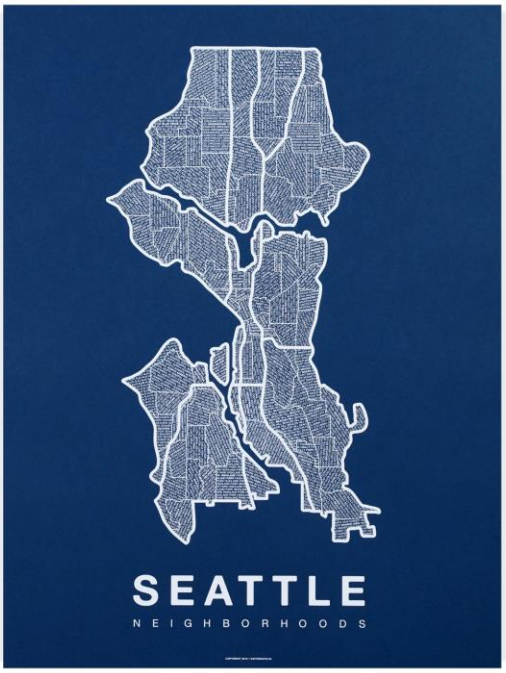
INSTITUTE OF BUSINESS ADMINISTRATION
KARACHI

Application Development Project Report

(Seattle AirBnB Rental Prices Analysis)



GROUP MEMBERS

S.NO	Name	ERP
1	Bilal Naseem	13216
2	Muhammad Salman Malik	27256

ACKNOWLEDGEMENT

We would like to thank Sir.Usman Ali, our instructor, for Application Development (Class:6097) and for his continuous support, patience, and motivation, and for unselfishly sharing his expertise from the initial to the final stage of this academic endeavor. We were able to understand and learn the fundamentals of Python from basic to advanced.

We are also grateful to our classmates and cohort members, for their editing help, honest feedback, and encouragement. Thanks should also go to the research assistants, and librarians from the university, who have been keeping us resourceful.

PROJECT GOALS AND OBJECTIVES

The Problem: Currently there is no convenient way for a new Airbnb host to decide the price of his or her listing. New hosts must often rely on the price of neighboring listings when deciding on the price of their own listing.

The Solution: A Predictive Price Modeling tool whereby a new host can enter all the relevant details such as location of the listing, listing properties etc. We would render a Choropleth Map using Dash which would be interactive and provide certain dropdown options to the user which would suggest the Price, Minimum stay and overall satisfaction/rating based on the relevant neighborhoods of Seattle. The Model would have previously been trained on similar data from already existing Airbnb listings.

TABLE OF CONTENTS

1 SYSTEM STUDY / DOMAIN ANALYSIS

1.1 Introduction and Scope

It was back in October of 2007 that Joe Gebbia and Brian Chesky decided to throw an air mattress in their living room and set up an "air bed and breakfast" for guests arriving in San Francisco for a major convention. From this lightbulb moment, the founders managed to attract investment and expand their vision to a global empire with a 2020 private valuation of USD $26 billion. Travelers now have the option of staying at short-term vacation rentals or "Airbnb's" in many parts of the world, creating a particularly efficient market for hosts who need their listings to be priced competitively. Major cosmopolitan cities like Seattle approaching nearly 10,000 listings will certainly create a demand for accurate price prediction and forecasting.

This analysis will highlight specific elements of the "Seattle Airbnb Listing '' Dataset from 1st Sep 2015 to 11th July 2017 which features 113676 rows and 19 columns that are allocated on 81 neighborhoods. The Data has been scraped and maintained by Tom Slee, and available for download on Airbnb Data Collection

We will begin by examining the unrefined data and perform any necessary tidying of the dataset. We will then determine which features provide most insight into price prediction and remove those that only serve to add noise. Data visualization and exploratory data analysis will then be performed on the tidied dataset to inform our modeling.

1.2 About Dataset

The Dataset can be classified as a Regression type dataset, as the main feature of interest price is a continuous variable.

The dataset has many features which can be categorized into following types:

- ➔ room_id: A unique number identifying an Airbnb listing.

- ➔ host_id: A unique number identifying an Airbnb host.
- ➔ room_type: One of “Entire home/apt”, “Private room”, or “Shared room”
- ➔ borough: A subregion of the city or search area for which the survey is carried out. The borough is taken from a shapefile of the city that is obtained independently of the Airbnb website. For some cities, there is no borough information; for others the borough may be a number.
- ➔ neighborhood: As with borough: a subregion of the city or search area for which the survey is carried out. For cities that have both, a neighborhood is smaller than a borough. For some cities there is no neighborhood information.
- ➔ reviews: The number of reviews that a listing has received. Airbnb has said that 70% of visits end up with a review, so the number of reviews can be used to estimate the number of visits.
- ➔ overall_satisfaction: The average rating (out of five) that the listing has received from those visitors who left a review.
- ➔ accommodates: The number of guests a listing can accommodate.
- ➔ bedrooms: The number of bedrooms a listing offers.
- ➔ price: The price (in $US) for a night stay. In early surveys, there may be some values that were recorded by month.
- ➔ minstay: The minimum stay for a visit, as posted by the host.
- ➔ latitude and longitude: The latitude and longitude of the listing as posted on the Airbnb site: this may be off by a few hundred meters.
- ➔ last_modified: the date and time that the values were read from the Airbnb website.

The first line of the CSV file holds the column headings which was a zip file containing the Airbnb data for Seattle over the span of 2 years. We used the glob module in Python which created a path to concat all the files and further pandas was used to create a DataFrame named 'seattle_df'.

