How Much are You Expecting to Get from Your Airbnb Listing?

Yizhen Luo and Xiao Bai

Abstract— **Based on Airbnb listing data in the city of Beijing, China, this project studies an optimal model for accurately predicting the monthly revenue of a listing in the Chinese market given the features of the listing, through Machine Learning algorithms. The project aims to provide a tool for the hosts and other stakeholders to have a better understanding of what makes a revenue-generating listing, and how to possibly attune the original characteristics of a listing to make more money. The study starts with data exploration and preprocessing. Then various types of machine learning models, including linear models, decision trees, and ensemble models are built to find the model with the best results of predictions as an appropriate solution. The application of the model is illustrated through an interactive interface then. The project concludes with a discussion about more potential analysis to be completed in the future research.**

I. INTRODUCTION

During the past few years since 2014, the unparalleled prosperity of emerging sharing economy has been coming into our sight; especially in China, online sharing services like Didi Chuxing and Meituan-Dianping have been transforming different aspects of peoples lives remarkably. Excited about this impressive phenomenon, we choose Airbnb, a global leader in accommodation sharing which entered the Chinese market in 2015, as the focus of our project. For Airbnb platform operators, as well as Airbnb hosts, one of their primary goals is to improve transaction activities and listing revenues. Therefore, we come up with the main research goal: to find an optimal model for predicting the revenue of an Airbnb listing in China, given its feature inputs. To solve this, we utilize Machine Learning tools to train our models with real life Airbnb dataset, and optimize the models' prediction results. Our ultimate intent is to build an interactive, user-friendly interface that could display predicted revenues for users based on the listing data they feed in.

There has been previous studies applying Machine Learning techniques to Airbnb data analytics. Choudhary et. al [1] study the pricing and availability prediction of new listings with no reviews for hosts and guests based on Airbnb listings data in San Francisco. The models employed included Linear Regression model with RMSE and MPE metrics, and Random Forest, as well as clustering and multinomial Naive Bayes. Calixte et al. [2] explore another direction to predict the likelihood of Airbnb customers (hosts) and guests returning to the platform. The authors employ decision tree with AUC metric to arrive at the conclusion that guests who stay with highly rated hosts and hosts with high recency and frequency will most likely return. Tang and Sangani [3] input textual and image features combined with other information extracted from Airbnb listing data in San Francisco, so as to predict each listings price and the neighborhood it belongs to via SVM classifier. While all these papers work on datasets from the InsideAirbnb.com, the vast majority of them concentrated on Airbnb in the developed market. There has been little literature before on this topic in the Chinese context, which we are exactly interested in pursuing.

II. METHODOLOGY

A. Dataset

The raw dataset which our project bases upon is a compiled list of all the latest Airbnb listings in Beijing (over 20,000 listings updated in Oct. 16, 2018 from http://insideairbnb.com/get-the-data.html), with 89 columns containing the main attributes of each Airbnb listings, including information of the host (host profile picture, host since, host location, host listing number, etc.), the listing information (accommodates, bedrooms, beds, bathrooms, availability, review scores, etc.) and booking requirements (cancellation policies, require guest profile picture or not, etc.), in forms of both discrete and continuous variables. (See Appendix for detailed description of attributes)

Comprehensive data exploration and preprocessing steps were applied to our dataset. We firstly explored our dataset visually through scatter plots, histograms, correlograms to understand the distributions of the variables and their correlation where apparent multicollinearity issue was not discovered. In data preprocessing stage, we firstly dropped all the unnecessary columns, such as thumbnail_url and state, leaving 34 variables (21 continuous-valued variables and 12 discrete ones) from the original 89 variables. To deal with missing values, we checked and dropped certain variable with too many missing values (for example, about a half data points missing), and applied the backfill method to finish imputation. To manage special variables such as time and categorical data, we extracted time index variables from original columns, and created dummy variables out of categorical values. Extreme outliers in continuous-valued variables observed from histograms were also removed.

