# UNVEILING THE SECRETS OF AIRBNB IN NYC: DATA INSIGHTS

ABHIJITH CV

FEB-2023

# AGENDA

Objective

Data life cycle

Analysis methods

Recommendations

Appendix:

- Data sources
- Data methodology
- Data model assumptions

# OBJECTIVE

To Conduct a thorough analysis of New York Airbnb Dataset.

Ask effective questions that can lead to data insights

process,analyze and share findings by data visualization and statistical techniques

# DATA LIFE CYCLE

In the first phase the data  captured and loaded into various envirnoment.

Once data is cleaned, EDA is done and new features are created.

Then Meaningful insights are derived using various analytical methods.

# 1. Importing libraries and reading the data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
inp0 = pd.read_csv('AB_NYC_2019.csv')
inp0.head(5)
```

| | id | name | host_id | host_name | neighbourhood_group | neighbourhood | latitude | longitude | room_type | price | minimum_nights |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2539 | Clean & quiet apt home by the park | 2787 | John | Brooklyn | Kensington | 40.64749 | -73.97237 | Private room | 149 | 1 |
| **1** | 2595 | Skylit Midtown Castle | 2845 | Jennifer | Manhattan | Midtown | 40.75362 | -73.98377 | Entire home/apt | 225 | 1 |
| **2** | 3647 | THE VILLAGE OF HARLEM....NEW YORK ! | 4632 | Elisabeth | Manhattan | Harlem | 40.80902 | -73.94190 | Private room | 150 | 3 |
| **3** | 3831 | Cozy Entire Floor of Brownstone | 4869 | LisaRoxanne | Brooklyn | Clinton Hill | 40.68514 | -73.95976 | Entire home/apt | 89 | 1 |
| **4** | 5022 | Entire Apt: Spacious Studio/Loft by central park | 7192 | Laura | Manhattan | East Harlem | 40.79851 | -73.94399 | Entire home/apt | 80 | 10 |

# 2. Creating features

**2.1 categorizing the "availability_365" column into 5 categories**

```python
def availability_365_categories_function(row):
    """
    Categorizes the "minimum_nights" column into 5 categories
    """
    if row <= 1:
        return 'very Low'
    elif row <= 100:
        return 'Low'
    elif row <= 200 :
        return 'Medium'
    elif (row <= 300):
        return 'High'
    else:
        return 'very High'
```

**2.2 categorizing the "minimum_nights" column into 5 categories**

```python
def minimum_night_categories_function(row):
    """
    Categorizes the "minimum_nights" column into 5 categories
    """
    if row <= 1:
        return 'very Low'
    elif row <= 3:
        return 'Low'
    elif row <= 5 :
        return 'Medium'
    elif (row <= 7):
        return 'High'
    else:
        return 'very High'
```

**2.3 categorizing the "number_of_reviews" column into 5 categories**

```python
def number_of_reviews_categories_function(row):
    """
    Categorizes the "number_of_reviews" column into 5 categories
    """
    if row <= 1:
        return 'very Low'
    elif row <= 5:
        return 'Low'
    elif row <= 10 :
        return 'Medium'
    elif (row <= 30):
        return 'High'
    else:
        return 'very High'
```

**Note:** By categorizing, we are able to better understand relationships and connections between things and better communicate our findings.

# 3. Fixing columns

**Fix:** reviews_per_month is of object Dtype. datetime64 is a better Dtype for this column.

```
inp0.last_review = pd.to_datetime(inp0.last_review)
inp0.last_review
```

```
0       2018-10-19
1       2019-05-21
2              NaT
3       2019-05-07
4       2018-11-19
           ...
48890          NaT
48891          NaT
48892          NaT
48893          NaT
48894          NaT
Name: last_review, Length: 48895, dtype: datetime64[ns]
```

```
inp0.columns
```

```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365', 'availability_365_categories',
       'minimum_night_categories', 'number_of_reviews_categories',
       'price_categories'],
      dtype='object')
```

There are no more Dtypes to be fixed and data does not contain inconsistencies such as shifted columns, which is need to align correctly.The columns necessery for the futher analysis are also derived.

# 4. Data types

### 4.1 Categorical

```
inp0.columns
```

```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365', 'availability_365_categories',
       'minimum_night_categories', 'number_of_reviews_categories',
       'price_categories'],
      dtype='object')
```

```
# Categorical nominal
categorical_columns = inp0.columns[[0,1,3,4,5,8,16,17,18,19]]
categorical_columns
```

```
Index(['id', 'name', 'host_name', 'neighbourhood_group', 'neighbourhood',
       'room_type', 'availability_365_categories', 'minimum_night_categories',
       'number_of_reviews_categories', 'price_categories'],
      dtype='object')
```

### 4.2 Numerical

