Term Project Report

Airbnb Data Model

Member

65070501005	Khetsophon	Liu	Sec A
65070501029	Theerakan	Thadawuth	Sec A
65070501045	Mawin	Srichat	Sec B
65070501077	Tamonwan	Tabloi	Sec B
65070501093	Waruntorn	Tosakulvong	Sec B

Data Model (CPE241)

Data Resource

Airbnb data : https://www.kaggle.com/datasets/paramvir705/airbnb-data

Interesting:

- **The requisite:** ชุดข้อมูลประกอบด้วย 70,000 แถว และจำนวนคอลัมน์ที่เหมาะสม ทำให้าสามารถทำงานตามระยะเวลาที่กำหนดได้อย่างมีประสิทธิภาพ
- **Overview:** โปรเจคนี้มีวัตถุประสงค์เพื่อคาดการณ์ราคาที่อยู่อาศัยโดยใช้เทคนิคการวิเคราะห์ข้อมูลต่างๆ
- **Objectives:** ทำความเข้าใจปัจจัยสำคัญที่ส่งผลต่อราคาที่อยู่อาศัย โดยสร้างแบบจำลองการคาดการณ์
- **Importance:** ให้ข้อมูลเชิงลึกสำหรับการลงทุนด้านอสังหาริมทรัพย์และการวิเคราะห์ตลาด ปัญหาและความเข้าใจข้อมูล

Data Explanation

ชุดข้อมูลนี้เกี่ยวกับการเช่าระยะสั้นจากแพลตฟอร์มเช่น Airbnb โดยมีข้อมูลที่ละเอียดเกี่ยวกับที่พักต่าง ๆ ที่มีให้เช่า รวมถึงรายละเอียดของเจ้าของที่พักและสถิติการรีวิว ข้อมูลนี้สามารถใช้ในการวิเคราะห์รูปแบบราคาค่าเช่า ความต้องการของผู้เข้าพัก การประเมินกิจกรรมและการตอบสนองของเจ้าของที่พัก และการวิเคราะห์ความคิดเห็นและคะแนนรีวิว

รายละเอียดของแต่ละคอลัมน์

1. **id**: รหัสประจำที่พัก (ID) ของรายการ
2. **log_price**: ลอการิทึมของราคาที่พัก
3. **property_type**: ประเภทของที่พัก (เช่น อพาร์ทเมนต์, บ้าน)
4. **room_type**: ประเภทของห้องที่เสนอให้เช่า (เช่น ทั้งที่พัก, ห้องส่วนตัว, ห้องรวม)
5. **amenities**: รายการสิ่งอำนวยความสะดวกที่มีในที่พัก แสดงเป็นสตริงในรูปแบบ JSON
6. **accommodates**: จำนวนผู้เข้าพักที่ที่พักสามารถรองรับได้
7. **bathrooms**: จำนวนห้องน้ำในที่พัก

8. **bed_type**: ประเภทของเตียงที่มี (เช่น เตียงจริง, ฟูก)

9. **cancellation_policy**: ประเภทนโยบายการยกเลิก (เช่น เข้มงวด, ปานกลาง, ยืดหยุ่น)

10. **cleaning_fee**: ระบุว่ามีการคิดค่าทำความสะอาดหรือไม่ (True หรือ False)

11. **city**: เมืองที่ตั้งของที่พัก

12. **description**: คำอธิบายเกี่ยวกับที่พัก

13. **first_review**: วันที่ได้รับรีวิวครั้งแรก

14. **host_has_profile_pic**: ระบุว่าเจ้าของที่พักมีรูปโปรไฟล์หรือไม่ (t สำหรับมี)

15. **host_identity_verified**: ระบุว่าเจ้าของที่พักได้รับการยืนยันตัวตนหรือไม่ (t สำหรับใช่, f สำหรับไม่)

16. **host_response_rate**: อัตราการตอบกลับของเจ้าของที่พัก (เช่น 100%)

17. **host_since**: วันที่เจ้าของที่พักเข้าร่วมแพลตฟอร์ม

18. **instant_bookable**: ระบุว่าที่พักสามารถจองได้ทันทีหรือไม่ (t สำหรับใช่, f สำหรับไม่)

19. **last_review**: วันที่ได้รับรีวิวล่าสุด

20. **latitude**: ละติจูดของที่ตั้งที่พัก

21. **longitude**: ลองจิจูดของที่ตั้งที่พัก

22. **name**: ชื่อของรายการที่พัก

23. **neighbourhood**: ชื่อย่านที่ตั้งของที่พัก

24. **number_of_reviews**: จำนวนรีวิวทั้งหมดที่ที่พักได้รับ

25. **review_scores_rating**: คะแนนเฉลี่ยที่ได้รับจากรีวิว

26. **thumbnail_url**: URL ของรูปภาพขนาดย่อของที่พัก

27. **zipcode**: รหัสไปรษณีย์ของที่ตั้งที่พัก

28. **bedrooms**: จำนวนห้องนอนในที่พัก

29. **beds**: จำนวนเตียงที่มีในที่พัก

การสร้าง Model วิเคราะห์ราคาที่พักใน Airbnb

Import Data

Data source: https://www.kaggle.com/datasets/paramvir705/airbnb-data

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

[ ] pd.set_option('display.max_columns', None)

    airbnb_df = pd.read_csv('/content/Airbnb_Data.csv')

[ ] print(airbnb_df.shape)
    airbnb_df.head()
```

	id	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_policy	cleaning_fee	city	description	first_review	host_has_profile_pic	host_identity_verified	host_response_rate	host_since	instant_boo
0	6901257	5.010635	Apartment	Entire home/apt	{"Wireless Internet","Air conditioning",Kitche...	3	1.0	Real Bed	strict	True	NYC	Beautiful, sunlit brownstone 1-bedroom in the ...	2016-06-18	t	t	NaN	2012-03-26	
1	6304928	5.129899	Apartment	Entire home/apt	{"Wireless Internet","Air conditioning",Kitche...	7	1.0	Real Bed	strict	True	NYC	Enjoy travelling during your stay in Manhattan...	2017-08-05	t	f	100%	2017-06-19	
2	7919400	4.976734	Apartment	Entire home/apt	{TV,"Cable TV","Wireless Internet","Air condit...	5	1.0	Real Bed	moderate	True	NYC	The Oasis comes complete with a full backyard ...	2017-04-30	t	t	100%	2016-10-25	
3	13418779	6.620073	House	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet",Ki...	4	1.0	Real Bed	flexible	True	SF	This light-filled home-away-from-home is super...	NaN	t	t	NaN	2015-04-19	
4	3808709	4.744932	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	2	1.0	Real Bed	moderate	True	DC	Cool, cozy, and comfortable studio located in ...	2015-05-12	t	t	100%	2015-03-01	

```
[ ] airbnb_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74111 entries, 0 to 74110
Data columns (total 29 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   id                      74111 non-null  int64
 1   log_price               74111 non-null  float64
 2   property_type           74111 non-null  object
 3   room_type               74111 non-null  object
 4   amenities               74111 non-null  object
 5   accommodates            74111 non-null  int64
 6   bathrooms               73911 non-null  float64
 7   bed_type                74111 non-null  object
 8   cancellation_policy     74111 non-null  object
 9   cleaning_fee            74111 non-null  bool
 10  city                    74111 non-null  object
 11  description             74111 non-null  object
 12  first_review            58247 non-null  object
 13  host_has_profile_pic    73923 non-null  object
 14  host_identity_verified  73923 non-null  object
 15  host_response_rate      55812 non-null  object
 16  host_since              73923 non-null  object
 17  instant_bookable        74111 non-null  object
 18  last_review             58284 non-null  object
 19  latitude                74111 non-null  float64
 20  longitude               74111 non-null  float64
 21  name                    74111 non-null  object
 22  neighbourhood           67239 non-null  object
 23  number_of_reviews       74111 non-null  int64
 24  review_scores_rating    57389 non-null  float64
 25  thumbnail_url           65895 non-null  object
 26  zipcode                 73145 non-null  object
 27  bedrooms                74020 non-null  float64
 28  beds                    73980 non-null  float64
dtypes: bool(1), float64(7), int64(3), object(18)
memory usage: 15.9+ MB
```

```
[ ] airbnb_df.shape

(74111, 29)

airbnb_df.isnull().sum()

id                          0
log_price                   0
property_type               0
room_type                   0
amenities                   0
accommodates                0
bathrooms                 200
bed_type                    0
cancellation_policy         0
cleaning_fee                0
city                        0
description                 0
first_review            15864
host_has_profile_pic      188
host_identity_verified    188
host_response_rate      18299
host_since                188
instant_bookable            0
last_review             15827
latitude                    0
longitude                   0
name                        0
neighbourhood            6872
number_of_reviews           0
review_scores_rating    16722
thumbnail_url            8216
zipcode                   966
bedrooms                   91
beds                      131
dtype: int64
```

1. Data Cleansing

1.1 Clean bathrooms

In the US, we have full, three-quarters, and half-baths. A full bath includes a tub and shower (either separately or combined), a sink, and a toilet. A three-quarter bath has a tub OR shower, a sink, and a toilet. And a half-bath (sometimes called a “powder room”) is just a sink and toilet. On very rare occasions, you’ll find a one-quarter bath, which is just a toilet, but those are pretty much confined to the northeastern part of the US and usually only in older homes

```
pd.unique(airbnb_df['bathrooms'])

array([1. , 1.5, 2. , nan, 2.5, 3. , 0.5, 4.5, 5. , 0. , 4. , 3.5, 5.5,
       7.5, 6. , 8. , 7. , 6.5])
```

- ทำการ drop แถวที่มีค่า null ใน bathrooms ทั้งหมด

```
[ ]  airbnb_df.dropna(subset='bathrooms', inplace=True)
```

1.2 Clean first review and last review

First review คือ วันที่รีวิวครั้งแรกโดยลูกค้าที่เข้าพัก

Last review คือ วันที่รีวิวครั้งล่าสุดโดยลูกค้าที่เข้าพัก

- ทำการสุ่มเลือก 6 แถวจากคอลัมน์ first review

```
[ ]  airbnb_df['first_review'].sample(6)

id
9880541     2017-09-04
3946239     2013-07-17
18020906    2015-07-06
10676252    2017-05-29
21138275    2016-12-31
17340546    2015-09-07
Name: first_review, dtype: object

[ ]  #airbnb_df.loc[airbnb_df['first_review'].isnull(), 'first_review'] = airbnb_df.loc[airbnb_df['first_review'
```

- ทำการ drop แถวที่มีค่า null ใน first_review ทั้งหมด

```
[ ] airbnb_df.dropna(subset='first_review', inplace=True)

[ ] airbnb_df.shape

(58087, 28)

[ ] #airbnb_df.drop(columns=['first_review', 'last_review'], inplace=True)

[ ] airbnb_df.isnull().sum()

log_price                   0
property_type               0
room_type                   0
amenities                   0
accommodates                0
bathrooms                   0
bed_type                    0
cancellation_policy         0
cleaning_fee                0
city                        0
description                 0
first_review                0
host_has_profile_pic      145
host_identity_verified    145
host_response_rate        9838
host_since                145
instant_bookable            0
last_review                 0
latitude                    0
longitude                   0
name                        0
neighbourhood            5101
number_of_reviews           0
review_scores_rating      858
thumbnail_url            5887
zipcode                   691
bedrooms                   68
beds                       44
dtype: int64
```

1.3 Clean host_has_profile_pic

รูปโปรไฟล์ และเครื่องหมายยืนยันตัวตนของ Host

- ดูค่าทั้งหมดใน host_has_profile_pic

```
[ ] pd.unique(airbnb_df['host_has_profile_pic'])

array(['t', nan, 'f'], dtype=object)
```

- ดูค่าทั้งหมดใน host_has_identity_verified

```
[ ] pd.unique(airbnb_df['host_identity_verified'])

array(['t', 'f', nan], dtype=object)
```

- ทำการหาค่าที่ไม่ซ้ำกันในคอลัมน์ host_identity_verified สำหรับแถวที่มีค่า host_has_profile_pic เป็นค่า null

```
[ ] pd.unique(airbnb_df[airbnb_df['host_has_profile_pic'].isnull()]['host_identity_verified'])

array([nan], dtype=object)
```

- เติมค่า null ทั้งในคอลัมน์ host_has_profile_pic และ host_identity_verified เป็น 'f'

```
[ ] airbnb_df.fillna({'host_has_profile_pic': 'f', 'host_identity_verified': 'f'}, inplace=True)
```

```
airbnb_df.isnull().sum()

log_price                 0
property_type             0
room_type                 0
amenities                 0
accommodates              0
bathrooms                 0
bed_type                  0
cancellation_policy       0
cleaning_fee              0
city                      0
description               0
first_review              0
host_has_profile_pic      0
host_identity_verified    0
host_response_rate        9838
host_since                145
instant_bookable          0
last_review               0
latitude                  0
longitude                 0
name                      0
neighbourhood             5101
number_of_reviews         0
review_scores_rating      858
thumbnail_url             5887
zipcode                   691
bedrooms                  68
beds                      44
dtype: int64
```

1.4 Clean host_response_rate

อัตราการโต้ตอบของ host (การตอบกลับข้อความ, การบริการ)

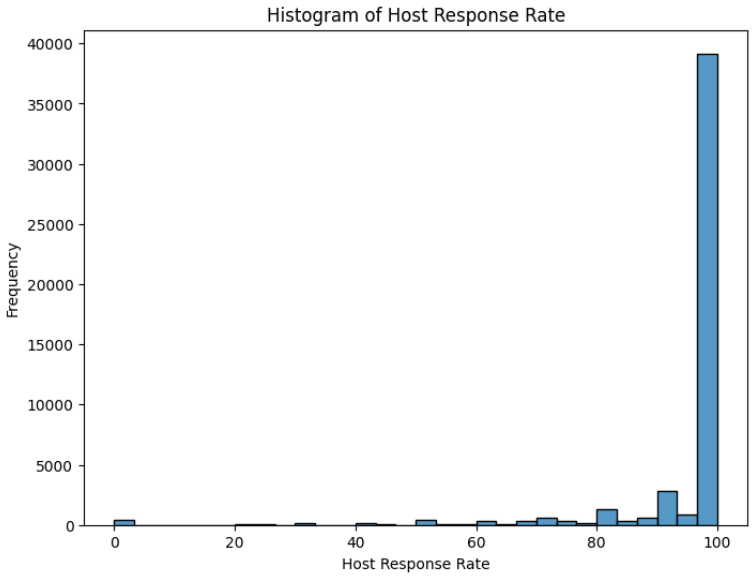
```
[ ] pd.unique(airbnb_df['host_response_rate'])

array([nan, '100%', '71%', '68%', '67%', '50%', '90%', '86%', '92%',
       '82%', '80%', '89%', '93%', '99%', '88%', '96%', '70%', '94%',
       '91%', '25%', '83%', '95%', '98%', '62%', '29%', '81%', '63%',
       '38%', '60%', '79%', '75%', '65%', '97%', '87%', '40%', '33%',
       '53%', '58%', '0%', '76%', '30%', '64%', '17%', '20%', '77%',
       '78%', '54%', '73%', '41%', '57%', '85%', '56%', '42%', '44%',
       '14%', '10%', '72%', '84%', '55%', '43%', '74%', '36%', '39%',
       '46%', '26%', '61%', '59%', '52%', '22%', '15%', '69%', '27%',
       '11%', '35%', '31%', '21%', '47%', '66%'], dtype=object)
```

- การลบเครื่องหมายเปอร์เซ็นต์ (%) ออกจากค่าทั้งหมดในคอลัมน์ host_response_rate

