```
TASK 1

1 Read the csv file and load it into a pandas dataframe.
2 Display the first five rows of your dataframe.
3 Display the data types of the columns.
```

In [63]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

In [64]:
```
#Read the csv file
df = pd.read_csv('Airbnb_Open_Data.csv',low_memory=False)
```

In [65]:
```
#Display the first five rows of your dataframe
df.head()
```

Out[65]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	...	service fee	minir ni
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	...	$193	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	...	$28	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...	$124	
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	United States	...	$74	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	...	$41	

5 rows × 26 columns

In [66]:
```
# Display the data types
df.dtypes
```

Out[66]:
```
id                                int64
NAME                             object
host id                           int64
host_identity_verified           object
host name                        object
neighbourhood group              object
neighbourhood                    object
lat                             float64
long                            float64
country                          object
country code                     object
instant_bookable                 object
cancellation_policy              object
room type                        object
Construction year               float64
price                            object
service fee                      object
minimum nights                  float64
number of reviews               float64
last review                      object
reviews per month               float64
review rate number              float64
calculated host listings count  float64
availability 365                float64
house_rules                      object
license                          object
dtype: object
```

```
TASK 2a: Data cleaning
1 Drop some of the unwanted columns. These include host id, id, country and country code from the dataset.
```

2 State the reason for not including these columns for your Data Analytics.

In [67]:
```
df.columns
```

Out[67]:
```
Index(['id', 'NAME', 'host id', 'host_identity_verified', 'host name',
       'neighbourhood group', 'neighbourhood', 'lat', 'long', 'country',
       'country code', 'instant_bookable', 'cancellation_policy', 'room type',
       'Construction year', 'price', 'service fee', 'minimum nights',
       'number of reviews', 'last review', 'reviews per month',
       'review rate number', 'calculated host listings count',
       'availability 365', 'house_rules', 'license'],
      dtype='object')
```

In [68]:
```
df.drop(columns=['id', 'host id','country', 'country code', ], axis=1, inplace=True)
```

In [69]:
```
df.head()
```

1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	moderate	Entire home/apt
2	THE VILLAGE OF HARLEM....NEW YORK !	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	True	flexible	Private room
3	NaN	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	moderate	Entire home/apt
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	moderate	Entire home/apt

Reason for dropping `host id`, `id`, `country` and `country code` columns:
`id` and `host id` are random ids so they don't add any value to the dataset, while 'country` and `country code` are having only categorical wvalue which is United States and US. Also we already know we only have data for USA so we don't need those values.

In [70]:
```
#Display data types of the columns
df.dtypes
```

Out[70]:
```
NAME                             object
host_identity_verified           object
host name                        object
neighbourhood group              object
neighbourhood                    object
lat                             float64
long                            float64
instant_bookable                 object
cancellation_policy              object
room type                        object
Construction year               float64
price                            object
service fee                      object
minimum nights                  float64
number of reviews               float64
last review                      object
reviews per month               float64
review rate number              float64
calculated host listings count  float64
availability 365                float64
house_rules                      object
license                          object
dtype: object
```

TASK 2b - Data Cleaning

In [71]:
```
# Check for missing values in the dataframe and display the count in ascending order.
# missing_data = df.isnuLL()
df.isnull().sum().sort_values()
```

Out[71]:
```
room type                           0
lat                                 8
long                                8
neighbourhood                      16
neighbourhood group                29
cancellation_policy                76
instant_bookable                  105
number of reviews                 183
Construction year                 214
price                             247
NAME                              250
service fee                       273
host_identity_verified            289
calculated host listings count    319
review rate number                326
host name                         406
minimum nights                    409
availability 365                  448
reviews per month               15879
last review                     15893
house_rules                     52131
license                        102597
dtype: int64
```

