Programming for Data Science

AIRBNB: What matters the most?

Lopes, Miguel[1], Magro, Helena[2], Ribeiro, Ana[3]

[1]M20180874, M20180874@novaims.unl.pt

[2]M20181404, helena.magro@gmail.com

[3]M20180956, ana_ribe87@gmail.com

Abstract: Airbnb - a global online marketplace and hospitality service -, founded in 2008, entered the Lisbon market in 2009. Since then it has been growing exponentially, not only in number of rental houses but also in profitability. Only in 2016, it registered an increase of 66% compared to 2015, reaching a total number of 718k visitors, with an over-night average of 4.1. It translated to a total of 42,8M€ of income earned by local households.

This article aims to decipher which are the main features of a rental place in Lisbon that influence the most the price per night in this region. Simultaneously, it also aims to reveal what the rental clients give more importance to.

Keywords: AirBnB, Webscrapping, Predictive Model, NPL, Lisbon

I. Introduction

Airbnb, founded in 2008, is a global online marketplace and hospitality service that lets people rent out their properties or spare rooms to guests who are looking for a place to stay.

As an increasing market the house short-term rentals are constantly growing, not only in number but in profit as well. Airbnb is in Lisbon since 2009 and since then the number of houses/ apartments has been growing. For that reason, the information in this website is becoming more valuable and useful for investment analysis.

Even though there are plenty of studies concerning the social and economic impact of Airbnb, none approaches the analysis of the features that are most valued by the users.

That said, this study aims to, in on hand, help a future investor to predict what is the fair price to request per night on his house; and in other and, to identify what Airbnb customers care the most about.

Programming for Data Science

1. Investor perspective

As an investor, we want to predict the fair market price per night to apply, given the conditions of the rental place and the investor characteristics.

2. Client perspective

The most important characteristic in Airbnb is the possibility that users have to leave reviews (giving a certain rating and/or leaving a comment). The goal with this analysis is to go through all the reviews/comments of the available rental places that exists in Lisbon and identify what Airbnb customers care the most about.

II. Data

1. Extraction

Since there is no previous knowledge about which variables were available for extraction and which data would matter the most, the first goal was to get as much information from Airbnb as we could.

For simplification matters, the search was constrained to the first week of August and the Lisbon region only. In terms of search parameters, for each district within Lisbon area (in total there are 16), we have performed a search of 1000 rooms. Regarding the guests, we decided for 2 adults, with no children.

Since Airbnb's website is heavily Javascript based, in order to get the information, we had to do a little more investigation than simply using a webscrapping tool. In the end, we had to perform 3 different requests to obtain all the data we needed.

The workflow of the program ended up being the following:

```
from ABNBRooms import ABNBRooms

counties = ["Alenquer", "Amadora", "Arruda dos Vinhos", "Azambuja",
"Cadaval", "Cascais", "Lisboa", "Loures", "Lourinhã", "Mafra",
"Odivelas","Oeiras", "Sintra", "Sobral de Monte Agraço", "Torres Vedras",
"Vila Franca de Xira"]

for county in counties:

    abnb = ABNBRooms(county, totalItems = 1000)

    #5 noites 1ª semana agosto
    abnb.checkin="2019-08-01"
    abnb.checkout="2019-08-06"
    #casal
    abnb.adults = 2

    roomList = abnb.getListings()

    df = roomList.toPandasDF()

    df.to_pickle("data\\"+county+".pkl")
```

Figure 1- Program Workflow

- ABNBRooms is instantiated and the user can choose to set, or not, multiple query parameters. Then the user has to call getListings() to start the search. This public method is where the first request is performed. After

getting the roomIDs, the __getRooms() private method is called from within, and it loops through all the roomsIDs and instantiates an ABNBRoom for each one, adding it to the instance of ABNBRooms. Finally, the user has the possibility of calling toPandasDF(), which transforms the list of ABNBRoom to a pandas dataframe.

- When ABNBRoom is instantiated, the __fillRoomInfo() is automatically called and performs, essentially, 3 tasks: getting data from the initial json (search), getting data from the 2nd request (room main info) and getting data from the third request (to get the reviews). There is also a public method, toDictionary(), that transforms the class to a dictionary, which is needed to then transform ABNBRooms to a pandas dataframe.

Besides these classes, there's also a general, static method that performs all the requests.

With all possible data gathered, we started by joining together the data from all the counties in a single dataframe, which ended up with 10375 rooms and 44 different parameters.

2. Data preprocessing

The following step concerned the cleaning of the data;

- **Remove duplicated rooms** – when gathering the data, we searched Airbnb by county. The problem is that many of the rooms overlap when searching on Airbnb, which created an excessive amount of duplicated rooms. This reduced the amount of rooms from 10375 to 1996;
- **Delete NA, None or NULLS** – after analyzing the dataframe, we found that more than 99% of the missing data was in the rating parameters, which we would not want to manipulate by any means, fearing that it could compromise the results. This reduced the amount of rooms from 1996 to 1216;
- **Deal with locations** – Regarding locations, from the data gathered from Airbnb, we had the location string of the room, which is a manual input of the Airbnb host, and, therefore, extremely unreliable. Our goal was to transform this string to one of the Lisbon counties and, by doing this, get a reliable categorical parameter. In order to do so, we had to standardize the input string. For that, we decided to use the Google Maps API. The remarkable thing about Google Maps is that one can search for "estoril, lisboa", "estoril, lisbon, cascais", "lisboa_ cascais| estoril" and it always returns the same information, in this case, "Estoril". This way, with google maps api we standardized the location strings. However, we still had a problem: in the case of the search string not being a county, how do we identify the county? For that, we decided to webscrape the website https://www.codigo-postal.pt to get the names of the counties. Then we identified the location by finding the minimum levenchstein distance between the string returned by Google and all the counties and all the freguesias. This worked great for almost all locations, however, we still had to manually change some locations. Namely, we had to remove rooms that were not located in the Lisbon district (but rather in Setúbal or Santarém) but were still picked up by Airbnb. We created a class Location to do all this logic. We also

added a new column to the dataframe called distToCenter, which was computed by finding the distance between the room coordinates (gotten from Airbnb) and the center of Lisbon, using the Haversine formula. This method is a static method of the class Location. After computing the distToCenter, we found that some rooms had a very unrealistic value for its county. This means that either the county classification was not well performed, or the coordinates were wrong from the start. Either way, we decided to define a maximum distance to the center of Lisbon, by county, and remove all the rooms that did not respect that limit. For instance, there were 2 rooms in the county of Lisbon that were more than 40km away from the center. This is obviously impossible. After this step, the amount of rooms was reduced from 1216 to 1127.