```
[ ]  airbnb_df['host_response_rate'] = airbnb_df['host_response_rate'].str.replace('%', '')

[ ]  pd.unique(airbnb_df['host_response_rate'])

array([nan, '100', '71', '68', '67', '50', '90', '86', '92', '82', '80',
       '89', '93', '99', '88', '96', '70', '94', '91', '25', '83', '95',
       '98', '62', '29', '81', '63', '38', '60', '79', '75', '65', '97',
       '87', '40', '33', '53', '58', '0', '76', '30', '64', '17', '20',
       '77', '78', '54', '73', '41', '57', '85', '56', '42', '44', '14',
       '10', '72', '84', '55', '43', '74', '36', '39', '46', '26', '61',
       '59', '52', '22', '15', '69', '27', '11', '35', '31', '21', '47',
       '66'], dtype=object)
```

- ทำการแปลงค่าทั้งหมดในคอลัมน์ host_response_rate ให้เป็นชนิดข้อมูลแบบ float และสร้าง histogram เพื่อดูการกระจายตัวของข้อมูล

```
[ ]  airbnb_df['host_response_rate'] = airbnb_df['host_response_rate'].astype('float')

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.histplot(data=airbnb_df, x='host_response_rate', bins=30, kde=False)
plt.title('Histogram of Host Response Rate')
plt.xlabel('Host Response Rate')
plt.ylabel('Frequency')
plt.show()
```



```
[ ] median_response = airbnb_df['host_response_rate'].median()
    median_response

100.0
```

- เปลี่ยนค่า null ในคอลัมน์ host_response_rate ทั้งหมดด้วยค่า median_response

```
[ ] airbnb_df.fillna({'host_response_rate': median_response}, inplace=True)

    airbnb_df.isnull().sum()

log_price                  0
property_type              0
room_type                  0
amenities                  0
accommodates               0
bathrooms                  0
bed_type                   0
cancellation_policy        0
cleaning_fee               0
city                       0
description                0
first_review               0
host_has_profile_pic       0
host_identity_verified     0
host_response_rate         0
host_since               145
instant_bookable           0
last_review                0
latitude                   0
longitude                  0
name                       0
neighbourhood           5101
number_of_reviews          0
review_scores_rating     858
thumbnail_url           5887
zipcode                  691
bedrooms                  68
beds                      44
dtype: int64
```

1.5 Clean host since

```
[ ] pd.unique(airbnb_df['host_since'])

array(['2012-03-26', '2017-06-19', '2016-10-25', ..., '2008-03-03',
       '2010-04-30', '2009-08-23'], dtype=object)

[ ] airbnb_df.dropna(subset='host_since', inplace=True)
```

1.6 Clean review_scores_rating

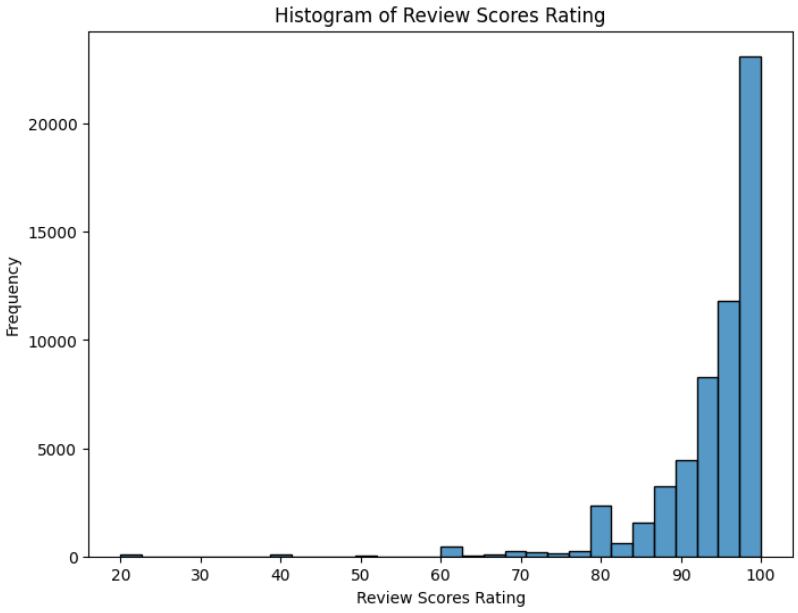
```
[ ]  pd.unique(airbnb_df['review_scores_rating'])
```

```
array([100.,  93.,  92.,  40.,  97.,  99.,  90.,  89.,  91.,  88.,  86.,
        72.,  98.,  95.,  96.,  nan,  84.,  80.,  94.,  87.,  85.,  70.,
        60.,  75.,  20.,  76.,  83.,  82.,  78.,  73.,  55.,  81.,  67.,
        71.,  77.,  79.,  47.,  74.,  68.,  66.,  63.,  50.,  53.,  65.,
        64.,  27.,  69.,  30.,  58.,  35.,  62.,  49.,  57.,  54.,  56.])
```

```
[ ]  pd.unique(airbnb_df[airbnb_df['review_scores_rating'].isnull()]['number_of_reviews'])
```

```
array([1, 2, 3, 5, 4, 7])
```

```
[ ]  # ดูการกระจายตัวของคอลัมน์ review_scores_rating
     plt.figure(figsize=(8, 6))
     sns.histplot(data=airbnb_df, x='review_scores_rating', bins=30, kde=False)
     plt.title('Histogram of Review Scores Rating')
     plt.xlabel('Review Scores Rating')
     plt.ylabel('Frequency')
     plt.show()
```



```
[ ] median_rating = airbnb_df['review_scores_rating'].median()
    median_rating

    96.0

[ ] # แทน missing value ด้วย median เพราะคอสัมน์นี้มีการการจายตัวที่เบี่ยงไปทางขวา
    airbnb_df.fillna({'review_scores_rating': median_rating}, inplace=True)

[ ] # ถ้าที่พักนั้นยังไม่เคยมีคนรีวิวให้ใส่ review_scores_rating เป็น 0
    airbnb_df.loc[airbnb_df['number_of_reviews'] == 0, 'review_scores_rating'] = 0
```

1.7 Clean neighborhood (fillna)

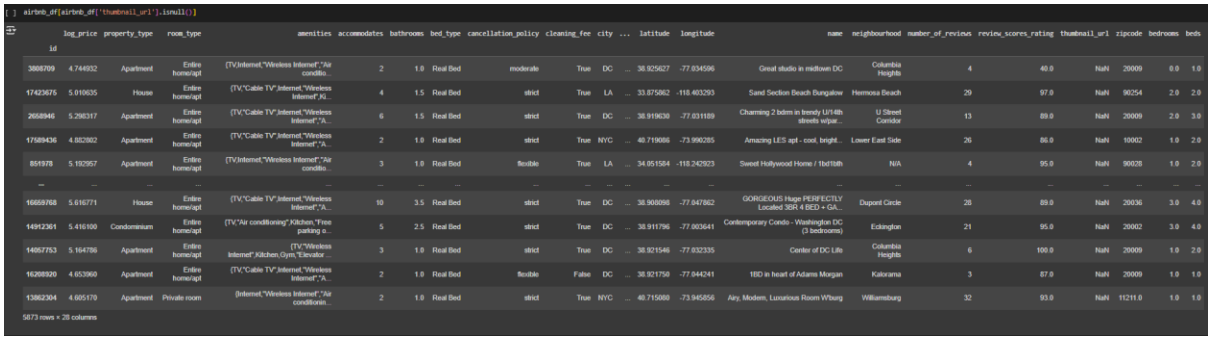
```
airbnb_df[airbnb_df['neighbourhood'].isnull()]
```

	id	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_policy	cleaning_fee	city	description	first_review	host_has_profile_pic	host_identity_verified	host_response_rate	host_since	instant_bookable
6	11825529	4.418841	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	3	1.0	Real Bed	moderate	True	LA	Warm and cozy studio with full kitchen and bat...	2017-03-10	t	f	100.0	2017-03-03	f
9	5385268	3.583519	House	Private room	{"Wireless Internet","Air conditioning",Kitche...	2	1.0	Real Bed	moderate	True	LA	Quiet community. Close to supermarkets,restaur...	2017-04-03	t	f	100.0	2017-03-12	f
14	583490	4.955827	Apartment	Entire home/apt	{Kitchen,Heating,"Smoke detector","Carbon mono...	2	1.0	Real Bed	strict	True	LA	This apartment is charming and wonderful. Beau...	2017-03-25	t	t	100.0	2012-11-20	f
26	19407360	4.553877	Apartment	Entire home/apt	{TV,"Wireless Internet",Pool,Kitchen,"Elevator...	2	1.0	Real Bed	flexible	True	LA	Retro, Melrose Place-esque studio apartment sl...	2016-07-17	t	t	100.0	2013-06-20	f
28	851978	5.192957	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	3	1.0	Real Bed	flexible	True	LA	This home is filled with love & laughter! Mode...	2015-12-30	t	t	100.0	2015-09-26	f
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
74098	8342838	4.553877	Apartment	Entire home/apt	{Internet,"Wireless Internet","Air conditionin...	4	1.0	Real Bed	flexible	True	LA	This bohemian vintage studio apartment located...	2015-02-28	t	t	100.0	2014-10-28	f
74099	4615631	4.276666	Condominium	Private room	{TV,Internet,"Wireless Internet","Air conditio...	2	1.0	Real Bed	moderate	True	LA	Charming Hollywood Luxury Condo! Enjoy Hollywo...	2016-07-25	t	t	100.0	2015-11-10	f
74101	18877717	4.584967	Apartment	Private room	{TV,Internet,"Wireless Internet",Pool,Kitchen,...	1	1.0	Real Bed	moderate	True	LA	A great comfortable bedroom available in the h...	2016-03-26	t	t	100.0	2014-04-01	f
74104	14934112	4.356709	Apartment	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet","A...	2	1.0	Real Bed	strict	True	Chicago	*Location location location!!! 1B,1BA availab...	2014-04-28	t	t	100.0	2014-03-01	f
74105	808802	4.248495	House	Private room	{TV,Internet,"Wireless Internet","Air conditio...	2	1.0	Real Bed	moderate	True	LA	Perfect for 1 person, will accept couples at a...	2016-03-25	t	t	75.0	2015-03-18	f

5099 rows × 29 columns

```
] # แทน missing value ด้วย N/A
  new_value = "N/A"
  airbnb_df['neighbourhood'].fillna(value=new_value, inplace=True)
```

1.8 Clean thumbnail_url (drop)



```
[ ] airbnb_df[airbnb_df['thumbnail_url'].isnull()]
```

id	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_policy	cleaning_fee	city	...	latitude	longitude	name	neighbourhood	number_of_reviews	review_scores_rating	thumbnail_url	zipcode	bedrooms	beds
3888709	4.744932	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	2	1.0	Real Bed	moderate	True	DC	...	38.925627	-77.034596	Great studio in midtown DC	Columbia Heights	4	40.0	NaN	20009	0.0	1.0
17423675	5.010635	House	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet",Ki...	4	1.5	Real Bed	strict	True	LA	...	33.875862	-118.403293	Sand Section Beach Bungalow	Hermosa Beach	29	97.0	NaN	90254	2.0	2.0
2658946	5.298317	Apartment	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet","A...	6	1.5	Real Bed	strict	True	DC	...	38.919630	-77.031189	Charming 2 bdrm in trendy U/14th streets w/par...	U Street Corridor	13	89.0	NaN	20009	2.0	3.0
17589436	4.882802	Apartment	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet","A...	2	1.0	Real Bed	strict	True	NYC	...	40.719086	-73.990285	Amazing LES apt - cool, bright...	Lower East Side	26	86.0	NaN	10002	1.0	2.0
851978	5.192957	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	3	1.0	Real Bed	flexible	True	LA	...	34.051584	-118.242923	Sweet Hollywood Home / 1bd1bth	N/A	4	95.0	NaN	90028	1.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16659768	5.616771	House	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet","A...	10	3.5	Real Bed	strict	True	DC	...	38.908898	-77.047862	GORGEOUS Huge PERFECTLY Located 3BR 4 BED + GA...	Dupont Circle	28	89.0	NaN	20036	3.0	4.0
14912361	5.416100	Condominium	Entire home/apt	{TV,"Air conditioning",Kitchen,"Free parking o...	5	2.5	Real Bed	strict	True	DC	...	38.911796	-77.003641	Contemporary Condo - Washington DC (3 bedrooms)	Eckington	21	95.0	NaN	20002	3.0	4.0
14057753	5.164786	Apartment	Entire home/apt	{TV,"Wireless Internet",Kitchen,Gym,"Elevator ...	3	1.0	Real Bed	strict	True	DC	...	38.921546	-77.032335	Center of DC Life	Columbia Heights	6	100.0	NaN	20009	1.0	2.0
16268920	4.653960	Apartment	Entire home/apt	{TV,"Cable TV",Internet,"Wireless Internet","A...	2	1.0	Real Bed	flexible	False	DC	...	38.921750	-77.044241	1BD in heart of Adams Morgan	Kalorama	3	87.0	NaN	20009	1.0	1.0
13862304	4.605170	Apartment	Private room	{Internet,"Wireless Internet","Air conditionin...	2	1.0	Real Bed	strict	True	NYC	...	40.715080	-73.945856	Airy, Modern, Luxurious Room Wburg	Williamsburg	32	93.0	NaN	11211.0	1.0	1.0

5873 rows × 28 columns