Borough, survey_id, city, bathrooms, and location_id columns were dropped as they had >75% of null values and had low importance with regards to objective of the analysis. Our dataset can be considered as a balanced dataset.

The price of the listing will serve as labels for the regression task. The goal of this project would be to predict the price of the listings.

1.3 Data Information

Data columns (total 19 columns):

#	Column	Non-Null Count	Dtype
0	room_id	113676 non-null	int64
1	host_id	113667 non-null	float64
2	room_type	113662 non-null	object
3	borough	0 non-null	float64
4	neighborhood	113676 non-null	object
5	reviews	113676 non-null	int64
6	overall_satisfaction	98323 non-null	float64
7	accommodates	109557 non-null	float64
8	bedrooms	107956 non-null	float64
9	price	113676 non-null	float64
10	minstay	60784 non-null	float64

11	latitude	113676 non-null	float64
12	longitude	113676 non-null	float64
13	last_modified	113676 non-null	object
14	survey_id	24615 non-null	float64
15	country	0 non-null	float64
16	city	24615 non-null	object
17	bathrooms	0 non-null	float64
18	location	24615 non-null	object

2 DATA CLEANING

2.1 ANALYZING AND CHANGING DATA TYPES

	column_name	percent_missing
room_id	room_id	0.000000
host_id	host_id	0.007917
room_type	room_type	0.012316
borough	borough	100.000000
neighborhood	neighborhood	0.000000
reviews	reviews	0.000000
overall_satisfaction	overall_satisfaction	13.505929
accommodates	accommodates	3.623456
bedrooms	bedrooms	5.031845
price	price	0.000000
minstay	minstay	46.528731
latitude	latitude	0.000000
longitude	longitude	0.000000
last_modified	last_modified	0.000000
survey_id	survey_id	78.346353
country	country	100.000000
city	city	78.346353
bathrooms	bathrooms	100.000000
location	location	78.346353

We calculated the missing values of all the columns in the DataFrame as a percentage to make the analysis easier and to decide which columns/rows we had to drop or impute in order to make our Data Handling and Preprocessing easier.

2.2 Handling Missing Values

Since room_type & host_id have 0.00123% and 0.0079% of data missing which is very insignificant so we're dropping these rows entirely. Based on the above data, we are dropping columns: borough, bathrooms, location, city, survey_id since they are irrelevant for our analysis. Wherever minstay is null, we'll impute the mean of the minstay of that neighborhood in which the room belongs to and for bedrooms we imputed the mean of the corresponding neighborhood and room_type. For the imputation of the null values from the overall_satisfaction column we created two separate columns named 'cat_price' and 'cat_reviews' which discretize the data into categorical groups. 'cat_price' was discretized based on natural grouping and 'cat_reviews' was discretized based on quartiles as shown below:

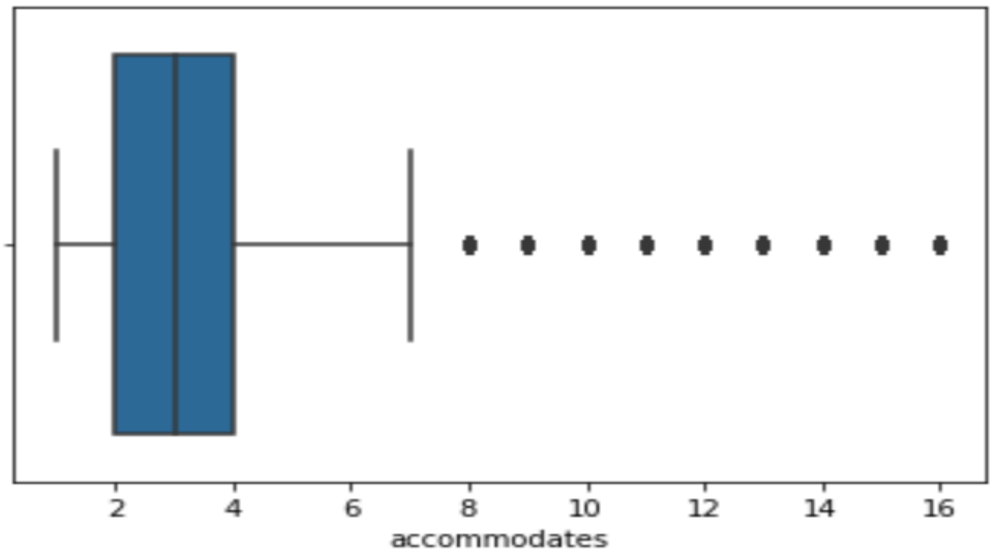
```
seattle_df["cat_price"].value_counts()
```

```
Normal       30918
High         28525
Very High    28303
low          25916
Name: cat_price, dtype: int64
```

