Title : AirBnB Listings and Reviews

Name : Marishel Lochan

Year : 2025

Introduction

This is an AirBnB Listing analysis project provided by

Maven Analytics

Objective:

The objective of this project is to analyze AirBnB Listings in Paris to determine the impact of recent regulations

In [ ]:

In [7]:
```
!pip install matplotlib
```

```
Collecting matplotlib
  Downloading matplotlib-3.10.1-cp311-cp311-win_amd64.whl.metadata (11 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.3.1-cp311-cp311-win_amd64.whl.metadata (5.4 kB)
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.57.0-cp311-cp311-win_amd64.whl.metadata (104 kB)
Collecting kiwisolver>=1.3.1 (from matplotlib)
  Downloading kiwisolver-1.4.8-cp311-cp311-win_amd64.whl.metadata (6.3 kB)
Requirement already satisfied: numpy>=1.23 in c:\users\shelm\appdata\local\packages\pythonsoftwarefoundation.pyt
hon.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from matplotlib) (2.2.4)
Requirement already satisfied: packaging>=20.0 in c:\users\shelm\appdata\local\packages\pythonsoftwarefoundation
.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\shelm\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from matplotlib) (11.1.0)
Collecting pyparsing>=2.3.1 (from matplotlib)
  Downloading pyparsing-3.2.3-py3-none-any.whl.metadata (5.0 kB)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\shelm\appdata\local\packages\pythonsoftwarefound
ation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from matplotlib) (2.9.0.post0
)
Requirement already satisfied: six>=1.5 in c:\users\shelm\appdata\local\packages\pythonsoftwarefoundation.python
.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (from python-dateutil>=2.7->matplotlib) (1
.17.0)
Downloading matplotlib-3.10.1-cp311-cp311-win_amd64.whl (8.1 MB)
   ---------------------------------------- 0.0/8.1 MB ? eta -:--:--
   -- ------------------------------------- 0.5/8.1 MB 4.2 MB/s eta 0:00:02
   ------- -------------------------------- 1.6/8.1 MB 5.6 MB/s eta 0:00:02
   ------------- -------------------------- 2.9/8.1 MB 5.2 MB/s eta 0:00:01
   ------------------- -------------------- 4.2/8.1 MB 5.9 MB/s eta 0:00:01
   ------------------------- ------------ 5.8/8.1 MB 5.9 MB/s eta 0:00:01
   ------------------------- ------------ 5.8/8.1 MB 5.9 MB/s eta 0:00:01
   ------------------------------ ------- 6.8/8.1 MB 4.8 MB/s eta 0:00:01
   ---------------------------------------  7.9/8.1 MB 4.9 MB/s eta 0:00:01
   ---------------------------------------- 8.1/8.1 MB 4.3 MB/s eta 0:00:00
Downloading contourpy-1.3.1-cp311-cp311-win_amd64.whl (219 kB)
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Downloading fonttools-4.57.0-cp311-cp311-win_amd64.whl (2.2 MB)
   ---------------------------------------- 0.0/2.2 MB ? eta -:--:--
   ------------------ --------------------- 1.0/2.2 MB 5.6 MB/s eta 0:00:01
   ---------------------------------------- 2.2/2.2 MB 5.2 MB/s eta 0:00:00
Downloading kiwisolver-1.4.8-cp311-cp311-win_amd64.whl (71 kB)
Downloading pyparsing-3.2.3-py3-none-any.whl (111 kB)
Installing collected packages: pyparsing, kiwisolver, fonttools, cycler, contourpy, matplotlib
Successfully installed contourpy-1.3.1 cycler-0.12.1 fonttools-4.57.0 kiwisolver-1.4.8 matplotlib-3.10.1 pyparsi
ng-3.2.3
```

