Airbnb price prediction using Gradient boosting

Liyan Chen
Electrical and Computer Engineering Department
University of California, San Diego
lic128@eng.ucsd.edu

Wen Liang
Electrical and Computer Engineering Department
University of California, San Diego
wel245@ucsd.edu

ABSTRACT

Using data from Inside Airbnb project, first clean the data, then extract useful features and information from the data set to train regression model, use the model and gradient boosting technique to predict the Airbnb offering price. In the mean time, build a price distribution map to have a intuitive impression on the Airbnb's price distribution. Different methods of regression are used for comparison, and the results shows that gradient boosting is a great technique for this regression model and the Airbnb listings price prediction task.

KEYWORDS

Linear model, regression, prediction, Airbnb price, Gradient boosting.

1 INTRODUCTION

With the popularization of Airbnb all around the world, Airbnb has changed the way people travel, more and more people choose Airbnb as their accommodation. In the mean time , the huge amount of information provided by the hosting list give us a window into how participants in the new sharing economy market their offerings. In the study, we aim at using the information provided by Inside Airbnb project, using gradient boosting to predict the price of a given offering.

New York City is well known for its diversity and population, and the Airbnb offering is also huge: about 40000.Thus we select New York City as our target city and our prediction task is to predict the listing price in New York City. Using several techniques, we visualize and analyze the relation between Airbnb listings price and other information provided to do the feature selection, during which we get a map about the distribution of price, from which we can get a intuitive sense on the distributions and correaltion. This process also included baseline description , feature selection and so on. Then we build and train a model to predict a listing's price, in the mean time observing the position of the listing. During the model building process, we first apply basic linear regressor and other basic regressor to predict the result, and render them as the baseline, then we apply gradient boosting to predict the price, which turn out to be better than other method.

2 RELATED WORK

After literature search, we find that there is not much published studies that apply machine learning techniques to data from the Inside Airbnb project. One of the most interesting paper we find is to predict the price using multiscale clustering skills in [1], since here we only care about one-room type offerings, there is not much clustering skills to apply. And we also notice that in [2] , the author find that 'social factors' like number of references, host response rate, and number of reviews are even more important determinants of room booking than conventional factors like price and amenities offered.

There are also many other topics involved algorithmic pricing. Hill [3] explains Airbnb's AI-based dynamic pricing tools, discussing how the original regression-based tool released in 2013, which used amenities of a listing and information about neighboring properties to predict the appropriate pricing, was refashioned into the company's most recent, reinforcement-learning based tool. And in [4], the author illustrates the relationship between the Neighborhood and the price prediction for San Francisco, which uses SVM as their main tool, and the output is the price interval.

3 DATASET STATISTICS

We use data from Airbnb detailed listings data for New York City which includes the price which is what we want to predict, location information, environment, accommodation attributes and rating of reviews. We also use data 2010 TIGER/Line Shapefiles from US Census Bureau which provides the map, zip code, shape and location of neighborhood blocks.

First, we did some data cleaning. The data is coming in a little raw, the 'number_reviews' and 'reviews_per_month' fields looks like they need some special processing to remove a large number of NAN values. Rather than discarding these outright, we set the value of 'reviews_per_month' to 0 where there is currently a NAN, because some quick analysis shows that this field is NAN only wherever 'number_of_reviews' is 0.There still exist some strange data,ex, the bedrooms, beds, price is nan or 0, we just drop them.
After the cleaning, we get a dataset about 27735 size.We divide the first dataset into training set and test set.

Table 1: Frequency of Special Characters

Datasets of Listings	Training Set	Test Set	Total
Size	18188	9547	27735

3.1 Properties

Before we focus on data analysis, we first analyze and select some informative properties from the raw datasets for our prediction.

Listings.csv: For each listings, we are provided with following features:

- 'price': (price per night),
- 'accommodates': (number of people can house accommodates),