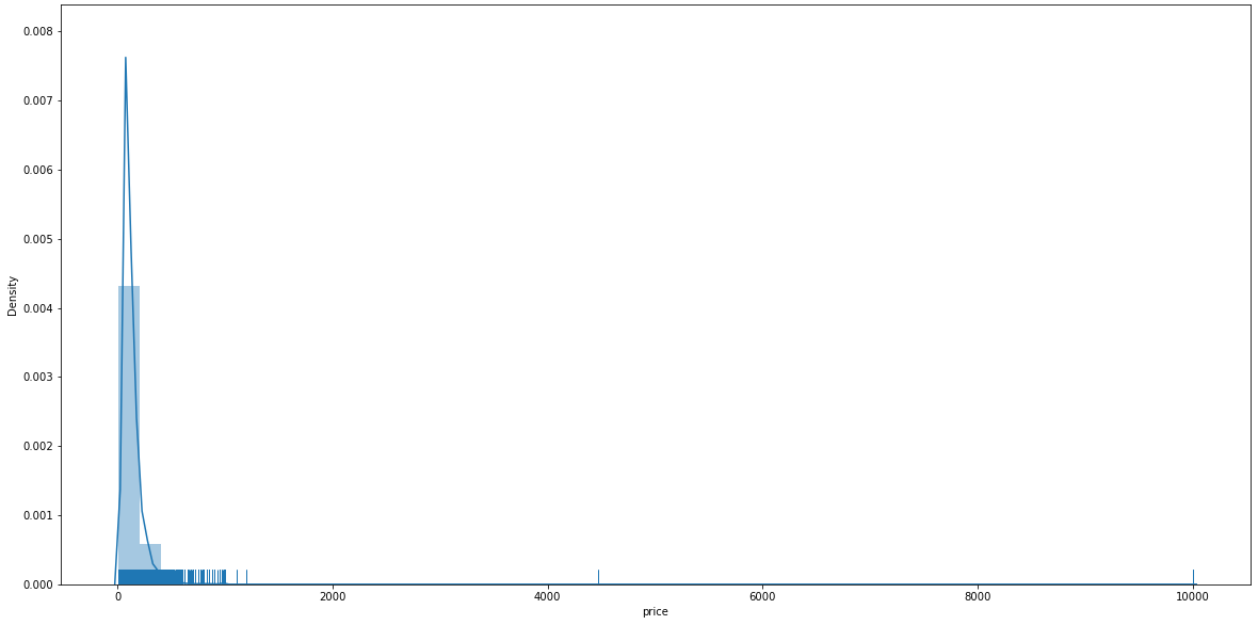
```
seattle_df['cat_reviews'].value_counts()
```

```
very less reviews                 31624
sufficient reviews                27599
low reviews                       26059
many reviews                      18074
extraordinary number of reviews   10306
Name: cat_reviews, dtype: int64
```

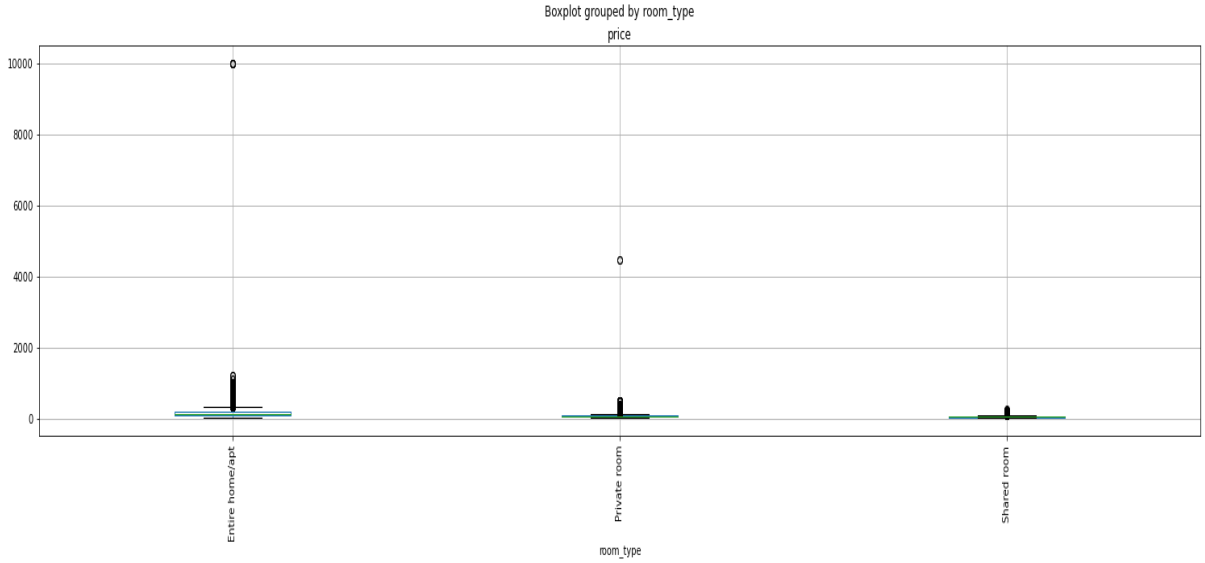
The overall_satisfaction column was grouped by the above two columns and the missing null values were transformed and replaced by the mean of the cat_price and cat_reviews in each row. Lastly, the accommodates column that showed the number of people that could be accommodated in room_type was grouped by room_type and bedrooms and the missing values were imputed based on the mean. We also plotted a box plot using seaborn to understand the spread of the data before we replaced the missing values as shown below:



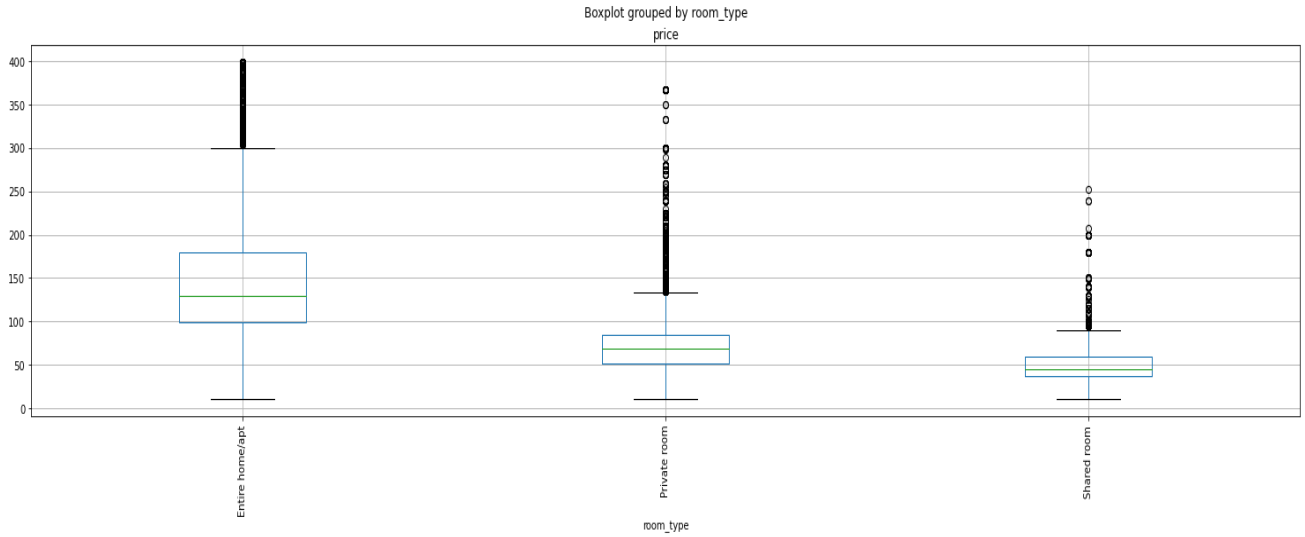
EXPLORATORY DATA ANALYSIS:



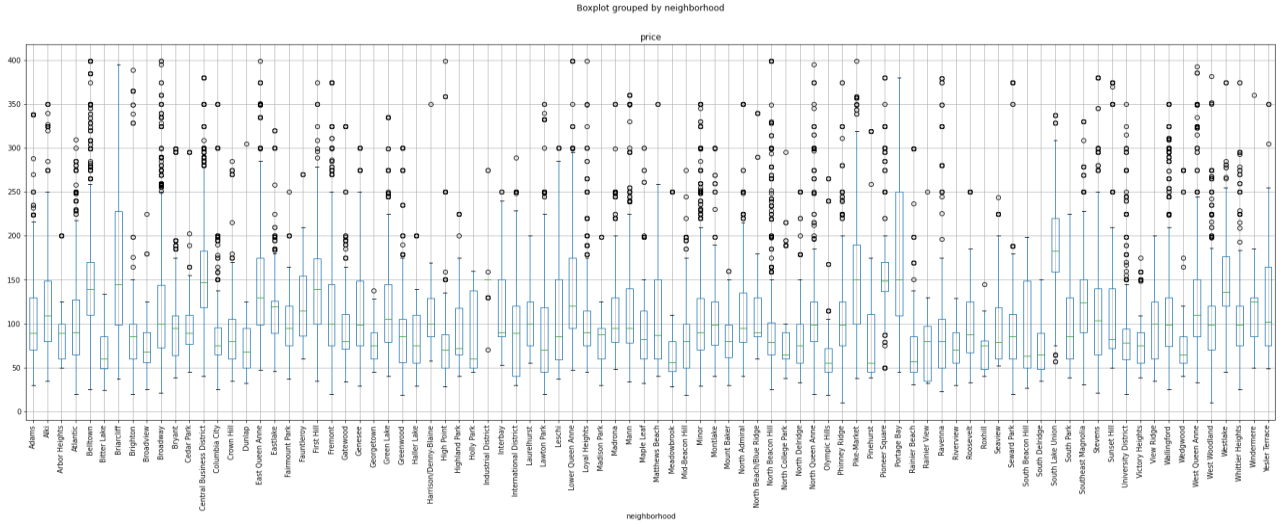
Kernel Distribution Estimation Plot which depicts the probability density function of the continuous or non-parametric data variables which in our case is price.



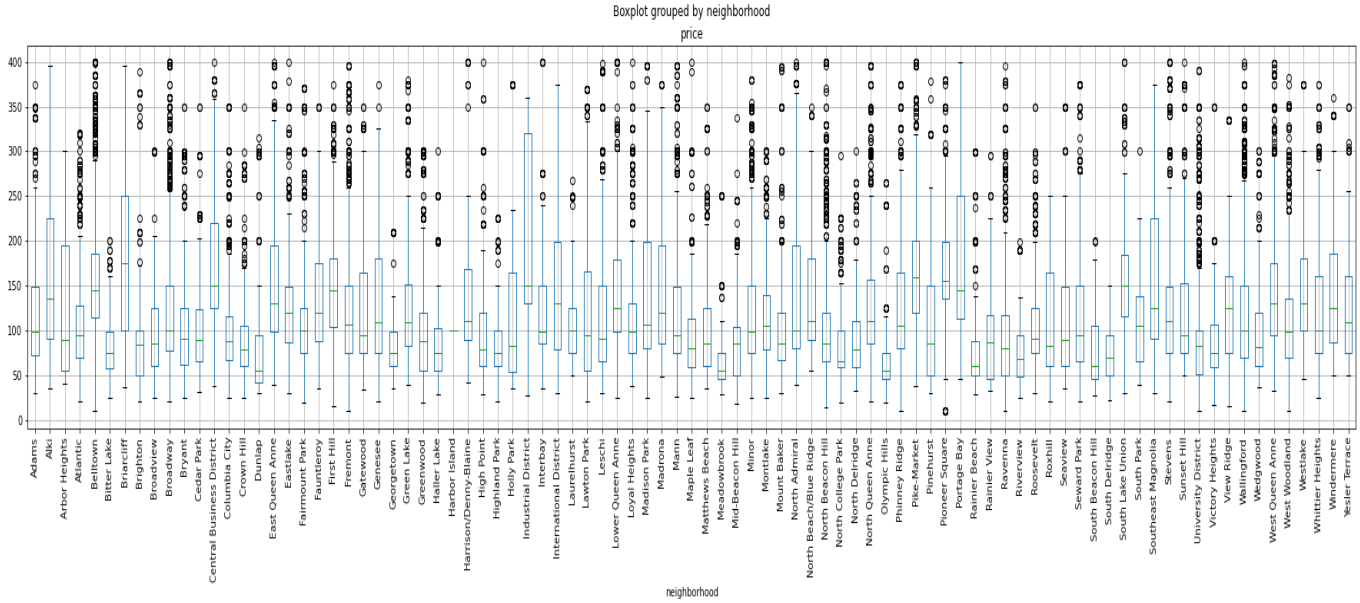
The following plot is a boxplot grouped by room_type which mainly shows that entire home/ apartment costs the most in comparison to private room and shared room (the cheapest).