```
numerical_columns = inp0.columns[[9,10,11,13,14,15]]
numerical_columns
```

```
Index(['price', 'minimum_nights', 'number_of_reviews', 'reviews_per_month',
       'calculated_host_listings_count', 'availability_365'],
      dtype='object')
```

```
inp0[numerical_columns].describe()
```

| | price | minimum_nights | number_of_reviews | reviews_per_month | calculated_host_listings_count | availability_365 |
|---|---|---|---|---|---|---|
| count | 48895.000000 | 48895.000000 | 48895.000000 | 38843.000000 | 48895.000000 | 48895.000000 |
| mean | 152.720687 | 7.029962 | 23.274466 | 1.373221 | 7.143982 | 112.781327 |
| std | 240.154170 | 20.510550 | 44.550582 | 1.680442 | 32.952519 | 131.622289 |
| min | 0.000000 | 1.000000 | 0.000000 | 0.010000 | 1.000000 | 0.000000 |
| 25% | 69.000000 | 1.000000 | 1.000000 | 0.190000 | 1.000000 | 0.000000 |
| 50% | 106.000000 | 3.000000 | 5.000000 | 0.720000 | 1.000000 | 45.000000 |
| 75% | 175.000000 | 5.000000 | 24.000000 | 2.020000 | 2.000000 | 227.000000 |
| max | 10000.000000 | 1250.000000 | 629.000000 | 58.500000 | 327.000000 | 365.000000 |

### 4.3 Coordinates and date

```
coordinates = inp0.columns[[5,6,12]]
inp0[coordinates]
```

| | neighbourhood | latitude | last_review |
|---|---|---|---|
| 0 | Kensington | 40.64749 | 2018-10-19 |
| 1 | Midtown | 40.75362 | 2019-05-21 |
| 2 | Harlem | 40.80902 | NaT |
| 3 | Clinton Hill | 40.68514 | 2019-05-07 |
| 4 | East Harlem | 40.79851 | 2018-11-19 |
| ... | ... | ... | ... |
| 48890 | Bedford-Stuyvesant | 40.67853 | NaT |
| 48891 | Bushwick | 40.70184 | NaT |
| 48892 | Harlem | 40.81475 | NaT |
| 48893 | Hell's Kitchen | 40.75751 | NaT |
| 48894 | Hell's Kitchen | 40.76404 | NaT |

48895 rows × 3 columns

# 5. Missing values

```
# Percentage of missing values
round((inp0.isnull().sum()/len(inp0))*100,2)
```