```
[ ]  # ดรอปคอลัมน์
     airbnb_df.drop(columns=['thumbnail_url'], inplace=True)
```

1.9 Clean zipcode (drop)

```
[ ]  # เช็คค่าในคอลัมน์ zipcode
     pd.unique(airbnb_df['zipcode'])
```

```
array(['11201', '10019', '10027', '20009', '94131', '90292', '90015',
       '94121', '91748', '10009.0', '90254', '90804', '60622', '02127',
       '10002', '11226.0', '91401', '11212.0', '11411.0', nan, '11374',
       '90028', '10016', '11225', '90291', '90026', '11212', '94118',
       '11211.0', '90057', '90046', '20037', '11237', '11233', '60608',
       '94127', '11249.0', '10037', '10011', '10011.0', '10032', '90027',
       '91601', '20001', '11226', '90013', '11217', '91016', '10014',
       '11101', '94110', '94134', '91501', '90230', '90039', '11221',
       '20019', '60647', '10025', '10013', '10040', '91604', '94117',
       '10039', '90036', '60614', '20002', '10029.0', '11222', '90042',
       '94114', '11238', '20007', '91106', '11206', '91208', '11223',
       '02114', '94103', '91107', '11355', '11220', '10003.0', '10024',
       '90004', '10012', '02130', '10023', '20012', '91205', '02132',
       '60611', '10473.0', '11249', '90019', '60639', '91423', '11238.0',
       '11232', '11377', '11216', '02129', '91803', '94115.0', '10128',
       '11207', '02118', '11213', '11215', '11229', '90064', '90024',
       '94117.0', '11216.0', '02111', '94107', '90265', '91331', '02119',
       '10002.0', '91384', '90405', '20010', '90717', '60601', '91505',
       '90807', '90068', '60654', '02120', '10035.0', '91606', '11218',
       '90012', '20018', '10036', '02135', '10022', '91607', '11378',
       '94102.0', '90018', '02113', '90404', '11692', '90403', '90014',
       '11230', '90302', '60610', '90212', '02108', ' ', '90025', '90038',
       '90048', '91411', '91040', '94112', '91325', '90802', '20011',
       '90731', '90069', '60660', '91202', '94102', '11385', '11205',
       '10010', '94133', '02109', '91104', '90007', '90029', '11237.0',
       '11225.0', '11213.0', '11206.0', '02110', '11105', '90272',
       '60642', '10282', '91741', '10001.0', '90065', '10034', '10030',
       '60605', '90016', '10475', '10031', '11231.0', '90010', '10017',
       '11102', '11361', '02124', '94109', '11103', '90250', '60661',
       '90277', '91745', '60641', '94124', '20003', '02116', '10455',
       '91306', '90505', '11428', '90045', '94115', '10033', '10044',
       '11370.0', '10021', '90732', '91405', '91765', '94116', '91711',
       '60625', '20015', '11211', '10028', '11354', '90803', '60657',
```

```
airbnb_df['zipcode'].tail()
```

```
id
14934112      60610
808802        90038
13281809      90254
18688039    11206.0
3534845       90802
Name: zipcode, dtype: object
```

```
airbnb_df.shape
```

```
(57942, 27)
```

```
# เขียนฟังก์ชันสำหรับทำให้ข้อมูลใน zipcode อยู่ใน format เดียวกัน
import re
import math
zip_code_pattern = r'^\d{5}(-\d{4})?$'


def fix_zipcode(zipcode):
    if pd.isna(zipcode):  # ถ้าเป็น NaN
        return zipcode
    zipcode = str(zipcode).strip()  # แปลง Zipcode ให้เป็น string และลบช่องว่าง
    if '.' in zipcode:
        zipcode = re.sub(r'.0$', '', zipcode)  # ลบ .0 ที่ท้ายออก
    if re.fullmatch(r'\d{5}', zipcode):  # ตรวจสอบว่าเป็นตัวเลข 5 หลัก
        return zipcode
    elif re.fullmatch(r'\d{5}-\d{4}', zipcode):  # ตรวจสอบว่าเป็นรูปแบบ Zip+4
        return zipcode
    else:
        return None  # หรือสามารถกำหนดค่าอื่น ๆ ที่เหมาะสมหากรูปแบบไม่ถูกต้อง

airbnb_df['zipcode'] = airbnb_df['zipcode'].apply(fix_zipcode)
```

```
pd.unique(airbnb_df['zipcode'])
```

```
array(['11201', '10019', '10027', '20009', '94131', '90292', '90015',
       '94121', '91748', '10009', '90254', '90804', '60622', '02127',
       '10002', '11226', '91401', '11212', '11411', nan, '11374', '90028',
       '10016', '11225', '90291', '90026', '94118', '11211', '90057',
       '90046', '20037', '11237', '11233', '60608', '94127', '11249',
       '10037', '10011', '10032', '90027', '91601', '20001', '90013',
       '11217', '91016', '10014', '11101', '94110', '94134', '91501',
       '90230', '90039', '11221', '20019', '60647', '10025', '10013',
       '10040', '91604', '94117', '10039', '90036', '60614', '20002',
       '10029', '11222', '90042', '94114', '11238', '20007', '91106',
       '11206', '91208', '11223', '02114', '94103', '91107', '11355',
       '11220', '10003', '10024', '90004', '10012', '02130', '10023',
       '20012', '91205', '02132', '60611', '10473', '90019', '60639',
       '91423', '11232', '11377', '11216', '02129', '91803', '94115',
       '10128', '11207', '02118', '11213', '11215', '11229', '90064',
       '90024', '02111', '94107', '90265', '91331', '02119', '91384',
       '90405', '20010', '90717', '60601', '91505', '90807', '90068',
       '60654', '02120', '10035', '91606', '11218', '90012', '20018',
       '10036', '02135', '10022', '91607', '11378', '94102', '90018',
       '02113', '90404', '11692', '90403', '90014', '11230', '90302',
       '60610', '90212', '02108', None, '90025', '90038', '90048',
       '91411', '91040', '94112', '91325', '90802', '20011', '90731',
       '90069', '60660', '91202', '11385', '11205', '10010', '94133',
       '02109', '91104', '90007', '90029', '02110', '11105', '90272',
       '60642', '10282', '91741', '10001', '90065', '10034', '10030',
       '60605', '90016', '10475', '10031', '11231', '90010', '10017',
       '11102', '11361', '02124', '94109', '11103', '90250', '60661',
       '90277', '91745', '60641', '94124', '20003', '02116', '10455',
       '91306', '90505', '11428', '90045', '10033', '10044', '11370',
       '10021', '90732', '91405', '91765', '94116', '91711', '60625',
```

```
[ ] airbnb_df['zipcode'].tail()

id
14934112    60610
808802      90038
13281809    90254
18688039    11206
3534845     90802
Name: zipcode, dtype: object

[ ] airbnb_df.shape

(57942, 27)

[ ] # เช็คว่ามี missing value กี่ row
    airbnb_df['zipcode'].isnull().sum()

695

[ ] airbnb_df.shape

(57942, 27)

[ ] review_scores_rating
    airbnb_df.dropna(subset='zipcode', inplace=True)

    airbnb_df['zipcode'].isnull().sum()

0
```

1.10 Clean beds & bedrooms

```
[ ] airbnb_df[airbnb_df['bedrooms'].isnull()].shape

(67, 27)

[ ] airbnb_df[airbnb_df['beds'].isnull()].shape

(41, 27)

[ ] # airbnb_df.loc[airbnb_df['bedrooms'].notnull(), 'beds'] = airbnb_df.loc[airbnb_df['bedrooms'].notnull(), '|

[ ] # Use .loc to avoid SettingWithCopyWarning and ensure the operation is applied to the dataframe
    airbnb_df.loc[:, ['bedrooms', 'beds']] = airbnb_df[['bedrooms', 'beds']].fillna(value=0)
```

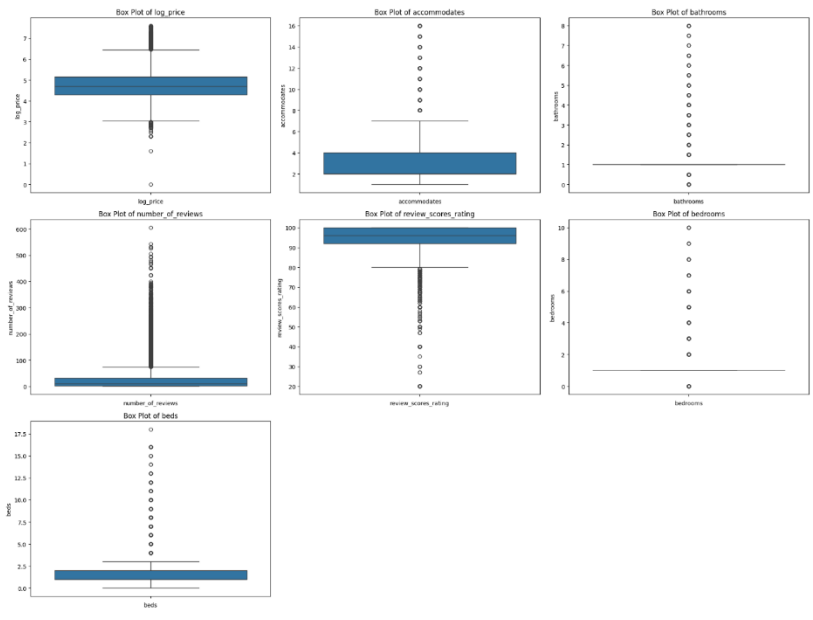
```
airbnb_df.isnull().sum()
```

```
log_price                0
property_type            0
room_type                0
amenities                0
accommodates             0
bathrooms                0
bed_type                 0
cancellation_policy      0
cleaning_fee             0
city                     0
description              0
first_review             0
host_has_profile_pic     0
host_identity_verified   0
host_response_rate       0
host_since               0
instant_bookable         0
last_review              0
latitude                 0
longitude                0
name                     0
neighbourhood            0
number_of_reviews        0
review_scores_rating     0
zipcode                  0
bedrooms                 0
beds                     0
dtype: int64
```

1.11 Detect outlier

```
[ ] numerical_columns = ['log_price', 'accommodates', 'bathrooms', 'number_of_reviews', 'review_scores_rating', 'bedrooms','beds']
```

```
plt.figure(figsize=(20, 15))

for i, column in enumerate(numerical_columns, 1):
    plt.subplot(3, 3, i)
    sns.boxplot(data=airbnb_df[column])
    plt.title(f'Box Plot of {column}')
    plt.xlabel(column)

plt.tight_layout()
plt.show()
```



```
[ ]  outlier_columns = ['log_price', 'accommodates', 'bathrooms', 'bedrooms','beds']
```

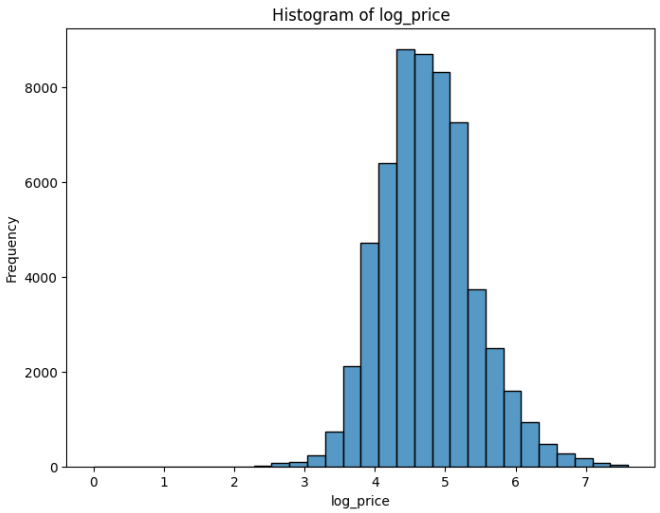
```
[ ]  # filter outlier ของ column ที่กำหนด
     import numpy as np

     outliers = {}
     for i in outlier_columns:
       Q1 = airbnb_df[i].quantile(0.25)
       Q3 = airbnb_df[i].quantile(0.75)
       IQR = Q3 - Q1
       lower_bound = Q1 - 1.5 * IQR
       upper_bound = Q3 + 1.5 * IQR
       outliers[i] = airbnb_df[(airbnb_df[i] < lower_bound) | (airbnb_df[i] > upper_bound)]
```

1.12 Outlier ของ log price

```
[ ]  outliers['log_price'].shape
```

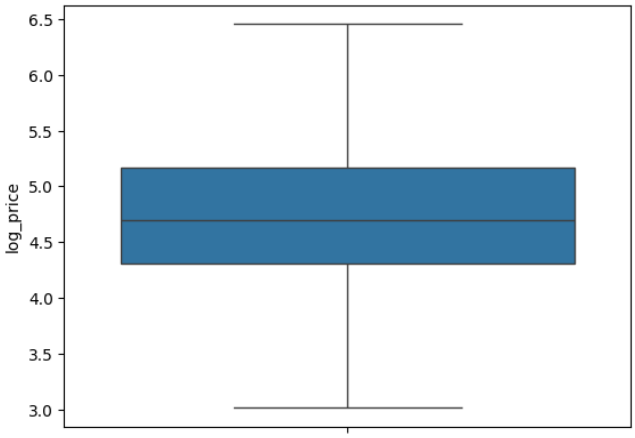
```
(940, 27)
```