- **Deal with lists** – for the two lists we had, amenities and languages, we decided to simplify and simply count the number for each one, by room;
- **Deal with dates** – we decided to convert the two datetimes we had, last update of the room and date of creation of host account, to numerical values, by subtracting it from today's date.
- **Further data cleaning -** We also removed all rooms with less than 3 reviews. It is very important to note that the data we originally gathered is not validated, i.e., anyone can put any room for sale for any price. The only way to validate it is by making sure the room has a minimum number of reviews. This way, we are more certain that the room has market and, therefore, it is a significant for our analysis.

At last, we also removed any room that doesn't have a bed or has personCapacity less than one, as well as all rooms that have pictureCounts equal to zero. After this step, the amount of rooms was reduced from 1127 to 1123.

Finally, we had the dataset prepared to be analyzed:

accurracyRating	locationRating
allowsChildren	minNights
allowsEvents	monthlyPriceFactor
allowsInfants	personCapacity
allowsPets	pictureCount
allowsSmoking	ratePerNight
bathrooms	rating
bedrooms	ratingC
beds	responseRate
checkinRating	reviewsCount
cleanlinessRating	roomType
communicationRating	serviceFee
isBusinessTravel	totalPrice
isFulleyRefundable	url
isHotel	distToCenter
isInstantBookPossible	monthSinceCreation
isSuperHost	daysSinceUpdate
isVerified	totalLanguages
location	totalAmenities

Figure 2- Variables in the dataset

Afterwards this first general cleaning data process, the data was used in two different analysis, that also included they own cleaning and preprocessing data step.

III. Predicting the Rental Fair Price per Night

There are three price variables in the dataset: the Total Price, the rate (€) per night and the service fee. Since the Total Price is the same as having the rate per night multiplied by 5 nights (as only the first week of August was considered), the Total Price variable was dropped from the analysis.

On other hand, within this analysis it was also realized that the service fee was not integrated in the rate per night, therefore it was maintained it the dataset.

Variables such as : url, minNights and reviews were removed in order to eliminate redundant information: 1) the reviews are string variables and for the purpose of this analysis, non-relevant (as an future investor does not know beforehand what reviews he/she will get); 2) the url has no information; and, 3) minimum nights is a not a relevant variable since we have pre-set the analysis to a 5 day occupancy rental.

The chosen dependent variable for this study is *raterPerNight*, as it reflects better the rental value:

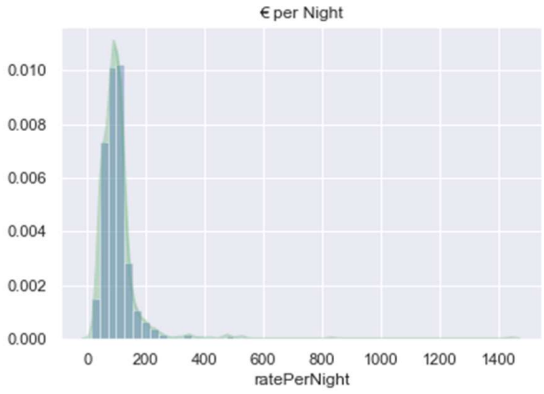


Figure 3- ratePerNight Histogram

1. **Numeric variable characterization**

The dataset was divided by numerical and categorical variables, and the histograms were plotted separately. The plot visualization indicates that the majority of the variables are not normal distributed and have the data concentrated in one or two classes. After that general observation, it was verified that there were not negative variables, and other

inconsistent data. As an example, the sBusinessTravel and isFullyRefundable variables have all data in one class and the isBusinessTarvel data is zero. These variables do not contribute to the predictive model and for that reason are removed further in the analyze. (Annex III)

2. Categorical variable characterization

The categorical variables are fewer than numerical: location, rating, responseTime and roomType. As expected, in the location histogram, Lisbon has the highest frequency followed by Cascais and Sintra. The most frequent room type is entire home/ apartment followed by private room. About the rating, it is distributed essentially by rating 5 and 4. (Annex IV)

3. Features selection

What are the most relevant features from the dataset? In the predictive model, the input variables should have the highest statistical significance. For that reason, before select the predictive model were tested some feature selection methods. These methods work in an automatic selection of the attributes that create an accurate predictive model. It can identify and remove unneeded, irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model or may in fact decrease the accuracy of the model.

In this project were tested three algorithms (selectKBest, tree decision classifier and random forest), one from each these three methods: Filter methods, wrapper methods and embedded methods. The information about the significance of each feature, from each method was summarized in a table. After that, the features chosen were the ones that were considered relevant (TRUE) in all of the three algorithms.

The features selected were: totalLanguages, totalAmenities, serviceFee, reviewsCount, PictureCount, monthsSinceCreation, distToCenter, daysSinceUpdate. (Annex V)

4. Multicolinearity verification

After having the features selected, it was plotted a heat map to identify if the more correlated variables, with correlation higher than 0.6, have been removed. The heat map visualization (Annex VI) shows the correlation between the selected features.

5. Outliers removal

After choosing the relevant variables to the predictive model, the outliers from these variables have to be removed. Even with less variables to analyze, it still being complex define an outlier without having all the variables related with each other. Could be removed a false outlier which result in loss of important data. For that reason, it was done a clustering method using K-Means algorithm.

Before applying the K-Means algorithm, the categorical variables were transformed in numeric variables and after that the optimal clusters number was defined by using Elbow method.

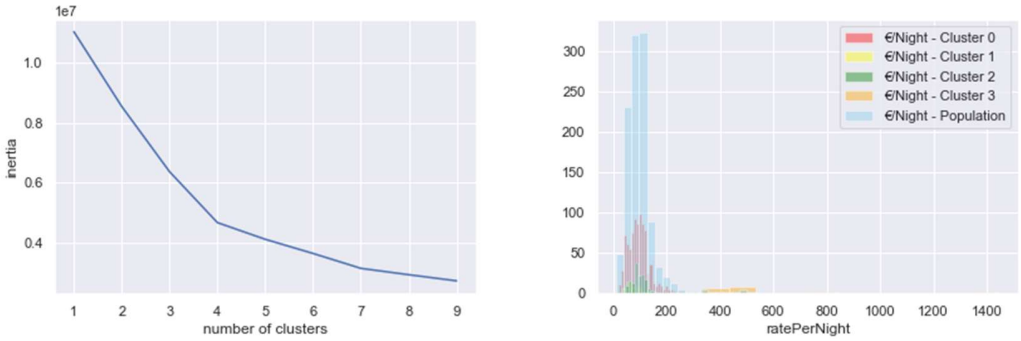


Figure 4- Clusters number definition and histograms

The number of clusters is 4, because after that the curve slope decreases, which means that with more clusters than four, the analyze becomes more complex without getting more relevant information.

For each cluster, only the observation within the 95.5% were considered as relevant observations and all the outliers were removed (Annexes VI and VIII).

6. Predictive Models

The main objective is to understand how the house features influence the rata per night, as predictive analytics is the process of discovering interesting and meaningful patterns in data, it is the way to predict the rate per night considering several input variables. It was used supervised learning methods, because there is target variable (rate per night) which is predicted with a given features.

It was tested three predictive algorithms: Decision tree regression, Random forest, K-NN Neighborhood.

The results from each model were as follows:

- Decision Tree training MSE = 0.0; MAE = 0.0; RMSE= 0.0; score is 1.0;

- Random Forest training MSE = 100.82; MAE = 1.56; RMSE = 10.04; score is 0.97;
- KNN Neighbors training MSE = 182.4; MAE= 7.1; RMSE = 13.51; score is 0.94.

Measure parameters:

- **Mean squared error (MSE)** is given by the average of the squares of the errors. In other words, the MSE is the difference between the predicted value and the real value, so the further lower it is, the better.
- **Mean absolute error (MAE):** this is the easiest to understand. It stands for the mean of the absolute values of each prediction error on all instances of the test dataset. This means that the predictions were perfect for the decision tree model but on average 1.23 away from the true prediction with the random forest model, and 7.13 with the KNN-Neighbors model.
- **Root mean squared error (RMSE)** is just the square root of MSE. The square root is introduced to make scale of the errors to be the same as the scale of targets.
- **Score** is measuring the accuracy of the model against the training data. (How well the model explains the data it was trained with)

Regarding the statistical information presented above (with the training dataset), it is concluded that the Decision Tree Regression is better than any of the other 2 models applied.

We re-ran the models with the test data, and the results were the following:

	MSE	MAE	RMSE	Score	MSE_Test	MAE_Test	RMSE_Test	Score_Test
DecisionTree	0.0	0.0	0.0	1.0	2097.7488151658767	7.606835071090047	45.80118792308642	0.18349749840058408
RandomForest	20.820676156583637	1.2330960854092523	4.562967910974571	0.9938702469098343	234.71824644549767	4.114218009478674	15.320517172912 2	0.9086410946781878
KNN	173.7404982206406	7.131435349940687	13.181065898501554	0.9488495787626889	138.4845497630332	8.221800947867 3	11.767945859963547	0.9460979405652107

Figure 5- Metrics for training and testing data

Even though the decision tree model showed perfect results on the training data for the testing one, it is actually performing worse than the random forest model or even the KNN model on the test data.

To validate the best model, all the models were plotted with the real rate per night and by observation, the conclusion is that the best model for this dataset is the random forest (different conclusion as previously).

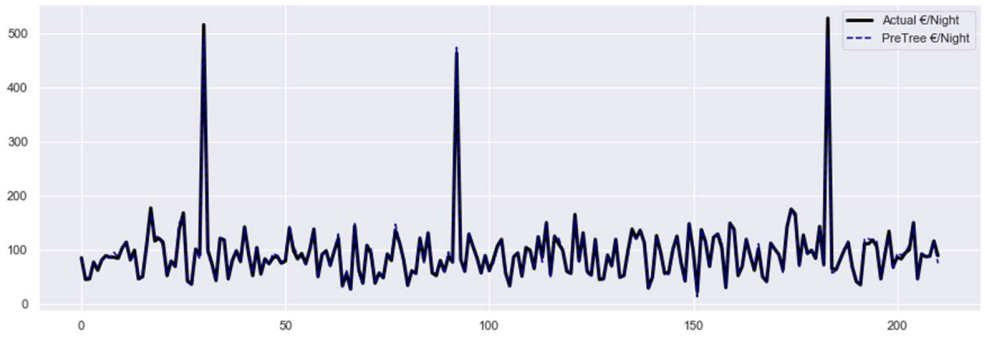


Figure 6 - Match of decision tree regression model with real rate per night data

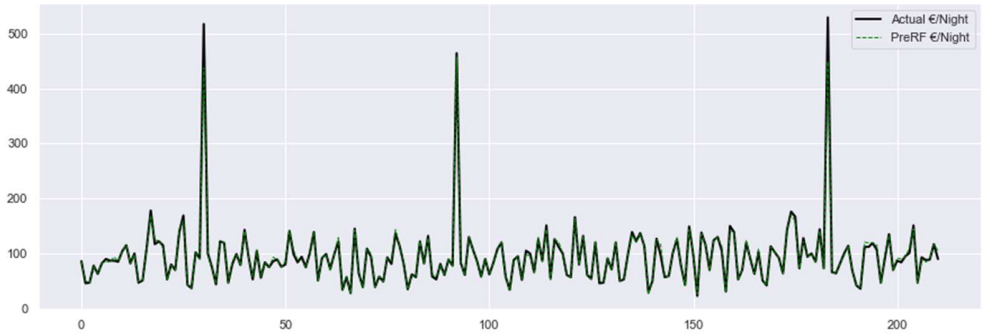


Figure 7 - Match of random forest model with real rate per night data

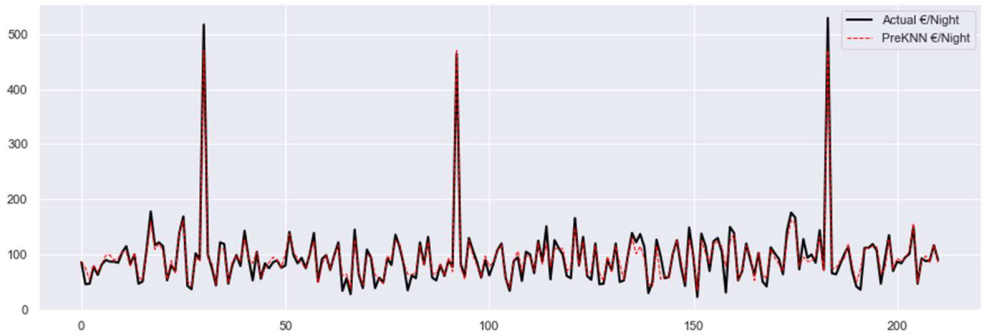


Figure 8 - Match of K-NN model with real rate per night data

IV. Word frequency analysis in reviews

Our goal with this analysis is to go through all the reviews/comments for all the rooms and identify what Airbnb customers care the most about, when rating the rooms. Our main strategy is a very simple form of Natural Language Processing (NLP), in which we will correlate word frequencies with room ratings.

1. Cleaning the data

Before starting to analyze text data, we first need to go through some text pre-processing techniques. We should note here that this cleaning process could easily go on forever, since there's always an exception to every cleaning step. Therefore, our approach is to follow a minimum viable approach, with clear simple cleaning processes. Our ultimate goal is to get a "bag of words" for each room, i.e., a list of relevant words used in each room.

As a reference, we'll always print here the result for the same roomID. We started with 2 comments:

```
['Amazing place very nice lady recommend to anywone', 'Lovely home
situated in the quiet village of Labrugeria.  Perfect for our needs and
wonderful hosts.  A great place to come home to after days out exploring
all the lovely sites, beaches and towns of the area.  45 minutes by car
to Lisbon and necessary for days out.']
```

Figure 9- Reviews sample

We proceeded with the following steps, by means of regular expressions:

- Merge all comments into one string, by room;
- Remove everything from the text data except letters and spaces;
- Lowercase every letter;
- Remove any word with less than 3 letters

The result is a string of words separated by a single space:

```
"amazing place very nice lady recommend anywone lovely home situated the
quiet village labrugeria perfect for our needs and wonderful hosts great
place come home after days out exploring all the lovely sites beaches and
towns the area minutes car lisbon and necessary for days out"
```

Figure 10 - Reviews after 1st clean

Next, we went through a slightly more advanced cleaning process. We used the library nltk which has a list of stopwords (very common words that bear no meaning) and a list of "all" English words. We then went through each string of words and converted it to a list of strings, splitting by space. Then we iterated over the list to remove all the stop words and words that don't exist in the English language, obtaining:

```
['amazing', 'place', 'nice', 'lady', 'recommend', 'lovely', 'home',
'situated', 'quiet', 'village', 'perfect', 'needs', 'wonderful', 'great',
'place', 'come', 'home', 'days', 'exploring', 'lovely', 'area', 'car',
'necessary', 'days']
```

Figure 11- Reviews with no Stopwords or typos

We can see that, for instance, “the” and “anywone” were removed because the first is a stop word and the latter is a typo.

Afterwards, we used nltk once again to categorize each word according to its grammatical class. We decided that the words that port value to our analysis are only nouns and adjectives. Keeping only those resulted in:

```
['amazing', 'place', 'nice', 'lady', 'home', 'quiet', 'village',
'perfect', 'wonderful', 'great', 'place', 'home', 'days', 'area', 'car',
'necessary', 'days']
```

Figure 12- Reviews with Categorization

Here we can see that “recommend” and “situated” were removed, due to being verbs. “lovely”, for instance, was identified as an adverb, although in this case it really is an adjective. Overall, after manually analyzing many of the thousand comments, we can say that this step was not very successful, misidentifying the words grammatically. Nonetheless, it was still better to perform this cleaning method than otherwise.

Then, we again used nltk to stemmatize words, i.e., convert every word to its most elementary stem. For instance, communication stems to commun. However, since these stems are not English words and are difficult to read, we substituted them, for simplicity, by the first word that was stemmatized to the same stem.

```
['amazing', 'place', 'nice', 'lady', 'home', 'quiet', 'village',
'perfect', 'wonderful', 'great', 'place', 'home', 'day', 'area', 'car',
'necessary', 'day']
```

Figure 13- Reviews with stemmatization

In this case, “days” was converted to “day”, and thus we decreased the number of words without losing information.

2. MatrixTerm and word frequency

By using sklearn, we converted the “bag of words” format to a matrix where we have the rooms in the rows and every word in the columns. The value is the frequency of the word for that room:

id	abb	ability	able	abnormal	abode	abrasive	abrupt	absence	absent	absolute	...	yummy	zag	zenith	zero	zest	zig	zigzag	zip	zone	zoo
21888626	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
23720498	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
22981563	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
5371649	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
18837008	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

Figure 14 – Matrix Term

With this matrix, we can easily find the most common words:

	great	place	nice	clean	stay	good	host	location	everything	apartment
count	1003	1000	951	943	931	887	887	887	877	850

Figure 15 - Most common words

However, this does not tell us much. A word that is used in every comment, effectively has no added value.

Therefore, we decided to filter out the words that appear too many times (>80%) or too few times (<20%) and, for the remaining, find the correlation between each word and the rating of the room. After removing some words that give no insights (eg: beautiful, great, amazing, etc), we generated the following wordcloud, with the top correlated words:

Figure 16 - Word Cloud of most relevant words for rating

Graphically, we can see the scattering of the frequency of the words:

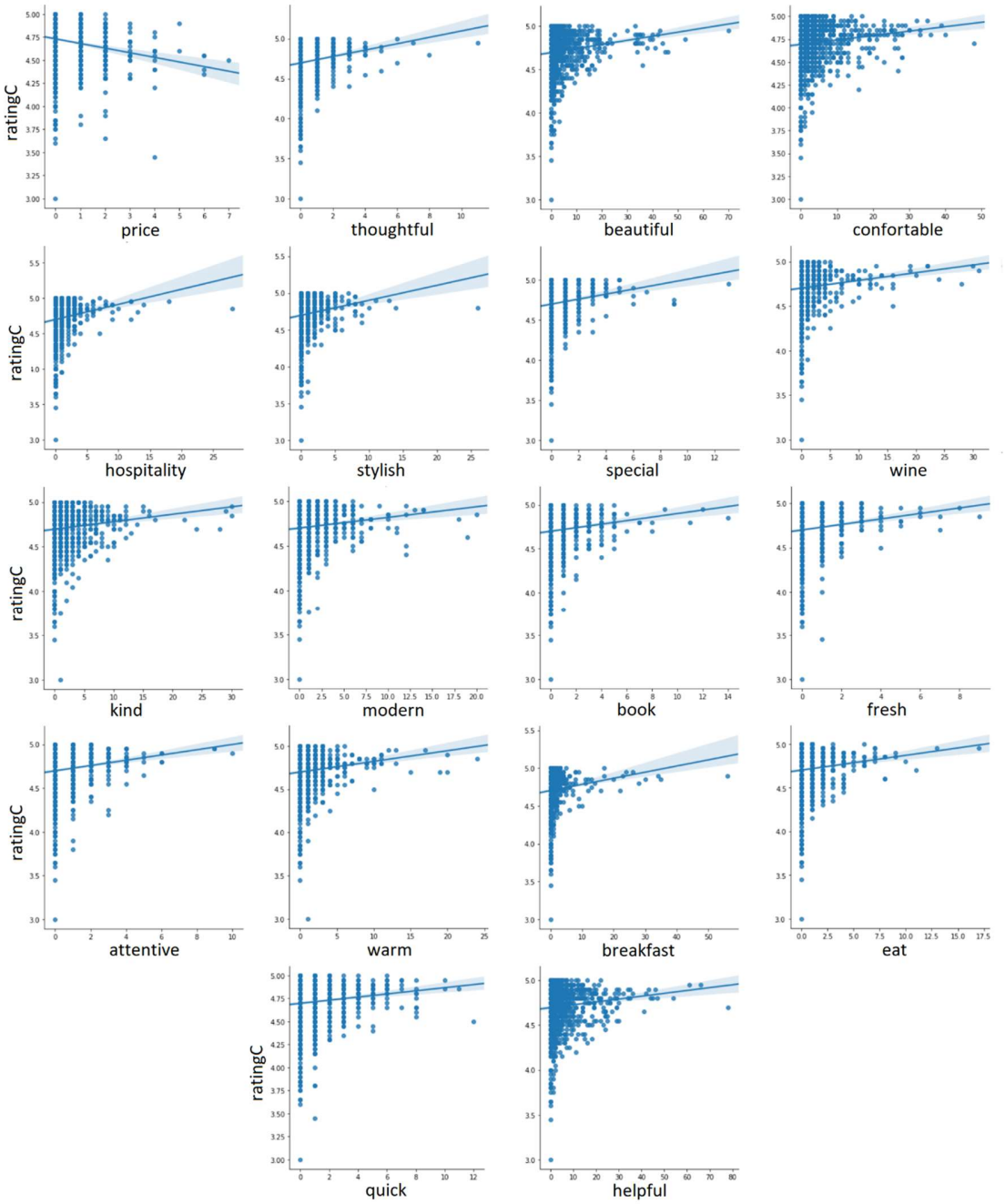


Figure 17 - Scatter rating vs most relevant words.

We can immediately see that the only word negatively correlated with the rating is price. This probably means that people only talk about price when they are unhappy.

It is alto interesting to note that, from the 18 words, the "hospitality" topic is the most common, including "thoughtful, "kind", "helpful" and "attentive". So, for an Airbnb host, hospitality and giving away food, namely wine and breakfast, would be an excellent combination.

Note, however, that all these words have very poor correlations with the rating, between 0.1 and 0.2. But that is somehow expected due to the nature of text variables, and most prominently word frequencies.

V. Conclusions

The house and host features that influence the most the price per night are: the number of amenities it offers, the number of languages that the host can speak, the service Fee charged per stay, the number of reviews the ad has, the number of pictures it posts, the distance to the Lisbon center, how long it has been on the platform, how long it was last updated , the time the host takes to respond to users, the type of room it offers and its location.

For example, an investor that creates an ad that offers 15 amenities, charges 20€ in Service Fee, has no reviews (since it is new), it has 10 pics, and it is located 1km from the center ([{'totalLanguages': 2, 'totalAmenities': 15, 'serviceFee': 20, 'reviewsCount': 0, 'pictureCount': 10, 'monthsSinceCreation':0, 'distToCenter':1, 'daysSinceUpdate':1, 'respTime':1 ,'room':2, 'loc':2 (arruda-dos-vinhos)}]) can charge 99,7€ per night

The best model to predict the €/Night, given the features analyzed is the Random Forest model. This model was chosen since its ability to predict the average prices is better than the other models tested, and it presents better metric scores overall.

Although simplistic, the word frequency analysis revealed powerful insights into the most relevant topics in the client perspective – price, hospitality and food.

VI. References

Airbnb. (2016). Overview of the Airbnb Community in Lisbon & Portugal.

André, M. F. (2018). Impacto das plataformas de economia partilhada,. Lisboa: Univrsidde Europeia.

Larose , Daniel T. & Larose, Chantal D. (2015). Data Mining and Predictive Analytics-Wiley

Abbott, Dean (2014). Applied Predictive Analytics_ Principles and Techniques for the Professional Data Analyst-Wiley

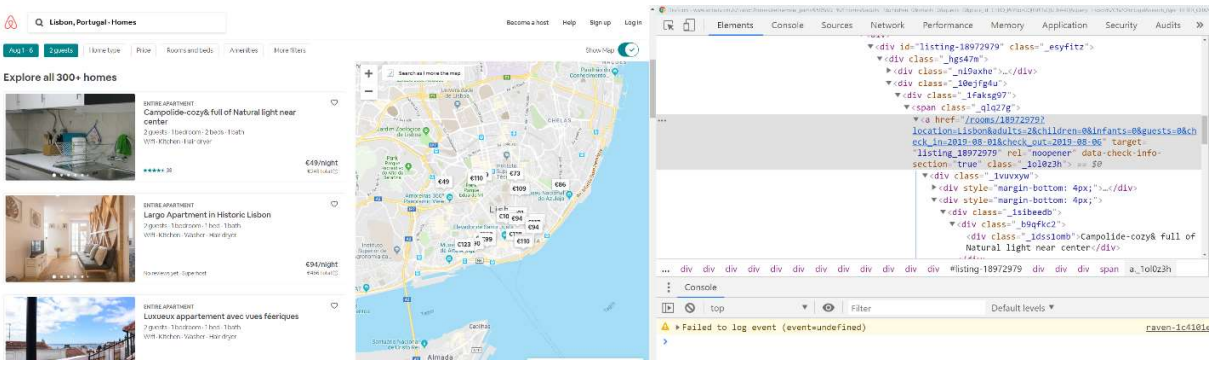
Bird, Steven & Klein, Ewan & Loper, Edward (2009). Natural language Processing with Python: Analyzing text with the Natural Language Toolkit – O'reilly

https://www.kaggle.com/sz8416/6-ways-for-feature-selection#Feature-Selection

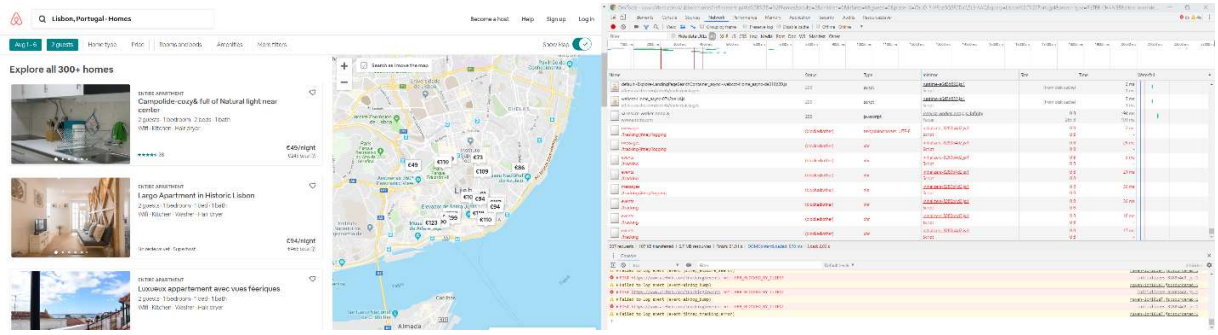
https://beportugal.com/airbnb-in-portugal/

VII. Annexes

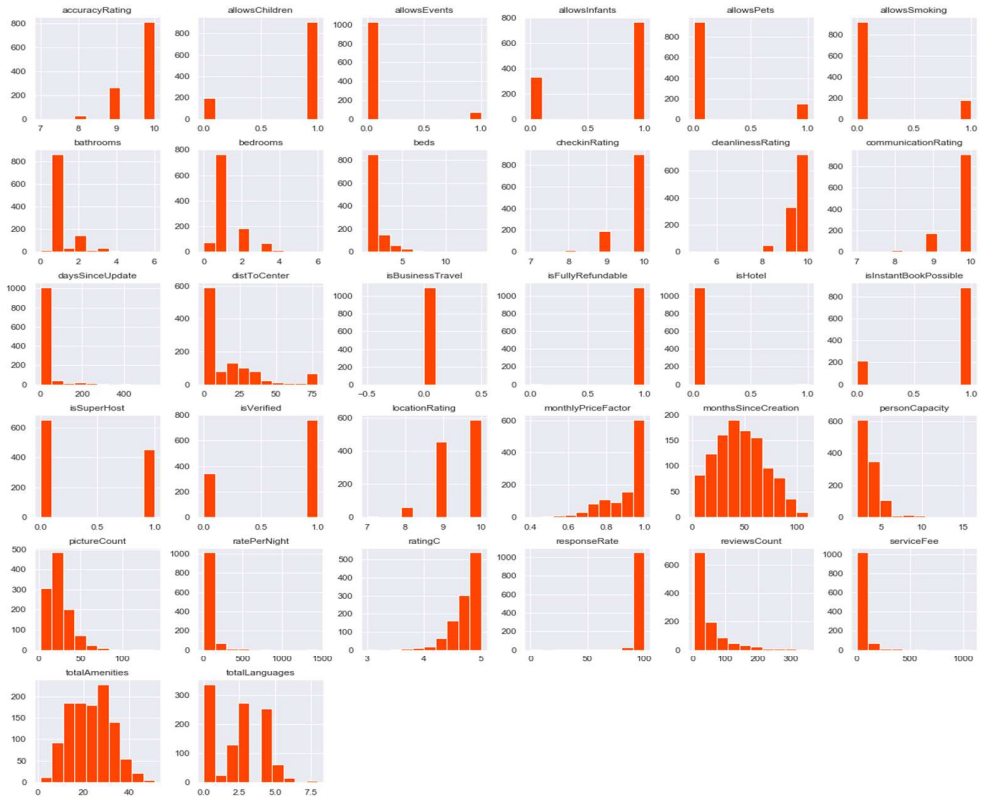
I. Inspect of an Airbnb page



II. Information given with webscrapping

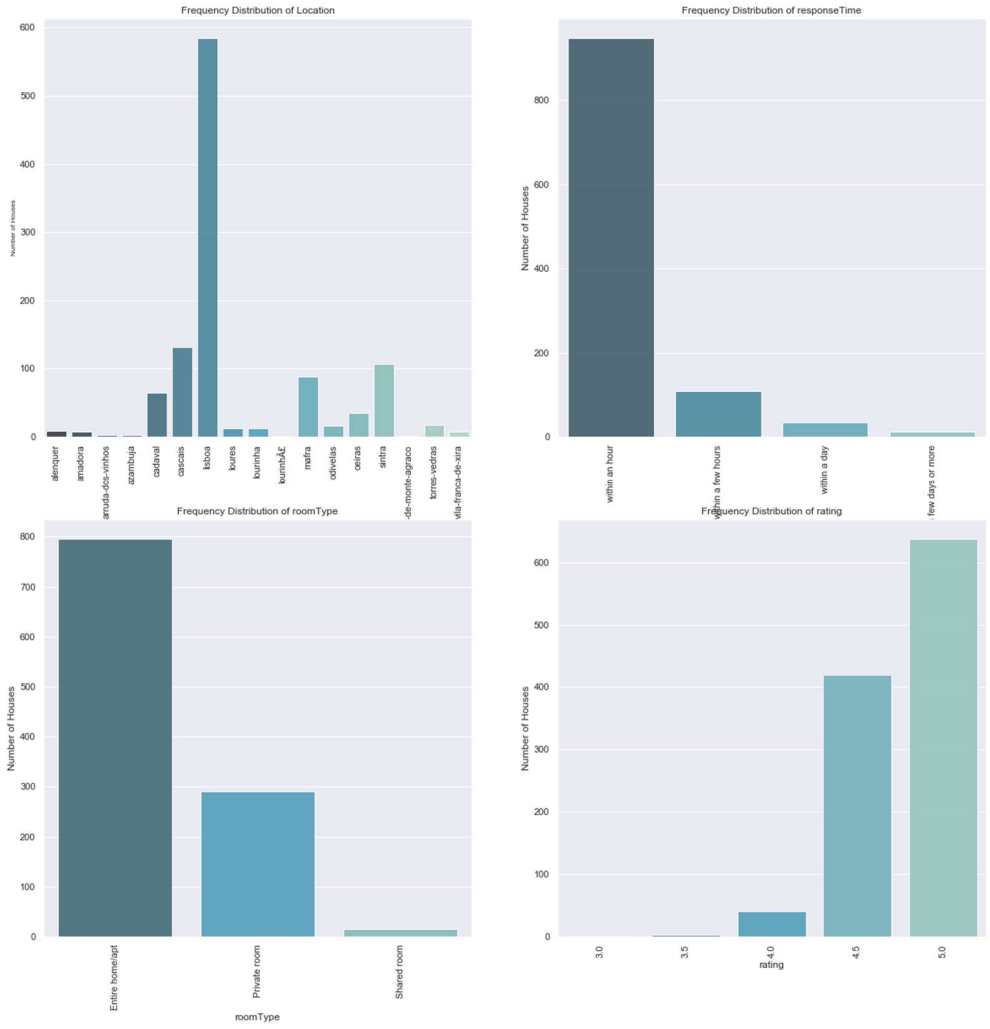


III. Numeric Variables Histogram

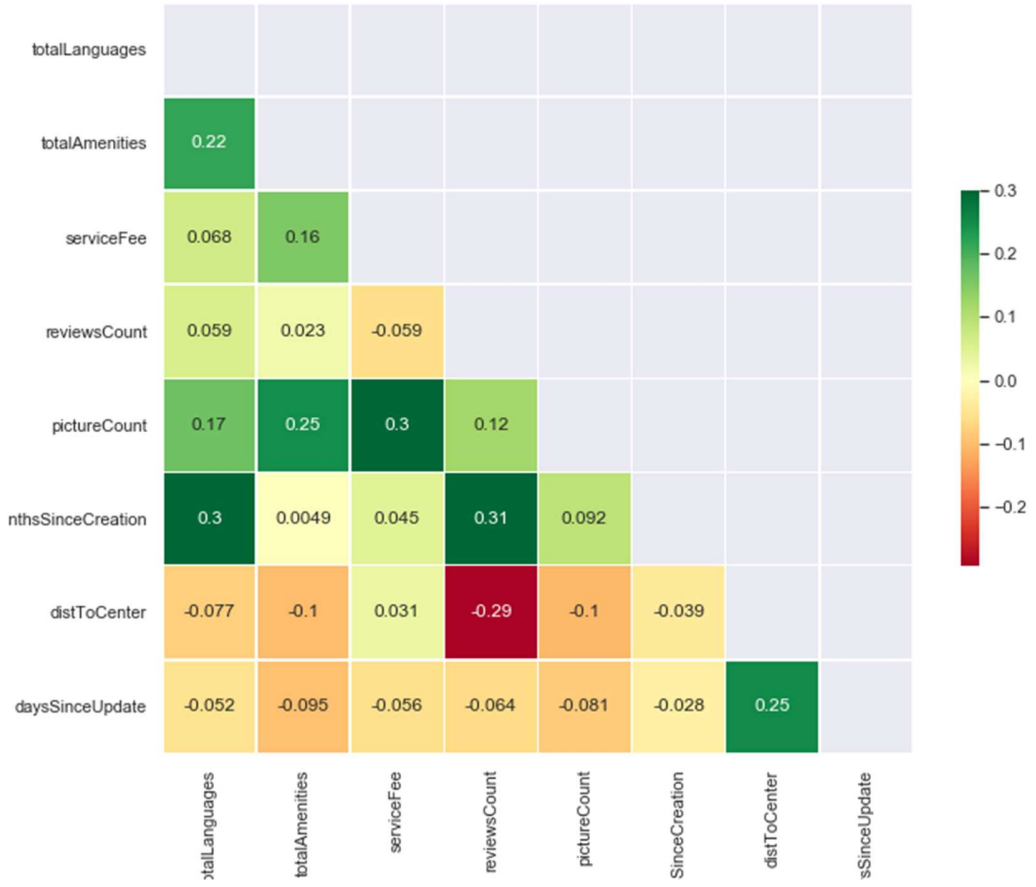


Programming for Data Science

IV. Categorical Variables Histogram

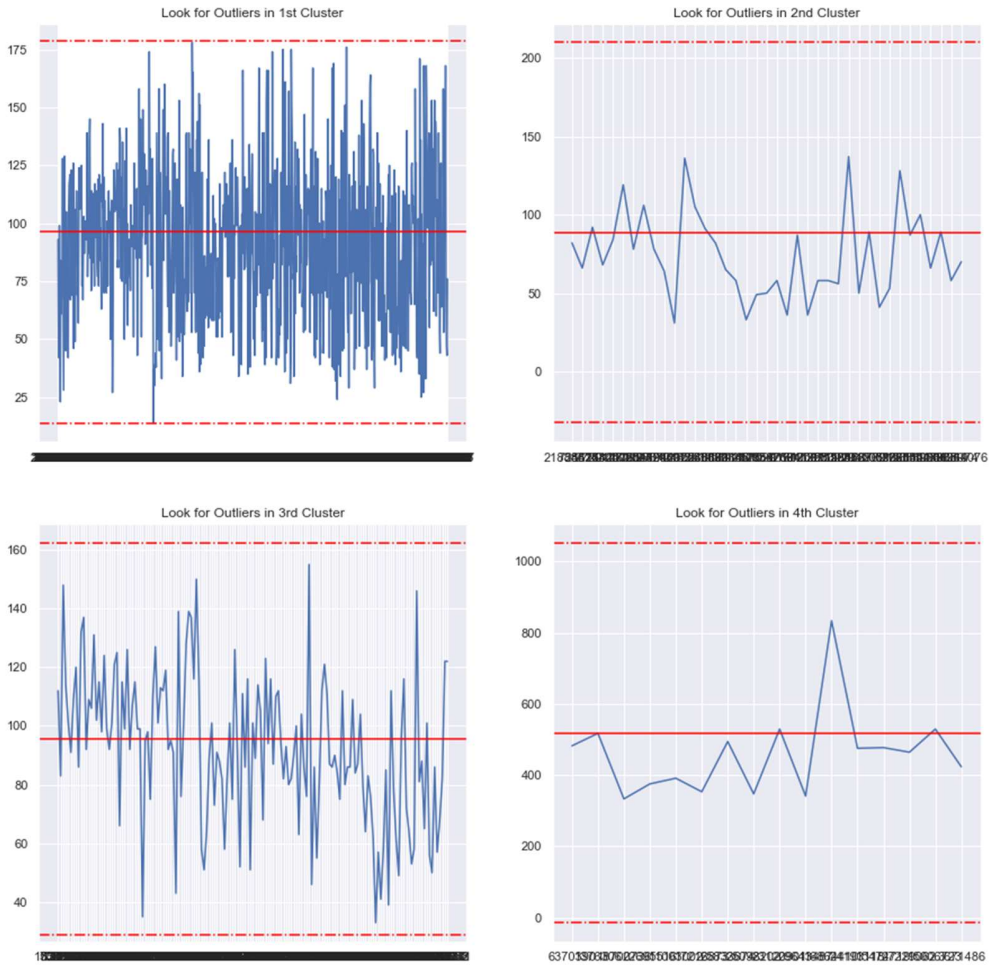


V. Heat Map Visualization



VI. Clusters Distribution before Outliers Removal

VII. Clusters Distribution After Outliers Removal



VIII. Features selected for the Predictive Model

	Feature	Chi-2	Random Forest	Tree Classifier	Total
1	totalLanguages	True	True	True	3
2	totalAmenities	True	True	True	3
3	serviceFee	True	True	True	3
4	reviewsCount	True	True	True	3
5	pictureCount	True	True	True	3
6	monthsSinceCreation	True	True	True	3
7	distToCenter	True	True	True	3
8	daysSinceUpdate	True	True	True	3
9	ratingC	False	True	True	2
10	personCapacity	True	True	False	2
11	monthlyPriceFactor	False	True	True	2
12	beds	True	True	False	2
13	locationRating	False	True	False	1
14	responseRate	False	False	False	0
15	isVerified	False	False	False	0
16	isSuperHost	False	False	False	0
17	isInstantBookPossible	False	False	False	0
18	isHotel	False	False	False	0
19	isFullyRefundable	False	False	False	0
20	isBusinessTravel	False	False	False	0
21	communicationRating	False	False	False	0
22	cleanlinessRating	False	False	False	0
23	checkinRating	False	False	False	0
24	bedrooms	False	False	False	0
25	bathrooms	False	False	False	0
26	allowsSmoking	False	False	False	0
27	allowsPets	False	False	False	0
28	allowsInfants	False	False	False	0
29	allowsEvents	False	False	False	0
30	allowsChildren	False	False	False	0
31	accuracyRating	False	False	False	0