```
id                                 0.00
name                               0.03
host_id                            0.00
host_name                          0.04
neighbourhood_group                0.00
neighbourhood                      0.00
latitude                           0.00
longitude                          0.00
room_type                          0.00
price                              0.00
minimum_nights                     0.00
number_of_reviews                  0.00
last_review                       20.56
reviews_per_month                 20.56
calculated_host_listings_count     0.00
availability_365                   0.00
availability_365_categories        0.00
minimum_night_categories           0.00
number_of_reviews_categories       0.00
price_categories                   0.00
dtype: float64
```

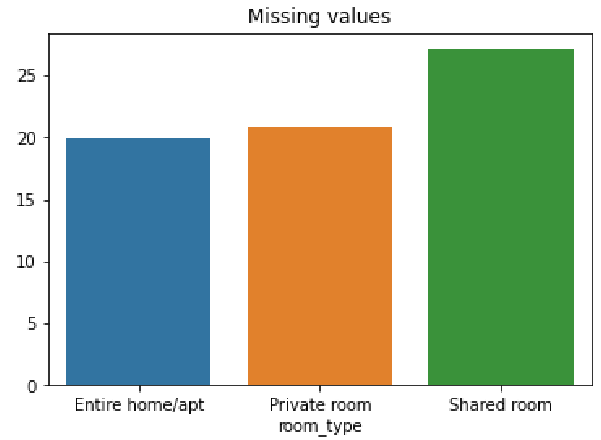
**- Two columns (last_review , reviews_per_month) has around 20.56% missing values. name and host_name has 0.3% and 0.4 % missing values**

**- We need to see if the values are, MCAR: It stands for Missing completely at random.**

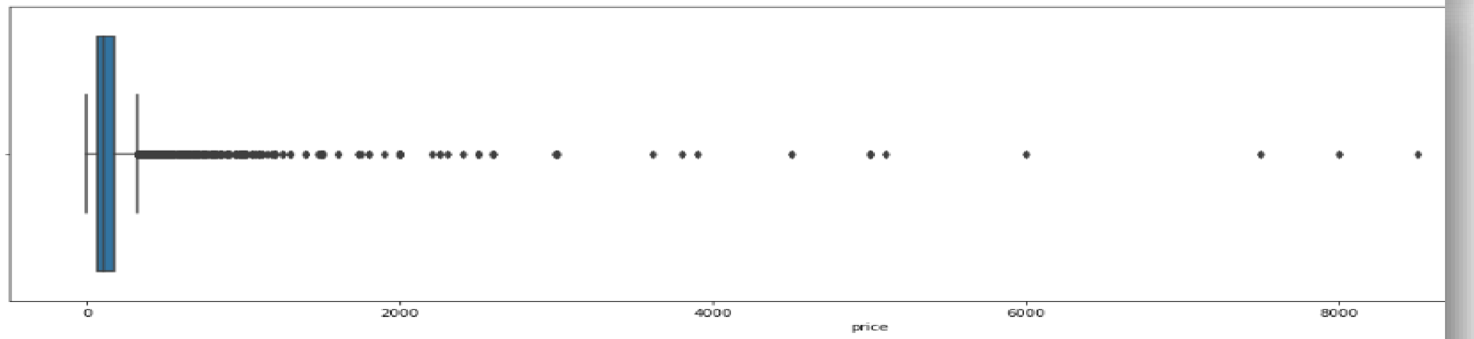
The reason behind the missing value is not dependent on any other features or if it is MNAR: It stands for Missing not at random. There is a specific reason behind the missing value.

**- There is no dropping or imputation of columns as we are just analyzing the dataset and not making a model.Also most of the features are important for our analysis.**

# 5.1 Missing value analysis




'Shared room' has the highest missing value percentage (27 %) for 'last_review' feature while to other room types has only about 20 %.




- The pricing is higher when 'last_review' feature is missing .
- reviews are less likely to be given for shared rooms
- When the prices are high reviews are less likely to be given
- The above analysis seems to show that the missing values here are not MCAR (missing completely at random)

# 6. Analysis

### 6.6 room_type

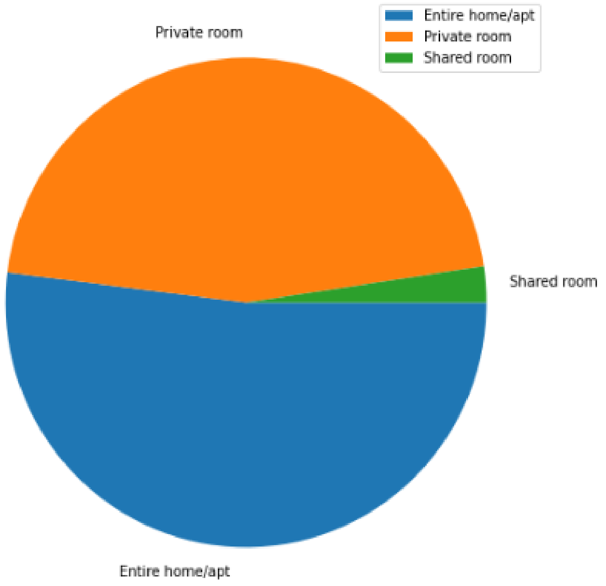
```
inp0.room_type.value_counts()
```

```
Entire home/apt    25409
Private room       22326
Shared room         1160
Name: room_type, dtype: int64
```

