AirBnB Listings: An in depth dive into the world of short-term sublets

Armandas Bartas, Alex Romanus, Braeden Norman, Gabriel Lanzaro

2021-12-07

Introduction

This project aims to investigate AirBnb listings and obtain insights into the most important features of short-term sublets. More specifically, the goal of this project is to classify the cities based on different attributes. The dataset, which was obtained from Kaggle, contains 10 cities from very distinct parts of the world: Bangkok, Cape Town, Hong Kong, Istambul, Mexico City, New York, Paris, Rio de Janeiro, Rome, and Sydney. The Airbnb data possesses 280 000 listings including information related to host info, geographical data, price, number of bedrooms, amenities, review scores, etc.

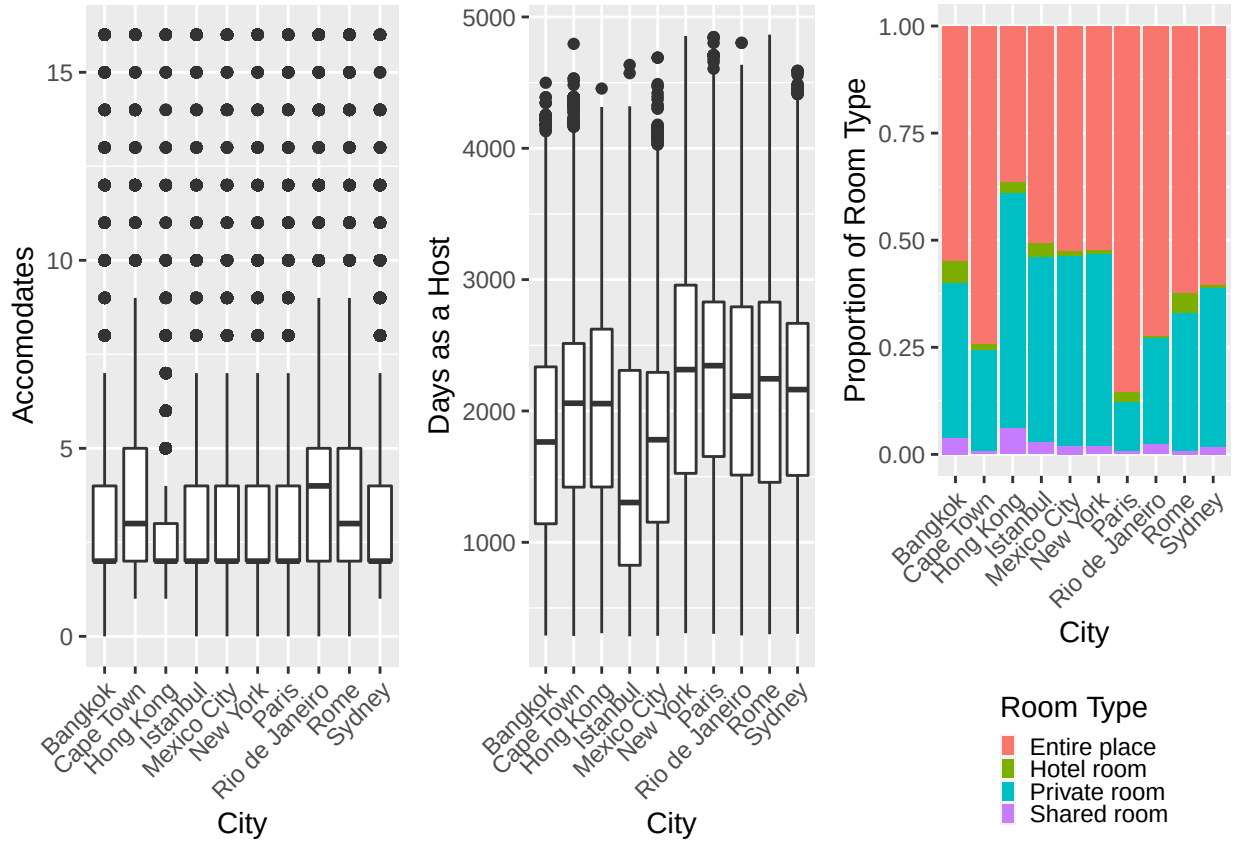
Previous research has shown that online review scores significantly contribute to selecting a place to stay (Zhao et al., 2015; Thomsen and Jeong, 2020). Moreover, the availability of locations with great review scores can influence the choice of a destination to spend a vacation. Therefore, it is crucial to develop models to further understand the destination selection. Local travel agencies can then target specific factors for improvement (e.g., reducing prices, setting mandatory amenities, demanding minimum review scores to keep hosts in the system).

The analysis can then reveal important aspects regarding how different attributes may characterize each city, for example:
- Which amenities are more important for each city when selecting a property?
- Does the host profile differ among different cities?
- Which types of accommodation are more common depending on the city?
- Can we predict the city based on different preferences related to the place to stay?
- What are the most important AirBnB variables to predict a city for destination?

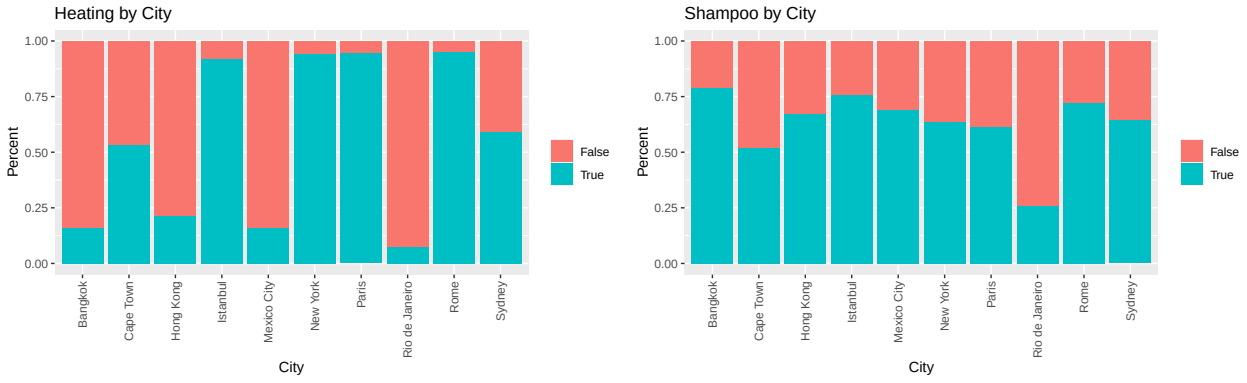
This project uses two classification algorithms to predict the cities: random forests and multinomial logistic regression. These methods have been selected as they provide different inferences about the data.

