for this fit is 0.4948 and the test set RMSE is 41.34. We choose this model as our final linear model.

Therefore, the final linear regression model is:

Price = 121.198 + 20.76(Brooklyn) + 47.003(Manhattan)+ 11.678(Queens) + 12.37(Staten Island) - 70.76(Private Room) - 94.47(Shared Room) - 0.857(Minimum_nights) - 0.02(Number of reviews) - 1.57(Reviews per month) + 0.15(calculated_host_listings_count) + 0.05(Availability_365) - 3.44(Miles) + 2.149(English)

Interpretation of a few selected coefficients:

- Intercept: When all the predictors are equal to 0, the predicted price of an Entire home/Apartment in Bronx is $121.198.
- Brooklyn: Given, everything else is constant, an Entire Home/Apt in Brooklyn costs $20.76 more than it does in Bronx. Coefficients for Manhattan, Queens and Staten Island can be interpreted similarly.
- Private Room: Everything else being constant, a private room in Bronx costs $70.76 less than an entire Home/Apt. The coefficient for Shared Room can be interpreted similarly.
- Minimum Nights: For every one unit increase in minimum nights, the price decreases by $0.857, given everything else is kept constant. Similar interpretations for Number of reviews, Reviews per month, Calculated host listings count and availability_365.
- Miles: Everything else being constant, for every one mile increase in distance from the nearest subway station, the price goes down by $3.44.
- English: Everything else being constant, a host with an English name, charges $2.149 more for the listing as compared to a host with a non-English name.

5.3.2 Logistic Regression

We use the final linear regression model as the base line for our logistic regression model. We divide the listing prices into four ranges: affordable ($< \$69$), medium ($\$69 - \$106$), expensive ($\$106 - \$175$), and very expensive ($> \$175$). We use one-hot encoding for the four new categorical variables. We choose the price thresholds as to roughly achieve equal proportions of each class in our data set. We then used a one-versus-rest strategy to classify each listing as either one of the four price classes.

First, we fit a binary logistic model on whether a listing classifies as "affordable" and "not affordable." Then, we fit the logistic model for the "medium," "expensive," and "very_expensive" price classes. We classify a given Airbnb listing as the price label with the highest of the four probabilities from our logistic models. Lastly, to evaluate the performances of models, we use five-fold cross validation. The process splits the data into training and validation sets, calculates the predicted classes of the validation set, and compares with the observed classes. We then obtain accuracy scores and MSEs for each model as given in Table 4 in the Results section.

5.3.3 LASSO Regression

We implement Lasso regression to reduce model complexity and prevent over-fitting, which might result from our implementation of linear regression. Since we have 221 unique neighbourhoods, 5 different neighbourhood groups, 3 different room_types and other numerical variables, we use LASSO regression to automate feature selection. Lasso will aid us in shrinking the coefficients and will push the coefficients for the less important parameters to 0, helping us optimise the complex model for prediction by avoiding over-fitting. The tuning parameter, lambda, which adjusts the amount of coefficient shrinkage, will be chosen via cross validation.

First, we create dummy variables for each level of every categorical variable in the data set i.e. for the 221 unique neighbourhoods, 5 neighbourhood groups and the 3 room types. We consider every level of each categorical variable as a viable predictor in predicting price before implementing lasso regression. In total, we have 237 predictor variables, including our previous predictors.

To choose the optimal lambda value, we perform the k-fold cross validation with k = 10. Here, the data is randomly split into 10 subsets, one subset is set aside and the model is trained on all the remaining subsets. We test the model on the validation subset. We repeat the procedure until each of the 10 sets has served as the validation set. Finally, we average the 10 errors.

The optimal value of lambda obtained after k-fold cross validation (k = 10) is 0.1529.
The lasso regression model is now fit on the training data using this optimal value of lambda. The R^2 value is **0.5593** and the test set RMSE is **38.98**. Therefore, we can see that Lasso regression, with its automated

feature selection and regularization, performs better than linear regression.