B. Training Methodology

Nearly 40% of the listings are not making revenue at all in the last 30 days. To correctly identify whether a listing would make any revenue and preven the 40% zero-revenue listings from skewing our prediction, our project is composed of two parts: firstly, to utilize classification models to determine whether a listing would make revenue or not. Secondly, if the listing is predicted to have revenue, we utilize regression/clustering models to predict exactly how much revenue the listing is expected to generate.

In the first step, classification, we firstly transform the monthly revenue variable into dummies (1: with revenue / 0: no revenue), and train our data with five classification models: Logistic Regression, Naive Bayes, Neural Network, Random Forest and AdaBoost. After tuning the parameters to obtain relatively good prediction results through cross validation in each of the five models, we compare the models' prediction accuracy on testing set to determine which model to choose.

In the second step, revenue prediction, we mainly focus on predicting the listings of which revenues are determined to be non-zero in the first step. Therefore, we filter out the non-revenue listing data and train our models with Linear Regression (including Linear Regression, Ridge/Lasso Polynomial Linear Regression, etc.) , Decision Tree, Ensemble methods (Bagging, Boosting and Stacking). We tune the parameters in each model to obtain relatively good prediction results, and compare model performance using Mean Squared Percentage Error metrics (A customized function was defined to calculate the MSPE for each prediction). After obtaining the optimal models in the first two steps, we construct an interactive interface for users to enter listing features and predict listing revenue in real time utilizing Jupyter Widgets.

C. Model Evaluation Metrics

We adopted two different evaluation metrics in the step 1 classification and step 2 classical models on the basis of the nature of our tasks.

During step 1, we utilized accuracy score for our classification task, which is one of the most intuitive and neat measures commonly used for classification problems.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

The accuracy measure represents the ratio of the number of our correct predictions divided by the number of our total predictions, which could precisely tell how a classification algorithm has performed especially when our data set is close to balanced.

During step 2, we chose the MSPE (Mean Squared Percentage Error) metric, which would bring us its special benefit of invariance to scales.

$$MSPE = \frac{1}{n}\sum_{t=1}^{n}\left(\frac{y_t - \hat{y}_t}{y_t}\right)^2$$

Since MSPE tends to remain unchanged regardless of whether scales are multiplied on the data, it saves us the work of normalizing the data in our preprocessing and also adds stability to our model training process.

III. MODEL RESULT

A. Step 1 Classification

In the first-step classification trying to predict whether the revenue equals to zero, the Naive Bayes model with 10-fold cross validation yields scores at 66.26% testing accuracy, whereas the Logistic Regression yields performance at 69.20% testing accuracy with 10-fold cross validation. Neural Network and Random Forest models both do slightly better which yield results at 70.15% and 70.30% testing accuracy. The AdaBoost model yields the best performance among the five models, with 70.34% cross validation accuracy. All the ML model results are significantly better than constant model with c = 0, 1, and random guess. Therefore, we choose the AdaBoost as our first-step model with 70.34% testing accuracy.

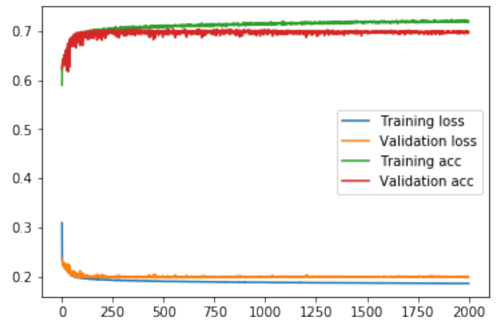


Fig. 1. Loss and accuracy across iterations in Neural Network model

Step 1 Model Accuracy	Training	Testing
Naïve Bayes	66.50%	66.26%
Logistic Regression	69.12%	69.20%
Neural Network	72.89%	70.15%
Random Forest	70.35%	70.30%
AdaBoost	70.83%	70.34%
Constant Model (c = 0)	39.85%	39.17%
Constant Model (c = 1)	60.15%	60.83%
Random Guess	49.96%	49.86%

Fig. 2. Step 1 Model training results

B. Step 2 Revenue Prediction

In the second step, we try to train the model to predict continuous target variable - monthly revenue of a listing. Different from the first step, performances vary greatly across models. Among the Linear Regression models, Polynomial Linear Regression with feature 2 yields the best results under Lasso regularization with alpha equal to 9. The training MSPE is 5.15, with testing MSPE slightly higher at 5.25. We found this result is very similar to Polynomial Linear Regression method under Stochastic Gradient Descent with 10-fold cross validation.

