APANPS5200

Kaggle Competition Report

Predicting Airbnb Rent in New York

QIAN ZHA

Part 1 Introduction

Airbnb pricing is important to get right, particularly in big cities like New York City. In this Kaggle competition, our goal is to construct a model using the dataset supplied and use it to predict the price of a set of Airbnb rentals included in scoringData.csv, and Success will be measured by lowering the root mean square error (RMSE).

This report will detail the findings with the central goal of optimizing the final predictive model. With the help of machine learning model, we can predict more accurately on New York City's rental price, thus give an insights for people who want to rent apartments in New York City.

Part 2 Data Preparation and Data Exploration

Data tidying and wrangling is important before we start to analyze the data because it makes raw data usable. It is considered as a backbone to the entire analysis part.
The analysis data collected by Airbnb contains nearly 50 thousand lines, and over 90 different variables. Data clean-up is necessary to ensure the integrity of the model is upheld. The results of our analysis will be skewed if we don't use good, clean data - sometimes to the point that they're actually more harmful than helpful.

In the Kaggle competition, several "tidying" strategies were implemented, and are critical to helping me to evaluate what variables would be the most impactful in the model analysis.

Data Splitting

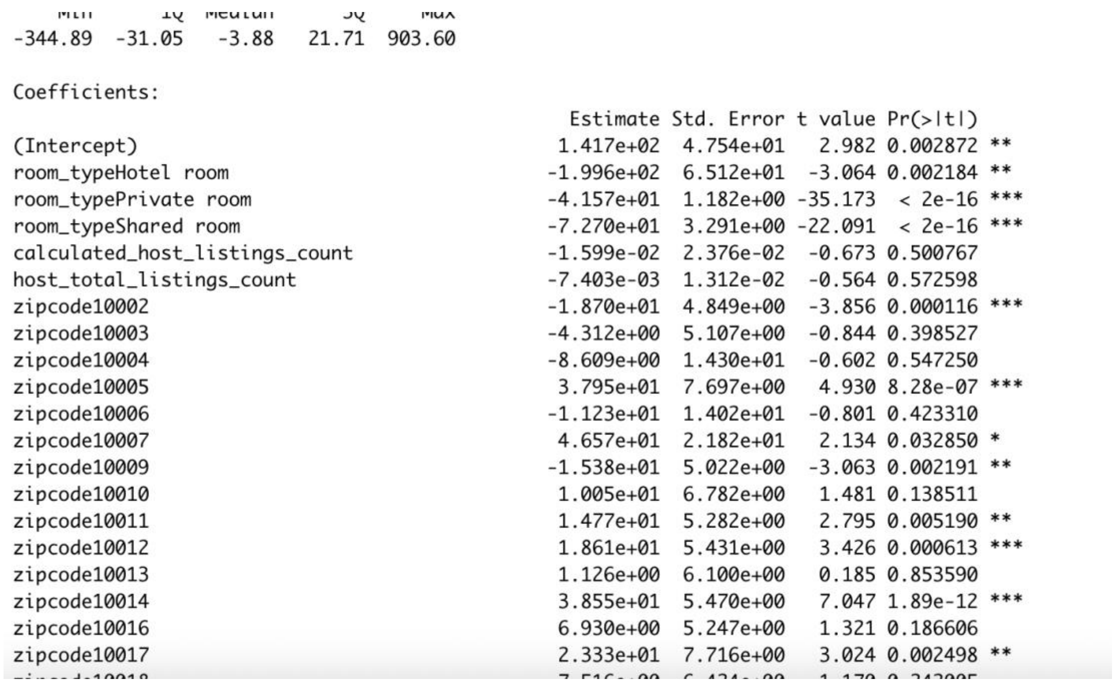
All preparation of this data was performed on the analysis data set, which was created to ensure that improvements to the data impacted both the analysis or train set, as well as the scoring data frame which would later serve as the test set for the rental price predictions. I used two different data splitting ways for different machine learning models—simple random split and stratified random sampling. For stratified random sampling, I use the function createDataPartition( ) from the caret package to proportionately split the data, funneling 70% of the values to be trained during model creation and 30% to be used in the testing sample. For simple random split, I use sample( ) function to randomly split data, but also 70% of the values to be trained during model creation and 30% to be used in the testing sample.

