1. Problem statement:

Airbnb is a home-sharing platform that allows property owners ('hosts') to put their spare room, apartment or home ('listings') online, so that guests can pay to stay in them in thousands of cities worldwide. Hosts are expected to set their own prices for their listings. Currently Airbnb and other sites provide some general guidance, but with this project we are trying to improve our base price services which help hosts price their properties using a wide range of data points.

Each year, Austin plays host to a large number of events such as ACL Festival and SXSW. We would expect that properties near downtown have higher asking prices based on the convenience of location to some of these events. Airbnb pricing is important to get right, particularly during these events where there is lots of competition and even small changes in prices can make a big difference. Price too high and no one will book. Price too low and you'll be missing out on a lot of potential income.

The client in this case would be Airbnb and they care about this problem because by suggesting the right rental price for each property, they can help their hosts to be profitable while at the same time helping their guests to get fair rental prices.
Instead of breaking down statistics by ZIP code, we want to generalize the sections of Austin for ease of discussion and analysis so we decided to break the city into five regions: Central (downtown and nearby areas), East, West, North, and South.

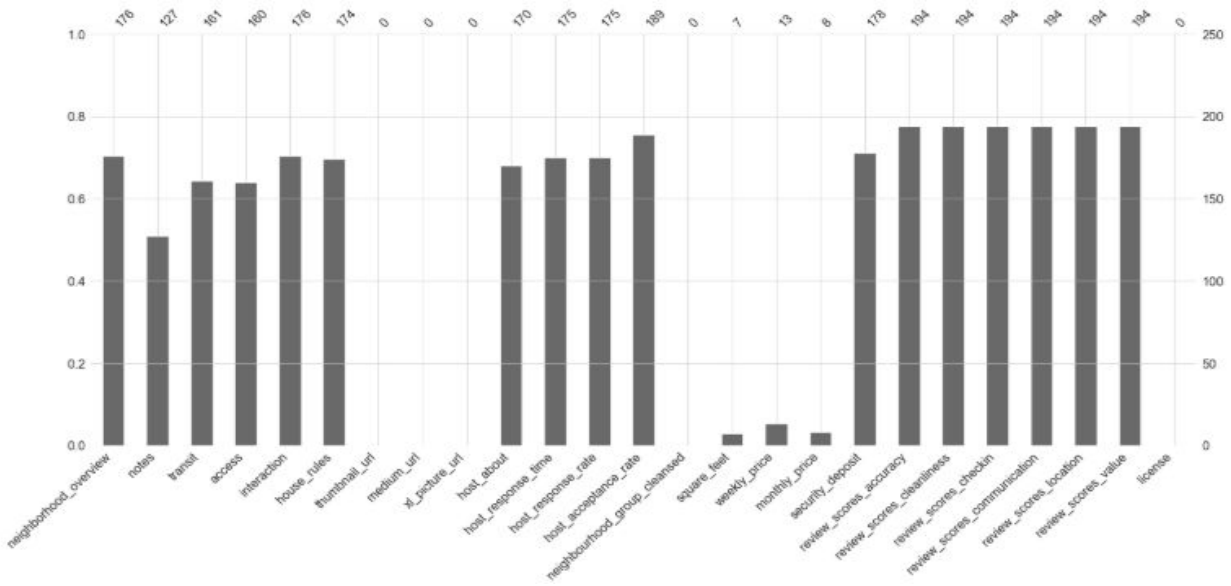
This project aims to predict the base price for properties in Austin by using machine learning.

2. Data:
The dataset that we are using in this project is listings.csv file which was compiled on March 17th, 2020 and was downloaded from http://insideairbnb.com/get-the-data.html for the city of Austin. The dataset contains 11,668 rows and 106 columns including the id column being used as the index.

First, we look at the top 25 columns with missing values to see if they add any value to our model or if we can drop them.

```
Columns with Null values by their count:

neighbourhood_group_cleansed     11668
medium_url                       11668
xl_picture_url                   11668
thumbnail_url                    11668
license                          11621
square_feet                      11476
monthly_price                    10931
weekly_price                     10793
notes                             6115
access                            4917
transit                           4370
host_about                        4099
interaction                       3862
house_rules                       3759
neighborhood_overview             3731
host_response_time                3462
host_response_rate                3462
security_deposit                  3100
host_acceptance_rate              2765
review_scores_location            2675
review_scores_value               2674
review_scores_checkin             2670
review_scores_communication       2670
review_scores_accuracy            2669
review_scores_cleanliness         2669
```



