Story of Airbnb: Price and Superhost Predicting

Han Zhang, Mengqi Xu, Xiao Chen, Yuqiang Zhang

2021.12.3

1. Introduction and Data inspection

Airbnb, Inc. is an American company that operates an online marketplace for lodging, primarily homestays for vacation rentals, and tourism activities. Price is one of the main factors guests consider when choosing where to stay, which even decides a guest's first and last impression. Also, from the perspective of landlords, no matter how beautiful your space, if it's priced higher than comparable listings in the area, there's a good chance you'll miss out on bookings. Additionally, unfair pricing is able to drive up the rental prices in the local areas, disturb economic order, and harm the local make sure that the price defines the maximum value of the product and at the same time it must appeal to the customers.

We get data from Inside Airbnb**http://insideairbnb.com/get-the-data.html**. We choose New The NYC dataset contains 37274 observations with 74 variables. And this **listing.csv** is linked to **reviews.csv** which we would also make use of later.

id	host_response_rate	longitude	maximum_maximum_nights	review_scores_rating
listing_url	host_acceptance_rate	property_type	minimum_nights_avg_ntm	review_scores_accuracy
scrape_id	host_is_superhost	room_type	maximum_nights_avg_ntm	review_scores_cleanliness
last_scraped	host_thumbnail_url	accommodates	calendar_updated	review_scores_checkin
name	host_picture_url	bathrooms	has_availability	review_scores_communication
description	host_neighbourhood	bathrooms_text	availability_30	review_scores_location
neighborhood_overview	host_listings_count	bedrooms	availability_60	review_scores_value
picture_url	host_total_listings_coun	beds	availability_90	license
host_id	host_verifications	amenities	availability_365	instant_bookable
host_url	host_has_profile_pic	price	calendar_last_scraped	calculated_host_listings_cou nt
host_name	host_identity_verified	minimum_nights	number_of_reviews	calculated_host_listings_cou nt_entire_homes
host_since	neighbourhood	maximum_nights	number_of_reviews_ltm	calculated_host_listings_cou nt_private_rooms
host_location	neighbourhood_cleansed	minimum_minimum_nights	number_of_reviews_l30d	calculated_host_listings_cou nt_shared_rooms
host_about	neighbourhood_group_clea nsed	maximum_minimum_nights	first_review	reviews_per_month
host_response_time	latitude	minimum_maximum_nights	last_review	

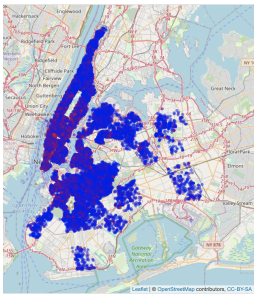
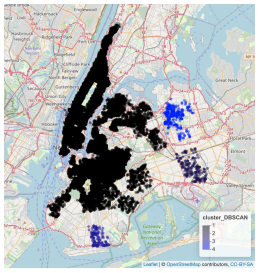
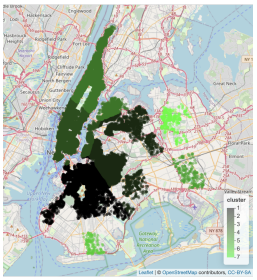
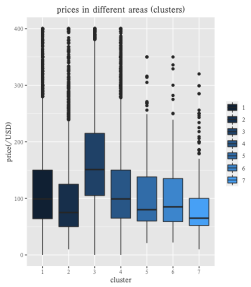
Figure 1: part overview of listings

This is a complex dataset including text(space, neighbourhood,transit,etc.), time(host_since,etc.), numeric(latitude, longitude, price, etc.) and logical(host_is_super_host, host_identity_verified) variables. Ignoring some useless variables such as url and zip code, we could conclude that all variables could be divided into two scopes: information about the house and information about the host. And we will use this criteria to do feature selection and build our price and superhost models. And we would also make use of the reviews text data to extract features to enhance our models. We hope our models could provide valuable insights for Airbnb, hosts and tenants.

2. Clustering and visualization

First, we plot a map (Plot 1) regarding the relationship between the price and the location of the house to have an intuitive view of the whole, where the red point represents a higher price . We can see that red points mainly gather at south of the central park.