DATA CLEANING

There are several steps in my data cleaning.

Cleaning data is very important before we analyzing the data because it helps to manipulate raw data within certain columns to be more valuable. I use functions like sum(is.na()),[!is.na()] and na_mean() particularly in this Kaggle competition. The function sum(is.na()) is useful to count missing values of a dataframe column in R. First I will use sum(is.na()) to check which columns have missing values, and once a list of variables with "NA" was compounded, I will use function [!is.na()] to remove the missing values. For example, *train[!is.na(train$host_is_superhost),];*
train< train[!is.na(train$calculated_host_listings_count),]
For columns with numeric data type such as *cleaning_fee, security_deposit, beds*, and *square_feet*, I will use na_mean() to impute the mean of the data in place of "NA" within the data set. For example,
analysis$security_deposit[is.na(analysis$security_deposit)]=mean(analysis$security_deposit,na.rm=TRUE)

To define what variables I should use in my final model, I first build a linear regression model to see the correlation with price (exabit 1 show part of the correlation).



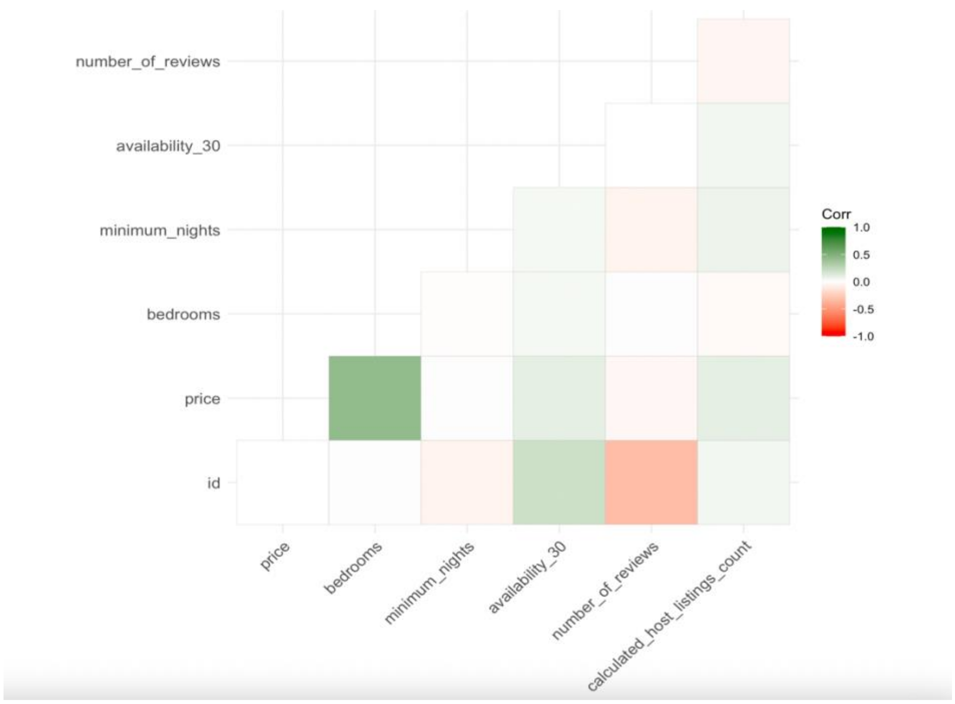
```
    Min      1Q  Median      3Q     Max
-344.89  -31.05   -3.88   21.71  903.60

Coefficients:
                                   Estimate Std. Error t value Pr(>|t|)
(Intercept)                       1.417e+02  4.754e+01   2.982 0.002872 **
room_typeHotel room              -1.996e+02  6.512e+01  -3.064 0.002184 **
room_typePrivate room            -4.157e+01  1.182e+00 -35.173  < 2e-16 ***
room_typeShared room             -7.270e+01  3.291e+00 -22.091  < 2e-16 ***
calculated_host_listings_count   -1.599e-02  2.376e-02  -0.673 0.500767
host_total_listings_count        -7.403e-03  1.312e-02  -0.564 0.572598
zipcode10002                     -1.870e+01  4.849e+00  -3.856 0.000116 ***
zipcode10003                     -4.312e+00  5.107e+00  -0.844 0.398527
zipcode10004                     -8.609e+00  1.430e+01  -0.602 0.547250
zipcode10005                      3.795e+01  7.697e+00   4.930 8.28e-07 ***
zipcode10006                     -1.123e+01  1.402e+01  -0.801 0.423310
zipcode10007                      4.657e+01  2.182e+01   2.134 0.032850 *
zipcode10009                     -1.538e+01  5.022e+00  -3.063 0.002191 **
zipcode10010                      1.005e+01  6.782e+00   1.481 0.138511
zipcode10011                      1.477e+01  5.282e+00   2.795 0.005190 **
zipcode10012                      1.861e+01  5.431e+00   3.426 0.000613 ***
zipcode10013                      1.126e+00  6.100e+00   0.185 0.853590
zipcode10014                      3.855e+01  5.470e+00   7.047 1.89e-12 ***
zipcode10016                      6.930e+00  5.247e+00   1.321 0.186606
zipcode10017                      2.333e+01  7.716e+00   3.024 0.002498 **
```

