AirBnb Bookings Analysis

Project Type - Exploratory Data Analysis

Contribution - Individual

Project Summary -

This project involves performing an analysis on the data having around 49000 observations. The objective is to seek-out meaningful insights from this data to provide crucial information to the management and the stakeholders to make decisions on what can be done to expand or improve the business, eventually leading to the growth of the business. This project will identify the patterns that are being formed in these observations and based on those patterns some suggestions can be taken in consideration. This analysis will help not only the guests to make better choices but it will also be helping the hosts, so that they may make the required changes in order to grow their business. As we are provided with information like:- Listing Counts Data Distributed as per specific neighborhood groups Prices Reviews data Room Type Preference Using this information we will be able to seek insights like:- Preference of the guests for their hosts Room-Type preference Prefered Price range Most preferred neighborhood We will also be able to create a filter system for our guests to provide them with the listings as per their budget and other preferences, we will be able to rank our hosts as per their ability to match the guests requirement, this way we will be targeting customers with specific needs and we will be fulfilling those needs, which will eventually enhance customer experience, and they will be getting utmost satisfaction when all of their needs will be fulfilled. We will also be able to provide our hosts with the information on what they can change or improve so that they can get more attention and also they will be able to fulfill the needs of their guests.

Majorly we will be doing data wrangling using "pandas" to frame this data more clearly and seek out meaningful information that can be used to analyze the requirements and the preference of the guests. There might also be a need to create other data frames so that information can be stored separately and can be accessed whenever needed.

For computations and calculations on the numerical data we will be using numpy so that we can seek insights mainly for ranking purposes. Numpy will be helping us to create required arrays so that the data can be stored and accessed in a systematic manner.

In order to simplify and study the results and the analysis more efficiently, the optimum presentation of the analysis is a must, which makes it easier for the stakeholders to understand the results of the analysis and make informed decisions. In order to achieve this we will be using Matplotlib and Seaborn for data visualization and for the presentation of the insights.

Now coming onto what I will be learning from this project, as this is a real life situation where I will have to do some research on how things really work in this field I will be getting the knowledge of the working model of this business, along with this I will get an idea on how do we need to think and create solutions for the problems, however identifying problems in any statement is also a skill in itself, this project will also help me to polish those skills as well, talking about data science, I will be learning the optimum utilization of the libraries which will help me to learn how to use the methods provided in the best way possible, I will also be working on critical thinking on how to apply some provided complex concepts on this given data so as to make it more meaningful as if it is telling a story.

This data analysis project will help me improve my knowledge and skills and also my understanding of the real time complexities.

Problem Statement

Business Objective - The business objective of this project is to identify opportuinites of improvements and also to derive patterns and insights of customer's preference which will ultimately help us in achieving customer satisfaction.

General Guidelines : -

Well-structured, formatted, and commented code is required.

Exception Handling, Production Grade Code & Deployment Ready Code will be a plus. Those students will be awarded some additional credits.

The additional credits will have advantages over other students during Star Student selection.

```
[ Note: - Deployment Ready Code is defined as, the whole .ipynb notebook should be executable in one go
          without a single error logged. ]
```

Each and every logic should have proper comments.

You may add as many number of charts you want. Make Sure for each and every chart the following format should be answered.

Chart visualization code

Why did you pick the specific chart? What is/are the insight(s) found from the chart? Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. You have to create at least 20 logical & meaningful charts having important insights. [Hints : - Do the Vizualization in a structured way while following "UBM" Rule.

U - Univariate Analysis,

B - Bivariate Analysis (Numerical - Categorical, Numerical - Numerical, Categorical - Categorical)

M - Multivariate Analysis]

Let's Begin !

1. Know Your Data

Import Libraries

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import math
import seaborn as sns
import openpyxl
pd.set_option('display.max_columns', 200)
```

Dataset Loading

```
# Load Dataset
airbnb_df = pd.DataFrame(pd.read_csv("/content/Airbnb NYC 2019.csv"))
```

Dataset First View

```
airbnb_df.head(5)
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nig
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	

```
airbnb_df.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'],
      dtype='object')
```

Dataset Rows & Columns count

```
airbnb_df.shape
```

```
(48895, 16)
```

Dataset Information

```
airbnb_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
```

Duplicate Values

```
duplicated_values = airbnb_df[airbnb_df.duplicated()]
duplicated_values
```

id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_

There are no exact duplicate values within the dataset.

```
duplicated_values = airbnb_df[airbnb_df.duplicated('name')]
duplicated_values
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_
330	81739	Loft w/ Terrace @ Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73842	-73.95312	Private room	249	
339	84010	Superior @ Box House	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73813	-73.95394	Private room	179	
580	219818	✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿	1138692	Keera (Jena)	Manhattan	Lower East Side	40.71892	-73.98401	Entire home/apt	199	
661	250537	The Lenox in Harlem	1313306	Yvette	Manhattan	Harlem	40.81122	-73.94279	Entire home/apt	400	
669	253471	Loft Suite @ The Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73641	-73.95330	Entire home/apt	199	
...	...	...	...	...	...	...	...	...	...	...	
48684	36382847	Comfort home	266211707	Yan	Brooklyn	Sunset Park	40.64439	-74.01816	Private room	185	
48735	36412461	Sunny, Cozy, Private Room In The Heart of Bush...	147515897	Flávia	Brooklyn	Bushwick	40.70366	-73.92728	Private room	84	
48759	36420404	Home Sweet Home	273656890	Liana	Manhattan	East Harlem	40.79266	-73.94740	Private room	50	
48791	36427922	Home away from home	238163900	Lucy	Queens	Cambria Heights	40.68557	-73.72731	Private room	50	
48826	36449743	Brooklyn's finest	66084717	Tim	Brooklyn	East Flatbush	40.65170	-73.92580	Entire home/apt	200	

989 rows × 16 columns

Now we can see that there are 998 rows that have duplicated names.

```
airbnb_df.query('name == "Superior @ Box House"')
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights
321	77765	Superior @ Box House	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73749	-73.95292	Private room	179	3
339	84010	Superior @ Box House	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73813	-73.95394	Private room	179	3
682	253846	Superior @ Box House	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73731	-73.95450	Private room	179	3

```
airbnb_df = airbnb_df[['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']].copy()
```

This dataframe is ready to handle duplicated values more efficiently.

```
airbnb_df.query('name == "Loft w/ Terrace @ Box House Hotel"')
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights
328	80700	Loft w/ Terrace @ Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73738	-73.95482	Private room	349	3
330	81739	Loft w/ Terrace @ Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73842	-73.95312	Private room	249	3
680	253839	Loft w/ Terrace @ Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73783	-73.95259	Private room	249	3