In [8]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```
```

Data

The Airbnb data has about 250,000+ listings across 10 major cities, along with ~5 million guest reviews.

In [ ]:
```
```

In [14]:
```
listings = pd.read_csv('Listings.csv',encoding='latin1', low_memory=False)
listings.head()
```

Out[14]:

	listing_id	name	host_id	host_since	host_location	host_response_time	host_response_rate	host_acceptance_rate	hos
0	281420	Beautiful Flat in le Village Montmartre, Paris	1466919	2011-12-03	Paris, Ile-de-France, France	NaN	NaN	NaN	
1	3705183	39 mÃ² Paris (Sacre CÃ âur)	10328771	2013-11-29	Paris, Ile-de-France, France	NaN	NaN	NaN	
2	4082273	Lovely apartment with Terrace, 60m2	19252768	2014-07-31	Paris, Ile-de-France, France	NaN	NaN	NaN	
3	4797344	Cosy studio (close to Eiffel tower)	10668311	2013-12-17	Paris, Ile-de-France, France	NaN	NaN	NaN	
4	4823489	Close to Eiffel Tower - Beautiful flat : 2 rooms	24837558	2014-12-14	Paris, Ile-de-France, France	NaN	NaN	NaN	

5 rows × 33 columns

In [ ]:

Objective 1

We needed to first AirBnB listings data, calculate basic profiling metrics, change column datatypes as necessary, and then filter down to only Paris Listings.

- First thing I did was change the format of the Dates in the data

In [ ]:

In [15]:
```
pd.to_datetime(listings['host_since'])
```

Out[15]:
```
0        2011-12-03
1        2013-11-29
2        2014-07-31
3        2013-12-17
4        2014-12-14
            ...
279707   2015-04-13
279708   2013-11-27
279709   2012-04-27
279710   2015-07-16
279711   2013-06-17
Name: host_since, Length: 279712, dtype: datetime64[ns]
```

In [ ]:

- I then was able to filter to only the listings with the city "Paris"

In [ ]:

In [35]:
```
paris = listings['city']=='Paris'
print(paris)
```

```
0         True
1         True
2         True
3         True
4         True
          ...
279707    True
279708    True
279709    True
279710    True
279711    True
Name: city, Length: 279712, dtype: bool
```

In [ ]:

- Before doing further analysis, I filled in any missing values in columns that will be delt with in this analysis

In [ ]:
```
```

In [33]:
```
listings.isnull().sum()
```

Out[33]:
```
listing_id                          0
name                              175
host_id                             0
host_since                        165
host_location                     840
host_response_time             128782
host_response_rate             128782
host_acceptance_rate           113087
host_is_superhost                 165
host_total_listings_count         165
host_has_profile_pic              165
host_identity_verified            165
neighbourhood                       0
district                       242700
city                                0
latitude                            0
longitude                           0
property_type                       0
room_type                           0
accommodates                        0
bedrooms                        29435
amenities                           0
price                               0
minimum_nights                      0
maximum_nights                      0
review_scores_rating            91405
review_scores_accuracy          91713
review_scores_cleanliness       91665
review_scores_checkin           91771
review_scores_communication     91687
review_scores_location          91775
review_scores_value             91785
instant_bookable                    0
dtype: int64
```

In [37]:
```
most_frequent_date = listings['host_since'].mode()
listings['host_since'].fillna(most_frequent_date)
```

Out[37]:
```
0         2011-12-03
1         2013-11-29
2         2014-07-31
3         2013-12-17
4         2014-12-14
             ...
279707    2015-04-13
279708    2013-11-27
279709    2012-04-27
279710    2015-07-16
279711    2013-06-17
Name: host_since, Length: 279712, dtype: object
```

In [ ]:
```
```

- After filling the missing values, I then created a 'paris_listings' that held only listings with the city 'Paris' but only kept the columns 'host_since', 'neighbourhood', 'city', 'accommodates' and 'price'.

In [ ]:
```
```

In [38]:
```
paris_listings = listings.loc[listings['city']=='Paris',['host_since','neighbourhood'
                                                                                ,'city'
                                                                                , 'accommodates',
                                                                                'price']]
print(paris_listings)
```