Exabit 1(part)

From the summary of model 1, I find that variables such as room_type, bathrooms, bedrooms, guests_included, extra_people, zipcode(etc.) have strong relation to Airbnb rental price, while variables such as bed_type, maximum_nights,host_response_time show no relation to Airbnb rental price.

To explore more on data, I library(ggcorrplot) and examine a correlation matrix to look evaluate bivariate correlations with selected variables.

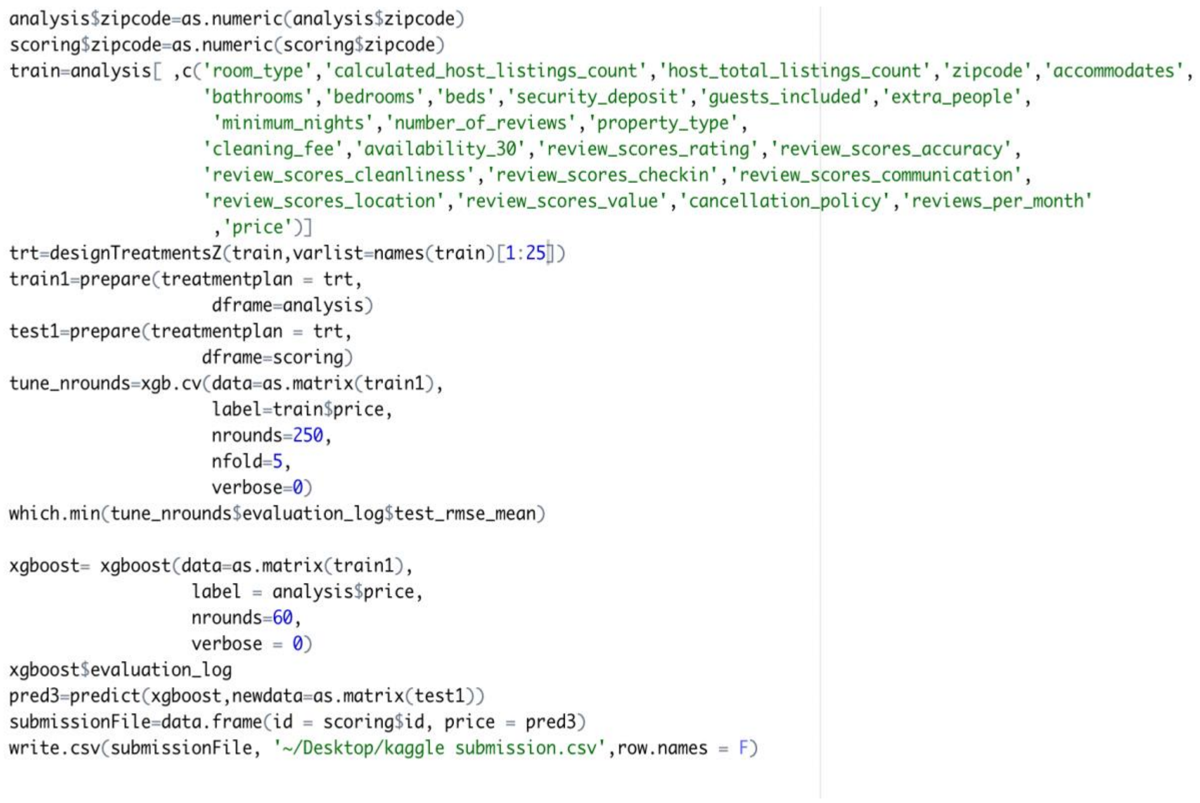
As the correlation matrix figure shows there is a positive link to the minimum number of nights, 30 days of availability, and the graph also indicates the least correlation with the number of reviews.