The figure below shows a screenshot of a few coefficients obtained after performing lasso regression (Figure 6).

minimum_nights	-0.89113315
number_of_reviews	-0.02773393
reviews_per_month	-1.05479641
calculated_host_listings_count	0.09274559
availability_365	0.05718129
miles	-2.11580477
English	1.29757450
V1_Bronx	-12.85150628
V1_Brooklyn	.
V1_Manhattan	35.73403107
V1_Queens	-3.89734516
V1_Staten Island	-4.84447897
V1_Entire home/apt	65.76549424
V1_Private room	.
V1_Shared room	-22.85003113
V1_Allerton	.
V1_Arden Heights	-5.20962540
V1_Arrochar	.
V1_Arverne	8.74478496
V1_Astoria	4.59162324
V1_Bath Beach	-8.44550889
V1_Battery Park City	7.06114510
V1_Bay Ridge	.
V1_Bay Terrace	42.28241794
V1_Bay Terrace, Staten Island	.
V1_Baychester	.
V1_Bayside	.
V1_Bayswater	.
V1_Bedford-Stuyvesant	.
V1_Belle Harbor	0.56768206
V1_Bellerose	2.47794719
V1_Belmont	.
V1_Bensonhurst	-11.24581425
V1_Bergen Beach	.
V1_Boerum Hill	25.11811099
V1_Borough Park	-17.02457369
V1_Breezy Point	135.06779534
V1_Briarwood	.
V1_Brighton Beach	-4.51068260
V1_Bronxdale	-14.42868713
V1_Brooklyn Heights	28.39419739
V1_Brownsville	-12.68447788
V1_Bull's Head	.
V1_Bushwick	-1.78565893
V1_Cambria Heights	0.69974553
V1_Canarsie	-12.41871629
V1_Carroll Gardens	25.83110289
V1_Castle Hill	-9.19527579
V1_Castleton Corners	2.11711730
V1_Chelsea	17.96491600
V1_Chinatown	-0.04614749
V1_City Island	.
V1_Civic Center	2.76180910
V1_Claremont Village	.
V1_Clason Point	-2.03620386
V1_Clifton	-9.92746435

Figure 6: Coefficients from Lasso Regression

5.3.4 Random Forest

Random Forest is based on bagging and feature randomness and constructs multiple trees based on a subset of data. This leads to a reduction in over-fitting and variance along with increasing the accuracy of the model fit. Furthermore, the Random Forest algorithm does not need normalization of data (feature scaling) and has a robustness towards outliers. It uses an in-built validation set to calculate the Out of Box (OOB) error.

First, given the size of our data set, we split it into training (70 %) and test (30%) sets. We compare the performance of our predictor variables with the categorical variables without encoding to the performance of our predictor variables with one-hot encoding for each level of every categorical variable we have (221 neighbourhoods, 5 neighbourhood groups and 3 room types). For our first analysis, we simply use the default Random Forest parameters in the ranger package in R (number of trees = 500, the number of variables to sample at each split (mtry) = number of predictors/ 3, node size = 5).

The R^2 for the non-dummy version is 0.578 and the Out of Box Prediction Error MSE is 1440.389 (RMSE = 37.96), whereas R^2 for the dummy version is 0.589 and the Out of Box Prediction Error MSE is 1401.582 (RMSE = 37.44). Hence, initial observation suggests that the model with dummy variables for every level

of each categorical variable performs better in comparison to the other model.

To maximize the predictive power of our model, we now perform a full grid search to tune the following hyperparameters:
- **mtry**: Number of variables to randomly sample at each split
- **nodesize**: Minimum number of samples within the terminal node
- **sampsize**: Number of samples to train on

To perform the grid search, we create a grid with a range of values for each parameter we want to tune and then loop through each parameter combination to evaluate the model. The model with the dummies created has 237 columns in total. Hence, the range of parameters set for mtry is (15,150) with a sequence of 20, nodesize is (3,9) with a sequence of 2 and sampsize = (0.632, 0.7, 0.8). Similarly, for the model with no dummies (10 columns),the range of parameters set for mtry is (2,8) with a sequence of 2, nodesize is (3,9) with a sequence of 2 and sampsize = (0.632, 0.7, 0.8).

The tables below shows the results (only the first 15) for the best combination of parameters arranged in increasing order of the Out of Box Prediction Error. As we can observe from the tables, the model with the dummy variables (Table 1) continues to perform marginally better than the one without the creation of dummy variables (Table 2).