```
        host_since      neighbourhood   city  accommodates  price
0       2011-12-03  Buttes-Montmartre  Paris             2     53
1       2013-11-29  Buttes-Montmartre  Paris             2    120
2       2014-07-31             Elysee  Paris             2     89
3       2013-12-17          Vaugirard  Paris             2     58
4       2014-12-14              Passy  Paris             2     60
...            ...                ...    ...           ...    ...
279707  2015-04-13       Observatoire  Paris             2    120
279708  2013-11-27  Buttes-Montmartre  Paris             2     60
279709  2012-04-27  Buttes-Montmartre  Paris             2     50
279710  2015-07-16         Popincourt  Paris             2    105
279711  2013-06-17  Enclos-St-Laurent  Paris             2     70

[64690 rows x 5 columns]
```

In [ ]:

- To understand the data a bit, I calcuated the mean, maximum and minimum of both the 'accommodates' and 'price' in the 'paris_listings'

In [ ]:

In [41]:
```
accommodation_avg = paris_listings['accommodates'].mean()
accommodation_minimum = paris_listings['accommodates'].min()
accommodation_maximum = paris_listings['accommodates'].max()
print(accommodation_avg)
print(accommodation_minimum)
print(accommodation_maximum)
```

```
3.0379965991652496
0
16
```

In [137…
```
price_avg = paris_listings['price'].mean()
price_minimum = paris_listings['price'].min()
price_maximum = paris_listings['price'].max()
print(price_avg)
print(price_minimum)
print(price_maximum)
```

```
113.09644458185191
0
12000
```

In [ ]:

- I saw for both the 'accommodates' and 'price' columns had a minimum od zero, as such I filtered the 'paris_listings' to ensure that there were zeros in the dataset.

In [ ]:

In [43]:
```
paris_listings.loc[paris_listings['price'] == 0]
```

Out[43]:

	host_since	neighbourhood	city	accommodates	price
98209	2020-07-20	Pantheon	Paris	0	0
203257	2020-02-04	Batignolles-Monceau	Paris	0	0
203258	2016-10-17	Opera	Paris	0	0
203259	2020-04-24	Luxembourg	Paris	0	0
203260	2020-04-24	Vaugirard	Paris	0	0
...	...	...	...	...	...
208881	2020-10-22	Pantheon	Paris	0	0
208882	2020-11-26	Enclos-St-Laurent	Paris	0	0
208883	2020-11-26	Vaugirard	Paris	0	0
208884	2020-12-21	Vaugirard	Paris	0	0
212834	2020-02-03	Enclos-St-Laurent	Paris	0	0

62 rows × 5 columns

In [ ]:

Objective 2:

The second objective was to prepare the data for visualization and this meant filtering and querying the data to produce valuable information.

- Firstly, a table was created to list all the neighbourhoods in 'Paris' and calculate their average prices.

In [ ]:

In [135…
```
paris_listings_neighbourhood = paris_listings.groupby('neighbourhood')['price'].mean().sort_values()
print(paris_listings_neighbourhood)
```

```
neighbourhood
Menilmontant            74.942257
Buttes-Chaumont         82.690182
Buttes-Montmartre       87.209479
Reuilly                 89.058402
Popincourt              90.559459
Gobelins                98.110184
Observatoire           101.866801
Batignolles-Monceau    102.612702
Enclos-St-Laurent      102.967156
Vaugirard              106.831330
Opera                  119.038644
Pantheon               122.662150
Temple                 138.446823
Hotel-de-Ville         144.472110
Bourse                 149.496801
Luxembourg             155.638639
Palais-Bourbon         156.856578
Passy                  161.144635
Louvre                 175.379972
Elysee                 210.536765
Name: price, dtype: float64
```

In [ ]:

- The Second findings were listing the average prices of each accommodation in the most expensive neighbourhood in Paris

In [ ]:

In [69]:
```
men = paris_listings_neighbourhood.idxmax() # gets the max index
print(men)
```

```
Elysee
```

In [70]:
```
men_listings= paris_listings.loc[paris_listings['neighbourhood']==men]
print(men_listings)
```