Finally I exclude variables that shows no correlation with rental price and include nearly all numeric variables in my predicting model. The final variables I choose are:

'room_type',	'calculated_host_listings_count'	'host_total_listings_count'	'zipcode'
'accommodates'	'bathrooms',	'bedrooms','beds',	'reviews_per_month'
'security_deposit'	'guests_included',	'extra_people'	'minimum_nights'
	'number_of_reviews',	'property_type'	'cleaning_fee'
'availability_30'	‘review_scores_rating',	'review_scores_accuracy'	'review_scores_cleanliness'
'review_scores_checkin'	review_scores_communication',	'review_scores_location'	'review_scores_value',
'cancellation_policy'			

Part 3 My Best Entry Model (XGBOOST)--R code



```r
analysis$zipcode=as.numeric(analysis$zipcode)
scoring$zipcode=as.numeric(scoring$zipcode)
train=analysis[ ,c('room_type','calculated_host_listings_count','host_total_listings_count','zipcode','accommodates',
                   'bathrooms','bedrooms','beds','security_deposit','guests_included','extra_people',
                    'minimum_nights','number_of_reviews','property_type',
                   'cleaning_fee','availability_30','review_scores_rating','review_scores_accuracy',
                   'review_scores_cleanliness','review_scores_checkin','review_scores_communication',
                   'review_scores_location','review_scores_value','cancellation_policy','reviews_per_month'
                    ,'price')]
trt=designTreatmentsZ(train,varlist=names(train)[1:25])
train1=prepare(treatmentplan = trt,
                    dframe=analysis)
test1=prepare(treatmentplan = trt,
                   dframe=scoring)
tune_nrounds=xgb.cv(data=as.matrix(train1),
                    label=train$price,
                    nrounds=250,
                    nfold=5,
                    verbose=0)
which.min(tune_nrounds$evaluation_log$test_rmse_mean)

xgboost= xgboost(data=as.matrix(train1),
                  label = analysis$price,
                  nrounds=60,
                  verbose = 0)
xgboost$evaluation_log
pred3=predict(xgboost,newdata=as.matrix(test1))
submissionFile=data.frame(id = scoring$id, price = pred3)
write.csv(submissionFile, '~/Desktop/kaggle submission.csv',row.names = F)
```

Part 4 What I have learnt

In order to gain the most accurate prediction of analysis data, I used several different methodologies, nearly all machine learning models in R that we have learnt in this class, including linear regression ,logistics regression, trees, boosting, bagging, random forest. Finally, the XGBoost prove to be the most powerful one in predicting the Airbnb New York rental in this competition. XGBoost referred to as an "ALL in One" algorithm because it is highly flexible and designed to handle missing data with its in-built features. It is a very powerful tool for both classification and regression.

For other models, I found that random forest with tuned are also powerful in predicting the rental, which can be used for both regression and classification models. However, compared to XGBoost Model, random forest model need a very clean and tidy dataset because it cannot deal with the missing values. At first, I am struggle with dealing with missing value so I just use less than 10 variables to predict the price, and the outcome is not satisfied. After I handle the missing value problem and choose different “mtry” for my random forest model, I continuously improve my performance and optimize the accuracy of random Forest model.

In conclusion, Implementation of Machine Learning will make it easier for people to understand these large amounts of data that are provided and find meaningful solutions in them.