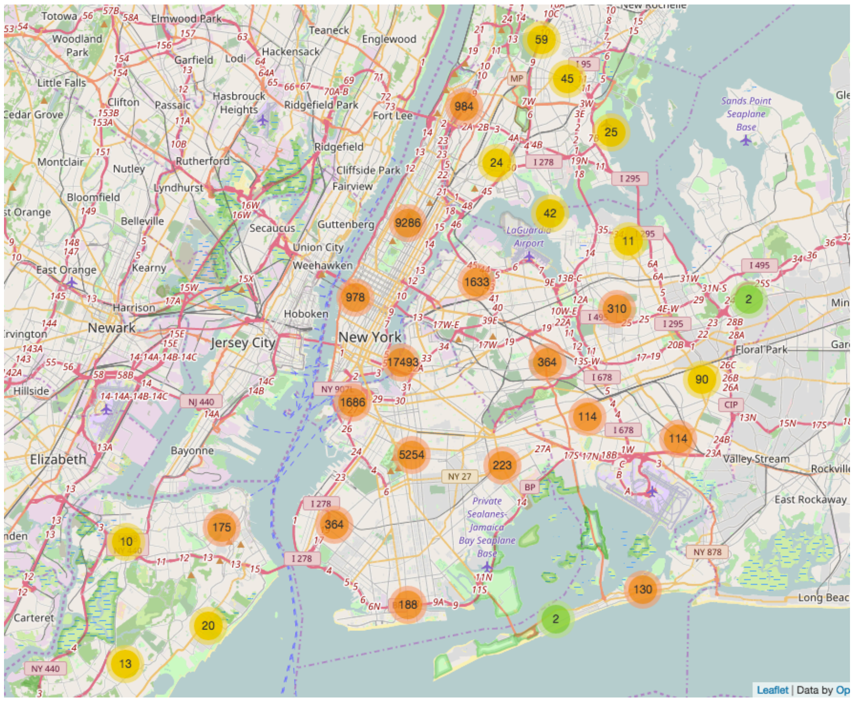

Figure 1: *Clustered Number of Airbnb Listings in New York City*

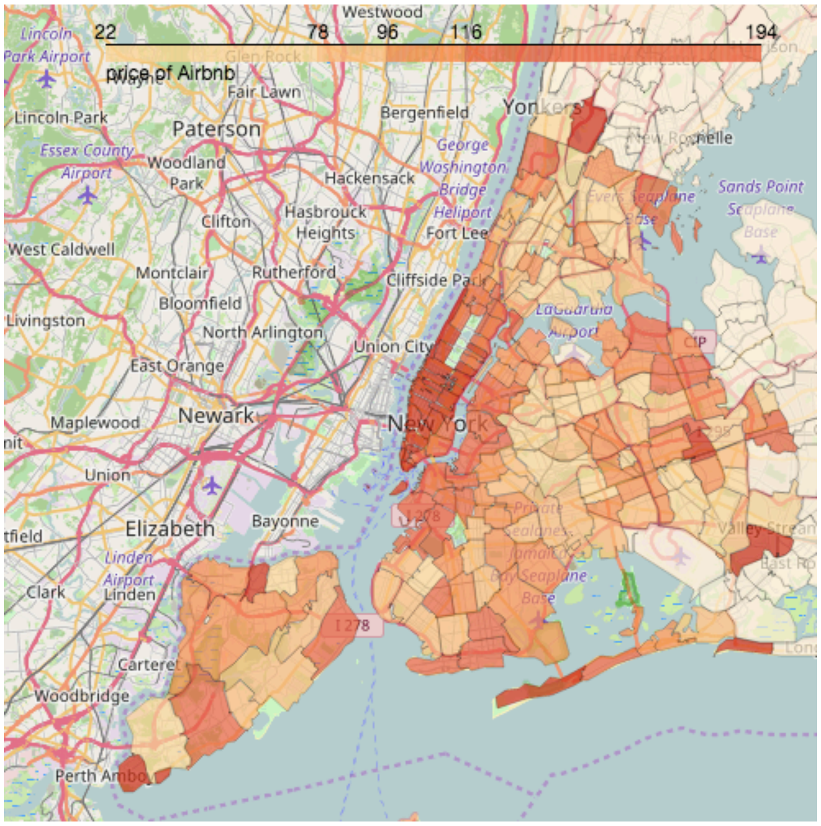

Figure 2: *Average Price of Airbnb Listings in blocks of New York City*

- 'bedrooms' :(number of bedrooms),
- 'beds'(number of beds),
- 'room_type',
- 'bed_type',
- 'property_type' (Apartment, House, Villa and some other types),
- 'neighbourhood_group_cleansed' (formatted name of neighbourhood),
- 'zipcode' (the zip code of the location),
- 'reviews_per_month' (number of reviews get per month),
- 'review_scores_rating' (average rating of reviews),
- … other scores of reviews (service, clean, accuracy),
- 'availability_30' (number of days available in 30 days),
- 'instant_bookable' (able to book instantly or not)

NYC.geojson: For each units of block, we are provided with following information:

- 'zip code', (zip code of the block)
- 'neighbourhood', (formatted names of neighbourhoods)
- 'geometry'(the block shapes described by longitude and latitude).