Hyperparameter Tuning				
	mtry	Node Size	Sample Size	Out of Box RMSE
1	55	7	0.7	37.332
2	55	7	0.8	37.344
3	55	5	0.8	37.345
4	55	3	0.7	37.349
5	55	9	0.632	37.351
6	55	3	0.632	37.352
7	55	7	0.632	37.353
8	55	9	0.7	37.354
9	55	5	0.7	37.356
10	55	5	0.632	37.358
11	55	9	0.8	37.362
12	55	3	0.8	37.366
13	75	9	0.7	37.389
14	75	5	0.7	37.401
15	75	7	0.632	37.415

Table 1: *Tuned parameter combinations obtained by grid search for the model with dummy variables for each level of every categorical variable.*

Hyperparameter Tuning				
	mtry	Node Size	Sample Size	Out of Box RMSE
1	4	9	0.7	37.838
2	4	9	0.632	37.860
3	4	7	0.632	37.868
4	4	9	0.8	37.883
5	4	7	0.7	37.892
6	6	9	0.632	37.913
7	4	5	0.7	37.923
8	4	5	0.632	37.928
9	4	7	0.8	37.931
10	4	5	0.8	37.939
11	3	5	0.7	37.945
12	7	5	0.632	37.948
13	3	5	0.632	37.949
14	9	5	0.7	37.950
15	9	5	0.8	37.967

Table 2: *Tuned parameter combinations obtained by grid search for the Non-dummy model*

Therefore, we find the optimal model fit as the one with dummy variables and hyperparameters - **mtry = 55, node size = 7 and sample size = 0.70**. This model gives an R^2 value of 0.5920 and an Out of Box (OOB) Prediction Error of 1395.366 (RMSE = 37.35). We evaluate the performance of this model on the test data and get a test RMSE of 37.67. These values show that the Random Forest model performs better than both, Lasso and Linear Regression.

The histogram (Figure 7) below shows the 25 most important variables in reducing the MSE of the model. We measure variable importance by recording the decrease in MSE each time a variable is used as a node split in a tree. Reduction in MSE is accumulated for each variable across all the trees. The higher reductions in accumulated MSE imply the greater importance of the variable. Distance of the listing from the closest subway station, miles, is the third most important variable in this respect.

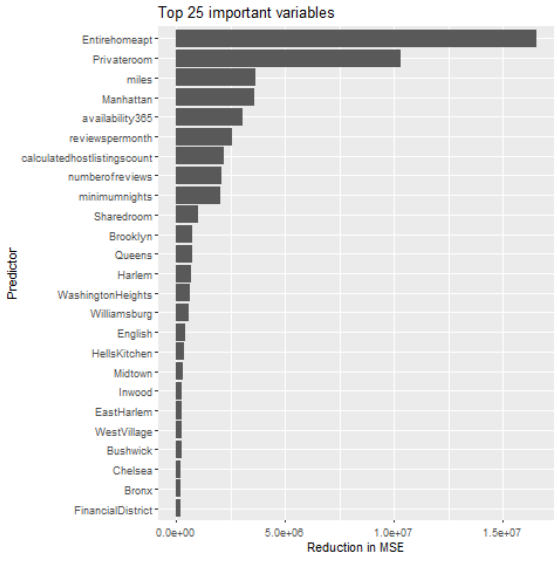


Figure 7: 25 most important predictors

An interesting aside:
Since the grid search has heavy time complexity, we quicken the process of tuning the hyperparameters via the h20 package, a scalable open-source machine learning platform that offers parallelized implementation of many supervised and unsupervised machine learning algorithms including Random Forest. After inputting the range of parameters to tune the model,the combination suggested by this tuning mechanism was **max depth = 100, min_rows =3, mtries = 40, number of trees = 300, and sample size = 0.55**. This fit gives an MSE of 1405.40 (RMSE = 37.49). Since, the RMSE with the grid search was better, we fit that model.

