Airbnb Vacation Rental Price Prediction

Hareesh P. Bahuleyan (2067 6609), Suryanarayanan P. H. (2067 9272)
Course: MSCI 723 - Big Data Analytics

April 8, 2018

Abstract

In this project, we study a business problem faced by residential property owners who wish to rent out rooms on the online marketplace Airbnb. Airbnb connects hosts who list home-stay accommodations to travelers looking for short-term rentals. With the help of information including property type, amenities provided and neighborhood facilities, we develop a regression model to predict the optimal price, for which a newly registered host can list his/her property. The listings data is available on the Airbnb website and is scraped for analysis, training and validating the model. Additionally, we mine for patterns in the data with unsupervised machine learning approaches such as association rule mining and clustering. Finally, we also study this problem as a binary classification task, where we use different variants of Naive Bayes algorithm to classify cheap versus expensive listings.

Contents

1 Introduction

1.1 Business Problem

Airbnb is a vacation rentals online marketplace where house owners (hosts) in different cities can list their properties for short term rentals. Travelers find this as a cheaper alternative to hotels coupled with a home-stay experience.

The business problem that we are trying to solve benefits hosts. Assume that Mike has an apartment to rent in the city of Toronto. As a property owner, Mike would be competing against 1000s of other hosts to get travelers to book his apartment. If Mike was to post his advertisement for the first time, he would be unsure of what price he should rent it out for. He could manually look for 'similar' postings in his neighborhood and decide on a price. However, this would be time consuming and not necessarily accurate.

1.2 Motivation

Airbnb has massive amounts of data about listings and their prices. This data could be used to develop a pricing model that would assist hosts like Mike to set an optimal price. The input information required by the pricing model should be easily available and understood by hosts. We hypothesize that there are two major factors that affect the price:

1. The amenities that a rental listing offers to a traveler. This can range from the number of bedrooms in the house to the availability of a pool or gym. Hosts can charge a higher price if they are able to provide better amenities.

2. The facilities in the neighborhood of the listing. If the house is located in a prime neighborhood with most facilities within a walkable distance, then it is likely to be priced higher.

A host would surely be aware of what amenities are available in his/her house. And for the neighborhood facilities, we can determine this information with OpenStreetMap/Google Maps, since the address of the listing is known. This also means that such a pricing model based on the above factors would be easy to use for hosts. Moreover, this information for the existing listings can be obtained by scraping the web which would facilitate the model building process.

Airbnb has been growing at incremental rates due to its disruptive business model. This means that the number of hosts as well as travelers who sign up on their website will increase with time. Our solution would benefit the newly registered property owners to set the right price so as to remain competitive in the rentals market and being profitable at the same time.

1.3 Related Work

The prediction of housing prices using linear regression dates back to as early as 1974 when Richardson et al. (1974) claimed that location and distance based variables play an important role along with house-specific characteristics in determining the price of a house [1]. Jorge

used multi-variate spatial methods kriging and cokriging to predict location price through continuous maps providing appraisers an overall view [2]. Jakob proposed a model using economy-wide national income, mortgage repayment and nominal lending rates [3]. This model is used to predict house prices in OECD countries. Mitra and Suman proposed a model using 14 independent variables to predict house rent using artificial neural networks [4]. When compared with hedonic price models, the proposed model gave better results. Lynn and Erik provide a method to predict the housing market trends [5]. They even demonstrated how Google predicts future business activities.

Most of the studies have addressed the issue of house price prediction with respect to a buyer/seller permanently moving in to a locality. However, Airbnb is a relatively new concept for short term rentals. Wu, Zhou and Li used gradient boosting model combined with one-vs-the-rest strategy to predict Airbnb's new user's first destination [6]. The prediction model was evaluated with various data sets by NDCG (Normalized Discounted Cumulative Gain). Hamel et. al scraped data from the Airbnb website to predict rental prices using apartment specific features [7]. However, their model lacked location specific features. Considering 25 explanatory variables in 5 categories for more than 180,533 listings from 33 cities in Airbnb, Dan and Juan identified the price determinants for sharing economy based accommodation [8]. Li et al. (2016) created clusters from Airbnb housing data with the help of information about landmark and facilities [9]. The authors then used a separate linear regression model for each cluster to predict the house prices. Emily and Kunal used InsideAirbnb's full listing data for San Francisco city to predict a listing's neighborhood and its price using text and image features along with other relevant variables [10].

Very few studies in the literature have looked at rental price prediction using Airbnb data, of which, most of them used the partial data provided at InsideAirbnb. We propose to use a lot more explanatory variables which can be obtained only by directly scraping the listing pages. Along with that, we use neighborhood amenity information by combining data from OpenStreetMap API.

2 Data

2.1 Data Collection

The data for this project has been obtained from 3 sources as described below:

2.1.1 Inside Airbnb

Airbnb provides opensource data about their listings on InsideAirbnb. We downloaded the CSV files for the listings in the city of Toronto. This file had information about the listing (URL, price, number of beds, latitude, longitude, description, etc.) and the host (name, response time, listings count, etc.). We also obtained the traveler reviews for these listings from this data source. In total, there were 12,029 records.

2.1.2 Web Scraping

A listing posted by a host on Airbnb includes information about the listing like number of bedrooms, number of guests, availability of amenities like Wifi, Cable TV, air conditioning, parking space and many more. A screenshot of a sample listing page is shown below in Figure 1.

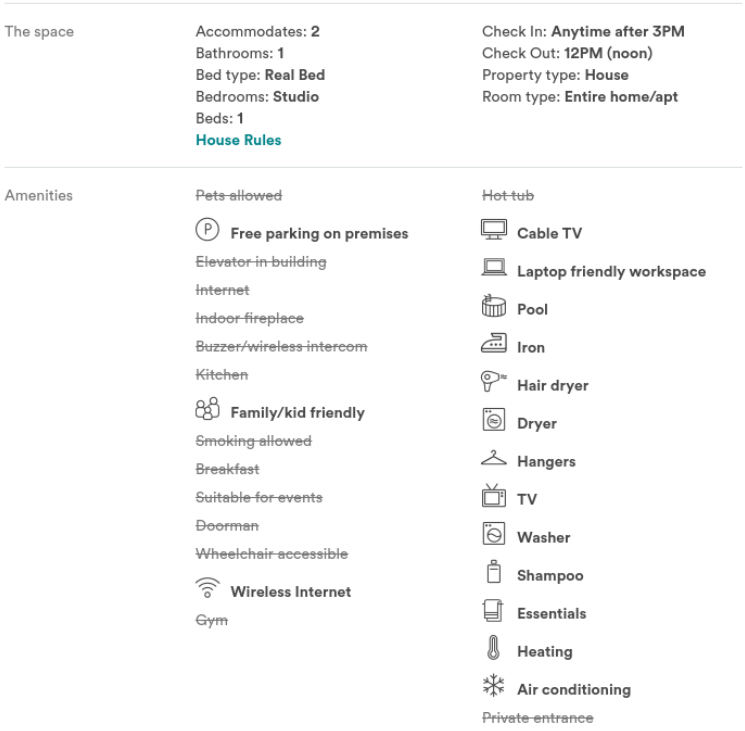


Figure 1: Sample Listings Page on Airbnb

For the purpose of obtaining amenity information of each listing, we had to crawl the data from the listing HTML pages. This was implemented through a Python program utilizing the library called Selenium. On using a standard web scraping library like BeautifulSoup, we were being blocked by the Airbnb (because the website hit was being made by a computer). On the other hand, Selenium allows us to access the website content by mimicking it the way a human would do it, i.e, open a browser first, enter the url in the search box, click certain buttons and download the required portions of the HTML page. We were able to scrape the amenity details for 9,795 of the original 12,029 listing records through this procedure which took 40 hours in total (done over multiple sessions). Some of the listings could not be scraped because either they were removed from the website by the hosts or they had a different HTML structure in which case the package failed to parse the webpage.

2.1.3 OpenStreetMap API

In order to collect the neighborhood data (such as distance of the house to the nearest bus stop), we made use of the latitude and longitude information of each listing. The goal was to use a maps API to automatically locate the nearest bus stop (and other neighborhood

facilities including restaurant, atm, cinema, hospital, nightclub, park, mall, gallery, museum, supermarket). We chose these facilities as we felt that these would be points of interest to a traveler in the city.

Figure 2: Amenities in the neighborhood of House No. 772 Richmond Street West

Our first attempt was using GoogleMaps API. However, the free version of this API has constraints on the number of hits allowed. This made us switch to OpenStreetMap, which is a free and open source alternative. We implement a Python program to query for the presence of the nearest, say bus stop, within a pre-specified radius, with the help of the Overpass API. Overpass has its own syntax to query the OpenStreetMap database to search for the presence of predefined facilities in the neighborhood of a given latitude and longitude. If the particular facility was present, then we calculate the distance in meters to the facility. Else, if it was not present within that radius, we assign an indicator flag of -1. This was completed in multiple sessions and took about 35 hours in total.

2.2 Data Description

The data obtained from the sources discussed in the previous section can be grouped in 3 categories of explanatory variables. This section describes the variables that come under each category along with information about the data type of the variable (whether numeric or categorical when it was scraped. Some of these numeric variables were later converted to categorical for specific tasks, which would be discussed in the respective sections). Note that `price` is the target variable which is numeric for the regression analysis and is converted to binary for the classification task.