```
inp0.room_type.value_counts(normalize=True)*100
```

```
Entire home/apt    51.966459
Private room       45.661111
Shared room         2.372431
Name: room_type, dtype: float64
```

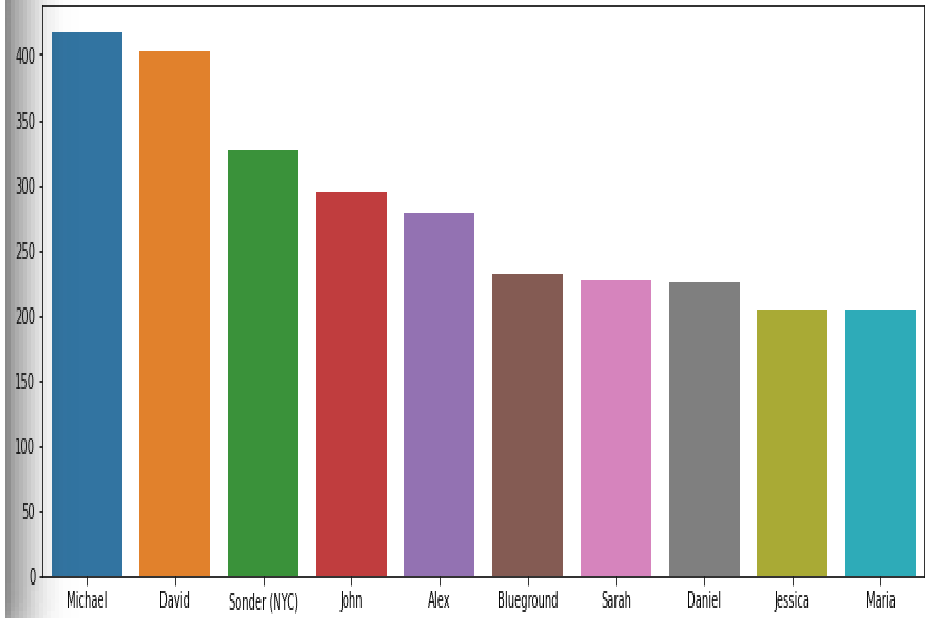
```
plt.figure(figsize=(8,8))
plt.pie(x = inp0.room_type.value_counts(normalize= True) * 100,labels = inp0.room_type.value_counts(normalize= True).
plt.legend()
plt.show()
```




### 6.3 host_name

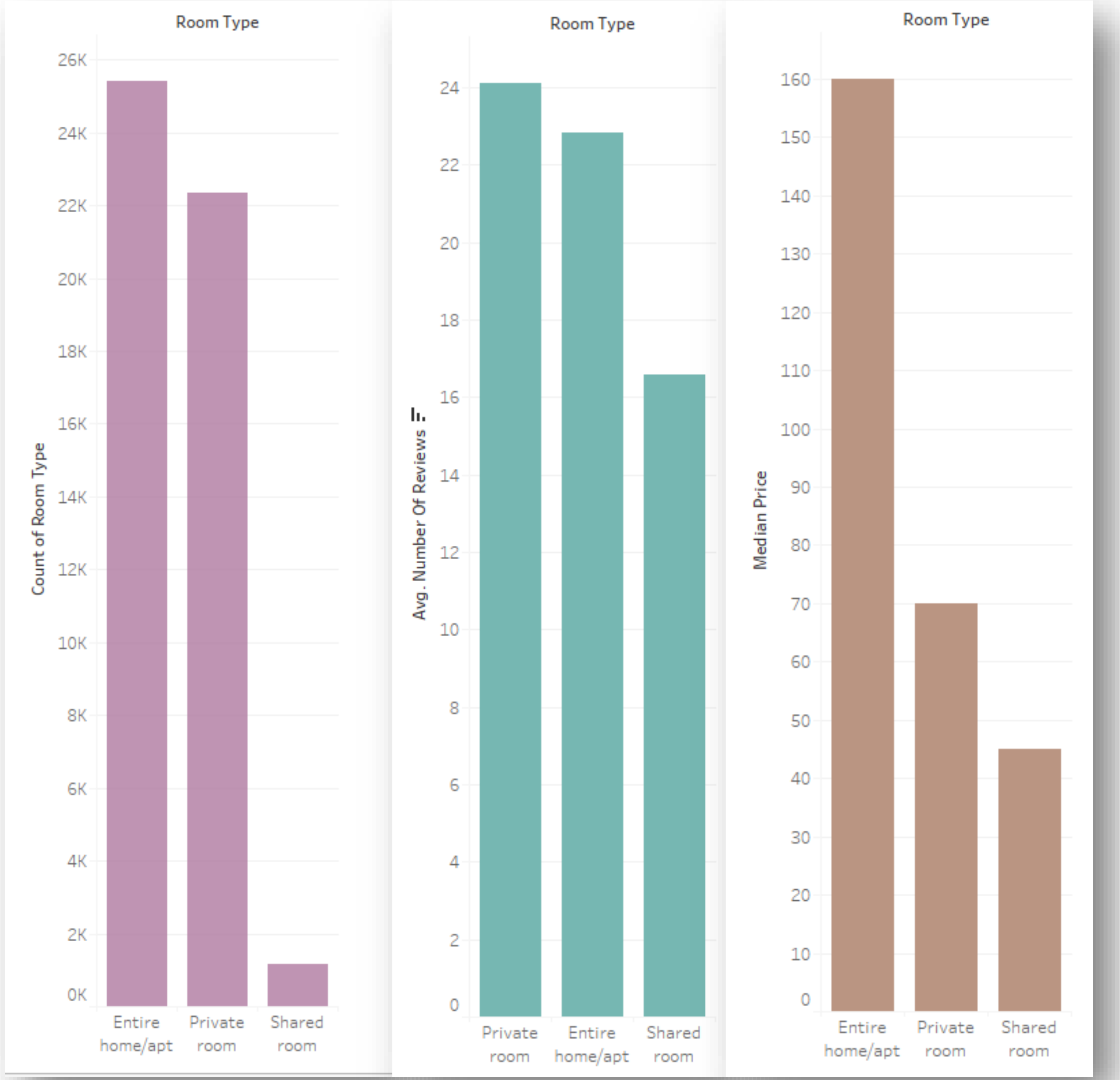
```
inp0.host_name.value_counts()
```

```
Michael               417
David                 403
Sonder (NYC)          327
John                  294
Alex                  279
                     ...
Rhonycs                 1
Brandy-Courtney         1
Shanthony               1
Aurore And Jamila       1
Ilgar & Aysel           1
Name: host_name, Length: 11452, dtype: int64
```

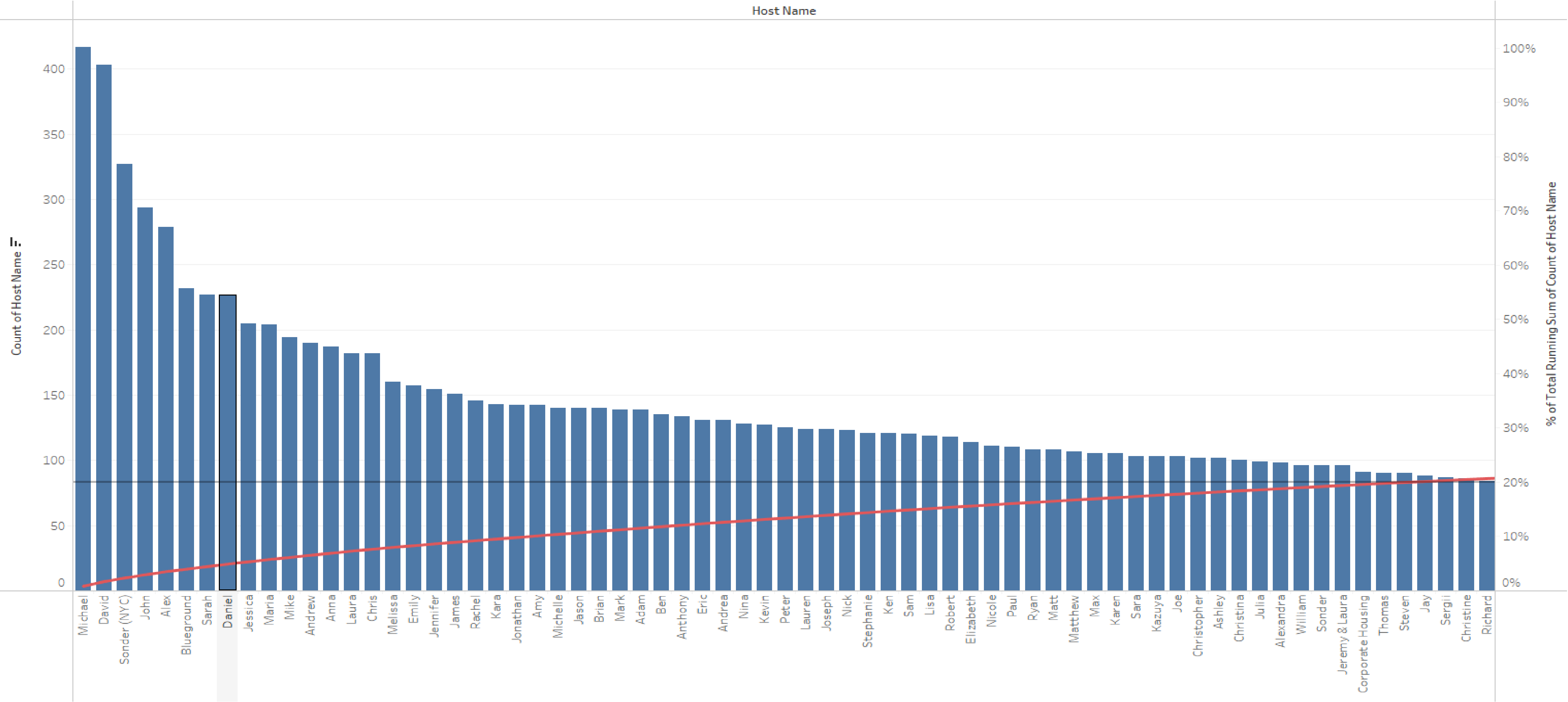


# THE PROBLEMS WITH SHARED ROOMS

- Shared rooms only account for 2 % of the total types of rooms.
- They are less likely to be reviewed.
- Median rates for shared rooms are significantly lower.

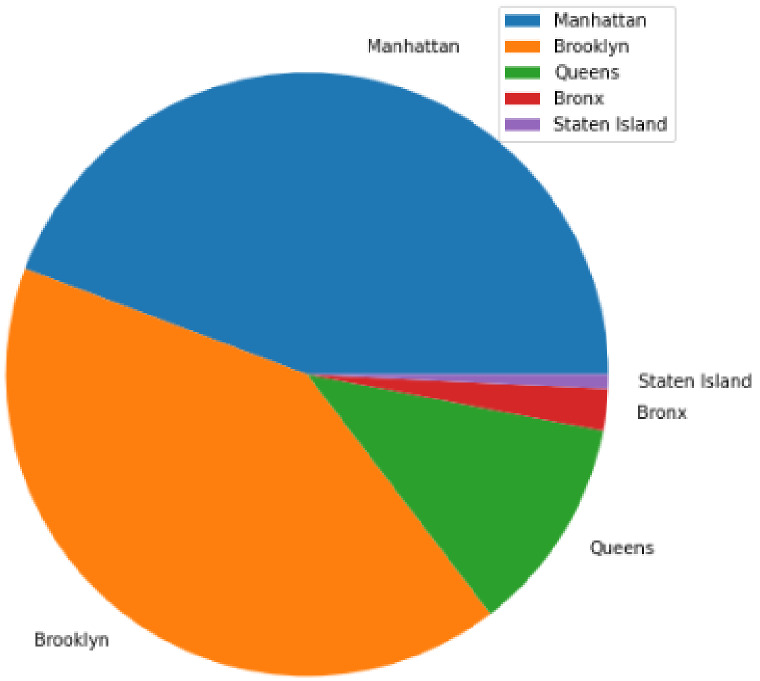


# EVERY HOST MATTER




- The top 60 hosts only make up 20% of the total host count!

# MOST CONTRIBUTING NEIGHBORHOODS

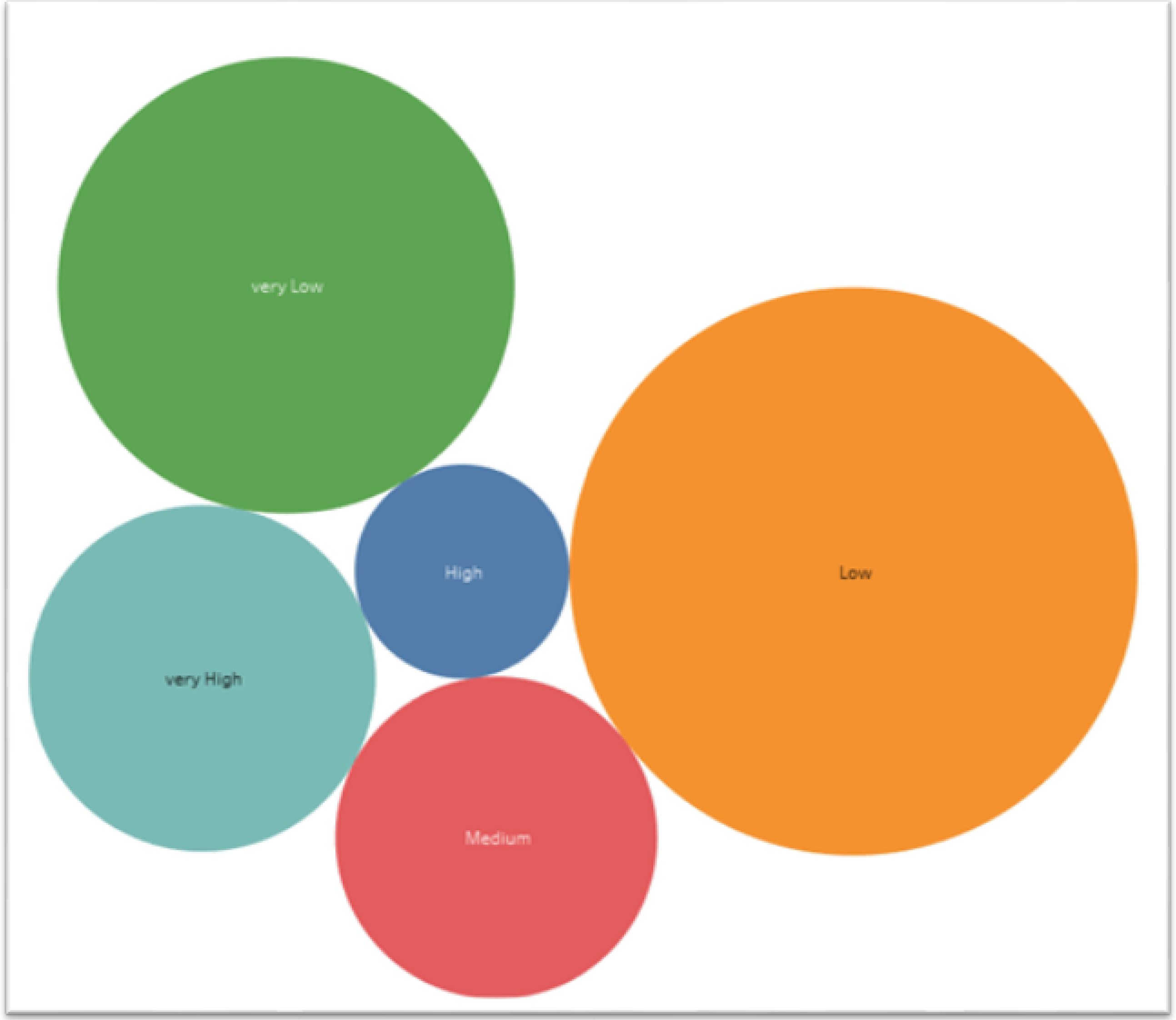



Neighborhood group percentages

| | |
|---|---|
| Manhattan | 44.301053 |
| Brooklyn | 41.116679 |
| Queens | 11.588097 |
| Bronx | 2.231312 |
| Staten Island | 0.762859 |

- 81 % of the listing are Manhattan and Brooklyn neighborhood group
- Staten Island has the lowest contribution.