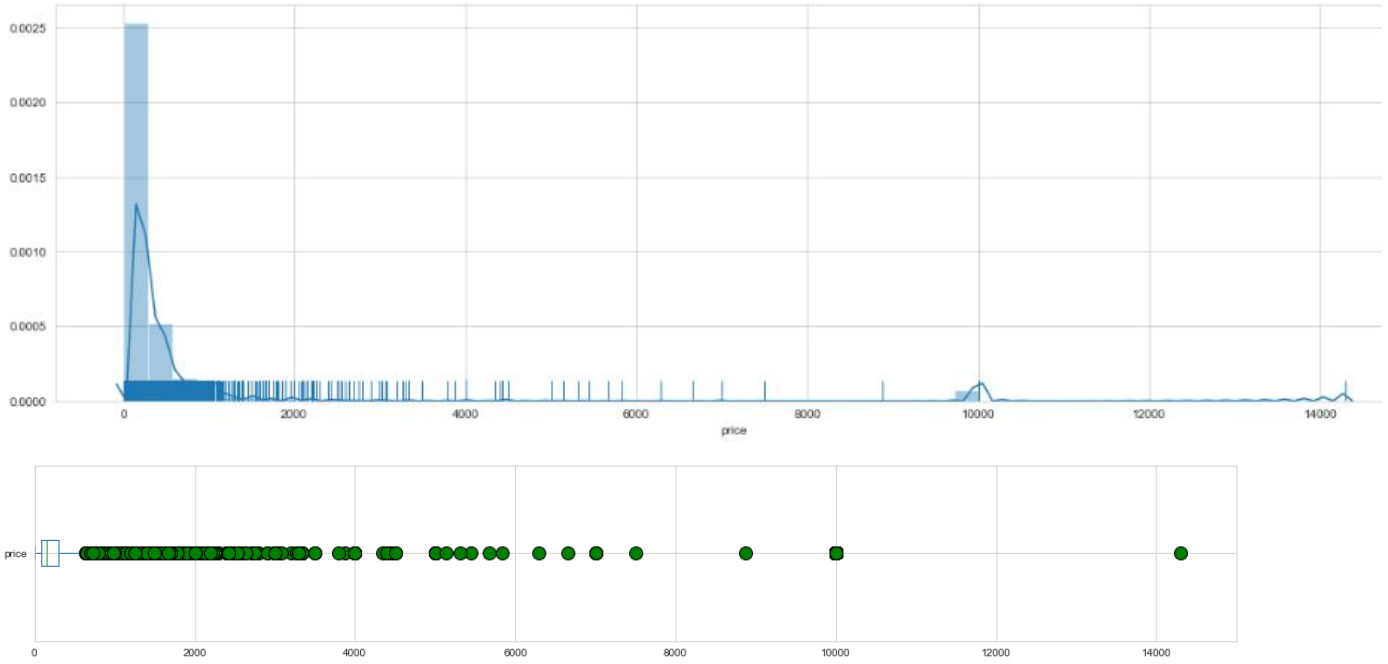
Features like 'square_feet', 'monthly_price' or 'weekly_price' have a lot of NULL values but we can use bedroom, bathroom, and accommodates data as a replacement for space size.

id	price	monthly_price	weekly_price
2265	$225.00	NaN	NaN
5245	$100.00	NaN	NaN
5456	$95.00	NaN	NaN
5769	$40.00	NaN	$160.00
6413	$99.00	$1,900.00	$700.00

Also use 'price' for our prediction instead of 'monthly_price' or 'weekly_price'. We can drop all of these columns since we have other columns that we can use as replacements.

After checking the descriptive statistics on the dataframe, we noticed that our target variable ‘price’ as well as 'cleaning_fee', 'security_deposit' and 'extra_people' costs are not included as part of the report. We did more investigation and noticed that they were an object data type. We then chained .str.replace().astype() methods to remove the ‘$’ and ‘,’ characters from these columns and also changed the data type to float.

Next, we decided to add a column called ‘Total_price’ which includes the sum of 'price', 'cleaning_fee', 'security_deposit' and 'extra_people' costs. After changing the data type, we looked at a few different plots like boxplot and distplot on the ‘price’ column and noticed that its distribution is not normal.



As we can see on the figures above, there are some outliers above $8,000 that we need to look at. There also seems to be a few prices below $15 that we need to check but the good news is that there are no null values in the ‘price’ column.

We start looking at properties where the ‘price’ is set to 0 and we notice that these are just a few incorrect values and we fill them by getting the median price based on the ‘accommodates’ rate for each of these properties.

We go up to the next level price which is set to $1 and we notice that all of these properties are for Hurricane Harvey Refugees and they haven't been updated since 2017 so we can drop them.

