**Pricing analysis and prediction**

**of Airbnb in Seattle**

Group 6

*1-May-2020*

|  |  |
| --- | --- |
| Name | Contribution |
| Jian Jian | Data Cleaning, Feature Selection - Lasso/Elastic Net Regression, |
| Tiehao Chen | Data Cleaning, Feature Selection  Regression Model |
| Chenghao Wu | Data Cleaning, sentiment analysis and recommender system |
| Yueyuan He | Data Cleaning, Recommender System |

# Abstract

**Project Overview:**

This project is to identify the dominant factors of the house price in Airbnb. It analyzed the sentiment of guests and provided market strategies to increase the revenue for the stakeholder of Airbnb. Airbnb has become a worldwide application that people use broadly during personal or business travels for its convenience and cost-effectiveness. For this project, we have explored, made in-depth research on some public databases and decided to focus on the dataset in Seattle eventually. By analyzing the strength of the correlation between the price and various dependent variables, the insights we gained from the dataset would help us to answer the major business question: How can we help Airbnb hosts to adjust their pricing strategy to obtain the optimal gross margin and what is the overall sentiment of the users?

**Dataset Description:**

This dataset consists of three parts - listings, reviews, and calendar. We will use listings and reviews in this project. The subset listing are intensively engaged in the analytic process to answer our major question - to predict price about houses by regression. Since we have a comparatively large number of attributes, first, we performed data cleaning and wrangling on a sample data. We categorized the columns by their different property(eg. Host info, Guest info, time, geography), and selected the columns related to the diverse kind of business questions. So the features can be manipulated in different ways regarding various business questions respectively.

There is a large number of features in the dataset, so the below is only a sample list:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **FeatureType** | **Host Info** | **Reviews** | **Geo** | **Time** | **House** | **Useless** |
| **Feature Name** | Host\_since | reviews\_per\_month | latitude | availability\_30 | bedrooms | host\_acceptance\_rate |
| Host\_verifications | number\_of\_reviews | longitude | availability\_60 | bathrooms | instant\_bookable |

**Analytic Approach:**

Based on business understanding, we selected the features that have a strong correlation to the housing price. Considering Databrick is the platform to run the consolidated codebase, our licence does not have the access of its collaborative functionality. So we used Google Colab as the cloud computing solution for team members collaboration. By streaming the analyzing process, the filtered dataset will be fed into the data pipeline. For a particular model, we will reduce the dimensions of the dataset, impute the missing value and outliers, scale and visualize the distribution of each feature to check the effectiveness of the data cleaning process. Using proper machine learning models, we can predict the housing price on the test dataset. Depending on the cases when analyzing, we will compare the model performance among Classification, Regression, Deep Learning, Clustering, Association Rules along with other models to generate the eventual conclusion.

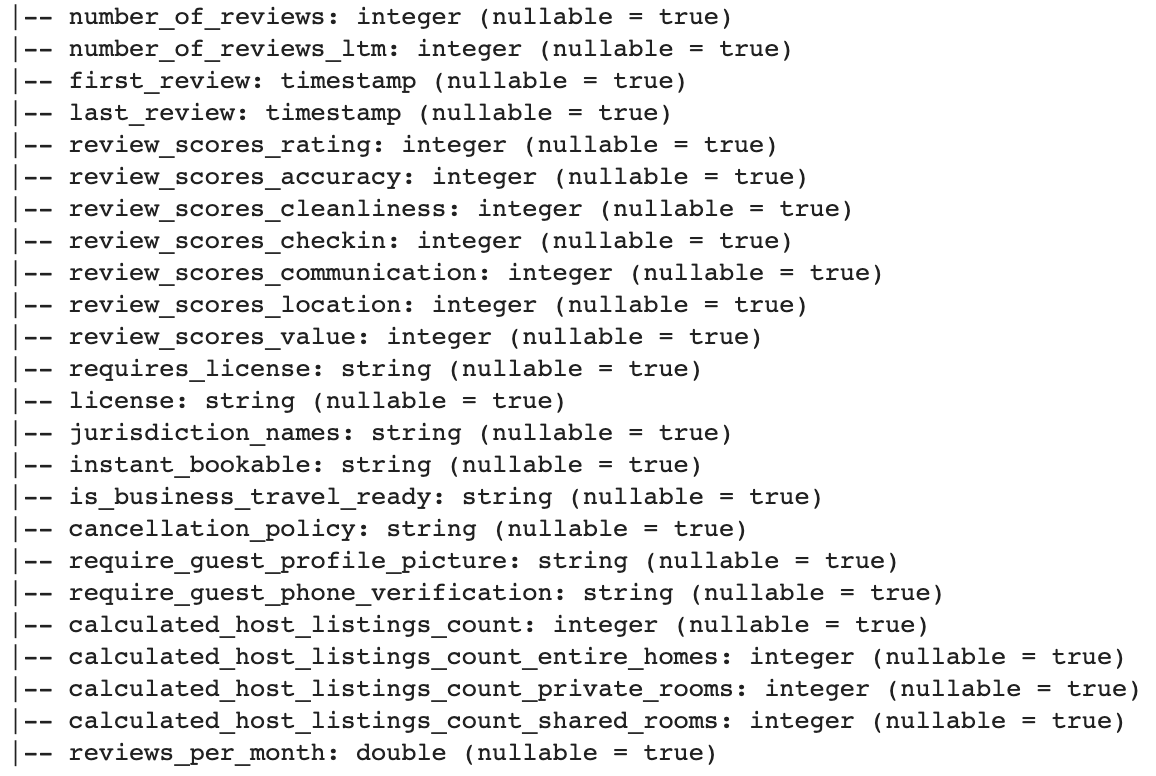
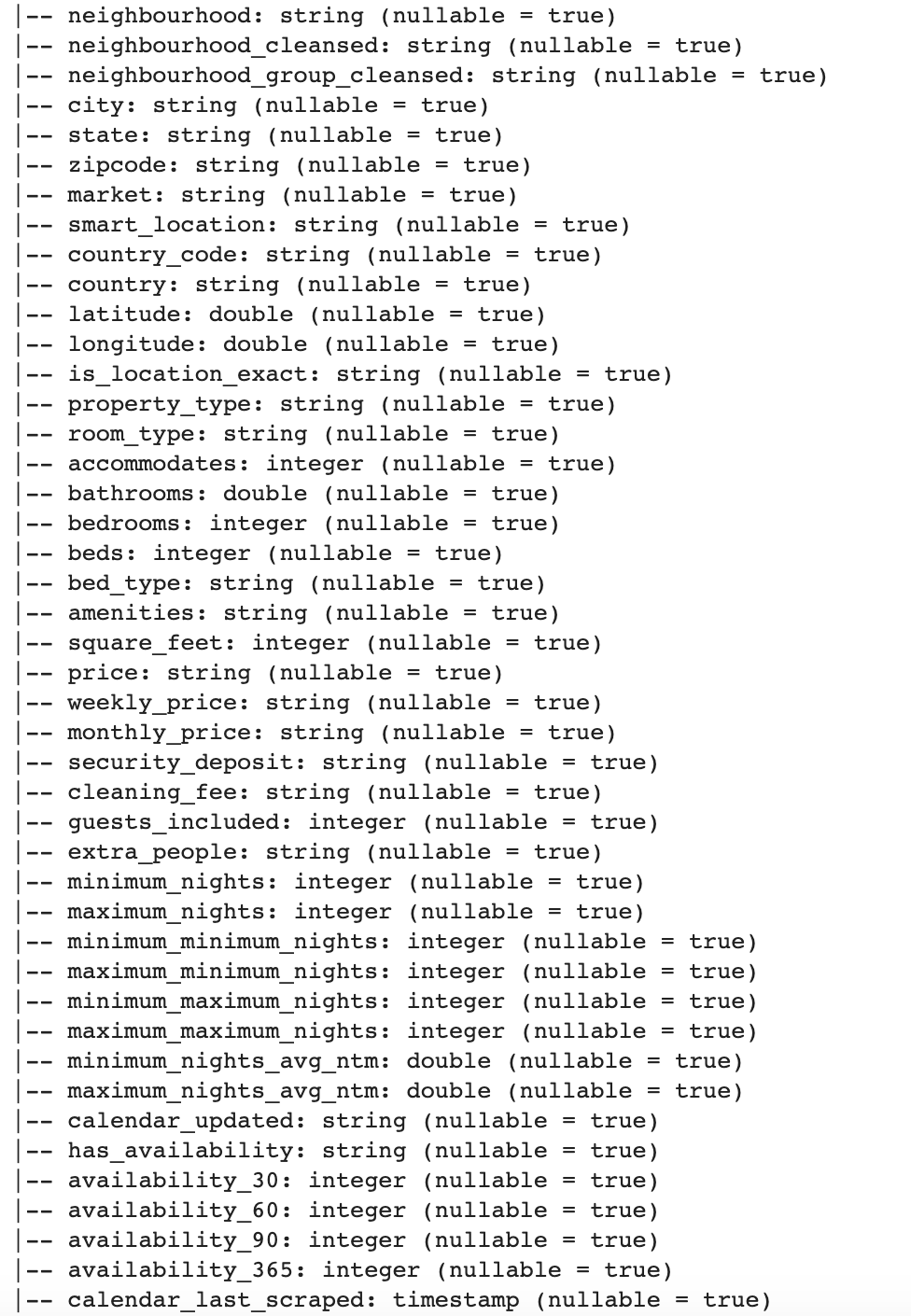
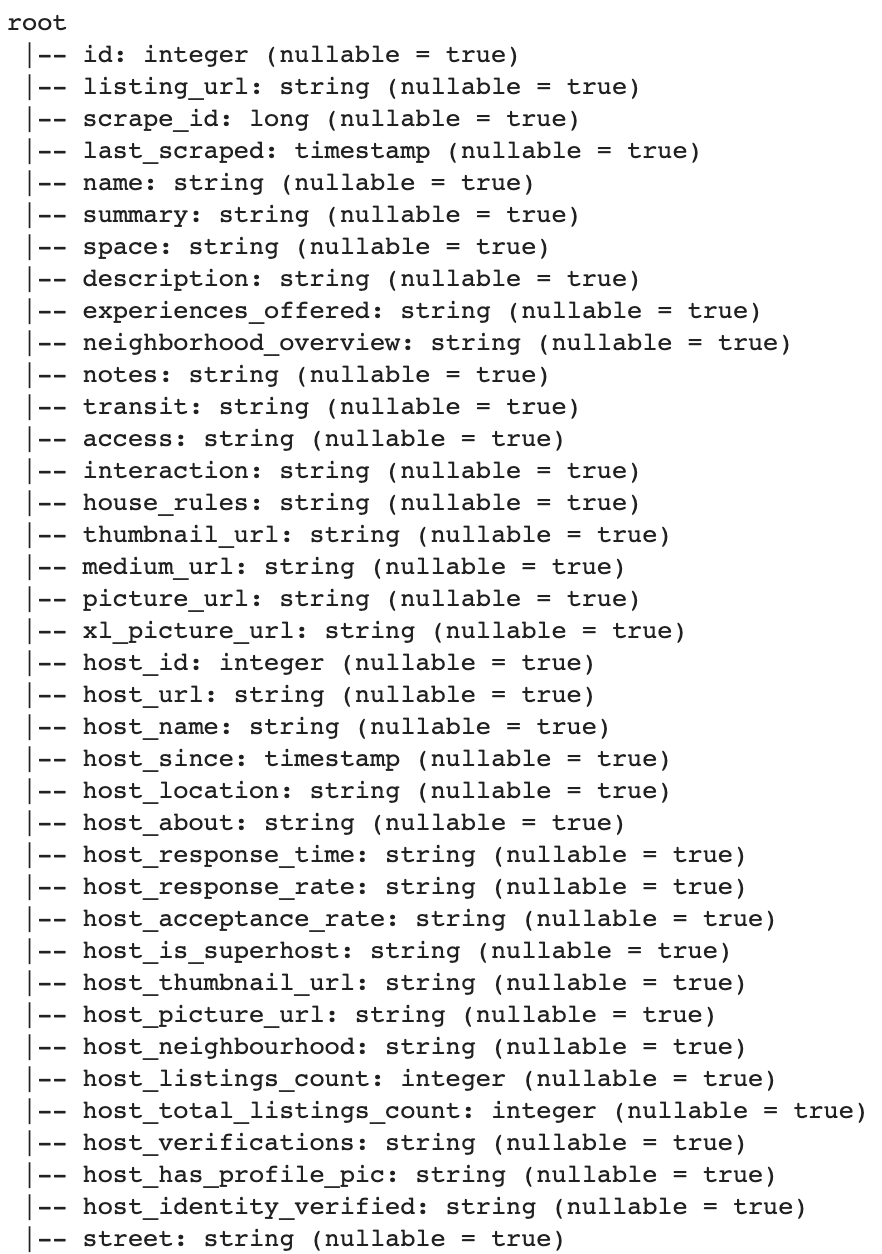
# Introduction