```
        host_since neighbourhood   city  accommodates  price
2       2014-07-31        Elysee  Paris             2     89
14      2015-12-30        Elysee  Paris             2     35
128     2015-03-26        Elysee  Paris             2     75
137     2015-08-23        Elysee  Paris             2     90
260     2014-07-18        Elysee  Paris             2    110
...            ...           ...    ...           ...    ...
278484  2016-07-22        Elysee  Paris             2     98
279043  2016-05-09        Elysee  Paris             2     75
279117  2014-11-20        Elysee  Paris             2    100
279299  2014-09-30        Elysee  Paris             2     87
279618  2013-04-15        Elysee  Paris             2    119

[1768 rows x 5 columns]
```

In [72]:
```
paris_listings_accommodations = men_listings.groupby('accommodates')['price'].mean().sort_values()
print(paris_listings_accommodations)
```

```
accommodates
0       0.000000
1      79.522222
3     152.828767
2     155.103352
4     212.096070
5     328.817073
6     355.508571
8     405.518519
7     411.538462
9     440.272727
10    500.857143
12    529.625000
16    800.000000
11    805.000000
13    842.500000
14    971.000000
Name: price, dtype: float64
```

In [ ]:

- Before proceeding, I took a look at the dataset to ensure I understand how I was going to filter the next table.

In [ ]:

In [73]:
```
paris_listings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 64690 entries, 0 to 279711
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   host_since     64657 non-null  object
 1   neighbourhood  64690 non-null  object
 2   city           64690 non-null  object
 3   accommodates   64690 non-null  int64 
 4   price          64690 non-null  int64 
dtypes: int64(2), object(3)
memory usage: 5.0+ MB
```

In [ ]:

- Last Table created was to find the average prices as well as the number of new hosts over the period of time from the first listing to the last.

- Before filtering, I decided to split the 'host_since' date column and split the year, month and day into their own columns in the 'paris_listings' to make it easier to filter.

In [ ]:

In [78]:
```
dates = paris_listings['host_since'].str.split('-',expand=True)
print(dates)
```

```
           0   1   2
0       2011  12  03
1       2013  11  29
2       2014  07  31
3       2013  12  17
4       2014  12  14
...      ...  ..  ..
279707  2015  04  13
279708  2013  11  27
279709  2012  04  27
279710  2015  07  16
279711  2013  06  17

[64690 rows x 3 columns]
```

In [86]:
```
paris_listings['Year'] = dates.iloc[:,0]
paris_listings['Month'] = dates.iloc[:,1]
paris_listings['Day'] = dates.iloc[:,2]
print(paris_listings)
```

```
        host_since      neighbourhood   city  accommodates  price  \
0       2011-12-03  Buttes-Montmartre  Paris             2     53
1       2013-11-29  Buttes-Montmartre  Paris             2    120
2       2014-07-31            Elysee  Paris             2     89
3       2013-12-17          Vaugirard  Paris             2     58
4       2014-12-14              Passy  Paris             2     60
...            ...                ...    ...           ...    ...
279707  2015-04-13       Observatoire  Paris             2    120
279708  2013-11-27  Buttes-Montmartre  Paris             2     60
279709  2012-04-27  Buttes-Montmartre  Paris             2     50
279710  2015-07-16         Popincourt  Paris             2    105
279711  2013-06-17  Enclos-St-Laurent  Paris             2     70

       (Year, Month, Day)  Year Month Day
0            [2011-12-03]  2011    12  03
1            [2013-11-29]  2013    11  29
2            [2014-07-31]  2014    07  31
3            [2013-12-17]  2013    12  17
4            [2014-12-14]  2014    12  14
...                   ...   ...   ...  ..
279707       [2015-04-13]  2015    04  13
279708       [2013-11-27]  2013    11  27
279709       [2012-04-27]  2012    04  27
279710       [2015-07-16]  2015    07  16
279711       [2013-06-17]  2013    06  17