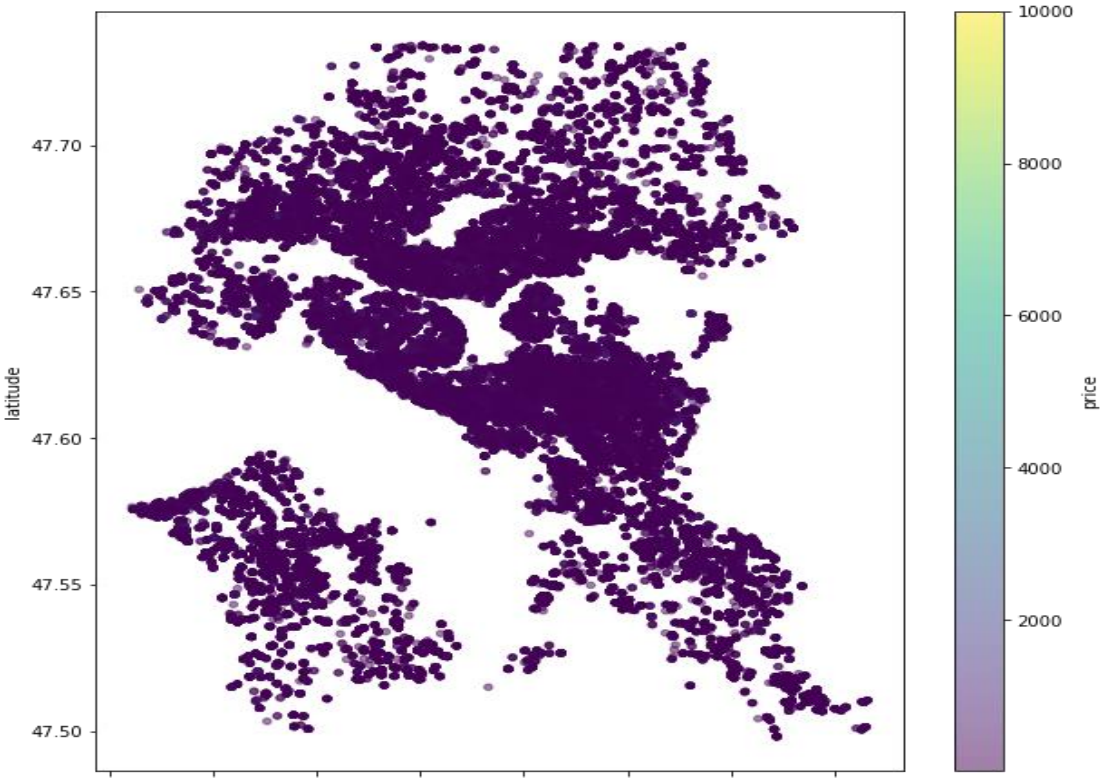
A similar boxplot but with better visualization after removal of outliers by setting price less than 400.