In [72]:
```
for col in df.columns:
    if df[str(col)].dtype == 'object':
        print(col)
        df[str(col)].fillna(value=df[str(col)].mode()[0], inplace=True)
    else:
        df[str(col)].fillna(value=df[str(col)].median(), inplace=True)
```

```
NAME
host_identity_verified
host name
neighbourhood group
neighbourhood
instant_bookable
cancellation_policy
room type
price
service fee
last review
house_rules
license
```

In [73]:
```
df.isnull().sum().sort_values()
```

Out[73]:
```
NAME                            0
availability 365                0
calculated host listings count  0
review rate number              0
reviews per month               0
last review                     0
number of reviews               0
minimum nights                  0
service fee                     0
price                           0
Construction year               0
room type                       0
cancellation_policy             0
instant_bookable                0
long                            0
lat                             0
neighbourhood                   0
neighbourhood group             0
host name                       0
host_identity_verified          0
house_rules                     0
license                         0
dtype: int64
```

In [74]:
```
#Check whether there are any duplicate values in the dataframe and if present remove them.
df.shape
```

Out[74]:
```
(102599, 22)
```

In [75]:
```
df.duplicated().sum()
```

Out[75]:
```
3461
```

In [76]:
```
#dropping the duplicate values
df.drop_duplicates(inplace=True)
```

In [77]:
```
df.shape
```

Out[77]:
```
(99138, 22)
```

```
#Task 3: Data Transformation
- Rename the column `availability 365` to `days_booked`
- Convert all column names to lowercase and replace the spaces in the column names with an underscore "_".
- Remove the dollar sign and comma from the columns `price` and `service_fee`. If necessary, convert these two columns to the appropriate data type.
```

In [78]:
```
#Rename the column 'availability 365' to 'days_booked'
df.rename(columns={'availability 365':'days_booked'}, inplace=True)
```

In [79]:
```
df.head(2)
```

Out[79]:

	NAME	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	instant_bookable	cancellation_policy	room type	...	servic fe
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	strict	Private room	...	$19
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	moderate	Entire home/apt	...	$2

2 rows × 22 columns

In [80]:
```
## Convert all column names to Lowercase and replace the spaces with an underscore

df.columns = [col.lower().replace(" ","_") for col in df.columns]
df.columns
```

Out[80]:
```
Index(['name', 'host_identity_verified', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'lat', 'long', 'instant_bookable',
       'cancellation_policy', 'room_type', 'construction_year', 'price',
       'service_fee', 'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'review_rate_number',
       'calculated_host_listings_count', 'days_booked', 'house_rules',
       'license'],
      dtype='object')
```

In [17]:
```
#Remove the dollar sign and comma from the columns `price` and `service_fee`.
#If necessary, convert these two columns to the appropriate data type.
```

In [81]:
```
df[['price','service_fee']].head()
```

Out[81]:

	price	service_fee
0	$966	$193
1	$142	$28
2	$620	$124
3	$368	$74
4	$204	$41

In [82]:
```
## Remove the dollar sign and comma from the columns. If necessary, convert these two columns to the appropriate data type.

def remove_dollar_comma_sign(value):
    if pd.isna(value):
        return np.NaN
    else:
        return value.replace('$','').replace(",","")
```

In [83]:
```
df['price'] = df['price'].apply(lambda x: remove_dollar_comma_sign(x))
```

In [84]:
```
df['service_fee'] = df['service_fee'].apply(lambda x: remove_dollar_comma_sign(x))
```

In [85]:
```
df[['price', 'service_fee']].head()
```

Out[85]:

	price	service_fee
0	966	193
1	142	28
2	620	124
3	368	74
4	204	41

In [104]:
```
df['price'] = df['price'].astype(int)
df['service_fee'] = df['service_fee'].astype(int)
```

TASK 4: Exploratory Data Analysis

- List the count of various room types avaliable in the dataset.
- Which room type has the most strict cancellation policy?
- List the average price per neighborhood group, and highlight the most expensive neighborhood to rent from.

