Basics of Data Science Course - Assignment 1

Soroush heidary

96222031

# Data Set : NYC Listings

Summary :

Beside the main 4 questions asked, I added 2 more assumptions to be tested.

- Is there any significant relationship between the points of interest in NYC and the features we had
- Is there any kind of NPL-techniques which could be applied to 'name' feature so that we could extract some meaningful information out of it

I could not manage my time properly to have all the assumptions and questions statistically tested and there's no statistical tests provided in the report, sorry :D

And don't mind the page amount, it's mostly pictures :D

General Data Cleaning :

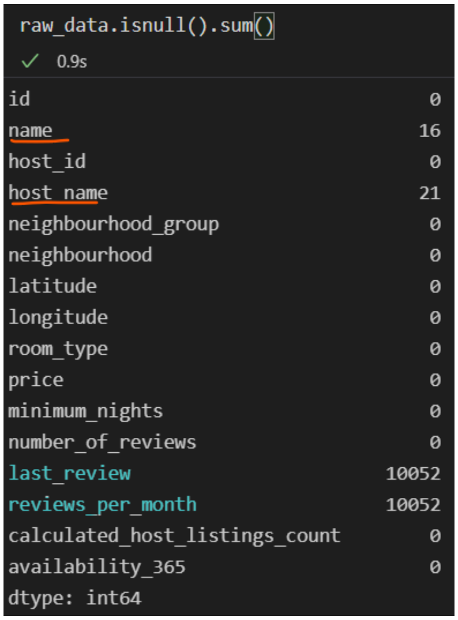
- Nulls values : as you can see some of our features has null values, for each we'll have a different approach:

  Name : this features null values basically means nothing so we can fill them with some value like 'Unknown'

  Host_name : just like the Name feature

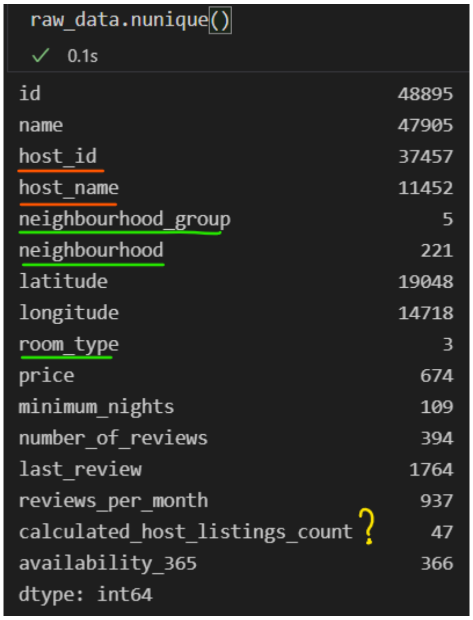
  Last_review : obviously when we don't have a last review date it means there were none (I did check the dataframe just to be sure, those with 0 number of reviews had this feature set to Nan) so we fill them by some date like (1900/01/01)

  Review_per_month : we simply set them to 0



```
raw_data.isnull().sum()
✓ 0.9s
id                                   0
name                                16
host_id                              0
host_name                           21
neighbourhood_group                  0
neighbourhood                        0
latitude                             0
longitude                            0
room_type                            0
price                                0
minimum_nights                       0
number_of_reviews                    0
last_review                      10052
reviews_per_month                10052
calculated_host_listings_count       0
availability_365                     0
dtype: int64
```

- Dtypes and Uniques: some of the features we see are actually utterly useless when it comes to predicting anything.
  Host_id and Host_name do have a looot of unique values and training any model on them would just lead to overfitting, same with 'id' we already have an indexing column, hence we won't need this one (although oddly this 'id' feature has a questionable correlation to the price, some value around 0.4 which is simply weird and way above my paygrade to delve into the whys and hows of that)
  As for other features we'll deal with them later on (some binning needs to be applied to some numerical features, and some one-hot-vectoring/label-encoding on the categorical features



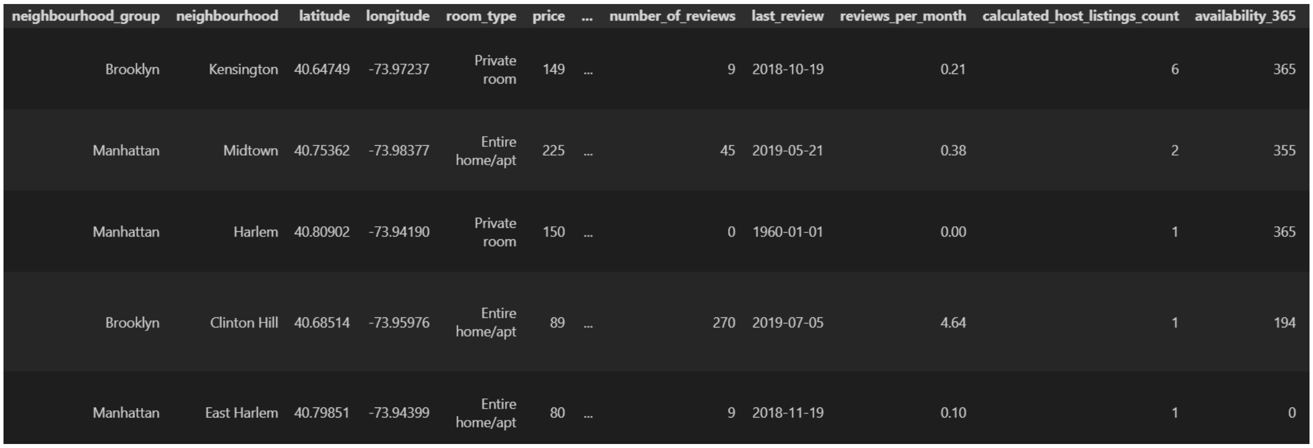
```
raw_data.nunique()
✓ 0.1s
id                                48895
name                              47905
host_id                           37457
host_name                         11452
neighbourhood_group                   5
neighbourhood                       221
latitude                          19048
longitude                         14718
room_type                             3
price                               674
minimum_nights                      109
number_of_reviews                   394
last_review                        1764
reviews_per_month                   937
calculated_host_listings_count ?     47
availability_365                    366
dtype: int64
```

Q1 – What can we learn about different hosts and Areas.

Quite a lot actually!

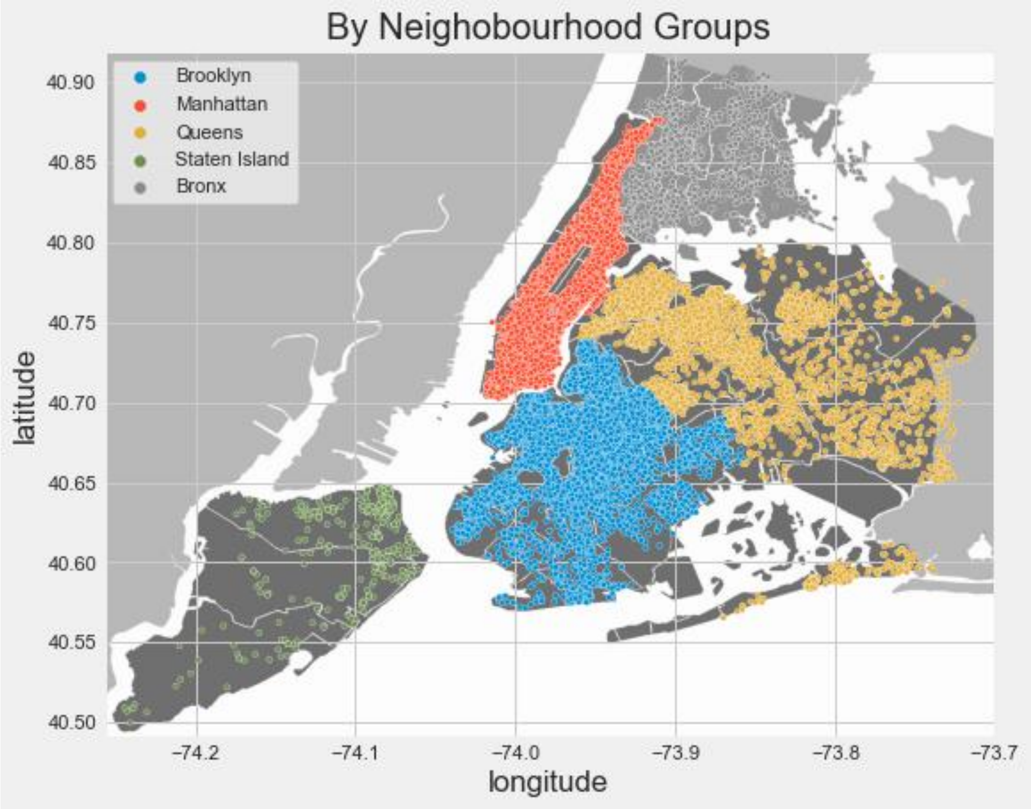
Features like density or popularity of different places, etc .. can be derived

Lets have a general look at our data

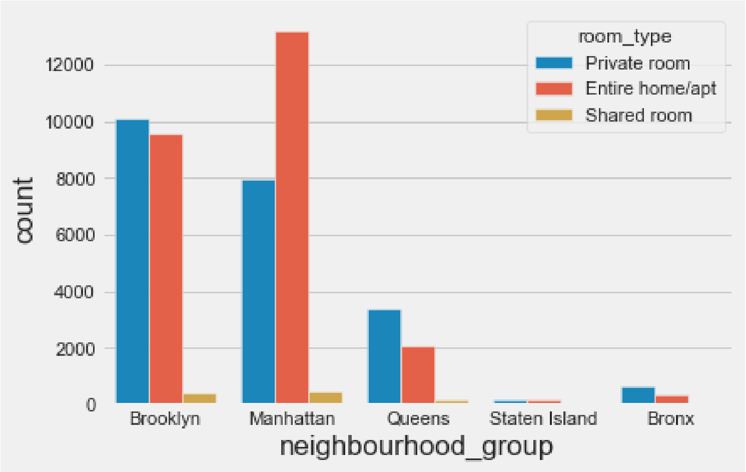


| neighbourhood_group | neighbourhood | latitude | longitude | room_type | price | ... | number_of_reviews | last_review | reviews_per_month | calculated_host_listings_count | availability_365 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Brooklyn | Kensington | 40.64749 | -73.97237 | Private room | 149 | ... | 9 | 2018-10-19 | 0.21 | 6 | 365 |
| Manhattan | Midtown | 40.75362 | -73.98377 | Entire home/apt | 225 | ... | 45 | 2019-05-21 | 0.38 | 2 | 355 |
| Manhattan | Harlem | 40.80902 | -73.94190 | Private room | 150 | ... | 0 | 1960-01-01 | 0.00 | 1 | 365 |
| Brooklyn | Clinton Hill | 40.68514 | -73.95976 | Entire home/apt | 89 | ... | 270 | 2019-07-05 | 4.64 | 1 | 194 |
| Manhattan | East Harlem | 40.79851 | -73.94399 | Entire home/apt | 80 | ... | 9 | 2018-11-19 | 0.10 | 1 | 0 |

Knowing that we have only 5 unique neighborhood groups, we'll start by exploring this feature, we will plot all the data using a scatterplot on the map using lats and lons, we'll give the hue according to their group and use the NYC map (which feels cool :D) as the background of our plot



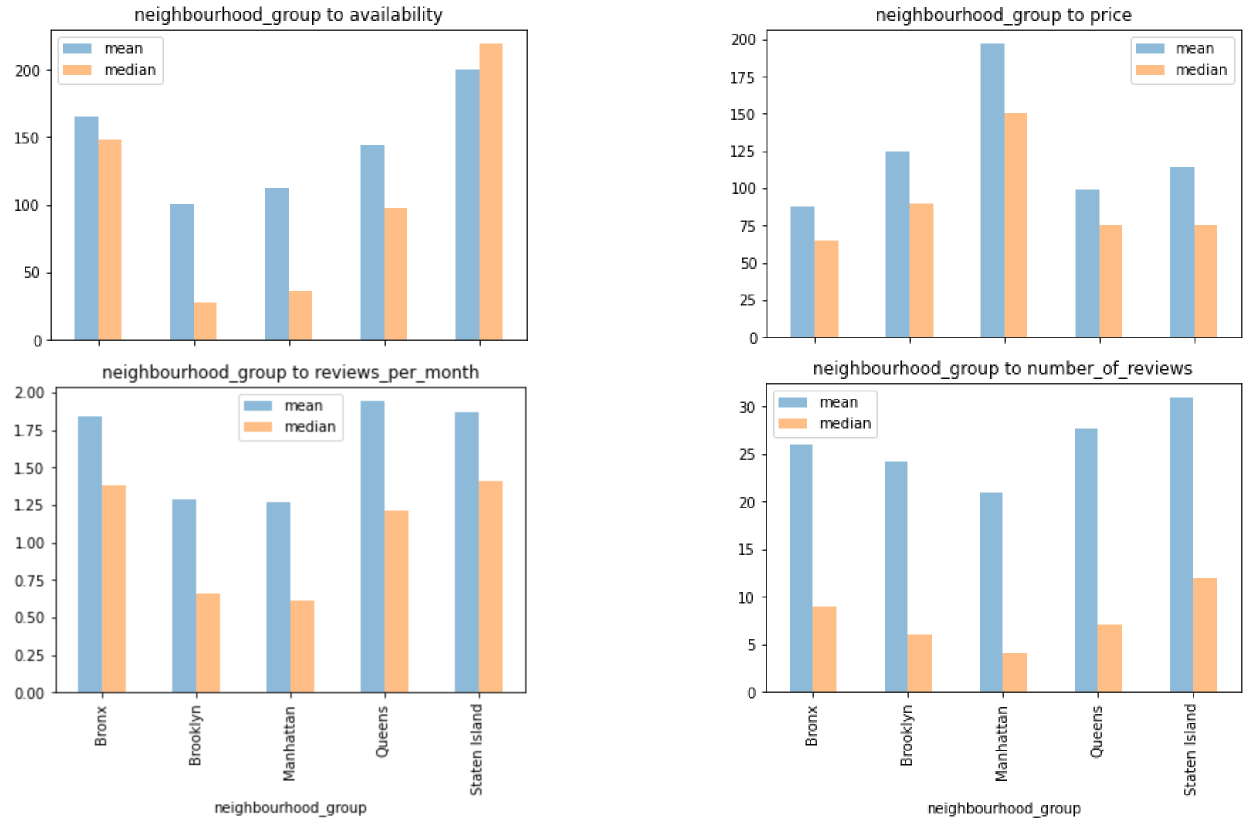
It seems that a lot of our data is taking place on Manhattan and Brooklyn, to test this, we'll have a bar plot, we'll also give it a hue based on the room_type since the room_type has only 3 features



seems about right! Also shared_rooms are not a popular choice around NYC it seems!