[64690 rows x 9 columns]
```

In [88]:
```
paris_listings_over_time = paris_listings.groupby('Year').agg(averagePrice=('price','mean'),newHosts=('Year','c
print(paris_listings_over_time)
```

```
      averagePrice  newHosts
Year
2008     77.750000         4
2009    159.641509       106
2010    125.031250       416
2011    124.828230      1339
2012    111.578615      4592
2013    107.096414      8142
2014    100.253800     10922
2015    103.646250     12147
2016    114.159847      8871
2017    108.658888      4585
2018    138.209362      4294
2019    129.757113      5694
2020    141.456038      3412
2021     93.488722       133
```

In [ ]:
```

```

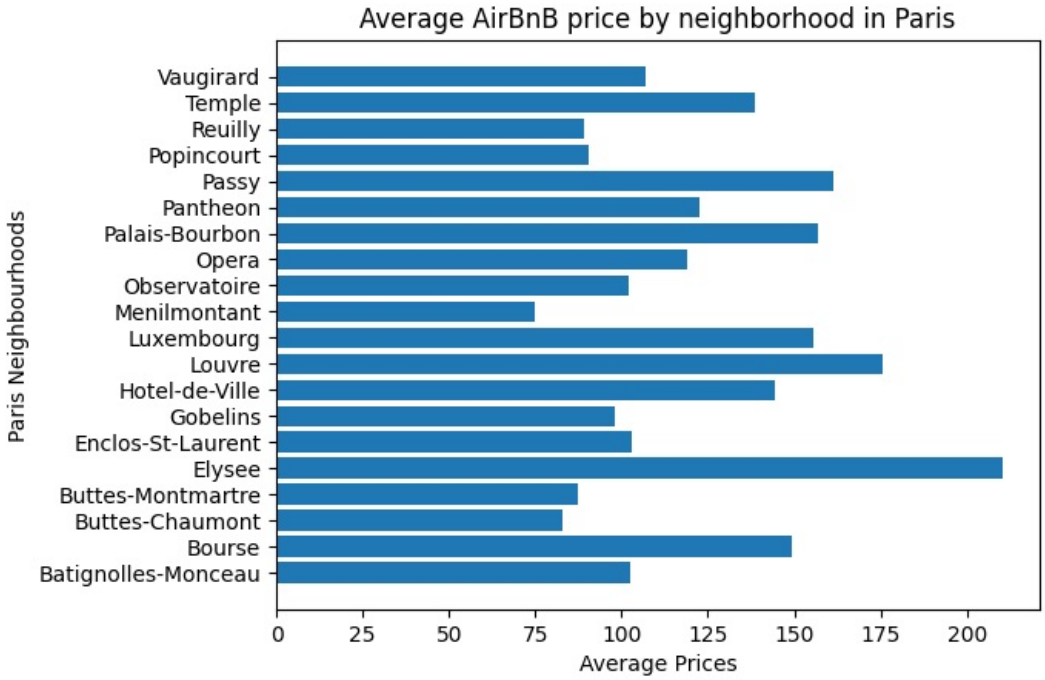
Objective3 : Visualization

In [ ]:
```

```

In [ ]:
```
paris_listings_neighbourhood = paris_listings_neighbourhood.reset_index()# to get index
```

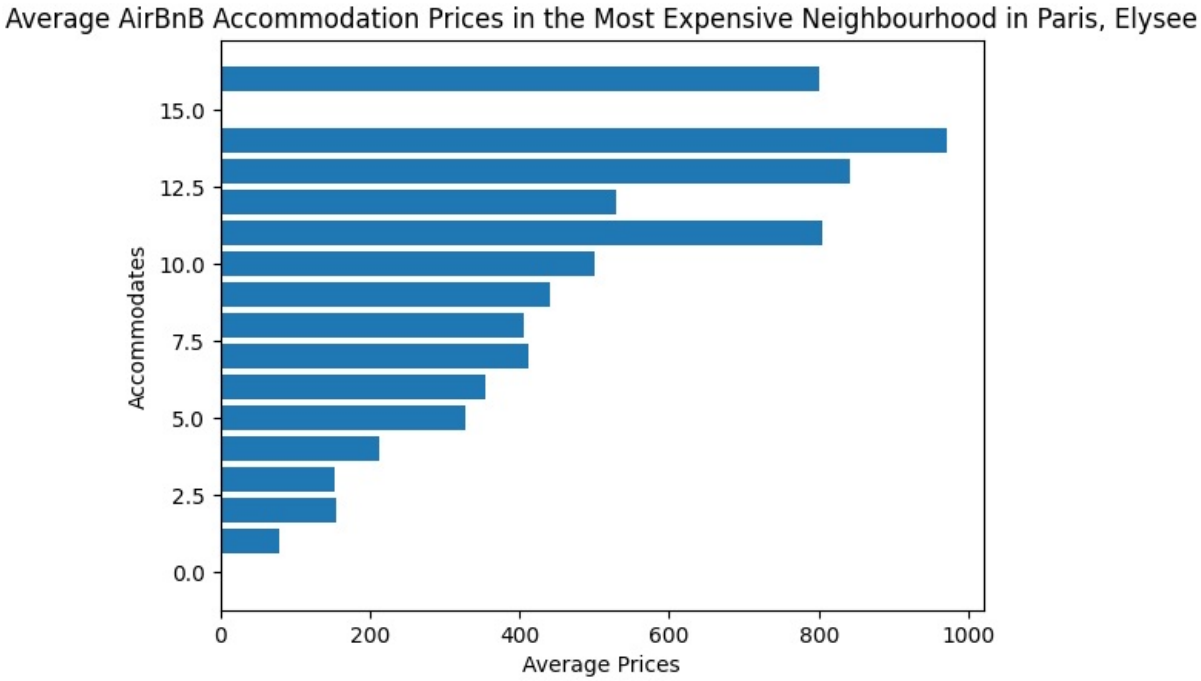
In [131…
```
plt.barh(paris_listings_neighbourhood['neighbourhood'], paris_listings_neighbourhood['price'])
plt.xlabel("Average Prices")
plt.ylabel("Paris Neighbourhoods")
plt.title("Average AirBnB price by neighborhood in Paris")
plt.show()
plt.clf()
```