```
# plot การกระจายตัวของ log_price
plt.figure(figsize=(8, 6))
sns.histplot(data=airbnb_df, x='log_price', bins=30, kde=False)
plt.title('Histogram of log_price')
plt.xlabel('log_price')
plt.ylabel('Frequency')
plt.show()
```



```
[ ]  # หา lower, upper bound
     Q1 = airbnb_df['log_price'].quantile(0.25)
     Q3 = airbnb_df['log_price'].quantile(0.75)
     IQR = Q3 - Q1
     lower_bound = Q1 - 1.5 * IQR
     upper_bound = Q3 + 1.5 * IQR

[ ]  # Capping outlier
     airbnb_df['log_price'] = np.where(airbnb_df['log_price'] > upper_bound, upper_bound, airbnb_df['log_price'])
     airbnb_df['log_price'] = np.where(airbnb_df['log_price'] < lower_bound, lower_bound, airbnb_df['log_price'])
```

```
sns.boxplot(airbnb_df['log_price'])
```



1.13 Dropping duplicate rows

```
[ ]  airbnb_df.drop_duplicates(inplace=True)

[ ]  airbnb_df.shape

     (57247, 27)

[ ]  airbnb_df.to_csv('AirbnbData.csv')
```

```
airbnb_df.isnull().sum()

log_price                 0
property_type             0
room_type                 0
amenities                 0
accommodates              0
bathrooms                 0
bed_type                  0
cancellation_policy       0
cleaning_fee              0
city                      0
description               0
first_review              0
host_has_profile_pic      0
host_identity_verified    0
host_response_rate        0
host_since                0
instant_bookable          0
last_review               0
latitude                  0
longitude                 0
name                      0
neighbourhood             0
number_of_reviews         0
review_scores_rating      0
zipcode                   0
bedrooms                  0
beds                      0
dtype: int64
```

2. EDA

```
airbnb_df.head()
```

	id	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_policy	cleaning_fee	city	description	first_review	host_has_profile_pic	host_identity_verified	host_respo
0	6901257	5.010635	Apartment	Entire home/apt	{"Wireless Internet","Air conditioning",Kitche...	3	1.0	Real Bed	strict	True	NYC	Beautiful, sunlit brownstone 1-bedroom in the ...	2016-06-18	t	t	
1	6304928	5.129899	Apartment	Entire home/apt	{"Wireless Internet","Air conditioning",Kitche...	7	1.0	Real Bed	strict	True	NYC	Enjoy travelling during your stay in Manhattan...	2017-08-05	t	f	
2	7919400	4.976734	Apartment	Entire home/apt	{TV,"Cable TV","Wireless Internet","Air condit...	5	1.0	Real Bed	moderate	True	NYC	The Oasis comes complete with a full backyard ...	2017-04-30	t	t	
4	3808709	4.744932	Apartment	Entire home/apt	{TV,Internet,"Wireless Internet","Air conditio...	2	1.0	Real Bed	moderate	True	DC	Cool, cozy, and comfortable studio located in ...	2015-05-12	t	t	
5	12422935	4.442651	Apartment	Private room	{TV,"Wireless Internet",Heating,"Smoke detecto...	2	1.0	Real Bed	strict	True	SF	Beautiful private room overlooking scenic view...	2017-08-27	t	t	

```
airbnb_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 57247 entries, 0 to 74110
Data columns (total 28 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   id                      57247 non-null  int64
 1   log_price               57247 non-null  float64
 2   property_type           57247 non-null  object
 3   room_type               57247 non-null  object
 4   amenities               57247 non-null  object
 5   accommodates            57247 non-null  int64
 6   bathrooms               57247 non-null  float64
 7   bed_type                57247 non-null  object
 8   cancellation_policy     57247 non-null  object
 9   cleaning_fee            57247 non-null  bool
 10  city                    57247 non-null  object
 11  description             57247 non-null  object
 12  first_review            57247 non-null  object
 13  host_has_profile_pic    57247 non-null  object
 14  host_identity_verified  57247 non-null  object
 15  host_response_rate      57247 non-null  float64
 16  host_since              57247 non-null  object
 17  instant_bookable        57247 non-null  object
 18  last_review             57247 non-null  object
 19  latitude                57247 non-null  float64
 20  longitude               57247 non-null  float64
 21  name                    57247 non-null  object
 22  neighbourhood           57247 non-null  object
 23  number_of_reviews       57247 non-null  int64
 24  review_scores_rating    57247 non-null  float64
 25  zipcode                 57247 non-null  object
 26  bedrooms                57247 non-null  float64
 27  beds                    57247 non-null  float64
dtypes: bool(1), float64(8), int64(3), object(16)
memory usage: 12.3+ MB
```

```
airbnb_df.describe(include='all')
```

	id	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_policy	cleaning_fee	city	description	first_review	host_has_profile_pic	host_identity_verified	host
count	5.724700e+04	57247.000000	57247	57247	57247	57247.000000	57247.000000	57247	57247	57247	57247	57247	57247	57247	57247	
unique	NaN	NaN	32	3	52625	NaN	NaN	5	5	2	6	56803	2533	2	2	
top	NaN	NaN	Apartment	Entire home/apt	{}	NaN	NaN	Real Bed	strict	True	NYC	Private room in the heart of Little Italy with...	2017-01-01	t	t	
freq	NaN	NaN	37697	32923	178	NaN	NaN	55664	27609	45284	24949	7	288	57152	41625	
mean	1.124600e+07	4.745517	NaN	NaN	NaN	3.216396	1.226623	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
std	6.083740e+06	0.654048	NaN	NaN	NaN	2.144833	0.562417	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
min	3.440000e+02	3.012984	NaN	NaN	NaN	1.000000	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
25%	6.237459e+06	4.304065	NaN	NaN	NaN	2.000000	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
50%	1.221197e+07	4.700480	NaN	NaN	NaN	2.000000	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
75%	1.639573e+07	5.164786	NaN	NaN	NaN	4.000000	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
max	2.123090e+07	6.455867	NaN	NaN	NaN	16.000000	8.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

2.1 Top 10 property types

```
# ตรวจสอบค่าว่างในคอลัมน์ PropertyType
print(airbnb_df['property_type'].isna().sum())
```

```
0
```

```
# นับจำนวนแต่ละประเภทของ PropertyType
property_counts = airbnb_df['property_type'].value_counts()
print(property_counts)
```

```
property_type
Apartment              37697
House                  12982
Condominium             2010
Townhouse               1285
Loft                    1001
Other                    423
Guesthouse               406
Bed & Breakfast          352
Bungalow                 305
Villa                    128
Dorm                     105
Guest suite               98
Camper/RV                 67
In-law                    63
Cabin                     62
Hostel                    55
Boutique hotel            47
Boat                      44
Timeshare                 33
Serviced apartment        16
Tent                      14
Castle                    13
Hut                        7
Treehouse                  6
Vacation home              6
Yurt                       6
Chalet                     5
Tipi                       3
Earth House                3
Train                      2
Cave                       2
Island                     1
Name: count, dtype: int64
```

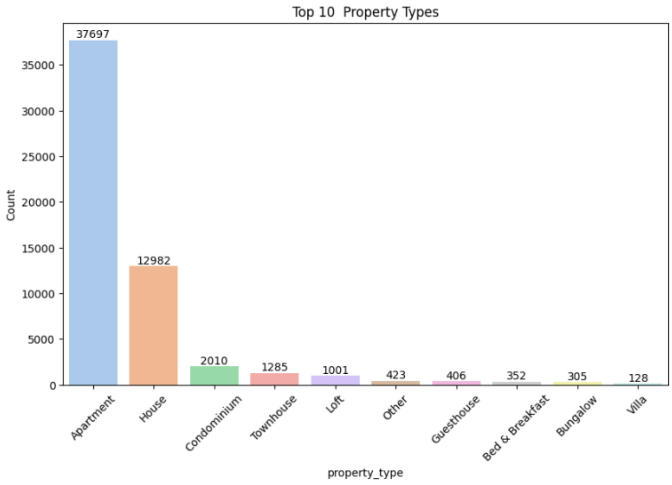
```
#นับจำนวน Type ทั้งหมด ที่มีใน property_type
num_property_types = airbnb_df['property_type'].nunique()
print(f"Number of unique property types: {num_property_types}")
```

```
Number of unique property types: 32
```

```
# เลือกประเภททรัพย์สิน 10 อันดับแรกที่ Airbnb มีมากสุด
top_10_property_types = property_counts.head(10)
print(top_10_property_types)
```

```
property_type
Apartment          37697
House              12982
Condominium         2010
Townhouse           1285
Loft                1001
Other                423
Guesthouse           406
Bed & Breakfast      352
Bungalow             305
Villa                128
Name: count, dtype: int64
```

```
plt.figure(figsize=(10,6))
property_type_count = airbnb_df['property_type'].value_counts().sort_values(ascending=False).head(10)
plt.ylabel('Count')
sns.barplot(x = property_type_count.index , y = property_type_count.values, palette = 'pastel')
for index, value in enumerate(property_type_count.values):
    plt.text(index, value, str(value), ha='center', va='bottom')
plt.xticks(rotation = 45)
plt.title('Top 10  Property Types')
plt.show()
```



ผลสรุป Top 10 property types

1. Apartment 37697
2. House 12982
3. Condominium 2010

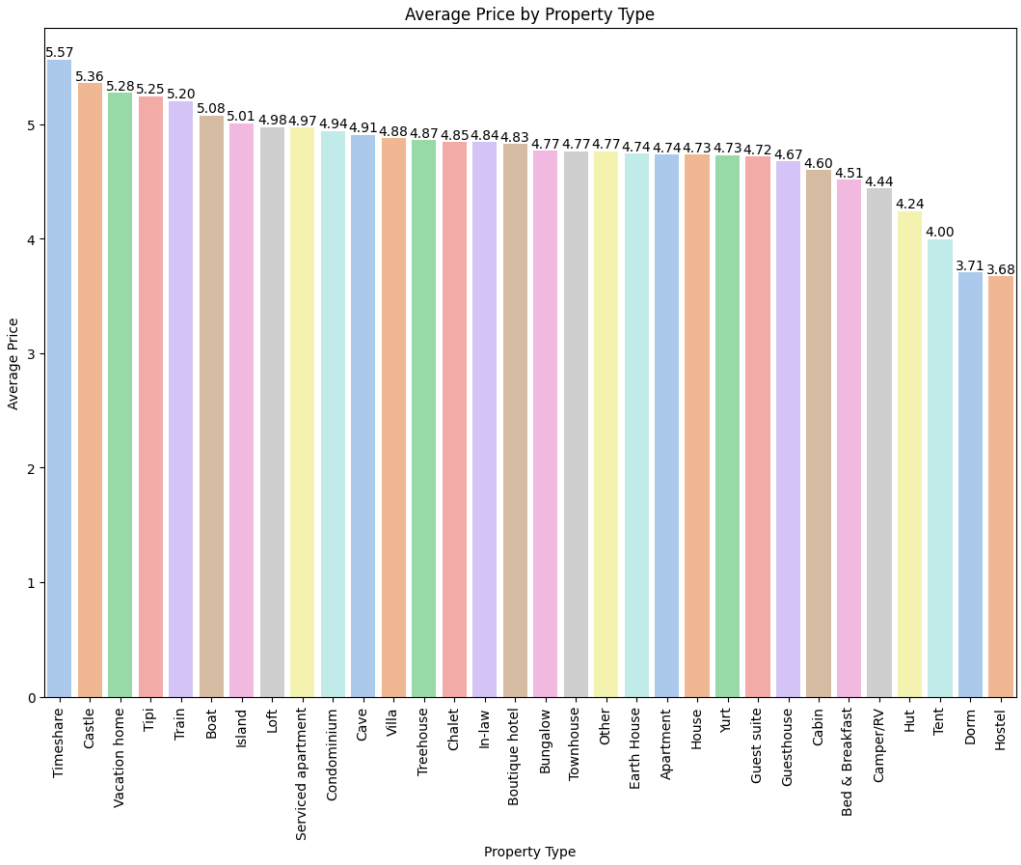
4. Townhouse 1285
5. Loft 1001
6. Other 423
7. Guesthouse 406
8. Bed & Breakfast 352
9. Bungalow 305
10. Villa 128

2.2 Average price per property type

```
#คำนวณราคาเฉลี่ยของแต่ละ PropertyType
avg_price_per_type = airbnb_df.groupby('property_type')['log_price'].mean().sort_values(ascending=False)
print(avg_price_per_type)
```

```
property_type
Timeshare             5.567173
Castle                5.357347
Vacation home         5.278115
Tipi                  5.246118
Train                 5.204399
Boat                  5.078666
Island                5.010635
Loft                  4.975895
Serviced apartment    4.974531
Condominium           4.943027
Cave                  4.909373
Villa                 4.879173
Treehouse             4.865014
Chalet                4.848429
In-law                4.842913
Boutique hotel        4.828464
Bungalow              4.767488
Townhouse             4.766615
Other                 4.765761
Earth House           4.744603
Apartment             4.738255
House                 4.733221
Yurt                  4.731928
Guest suite           4.717421
Guesthouse            4.674414
Cabin                 4.600159
Bed & Breakfast       4.513852
Camper/RV             4.440293
Hut                   4.242541
Tent                  3.997118
Dorm                  3.707582
Hostel                3.675375
Name: log_price, dtype: float64
```

```python
# สร้างกราฟแท่งแสดงราคาเฉลี่ยของแต่ละ PropertyType
plt.figure(figsize=(13, 9))
sns.barplot(x=avg_price_per_type.index, y=avg_price_per_type.values, palette='pastel')

# เพิ่มจำนวนราคาเฉลี่ยบนกราฟแท่ง
for index, value in enumerate(avg_price_per_type.values):
    plt.text(index, value, f'{value:,.2f}', ha='center', va='bottom')

# ตั้งค่า title และ labels
plt.xticks(rotation=90)
plt.title('Average Price by Property Type')
plt.xlabel('Property Type')
plt.ylabel('Average Price')