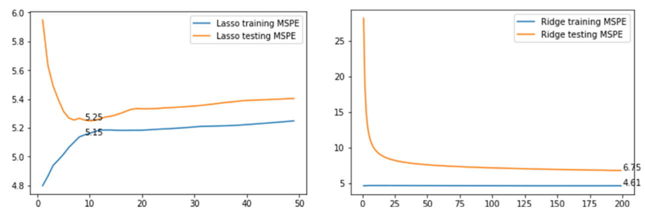


Fig. 3. Lasso and Ridge on different alphas in Polynomial LR

In Decision Tree model, we have initially observed a severe overfitting problem using the default decision tree module in sklearn without tuning any parameters. To resolve the problem, we try to restrict the complexity of the tree through exploring the tuning of three possible hyperparameters: the maximum depth of the tree, the minimum number of samples required in leaf node, and the maximum number of features considered when searching for best splits on each node. In preliminary attempts for each parameter, we drew the plots about the trend of change in training MSPE and testing MSPE over every integer value in an appropriate interval to help judge an ideal range of values for tuning later. Then, we used a 10-fold GridSearchCV to look for the optimal combination of the two parameters chosen(max_depth and min_samples_leaf), which gave us the result of a tree with maximum depth = 10, minimum samples per leaf = 10 here. Under this best estimator model, the training MSPE is about 2.33, while the testing MSPE is about 4.32, which indicates us that the overfitting issue has largely been handled compared to the originally unrestricted tree.

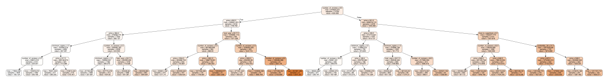


Fig. 4. Decision Tree Visualization

In terms of Ensembles methods, we used Random Forest, XGBoost, and Stacking methods. In Random Forest model, the optimal MSPE output through GridSearchCV (maximum depth = 10, minimum sample leaf = 10) is 2.35 on training set, and 3.43 on testing set. XGBoost model gives even better results after searching the optimal maximum depth as 6, with training MSPE 1.38, testing MSPE 2.19. Stacking these two most outperforming results excels even further, giving us training MSPE 1.34, testing MSPE 2.16.

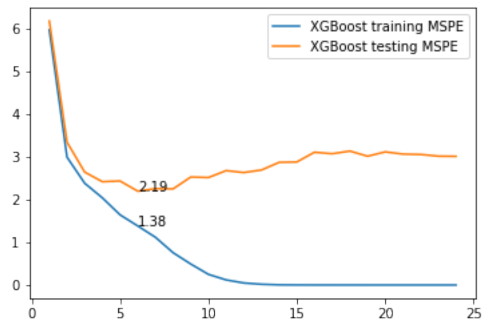


Fig. 5. XGBoost on different maximum depths

Step 2 Model MSPE	Model Parameters	Traing	Testing
Linear Regression	-	8.67	9.91
Polynomial LR	*Lasso, alpha = 9*	5.15	5.25
Decision Tree	*max_depth = 10, min_samples_leaf = 5*	2.33	4.32
Random Forest	*max_depth = 10, min_samples_leaf = 10*	2.35	3.43
XGBoost	*max_depth = 6*	1.38	2.19
Stacking	*Random Forest + XGBoost*	1.34	2.16
Constant Model	*c = mean*	31.94	32.76

Fig. 6. Step 2 Model training results

All the Machine Learning models we used in step 2 perform significantly better than the constant model with c = mean. Among all these models, we decide to choose the top-performing Stacking model as our final model.