In [86]:
```
df['room_type'].unique()
```

Out[86]:
```
array(['Private room', 'Entire home/apt', 'Shared room', 'Hotel room'],
      dtype=object)
```

In [87]:
```
## List the count of various room types avaliable with Airbnb
df['room_type'].value_counts()
```

Out[87]:
```
room_type
Entire home/apt    51987
Private room       44887
Shared room         2149
Hotel room           115
Name: count, dtype: int64
```

In [88]:
```
df['cancellation_policy'].unique()
```

Out[88]:
```
array(['strict', 'moderate', 'flexible'], dtype=object)
```

In [89]:
```
## Which room type adheres to more strict cancellation policy?
df_group_prep = df[df['cancellation_policy']=='strict']
```

In [90]:
```
df_group_prep.shape
```

Out[90]:
```
(32926, 22)
```

In [92]:
```
df_group_prep.head(2)
```

Out[92]:

	name	host_identity_verified	host_name	neighbourhood_group	neighbourhood	lat	long	instant_bookable	cancellation_policy	room_typ
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	strict	Privat roor
8	Large Furnished Room Near B'way	verified	Evelyn	Manhattan	Hell's Kitchen	40.76489	-73.98493	True	strict	Privat roor

2 rows × 22 columns

In [93]:
```
df_group_prep['room_type'].value_counts()
```

Out[93]:
```
room_type
Entire home/apt    17238
Private room       14936
Shared room          718
Hotel room            34
Name: count, dtype: int64
```

```
Entire home/apt has the highest strict policy
```

In [106]:
```
## List the average price per neighborhood group, and highlight the most expensive neighborhood to rent from.
grp_avg = df['price'].groupby(df['neighbourhood_group']).mean().sort_values(ascending=False).reset_index()
grp_avg
```

Out[106]:

	neighbourhood_group	price
0	Queens	628.668822
1	Brooklyn	625.471627
2	Bronx	625.271511
3	Staten Island	625.060870
4	Manhattan	621.666140
5	brookln	580.000000
6	manhatan	460.000000

In [107]:
```
grp_avg = df['price'].groupby(df['neighbourhood_group']).min().sort_values(ascending=True).reset_index()
grp_avg
```

Out[107]:

	neighbourhood_group	price
0	Bronx	50
1	Brooklyn	50
2	Manhattan	50
3	Queens	50
4	Staten Island	50
5	manhatan	460
6	brookln	580

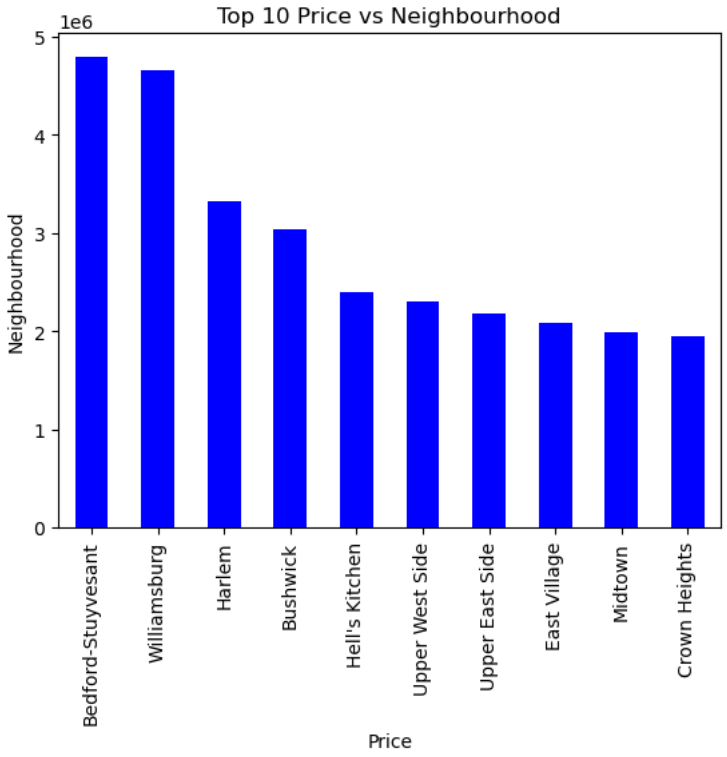
```
#Task 5a: Data Visualization
* Create a horizontal bar chart to display the top 10 most expensive neighborhoods in the dataset
* List the neighborhoods which offer short term rentals within 10 days. Illustrate with a bar graph
* List the prices with respect to room type using a bar graph and also state your inferences.
* Create a pie chart that shows distribution of booked days for each neighborhood group
```

