Rental prices- Data Cleaning

Introduction

A dataset from a A housing rental company includes details about each property rented, Number of bedrooms, Number of bathrooms, as well as the price charged per night is provided. Data analysis of the the given data is done to answer the follwing questions:

1. What are main factors affect rental price?.
2. Do number of bathrooms has significant effect on price?.
3. Are there certain property or room types that have higher rental prices?.

A regression model is to be implemnted to help people estimate the money they could earn renting out their living space.

Data Wrangling

Assesing and cleaning Data

In [1]:
```
# Importing packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Loading data and printing out a few lines.
df = pd.read_csv('rentals.csv')
df.head()
```

Out[1]:

	id	latitude	longitude	property_type	room_type	bathrooms	bedrooms	minimum_nights	price
0	958	37.76931	-122.43386	Apartment	Entire home/apt	1.0	1.0	1	$170.00
1	3850	37.75402	-122.45805	House	Private room	1.0	1.0	1	$99.00
2	5858	37.74511	-122.42102	Apartment	Entire home/apt	1.0	2.0	30	$235.00
3	7918	37.76669	-122.45250	Apartment	Private room	4.0	1.0	32	$65.00
4	8142	37.76487	-122.45183	Apartment	Private room	4.0	1.0	32	$65.00

In [2]:
```
# Getting the dataset information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8111 entries, 0 to 8110
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              8111 non-null   int64  
 1   latitude        8111 non-null   float64
 2   longitude       8111 non-null   float64
 3   property_type   8111 non-null   object 
 4   room_type       8111 non-null   object 
 5   bathrooms       8099 non-null   float64
 6   bedrooms        8107 non-null   float64
 7   minimum_nights  8111 non-null   int64  
 8   price           8111 non-null   object 
dtypes: float64(4), int64(2), object(3)
memory usage: 570.4+ KB
```

In [3]:
```
# Checking for null values and duplicates
df.isnull().sum()
```

Out[3]:
```
id                0
latitude          0
longitude         0
property_type     0
room_type         0
bathrooms        12
bedrooms          4
minimum_nights    0
price             0
dtype: int64
```

In [4]:
```
# Dropping null values
df = df.dropna()
```

In [5]:
```
# Checking
df.isnull().sum()
```

Out[5]:
```
id                0
latitude          0
longitude         0
property_type     0
room_type         0
bathrooms         0
bedrooms          0
minimum_nights    0
price             0
dtype: int64
```

In [6]:
```
# Removing the dollar sign from the price column
df.price = df.price.str.replace(r'\D+', '')
df.price.head() #cheking
```

Out[6]:
```
0    17000
1     9900
2    23500
3     6500
4     6500
Name: price, dtype: object
```

In [7]:
```
# Changing the type of the price into int
df.price = df.price.astype(int)
```

In [8]:
```
# Checking
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8095 entries, 0 to 8110
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              8095 non-null   int64  
 1   latitude        8095 non-null   float64
 2   longitude       8095 non-null   float64
 3   property_type   8095 non-null   object 
 4   room_type       8095 non-null   object 
 5   bathrooms       8095 non-null   float64
 6   bedrooms        8095 non-null   float64
 7   minimum_nights  8095 non-null   int64  
 8   price           8095 non-null   int32  
dtypes: float64(4), int32(1), int64(2), object(2)
memory usage: 600.8+ KB
```

In [9]:
```
# Checking for duplicate rows
df.duplicated().sum()
```

Out[9]:
```
0
```

Data has no duplicates

In [10]:
```
# descriptive statistics for numeric variables
df.describe()
```

Out[10]:

	id	latitude	longitude	bathrooms	bedrooms	minimum_nights	price
count	8.095000e+03	8095.000000	8095.000000	8095.000000	8095.000000	8.095000e+03	8095.000000
mean	2.026698e+07	37.766017	-122.430126	1.395862	1.346387	1.236963e+04	22564.632489
std	1.226930e+07	0.022937	0.026974	0.923114	0.925888	1.111454e+06	41257.579732
min	9.580000e+02	37.704630	-122.513060	0.000000	0.000000	1.000000e+00	0.000000
25%	8.933734e+06	37.751430	-122.442855	1.000000	1.000000	2.000000e+00	10000.000000
50%	2.161924e+07	37.769090	-122.424670	1.000000	1.000000	4.000000e+00	15000.000000
75%	3.120025e+07	37.785600	-122.410625	1.500000	2.000000	3.000000e+01	24000.000000
max	3.935418e+07	37.828790	-122.368570	14.000000	14.000000	1.000000e+08	1000000.000000