Exploratory Analysis

The exploratory analysis revealed some important information about the dataset. For example, the next figure shows graphs that present (1) the number of guests the listing accommodates, (2) for how long the host has been renting properties in AirBnB, and (3) the room types for different categories. The first boxplot indicates that cities such as Cape Town, Rio de Janeiro, and Rome tend to offer listings with more guests, which might be suitable for group or family trips. The second boxplot shows that AirBnb has been used in some cities for more time than in others. For example, New York and Paris have, on average, hosts with more AirBnb time than in Istambul. It might show that AirBnb has been widely used in Istambul only recently. The third plot shows that most of the accommodations in Paris and Cape Town are for the entire place, and most of the accommodations in Hong Kong are for private rooms.



In the original dataset, there is a column with a list of the possible amenities a listing has. There were 2865 different amenities and a lot of them only have a count of 1. So, to better view the distribution, we removed all amenities that had below 100 observations. The plots below represent the percent TRUE/FALSE values of 2 of the 60 included amenities in the updated dataset. For example, in Heating, we only have 4 cities that have almost no listings with heating (i.e., Bangkok, Hong Kong, Mexico City, and Rio de Janeiro). These cities tend to have the highest temperatures throughout the year.



Exploring Interactions

The addition of interaction terms is a form of basis expansion. Interaction terms model the change in response variables against the change in a product or quotient of two or more predictor variables. They help in the case where the added effects of two predictor variables do not stack linearly, rather, they compound

on each other. The right interaction terms can improve a model, but adding too many terms, or the wrong terms, increase variance and can actually worsen out of sample performance.