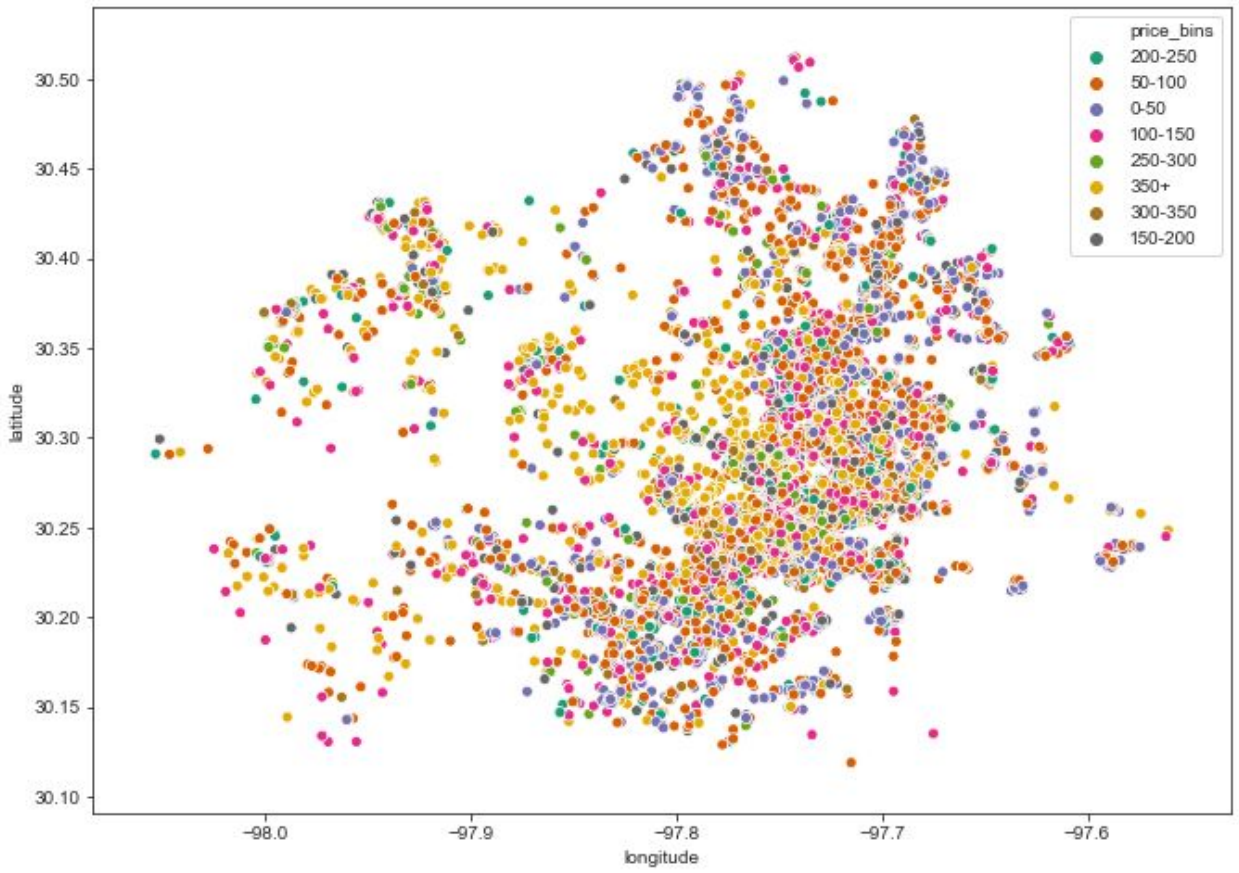
The next few levels for prices up to $16 can be dropped as these are camp site rentals and they do not add any value to solving our problem.

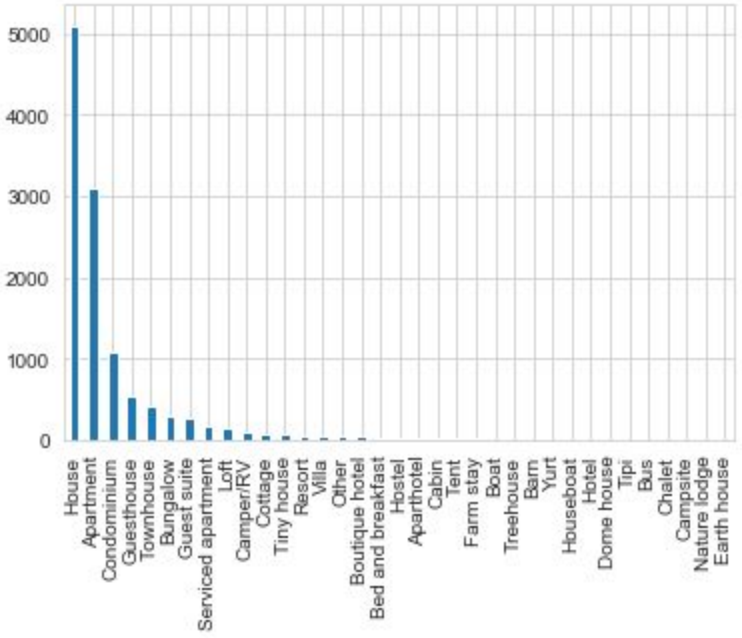
```
1 df.price.describe()
```

```
count    11641.000000
mean       485.390688
std       1481.881108
min         16.000000
25%         80.000000
50%        149.000000
75%        300.000000
max      14298.000000
```

We ran descriptive statistics on the price column and now the minimum is set to $16 and the maximum is $14,298!

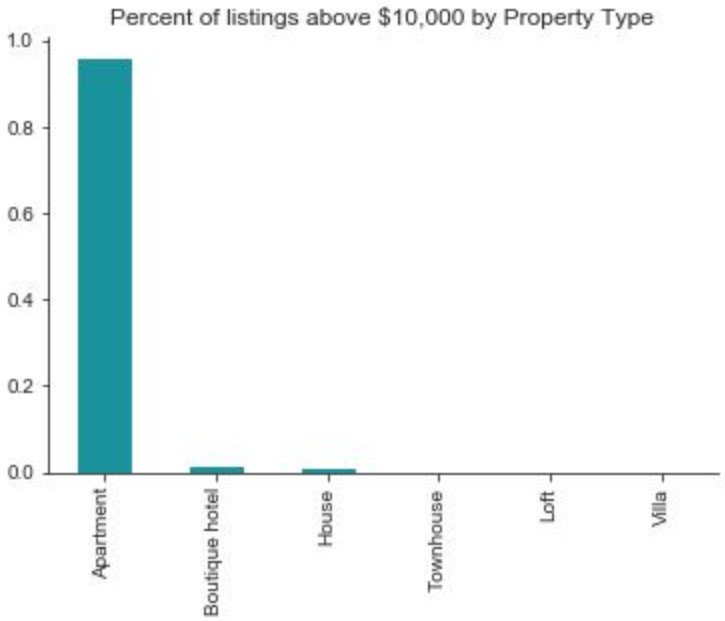
In order to do some more visualization on the ‘price’ column, we break it into 8 bins with each bin incrementing by $50. Then we use this new ‘price_bins’ column as the value for hue in the scatterplot and by doing this, we can see that most of the properties above the 350 price range are located on the West side of Austin.