Then we conduct clustering analysis based on the coordinates (i.e. longitude and latitude) using DBSCAN (Plot 2). By DBSCAN we divide the houses into four clusters initially, and we can see that cluster1 is pretty large and there are over 30,000 houses in it. Therefore, we continue to use k-means clustering to further cluster points in initial cluster1. Eventually we get 7 clusters (including 4 k-means clusters in initial cluster1 and remaining 3 clusters) as Plot 3 shows. We find that there are obvious differences between the prices of houses of different clusters as the box-plot (Plot 4), so we confirm the location of the house effects its price in some way.

plot 1	plot 2 DBSCAN cluster map	plot 3	plot 4 Boxplot
			

3. Feature Selection and Data Exploration

a) Convert text features to numeric/categorical and Drop useless features

We first use regular expression to extract price value from **price** and other features like **bathroom_text** would also be converted to numeric values. Meanwhile, there are some useless features considering our purposes in the dataset. For instance, we do not want to rely on reviews information because for a new host or department to predict prices because there could be no reviews. And we do not need to consider **listing_url** and other variables similar. So we would first drop some features seperately for prices models and superhost models.

b) Check missing data

We should remove those features with a large number of missing values. Here are the plots for missing data percentage.

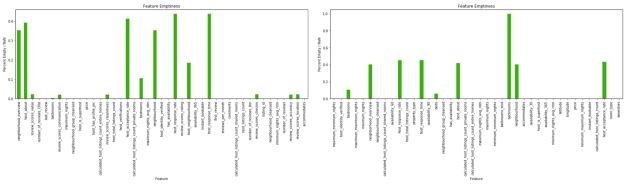


Figure 2: Missing data percentage in different features

We would drop those features with more than 20% missing values.

c) Check density of categorical features

We make sure our categorical features perform appropratly by checking their density and uniqueness. We notice that **amenities** has almost every unique value and it's hard for us to deal so we would also drop this important features. Meanwhile, we find **neighbourhood_cleansed** is pretty long tailed with some labels having only one sample. So we would just cut the dataset with samples of **neighbourhood_cleansed** more than 50.

d) Check features correlation

We would draw the correlation matrix plot with remain features and filter out those features having severe multicollinearity.

The whole process above is done similar for preparing dataset for superhost models and prices models with only a litlle slient different details considering our purposes. Moreover, for **price** values, it would be better for our models to replace **price** with log(**price**). This would help us transform a long tailed distribution to a almost normal distribtuion showed in our code. And another thing important is that as for $price >= 1000$, we could just regard them as outliers and delete these data in prices dataset. Meanwhile, there is text

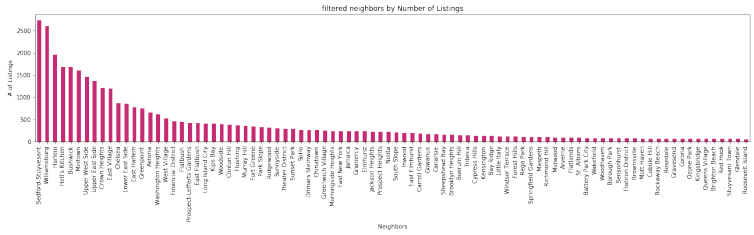


Figure 3: feature counting for neighbourhood cleansed

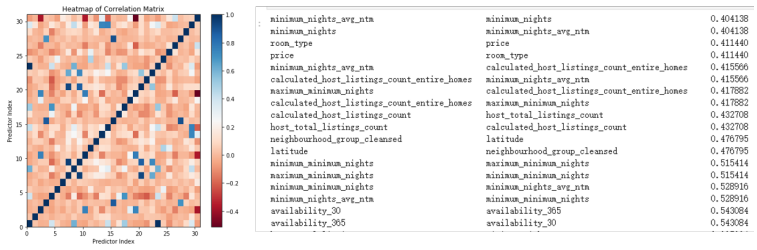


Figure 4: feature correlations and filtering parts

information about reviews comments in **reviews.csv**. We would also use **Bag-of-words model** to deal with this.

