AIRBNB, NYC STORYTELLING CASE STUDY

DATA INSIGHTS

RUSHABH PATEL

C59 BATCH, 2024

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.

Importing libraries and reading the data

Importing libraries and reading data:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data1 = pd.read_csv('AB_NYC_2019.csv')
data1.head()
```

Out[2]:

	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	

Creating features

Categorizing the "price" column into 5 categories:

```
def price_cat_func(row):
    """
    Categorizes the "number_of_reviews" column into 5 categories
    """
    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'
```

```
data1['price_categories'] = data1.minimum_nights.map(price_cat_func)
data1['price_categories']
```

```
0        Very Low
1        Very Low
2             Low
3        Very Low
4          Medium
```

Categorization:

Categorizing the "availability_365" column into 5 categories:

```
def availability_365_cat_func(row):
    """
    Categorizes the "minimum_nights" column into 5 categories
    """
    if row <= 1:
        return 'Very Low'
    elif row <= 100:
        return 'Low'
    elif row <= 200 :
        return 'Medium'
    elif (row <= 300):
        return 'High'
    else:
        return 'Very High'
```

```
data1['availability_365_categories'] = data1.availability_365.map(availability_365_cat_func)
data1['availability_365_categories']
```

```
0        Very High
1        Very High
2        Very High
3           Medium
4         Very Low
```

Categorizing the "minimum_nights" column into 5 categories:

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

```
data1['minimum_night_categories'] = data1.minimum_nights.map(min_night_cat_func)
data1['minimum_night_categories']
```

```
0         Very Low
1         Very Low
2              Low
3         Very Low
4        Very High
           ...
```

EDA

EDA:

4]:
```
## let's check null counts and data types:

data1.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
```

```
data1.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  datetime64[ns]
 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  availability_365_categories     48895 non-null  object
 17  minimum_night_categories        48895 non-null  object
 18  number_of_reviews_categories    48895 non-null  object
 19  price_categories                48895 non-null  object
dtypes: datetime64[ns](1), float64(3), int64(7), object(9)
memory usage: 7.5+ MB
```

Data Types

Categorical Data Types:

```
data1.columns
```

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

```
## Let's differentiate categorical columns:

categorical_columns = data1.columns[[0, 1, 3,4,5,8,16,17,18,19]]
categorical_columns
```

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

Numerical Data Types:

```
]: numerical_columns = data1.columns[[9,10,11,13,14,15]]
numerical_columns
```

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

```
]: data1[numerical_columns].head()
```

]:

	price	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	availability_365
0	149	1	9	0.21	6	365
1	225	1	45	0.38	2	355
2	150	3	0	NaN	1	365
3	89	1	270	4.64	1	194
4	80	10	9	0.10	1	0

```
]: data1.describe()
```

Missing Values

Missing value analysis:

```
# Selecting the data with missing values for 'last_review' feature

data2 = data1.loc[data1.last_review.isnull(),:]
data2
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	numbe
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150	3	
19	7750	Huge 2 BR Upper East Cental Park	17985	Sing	Manhattan	East Harlem	40.79685	-73.94872	Entire home/apt	190	7	
26	8700	Magnifique Suite au N de Manhattan - vue Cloitres	26394	Claude & Sophie	Manhattan	Inwood	40.86754	-73.92639	Private room	80	4	
36	11452	Clean and Quiet in Brooklyn	7355	Vt	Brooklyn	Bedford-Stuyvesant	40.68876	-73.94312	Private room	35	60	
38	11943	Country space in the city	45445	Harriet	Brooklyn	Flatbush	40.63702	-73.96327	Private room	150	1	

Missing Value Analysis

Missing values Analysis (MvA) ('neighbourhood_group' feature)

```
# Count of 'neighbourhood_group' with missing values

data2.groupby('neighbourhood_group').neighbourhood_group.count()
```

```
neighbourhood_group
Bronx             215
Brooklyn         3657
Manhattan        5029
Queens           1092
Staten Island      59
Name: neighbourhood_group, dtype: int64
```

```
# Count of 'neighbourhood_group'

data1.groupby('neighbourhood_group').neighbourhood_group.count()
```

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

```
(data2.groupby('neighbourhood_group').neighbourhood_group.count() /
 data1.groupby('neighbourhood_group').neighbourhood_group.count())*100
```

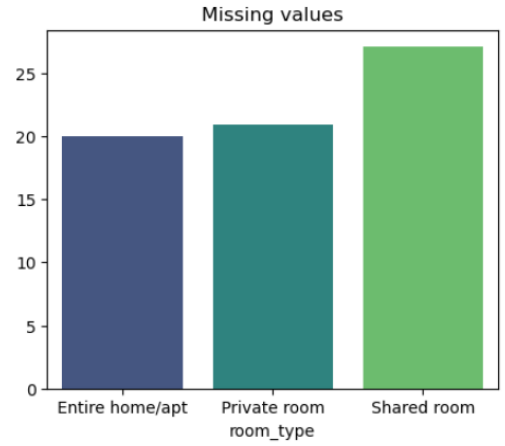
MvA 'room_type' feature

```
# Count of 'room_type' with missing values

data3 = (data2.groupby('room_type').room_type.count() / data1.groupby('room_type').room_type.count())*100
data3
```

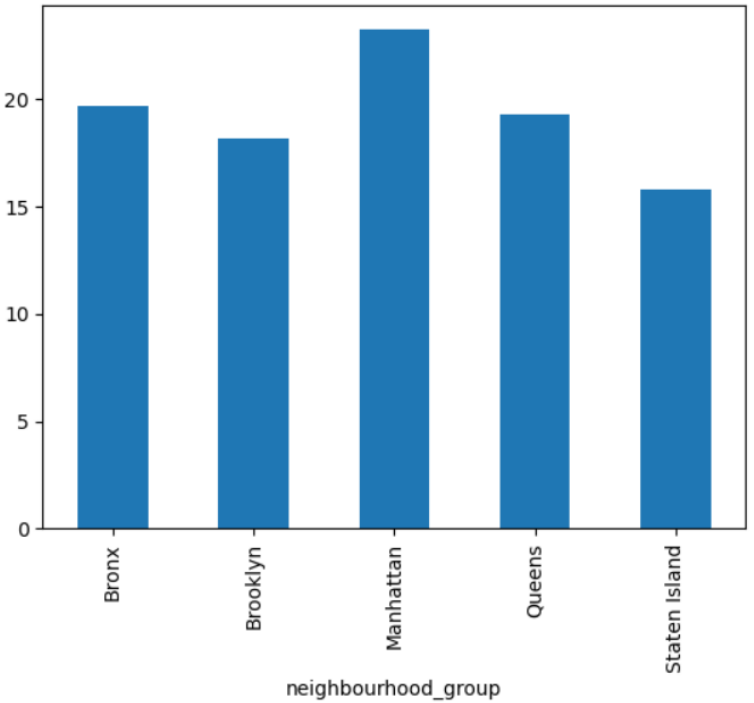
```
room_type
Entire home/apt    19.981109
Private room       20.877004
Shared room        27.068966
Name: room_type, dtype: float64
```

```
plt.figure(figsize=[5,4])
plt.title('Missing values')
sns.barplot(x = data3.index, y = data3.values, palette='viridis')
plt.show()
```



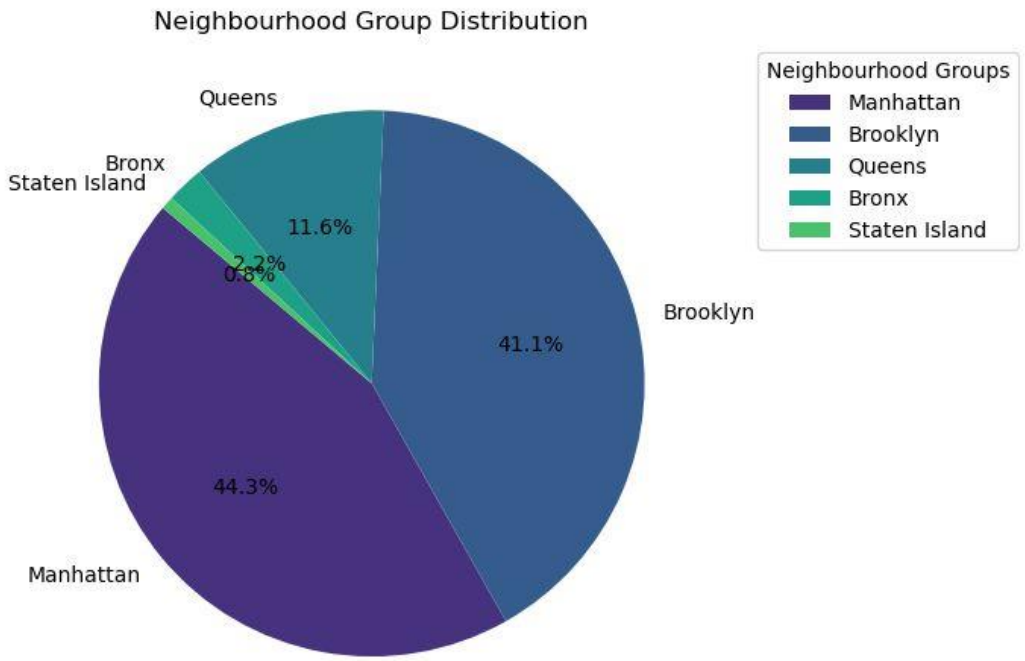

```
((data2.groupby('neighbourhood_group').neighbourhood_group.count() /
  data1.groupby('neighbourhood_group').neighbourhood_group.count())*100).plot.bar()
```

```
<Axes: xlabel='neighbourhood_group'>
```

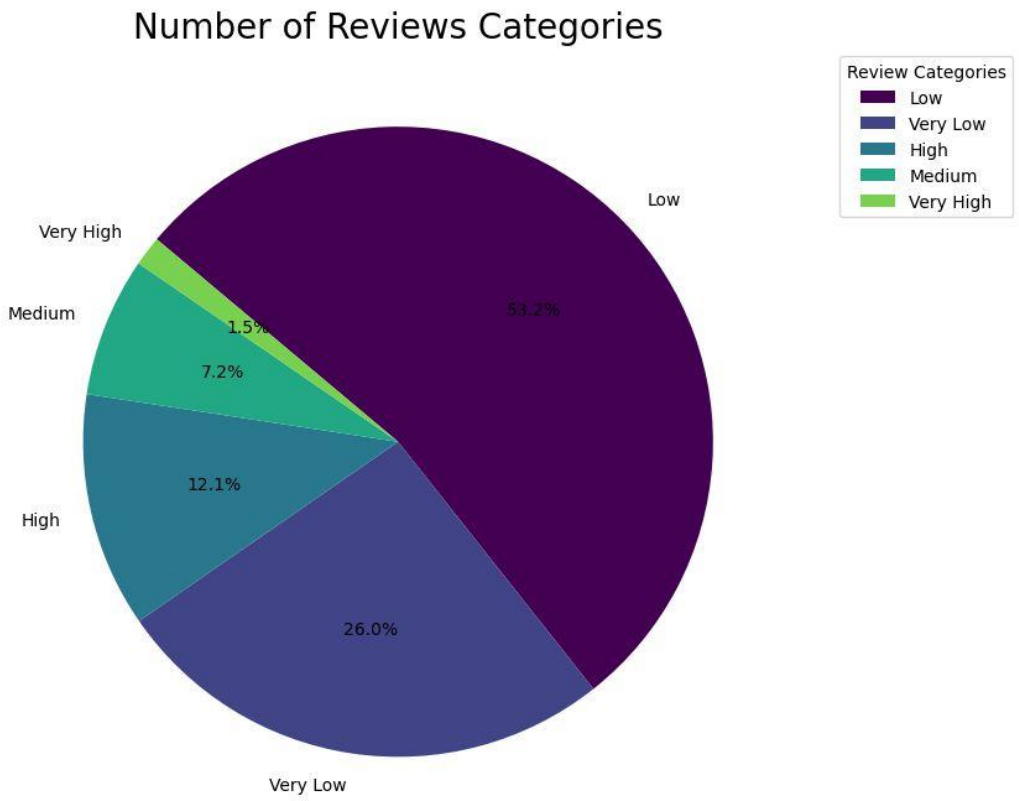


MOST CONTRIBUTING NEIGHBORHOODS

- 85 % of the listing are Manhattan and Brooklyn neighborhood group.
- Staten Island has the lowest contribution.



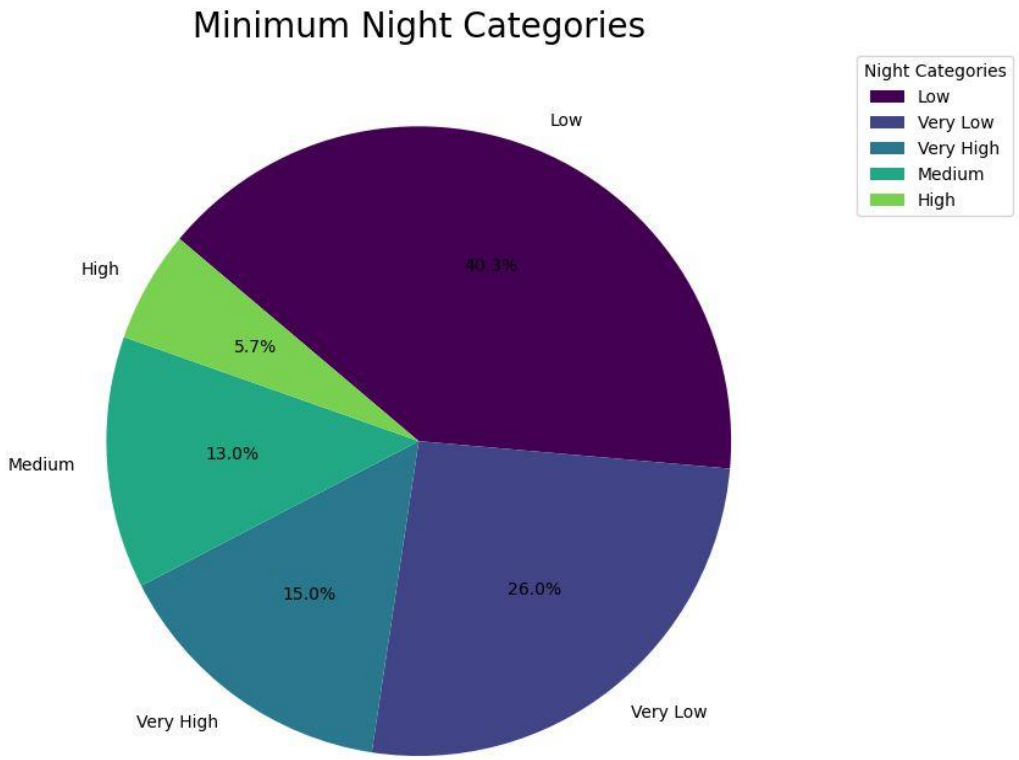
MINIMUM NIGHT CATEGORIES



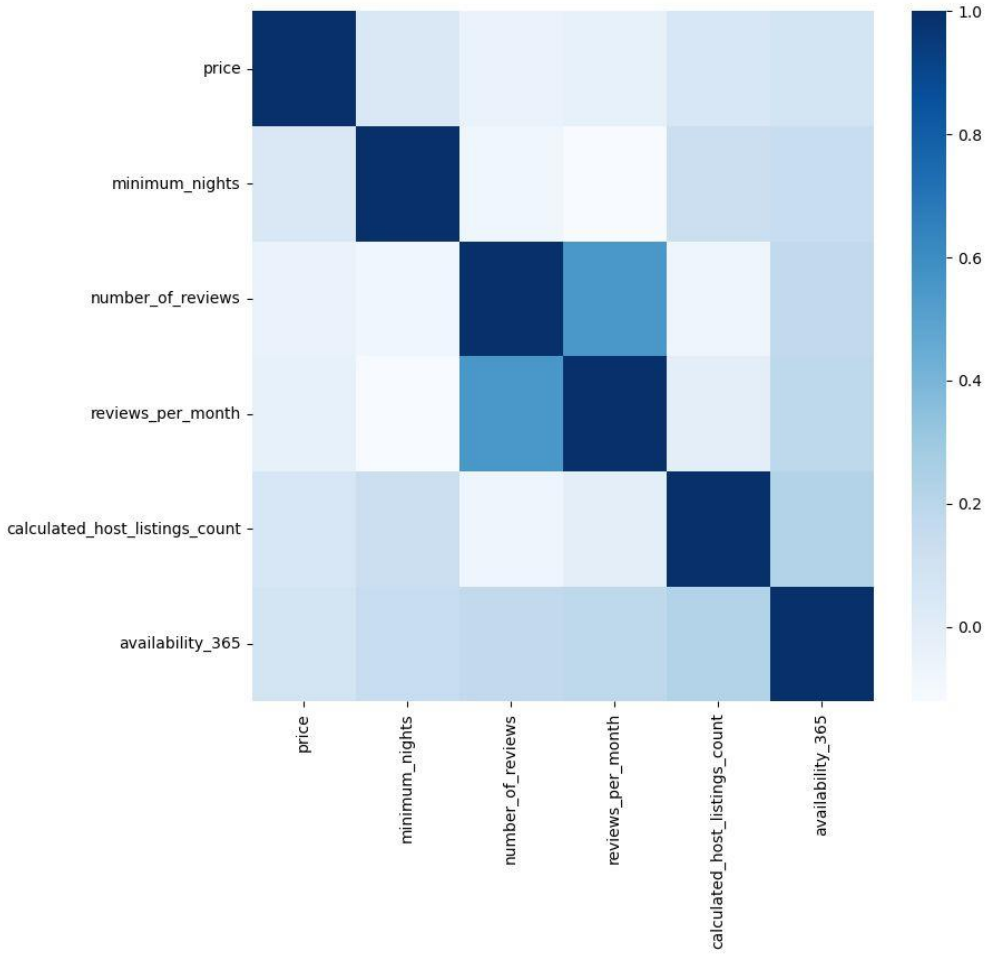
- Low category in minimum night feature contributes 40% and Very Low category contributes 26%.

EFFECT OF MINIMUM NIGHT ON REVIEWS

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



Bivariate and Multivariate Analysis



APPENDIX - DATA SOURCES

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

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

APPENDIX - DATA ASSUMPTIONS

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

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

Location Varibles:
    - latitude
    - longitude

Time Varibale:
    - last_review
```

APPENDIX –DATA METHODOLOGY

- Performed a comprehensive analysis of the New York Airbnb dataset, which included the following steps:

- Data Cleaning: Utilized Python to clean and preprocess the dataset.
- Feature Engineering: Extracted and derived essential features.
- Statistical Analysis: Employed group aggregation, pivot tables, and other statistical techniques.

CONCLUSION

Here are the insights presented in a different way:

- Significant insights have been derived from various dataset attributes.
- A wide range of visuals has been used in presentations for stakeholders.
- The data collection team should gather data on review scores to enhance future analysis.
- A clustering machine learning model can be developed to identify groups of similar objects in datasets with multiple variables.