```
airbnb_df.loc[airbnb_df.duplicated(subset= ['name', 'host_name', 'neighbourhood_group', 'neighbourhood', 'room_type'])]
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minim
330	81739	Loft w/ Terrace @ Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73842	-73.95312	Private room	249	
339	84010	Superior @ Box House	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73813	-73.95394	Private room	179	
580	219818	✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿	1138692	Keera (Jena)	Manhattan	Lower East Side	40.71892	-73.98401	Entire home/apt	199	
669	253471	Loft Suite @ The Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73641	-73.95330	Entire home/apt	199	
670	253475	Loft Suite @ The Box House Hotel	417504	The Box House Hotel	Brooklyn	Greenpoint	40.73794	-73.95254	Entire home/apt	199	
...	...	...	...	...	...	...	...	...	...	...	
47876	35966653	Bright, contemporary and best location	24232061	Tracy	Manhattan	Upper East Side	40.77297	-73.95530	Private room	122	
48026	36039574	★Premier Queen Room with Balcony ★	270874051	Hotel Vetiver	Queens	Long Island City	40.75300	-73.93485	Private room	99	
48207	36139806	30 mins to Times Square!! 15 mins LGA, 25mins ...	260209224	Lotay	Queens	Jackson Heights	40.75077	-73.87020	Entire home/apt	67	
48662	36372006	Very Clean Private Room Near Buses & Restauran...	118405437	PengYu	Queens	Woodhaven	40.69411	-73.86877	Private room	66	
48684	36382847	Comfort home	266211707	Yan	Brooklyn	Sunset Park	40.64439	-74.01816	Private room	185	

233 rows × 16 columns

```
airbnb_df.query('name == "✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿"')
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights
579	219793	✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿	1138692	Keera (Jena)	Manhattan	Lower East Side	40.71813	-73.98416	Entire home/apt	199	
580	219818	✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿	1138692	Keera (Jena)	Manhattan	Lower East Side	40.71892	-73.98401	Entire home/apt	199	

All the values, included duplicared values, are same except for last_review and reviews_per_month, which are not so important regarding duplicacy. So, we can drop the duplicated values to sort out dataframe on the basis of latest entries.

```
airbnb_df['last_review'] = pd.to_datetime(airbnb_df['last_review'])
```

```
airbnb_df = airbnb_df.sort_values(by='last_review', ascending=False).reset_index(drop=True)
```

```
airbnb_df['last_review'].replace(np.nan,airbnb_df['last_review'].max(), inplace=True)

airbnb_df = airbnb_df.sort_values(by='last_review', ascending=False).reset_index(drop=True)
```

```
<ipython-input-19-ce4e210adafb>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col


  airbnb_df['last_review'].replace(np.nan,airbnb_df['last_review'].max(), inplace=True)
```

```
# Dropping duplicated values
airbnb_df = airbnb_df.drop_duplicates(subset=['name', 'host_name', 'neighbourhood_group', 'neighbourhood', 'room_type'], keep='first').r

airbnb_df.shape # (48655, 14)
```

```
airbnb_df[airbnb_df.duplicated(subset=['name', 'host_name', 'neighbourhood_group', 'neighbourhood', 'room_type'])] # No values
airbnb_df.query('name == "✿✿✿ COUNTRY COTTAGE IN THE CITY✿✿✿    "')
```

id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_

Missing Values/Null Values

```
null_values = airbnb_df.isnull()
null_value_count = null_values.sum()

null_value_count
```

	0
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	0
reviews_per_month	9989
calculated_host_listings_count	0
availability_365	0

dtype: int64

```
airbnb_df.describe()
```

	id	host_id	latitude	longitude	price	minimum_nights	number_of_reviews	last_review	rev
count	4.866200e+04	4.866200e+04	48662.000000	48662.000000	48662.000000	48662.000000	48662.000000	48662	
mean	1.900484e+07	6.746247e+07	40.728926	-73.952196	152.658173	7.002219	23.337738	2018-11-30 01:10:55.903990784	
min	2.539000e+03	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	0.000000	2011-03-28 00:00:00	
25%	9.462062e+06	7.802407e+06	40.690000	-73.983068	69.000000	1.000000	1.000000	2018-11-04 00:00:00	
50%	1.965763e+07	3.070801e+07	40.722985	-73.955660	105.000000	2.000000	5.000000	2019-06-14 00:00:00	
75%	2.913386e+07	1.074344e+08	40.763130	-73.936270	175.000000	5.000000	24.000000	2019-07-04 00:00:00	

```
airbnb_df.rename(columns={
    "calculated_host_listings_count": 'listings'
}, inplace=True)
```

```
airbnb_df['price'].describe()
```

	price
count	48662.000000
mean	152.658173
std	240.418499
min	0.000000
25%	69.000000
50%	105.000000
75%	175.000000
max	10000.000000

dtype: float64

```
##Check the minimum_nights column.
airbnb_df['minimum_nights'].describe()
airbnb_df['minimum_nights'].replace(range(366, 1251), 365, inplace=True) # replacing values
airbnb_df['minimum_nights'].describe()
```

```
<ipython-input-25-63cb7f09c3bc>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained as:
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[co]

  airbnb_df['minimum_nights'].replace(range(366, 1251), 365, inplace=True) # replacing values
```

	minimum_nights
count	48662.000000
mean	6.914821
std	17.543710
min	1.000000
25%	1.000000
50%	2.000000
75%	5.000000
max	365.000000

dtype: float64

What did you know about your dataset? The Airbnb NYC 2019 Dataset has:- Rows = 48895 Columns = 16

We can see the division of Categorical and Numerical values in our dataset, We can see:-

3 columns - float64
7 columns - int64 6 columns - object data type (Categorical)

The dataset contains both numerical and categorical data.

The primary key of our dataset is the "id" column, having a unique IDs for the hotel names.

This dataset had no exact duplicated values, however in the 'name' column there were 998 rows that were duplicates. Where all the values were almost the same, except for the prices, there might be a chances that the prices were altered with time as the hotel was the same.

To handle these values we sorted our dataframe on the basis of latest entries, for which we needed a timestamp in our dataset. In our dataset we used the 'last_review' column as the timestamp for our dataset.

After all these operations, we were successfully able to handle our duplicated values.

These are columns that majorly has the null values:-

last_review = 10052 reviews_per_month = 10052 As the date column was missing values we replaced the NA values with the latest date present in our dataset, so that the time stamp could be efficient.

We are also missing few "Names" as well as "Host Names":-

name = 16 host_name = 21 We have now successfully formatted our dataframe and it is now ready for data wrangling.

2. Understanding Your Variables

```
airbnb_df.head(5)
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	m
0	36455809	Cozy Private Room in Bushwick, Brooklyn	74162901	Christine	Brooklyn	Bushwick	40.69805	-73.92801	Private room	30	
1	15968426	Comfy spacious Astoria aptmt 15m from Manhattan	2753243	Jwanah	Queens	Astoria	40.76248	-73.92422	Private room	56	
2	16000062	Charming Carriage House near Prospect Park	4366974	Ariana	Brooklyn	Crown Heights	40.67320	-73.96195	Entire home/apt	300	
3	15998231	Cozy Room in Brownstone	20327528	Miller	Brooklyn	Bedford-Stuyvesant	40.68631	-73.95597	Private room	35	
4	15987034	near Williamsburg/Manhattan/ Greenpoint - 1 b/1b	19803201	Gino	Brooklyn	Greenpoint	40.72388	-73.94040	Entire home/apt	150	

```
# Dataset Columns
df_columns = airbnb_df.columns
```

```
df_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', 'listings', 'availability_365'],
      dtype='object')
```

```
# Dataset Describe
df_describe = airbnb_df.describe()
df_describe
```

	id	host_id	latitude	longitude	price	minimum_nights	number_of_reviews	last_review	rev
count	4.866200e+04	4.866200e+04	48662.000000	48662.000000	48662.000000	48662.000000	48662.000000	48662	
mean	1.900484e+07	6.746247e+07	40.728926	-73.952196	152.658173	6.914821	23.337738	2018-11-30 01:10:55.903990784	
min	2.539000e+03	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	0.000000	2011-03-28 00:00:00	
25%	9.462062e+06	7.802407e+06	40.690000	-73.983068	69.000000	1.000000	1.000000	2018-11-04 00:00:00	
50%	1.965763e+07	3.070801e+07	40.722985	-73.955660	105.000000	2.000000	5.000000	2019-06-14 00:00:00	
75%	2.913386e+07	1.074344e+08	40.763130	-73.936270	175.000000	5.000000	24.000000	2019-07-04	

