ORIE 4741

Fall 2017

Project Final Report
Optimization of Airbnb Revenues - New York

Kerou Gao (kg486)
Kartikay Gupta (kg477)

Cornell University, Ithaca, NY

December 5, 2017

Acknowledgement: We would like to thank **Mr. Jialin Liu**, a PhD student in the Electrical and Computer Engineering Department, for providing valuable inputs in this project.

Introduction

Airbnb is an online hospitality service that connects people, who are willing to rent or lease their extra house space, with people who are looking for short term lodging. This lucrative model enables people to find accommodation at reasonable prices in lieu of going for expensive hotels. Airbnb has garnered immense popularity over the past few years and has over 3 million lodging listings in 65 thousand cities and 191 countries.

This project is aimed at finding a pricing strategy that can maximize hosts' revenues, which can be used for guiding both new and old owners to set prices for their properties. Our methodology is listed below:

- Data analysis: Generating machine learning model to predict occupancy(outputs) for each listing using price and other characteristics of houses(inputs).
- Optimization: Applying this model to each listing and find the best price that maximize host's revenue. Recall: revenue = price*occupancy

Data Set Description and Feature Engineering

The data set that we use for training our models has been obtained from "Inside Airbnb". We chose New York City for our analysis as it contains comprehensive information of 44317 Airbnb listings. There are in total 93 features that are provided in the data set. These features give us a plethora of information ranging from:

- Listing Location - Coordinates, Neighbourhood, Street
- Listing Characteristics - Room type, Amenities, Number of Bedrooms, Annual Availability, Transit proximity information etc.
- Listing Reviews - Statements by guests, number of reviews, review scores etc.

Since "Occupancy" is not in our data set, we made an assumption that Occupancy and average-monthly-review-number have positive correlation. Therefore, we replace occupancy by average-monthly-review-number, but still call it "Occupancy" in our report.

For our preliminary analysis, we extracted all the continuous and discrete values from our data set and created a new data frame from these values. The new data set consisted of 31 features in total. Out of these features, we used data visualization tools such as scatter plots (Figure 1,2) and bubble plots (Figure 3) to select features that displayed correlation with the listing occupancy. In addition, we also added an offset to get a better fit to our data.

The feature space that we used consisted of:

- price: price of each listing
- accommodates: number of persons that can be accommodated in a listing

- bathrooms: number of bathrooms
- bedrooms: number of bedrooms
- beds: number of beds
- guests included: number of guests included in a listing price
- latitude & longitude: coordinates of the listing
- review score: based on tenants' review, range in [1,100]
- cleaning fee: cleaning fee for each listing
- security-deposit: security deposit for each listing, will be returned when check out