4. Model and results

First we would introduce some classical models we use for classification and regression. We would use **linear regression** as baseline for price predicting models and **logistic regression** as baseline for superhost models which is a binary classification problem.

a) Random forest: Classifier/Regressor Random forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random forest is built using a method called **bagging** in which each decision tree is used as a parallel estimator.

b) Gradient boosting decision tree(GBDT): Classifier/Regressor GBDT is a generalization of boosting to arbitrary differentiable loss functions. GBDT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems in a variety of areas. It uses **boosting** technique to create an ensemble learner.

c) XGBoost: Classifier/Regressor XGBoost is a specific implementation of the Gradient Boosting method which delivers more accurate approximations by using the strengths of second order derivative of the loss function, L1 and L2 regularization and parallel computing. XGBoost uses advanced regularization ($l1$ & $l2$), which improves model generalization capabilities.It also delivers high performance as compared to GBDT.

d) Multilayer Perceptron: Classifier/Regressor A multilayer perceptron (MLP) is a class of feedforward artificial neural network. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

e) Naive Bayes method: Classifier Naive Bayes is a simple, yet effective and commonly-used, machine learning classifier. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting. It can also be represented using a very simple Bayesian network.

4.1 Price models: Regressor

We would use the dataset prepared for prices with shape of $(31339, 18)$. After creating dummy variables for categorical features, shape of data should be $(31339, 114)$. Then we split the dataset into train and test datasets. And stored them with **pickle** to make sure our method is solid and repeatable. The train dataset has shape of $(23504, 113)$ and test dataset has shape of $(7835, 113)$. As for regression problems, we would use **MAE** and **RMSE** to evaluate the results. Moreover, we would draw residual plot and predictions v.s. lables plot for the best result(limited to the pages). We would use **linear regression** as baseline. Moreover, we use **randomized-search** and **grid-search** with cross validations to conduct model tuning as showed in the table.

Table 2: Prices models evaluation result with MAE and RMSE.

Esti\Models	Linear Reg	Random Forest	RF Tuned	GBDT	GBDT Tuned	XGBoost	XGBoost Tuned	MLP	MLP Tuned
MAE	0.2926	0.1972	0.1979	0.2365	0.2028	0.2058	0.2043	0.2562	0.2476
RMSE	0.4705	0.3915	0.381	0.4115	0.3783	0.3806	0.3776	0.4485	0.4312

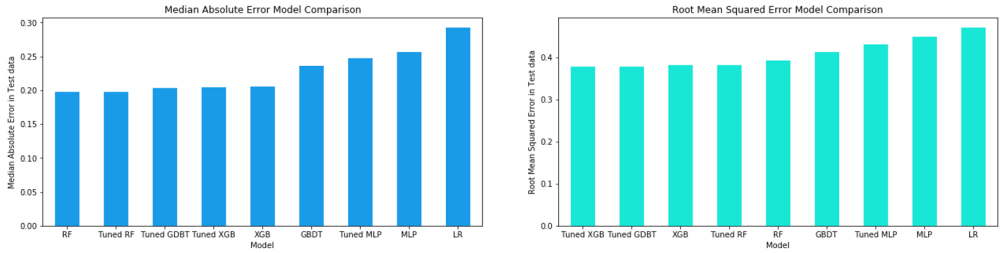


Figure 5: Price Models Comparison

Considering both **MAE** and **RMSE**, the tuned Random forest and tuned XGBoost models perform best on test dataset. But it's still possible that we would get better results with more tuning on important hyperparameters. Limited to pages, we would just show residual plot and predictions v.s. lables plot for tuned XGBoot here.

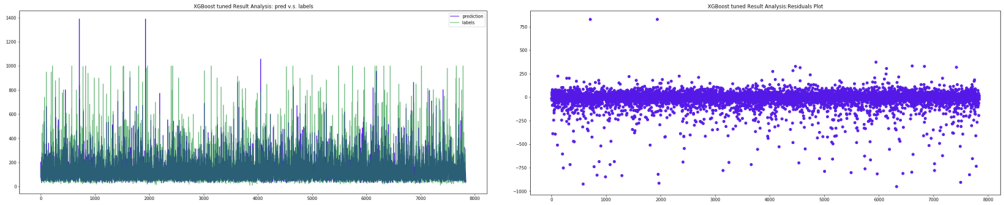


Figure 6: Tuned XGBoost prediction illustration

4.2 Superhost models: Classifier

We use **TfidfVectorizer** (abbr. **Tfidf**) and **CountVectorizer** (abbr. **CV**) as bag-of-words model for text data. CountVectorizer converts a collection of text documents to a matrix of token counts: the occurrences of tokens in each document. This implementation produces a sparse representation of the counts. TfidfVectorizer weights the word counts by a measure of how often they appear in the documents. Besides, we have feature selected data from **listings.csv**. So we would fit differnt classifier models to check their performence on predicting superhost. We would build models with feature selected data (abbr. **FSD**) from previous sections or bag-of-words alone. We tried to combine them together as multi-modal model but this encountered