6 Results

Comparison of Methods and Models for prediction of Price				
	Method	Model Number	R^2	Test RMSE
1	Linear Regression	1	0.1116	201.749
2	Linear Regression	2	0.1116	201.745
3	Linear Regression	3	0.5155	41.23
4	Linear Regression	4	0.4971	42.03
5	Linear Regression	5	0.4948	41.34
6	LASSO Regression	1	0.5593	38.98
7	Random Forest (Without dummy variables)	1	0.5806	38.75
8	Random Forest (With dummy variables)	2	0.5921	37.67

Table 3: *Comparison of the performances of various models and methods*

As mentioned in the implementation section, Linear regression model 1 includes predictors: neighbourhood_group, latitude, longitude, room_type, minimum_nights, number_of_reviews, reviews_per_month, calculated_host_listings_count and availability_365. For Model 2, we include miles as another predictor. Model 3 onwards price is filtered to include only values falling within the 90% percentile ($269). For interpretation purposes, we get rid of latitude and longitude in Model 4. Finally, we add English as our predictor in Model 5. Linear Regression Model 5 is our chosen Linear Model in comparison to Model 3 as the predictor English is significant.

As the table suggests, in predicting price, Random Forest with dummy variables created for each level of every categorical variable (Random Forest Model 2) performs the best amongst all the models and methods involved. In terms of performance, it is followed by Random Forest Model 1 (No dummy variables created), LASSO regression and Linear Regression Model 5. Linear Regression performs the worst amongst all the algorithms implemented and models fitted. Table 3 summarises the R^2 value and Test RMSE for every fit.

Table 4 below summarises the accuracy and cross-vallidated MSE of classification for every price range. The price range affordable has the highest classification accuracy followed by Very Expensive, Expensive and Medium.

Classification Accuracy for different levels of Price			
	Price Range	Test Set Classification Accuracy	Cross-Vallidated MSE
1	Affordable	0.8244	0.1172
2	Medium	0.7421	0.1814
3	Expensive	0.7584	0.1671
4	Very Expensive	0.7942	0.1356

Table 4: *Classification accuracy for different price ranges*

7 Conclusion

Our paper takes a variation of the approach in the Digital Discrimination paper. Instead of classifying ethnicity by the host's picture, our paper classifies by the host's given name in the data set. One caveat arises from the lack of a last name, which the "ethnicolor" model suggests, but for our analysis, we assumed that first name carries as much ethnic information as last names. We expect some underestimation in non-English classification of hosts. Furthermore, to enhance our analysis, we accounted for geographical factors of the listings, finding distances of the listings to their nearest subway stations. Our paper finds that price decreases about \$3.44 as the listing location for every mile away from the nearest subway station. Additionally, a host with English name, on average, post \$2.15 higher listing prices than hosts with non-English names for similar listing attributes. Both results fall in line with the conclusions of the respective papers we followed.

On comparing the performance of various model fits for three different regression techniques (Linear Regression, LASSO, Random Forest), our paper finds that that in terms of predicting price, Random Forest prevails as the most accurate regression technique with the highest R^2 value and the least Residual Mean Squared Error (RMSE). For speed and memory optimization, we use the package h20 to tune the hyperparameters for Random Forest after doing a manual grid search. Overall, any Random Forest model fit suggested by these approaches, outperforms the fit suggested by Linear and LASSO regression.

8 Code Appendix

All available code can be found on the public Github repo https://github.com/dlsimpao/AirbnbStudy. Code is sorted by Visualization, ModelBuilding, and Python notebook files.