Variables Description

There are 3 types of variables :-

Numerical Variables: represent quantitative data and can be further categorized into:-

Continuous Variables: take any value within the given number of range. Discrete Variables: having a specific value with a specific identity. Categorical Variables: represent qualitative data and can be further categorized into:-

Nominal Variables: They are random and do not follow any order or ranking. Ordinal Variables: They are according to an order and can be ranked as well. Time Variables: They are basically date and time variables having a timestamp.

Check Unique Values for each variable.

```
def unique_value(df):
  for column in df.columns:
    unique_values = df[column].unique()
    print(f"Unique values for '{column}': {unique_values}")
```

3. Data Wrangling

Data Wrangling Code

```
##Create a new column with the price range distribution
start = 0
end = 10000
breakpoints = np.linspace(start, end, num=101)
breakpoints = breakpoints.astype(int)
def price_range(amt):
    bp = breakpoints
    for i in range(len(breakpoints)-1):
        if bp[i] <= amt <= bp[i+1]:
            return f"{bp[i]} - {bp[i+1]}"


airbnb_df['price_range'] = airbnb_df['price'].apply(lambda amt: price_range(amt))
airbnb_df['price_range']
```

	price_range
0	0 - 100
1	0 - 100
2	200 - 300
3	0 - 100
4	100 - 200
...	...
48657	0 - 100
48658	100 - 200
48659	0 - 100
48660	200 - 300
48661	0 - 100

48662 rows × 1 columns

dtype: object

```
airbnb_df.head()
```

```
airbnb_df.sort_values(by='price', ascending=False, inplace=True)
```

```
airbnb_df.head()
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_
42906	13894339	Luxury 1 bedroom apt. - stunning Manhattan views	5143901	Erin	Brooklyn	Greenpoint	40.73260	-73.95739	Entire home/apt	10000	
1809	22436899	1-BR Lincoln Center	72390391	Jelena	Manhattan	Upper West Side	40.77213	-73.98665	Entire home/apt	10000	
46635	7003697	Furnished room in Astoria apartment	20582832	Kathrine	Queens	Astoria	40.76810	-73.91651	Private room	10000	
46974	9528920	Quiet, Clean, Lit @ LES & Chinatown	3906464	Amy	Manhattan	Lower East Side	40.71355	-73.98507	Private room	9999	
9953	31340283	2br - The Heart of NYC: Manhattans Lower East ...	4382127	Matt	Manhattan	Lower East Side	40.71980	-73.98566	Entire home/apt	9999	

```
airbnb_df.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', 'listings', 'availability_365', 'price_range'],
      dtype='object')
```

Considering the following columns to be the selected features to work with :-

id, host_id, neighbourhood_group, neighbourhood, room_type, price, price_range, minimum_nights, number_of_reviews, reviews_per_month, listings, availability_365

```
feature_df = airbnb_df[['id', 'name', 'host_id', 'neighbourhood_group',
       'neighbourhood', 'room_type', 'price', 'minimum_nights',
       'number_of_reviews', 'reviews_per_month', 'listings',
       'availability_365', 'price_range']]
```

```
feature_df = feature_df.reset_index(drop=True)
feature_df.head()
```

	id	name	host_id	neighbourhood_group	neighbourhood	room_type	price	minimum_nights	number_of_reviews	reviews_p
0	13894339	Luxury 1 bedroom apt. - stunning Manhattan views	5143901	Brooklyn	Greenpoint	Entire home/apt	10000	5	5	
1	22436899	1-BR Lincoln Center	72390391	Manhattan	Upper West Side	Entire home/apt	10000	30	0	
		Furnished room in				Private				

```
feature_df.head()
```

	id	name	host_id	neighbourhood_group	neighbourhood	room_type	price	minimum_nights	number_of_reviews	reviews_p
0	13894339	Luxury 1 bedroom apt. - stunning Manhattan views	5143901	Brooklyn	Greenpoint	Entire home/apt	10000	5	5	
1	22436899	1-BR Lincoln Center	72390391	Manhattan	Upper West Side	Entire home/apt	10000	30	0	
		Furnished room in				Private				

```
##Check the top performing Host as per the total listing count
host_groups = feature_df.groupby('name')
hosts = []
no_of_listings = []
host_prices = []
for host, data in host_groups:
  hosts.append(host)
  no_of_listings.append(data['listings'].sum())
  host_prices.append(data['price'].mean())

host_df = pd.DataFrame({
    'Host Name': hosts,
    'Total Listings': no_of_listings,
    'Price': host_prices
})


host_df = host_df.sort_values(by='Total Listings', ascending=False).reset_index(drop=True)
host_df = host_df.drop_duplicates(subset='Total Listings').reset_index(drop=True)
top_10_hosts = host_df.head(10)

host_df['Revenue'] = (host_df['Total Listings'])*(host_df['Price'])
top_host_revenue = host_df.sort_values(by='Revenue', ascending=False).reset_index(drop=True)
top_host_revenue.head(10)
```

	Host Name	Total Listings	Price	Revenue
0	Pleasant 1BR in Midtown East by Sonder	654	197.000000	128838.0
1	Sonder | Stock Exchange | Cozy 1BR + Lounge	327	229.000000	74883.0
2	West 55th street, Lux 1bd Serviced Apartment	261	251.666667	65685.0
3	Gorgeous + Bright Midtown East 1BR, Doorman, G...	232	241.000000	55912.0
4	*NO GUEST SERVICE FEE* Beekman Tower One Bedro...	49	714.000000	34986.0
5	West 15th Street Cozy Chelsea 1bd Serviced Apt	174	200.000000	34800.0
6	Stunning 1 Bedroom Apt. in NYC, w/d in the unit!	121	239.000000	28919.0
7	2BR WITH PRIVATE PATIO EAST VILLAGE	114	249.000000	28386.0
8	DOORMAN/ GYM/ MODERN 2 BR ON EAST 52ND ST	65	385.000000	25025.0
9	Prime Location One Bed Doorman Gym Deck!5223	96	250.000000	24000.0

