NYC AirBnB datamine
A visual data mining analysis of New York City AirBnBs

Leonardo Balzoni - 1870364

May 14, 2020

1 Introduction to the analysis

Housing websites like *AirBnB* [1] are among the websites that generate the most traffic worldwide[2]. Even with an immense amount of unseen work that ensures a pleasurable user experience, it can still be difficult for visitors to find the information they are looking for, given the sheer amount of data available. This project thereby tries to give some otherwise inaccessible insights about the announcements, with the help of data mining tools and graphs visualization. Conducting this research on a global scale would be too computationally heavy for my resources, therefore the analysis is limited to *New York City*, one of the cities that has the most announcements on the website.

2 Related work

Before starting to develop this tool, I wanted to understand which is the state-of-the-art for housing websites' visualizations. What I found out is that in almost all the cases the visualizations were quite limited and only offered few insights while focusing on mainly one topic in each case.

- *AirBnBShare* [3]: A trend-exploration tool aimed to help people to efficiently construct their travel experience and encourage the participation into the sharing economy. The statement on which their analysis is built is that Airbnb is known for its core philosophy of connecting people and culture by providing a residential-space sharing platform. However, through data exploration, they noticed a drastic gap between the ratio of the Entire Home listings and Shared Room listings.
  The issue is that the tool focuses mainly on shared rooms, which is for most tourists not the ideal accommodation.

- *InsideAirBnB* [4]: The question at the base of the visualization is how is Airbnb really being used in and affecting the neighbourhoods of your city. The focus of this analysis is that data shows that the majority of Airbnb listings in most cities are entire homes, many of which are rented all year round - disrupting housing and communities.
  Even in this case the tool is used more as a way to prove their statement than to actually help users that want to use AirBnB.

3 Description of the dataset

The dataset that is going to be used was provided from *Kaggle* [5], and it is made of around 50000 tuples with 16 attributes each [6]. I am aware of the fact that doing data mining tasks on this big of a dataset, on a web application, could result in a slow navigation; the best solution would be to delegate the heavy lifting to a backend server but for the moment I instead faced 2 options: reduce the amount of attributes or reduce the amount of tuples.
What I ended up doing is a mix of the 2, as I dropped useless attributes such as the name of the host or the date of the last review, and reduced the amount of tuples to 10000. The dataset is still quite big, but the web browser was able to manage it.
Each tuple of the set represents an announcement on the AirBnB website for a house in New York City, some of the most useful attributes are: the location of the house (both its neighbourhood and its coordinates), its cost, the average availability, and the amount of reviews. These information will be used to identify patterns and gather insights about the distribution of rental housing in NYC in order to create a user friendly visual representation of the best options available.

4 Goals of the Analysis

An analysis of this kind can be used to fulfill multiple objectives; firstly as already stated the multiple visualizations will allow to see otherwise hidden information, such as for example the average price in each neighbourhood, or the estimated income of the various hosts. This means that both someone who is willing to put a announcement on the website and someone who is looking for an accomodation, will be able to find useful information.
Moreover I applied some dimensionality reduction algorithms such as *PCA* or *t-SNE* to gather informations about similar announcements, which could for example be used to find an announcement similar to one that we like, but that might be unavailable.
Lastly I also created a tool for *"Intelligent pricing"*, what I did was creating a machine learning algorithm using a *random forest regressor* [7], which is able to suggest to the users the optimal price at which to create a posting for one of their houses basing itself on houses with similar features already on the website.
As for non-functional requirements, I want to make sure that the tool can

be used on a wide range of systems and devices, this means that responsiveness and performance will be always taken into consideration during its development.

5 Dataset preprocessing

The given dataset was immediately almost ready to be used, as there were no duplicates and the attribute's types were coherenent and almost always not null.
Even with this said some preprocessing was necessary, I began by dropping some useless columns such as the name of the host or the date of the last review. Moreover there were some announcements with missing data for important fields, so I dropped those (circa 20 out of 50000).
One more thing that I did was using the information available to evalauate the estimated monthly income for each of the announcements, this was done by multiplying the price for the amounts of days in which the house is rented (365 - availability) and by dividing everything by 12.
Moreover since I saw that only a minuscule percentage of announcements had a cost per night greater than 500$, I decided to drop these ones. I did this so that the graphs would not be extremely skewed just to represent a couple of announcements.
Lastly there was the "Reviews per month" attribute that was null in the case of 0 reviews on the announcement. Instead of dropping these, what I did was to fill the null values with 0 in the case of no reviews for the house.

5.1 PCA and RFR preprocessing

While what I said is enough for most of the analyses that I ended up doing, evaluating PCA and working with the Random Forest Regressor requires for more attention. This is due to the fact that these algorithms only accept numerical values for the various fields, and the dataset instead has some categorical ones (i.e. "neighbourhood" or "room_type"). What I did was creating some sort of one-hot encoding for those fields; for example "room_type" could be: an entire apartment, a private room or a shared room, so I replaced the "room_type" field with 3 new ones: "room_type_apt", "room_type_pvtroom" and "room_type_shroom", and the values are all 0 except for one which is 1. Doing the same for the rest of the categorical fields I finished the preprocessing for the dataset for these algorithms.

6 Tools used

The code is mainly written in *Javascript* and *HTML*, although at the core of our analysis is *D3.js* [8], a small, free javascript library for manipulating documents based on data. Instead of using the basic D3 functions I also utilised *PlotlyJS* [9], a high-level, declarative charting library built on top of D3, which ships with lots of chart types.
It also has to be stated that the preprocessing of the dataset was done in *Python* [10] with the aid of *Pandas* [11], a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language, and the data mining algorithms were implemented with the help of *Scikit-learn* [12].
Lastly to make things like HTML document traversal, manipulation and event handling much simpler I also used *jQuery* [13] .

7 Visualizations

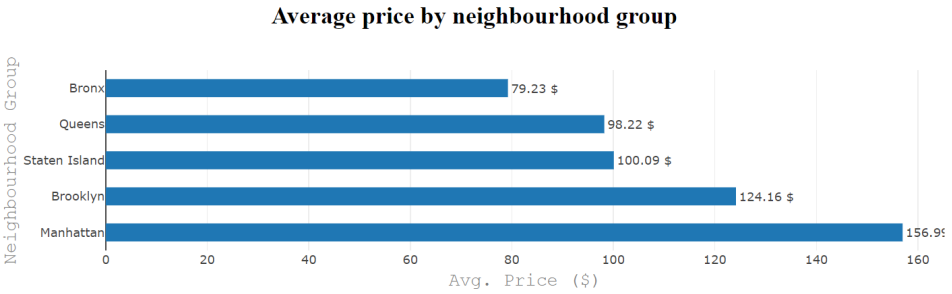
I will now show more in detail each of the graphs that I created, for each one I will explain its function and the available interactions that it offers, starting from the more basic ones, and proceeding to the most interesting.

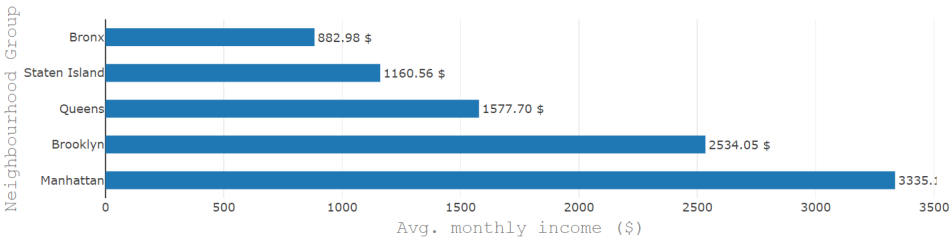
7.1 Average neighbourhood price and income barcharts

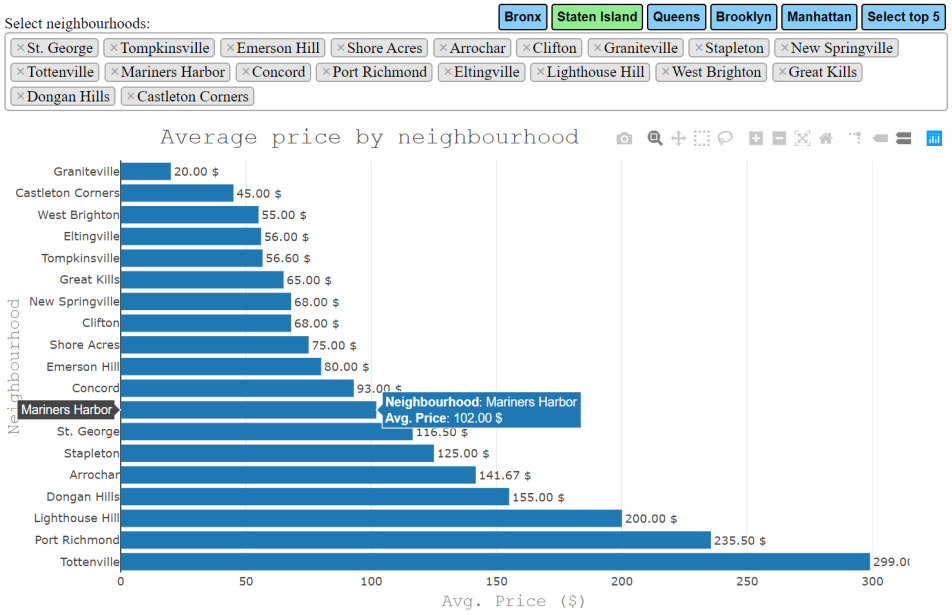
I began by creating some barcharts to visualize some pricing information across the city:

- The first one shows the average price of announcements in each of the NYC neighbourhood groups (Bronx, Staten Island, Queens, Brooklyn, Manhattan).
- The second one shows the average monthly income of announcements in each of the NYC neighbourhood groups.
- The third one is more complex, it allows to see the average price in each single neighbourhood of the city, offering to the user the option to manually select the neighbourhoods he is willing to compare. Moreover I created some buttons that allow to quickly compare all the neighbourhoods inside each of the neighbourhood groups.

Each of these charts allows to pan, zoom and select neighbourhoods. By selecting one or more hoods, all the announcements belonging to them are selected across all the other graphs.

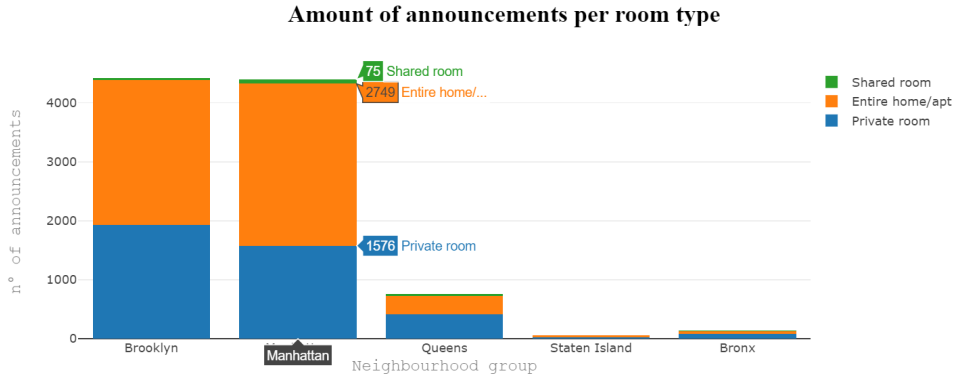

Average price by neighbourhood group
Neighbourhood Group
Bronx
Queens
Staten Island
Brooklyn
Manhattan
79.23 $
98.22 $
100.09 $
124.16 $
156.9
0
20
40
60
80
100
120
140
160
Avg. Price ($)


Average monthly income by neighbourhood group
Neighbourhood Group
Bronx
Staten Island
Queens
Brooklyn
Manhattan
882.98 $
1160.56 $
1577.70 $
2534.05 $
3335.
0
500
1000
1500
2000
2500
3000
3500
Avg. monthly income ($)


Select neighbourhoods:
Bronx
Staten Island
Queens
Brooklyn
Manhattan
Select top 5
St. George
Tompkinsville
Emerson Hill
Shore Acres
Arrochar
Clifton
Graniteville
Stapleton
New Springville
Tottenville
Mariners Harbor
Concord
Port Richmond
Eltingville
Lighthouse Hill
West Brighton
Great Kills
Dongan Hills
Castleton Corners
Average price by neighbourhood
Neighbourhood
Graniteville 20.00 $
Castleton Corners 45.00 $
West Brighton 55.00 $
Eltingville 56.00 $
Tompkinsville 56.60 $
Great Kills 65.00 $
New Springville 68.00 $
Clifton 68.00 $
Shore Acres 75.00 $
Emerson Hill 80.00 $
Concord 93.00 $
Mariners Harbor
Neighbourhood: Mariners Harbor
Avg. Price: 102.00 $
St. George 116.50 $
Stapleton 125.00 $
Arrochar 141.67 $
Dongan Hills 155.00 $
Lighthouse Hill 200.00 $
Port Richmond 235.50 $
Tottenville 299.0
0
50
100
150
200
250
300
Avg. Price ($)

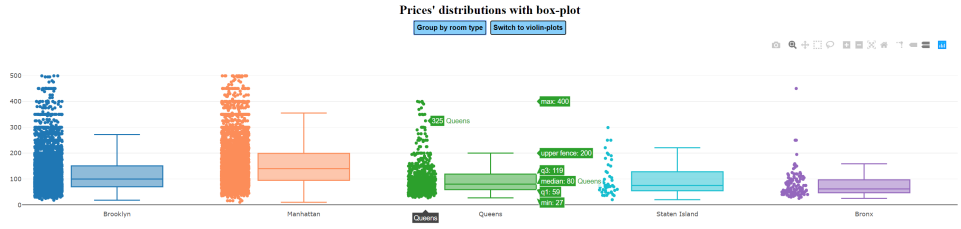
7.2 Amount of announcements per neighbourhood barchart

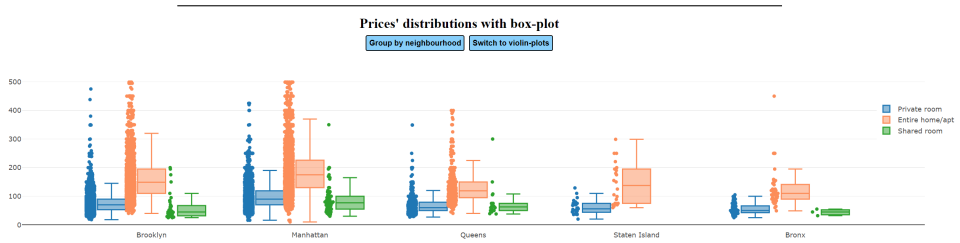
Following the price barcharts, I noticed a huge disparity not only in the amounts of announcements in the different neighbouhoods groups but also in the kinds of room types available for rental, so I created this stacked barchart to visualize it.
The available housing types are: entire apartment, private room or shared room. I divided the barchart per neighbourhood group, hovering over one of these shows the exact housing type distribution in that neighbourhood.



7.3 Box-plot for pricing distribution across neighbourhoods

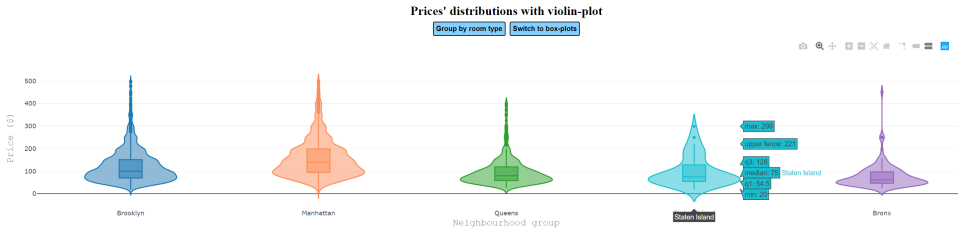
Here I created boxplots showing the price distribution across neighbourhood groups, hovering on each one shows informations about the distribution (max, min, upper and lower quartiles and median)
Next to each boxplot I also put the points that generated it, allowing users to make selections in whichever price section they want.
Also in this case when the user makes a selection, the same announcements are selected across all the other graphs.
Via a button the user can also switch the visualization to one that shows detailed boxplots for each of the housing types (entire apartment, private room, shared room).





7.4 Violin-plot for pricing distribution across neighbourhoods

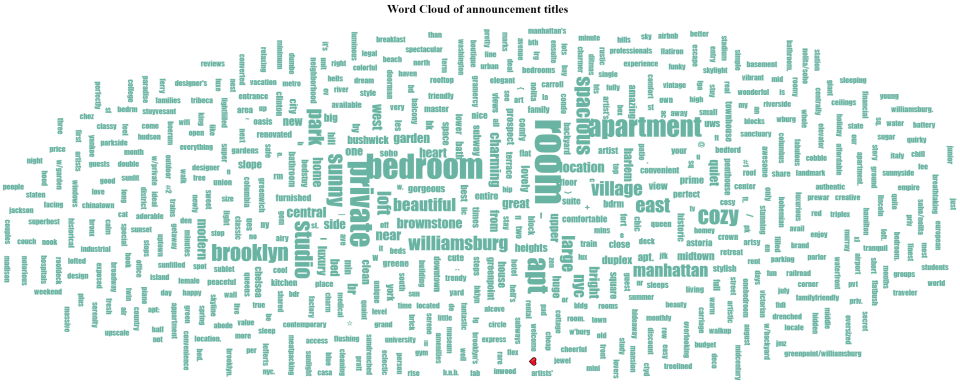
From the box-plot section the user is able to click a button that switch the graph to a violin plot one. Since the information gainable from these 2 graphs is the same I thought that having both viewable at the same would be useless, so the user is allowed to use the one he prefers.
Also in this graph via a button the user can switch the visualization to one that shows detailed violin-plots for each of the housing types (entire apartment, private room, shared room).



7.5 Word Cloud

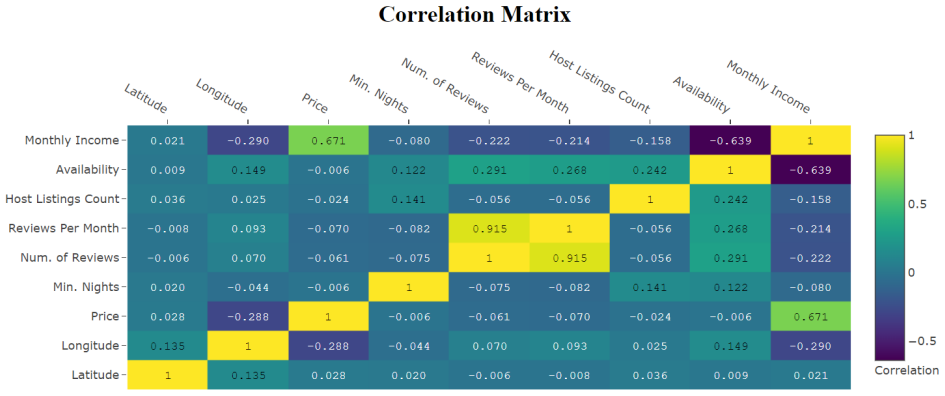
By exploring the dataset it was immediately clear that hosts tend to put catchy words in their announcements to make the house sound more appealing. So by counting each word's occurrences in all the titles of the announcements in the datastet I was able to find the most used ones (after eliminating stopwords) and to plot them in a word cloud view.

The dimension of the words in the word cloud is linearly dependant on their occurrence in the announcements' titles.



7.6 Correlation Matrix

While searching for ideas of information to gain from the dataset, I wanted to see if some of the attributes of the announcements were tied to eachother. The best way to do so is to create a correlation matrix between attributes able to highlight the attributes that are related to eachother.

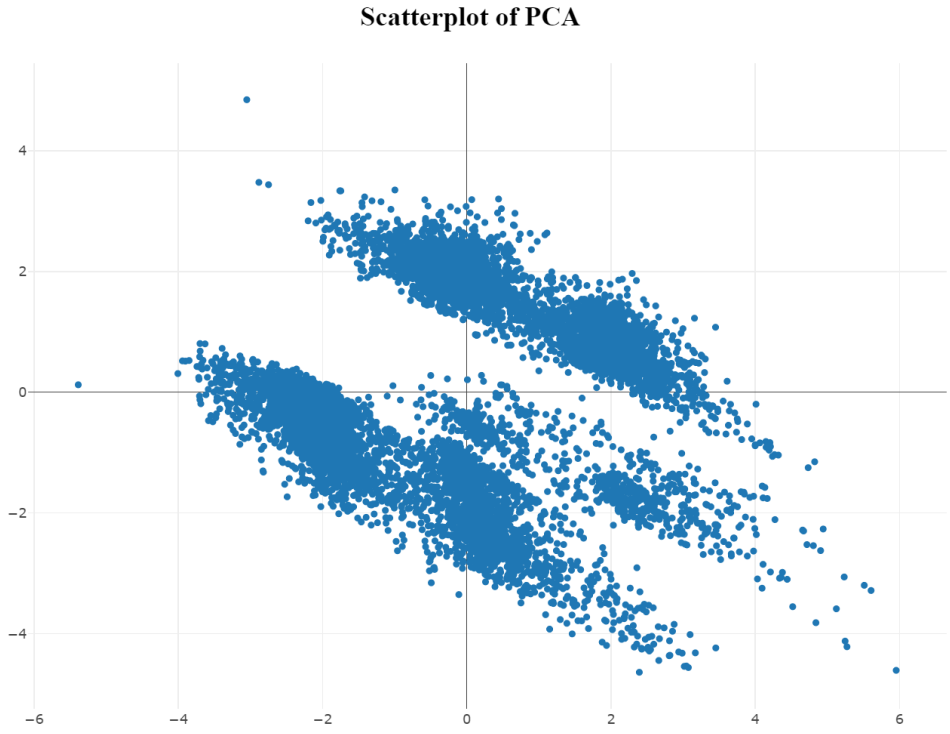


As one can see, its easy to tell that there are no statistically relevant correlations except for the obvious ones like between price and income or between total number of reviews and reviews per month.

7.7 PCA

Something that can be especially useful in sites like AirBnB is a way to find similar announcements. One of the ways to do so is to use a dimensionality

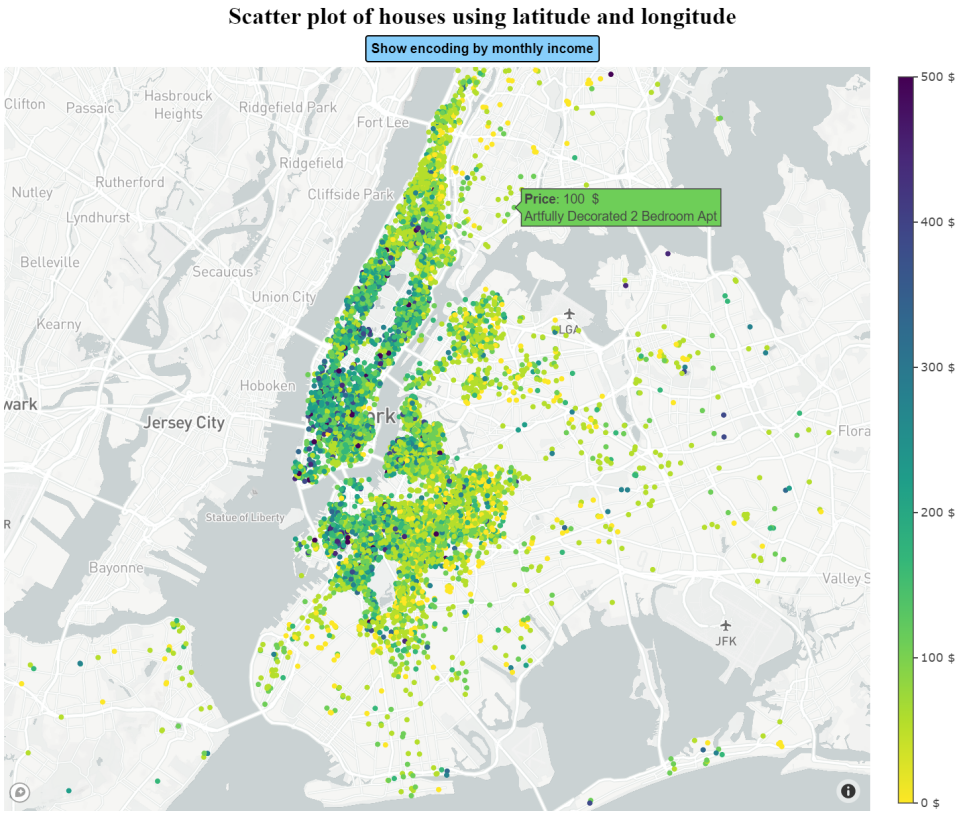
reduction algorithm, I chose PCA. As a normalization algorithm *z-norm* was used, in this way the closeness between the announcements is preserved.
Since the PCA evaluation is extremely computationally heavy what I did was evaluating it once and then saving in a file its results. In this way the loading of the browser remains snappy.
Of course I made it possible to select announcements from the scatterplot to others graphs, in this way the user is able to see how the announcements in the clusters created from PCA are related to eachother.
In a later section I will show the importance of this and how interesting are the insights that this interaction provides to the user.

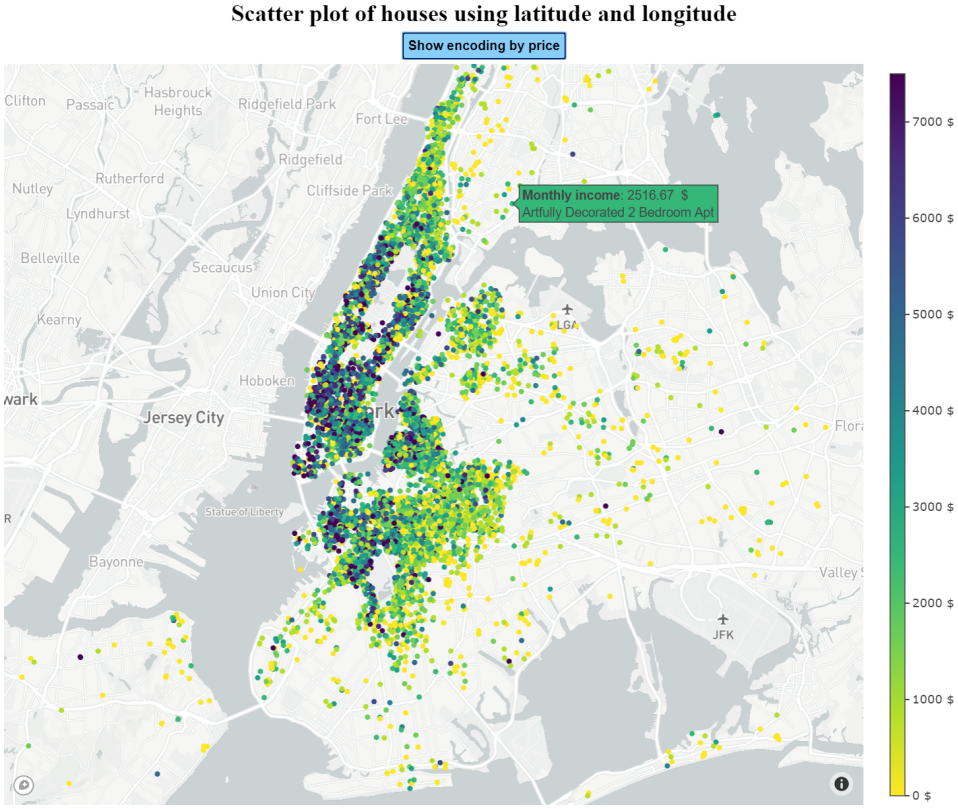


7.8 Map Scatterplot with encoding for price and average income

Last but not least I used the announcements' latitude and longitude values to create a scatterplot of the announcements using their location. I overlayed this info to a New York City map obtaining an interesting result.
As an added encoding I used the color of the points in the scatterplot to show either the announcements' price per night or their average monthly

income.
Even in this case when the user selects announcements the selection carries across other graphs. As I will show in the next section this allows for some extremely interesting insights.



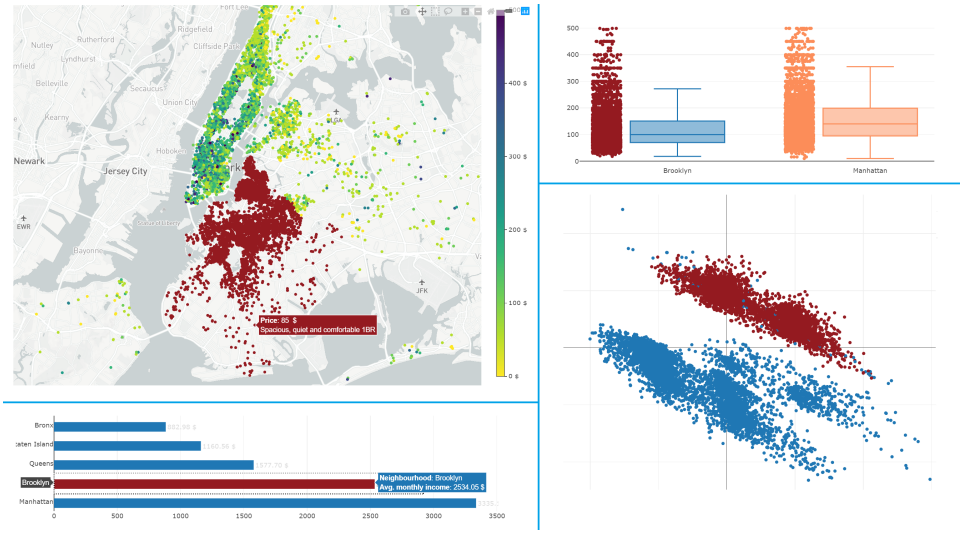


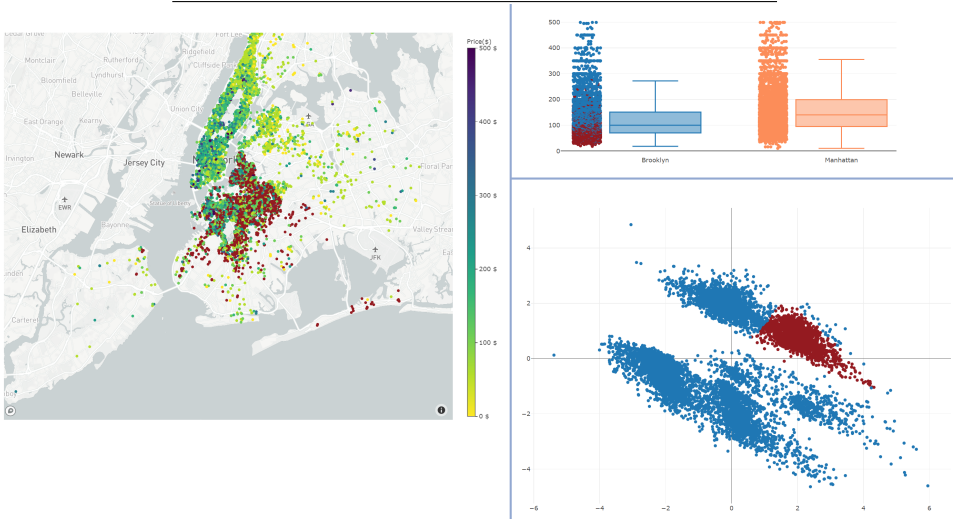
8 Interaction design

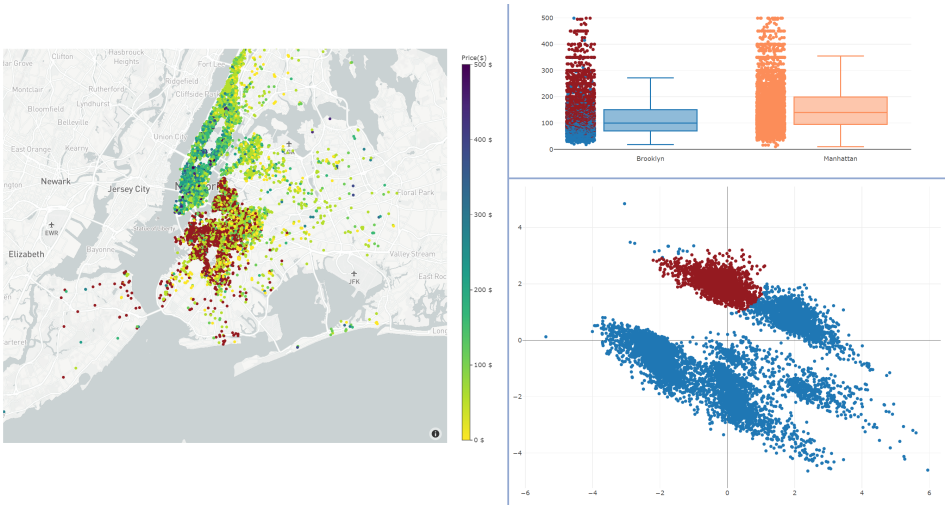
Visualizing data is one thing, being able to interact with it is another. I wanted to make it possible for the user to be able to interact with all the graphs in the webapp, and the choice of using *PlotlyJS* surely helped in this regard. Thanks to it, all the graphs done using the library are: pannable, zoomable and selectable both with a box selection or a lasso selection tool; moreover the user is also able to reset the graph to its original view and download it as a png image.

Even though this already makes a huge difference with respect to static data, I wanted something more. There are multiple scatterplots across the project (in the pricing distribution graph, on the map, and on the PCA representation) and I thought that it would have been interesting to make the selection in a across-project manner. This means that when the user selects some houses in one of these graphs, the selection is carried across to all the others (I also allowed selection from the pricing barcharts; if the user

selects a neighbourhood, the selection transfers to the scatterplots).
This is especially useful to see how the clusters created by PCA are reflected in the actual pricing/position of the announcements, the images below show how clearly announcements are clustered taking into consideration both the price and the neighbourhood:







Notice how clearly the big cluster corresponds to the houses in the Brooklyn area and how the 2 small subclusters belonging to the big one correspond one to houses with a high cost, and one to houses with a low cost. By using the tool this selection is able to show even more interesting results.

9 Intelligent Pricing

As I stated at the beginning of this paper I also developed an *"Intelligent pricing"* algorithm. Note that this was not done on the web application but on *"Google Colab"* [14] as the computation needed to train the algorithm is exremely heavy I used a workstation provided by Google.
What I basically did was training the algorithm using Random Forest Regression [7], a machine learning algorithm based on bagging, to learn how the different characteristics of the announcements on the website influence the pricing of the houses.
It's obvious that this feature could be extremely useful for someone that is willing to add an announcement to the site but is unsure about the best price to put it at. Since RFR have tons of parameters to tune, after estimating a baseline with the default values, I did some extensive parameters tuning to optimize the results as much as possible [15].
With the tuned hyperparameters the algorithm was able to estimate the pricing of the houses on the test set with impressive evaluation metrics.

	algorithm	CV error	CV std	training error	test error	training_r2_score	test_r2_score
Random Forest Regressor	0.17424	0.005687	0.056464	0.175433	0.877138	0.616326

10 Future work

While I am quite proud of the results that I was able to obtain with this project, one main thing that I have in mind and that for sure would improve the tool, is to create a backend. In this way I could delegate the heavy lifting for example to a Node.js server instead than doing everything on the web server.
By adding this feature, even if the dataset was constantly changing one could re-evaluate the PCA every few hours and continuosly improve the intelligent pricing algorithm to keep the changes into consideration.

References

[1] "AirBnB's website"
https://www.airbnb.it/.

[2] "Web traffic on AirBnB"
https://www.similarweb.com/website/airbnb.com.

[3] "AirBnBShare data exploration"
http://tangyidesign.com/tableau-airbnb-la-data-visualization-1.

[4] "Inside AirBnB's data exploration results"
http://insideairbnb.com/.

[5] "Kaggle website'
https://www.kaggle.com/.

[6] "New York City AirBnB's dataset'
https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data/.

[7] "Random Forest Regressor algorithm'
https://towardsdatascience.com/random-forest-and-its-implementation-71824ced454f.

[8] "Data-Driven Documents website'
https://d3js.org/.

[9] "PlotlyJS website'
https://plot.ly/javascript/.

[10] "Python website'
https://www.python.org/.

[11] "Pandas website'
https://pandas.pydata.org/.

[12] "Scikit-learn website'
https://scikit-learn.org/stable/.

[13] "jQuery website'
https://jquery.com/.

[14] "Google Colab'
https://colab.research.google.com/.

[15] "Hyperparameters tuning in RFR'
https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-

[16] "Lasso regression explained'
https://en.wikipedia.org/wiki/Lasso_(statistics).

[17] "Extreme Gradient Boosting documentation'
https://xgboost.readthedocs.io/en/latest/.