In [11]:
```
# Check for bathoroom values
df.bathrooms.value_counts()
```

Out[11]:
```
1.0     5668
2.0     1111
1.5      579
2.5      234
3.0      149
5.0      113
3.5       62
4.0       61
0.0       38
10.0      19
0.5       17
4.5       14
8.0       14
6.0        9
7.0        5
6.5        1
14.0       1
Name: bathrooms, dtype: int64
```

In [12]:
```
# Rounding up values of bathrooms
rounded_values = []
for value in df['bathrooms']:
    value = round(value)
    rounded_values.append(value)

df['bathrooms'] = rounded_values

# Checking
df.bathrooms.value_counts()
```

Out[12]:
```
1     5668
2     1924
3      149
4      137
5      113
0       55
10      19
8       14
6       10
7        5
14       1
Name: bathrooms, dtype: int64
```

In [13]:
```
# From the describe, price column has some outliers for the property_type Boutique hotel
df.query('price >= 1000000')
```

Out[13]:

	id	latitude	longitude	property_type	room_type	bathrooms	bedrooms	minimum_nights	price
7345	36185102	37.78898	-122.41659	Boutique hotel	Private room	1	1.0	1	1000000
7346	36185260	37.79240	-122.42060	Boutique hotel	Private room	1	1.0	1	1000000
7347	36185321	37.79404	-122.42202	Boutique hotel	Private room	1	1.0	1	1000000
7348	36185365	37.79196	-122.42184	Boutique hotel	Private room	1	1.0	1	1000000
7349	36185403	37.79396	-122.42200	Boutique hotel	Private room	1	1.0	1	1000000
7350	36185434	37.79334	-122.42046	Boutique hotel	Private room	1	1.0	1	1000000
7351	36185495	37.79341	-122.42051	Boutique hotel	Private room	1	1.0	1	1000000

In [14]:
```
# Changing the price of the outliers with the mean
Boutique_hotel_mean_price = df.query('property_type == "Boutique hotel"').price.mean()
df.loc[df['price'] == 1000000, 'price'] = Boutique_hotel_mean_price
```

In [15]:
```
# Checking
df.query('price == 1000000')
```

Out[15]:

id	latitude	longitude	property_type	room_type	bathrooms	bedrooms	minimum_nights	price

In [16]:
```
# From the describe, minimum_nights column has outliers
df.query('minimum_nights == 365')
```

Out[16]:

	id	latitude	longitude	property_type	room_type	bathrooms	bedrooms	minimum_nights	price
57	51374	37.76519	-122.45613	Apartment	Entire home/apt	1	2.0	365	999900.0
312	505763	37.75081	-122.44524	Apartment	Entire home/apt	1	1.0	365	20000.0
555	1084068	37.77967	-122.40379	Loft	Entire home/apt	2	1.0	365	18000.0
637	1299242	37.74272	-122.42144	Apartment	Entire home/apt	2	3.0	365	20000.0
1349	4638176	37.76035	-122.39416	Apartment	Entire home/apt	2	3.0	365	29600.0
2018	8818098	37.78818	-122.39181	Apartment	Entire home/apt	2	1.0	365	16000.0
2479	12361066	37.78538	-122.38997	Apartment	Entire home/apt	2	2.0	365	20200.0
4967	25785670	37.78862	-122.38892	Condominium	Entire home/apt	1	0.0	365	12100.0
5550	29107044	37.75388	-122.46552	Apartment	Entire home/apt	1	2.0	365	380000.0

In [17]:
```
# Setting the maxmuim minimum_nights for rental to be a year
df.loc[df['minimum_nights'] > 365, 'minimum_nights'] = 365
```

In [18]:
```
df.describe()
```

Out[18]:

	id	latitude	longitude	bathrooms	bedrooms	minimum_nights	price
count	8.095000e+03	8095.000000	8095.000000	8095.000000	8095.000000	8095.000000	8095.000000
mean	2.026698e+07	37.766017	-122.430126	1.419024	1.346387	16.196788	21739.71865
std	1.226930e+07	0.022937	0.026974	0.931206	0.925888	23.220672	29592.82322
min	9.580000e+02	37.704630	-122.513060	0.000000	0.000000	1.000000	0.00000
25%	8.933734e+06	37.751430	-122.442855	1.000000	1.000000	2.000000	10000.00000
50%	2.161924e+07	37.769090	-122.424670	1.000000	1.000000	4.000000	15000.00000
75%	3.120025e+07	37.785600	-122.410625	2.000000	2.000000	30.000000	24000.00000
max	3.935418e+07	37.828790	-122.368570	14.000000	14.000000	365.000000	999900.00000

In [19]:
```
# Saving the file
df.to_csv('rentals_cleaned_csv')
```