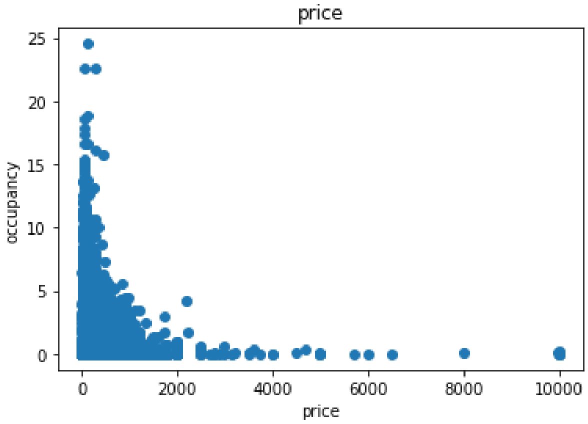

Figure 1: Correlogram Occupancy-Price

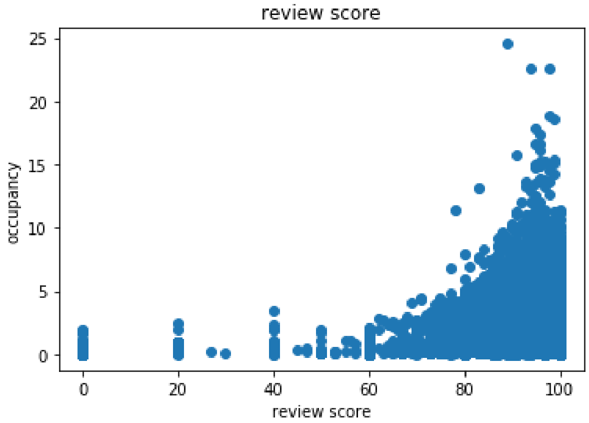

Figure 2:Correlogram Occupancy-review score

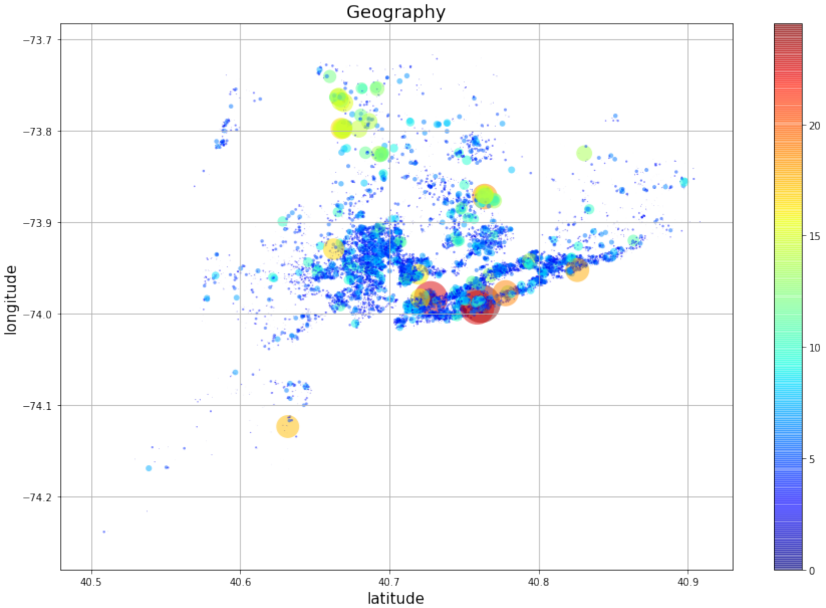

Figure 3: Bubble plot depicting listing price as a function of location

In the next step, we selected categorical data like "Property type" ,"Amenities" and encoded them using **One-Hot Encoding** method. In addition, we decided to **delete** the missing values in our data set as they were quite less in number and therefore deleting them would cause no significant truncation of the data set. Thus our final data set consists of 43825 rows and 196 columns.

We used **"train_test_split"** to separate our data into a training set and a training set. 10% of the data was kept aside for testing purposes and was not touched during the individual model training phase.

Regression Methods

Since our output space Y = R, our course of action first involved using regression techniques to predict the average number of monthly reviews. In tandem with the concepts taught in lectures, we tried 3 different regression techniques on the training data. To gauge the performance of our methods, we took the mean of our output space, 1.56, as our benchmark and then compared the RMSE obtained against this benchmark:

Linear Regression

In case of linear regression, the tunable parameter that we have is the degree of the polynomial that we fit to the data. From our analysis of the initial linear regression coefficients, we identified that "price", "availability" and "review_scores_rating" had a major impact while predicting the average number of monthly reviews. Consequently we decided to expand our feature space by including higher degrees of these features. The results of this inclusion have been shown in the table below:

Table 1: Polynomial Fitting Results

Feature Space	**Training RMSE**	**Test RMSE**
Baseline	1.19325	1.19834
Baseline + price^2	1.19321	1.19816
Baseline +rsr^2	1.19054	1.19675
Baseline +avail^2	1.18106	1.18727
Baseline +price^2 + price^3	1.19319	1.19798
Baseline +rsr^2 + rsr^3	1.17984	1.18905
Baseline +avail^2 + avail^3	1.18105	1.18710

avail = "availability", rsr = "review_scores_rating",
RMSE = Root Mean Square Error

Thus, we observe that having a higher degree polynomial certainly results in better predictions as the RMSE value decreases. But the decrease is quite minimal and we still have high window of error around our mean of 1.56 which does not justify the use of a higher degree polynomial. Thus we decide to progress ahead without appending any higher degree features to our covariate space.

Ridge Regression

To obtain the optimal value of the **regularization parameter** λ, we adopted a **10-fold cross validation** approach on our training data and plotted a corresponding validation curve as shown below (this method is also used for Lasso Regression):