# แสดงกราฟ
plt.show()
```



ผลสรุปราคาที่อยู่ระหว่าง 3.675375 – 5.567173 โดยเรียงลำดับดังนี้

1. Timeshare 5.567173
2. Castle 5.357347
3. Vacation home 5.278115
4. Tipi 5.246118
5. Train 5.204399

6. Boat 5.078666
7. Island 5.010635
8. Loft 4.975895
9. Serviced apartment 4.974531
10. Condominium 4.943027
11. Cave 4.909373
12. Villa 4.879173
13. Treehouse 4.865014
14. Chalet 4.848429
15. In-law 4.842913
16. Boutique hotel 4.828464
17. Bungalow 4.767488
18. Townhouse 4.766615
19. Other 4.765761
20. Earth House 4.744603
21. Apartment 4.738255
22. House 4.733221
23. Yurt 4.731928
24. Guest suite 4.717421
25. Guesthouse 4.674414
26. Cabin 4.600159
27. Bed & Breakfast 4.513852
28. Camper/RV 4.440293
29. Hut 4.242541
30. Tent 3.997118
31. Dorm 3.707582
32. Hostel 3.675375

2.3 Most city in AirBNB

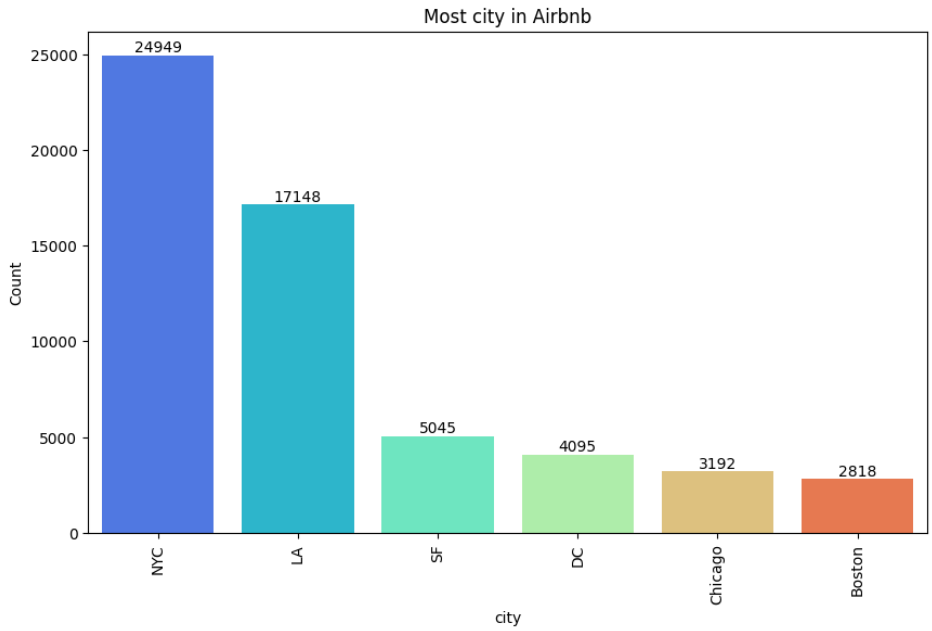
```
# ตรวจสอบค่าว่างในคอลัมน์ City
print(airbnb_df['city'].isna().sum())
```

```
0
```

```
# นับจำนวนแต่ละประเภทของ city
city_counts = airbnb_df['city'].value_counts()
print(city_counts)
```

```
city
NYC        24949
LA         17148
SF          5045
DC          4095
Chicago     3192
Boston      2818
Name: count, dtype: int64
```

```
plt.figure(figsize=(10,6))
property_type_count = airbnb_df['city'].value_counts().sort_values(ascending=False).head(6)
plt.ylabel('Count')
sns.barplot(x = property_type_count.index , y = city_counts.values, palette = 'rainbow')
for index, value in enumerate(city_counts.values):
    plt.text(index, value, str(value), ha='center', va='bottom')
plt.xticks(rotation = 90)
plt.title('Most city in Airbnb')
plt.show()
```



ผลสรุปว่าเมืองไหนที่ลงทะเบียนเยอะสุดใน AirBNB สรุปได้ดังนี้

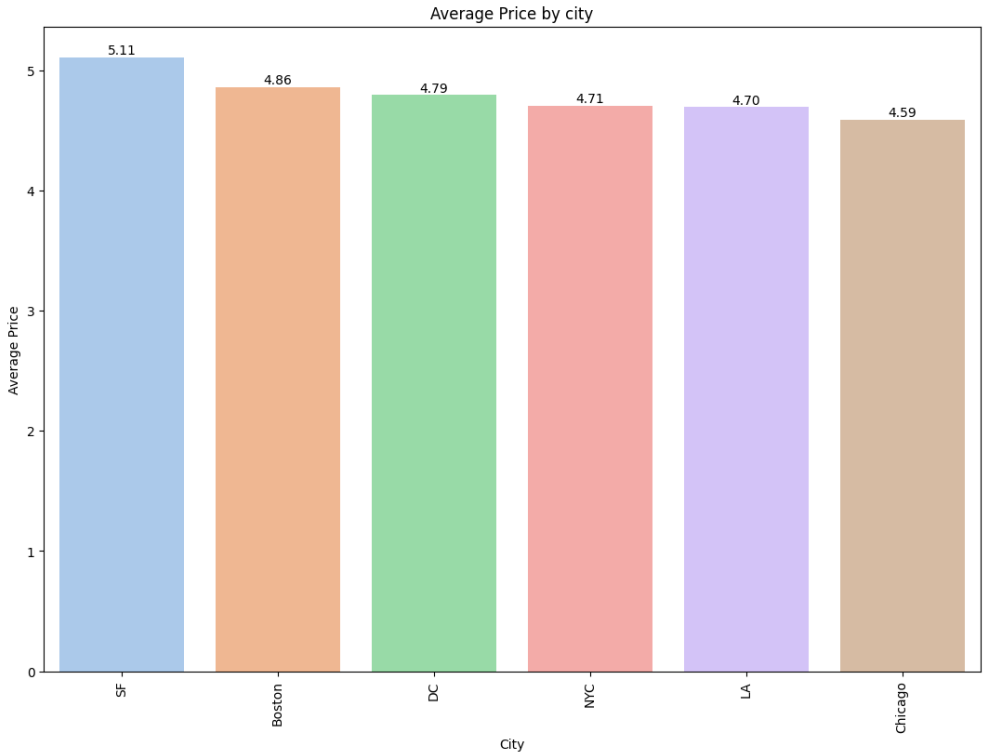
1. NYC 24949
2. LA 17148
3. SF 5045
4. DC 4095
5. Chicago 3192
6. Boston 2818

2.4 Average price per city

```
#คำนวณราคาเฉลี่ยของแต่ละ city
avg_price_per_city = airbnb_df.groupby('city')['log_price'].mean().sort_values(ascending=False)
print(avg_price_per_city)
```

```
city
SF         5.107322
Boston     4.860853
DC         4.794949
NYC        4.705524
LA         4.695440
Chicago    4.590058
Name: log_price, dtype: float64
```

```
# สร้างกราฟแท่งแสดงราคาเฉลี่ยของแต่ละ City
plt.figure(figsize=(13, 9))
sns.barplot(x=avg_price_per_city.index, y=avg_price_per_city.values, palette='pastel')

# เพิ่มจำนวนราคาเฉลี่ยบนกราฟแท่ง
for index, value in enumerate(avg_price_per_city.values):
    plt.text(index, value, f'{value:,.2f}', ha='center', va='bottom')

# ตั้งค่า title และ labels
plt.xticks(rotation=90)
plt.title('Average Price by city')
plt.xlabel('City')
plt.ylabel('Average Price')

# แสดงกราฟ
plt.show()
```



ผลสรุปว่าค่าเฉลี่ยของแต่ละ City ราคาอยู่ที่ 4.59-5.12

1. SF 5.107322
2. Boston 4.860853
3. DC 4.794949
4. NYC 4.705524
5. LA 4.695440
6. Chicago 4.590058

2.5 Top room types

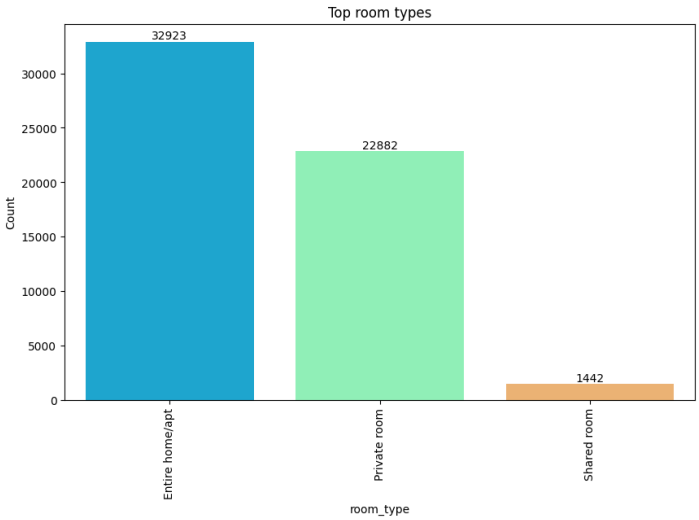
```
# ตรวจสอบค่าว่างในคอลัมน์ room_type
print(airbnb_df['room_type'].isna().sum())
```

```
0
```

```python
# นับจำนวนแต่ละประเภทของ room_type
room_type_counts = airbnb_df['room_type'].value_counts()
print(room_type_counts)
```

```
room_type
Entire home/apt    32923
Private room       22882
Shared room         1442
Name: count, dtype: int64
```

```python
plt.figure(figsize=(10,6))
property_type_count =airbnb_df['room_type'].value_counts().sort_values(ascending=False).head(6)
plt.ylabel('Count')
sns.barplot(x = property_type_count.index , y =room_type_counts.values, palette = 'rainbow')
for index, value in enumerate(room_type_counts.values):
    plt.text(index, value, str(value), ha='center', va='bottom')
plt.xticks(rotation = 90)
plt.title('Top room types')
plt.show()
```



ผลสรุปว่าประเภทของห้องนอนว่ามีห้องนอนแบบไหน จำนวนเท่าไหร่ โดยเรียงลำดับจากมากไปน้อย

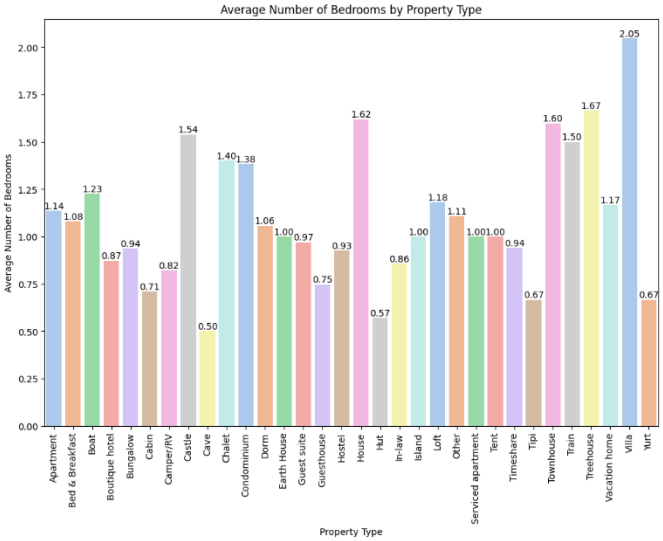
1. Entire home/apt 32923
2. Private room 22882
3. Shared room 1442

2.6 Property type VS Bedrooms

```python
# คำนวณค่าเฉลี่ยของจำนวนห้องนอนในแต่ละประเภทของทรัพย์สิน
avg_bedrooms = airbnb_df.groupby('property_type')['bedrooms'].mean().reset_index()
print(avg_bedrooms)
```

```
          property_type  bedrooms
0             Apartment  1.135104
1       Bed & Breakfast  1.079545
2                  Boat  1.227273
3        Boutique hotel  0.872340
4              Bungalow  0.937705
5                 Cabin  0.709677
6             Camper/RV  0.820896
7                Castle  1.538462
8                  Cave  0.500000
9               Chalet  1.400000
10         Condominium  1.383582
11                Dorm  1.057143
12         Earth House  1.000000
13         Guest suite  0.969388
14          Guesthouse  0.746305
15              Hostel  0.927273
16               House  1.618857
17                 Hut  0.571429
18              In-law  0.857143
19              Island  1.000000
20                Loft  1.180819
21               Other  1.108747
22  Serviced apartment  1.000000
23                Tent  1.000000
24           Timeshare  0.939394
25                Tipi  0.666667
26           Townhouse  1.597665
27               Train  1.500000
28           Treehouse  1.666667
29      Vacation home  1.166667
30               Villa  2.046875
31                Yurt  0.666667
```

```python
# สร้างกราฟแสดงค่าเฉลี่ยของจำนวนห้องนอนในแต่ละประเภทของ Property
plt.figure(figsize=(12, 8))
ax = sns.barplot(x='property_type', y='bedrooms', data=avg_bedrooms, palette='pastel')

# เพิ่มจำนวนเฉลี่ยบนกราฟแท่ง
for index, row in avg_bedrooms.iterrows():
    ax.text(index, row['bedrooms'], f'{row["bedrooms"]:.2f}', ha='center', va='bottom')

# ตั้งค่า title และ labels
plt.xticks(rotation=90)
plt.title('Average Number of Bedrooms by Property Type')
plt.xlabel('Property Type')
plt.ylabel('Average Number of Bedrooms')