Well assumingly some kind of distinguishment is there to be seen based on prices, availability, etc in each group, we'll test it by using bar plots

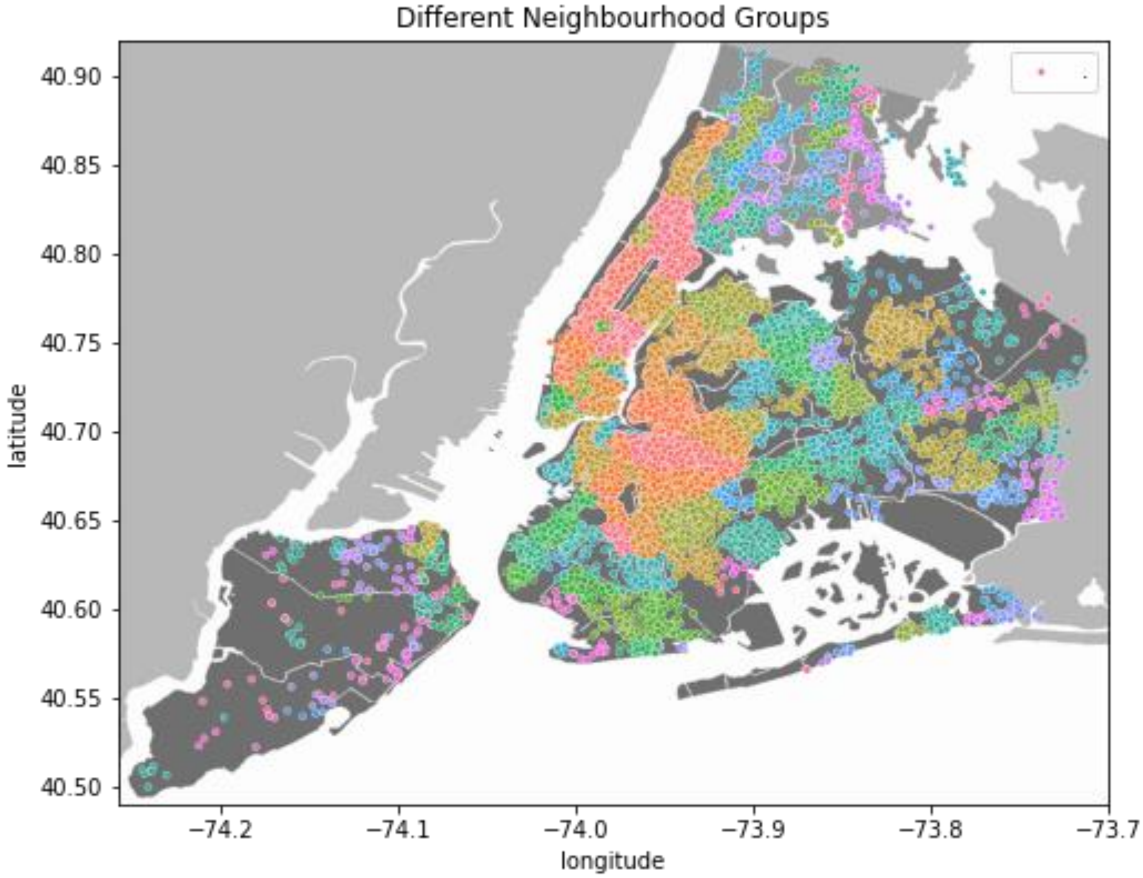
These bar plots have been created by using a groupby method in Pandas and then applying a mean aggregation on the grouped data (columns share the x axis)



While Manhattan has the highest prices, Staten Island has the highest availability (maybe they should rethink their prices though …), and Brooklyn and Manhattan seem to be the most popular for tourists

Now we'll try to see if there's any meaning in the neighborhood feature.

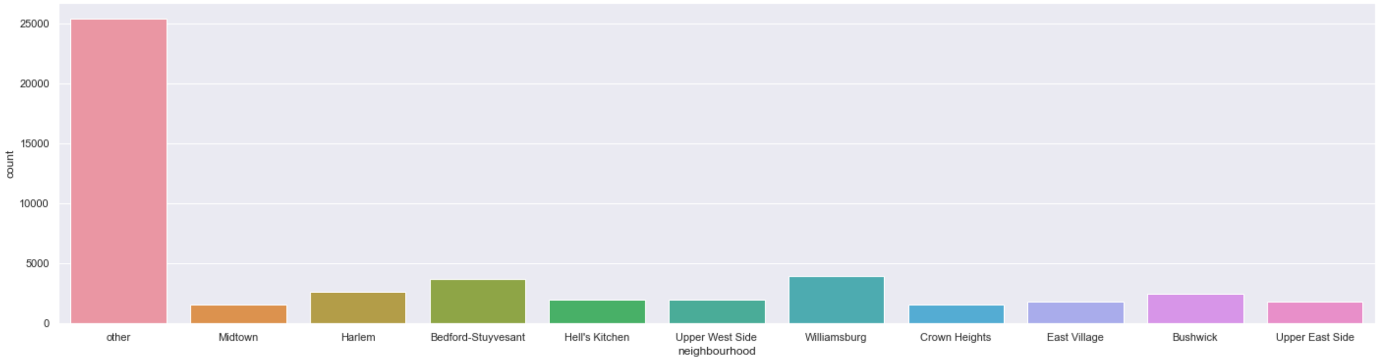
As we saw above, there is a lot of variety in this column. Just to demonstrate the variety we can look at the plot below



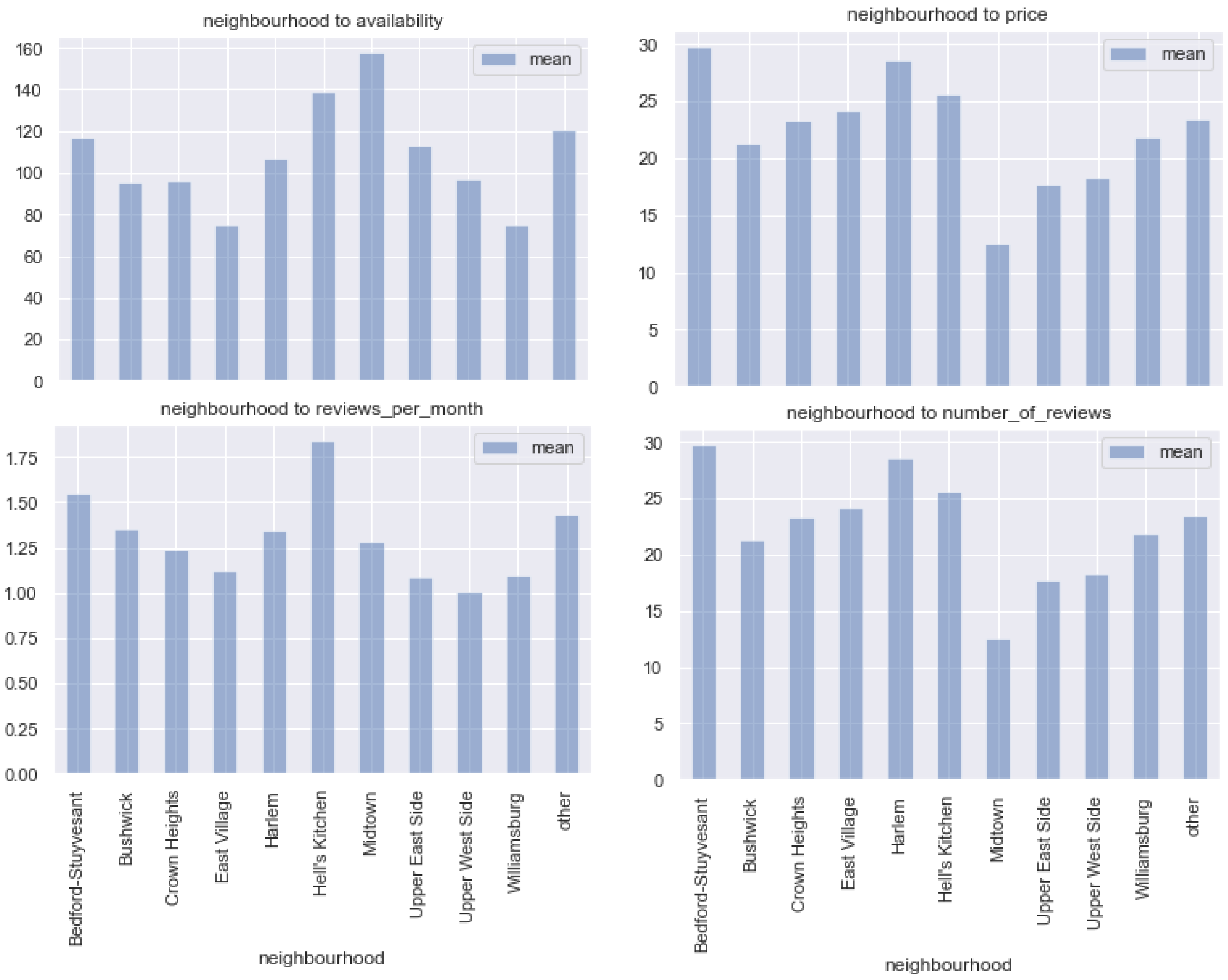
For one thing; there not enough damn colors to even show the variety!

So we'll try to lessen the unique one based on their value counts into its top 10 (11 with Other)

The rest we'll just get named 'Other', below Is a chart of value counts



Below you can see the aggregations applied to these types to see whether or not there relies some connection to price, availability and reviews



Well not that there’s no difference, there’s no distinguishable difference

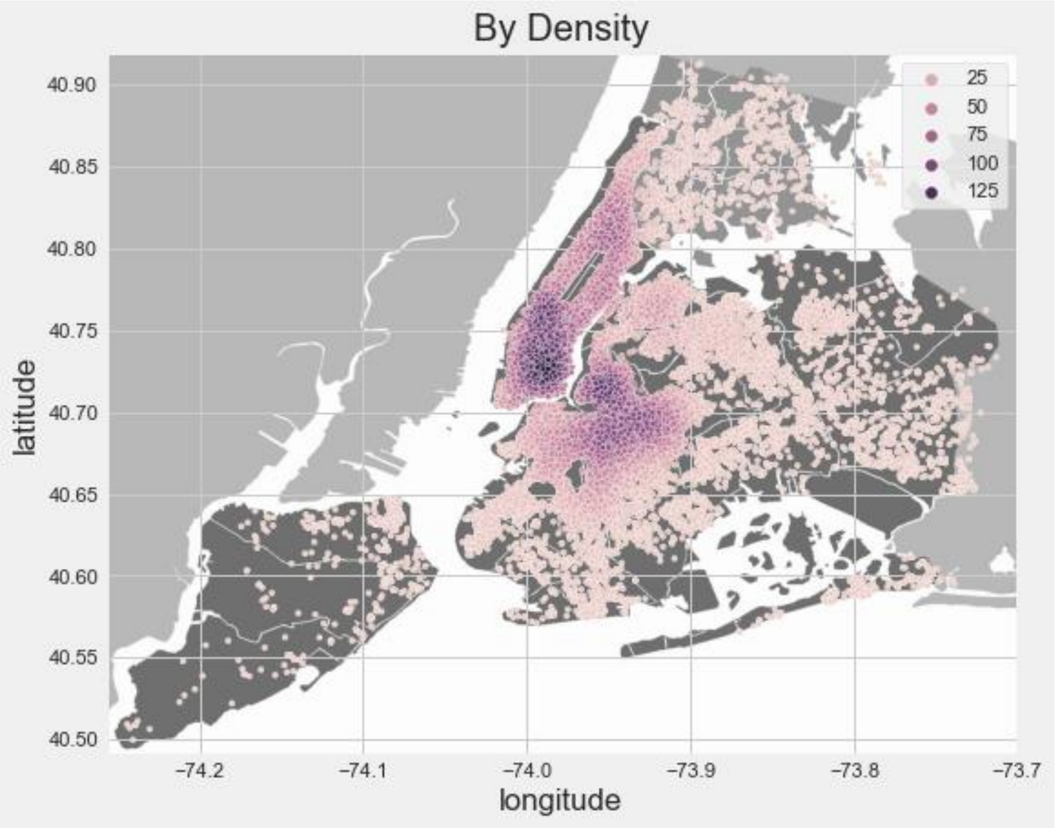
Though there is a quite usefull feature can be derived the coordinates, and that’s the density of how the hotels are placed.