## Dataset

### Data Source

The Airbnb dataset contains 106 columns and 9024 rows, which is available at[**http://insideairbnb.com/seattle/**](http://insideairbnb.com/seattle/). This website is an independent, non-commercial set of tools and data that provide a lot of the data of Airbnb which is really being used in cities around the world. By analyzing data from this website, we can use the regression analysis to get a better price prediction. Our focus is on Seattle, since Seattle is becoming one of the prominent developing cities in the Area of North America in recent years.

### Data Composition



*Figure 1 - Dataset Schema*

The complete dataset includes listing, reviews, and calendar. Listing consisted of 106 features with 9023 observations about the detail of houses, reviews has more than 40,000 records about the reviews of houses from 2008 to 2019, calendar indicates the availability of a particular house, which would not be used in this project.



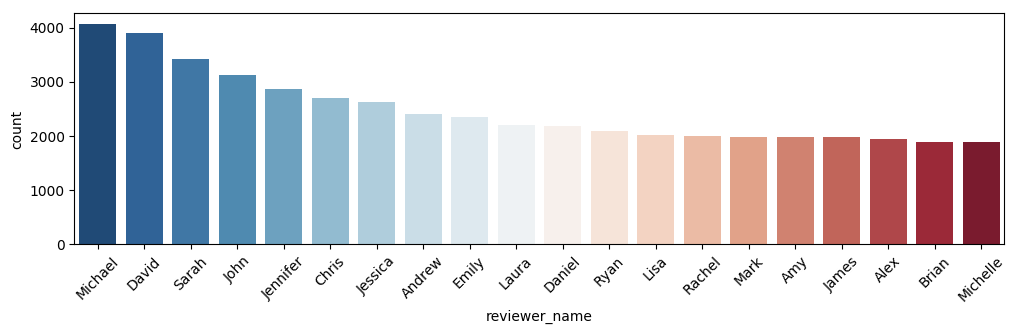
*Figure 2 - Columns of incorrect format*

The majority type of columns is number.However, some features have a string format, like this screenshot shown on the above. Data conversion is needed. And some features are categorical variables that can be turned into number format also. To get cleaned data, we built a series of transformers, and injected them into the cleaning pipeline, so that team members can use this pipeline to train and test the dataset in a unified way. Some unstructured records are also available in this dataset, such as the user review, amenity. So we separated the data into 2 subsets. One is prepared for structured data analysis, which only has numerical features. Another one is unstructured text features. It will be used as one part of sentiment analysis.

## Data exploration

### Who is the frequent customer?

First, we counted the total number of the records by the user individually. From the plot we learned that, during the year from 2008 to 2011, the number for the most frequent customer is 4061, which indicates that this should be an organization, such as the travel agency or a conglomerate. In this case, there would be a potential submarket to be further explored.

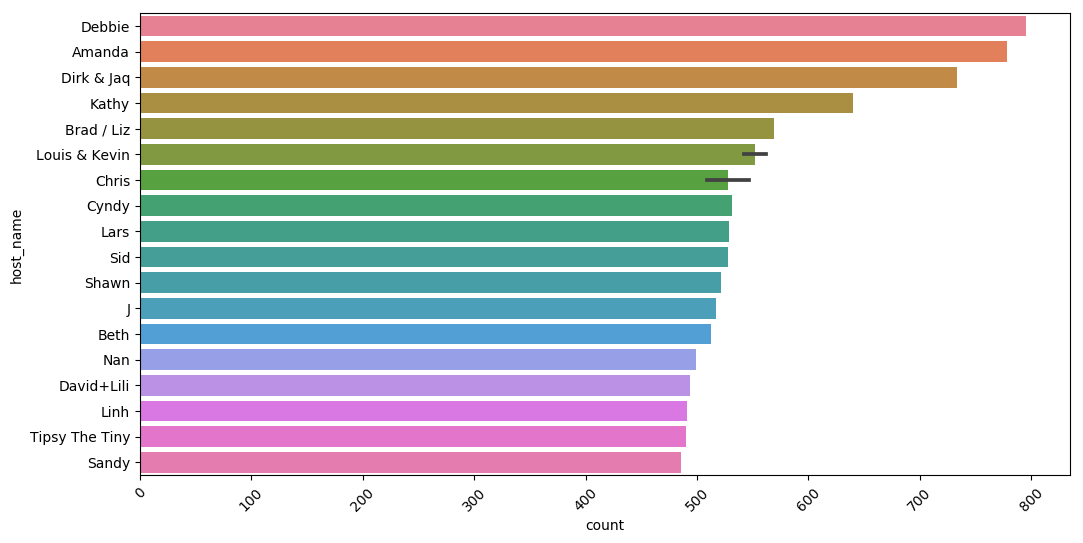


*Figure 3 - top 20 loyalty customers*

### Which house is on fire？



*Figure 4 - top 20 popular houses*



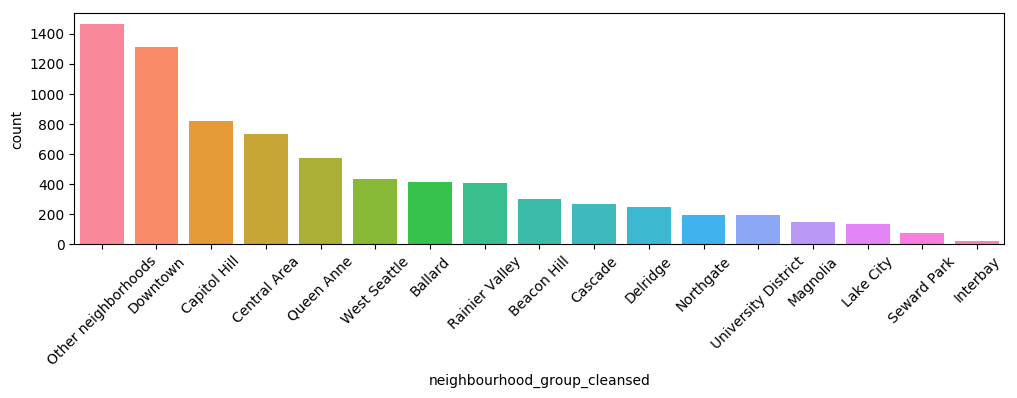
*Figure 5 - top 20 best homeowners*

Second, we visualized the frequency of each house. It helps to answer below 2 questions:

1. Which house is the most popular?
2. The occupancy of a house in a particular period?

According to the bar chart above, the average number is 795, which indicates that it is valuable to be further explored. So we joined the result with its concrete username(Figure 5). The hottest house host is Debbie. This host must have a unique property. This will guide the market team on the way to carry out the investigation on the quality of host services.

### Which place is most popular？



*Figure 6 - popular living area ranking*

Also, we analyzed the geographical location of the houses used. Except for other neighborhoods, downtown is the most popular place which people like to live in Seattle. This aligns with common sense, so the dataset has a reasonable quality.

## Business Questions

The business questions we are going to resolve are based on the Airbnb hosts and stakeholders demand. The pricing strategy and users’ feedback could be crucial for them. So the business questions we raised are that:

1. Which features may have a big impact on the price?

2. What is the users’ attitude towards the experience in Airbnb in Seattle?

By answering these two questions we could draw conclusions and give hosts and stakeholders a couple of marketing recommendations. The main objective of this project is to get the analyzing result of the above business questions. We will walk through the dataset to find out which features are important to price by using Linear Regression - Lasso, Elastic Net and Tree-Based Regression - Random Forest and Gradient Boosting Tree model. Besides, the overall feedback sense can be obtained by Natural Language Processing and Sentiment Analysis.

## Regression

Hosts do not always tag their house with a reasonable price based on their own experience. But Airbnb has a large volume of information about hosts and users, and we can mine the data and predict the price to create fair trade opportunities. Reasonable prices can attract more customers to use the airbnb to book their houses or apartments for their trip. Incidents may arise from time to time, like the Covid-19 this year. It is important to make the adjustment of price to cope with the market changes. This way, Airbnb can help to keep the retention rate, and even generate more for the hosts and itself.

## Natural Language Processing

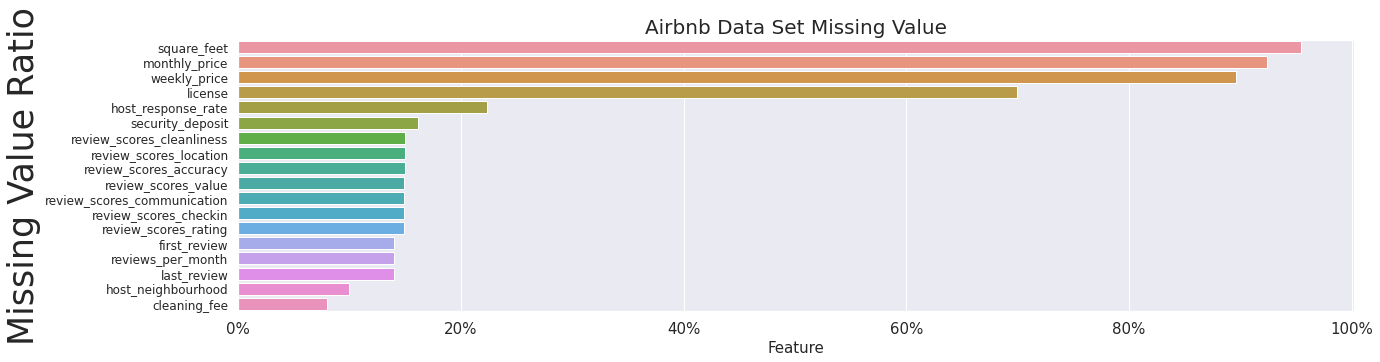
The feedback from our customers are the first-hand materials to be analyzed. Only if hosts get an overall attitude on them, can the hosts know the direction to improve their services and strategies. A 40000 comments on Airbnb in Seattle from guests is a valuable acquisition. By analyzing the term frequency in hosts description , we may find out what are crucial elements(words) hosts can use to emprise their property to get a higher rating score. And eventually, we found out that we have positive reviews nearly 40 times more than negative ones, which means most of Airbnb has done a decent job while some of them need to be improved.

## Recommender system

In the past few years, with the rapid rise of the big data industry, companies like Amazon, Netflix, Linkedin developed advanced intelligent recommender systems. They no longer just barely take the company’s own ideas as the recommendation. For now, the system can help to direct their products to the just-right customers according to the preference of the individuals. Thus, customers can get what they indeed-needs within a key strike. In the meanwhile, companies are able to improve their services and increase sales. So the recommender system is pretty significant for a company. Through the above data analysis we know, there are a lot of people using Airbnb frequently and the most popular house was booked 795 times, so this house must have something special to attract customers. Everyone has their unique preference that shares some common with a particular group of people. Therefore it is very important and intimate to recommend similar conditions of house to users. The system can help customers to book rooms to please them as much as possible.

# Analytic Methodology

## Data cleaning



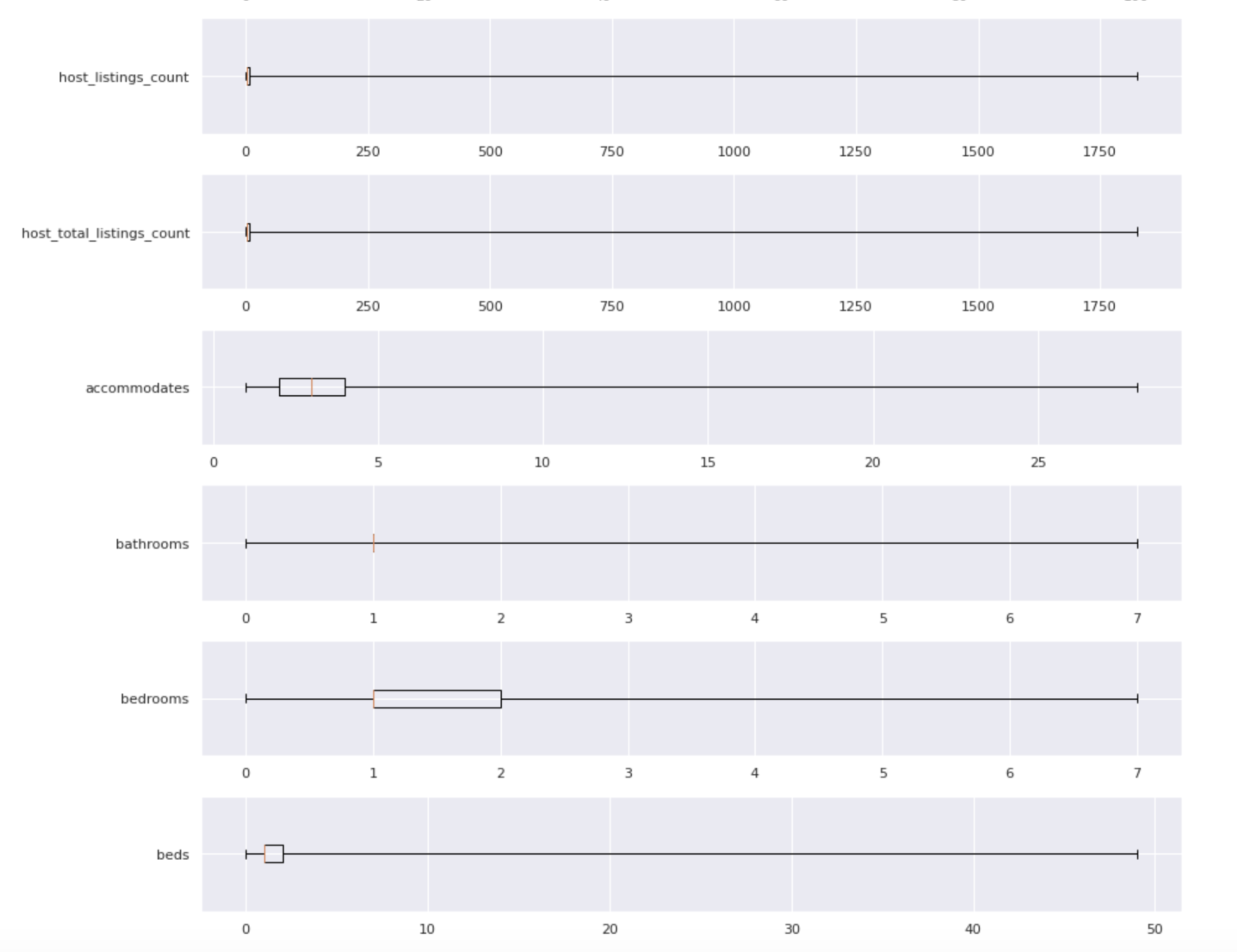
*Figure 7 - The ratio of missing value of each column*

To impute the missing value, we visualized the missing ratio of each feature. Some of the features lost over half of the records. Any approaches to refilling the data will introduce bias into the machine learning model because we do not know their original distribution.