# แสดงกราฟ
plt.show()
```



ผลสรุปว่าประเภทของ Property ว่ามีขนาดโดยเฉลี่ยเป็นเท่าไหร่ โดยสรุปได้ดังนี้

1. Apartment 1.135104
2. Bed & Breakfast 1.079545
3. Boat 1.227273
4. Boutique hotel 0.872340
5. Bungalow 0.937705
6. Cabin 0.709677
7. Camper/RV 0.820896
8. Castle 1.538462
9. Cave 0.500000
10. Chalet 1.400000
11. Condominium 1.383582
12. Dorm 1.057143
13. Earth House 1.000000
14. Guest suite 0.969388
15. Guesthouse 0.746305
16. Hostel 0.927273
17. House 1.618857
18. Hut 0.571429
19. In-law 0.857143
20. Island 1.000000
21. Loft 1.180819
22. Other 1.108747
23. Serviced apartment 1.000000
24. Tent 1.000000
25. Timeshare 0.939394
26. Tipi 0.666667
27. Townhouse 1.597665
28. Train 1.500000
29. Treehouse 1.666667

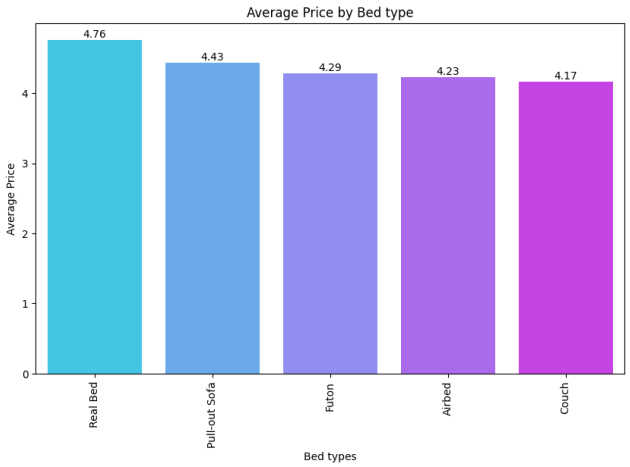
30. Vacation home 1.166667
31. Villa 2.046875
32. Yurt 0.666667

2.7 Bed type VS Log_price

```
#หาค่าเฉลี่ยราคาห้อง
avg_price_per_bed = airbnb_df.groupby('bed_type').log_price.mean().sort_values(ascending = False)
print(avg_price_per_bed)
```

```
bed_type
Real Bed         4.757994
Pull-out Sofa    4.432184
Futon            4.285674
Airbed           4.232398
Couch            4.166315
Name: log_price, dtype: float64
```

```
#สร้างกราฟ
plt.figure(figsize=(10,6))
sns.barplot(x = avg_price_per_bed.index , y = avg_price_per_bed.values, palette = 'cool')

# เพิ่มจำนวนราคาเฉลี่ยบนกราฟแท่ง
for index, value in enumerate(avg_price_per_bed.values):
    plt.text(index, value, f'{value:,.2f}', ha='center', va='bottom')

# ตั้งค่า title และ labels
plt.xticks(rotation=90)
plt.title('Average Price by Bed type')
plt.xlabel('Bed types')
plt.ylabel('Average Price')
```



ผลสรุปว่าราคาเฉลี่ยของ Bed type สรุปได้ดังนี้

1. Real Bed 4.757994
2. Pull-out Sofa 4.432184
3. Futon 4.285674

4. Airbed 4.232398
5. Couch 4.166315

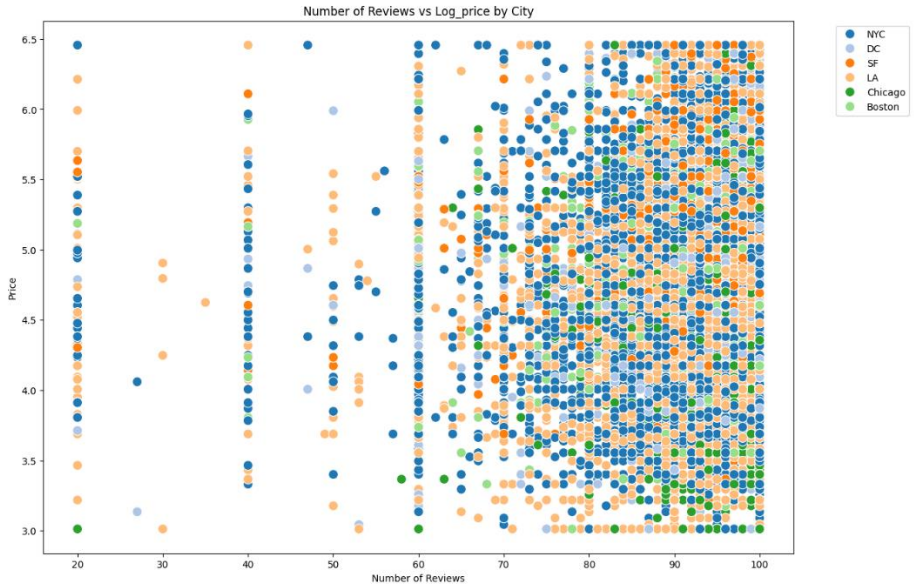
2.8 Number of reviews VS price

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='review_scores_rating', y='log_price', data=airbnb_df, color='skyblue', edgecolor='black')
plt.title('Scatter Plot of review_scores_rating vs Review Scores Rating')


# ตั้งค่า title และ labels
plt.title('Number of Reviews vs Price')
plt.xlabel('Number of Reviews')
plt.ylabel('Price')

plt.grid(True)  # Add grid lines
plt.show()
```

ผลสรุปว่าคะแนนรีวิวต่อราคาไม่ว่าราคาจะเท่าไหร่แต่ส่วนมากจะรีวิวไปในเรต 80-100

2.9 Number of reviews VS City

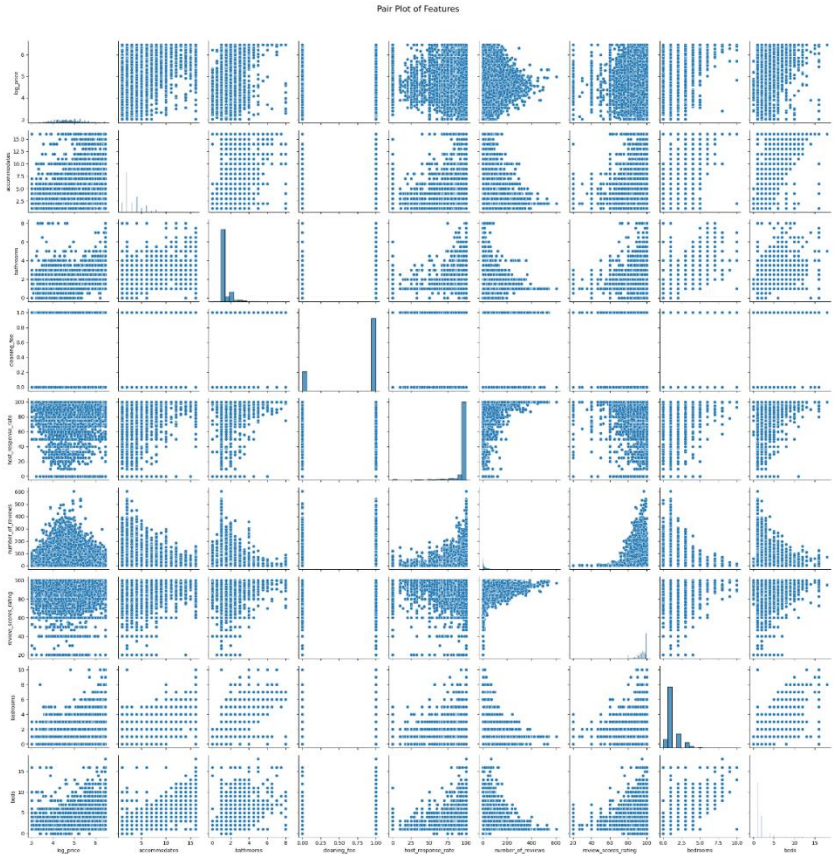
```
# สร้างกราฟกระจายเพื่อแสดงความสัมพันธ์ระหว่างจำนวนรีวิว, ราคา และเมือง
plt.figure(figsize=(14, 10))
sns.scatterplot(x='review_scores_rating', y='log_price', hue='city', data=airbnb_df, palette='tab20', s=100)

# ตั้งค่า title และ labels
plt.title('Number of Reviews vs Log_price by City')
plt.xlabel('Number of Reviews')
plt.ylabel('Price')

# แสดงกราฟ
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



ผลสรุปว่าคะแนนรีวิวต่อราคาและเมือง ไม่ว่าราคาเท่าไหร่หรือเมืองอะไรแต่ส่วนมากจะรีวิวไปในเรต 80-100

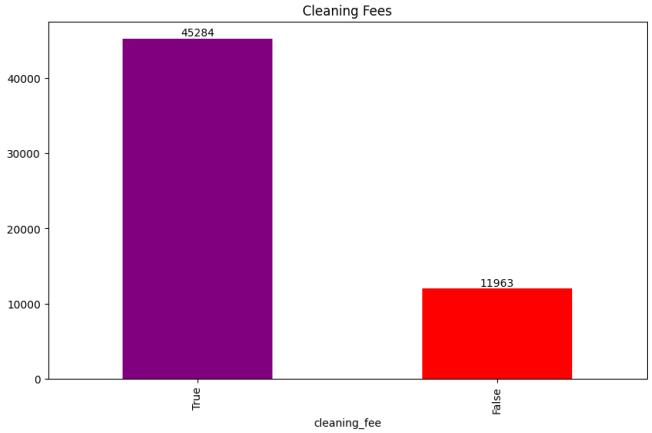
2.10 Pair plot of each column

```
# Columns to exclude
columns_to_exclude = ['id', 'latitude', 'longitude']

# Select columns to include in the pair plot
columns_to_include = [col for col in airbnb_df.columns if col not in columns_to_exclude]

# Create the pair plot
sns.pairplot(airbnb_df[columns_to_include], kind='scatter')

# Set the title
plt.suptitle('Pair Plot of Features', y=1.02, fontsize=16)
plt.tight_layout()
plt.show()
```



2.11 Number of cleaning fee

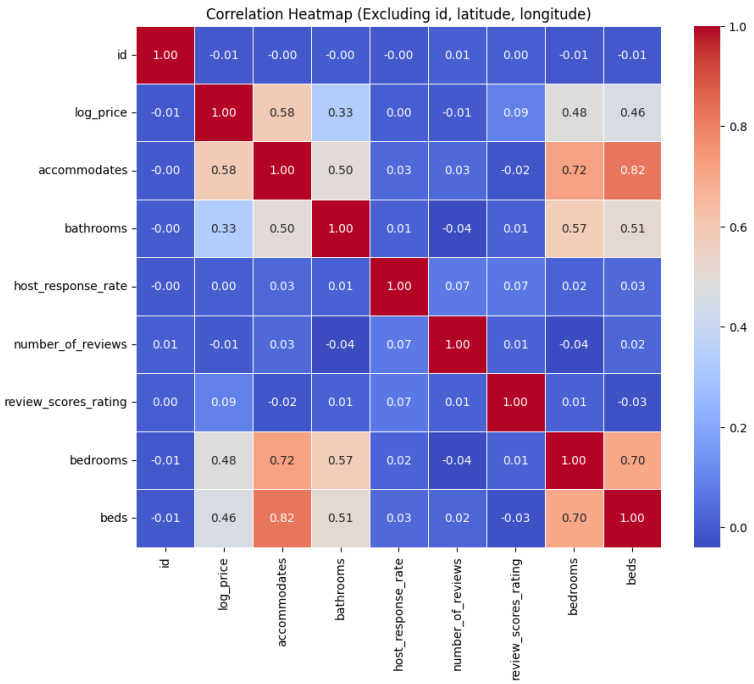
```
# สร้างกราฟแท่งเพื่อแสดงค่าจำนวนของ cleaning_fee
plt.figure(figsize=(10, 6))
cleaning_fee_counts = airbnb_df['cleaning_fee'].value_counts()
ax = cleaning_fee_counts.plot(kind='bar', color=['purple', 'red'])

# เพิ่มจำนวนนับบนกราฟแท่ง
for p in ax.patches:
    ax.text(p.get_x() + p.get_width() / 2, p.get_height(), str(int(p.get_height())), ha='center', va='bottom')

# ตั้งค่า title
plt.title('Cleaning Fees')

# แสดงกราฟ
plt.show()
```



ผลสรุปว่าที่พักส่วนมากใน AirBNB 79.1028350831% มีการเก็บ Cleaning Fee

2.12 Heatmap

```
# Drop 'id', 'latitude', and 'longitude' columns
airbnb_df_filtered = airbnb_df.drop(columns=['latitude', 'longitude'])

# Create heatmap
plt.figure(figsize=(10, 8))
num_cols_filtered = airbnb_df_filtered.select_dtypes(np.number)
heatmap_filtered = sns.heatmap(num_cols_filtered.corr(), annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
heatmap_filtered.set_title('Correlation Heatmap (Excluding id, latitude, longitude)')
plt.show()
```



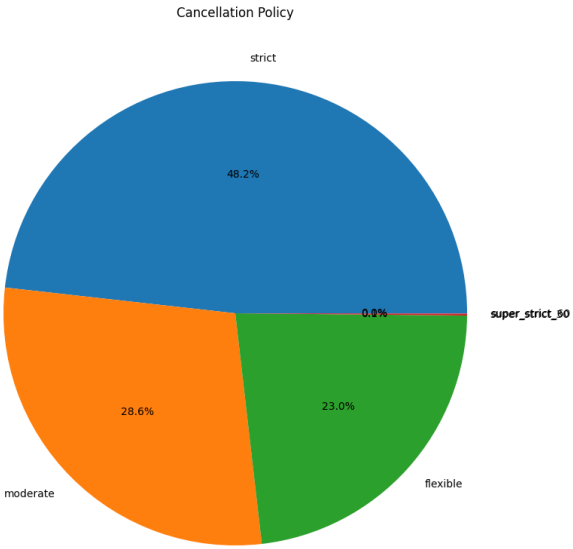
ผลสรุปว่าจากกราฟ สีที่เข้มและค่าที่สูง ใกล้ 1 หรือ -1 แสดงถึงความสัมพันธ์ที่แข็งแรงระหว่างตัวแปรสองตัว สีที่อ่อนและค่าที่ใกล้ 0 แสดงถึงความสัมพันธ์ที่อ่อนหรือไม่มีความสัมพันธ์ระหว่างตัวแปรสองตัว

2.13 Percentile of each cancellation policy

```
[ ]  policy_counts = airbnb_df['cancellation_policy'].value_counts()
     for policy, count in policy_counts.items():
         print(f"นโยบาย: {policy}, จำนวน: {count} ({count / len(airbnb_df) * 100:.1f}%)")
```