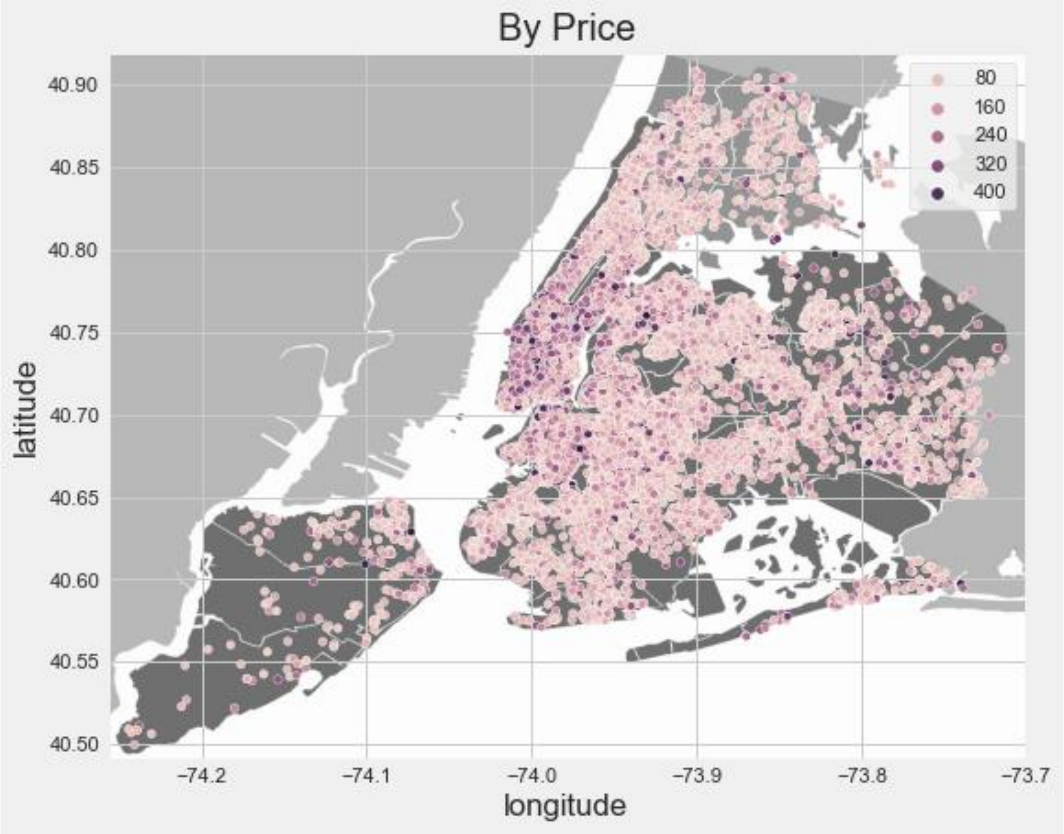
In a chart below you can see the density of each data, for aquiring this value a GaussianKDE is used

Also after that we’ll see a chart showing each data with the hue based on It’s price, the more thick the color, the more costly the listing would be … (for this chart I had to use not all the data but those who had their price in this range :

price < price.mean()*3

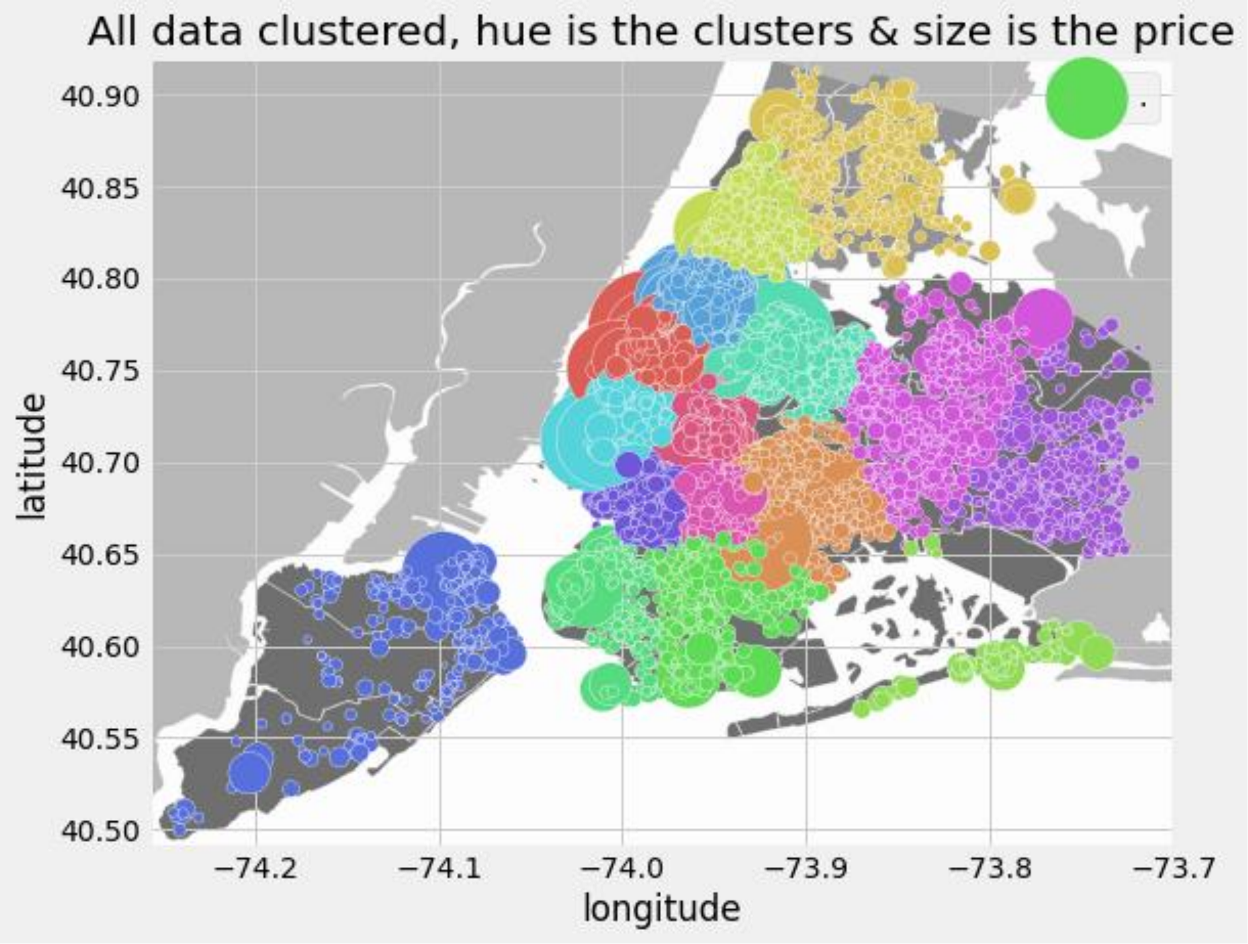
because there were some outlier that made the hue almost impossible to ditinguish


By Density


By Price

There’s something else we can try with this question.

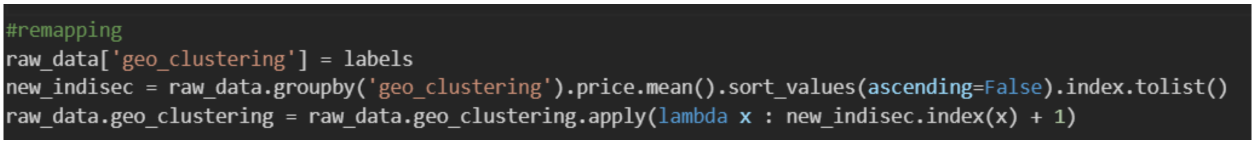
We’ll cluster the data with K-Means clustering technique to see if it could correlate to the price



The bigger the sizes, the more costly the place is, I used different paramethers for the K-Means clustering but the best correlation (and the one who wouldn’t lead to overfitting) were proved to be around 12 – 16 clusters and a correlation of 0.22 were recorded when comparing the cluster labels and prices of each data.

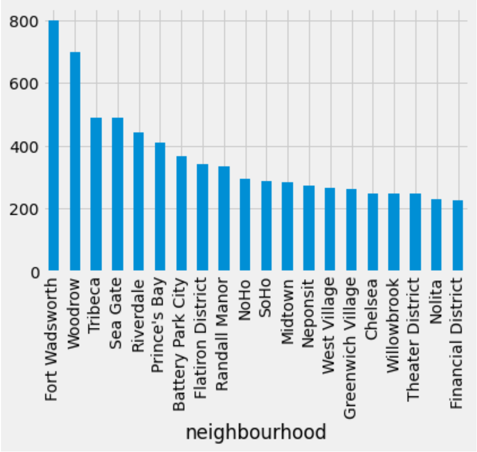
(for this purpose the clusters had to be sorted in such a way that cluster with label of 1 has the highest mean of price in it.)

This is the code I wrote for that purpose (labels are the cluster label of each data)



```
#remapping
raw_data['geo_clustering'] = labels
new_indisec = raw_data.groupby('geo_clustering').price.mean().sort_values(ascending=False).index.tolist()
raw_data.geo_clustering = raw_data.geo_clustering.apply(lambda x : new_indisec.index(x) + 1)
```

Before we move on to question 2, this picture shows the mean price of each neighbourhood after we reduced it's dimention



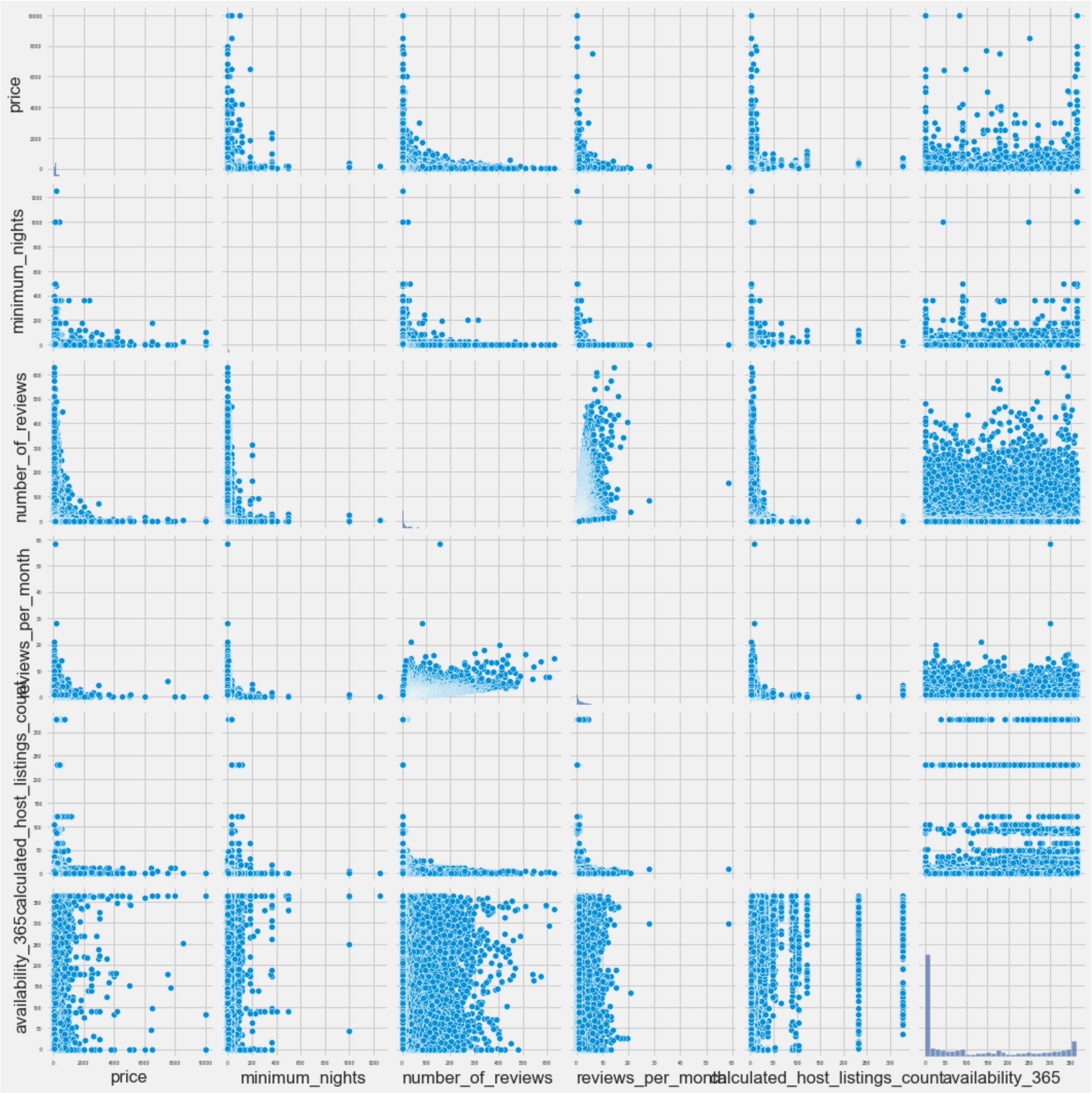
Upper West
East Side
Hell Kitchen
Bedford Stuyvesant
Crown Heights
West Side
Washington Heights
Upper East
East Village
Financial District
Island City
Midtown
Greenpoint
Clinton Hill
Long Island
Harlem
Park Slope
Sunset Park
Murray Hill
Lower East
East Flatbush
Prospect Lefferts
Fort Greene
Lefferts Gardens
Astoria
Bushwick
Flushing
West Village
Chelsea
Flatbush
Kips Bay
East Harlem
Chinatown
Ridgewood
Stuyvesant Bedford
Greenwich Village
Williamsburg
Prospect Heights
SoHo

And the wordcloud you see shows which of the neighbourhood before dimention reduction were the most popular according to their listings, not something really usefull I know, but it was damn pretty to look at, I also copied the code for the wordcloud from a webpage which I lost the link to

Q2 – What can we learn from predictions.

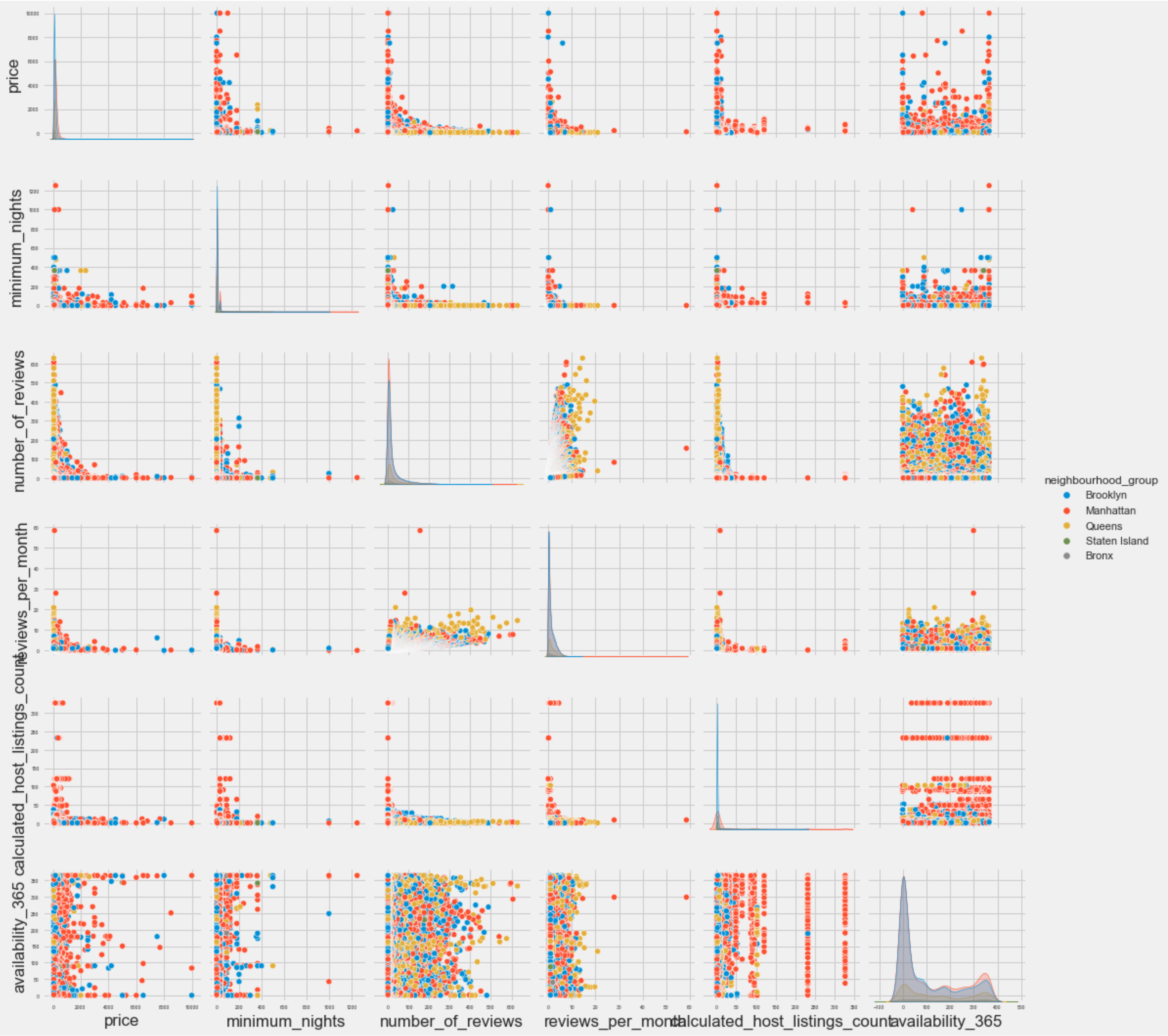
Well the question is sooo general that only a general answer could be a worthy opponent :

So here's some pair plots for you yall



Which does mean something, at the very least you could see a clear regression line in the plot of reviews_pre_month vs number_of_reviews, other than that, nothing much obvious is grathered from the paiplot

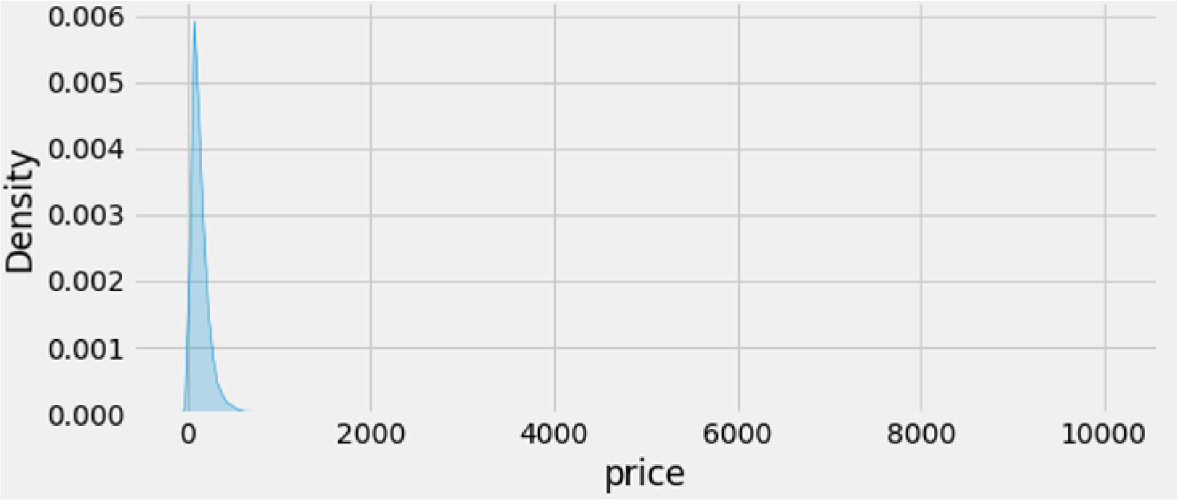
And here's a hued version of this plot just for the sake of more generality …



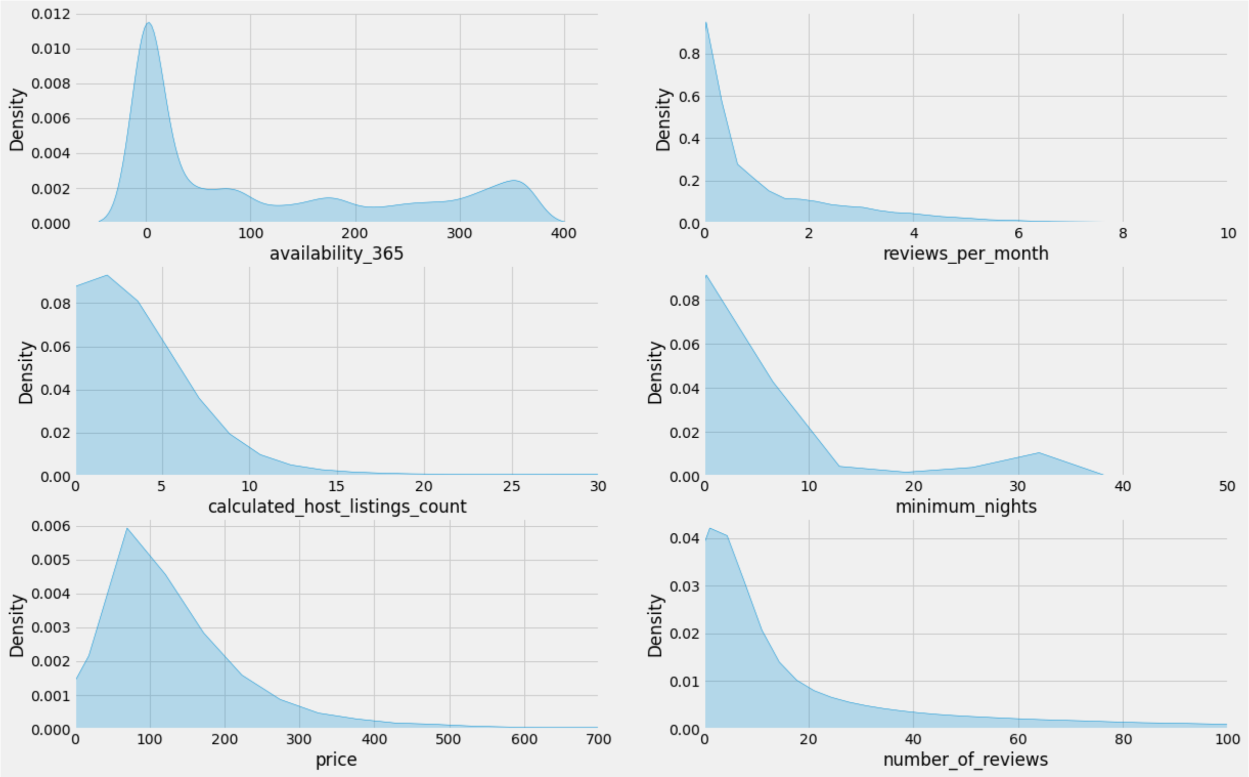
But all jokes aside, for any kind of predictions to be made from this data set we need to deal with the data itself, like I mentioned before, some categorical features have to be one hot vectored others either dropped or label encoding applied to them

And as for the numerical features, it's probably better to apply some binning process on them

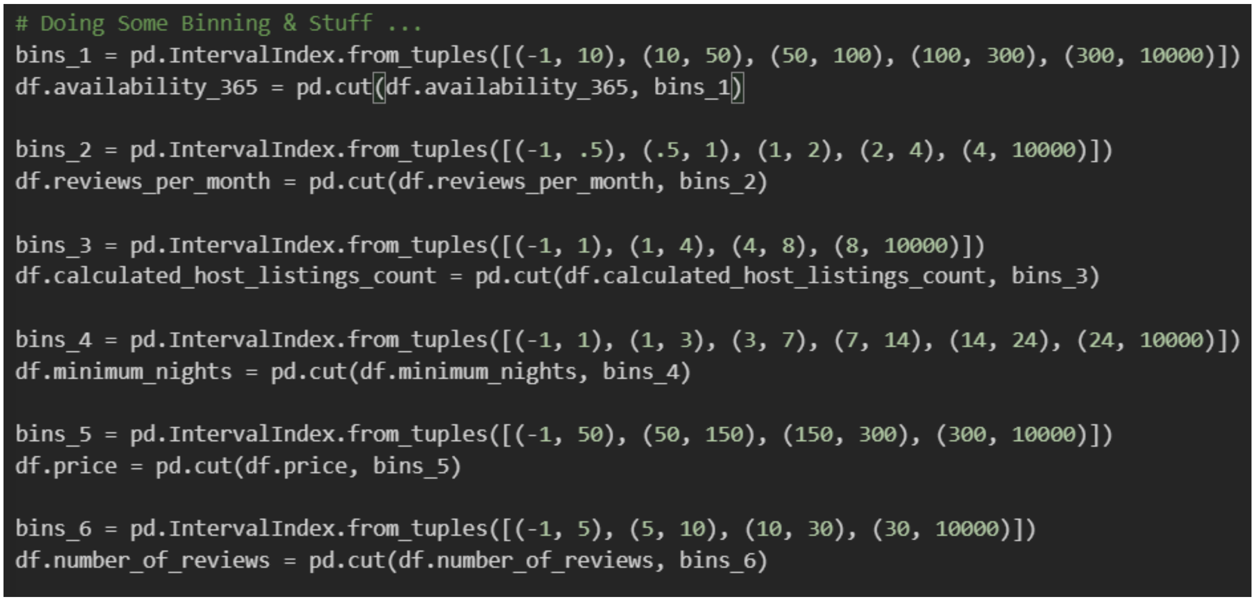
These numerical ones are the 6 ones we see in the chart below, to decide the bins for each of them we need to know the distribution of its data and for that we can take a look at their KDE plots but due to the outliers in each feature what we would see In the kde plot would look something like this



And this clearly doesn’t help us choose our bins, to avoid this we limit the x axis to some reasonable degree



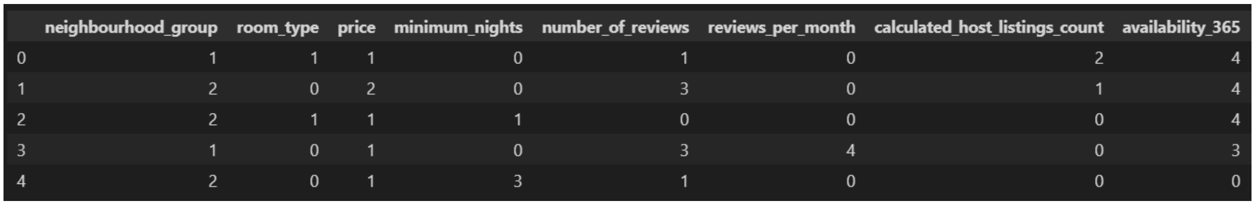
We choose bins by intuition (actually I test a few and the best results came out of this choice for each)



```
# Doing Some Binning & Stuff ...
bins_1 = pd.IntervalIndex.from_tuples([(-1, 10), (10, 50), (50, 100), (100, 300), (300, 10000)])
df.availability_365 = pd.cut(df.availability_365, bins_1)

bins_2 = pd.IntervalIndex.from_tuples([(-1, .5), (.5, 1), (1, 2), (2, 4), (4, 10000)])
df.reviews_per_month = pd.cut(df.reviews_per_month, bins_2)

bins_3 = pd.IntervalIndex.from_tuples([(-1, 1), (1, 4), (4, 8), (8, 10000)])
df.calculated_host_listings_count = pd.cut(df.calculated_host_listings_count, bins_3)

bins_4 = pd.IntervalIndex.from_tuples([(-1, 1), (1, 3), (3, 7), (7, 14), (14, 24), (24, 10000)])
df.minimum_nights = pd.cut(df.minimum_nights, bins_4)

bins_5 = pd.IntervalIndex.from_tuples([(-1, 50), (50, 150), (150, 300), (300, 10000)])
df.price = pd.cut(df.price, bins_5)

bins_6 = pd.IntervalIndex.from_tuples([(-1, 5), (5, 10), (10, 30), (30, 10000)])
df.number_of_reviews = pd.cut(df.number_of_reviews, bins_6)
```

Like I said there's no rule to apply these bins but for example we would want to know if the minimum_nights_alowed on a hotel is exactly 1 or not (because for some "OBVIOUS" reasons some people tend to borrow a room of size 2 for exactly 1 "special" night …)

For now we do label encoding on neighborhood_group and room type, but we'll have another dataset which has these 2 features one hot vectored into 8 distinct Boolean features, and we'll have its correlation heatmap to see if this helps any or not

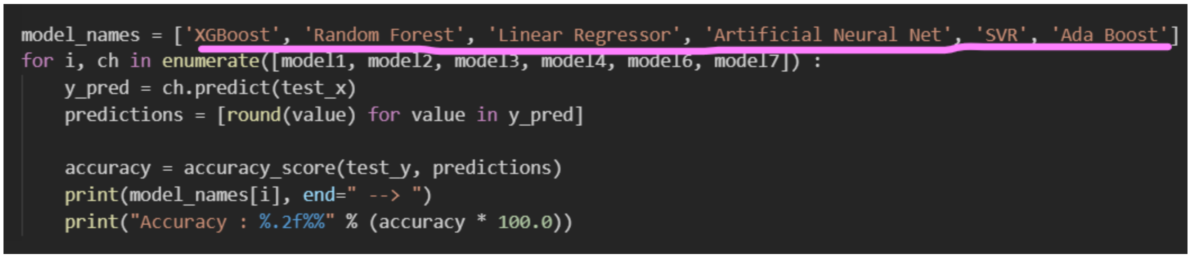


| | neighbourhood_group | room_type | price | minimum_nights | number_of_reviews | reviews_per_month | calculated_host_listings_count | availability_365 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 2 | 4 |
| 1 | 2 | 0 | 2 | 0 | 3 | 0 | 1 | 4 |
| 2 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 4 |
| 3 | 1 | 0 | 1 | 0 | 3 | 4 | 0 | 3 |
| 4 | 2 | 0 | 1 | 3 | 1 | 0 | 0 | 0 |

This picture shows the modified dataframe ready to be fed to some models, to see if we can predict any of the columns and then try to test if removing any feature has an effect to the accuracy

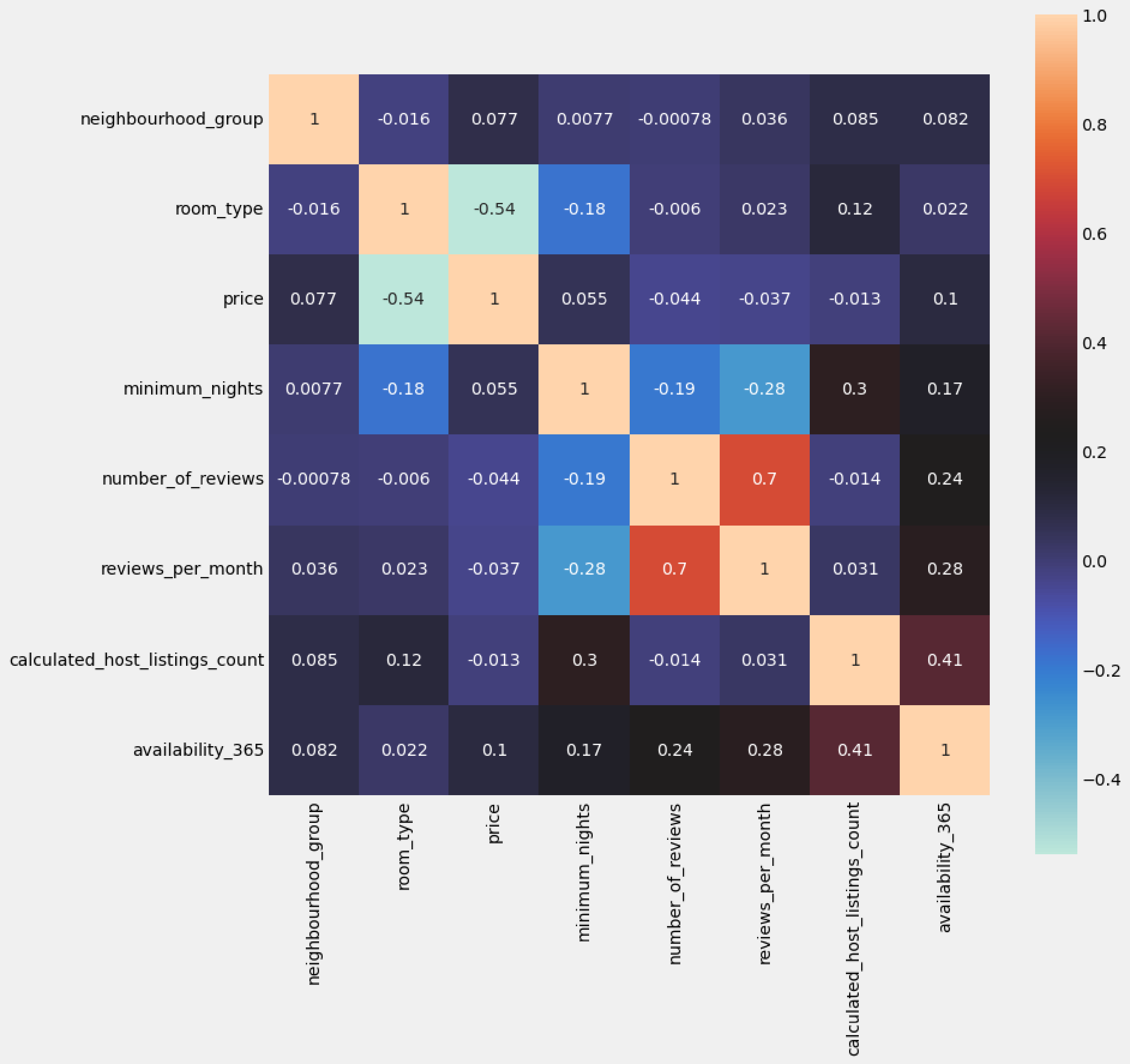
this was my understanding of the question 2, that if we make a model to make predictions, how would changing the features affect the optimum accuracy meaning that feature had an obvious or non-obvious relation to the predicted column and for that purpose I tried out a few models containing a ensemble models and regressors and a neural network, all failed miserably (or I did *sad soroush noises*) but it was still worthful of a mention

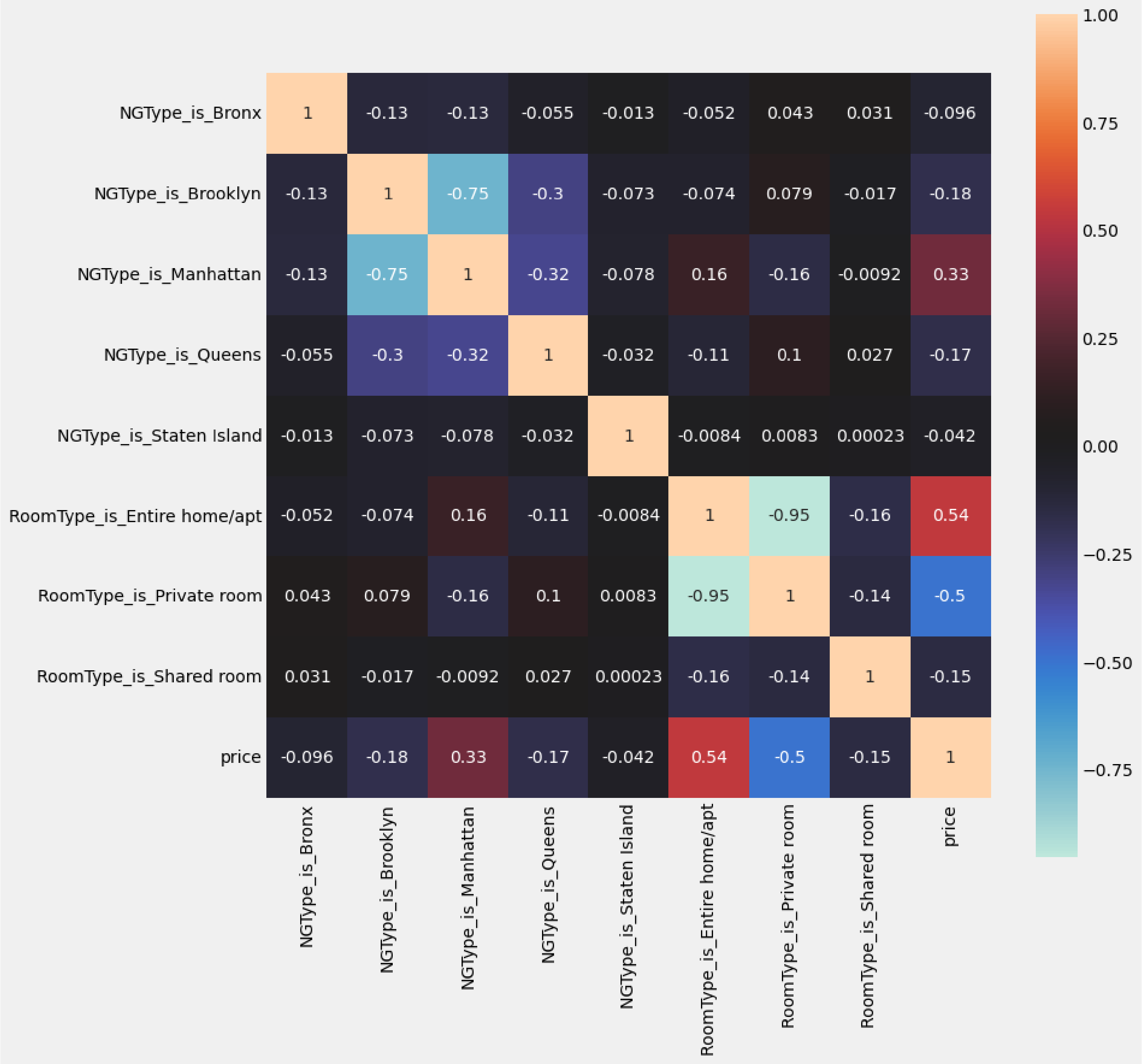
here's the models I tried



```
model_names = ['XGBoost', 'Random Forest', 'Linear Regressor', 'Artificial Neural Net', 'SVR', 'Ada Boost']
for i, ch in enumerate([model1, model2, model3, model4, model6, model7]) :
    y_pred = ch.predict(test_x)
    predictions = [round(value) for value in y_pred]

    accuracy = accuracy_score(test_y, predictions)
    print(model_names[i], end=" --> ")
    print("Accuracy : %.2f%%" % (accuracy * 100.0))
```

well anyway we see the correlation heatmap of the modified dataframe, and below that the correlation heatmap of one-hot-vectored catagories





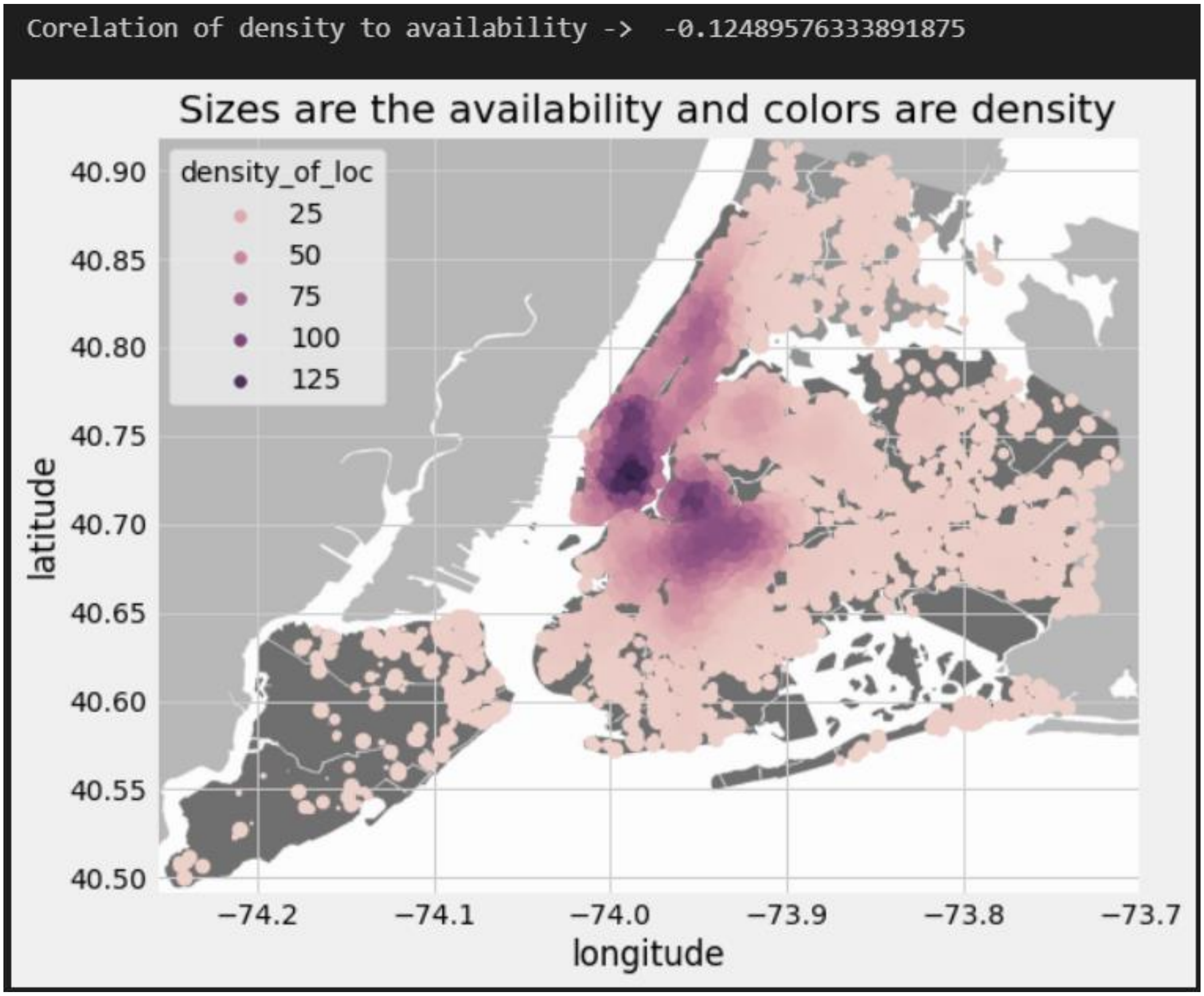
| | NGType_is_Bronx | NGType_is_Brooklyn | NGType_is_Manhattan | NGType_is_Queens | NGType_is_Staten Island | RoomType_is_Entire home/apt | RoomType_is_Private room | RoomType_is_Shared room | price |
|---|---|---|---|---|---|---|---|---|---|
| NGType_is_Bronx | 1 | -0.13 | -0.13 | -0.055 | -0.013 | -0.052 | 0.043 | 0.031 | -0.096 |
| NGType_is_Brooklyn | -0.13 | 1 | -0.75 | -0.3 | -0.073 | -0.074 | 0.079 | -0.017 | -0.18 |
| NGType_is_Manhattan | -0.13 | -0.75 | 1 | -0.32 | -0.078 | 0.16 | -0.16 | -0.0092 | 0.33 |
| NGType_is_Queens | -0.055 | -0.3 | -0.32 | 1 | -0.032 | -0.11 | 0.1 | 0.027 | -0.17 |
| NGType_is_Staten Island | -0.013 | -0.073 | -0.078 | -0.032 | 1 | -0.0084 | 0.0083 | 0.00023 | -0.042 |
| RoomType_is_Entire home/apt | -0.052 | -0.074 | 0.16 | -0.11 | -0.0084 | 1 | -0.95 | -0.16 | 0.54 |
| RoomType_is_Private room | 0.043 | 0.079 | -0.16 | 0.1 | 0.0083 | -0.95 | 1 | -0.14 | -0.5 |
| RoomType_is_Shared room | 0.031 | -0.017 | -0.0092 | 0.027 | 0.00023 | -0.16 | -0.14 | 1 | -0.15 |
| price | -0.096 | -0.18 | 0.33 | -0.17 | -0.042 | 0.54 | -0.5 | -0.15 | 1 |

Not much better than before but still some correlations are seen here, especially room_type vs price

Q3 – which hosts are busiest and why

While I tried a couple of different approaches, the most solid answer lies within the density

In the picture below you can see (not clearly but still you can) that with the colors getting thicker and thicker the size gets bigger too …



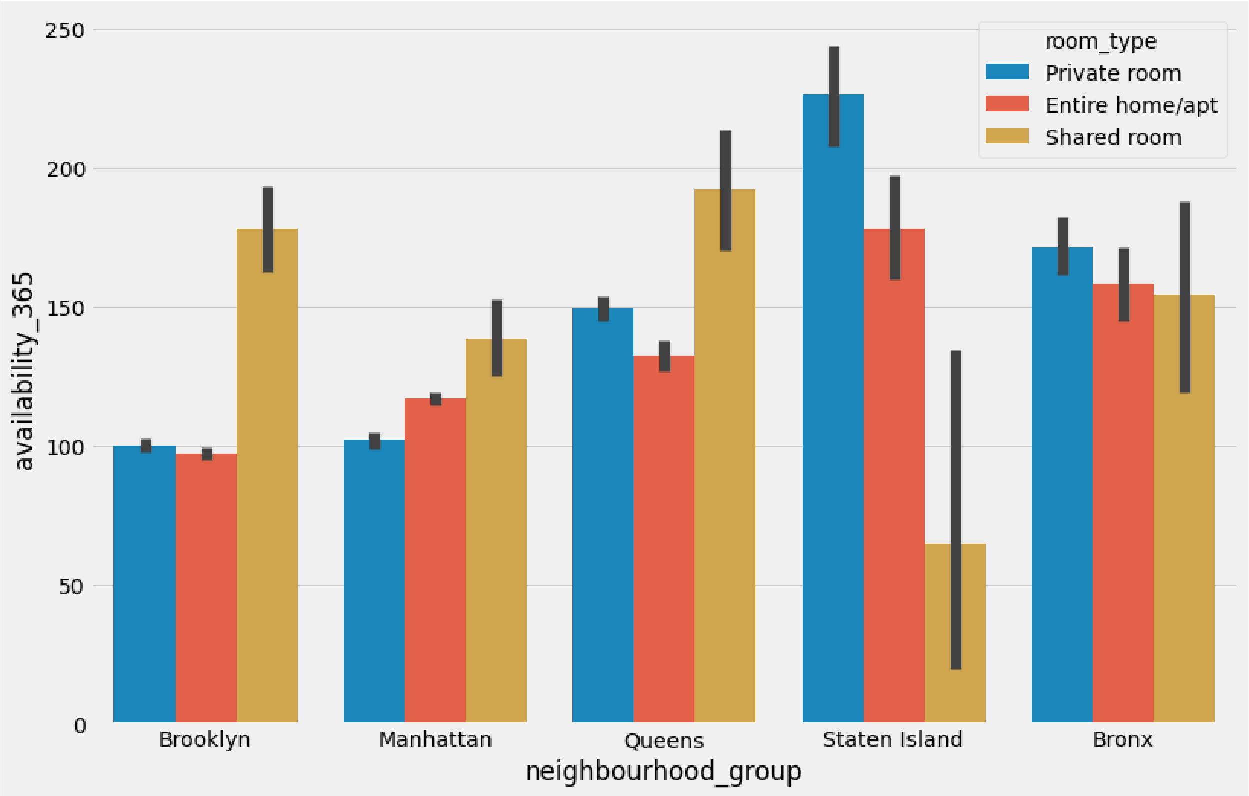
```
Corelation of density to availability ->  -0.12489576333891875
```

Which actually seems reasonable, the more concentrated an area, the more need of the number of hosts

Or it could be because of the central park located at the center of Manhattan which is by it self a point of intrest in NYC that tourist wouldn't hate for their hotel to be close to it

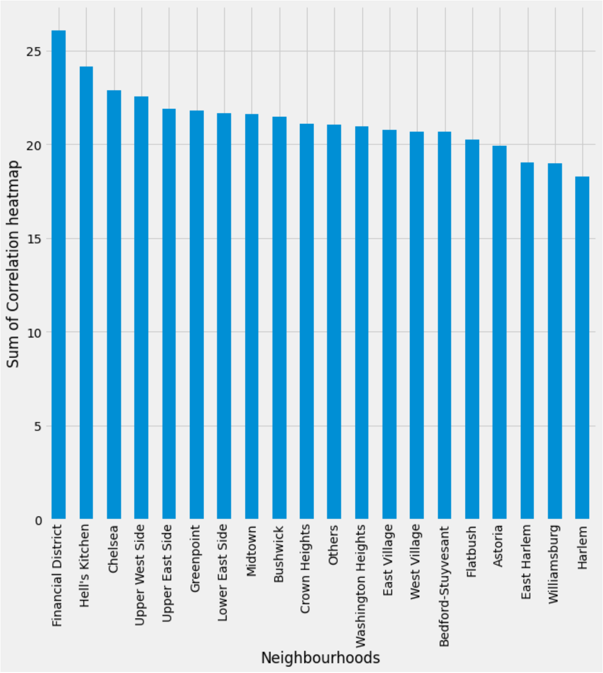
But beside that lets look at the availability distribution on each neighbourhood

And then hue it by the room_type (because Why the hell not!)

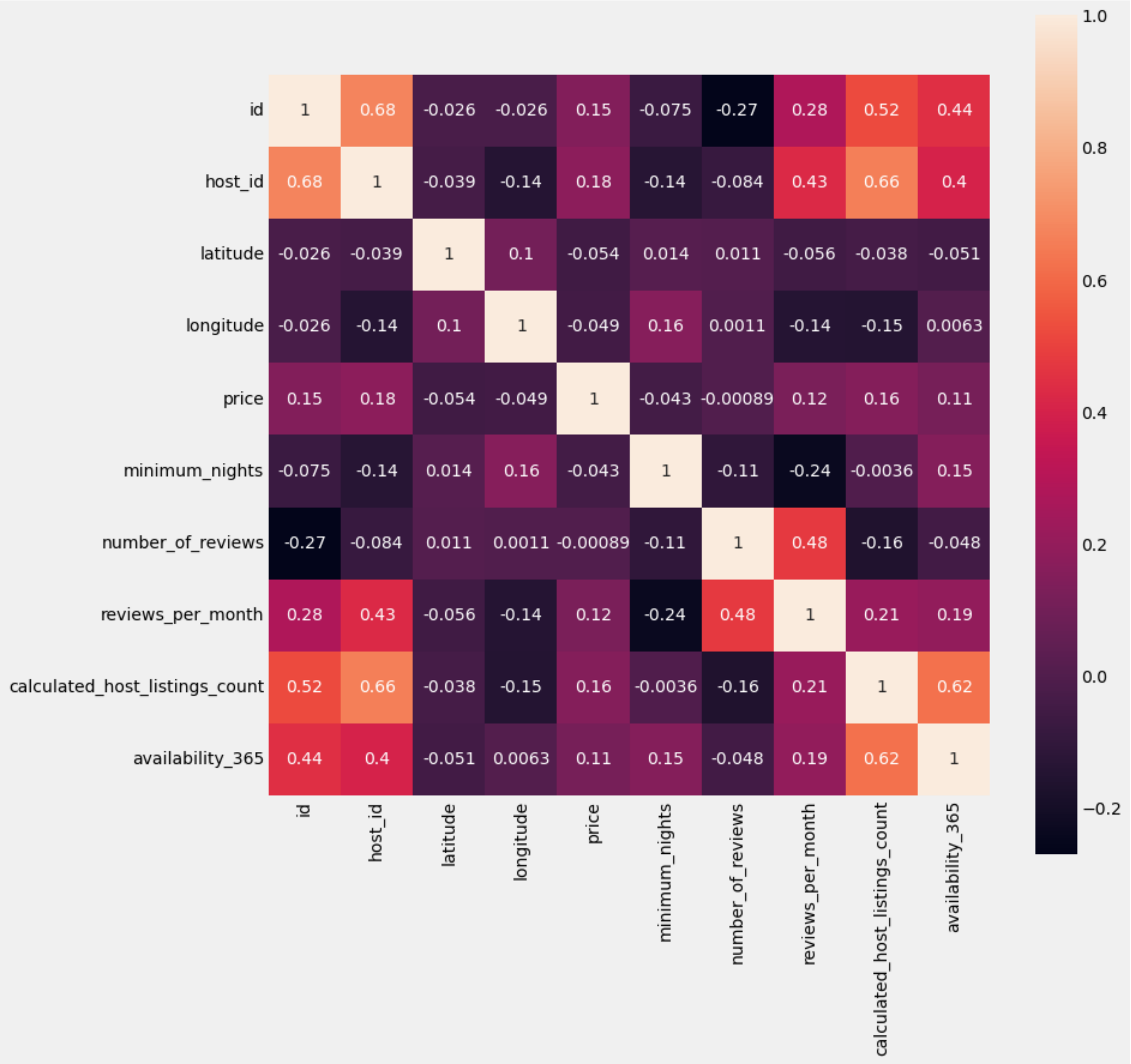


What we'll try to do now is to search each and every neighborhood and see which one has a more meaningful correlation heatmap, meaning in which correlation heatmap we have more values tending to 1 or -1, in other words which heatmap has the highest summation of summation of it's absolute values, the idea behind this laborious work is to find a heatmap so maybe we could get some insight of what causes the availability to be higher or lower, we loop over all the neighbourhoods and save the procedure in a dictionary

Lastly the highest ones are listed here



And the heatmap of ‘Financial District’ group is as follows :



Though still very vague for me that why ‘id’ and ‘host_id’ would show this kind of correlation to availability but what we see here is that, calculated_listings of hosts does show a promising relation to availability (though this needs a statistical test to prove but like I couldn’t manage my time properly to do all the things), but also; reviews_per_month does have an effect, almost like as till the time a host has some active critics meaning the host has some active changes (hopefully for the betterment) the availability is affected

I also tried another approach,

We know that decision trees work really well when it comes to machine learning Explainability

So if we could train a decision tree with our dataset on the availability column it could show us which factors have the most effect (more precisely, gini impurity) with them.

But as I said in question 2 all my models failed miserably in giving out any statistically meaningfull outputs (though the correct way to use this tree here was to booleanize a lot of features that I did not have the time to do)

Just to show you a glimpse of my miserable failure this is a tiny part of my tree (my computer couldn't zoom into that :D)

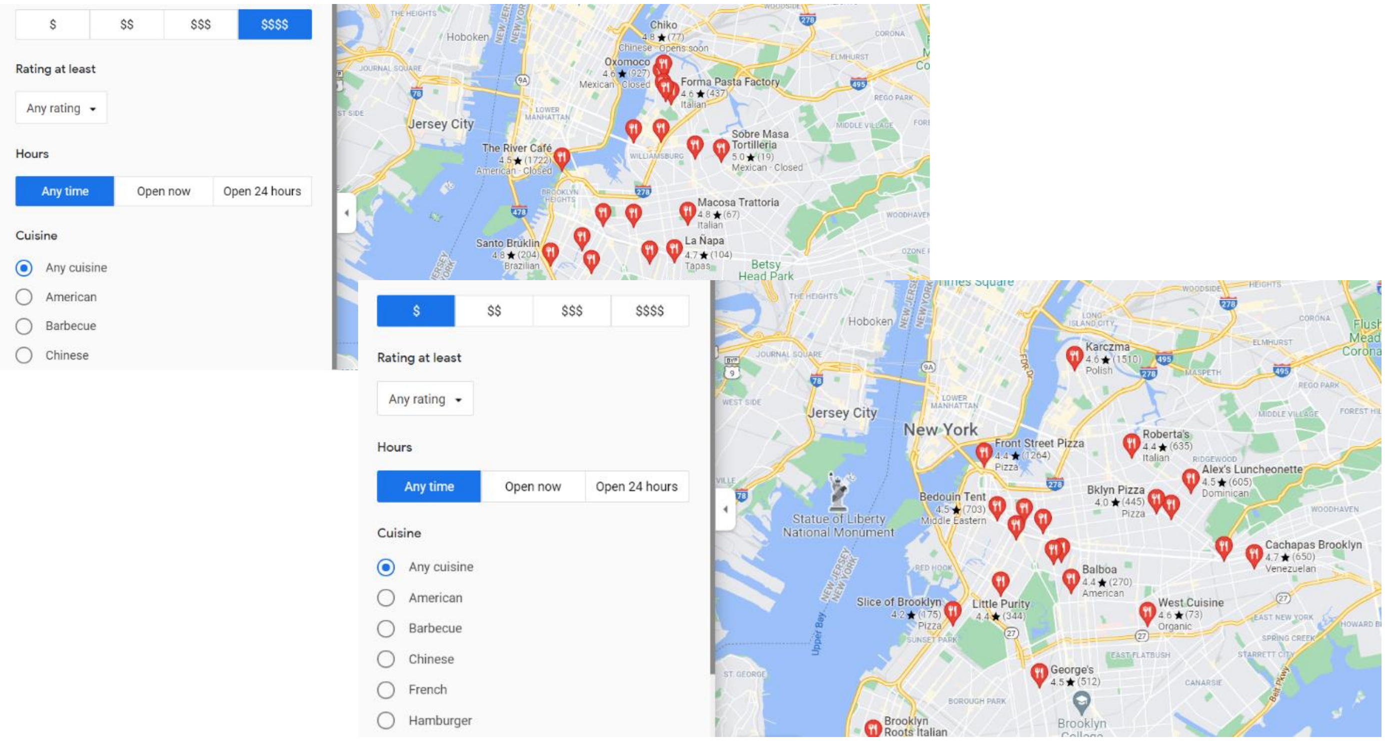
Q4 – Is there any noticeable difference in traffic among different areas and what could be the reason for it

If by 'traffic' the question means traffic of the host (which hopefully it does) there's some fun stuff to do here :

Given the current data set, traffic means the density which we introduced above, but there's something we can do to make it more accurate

I extracted some data from google map and some newyorkcity databases which are the geo-locations of shopping centers, museums, restaurants, etc…

There was this idea to exctract cheap vs expensive restuarants from google map using its filters but I did not find the time to complete it, instead I used a more generalized extracted data



$ | $$ | $$$ | [✓ $$$$]

Rating at least: Any rating

Hours: [✓ Any time] | Open now | Open 24 hours

Cuisine: ● Any cuisine ○ American ○ Barbecue ○ Chinese

[✓ $] | $$ | $$$ | $$$$

Rating at least: Any rating

Hours: [✓ Any time] | Open now | Open 24 hours

Cuisine: ● Any cuisine ○ American ○ Barbecue ○ Chinese ○ French ○ Hamburger

Hoboken
Jersey City
Chiko 4.8 ★ (77) Chinese · Opens soon
Oxomoco 4.6 ★ (927) Mexican · Closed
Forma Pasta Factory 4.6 ★ (437) Italian
Sobre Masa Tortilleria 5.0 ★ (19) Mexican · Closed
The River Café 4.5 ★ (1722) American · Closed
Macosa Trattoria 4.8 ★ (67) Italian
Santo Brúklin 4.8 ★ (204) Brazilian
La Napa 4.7 ★ (104) Tapas
Betsy Head Park
New York
Karczma 4.6 ★ (1510) Polish
Front Street Pizza 4.4 ★ (1264) Pizza
Roberta's 4.4 ★ (635) Italian
Alex's Luncheonette 4.5 ★ (605) Dominican
Bedouin Tent 4.5 ★ (703) Middle Eastern
Bklyn Pizza 4.0 ★ (445) Pizza
Statue of Liberty National Monument
Cachapas Brooklyn 4.7 ★ (650) Venezuelan
Balboa 4.4 ★ (270) American
Slice of Brooklyn 4.2 ★ (175) Pizza
Little Purity 4.4 ★ (344)
West Cuisine 4.6 ★ (73) Organic
George's 4.5 ★ (512)
Brooklyn Roots Italian
Brooklyn College
Upper Bay

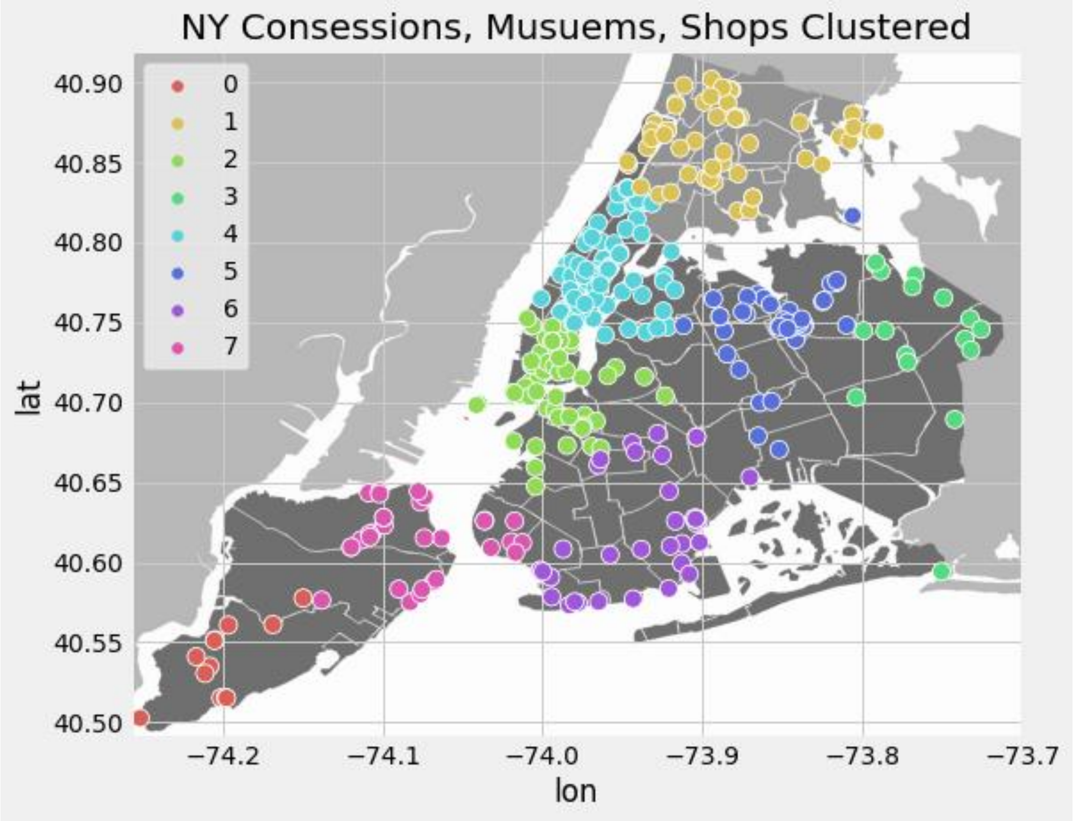
There 3 parts of extracted data, the first one is the shopping centers which I took from the source code of the HTML page because seemingly google map api needs a private key which I couldn't acquire from internet

Next part comes from the museums which I found on database of NYC open database

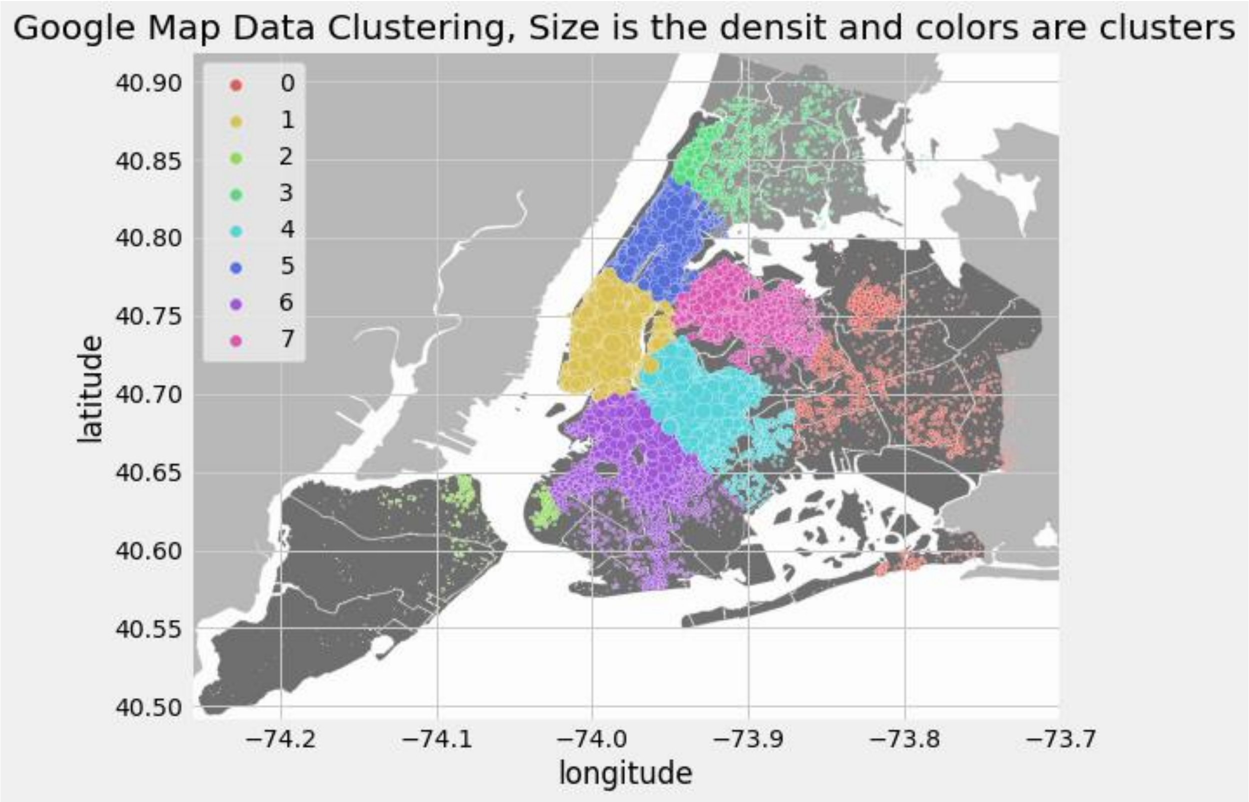
(https://opendata.cityofnewyork.us)

And the last one consist of some points of interest including some restaurants, golf courses, amusement parks, stadiums, etc…

In the picture below you'll see them all together clustered into 8 parts

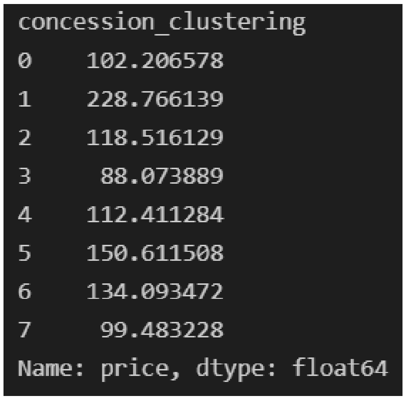


Now that we trained a K-means clustering model on our extracted data, we can use the same cluster centers and plot our original data set using these clusters, this would let us know if these points of interest would cause any traffic (density) of our hosts. This means some of our clusters has to be more dense and were the clusters emerge, we should see more density of hosts



The theory seems to be correct, the yellow cluster has the biggest circles meaning the hosts there are denser than usual, but some other (like the red cluster) do not!

We see the mean density of each cluster here

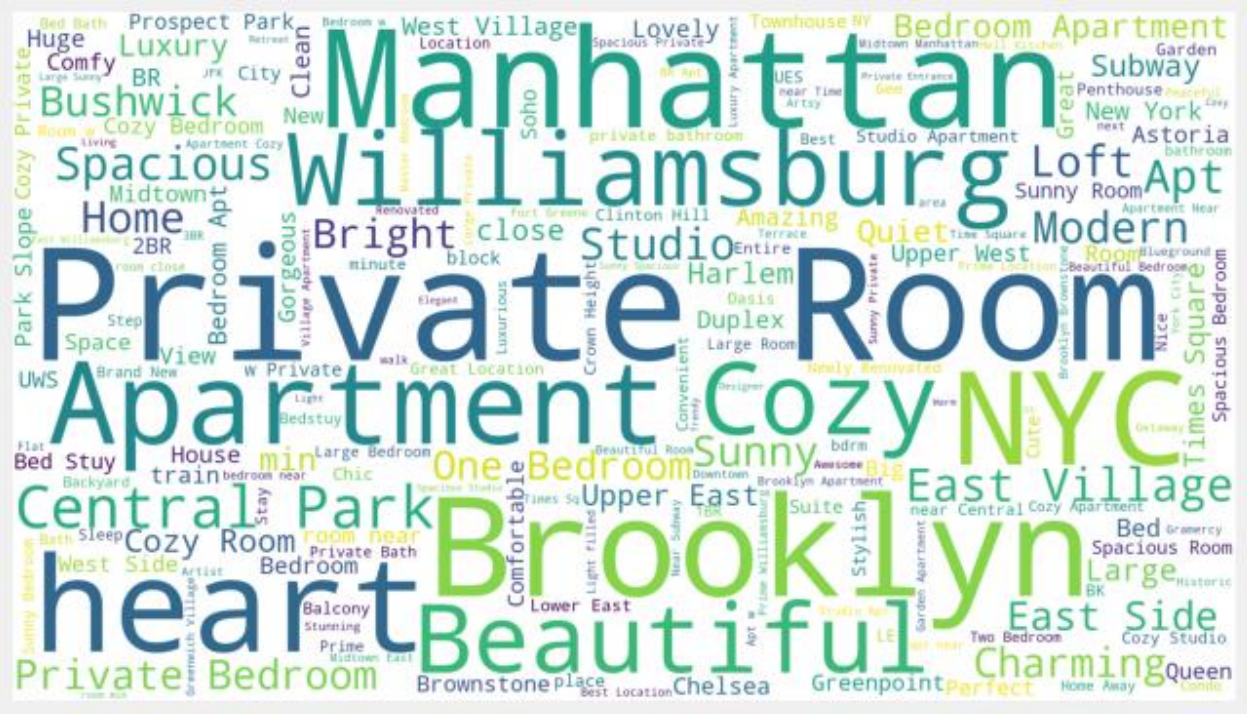


```
concession_clustering
0    102.206578
1    228.766139
2    118.516129
3     88.073889
4    112.411284
5    150.611508
6    134.093472
7     99.483228
Name: price, dtype: float64
```

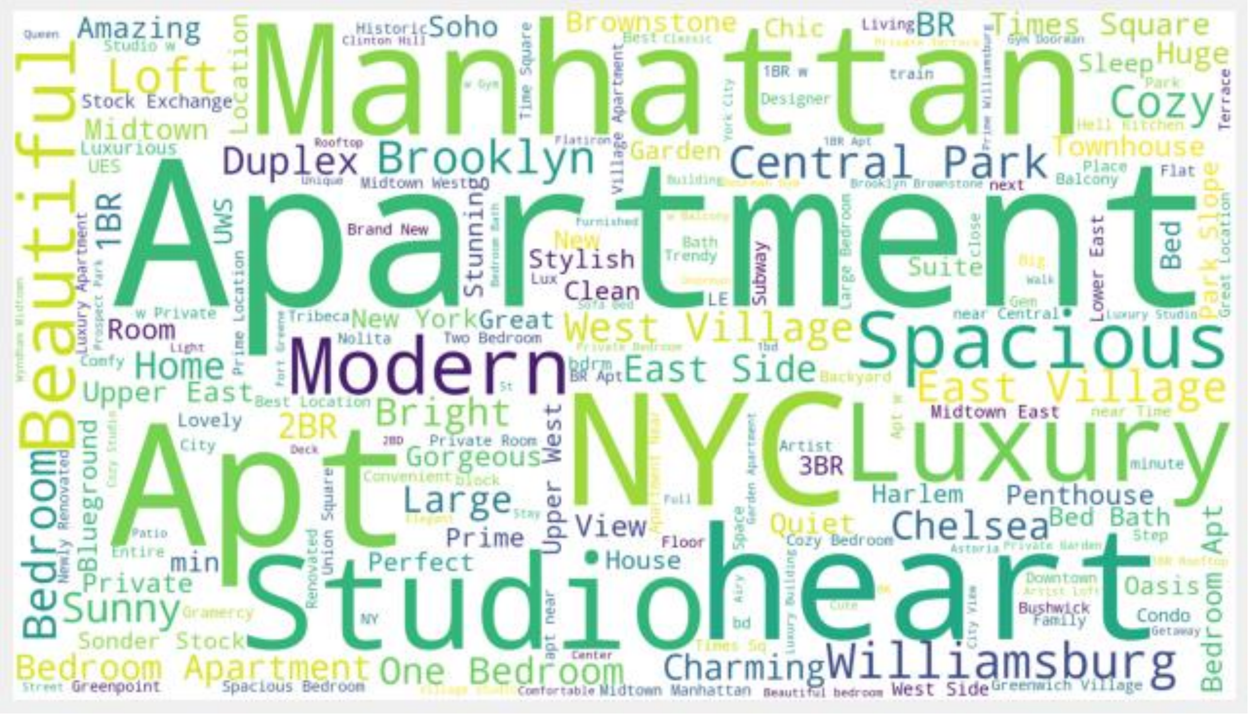
There’s also an additional question to be answered, does the name feature mean anything ?

To answer that take a look at the wordclouds below

This first one show the most common words used in the name column of all of our data

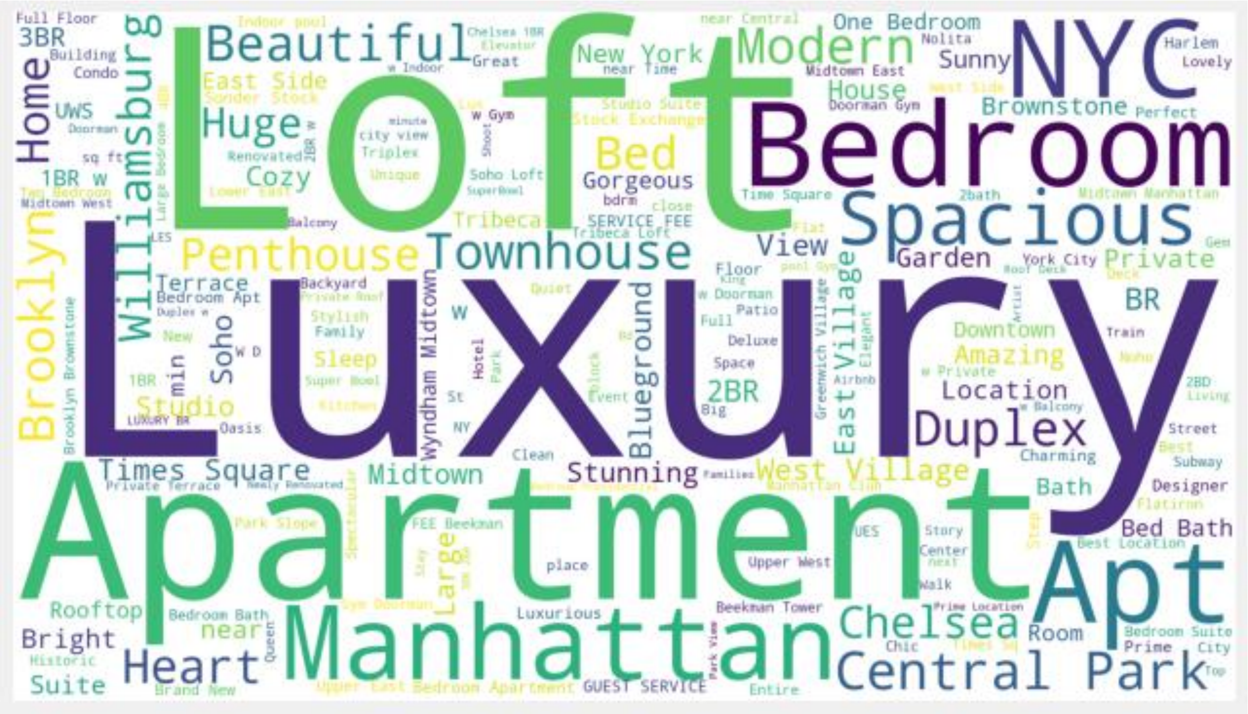


Next we create another wordcloud, but this time only with the ones who have a price higher than average



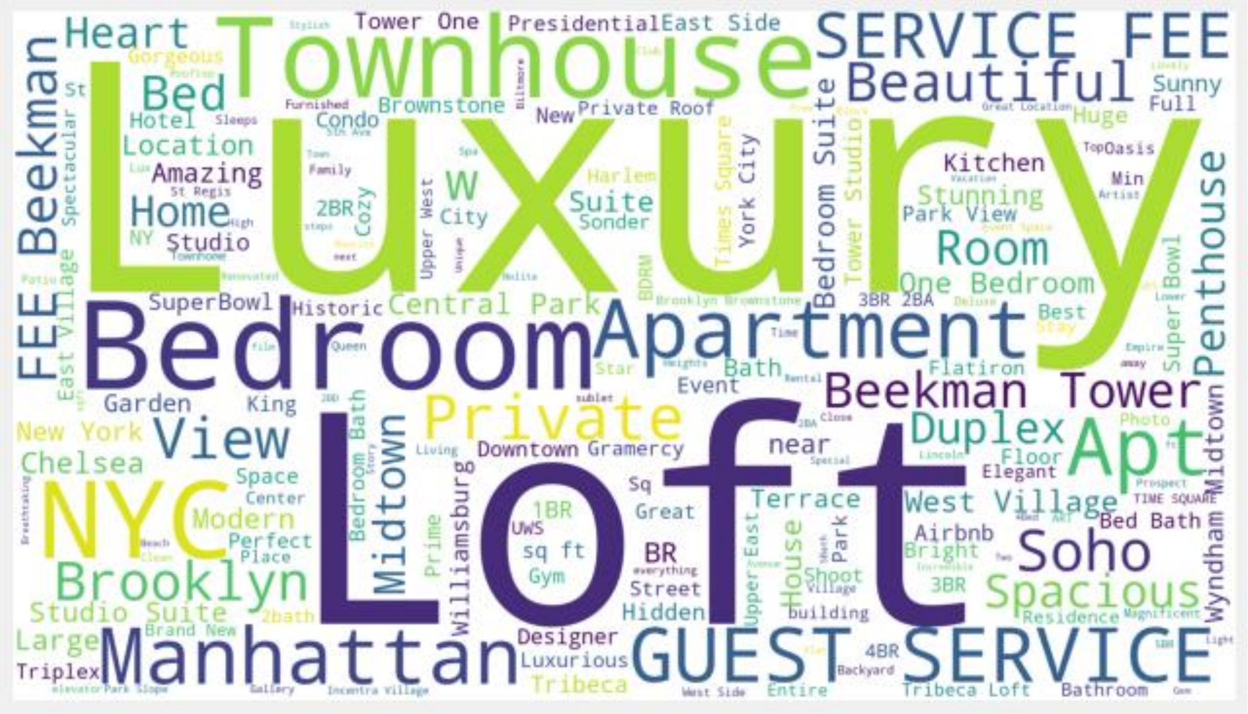
What we see is that the higher the price goes some words start emerge which were not common before an example of these words is ‘Luxury’ or ‘Spacious’ or ‘Modern’ also ‘Apartment’ has more intensity now

This time we look at the wordcloud of those who have a price higher than the average*2



Some words like 'Luxury' and 'Loft' are getting a lot of attention meaning that maybe the host would require more money

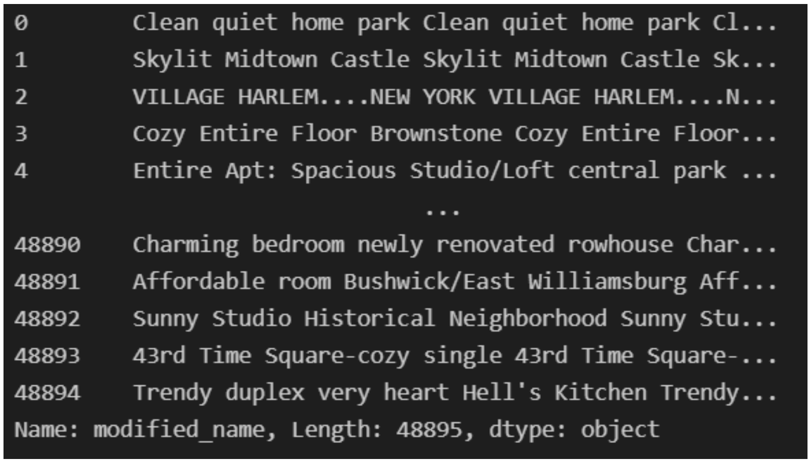
But when we proceed furthur with average*4 not much difference is seen



Now we'll try another approach too, we remove words like 'by' or 'in' or 'is' from the name column and for each name we multiply the value (which is a string) by it's price/10

In this way rows who belong to more costly stays would have their names (respectively the words used in their name) scaled to an extent of their price

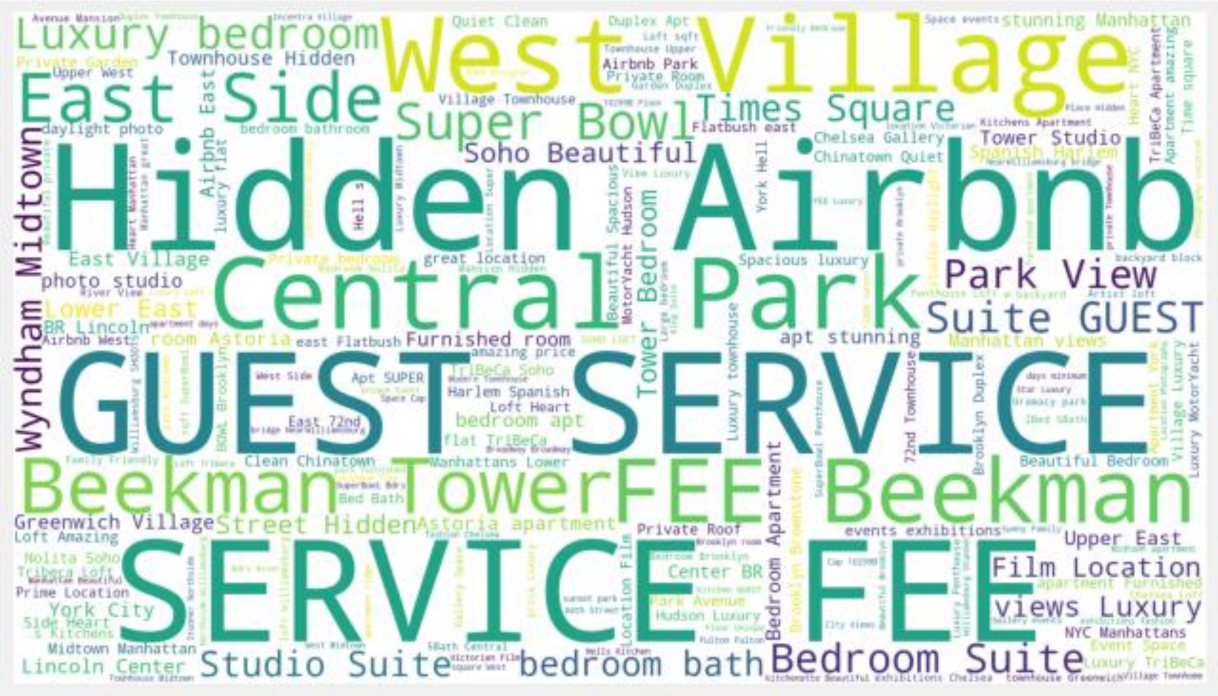
For example if the name of a host in 'Luxury Stay' and it's price is 100 the new feature would have 100 'Luxury Stay' in it's name



```
0        Clean quiet home park Clean quiet home park Cl...
1        Skylit Midtown Castle Skylit Midtown Castle Sk...
2        VILLAGE HARLEM....NEW YORK VILLAGE HARLEM....N...
3        Cozy Entire Floor Brownstone Cozy Entire Floor...
4        Entire Apt: Spacious Studio/Loft central park ...
                               ...
48890    Charming bedroom newly renovated rowhouse Char...
48891    Affordable room Bushwick/East Williamsburg Aff...
48892    Sunny Studio Historical Neighborhood Sunny Stu...
48893    43rd Time Square-cozy single 43rd Time Square-...
48894    Trendy duplex very heart Hell's Kitchen Trendy...
Name: modified_name, Length: 48895, dtype: object
```

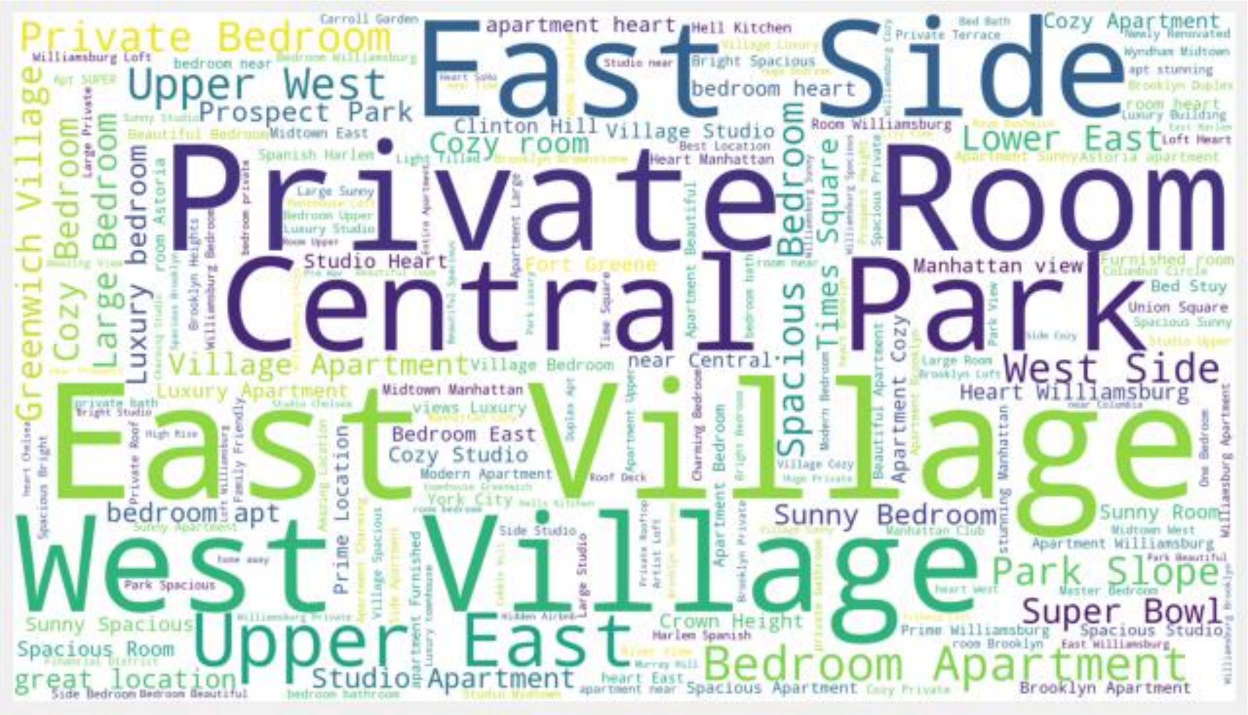
Which ofcourse has it's flaws but is worth a try

These are the ones when extracted from the modified name feature, and it shows a correlation of 0.13



The same principle was applied again, this time working with availability meaning that these words were more common when looking at a row with low availability and the results are :

And as expected hosts around centeral park are the busiest!


Private Room
Central Park
East Village
West Village
East Side
Upper East
Upper West
Private Bedroom
Bedroom Apartment
Greenwich Village
Cozy Bedroom
Large Bedroom
Luxury bedroom
Village Apartment
Spacious Bedroom
Times Square
West Side
Lower East
Park Slope
Super Bowl
Prospect Park
Clinton Hill
Village Studio
Cozy room
bedroom heart
apartment heart
Hell Kitchen
Cozy Apartment
Studio Heart
Manhattan view
Heart Williamsburg
Sunny Bedroom
Studio Apartment
great location
Spacious Room
Sunny Spacious
Crown Height
bedroom apt
Cozy Studio
Bedroom East
Midtown Manhattan
Luxury Apartment
Village Bedroom
near Central
Prime Location
Sunny Room
Spacious Apartment
Brooklyn Apartment
Spacious Studio
Prime Williamsburg
Apartment Williamsburg
Fort Greene
Furnished room
Bed Stuy
Union Square
Heart Manhattan
Midtown East
Spanish Harlem
Carroll Garden
Williamsburg Loft
York City
Modern Apartment
Apartment Furnished
heart East
Harlem Spanish
Park Spacious