AirBNB Data Analytics Project 1: Load Data and Basic Statistics

In this notebook we will download airbnb data and You will:

- Use python functions to compute important summary statistics
- Derive insights from the dataset to extract important features
- Report basic statistics like mean, standard deviation, range, max, min, percentiles for features.
- use histogram, pie chart etc. to visualize the data

In this notebook you will be provided with some already complete code as well as some code that you should complete yourself to finish Project 1. The code we provide to complte is optional and is there to assist you with solving the problems but feel free to ignore the helper code and write your own.

Download Data

There are many ways to download data from Airbnb.com. Our group has tried to download the data using web scrapping tool directly from the website (this works for all areas that airbnb has services) and also tried to download data from insideairbnb.com (limited region only). Please feel free to explore new ways that are unknown to us and focus on the area for your project only for the analysis.

If you are assigned analysis work for our clients in one of the regions that is supported by insideairbnb.com, it is suggested to use data set there since the data is well structured and easier to implement.

There are four main data tables:

- listings - Detailed listings data showing ~100 attributes for each of the listings. Some of the attributes used in the analysis are price(continuous), longitude (continuous), latitude (continuous), listing_type (categorical), is_superhost (categorical), neighbourhood (categorical), ratings (continuous) among others.
- reviews - Detailed reviews given by the guests. Key attributes include date (datetime), listing_id (discrete), reviewer_id (discrete) and comment (textual).
- calendar - Provides details about booking for the next year by listing. Four attributes in total including listing_id (discrete), date(datetime), available (categorical) and price (continuous).
- neighborhood information - Provide basic neighborhood information.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Reading in listings data

```
listings = pd.read_csv('listings.csv.gz')
listings.head()
```

```
         id                              listing_url       scrape_id  \
0  35066424  https://www.airbnb.com/rooms/35066424  20220912200136
1  34843927  https://www.airbnb.com/rooms/34843927  20220912200136
2  35067513  https://www.airbnb.com/rooms/35067513  20220912200136
3  35069027  https://www.airbnb.com/rooms/35069027  20220912200136
4  34857616  https://www.airbnb.com/rooms/34857616  20220912200136

  last_scraped       source
name  \
0   2022-09-13  city scrape     Spacious 3 Bedroom 3 Bath + Loft at
Alii Cove
1   2022-09-12  city scrape                          Simply Paradise
Glamping
2   2022-09-13  city scrape  Noah’s Hideaway Maui, Luxury B&B, Walk to
Beach!
3   2022-09-13  city scrape                                Private
Beach Suite
4   2022-09-13  city scrape     Point at Poipu Resort- 2 bedrooms
garden view

                                        description  \
0  Largest 3 Bedroom 3 bath at Alii Cove across f...
1  Won·der:  a feeling of surprise mingled with a...
2  PRIVATE AND SECLUDED: Noah's Hideaway, a Luxur...
3  Best location in all NS<br />Quiet, safe, frie...
4  Drenched in radiant island sunshine, The Point...

                               neighborhood_overview  \
0  It's summer time all year long! Beautiful Kona...
1  Located in historic Holualoa and just a short ...
2  We live in a tropical oasis, our own little pa...
3                                                NaN
4                                                NaN

                                        picture_url
host_id  ...  \
0  https://a0.muscache.com/pictures/0d072215-a1d1...  264152810  ...

1  https://a0.muscache.com/pictures/miso/Hosting-...  262664392  ...

2  https://a0.muscache.com/pictures/miso/Hosting-...  264162605  ...

3  https://a0.muscache.com/pictures/cd562c88-2744...   38304048  ...

4  https://a0.muscache.com/pictures/d4125003-41be...   31214940  ...
```

```
  review_scores_communication review_scores_location
review_scores_value  \
0                         5.00                    5.00
4.50
1                         4.61                    4.66
4.49
2                         5.00                    4.99
4.93
3                         4.88                    5.00
4.88
4                         5.00                    5.00
5.00

                            license instant_bookable  \
0                               NaN                 f
1                               NaN                 t
2  440090330000, TA-197-216-9216-01                 t
3               GE-095- 995-512-01R                 f
4  280210010000, TA-148-664-9856-01                 f

  calculated_host_listings_count
calculated_host_listings_count_entire_homes  \
0                               1
1
1                               3
0
2                               2
2
3                               1
0
4                              31
10

  calculated_host_listings_count_private_rooms  \
0                                            0
1                                            3
2                                            0
3                                            1
4                                           21

  calculated_host_listings_count_shared_rooms reviews_per_month
0                                           0              0.25
1                                           0              5.90
2                                           0              1.80
3                                           0              0.24
4                                           0              0.05

[5 rows x 75 columns]
```

Reading the 'calender' dataset

```
calendar = pd.read_csv('calendar.csv.gz')
calendar.head()
```

```
   listing_id        date available    price adjusted_price
minimum_nights  \
0    35066424  2022-09-13         f  $175.00        $175.00
30.0
1    35066424  2022-09-14         f  $175.00        $175.00
30.0
2    35066424  2022-09-15         f  $175.00        $175.00
30.0
3    35066424  2022-09-16         f  $175.00        $175.00
30.0
4    35066424  2022-09-17         f  $175.00        $175.00
30.0

   maximum_nights
0           365.0
1           365.0
2           365.0
3           365.0
4           365.0
```

Reading the 'reviews' dataset

```
reviews = pd.read_csv('reviews.csv.gz')
reviews.head()
```

```
   listing_id         id        date  reviewer_id reviewer_name  \
0    34843927  585561271  2019-12-31     56285682          Eric
1    34843927  589316569  2020-01-05    149607043       Vanessa
2    34843927  589852238  2020-01-06    160618306          Ryan
3    34843927  590205146  2020-01-07    250306395        Mollie
4    34843927  590886527  2020-01-09    310087862       William

                                            comments
0  I truly felt the “Aloha Hospitality” here!  Ju...
1  Wow vraiment beau et propre! Nous n'avons que ...
2  Thanks for the accommodations was a safe comfo...
3  Beautiful star gazing. Cool tent. A little har...
4  We really enjoyed our stay. The balcony with t...
```

```
reviews.shape
```

```
(926310, 6)
```

Reading the 'neighbourhoods' dataset

```
neighbourhoods = pd.read_csv('neighbourhoods.csv')
neighbourhoods.head()
```

```
  neighbourhood_group neighbourhood
0              Hawaii       Hamakua
1              Hawaii           Kau
2              Hawaii    North Hilo
3              Hawaii  North Kohala
4              Hawaii    North Kona
```

show the first 5 rows of the data in the listing dataset.

```
listings.head()
```

```
         id                              listing_url       scrape_id  \
0  35066424  https://www.airbnb.com/rooms/35066424  20220912200136
1  34843927  https://www.airbnb.com/rooms/34843927  20220912200136
2  35067513  https://www.airbnb.com/rooms/35067513  20220912200136
3  35069027  https://www.airbnb.com/rooms/35069027  20220912200136
4  34857616  https://www.airbnb.com/rooms/34857616  20220912200136

  last_scraped       source
name  \
0   2022-09-13  city scrape     Spacious 3 Bedroom 3 Bath + Loft at
Alii Cove
1   2022-09-12  city scrape                          Simply Paradise
Glamping
2   2022-09-13  city scrape  Noah’s Hideaway Maui, Luxury B&B, Walk to
Beach!
3   2022-09-13  city scrape                                Private
Beach Suite
4   2022-09-13  city scrape     Point at Poipu Resort- 2 bedrooms
garden view

                                        description  \
0  Largest 3 Bedroom 3 bath at Alii Cove across f...
1  Won·der:  a feeling of surprise mingled with a...
2  PRIVATE AND SECLUDED: Noah's Hideaway, a Luxur...
3  Best location in all NS<br />Quiet, safe, frie...
4  Drenched in radiant island sunshine, The Point...

                             neighborhood_overview  \
0  It's summer time all year long! Beautiful Kona...
1  Located in historic Holualoa and just a short ...
2  We live in a tropical oasis, our own little pa...
3                                              NaN
4                                              NaN

                                        picture_url
host_id  ...  \
0  https://a0.muscache.com/pictures/0d072215-a1d1...  264152810  ...

1  https://a0.muscache.com/pictures/miso/Hosting-...  262664392  ...
```

```
2  https://a0.muscache.com/pictures/miso/Hosting-...  264162605  ...

3  https://a0.muscache.com/pictures/cd562c88-2744...   38304048  ...

4  https://a0.muscache.com/pictures/d4125003-41be...   31214940  ...


  review_scores_communication review_scores_location
review_scores_value  \
0                        5.00                   5.00
4.50
1                        4.61                   4.66
4.49
2                        5.00                   4.99
4.93
3                        4.88                   5.00
4.88
4                        5.00                   5.00
5.00

                           license instant_bookable  \
0                              NaN                f
1                              NaN                t
2  440090330000, TA-197-216-9216-01                t
3               GE-095- 995-512-01R                f
4  280210010000, TA-148-664-9856-01                f

  calculated_host_listings_count
calculated_host_listings_count_entire_homes  \
0                               1
1
1                               3
0
2                               2
2
3                               1
0
4                              31
10

  calculated_host_listings_count_private_rooms  \
0                                            0
1                                            3
2                                            0
3                                            1
4                                           21

  calculated_host_listings_count_shared_rooms reviews_per_month
```

```
0                                                   0               0.25
1                                                   0               5.90
2                                                   0               1.80
3                                                   0               0.24
4                                                   0               0.05

[5 rows x 75 columns]
```

```
#Shape of the dataset
listings.shape
```

```
(28580, 75)
```

print all the columns in the listings data to get familiar with the data

```
for col in listings.columns:
    print(col)
```

```
id
listing_url
scrape_id
last_scraped
source
name
description
neighborhood_overview
picture_url
host_id
host_url
host_name
host_since
host_location
host_about
host_response_time
host_response_rate
host_acceptance_rate
host_is_superhost
host_thumbnail_url
host_picture_url
host_neighbourhood
host_listings_count
host_total_listings_count
host_verifications
host_has_profile_pic
host_identity_verified
neighbourhood
neighbourhood_cleansed
neighbourhood_group_cleansed
latitude
longitude
property_type
```

```
room_type
accommodates
bathrooms
bathrooms_text
bedrooms
beds
amenities
price
minimum_nights
maximum_nights
minimum_minimum_nights
maximum_minimum_nights
minimum_maximum_nights
maximum_maximum_nights
minimum_nights_avg_ntm
maximum_nights_avg_ntm
calendar_updated
has_availability
availability_30
availability_60
availability_90
availability_365
calendar_last_scraped
number_of_reviews
number_of_reviews_ltm
number_of_reviews_l30d
first_review
last_review
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
license
instant_bookable
calculated_host_listings_count
calculated_host_listings_count_entire_homes
calculated_host_listings_count_private_rooms
calculated_host_listings_count_shared_rooms
reviews_per_month
```

Understand the data and start doing analysis:

To analyze the data to generate insights, we have to pre-process and aggregate the data to generate summary table for listings for our analytical purposes. Here are a couple of ideas if you are not already familiar with residential real estate:

- For any real estate, the most important things are location, location, location! look at city, zipcode, neighbourhood_cleansed and other related columns to get an idea about the location.
- room_type and property_type are both important since it gives information about the property and how it rents on AirBNB. For our client's purposes, you can focus on Entire home/apt for room_type and in most of the cases, you can research House only.
- accommodates, bathrooms, beds, bed_type are important columns about sleeping capacity.
- square_feet is important columns related to hosting price.
- price, security_deposit, cleaning_fee are sensitive information for guests.
- minimum)nights and maximum_nights are important information about the hosting.
- other columns like review_scores_rating, cancellation_policy, reviews_per_month, number_of_reviews, number_of_reviews_itm etc are also important factors for guests to decide which property to stat at.

```
# your code and analysis here#
#selecting the above mentioned columns from the dataset.
listings[['neighbourhood', 'neighbourhood_cleansed','latitude',
'longitude',
'property_type', 'room_type', 'accommodates', 'bathrooms', 'bedrooms',
'beds',
'price', 'minimum_nights', 'maximum_nights', 'number_of_reviews',
'number_of_reviews_ltm',
'number_of_reviews_l30d', 'review_scores_rating',
'review_scores_accuracy', 'review_scores_cleanliness',
'review_scores_checkin', 'review_scores_communication',
'review_scores_location', 'review_scores_value'
]]
```

```
                         neighbourhood neighbourhood_cleansed
latitude  \
0        Kailua, Hawaii, United States            North Kona
19.62768
1      Holualoa, Hawaii, United States            North Kona
19.66220
2       Lahaina, Hawaii, United States               Lahaina
20.91764
3                                   NaN       North Shore Oahu
21.58206
4                                   NaN           Koloa-Poipu
21.90589
...                                 ...                   ...       ..
.
28575     Puako, Hawaii, United States          South Kohala
19.96992
28576                               NaN                   Ewa
21.32377
28577   Kamuela, Hawaii, United States          South Kohala
```

```
19.96680
28578                              NaN            Koloa-Poipu
21.87031
28579                              NaN      North Shore Kauai
22.22897

       longitude             property_type        room_type
accommodates  \
0     -155.98543          Entire townhouse  Entire home/apt
6
1     -155.95681      Private room in yurt     Private room
3
2     -156.68840        Entire guest suite  Entire home/apt
2
3     -158.14168      Private room in home     Private room
1
4     -159.46809    Private room in resort     Private room
6
...          ...                       ...              ...
...
28575 -155.84321  Private room in bungalow     Private room
2
28576 -157.97581      Private room in home     Private room
2
28577 -155.84840               Entire home  Entire home/apt
8
28578 -159.44706             Entire condo  Entire home/apt
8
28579 -159.47531             Entire condo  Entire home/apt
5

       bathrooms  bedrooms  beds  ... number_of_reviews  \
0            NaN       3.0   4.0  ...                  2
1            NaN       1.0   2.0  ...                194
2            NaN       1.0   1.0  ...                 70
3            NaN       1.0   1.0  ...                  8
4            NaN       2.0   3.0  ...                  1
...          ...       ...   ...  ...                ...
28575        NaN       1.0   1.0  ...                 32
28576        NaN       1.0   NaN  ...                  0
28577        NaN       4.0   4.0  ...                  8
28578        NaN       4.0   6.0  ...                  0
28579        NaN       2.0   3.0  ...                  0

       number_of_reviews_ltm  number_of_reviews_l30d
review_scores_rating  \
0                          2                       0
5.00
1                         96                       3
4.41
```

```
2                           5                           0
5.00
3                           0                           0
4.88
4                           0                           0
5.00
...                       ...                       ...
...
28575                      10                           0
4.72
28576                       0                           0
NaN
28577                       3                           0
5.00
28578                       0                           0
NaN
28579                       0                           0
NaN

       review_scores_accuracy  review_scores_cleanliness  \
0                        5.00                       5.00
1                        4.46                       4.40
2                        4.97                       4.99
3                        4.75                       4.88
4                        5.00                       5.00
...                       ...                        ...
28575                    4.78                       4.53
28576                     NaN                        NaN
28577                    5.00                       5.00
28578                     NaN                        NaN
28579                     NaN                        NaN

       review_scores_checkin  review_scores_communication  \
0                       5.00                        5.00
1                       4.65                        4.61
2                       4.99                        5.00
3                       4.88                        4.88
4                       5.00                        5.00
...                      ...                         ...
28575                   4.78                        4.78
28576                    NaN                         NaN
28577                   5.00                        5.00
28578                    NaN                         NaN
28579                    NaN                         NaN

       review_scores_location  review_scores_value
0                        5.00                 4.50
1                        4.66                 4.49
2                        4.99                 4.93
3                        5.00                 4.88
```

```
4                        5.00                   5.00
...                       ...                    ...
28575                    4.91                   4.72
28576                     NaN                    NaN
28577                    5.00                   4.38
28578                     NaN                    NaN
28579                     NaN                    NaN

[28580 rows x 23 columns]
```

One example of the analysis can be: what types of listings are available in the area?

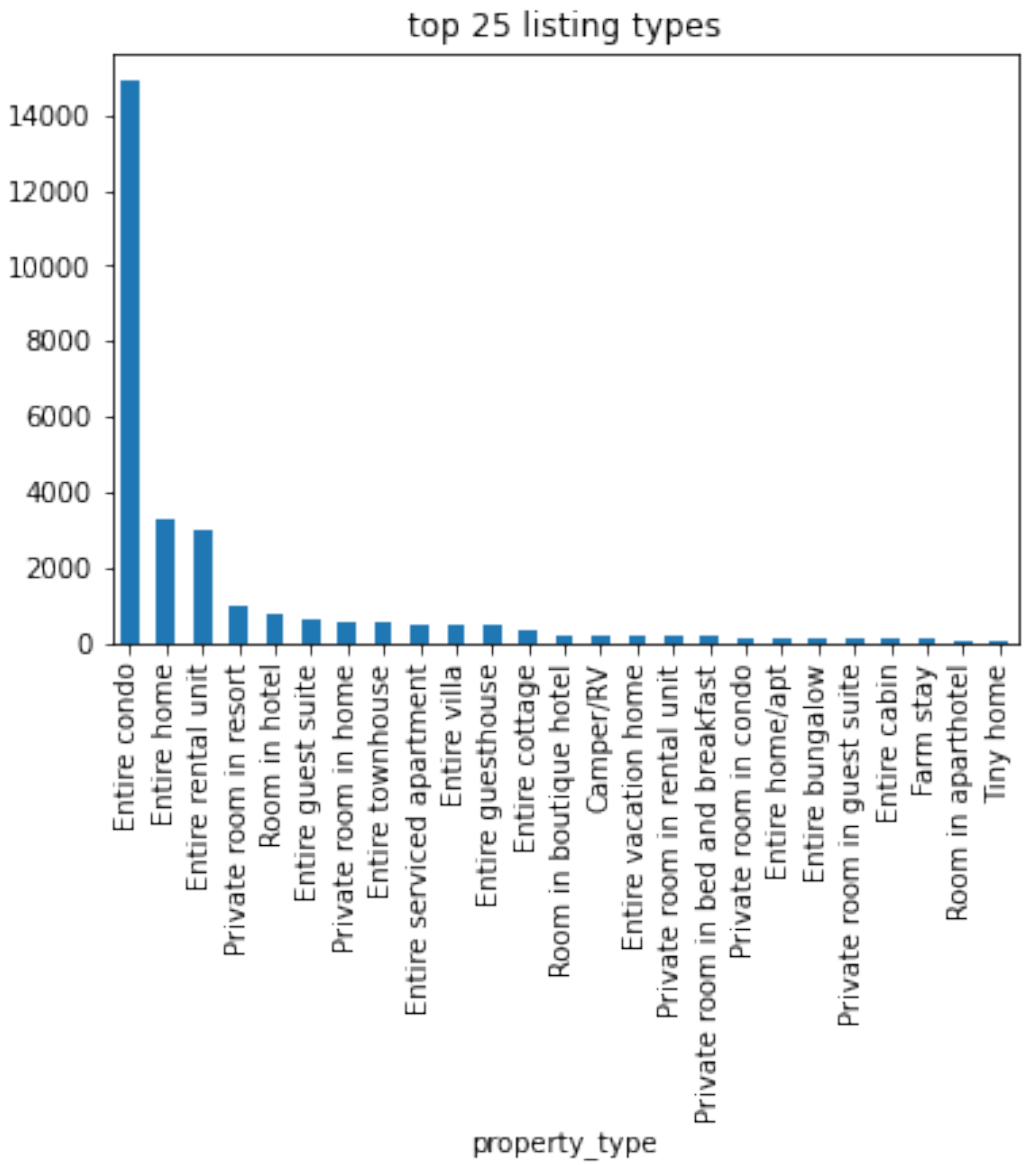
```
# your code and analysis here#
type_of_listing_df = listings.groupby(listings['property_type'])
['id'].count()
type_of_listing_df.sort_values(ascending = False).reset_index()
```

```
             property_type     id
0              Entire condo  14889
1               Entire home   3288
2       Entire rental unit   3011
3   Private room in resort    964
4             Room in hotel    734
..                      ...    ...
81                  Ranch      1
82                 Island      1
83   Private room in minsu      1
84  Private room in chalet      1
85                   Barn      1

[86 rows x 2 columns]
```

There are 86 listing types in this dataset. Among them the higest 25 types are graphed below.

```
type_of_listing_df.sort_values(ascending =
False).head(25).plot(kind='bar')
plt.title("top 25 listing types")
plt.show()
```



Some ideas on the analysis

- x unique listing in the area in total.
- The first rental in the area was up in x date in x neighborhood.
- x reviews have been written by guests since then.
- The price for a listing ranges from x per night to x per night.
- Listing with >$x price tag are in xx, xx, xx neighborhood
- highest average listing price is in xx neighborhood and lowest average listing price is in xx neighborhood.

Analyze time series trending in supply and demand

To get the supply in the area, column host_since provides a good estimation on how many new hosts are added each year and month. You may want to get the year and month of this column and plot unique hosts vs. date (year and month).

```
# your code and analysis here#

#Number of unique host IDs
listings['host_id'].nunique()
```

```
8110
```

```
listings['host_since'].dtype
```

```
dtype('O')
```

```
# get the year and month of host_since variable
listings['host_year'] = listings['host_since'].str.split('-').str[0]
listings['host_month'] = listings['host_since'].str.split('-').str[1]
listings[['host_year', 'host_month']].head()
```

```
  host_year host_month
0      2019         05
1      2019         05
2      2019         05
3      2015         07
4      2015         04
```

```
#Combining Year and Month
listings['host_year_and_month'] = listings.host_year + "-" +
listings.host_month

#Unique number of host ids for each month of the year from 2008 to
2022
time_series =
listings.groupby(['host_year_and_month']).host_id.nunique().reset_inde
x()
time_series.head()
```

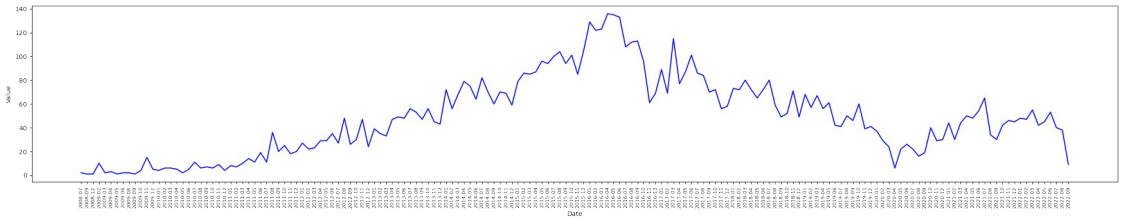
```
  host_year_and_month  host_id
0             2008-07        2
1             2008-09        1
2             2008-12        1
3             2009-02       10
4             2009-03        2
```

```
import matplotlib.pyplot as plt

# Draw Plot
def plot_df(df, x, y, title="", xlabel='Date', ylabel='Value',
dpi=100):
```

```
    plt.figure(figsize=(30,5), dpi=dpi)
    plt.plot(x, y, color='blue')
    plt.xticks(fontsize=8,rotation = 90)
    plt.gca().set(title=title, xlabel=xlabel, ylabel=ylabel)
    plt.show()

# Line Chart of monthly number of unique host IDs
plot_df(time_series, x=time_series.host_year_and_month,
y=time_series.host_id)
```



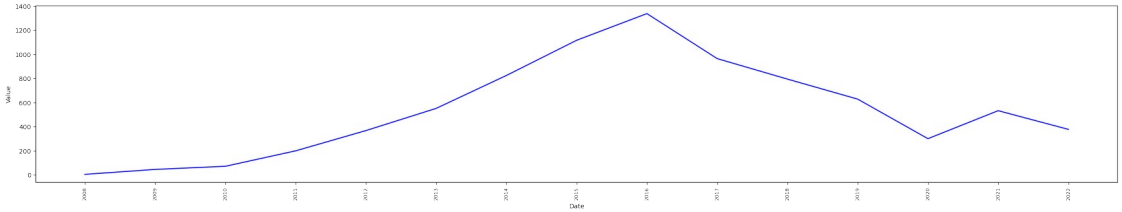
```
#Unique number of host ids for each month of the year from 2008 to
2022
time_series =
listings.groupby(['host_year_and_month']).host_id.nunique().reset_inde
x()
time_series.head()
```

	host_year_and_month	host_id
0	2008-07	2
1	2008-09	1
2	2008-12	1
3	2009-02	10
4	2009-03	2

```
time_series_year =
listings.groupby(['host_year']).host_id.nunique().reset_index()
time_series_year.head()
```

	host_year	host_id
0	2008	4
1	2009	45
2	2010	71
3	2011	199
4	2012	367

```
#Yearly View of unique host IDs
plot_df(time_series_year, x=time_series_year.host_year,
y=time_series_year.host_id)
```



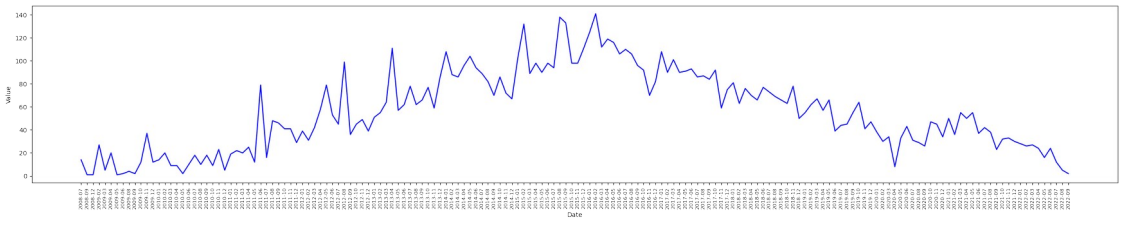
To get the demand in the area, notice that the data on the number of booking made on Airbnb over the years is not available and is probably confidential information. Instead, use 'number of reviews' as a proxy for the demand for Airbnb rentals. As per the company, about 50% of the guests review the hosts/listings, hence studying the number of reviews will give us a good estimation of the demand. Plot unique reviews vs data (year and month).

```
#get the year and month for number of unique reviews
time_series_reviews =
listings.groupby(['host_year_and_month']).number_of_reviews.nunique().
reset_index()
time_series_reviews
```

```
    host_year_and_month  number_of_reviews
0               2008-07                 14
1               2008-09                  1
2               2008-12                  1
3               2009-02                 27
4               2009-03                  5
..                  ...                ...
161             2022-05                 16
162             2022-06                 24
163             2022-07                 12
164             2022-08                  5
165             2022-09                  2

[166 rows x 2 columns]
```

```
#Monthly view for the number of unique reviews
plot_df(time_series_reviews,
x=time_series_reviews.host_year_and_month,
y=time_series_reviews.number_of_reviews)
```

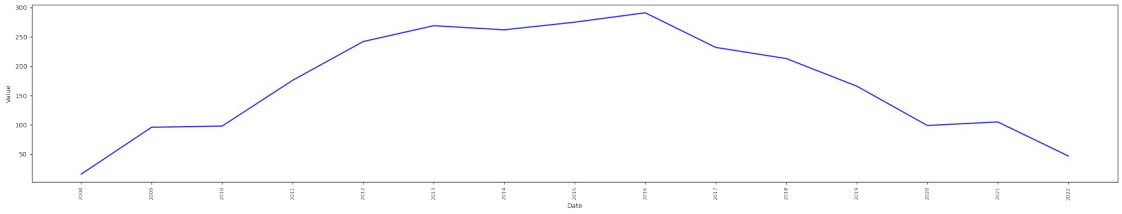


```
time_series_reviews2 =
listings.groupby(['host_year']).number_of_reviews.nunique().reset_inde
```

```
x()
time_series_reviews2
```

	host_year	number_of_reviews
0	2008	16
1	2009	96
2	2010	98
3	2011	176
4	2012	242
5	2013	269
6	2014	262
7	2015	275
8	2016	291
9	2017	232
10	2018	213
11	2019	166
12	2020	99
13	2021	105
14	2022	47

```
#Yearly View for th number of reviews
plot_df(time_series_reviews2, x=time_series_reviews2.host_year,
y=time_series_reviews2.number_of_reviews)
```



What do you see in supply and demand? Do you think it is a good time to invest residential real estate for short term rental like Airbnb in this area?

As it was observed from the time vs unique host ID counts, there has been a gradual increasing trend upto May, 2016. Upto 2020 the trend has been decreasing. Aterwards the trend is up and down and at the present the supply is at its lowest. Therefore as the supply is less the prices could be really high as the demand is also high. Therefore it can not be reccommended to invest for short term at the momemt.

Do you see seasonality in the supply and demand analysis? Is there a better time to start hosting (when demand is higher than supply)? How does number of reviews change across months in each year?

plot number of reviews vs. month for each year.

The two line charts as requested in the question are plotted in the previous section. The graps showed a seasonal pattern in the number of reviews (demand). Therefore again number of reviews were plotted against the month as below to observe the month on

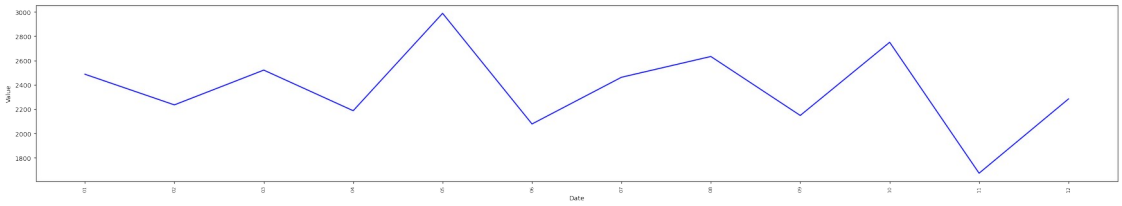
month count. According to that May has the highest demand while October has the second highest demand. November demand is lowest and should not start hosting in November. Therefore October and May are recommended for hosting.

```
# your code and plot here

# Monthly number of reviews
time_series_reviews3 =
listings.groupby(['host_month']).number_of_reviews.count().reset_index
()
time_series_reviews3
```

```
   host_month  number_of_reviews
0          01               2487
1          02               2235
2          03               2521
3          04               2187
4          05               2988
5          06               2078
6          07               2462
7          08               2633
8          09               2148
9          10               2750
10         11               1673
11         12               2284
```

```
#plot the month vs revies
plot_df(time_series_reviews3, x=time_series_reviews3.host_month,
y=time_series_reviews3.number_of_reviews)
```




```
months_in_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul',
'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

#First five rows of monthly number of reviews table
time_series_reviews.head()
```

```
  host_year_and_month  number_of_reviews
0             2008-07                 14
1             2008-09                  1
2             2008-12                  1
```

```
3             2009-02                 27
4             2009-03                  5
```

```
#Extracting the year and month from above table
time_series_reviews['host_year'] =
time_series_reviews['host_year_and_month'].str.split('-').str[0]
time_series_reviews['host_month'] =
time_series_reviews['host_year_and_month'].str.split('-').str[1]
time_series_reviews.head()
```

```
  host_year_and_month  number_of_reviews host_year host_month
0             2008-07                 14      2008         07
1             2008-09                  1      2008         09
2             2008-12                  1      2008         12
3             2009-02                 27      2009         02
4             2009-03                  5      2009         03
```

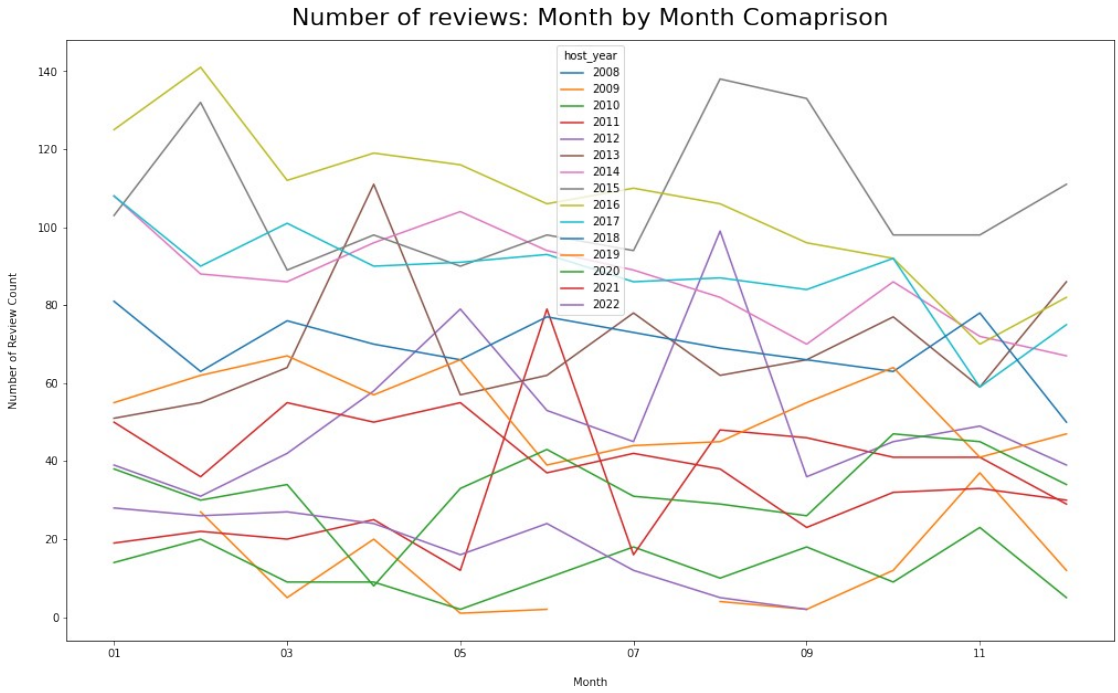
```
#Creating a pivot plot host month, host yeast vs number of reviews
df_pivoted = time_series_reviews.pivot(index='host_month',
                           columns='host_year',
                           values='number_of_reviews')
df_pivoted
```

```
host_year   2008  2009  2010  2011  2012   2013   2014   2015   2016
2017  \
host_month

01           NaN   NaN  14.0  19.0  39.0   51.0  108.0  103.0  125.0
108.0
02           NaN  27.0  20.0  22.0  31.0   55.0   88.0  132.0  141.0
90.0
03           NaN   5.0   9.0  20.0  42.0   64.0   86.0   89.0  112.0
101.0
04           NaN  20.0   9.0  25.0  58.0  111.0   96.0   98.0  119.0
90.0
05           NaN   1.0   2.0  12.0  79.0   57.0  104.0   90.0  116.0
91.0
06           NaN   2.0  10.0  79.0  53.0   62.0   94.0   98.0  106.0
93.0
07          14.0   NaN  18.0  16.0  45.0   78.0   89.0   94.0  110.0
86.0
08           NaN   4.0  10.0  48.0  99.0   62.0   82.0  138.0  106.0
87.0
09           1.0   2.0  18.0  46.0  36.0   66.0   70.0  133.0   96.0
84.0
10           NaN  12.0   9.0  41.0  45.0   77.0   86.0   98.0   92.0
92.0
11           NaN  37.0  23.0  41.0  49.0   59.0   72.0   98.0   70.0
59.0
12           1.0  12.0   5.0  29.0  39.0   86.0   67.0  111.0   82.0
75.0
```

```
host_year   2018  2019  2020  2021  2022
host_month
01          81.0  55.0  38.0  50.0  28.0
02          63.0  62.0  30.0  36.0  26.0
03          76.0  67.0  34.0  55.0  27.0
04          70.0  57.0   8.0  50.0  24.0
05          66.0  66.0  33.0  55.0  16.0
06          77.0  39.0  43.0  37.0  24.0
07          73.0  44.0  31.0  42.0  12.0
08          69.0  45.0  29.0  38.0   5.0
09          66.0  55.0  26.0  23.0   2.0
10          63.0  64.0  47.0  32.0   NaN
11          78.0  41.0  45.0  33.0   NaN
12          50.0  47.0  34.0  30.0   NaN
```

```
df_pivoted.plot(kind='line', figsize=(17, 10))
plt.title("Number of reviews: Month by Month Comaprison", y=1.013,
fontsize=22)
plt.xlabel("Month", labelpad=16)
plt.ylabel("Number of Review Count", labelpad=16);
```



How about average nightly rent? does it show change over the years and seasonality?

Answer:

```
# your code and plot here
```

The average nightly rent is about $476 according to the dataset. A line chart was plotted to show the average monthly rate vs month. The graph has a irreguar pattern

and does not include a trend. In 2021 October a huge sudden growth is observed. Might be due to some outlieres in that month, as the next month has just an average range. Another peak is observed in April, 2018. However the patterns are different year to year therefore can not specifiacally point out seasonal behavior.

```
#The values in price columns included with $ mark therefore we need to
remove them.
listings['price']
```

```
0          $175.00
1           $83.00
2          $622.00
3           $70.00
4          $206.00
           ...
28575      $300.00
28576       $55.00
28577    $2,300.00
28578      $999.00
28579      $425.00
Name: price, Length: 28580, dtype: object
```

```
#Removing the $ symbol and  converting into float data type.
listings['price'] = listings['price'].str.replace('[$, ]',
'').astype(float)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2:
FutureWarning: The default value of regex will change from True to
False in a future version.
```

```
#First 5 rows of price ($ removed)
listings['price'].head()
```

```
0    175.0
1     83.0
2    622.0
3     70.0
4    206.0
Name: price, dtype: float64
```

```
#Average Price
listings['price'].mean()
```

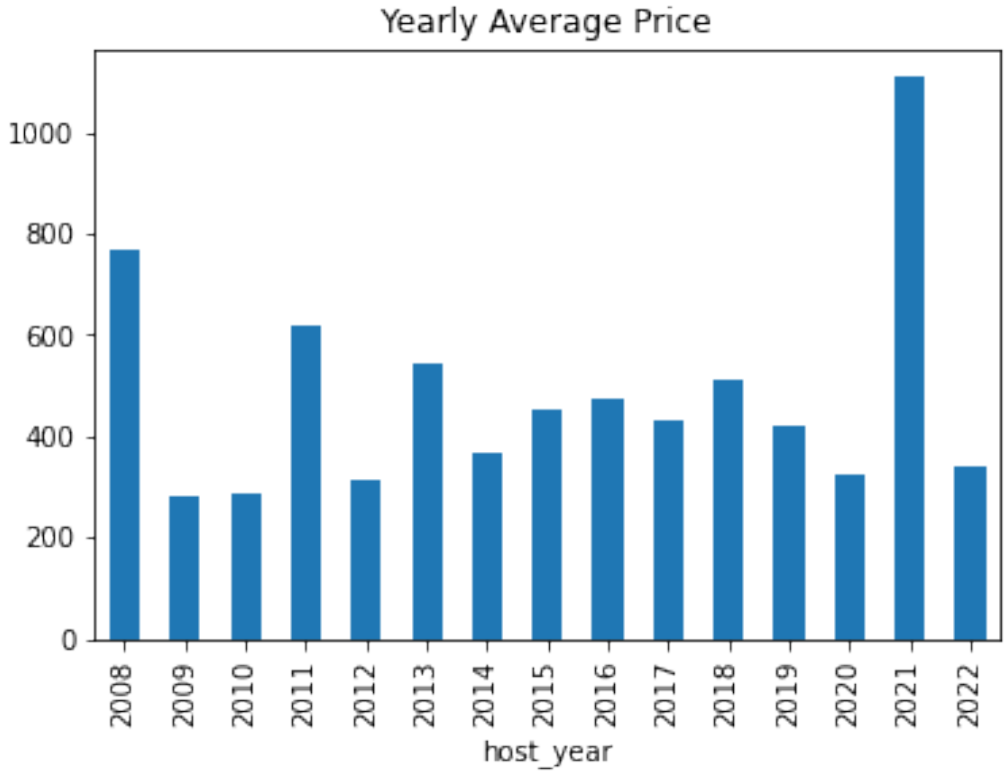
```
476.1075577326802
```

```
#Yearly Average Price
yearly_avg_price = listings.groupby('host_year')['price'].mean()
yearly_avg_price
```

```
host_year
2008     766.400000
```

```
2009     280.828571
2010     287.060440
2011     619.852978
2012     314.863229
2013     545.652084
2014     365.040593
2015     452.049553
2016     475.278613
2017     430.828709
2018     512.635662
2019     421.274272
2020     325.493066
2021    1108.717742
2022     338.502591
Name: price, dtype: float64
```

```
yearly_avg_price.plot(kind='bar')
plt.title("Yearly Average Price")
plt.show()
```

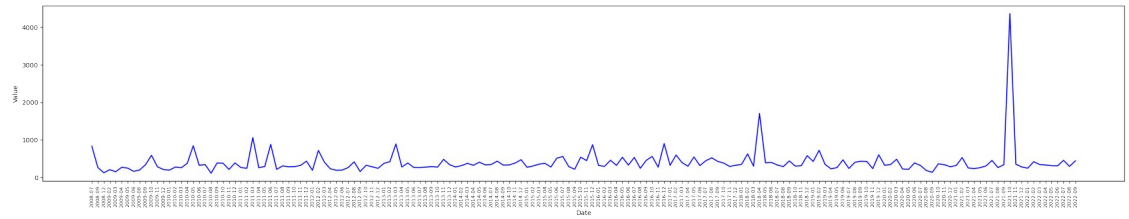



```
# Monthly average price
time_series_price =
listings.groupby(['host_year_and_month']).price.mean().reset_index()
time_series_price.head()
```

```
  host_year_and_month       price
0             2008-07  829.944444
```

```
1              2008-09  264.000000
2              2008-12  125.000000
3              2009-02  207.193548
4              2009-03  152.800000
```

```
plot_df(time_series_price, x=time_series_price.host_year_and_month,
y=time_series_price.price)
```



Analyze the data to answer the question: if average prices on certain days were higher compared to the other days. To be more specific, if it is more expensive to travel on weekends?

Plot a box plot of average prices by day of the week to understand this phenomenon.

```
# your code and plot here
```

```
#first 5 rows of the calender dataset
calendar.head()
```

```
   listing_id        date available    price adjusted_price
minimum_nights  \
0    35066424  2022-09-13         f  $175.00        $175.00
30.0
1    35066424  2022-09-14         f  $175.00        $175.00
30.0
2    35066424  2022-09-15         f  $175.00        $175.00
30.0
3    35066424  2022-09-16         f  $175.00        $175.00
30.0
4    35066424  2022-09-17         f  $175.00        $175.00
30.0

   maximum_nights
0           365.0
1           365.0
2           365.0
3           365.0
4           365.0
```

```
calendar.shape
```

```
(10431371, 7)
```

The dataset doesnot contain a weekday variable, therefore we need to extract weekday from the date variable.

```
#modify the calendar dataframe to add new column for month and weekday
for that particular date

#run with a limit as high power is required to run full.
#calendar_small = calendar.copy()
calendar_small = calendar.head(1000000)

#modify the calendar dataframe to add new column for month and weekday
for that particular date
#this will take more than 7mins
from datetime import datetime
calendar_small['weekday'] =
[datetime.strptime(calendar_small.iloc[i,1],'%Y-%m-%d').weekday() for
i in range(len(calendar_small.iloc[:]))]
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
```

```
calendar_small.head()
```

```
   listing_id        date available    price adjusted_price
minimum_nights  \
0    35066424  2022-09-13         f  $175.00        $175.00
30.0
1    35066424  2022-09-14         f  $175.00        $175.00
30.0
2    35066424  2022-09-15         f  $175.00        $175.00
30.0
3    35066424  2022-09-16         f  $175.00        $175.00
30.0
4    35066424  2022-09-17         f  $175.00        $175.00
30.0

   maximum_nights  weekday
0           365.0        1
1           365.0        2
2           365.0        3
3           365.0        4
4           365.0        5
```

```
calendar_small['weekday'].nunique()
```

```
7
```

```
#Remove the $ mark in price and converting to float type.
calendar_small['price'] = calendar_small['price'].str.replace('[$, ]',
'').astype(float)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2:
FutureWarning: The default value of regex will change from True to
False in a future version.

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
```

```
calendar_small['weekday'] = calendar_small['weekday'].map({0:
'Monday',
                                                           1:
'Tuesday',
                                                           2:
'Wedensday',
                                                           3:
'Thursday',
                                                           4:
'Friday',
                                                           6:
'Saturday',
                                                           6:
'Sunday',
                                                           })
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  import sys
```

```
calendar_small['weekday']
```

```
0          Tuesday
1        Wedensday
2         Thursday
```

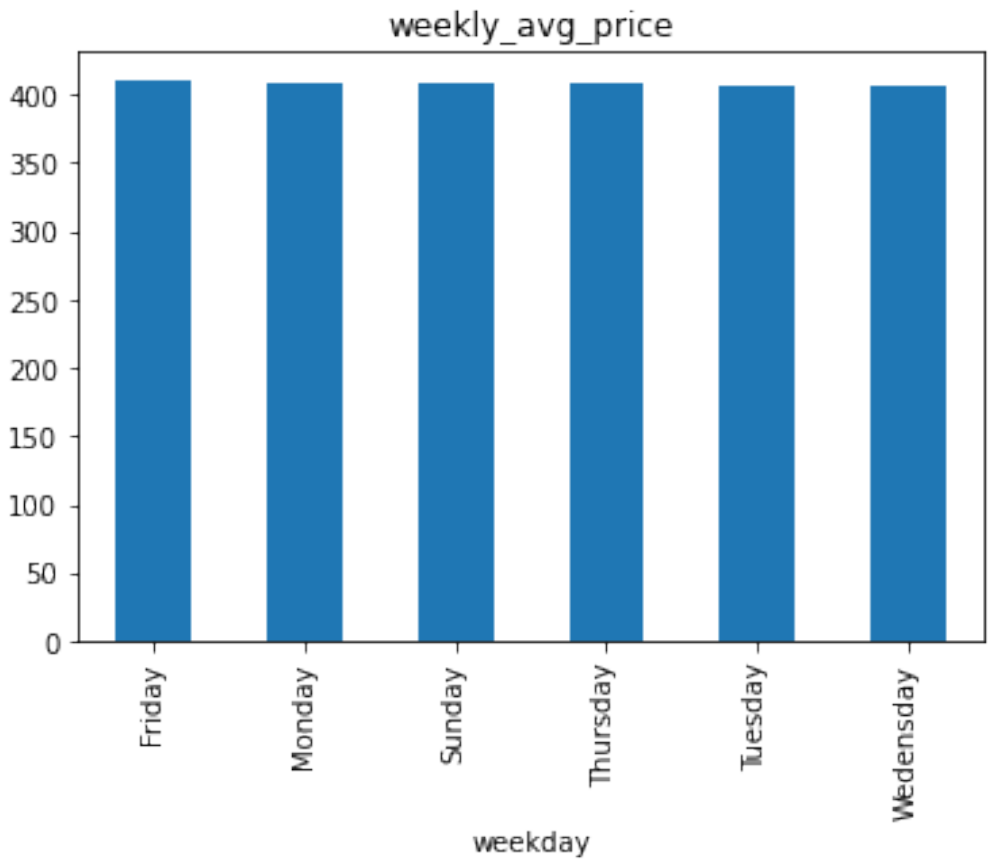
```
3            Friday
4               NaN
            ...
999995       Sunday
999996       Monday
999997      Tuesday
999998    Wedensday
999999     Thursday
Name: weekday, Length: 1000000, dtype: object
```

```
weekly_avg_price = calendar_small.groupby('weekday')['price'].mean()
```

```
weekly_avg_price
```

```
weekday
Friday       410.732376
Monday       407.452292
Sunday       408.879472
Thursday     408.064745
Tuesday      407.267835
Wedensday    407.401222
Name: price, dtype: float64
```

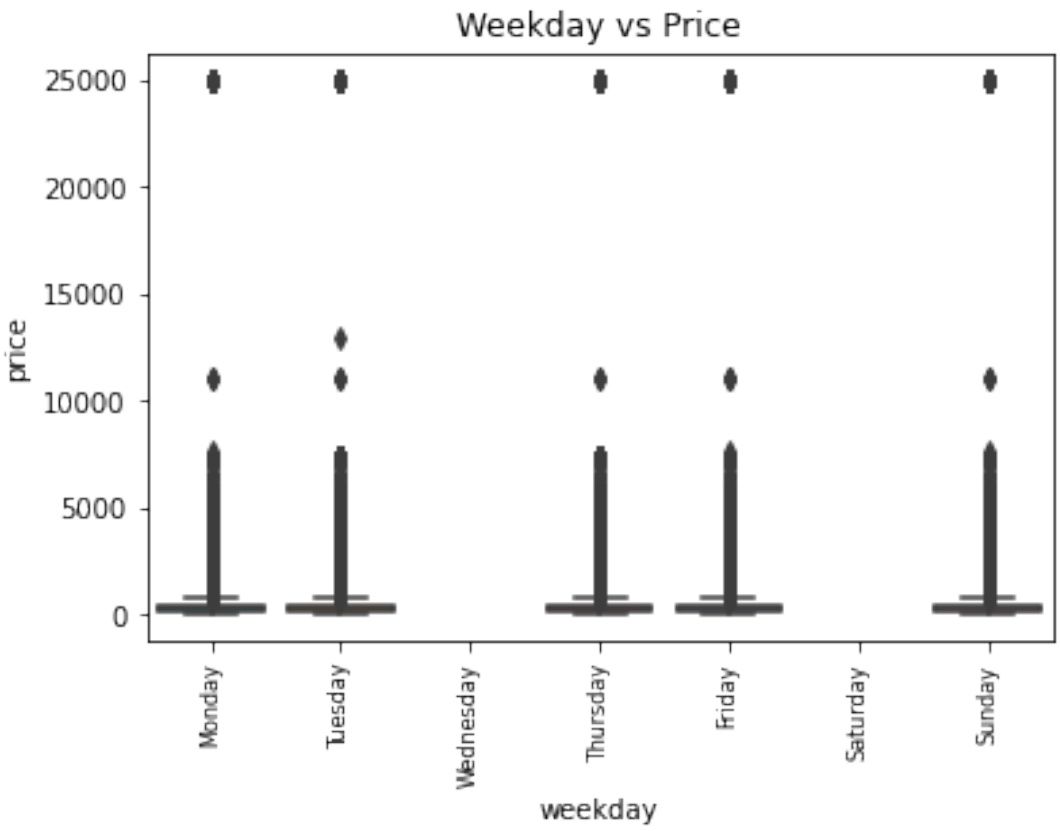
```
weekly_avg_price.plot(kind='bar')
plt.title("weekly_avg_price")
plt.show()
```




```
weekly_avg_price1 = calendar_small.groupby('weekday')
['price'].mean().reset_index()
weekly_avg_price1.head()
```

```
    weekday       price
0    Friday  410.732376
1    Monday  407.452292
2    Sunday  408.879472
3  Thursday  408.064745
4   Tuesday  407.267835
```

```
import seaborn as sns
order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',
'Saturday', 'Sunday']
sns.boxplot(x="weekday", y="price", data=calendar_small, order=order)
plt.title("Weekday vs Price")
plt.xticks(fontsize=8,rotation = 90)
plt.show()
```


Weekday vs Price
price
25000
20000
15000
10000
5000
0
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
weekday