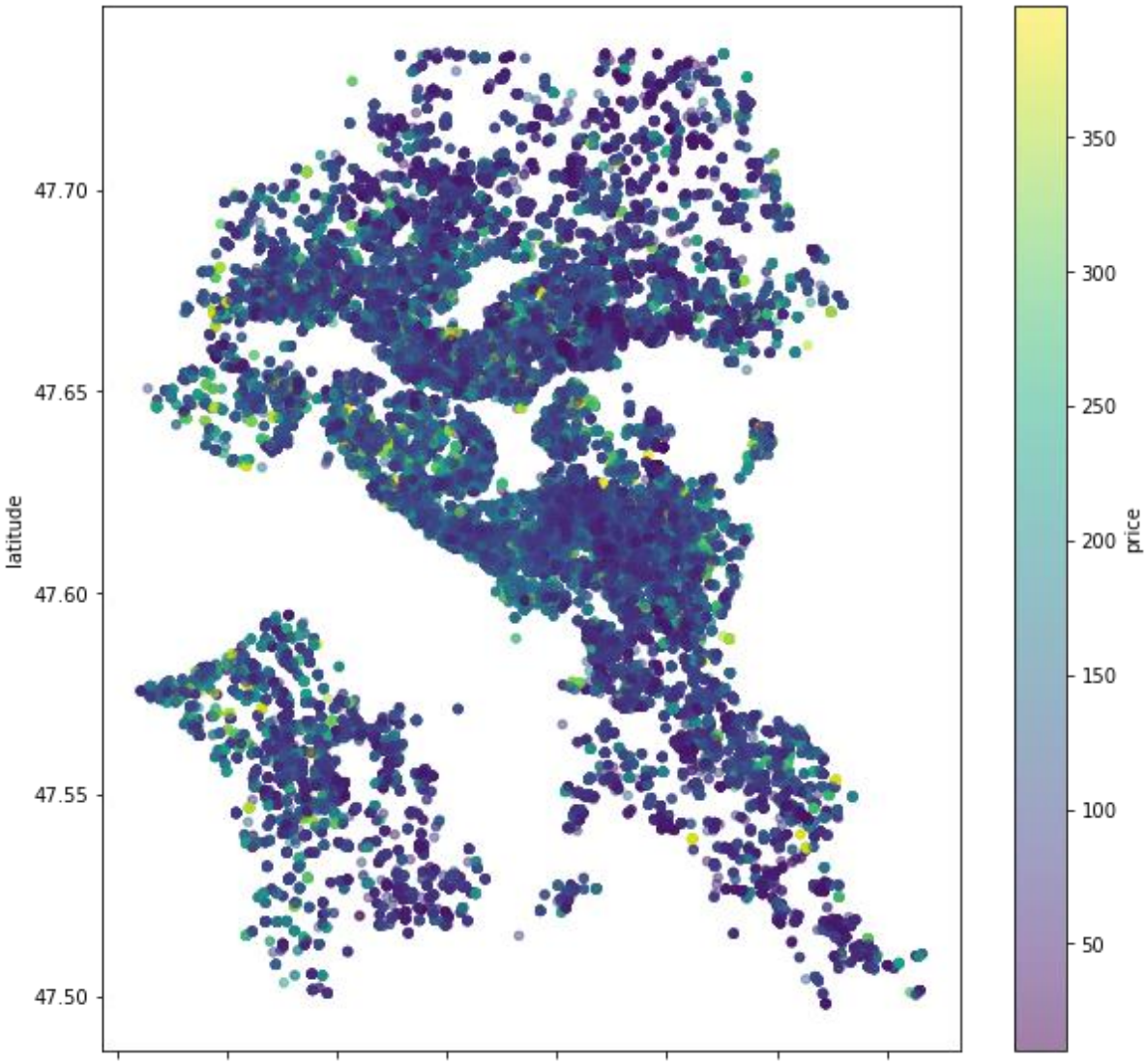
Boxplot as per each neighborhood which in our dataset equal to 81. The boxplots have been rotated by 90 for better visualization. It won't be wrong to depict that 'Briarcliff' is the most expensive neighborhood in Seattle.



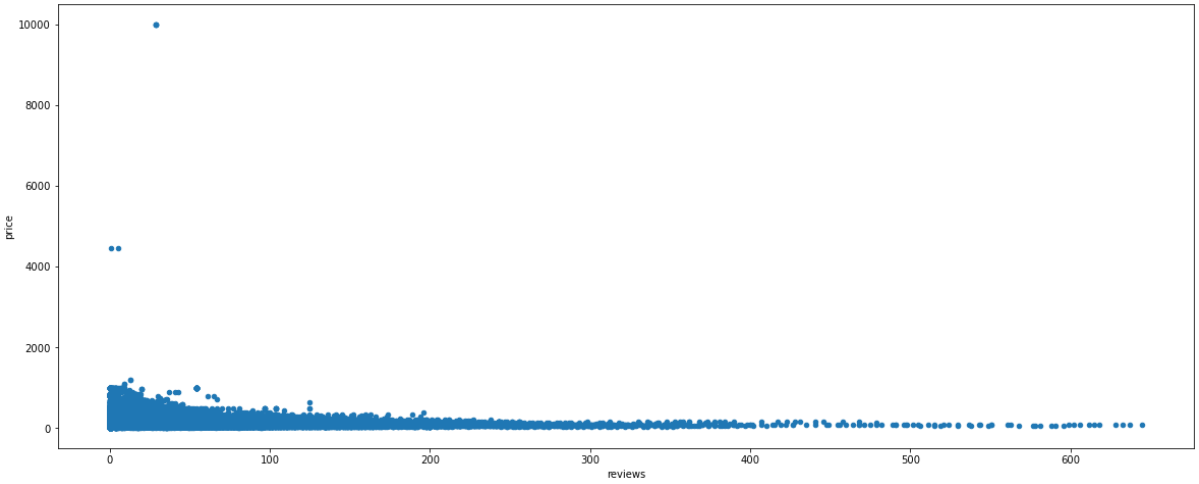
Similar vertical boxplot after removal of price outliers by setting price lower than 400 for better visualization.



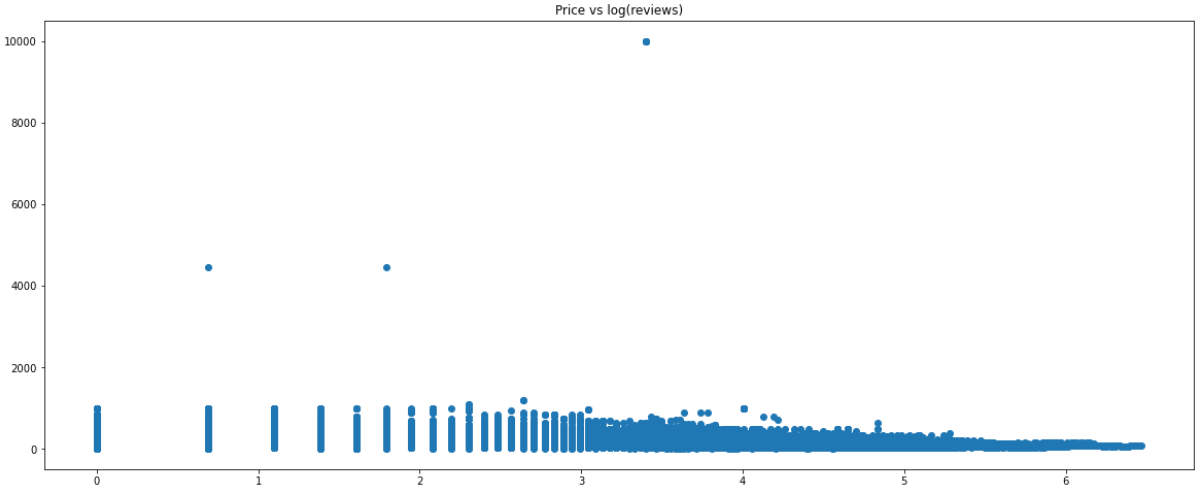
Latitude and Longitude coordinates were used to plot a scatter plot for each neighborhood.