C. Model Application

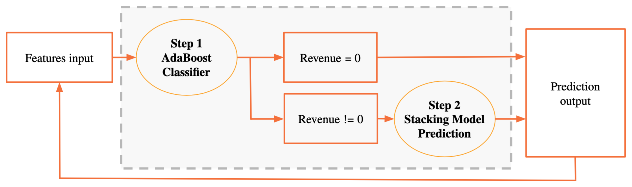
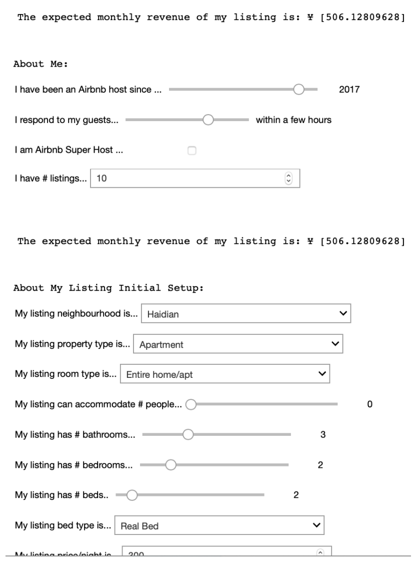


Fig. 7. Model integration mechanism

After we have obtained relatively optimal models for step 1 (determine whether revenue equals zero) and step 2 (determine how much the revenue would be), we have developed an interactive tool (Figure 8) to utilize the two models in real-life scenarios. Empowered by Jupyter Widgets, the tool takes the parameter input from the user, and then predicts and displays the expected revenue in real time. Also, the user can tune the parameters and see the expected revenue change instantly.



```
The expected monthly revenue of my listing is: ¥ [506.12809628]

About Me:
```

I have been an Airbnb host since ... 2017

I respond to my guests... within a few hours

I am Airbnb Super Host ...

I have # listings... 10

```
The expected monthly revenue of my listing is: ¥ [506.12809628]

About My Listing Initial Setup:
```

My listing neighbourhood is... Haidian

My listing property type is... Apartment

My listing room type is... Entire home/apt

My listing can accommodate # people... 0

My listing has # bathrooms... 3

My listing has # bedrooms... 2

My listing has # beds.. 2

My listing bed type is... Real Bed

My listing price/night is... 300

Fig. 8. Interactive page integrating the ML models

For example, in the sample listing in Figure 8, the expected monthly revenue is 506 yuan. Here if the host is an Airbnb Super Host, their revenue could increase to 775 yuan. We can see that at this specific condition, being an Airbnb Super Host could substantially increase the listing revenue by 53%, so the host may wish to be a Super Host in order to maximize economic return.

Similarly for pricing strategy recommendation, in the example, if we increase the price from 500 to 1000 yuan per night, the monthly revenue will increase to 1981 yuan, which means that the listing at this situation deserves a higher price in order to gain higher return. However, if the listing price continues to increase to 2500 yuan per night, the revenue would become 0. This would give the host a clear sense about what might be the optimal price for their listings.

IV. CONCLUSION AND FUTURE WORK

We believe the research output will make significant impact on many. For the hosts, the project will be a practical tool for them to estimate their listings' potential revenue beforehand, and tune their listing features to maximize their return. In large, our project would also cast new light on the understanding of the present patterns of accommodation sharing economy in the Chinese context, which may inspire regulators and market organizers on better resource allocation and efficiency.

Given more time, we foresee our project could be further refined in several aspects.

Firstly, our model could be trained with more listing features for more accurate prediction. The dataset we obtained also contains listing descriptions and reviews, along with the pictures of the listings as well as the hosts profile picture. Text mining, sentiment analysis and image recognition techniques could be used to understand these text and imagery information and further improve our models.

Secondly. A suggestion system for feature tuning could also be interesting to explore. Since our model provides over 30 features for users to tune, it would be better for the model to provide hosts with suggestions on which features should be prioritized to tune so that they can maximize their monthly revenue most easily, such as pricing suggestions, room configuration suggestions based on other listing features and other competing listings.