8.1 Additional Plots

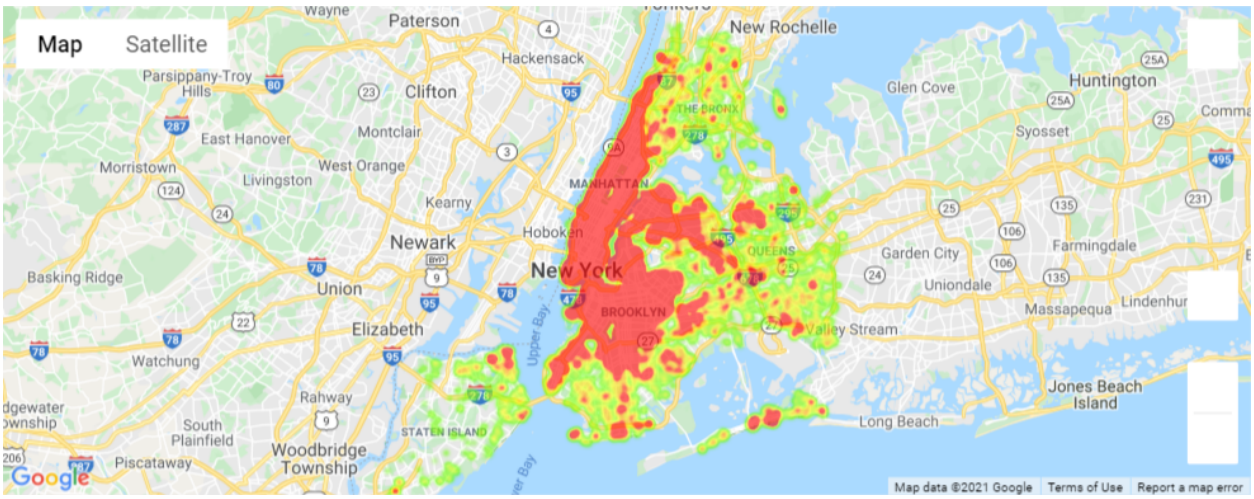


Figure 8: Price Heat Map

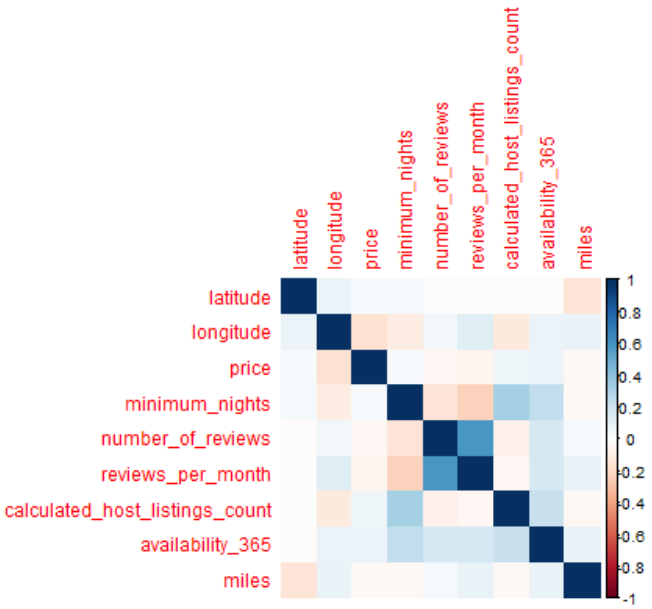


Figure 9: Correlation Plot

9 Supplementary Material

NYC Subway dataset is accessible in the nyc-transit-subway-entrance-and-exit-data.csv. Apart from ethnicity predictions, the project is implemented in R. Ethnicity prediction is performed in Python and is accessible through the python notebook, Ethnicolor code.impynb. The predictions are then stored as a csv, which is then imported into R. The csv is accessible through EthnicityPredictions.csv. Furthermore, airbnb_wdistances.csv includes the original airbnb dataset along with the closest station name and distance to the nearest subway for each listing (Code for this is available in the Model Building Rmd). Figure 8 in Additional Plots is done in Python and can be accessed through the Python Heatmap.ipynb. Code for Pre-processing, Linear, LASSO and Random Forest model building can be found in the Airbnb Model Building.Rmd file. Code for the leaflet plots can be found in the Airbnb Visualization.Rmd file.

10 References

1. Key Factors Affecting the Price of Airbnb Listings: A Geographically Weighted Approach:
   www.researchgate.net/publication/319865700_Key-Factors-Affecting-the-Price-of-Airbnb-Listings-A-Geographically-Weighted-Approach
2. Digital Discrimination: The Case of Airbnb.com: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2377353
3. Heatmap of Price in Python: https://towardsdatascience.com/ai-and-real-state-renting-in-amsterdam-part-1-5fce18238dbc
4. LASSO Regression Implementation in R: https://www.pluralsight.com/guides/linear-lasso-and-ridge-regression-with-r
5. LASSO Regression Mathematical Equation: https://www.mygreatlearning.com/blog/understanding-of-lasso-regression/
6. Random Forest Implementation and Hyperparameter tuning in R: https://uc-r.github.io/random_forests
7. Classification (Logistic): https://kavita-ganesan.com/news-classifier-with-logistic-regression-in-python/#.YE01E50zZPY
8. Inside Airbnb: http://insideairbnb.com/
9. Geosphere: https://stackoverflow.com/questions/32363998/function-to-calculate-geospatial-distance-between-two-points-lat-long-using-r
10. Ethnicolor: https://pypi.org/project/ethnicolr/