House	0.437935
Apartment	0.266901
Condominium	0.091573
Guesthouse	0.046216
Townhouse	0.035736
Bungalow	0.025599
Guest suite	0.023194
Serviced apartment	0.014260
Loft	0.011855

Grouping properties by every unique type shows us that 44% of the properties are Houses, 27% are Apartments, 9% are Condominiums, 5% are Guesthouses and 3% areTownhouse. The rest of the property types hold lower than 3% counts and we can group them together later on to help our model with only keeping the top 90% values and the rest be set to ‘Other’.



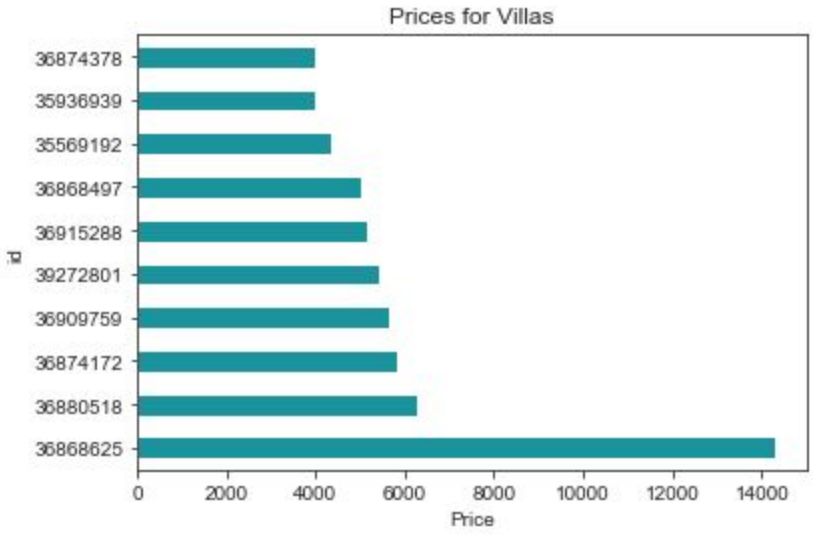
Apartment	0.960159
Boutique hotel	0.015936
House	0.011952
Townhouse	0.003984
Loft	0.003984
Villa	0.003984

We look at the unique property types above $10,000 per night to see if we can find which property types the outliers are related to. By looking at the chart above, we see that 96% of these prices are for Apartment property type. We need to look at random Apartment listings using the listing_url to confirm if the prices are actually above $10,000 for these listings.

id	bedrooms	bathrooms	accommodates	price	listing_url
37997840	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/37997840
37998870	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/37998870
41718441	4.0	2.0	10	10000.0	https://www.airbnb.com/rooms/41718441
39560085	2.0	1.0	6	10000.0	https://www.airbnb.com/rooms/39560085
39580699	2.0	2.0	6	10000.0	https://www.airbnb.com/rooms/39580699
...	...	...	...	...	...
38214597	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38214597
38212404	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38212404
42455201	0.0	1.0	2	10000.0	https://www.airbnb.com/rooms/42455201
38209274	1.0	1.0	2	10000.0	https://www.airbnb.com/rooms/38209274
42454474	0.0	1.0	2	10000.0	https://www.airbnb.com/rooms/42454474

Next, we check the price of apartments that are more than $10,000 per night. We manually checked some of these listings randomly and only 1 of them has a price above $10,000/night. We will replace these Apartment prices with the median price associated with the accommodates rate for each apartment.

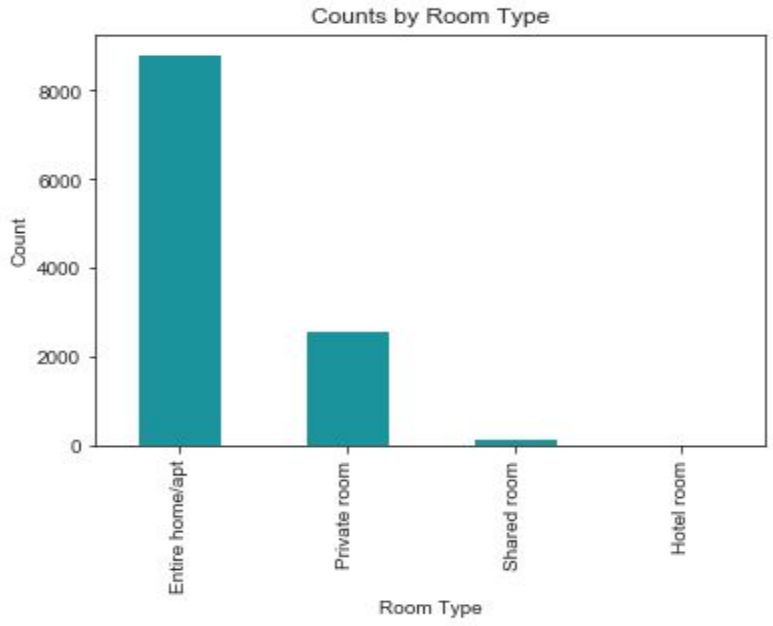
The max price is $14,298 and we couldn't find this price in Apartments. The next property type that we should check is Villa. These properties are larger houses, having an estate and consisting of farm and residential buildings arranged around a courtyard and it would make sense if they have higher rental prices.



id	
36868625	14298.0
36880518	6291.0
36874172	5832.0
36909759	5669.0
39272801	5443.0

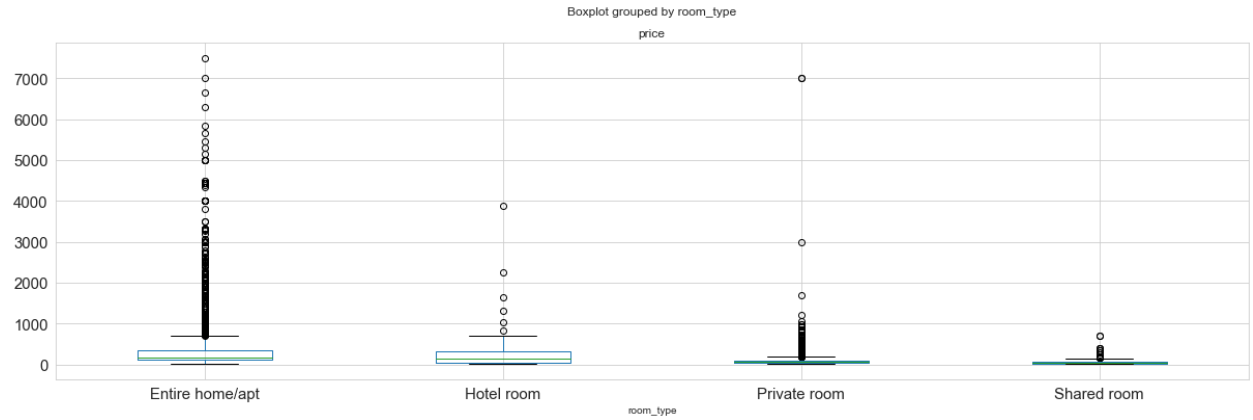
We see that our $14,298 per night property is listed under Villa property type. The median price for Villas is $1,344 and we could use that value to replace these outliers but since Villas are not in the top 90% property types, they will be included in the 'Other' property type at the end.

Using a mask for all the properties with prices above $8,000, prices are set to NaN values and then filled with the median price for each property type.

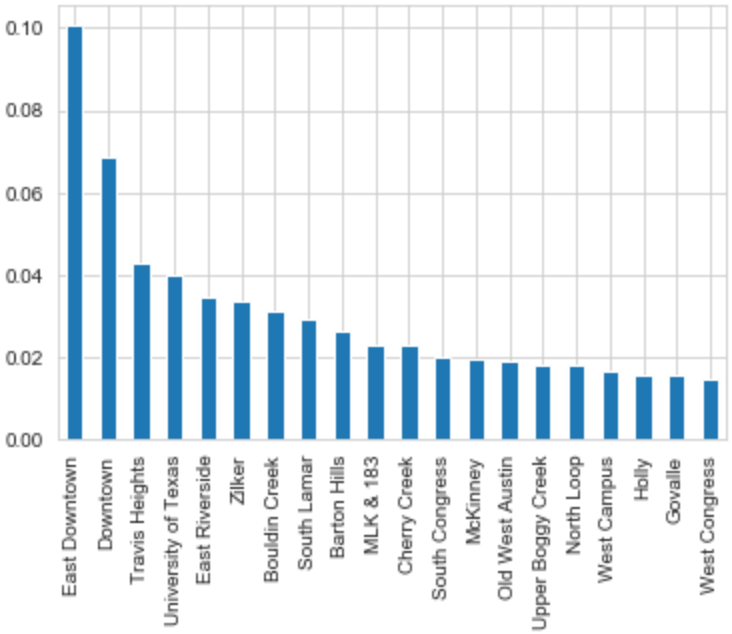


Entire home/apt	0.758698
Private room	0.223349
Shared room	0.013573
Hotel room	0.004381

Looking at the room types, we see that 76% of them are either the entire home or the entire apartment.

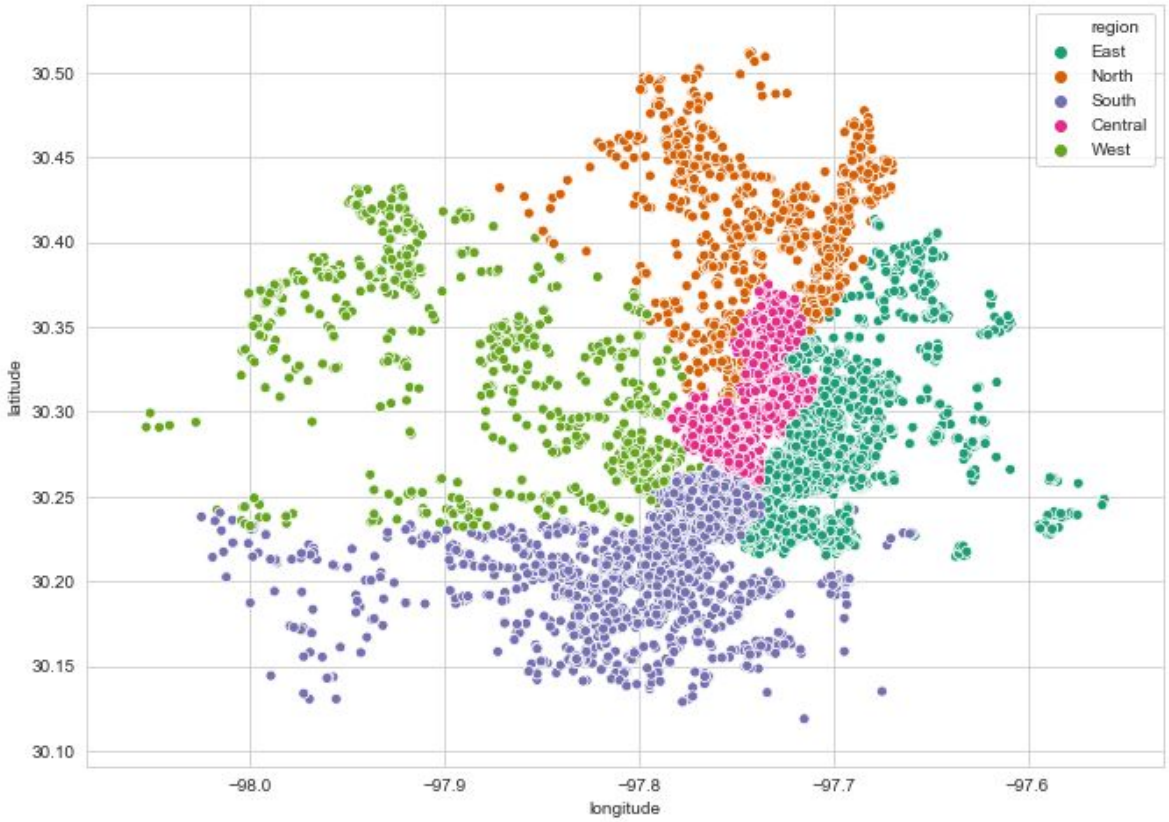


Using boxplot above, shows us that most of the higher rental prices are for Entire home/apt but there are a few high rental prices for Hotel room and Private room types that needs to be checked.

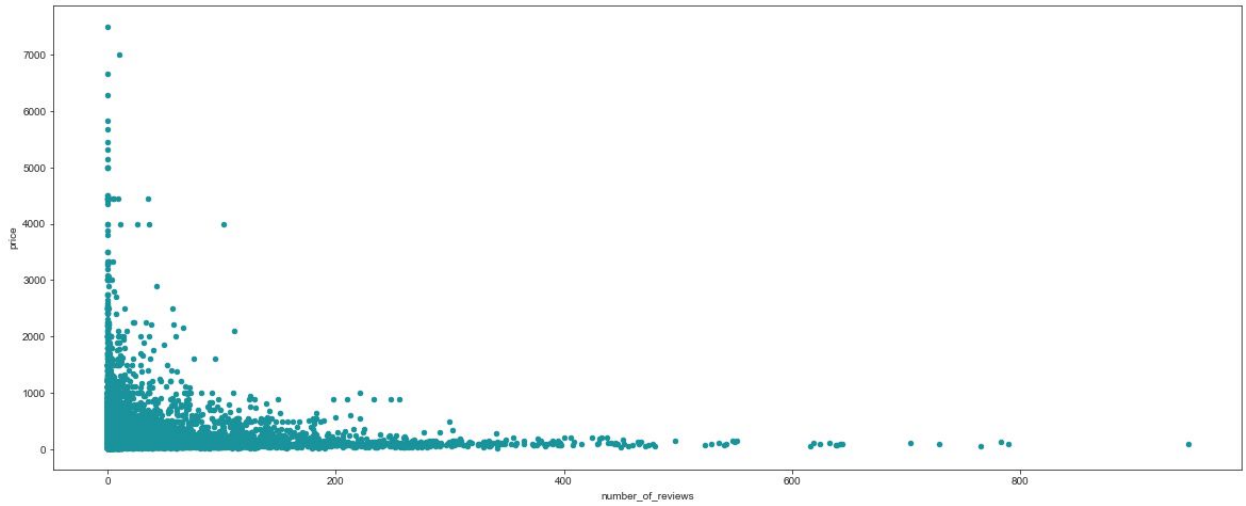


Looking at different neighbourhoods, we can see that 10% of the properties are located in the East Downtown neighbourhood, 7% in Downtown, 4% in Travis Heights and University of Texas each.

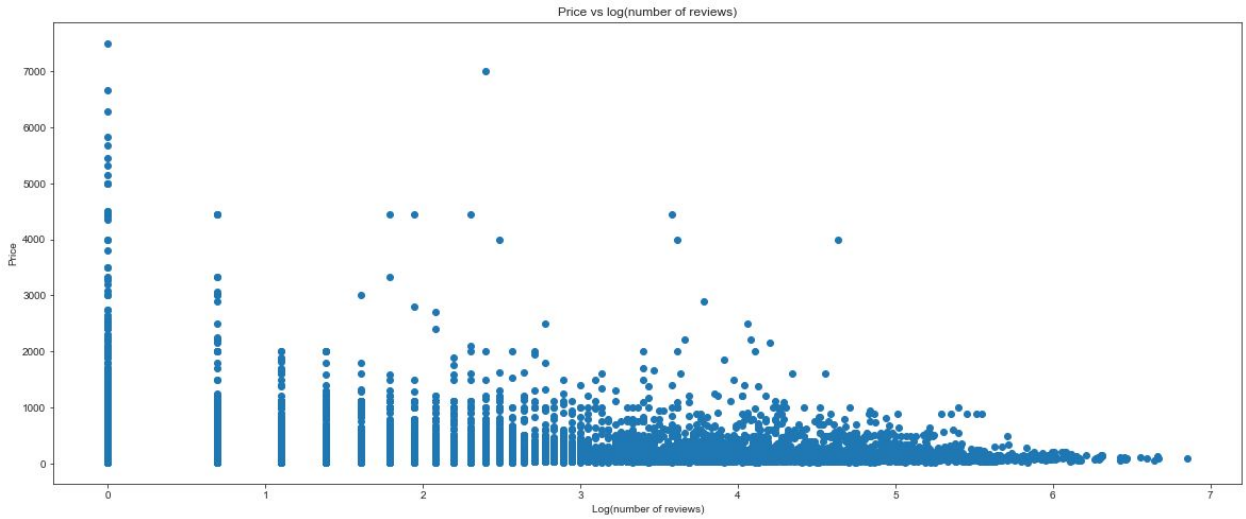
Next, we divide all the zip codes into four different regions. This will help us with some visualizations and analysis.



Looking at the scatter plots above, we can see that the most expensive rentals are in the West region. This makes sense because this region has the most count of Villas and properties are bigger as well. They also have the highest minimum nights in their listings.

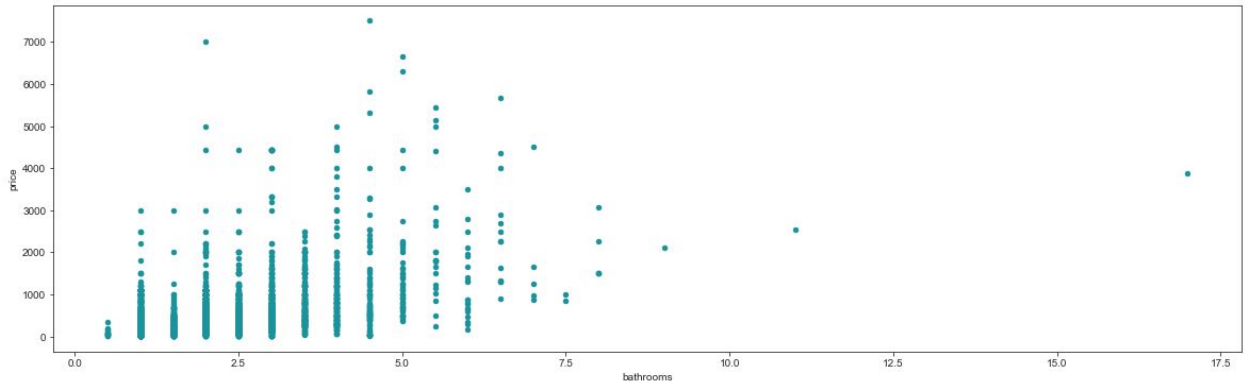


Looking at the number of reviews, we can see that some high rental prices, either don't have a review or the number of reviews are small. These properties might be good candidates for dropping later.

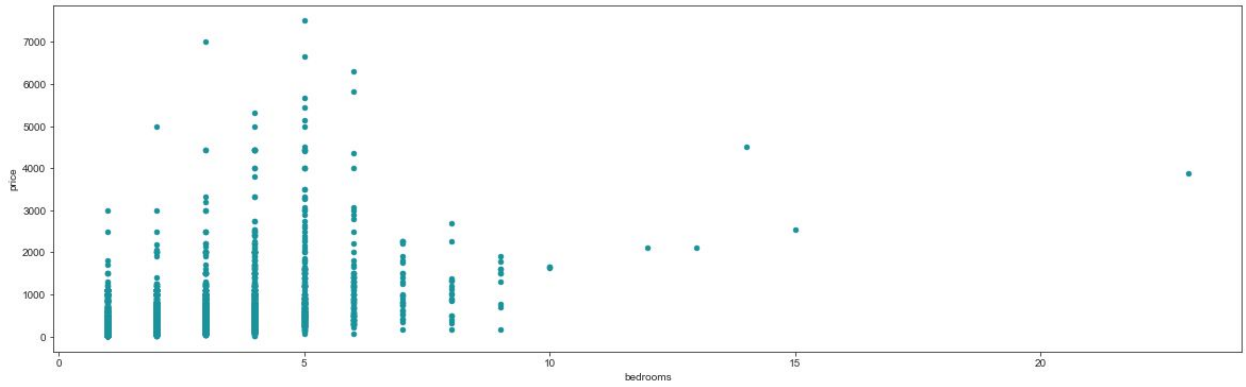


Looking at the log of the number of reviews, we can see in more detail that properties with zero reviews have high prices ranging above $7,000!

