# An Aspect Based Sentiment Analysis Approach to Airbnb Price Estimation Models

Dário Baltazar
STUDENT NUMBER: 2052603

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY DATA SCIENCE SOCIETY
DEPARTMENT OF COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
TILBURG UNIVERSITY

Thesis committee:

Dr. Raquel G. Alhama
Dr. Giacomo Spigler

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science & Artificial Intelligence
Tilburg, The Netherlands
June 2021

**Preface**

I would like to thank my supervisor Dr. Raquel Garrido Alhama, for her guidance during the development of this thesis. She has provided insightful feedback that enabled my progress on the thesis. I would also like to acknowledge the emotional support I got from all my university colleagues, friends, family and even everyone at the company I work for, for their understanding in some challenging moments. I also thank InsideAirbnb.com for providing the data used during this research.

# An Aspect Based Sentiment Analysis Approach to Airbnb Price Estimation Models

Dário Baltazar

*Airbnb is an online marketplace that specializes in offering peer-to-peer solutions to accommodations and tourism activities. Assessing a listing's price is a challenging task: on the one hand, property owners need a competitive price to attract customers, and, on the other hand, online customers need to evaluate the price of a listing based on its inherent characteristics. This research intends to improve price prediction accuracy by incorporating Sentiment Analysis techniques into existing machine learning price prediction models. Accordingly, customer reviews were integrated into an existing Airbnb dataset as a feature by calculating their sentiment score. This project addresses the related literature gap by introducing Aspect-based Sentiment Analysis for the calculation of the sentiment score from customer reviews. After testing and evaluating several models, it was determined that the inclusion of sentiment score extracted does not improve the predictive accuracy of the models presented in this project. In the end, an XGBoost model trained without the sentiment score feature provided the best results for price prediction.*

## 1. Introduction

The technologies brought by the development of Web 2.0 have encouraged the content sharing, on-line and off-line collaboration, and user-generated content from websites. This led to the emerging trend of a so-called sharing economy. Staff (2021) defines sharing economy as an economic model that uses peer-to-peer (P2P) activities such as the acquisition, provision, or the sharing of access to product and services enhanced through a community-based online platform. According to Teubner (2014), Airbnb is a prime example of this type of economic practice, which back in 2013 was already generating comparable numbers of stays to big hotel chains.

Airbnb is a company founded in 2007 that specializes in offering an online solution to booking accommodations and experiences by allowing guests to directly connect with hosts that offer accommodations or other services such as guided tours. Since its foundation, Airbnb has expanded to over 220 countries, with an average of 150 million users worldwide and a valuation of about 101 billion dollars (Bustamante 2021).

Due to the widespread use of Airbnb, it is important that the rental price presented for the listings reflects a fair price evaluation so both guests and hosts can take the maximum benefit from the platform. Therefore, one of the challenges presented to Airbnb owners is coming up with a price that will positively impact the number of customers renting their place. At the same time, customers want a fair and transparent estimation of the listing's price. As outlined in AirbnbEng (2016), currently Airbnb's dynamic pricing tool simply offers price suggestions to property owners based on the price of similar listings in their neighbourhood. It also takes into consideration reviews, but only the total number of reviews and the rating. This research aims to establish a fair model for Airbnb price evaluation of listings in Amsterdam. It will build upon studies

conducted on price estimation models for Airbnb listings. Machine learning models for price prediction of rental units were previously developed by Yu and Wu (2016) and Ma et al. (2018). In both cases, the models took no text data into account to estimate the price of rental units.

More recently, Kalehbasti, Nikolenko, and Rezaei (2019) improved these models by employing a Feed-Forward Neural Network model (something that had not been implemented in this area of research) and including customer reviews through sentiment analysis in the models. This type of approach takes a review and assigns a single polarity value to it. In their research, the sentiment from a review was considered positive or negative. On the other hand, in this project, rather than calculating only the general sentiment of a review, an Aspect-Based Sentiment Analysis will be performed to capture different aspects discussed in a customer review and sentiments expressed about these (de Kok et al. 2018). This technique allows for a more detailed analysis of the text provided by the reviews.

The research's main aim is to extend existing models with a more fine-grained type of sentiment (aspect-based sentiment analysis) that assesses the influence of opinion on different attributes of the property, as well as to aid Airbnb listing owners in evaluating their listing's price given a set of available features about the listing and the reviews submitted on Airbnb related to that listing. The research questions address the problem in the following ways:

R.1) Does including aspect-based sentiment analysis into a price evaluation model make the model perform better than an opinion mining sentiment analysis State-Of-The-Art approach (such as in Kalehbasti, Nikolenko, and Rezaei (2019))?
- R.1.1) Do pre-processing methods improve performance?
- R.1.2) Which features have the most influence over the outcomes?

## 2. Related Work

For the past few years, Airbnb has been growing its importance in the travel and tourism industry, and with this growth, it recently became a focus of academic researches. Although multiple studies have been published in recent years regarding Airbnb data, it is still a new topic worthy of investigation. By conducting this research, I aim to find a different methodology to evaluate Airbnb unit prices with the goal of improve on previous literature. So far, the approaches from previous literature differ substantially from each other. Beside using different methodologies, which will be discussed later in this section, these studies use distinct Airbnb data. Some use Airbnb data of a specific city; others use Airbnb data from multiple places around the world. All of these arguments makes that every research uses different features to predict the price. This diversity of procedures helps with the investigation in this paper by recognising models and methods that correlate with a better outcome and by identifying features that have high importance for the task at hand.

McNeil (2020) mentioned in his paper that the task of pricing properties is still one of the biggest challenges that Airbnb faces. Airbnb owners want to estimate a price that will positively impact the number of customers renting their property. At the same time, customers want to develop a fair evaluation of the offered price for a property. McNeil (2020) argues that this challenge comes from Airbnb granting owners the freedom to set their price, providing only a little information on similar listings in the neighbourhood of the property that could be used as reference. Airbnb has since adopted a dynamic-

pricing tool which, as outlined in AirbnbEng (2016), facilitate the owners with the task of setting up their price by showing them the predicted demand given a set of attributes, such as location. This research did not use dynamic attributes due to the lack of temporal data necessary for such task. It did however include something that still lacks from the Airbnb tool, which is the sentiment that reviews on listings have and how they could impact the price. Currently, the dynamic-pricing tool used by Airbnb takes into consideration reviews, but only the total number of reviews and their rating AirbnbEng (2016). Hinckley (2015) assessed the impact of online reviews by asking the following question to a group of participants selected for their study: "When making a major purchase such as an appliance, a smartphone, or even a car, how important are online reviews in your decision making?" (Hinckley 2015). They estimated that reviews regarding their purchasing decision impacted 67% of the consumers. Therefore, including the sentiment from customer reviews as a feature for the price estimation of Airbnb listings could improve a model's predictive performance.

Background analysis has been conducted by reading through a number of relevant pieces of literature that targeted price prediction models using similar datasets, to the one used for this thesis. Literature relevant to the task of aspect-based sentiment analysis was also analysed, given that this task is one of the most important steps in this research. Thus far, numerous researches regarding property pricing have been conducted. For instances, Yu and Wu (2016) built a real estate price prediction model for residential houses in Ames (Iowa) using Principal Component Analysis (PCA) for feature selection and various regression techniques such as Linear Regression, Support Vector Regression (SVR) and Random Forest Regression. Beside running regression models, they also classified the prices into different classes using Naive Bayes, Multi Nominal Logistic Regression, Support Vector Machine Classification (SVC) and Random Forest Classification. Notably none of the features used in Yu and Wu (2016) study - e.g. number of bathrooms/bedrooms/living rooms, housing prices, Etc. - relates to reviews or any textual data. In their conclusion, it is stated that, for the regression methods, the best model was the SVR model with an RMSE of 0.53 and, for the classification methods, the best classification model was the SVC model with an accuracy of 69%. In another paper, Ma et al. (2018) used three techniques to analyse the price of warehouse rentals in Beijing. Among them are Linear Regression, Regression Tree, Random Forest Regression and Gradient Boosting Regression. In their case, the best performing model was the Regression Tree with an RMSE of CNY/$m^2$.day.

More recently, Kalehbasti, Nikolenko, and Rezaei (2019) tried to improve those methods by implementing a Deep Learning model - Feed-Forward Neural Network - and leveraging the customer reviews through sentiment analysis in order to estimate Airbnb prices for units in the city of New York. They proposed to analyse the reviews for each listing in their data-set using the sentiment analysis library TextBlob. TextBlob allows to get a range of score values for the sentiment in the reviews that went from -1 to 1. A -1 value meant that the review portrayed a very negative sentiment, and a value of 1 meant that the review portrayed a very positive sentiment. Subsequently, they included the averaged sentiment result for each listing as a feature in the Price Prediction model. Finally, they proceeded with applying five different models - Linear Regression, Ridge Regression, Gradient Boost, K-means with Ridge Regression, Support Vector Regression (SVR) and Neural Network. Among those models, the one that performed the best was SVR with an $R^2$ of 69% and an MSE of 0.147.