```
#Let's check the number of neighbourhood_groups are there.
feature_df['neighbourhood_group'].unique()


n_groups = feature_df.groupby('neighbourhood_group')
```

```
groups = []
listings = []
reviews = []
max_price = []
min_price = []
for group, data in n_groups:
  groups.append(group)
  listings.append(data['listings'].sum())
  reviews.append(data['number_of_reviews'].sum())
  max_price.append(data['price'].max())
  min_price.append(data['price'].min())

group_feat_df = pd.DataFrame({
    'Group': groups,
    'Listing Count': listings,
    'No._of_reviews': reviews,
    'Min Price': min_price,
    'Max Price': max_price
})

group_feat_df
```

	Group	Listing Count	No._of_reviews	Min Price	Max Price
0	Bronx	2410	28326	0	2500
1	Brooklyn	44976	485088	0	10000
2	Manhattan	269295	453964	0	10000
3	Queens	22577	156743	10	10000
4	Staten Island	864	11540	13	5000

```
# Now let's divide our dataset based on room types, and create their groups.

feature_df['room_type'].unique()

room_groups = feature_df.groupby('room_type')
rooms = []
room_listings = []
room_reviews = []
max_room_price = []
min_room_price = []
for room_type, room_data in room_groups:
    rooms.append(room_type)
    room_listings.append(room_data['listings'].sum())
    max_room_price.append(room_data['price'].max())
    min_room_price.append(room_data['price'].min())

room_feat_df = pd.DataFrame({
    'Group': rooms,
    'Listing Count': room_listings,
    'Min Price': min_room_price,
    'Max Price': max_room_price
})

room_feat_df
```

	Group	Listing Count	Min Price	Max Price
0	Entire home/apt	263986	0	10000
1	Private room	70813	0	10000
2	Shared room	5323	0	1800

```
# Now let us check how many different neighbourhoods are there in total.
feature_df['neighbourhood'].count()
```

```
np.int64(48662)
```

```
# List the top 10 most prefered neighbourhoods and check their average pricing and average price range.
area_groups = feature_df.groupby('neighbourhood')
areas = []
listing_count = []
avg_price = []
for area, n_data in area_groups:
  areas.append(area)
  listing_count.append(n_data['listings'].sum())
  avg_price.append(round(n_data['price'].mean(), 2))
```

```
area_feat_df = pd.DataFrame({
    'Area': areas,
    'Listing Count': listing_count,
    'Average Price': avg_price,
})

area_feat_df = area_feat_df.sort_values(by='Listing Count', ascending=False).reset_index(drop=True)
area_feat_df.head(10)
```

	Area	Listing Count	Average Price
0	Financial District	84942	226.01
1	Hell's Kitchen	24754	205.32
2	Murray Hill	24726	221.12
3	Midtown	24647	282.66
4	Chelsea	17483	248.34
5	Theater District	16151	242.67
6	Upper East Side	14909	189.08
7	Upper West Side	13201	211.23
8	Bedford-Stuyvesant	9605	107.73
9	Tribeca	7519	492.23

```
avg_room_price = []
for room, data in room_groups:
  avg_room_price.append(data['price'].mean())

room_vs_price = pd.DataFrame({
    'Room Type': rooms,
    'Avg_Price': avg_room_price
})

room_vs_price
```

	Room Type	Avg_Price
0	Entire home/apt	211.681940
1	Private room	89.611256
2	Shared room	70.241768

```
# Relationship: Reviews per Month vs. Room Type

total_reviews = []
for room, data in room_groups:
  total_reviews.append(data['reviews_per_month'].sum())

room_vs_reviews = pd.DataFrame({
    'Room Type': rooms,
    'Reviews': total_reviews
})

room_vs_reviews
```

	Room Type	Reviews
0	Entire home/apt	26525.13
1	Private room	25415.29
2	Shared room	1232.65

What all manipulations have you done and insights you found?

This is a large dataset, having effiecient information for analysis. Here, locations play a very crucial role in these values and the inforamtion was distributed based on locations and their respective areas.

We have done number of manipulations to seek information which can be beneficial for the stake holders to make decisions for the business, not only that our hosts will also get good idea about the preferences of their customers, as we promised in our agenda.

We divided the complete manipulation into 2 major parts based on features and the relationships, for gaining insights accordingly.

These are the manipulations we followed:-

- Created a new price range column: to categorize the price distribution and to simplify the judgement to get an idea of the budget of our customer
- Sorted the dataset as per price: As the data was neither sorted nor having any significance regaring any value, we sorted it as per the pricing, keep the most expensive ones on the top.
- We filtered the data so that it becomes simple and easy to understand and work only with the required columns for the data analysis and feature engineering.
- Divided the data accoring to the categorical groups to derive insights accordingly, it helped us to identify the outcomes in a structed manner with respect to the considered groups, it provided us with more specific information about the different groups to identify insights for each group individually.

These are the diferent groups we created and worked with:- Created Nighbourhood Group: This group helped us to get an idea of which neighbourhood is the most prefered one in all of the neighbourhoods.

Created room type groups: This group helped us to identify which room type is the most prefered by our customers.

Created Neighbourhood area groups: This group helped us to identify which is the most prefered area within the neighbourhoods.

We also worked on few of the relationships that helped us in few comparisons that will help our stake holders to make decisions accordingly. These are the relationships we worked with:-

Relationship: Price vs. Room Type

Checked the distribution of prices for different room types. Determined which room type is having the most expensive price distribution. Relationship: Reviews per Month vs. Room Type

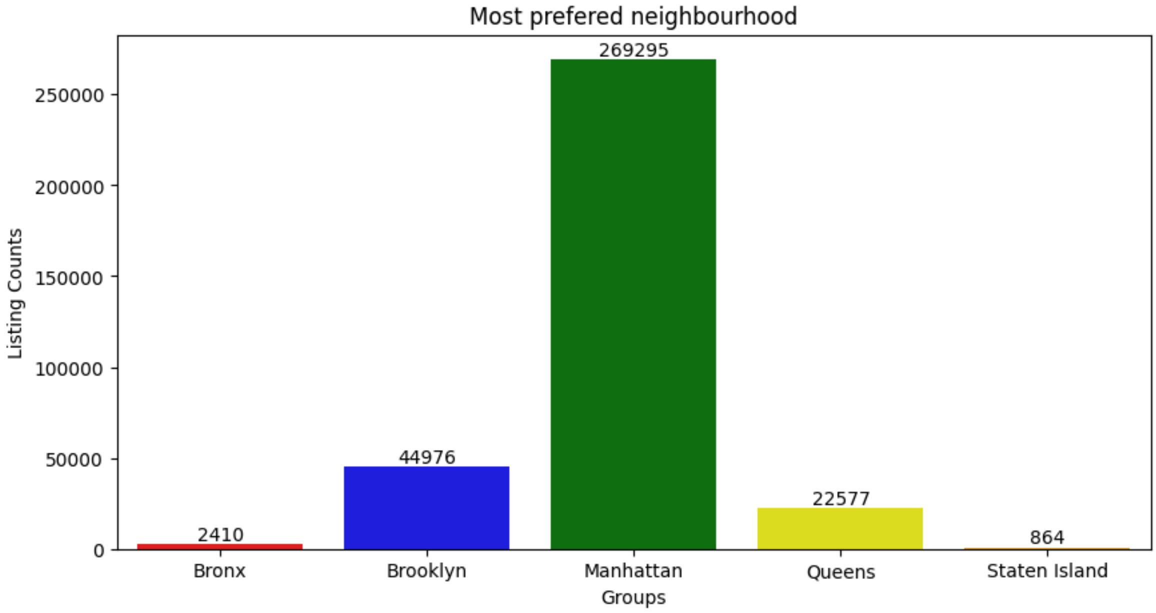
Compared the distribution of reviews per month for different room types. Explored the level of engagement and satisfaction of our customers as per different room types. Relationship: Price vs. Neighbourhood

Compared the distribution of prices across different neighbourhoods. Identified neighbourhoods with higher or lower average prices and explored price variations as per the areas. These are the manipulations that we did in order to derive useful information so that it can contribute into the growth of our business and also the busniness of our hosts, and ultimately grow and improve tavelling experience for our customers.

4. Data Vizualization, Storytelling & Experimenting with charts : Understand the relationships between variables

Chart - 1

```
# Let's visualize the most prefered neighbourhood group with a bar chart

group_feat_df

plt.figure(figsize=(10,5))

colors = ['red', 'blue', 'green', 'yellow', 'orange'] # To make each bar with a different color
sns.barplot(x="Group", y='Listing Count', data=group_feat_df, hue='Group', palette=colors, legend=False)
plt.xlabel('Groups')
plt.ylabel('Listing Counts')
plt.title('Most prefered neighbourhood')
for x, y in zip(range(len(groups)), listings):
    plt.text(x, y, f'{y}', ha='center', va='bottom') # To annotate each bar with the exact value
```



1. Why did you pick the specific chart?

Considering the highest preference group, we need to compare the values attained by each group. For comparison, bar chart is the best option.

2. What is/are the insight(s) found from the chart?

From the analysis we can clearly see that Manhattan is the most prefered group among all and it is outperforming all the other groups with a huge difference, Brooklyn is the 2nd prefernce of our customers followed by Manhattan.

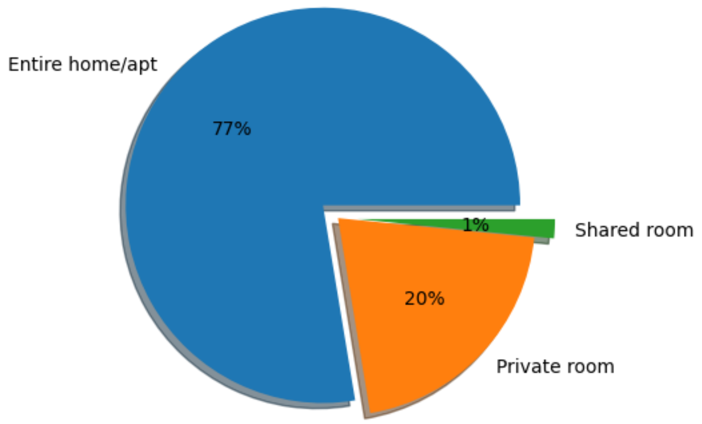
3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Absolutely, as we know what are the preferences of our customers we will be able to work on those things that are mostly in demand and we will be able to meet their requirements for their satisfaction, as here in this case we know that the most prefered group is Manhattan, we can target our customers with the availabilites in that area.

If we talk about the negative growth, it is as there is low demand in the other neighbourhoods, however we can handle if we try to find out why is that, the most prefered group is Manhattan, if we do that we will be able to identify the cause for low demad in those areas, we need to focus on the reasons and the difference so that we can get to the roots of this.

Chart - **2**

```
# Let's visualize the most prefered room type using a pie chart
room_feat_df

plt.pie(room_feat_df['Listing Count'], labels=room_feat_df['Group'], autopct="%1i%%", explode=(0.1, 0, 0.1), shadow=True)
plt.show()
```



1. Why did you pick the specific chart?

There are 3 types of rooms. To understand the distribution of the listings on a pie chart and also to see the difference between the room types. In this we are also able to see the difference in percentage which gave us a wider view on the data outcome.

2. What is/are the insight(s) found from the chart?

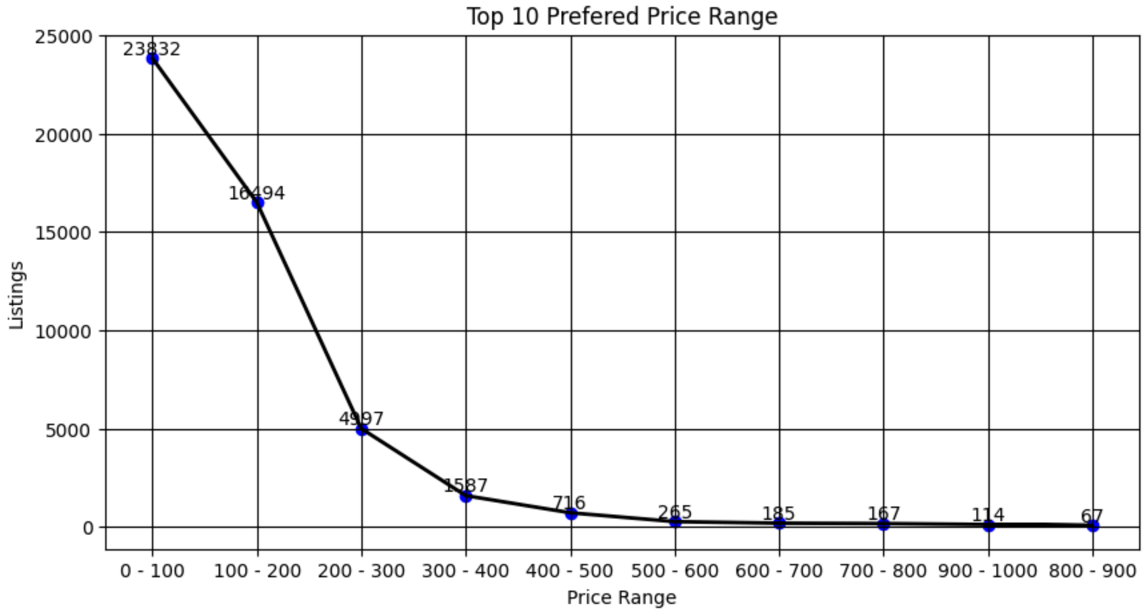
It is clearly visible that the "Entire Home/Apt" room type is the most prefered one, this indicates that the people are prioritising without having any type interference, as we see that the shared rooms are the least prefered. Those are mostly prefered by the students that come from outside so that their accommodation can be affordable. This information can be easily used for more specific target approaches.

3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Based on customer's preferences, it will be able to spend money on advertisements more efficiently which will help us to minimize our cost wastage, not only for the majority as what sort of customers prefer the other room types, we will be able to target them with their needs which will increase our consumer market even in the low demand sectors.

Chart - 3

```
# Now let's check which price range has the most number of listings, and plot the top 10 most prefered price range on chart.
prefered_range = airbnb_df['price_range'].value_counts().head(10)
plt.figure(figsize=(10,5))

prefered_range_sorted = prefered_range.sort_index()
plt.scatter(prefered_range.index, prefered_range, color='blue', marker='o')
plt.plot(prefered_range_sorted.index, prefered_range_sorted, color='black', linestyle='-', linewidth=2, label='Line Connecting Dots')

for x, y in zip(prefered_range_sorted.index, prefered_range_sorted): # To annotate each plotted value
    plt.text(x, y, f'{y}', ha='center', va='bottom')
plt.title("Top 10 Prefered Price Range")
plt.xlabel("Price Range")
plt.ylabel("Listings")
plt.grid(color='black')
```



Why did you pick the specific chart?

The plot chart can give us the graphical representation of the preferences of our customers regarding prices and how do they go through each of the price ranges. As we can see the chart is giving us a fall in the preferences as the price range increases.

2. What is/are the insight(s) found from the chart?

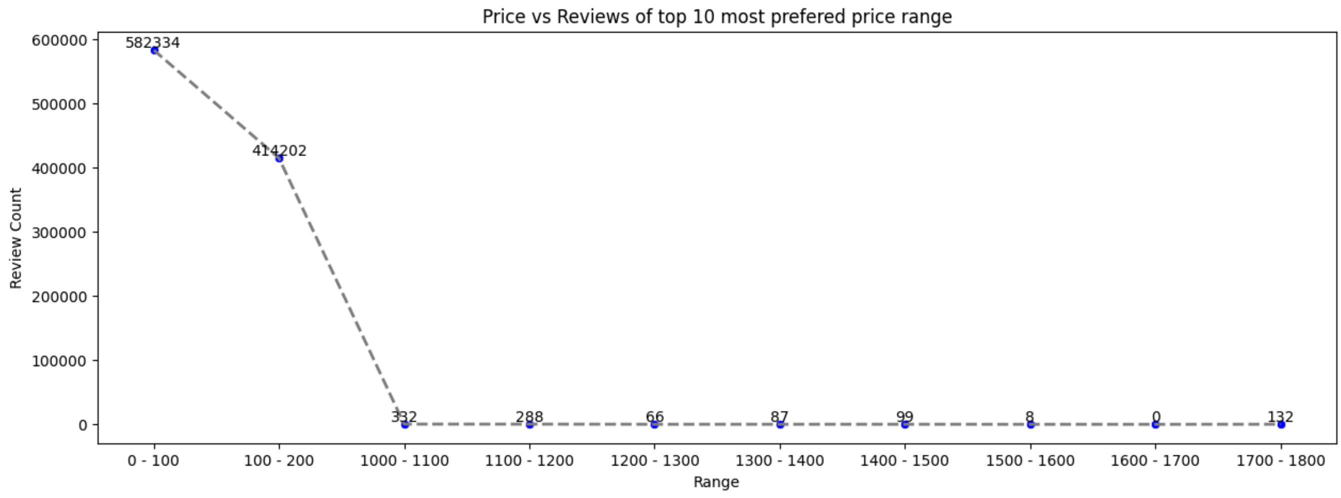
Through the chart we can cleary see the budget preferences of our customers and their most prefered price range that is 0-100. This can give us an idea on the spending power of our customers which will help us in setting better and more specific prices, also at the time of listings we can even make recommendations to our customers with their prefered price ranges.

3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Definitely, as customers like personalized interfaces and options, if we will focus on providing them what they prefer, it would be really appreciated by them as they will not have to go through a lot while searching for things they are looking for specifically.

Chart - 4

```
# Let's visualize the relationship between listing price and the number of reviews received.

sorted_range = feature_df.sort_values(by='price_range', ascending=False)
sorted_range = sorted_range.groupby('price_range')
range_group = []
review_count = []
for range, data in sorted_range:
  range_group.append(range)
  review_count.append(data['number_of_reviews'].sum())

range_df = pd.DataFrame({
    'Range': range_group,
    'Review Count': review_count
})
plt.figure(figsize=(15, 5))
sns.scatterplot(x='Range', y='Review Count', data=range_df.head(10), color='blue', legend=False, marker='o')
plt.plot(range_df['Range'].head(10), range_df['Review Count'].head(10), color='grey', linestyle='--', linewidth=2, label='Line Connectir
plt.title("Price vs Reviews of top 10 most prefered price range")
for x, y in zip(range_df['Range'].head(10), range_df['Review Count'].head(10)):
  plt.text(x, y, f'{y}', ha='center', va='bottom')
```



1. Why did you pick the specific chart?

As the difference between the no. of reviews in the price range is very huge, using a plot chart is quiet handy so that the pointers can be seen clearly with respect to their values and also their differences.

2. What is/are the insight(s) found from the chart?

As the price range and the no. of reviews are inversely proportional, it cleary states that the budget of our majortiy customer base lies between the range 0 - 200. We can also see that there are few preferences in the higher budget section as well where there is a competetion in the prices.

3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

The results suggests that it is posiible to make our pricing policies more targeted and more specific resulting in increasing customer base by providing them with the prices that are under their budget.

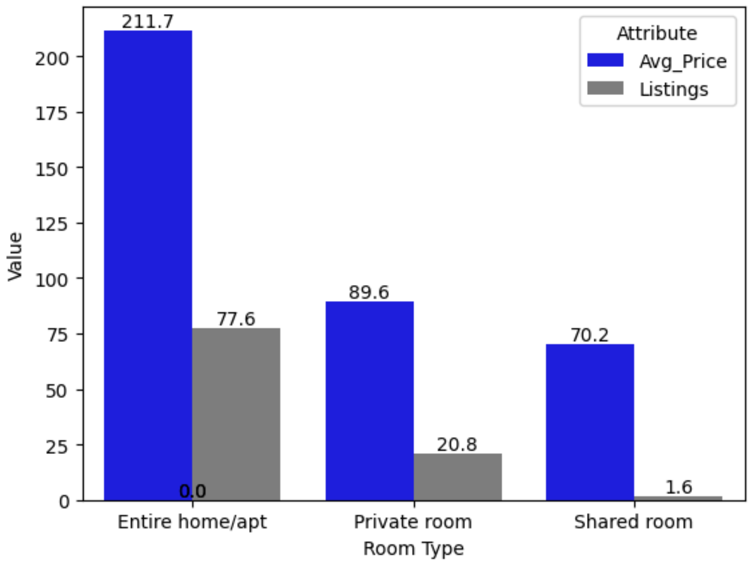
Chart - 5

1. Why did you pick the specific chart?

Double-click (or enter) to edit

```
# Chart - 4 visualization code
# Let us now visualize the average price division with room types.

data = {
    'Room Type': room_vs_price['Room Type'],
    'Avg_Price': room_vs_price['Avg_Price'],
    'Listings': ((room_feat_df['Listing Count']/room_feat_df['Listing Count'].sum())*100)
}

# Creating a DataFrame
df = pd.DataFrame(data)

df_melted = pd.melt(df, id_vars='Room Type', var_name='Attribute', value_name='Value')
# Creating a strip plot
sns.barplot(x='Room Type', y='Value', data=df_melted, hue="Attribute", palette=['blue', 'grey'], legend=True)

# Applying annotations on the values
for p in plt.gca().patches:
    plt.gca().annotate('{:.1f}'.format(p.get_height()), (p.get_x() + p.get_width() / 2., p.get_height()),
                       ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                       textcoords='offset points')
```

```
plt.show()
```



1. Why did you pick the specific chart?

As there are 3 types of rooms, it was better to use the bar chart for better visualizing the divisions of the average price and the percentage of the listings divided among these values altogether.

2. What is/are the insight(s) found from the chart?

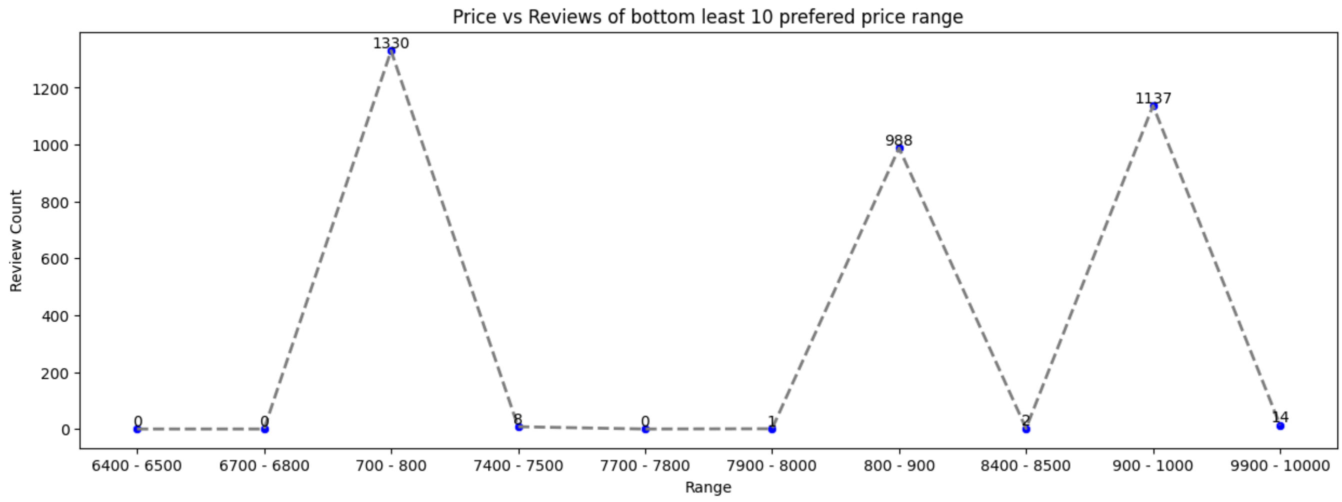
As we can clearly see that the average pricing the listing counts has a direct relationship, the highest pricing is in the Entire Home/Apt room type, and for that the listings division, which can be quite using in making decisions like pricing.

3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Yes, as we can understand what percentage is ready to pay which amount for their preferences we will be able to make better pricing strategies which will definitely assist us in the growth.

Chart - 6

```
# Let us also check for the least 10 prefered price ranges
plt.figure(figsize=(15, 5))
sns.scatterplot(x='Range', y='Review Count', data=range_df.tail(10), color='blue', legend=False, marker='o')
plt.plot(range_df['Range'].tail(10), range_df['Review Count'].tail(10), color='grey', linestyle='--', linewidth=2, label='Line Connectir
plt.title("Price vs Reviews of bottom least 10 prefered price range")
for x, y in zip(range_df['Range'].tail(10), range_df['Review Count'].tail(10)):
  plt.text(x, y, f'{y}', ha='center', va='bottom')
```



1. Why did you pick the specific chart?

As the difference between the no. of reviews in the price range is very huge, using a plot chart is quiet handy so that the pointers can be seen clearly with respect to their values and also their differences.

2. What is/are the insight(s) found from the chart?

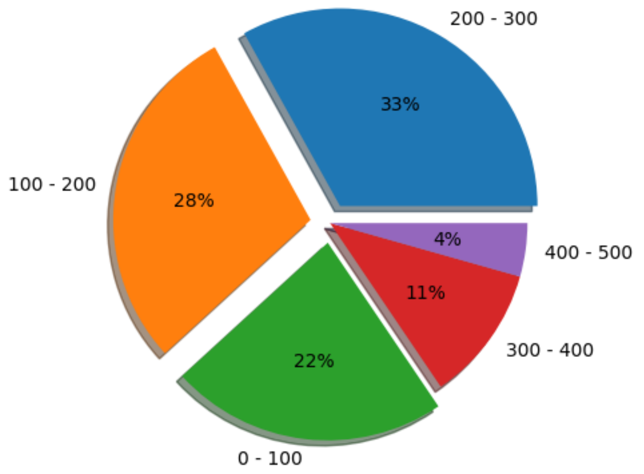
As the price range and the no. of reviews are inversely proportional, it cleary states that the budget of our majortiy customer base lies between the range 0 - 200. We can also see that there are few preferences in the higher budget section as well where there is a competetion in the prices.

3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

This suggests that we will be able to make our pricing policies more targeted and more specific resulting in increasing customer base by providing them with the prices that are under their budget.

Chart - 7

```
# Let us know the price range having the highest listings so as to get more specific idea on price preference.

sorted_range
range_group
range_listing_count = []

for range, data in sorted_range:
  range_listing_count.append(data['listings'].sum())


range_list_df = pd.DataFrame({
    'Range': range_group,
    'Listing Count': range_listing_count
})

range_list_df = range_list_df.sort_values(by='Listing Count', ascending=False).reset_index(drop=True)

# We will be taking on the top 5 prefered ranges.
plt.pie(range_list_df['Listing Count'].head(), labels=range_list_df['Range'].head(), autopct="%1i%%", explode=(0.1, 0.1, 0.1, 0, 0), shad
plt.show()
```



1. Why did you pick the specific chart?

Pie chart is an effective chat to clearly see the distrubutions and preferences as in proportions, it becomes easier to notice the division of listngs by the price range.

2. What is/are the insight(s) found from the chart?

As we can see form the chart:- The top 3 price ranges that are having the most listings are:-

- 200-300: 33%
- 100-200: 28%
- 0-100: 22%

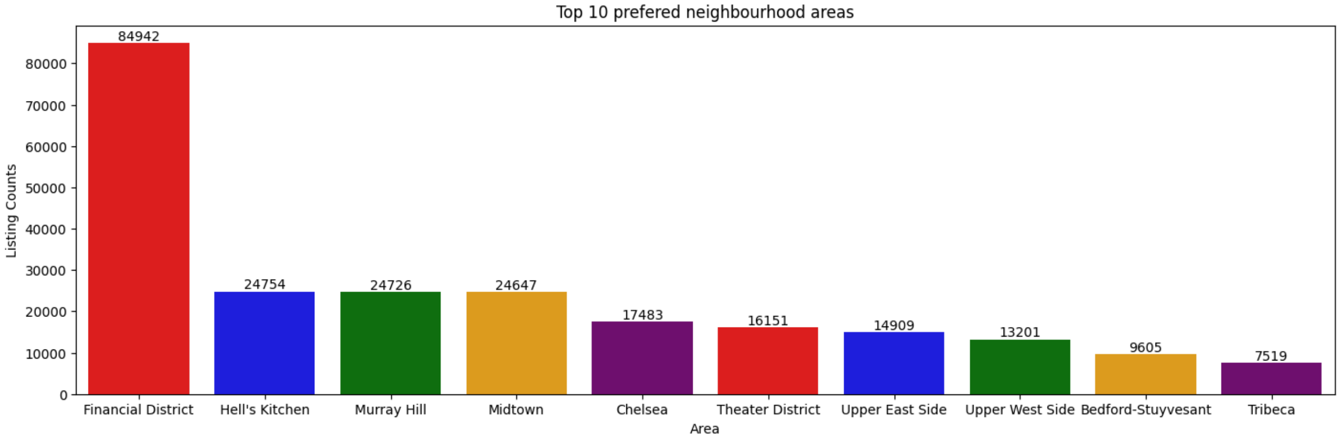
3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

The number of listings are closely divided among these sectors and this can be a great was to keep a track of all the hostels that are within this price range and create recommendations according to their prefered neighbourhoods.

Chart - **8**

```
# Chart - 7 visualization code
# As now we are having the price ranges with the most listings let us check the neighbourhoods with the most listings.

areas = []
area_listing = []
area_group = feature_df.groupby('neighbourhood')
for area, data in area_group:
    areas.append(area)
    area_listing.append(data['listings'].sum())

area_data_df = pd.DataFrame({
    "Area": areas,
    "Values": area_listing
})

area_data_df = area_data_df.sort_values(by='Values', ascending=False).reset_index(drop=True)
area_data_df
plt.figure(figsize=(17,5))
# plt.plot(groups,listings, color='black', marker = "o", markerfacecolor = 'blue', markeredgecolor='blue',linestyle='-')
colors = ['red', 'blue', 'green', 'orange', 'purple']
sns.barplot(x="Area", y='Values', data=area_data_df.head(10), hue="Area", palette=colors, legend=False)
plt.title('Top 10 prefered neighbourhood areas')
plt.xlabel('Area')
plt.ylabel('Listing Counts')
for x, y in zip(area_data_df['Area'].head(10), area_data_df['Values'].head(10)):
    plt.text(x, y, f'{y}', ha='center', va='bottom')
```

```
<ipython-input-51-30fc849892e1>:21: UserWarning:
The palette list has fewer values (5) than needed (10) and will cycle, which may produce an uninterpretable plot.
  sns.barplot(x="Area", y='Values', data=area_data_df.head(10), hue="Area", palette=colors, legend=False)
```



1. Why did you pick the specific chart?

The bar chart clearly reflects the differece between the areas and the gap between them as per the listing counts, we can see the top ten most listed neighbourhoods.

2. What is/are the insight(s) found from the chart?

We can see that the most listed neighbourhood area is the 'Financial District' and it is also having a major gap between the other ones in the list, we can clearly see that the Financial District is the most prefered neighbourhood of our customers.

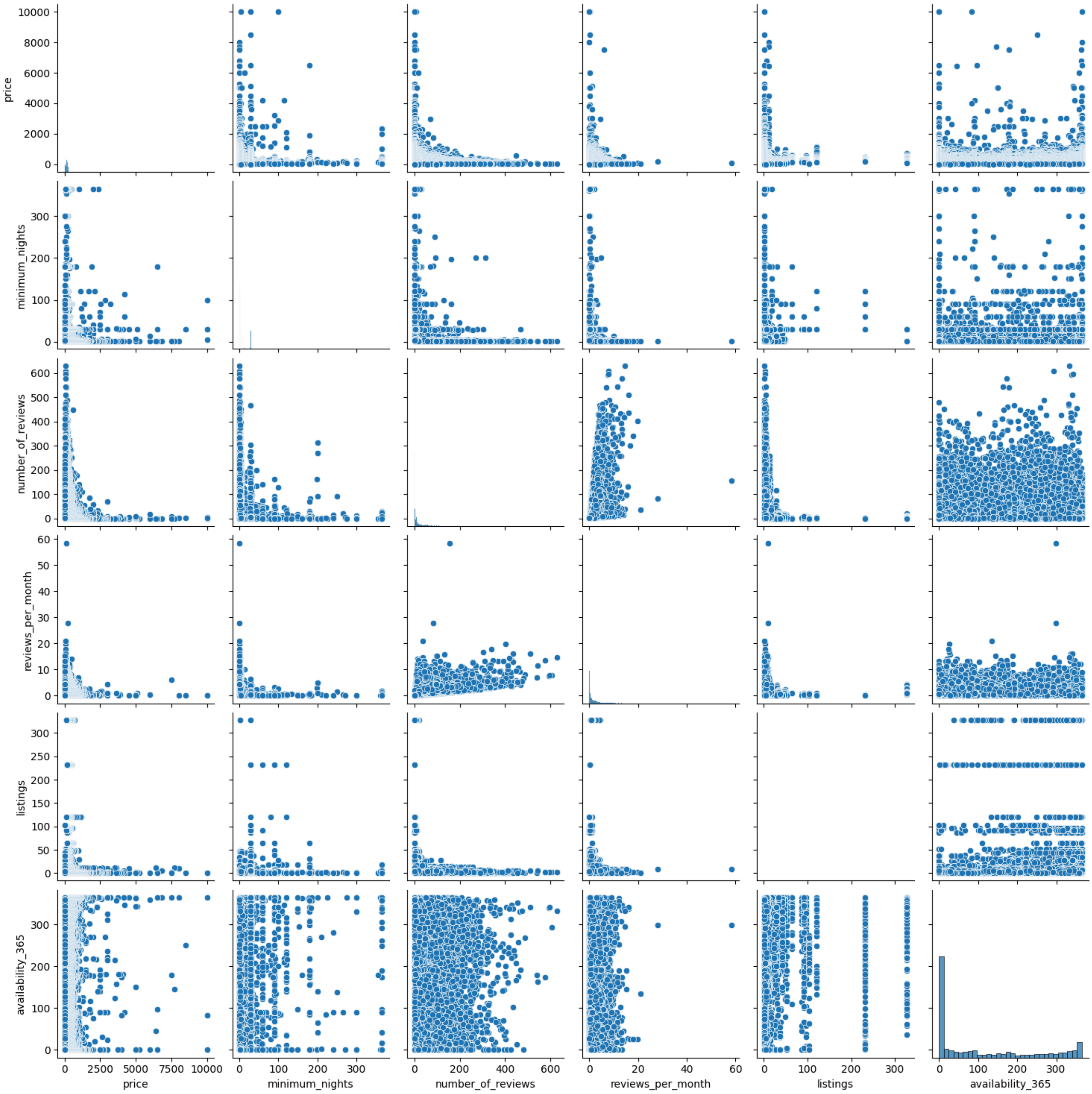
3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Definitely, as this insight suggests the areas where there is a great scope for business and we can create more opportunities and also increase our Advertisements in a more targgetted manner.

Chart - **9**

```
# Chart - 9 visualization code
# Let's create a pairplot to know the relationship between our few variables.

pair_df = feature_df[['price', 'minimum_nights', 'number_of_reviews', 'reviews_per_month', 'listings', 'availability_365', 'price_range
sns.pairplot(pair_df)
```

```
<seaborn.axisgrid.PairGrid at 0x7b24270a2350>
```



1. Why did you pick the specific chart?

Here we can easily see the distribution of our numerical variables, it will also give us an overview of our dataset and the relationships between our variables.

2. What is/are the insight(s) found from the chart?