A PROJECT REPORT

ON

Data Driven Insights into Toronto's Airbnb Market

DATA 601 University of Calgary

Master of Data Science and Analytics

Submitted to: Dr Mehdi Karamollahi

By: Adam Virani, Sharlin Kahlon, Yaowei Sun, Xiao Xiao

Table of Contents

1. Abstract

This project presents the successful implementation of data analysis techniques to visualize and narrate insights from an official Airbnb dataset. As a team of four professionals with expertise in finance, business analysis and data analysis, we leveraged our business

knowledge to interpret the data effectively and uncover meaningful trends in Airbnb listings. The analysis includes data cleaning, exploratory data analysis (EDA), and the application of various data visualization methods to tell a compelling data-driven story. We explored key factors affecting Airbnb listings, such as pricing, neighbourhood, and amenities, providing valuable business insights for both hosts and potential investors. In the final report, we conducted thorough data cleaning, performed EDA to uncover patterns and anomalies, and concluded based on our analysis. Our approach balances business insight with technical expertise, providing a comprehensive understanding of the Airbnb landscape via guiding questions.
Keywords: Data Cleaning, Exploratory Data Analysis, Data Visualization, Airbnb Dataset, Business Insight

2. Introduction

In recent years, the short-term rental sector has experienced substantial growth due to the transformation of travelers' lodging options through platforms like Airbnb. Airbnb represents more than a mere travel booking website; it symbolizes a growing community that is reshaping how people view home, travel, and accommodation. By attracting tourists, business travelers, and short-term tenants, Airbnb has become a major influencer in the real estate sector. ([1].The Future of Airbnb: Navigating Regulatory Challenges and Embracing New Trends, 2024)

This project aims to perform a comprehensive analysis of Toronto Airbnb Official Dataset provided by Inside Airbnb. The dataset contains extensive details about Airbnb operations in Toronto, property types, host profiles, pricing, availability, amenities and host reviews enabling a thorough examination of rental landscape in Canada. By leveraging this rich dataset and with a focus on understanding the variances in Toronto, we hope to investigate important market trends, year-over-year comparisons, comparing metropolitan city with the capital city of Ottawa and other Canadian cities, variables influencing patterns, average daily rate, host behavior, and the impact of growing population.

The target subject area or domain that we are focusing on in our project is real estate analytics and tourism. Our objective is to use cutting-edge data analysis tools to find significant trends and offer useful insights that can guide decision-making for all parties involved in Canada's developing Airbnb business. The findings of this analysis will be valuable not just to potential investors or current hosts, but also to local governments in shaping rules. Understanding the dynamics of Airbnb's influence on the housing market in Toronto will help them shape effective regulations and policies that address the needs of the community while fostering a thriving tourism sector. Ultimately, our work seeks to contribute to a more informed dialogue around the implications of short-term and long-term rentals in urban environments.

2.1. Guiding Questions Overview & Team Contributions

Through extensive investigation of given data and by leveraging data analytics techniques, we would like to understand:

1. What are the characteristics of Toronto's Airbnb market compared with other Canadian cities? (Yaowei)
2. Which factors have the most significant impact on the pricing of Airbnb listings in Toronto? (Sharlin)
3. How does a year-over-year comparison account for changes in the Airbnb market in Toronto? (Adam)
4. What aspects of the Airbnb platform affect hosts' reviews and performance, do top rated hosts have anything in common, are there any differences between hosts with few listings vs many? (Xiao)

Reading different articles on Airbnb and the way it took over the world , we came across that the Airbnb rise has come at a huge cost to urban life and has also led to an increase in real estate prices. (Sherwood, 2019) . Studying the Airbnb trend in one of the popular cities in Canada-Toronto involves examining the effects of different factors on pricing, the growing population, a comparison with Ottawa and other Canadian Cities, analyzing hosts' behavior and the potential impact of reviews to gain a deeper insight into Airbnb's strategy. Through responding to these inquiries, we will gain further insight into the Toronto Airbnb sector and determine the prudence of investing in high-priced properties.

3. Project Methodology

a.) Data Collection: Data collection is an organized collection of information from a variety of sources in order to answer specific questions or evaluate results. It can be carried out in a variety of ways, including observations, experiments, or data extraction from existing records.

b.) Data Cleaning: It is the process of removing missing values, incorrect or duplicate data within dataset. Also called data scrubbing or data munging.

c.) Exploratory Data Analysis: EDA is statistical approach to measure given dataset and its attributes. It helps to understand if data is left skewed, right skewed or has balanced distribution. Either categorical or numerical, it performs valuable investigations, discover patterns and present visual representations.

d.) Visualizations and Findings: Visualizations are vital for interpreting and presenting data analysis findings. They convert unprocessed data into comprehensible representations, which facilitates the discovery of trends, patterns, and insights.

4. Data Collection

Data Collection: The data for this project was obtained from the Official Airbnb platform, which scrapes the official Airbnb site for data related to listings and reviews. The data collected covers eight major Canadian cities over a period from June 2023 to September 2024, with some portions from 2020 and 2021. The dataset is structured in a .XLSX or .CSV format, ensuring that it can be processed easily for analysis. The dataset includes detailed information about Airbnb listings, hosts, and reviews, with a total of 79 columns, though some columns are duplicates and will be removed as part of the data cleaning process. Approval for the use of this data, as well as proper attribution to the source, has been referenced. We have used four datasets for our analysis:-

Dataset URL\

Gathered_csv https://insideairbnb.com/get-the-data/

reviews.csv https://insideairbnb.com/get-the-data/

listings 2021 Toronto.csv https://www.kaggle.com/datasets/michasolo/toronto-airbnb-new-listings\

listings_sep_09_2020.csv https://www.kaggle.com/datasets/robinkongninglo/toronto-airbnb-dataset?resource=download\

Datasets uploaded for download:

listings_sep_09_2020.csv https://uofc-my.sharepoint.com/:x:/g/personal/adam_virani_ucalgary_ca/EVf-2obzeMxAmuMo9vFCEpUBxRGwnaS2fva8mWqX50ITxQ?e=RDsXKh

listings 2021 Toronto.csv https://uofc-my.sharepoint.com/:x:/g/personal/adam_virani_ucalgary_ca/EWuI_RYHrt9FisTEjyyw5hwBem0n6ny
e=PIbNE9

Gathered_data.csv https://uofc-my.sharepoint.com/:x:/g/personal/adam_virani_ucalgary_ca/EScs8--4txlBj-kq9h6pIuMBg8m1ZBr78YG6LWUi-FgDbw?e=AMYrsw

reviews.csv https://onedrive.live.com/?authkey=%21ADYf0136eLIL4bs&id=F10C828FB89FA24D%21408304&cid=F10C828FB89FA24D&

4.1 Dataset Attributes

Licensing:

This data is licensed under a Creative Commons Attribution 4.0 International License. Dataset Details: The dataset is divided into several key categories of information, each represented by a number of attributes:

• Listing Information:
o id, listing_url, scrape_id, last_scraped, source, name, description, neighborhood_overview, picture_url

• Host Information:
o host_id, host_url, host_name, host_since, host_location, host_about, host_response_time, host_response_rate, host_acceptance_rate, host_is_superhost, host_thumbnail_url, host_picture_url, host_neighbourhood, host_listings_count, host_total_listings_count, host_verifications, host_has_profile_pic, host_identity_verified

• Location Information:
o neighbourhood, neighbourhood_cleansed, neighbourhood_group_cleansed, latitude, longitude, city

• Property Information: o property_type, room_type, accommodates, bathrooms, bathrooms_text, bedrooms, beds, amenities

• Pricing and Booking Information: o price, minimum_nights, maximum_nights, minimum_minimum_nights, maximum_minimum_nights, minimum_maximum_nights, maximum_maximum_nights, minimum_nights_avg_ntm, maximum_nights_avg_ntm

• Availability Information: o has_availability, availability_30, availability_60, availability_90, availability_365, calendar_updated, calendar_last_scraped\

• Review Information: o number_of_reviews, number_of_reviews_ltm, number_of_reviews_l30d, first_review, last_review, review_scores_rating, review_scores_accuracy, review_scores_cleanliness, review_scores_checkin, review_scores_communication, \review_scores_location, review_scores_value, reviews_per_month

• Other Attributes: o license, instant_bookable, calculated_host_listings_count, calculated_host_listings_count_entire_homes, calculated_host_listings_count_private_rooms, calculated_host_listings_count_shared_rooms\

