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

```
df=pd.read_csv('/content/AirBnB-nyc.csv')
df
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimu
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	
...	...	...	...	...	...	...	...	...	...	...	
48890	36484665	Charming one bedroom - newly renovated rowhouse	8232441	Sabrina	Brooklyn	Bedford-Stuyvesant	40.67853	-73.94995	Private room	70	
48891	36485057	Affordable room in Bushwick/East Williamsburg	6570630	Marisol	Brooklyn	Bushwick	40.70184	-73.93317	Private room	40	
48892	36485431	Sunny Studio at Historical Neighborhood	23492952	Ilgar & Aysel	Manhattan	Harlem	40.81475	-73.94867	Entire home/apt	115	
48893	36485609	43rd St. Time Square-cozy single bed	30985759	Taz	Manhattan	Hell's Kitchen	40.75751	-73.99112	Shared room	55	
48894	36487245	Trendy duplex in the very heart of Hell's Kitchen	68119814	Christophe	Manhattan	Hell's Kitchen	40.76404	-73.98933	Private room	90	

48895 rows × 16 columns

```
missing_values = df.isnull().sum()
print(missing_values)
```

```
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
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              48895 non-null  int64
 1   name                            48879 non-null  object
 2   host_id                         48895 non-null  int64
 3   host_name                       48874 non-null  object
 4   neighbourhood_group             48895 non-null  object
 5   neighbourhood                   48895 non-null  object
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object
 9   price                           48895 non-null  int64
 10  minimum_nights                  48895 non-null  int64
 11  number_of_reviews               48895 non-null  int64
 12  last_review                     38843 non-null  object
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64
 15  availability_365                48895 non-null  int64
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```

1.General statistics for prices in different neighnourhoods

```
stats=df.groupby('neighbourhood')['price'].agg(
    average_price=np.mean,
    median_price=np.median,
    min_price=np.min,
    max_price=np.max
).reset_index()
stats['price_range'] =stats['max_price']-stats['min_price']
print(stats)
```

```
        neighbourhood  average_price  median_price  min_price  max_price  \
0            Allerton      87.595238          66.5         33        450
1       Arden Heights      67.250000          72.5         41         83
2            Arrochar     115.000000          65.0         32        625
3             Arverne     171.779221         125.0         35       1500
4             Astoria     117.187778          85.0         25      10000
..                ...            ...           ...        ...        ...
216  Windsor Terrace     138.993631         123.0         38        450
217        Woodhaven      67.170455          52.0         10        250
218         Woodlawn      60.090909          68.0         29         85
219          Woodrow     700.000000         700.0        700        700
220         Woodside      85.097872          60.0         28        500

     price_range
0            417
1             42
2            593
3           1465
4           9975
..           ...
216          412
217          240
218           56
219            0
220          472

[221 rows x 6 columns]
```

2. Room distribution-calculates the percentage distribution of each room type.

```
room_type_distribution = df['room_type'].value_counts(normalize=True) * 100
print(room_type_distribution)
```

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

HOST AND LISTING CHARACTERISTICS:

1.What is the average number of listings per host? 2.Which hosts have the most listings? 3.How does the number of reviews correlate with the price or type of listing?

```
Q1= df['host_id'].value_counts().mean()
print(Q1)
```

```
1.3053634834610353
```

```
Q2=df['host_id'].value_counts().reset_index()
print(Q2)
```

```
          host_id  count
0      219517861    327
1      107434423    232
2       30283594    121
3      137358866    103
4       16098958     96
...          ...    ...
37452   23727216      1
37453   89211125      1
37454   19928013      1
37455    1017772      1
37456   68119814      1

[37457 rows x 2 columns]
```

```
top_hosts = Q2.head(10)
print(top_hosts)
```

```
      host_id  count
0  219517861    327
1  107434423    232
2   30283594    121
3  137358866    103
4   16098958     96
5   12243051     96
6   61391963     91
7   22541573     87
8  200380610     65
9    7503643     52
```

Correlation between the number of reviews and the price

```
correlation=df[['price','number_of_reviews']].corr()
print(correlation)
```

```
                      price  number_of_reviews
price              1.000000          -0.047954
number_of_reviews -0.047954           1.000000
```