```
<Figure size 640x480 with 0 Axes>
```

In [113…
```
paris_listings_accommodations = paris_listings_accommodations.reset_index()
```

In [130…
```
plt.barh(paris_listings_accommodations['accommodates'],paris_listings_accommodations['price'])
plt.xlabel("Average Prices")
plt.ylabel("Accommodates")
plt.title("Average AirBnB Accommodation Prices in the Most Expensive Neighbourhood in Paris, " + men)
plt.show()
plt.clf()
```

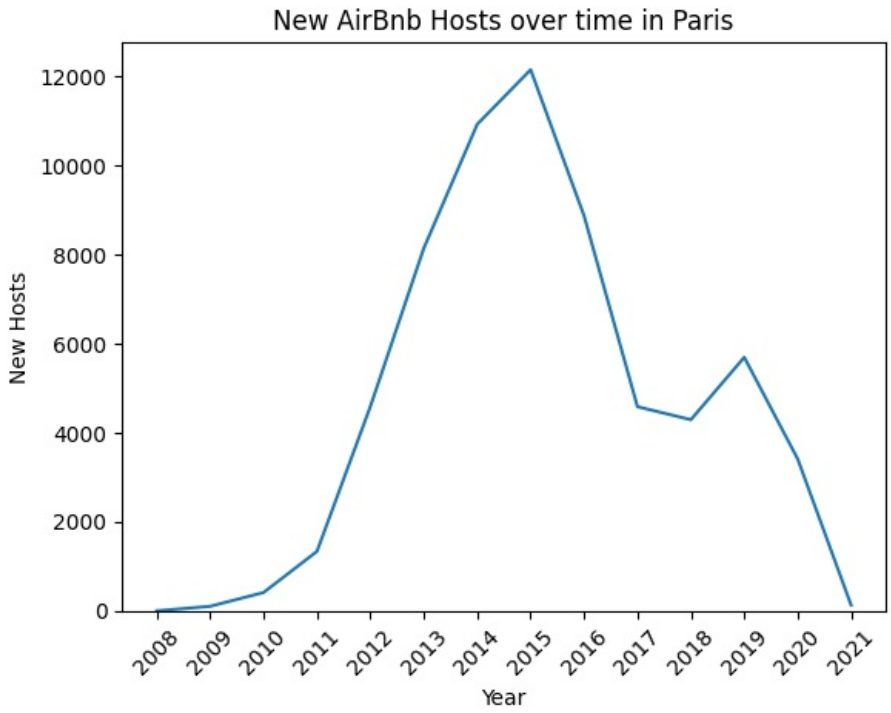