*Figure 8 - Word cloud of customer feedback*

In addition, some of the columns share the same prefix review, which means they are unstructured text comments. We dropped these columns, but when processing sentiment analysis, we can retrieve them back. Since we use a pipeline to clean the data, every single module is reusable. And eventually, we replace the rest missing values with the mean for numerical continuous features, and the mode for the boolean features. The host response rate, security deposit, and cleaning fee are identified to be important to the price, that's why they are maintained and imputed.

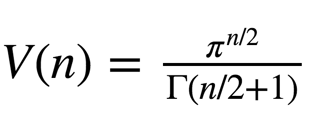
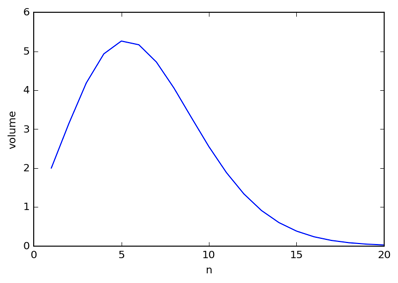


*Figure 9 - The distribution of skewed features*

Features may not be linearly correlated, to better identify the distribution, we calculated the minimum, maximum, mean, first, and the third quartile of each feature. And use this detail to generate the boxplots to get a better intuition of the feature distribution. The reason we were not using the box plot directly is that the data set may be huge in the big data environment. It is not possible to extract the entire dataset all at once. By box plot visualization, some skewed features are needed to be transformed with logarithm.

## Feature Selection

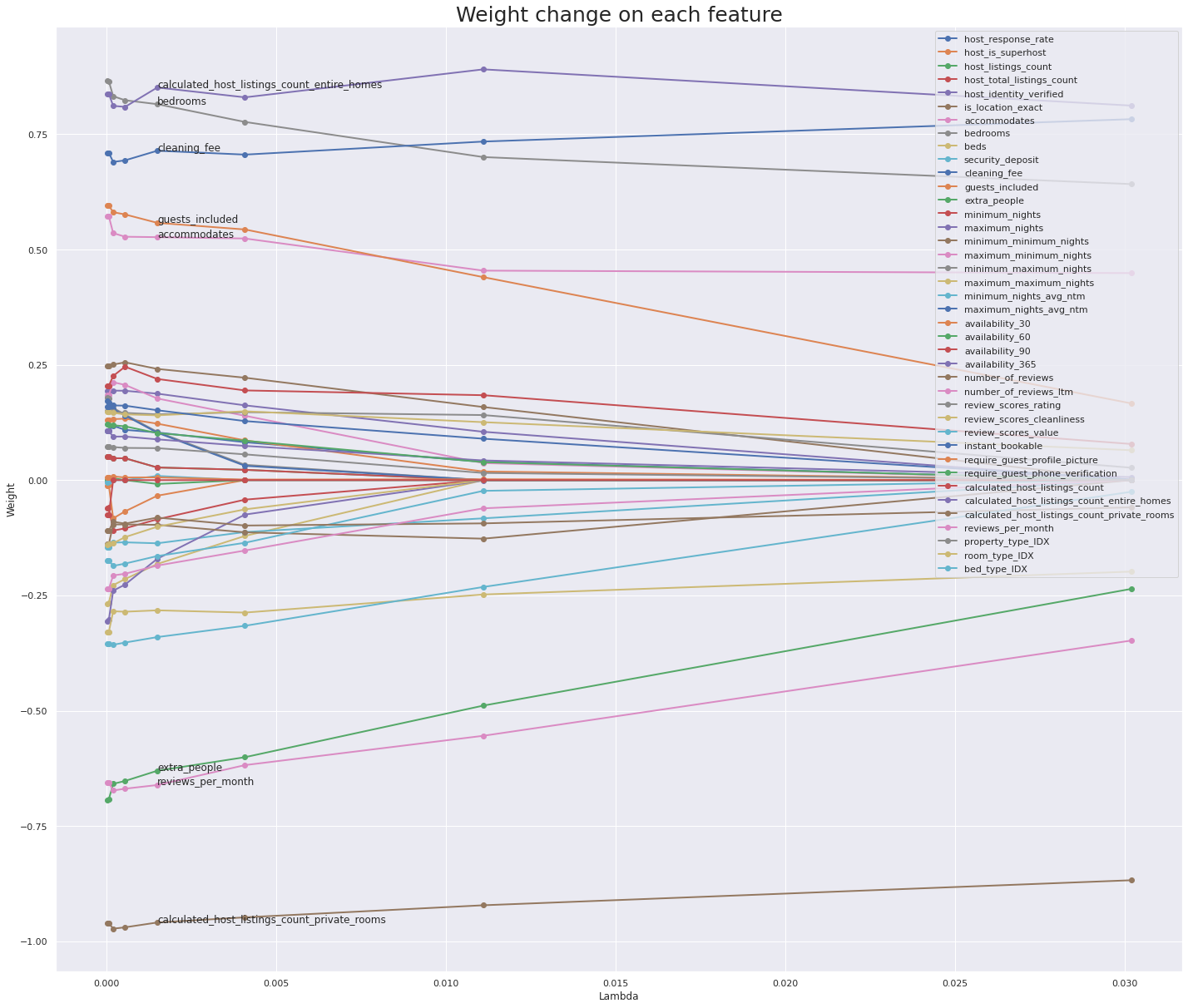
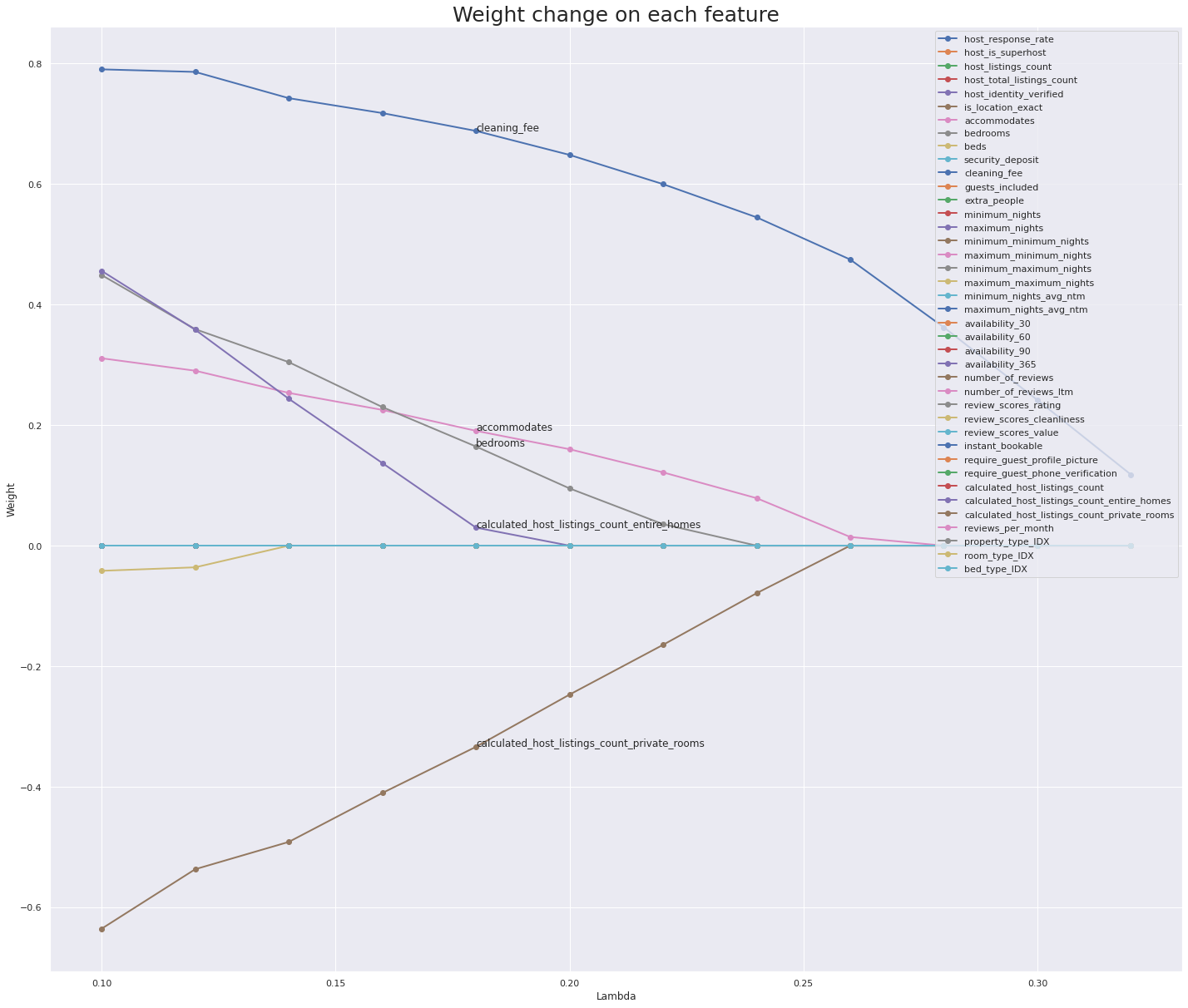
### Theory - the curse of dimensionality



*Figure 10 - Gamma function*

It gets harder to get a pair of similar observations, because the value of the gamma function decreases when the number of dimensions is close to infinity. Also higher dimensional dataset means a higher computational cost. That's why we need to carry out the feature reduction. Considering the huge volume of the dataset that comprised 106 features and over 9000 observations are far from feasible to be cleaned, we've taken a sample subset to build our feature selection model first.