As it stands, this paper's proposed approach follows Kalehbasti, Nikolenko, and Rezaei (2019) work, but differs in the way the sentiment analysis is treated. Kalehbasti, Nikolenko, and Rezaei (2019) use an opinion mining approach to perform sentiment

analysis. This type of approach takes a review and assigns a single polarity value to it. Hence, it does not take different sentiments that might be expressed in the review into consideration. This research proposes implementing an aspect-based sentiment analysis to Airbnb reviews and using the results on the price estimation models. de Kok et al. (2018) states that aspect-based sentiment analysis is divided into two main tasks: aspect detection and aspect sentiment classification. As described in their work, aspect detection refers to discovering which aspects are associated with a particular sentence or review; aspect sentiment detection refers to assigning sentiment to the aspects discovered in said review/sentence. Luo (2018) describes the data analysis procedures for aspect-based sentiment analysis on reviews as a set of topics relevant for this research: data pre-processing, information extraction, information categorisation, aspect segmentation and sentiment detection. This schema will serve as the basis for the aspect-based sentiment analysis done on this research.

This thesis will take into consideration all the work described on the literature presented in the hope of extending the existing price estimation models with a more fine-grained type of sentiment analysis - aspect-based sentiment analysis. Additionally, data collected from Ins (2021), a non-commercial set of independent tools that allow users to explore Airbnb data, for the city of Amsterdam will be used. This data has not been used in any of the studies being mentioned in this thesis.

## 3. Experimental Setup

The work was divided into two main steps: aspect-based sentiment analysis and price estimation models creation. An overview of the full flow of work is displayed in Appendix A. Before going into detail about the prediction models used in this thesis, it is relevant to mention the work done related to data cleaning and data preprocessing. Data cleaning and preprocessing related to textual data from customer reviews will be described in the Aspect-Based sentiment analysis subsection since they are pertinent to aspect detection. When it comes to the price estimation models' creation, a data cleaning and preprocessing subsection was created to highlight all the important steps taken.

### 3.1 Data

The data was collected from Inside Airbnb, a non-commercial set of independent tools that allow users to explore Airbnb data from cities worldwide. The data-sets used in this project represent an overview of the Airbnb properties in Amsterdam's city on the 8th of February 2021. The data collected is divided into two different files: "Listings" and "Reviews detailed". "Listings" contains all the listings in Amsterdam on the 8th of February 2021. It contains 96 features and 18291 observations of different Amsterdam Airbnb listings. This data-set includes continuous, categorical and text features. Since not all columns are relevant for the proposed research, columns that were applicable to answer the proposed researched questions were manually selected. "Reviews detailed" contains a total of 47882460 observations. There are six columns in this data-set: listing id, review id, date of the review, reviewer id, reviewer name and the actual review/comment. Because of a large amount of data in both data-sets, data cleaning and preprocessing techniques were used to create a final data-set that served as the one to build the models.

### 3.2 Method / Models

To meet the goals set for this thesis and find the answers to the research questions, Natural Language Processing techniques, as well as machine learning methods, were used. The first step was to apply aspect-based sentiment analysis on the available reviews. In order to do so, several steps had to be taken, such as tokenization, stop word removal, lower casing, aspect detection, among others. The aim of this first step was to get a more realistic sentiment score per review. After establishing a sentiment score for each review, scores were averaged across all the reviews linked to a property. This final score was then merged with the listings data and set as a new feature to be used in the models.

To evaluate the price estimation models, Linear Regression was selected as the baseline model. After establishing the baseline, multiple machine learning models were employed, such as Ridge Regression, XGBoost and Support Vector Regression. To establish a possible answer to the research question of how sentiment might affect the results, two versions of the models were fitted to the data: one that used the sentiment score as a feature and one without it. A deep learning approach, Neural Networks, was then applied to the data.

In the following subsections, a more detailed description of the implementation process and description of each model is presented. [1]

**3.2.1 Aspect-Based Sentiment Analysis.** (Wang and Liu 2015) describes the traditional way of extracting sentiment from a text as one that assumes the text has only one aspect associated with it. A more complete method is to predict the aspects present in a given text followed by the sentiment associated with each one of them. As already mentioned in section 2, this method is called aspect-based sentiment analysis (ABSA), and it can be divided into two main tasks - aspect detection and aspect classification (de Kok et al. 2018). In this research, the first step into applying ABSA was detecting the aspects in the reviews, which required some text pre-processing tools. Aspect detection consisted of the following tasks:

**a)** **Removing non-English reviews** To remove non-English reviews it was necessary to use the *detect* function available through *langdetect*. I have built a function that analysed all the reviews in the dataset and assign the language by adding the information in a new column. After this, I dropped all the rows where the value in the new column was different than 'en'.

**b)** **Normalization** In this task the reviews were transformed into a more basic format by ignoring reviews where the length was shorter than one character; removing accented characters; converting short forms such as 'thx', into full forms like 'thanks', which was done by creating a dictionary of the most usual short forms in English and what are their full forms; and lower casing.

**c)** **Extraction of opinion units** To extract the opinion on each review a method similar to the one used by Htay and Lynn (2013) was used. With the help of model "en_core_web_lg" available in the Spacy (2021) library, a part-of-speech tagger was used to identify and split phrases in the review that contained verbs as opinion phrases.

1 Code can be found here:
https://github.com/DarioBaltazar/ABSA_PriceEstimation_Thesis

**d) Extraction of tags per opinion units** To understand which tags were associated with each of the opinion units there was a need to tokenize, remove punctuation, remove stop-words and lemmatize the opinion units. All of this was done with the help of Spacy (2021) library.

After the extraction of the opinion units and their tags, the following step was aspect classification. To help with this task, various packages were used. First, *SentiWordNet* developed by Esuli and Sebastiani (2019) was used as the primary resource to get sentiment scores. *SentiWordNet* is a lexical resource used for opinion mining, and it is used to assign one of three sentiment scores: positive, negative or objective/neutral. Because the part of speech tagger used before gave us different tags than the ones accepted by *SentiWordNet*, a function was used to convert the notation. For instance, if a part of speech tagger was a *'J'*, the new tagger would be *'ADJ'*.

As previously mentioned, *SentiWordNet* is used as the primary resource for the sentiment score, but if for some reason a word is not found in this lexicon list, then *AFFIN* developed by Nielsen (2016) is used; if a word is still not found in the previous resources, then a lexicon list of positive and negative words (Sengupta 2018) is used. As last resort, if the word is not found in either of the last 3 cases, *sentiment.vader* from *NLTK* library (Bird and Tang 2021) is used.

**3.2.2 Cleaning and Preprocessing.** The data preparation process began by joining the sentiment result from aspect-based sentiment analysis with the *listings* data-set, based on the column *listing_id* and by averaging the sentiment score per review id and per listing id. Figure 1 shows that most of the reviews analysed were positive and the sentiment score results were distributed from -2.625 (negative sentiment) to 9.875 (positive sentiment).

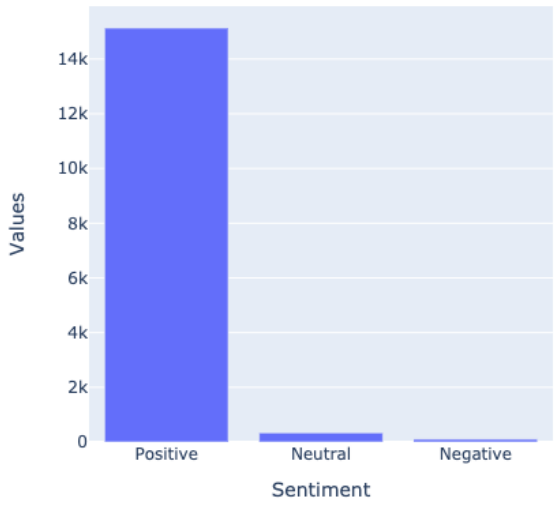



Figure 1: Sentiment distribution of the reviews per listing.

After this, irrelevant columns were dropped, resulting in 23 columns remaining in the data-set, as shown in in Appendix B. Since the price column had the character $, the next step was to strip away the symbol from this column. I then proceed with analysing the price column given that this was going to be the target feature (Figure 2).

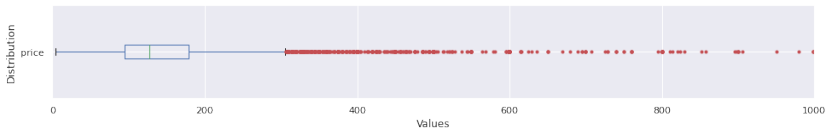



Figure 2: Price distribution.

I decided to keep only prices above 20 in order to avoid outliers and consequently make a better price prediction. Further analyses of the price feature, such sentiment polarity distribution on price, can be found in Appendix C.

Dealing with missing values was the following task. Missing values were identified in columns host_since, host_is_superhost, host_identity_verified, neighbourhood, neighbourhood_group_cleansed, bathrooms, bedrooms, beds, first_review, last_review, reviews_per_month. For columns bedrooms and beds the missing values were replaced with 0. In the case of first_review, last_review and reviews_per_month, the rows with missing values were dropped. The remaining columns contained a lot of missing values and since they were not relevant for the task in hands, I decided to drop them. The subsequent data-set was composed of 15562 rows and 17 features.

After dealing with missing values I proceeded to check columns *latitude* and *longitude*. Since the coordinates by themselves do not portray much information for the models being applied, I decided to create a new feature to cluster the different locations based on *longitude* and *latitude*. To achieve this, and with following the guide from Magiya (2019), K-means clustering was used. I decided to extract 22 different clusters since originally, all the listings in the data-set are from 22 different neighbourhoods in Amsterdam. The results were then merged with the main data-set as a new feature called *cluster_locations* and columns *latitude* and *longitude* were dropped. A map with the locations clustered can be found in Figure 3.