In [112]:
```
grp2 = df['price'].groupby(df['neighbourhood']).sum().sort_values(ascending=False)
grp2.head(10)
```

Out[112]:
```
neighbourhood
Bedford-Stuyvesant    4793673
Williamsburg          4663153
Harlem                3317743
Bushwick              3038762
Hell's Kitchen        2394881
Upper West Side       2306230
Upper East Side       2177795
East Village          2081467
Midtown               1985830
Crown Heights         1943244
Name: price, dtype: int32
```

In [116]:
```
# 1 Create a horizontal bar chart to display the top 10 most expensive neighborhoods in the dataset
grp2.head(10).plot(kind='bar',color={'blue'})
plt.xlabel('Price')
plt.ylabel('Neighbourhood')
plt.title('Top 10 Price vs Neighbourhood')
plt.show
```

Out[116]: <function matplotlib.pyplot.show(close=None, block=None)>



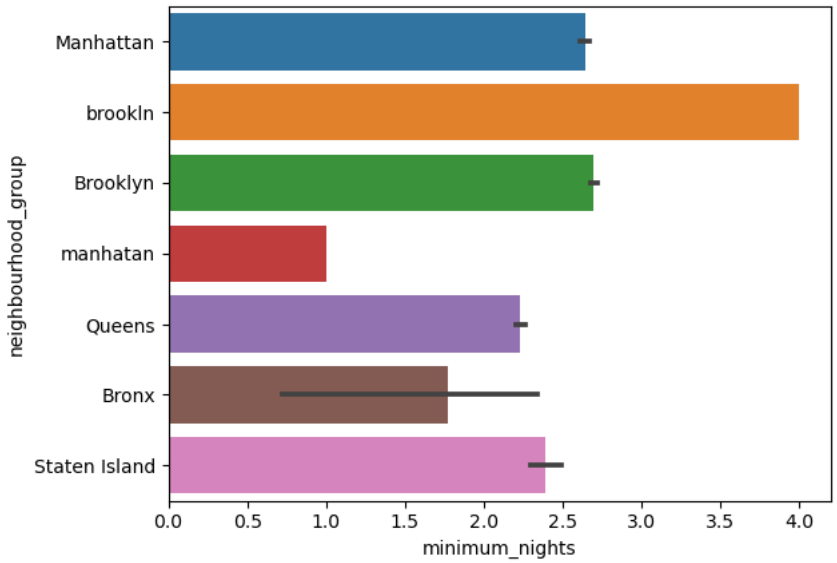
In [117]:
```
grp2.head(10).plot(kind='barh',color={'blue'})
plt.xlabel('Price')
plt.ylabel('Neighbourhood')
plt.title('Top 10 Price vs Neighbourhood')
plt.show
```

Out[117]: <function matplotlib.pyplot.show(close=None, block=None)>

In [123]:
```
# List the neighborhoods which offer short term rentals within 10 days. Illustrate with a bar graph
df_filter_min_nights = df[df['minimum_nights']<10]
df_filter_min_nights['neighbourhood_group'].value_counts()

sns.barplot(x='minimum_nights',
            y='neighbourhood_group',
            data=df_filter_min_nights, orient='h')
```