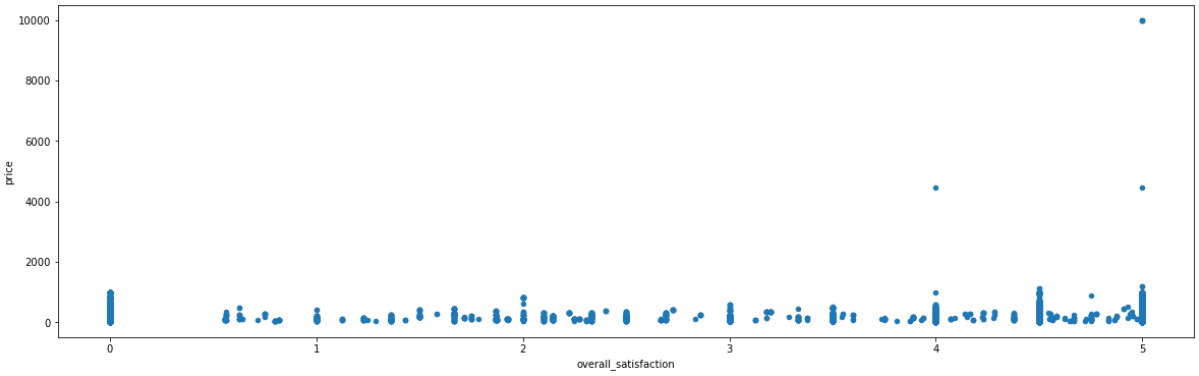


After removing outliers of price by setting price below 400 a better visualization of the scatter plot could be seen against the geometry coordinates as it correctly displays each point on the neighborhood according to the color scaler on the right of the plot.

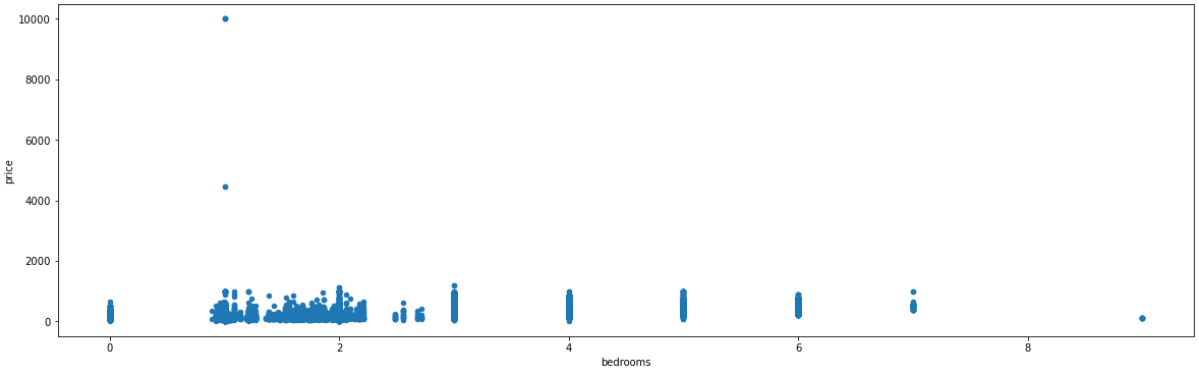


Reviews are plotted in a scatter plot but does not show accuracy as most of the reviews are skewed towards the left between 0 and 100. For better representation reviews were taken on a logarithmic scale and then plotted in the plot below so that the values lie on a closer magnitude.

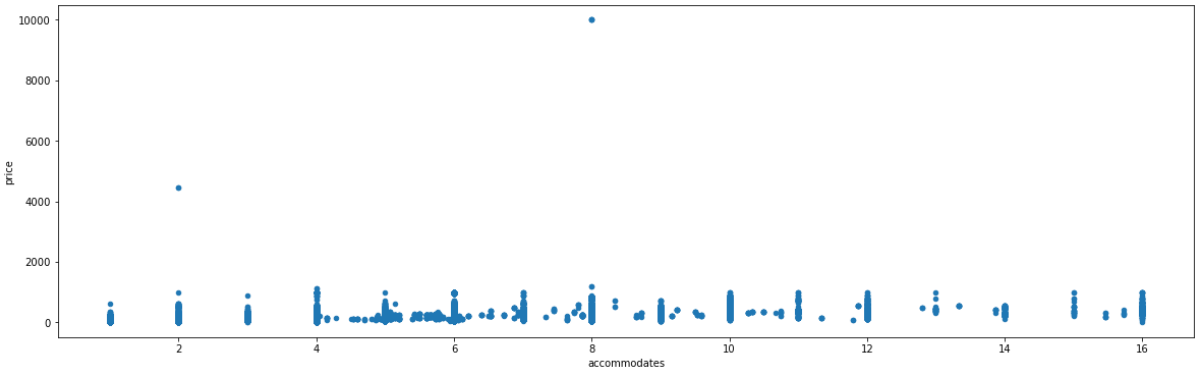




A scatter plot of overall_satisfaction against the price which shows that the satisfaction level for more expensive Airbnb's is much higher.