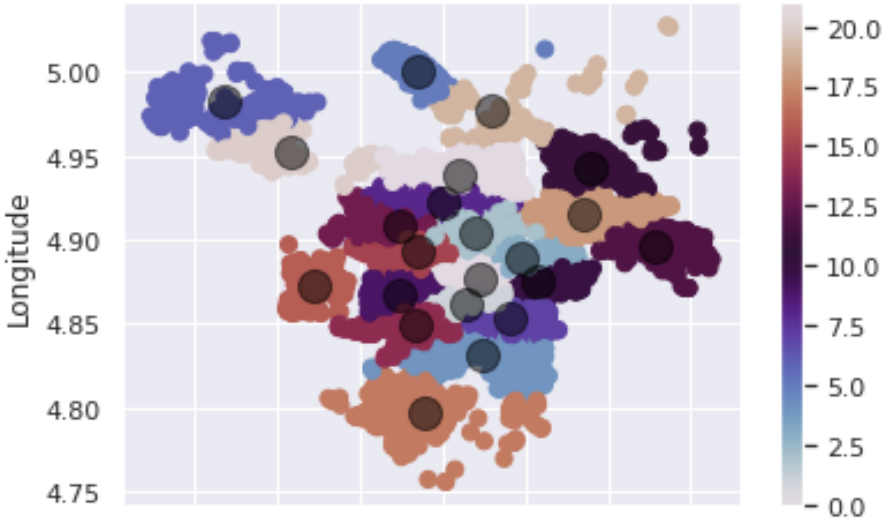


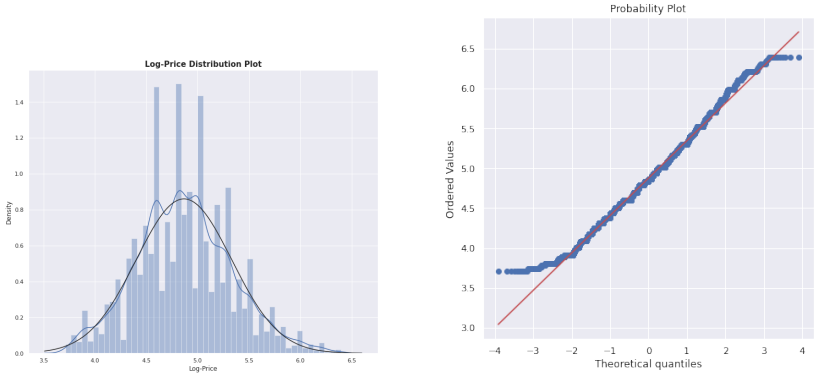

Figure 3: Amsterdam clustered locations.

Another feature was then created by dividing the number of bedrooms on each listing by the number of people it could accommodate. This gave the information of how many people could stay in the bedrooms.

Figure 4 shows that there is a positive-skewed distribution of the target feature *price*. In contrast to a normal distribution, where the median and mean of the distribution are the same, in a positive-skewed distribution, the mean is greater than the median. As we can note from the values of the mean (145.26) and median (127.0) of *price*, proving that a positive-skewed distribution is happening in this case.


Figure 4: Price Distribution

Therefore, values to the right will start acting as outliers which will most likely influence the performance of the price prediction models. To deal with the extreme values, log transformation of *price* was used. The result was saved as a new feature called *price_log*. As we can see from Figure 5 and Table 1 the skewness problem was




(a) (b)

Figure 5: Log Price distribution and goodfit.

| Mean | Median |
|---|---|
| 4.86 | 4.84 |

Table 1: Mean and Median Log Price

successfully dealt with by using the log transformation method, since both the mean and median of *log_price* are fairly similar.

After creating the *price_log* feature, the existence of outliers and their treatment is checked again (results in Appendix D).

To evaluate the relationship between the target feature *price_log* and all the other features, a correlation matrix was created with the help of Seaborn library (Waskom 2020). The correlation matrix helps to identify features that are highly correlated with the target feature, and therefore need to be removed. This is an important step to take since it helps us avoid multicolinearity which, according to Daoud (2017), can cause regression models to be unreliable. The result can be seen in Figure 6 and it shows that *accommodates* has the highest correlation to *price_log*, but even with this feature we can conclude that there is not a strong relationship with the target feature to the point where it needs to be dropped.

This relationship was further explored by checking Multicollinearity between the features. To do it, the Eigenvector of the correlation matrix was calculated. The results can be found in Appendix E.

Because there were still some categorical variables present in the data-set (such as *property_type* or *room_type*), the creation of dummy variables was the next step. To do so I extracted the variables which had the datatype *object* and proceeded with the creation of the dummy variables using the following code:

```
create_categorical_dum = x.select_dtypes(include=['object'])
create_categorical_dum.head()
object_dum = pd.get_dummies(create_categorical_dum)
object_dum.head()
x = x.drop(list(create_categorical_dum.columns), axis=1)
x = pd.concat([x,object_dum], axis=1)
```

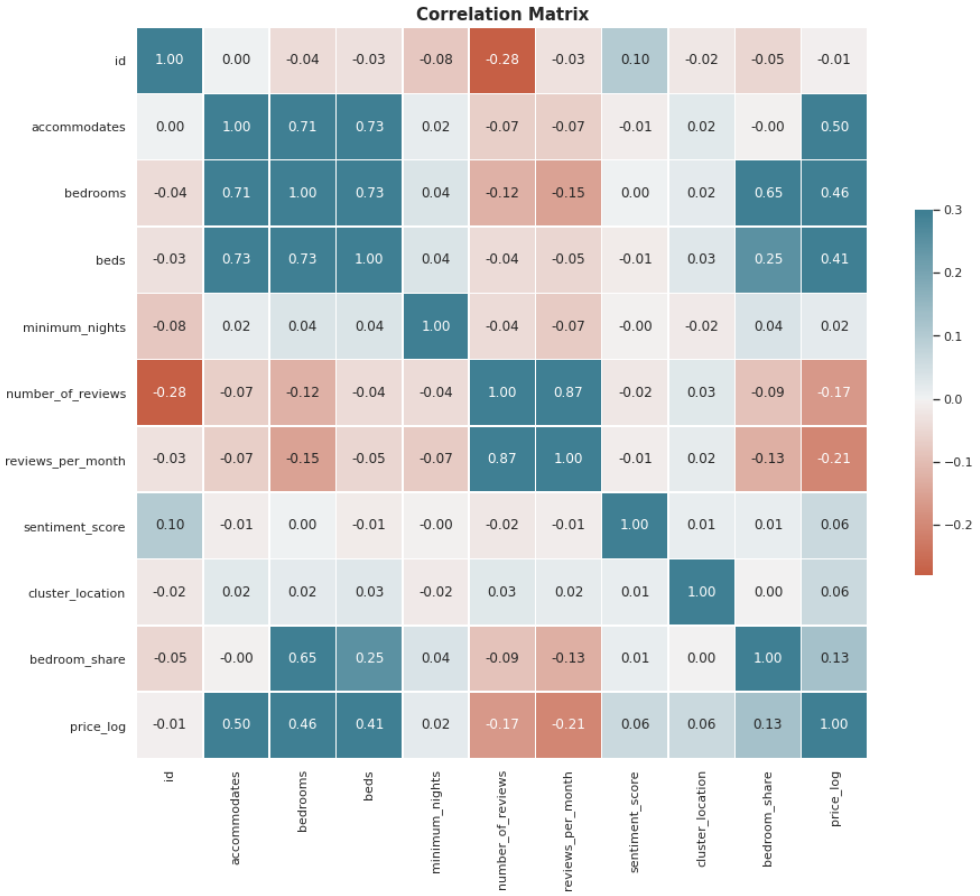



Figure 6: Correlation matrix

Since two versions of each model were going to be created - one trained on data that included the sentiment feature and one without the sentiment feature - I created a new version of the data-set by dropping the *sentiment* feature. The next task was to split these two final data-sets into train and test data-sets. To accomplish it, functions from the Scikit-learn library were used and the datasets were divided with an 80%-20% ratio.

Finally, I check the feature importance by taking the results from the correlation matrix. The results can be found on the tables that are represented in Figure 7. The process was done twice, one before creating dummy variables for the categorical features and another after it. As we can note from it, *sentiment_score* has a low score and therefore points towards low importance of the feature on the prediction of the listing prices. Nevertheless, the original plan is still going to be followed (dividing the models into two versions like explained before, and check how is the model's prediction power affected).

| | rank | weight |
|---|---|---|
| accommodates | 0 | 0.246848 |
| bedrooms | 1 | 0.216090 |
| beds | 2 | 0.164721 |
| reviews_per_month | 3 | 0.042107 |
| bedroom_share | 4 | 0.016024 |
| sentiment_score | 5 | 0.004147 |
| minimum_nights | 6 | 0.000347 |
| id | 7 | 0.000034 |

(a)

| | rank | weight |
|---|---|---|
| accommodates | 0 | 0.246848 |
| bedrooms | 1 | 0.216090 |
| room_type_Entire home/apt | 2 | 0.186720 |
| room_type_Private room | 3 | 0.184851 |
| beds | 4 | 0.164721 |
| reviews_per_month | 5 | 0.042107 |
| bedroom_share | 6 | 0.016024 |
| cluster_location_8 | 7 | 0.008712 |
| cluster_location_9 | 8 | 0.007754 |
| cluster_location_1 | 9 | 0.007514 |

(b)

Figure 7: (a) Feature importance without dummy variables; (b) Top 10 feature importance with dummy variables

**3.2.3 Baseline Model: Linear Regression.** Linear Regression was chosen as the baseline model to evaluate the performance of the other models. As already mentioned, two versions of the models were fitted to the data - one with the sentiment feature and one without.

Linear Regression needs a linear relationship between the dependent and independent variables. Although, as already proven, the features present in this data-set are more complex than a linear relationship. In order to be able to use linear regression models with non-linear data, we can apply Polynomial Regression. To do this we simply transform our features into higher-order terms. *PolynomialFeatures* function from the Scikit-learn pre-processing library was used to achieve this.

Features on both the train and test set were normalized using the *StandardScaler* function from the Scikit-learn library before training any model.

To find the best hyper-parameters, functions were created around the functionality of *GridSearchCV* algorithm from Scikit-learn model_selection library. First a function to find the hyper-parameters of the model in question was created. After coming up with possible values for the hyper-parameters, I applied the *GridSearch* algorithm, with cross-validation of 5 and checked the best results. Note that this process was created for all the models that were implemented, not only Linear Regression. The result was then used to select the hyperparameters, and finally, the Linear Regression model was fitted to the training sets.

**3.2.4 Ridge Regression.** Ahn et al. (2012) stated that the criticism related to Linear Regression models such as non-linearity within the features in the data, or multi-collinearity issues or even issues regarding the inclusion of outliers can be overcome with Ridge Regression models.

The overall idea of ridge regression is to attempt to reduce the variability amongst the data by increasing the bias of the model. In other words, by using Ridge Regression we want the model to perform a bit worse during training for the sake of overcoming overfitting and have a model that can be more generalised. To do this, Ridge Regression model includes a weighted L2 regularization penalty term to the error cost function in

order to help the algorithm reduce overfitting (Kalehbasti, Nikolenko, and Rezaei 2019). Mathematically, Ridge Regression hopes to minimize $J(\theta)$ where $X$ is a matrix and $\alpha$ is a hyperparameter (1).

$$J(\theta) = ||y - X\theta||_2^2 + \alpha||\theta||_2^2 \tag{1}$$

To apply this model I started by using *GridSearch* algorithm with cross-validation equal to 5. The result was then used to select the hyperparameters and the Ridge Regression model was created and fitted to the training sets.

**3.2.5 Support Vector Regression.** The goal of the support vector machine algorithm is to find a hyperplane in an N-dimensional space that can distinctly classify the data points. Even though it was developed for the task of classification, it has been used for regression as well. Therefore, when implemented as a regression model, it has the goal of finding the optimal solution of a boundary for the regression line in a high dimensional space, as shown in Figure 8. This means that if we consider a feature matrix $X$, and $Y(true)$ as our target vector of the training set, the goal is to find a function $f(x)$ such that the difference between $f(x)$ and $Y(true)$ is less than or equal to $\epsilon$ (threshold between the boundaries and the regression line) (Figure 8).

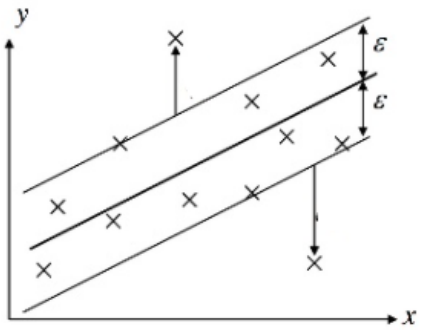



Figure 8: Support Vector Machine

Even though one of the main attractions of SVM is its suitability to work with high dimensional spaces - which is the case in this research - it is not a good model to use in large datasets since it requires a long period of training time. Regardless, one of the goals of this thesis is to compare results with prior research and, because this model had a good performance in the work done by Kalehbasti, Nikolenko, and Rezaei (2019), the model was picked to be used. It is worth noting that SVM has the ability to model non-linear relationships and, as already mentioned, this is the case in our feature space.

The hyperparameter *kernel* used for the model was RBF (Radial Basis Function) so it could identify the linear boundary in the N-dimensional feature space, as explained by Kalehbasti, Nikolenko, and Rezaei (2019). For the remaining hyperparameters, the *GridSearch* algorithm was again used with cross-validation equal to 5. As in the previous models, the result was then used to select the remaining hyperparameters, the SVM model was created and then fitted to the train datasets.

**3.2.6 XGBoost.** According to Fei (2020), XGBoost is a machine learning algorithm that has been showing promising results in real-world problems related to rental prices. The XGBoost is an ensemble tree method, similar to the gradient boosting algorithm, that applies the principle of boosting weak learners using the gradient descent. Initially created by Tianqi Chen, XGBoost or Extreme Gradient Boosting is now an open-source library. Among other arguments, Fei (2020) states that this algorithm works well with large datasets and it should take a short time to implement. Furthermore, results have been showing a higher accuracy when compared to the models previously mentioned. Because of its novelty and promising results it seemed pertinent to include the model in this research.

As in the other models, *GridSearch* algorithm was again used with cross-validation equal to 2, followed by the creation of XGBoost Regressor model with the hyperparameter results from the *GridSearch* and fitting it to the train datasets.

**3.2.7 Neural Network.** Neural Networks are used for a range of prediction tasks, including price prediction. They usually perform better in such tasks when compared to other models due to the hidden layers in their architecture. These layers are used to improve the accuracy in predictions. Besides containing hidden layers, a Neural Network has an input layer(s) and an output layer(s). Each layer consists of a certain number of neurons that are connected with each other, mimicking the way a human brain works. An example of a simple Neural Network can be seen in Figure 9. In this example the input layer contains 3 neurons, the hidden layer has 5 neurons and the output layer one neuron.

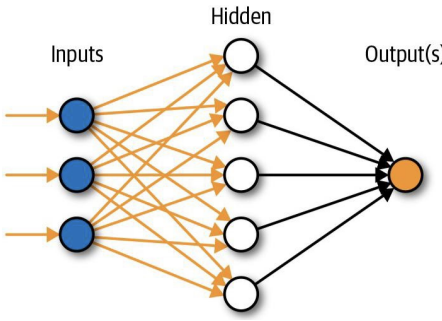



Figure 9: Neural Network

For this research, a Neural Network with similar architecture as the one used in Kalehbasti, Nikolenko, and Rezaei (2019) paper is created. The Neural Network was built using the the Sequential model available in Keras library. The architecture of the Neural Network consisted of 6 fully connected layers: the input layer with 48 neurons and relu activation function using a normal distribution to initialise the weights; 32 neurons in the first hidden layer, 24 neurons on the second hidden layer, 16 neurons on the third hidden layer and finally 8 neurons on the fourth hidden layer, all using relu activation functions; the output layer had 1 neuron with a linear activation function. A detailed picture of this architecture can be found in Appendix F. The compile function was then used with *Mean Squared Error* as the *loss function* parameter, *Adam* as the *optimizer* parameter, *Mean Squared Error* and *Mean Absolute Error* as the *metrics* parameter. The model was then fitted to the training data sets, using 20 epochs and a batch size of 30. The model loss can be seen in Figure 10.

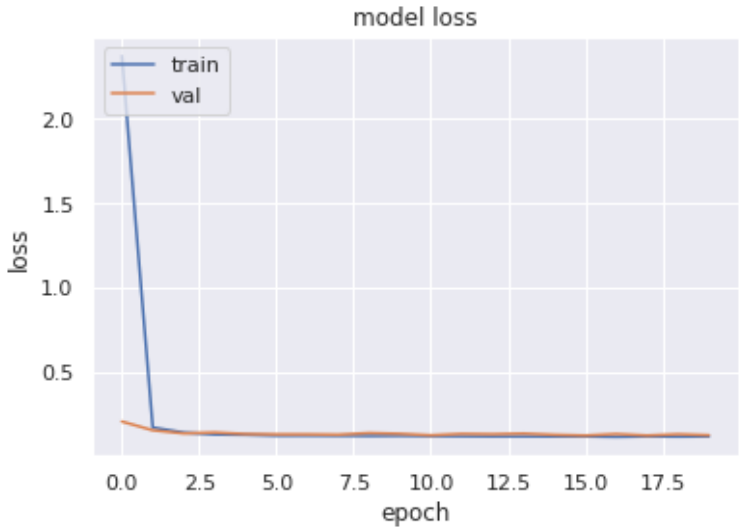



Figure 10: Neural Network model's loss

### 3.3 Evaluation metrics

To evaluate and compare the results from the models analysed, three different evaluation metrics were used: $R^2$, *RMSE* and *MAE*. *MAE* or *Mean Absolute Error* measures the average magnitude of errors between the predicted values and the actual values and it can be calculated with formula (2), where $y_i$ represents the true value of $y$ and $\tilde{y}_i$ represents the predicted value of $y$. An *MAE* value of 0 indicates that there was no error done by the model, in other words, the model made a perfect prediction.

$$MAE = \frac{1}{n}\sum |y_i - \hat{y}_i| \tag{2}$$

*RMSE* or *Root Mean Square Error* is simply the standard deviation of the prediction errors. In other words, *RMSE* gives us information on how the prediction errors are. When analysing the *RMSE* results of a model we get an idea of how much error the model typically make in its predictions. It can be calculated using formula (3).

$$RMSE = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n}} \tag{3}$$

$R^2$ measures how close the data are to the fitted line. In other words, it represents the proportion of variance for a dependent variable that is explained by an independent variable. It can be calculated using the formula (4).

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \overline{y})^2} \tag{4}$$

According to Chai and Draxler (2014) research, MAE is a more appropriate score to measure prediction error, therefore I decided to include both MAE and RMSE in my results. The best model will be the one with the lowest error rate and the highest $R^2$.

## 4. Results

As mentioned the first task was to perform Aspect Based Sentiment Analysis so we could get a more realistic sentiment linked to each Airbnb property present in the dataset. Figure 11 shows step (1) of aspect detection being applied. English reviews were identified by adding a new column as we can note from the image.

| | listing_id | id | comments | langs |
|---|---|---|---|---|
| **128076** | 4413032 | 632770021 | It's a good room. It's near the city. | en |
| **267422** | 14172612 | 165981671 | We are a family of 4 traveling with our 2 daug... | en |
| **312317** | 17875133 | 554903297 | What a wonderful place to stay in Amsterdam, c... | en |
| **371571** | 23209973 | 299651385 | We had a great stay at Cherique's apartment! S... | en |
| **73372** | 1723551 | 414620920 | Very cozy place, offering all you need for a c... | en |

Figure 11: Aspect detection step 1

Step (2), step (3) and step(4) refer to the normalization of the reviews and the extraction of opinion units and spacy tags. A result of these steps applied to a review can be seen in Figure 12.

| | REVIEWS | OPINION_UNITS | spacy_tag |
|---|---|---|---|
| **9** | we had a great stay at cheriques apartment! sh... | we had a great stay at cheriques apartment! | [(great, JJ), (stay, NN), (cherique, NNS), (ap... |
| **10** | we had a great stay at cheriques apartment! sh... | she responded quickly, provided tips and when ... | [(respond, VBD), (quickly, RB), (provide, VBN)... |
| **11** | we had a great stay at cheriques apartment! sh... | the apartment is really nice, spacious and eve... | [(apartment, NN), (really, RB), (nice, JJ), (s... |
| **12** | we had a great stay at cheriques apartment! sh... | the location is really great! | [(location, NN), (really, RB), (great, JJ)] |
| **13** | we had a great stay at cheriques apartment! sh... | its within walking distance of the city center... | [(walk, VBG), (distance, NN), (city, NN), (cen... |
| **14** | we had a great stay at cheriques apartment! sh... | we would definitely stay here again | [(would, MD), (definitely, RB), (stay, VB), (h... |

Figure 12: Aspect detection on a review steps 2 - 4

After aspect detection, aspect classification had to be done. Figure 13 shows the result of the sentiment calculation on the review previously seen. On the 'Sentiment Score' column we have the score that was calculated from the function described in the previous section. The 'Sentiment Polarity' column was defined by looking at that score. For example, if the score is negative then the 'Sentiment Polarity' will state 'Negative'.

| | REVIEWS | Sentiment Polarity | Sentiment Score | OPINION_UNITS | spacy_tag |
|---|---|---|---|---|---|
| **9** | we had a great stay at cheriques apartment! sh... | Positive | 3.000 | we had a great stay at cheriques apartment! | [(great, JJ), (stay, NN), (cherique, NNS), (ap... |
| **10** | we had a great stay at cheriques apartment! sh... | Negative | -1.875 | she responded quickly, provided tips and when ... | [(respond, VBD), (quickly, RB), (provide, VBN)... |
| **11** | we had a great stay at cheriques apartment! sh... | Positive | 2.000 | the apartment is really nice, spacious and eve... | [(apartment, NN), (really, RB), (nice, JJ), (s... |
| **12** | we had a great stay at cheriques apartment! sh... | Positive | 3.000 | the location is really great! | [(location, NN), (really, RB), (great, JJ)] |
| **13** | we had a great stay at cheriques apartment! sh... | Neutral | 0.000 | its within walking distance of the city center... | [(walk, VBG), (distance, NN), (city, NN), (cen... |
| **14** | we had a great stay at cheriques apartment! sh... | Positive | 1.000 | we would definitely stay here again | [(would, MD), (definitely, RB), (stay, VB), (h... |

Figure 13: Aspect classification in one example.

As already mentioned in the previous chapter, the results were merged with the dataset listings so the models could be trained with the sentiment score as a feature.

This research used various machine learning models to analyse the relationship between the feature target price and features *accommodates*, *bedrooms*, *beds*, *minimum_nights*, *reviews_per_month*, *sentiment_score*, *bedroom_share*, *host_is_superhost*, *host_identity_verified*, *room_type* and *cluster_location*. Table 2 shows the results with the evaluation metrics scores that were selected. These results are in regards to the models that were trained with the dataset that contained the *sentiment_score* feature.

| **Method** | **Train $R^2$** | **Train RMSE** | **Train MAE** | **Test $R^2$** | **Test RMSE** | **Test MAE** |
|---|---|---|---|---|---|---|
| Linear Regression | 0.449 | 0.344 | 0.266 | 0.439 | 0.351 | 0.270 |
| Ridge Regression | 0.449 | 0.344 | 0.266 | 0.440 | 0.350 | 0.270 |
| SVR | 0.524 | 0.320 | 0.241 | 0.417 | 0.357 | 0.274 |
| **XGBoost** | **0.554** | **0.310** | **0.238** | **0.453** | **0.346** | **0.266** |
| Neural Network | 0.424 | 0.352 | 0.273 | 0.347 | 0.378 | 0.292 |

Table 2: Results from models trained with sentiment score feature

Looking at the results on Table 2 and the values of $R^2$ we can conclude that overall, the performances of the models were low, with the highest accuracy value being only 45.3%. Possible reasons for this will be laid out on the next section. By looking at the results of Linear Regression and Ridge Regression models we can see that their performance was very similar, making their error scores on the train set almost the same - 0.34 RMSE and 0.26 MAE. If we compare the RMSE and MAE values between train and test for all the models we see that there is a slight increase of the values. This may imply overfitting is happening. One possible reason for this may be the fact that cross-validation was not used when training the models. With an accuracy score of 55.4% on train and 45.3% on test, XGBoost was the best performing model. Not only the accuracy was the best but the model also showed the lowest train RMSE (0.310), lowest train MAE (0.238), lowest test RMSE (0.346) and lowest test MAE (0.266). Neural Network model appeared to be the worst model among all with an accuracy of only 42.4% on train and 34.7% on test. Possible reasons for this outcome as well as future changes to the model will be discussed on the next section. Overall, by looking at the error scores, the only models that might be generalizable are Linear Regression, Ridge Regression and Neural Network, given that the difference of those score between train and test is small.

Table 3 shows the results in regards to the models that were trained with the dataset that did not the *sentiment_score* feature.

| **Method** | **Train $R^2$** | **Train RMSE** | **Train MAE** | **Test $R^2$** | **Test RMSE** | **Test MAE** |
|---|---|---|---|---|---|---|
| Linear Regression | 0.448 | 0.344 | 0.266 | 0.434 | 0.350 | 0.270 |
| Ridge Regression | 0.448 | 0.344 | 0.266 | 0.440 | 0.350 | 0.270 |
| SVR | 0.512 | 0.324 | 0.245 | 0.422 | 0.356 | 0.273 |
| **XGBoost** | **0.537** | **0.315** | **0.243** | **0.456** | **0.345** | **0.265** |
| Neural Network | 0.459 | 0.341 | 0.263 | 0.388 | 0.366 | 0.282 |

Table 3: Results from models trained without sentiment score feature

Again Linear Regression and Ridge Regression performed similarly during training and on the test set, with their test error scores still being slightly higher than the training error scores. The accuracy of the SVR model, both on the train and on the test set, slightly dropped when compared to the SVR results on Table 2. Again XGBoost was the best performing model showing a 45.6% accuracy on the test set, an RMSE of 0.345 and an MAE of 0.265. Neural Network was still the worst performing model with a test accuracy of 38.8%, however, the results from train and the error test scores seemed to improve this time around.

As it was led to believe by the results shown on Figure 7, the absence of sentiment score feature did not seem to greatly impact the performance of the models. After training the models without the sentiment scores, the performance results of the models dropped slightly in some models, but in others, it increased slightly. The best result was shown by the XGBoost model trained without the data that included sentiment score (45.6%).

## 5. Discussion

This research aimed to explore how sentiment scores could play a role in price estimation models for Airbnb listings. This entailed the comparison of machine learning models' performance when using sentiment scores that were calculated through a novel Natural Language Processing technique called aspect-based sentiment analysis. In this section an interpretation of all the results will be given as well as possible solutions to be used in future research.

### 5.1 Interpreting models' results and comparison with previous literature

Looking at table 4 and comparing the results of the best model in this research (XGBoost) with the results from the best model in Kalehbasti, Nikolenko, and Rezaei (2019) (SVR), MAE and RMSE values shown in this research seem to be reasonably close with previous literature. Although when I compare the results for $R^2$, I conclude that in my case the values are smaller. This means that my models are considerably more noisy than the ones built in previous literature. In other words, the data points in my case are more distant from the fitted line than with the ones from the compared research.

| **Best Model** | **Train $R^2$** | **Train RMSE** | **Train MAE** | **Test $R^2$** | **Test RMSE** | **Test MAE** |
|---|---|---|---|---|---|---|
| XGBoost | 0.537 | 0.315 | 0.243 | 0.456 | 0.345 | 0.265 |
| SVR | 0.777 | 0.327 | 0.213 | 0.690 | 0.3835 | 0.276 |

Table 4: Comparison between this research and Kalehbasti, Nikolenko, and Rezaei (2019)

Due to the limited amount of data to train the models, it was expected that the accuracy of the models would be somewhat low. As evidenced in the results, models showed a decrease in the performance when the unseen test set was used in both cases - models trained with sentiment score and without sentiment score. A range of reasons can be attributed to this, quality of data used being the most realistic. As shown in the information given by the correlation matrix (Figure 6), it appears that there were not enough individual variables correlated with price. The results from the Eigenvector

values (Appendix E) seem to point to some multicollinearity between features, which may be a cause for poor results in my models since it can affect their precision. The curse of dimensionality phenomenon might have also posed as a reason for the bad results. This phenomenon may arise when the data being used to train the models has too many features, risking the models to overfit or making the observations in our data to appear equidistant from the other ones (making them harder to cluster).

Class imbalance might have also affected the results of my models. Even though sentiment was not used as a class variable, the variable *cluster_location* was created and used to train the models. Class imbalance refers to a problem present in Machine Learning where the total number of a class in the data is far less than another class. By looking at Figure 14, it is notable that some location clusters have a far less number than others. For future research, this problem could be mitigated through sampling based approaches or cost function based approaches. Sampling based approaches refer to either oversampling the minority class, undersampling the majority class or approaching it as a mix of oversampling and undersampling. Cost function based approaches are related to using customized cost functions on the models to deal with the unbalanced data.

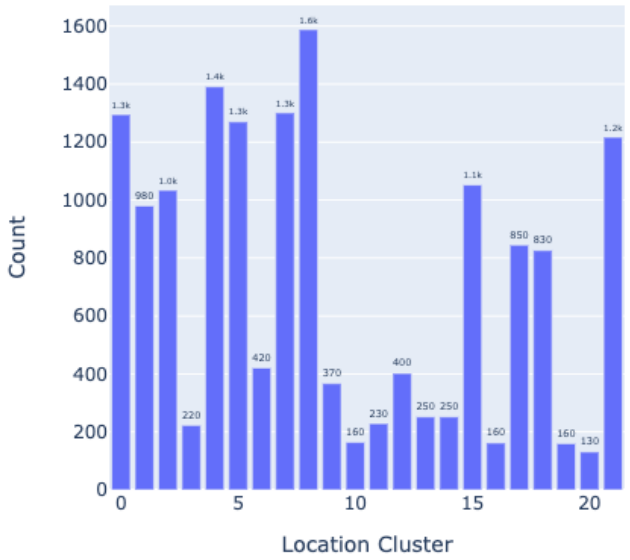



Figure 14: Location cluster distribution

Another realistic reason for the low performance might be related to the price target feature nature. Price in this case, merely represents an average of the price per listing although, since this price changes throughout the year, it makes the nature of our target feature less reliable. For future work, this problem could be seen as a time series problem and get data related to the listings price over time.

### 5.2 Interpreting feature importance results

To find the answer to the research question about the feature importance and, to some extent the main research question, two tables regarding feature importance were calculated through the correlation matrix (Figure 7). The findings presented were in line with the ones found in the literature like the research by Brownlee (2021), with accommodates and bedrooms being the most important features in the dataset. From this result, it was speculated that sentiment score would not have a big importance in the performance of the models being built. This was confirmed by training the models without sentiment score and seeing that the performance of the models did not see a significant difference.

### 5.3 Aspect Based Sentiment Analysis results

Another consideration to be taken into account are the results from aspect-based sentiment analysis. It appears that the reviews used were inflated since sentiment polarity distribution was overwhelmingly positive. This may be attributed to the methods used in this research which found mainly positive opinions in each review; or simply by the nature of the dataset provided. These results might have originated a class imbalance problem. As already stated, class imbalance is a phenomenon that appears when we have one class overwhelmingly present. This could be seen when I transformed the sentiment into 3 classes: *Positive*, *Negative* and *Neutral* (Figure 1). In my case, as already stated, the class *Positive* is overwhelmingly present.

I would argue that the nature of the data is what is causing this problem since similar results have been found in previous papers, as the one developed by Liu (2021), even though sentiment scores were calculated differently. A possible suggestion for future work would be to use a different method for aspect modelling: LDA or latent Dirichlet allocation since it has been reporting good results.

## 6. Conclusion

This research aimed to find answers to the research questions described in the introduction section. To get to these answers, machine learning models alongside aspect-based sentiment analysis were introduced. The main drive of the research was to prove that customer reviews play a big role in improving price prediction since customers and property hosts evaluate price based on previous feedback. To evaluate if including sentiment score as a feature in the price prediction models would make the models perform better, two versions of the models were built. The sentiment score was calculated by using aspect-based sentiment analysis, allowing for a more detailed picture of the sentiment a customer review presents. Unfortunately, the results of the two versions of the models were not satisfactory and it was concluded that the sentiment score fetched from the customer reviews does not have a big impact on the prediction of Airbnb prices. The identification of the features with the most importance was also achieved. The results aligned with what was discovered in previous papers. In addition to the questions a fairly new prediction model was used, XGBoost. It was concluded that, even though the results were sub-par, XGBoost was the model that provided the most satisfactory scores among all.

Some enhancements that can be made in the future regarding this research include increasing the number of features with a higher correlation with the price. For example, seasonality is something that is known to directly affect the price of Airbnb listing since it is observed that rental prices increase during holiday seasons. I also believe that,

besides using a larger dataset, it would be important to include customer reviews that overall have a more balanced sentiment distribution. A well-balanced dataset should have a positive impact on building a more accurate model. To deal with class imbalance some methods were suggested to be used, such us undersampling or oversampling the unbalanced classes.

The biggest challenge faced during this research was the scattered information that was available regarding aspect-based sentiment analysis alongside the processing time required for this task. Besides this, the low number of features that had a relevant correlation with the target feature was also a challenge that had a great impact on the final results. Nevertheless, I believe that if the suggestions given were to be met, the results could achieve improvements.

## References

2021. Get the data - inside airbnb. adding data to the debate. http://insideairbnb.com/get-the-data.html.

Ahn, Jae Joon, Hyun Woo Byun, Kyong Joo Oh, and Tae Yoon Kim. 2012. Using ridge regression with genetic algorithm to enhance real estate appraisal forecasting. *Expert Systems with Applications*, 39(9):8369–8379.

AirbnbEng. 2016. Aerosolve: Machine learning for humans. https://medium.com/airbnb-engineering/aerosolve-machine-learning-for-humans-55efcf602665.

Bird, Steven and Liling Tang. 2021. Nltk sentiment package. https://www.nltk.org/api/nltk.sentiment.html.

Brownlee, Jason. 2021. Xgboost for regression. https://machinelearningmastery.com/xgboost-for-regression/.

Bustamante, Jaleesa. 2021. Airbnb statistics. https://ipropertymanagement.com/research/airbnb-statistics.

Chai, Tianfeng and Roland R Draxler. 2014. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250.

Daoud, Jamal I. 2017. Multicollinearity and regression analysis. In *Journal of Physics: Conference Series*, volume 949, page 012009, IOP Publishing.

Esuli, Andrea and Fabrizio Sebastiani. 2019. Sentiwordnet. https://github.com/aesuli/sentiwordnet.

Fei, Yue. 2020. *California Rental Price Prediction Using Machine Learning Algorithms*. Ph.D. thesis, UCLA.

Hinckley, Dan. 2015. New study: Data reveals 67% of consumers are influenced by online reviews. https://moz.com/blog/new-data-reveals-67-of-consumers-are-influenced-by-online-reviews.

Htay, Su Su and Khin Thidar Lynn. 2013. Extracting product features and opinion words using pattern knowledge in customer reviews. *The Scientific World Journal*, 2013.

Kalehbasti, Pouya Rezazadeh, Liubov Nikolenko, and Hoormazd Rezaei. 2019. Airbnb price prediction using machine learning and sentiment analysis. *arXiv preprint arXiv:1907.12665*.

de Kok, Sophie, Linda Punt, Rosita van den Puttelaar, Karoliina Ranta, Kim Schouten, and Flavius Frasincar. 2018. aggregated aspect-based sentiment analysis with ontology features. *Progress in Artificial Intelligence*, 7(4):295–306.

Liu, Peilu. 2021. Airbnb price prediction with sentiment classification.

Luo, Yi. 2018. What airbnb reviews can tell us? an advanced latent aspect rating analysis approach.

Ma, Yixuan, Zhenji Zhang, Alexander Ihler, and Baoxiang Pan. 2018. Estimating warehouse rental price using machine learning techniques. *International Journal of Computers Communications & Control*, 13(2):235–250.

Magiya, Joseph. 2019. Clustering gps coordinates and forming regions with python. https://levelup.gitconnected.com/clustering-gps-co-ordinates-forming-regions-4f50caa7e4a1.

McNeil, Brandon. 2020. Price prediction in the sharing economy: A case study with airbnb data.

Nielsen, Finn Aarup. 2016. Affin lexicon. https://pypi.org/project/afinn/, journal=PyPI.

Sengupta, Paramartha. 2018. Positive and negative words. https://www.kaggle.com/harshaiitj08/positive-and-negative-words#positive-words.txt.

Spacy. 2021. Spacy en_core_web_lg-3.0.0. https://github.com/explosion/spacy-models/releases//tag/en_core_web_lg-3.0.0.

Staff, Investopedia. 2021. Sharing economy definition. https://www.investopedia.com/terms/s/sharing-economy.asp.

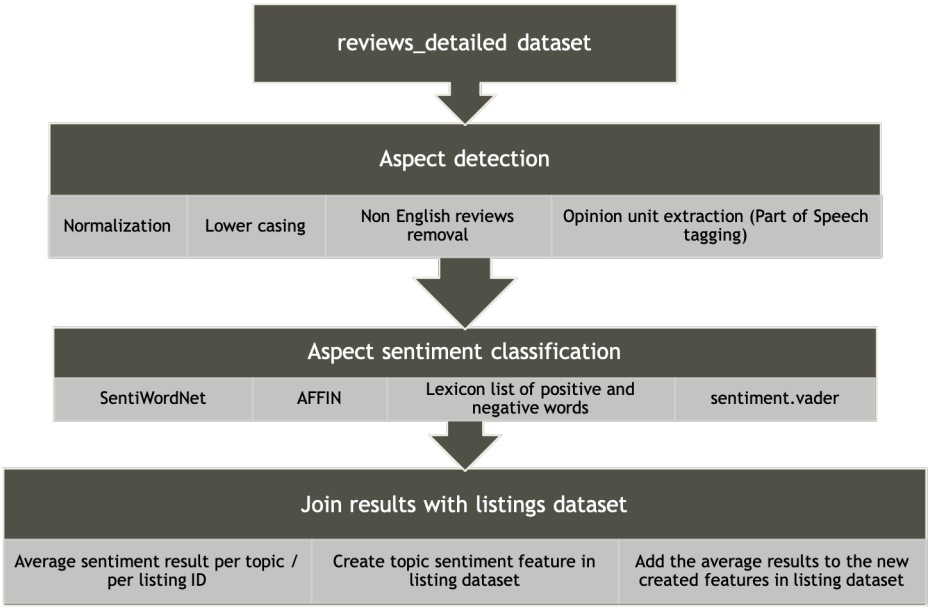
Teubner, Timm. 2014. Thoughts on the sharing economy. In *Proceedings of the International Conference on e-Commerce*, volume 11, pages 322–326.

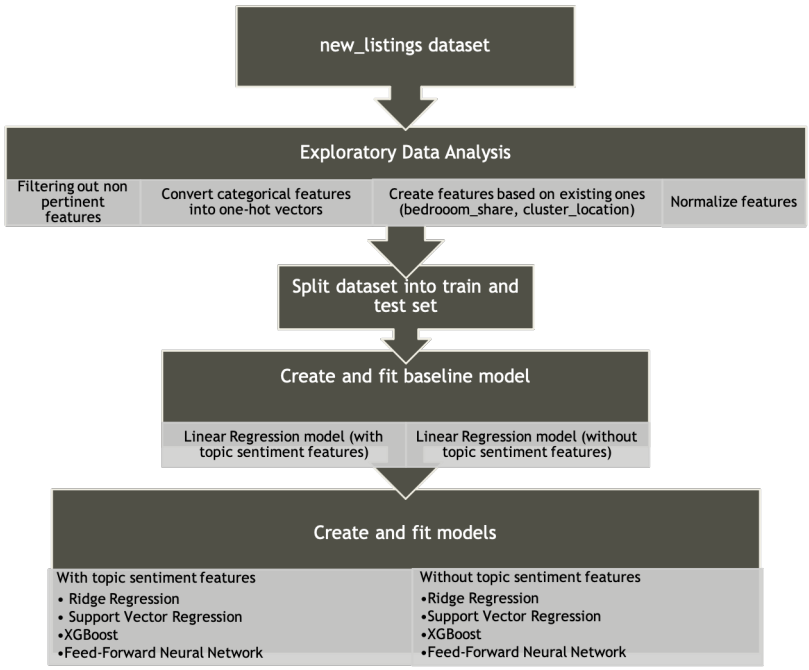
Wang, Bo and Min Liu. 2015. Deep learning for aspect-based sentiment analysis. *Stanford University report*.

Waskom, Michael. 2020. https://seaborn.pydata.org/.

Yu, Hujia and Jiafu Wu. 2016. Real estate price prediction with regression and classification. *CS229 (Machine Learning) Final Project Reports*.

## Appendix A: Flow of work

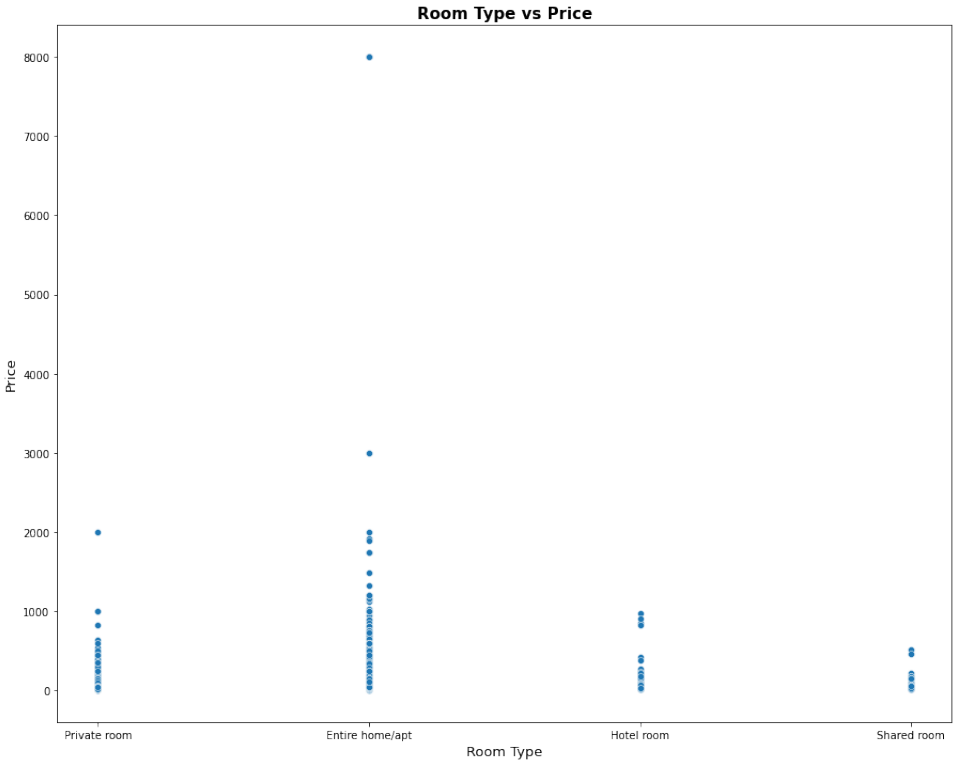


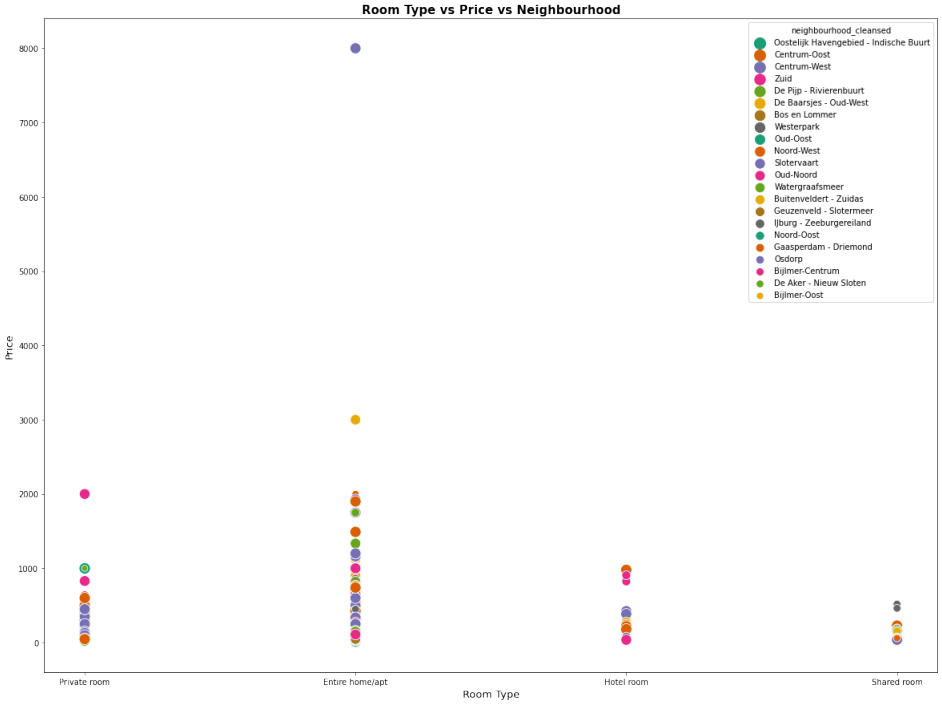



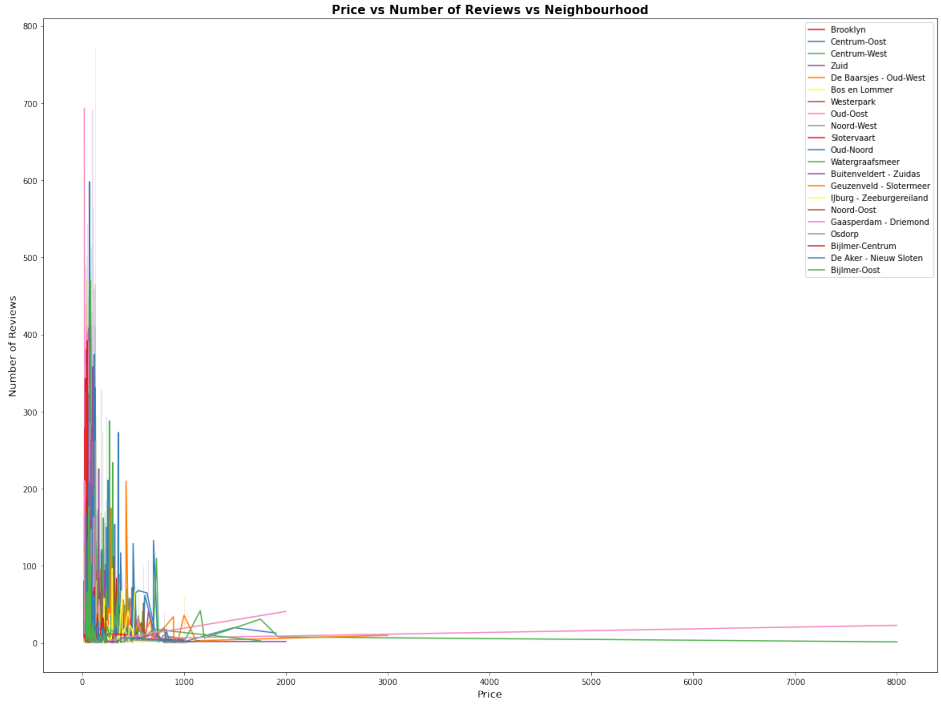
**Appendix B: Listing data-set after dropping irrelevant columns**

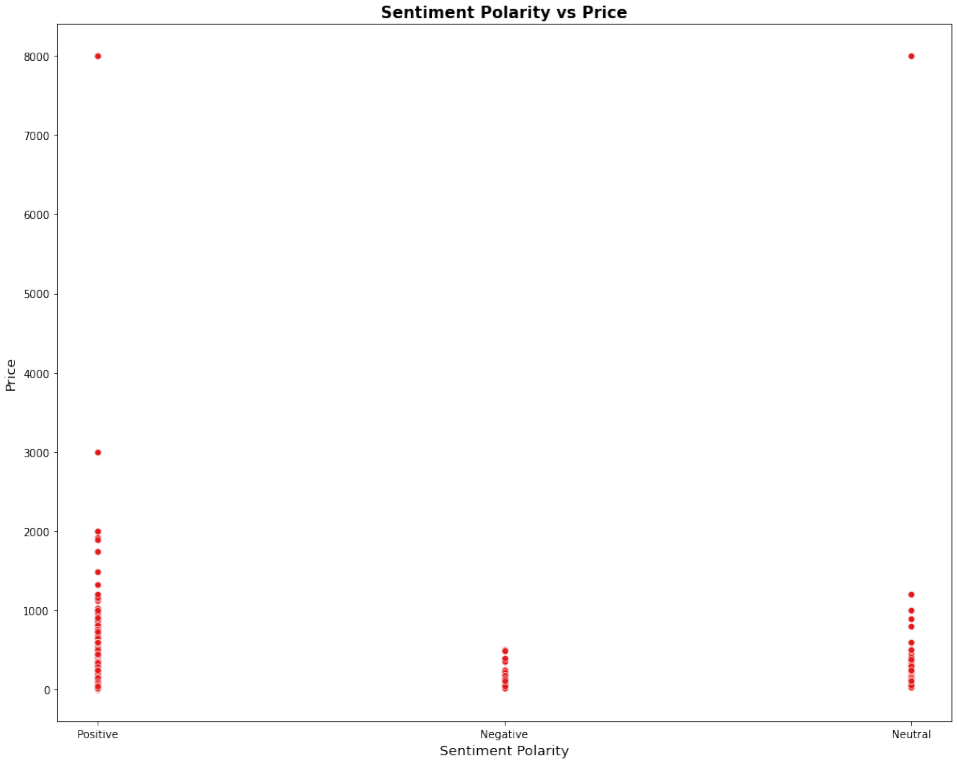
| Column | Min | Max | DataType |
|---|---|---|---|
| id | 2818 | 47882460 | int64 |
| host_since | 2008-09-24 | 2018-11-29 | object |
| host_is_superhost | f | t | object |
| host_identity_verified | f | t | object |
| neighbourhood_cleansed | BijlmerCentrum | Zuid | object |
| neighbourhood | Amsterdam, North Holland | NaN | object |
| neighbourhood_group_cleansed | NaN | NaN | float64 |
| latitude | 52.2893 | 52.4251 | float64 |
| longitude | 4.75594 | 5.02769 | float64 |
| property_type | Barn | Yurt | object |
| room_type | Entire home/apt | Shared room | object |
| accommodates | 1 | 16 | int64 |
| bathrooms | NaN | NaN | float64 |
| bedrooms | 1 | 30 | float64 |
| beds | 0 | 33 | float64 |
| price | $1,000.00 | $999.00 | object |
| minimum_nights | 1 | 1001 | int64 |
| number_of_reviews | 1 | 859 | int64 |
| first_review | 2009-03-30 | 2021-02-04 | object |
| last_review | 2011-06-26 | 2021-02-08 | object |
| reviews_per_month | 0.01 | 35.53 | float64 |
| sentiment_score | -2.625 | 9.875 | float64 |
| polarity | Negative | Positive | object |

## Appendix C: Price analysis with other attributes

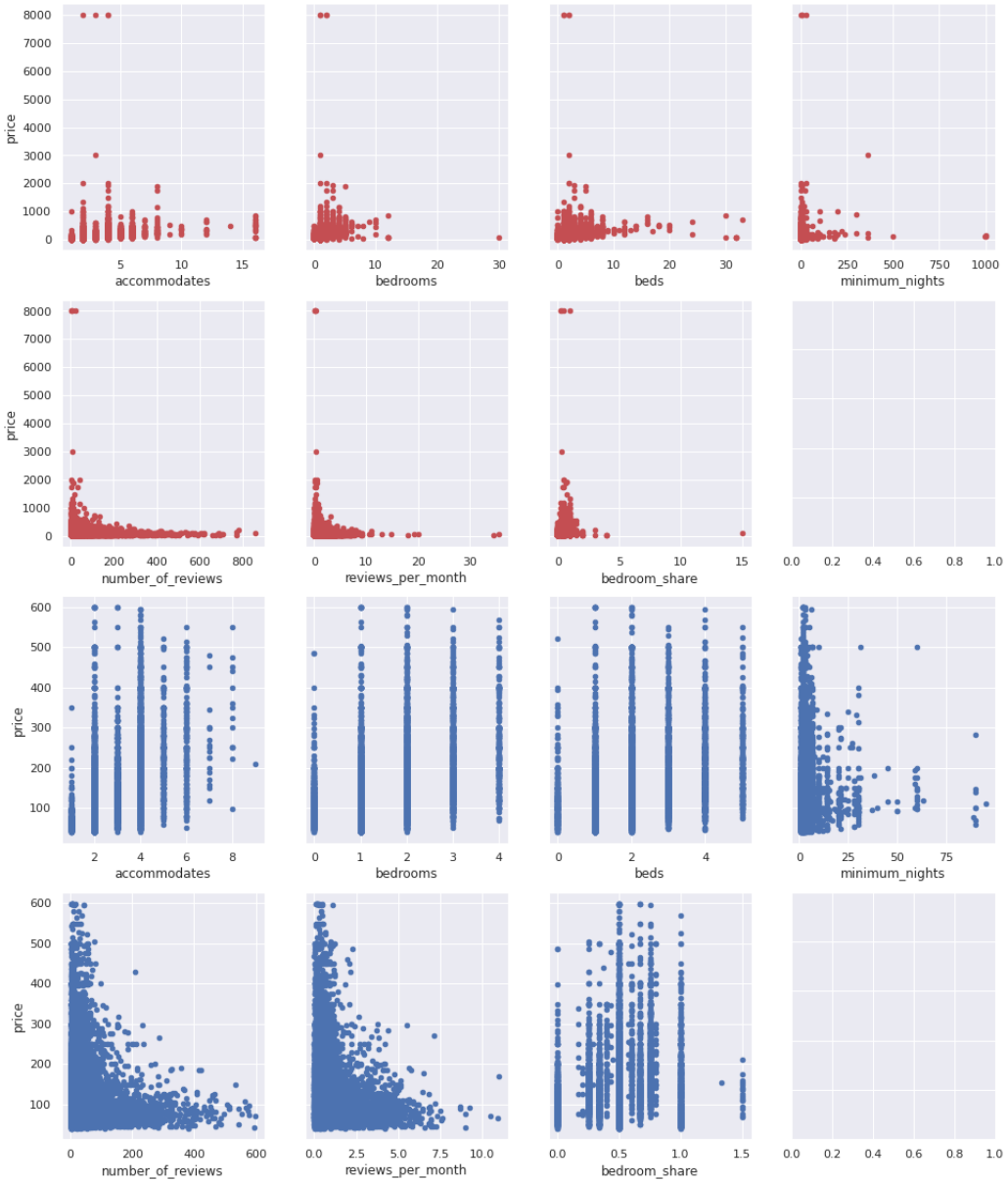



Price vs Number of Reviews vs Neighbourhood
Brooklyn
Centrum-Oost
Centrum-West
Zuid
De Baarsjes - Oud-West
Bos en Lommer
Westerpark
Oud-Oost
Noord-West
Slotervaart
Oud-Noord
Watergraafsmeer
Buitenveldert - Zuidas
Geuzenveld - Slotermeer
IJburg - Zeeburgereiland
Noord-Oost
Gaasperdam - Driemond
Osdorp
Bijlmer-Centrum
De Aker - Nieuw Sloten
Bijlmer-Oost
Number of Reviews
Price


Sentiment Polarity vs Price
Price
Positive
Negative
Neutral
Sentiment Polarity

## Appendix D: Outlier analysis



**Appendix E: Multicollinearity: Eigen vector results**

```
[270] #Eigen vector of a correlation matrix.
      multicollinearity, V=np.linalg.eig(corr)
      multicollinearity

      array([3.07760088, 1.8496101 , 0.03769868, 0.11739647, 0.24309122,
             0.58336157, 1.15814572, 0.87294272, 1.0629219 , 0.97669195,
             1.02053879])
```

**Appendix F: Neural Network model architecture**