3.2 Properties Statistics

We visualize the whole dataset in Figure 1. We also want a map to visualize the listings price differences in the city so we use zip codes and corresponding geojson file to plot out this information in Figure 2.

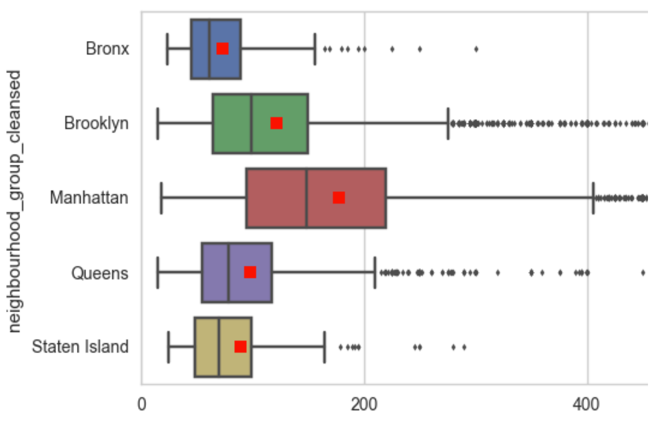

Figure 3: *Box Plot of neighborhood categories and price*

As expected, 5 different regions have different distributions, the average prices (colors in Figure 2 and the red points in Figure 3) have great difference with each other and the relation between price and neighborhoods is shown in the box plot. The listings in Manhattan are more likely have higher prices and other 4 regions have similar distribution. For other categorical features, we can also visualize them by using box plot in Figure 4.

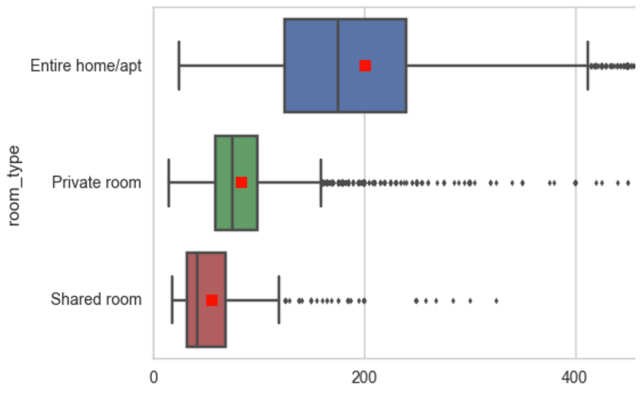

Figure 4: *Box Plot of room type categories and price*

We can see that there are great differences in position and shape of the boxes for different type of rooms which means that prices from three categories have different distributions and the whole apartment or entire home has high probability that its price is greater than two other type of room. For more accurate test of dependency, we use ANOVA (Analysis of variance) test to test the null hypothesis that $\mu_{class1} = \mu_{class2} = \mu_{class3}$ (the population means of three groups are equal). Because our features distributions are similar to normal distribution so the assumption of ANOVA is satisfied in this dataset. The result for this 3 classes is p-value = 1.0369e-12 and the variance explained by this feature is $\omega^2 = \frac{SST}{totalSS} = 0.2038$. The extremely small p-value means the we have great confidence to reject this null hypothesis and believe that the price has relation with this categorical feature and the ω^2means that this feature can explain 20.38% of the price data variance. Therefore, this feature contains very useful information for our price prediction tasks. Similarly, the box plots and ANOVA test help us to select informative features such as cancellation policy type, room type, bed type and cancellation policy. Those features can indicate the size, environment of the accommodation or the service quality.

Table 2: ANOVA test p-values

Categorical features	p-value
Neighborhood	0
Bed Type	2.0617e-10
Instant Bookable	0.0149
Room Type	1.0369e-12

For continuous features, we use scatter plots and pearson correlation coefficients to find association between features and price.