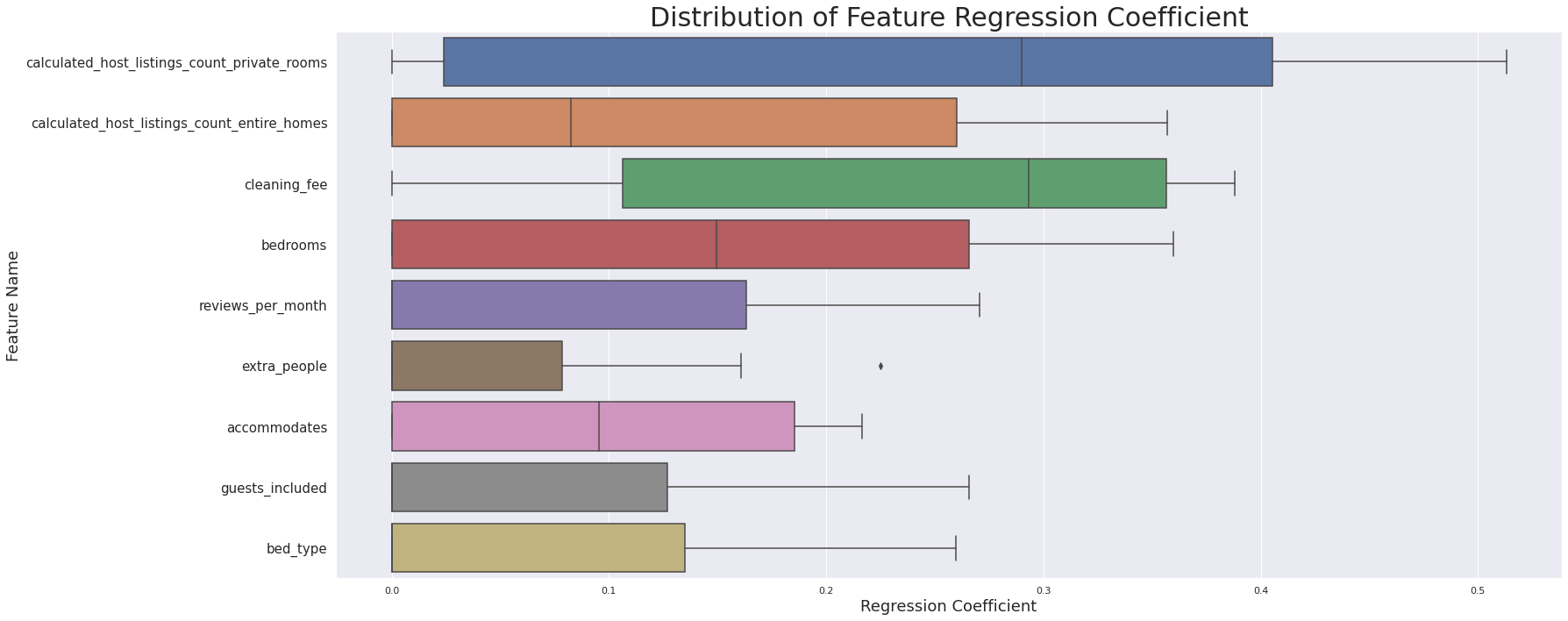
### Lasso Regression



*Figure 11 - Feature selection by Lasso regression*

For lasso regression, we extracted the regression coefficient against different lambda values. As the Lasso regression uses the L1 penalty term, it can be used for feature selection. We have run several rounds of parameter tuning exploratory. And finally decided to use the 0.4 as the threshold. And 9 features have been targeted as useful variables.

### Elastic Net



*Figure 12 - Stability of regression coefficient.*

The above boxplot is to show, for different degrees of penalty level, how stable the importance degree of each feature is. The elastic net was trained by the combination of parameters. We recorded the regression coefficient of each model. So features have a narrowed variation range, which means customers have a more unified attitude towards it. To future examine the determinant of the price, we’ve built a few models to get more insights of the data.

### Random Forest

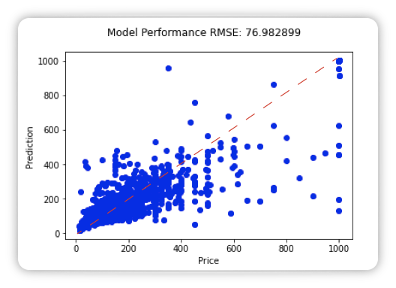
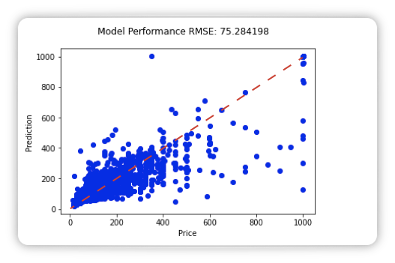
### 

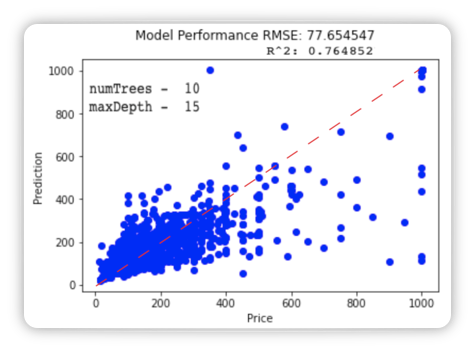
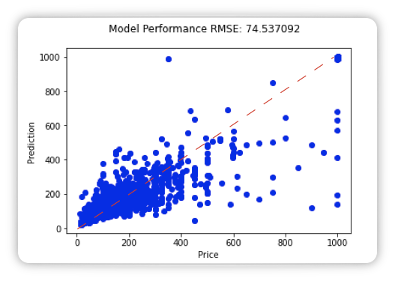
*Figure 13 - Feature importance by random forest regression*

The regression model of random forest helps to measure the feature importance. Categorical features, like ‘neighborhood group cleaned’, ‘cancellation policy’, ‘property type’, are transformed into ordinal variables by StringIndexer. After that, we used VectorAssembler to integrate all numeric variables into a feature. Above figure is a bar plot that displays importance for features in the dataset. Different thresholds(0, 0.01, 0.02, 0.03) have been applied to select features for regression models.

Using RegressionEvaluator, the pyspark build-in function helps to calculate the performance of R square and RMSE to evaluate my mode. We fully searched and tested the performance of models trained by the combination of hyperparameters using the functionality provided by the cross-validation. (number of trees as 5, 10, 15, max depth as 5, 10, 10).

Below are the 4 different random forest regression models by using different features to predict price.

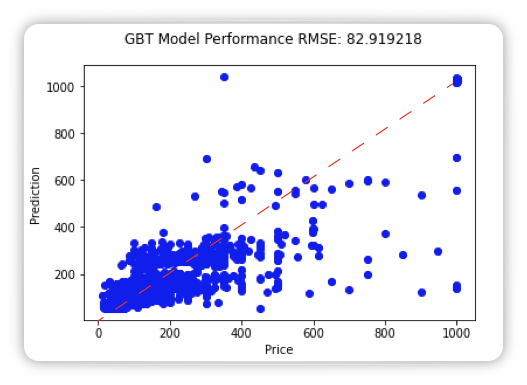
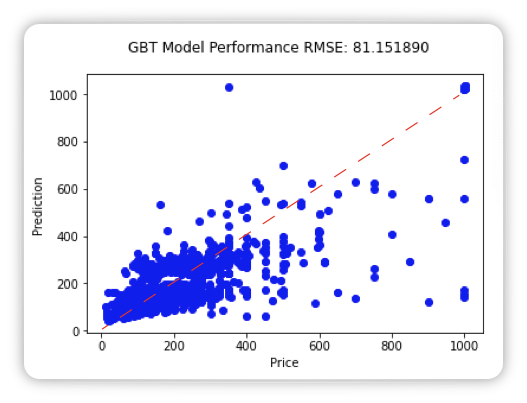


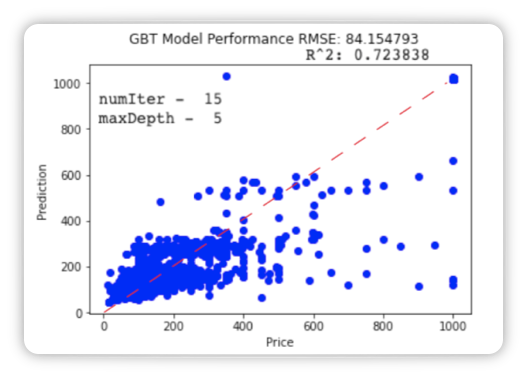
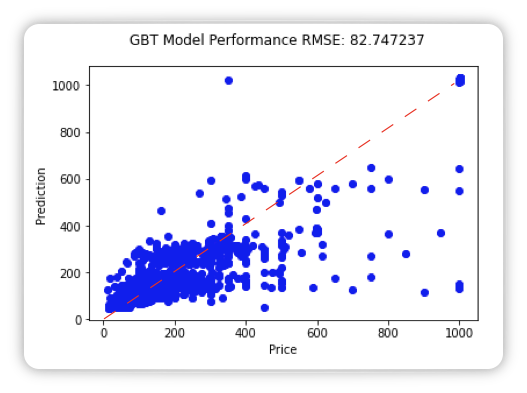


*Figure 14 - Random forest regression model*

We filtered the feature importance across different thresholds. The R^2 varies from 0.745 to 0.769. In fact, there are too many features selected for Random Forest Regression, some features are insignificant. After filtering, the feature has reduced from 52 to 15.

### Gradient Boosting Regressor

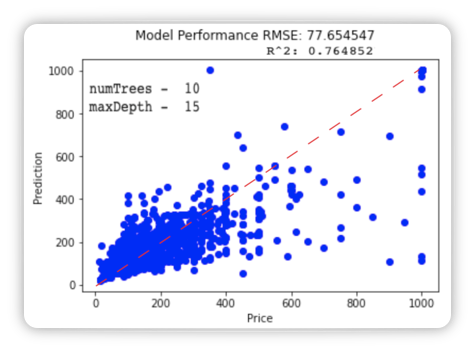




*Figure 19 - Gradient-Boosted Regression*

All the models of gradient boosting shares a close performance with around 0.75.

We’ve tried a few thresholds. It helps to reduce features from 15 to 5.



*Figure 35- Final model for predicting price*

After the comparison among models, we have decided this random forest regression to be the ultimate predictor of house prices. Its predictions are accurate and take a little time only. With this model, the hosts will know the marked reflection on their pricing strategy in advance. It is a toll to assist them on figuring out the way to adjust prices for special situations to get business success.

## Modeling

### Sentiment analysis

#### Description Term Frequency Analysis

In this part, we are going to get some insights from the description column by turning the unstructured data into the term frequency score. Our plan is to visualize TF-IDF score with a word cloud to demonstrate frequency words used in the description. First, we extracted rows that rating score equals 100 and marked them as high score hoster’s description from listing dataframe. We have 1732 rows in total which are plenty enough to analyze. By scrutinizing these words, we found there are five different languages which are English, Korean, French, Germany and Chinese. To better get an accurate analysis result, we extracted only English descriptions by the help of language detect package. Then I tokenized the text and removed stop words, and got the TF and TF-IDF score of each word in description. We listed the top 20 TF-IDF score words. But they have the same TF-IDF score and some of them are hard to interpret. So we used a word cloud to show the frequency of words.

### 

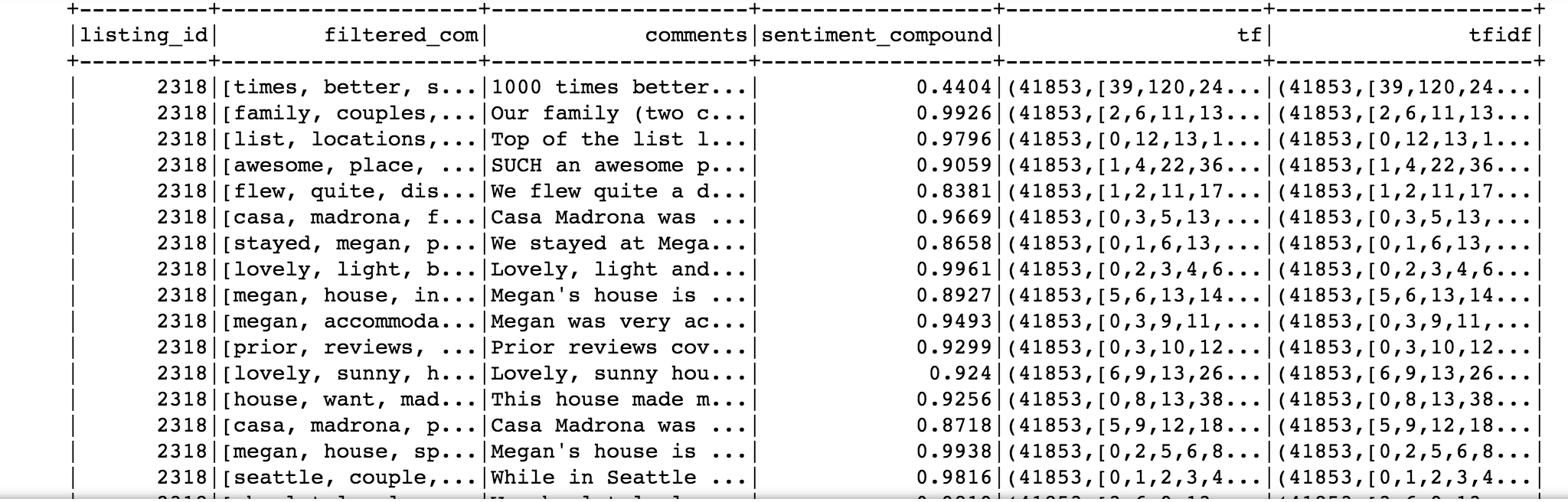
*Figure 23 - Term frequency and word cloud*

The above figure implies that ‘caphill’ ,’ caravan’, ‘tangerine’,’miranda’,‘John’ alone with other words are frequently involved in high rated Airbnb. So the data was combined with other pieces of information in the dataset:

1. The airbnb in Capitol hill tends to get a high rating score.
2. Caravan airbnb surprisingly appears the most in high rated description, which means the caravan type airbnb are popular.
3. Miranda,John might be a good host since her airbnb tends to get high review scores.
4. Tangerine (a spa hotel) could be a good Airbnb to choose.

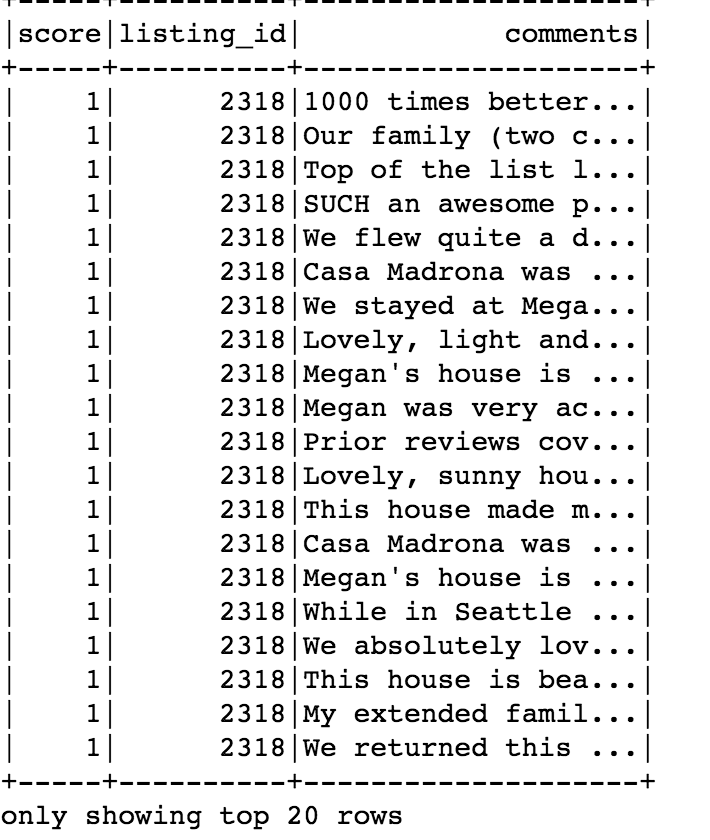
#### Reviews Sentiment Analysis

Since our intention is to get some marketing insights of airbnb, it seems crucial to know the attitude of guests towards the places they stayed so that we are able to get feedback to improve the service. So we used the review table to analyze the sentimental comments and detect the ratio of positive reviews against negatives. Besides, we have built a sentiment prediction model by using logistic regression. As the same thing we did in term frequency analysis. we extracted only English reviews and dropped all NA in review columns. Then, we found out that the reviews are not labeled, so we imported vader\_lexicon from nltk package. It is a ready-to-use tool that helps me give a compound sentiment score to each text. After tokenizing and removing stop words, we created a dataframe as below.