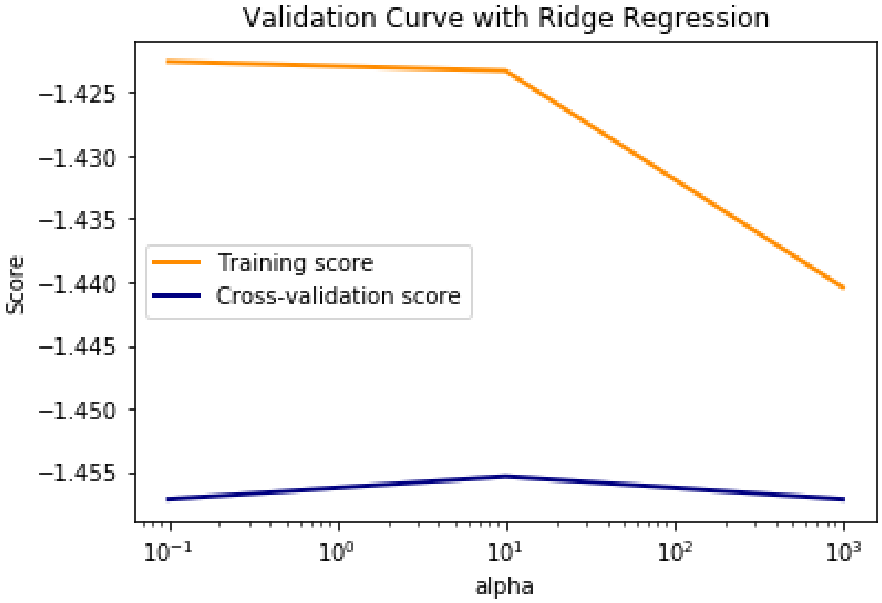

Figure 4: Validation Curve for Hyperparameter Tuning

The negative value on the y-axis denotes the negative mean sqaured error. Thus, we pinpoint a value of 10 for λ. Using this value, we fit a model to our training data and determine the performance of the classifier on the test set.

Lasso Regression

To obtain the optimal value of the **regularization parameter** λ, we again adopt a **5-fold cross validation** approach on our training data and arrive at a value of 0.005 for λ. Using this value, we fit a model to our training data and determine the performance of the classifier on the test set.

The performance of the three models in predicting the average number of monthly reviews is shown in the table below:

Table 2: Performance of Different Regression Methods

Technique	**Training RMSE**	**Test RMSE**
Linear Regression	1.1932	1.1983
Ridge Regression	1.1980	1.2034
Lasso Regression	1.2043	1.2074

From the above results, it is evident that linear regression has the best performance among all the methods. But still the RMSE obtained is far from ideal and the poor performance of regression techniques warrants the use of advanced prediction techniques which have been discussed in the next section.

Ensemble Learning Methods

In general, Ensemble techniques improve predictive performance by creating a strong classifier from a bunch of weak classifiers. The resulting models offer the advantage of reduced variance and reduced bias[1]. Thus, in order to obtain a more robust model, we tried three advanced ensemble methods–“Random Forest”[2], “XGboost”[3] and “AdaBoost”[4].

Random Forest

As mentioned above, ensemble learning methods combine a group of weak learners to form a strong learner. In this case, **“Decision Trees”** are the weak learners and they come together to form Random Forest which is a strong learner and is an example of bagging technique. In general, higher the number of trees in the forest, greater is the accuracy of the model[5]. Random Forest can be used for both classification and regression. Let us consider the case of regression since that is the main focus of our project. From a given dataset, cases are taken at random with replacement to create a subset. Typically 66% of the data is used to create a subset. A decision tree is then trained on this dataset and the value of the terminal node is recorded. This process of creating a random subset and fitting a decision tree to it is repeated and the value of terminal nodes is recorded in each case. The randomness in selecting data for training the model ensures that the predictive model has low variance. The final prediction is the average or the weighted average of the values obtained from the group of decision trees [6].

XGBoost

XGBoost stands for “Extreme Gradient Boosting” and is an implementation of gradient boosted regression trees which are designed for speed and performance. Gradient boosting framework generally involves three elements: Loss Function, Weak Learner and an Additive Model[7]. The loss function generally depends on the type of the problem being solved and for our regression setting, a squared loss function is used. For weak learners, decision trees are generally used and these are constructed in a greedy manner. Specifically regression trees are used that output real values for splits and whose output can be added together, allowing subsequent models outputs to be added and “correct” the residuals in the predictions. The next step involves adding decision trees one at a time and typically a gradient descent procedure is adopted to minimize the loss while adding trees, hence the name gradient boosting. This addition of trees is continued until a fixed number of trees have been added or the error has been reduced to an acceptable threshold and no further reductions are observed.

AdaBoost

AdaBoost is a short form for "Adaptive Boosting" and is quite similar in working to the random forest in the sense that the weak learners in this case are generally decision trees. However, unlike random forest, the dataset is not randomly split into subsets. Rather the data is repeatedly modified by applying weights to each of the training samples[8]. The initial iteration involves assigning a weight of (1/N), where N is the number of training samples, to each sample and a weak learner is fit to the model. For successive iterations, the training samples that were incorrectly predicted have their weights increased and the weights are decreased for the samples that were correctly predicted. the next step involves applying the learning algorithm to the reweighted data. This process is repeated and the final prediction consists of the average or the weighted average of the predictions obtained from the iterations.

A summary of the performance of the three Ensemble Learning methods is shown in table 4 below:

Table 4: Performance of Ensemble Learning Algorithms

Technique	Training RMSE ($)	Test RMSE ($)
Random forest	0.543	0.978
XGboost	0.569	0.983
Adaboost	0.611	1.06

Compare to regression method, these three Ensemble Learning method improve accuracy by 20%. Now prediction error is slightly below 1.

Optimization and Result

When we have the model to predict occupancy of listing, the next thing to do is to maximize revenues by changing price. We picked 8 listings from the test set, simulating revenue of each listing by changing price(range in [0, 2*original price]) and picked the best price that gave the highest revenue. The result for "Random Forest", "XGboost" and "Adaboost" are listed in table 5:
Our result reveals several patterns:

1. The results of all three model are in same direction even though the actual values vary. This validates our optimization methodology and establishes its consistency.

2. Almost all of the optimal prices hit their upper bonds, which means that our model tends to bring up revenues by increasing prices and decreasing occupancies. Unfortunately, it seems to violate Airbnb's principle. One way to think about it is that we are trying to maximize revenues from the hosts' perspective, which can be different from Airbnb's goal which is to increase market share.

3. We highlight unreasonable examples by red in table 5. In these examples, we see that occupancies increase when prices increase. We believe this problem due to inaccuracy of our prediction. All of those examples' original occupancies are lower than 1. The reason for this phenomenon is that our average error is about 1, thereby leading to a huge impact on an output that is below 1. Therefore, our current model cannot be used to predict small outputs. Besides, occupancy and average-monthly-review-number may not have strict linear relationship, thereby leading to more errors in prediction.

Table 5: Optimization Results

Random Forest						
	Old Rev	**Opt Rev**	**Old Price**	**Opt Price**	**Old Occ**	**Opt Occ**
Listing 1	720	981	180	359	4	2.7
Listing 2	339	588	85	169	4	3.5
Listing 3	50	398	115	229	0.5	1.7
Listing 4	120	187	35	69	3.45	2.7
Listing 5	12	53	300	599	0.04	0.09
Listing 6	698	1094	99	197	7	5.5
Listing 7	253	520	75	149	3.38	3.49
Listing 8	90	556	300	599	0.3	0.9
XGboost						
	Old Rev	**Opt Rev**	**Old Price**	**Opt Price**	**Old Occ**	**Opt Occ**
Listing 1	720	897	180	359	4	2.5
Listing 2	339	551	85	169	4	3.3
Listing 3	50	447	115	229	0.5	1.9
Listing 4	120	143	35	69	3.45	2.1
Listing 5	12	21	300	599	0.04	0.03
Listing 6	698	705	99	197	7	3.6
Listing 7	253	467	75	147	3.38	3.18
Listing 8	90	336	300	599	0.3	0.6
Adaboost						
	Old Rev	**Opt Rev**	**Old Price**	**Opt Price**	**Old Occ**	**Opt Occ**
Listing1	720	767	180	359	4	2.13
Listing2	339	477	85	169	4	2.8
Listing 3	50	385	115	229	0.5	1.68
Listing4	120	149	35	69	3.45	2.3
Listing 5	12	313	300	599	0.04	0.52
Listing6	698	1076	99	197	7	5.5
Listing7	253	469	75	149	3.38	3.15
Listing 8	90	667	300	599	0.3	1.13

revenue = price*occupancy
Old Rev = "Original Revenues", Opt Rev = "Optimal Revenues",
Old Price = "Original Price", Opt Price = "Optimal Price"
Old Rev = "Original Occupancy", Opt Rev = "Optimal Occupancy"

Scope for Further Work

The biggest problem of our model is under-fitting. Here is what we think can help improve our models accuracy:

1. Replace output space by real occupancy data. The assumption that Occupancy and average-monthly-review-number have positive correlation may be wrong at the first place. For example, tenants are more likely to write review when they have really good or bad experience, or when the price is high.

2. Consider quality and quantity of picture. Picture of listing plays an important role in tenants' decision making process.

3. Natural language processing. Another aspect that is quite important is the verbal reviews given by the guests for a particular listing as a listing with positive reviews is bound to attract more customers. Thus, inclusion of Natural Language Processing to gather insights from verbal reviews will increase the accuracy of our predictive model.

Other than those listed above, we also need to extend our model to other cities other than New York. We could do it by adding a new feature point to which city this listing belongs.

References

[1] *Ensemble Learning to Improve Machine Learning Results* [Online]. Available: https://blog.statsbot.co/ensemble-learning-d1dcd548e936

[2] *Random Forest* [Online]. Available: https://en.wikipedia.org/wiki/Random_forest

[3] *XgBoost* [Online]. Available: http://xgboost.readthedocs.io/en/latest/model.html

[4] *AdaBoost* [Online]. Available: https://en.wikipedia.org/wiki/AdaBoost

[5] *How the Random Forest algorithm works in machine learning* [Online]. Available: http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/

[6] *Random Forest* [Online]. Available: http://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics

[7] *Gradient Boosting* [Online]. Available: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/

[8] *Adaboost-Scikit* [Online]. Available: http://scikit-learn.org/stable/modules/ensemble.html#adaboost