Airbnb.com Case Study

code file link :

Dataset link :

❖ About Airbnb

Airbnb is an online platform that helps people rent out their homes or properties to guests, usually for short stays. It started in 2008 and gives hosts an easy way to make extra money by sharing their space. Guests often prefer Airbnb because it's usually cheaper and feels more like home compared to hotels. Airbnb makes money by charging a fee to both the host and the guest. With millions of listings in over 220 countries, Airbnb now also offers experiences like guided tours and activities to make travel more enjoyable.

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

❖ Tools/ Technologies used

python™

jupyter

Microsoft® Excel

Lets start . . .

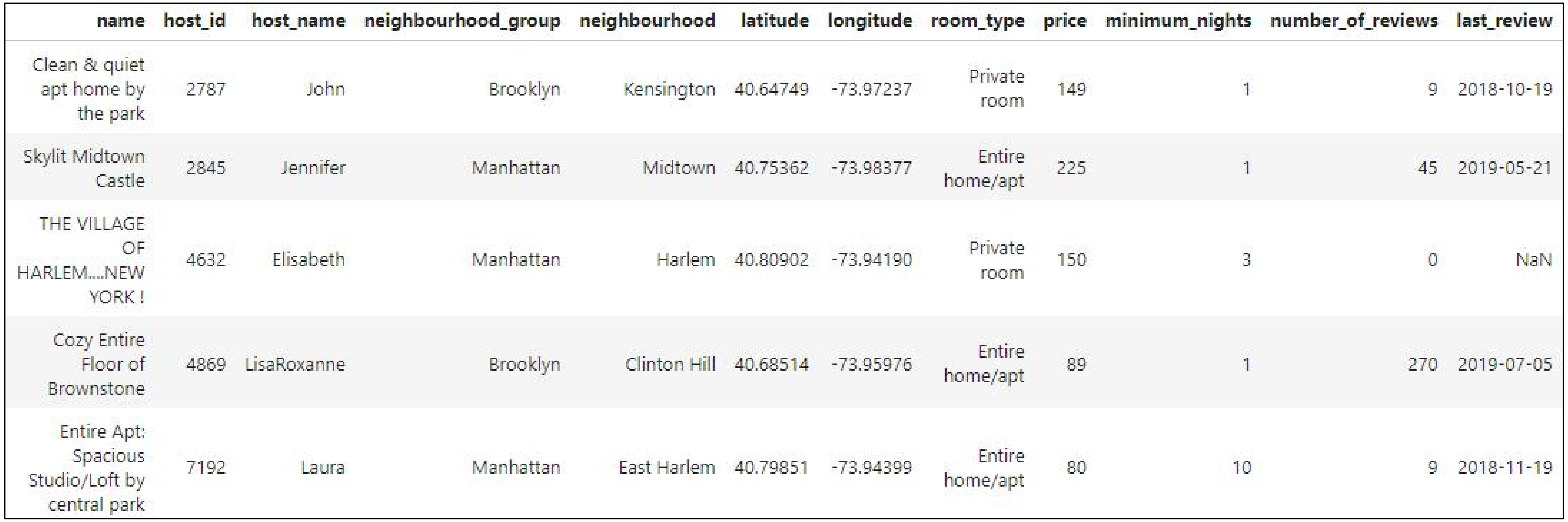
❖ Importing modules

```
#importing all required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

❖ Loading dataset

Code: df = pd.read_csv("D:\W\Practice datasets\Airbnb_data.csv")

Output:

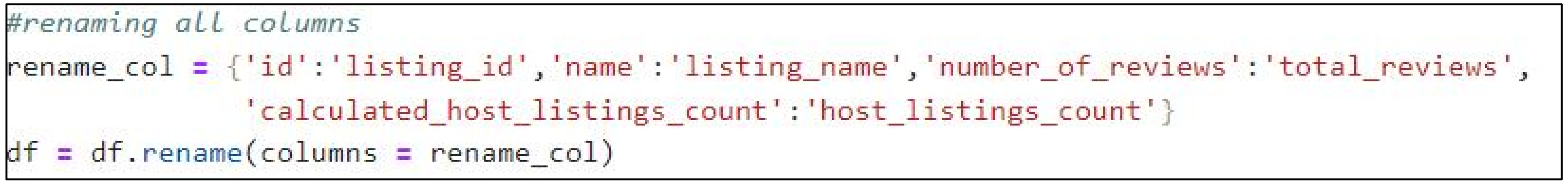


name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review
Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	1	9	2018-10-19
Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	1	45	2019-05-21
THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	0	NaN
Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	1	270	2019-07-05
Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	9	2018-11-19

❖ Data cleaning

- Renaming all columns

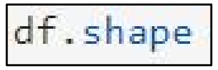
Code :



```
#renaming all columns
rename_col = {'id':'listing_id','name':'listing_name','number_of_reviews':'total_reviews',
              'calculated_host_listings_count':'host_listings_count'}
df = df.rename(columns = rename_col)
```

- Determining the shape of dataframe

Code :



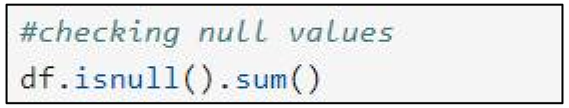
```
df.shape
```

Output :
(48895, 16)

So, there are 16 columns and 48,895 rows in a dataframe

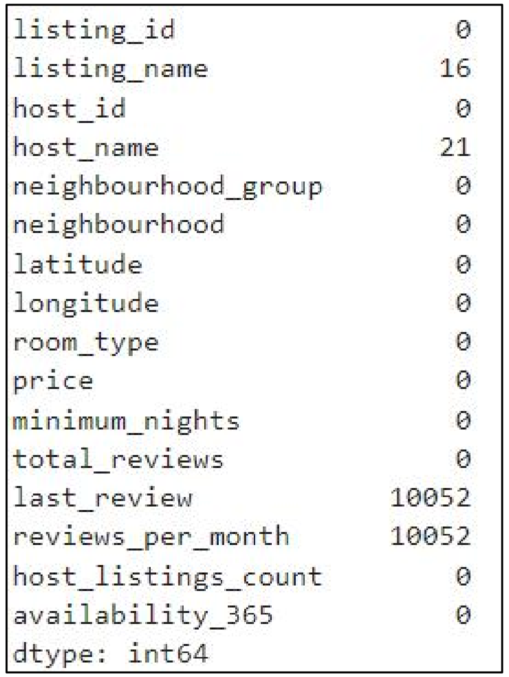
- Checking null values

Code:



```
#checking null values
df.isnull().sum()
```

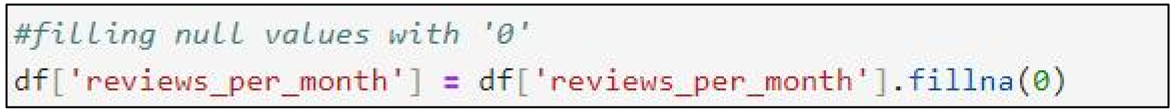
Output :



```
listing_id                    0
listing_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
total_reviews                 0
last_review               10052
reviews_per_month         10052
host_listings_count           0
availability_365              0
dtype: int64
```

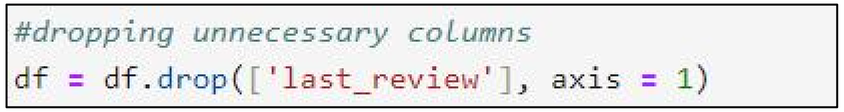
There are 16 null values in listing_name, 21 in Host_name, last_review and reviews per month has 10,052

- Filling na values with 0

Code: 

```
#filling null values with '0'
df['reviews_per_month'] = df['reviews_per_month'].fillna(0)
```

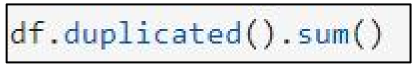
- Dropping the unneccesary columns

Code: 

```
#dropping unnecessary columns
df = df.drop(['last_review'], axis = 1)
```

Output: 0

- Checking for duplicates

Code: 

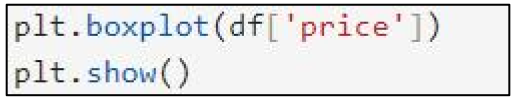
```
df.duplicated().sum()
```

Output: 0

❖ Exploratory data analysis (EDA)

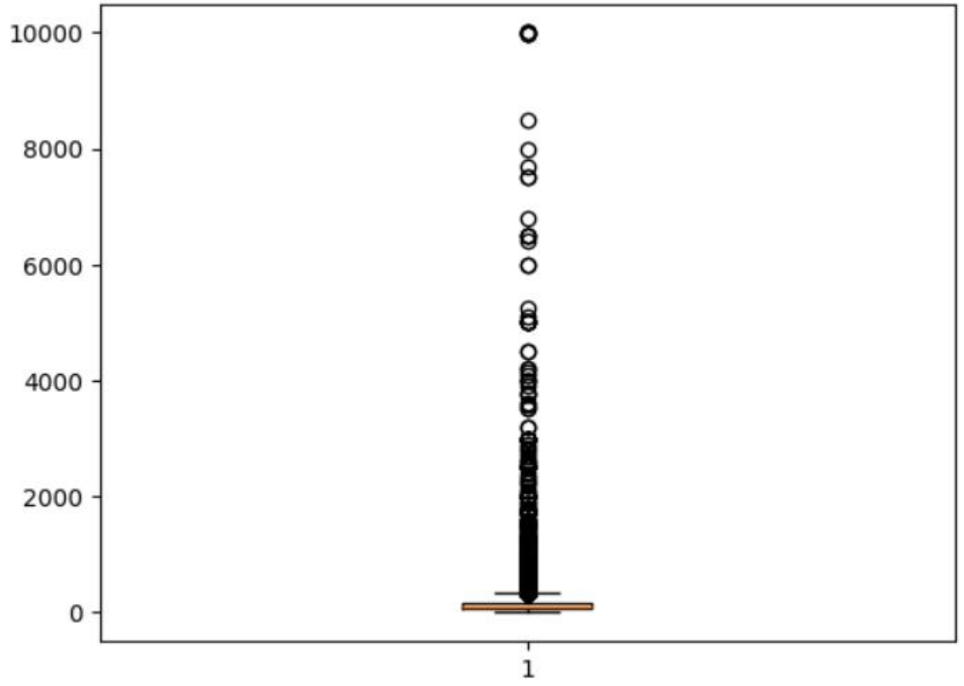
➢ Boxplot for ‘Price’

Code :



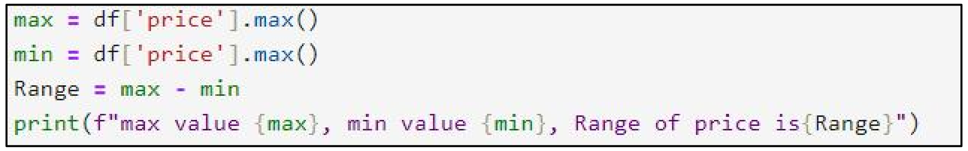
```
plt.boxplot(df['price'])
plt.show()
```

Output :



- The box plot shows that the dataset has a lot of unusual values, called outliers, which are represented by the points above the top line.
- Most of the data is close together in a small range, but there are some very large values going up to 10,000. This means the data is not evenly spread and is pulled up by these high numbers, making it look uneven.

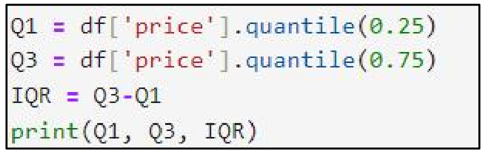
➢ Calculating range of price

Code: 

```
max = df['price'].max()
min = df['price'].max()
Range = max - min
print(f"max value {max}, min value {min}, Range of price is{Range}")
```

Output : max value 10000, min value 10000, Range of price is 0

➢ Interquartile range of price

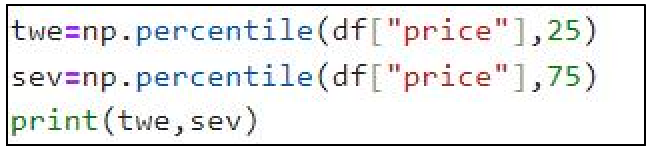
Code : 

```
Q1 = df['price'].quantile(0.25)
Q3 = df['price'].quantile(0.75)
IQR = Q3-Q1
print(Q1, Q3, IQR)
```

Output: 69.0 175.0 106.0

The output shows that 25% of prices are below 69 and 75% are below 175, with an interquartile range (IQR) of 106. The IQR represents the spread of the middle 50% of the data.

➢ Percentile value for price

Code: 

```
twe=np.percentile(df["price"],25)
sev=np.percentile(df["price"],75)
print(twe,sev)
```

Output: 69.0 175.0

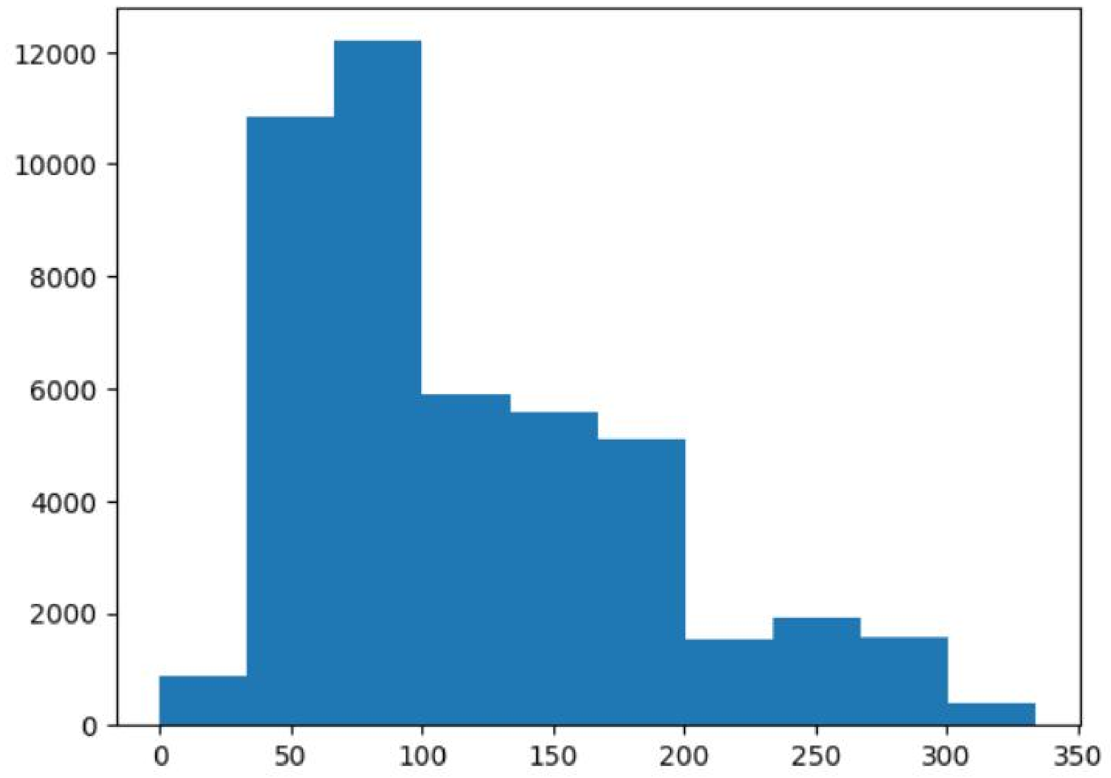
The output shows that 25% of the prices are below 69 and 75% are below 175. These values represent the 25th and 75th percentiles of the price column.

➢ Plotting Histogram

code:

```
plt.hist(df['price'])
plt.show()
```

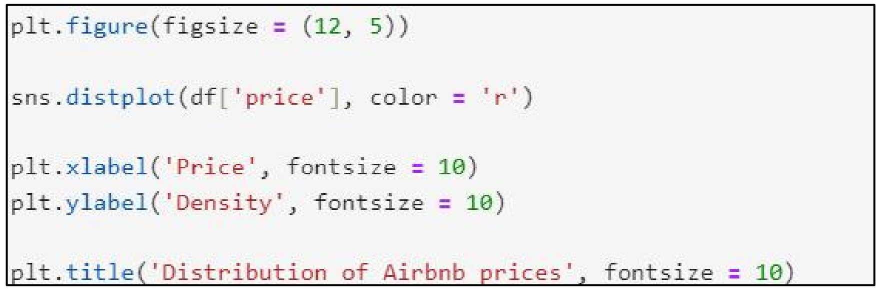
Output:



- This histogram represents the frequency distribution of a dataset, with the x-axis depicting the range of values from 0 to 350 and the y-axis showing the corresponding frequency counts.
- The distribution exhibits a right-skew, with the highest frequency concentrated between 50 and 100, and a gradual decrease in frequency as the values increase.
- The histogram reflects an asymmetric distribution, tapering off towards the higher value range.

➢ Distribution of Airbnb prices

code:



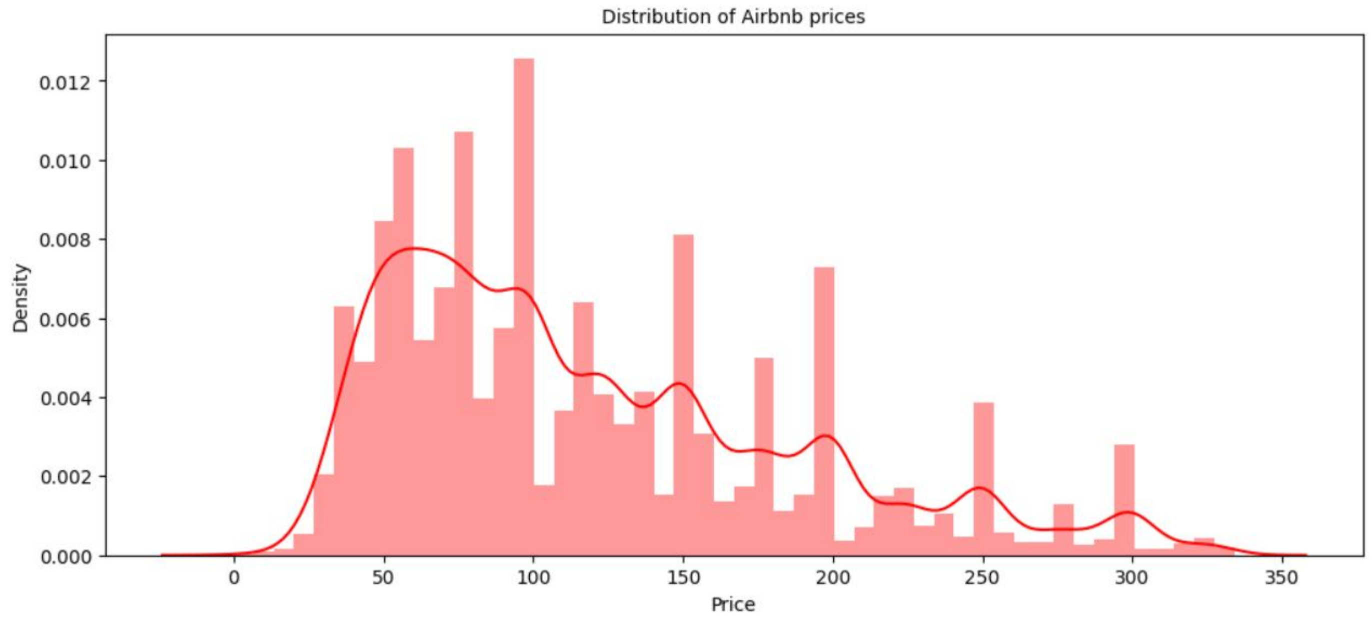
```
plt.figure(figsize = (12, 5))

sns.distplot(df['price'], color = 'r')

plt.xlabel('Price', fontsize = 10)
plt.ylabel('Density', fontsize = 10)

plt.title('Distribution of Airbnb prices', fontsize = 10)
```

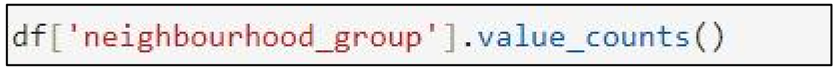
Output:



- The image shows the distribution of Airbnb prices using a histogram with a kernel density estimate (KDE) overlaid.
- The red-shaded histogram represents the frequency of listings within specific price ranges, while the smooth red KDE line provides a continuous estimate of the price distribution. The data is right-skewed, meaning most listings are priced below $150, with the highest concentration between $50 and $100.
- The KDE curve helps visualize the overall price trends by smoothing the fluctuations in the histogram.

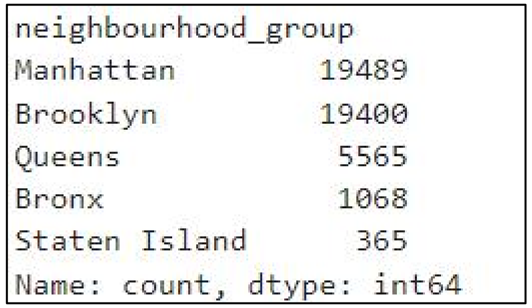
- Count of neighbourhood group

  Code:



```
df['neighbourhood_group'].value_counts()
```

  Output:

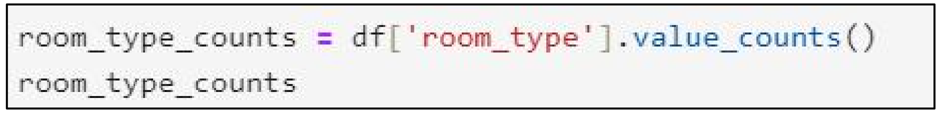


```
neighbourhood_group
Manhattan        19489
Brooklyn         19400
Queens            5565
Bronx             1068
Staten Island      365
Name: count, dtype: int64
```

  Output gives us the value counts for each neighbourhood group

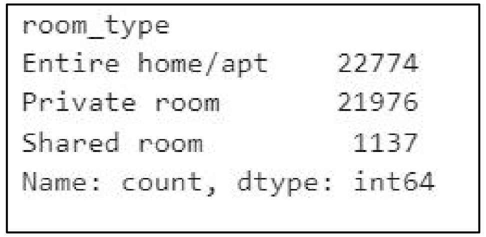
- Count of room_types

  code :



```
room_type_counts = df['room_type'].value_counts()
room_type_counts
```

  Output :

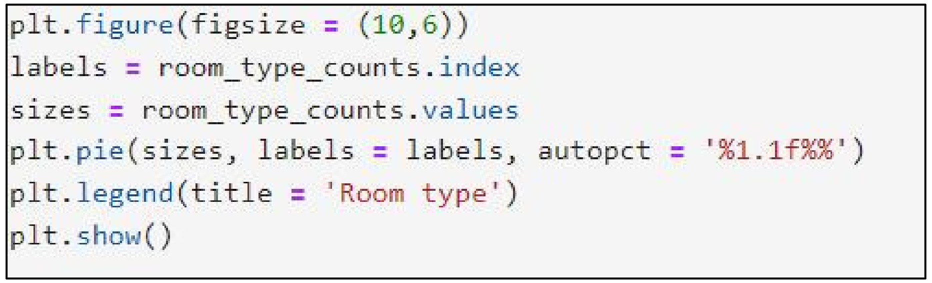


```
room_type
Entire home/apt    22774
Private room       21976
Shared room         1137
Name: count, dtype: int64
```

Output gives us the value counts for each room type

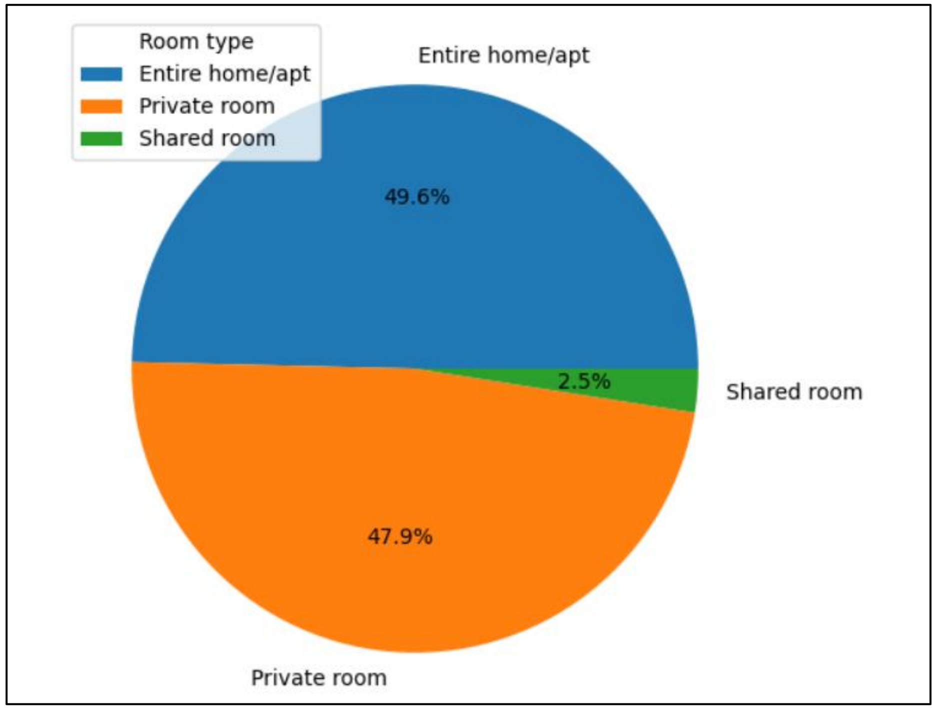
➢ Proportion of room types by category

Code:



```
plt.figure(figsize = (10,6))
labels = room_type_counts.index
sizes = room_type_counts.values
plt.pie(sizes, labels = labels, autopct = '%1.1f%%')
plt.legend(title = 'Room type')
plt.show()
```

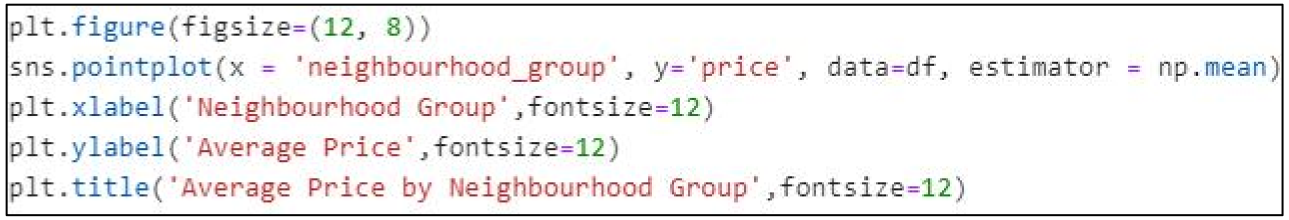
Output:



- This pie chart shows the proportion occupied by the each room type.
- Entire home/apt has the majority proportion of 49.6%
- Private room has 47.9% proportion and the Shared room type has 2.5% proportion.

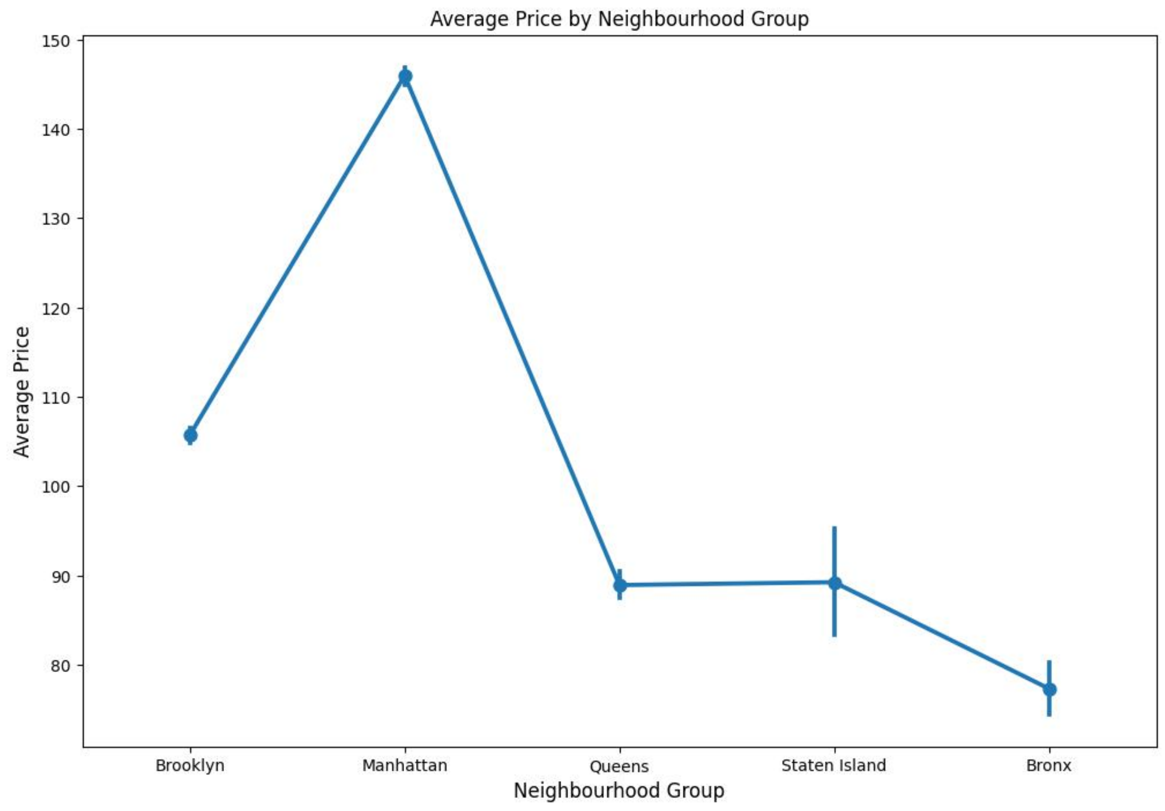
➢ Average price by neighbourhood group

code :



```
plt.figure(figsize=(12, 8))
sns.pointplot(x = 'neighbourhood_group', y='price', data=df, estimator = np.mean)
plt.xlabel('Neighbourhood Group',fontsize=12)
plt.ylabel('Average Price',fontsize=12)
plt.title('Average Price by Neighbourhood Group',fontsize=12)
```

Output:



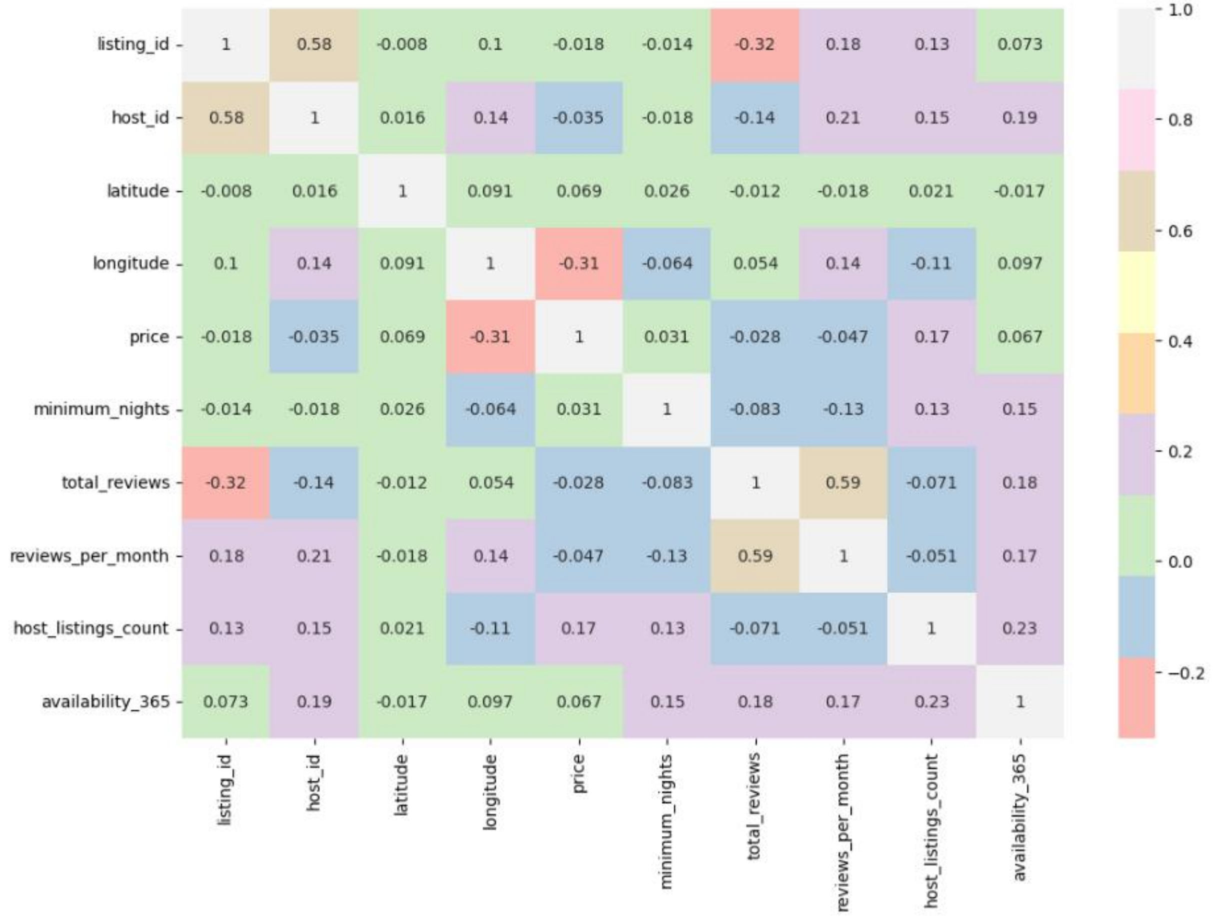
- By looking at the graph, we can clearly coclude that Manhattan has highest average price of 148 and lowest is for Bronx 76.
- Brooklyn, Queens and Staten Island has the average price value 104, 88 and 90 respectively.

❖ Correlation analysis

- Code:

```
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(numeric_only=True),annot=True,cmap='Pastel1')
```

- Output:



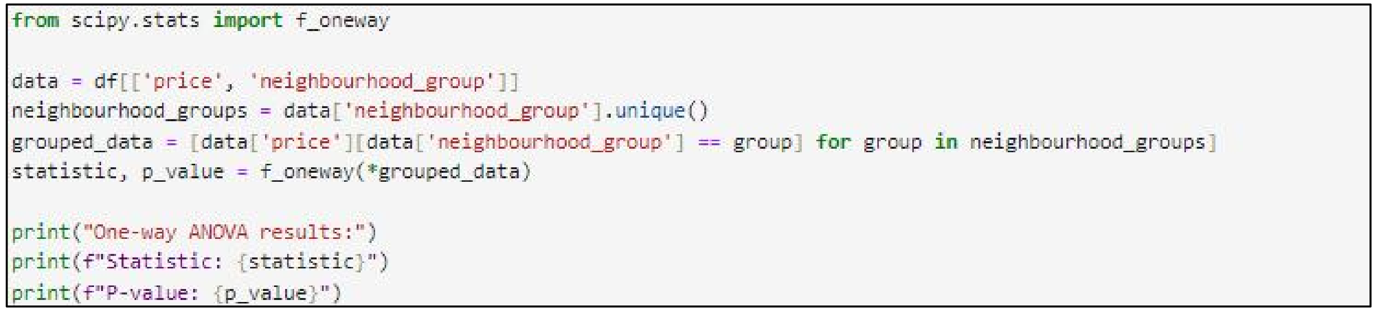
	listing_id	host_id	latitude	longitude	price	minimum_nights	total_reviews	reviews_per_month	host_listings_count	availability_365
listing_id	1	0.58	-0.008	0.1	-0.018	-0.014	-0.32	0.18	0.13	0.073
host_id	0.58	1	0.016	0.14	-0.035	-0.018	-0.14	0.21	0.15	0.19
latitude	-0.008	0.016	1	0.091	0.069	0.026	-0.012	-0.018	0.021	-0.017
longitude	0.1	0.14	0.091	1	-0.31	-0.064	0.054	0.14	-0.11	0.097
price	-0.018	-0.035	0.069	-0.31	1	0.031	-0.028	-0.047	0.17	0.067
minimum_nights	-0.014	-0.018	0.026	-0.064	0.031	1	-0.083	-0.13	0.13	0.15
total_reviews	-0.32	-0.14	-0.012	0.054	-0.028	-0.083	1	0.59	-0.071	0.18
reviews_per_month	0.18	0.21	-0.018	0.14	-0.047	-0.13	0.59	1	-0.051	0.17
host_listings_count	0.13	0.15	0.021	-0.11	0.17	0.13	-0.071	-0.051	1	0.23
availability_365	0.073	0.19	-0.017	0.097	0.067	0.15	0.18	0.17	0.23	1

- Total Reviews & Reviews per Month (0.59) show that listings with more frequent monthly reviews tend to accumulate more total reviews, indicating consistent bookings.
- Host Listings Count & Availability_365 (0.23) suggests that hosts with more properties generally have higher yearly availability, possibly due to lower occupancy.
- Price & Longitude (-0.31) reveals that properties located further east tend to be more affordable.
- Total Reviews & Listing ID (-0.32) indicates that newer listings (higher listing IDs) have fewer reviews, likely due to their shorter time on the market.

❖ Statistical analysis

➢ One way annova test for Price and neighbourhood groups

code:

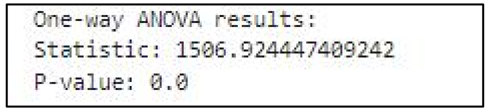


```
from scipy.stats import f_oneway

data = df[['price', 'neighbourhood_group']]
neighbourhood_groups = data['neighbourhood_group'].unique()
grouped_data = [data['price'][data['neighbourhood_group'] == group] for group in neighbourhood_groups]
statistic, p_value = f_oneway(*grouped_data)

print("One-way ANOVA results:")
print(f"Statistic: {statistic}")
print(f"P-value: {p_value}")
```

output :

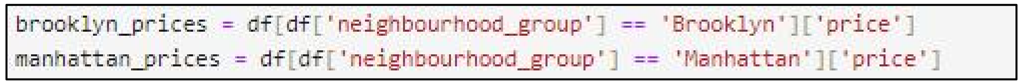


```
One-way ANOVA results:
Statistic: 1506.924447409242
P-value: 0.0
```

Based on the very high F-statistic (1506.92) and the p-value of 0.0, we can say that there are important differences in average prices between the different neighborhood groups.This means some neighborhoods likely have higher or lower prices than others.

- Comparing the prices of neighbourhood group Manhattan and Brooklyn
  - Filtering the prices for Manhattan and Brookyn neighbourhood group

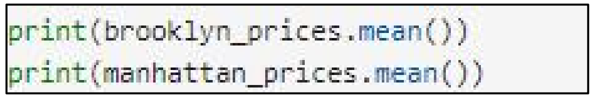
Code:



```
brooklyn_prices = df[df['neighbourhood_group'] == 'Brooklyn']['price']
manhattan_prices = df[df['neighbourhood_group'] == 'Manhattan']['price']
```

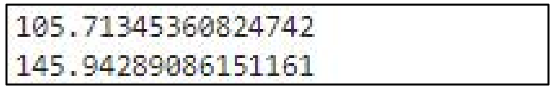
  - Calculating their mean

Code:



```
print(brooklyn_prices.mean())
print(manhattan_prices.mean())
```

Output:

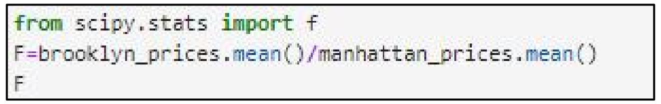


```
105.71345360824742
145.94289086151161
```

Output suggests that the mean of brooklyn and manhattan prices are 105.713 and 145.942 respectively.

  - Calculating the ratio of their means

Code :



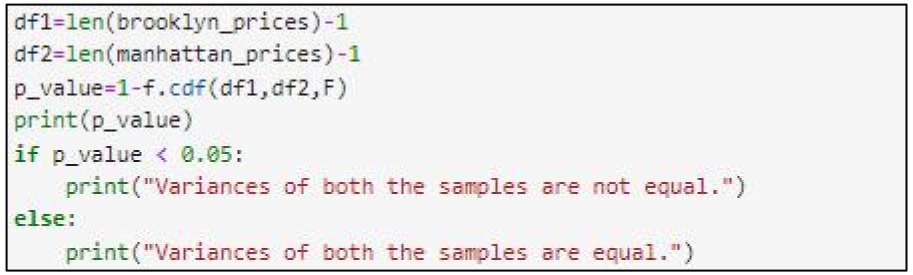
```
from scipy.stats import f
F=brooklyn_prices.mean()/manhattan_prices.mean()
F
```

Output: 0.724

Output shows that the ratios of their mean prices is 0.724

➢ Calculating P- value

Code :



```
df1=len(brooklyn_prices)-1
df2=len(manhattan_prices)-1
p_value=1-f.cdf(df1,df2,F)
print(p_value)
if p_value < 0.05:
    print("Variances of both the samples are not equal.")
else:
    print("Variances of both the samples are equal.")
```

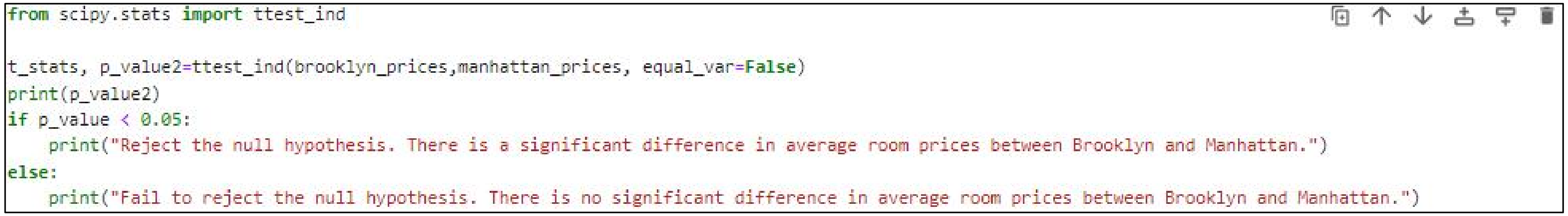
Output:

```
0.021773985421123965
Variances of both the samples are not equal.
```

Given output shows that variances of Manhattan prices and Brookly prices are not equal.

➢ Performing 2 sample T-test

code :



```
from scipy.stats import ttest_ind

t_stats, p_value2=ttest_ind(brooklyn_prices,manhattan_prices, equal_var=False)
print(p_value2)
if p_value < 0.05:
    print("Reject the null hypothesis. There is a significant difference in average room prices between Brooklyn and Manhattan.")
else:
    print("Fail to reject the null hypothesis. There is no significant difference in average room prices between Brooklyn and Manhattan.")
```

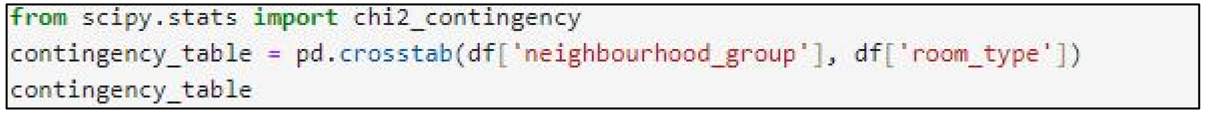
Output:

0.0

Reject the null hypothesis. There is a significant difference in average room prices between Brooklyn and Manhattan.

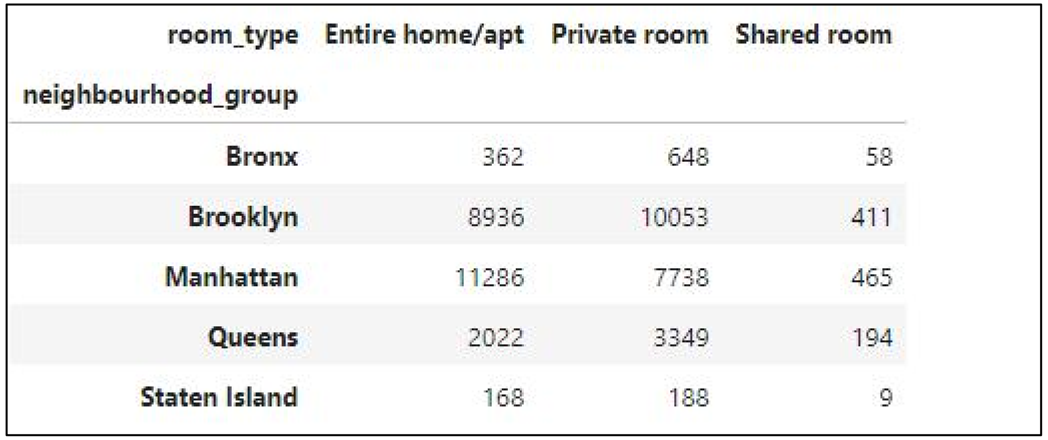
➢ Checking association between two samples using Contigency table

Code:



```
from scipy.stats import chi2_contingency
contingency_table = pd.crosstab(df['neighbourhood_group'], df['room_type'])
contingency_table
```

Output:



room_type neighbourhood_group	Entire home/apt	Private room	Shared room
Bronx	362	648	58
Brooklyn	8936	10053	411
Manhattan	11286	7738	465
Queens	2022	3349	194
Staten Island	168	188	9

- Given contigency table gives the frequency of the Neighbourhood groups and the Room types
- Manhattan has the 11286 Entire home/apt, 7738 Private rooms and 465 Shared rooms.
- Brooklyn has 8936 Entire home/apt, 10053 Private rooms and Shared rooms.
- Staten Island have the lowest frequency with Shared room 9.