2.2.1 Listing/Host Characteristics

1. `minimum_nights` (Numeric): Minimum number of night for which the booking has to be made

2. `cancellation_policy` (Categorical): Whether strict, flexible or moderate

3. `number_of_reviews` (Numeric): Number of reviews for the listing

4. `instant_bookable` (Binary): Whether the booking can be made instantaneously
5. `calculated_host_listings_count` (Numeric): Number of listings posted by the same host
6. `num_page_saved` (Numeric): Number of travelers who have saved this listing
7. `property_type` (Categorical): House, Apartment, Condominium, Tent, etc.
8. `room_type` (Categorical): Entire House or Private room or Shared room
9. `bed_type` (Categorical): Whether real bed or other types like couch, airbed, etc.
10. `accommodates` (Numeric): Number of guests that can be accommodated
11. `bathrooms` (Numeric): Number of bathrooms
12. `bedrooms` (Numeric): Number of bedrooms
13. `beds` (Numeric): Number of beds

2.2.2 Amenities

The list of amenities provided at the house/apartment are binary variables - 1 if present 0 otherwise. These include:

1. Elevator in building
2. Internet
3. Family/kid friendly
4. Wireless Internet
5. Buzzer/wireless intercom
6. Kitchen
7. Doorman
8. Wheelchair accessible
9. Cable TV
10. Hot tub
11. Gym
12. Pool
13. TV
14. Dryer
15. Washer
16. Essentials
17. Shampoo
18. Heating
19. Air conditioning
20. Pets allowed
21. Suitable for events
22. Smoking allowed
23. Indoor fireplace
24. Breakfast
25. Laptop-friendly workspace
26. Iron
27. Hangers
28. Hair dryer
29. Private living room
30. Private entrance
31. Parking

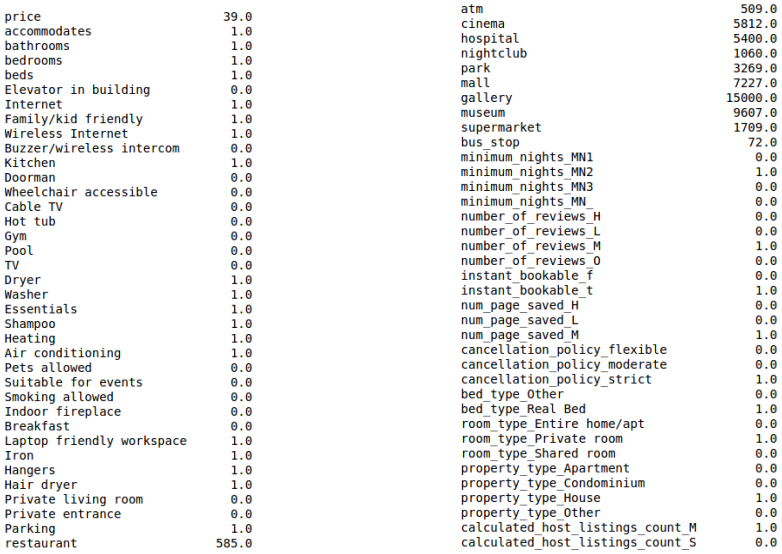


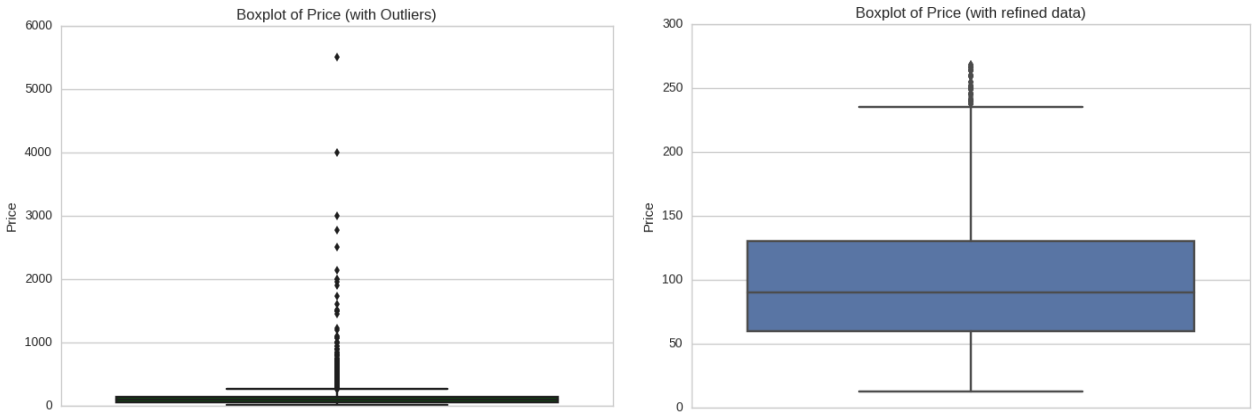
Figure 3: Sample record in the dataset

Table 1: Search Radius for Neighborhood Facilities

Facility	**Search Radius (km)**
`restaurant`	2
`atm`	5
`cinema`	15
`hospital`	15
`nightclub`	10
`park`	10
`mall`	10
`gallery`	10
`museum`	10
`supermarket`	5
`bus_stop`	1

2.2.3 Neighborhood Facilities

These variables represent the distance of the house to the corresponding facility and hence are of numeric data type. The search radius while querying in OpenSteetMap API was set in a logical format, i.e., a higher radius allowance for uncommon facilities like gallery or museum and a lower search radius for more common ones like atm and bus stop. The exact values used are shown in Table 1. A sample record of the data is shown in Figure 3.



(a) Before Outlier Removal

(b) With Refined Data

Figure 4: Box Plot of Price

3 Exploratory Data Analysis

EDA is an important first step which has to be carried out before modeling the data. There can be some patterns, trends or insights which we might be able to uncover either by simple descriptive statistics or visualizing the data in the form of different charts.

3.1 Dependent Variable

First, we focus on the dependent variable, which is price in our case. The mean value was found to be $123 per night. However, with a box plot, we we able to detect the presence of large number of outliers (4). The maximum price was surprisingly around $5500 per night for one listing, which was indeed a very luxurious resort style home. The presence of such data points would shift the mean up by a considerable amount and affect the model fitting process. Hence, data had to be cleaned before proceeding further.

3.1.1 Data Cleaning

We adopted the common technique of outlier removal based on inter quartile range. This is done by first calculating the inter-quartile range which is the difference between the 75^{th} quantile and the 25^{th} quantile.

$$IQR = q_{0.75} - q_{0.25} \tag{1}$$

Any data point that has a value greater than 75^{th} quartile $+ 1.5 * IQR$ or 25^{th} quartile $- 1.5 * IQR$. Based on this criteria, 563 records were discarded. The refined dataset had 9232 records with the price variable now having a mean value of $101 and a median value of $90. The maximum price was $268 and the minimum was $13, and the standard deviation was $52. The box plot of price in the refined data is illustrated in Figure 4.

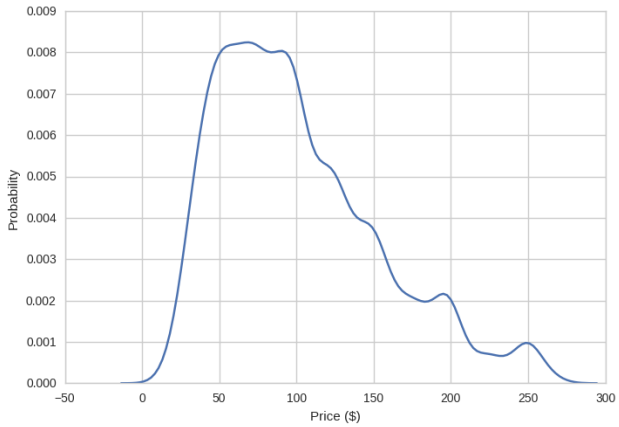


Figure 5: Distribution of Price

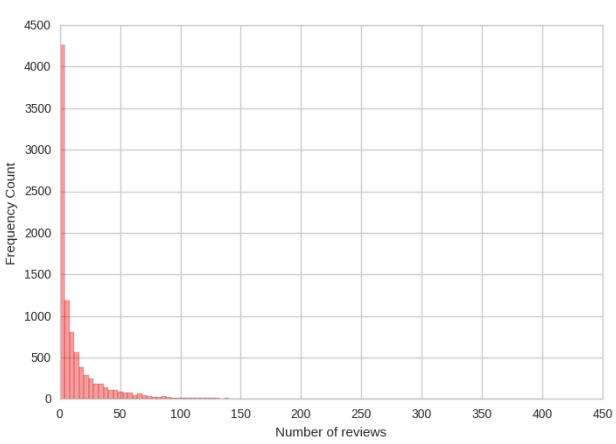
Additionally, a probability density plot of price variable showed the dependent variable to be normally distribution with a positive skew (Figure 5). The skewness was computed and found to have a value of 0.87.