Out[123]: <Axes: xlabel='minimum_nights', ylabel='neighbourhood_group'>



In [128]:
```
# List the prices with respect to room type using a bar graph and also state your inferences.
df1 = df.groupby(['room_type']).agg(mean_price=('price','mean'))
```

In [129]:
```
df1 = df1.reset_index()
```

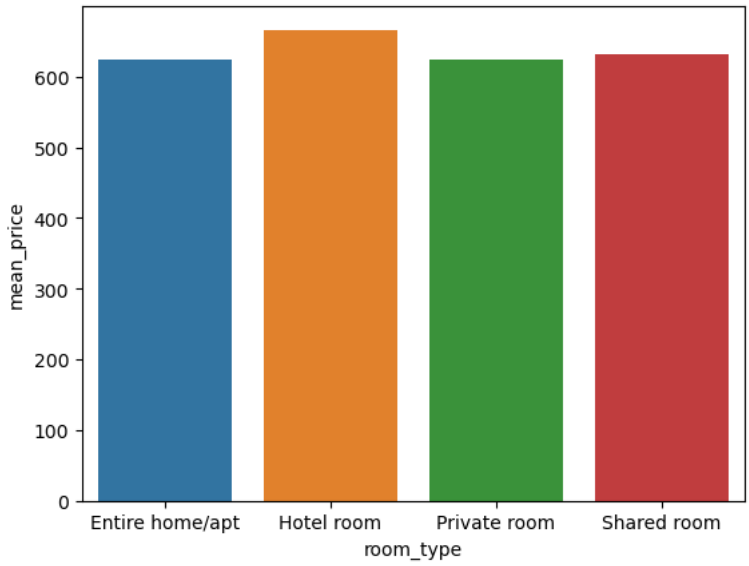
In [132]:
```
df1.head()
```

Out[132]:

	room_type	mean_price
0	Entire home/apt	624.227711
1	Hotel room	666.391304
2	Private room	623.842516
3	Shared room	630.912517

In [134]:
```
sns.barplot(x='room_type',
            y='mean_price',
            data=df1)
```

Out[134]: <Axes: xlabel='room_type', ylabel='mean_price'>



In [ ]:
```
#conclusion: Hotel room are more expensive than Airbnb room, and also to entire home/apt
```

In [135]:
```
##Create a pie chart that shows distribution of booked days for each neighborhood group

grp3=df['days_booked'].groupby(df['neighbourhood_group']).mean().sort_values().reset_index()
grp3
```

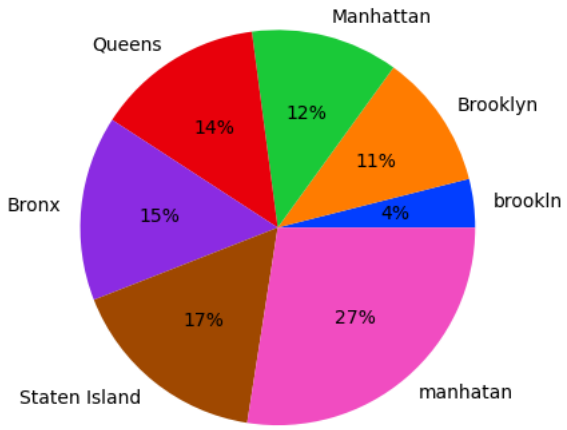
Out[135]:

	neighbourhood_group	days_booked
0	brookln	47.000000
1	Brooklyn	130.765437
2	Manhattan	142.697820
3	Queens	162.424977
4	Bronx	178.993117
5	Staten Island	195.989130
6	manhatan	325.000000

In [141]:
```
#define Seaborn color palette to use
palette_color = sns.color_palette('bright')

# plotting data on chart
plt.pie(grp3['days_booked'], labels=grp3['neighbourhood_group'], colors=palette_color, autopct='%.0f%%')
```