# MINIMUM NIGHT CATEGORIES

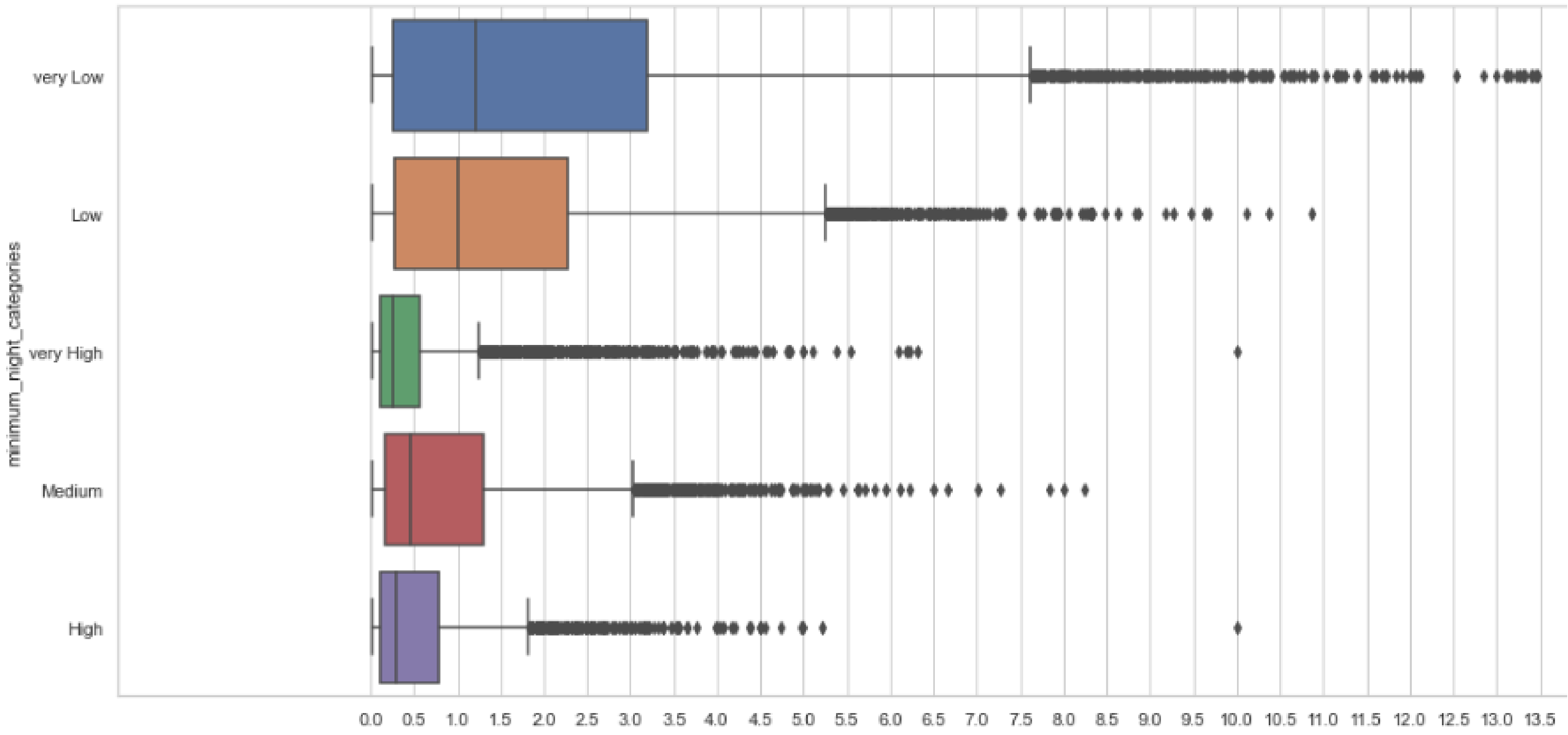



Minimum night category percentages

| | |
|---|---|
| Low | 40.280192 |
| very Low | 26.014930 |
| very High | 14.997444 |
| Medium | 12.960425 |
| High | 5.747009 |

- Low category in minimum night feature contributes 40 %

# EFFECT OF MINIMUM NIGHT ON REVIEWS




- Customers are more likely to leave reviews for lower number of minimum nights.

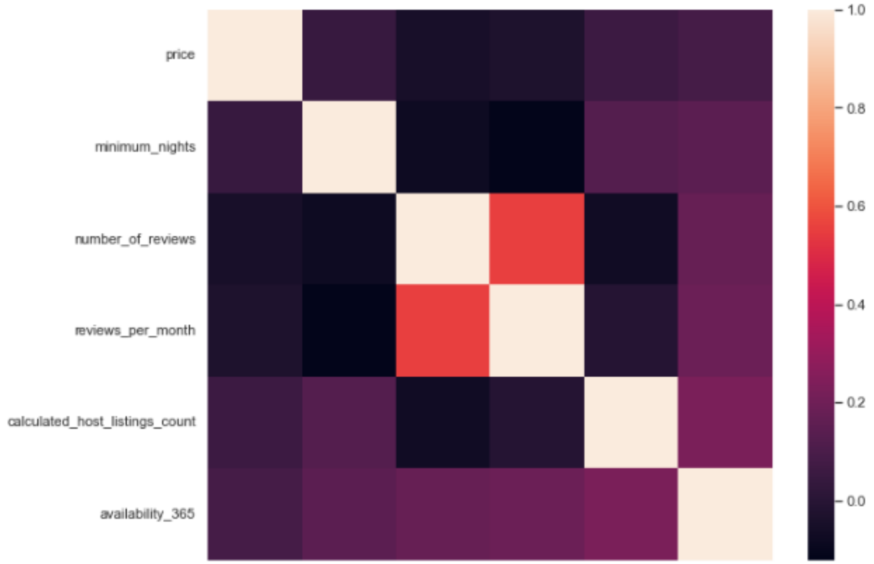
# 7. Bivariate and Multivariate Analysis

### 7.1 Finding the correalations