```
นโยบาย: strict, จำนวน: 27609 (48.2%)
นโยบาย: moderate, จำนวน: 16372 (28.6%)
นโยบาย: flexible, จำนวน: 13176 (23.0%)
นโยบาย: super_strict_30, จำนวน: 81 (0.1%)
นโยบาย: super_strict_60, จำนวน: 9 (0.0%)
```

```
# สร้างกราฟวงกลมเพื่อแสดงนโยบาย
plt.figure(figsize=(10, 10))  # ปรับขนาดกราฟ
airbnb_df['cancellation_policy'].value_counts().plot(kind='pie', autopct = "%1.1f%%")

# ตั้งค่า title และลบ ylabel
plt.ylabel("")
plt.title("Cancellation Policy")
plt.show()
```



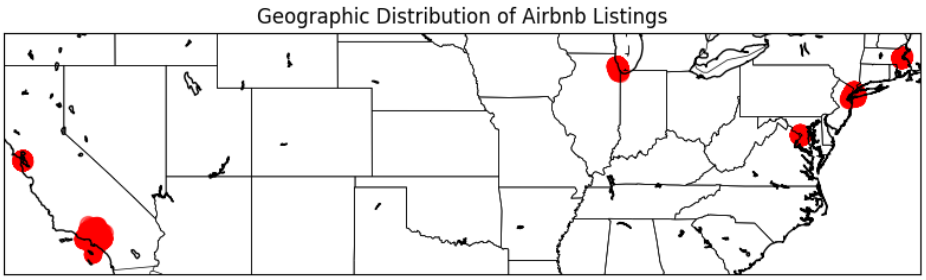
วิเคราะห์ที่พักใน Airbnb ส่วนใหญ่ Cancellation Policy เป็นแบบไหน ได้ผลสรุปดังนี้

1. นโยบาย: strict, จำนวน: 27609 (48.2%)
2. นโยบาย: moderate, จำนวน: 16372 (28.6%)
3. นโยบาย: flexible, จำนวน: 13176 (23.0%)
4. นโยบาย: super_strict_30, จำนวน: 81 (0.1%)
5. นโยบาย: super_strict_60, จำนวน: 9 (0.0%)

2.14 Location

```
pip install matplotlib basemap

Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: basemap in /usr/local/lib/python3.10/dist-packages (1.4.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: basemap-data<1.4,>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from basemap) (1.3.2)
Requirement already satisfied: pyshp<2.4,>=1.2 in /usr/local/lib/python3.10/dist-packages (from basemap) (2.3.1)
Requirement already satisfied: pyproj<3.7.0,>=1.9.3 in /usr/local/lib/python3.10/dist-packages (from basemap) (3.6.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from pyproj<3.7.0,>=1.9.3->basemap) (2024.2.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

```
from mpl_toolkits.basemap import Basemap

plt.figure(figsize=(10, 7))
m = Basemap(projection='merc', llcrnrlat=airbnb_df['latitude'].min() - 1, urcrnrlat=airbnb_df['latitude'].max() + 1,
            llcrnrlon=airbnb_df['longitude'].min() - 1, urcrnrlon=airbnb_df['longitude'].max() + 1, resolution='i')

m.drawcoastlines()
m.drawcountries()
m.drawstates()

x, y = m(airbnb_df['longitude'].values, airbnb_df['latitude'].values)
m.scatter(x, y, s=100, color='red', marker='o', alpha=0.5)

plt.title('Geographic Distribution of Airbnb Listings')
plt.show()
```



จากแผนภาพจะเห็นได้ว่าจะกระจุกตัวอยู่ 6 เมือง ตามลำดับความมากน้อยดังต่อไปนี้

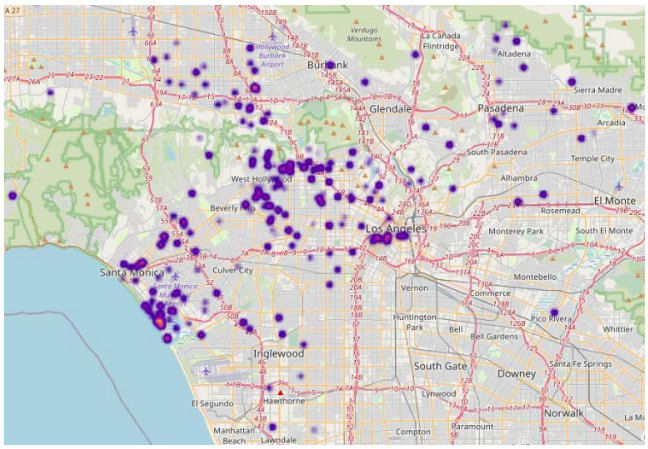
1. NYC 24949
2. LA 17148
3. SF 5045
4. DC 4095
5. Chicago 3192
6. Boston 2818

2.15 Density of AirBNB accommodations

```
! pip install plotly

Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (5.15.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly) (8.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly) (23.2)
```

```python
import plotly.express as px
fig = px.density_mapbox(airbnb_df.head(1000), lat = 'latitude', lon = 'longitude', z = 'id', radius = 8,
 center = dict(lat =34.0545 , lon = -118.3031),
 zoom = 10,
 mapbox_style = 'open-street-map',
 width=1100,
 height=800,
 title="Heatmap of Crime Locations"
 )
fig.show()
```

```python
airbnb_df.columns
```



แผนภาพแสดงตำแหน่งและความหนาแน่นของโรงแรม

3. Modeling

Prepare Data

```python
spare = airbnb_df.copy()
```

```python
airbnb_df.describe(include='all')
```

	log_price	property_type	room_type	amenities	accommodates	bathrooms	bed_type	cancellation_
count	57247.000000	57247	57247	57247	57247.000000	57247.000000	57247	
unique	NaN	32	3	52625	NaN	NaN	5	
top	NaN	Apartment	Entire home/apt	{}	NaN	NaN	Real Bed	
freq	NaN	37697	32923	178	NaN	NaN	55664	
mean	4.745517	NaN	NaN	NaN	3.216396	1.226623	NaN	
std	0.654048	NaN	NaN	NaN	2.144833	0.562417	NaN	
min	3.012984	NaN	NaN	NaN	1.000000	0.000000	NaN	
25%	4.304065	NaN	NaN	NaN	2.000000	1.000000	NaN	
50%	4.700480	NaN	NaN	NaN	2.000000	1.000000	NaN	
75%	5.164786	NaN	NaN	NaN	4.000000	1.000000	NaN	
max	6.455867	NaN	NaN	NaN	16.000000	8.000000	NaN	

```
airbnb_df[['room_type','bed_type','cancellation_policy','city','property_type']]
```

	room_type	bed_type	cancellation_policy	city	property_type
0	Entire home/apt	Real Bed	strict	NYC	Apartment
1	Entire home/apt	Real Bed	strict	NYC	Apartment
2	Entire home/apt	Real Bed	moderate	NYC	Apartment
4	Entire home/apt	Real Bed	moderate	DC	Apartment
5	Private room	Real Bed	strict	SF	Apartment
...	...	...	...	...	...
74104	Entire home/apt	Real Bed	strict	Chicago	Apartment
74105	Private room	Real Bed	moderate	LA	House
74107	Entire home/apt	Real Bed	moderate	LA	Apartment
74108	Entire home/apt	Real Bed	moderate	NYC	Apartment
74110	Entire home/apt	Real Bed	moderate	LA	Boat

57247 rows × 5 columns

```
cols = ['room_type','bed_type','cancellation_policy','city','property_type']
df = pd.get_dummies(airbnb_df, columns=cols, drop_first=True)

df['host_identity_verified'] = df['host_identity_verified'].replace({'t':True,'f':False}).infer_objects(copy=False)

df['host_has_profile_pic'] = df['host_has_profile_pic'].replace({'t':True,'f':False}).infer_objects(copy=False)

df['instant_bookable'] = df['instant_bookable'].replace({'t':True,'f':False}).infer_objects(copy=False)

df.drop(columns = ['id', 'amenities', 'latitude', 'longitude', 'name', 'first_review', 'neighbourhood', 'last_review', 'description', 'zipcode', 'host_since'], inplace = True)
```

```
df.columns
```

```
Index(['log_price', 'accommodates', 'bathrooms', 'cleaning_fee',
       'host_has_profile_pic', 'host_identity_verified', 'host_response_rate',
       'instant_bookable', 'number_of_reviews', 'review_scores_rating',
       'bedrooms', 'beds', 'room_type_Private room', 'room_type_Shared room',
       'bed_type_Couch', 'bed_type_Futon', 'bed_type_Pull-out Sofa',
       'bed_type_Real Bed', 'cancellation_policy_moderate',
       'cancellation_policy_strict', 'cancellation_policy_super_strict_30',
       'cancellation_policy_super_strict_60', 'city_Chicago', 'city_DC',
       'city_LA', 'city_NYC', 'city_SF', 'property_type_Bed & Breakfast',
       'property_type_Boat', 'property_type_Boutique hotel',
       'property_type_Bungalow', 'property_type_Cabin',
       'property_type_Camper/RV', 'property_type_Castle', 'property_type_Cave',
       'property_type_Chalet', 'property_type_Condominium',
       'property_type_Dorm', 'property_type_Earth House',
       'property_type_Guest suite', 'property_type_Guesthouse',
       'property_type_Hostel', 'property_type_House', 'property_type_Hut',
       'property_type_In-law', 'property_type_Island', 'property_type_Loft',
       'property_type_Other', 'property_type_Serviced apartment',
       'property_type_Tent', 'property_type_Timeshare', 'property_type_Tipi',
       'property_type_Townhouse', 'property_type_Train',
       'property_type_Treehouse', 'property_type_Vacation home',
       'property_type_Villa', 'property_type_Yurt'],
      dtype='object')
```

```
df.select_dtypes(include = ['bool'])
```

	cleaning_fee	host_has_profile_pic	host_identity_verified	instant_bookable	room_type_Private room
0	True	True	True	False	False
1	True	True	False	True	False
2	True	True	True	True	False
4	True	True	True	True	False
5	True	True	True	True	True
...	...	...	...	...	...
74104	True	True	True	False	False
74105	True	True	True	False	True
74107	True	True	False	False	False
74108	True	True	True	True	False
74110	False	True	True	False	False

57247 rows x 50 columns

```
df.columns
```

```
Index(['log_price', 'accommodates', 'bathrooms', 'cleaning_fee',
       'host_has_profile_pic', 'host_identity_verified', 'host_response_rate',
       'instant_bookable', 'number_of_reviews', 'review_scores_rating',
       'bedrooms', 'beds', 'room_type_Private room', 'room_type_Shared room',
       'bed_type_Couch', 'bed_type_Futon', 'bed_type_Pull-out Sofa',
       'bed_type_Real Bed', 'cancellation_policy_moderate',
       'cancellation_policy_strict', 'cancellation_policy_super_strict_30',
       'cancellation_policy_super_strict_60', 'city_Chicago', 'city_DC',
       'city_LA', 'city_NYC', 'city_SF', 'property_type_Bed & Breakfast',
       'property_type_Boat', 'property_type_Boutique hotel',
       'property_type_Bungalow', 'property_type_Cabin',
       'property_type_Camper/RV', 'property_type_Castle', 'property_type_Cave',
       'property_type_Chalet', 'property_type_Condominium',
       'property_type_Dorm', 'property_type_Earth House',
       'property_type_Guest suite', 'property_type_Guesthouse',
       'property_type_Hostel', 'property_type_House', 'property_type_Hut',
       'property_type_In-law', 'property_type_Island', 'property_type_Loft',
       'property_type_Other', 'property_type_Serviced apartment',
       'property_type_Tent', 'property_type_Timeshare', 'property_type_Tipi',
       'property_type_Townhouse', 'property_type_Train',
       'property_type_Treehouse', 'property_type_Vacation home',
       'property_type_Villa', 'property_type_Yurt'],
      dtype='object')
```

3.1 Test with Multiple Models

เริ่มจากทำการทดลองหลายๆ Regression Model โดยการวัดประสิทธิภาพของโมเดล ด้วยค่า 2 อย่าง คือ ค่า R_Squared และค่า Mean Sqaured Erro rซึ่งโมเดลที่มีประสิทธิภาพมากที่สุด

```python
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.pipeline import Pipeline

# List of regression models
models = {
    'Linear Regression': LinearRegression(),
    'Ridge Regression': Ridge(),
    'Lasso Regression': Lasso(),
    'ElasticNet': ElasticNet(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(),
    'Gradient Boosting': GradientBoostingRegressor(),
}

# Train and evaluate each model
results = {}
for model_name, model in models.items():
    # Train the model
    model.fit(X_train, y_train)
    # Make predictions
    y_pred = model.predict(X_test)
    # Evaluate the model
    mse = mean_squared_error(y_test, y_pred)
    results[model_name] = mse
    # print(f'{model_name} Mean Squared Error: {mse}')

    r_squared =  r2_score(y_test, y_pred)
    print(f'{model_name} model accuracy: {r_squared}')

# Display all results
print("\nModel Evaluation Results:")
for model_name, mse in results.items():
    print(f'{model_name}: {mse}')
```

```
Linear Regression model accuracy: 0.6075889629865713
Ridge Regression model accuracy: 0.6076208925272988
Lasso Regression model accuracy: -0.00010950539730814057
ElasticNet model accuracy: -0.00010950539730814057
Decision Tree model accuracy: 0.309344470042919
Random Forest model accuracy: 0.6072047409683998
Gradient Boosting model accuracy: 0.6340250180964568

Model Evaluation Results:
Linear Regression: 0.16759588301008432
Ridge Regression: 0.1675822461368377
Lasso Regression: 0.4271394529051042
ElasticNet: 0.4271394529051042
Decision Tree: 0.294973923975012
Random Forest: 0.16775998142305826
Gradient Boosting: 0.15630523728012516
```

ซึ่งโมเดลที่มีประสิทธิภาพมากที่สุด จากการจัดการข้อมูลด้วยวิธีของกลุ่มเรา และวัดผลด้วยค่า R_Squared คือ Gradient Boosting Model