```
<Figure size 640x480 with 0 Axes>
```

In [ ]:
```
```

In [119…
```
paris_listings_over_time = paris_listings_over_time.reset_index()
```

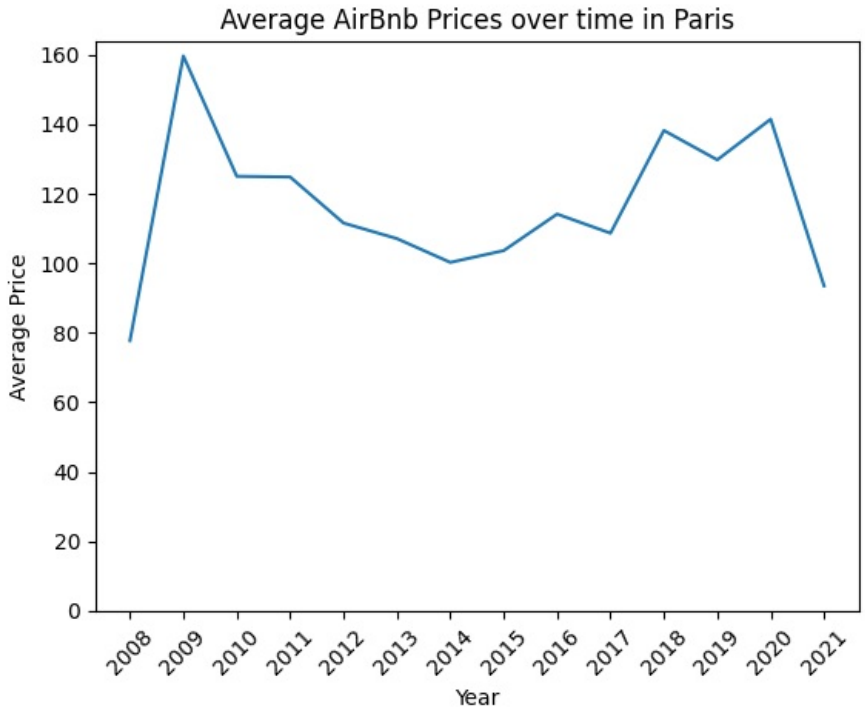
In [132…
```
plt.ploat(paris_listings_over_time['Year'], paris_listings_over_time['newHosts'])
plt.xticks(rotation=45)
plt.ylim(bottom=0)
plt.xlabel("Year")
plt.ylabel("New Hosts")
plt.title("New AirBnb Hosts over time in Paris")
plt.show()
plt.clf()
```




```
<Figure size 640x480 with 0 Axes>
```

In [133…

```
plt.plot(paris_listings_over_time['Year'], paris_listings_over_time['averagePrice'])
plt.xticks(rotation=45)
plt.ylim(bottom=0)
plt.xlabel("Year")
plt.ylabel("Average Price")
plt.title("Average AirBnb Prices over time in Paris")
plt.show()
plt.clf()
```




```
<Figure size 640x480 with 0 Axes>
```

In [ ]:

Findings

- Elysee is the most expensive neighbourhood in Paris
- 2014 and 2015 had the most amount of new Hosts yet though and were the two years with the lowest average prices.