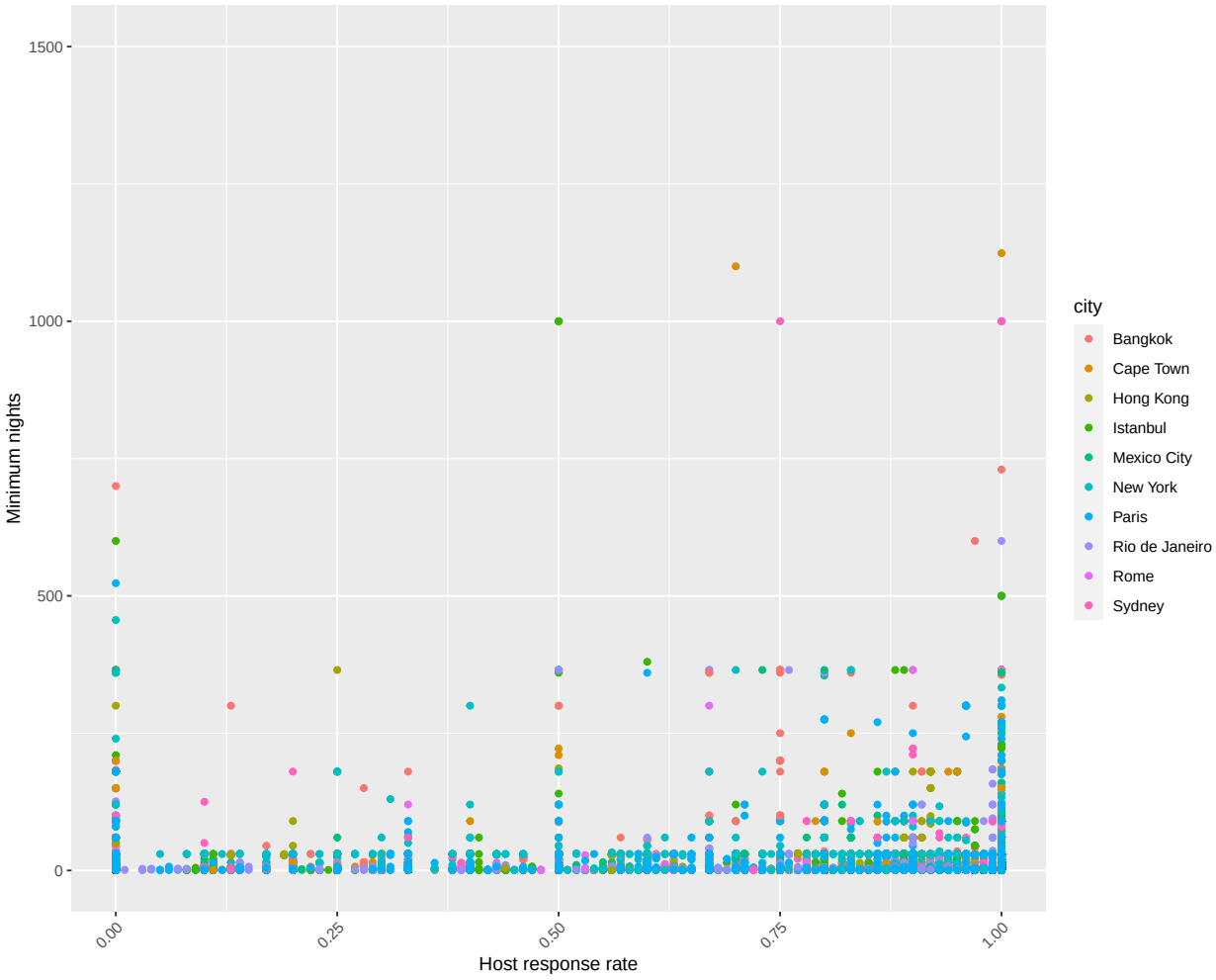
Method

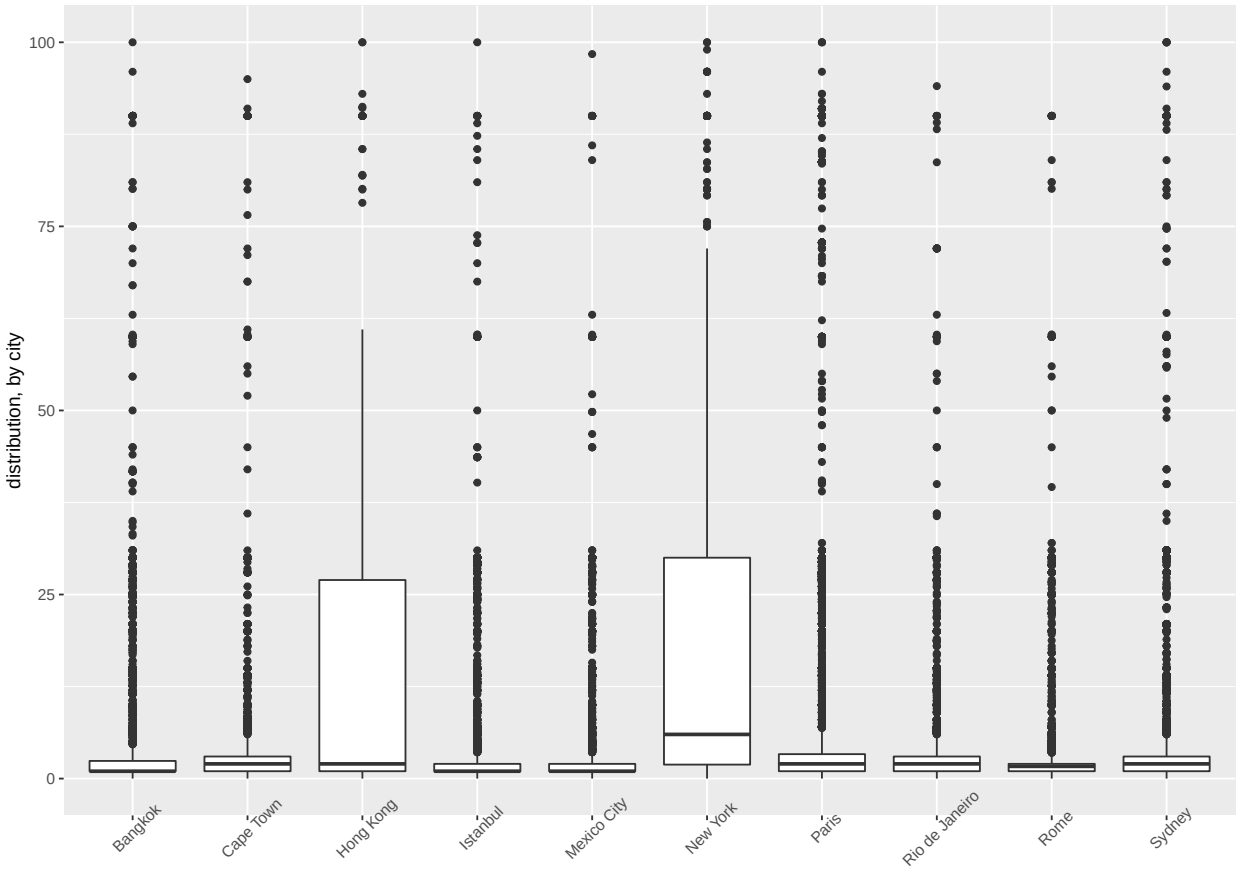
The method was to systematically fit a separate model for each interaction term. The interaction terms used were products of pairs of predictor variables. As such, the categorical variables were not included in the analysis of interactions. The true/false variables were converted into binary 0/1. Some data cleaning had to be done: ListingID and name were removed as they uniquely identify the corresponding observation, geographical information, such as neighbourhood, longitude, latitude, district, and location were removed as they also eliminate the challenge of predicting the city. Amenities were also removed for this analysis as they provided too much trouble. Each combination of amenities is considered a unique categorical variable. Overcoming this challenge is covered within this report, but it was not necessary for this particular analysis. The data set was cut down from ~300k observations to 1k observations, blocked by city and randomized, with 100 observations for each one. This was done to expedite the process of fitting so many models. The model used was a basic LDA model over all variables, minus the aforementioned removals. For each model, one unique product of two numeric variables was added to the predictor space, and the misclassification rate was recorded.

Results

The best performing models are seen in the table below. The default (no interaction terms added) misclassification rate was 0.013. There were two interaction terms that provided a rate of 0.008. The first was host_response_rate * minimum_nights, and the second was host_identity_verified * review_scores_communication. Five interaction terms gave a misclassification rate of 0.009, three of which contained the variable host_is_superhost, and the other two contained the variable bedrooms. Of these seven interaction terms, four contained a binary 0/1 variable, essentially meaning that the other term in the interaction had a different effect on the city depending on some binary condition.

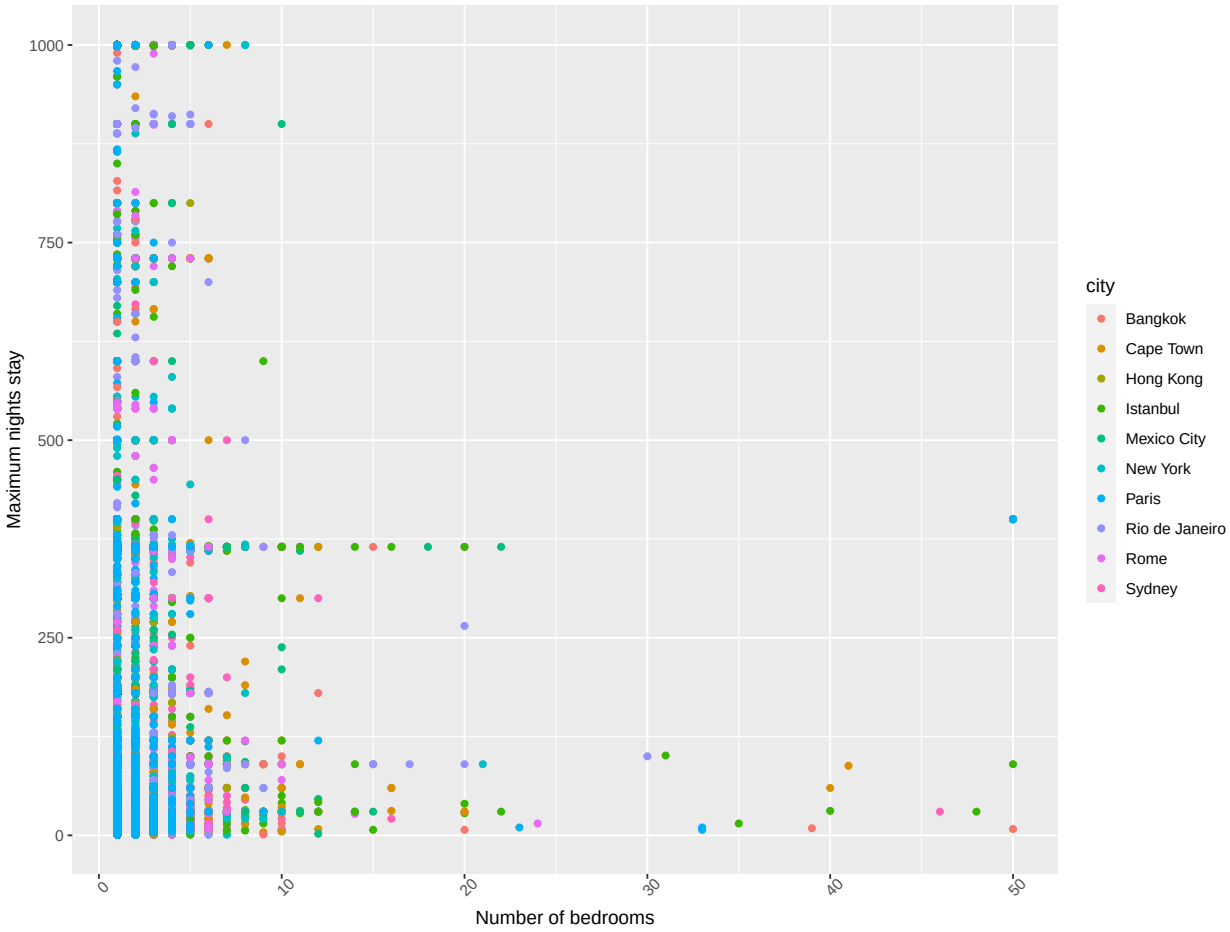
```
##      [,1]
## [1,] "misclass error"
## [2,] "0.008"
## [3,] "0.008"
## [4,] "0.009"
## [5,] "0.009"
## [6,] "0.009"
## [7,] "0.009"
## [8,] "0.009"
##      [,2]
## [1,] "model formula"
## [2,] "city ~ . + host_response_rate * minimum_nights"
## [3,] "city ~ . + host_identity_verified * review_scores_communication"
## [4,] "city ~ . + host_is_superhost * host_total_listings_count"
## [5,] "city ~ . + host_is_superhost * price"
## [6,] "city ~ . + host_is_superhost * review_scores_location"
## [7,] "city ~ . + bedrooms * maximum_nights"
## [8,] "city ~ . + bedrooms * review_scores_location"
```


1500
1000
500
0
Minimum nights
0.00
0.25
0.50
0.75
1.00
Host response rate
city
Bangkok
Cape Town
Hong Kong
Istanbul
Mexico City
New York
Paris
Rio de Janeiro
Rome
Sydney

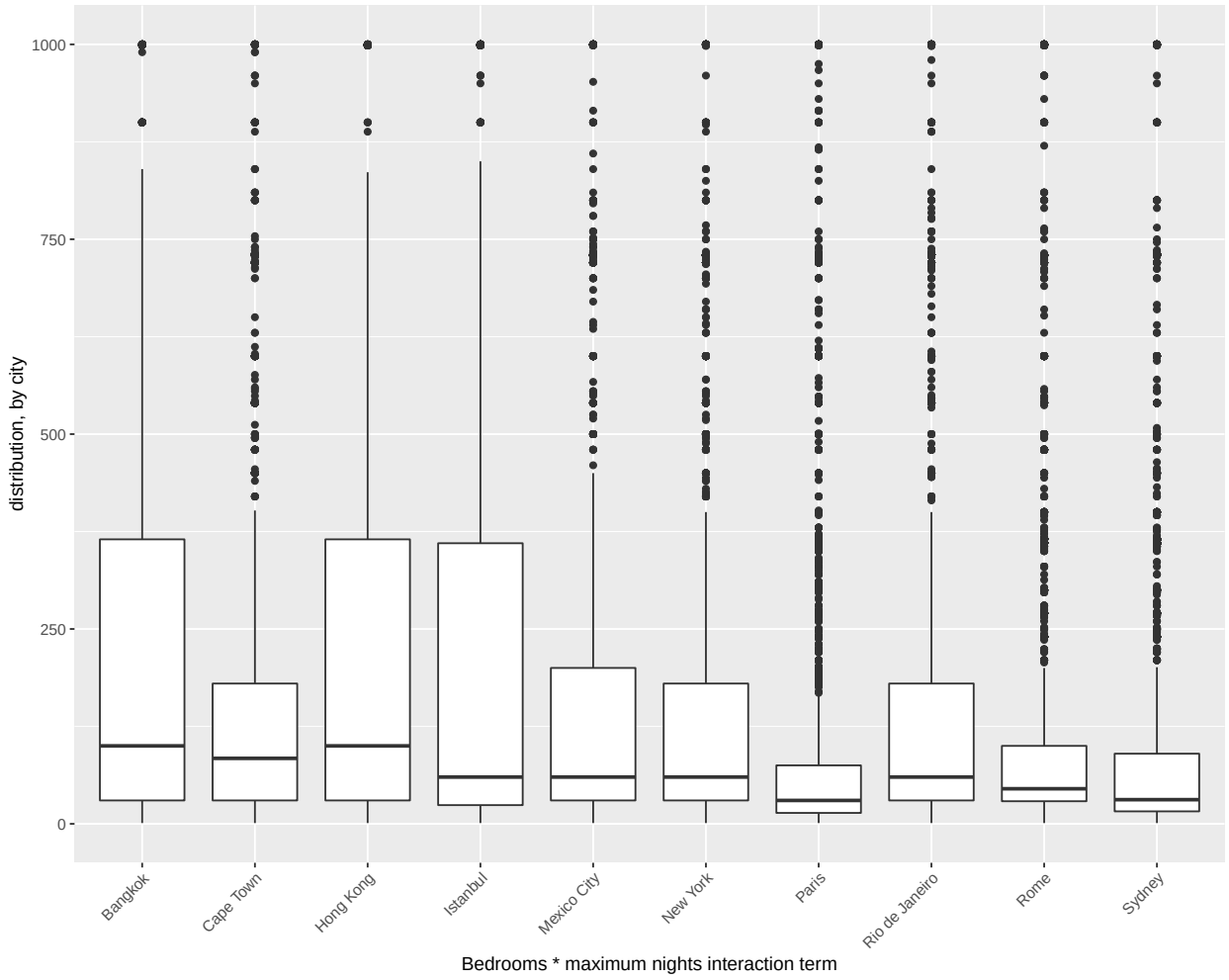


Host response rate * minimum nights interaction term

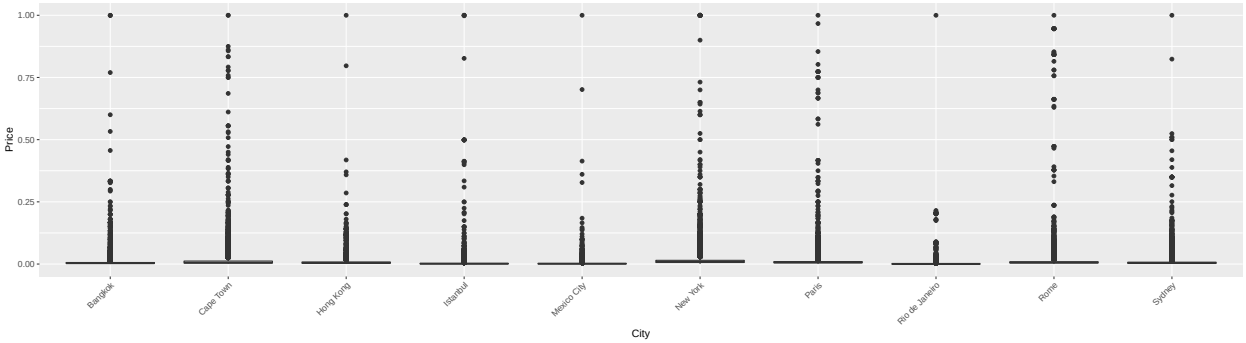
As we can see, this interaction term helps to identify observations as New York or Hong Kong Airbnbs because their distributions are much higher than the other cities in the dataset. Adding this interaction term to the model may improve the model's ability to correctly identify these two cities, as well as deter it from misidentifying other cities as New York or Hong Kong.



We can see that the observations' cluster along the axes of the plot, resulting on low product interaction terms. We see a cluster of observations coloured 'Paris' in the bottom corner, implying that Parisian Airbnbs have low values for their interaction terms. Looking at the next plot, a boxplot of the interaction term by city, we see this is confirmed.

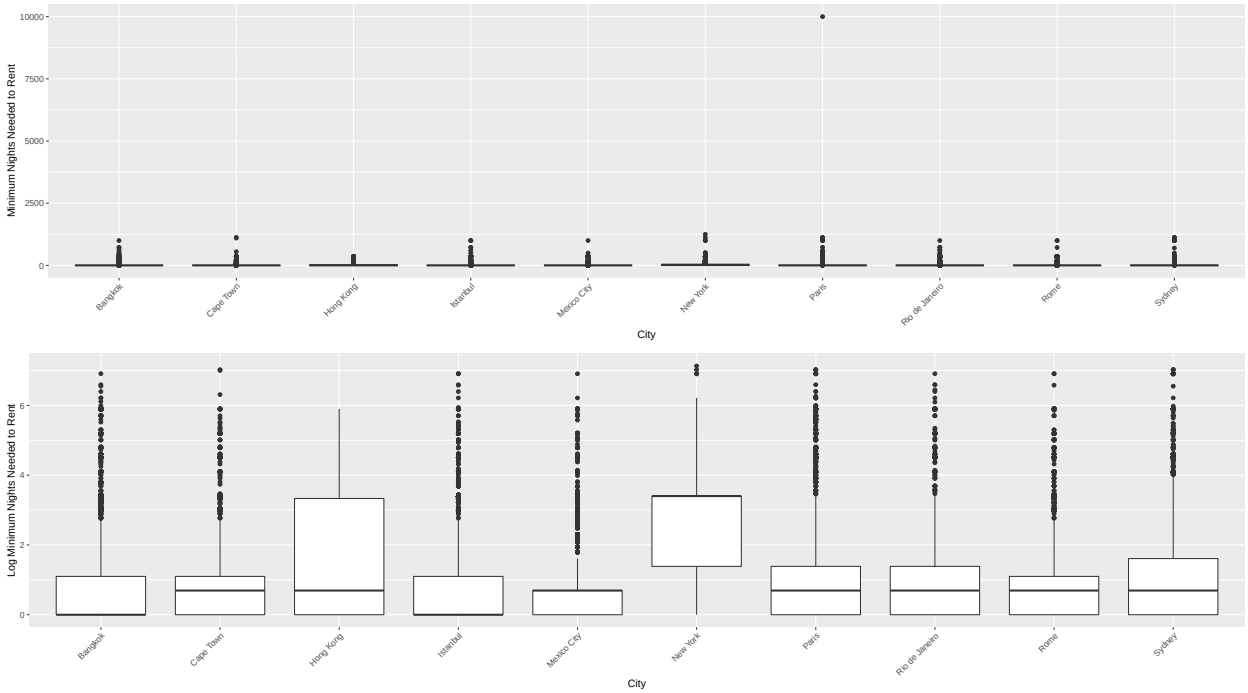


Exploring Price



Since prices of Airbnb's are recorded in each city's country's own currency, to make the prices comparable, it makes sense to standardize the variable. The best way to make all listings' prices comparable is to divide each observation's price by its city's costliest Airbnb. That way, the spreads of the data separated by city are preserved while making the price variable unitless to allow for comparison of observations between cities.

Exploring Minimum Nights



Since no other observations have anywhere near 10000 minimum nights required to rent, which no person would realistically want to rent anyway, it is reasonable to remove this observation from the data set. Furthermore, there are still many outliers, i.e., values above the whiskers, preventing the spread from being reasonably assessed, so we apply a log transformation to minimum_nights to make it more readable.

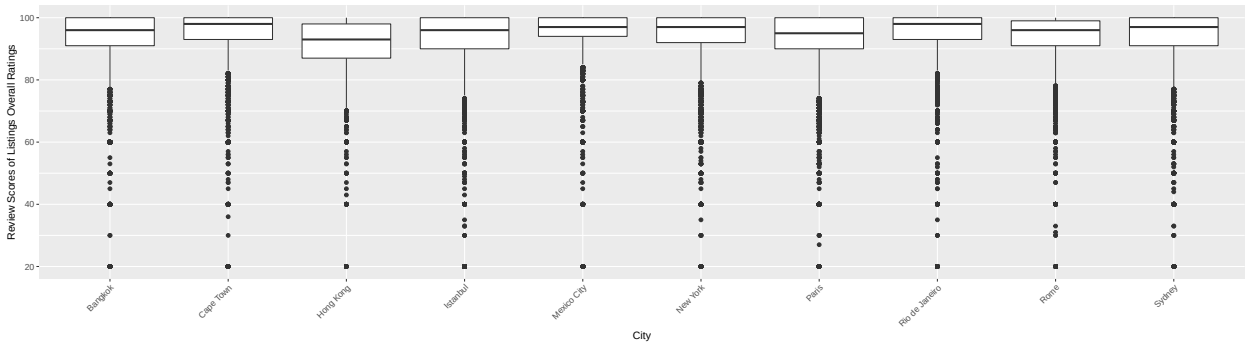
Upon transforming the data, it is clear that most cities have many listings with very few minimum nights required to rent, as many of their boxes show Q1's hovering right at 0. New York is the only exception, with a Q1 value of about 1.5 log days, or about 4.5 days, and a median of about 12 days. Conversely, Istanbul and Bangkok have medians hovering just above 0. These distinct spreads of values of minimum days may make Istanbul, Bangkok, and especially New York much easier to predict if this variable is significant in our model.

Exploring Review Score of Overall Rating

Many observations had to be removed from the data to accommodate the large amount of NANs in the review_score_ratings variable.

The high medians of and Q1 values of Cape Town and Rio suggest that these cities may be easier to predict than the rest. Hong Kong also seems to have a significantly lower rating spread of ratings than the rest of the cities, suggesting it may also be easier to predict than the rest.

However, unless the model used is sensitive to subtle differences in the values of these ratings, it likely won't be very useful for classification. This is because the spreads of the data for these reviews are all very similar, with almost all quantiles above a score of 90. This contradicts what we originally thought, as we predicted that overall reviews may differ significantly from city to city. Generally, it seems Airbnb customers seem to give high reviews.



Exploring Review Scores of Location

Similar to overall review scores, many observations had to be removed to analyze review_scores_location due to the high volume of NANs. Some cities, such as Hong Kong, were significantly effected, as they have far fewer location reviews than, say, Paris.

The largest proportion of high reviews, i.e., 9s and 10s, belong to Paris at 24.99% and 26.09% respectively. Also, Istanbul has large shares of low reviews, with 23.15%, 28.57%, and 22.22% of 2s, 3s, and 4s respectively, compared to the other cities. These heavy tails may make it easier to classify these two cities if review scores based on location are a significant predictor in our model.

```
##            city    10   2 3  4  5   6   7    8    9
##         Bangkok  5598  62 0 33 15 181 217 1230 3832
##       Cape Town 10900  50 1 10  9  94  59  426 1847
##       Hong Kong  2698  29 0 12  4  33  24  175  804
##        Istanbul  7886 147 2 52 28 212 137  713 2042
##     Mexico City 12674  70 0  8  2  78  42  233 1336
##        New York 18819  64 0 27 14 200 151 1245 6220
##           Paris 36662  76 1 22 12 243 239 1882 8833
##  Rio de Janeiro 13657  46 1 20  7 114  63  497 1701
##            Rome 13842  29 1 27 21 138 130 1076 5548
##          Sydney 17759  62 1 23  8 187  89  988 3186
```

```
##            city     10      2      3      4      5      6      7      8      9
##         Bangkok  3.98%  9.76%  0.00% 14.10% 12.50% 12.23% 18.85% 14.53% 10.84%
##       Cape Town  7.76%  7.87% 14.29%  4.27%  7.50%  6.35%  5.13%  5.03%  5.23%
##       Hong Kong  1.92%  4.57%  0.00%  5.13%  3.33%  2.23%  2.09%  2.07%  2.27%
##        Istanbul  5.61% 23.15% 28.57% 22.22% 23.33% 14.32% 11.90%  8.42%  5.78%
##     Mexico City  9.02% 11.02%  0.00%  3.42%  1.67%  5.27%  3.65%  2.75%  3.78%
##        New York 13.39% 10.08%  0.00% 11.54% 11.67% 13.51% 13.12% 14.71% 17.60%
##           Paris 26.09% 11.97% 14.29%  9.40% 10.00% 16.42% 20.76% 22.23% 24.99%
##  Rio de Janeiro  9.72%  7.24% 14.29%  8.55%  5.83%  7.70%  5.47%  5.87%  4.81%
##            Rome  9.85%  4.57% 14.29% 11.54% 17.50%  9.32% 11.29% 12.71% 15.69%
##          Sydney 12.64%  9.76% 14.29%  9.83%  6.67% 12.64%  7.73% 11.67%  9.01%
```

Logistic Regression Analysis

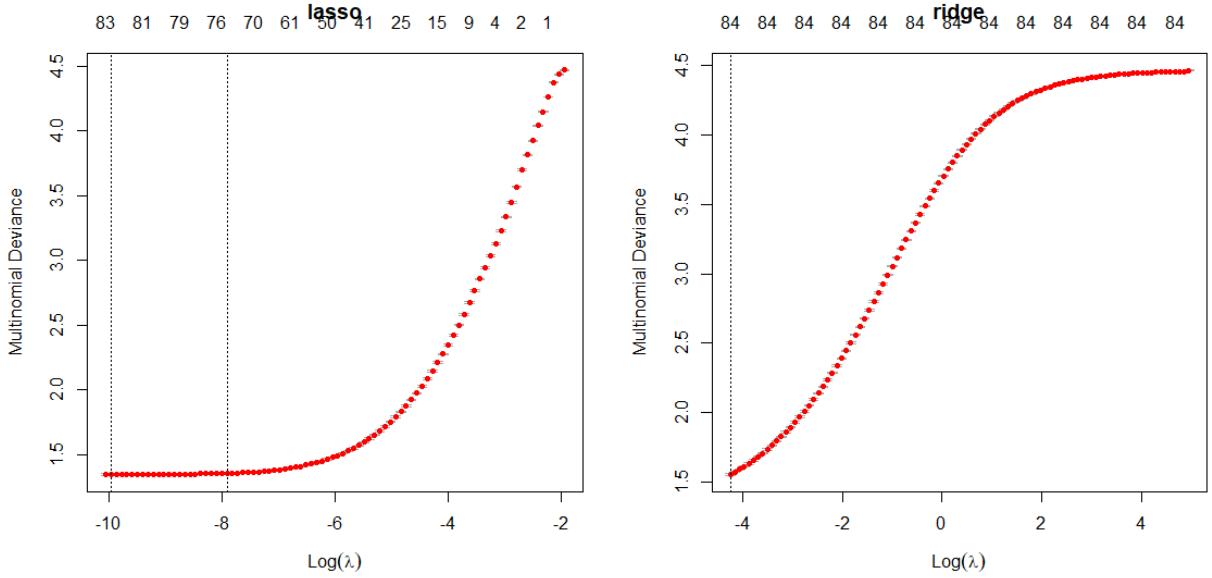
This report considered classification models to determine the different cities based on the AirBnb data. Seven different classification models were applied to the dataset: (1) lasso logistic regression with default lambda,