กลุ่มเราได้ทำการเลือกโมเดลเพื่อมา วิเคราะห์ต่อ คือ LinearRegression ซึ่งให้ค่าความถูกต้องแม่นยำในระดับกลาง ๆ ถ้าเทียบกับโมเดล อื่น ๆ เนื่องจาก LinearRegression เป็นโมเดลที่เราได้ศึกษาจากอาจารย์ผู้สอน

3.2 Focus on Linear Regression

ทำการสร้างโมเดล

```python
#สร้างโมเดล
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

y_pred = linear_model.predict(X_test)
```

```
{'copy_X': True, 'fit_intercept': True, 'n_jobs': None, 'positive': False}
```

Check ค่า Coefficient

```python
pd.DataFrame({"Columns":X.columns, "Coeffecient":linear_model.coef_}).sort_values(by='Coeffecient', ascending = False)
```

ผลลัพธ์โดยประมาณ :

	Columns	Coeffecient
44	property_type_Island	0.815094
50	property_type_Tipi	0.730752
49	property_type_Timeshare	0.610877
52	property_type_Train	0.608209
20	cancellation_policy_super_strict_60	0.493588
32	property_type_Castle	0.492569
19	cancellation_policy_super_strict_30	0.365942
33	property_type_Cave	0.307521
54	property_type_Vacation home	0.273986
25	city_SF	0.259543
27	property_type_Boat	0.229381
56	property_type_Yurt	0.226710
28	property_type_Boutique hotel	0.213894
0	accommodates	0.153564
45	property_type_Loft	0.141622
26	property_type_Bed & Breakfast	0.126587

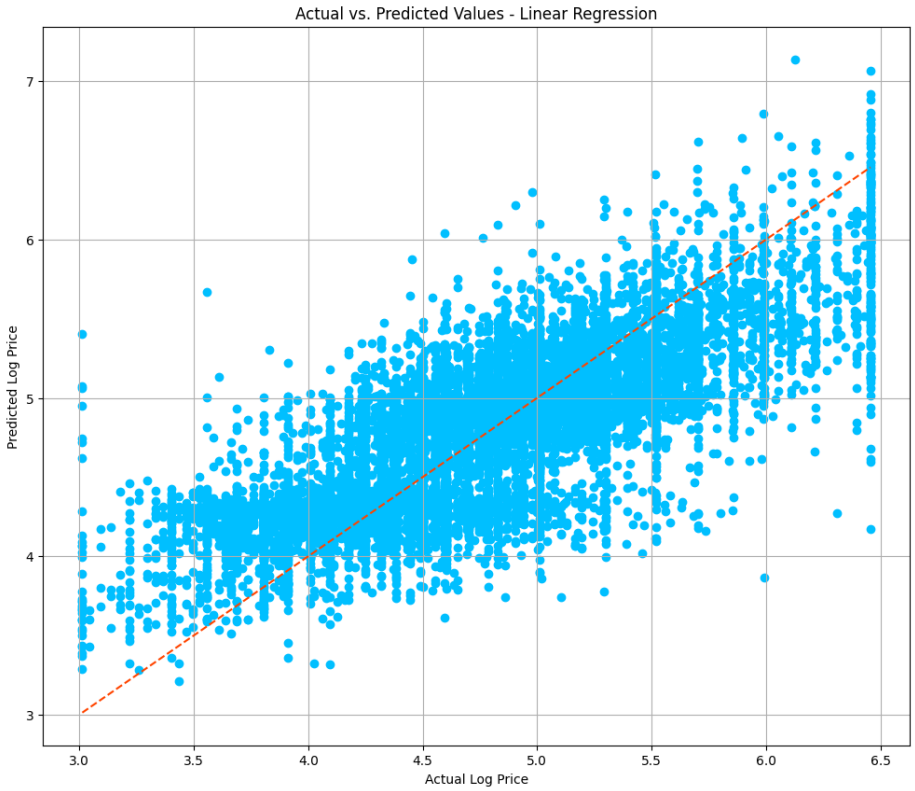
- ค่าสัมประสิทธิ์ที่เป็นบวกที่สุดคือ: property_type_Islan
- ค่าสัมประสิทธิ์ลบน้อยที่สุดคือ: room_type_Roomshare

หาค่า C ในสมการ Linear Regression

```
[ ]  #ค่า C
     linear_model.intercept_

     5.0360187074041685
```

Check graph Actual vs. Predicted Values - Linear Regression

```
plt.figure(figsize=(12, 10))
plt.scatter(y_test, y_pred, color='deepskyblue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='orangered')
plt.xlabel('Actual Log Price')
plt.ylabel('Predicted Log Price')
plt.title('Actual vs. Predicted Values - Linear Regression')
plt.grid(True)
plt.show()
```



ทำการเช็คค่าต่างๆ

```
#ดูค่าวัดความถูกต้อง
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score, explained_variance_score

# Calculating metrics
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)
explained_variance = explained_variance_score(y_test, y_pred)

# Printing the regression report
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"R-squared (R²): {r2}")
print(f"Explained Variance Score: {explained_variance}")
```

```
Mean Absolute Error (MAE): 0.31918213295320114
Mean Squared Error (MSE): 0.16759588301008432
Root Mean Squared Error (RMSE): 0.40938476157532333
R-squared (R²): 0.6075889629865713
Explained Variance Score: 0.6075929865877394
```

3.3 ทดสอบความถูกต้องของโมเดลโดยใช้ KFold

การตั้งค่าจำนวน folds

```
#ทดสอบความถูหต้องของโมเดล แบ่งจำนวน 5 กอง เล็อกมา1กอง มาเป็น Target 4 ตัวที่เหลือเป็นตัว Train
k = 5  # Number of folds
kf = KFold(n_splits=k, shuffle=True, random_state=42)
```

- k กำหนดจำนวน folds เป็น 5
- KFold ตั้งค่าแบ่งข้อมูลออกเป็น 5 ส่วน พร้อมทั้งการสุ่ม shuffle ข้อมูล

การสร้างลิสต์เพื่อเก็บค่าตัวชี้วัด

เพื่อเก็บค่าตัวชี้วัดต่างๆ ในแต่ละรอบของ Cross-Validation

```
mae_list = []
mse_list = []
rmse_list = []
r2_list = []
explained_variance_list = []
```

การแบ่งข้อมูลและการฝึกโมเดล

```
for train_index, test_index in kf.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Train the model
    model = LinearRegression()
    model.fit(X_train, y_train)

    # Predict
    y_pred = model.predict(X_test)

    # Calculate metrics
    mae_list.append(mean_absolute_error(y_test, y_pred))
    mse_list.append(mean_squared_error(y_test, y_pred))
    rmse_list.append(mean_squared_error(y_test, y_pred, squared=False))
    r2_list.append(r2_score(y_test, y_pred))
    explained_variance_list.append(explained_variance_score(y_test, y_pred))
```

- ใช้ kf.split(X) เพื่อแบ่งข้อมูลออกเป็น folds ต่างๆ
- X_train, X_test, y_train, y_test แบ่งข้อมูลสำหรับฝึกและทดสอบโมเดล
- สร้างและฝึกโมเดล Linear Regression
- ทำนายค่าจากข้อมูลทดสอบ (y_pred)
- คำนวณค่าตัวชี้วัดต่างๆ และเก็บค่าไว้ในลิสต์ที่สร้างขึ้น

การคำนวณค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐานของตัวชี้วัด

```
mean_mae = np.mean(mae_list)
std_mae = np.std(mae_list)

mean_mse = np.mean(mse_list)
std_mse = np.std(mse_list)

mean_rmse = np.mean(rmse_list)
std_rmse = np.std(rmse_list)

mean_r2 = np.mean(r2_list)
std_r2 = np.std(r2_list)

mean_explained_variance = np.mean(explained_variance_list)
std_explained_variance = np.std(explained_variance_list)
```

```
print(f"Mean Absolute Error (MAE): {mean_mae:.4f} ± {std_mae:.4f}")
print(f"Mean Squared Error (MSE): {mean_mse:.4f} ± {std_mse:.4f}")
print(f"Root Mean Squared Error (RMSE): {mean_rmse:.4f} ± {std_rmse:.4f}")
print(f"R-squared (R²): {mean_r2:.4f} ± {std_r2:.4f}")
print(f"Explained Variance Score: {mean_explained_variance:.4f} ± {std_explained_variance:.4f}")
```

```
Mean Absolute Error (MAE): 0.3190 ± 0.0021
Mean Squared Error (MSE): 0.1680 ± 0.0032
Root Mean Squared Error (RMSE): 0.4099 ± 0.0039
R-squared (R²): 0.6072 ± 0.0042
Explained Variance Score: 0.6072 ± 0.0042
```

Result and Discussion

Results

พวกเราได้ทำการทดลองกับโมเดล Linear Regression 2 วิธี คือการแบ่ง data ด้วย train_test_split และวิธี k-fold เพื่อทำการเช็คความถูกต้องของโมเดล ซึ่งทั้ง 2 วิธีก็ให้ผลลัพธ์ที่ใกล้เคียงกัน

พวกเราได้นำ Linear Regression Model มาใช้ในการทำนาย log-transformed prices ของที่พักต่าง ๆ ที่ปล่อยให้เข้าพัก (เช่าระยะสั้น) โดยอิงตามคุณสมบัติต่างๆ ของชุดข้อมูล ซึ่งประสิทธิภาพของโมเดลประเมินผ่านการใช้หลากหลายตัวแปรดังนี้:

- Mean Absolute Error (MAE): 0.31918213295320114
- Mean Squared Error (MSE): 0.16759588301008432
- Root Mean Squared Error (RMSE): 0.40938476157532333
- R-squared (R^2): 0.6075889629865713
- Explained Variance Score: 0.6075929865877394

Discussion

ตัวชี้วัดประสิทธิภาพบ่งชี้ว่า Linear Regression Model มีความเหมาะสมกับข้อมูลในระดับที่น่าพอใจ และนี่คือรายละเอียดของตัวแปรแต่ละตัวที่เกี่ยวข้องกับโมเดล:

- **Mean Absolute Error (MAE):** ค่า MAE เท่ากับ 0.319 แสดงให้เห็นว่าโดยเฉลี่ยแล้ว การทำนายของโมเดลเบี่ยงเบนจากค่าจริงของ log prices ประมาณ 0.319 หน่วย นี่เป็นการวัดขนาดเฉลี่ยของค่า errors ในโมเดล โดยไม่คำนึงถึงทิศทางของความผิดพลาด ค่า MAE ที่ต่ำสื่อถึงประสิทธิภาพของโมเดลที่ดี

- **Mean Squared Error (MSE):** ค่า MSE เท่ากับ 0.1676 แสดงถึงค่าเฉลี่ยของความแตกต่างกำลังสองระหว่างค่าที่คาดการณ์ไว้กับค่าจริง จะเกิดค่า errors ที่มากกว่า MAE ซึ่ง MSE ที่ค่อนข้างต่ำบ่งชี้ว่าการคาดการณ์ของแบบจำลองนั้นใกล้เคียงกับค่าจริง โดยเกิดค่า errors เพียงเล็กน้อย

- **Root Mean Squared Error (RMSE):** ค่า RMSE ที่ 0.4094 คือค่ารากที่สองของ MSE และครอบคลุมการวัดข้อผิดพลาดในหน่วยเดียวกับตัวแปรเป้าหมาย (log price) ค่านี้บ่งชี้ว่าการกระจายตัวของค่าความคลาดเคลื่อน (errors) มีลักษณะอย่างไร ในกรณีของเรา RMSE ที่ 0.4094 บ่งชี้ว่าข้อผิดพลาดในการคาดการณ์ทั่วไปอยู่ที่ประมาณ 0.4094 log price units

- **R-squared (R^2):** ค่า R^2 0.6076 บ่งชี้ว่าประมาณ 60.76% ของความแปรปรวนใน log-transformed prices อธิบายได้ด้วยแบบจำลอง ซึ่งบ่งบอกความพอดีของแบบจำลองเรา ขณะที่ค่า R^2 เข้าใกล้กับ 1 จะเป็นตัวบ่งชี้ว่าเป็นแบบจำลองที่อธิบายความแปรปรวนส่วนใหญ่ได้ ซึ่งค่านี้จะชี้ให้เห็นถึง moderate level ของค่า explanatory power

- **Explained Variance Score:** ค่า Explained variance score ของ 0.6076 มีความใกล้เคียงกับค่า R^2 มาก, ซึ่งสามารถวัดสัดส่วนของความแปรปรวนในตัวแปรตามที่สามารถคาดเดาได้จากตัวแปรอิสระอีกด้วย เราสามารถยืนยันได้ว่าแบบจำลองของเราอธิบายความแปรปรวนของ log prices ได้ประมาณ 60.76%

Conclusion

หลังจากการนำชุดข้อมูล Airbnb จาก Kaggle มาทำการจัดการด้วยวิธี data preparation process, EDA and Visualize of Data, Modeling and Evaluation ทำให้เราได้ทำความเข้าใจเกี่ยวกับข้อมูลเกี่ยวกับ Data รวมทั้ง Cleansing ข้อมูล ที่มีความผิดปกติ หลังจากนั้น เราได้ทำการวิเคราะห์ข้อมูลแล้วพล็อตออกมาเป็นกราฟ เพื่อทำความเข้าใจกับข้อมูลเชิงลึก หลังจากนั้นนำชุดข้อมูลมา Preprocessing อีกรอบ แล้วนำข้อมูลมา Train ด้วย Regression Models ซึ่งพวกเราได้ทำการเลือก Linear Regression Model มาเป็นโมเดลหลักของเรา ซึ่งค่าการประเมินผลที่ได้ ก็คือ

- Mean Absolute Error (MAE): 0.31918213295320114,
- Mean Squared Error (MSE): 0.16759588301008432
- Root Mean Squared Error (RMSE): 0.40938476157532333
- R-squared (R^2): 0.6075889629865713
- Explained Variance Score: 0.6075929865877394

หรือถ้าสรุปผลก็คือ โมเดลเราสามารถทำนายราคาบ้านออกมา เป็น log_price ที่มีค่า คาดเคลื่อน ±0.31918213295320114 นั่นเอง เป็นค่าที่สรุปมาได้จาก ค่า MAE