Out[141]:
```
([<matplotlib.patches.Wedge at 0x1d3fa34bdd0>,
  <matplotlib.patches.Wedge at 0x1d3fa35cb90>,
  <matplotlib.patches.Wedge at 0x1d3fa35e290>,
  <matplotlib.patches.Wedge at 0x1d3fa35db50>,
  <matplotlib.patches.Wedge at 0x1d3fa368e50>,
  <matplotlib.patches.Wedge at 0x1d3fa36a5d0>,
  <matplotlib.patches.Wedge at 0x1d3fa36bb90>],
 [Text(1.0914410636123797, 0.13695402389370517, 'brookln'),
  Text(0.9097556964016024, 0.6183401756839318, 'Brooklyn'),
  Text(0.2695299631376125, 1.066467814315574, 'Manhattan'),
  Text(-0.5869396038310039, 0.9303235466517573, 'Queens'),
  Text(-1.0943728547937364, 0.1111218011490439, 'Bronx'),
  Text(-0.6883065813100627, -0.8580408207802552, 'Staten Island'),
  Text(0.7150354846918396, -0.8358972757650943, 'manhatan')],
 [Text(0.5953314892431161, 0.0747021948511119, '4%'),
  Text(0.4962303798554194, 0.3372764594639628, '11%'),
  Text(0.1470163435296068, 0.5817097168994039, '12%'),
  Text(-0.3201488748169111, 0.5074492072645949, '14%'),
  Text(-0.5969306480693107, 0.060611891535842115, '15%'),
  Text(-0.3754399534418523, -0.4680222658801392, '17%'),
  Text(0.3900193552864579, -0.4559439685991423, '27%')])
```



In [142]:
```
#TASK 5b
#Does service price and room price have an impact on each other. Illustrate this relationship with a scatter plot and state y
```

In [157]:
```
plt.figure(figsize=(15,10))
plt.title('Relationship bewtween price and service fee', size=25, color='red')
sns.scatterplot(x=df['price'], y=df['service_fee'], hue=df['room_type']), size==30
```

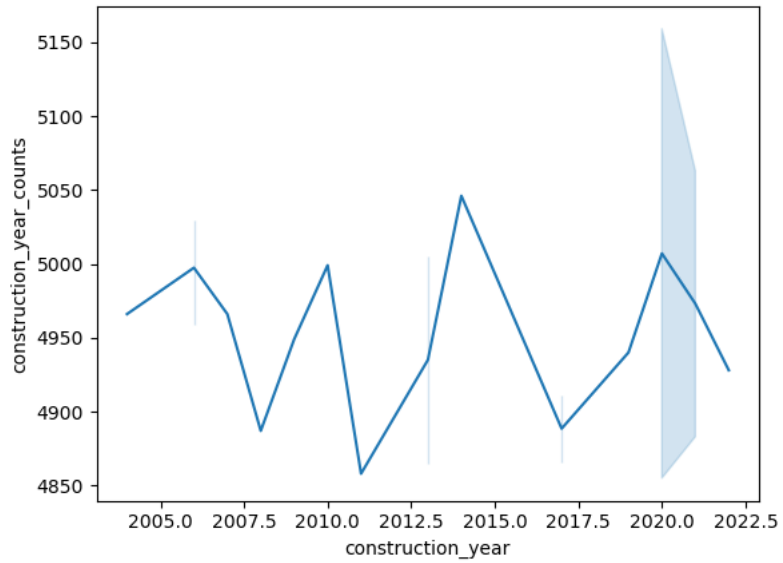
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[157], line 3
      1 plt.figure(figsize=(15,10))
      2 plt.title('Relationship bewtween price and service fee', size=25, color='red')
----> 3 sns.scatterplot(x=df['price'], y=df['service_fee'], hue=df['room_type']), size==30

