DS-GA 1001: Introduction to Data Science

Predicting NYC Airbnb Cleared Prices

Anhthy (Ant) Ngo (`an3056`), Eelis Virtanen (`ev933`), Aren Dakessian (`agd387`)

New York University

December 22, 2019

Abstract

For Airbnb, host price formulation plays a large role on the dollar amount of services fees that are accrued. Service fees incurred by both parties of the transaction are one of Airbnb's main sources of revenue. The purpose of this paper is to propose a proof-of-concept ML model that will help Airbnb accurately predict listings "optimal" prices within one of their largest markets, New York City. Additionally, accurate pricing across listings balances affordability for users and profitability for hosts - thus enhancing user experience altogether.

1 Business Understanding

Recent technology advancements have allowed for greater connectivity and as a result fueled a "sharing economy" that allows buyers and sellers to distribute goods and resources. Airbnb is one of the many companies taking advantage of the growth of the sharing economy. Airbnb operates as an online marketplace that allows travelers to browse and book lodging worldwide. Similar to how ride-sharing services are taking over the taxi industry, Airbnb is quickly taking over the hotel industry. In fact, Airbnb has over 5 million listings worldwide and 49% of users used it as an alternative to hotels in 2016 [1]. The business model is a multi-sided marketplace that connects travelers with host and experience providers. Airbnb's revenue is generated primarily from booking service fees. Generally, guests are charged around 6-12% depending on the price of the reservation and hosts are charged around 3% to cover guest payment processing [3]. Optimizing rate of bookings will increase Airbnb's revenue even when service fee becomes minimal. Our teams' goal is to accurately predict cleared listing prices in New York City. By predicting prices accurately, hosts are more likely to sell their listings. Airbnb pricing is important to get right, particularly in big cities like New York, where there are an overwhelming amount of listings and subtle pricing mishaps can make a significant difference between a closed sale or not. Pricing is both an art and a science, namely, prices that are too high will not be booked and prices that are too low will miss out on potential income. To predict Airbnb cleared listing prices, we will utilize historical Airbnb listings data with the goal of extracting useful information to tackle the business problem of optimizing booking rates. By data mining the historical Airbnb listings in New York City, we can identify the most important features that consumers seek when looking for lodging and build a model that will predict the "sweet spot" price. By evaluating different model performances and choosing a "best" model, Airbnb can utilize predicted prices as recommendations to their hosts. Newer hosts who are uninformed about the current market can benefit from the price recommendation, while experienced hosts can use the suggestion as a reference. The goal of the recommendation is generating more informed pricing decisions that will ultimately lead to higher profits for Airbnb and the hosts. The model we provide is a proof of concept, that aims to assess the value and feasibility of our price prediction before Airbnb can choose to implement it.

2 Data Understanding

2.1 Data Collection

Our data is collected from InsideAirbnb, which is an independent, non-commercial group that web-scrapes Airbnb listings, reviews, and calendar data from multiple cities worldwide. For our purposes, we will be looking at data for New York City and primarily focus on the `listings.csv` file, which contains Airbnb summary information and metrics for listings in New York City (i.e., geographic, host, review information) that will be useful in predicting our target variable `price`. Price is defined as the listing cost for a single day and only prices which were "cleared" (booked) were included in the dataset. This would aid in removing outlier prices made by uninformed hosts. The original data was scraped on 9/13/2019 and contains 48,377 unique NYC listings.

2.2 Selection Bias and Reliability

Since our data is collected from a secondary source, we cannot confidently verify the dataset's validity. The dataset contains listing information taken at a snapshot (9/13/2019), therefore hosts who do not keep their price calendar updated will still be incorporated into the data set. Additionally, fraudulent listings that have not been identified by Airbnb will also be incorporated into the dataset, however InsideAirbnb suggests that this subset is small and will not affect analysis [2]. There is an additional bias in our dataset due to the fact that the prices are assumed to be "optimal". For example, uninformed hosts can price listings well-below or well-above the current market value. However, we assume the prices to be “optimal”, induced by the supply and demand characteristics of the current market.

3 Data Preparation

3.1 Removing Irrelevant Features

There were several steps that were performed to properly prepare/clean our dataset for the modeling phase. The first thing we did was drop columns that were clearly irrelevant for predicting our target variable `price`. Some example features include `country`, `scrape_id`, `listing_url`, `thumbnail_url`, etc. A subsequent issue we had to address was features with a significant amount of missing values. Specifically, we found six columns that had 50% or more missing values, and removed those features altogether. Additionally, we removed rows with null values that only accounted for less than 1% of the total dataset since doing so would not impact our analysis. Currency-based features that were originally encoded as strings were converted to floats in order to have the right data type in place for the model.

3.2 One Hot Encoding

The next action item was to consider categorical features with a large number of unique values which needed to be binarized. To avoid feature explosion, we chose to encode binary values for categorical variables that had unique values of less than 20. Our aim was to find a viable trade-off between feature explosion and maintaining the power some categorical variables have (i.e the listing is in Manhattan as opposed to Staten Island). To avoid overfitting, we broke out the neighborhood feature to only contain the major boroughs of NYC. The same process was conducted on `listing_type`. Additional features that were encapsulated in arrays (such as `amenities`) were converted to their respective columns and one-hot encoded.

3.3 Handling Outliers

Handling outliers was a key step in our data preparation process. Most features were skewed and did not resemble a normal distribution at all. In order to remove the extreme price deviations that would adversely affect our price-prediction scenario, observations above the 98th percentile and below the 2nd percentile were removed. Additionally, in order to fulfill the “normality” assumption, we opted to log-transform our target variable `price` (which was originally heavily right-skewed), which resembles more of a Gaussian distribution after the transformation. We opted for this transformation because we ultimately used MSE as the loss metric.

3.4 Feature Selection/ Imputation

In order to treat missing values, the median was imputed within each fold in cross-validation in order to prevent occurrences of data leakage.

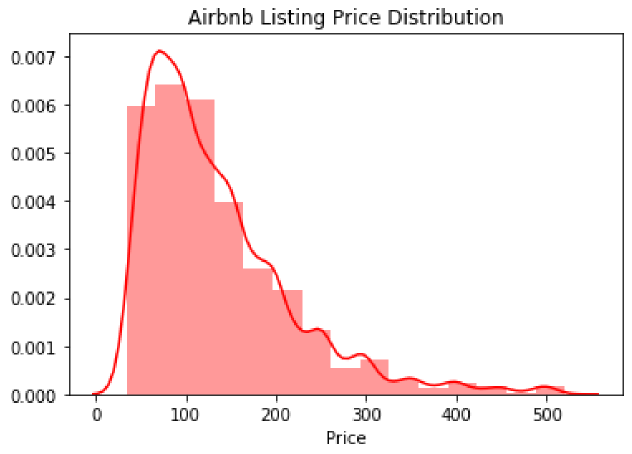


Figure 1: Left-skewed distribution for price feature.

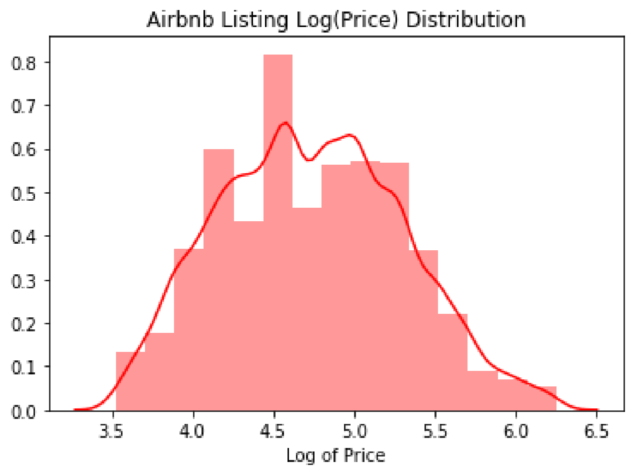
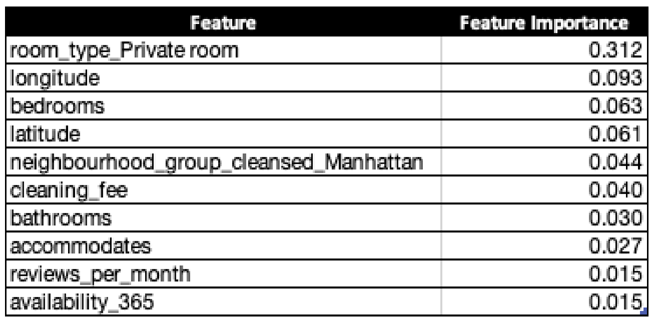


Figure 2: Transforming to the log of price will meet the normality assumption.

The original dataset contained 300+ features, which is why feature selection was incorporated in the data preparation to maximize the removal of irrelevant features while minimizing the variance lost. Utilizing sklearn feature importance and setting an arbitrary cutoff of 0.5%, the number of variables were reduced to ∼10% of the original amount. Specifically, we found that we were able to account for ∼80% of the feature importance with just 23 of the original 300+ features.



Feature	Feature Importance
room_type_Private room	0.312
longitude	0.093
bedrooms	0.063
latitude	0.061
neighbourhood_group_cleansed_Manhattan	0.044
cleaning_fee	0.040
bathrooms	0.030
accommodates	0.027
reviews_per_month	0.015
availability_365	0.015

Table 1: Top 10 features from feature selection.

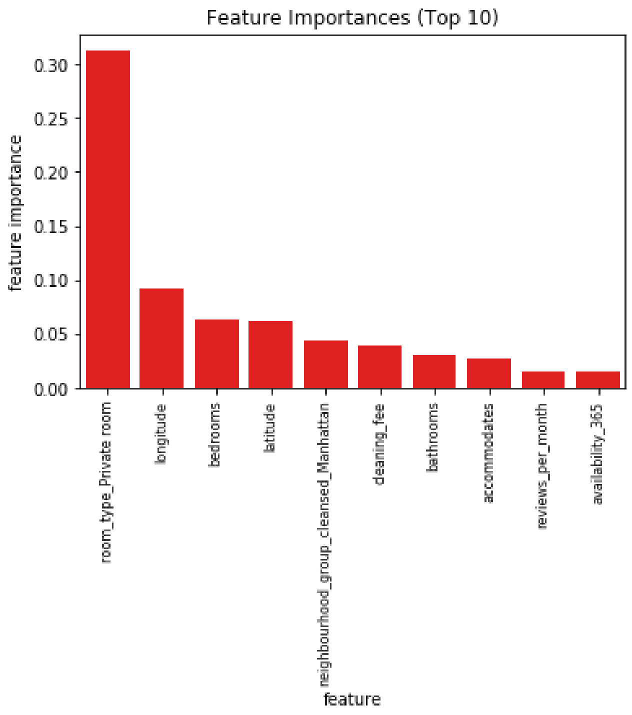


Figure 3: Top 10 features from feature selection.

4 Modeling & Evaluation

4.1 Evaluation Metric

Before implementing the baseline model, one important question to address was to decide which loss function we would evaluate the models according to. Various options exist such as mean-squared-error (MSE) and mean-absolute-error (MAE). Different loss functions can be used to achieve different goals from a practical and theoretical perspective. For example, MSE will penalize outliers more than the MAE. Similarly, from a probabilistic perspective, the MSE assumes that the errors are distributed as standard normal random variables. In our case, the choice of the loss function was influenced by the underlying business objective. The goal was to predict the market-clearing price for a typical range of prices (∼\$30-\$300 USD). In our case, it was decided that large prediction errors would be quite problematic since large discrepancies would not likely be able to clear in the market. On the other hand, small deviations were assumed to not be as problematic since there is perhaps more noise in supply and demand characteristics within smaller price ranges and small errors could still clear even if they were off by a little bit from their "optimal" amounts. Hence, the MSE was used in order to achieve the objective of trying to penalize larger errors with more vigor than say a MAE loss metric which would have penalized them less.

4.2 Baseline: Linear Regression

The modeling task was to use the data that had been pre-processed, cleaned and engineered in order to predict the clearing price for Airbnb on a particular date. Our baseline model was chosen to be the linear regression, which is one of the simplest models available and will generally not overfit the underlying data. The primary assumption is that the response variable can be modeled as a linear combination of the features in an additive manner. This can potentially be problematic if the data is highly non-linear but given that we are predicting prices for Airbnb this assumption does not seem completely unreasonable. See Figure 3 for the 10-fold cross-validation results for linear regression performed on the dataset. We can see that the Linear Regression performed quite well with an average MSE of 0.13 across all folds with a small standard deviation. The fact that each fold gave a similar MSE gives us some confidence that we are not overfitting the model on the data. The task for the remaining section will be to improve the baseline results.

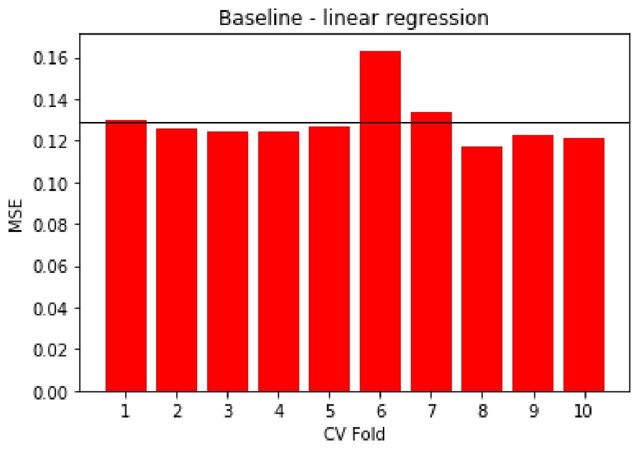
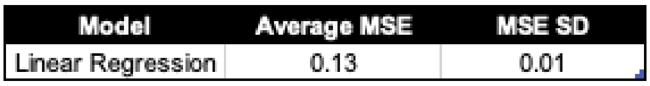


Figure 4: 10-fold cross-validation linear regression MSE.



Model	Average MSE	MSE SD
Linear Regression	0.13	0.01

Table 2: Top 10 features from feature selection.

4.3 Alternative Models

A total of five different models were used in order to improve the results over the baseline. These models can be categorized into roughly two different approaches: regularized linear models and non-linear models.

4.3.1 Regularized Linear Models

Regarding the linear models, the elastic net was a natural candidate as the model to follow the linear regression. The advantage of the elastic net was that it would allow us to regularize the linear model and prevent potential overfitting which should reduce the expected generalization error. Within the linear domain, we also evaluated a model which first implemented principle component analysis on the initial features. The rational was due to the fact that some features where correlated so performance could potentially be improved by using PCA in order to get a set of orthogonal and non-correlated features which could present clearer signals to the model (see Appendix for feature correlation matrix).

4.3.2 Non-linear Models

Regarding the non-linear models, random forest and gradient boosting were used in order to see if non-linear models could achieve higher performance than the earlier linear models. Finally, K-nearest neighbors was used which effectively performs a look-up on the training data and computes the average of the K nearest data points.

4.4 Experimental Framework

For each model, we ran grid-search in sk-learn in order to find the optimal hyper-parameter combination. For each configuration, we ran 5-fold cross-validation in order to add more confidence to the results (only 3-folds for Random Forest and Gradient Boosting). Then, the hyper-parameter configuration with the best performance was chosen to be the final model for that specific model. The best performing model amongst all of these optimally tuned model candidates would be then chosen to be the model that would be used to solve the business problem and its performance would be evaluated on a hold-out data set in order to get a final estimate of its performance.

4.4.1 Hyper-parameter tuning

Regarding the specifics, one consideration was whether to use grid search or random search in order to tune the hyper-parameters. One argument for random search was that we were unlikely to be able to pick good values based upon our own intuition hence random grid search could have a better chance of randomly picking a good configuration. Similarly, in grid search, we would likely be spending a lot of computing resources enumerating through a hyper-parameter space for values that are nowhere near the optimal ones which could result in a lot of downtime. Due to this, a random grid search through a wide predetermined range was used initially.

However, in practice each iteration itself was taking a long time using our computing resources which meant that the random search was not searching enough combinations so that we could have confidence in the results. Hence, an iterative grid search method was utilized where we initially guessed reasonable combinations and kept on tweaking them within a small range until future performance gains were minimal. This was not the optimal solution and seems contradictory in light of the earlier arguments, however given the computing constraints, it seemed to be the most reasonable approach.

4.4.2 Insights gained from experimental framework

Another insight was how some of our modeling experiences cycled back into our data preparation process in order to create an iterative loop. The main insights were to take the log of the prices in order to make their distribution more normal due to the right skew of the price distribution. Similarly, we looked at the distribution of each feature in more detail in order to examine whether each feature distribution made sense. For example, since we are looking at predicting the price for a typical Airbnb customer, does it make sense for a listing to have 20 bathrooms for the apartment? Hence, a lot of time was spent removing outliers from the features which were hurting the MSE performance.

4.5 Results

4.5.1 Model Comparison

It can be seen by Figure 4, that the linear models performed essentially the same as the baseline. Gradient Boosting performed the best with Random Forest performing almost as well. Finally, KNN performed the worst by a considerable margin with the rational that we probably had too few instances relative to the dimensionality of our problem.

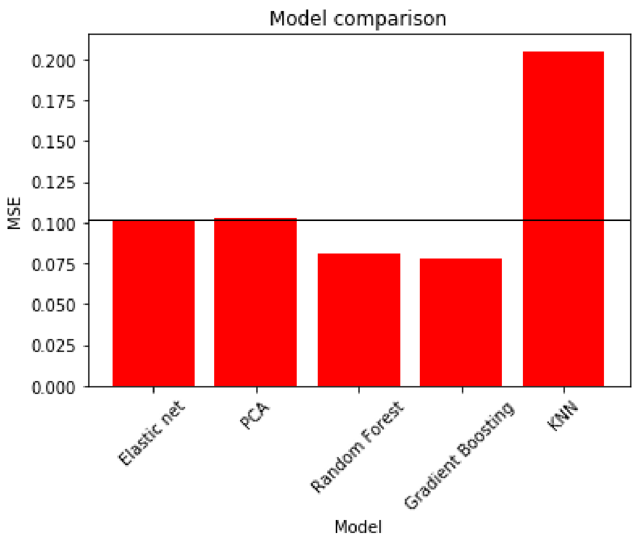


Figure 5: Model MSE

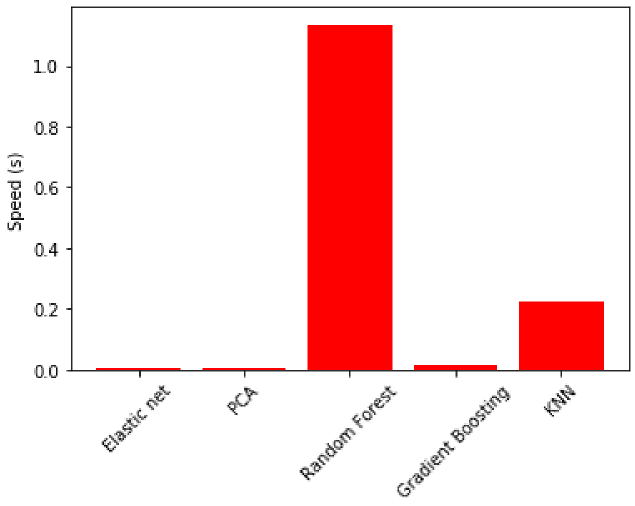


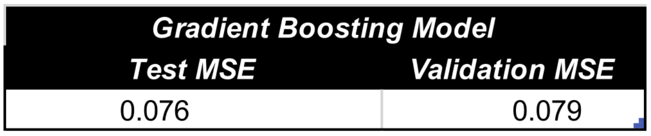
Figure 6: Model MSE

4.5.2 Prediction Speed Comparison

Additionally, the time to predict the test instances was evaluated. From Figure 5, it is clear that the Random Forest Regressor is by far the slowest taking over 1 second in order to make a prediction. This may be due to using too many estimators or having too much depth in the model. Irregardless, we can see that Gradient Boosting which was the best model from an MSE perspective had an acceptable prediction time.

4.5.3 Best Model

Based upon the results the gradient boosting performed the best with a test MSE of 0.076 and a validation MSE of 0.079. This model was selected to be utilized in the production environment.



Gradient Boosting Model	
Test MSE	Validation MSE
0.076	0.079

Table 3: Gradient Boosting MSE

5 Deployment

5.1 Deployment of Results

Now that we have a model, the next step is to integrate our results into Airbnb's production servers. Collaboration with the IT Team, and Data/Software Engineers is a crucial component to efficiently getting this task done. Before launching into production, we would first evaluate our system for a few weeks on live data in a testing environment. Once we have confidence in our solution, we would implement this solution in a language like C which would run faster in a real-time. Training the models would be pushed to a backend system whereas solely the trained model would be deployed on the frontend in order for the service to best fast enough for the host. The maximum prediction time for the model should be of the order of 0.1 seconds. We saw in the previous section that Gradient Boosting achieved this easily for the entire training set so the prediction of a single instance should be very fast.

5.2 Monitoring and Evaluation

There should be some outlier detection processes incorporated into the production system to monitor the modeling portion. Hosts obviously have the freedom to list at whatever price they want, but if a listing clears at a price that deviates (in either direction) past some arbitrary threshold relative to our suggestion, said listings should be flagged by the system for some further automated inspection to see if there was something the model could have caught that it didn't.

Another factor to consider in a production environment is how often we re-train the model to avoid concept drift. This is a question that would be worth taking more time for when this generalizes and things like seasonality can be addressed/incorporated into the model. However, at a high level, it would make sense to re-train in a way that catches the shifting dynamics of the Airbnb environment between weekdays and weekends. When not incorporating holidays and seasonality, the most simple (yet proven to be effective) segment to re-train on would be during the week vs. weekend. Having multiple models could be an option as well.

Due to the simple numerical interpretation of the model, standard error metrics can be used as a baseline for evaluation. However, this is a scenario in which interpretation of error terms should be adjusted within the context of the scenario. For example, it's probably pretty rare that the host ends up using the exact price that our model suggests. However, if we can get within ± 5% of our suggested price, we know that we gave them a fair baseline to operate off of, which is exactly what we are trying to achieve. Everything we would want to know can be summed up in two tables.

ListingID	User_Age	Predicted	Actual	%Difference	TimeToBook

Table 4: Sample Error Table for Given Date

Date	SumPredicted	SumActual	%Difference

Table 5: Aggregation of Error Table (Snapshot)

5.3 Issues, Risks, Ethical Considerations

If they were to deploy this system, one risk is the implicit fact that we are "trusting" the user. For example, if the actual market price for a listing is around $300 and a host lists it for $50 and it clears, the model would assume that $50 is within the ideal market price when it trains new data for a given day. There isn't really a great way to mitigate this risk, since outlier detection practices are more geared towards new listing prices being significantly different from what our model suggests. We would have to "bake in" an assumption, that on average, hosts will not list outlandish prices and have at least minimal understanding of what market price of their listing should be around. On average, this should wash the sub-optimal listings and help our model train on data more reflective of realistic market conditions. More technically, there is an additional bias in the model due to the target variable price being relied completely upon user-input.

One underlying assumption we made was to assume that the hosts have been in operation for long enough such that features like "reviews per month" are informative. However, new hosts will not have such data, presenting a cold-start problem since we would still like to make predictions for new hosts. One option could be to develop additional models which would not use features that new hosts do not have and use them in such cases.

The biggest ethical consideration to be made with this product has to do with the underlying goals of Airbnb as a company. Say, for example, Airbnb is in a stage where they care more about maximizing volume instead of a fair experience for the host. In this case, we could tune the model to "low-ball" the price predictions, and thus get more listings booked at a price that is sub-optimal relative to the user. This would allow Airbnb to gain market share in the booking market whilst hosts would be getting slightly below the market price. Hence, the interests of Airbnb may not be exactly aligned with the hosts.

6 Conclusion and Future Considerations

We demonstrated a proof-of-concept about a price prediction system that could be deployed at Airbnb to improve its business. This paper touched upon all aspects of this solution from data preparation up to deployment. Based upon the favorable results of this paper, future work would be aimed at generalizing these results towards a more realistic environment. This environment would include all possible dates as well as more flexible durations since our current solution is only valid for one particular date. Other important incorporations such as seasonality, holidays, and weekends could be included into the system since these features would likely have a large influence on the clearing price. One possibility could be to have multiple models trained for specific dates (holidays, high season, etc) that would be utilized depending on what dates the host was looking to rent their property on. Engineering additional features such as time elapsed from listing generation to booking time could be a powerful metric that would allow for deeper analysis on price optimization.

References

[1] Airbnb by the Numbers: Usage, Demographics, and Revenue Growth. (n.d.). Retrieved from https://muchneeded.com/Airbnb-statistics/

[2] Inside Airbnb Adding data to the debate. (n.d.). Retrieved from http://insideAirbnb.com/

[3] How Airbnb Makes Money. (22, July, 2019.). https://www.investopedia.com/articles/investing/112414/how-Airbnb-makes-money.asp

Appendix

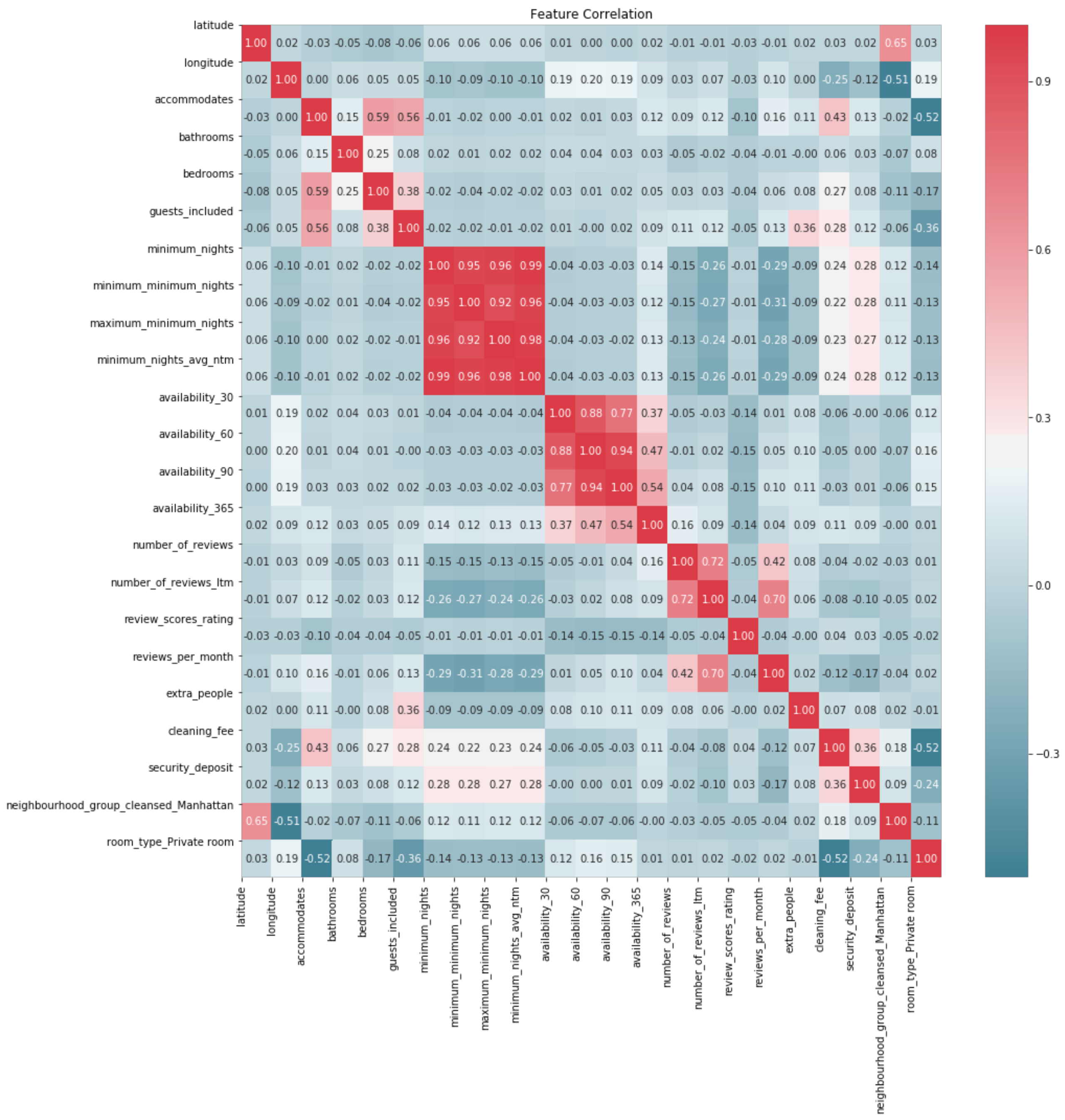
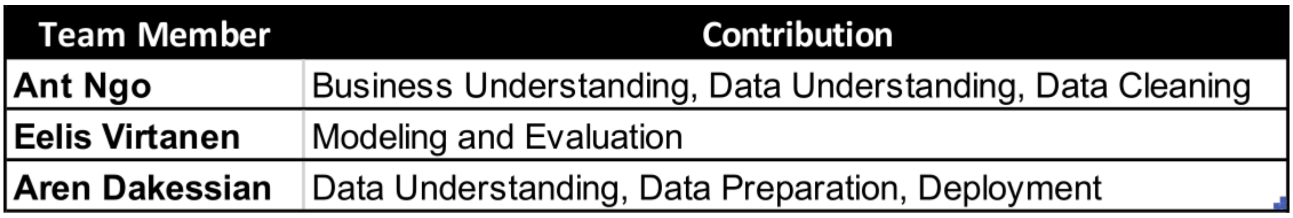


Figure 7: Feature correlations



Team Member	Contribution
Ant Ngo	Business Understanding, Data Understanding, Data Cleaning
Eelis Virtanen	Modeling and Evaluation
Aren Dakessian	Data Understanding, Data Preparation, Deployment

Table 6: Team contributions