Lastly, real time data updates from Airbnb servers. Our model is trained based on a fixed dataset collected on Oct. 16, 2018. Therefore, the prediction accuracy is severely diminished as it only based on the situation at certain timing. If we could train our model through latest data updated in real time, we could substantially improve our prediction performance.

ACKNOWLEDGMENT

This research was supported by Machine Learning course during Fall 2018 semester at New York University Shanghai. We thank our professor Dr. Enric Junqu de Fortuny and teaching assistants Dr. Xianbin Gu, Mr. Ruowen Tan who provided insight and expertise that greatly assisted the research.

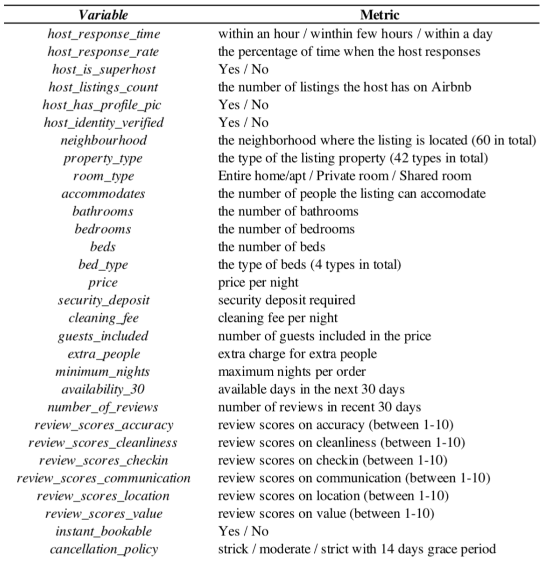
REFERENCES

[1] P. Choudhary, A. Jain, and R. Baijal, Unravelling Airbnb Predicting Price for New Listing, arXiv:1805.12101 [q-fin], May 2018. [Online serial]. Available: http://arxiv.org/abs/1805.12101. [Accessed Dec. 5, 2018].

[2] K. Calixte, J. Curran, L. Y. S. Lai, and E. Ram, Airbnb: Predicting Loyalty, Semantic Scholar, 2016. [Online], Available: https://bit.ly/2UjJsEB. [Accessed Dec. 5, 2018].

[3] E. Tang and K. Sangani, Neighborhood and Price Prediction for San Francisco Airbnb Listings, Stanford School of Engineering, Stanford University, Stanford, CA, 2015. [Online serial]. Available: http://cs229.stanford.edu/proj2015/236_report.pdf. [Accessed Dec. 5, 2018].

APPENDIX



Variable	**Metric**
host_response_time	within an hour / winthin few hours / within a day
host_response_rate	the percentage of time when the host responses
host_is_superhost	Yes / No
host_listings_count	the number of listings the host has on Airbnb
host_has_profile_pic	Yes / No
host_identity_verified	Yes / No
neighbourhood	the neighborhood where the listing is located (60 in total)
property_type	the type of the listing property (42 types in total)
room_type	Entire home/apt / Private room / Shared room
accommodates	the number of people the listing can accomodate
bathrooms	the number of bathrooms
bedrooms	the number of bedrooms
beds	the number of beds
bed_type	the type of beds (4 types in total)
price	price per night
security_deposit	security deposit required
cleaning_fee	cleaning fee per night
guests_included	number of guests included in the price
extra_people	extra charge for extra people
minimum_nights	maximum nights per order
availability_30	available days in the next 30 days
number_of_reviews	number of reviews in recent 30 days
review_scores_accuracy	review scores on accuracy (between 1-10)
review_scores_cleanliness	review scores on cleanliness (between 1-10)
review_scores_checkin	review scores on checkin (between 1-10)
review_scores_communication	review scores on communication (between 1-10)
review_scores_location	review scores on location (between 1-10)
review_scores_value	review scores on value (between 1-10)
instant_bookable	Yes / No
cancellation_policy	strick / moderate / strict with 14 days grace period

Fig. 9. Dataset variable description