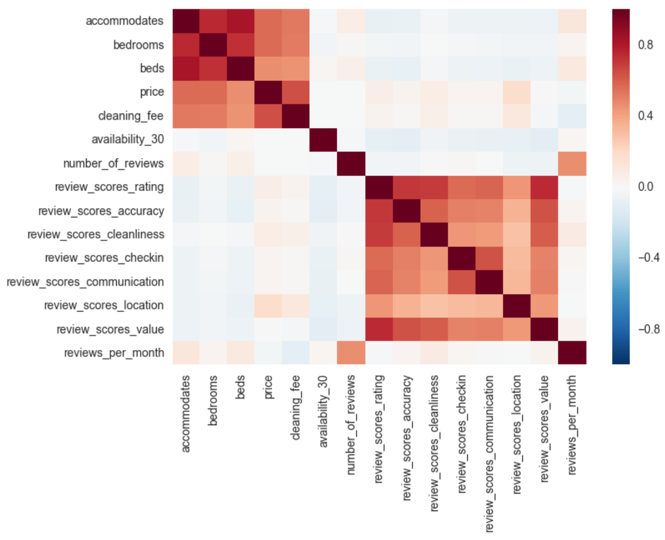

Figure 5: *Box Plot of room type categories*

From Figure 5, we can see that there are several features that are highly correlated with price such as accommodates, number of bedrooms, number of bed.

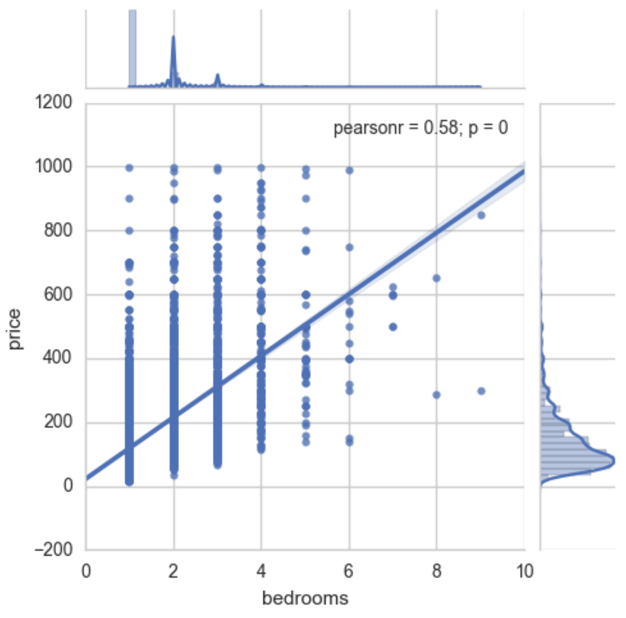

Figure 6: *The scatter plot and correlation between number of bedrooms and price*

Using number of bedrooms in Figure 6 as an example, there is a clear linear relation between price and number of bedrooms. This is reason why we do not define the number of bedrooms as a categorical feature although the value is discrete from 1 to 9. The pearson correlation coefficient r = 0.58, suggesting that the number of bedrooms is a good feature for our prediction task. The reason is the fact that more bedrooms indicate more space and higher prices. Also, since we do not have accurate data of area of the room or house, the number of bedrooms is the feature that reveals the area. Moreover, we need chi-square test to get more solid mathematical results for feature selection. For categorical features, we use ANOVA test. For continuous data, we should use chi-square test to test the ability of continuous features to explain the target data. The p-value for number of bedrooms is 2.80819856e-280, which means there is association between the feature and price. We do similar process for each of continuous features and select some features such as bedrooms, beds, accommodates, cleaning fee, location review score, number of reviews and cleanliness review score.

Table 3: chi-square test p-values for selected features

features	p-value
Bedrooms	2.80819856e-280
Beds	0
Accommodates	0
Cleaning Fee	0
Location Review Score	2.81648877e-060
Number of Reviews	0
Cleanliness Review Score	2.04605177e-43

4 MODEL

We use different model to predict the price of a given Airbnb hosting, but the idea is the same: using features: ‘price’, ‘accomodates’, ‘bedrooms’, ‘beds’, ‘neighbourhood_cleansed’, ‘room_type’, ‘cancellation_policy’, ‘instant_bookable’, ‘reviews_per_month’, ‘number_of_reviews’, ‘availability_30’, ‘review_scores_rating’ to train a linear model to predict the price. Ex.

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_{12} x_{12}$$

But here is still some problems, there are a few columns that contain categorical variables, which is represented by a string , but when we do regression, the input must be numeric. Thus we must do preprocessing to convert these to numeric variables. We use get_dummies function in Pandas to achieve this. This process is also known as “one hot” encoding, which turns the categorical variables into a unit binary vector at which the position of its value is 1 otherwise 0. After preprocessing, we get a 213 length feature representation of the dataset, using which we can apply a bunch of different linear regression models on.
The model we’d like to use here is Gradient boosting, which is a machine learning ensemble meta-algorithm for primarily reducing bias, and also variance in supervised learning. The basic idea of boosting is to convert weak learners to strong ones. Starting from a weak learner, we do iterations, at each iteration, we change the weight distribution of the data, then apply new weak learner on the changed data, thus getting a bunch of different weak learners, combine those weak learners, we get the final strong learners.
Here remains two questions:

- How to change the weight distribution of the dataset in each iteration?
- How to combine different weak learners into a strong learner?

The answer follows:

- If the cost of a data point is great, we assign a greater weight on the data point in next iteration and assign less weight on the points which the cost is less.

- If the cost of a weak learner is great, we assign less weight on it when combining the learners and assign greater weight on the learners which the cost is less.

Below is a description of the Gradient boosting algorithm:

```
{
Input: Training set:
D = $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \dots (x^{(M)}, y^{(M)})\}$,
$x^i \in \chi \subseteq R^n$,     $Cost\ Function: L(y, f(x))$;
Output: Strong learner $f_s(x)$
    1. Initialize the model:
        $f_0(x) = \arg min_c \sum_{i=1}^{M} L(y^{(i)}, c)$;
    2. Train K models iteratively k = 1,2, … , K
        • Compute the residual: for $i = 1,2, \dots, M$:
            $r_{ki} = -[\frac{\partial L(y^i, f(x^i))}{\partial f(x^i)}]_{f(x)=f_{k-1}(x)}$
        • Train a regressor for the residual, get the leafnode $R_{kj}, j = 1,2,3 \dots, J$
        • For $j = 1,2,3 \dots, J$, compute:
            $c_{kj} = \arg min_c \sum_{x_i \in R_{kj}} L(y^i, f_{k-1}(x_i) + c)$
        • Update the model:
            $f_k(x) = f_{k-1}(x) + \sum_{j=1}^{J} c_{kj} I(x \in R_{kj})$
    3. Get the final Strong learner:
        $f_s(x) = f_K(x) = \sum_{k=1}^{K} \sum_{j=1}^{J} c_{kj} I(x \in R_{kj})$
}
```

Here, we use the basic linear regression, ridge and lasso regression, ElasticNet, Bayesian ridge and OMP regressor to perform different regression. And compare the result of those regressor to illustrate the accuracy of the gradient boosting algorithm.

5 EXPERIMENT AND RESULTS

To evaluate which model are doing better, we will need some way to score the results. We choose median absolute error here as the score, for the reason that the result of median absolute error is more intuitive: it present the difference in dollars between the prediction and the actual price.

Table 4: Results of Different Regressors for This Model

REGRESSOR	ERROR(MEDIAN ABSOLUTE ERROR)
LINEAR	19.56
RIDGE	19.50
LASSO	25.88
ELASTICNET	31.02
BAYESRIDGE	19.40
OMP	21.26

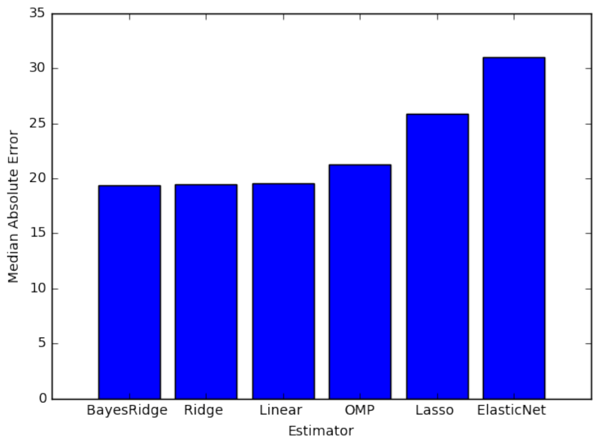

Figure 7: *The median absolute error of different estimators*

Looking at the error from each of these six estimators, BayesRidge , Ridge,OMP and Linear appear to be roughly the same, while Lasso and ElasticNet is obviously larger. However, most of the estimators are able to predict the price with a median error about 20 dollars. Having such close result is due to the fact that we haven't done any tuning.