NameError: name 'size' is not defined
```

In [158]:
```
plt.figure(figsize=(10,5))
df['construction_year_counts']=df['construction_year'].value_counts()
```

```
<Figure size 1000x500 with 0 Axes>
```

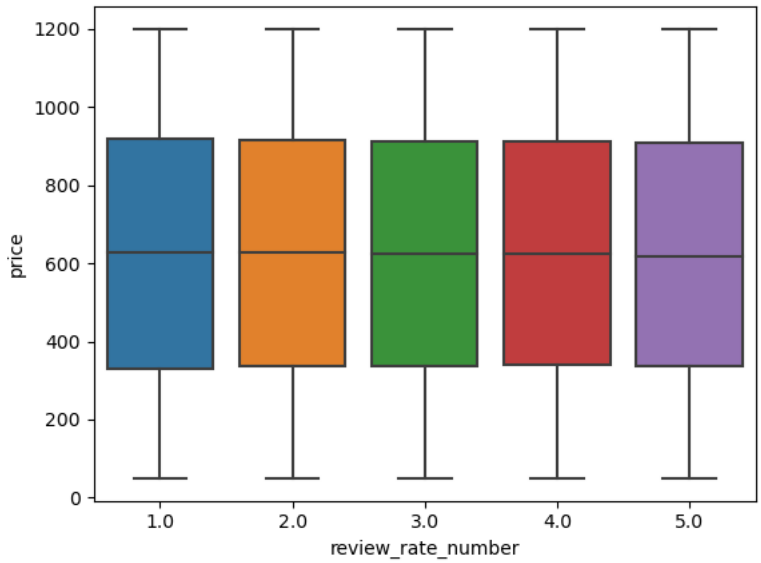
In [160]:
```
# Using a line graph show in which year the maximum construction of rooms took place.
sns.lineplot(x='construction_year', y='construction_year_counts', data=df)
plt.show()
```



```
TASK 5c: Data Visualization
* With the help of box plots illustrate the following
1 Effect of Review Rate number on price
2 Effect of host identity verified on price
```

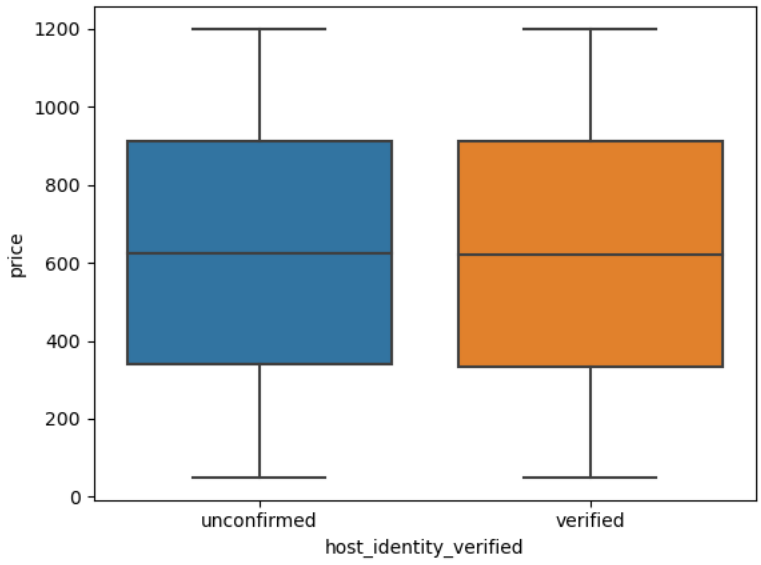
In [161]:
```
sns.boxplot(x='review_rate_number', y='price', data=df)
```

Out[161]:
```
<Axes: xlabel='review_rate_number', ylabel='price'>
```



In [163]:
```
sns.boxplot(x='host_identity_verified', y='price', data=df)
```

Out[163]:
```
<Axes: xlabel='host_identity_verified', ylabel='price'>
```



In [ ]: