International Institute of Information Technology Bangalore

upGrad

STORYTELLING CASE STUDY : AIRBNB NYC

NAME : PRITKUMAR PARMAR

OVERVIEW

- OBJECTIVE
- DATA LIFE-CYCLE
- ANALYSIS METHODS
- CONCLUSION
- APPENDIX

OBJECTIVE

- To handle an analysis of NewYork AirBNB dataset.
- Detect meaningful insights from dataset
- Prepare data to do Data Visulisation and extract important insights.

DATA LIFE-CYCLE

- In first stage, load the data and import the necessary libraries
- In second stage, Clean the data and do the needful opeartions for null values
- In the third stage, Prepare the data for Data Visulisation/EDA
- In fourth stage, extract meaningful insights from data

IMPORT LIBRARIES AND LOAD DATA

In [448]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [449]:
```
inp0 = pd.read_csv('AB_NYC_2019.csv')
inp0.head()
```

Out[449]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_revie
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	2
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	10	

NULL VALUE ANALYSIS

```
Out[450]: (48895, 16)

In [451]: inp0.columns

Out[451]: 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')

In [452]: inp0.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

In [453]: inp0.isnull().sum()

Out[453]: 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
```

CREATING FEATURES OF CATEGORICAL VARS

2.1 CATEGORISING 'AVAILABILTY 365' FEATURE INTO 5 CATEGORIES

In [455]:
```
def availability_365_categories_function(row):

    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'
```

In [456]:
```
inp0['availabilty_365_categories'] = inp0.availability_365.map(availability_365_categories_function)
inp0['availabilty_365_categories'].head(15)
```

Out[456]:
```
0     very High
1     very High
2     very High
3        Medium
4      very Low
5        Medium
6      very Low
7          High
8      very Low
9        Medium
10          Low
11          Low
12    very High
13    very High
14     very Low
Name: availabilty_365_categories, dtype: object
```

In [457]:
```
inp0.availabilty_365_categories.value_counts()
```

Out[457]:
```
very Low     17941
Low          11829
very High     8108
Medium        5792
High          5225
Name: availabilty_365_categories, dtype: int64
```

2.2 CATEGORISING 'MINIMUM NIGHTS' COLUM INTO 5 CATEGORIES

In [458]:
```
def minimum_night_categories_function(row):

    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'
```

In [459]:
```
inp0['minimum_nights_categories'] = inp0.minimum_nights.map(minimum_night_categories_function)
inp0['minimum_nights_categories'].head(15)
```

Out[459]:
```
0      very Low
1      very Low
2           Low
3      very Low
4     very High
5           Low
6     very High
7           Low
8           Low
9      very Low
10       Medium
11          Low
12       Medium
13          Low
14    very High
Name: minimum_nights_categories, dtype: object
```

In [460]:
```
inp0.minimum_nights_categories.value_counts()
```

Out[460]:
```
Low          19695
very Low     12720
very High     7333
Medium        6337
High          2810
Name: minimum_nights_categories, dtype: int64
```

2.3 categorizing the "number_of_reviews" column into 5 categories

In [461]:
```
def number_of_reviews_categories_function(row):

    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'
```

In [462]:
```
inp0['number_of_reviews_categories'] = inp0.minimum_nights.map(number_of_reviews_categories_function)
inp0['number_of_reviews_categories'].head(10)
```

Out[462]:
```
0     very Low
1     very Low
2          Low
3     very Low
4       Medium
5          Low
6     very High
7          Low
8          Low
9     very Low
Name: number_of_reviews_categories, dtype: object
```

In [463]:
```
inp0.number_of_reviews_categories.value_counts()
```

Out[463]:
```
Low          26032
very Low     12720
High          5893
Medium        3503
very High      747
Name: number_of_reviews_categories, dtype: int64
```

In [466]:
```
inp0[inp0.price == 0].shape
```

Out[466]:
```
(11, 19)
```

In [467]:
```
def price_categories_function(row):

    if row <= 1:
        return 'very Low'
    elif row <= 4:
        return 'Low'
    elif row <= 15 :
        return 'Medium'
    elif (row <= 100):
        return 'High'
    else:
        return 'very High'
```

In [468]:
```
inp0['price_categories'] = inp0.minimum_nights.map(price_categories_function)
inp0['price_categories'].head(10)
```

Out[468]:
```
0     very Low
1     very Low
2          Low
3     very Low
4       Medium
5          Low
6         High
7          Low
8          Low
9     very Low
Name: price_categories, dtype: object
```

In [469]:
```
inp0.price_categories.value_counts()
```

Out[469]:
```
Low          22998
very Low     12720
Medium        7556
High          5447
very High      174
Name: price_categories, dtype: int64
```

FIXING COLOMS

3. FIXING COLUMNS

In [470]:
```
inp0.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 20 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  
 16  availabilty_365_categories      48895 non-null  object 
 17  minimum_nights_categories       48895 non-null  object 
 18  number_of_reviews_categories    48895 non-null  object 
 19  price_categories                48895 non-null  object 
dtypes: float64(3), int64(7), object(10)
memory usage: 7.5+ MB
```

In [471]:
```
inp0.last_review.dtype
```

Out[471]:
```
dtype('O')
```

In [472]:
```
inp0.last_review = pd.to_datetime(inp0.last_review)
inp0.last_review
```

```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_15496\875808853.py:1: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consistent parsing.
  inp0.last_review = pd.to_datetime(inp0.last_review)
```

Out[472]:
```
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]
```

In [473]:
```
inp0.last_review.dtypes
```

Out[473]:
```
dtype('<M8[ns]')
```

In [474]:
```
print([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',
```

DATA TYPES

4. DATA TYPES

4.1 CATEGORICAL COLOMS

In [475]:
```
inp0.columns
```

Out[475]:
```
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', 'availabilty_365_categories',
       'minimum_nights_categories', 'number_of_reviews_categories',
       'price_categories'],
      dtype='object')
```

In [476]:
```
categorical_columns = inp0.select_dtypes(include=['object']).columns.tolist()
categorical_columns
```

Out[476]:
```
['name',
 'host_name',
 'neighbourhood_group',
 'neighbourhood',
 'room_type',
 'availabilty_365_categories',
 'minimum_nights_categories',
 'number_of_reviews_categories',
 'price_categories']
```

4.2 NUMERICAL COLOMS

In [478]:
```
numerical_columns = inp0.select_dtypes(include=['int64']).columns.tolist()
numerical_columns
```

Out[478]:
```
['id',
 'host_id',
 'price',
 'minimum_nights',
 'number_of_reviews',
 'calculated_host_listings_count',
 'availability_365']
```

In [479]:
```
inp0[numerical_columns].head()
```

Out[479]:

	id	host_id	price	minimum_nights	number_of_reviews	calculated_host_listings_count	availability_365
0	2539	2787	149	1	9	6	365
1	2595	2845	225	1	45	2	355
2	3647	4632	150	3	0	1	365
3	3831	4869	89	1	270	1	194
4	5022	7192	80	10	9	1	0

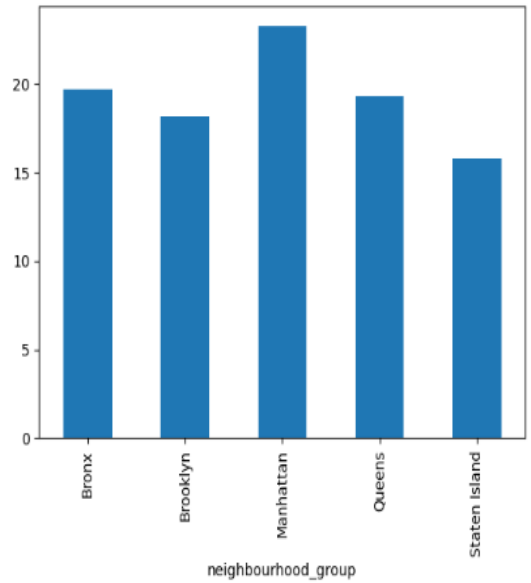
MISSING VALUE ANALYSIS

5.2 MISSING VALUE ANALYSIS (NEIGHBOURHOOD GROUP FEATURE)

In [485]:
```
inp0.neighbourhood_group.value_counts(ascending=False)
```

Out[485]:
```
Manhattan        21661
Brooklyn         20104
Queens            5666
Bronx             1091
Staten Island      373
Name: neighbourhood_group, dtype: int64
```

In [486]:
```
((inp1.groupby('neighbourhood_group').neighbourhood_group.count()/inp0.groupby('neighbourhood_group').neighbourhood_group.count(
plt.show()
```



In [487]:
```
((inp1.groupby('neighbourhood_group').neighbourhood_group.count()/inp0.groupby('neighbourhood_group').neighbourhood_group.count(
```

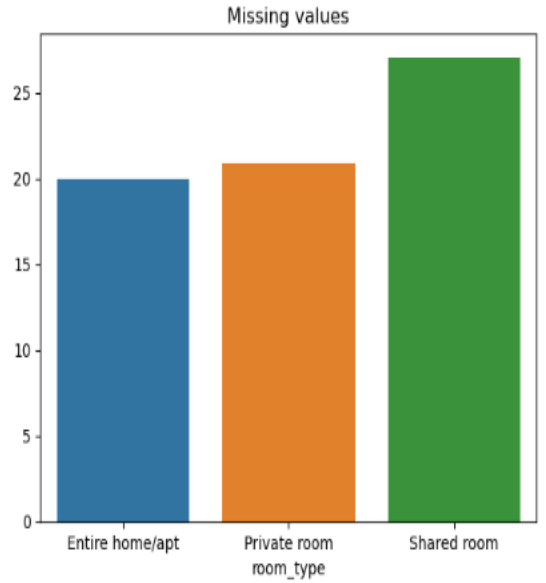
Out[487]:
```
19.240898461107257
```

5.3 MISSING VALUE ANALYSIS (ROOM TYPE FEATURE)

In [488]:
```
# Count of 'room_type' with missing values
inp3 = (inp1.groupby('room_type').room_type.count()/inp0.groupby('room_type').room_type.count())*100
inp3
```

Out[488]:
```
room_type
Entire home/apt    19.981109
Private room       20.877004
Shared room        27.068966
Name: room_type, dtype: float64
```

In [489]:
```
plt.title('Missing values')
sns.barplot(x = inp3.index, y = inp3.values)
plt.show()
```

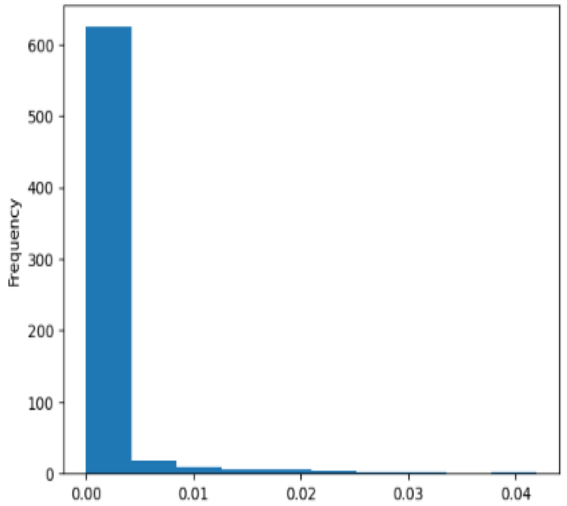


'Shared room' has the highest missing value percentage (27 %) for 'last_review' feature while to other room types has only about 20 %.

UNIVARIATE ANALYSIS

6.7 Price

In [510]:
```
inp0.price.value_counts(normalize=True).plot.hist()
plt.show()
```



In [511]:
```
sns.distplot(inp0.price,kde=True)
plt.show()
```

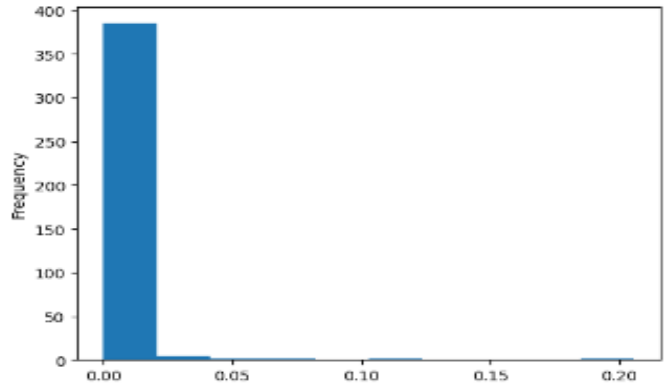
```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_15496\4203382358.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(inp0.price,kde=True)
```

6.9 NUMBER OF REVIEWS

In [516]:
```
inp0.number_of_reviews.describe()
```

Out[516]:
```
count    48895.000000
mean        23.274466
std         44.550582
min          0.000000
25%          1.000000
50%          5.000000
75%         24.000000
max        629.000000
Name: number_of_reviews, dtype: float64
```

In [517]:
```
inp0.number_of_reviews.value_counts(normalize=True).plot.hist()
plt.show()
```



In [518]:
```
sns.distplot(inp0.number_of_reviews)
plt.show()
```

```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_15496\4258575927.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(inp0.number_of_reviews)
```

BIVARIATE/MULTIVARIATE ANALYSIS

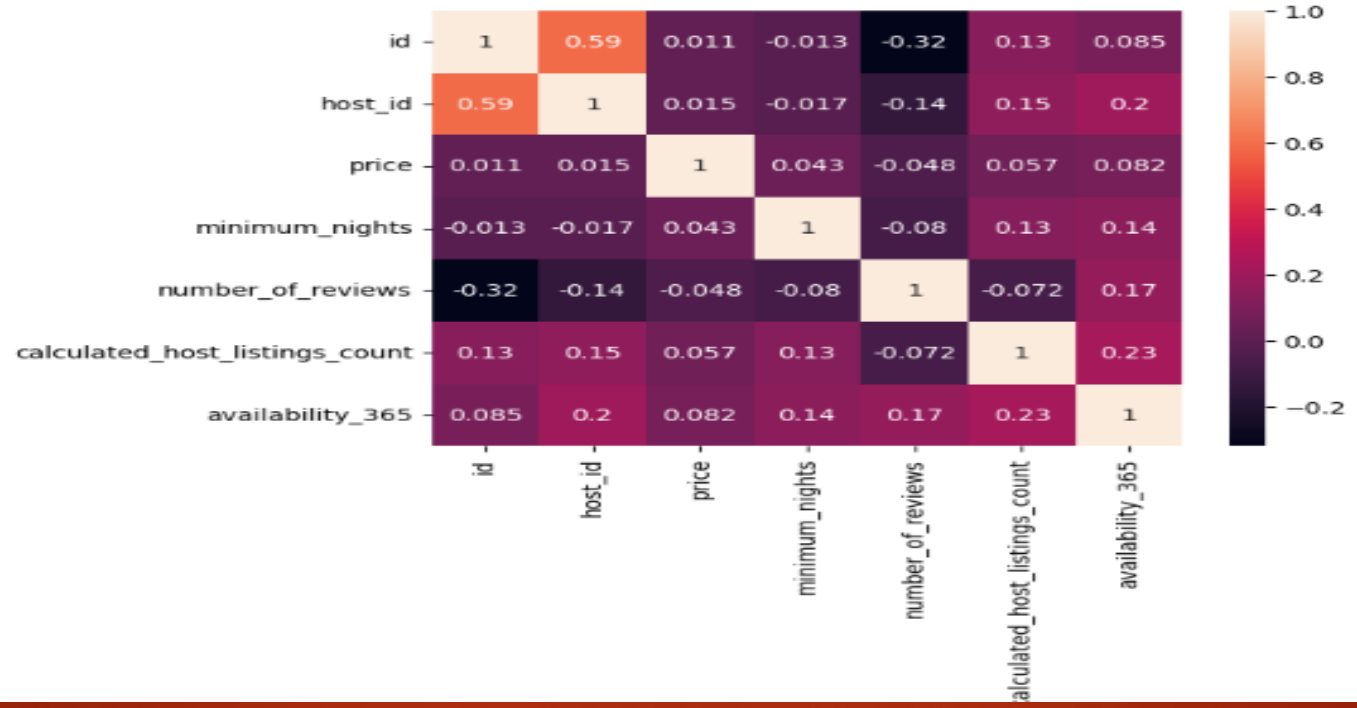
7. BIVARIATE AND MULTIVARIATE ANALYSIS

In [528]:
```
inp0[numerical_columns].head()
```

Out[528]:

	id	host_id	price	minimum_nights	number_of_reviews	calculated_host_listings_count	availability_365
0	2539	2787	149	1	9	6	365
1	2595	2845	225	1	45	2	355
2	3647	4632	150	3	0	1	365
3	3831	4869	89	1	270	1	194
4	5022	7192	80	10	9	1	0

In [529]:
```
res = inp0[numerical_columns].corr()
sns.heatmap(res, annot=True)
plt.show()
```



CONCLUSION

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

APPENDIX : DATA SOURCE

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 : FEATURE DATATYPE

```
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
```


Thank You !