Scatter plot of number of reviews vs price

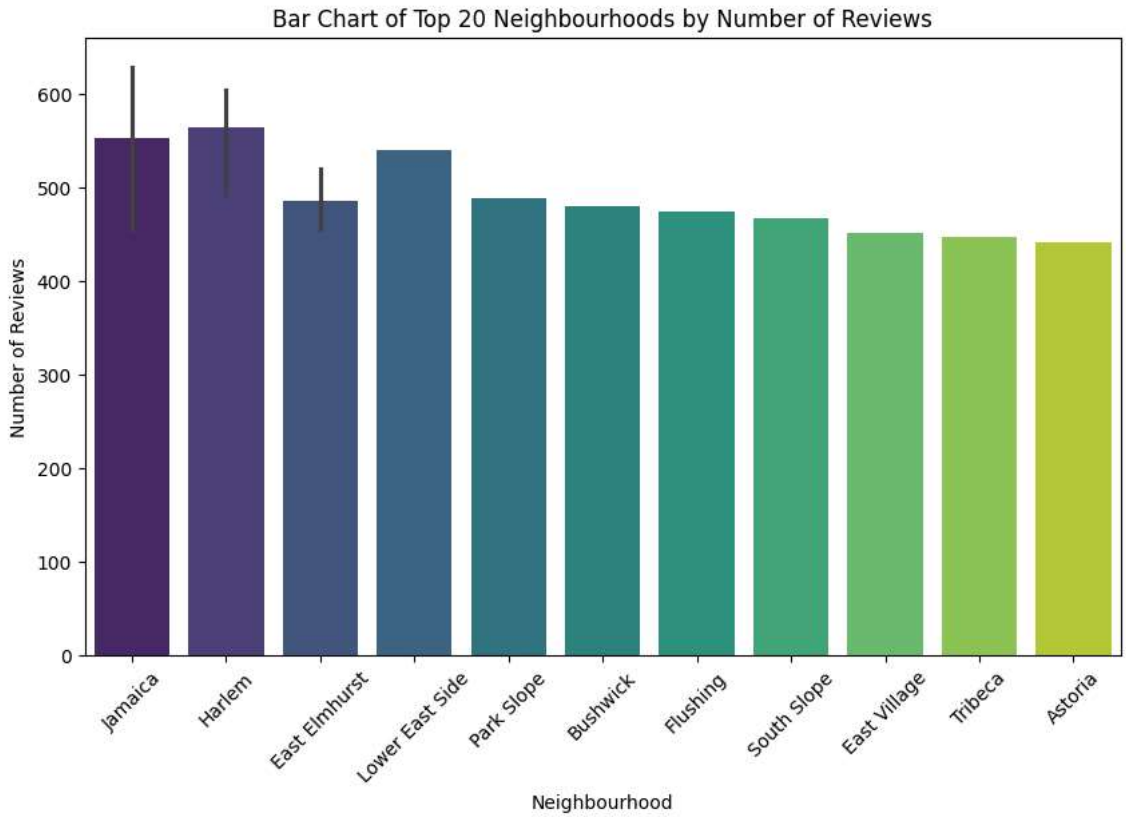
```
top_20_reviews = df.nlargest(20, 'number_of_reviews')

# Plot bar chart for top 10 reviews
plt.figure(figsize=(10, 6))
sns.barplot(x='neighbourhood', y='number_of_reviews', data=top_20_reviews, palette='viridis')
plt.title('Bar Chart of Top 20 Neighbourhoods by Number of Reviews')
plt.xlabel('Neighbourhood')
plt.ylabel('Number of Reviews')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-11-3a3e8d680779>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.barplot(x='neighbourhood', y='number_of_reviews', data=top_20_reviews, palette='viridis')
```