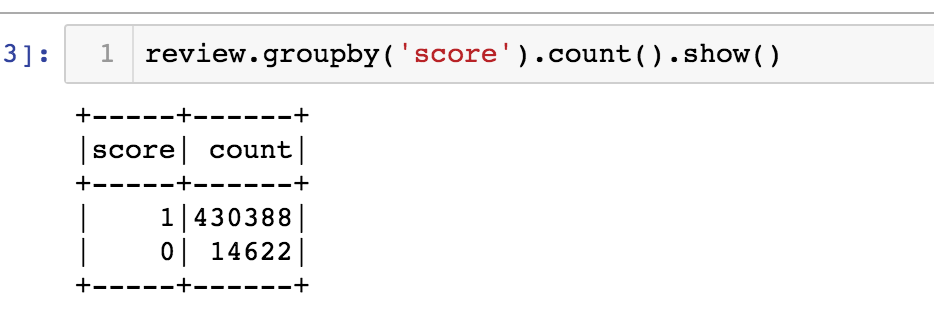
*Figure 24 - comments with sentiment compound score*

As every comment has a sentiment\_compound score now, it’s time to give them an appropriate label. The compound score given by vader\_lexicon means that when a comment is identified as a positive one. Given that case, the score will be positive, and negative otherwise. The score ranges from -1 to 1, the greater the absolute value, the stronger emotion it has. So we labeled the comments to 1 for the records with positive compound score and 0 otherwise. After these sequences of transformation, we created a new dataframe as below.



*Figure 25 - labeled comments*

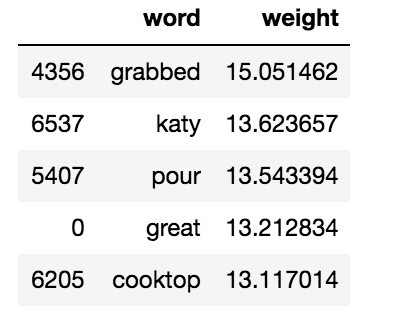
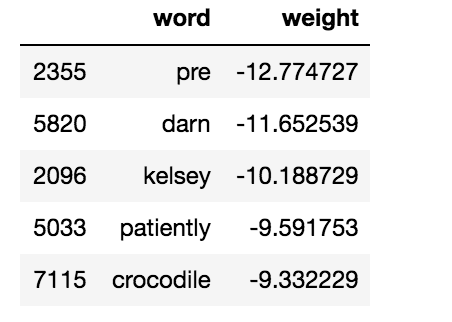
By grouping the label helps to explain how many positive comments or negative ones are there. Fortunately, there are 430388 positive reviews versus 14622 negative reviews, so we can draw a conclusion that most guests in Seattle Airbnb have a positive attitude.



*Figure 26 - sentiment analysis result*

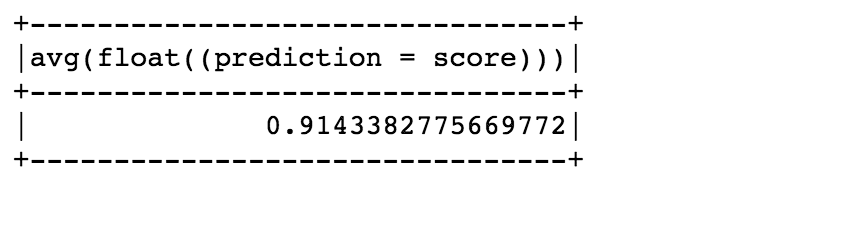
After sentiment analysis, we built a sentiment prediction model by using logistic regression.We sampled the dataset by selecting 14622 positive reviews out of 430388 to balance the data. The unbalanced data will cause underfitting and will cause problems when we tune the model. We used all default parameters of the logistic regression model for the first time and got an accuracy which is 0.856. It is an acceptable performance, while when I dig deeper, there are many problems.



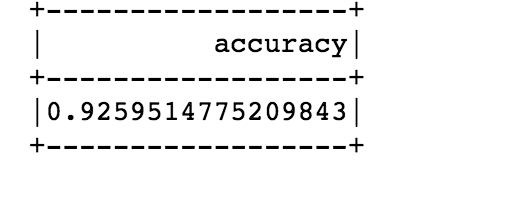
*Figure 27 - accuracy without tuning and the top/least 5 weight words*

The accuracy is reasonable, when diving deeper, there are many problems. As the top 5 and the least 5 weight words above, we found out that there is too much noise. Some meaningless words are contributing to the prediction. It will also cause overfitting since we have too many features compared to examples. So the next thing is tuning the model by using Elastic net regularization to punish the meaningless words. I used lamda 0.02 and alpha 0.3 for the first time for testing if it works and got accuracy 0.91.



*Figure 28 - accuracy after tuning with random parameter*

It seems we improve the performance a little bit, but there are still many insignificant words (with no importance at all) that contribute to the analysis, actually 96% of our features are useless. So we use ParamGridBuilder to find the best lambda and alpha value. After setting the lambda as 0,0.01, 0.2 and alpha as 0,0.2,0.4 and grid the model. I found the best performance is 0.92 when lambda equals 0.01 and alpha equals 0.2.



*Figure 29 - best accuracy*

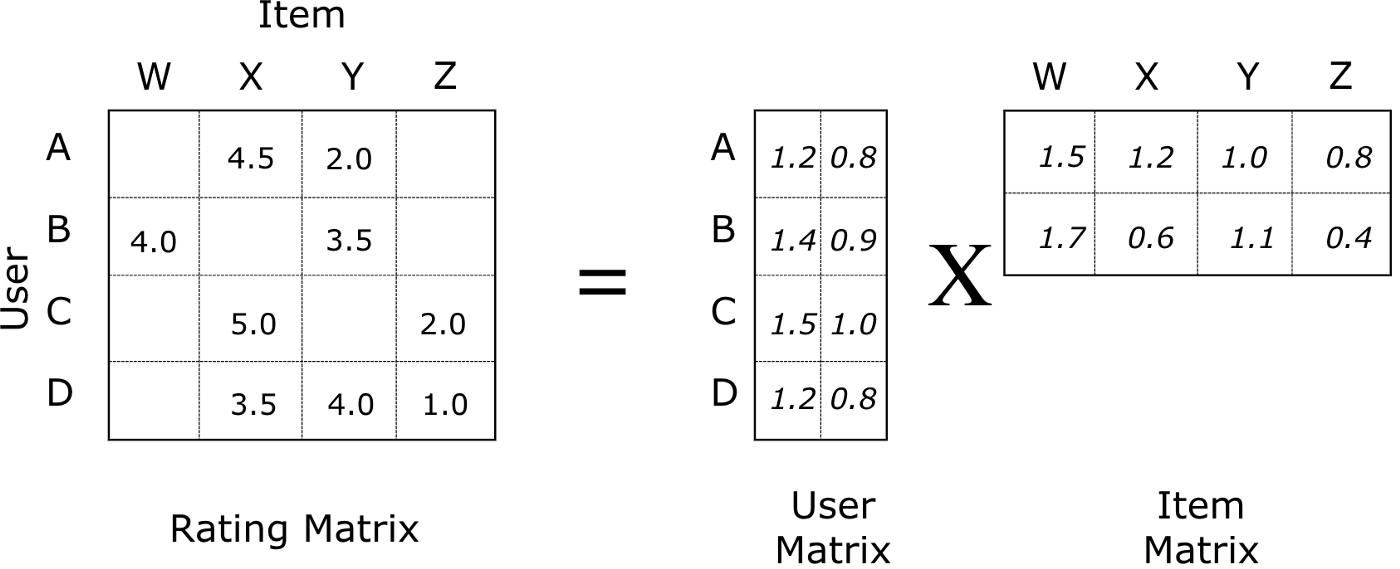
### 

In the databrick platform, to decrease the time consuming process and get a comparatively accurate result ro demonstrate. So we subset 5000 samples instead of whole comments, and got the same result at a high level . We have 4797 positive reviews and 109 negative ones. And I increase the model accuracy from 0.58 to 0.75 with the same lamba 0.02 and alpha 0.2.

## Recommender System

#### Collaborative filtering

Collaborative filtering is to use the preferences of a group of similar interests and common experience to recommend information that users are interested in. Users give a considerable degree of response, such as ratings, and then to filter out items that users are not interested in, so as to meet users’ real needs.It is commonly used for recommender systems.it is not possible that everyone used every item and gave a rating in a real life, so the user-item association matrix would have a lot of zeros, as you can see in figure 30. Collaborative filtering can help fill in the missing entries of this user-item association matrix. The first thing is to calculate the similarity between the evaluated item and the item to be predicted, and use the similarity as the weight to weight the score of each evaluated item to obtain the predicted value of the item to be predicted. For example, to perform similarity calculations on project A and project B, then to find the combination that scores both A and B, and perform similarity calculation on these combinations.