(2) lasso logistic regression with the lambda selected via cross-validation, (3) ridge logistic regression with default lambda, (4) lasso logistic regression with lambda selected via cross-validation, (5) linear discriminant analysis (LDA), (6) quadratic discriminant analysis (QDA), and (7) logistic regression.

The dataset was split into two sets: a training dataset with 80% of the data and a test dataset with 20% of the data. The errors were then computed using the test dataset. Out of the seven proposed models, the lasso logistic regression with lambda selected via cross-validation presented the best performance because of its lower error (i.e., higher accuracy).

The plots below show different values of lambda for the lasso and the ridge logistic regressions. It also shows the number of predictors that were used in each model. Lasso performs better because it uses a subset of the predictors, and some of the predictors were not useful in the final model. Therefore, the accuracy of the lasso model with fewer predictors outperformed the ridge model.

```
##       Lasso.Logit.cv Lasso.Logit Ridge.Logit.cv Ridge.Logit    LDA  QDA
## Error         0.2342      0.3219          0.244      0.5506 0.2619 0.31
##       multinom
## Error    0.532
```



Preprocessing was also used to try to increase the model performance.The table below shows the accuracies with different preprocessing techniques. By default, glmnet standardizes the data. Therefore, no differences were found between the accuracy reported previously and the accuracy with standardization below. In addition, there were no great differences by implementing the min-max normalization.

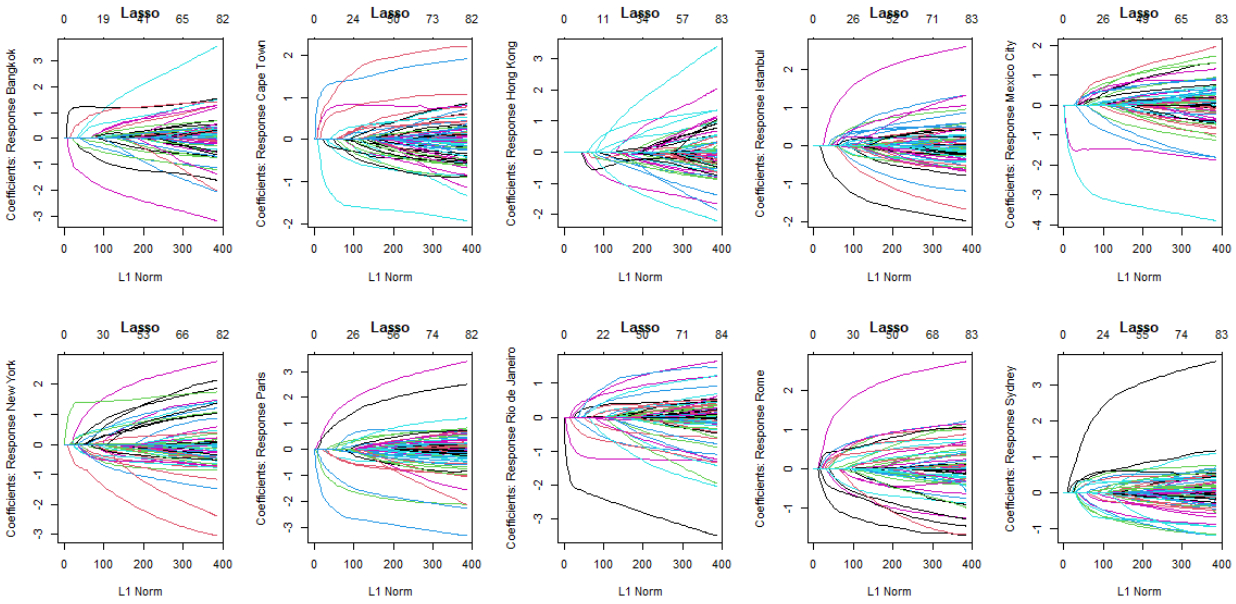
```
##       Lasso.Without Lasso.With.Standardization Lasso.With.Min.Max.Scaling
## Error        0.8566                    0.2342                     0.7842
```

The tables below shows the accuracy of the model by city. The first table is a confusion matrix, and the second is the accuracy by city. Rio de Janeiro had the highest accuracy of all cities: it presented an accuracy of approximately 83%.

```
##                Reference
## Prediction       Bangkok Cape Town Hong Kong Istanbul Mexico City New York
##   Bangkok           5041        84       267      117          59       21
##   Cape Town           57      6707        15      281         461       73
##   Hong Kong          170         6      1182       39          11       43
##   Istanbul           147       241       101     4516         198      353
##   Mexico City         53       735        17      204        8826       50
##   New York            46       139       128      561          95    10123
##   Paris               10       124        68      649         321      448
##   Rio de Janeiro     387       342       210      120         420       64
##   Rome                75       247       194     1069         160      276
##   Sydney             121       632        70      346         211      425
##                Reference
## Prediction       Paris Rio de Janeiro  Rome Sydney
##   Bangkok           23            366    47    144
##   Cape Town         85            264   207    535
##   Hong Kong         30             80    57     59
##   Istanbul         398             93   787    350
##   Mexico City      169            642   115    334
##   New York         491             85   455    550
##   Paris          11624             53   639    574
##   Rio de Janeiro    68           9147   143    123
##   Rome             505            184  9158    195
##   Sydney           297             81   114   5897
```