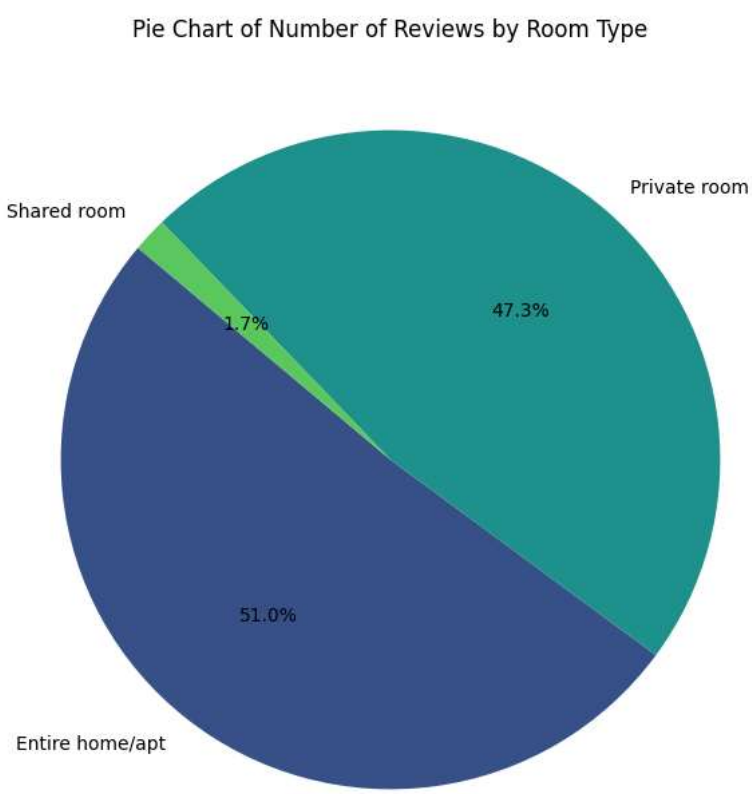

```
# Correlation between the number of reviews and the room type
# First, we need to convert the room_type to numerical values
room_type_mapping = {
    'Entire home/apt': 1,
    'Private room': 2,
    'Shared room': 3,
    'Hotel room': 4
}
df['room_type_num'] = df['room_type'].map(room_type_mapping)
correlation_reviews_room_type = df[['number_of_reviews', 'room_type_num']].corr()
print(correlation_reviews_room_type)
```

```
                    number_of_reviews  room_type_num
number_of_reviews           1.000000       0.002724
room_type_num               0.002724       1.000000
```

Box plot of number of reviews by room type

```
room_type_reviews = df.groupby('room_type')['number_of_reviews'].sum()

# Plot pie chart
plt.figure(figsize=(8, 8))
plt.pie(room_type_reviews, labels=room_type_reviews.index, autopct='%1.1f%%', startangle=140, colors=sns.color_palette('viridis', n_colors=l
plt.title('Pie Chart of Number of Reviews by Room Type')
plt.show()
```



Neighbourhood Comparison

```
Neighbour_hood=df.groupby('neighbourhood').agg(average_price=('price', 'mean'),
    average_availability=('availability_365', 'mean'),
    average_reviews=('number_of_reviews', 'mean')
).reset_index()
print(Neighbour_hood)
```

```
        neighbourhood  average_price  average_availability  average_reviews
     0           Allerton      87.595238            163.666667        42.928571
     1      Arden Heights      67.250000             94.250000         7.750000
     2           Arrochar     115.000000            255.809524        14.619048
     3            Arverne     171.779221            188.428571        29.259740
     4            Astoria     117.187778            109.191111        21.455556
     ..               ...            ...                   ...              ...
     216  Windsor Terrace     138.993631             81.885350        27.541401
     217        Woodhaven      67.170455            200.920455        31.727273
     218         Woodlawn      60.090909             98.272727        44.000000
     219          Woodrow     700.000000              0.000000         0.000000
     220         Woodside      85.097872            130.217021        21.425532

     [221 rows x 4 columns]
```

Outliers in average price:-

```
neighbourhood_avg_price = df.groupby('neighbourhood')['price'].mean().reset_index()
# Detect outliers using IQR method
Q1 = neighbourhood_avg_price['price'].quantile(0.25)
Q3 = neighbourhood_avg_price['price'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers = neighbourhood_avg_price[(neighbourhood_avg_price['price'] < lower_bound) | (neighbourhood_avg_price['price'] > upper_bound)]
print(outliers)
```

```
          neighbourhood       price
     6    Battery Park City  367.557143
```

```
75   Flatiron District  341.925000
82      Fort Wadsworth  800.000000
92   Greenwich Village  263.405612
127            Midtown  282.719094
139           Neponsit  274.666667
144               NoHo  295.717949
157       Prince's Bay  409.500000
161      Randall Manor  336.000000
167          Riverdale  442.090909
174           Sea Gate  487.857143
178               SoHo  287.103352
197            Tribeca  490.638418
209       West Village  267.682292
219            Woodrow  700.000000
```

```
top_5_neighbourhoods = neighbourhood_avg_price.nlargest(5, 'price')

# Plot
plt.figure(figsize=(14, 7))
sns.scatterplot(data=top_5_neighbourhoods, x='neighbourhood', y='price', color='blue', label='Top 5 Neighbourhoods')
sns.scatterplot(data=outliers[outliers['neighbourhood'].isin(top_5_neighbourhoods['neighbourhood'])], x='neighbourhood', y='price', color='r
plt.xticks(rotation=90)
plt.title('Scatter Plot of Top 5 Average Prices by Neighbourhood with Outliers Highlighted')
plt.xlabel('Neighbourhood')
plt.ylabel('Average Price')
plt.legend()
plt.show()
```