This scatter plot was made against the price and bedrooms. It doesn't talk much about the relation however it could be seen that 1- and 2-bedroom Airbnb's are rented out the most.



This scatter plot was made against the price and accommodates (number of people the Airbnb space can occupy).

Data preprocessing:

During this project we found out that properties with very few number of reviews are very difficult to predict. And we opted to fix the minimum number of reviews to 10 and similarly removed the outliers of price by fixing it to below 400.

keeping only data of properties having:

- minimum number of reviews = 10
- price < 400

For normalization of numerical columns we used the standard scaler to normalize the following column heads:

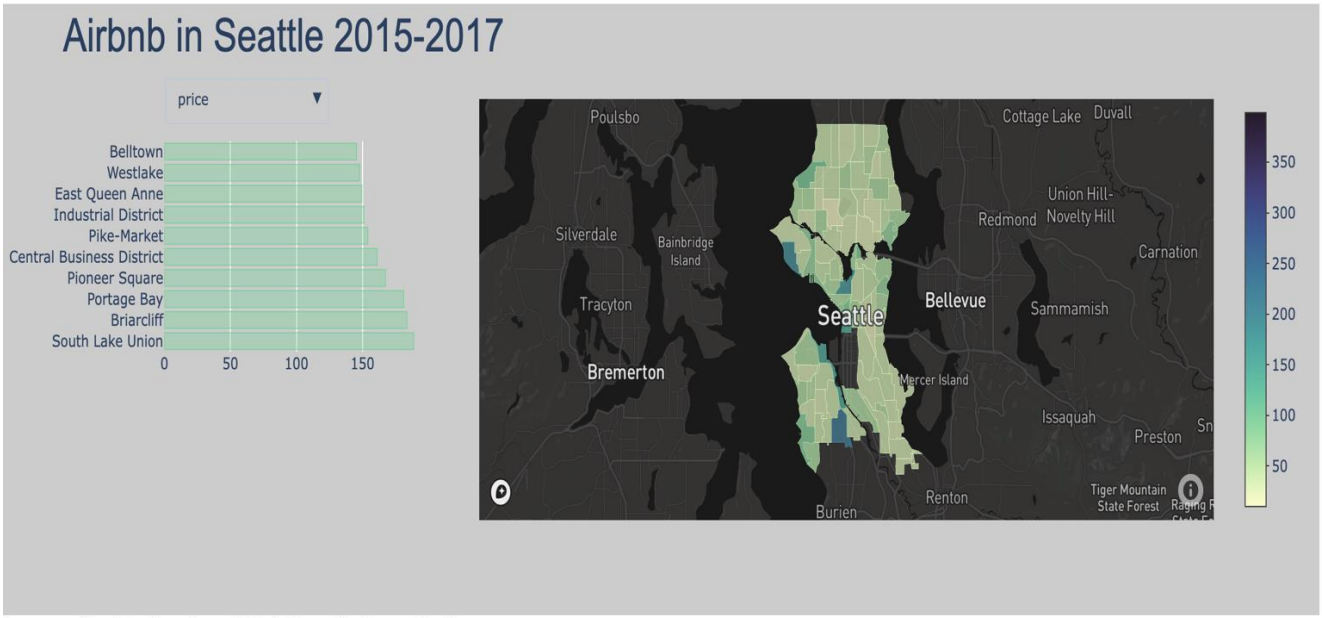
- reviews
- overall_satisfaction
- accommodates
- bedrooms
- price
- mainstay

For the train and test split a df_dummies, Data Frame was created and used, which was obtained by applying One Hot Encoding on the entire data set.

Since our data set had no class column and we wanted to test the data against price hence we s plit the data on price with a test size of 30%. We calculated median on our train data. If we just take the median value, as our baseline, we would say that an overnight stay in Seattle costs: 99. 0.

3 DASHBOARD

We Build an Interactive Choropleth Map with Plotly and Dash along with a bar plot using plotly.graph_objects as go. We scrapped the overall satisfaction, price, and minimum stay for each Airbnb, grouped by different neighborhoods in Seattle. The bar plot was used to show the top 10 neighborhoods with the highest prices, satisfaction, and duration of stay accordingly.



Data source from http://tomslee.net/airbnb-data-collection-get-the-data

The general principle in building this dashboard via plotly and dash is to:

1) arrange and combine different elements together on a defined canvas.

2) compile all elements in one container: `fig=go.Figure(data=trace2 + trace1, layout=layout)`.

3) pass this container to `dcc.Graph`, in which `dcc` is the dash core components, and `dash` will create a html-based web application for the dashboard.

A key function of this dashboard is to show a specific parameter (i.e. price, overall_satisfaction and minstay) in the two traces (choropleth map and bar plot) via the dropdown menu. My method is to create four layers with an argument visible=False for each trace. Then use the buttons feature of updatemenus to turn visible=True for a given parameter.

Thus, there are four figure layers in each trace and only one is visible depending on which button is clicked in each time point.

Since maps are not plotted on a cartesian system of coordinates (x/y), which is the coordinate system used in the bar plot, I set up two coordinate systems in the dashboard for the map and bar plot, respectively. As for the bar plot(trace2), its axes are assigned to xaxis='x2', yaxis='y2'. Instead, the map (trace1) has its own features within Layout, which was assigned to the variable mapbox1, the numbers following mapbox and x/y are arbitrary.

Then within the Layout settings, we do adjustments for these two coordinate systems individually. Such as the position of traces in the dashboard via domain, the appearance of ticks on each axis via showticklabels, and the ascending order of bar via autorange.

After concatenating all elements within fig=go.Figure, We assigned fig to dcc.Graph, wrapped up all codes as a py file, and run it.