Next, we will try gradient boosting on the dataset,. First let's try the basic regressor : Gradient BoostingRegressor() without tuning, we get result : 17.34199425157995.
using GridSearchCV to take exhaustive and cross-validation to perform the parameter tuning, after which we get the tuned parameter:

Table 5: Tuned Parameters

Parameter	Value
Max_depth	4
Min_samples_split	2
n_estimators	1500
Loss	"lad"

And the result of the tuned parameter is a median error of $ 16.763842060613154.
After the result, we'd like to analyze the deviation's correlation with boosting iterations, to make sure that our model is not overfitting, as we can see from Figure 8. The deviation decrease as the boosting iteration increase and become flat at the point about 1500, which means that we are not overfitting.
And after getting the final regressor, we are also interested in the most influential features, as we can see from Figure 9, the most influential feature are the reviews_per_month, which is in our expectation, and make perfect sense.

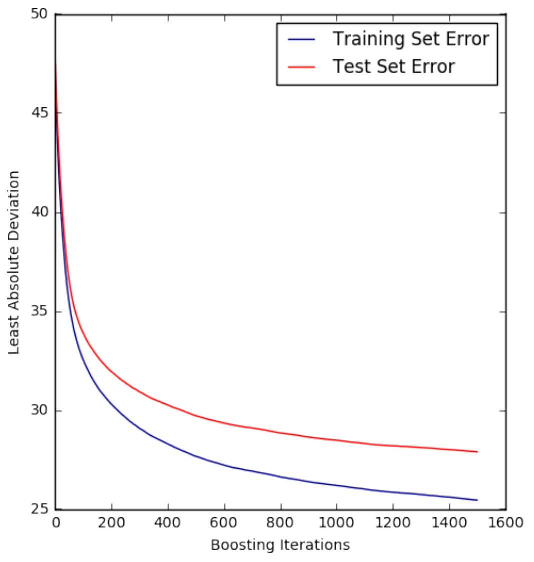

Figure 8: *Least Absolute Deviation vs Boosting iterations*

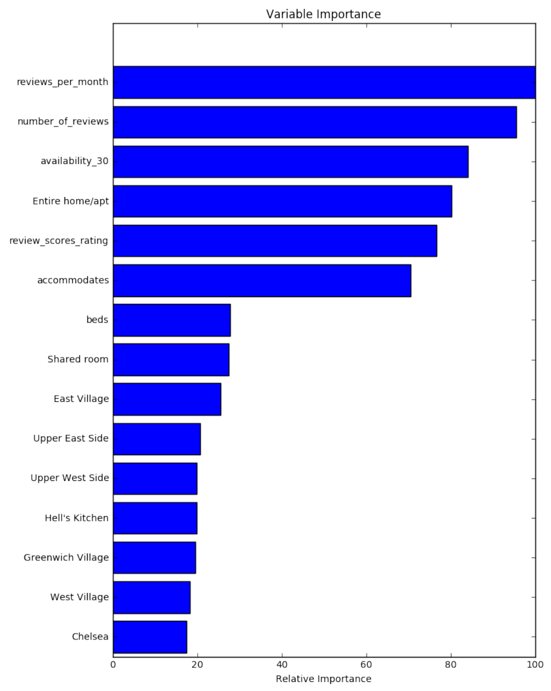

Figure 9: *The most influential features.(East Village, Upper East Sidem Upper West Side, Hell's Kitchen, Greenwich Viallage, West Village and Chelsea are name of neghborhoods)*

6 CONCLUSION AND FUTURE WORK

Our analysis of the Inside Airbnb project's listings for New York shows that we can successfully predict price using an array of features extracted from listings. Using different regressor as comparison, the gradient boosting regressor show great accuracy and ability in avoiding overfitting. All the regressor used in the experiment somehow shows good performance, considering the median absolute error is around 20, which suggests that the feature we use is reasonable.

In the future work, our final goal is to construct a recommendation system for Airbnb users, in which the user input an price, then the system offers a good option. To achieve such goal, latent factor and text-mining can be used, to guarantee a better result.

REFERENCES

[1] Reasonable price recommendation on Airbnb using Multi-Scale clustering. IEEE Control Conference (CCC), 2016 35th Chinese ISBN: 978-9-8815-6391-0
[2] Lee, D., Hyun, W., Ryu, J., Lee, W. J., Rhee, W., & Suh, B. (2015, February). An Analysis of Social Features Associated with Room Sales of Airbnb. In Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work & Social Computing (pp. 219-222). ACM
[3] Hill, D. (2015). How much is your spare room worth?. Spectrum, IEEE, 52(9), 32-58
[4] Neighborhood and Price Prediction for San Francisco Airbnb Listings, Emily Tang, Kunal Sangani.