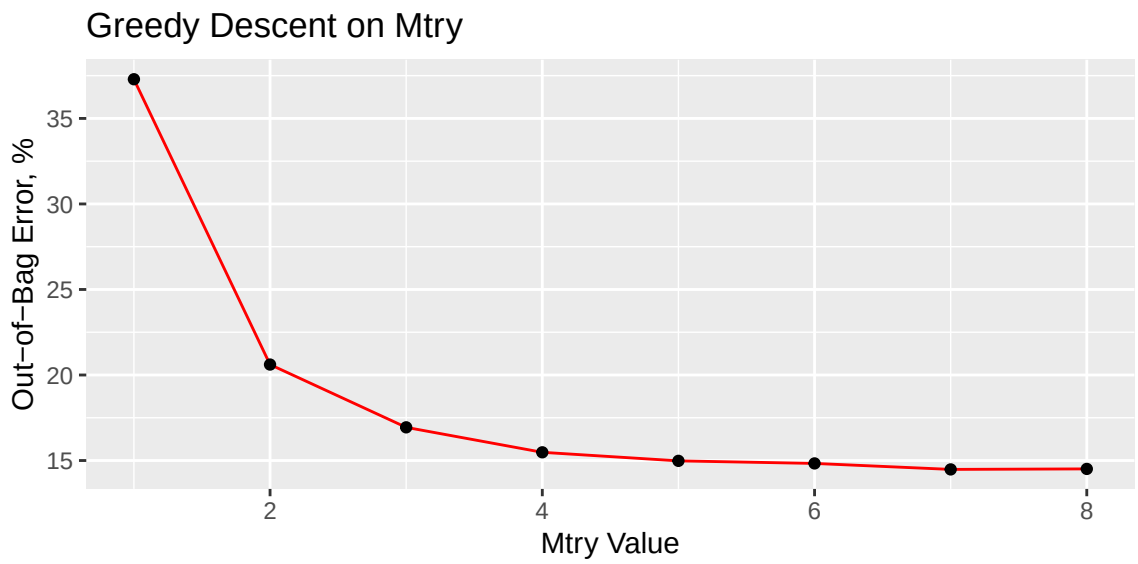
```
##        Bangkok      Cape Town      Hong Kong       Istanbul    Mexico City
##      0.8171503      0.7722510      0.7048301      0.6286192      0.7919246
##       New York          Paris Rio de Janeiro           Rome         Sydney
##      0.7987848      0.8011027      0.8297351      0.7591810      0.7196729
```

Finally, the plots below display the coefficient values for the different cities. The figures show that the coefficients may vary considerably depending on the city. For example, the coefficients for Sydney are very different from the coefficients for Cape Town. It shows the variables may exert different degrees of influence in different cities.

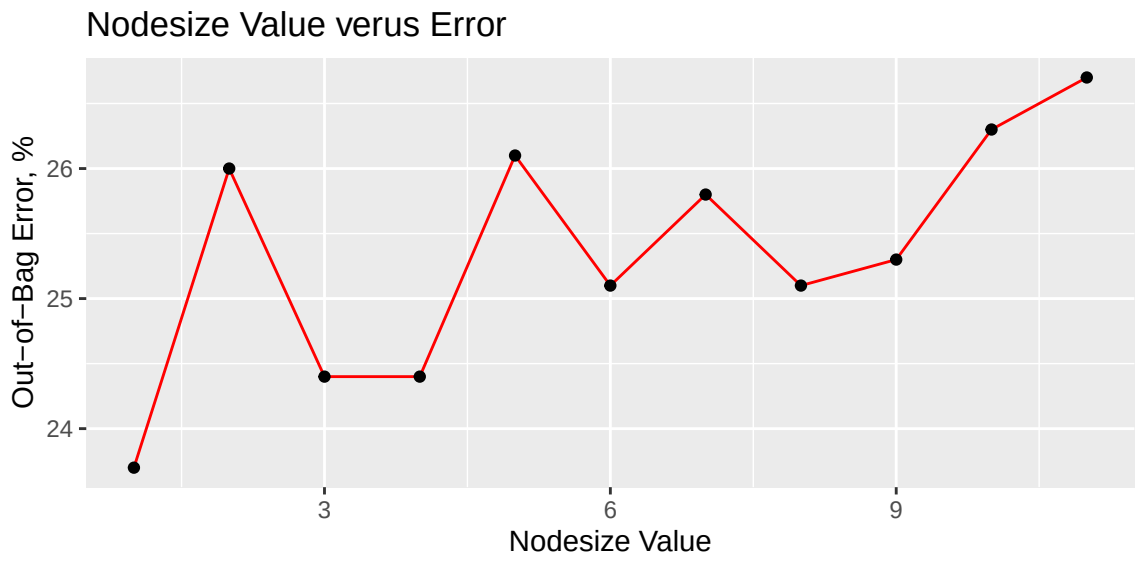


Random Forest Analysis

The most important tuning parameter with regards to random forest is mtry - the number of randomly selected predictors to use when building each individual tree. Greedy descent was performed on mtry, starting at mtry = 1, and adding +1 at each iteration until doing so no longer improved the out of bag error. This analysis, and the following on nodesize, were done using a 10 000 row subset of the original data set in order to decrease computation time. Models with an mtry value of 7 performed slightly better than those with the default parameter of 9. No analysis was done on higher values of mtry.



A greedy descent was also done on the maxnodes parameter - the maximum number of terminal nodes in each tree. However, larger values or maxnodes took too long to compute and analysis was abandoned. Nodesize - the minimum size of terminal nodes, was found to be a better tuning parameter to modify. Trying different values for nodesize revealed that the default value for classification, 1, gave the best out of bag error.



Class weights were added in an attempt to improve the model. My hypothesis was that weighing the harder-to-predict classes more heavily would improve the classification error - it did not. For completeness, I tried weighing those classes more lightly compared to the easier-to-predict classes. This also did not have any substantial effect on the error.

Variable selection was evaluated using the rfcv() function with a reduced data set. It was found that eliminating variables increased the out-of-bag error. It is worth noting that these reduced models would theoretically have lower variance but increased bias. It is also worth noting that the increase in error resulting from variable elimination was larger when fitted on a larger data set. The reduced model with 42 predictor variables, and the full model with 83 variables, were then fitted on the full data set (complete cases only).

Results

After all the analysis, two random forest models were fitted with mtry set to 7. The first model was fitted over all variables and gave an out-of-bag classification error of 10.01%. The second model was fitted over the 42 variables marked most important by the random forest algorithm in the first model. It achieved an out-of-bag classification error of ##%. The easiest class to classify was Bangkok. The most difficult were Hong Kong and Istanbul, which often were mistaken for each-other. Sidney was also misclassified often as either New York or Paris. Overall, the random forest model performed very well and with minimal adjustments from the default parameters.