```
inp0[numerical_columns].corr()
```

| | price | minimum_nights | number_of_reviews | reviews_per_month | calculated_host_listings_count | availability_365 |
|---|---|---|---|---|---|---|
| **price** | 1.000000 | 0.042799 | -0.047954 | -0.030608 | 0.057472 | 0.081829 |
| **minimum_nights** | 0.042799 | 1.000000 | -0.080116 | -0.121702 | 0.127960 | 0.144303 |
| **number_of_reviews** | -0.047954 | -0.080116 | 1.000000 | 0.549868 | -0.072376 | 0.172028 |
| **reviews_per_month** | -0.030608 | -0.121702 | 0.549868 | 1.000000 | -0.009421 | 0.185791 |
| **calculated_host_listings_count** | 0.057472 | 0.127960 | -0.072376 | -0.009421 | 1.000000 | 0.225701 |
| **availability_365** | 0.081829 | 0.144303 | 0.172028 | 0.185791 | 0.225701 | 1.000000 |

```
plt.figure(figsize=(10,8))
sns.heatmap(data = inp0[numerical_columns].corr())
plt.show()
```



# CONCLUSION

- Strong significant insights are derived based on various attributes in the dataset.
- Ample amount and variety of visuals have can used in the presentations for the stake-holders.
- Data collection team should collect data about review scores so that it can strengthen the later analysis.
- A clustering machine learning model to identify groups of similar objects in datasets with two or more variable quantities can be made.

# APPENDIX - DATA SOURCES

The columns in the dataset are self-explanatory. You can refer to the diagram given below to get a better idea of what each column signifies.

| Column | Description |
|---|---|
| id | listing ID |
| name | name of the listing |
| host_id | host ID |
| host_name | name of the host |
| neighbourhood_group | location |
| neighbourhood | area |
| latitude | latitude coordinates |
| longitude | longitude coordinates |
| room_type | listing space type |
| price | |
| minimum_nights | amount of nights minimum |
| number_of_reviews | number of reviews |
| last_review | latest review |
| reviews_per_month | number of reviews per month |
| calculated_host_listings_count | amount of listing per host |
| availability_365 | number of days when listing is available for booking |

# APPENDIX –DATA METHODOLOGY

- Conducted a thorough analysis of NewYork Airbnbs Dataset.
- Cleaned the data set using python.
- Derived the necessary features.
- Used group aggregation, pivot table and other statistical methods.
- Created charts and visualizations using Tableau.

# APPENDIX - DATA ASSUMPTIONS

```
Categorical Variables:
     - room_type
     - neighbourhood_group
     - neighbourhood

Continous Variables(Numerical):
     - Price
     - minimum_nights
     - number_of_reviews
     - reviews_per_month
     - calculated_host_listings_count
     - availability_365
- Continous Variables could be binned in to groups too

Location Varibles:
     - latitude
     - longitude

Time Varibale:
     - last_review
```