3.2 Explanatory Variables

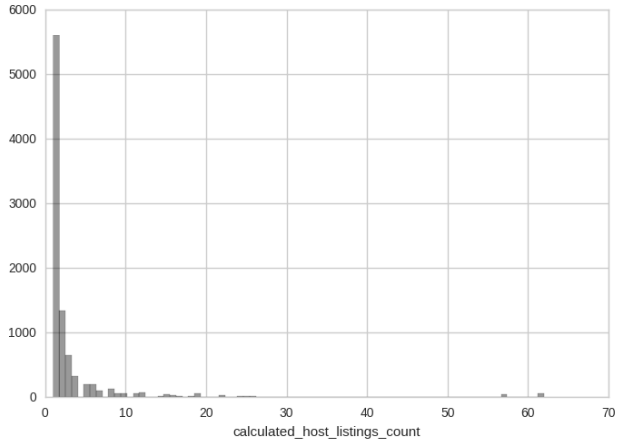
This section is categorized by the type of plots that were graphed for the different explanatory variables. Only select plots have been discussed in the report, however, all the plots have been provided in the Appendix.

3.2.1 Histograms

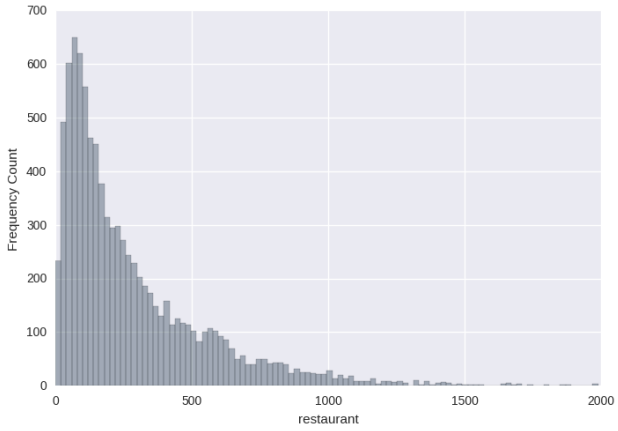
Histograms are helpful for analyzing the distribution of numeric variables. Although, we charted out the histograms for most of the numeric variables, we have selectively chosen a few for discussion in this report. The first histogram shown in Figure 6(a) is that of the variable `number_of_reviews`. Most hosts/listings do not have even 1 traveler review. The maximum number of reviews that a listing has received is 401. The next one is for the variable `calculated_host_listings_count`. 5604 hosts out of the total 9232 have just one property listed, 1335 have two. Very few owners have multiple properties listed. The maximum number of properties listed by a single host is 62. It is possible that this user corresponds to a real-estate agency with a large number of houses/apartments for rent. Histograms were also plotted for all the numeric neighborhood facility variables. In general these variables show a positively skewed but almost normally distributed trend. A sample of `restaurant` is illustrated in Figure 6.



(a) `number_of_reviews`



(b) `calculated_host_listings_count`



(c) `restaurant` distance in metres

Figure 6: Histograms

3.2.2 Bar Plots

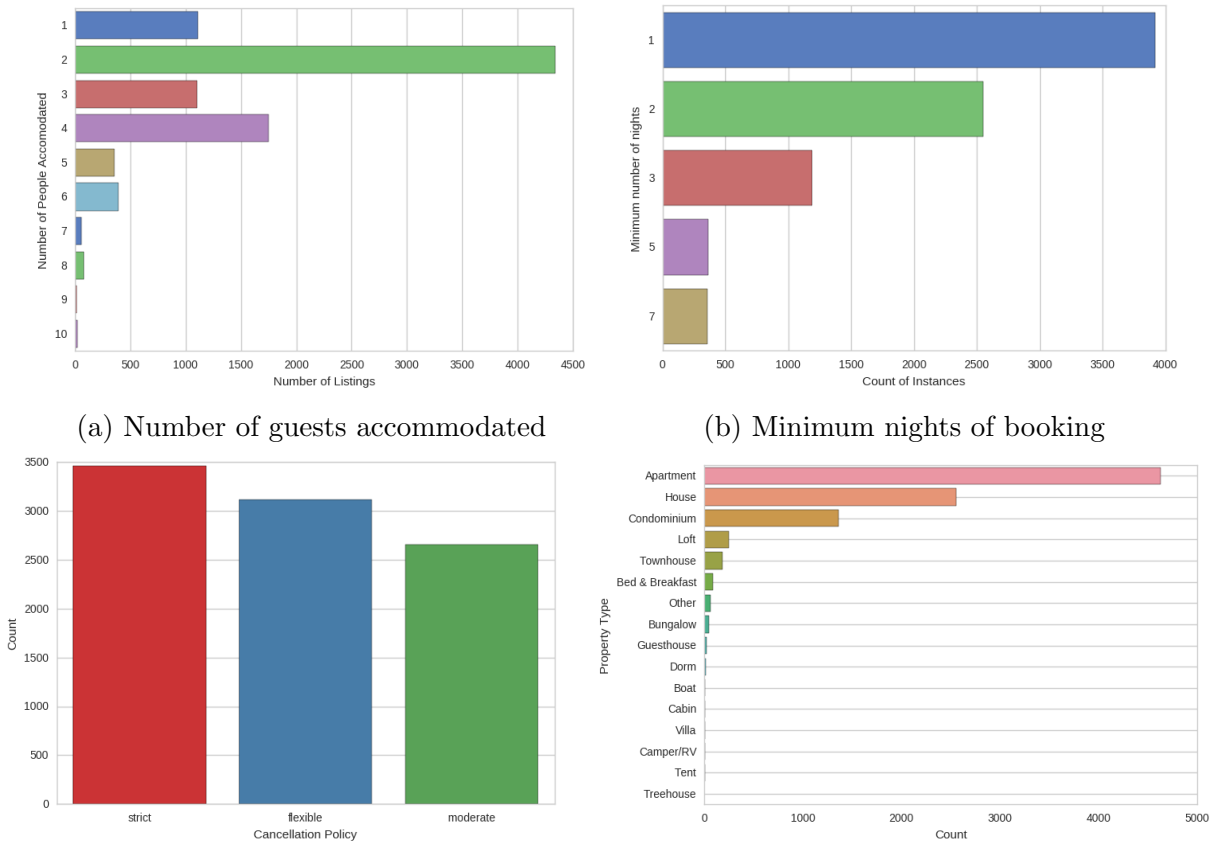
The analysis of the bar plots in Figure 7 can be described as below:

(a) Most of the listings accommodate 2 people. The count of listings that can accommodate more than 4 are very few.

(b) Most of the listings have a criterion that the client or the customer has to stay for a minimum of 1 night. However, there are listings that impose more number of minimum nights constraint. The graph only shows the most frequent 5 values.

(c) The most common cancellation policy is strict, followed by flexible and moderate.

(d) Apartments, house and condos comprise more than 80% of the listings. The uncommon ones include tents and treehouses.



(a) Number of guests accommodated

(b) Minimum nights of booking

(c) Cancellation Policy

(d) Property Type

Figure 7: Bar Plots (Part 1)

The analysis of the bar plots in Figure 7 can be described as below:

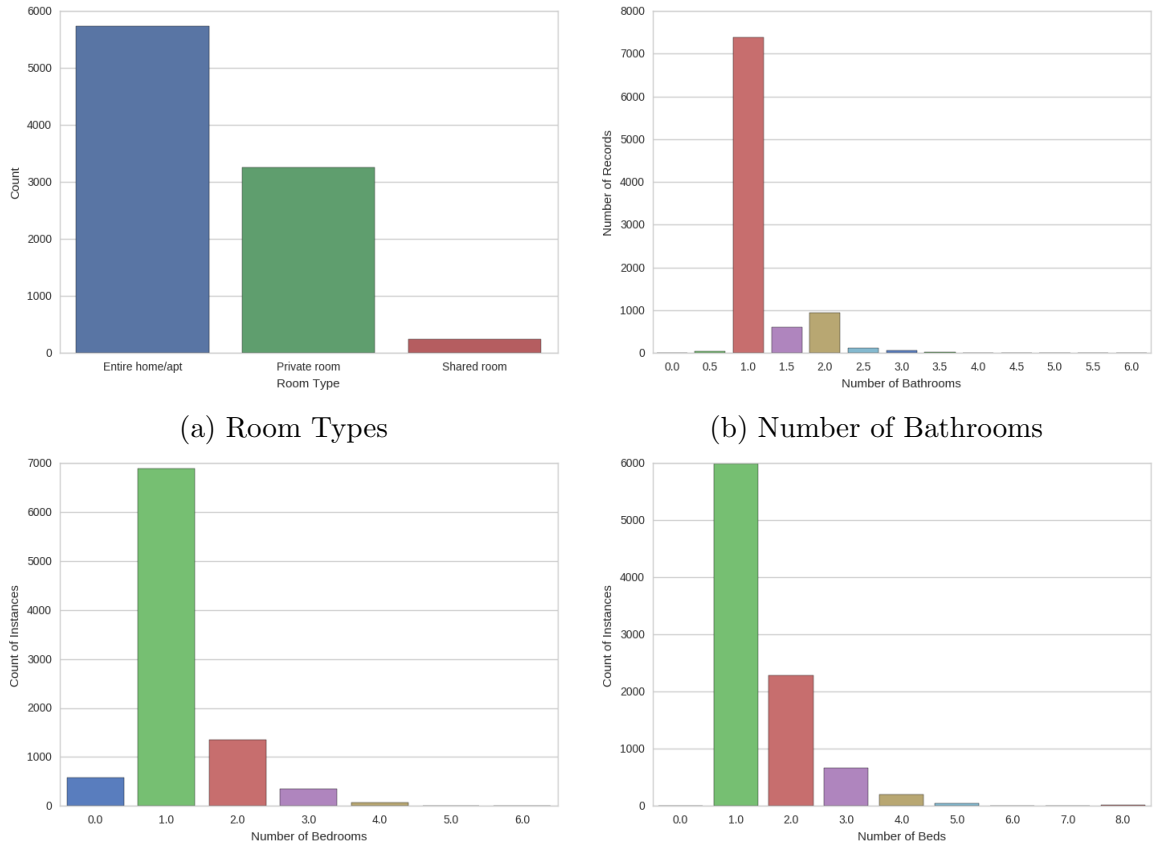
(a) There are 3 room types: entire room/apt, private room and shared room. Most of the listings had room type as entire room/apt. Over 5000 listings provide an entire apartment for rent and over 3000 listings provide one private room for the night to stay.