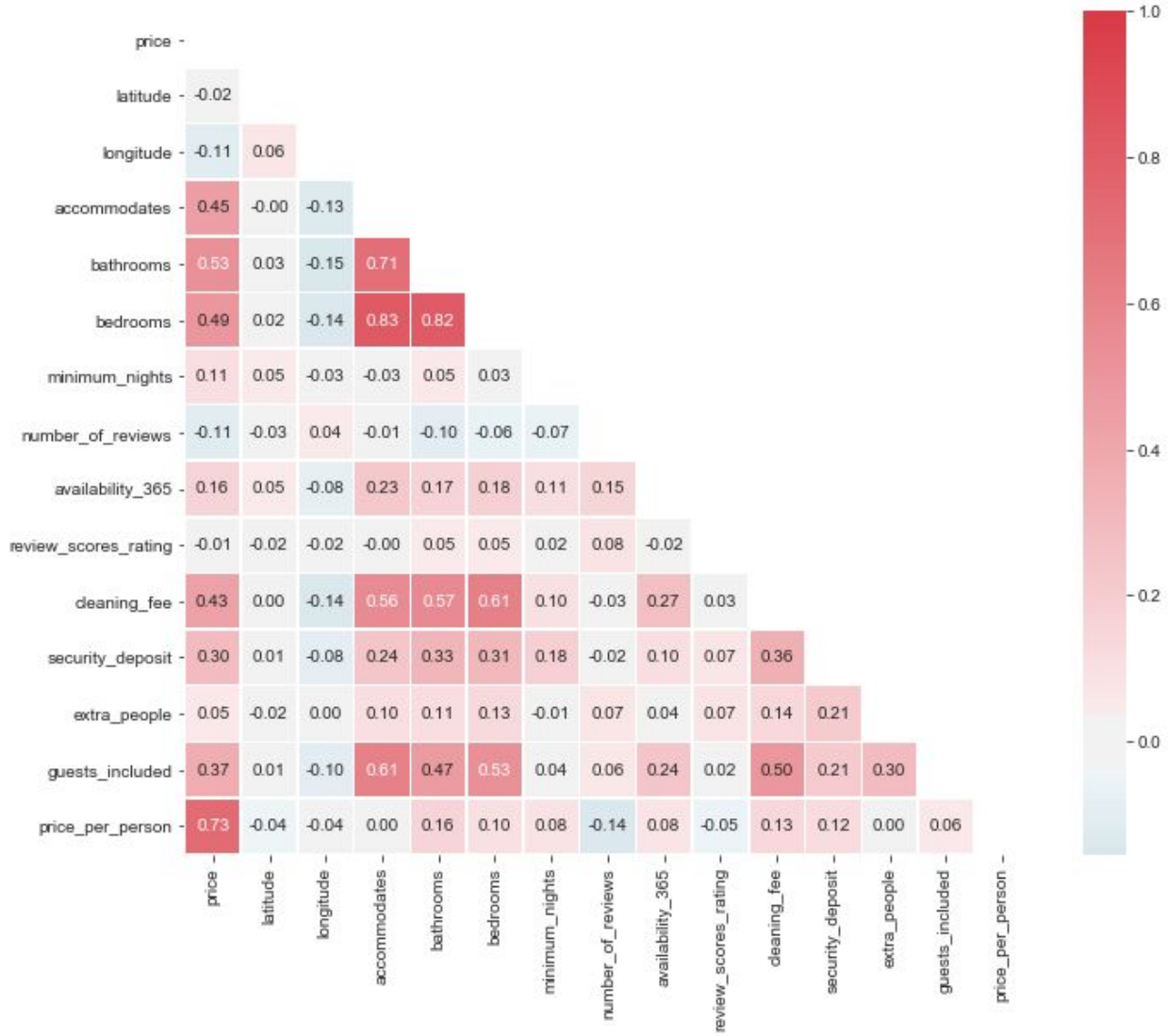
We look at the bathrooms, bedrooms and accommodates columns next. The bathrooms column has 13 missing values as well as 33 values set to zero. The bedrooms column has 15 missing values and 706 values set to 0. We set these 0 values to NaN values and then use the accommodates column to fill them with the median value.



Looking at the chart above, properties with 4 bathrooms or more have higher rental prices!



Looking at the chart above, properties with 4 or more bedrooms have higher rental prices!



The correlation matrix above shows that price is correlated with accommodates, bathrooms, bedrooms, cleaning_fee, security_deposit and guests_included. Bedrooms are also correlated with accommodates and bathrooms.

After replacing all the NaN values and dropping some observations, we now have 11,608 observations and 25 features. We save the file to our data folder so it can be used for our Machine Learning steps.