computing ability issue with extreme large memory usage. And this could be a future work direction. Here is the results of our superhost models. Notice that due to the imbalance, after spliting train and test data, we do upsampling for labels **superhost** in train data. Number of superhost and Non-superhost is 1581 : 4647 in test data.

super(Non-super)	Log Reg			Naive Bayes		
	CV	Tfidf	FSD	CV	Tfidf	FSD
precision	0.56(0.84)	0.49(0.89)	0.46(0.88)	0.37(0.87)	0.28(0.98)	0.28(0.88)
recall	0.51(0.86)	0.73(0.75)	0.71(0.72)	0.77(0.55)	0.99(0.12)	0.93(0.18)
F1	0.53(0.85)	0.59(0.81)	0.56(0.79)	0.50(0.67)	0.43(0.21)	0.43(0.30)
test accuracy	0.77	0.74	0.71	0.60	0.34	0.37
super(Non-super)	Random Forest			XGBoost		
	CV	Tfidf	FSD	CV	Tfidf	FSD
precision	0.53(0.82)	0.59(0.81)	0.73(0.88)	0.62(0.84)	0.63(0.84)	0.72(0.89)
recall	0.45(0.86)	0.38(0.91)	0.65(0.92)	0.48(0.90)	0.47(0.91)	0.67(0.91)
F1	0.49(0.84)	0.46(0.86)	0.69(0.90)	0.54(0.87)	0.54(0.87)	0.69(0.90)
test accuracy	0.76	0.78	0.85	0.79	0.80	0.86

Figure 7: Host models results

From the results we could see that using **feature selected data** with **XGBoost** would get better models. But due to the imbalance, our model could not predict superhost labels as well as non-superhost. Here is ROC curves for XGBoost method.

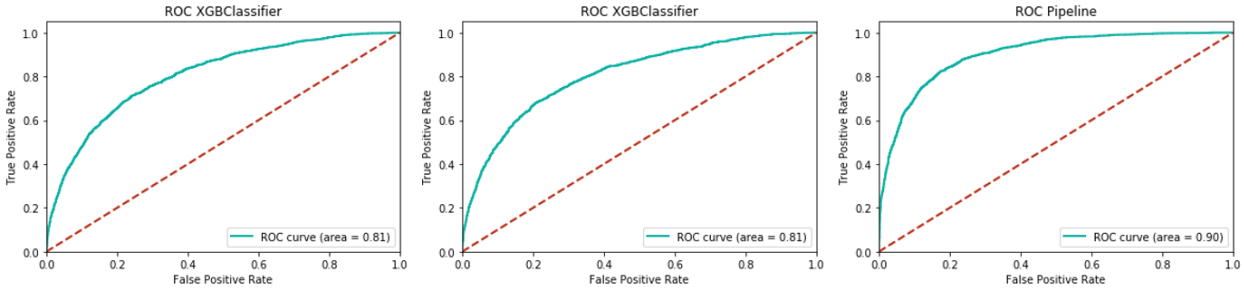


Figure 8: ROC Curves: CV/Tfidf/FSD

5. Conlusions and Discussions

In this project, we explore the Airbnb datasets with several visualization method. We build our train and test dataset with feature selection and deal with text data using bag-of-words model. After this we train classifier models to predict labels of superhost and regressor models to predict houses' prices. And we found using tuned XGBoost with grid-search cross validations help us got not bad results on both host and price dataset. And our models could help host to determine their prices at New York and how they could improve their host services to be named as "Superhost" and acquire priority. However, due to imbalance dataset, our superhost model perform worse predicting "superhost" label than "non-superhost" label. As future work, we could combine feature selection data with bag-of-words model for reviews text data to conduct a multi-modal learning for better results. Features in this two modal data could possibly enhance final models' performance as we see in their single results.

Project Github: https://github.com/sharechanxd/bios625_fp