(b) Over 80% of the listings provide a single bathroom but there are some listings that provide 0.5 bathrooms (shared ones). Some listings have zero bathrooms! We realize that these are the ones where the listing refers to a property type of 'tent'.

(c) Singe bedroom is the most common. Zero bedrooms could either refer to a listing type like 'tent' or it could correspond to listings that where the living room/couch is rented.

(d) Here again, zero could refer to listings that only have couches and no bed. However, most listings offer a single bed.

The final bar plot is a stacked bar plot of the binary amenity variables showing the percentage of listings that provide each amenity. Referring to Figure 9, for instance, 41.8% of the listings provide parking facility whereas only 9.8% of the listings have an indoor fireplace.



(a) Room Types

(b) Number of Bathrooms

(c) Number of Bedrooms

(d) Number of Beds

Figure 8: Bar Plots (Part 2)

3.2.3 Scatter Plots

Scatter plots were made between all neighborhood distance variables and price. A sample for `restaurant` is shown in Figure 10. As evident from the graph, the price is negatively correlated with `restaurant`, i.e., if the distance to the facility is less, then the price is more. This trend was observed for most other distance variables (refer Appendix).

3.2.4 Correlation Analysis

Pearson Correlation coefficient was calculated between price and other explanatory variables. As expected the amenity variables have a slight positive correlation (between 0.1 and 0.3) with price, indicating that the presence of the amenity can result in higher listing price. On the other hand the distance variables have a negative correlation (between -0.05 and -0.25) with price. This is intuitive because if the distance to the facility is less, then the price is more. The actual values of the correlation coefficients have been provided in the Appendix.

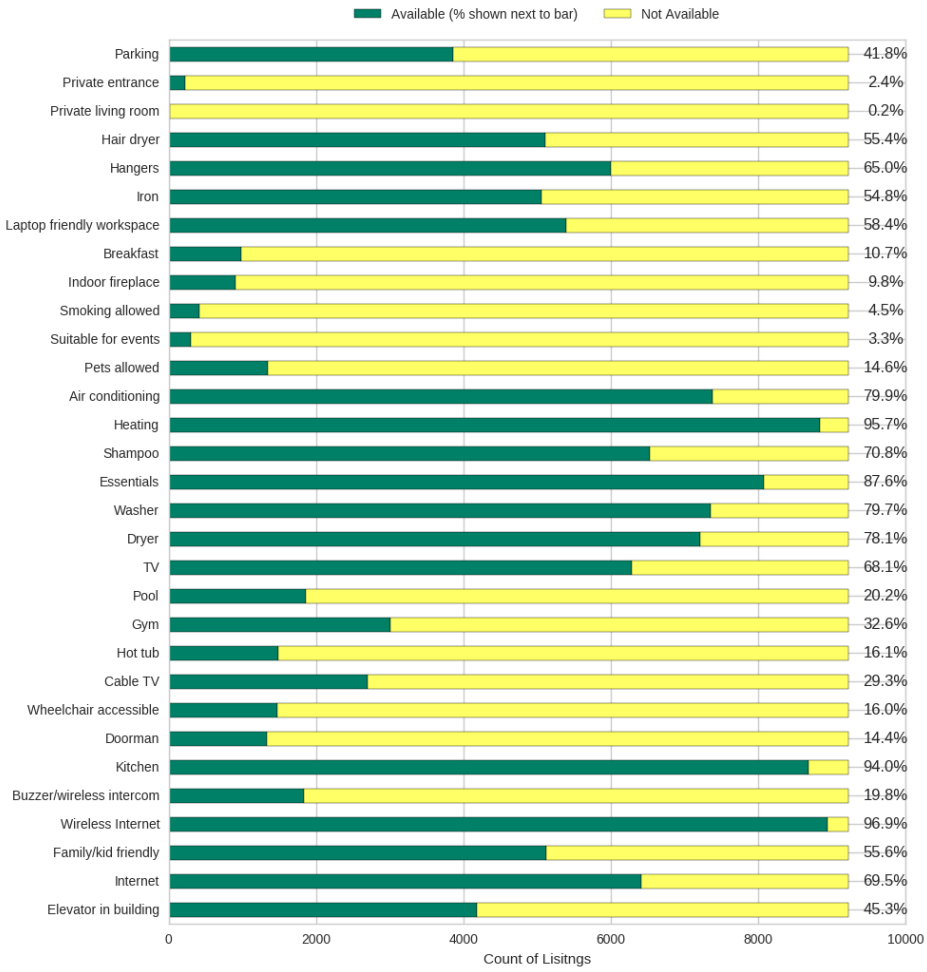


Figure 9: Stacked Bar Plot Amenities available with percentages (% of total listings that provide each amenity)

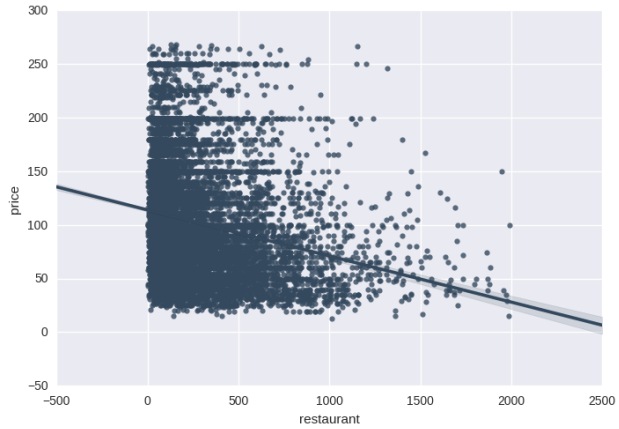
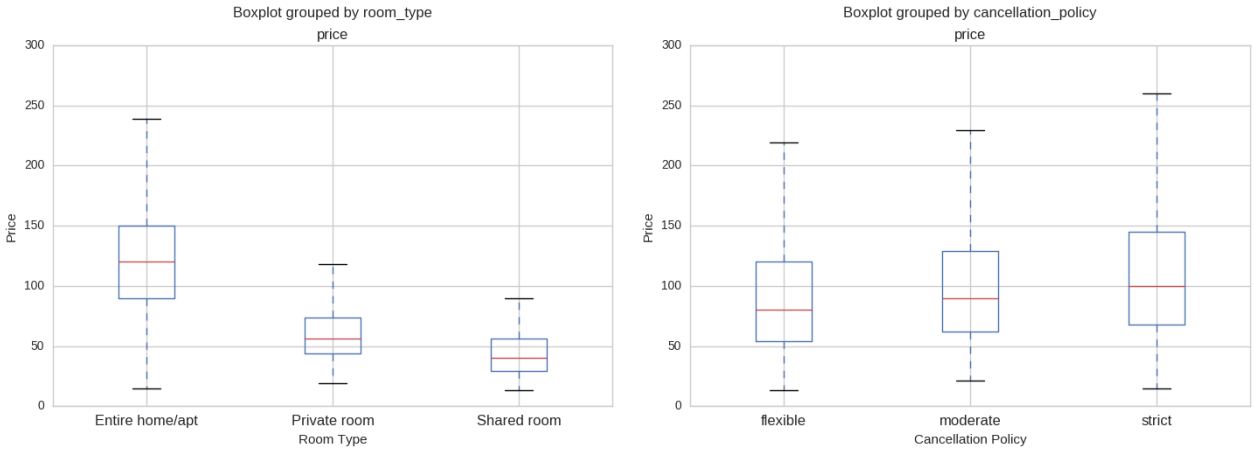


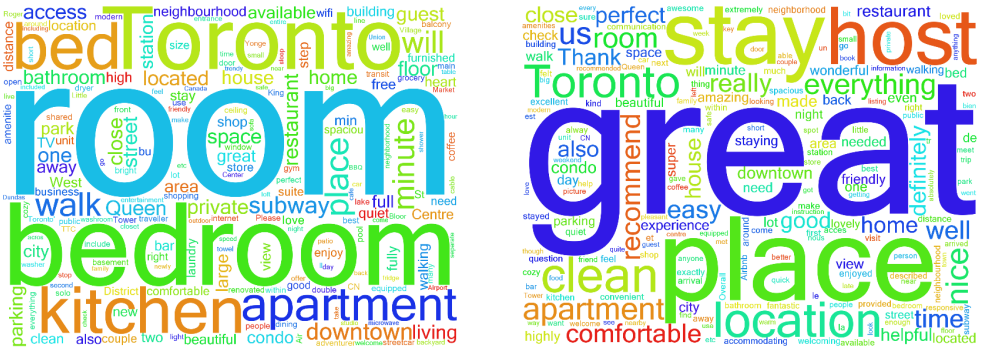
Figure 10: Scatter plot between `price` and `restaurant`



(a) Price by categories of room type

(b) Price by categories of cancellation policy

Figure 11: Box Plots



(a) Listing Description

(b) Traveler Reviews

Figure 12: Word Clouds

3.2.5 Box Plots

The two box plots in Figure 11 can be interpreted as follows:

(a) The 3 room types have different price ranges. The price of shared room is comparatively lower than a private room or the entire house.

(b) Strict cancellation policy house owners tend to price houses marginally higher.

3.2.6 Wordclouds

We constructed word clouds (Figure 12) for description and reviews which were the text data available, and discovered that 'room' and 'bedroom' were the most frequent words used in the listing descriptions. And for reviews, the most common words used were 'great' and 'place'.

4 Regression Analysis

This section describes the process of fitting a linear regression to predict the listing price using the different groups of explanatory variables discussed in the previous section.

4.1 Data Preprocessing

Some of the variables in the original data had to be transformed before being used for model fitting.

1. Variables having a huge range of values were converted from numeric to categorical. For example, `number_of_reviews` variable has most of the values around zero and the highest being 401 review. This skewed trend was observed for 2 other variables: `minimum_nights` and `num_page_saved`. The transformations are listed below:

    - `number_of_reviews` was grouped into 4 categories: 1) O - Zero Reviews, 2) L - Between 1 and 10 reviews, 3) M - Between 10 and 100 reviews, 4) H - More than 100.
    - `minimum_nights` was grouped into 4 categories based on the minimum number of nights of stay required: 1 night, 2 nights, 3 nights and the 4th group for any number of nights greater than 3.
    - `num_page_saved` was grouped into 3 categories: 1) L - Between 0 and 10, 2) M - Between 10 and 100 reviews, 3) H - More than 100.
    - `calculated_host_listings_count` was grouped into 2 categories: 1) S - single, corresponds to hosts who have a single listing 2) M - multiple, corresponds to hosts who have more than one listing on Airbnb.

2. Some variables which were already categorical in nature had some of their groups combined in the following manner:

    - `property_type`: Keeping only the most common categories (House, Apartment, Condominium), all the other property types were grouped into 'Other'.
    - `bed_type`: This was converted into binary with 'Real Bed' being the most common bed type. Every other bed types such as Couch, Airbed, Pull-out sofa, etc. were grouped together and named 'Other'.

3. All the categorical variables with more than one category were One-Hot encoded.

4. The neighborhood facility numerical variables had certain rows with values -1 (which meant that the particular facility was not found within the search radius listed in Table 1). In this case, such records had to be imputed with data. If a facility is not present, we need to impute the record with a high value. We decided to set this value to be equal to 1.5 times the search radius for each facility. For example, if no supermaket was present in the 5km radius, we assign a value of 7.5km to that listing for the `supermarket` column.

4.2 Model Fitting

For fitting the linear model, we make use of the Python library `statsmodels` instead of `sklearn`. This is because `statsmodels` outputs the model and co-efficient significance along with other evaluation metrics. After the data transformations and one-hot encoding, there were 73 explanatory variables for building the model. Model fitting was done using ordinary least squares. Some of the important results are discussed below:

- As shown in the model summary below, the R-squared is 0.58. The F-statistic is 200.8, which is larger than the F-stat at 5% alpha (as indicated by the value of Prob(F-statistic) in the model summary). This means that the overall model is significant.

Dep. Variable:	price	**R-squared:**	0.584
Model:	OLS	**Adj. R-squared:**	0.582
Method:	Least Squares	**F-statistic:**	200.8
Date:	Sun, 19 Mar 2017	**Prob (F-statistic):**	0.00
Time:	19:19:44	**Log-Likelihood:**	-45533.
No. Observations:	9202	**AIC:**	9.120e+04
Df Residuals:	9137	**BIC:**	9.166e+04
Df Model:	64		
Covariance Type:	nonrobust		

Figure 13: Linear Regression Model Summary

- The next step was to determine which coefficients in the model were significant at 5% alpha. The table below shows the features that had to be discarded at this stage because their Prob $> |t|$ was greater than 0.05.
  Interpretation: The variables listed in Table 2 turned out to be not significant. The interpretation could be two folds: 1) Some facilities like Internet were being provided by most homes and hence would not be a useful feature in predicting the price of the listing. 2) Some other facilities like Wheelchair accessible and Private entrance are provided by very few listings. As a result, the regression model would not be able to predict the price based on these features.

4.3 Recursive Feature Elimination

After removing the non-significant variables, we still have 52 variables in our model. The next step to feature selection would be to use recursive feature elimination. We start with 52 variables and keep discarding 1 variable at each iteration (that results in least reduction in mean squared error) in a greedy manner. A graph was plotted depicting the variation of MSE (when evaluated on the training set) with the number of features (Figure 14). The MSE when all the variables were used was 1166. When only 1 variable was used in the model,

Table 2: Non-significant variables which were removed from the model

Variable	**Prob($>|t|$)**
Internet	0.37
Wireless Internet	0.31
Doorman	0.29
Wheelchair accessible	0.64
Hot tub	0.13
Pool	0.54
Dryer	0.35
Washer	0.55
Shampoo	0.13
Suitable for events	0.82
Smoking allowed	0.14
Breakfast	0.09
Iron	0.63
Private living room	0.58
Private entrance	0.11
hospital	0.09
mall	0.17
supermarket	0.46
number_of_reviews_H	0.61
number_of_reviews_O	0.45
property_type_Condominium	0.71

the MSE was 2754. Using more features evidently results in a decrease in MSE, however, the gains in terms of the decrease in MSE is not significant beyond a certain number of features in the model. This is also validated by the elbow-shaped graph where beyond 21 features the slope of the curve is low, i.e., the graph become almost flat. The MSE returned by the model when using the 21 optimal features is 1424. These features along with their coefficients are shown in Figure 15 (Note that some of the above variables are categorical and have been one-hot encoded columns.):

It is interesting to note that all of the neighborhood facility features were eliminated in the feature selection process. Our initial hypothesis was that the distance of neighborhood facility would play a significant role in predicting the price of the listing. The idea was drawn from the fact that real estate prices are higher in a location with better facilities nearby. For example, having a mall or restaurant in the vicinity of an apartment would be a reason to price it high. However, this was proven to be wrong from our regression model since none of the neighborhood facility distance variables were helpful in decreasing MSE significantly. We further investigate the effect of these variables for a binary classification task which is described later in this report.

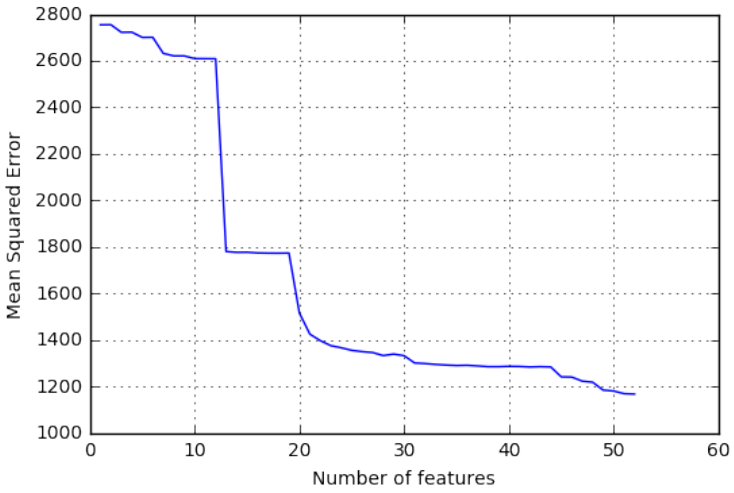


Figure 14: Variation of MSE with number of features

4.4 Interpretation of Coefficients

As shown in Figure 15, the interpretation in terms of the magnitude and sign of the coefficients for most variables are intuitive and as expected. For example, `bedrooms` has a positive coefficient of 19.1, which means that the price increases by $19.1 for every 1 unit increase in number of bedrooms, assuming that all other variables are kept at the same level. The interpretation for the other variables have been provided in the table. The final regression equation is as follows:

$$\begin{aligned} Price = {} & 9.3 + 19.1 * (bedrooms) + 8.2 * (elevator) + 2.9 * (min\_nights\_MN1) \\ & + 4.8 * (min\_nights\_MN2) + 4.1 * (min\_nights\_MN3) - 2.4 * (min\_nights\_MN\_) \\ & + 7.6 * (instant\_bookable\_f) + 1.7 * (instant\_bookable\_t) - 6.6 * (num\_page\_saved\_H) \\ & + 14.1 * (num\_page\_saved\_L) + 1.9 * (num\_page\_saved\_M) \\ & + 2.8 * (cancellation\_policy\_flexible) + 2.1 * (cancellation\_policy\_moderate) \\ & + 4.5 * (cancellation\_policy\_strict) + 2.1 * (bed\_type\_other) + 7.2 * (bed\_type\_real) \\ & + 36.6 * (room\_type\_entire\_home) - 5.1 * (room\_type\_private) \\ & - 22.2(room\_type\_shared) + 3.3 * (calculated\_host\_listings\_count\_M) \\ & + 5.9 * (calculated\_host\_listings\_count\_S) \end{aligned} \tag{2}$$

4.5 Cross-validation

A 10-fold cross validation was done and the MSE values from each fold are reported in Table 3. The results are promising since the average cross-validation MSE (=1429) is approximately close to the MSE when the entire data was used for both train and test (=1424). From this we can conclude that there is no over-fitting.

Figure 15: Coeffients and their Interpretations for the varaibles in the model

Variable	Coefficient	Interpretation
Bedrooms	19.1	Unit increase in bedroom results in additional charges of $19.1 in rent.
Elevatorinbuilding	8.2	Presence of elevator indicates that the price would increase by $8.2
minimum_nights_MN1	2.9	If the minimum number of nights is greater than 3, for example 7, the host is likely to give a discounted price. This explains the negative coefficient . And apart from that, the price is increased more if the minimum number of nights required is 2 rather than 1.
minimum_nights_MN2	4.8	
minimum_nights_MN3	4.1	
minimum_nights_MN_	-2.4	
instant_bookable_f	7.6	The price goes up for the properties which does not need a instant bookable feature by $7.6
instant_bookable_t	1.7	
num_page_saved_H	-6.6	People tend to book cheaper properties. Hence they save such property pages more often. The coefficient implies that lower priced listings are more popular.
num_page_saved_L	14.1	
num_page_saved_M	1.9	
cancellation_policy_flexible	2.8	The cancellation policy is stricter for most of the established and renowned properties owned by real estate owners. These properties have higher prices. The model explains this by having a greater coefficient of 4.5 for strict cancellation policy
cancellation_policy_moderate	2.1	
cancellation_policy_strict	4.5	
bed_type_Other	2.1	There are lot of bed types including couches, hence the price increases by $7.2 for real beds and $2.1 for other types.
bed_type_Real_Bed	7.2	
room_type_Entire_home/apt	36.6	The price increases by $36.6 if the property type is entire house or apartment. The prices are usually higher for entire apartment. Similarly, for shared room, the price goes down by $22.2.
room_type_Private_room	-5.1	
room_type_Shared_room	-22.2	
calculated_host_listings_count_M	3.3	Multiple listings are maintained by real estate owners and they have a competitive prices when compared to normal people listing their property which is usually 1. Hence the price goes up by $5.9 if the number of host listing is one.
calculated_host_listings_count_S	5.9	

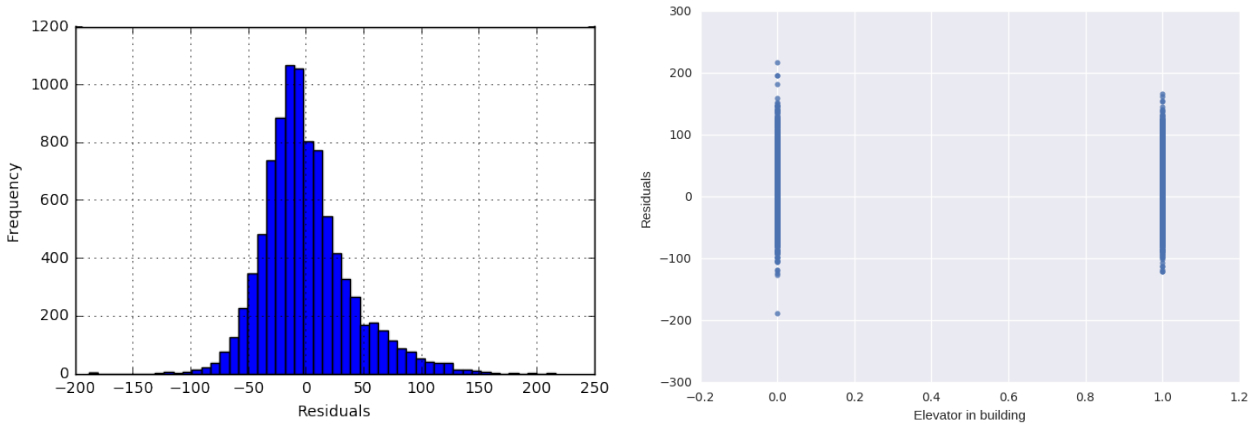
Table 3: Cross Validation MSE for each fold

Fold	1	2	3	4	5	6	7	8	9	10	**Avg**
MSE	1472	1395	1392	1440	1536	1281	1499	1390	1407	1460	1429

4.6 Residual Plots

In order to ensure that the linear regression assumptions of constant variance and normally distributed residuals hold, we analyze the residual plots.

- **Residual Histogram**: The residual histogram as seen in Figure 16 is approximately normal distributed with a mean value of -0.017. The graph and the the mean value which is approximately equal to zero can be used to conclude that there is no violation.
- **Residuals vs X**: Since most of the variables in the model are binary in nature, the x-label would be either 0 or 1 and hence the residuals when plotted against x would be of the form shown in Figure 16. The graph shown here is for a sample explanatory variable (Elevator presence in building). The graphs for the remaining variables are included in the appendix. Most of these graphs have no pattern which means that the model does not overestimate or underestimate for any specific x values.



(a) Residual Histogram

(b) Residuals vs Elevator Presence Variable

Figure 16: Residual Plots

5 Association Rule Mining

Apriori algorithm for association rules and frequent itemsets was implemented via the Orange package in Python. Association rule mining requires the variables to be discrete. All the amenity features that we have are binary and hence can be used for rule mining. The goal is to identify which set of amenities are simultaneously provided by hosts. For example, rentals with TV (`tv=1`) would mostly have a cable connection (`cable_tv=1`). We could also look for some not very obvious rules, for instance - does a listing which provides Gym facility also have an elevator in the building?

5.1 Data Preprocessing

Orange when used with Python requires the input to be in a 'market basket' format rather than a 'transaction' matrix. The difference between the two is illustrated via an example in Figure 17. Therefore, the preprocessing for implementing the apriori algorithm involves doing such a transformation of the binary amenities matrix.

5.2 Varying Support and Confidence

Frequent itemset generation and association rule mining was done at different levels of support and confidence thresholds. Support thresholds of 0.2 or lower results in `KernelException` because the program runs out of memory due to the large number of itemsets generated. Hence, we start with support threshold of 0.3 and vary it till 0.7. The number of frequent itemsets for different levels of support thresholds are listed in Table 4. Based on the the values, we decided to set an optimal support threshold of 0.5, where the number of frequent itemsets are not very large or very few.
Next task would be to pick the confidence level. Variation of the number of rules generated by the apriori algorithm at different support and confidence thresholds have been tablulated (refer to Table 5). Setting a confidence level of 0.8 at the decided support level of 0.5 would

Listing ID	Elevator	Gym	Heating	Air-conditioning
1	1	1	1	0
2	0	1	1	1
3	0	0	1	1
4	1	0	0	1

(a) Transaction Matrix Format

Listing ID	Amenities
1	Elevator, Gym, Heating
2	Gym, Heating, Air-conditioning
3	Heating, Air-conditioning
4	Elevator, Air-conditioning

(b) Market Basket Format

Figure 17: Data transformation required for rule mining

Table 4: Number of Frequent Itemsets for different support levels

Support	0.3	0.4	0.5	0.6	0.7
Number of Frequent Itemsets	8913	2029	578	167	59

Table 5: Number of rules for different support and confidence levels

	Confidence	
Support	0.8	0.9
0.7	193	124
0.6	594	210
0.5	1264	386
0.4	2495	431
0.3	3556	668

result in a very large number of rules. Hence, it would be ideal to set the confidence level to a value of 0.9. This is also the usual industry standard for market basket analysis tasks.

5.3 Insights from Frequent Itemsets and Rules

The insights from the rule mining described above can be used by hosts to understand which amenities come as a package, i.e., the set of amenities that are provided together. We can think of it as travelers expect certain amenities to be provided as a combo. Table 6 provides a list of some of the larger frequent itemsets (along with their supports) generated from step 1 of the apriori algorithm.

A sample set of the rules generated along with their supports and confidence levels are shown in Table 7 (The entire list is provided in the Appendix). The rules can be interpreted as follows: If a listing provides amenity A, then it also provides amenities B and C with the specified confidence. Hosts could use this information to decide on which sets of amenities to

Table 6: Frequent Itemsets with Amenities

Support	Itemset
0.52	TV, Air conditioning, Heating, Essentials, Kitchen, Wireless Internet
0.51	Hangers, Laptop friendly workspace, Heating, Kitchen, Wireless Internet
0.54	Heating, Shampoo, Essentials, Washer, Dryer, Kitchen, Wireless Internet

Table 7: Association Rules for Amenities

Support	Confidence	Rule
0.51	0.95	Iron, Kitchen, Wireless Internet $\rightarrow$ Essentials
0.54	0.91	Laptop friendly workspace $\rightarrow$ Hangers
0.76	0.98	Air conditioning, Kitchen $\rightarrow$ Heating
0.85	0.98	Essentials, Wireless Internet $\rightarrow$ Heating

Table 8: Association Rules that have Property/Room Type on LHS

Support	Confidence	Rule
0.59	0.96	room_type_Entire home/apt $\rightarrow$ Kitchen, Wireless Internet
0.59	0.95	room_type_Entire home/apt Kitchen $\rightarrow$ Heating
0.56	0.90	room_type_Entire home/apt $\rightarrow$ Essentials
0.33	0.93	room_type_Private room $\rightarrow$ Heating, Wireless Internet
0.46	0.91	property_type_Apartment $\rightarrow$ Heating, Kitchen, Wireless Internet

provide to travelers, especially when under budget constraints. For example, from the first rule in Table 7, say we know that a host already has Iron, Kitchen and Wireless Internet at his apartment. Assume that the host has the budget to invest in just one more facility. In this case he can choose Essentials (bedsheets, towels) because this rule has a confidence of 95%, i.e., most other listings which provide Iron, Kitchen and Wireless Internet also provide Essentials.

5.4 Property/Room Type and Amenities

In addition to the rule mining with just the amenity variables as described above, we also come up with rules that take into account the `property_type` and `room_type`. The variables `property_type` and `room_type` are one-hot encoded and provided as input to the the apriori algorithm along with the binary amenity features to generate frequent itemsets and rules. The support threshold was set to 0.3 (anything lower would result in too many frequent itemsets and Orange gives a `KernelException`). Setting higher values of support threshold would result in itemsets that did not have the variables `property_type` and `room_type`. For generating the association rules, the confidence was set to 0.9 (same as the previous analysis). In this case, we are only interested in the rules that have `property_type` and `room_type` on the LHS. The apriori algorithm return 44 such rules based on the minimum support and confidence. A sample list of the rules are shown in Table 8.

5.4.1 Interpretation

The sample rules shown in Table 8 can provide some useful insights to hosts. The 1st rule in the table says that listings which are entire home/apartments come with Kitchen and Wireless Internet with a 96% confidence. From the 3rd rule in the table, we know with 95% confidence that entire home/apartment listings provide Essentials (bedsheets, towels, soap). The other rules with property type apartment can be interpreted in a similar manner. This information could prove to be very useful for a new host. For example, if a new hosts wishes to rent out an entire home or apartment unit, then he would need to provide Kitchen, Wireless Internet, Essentials at the bare minimum to remain competitive on Airbnb. On the other hand, somebody who is only renting out a private room need not provide Essentials since most other listings also do not offer such amenities. However, even private room owners are expected to provide Heating and Wireless Internet (confidence of 93%) in a city like Toronto.

6 Clustering Analysis

In this section, we describe how the numeric variables in our dataset was used to identify patterns in an unsupervised manner. Specifically, we use the KMeans clustering algorithm and also tune the value of k, and find the optimum number of clusters.

6.1 Motivation

The input variables to the clustering algorithm are the 11 neighborhood facility distance variables and the price variable. The motivation for such an analysis was the belief that listings could certainly be grouped based on how far they are from facilities and this indirectly would also affect the price. Also, at the regression analysis stage, all the neighborhood facility distance variables had to be discarded during recursive feature elimination. Hence, we were motivated to use clustering to see if these variables when put together with the price gave rise to any distinct clusters. Our initial hypothesis is that there would be 2 distinct clusters - one representing the houses nearer to facilities and the other being the set of houses far away from facilities.

6.2 Data Preprocessing

The preprocessing required before clustering is data normalization. This is necessary because different numeric variables have different ranges. We use the Euclidean distance metric and therefore by normalizing we ensure that a unit difference in variable A has the same effect as that of a unit difference in another variable. For this purpose, we apply min-max normalization for each column (X) in the data, as depicted by Equation 3.

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3}$$

where, X_{min} is the minimum value in the column and X_{max} is the maximum value in the column.

Table 9: Inertia for different values of k

Number of Clusters	1	2	3	4	5	6	7	8	9	10
Inertia	3674	2078	1733	1519	1343	1230	1136	1060	1009	966

6.3 KMeans Clustering

As mentioned earlier, we use Euclidean distance metric because all the variables that we input to the clustering algorithm are of numeric data type. The normalized data was fed into KMeans clustering and the 'number of clusters' hyperparameter was varied from 1 to 10. The graph in Figure 18 shows the variation of inertia (sum of distances of records to their closest cluster center) and number of clusters. As the number of clusters increase, the inertia decreases. The 'elbow' shape of the graph can be used to decide on the optimum number of clusters. Alternatively, we could use the information about the change in slope of the graph between different values of k. As seen from the graph and Table 9, the steepest slope is from $k = 1$ to $k = 2$. So 2 is the chosen optimal number of clusters.

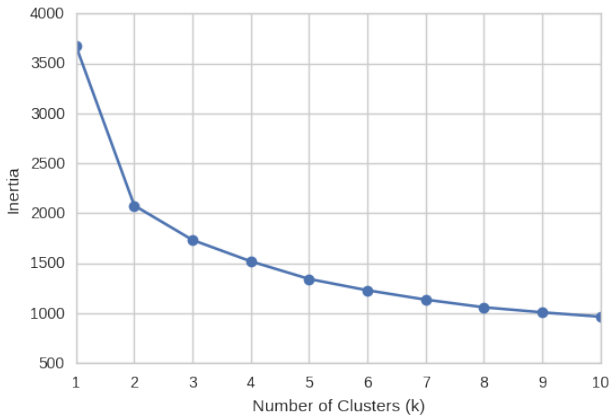


Figure 18: Variation of Inertia with number of clusters

6.4 Insights

The insights from the clustering are in line with our original hypothesis. There are 2 groups - one having the listings with neighborhood facilities nearby (and as such have a higher price) and the other having facilities farther away (and as such have a lower price). The cluster centers for each of the variables are shown in Table 10. For example, listings in cluster 2 (Expensive Listings) have ATMs within 760 metres (on an average) whereas for listings in cluster 1 (Cheap Listings) this value is 1593 metres. The same is true with most other distance variables except for parks and bus stops. The mean values for these two variables are approximately same for both the clusters. Parks tend to be farther away from most listings and bus stops are more common and can be found within 200 metres of every listing in a city like Toronto.

Table 10: Cluster centers for 'Cheap' (Cluster 1) and 'Expensive' (Cluster 2) clusters

	Cluster	
Variable	**1**	**2**
price	80	110
restaurant	533	202
atm	1593	760
cinema	2630	1339
hospital	3882	1607
nightclub	8253	1544
park	1785	1747
mall	7024	2204
gallery	11459	2411
museum	6111	1277
supermarket	756	323
bus_stop	194	186

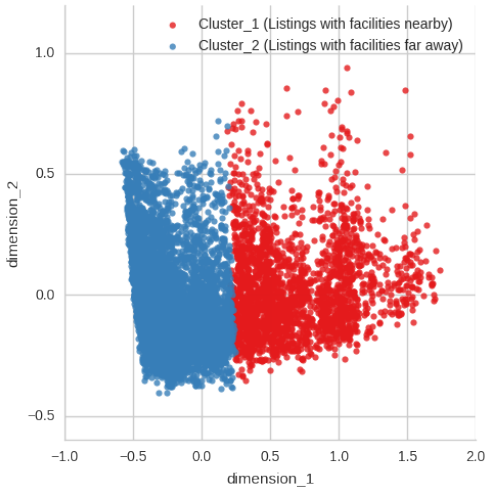


Figure 19: Scatter plot of the new dimensions based on PCA

6.5 Principal Component Analysis

In order to represent the graphs in a 2-dimensional plot, PCA dimensionality reduction was applied on the original data. Principal Component Analysis is a linear method of reducing higher dimensional data into a low dimensional space. In our case, the 12-dimensional data was transformed in 2-dimension for representational purposes. The exact meaning of what these 2 dimensions represent cannot be clearly explained. Next, we create a scatter plot with these 2 new dimensions on either axes and choose a different color for each cluster (Figure 19). It can be observed that the 2 clusters are distinctly separated on the graph, which again confirms our initial hypothesis.

7 Classification

From the clustering section, we realize that the data indeed had 2 categories or clusters of listings based on price - Cheap and Expensive. That motivated us to carry out a binary classification task on the dataset. In particular, we chose Naive Bayes classifier. Experiments were run with different variants of the Naive Bayes (NB) classifier - GaussianNB on the numeric data, MultinomialNB on the categorical data and various other combinations which are explained in detail in the subsections below.

7.1 Data Preprocessing

Since the price variable in the original dataset was of numeric type, we had to find a way to convert it into categorical for implementing a classification task. For this purpose, we used a 1-d KMeans clustering applied on the price variable to obtain 2 clusters of records. The one with the higher cluster mean ($162) corresponds to expensive listings and the lower cluster mean ($69) referred to the cheaper listings. We use these cluster labels as class labels in our binary classification problem. The new price column has 6048 records from the 'cheap' listings and 3148 records from the 'expensive' listings.

Next, we transform the features so that they are suitable to be used as inputs to a NB classifier. The transformations are same as the regression analysis case. Some numeric variables (`number_of_reviews`, `minimum_nights`, `num_page_saved`, `calculated_host_listings_count`) were transformed into categorical. No one-hot is done for categorical variables. Instead the categories are kept as numbers, for instance, 1, 2 and 3 instead of low, medium, high (because `sklearn` models cannot take strings as input, and MultinomialNB by default assumes all variables to be categorical).

7.2 Naive Bayes Variants

Six variants of the Naive Bayes classifier were implemented:

1. **NB_Numeric**: Gaussian NB using only the numeric type variables.
2. **NB_Categorical**: Multinomial NB using only the categorical variables.
3. **NB_Numeric_Binned**: Convert numeric variables to categories by binning (into High/Medium/Low) based on quantile values (equal bin sized) and then train a Multinomial NB classifier.
4. **NB_Categorical_Converted**: Step 1 - train Multinomial NB on categorical features and get predicted probabilities. Step 2 - use these probabilities as new features along with the existing numeric variables to train a Gaussian NB.
5. **NB_Numeric_Converted**: Step 1 - train Gaussian NB on numeric features and get predicted class labels. Step 2 - use these class labels as new features along with the existing categorical variables to train a Multinomial NB.

Table 11: Classification Accuracy of Naive Bayes Classifier

	Accuracy		
Model	**Train Set**	**CV Averaged**	**Test Set**
NB_Numeric	0.678	0.677	0.688
NB_Categorical	0.718	0.717	0.721
NB_Numeric_Binned	0.715	0.715	0.724
NB_Categorical_Converted	0.705	0.726	0.712
NB_Numeric_Converted	0.696	0.695	0.738
NB_Combined	0.731	0.731	0.741

6. **NB_Combined**: Step 1 - Separately train Multinomial NB on categorical variables and Gaussian NB on numeric variables. Step 2 - use predicted probabilities from both the models in Step 1 as inputs to train a new Gaussian NB.

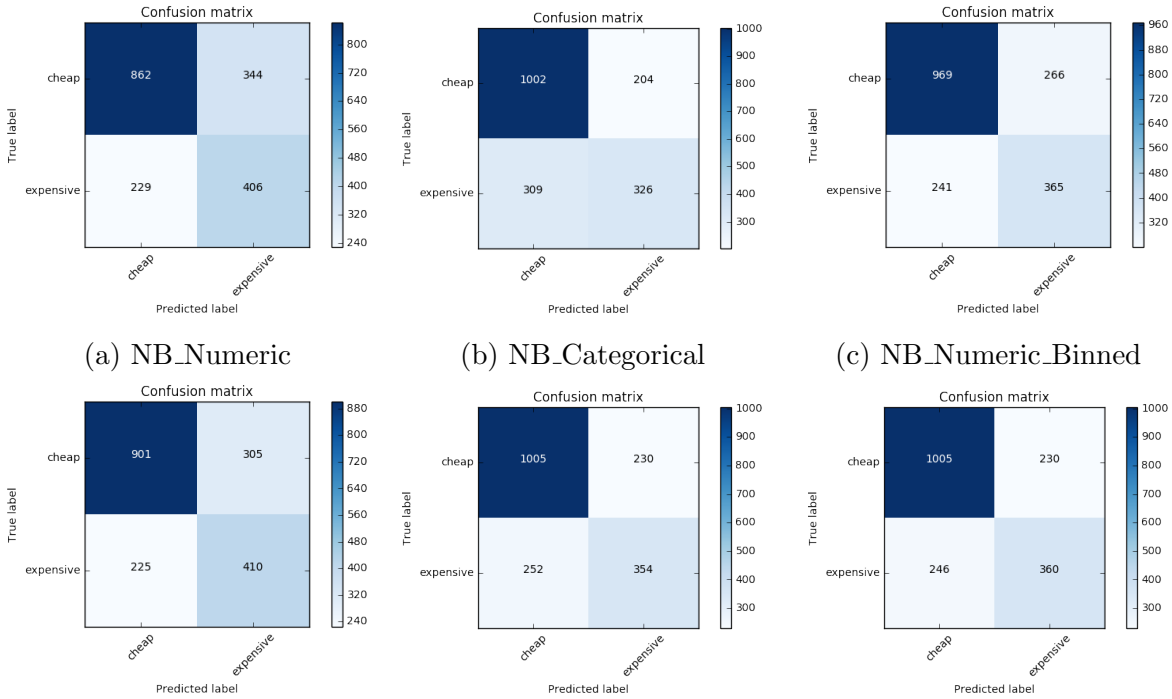
7.3 Discussion of Results

In order to evaluate which of the Naive Bayes model variant performs the best, we do the following:

- 10-fold cross validation and report the average classification accuracy
- Split the original data into test-train split with the same random seed. This way we ensure that all models are trained with the same train set and evaluated on the same test set. We report the confusion matrix and the classification accuracy on the test set.

From the accuracy values reported in Table 11, one can conclude that none of the models are overfitting on the training data. This is because the test and cross-validation accuracy are approximately same as the training accuracy for all the NB variants. In terms of performance, we start by comparing the Gaussian NB_Numeric with the Multinomial NB_Categorical. NB_Numeric has a lower accuracy score than the NB_Categorical, and it is also the model with the least accuracy. However, when the numeric variables are binned into High/Medium/Low categories and trained with a Multinomial NB, the accuracy on the test set increased to 0.724. The NB_Categorical_Converted and NB_Numeric_Converted had test set accuracy of 0.712 and 0.738, performance being similar to the NB_Categorical. However, the best performance was given by NB_Combined which is a model trained with features as the predicted probabilities from separate Gaussian and Multinomial NB classifiers. The classification obtained by combining multiple models is indeed promising because we are able to achieve the best accuracy of 0.741 as a result of this.
Next, we analyze the confusion matrices. The major findings are listed below:

- NB_Numeric and NB_Categoric_Converted were observed to be good with predicting expensive listings. However, they also predicted a lot of cheap listings as expensive.
- NB_Categorical, NB_Numeric_Converted and NB_Combined were good at predicting the cheap listings correctly.



(a) NB_Numeric (b) NB_Categorical (c) NB_Numeric_Binned

(d) NB_Categoric_Converted (e) NB_Numeric_Converted (f) NB_Combined

Figure 20: Confusion Matrices for the NB model variants on the same test set

- NB_Categorical has the highest mis-classifications in terms of listings that were actually expensive but being predicted as cheap.
- NB_Combined had the least total mis-classifications with only 476 out of 1841 as wrongly classified.
- NB_Categorical has the highest mis-classifications in terms of listings that were actually cheap being predicted as expensive.
- NB_Categorical_Converted made least errors while classifying expensive, with only 225 out of 635 expensive being classified as cheap.

8 Conclusion

The main objective of this study was to develop a price prediction model for new host owners on Airbnb to assist them in setting a price for their listing based on the amenities that they offer and the facilities available in the neighborhood. The model was build using linear regression and the features were selected based on recursive feature elimination. Apriori algorithm was also applied on the dataset to generate association rules using binary amenity variables. Clustering analysis was carried out to identify the presence of groups of similar records in the data. The result was the discovery of the presence of 2 distinct clusters in the dataset - one corresponding to listings with neighborhood facilities closeby and the other

with such facilities far away. Finally, we convert the price variable into binary (cheap and expensive) and study this problem as a binary classification task. Different variants of the Naive Bayes classifier were implemented and the results show that combination of separately trained models can improve prediction accuracy.
For the price prediction model, this could be developed into a full-fledged web application that can be used by hosts on Airbnb, who would just be required to input information about amenities and neighborhood facilities. This part of the project would be considered for future work.

References

[1] Harry W Richardson, Joan Vipond, and Robert A Furbey. Determinants of urban house prices. *Urban Studies*, 11(2):189–199, 1974.

[2] Jorge Chica-Olmo. Prediction of housing location price by a multivariate spatial method: cokriging. *Journal of Real Estate Research*, 2009.

[3] Jakob B Madsen. A behavioral model of house prices. *Journal of Economic Behavior & Organization*, 82(1):21–38, 2012.

[4] Suman Kumar Mitra. Applicability of artificial neural network in predicting house rent. 2008.

[5] Lynn Wu and Erik Brynjolfsson. The future of prediction: How google searches foreshadow housing prices and sales. In *Economic analysis of the digital economy*, pages 89–118. University of Chicago Press, 2014.

[6] Yingzhi Wu, Zhimin Zhou, and Jingyuan Li. New user booking prediction for airbnb historical data. 2015.

[7] What Price Should I List My Apartment For On Airbnb? http://hamelsmu.github.io/AirbnbScrape/. Accessed: 2017-01-28.

[8] Dan Wang and Juan L Nicolau. Price determinants of sharing economy based accommodation rental: A study of listings from 33 cities on airbnb. com. *International Journal of Hospitality Management*, 62:120–131, 2017.

[9] Yang Li, Quan Pan, Tao Yang, and Lantian Guo. Reasonable price recommendation on airbnb using multi-scale clustering. In *Control Conference (CCC), 2016 35th Chinese*, pages 7038–7041. IEEE, 2016.

[10] Emily Tang and Kunal Sangani. Neighborhood and price prediction for san francisco airbnb listings.

A Appendix I - Code on Github

The links to the Jupyter Notebooks with code/plots/results are available on Github:

1. Web Scraping Part 1 and Part 2
2. OpenStreetMap Data Collection
3. Exploratory Analysis
4. Regression Analysis
5. Clustering Analysis
6. Association Rule Mining
7. Binary Classification

B Appendix II - Association Rules

The association rules and frequent itemsets with the corresponding supports and thresholds are provided as text files on Github (because the list was very long).

1. Transformed in Orange market basket format
2. Frequent Itemsets
3. Association Rules
4. Rules with Property Type on LHS