*Figure 30 - Collaborative Filtering*

Spark currently supports model-based collaborative filtering, in which users and products are described by a small set of latent factors that can be used to predict missing entries. It uses the alternating least squares (ALS) algorithm to learn these latent factors. A user-item pair with an interactive relationship accounts for only a small part. In other words, the user-item relationship list is very sparse. Unlike SVD matrix decomposition, the matrix decomposition technique used by ALS does not need to fill the coefficient matrix into a dense matrix before decomposition, which not only greatly reduces the storage space, but spark can use this sparseness to use simple linear algebra calculation and solution. These points make this algorithm calculate very fast on large-scale data, explaining why spark mllib currently has only one ALS recommended algorithm.Based on the natural language processing analysis of Chenghao, I use the ALS algorithm to solve Matrix factorization, the item is the house, and the rating is the score of reviews.

#### Based on Items

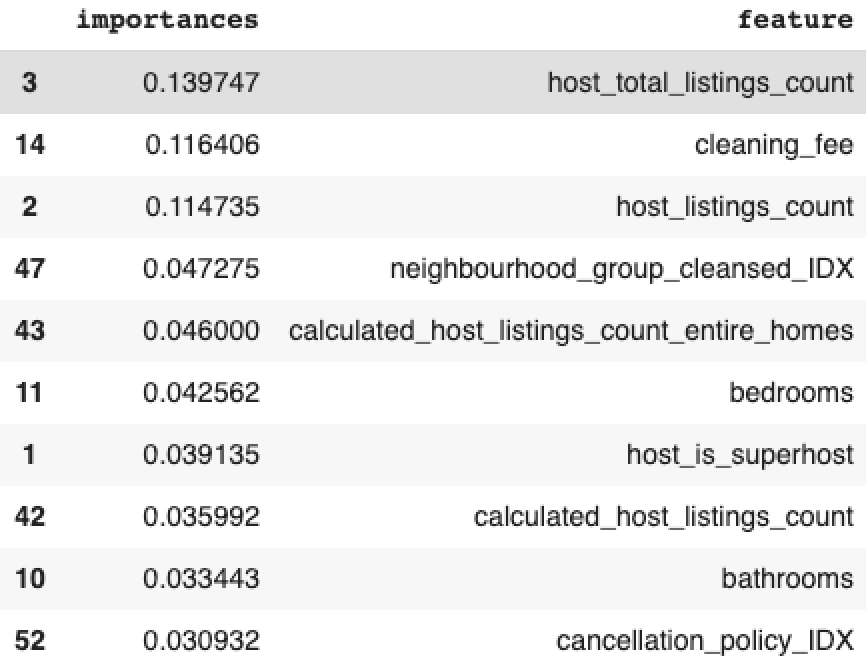
To get more insight of the data space, we built another recommender system based on the description of house by computing the distance between each two houses in the top 100 most popular house in Seattle.we selected the columns which we would use to a new dataframe, they are ‘listing id’, ‘listing url’, ‘host\_name’, ‘description’, then make this dataframe join with the data frame of most living which we did before, because the calculation of distances of all houses are so large that the computational cost is too large to be accepted. In addition, most people would like to live in a house which is the top popular, so we just used the top 100 popular houses to recommend.

First, we tokenized the description of houses further to filter stop words and changed them into TFIDF, then standardized and normalized the processed data, and imported them to KMeans for clustering, we set the number of clusters is 10 because we want to split 100 houses into 10 groups, last is to do PCA. All of the above steps are made by pipeline, so that’s really easy and convenient to build this recommendation model. At last, by using a query of dataframe and UDF function to get the final recommendation list for users.

# Results

## Feature Importance

Using these features in the elastic net regression, we conclude that the price is more sensitive to the room's cleaning fee, which means the cleaner the room, the higher class of the houses it is. Guests tend to have a unified attitude towards the extra people. People may request a flexible arrangement, which means vouchers rewards would please a particular group of guests. This is a cost to Airbnb, but it helps to maintain the retention rate. Bed type, accommodate, bedroom and all other features indicated that comfort is a determinant for a guest to choose a house. And they could measure it by gauging the review per month.



*Figure 31- Random forest model feature selection*

The random forest yielded a slightly different set of feature importance. The most important feature is host\_total\_listings\_count. This feature means that the hosts with more property provide a better (or more expensive). This evidence endorsed the preliminary exploration we did in the first place. Like the Elastic Net model implies, cleaning determines the quality of service, which was reflected onto the price. neighbourhood\_group\_cleansed reflects the location of hosts’ house or apartment, particular districts will have higher prices. More rooms and bathrooms mean higher prices. What’s more, if hosts are superhost, that means they provide better service to customers, the price is higher. Cancellation\_policy is also important. Some properties have very flexible cancellation policies, that means hosts need to take a higher risk. So flexibility affected the pricing of the houses.

## Regression Mode Performance

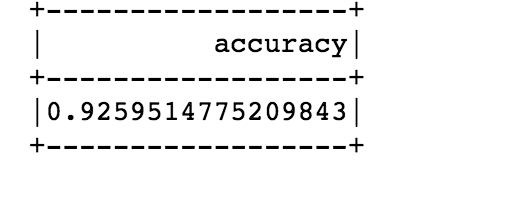
The RMSE of the final random forest regression model is 77.654547.The R^2 of the final random forest regression model is 0.764853. we randomly split data -- 70% training data, 30% testing data. We used cross-validation and added hyperparameters to get the best model. Each time we get 9 models, they have different performances. Because the data has been splitted into 3 different folders, this way can solve overfitting or underfitting problems.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Hyperparameter | Value | RMSE | R^2 | Running time |
| Random Forest Regression | Num of Trees  Max Depth | 10  15 | 77.654547 | 0.764853 | Around 8 minutes |
| Gradient-Boosted Regression | Num of Iteration  Max Depth | 15  5 | 84.154793 | 0.723838 | Around 40 minutes |

Random Forest is much better than Gradient-Boosted Regression for predicting price. Because the random forest model has lower RMSE values. What’s more important is that the random forest model cost time much less than that of the GBT model.

## Natural Language Processing

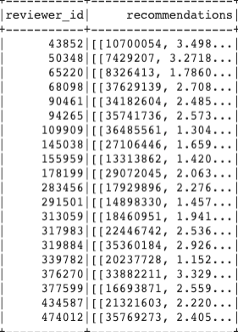
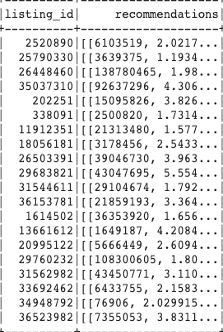
As what has already been stated above, sentiment analysis shows that we have positive reviews way much more than negative ones. We have 430388 positive ones and 14622 negative ones. For the prediction model, we achieved an accuracy which is 0.9259 by using a logistic model. Elastic net regularization tuning is used to tune the model, the first time accuracy with not tuning is 0.856. The first problem I met is the overfitting caused by unbalanced data since we have too many positive reviews. So I fixed this issue by randomly selecting the same number of positive labels as negative ones. Another problem is that we have too many features compared to examples ,which will also result in overfitting. For the model tuning I used L1 and L2 regularization to punish the terms that are meaningless. And use gridbuilder to select the best parameters lambda and alpha. The best of them are 0.01 and 0.2.



*Figure 32- accuracy before and after tuning*

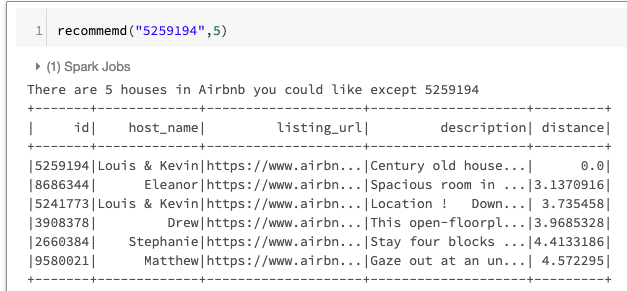
## Recommender System

After building this recommender system, I can recommend each user 10 houses by using “recommendForAllUsers” which is already assembled in spark api , and 10 potential users are selected for each house by using “recommendForAllItems”. It takes 18 minutes in the training process and prediction. I use RMSE (Root Mean Square Error) to calculate the distances between the house which users could live in and the house I recommend to them. The result of RMSE is 0.6588.

*Figure 33 - Collaborative filtering recommendation results*

In my second recommender system, I just need to input listing\_id and how many other houses you want from the recommender system, then you can get the houses you probably like. As you can see from figure 34,this recommendation result return 5 other results, the closest distance is 3.137 and the second closest distance is 3.735, and the homeowner of this second result is Louis & Kevin who are same people as the house we want the recommender system to recommend, houses that are belonged to same people could probably be same, so the result is fairly good.



*Figure 34 - Results based on items to recommend*

# Conclusion

To please the guest, the comfort and class of the room should be the thing that catches their eyes. And for a different member, we should have a different accommodation to meet their expectations respectively.

Overall feedback of the guests is positive, it means the hosts should keep their current service level and maybe only need to improve little. The airbnbs in Downtown and capitol hill tend to get high review scores, we can suggest people who tend to set an Airbnb start their business in these regions. Caravans Airbnb surprisingly has a high rating score, so people could own a popular Airbnb business easily if they have a caravan.