5. Data Cleaning

1. Data Collection
2. Drop Unnecessary Columns
3. Replace placeholders with null
4. Check Datatypes

5. Data Imputation
6. Filtering Toronto Dataset

In [88]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
pd.set_option('display.max_columns',50)
df = pd.read_csv(r'Gathered_data.csv', low_memory=False)
df.head()
```

Out[88]:

	Unnamed: 0	id	listing_url	scrape_id	last_scraped	sou
0	0	2843	https://www.airbnb.com/rooms/2843	20240913025610	2024-09-13	sci
1	1	29059	https://www.airbnb.com/rooms/29059	20240913025610	2024-09-13	sci
2	2	29061	https://www.airbnb.com/rooms/29061	20240913025610	2024-09-13	sci
3	3	31847	https://www.airbnb.com/rooms/31847	20240913025610	2024-09-13	sci
4	4	34715	https://www.airbnb.com/rooms/34715	20240913025610	2024-09-13	sci

5 rows × 79 columns

Size of dataset after gathering all Canadian cities dataset on official Airbnb site.

In [89]:
```
df.shape
```

Out[89]:
```
(474954, 79)
```

In [90]:
```
columns_df = pd.DataFrame(df.columns, columns=["Column Names"])
# Display the column names in a table format
columns_df
```

Out[90]:

	Column Names
0	Unnamed: 0
1	id
2	listing_url
3	scrape_id
4	last_scraped
...	...
74	calculated_host_listings_count_shared_rooms
75	reviews_per_month
76	mark
77	city
78	price+

79 rows × 1 columns

In [91]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 474954 entries, 0 to 474953
Data columns (total 79 columns):
 #   Column                                        Non-Null Count   Dtype  
---  ------                                        --------------   -----  
 0   Unnamed: 0                                    474954 non-null  int64  
 1   id                                            474954 non-null  int64  
 2   listing_url                                   474954 non-null  object 
 3   scrape_id                                     474954 non-null  int64  
 4   last_scraped                                  474954 non-null  object 
 5   source                                        474954 non-null  object 
 6   name                                          474954 non-null  object 
 7   description                                   474954 non-null  object 
 8   neighborhood_overview                         474954 non-null  object 
 9   picture_url                                   474954 non-null  object 
 10  host_id                                       474954 non-null  int64  
 11  host_url                                      474954 non-null  object 
 12  host_name                                     474954 non-null  object 
 13  host_since                                    474954 non-null  object 
 14  host_location                                 474954 non-null  object 
 15  host_about                                    474954 non-null  object 
 16  host_response_time                            474954 non-null  object 
 17  host_response_rate                            474954 non-null  object 
 18  host_acceptance_rate                          474954 non-null  object 
 19  host_is_superhost                             474954 non-null  object 
 20  host_thumbnail_url                            474954 non-null  object 
 21  host_picture_url                              474954 non-null  object 
 22  host_neighbourhood                            474954 non-null  object 
 23  host_listings_count                           474954 non-null  object 
 24  host_total_listings_count                     474954 non-null  object 
 25  host_verifications                            474954 non-null  object 
 26  host_has_profile_pic                          474954 non-null  object 
 27  host_identity_verified                        474954 non-null  object 
 28  neighbourhood                                 474954 non-null  object 
 29  neighbourhood_cleansed                        474954 non-null  object 
 30  neighbourhood_group_cleansed                  474954 non-null  object 
 31  latitude                                      474954 non-null  float64
 32  longitude                                     474954 non-null  float64
 33  property_type                                 474954 non-null  object 
 34  room_type                                     474954 non-null  object 
 35  accommodates                                  474954 non-null  int64  
 36  bathrooms                                     474954 non-null  object 
 37  bathrooms_text                                474954 non-null  object 
 38  bedrooms                                      474954 non-null  object 
 39  beds                                          474954 non-null  object 
 40  amenities                                     474954 non-null  object 
 41  price                                         474954 non-null  object 
 42  minimum_nights                                474954 non-null  int64  
 43  maximum_nights                                474954 non-null  int64  
 44  minimum_minimum_nights                        474954 non-null  int64  
 45  maximum_minimum_nights                        474954 non-null  int64  
 46  minimum_maximum_nights                        474954 non-null  int64  
 47  maximum_maximum_nights                        474954 non-null  int64  
 48  minimum_nights_avg_ntm                        474954 non-null  float64
 49  maximum_nights_avg_ntm                        474954 non-null  float64
 50  calendar_updated                              474954 non-null  object 
```

```
 51  has_availability                                474954 non-null  object
 52  availability_30                                 474954 non-null  int64
 53  availability_60                                 474954 non-null  int64
 54  availability_90                                 474954 non-null  int64
 55  availability_365                                474954 non-null  int64
 56  calendar_last_scraped                           474954 non-null  object
 57  number_of_reviews                               474954 non-null  int64
 58  number_of_reviews_ltm                           474954 non-null  int64
 59  number_of_reviews_l30d                          474954 non-null  int64
 60  first_review                                    474954 non-null  object
 61  last_review                                     474954 non-null  object
 62  review_scores_rating                            474954 non-null  object
 63  review_scores_accuracy                          474954 non-null  object
 64  review_scores_cleanliness                       474954 non-null  object
 65  review_scores_checkin                           474954 non-null  object
 66  review_scores_communication                     474954 non-null  object
 67  review_scores_location                          474954 non-null  object
 68  review_scores_value                             474954 non-null  object
 69  license                                         474954 non-null  object
 70  instant_bookable                                474954 non-null  object
 71  calculated_host_listings_count                  474954 non-null  int64
 72  calculated_host_listings_count_entire_homes     474954 non-null  int64
 73  calculated_host_listings_count_private_rooms    474954 non-null  int64
 74  calculated_host_listings_count_shared_rooms     474954 non-null  int64
 75  reviews_per_month                               474954 non-null  object
 76  mark                                            474954 non-null  int64
 77  city                                            474954 non-null  object
 78  price+                                          474954 non-null  float64
dtypes: float64(5), int64(23), object(51)
memory usage: 286.3+ MB
```

In [92]:
```
df.isnull().sum()
```

Out[92]:
```
Unnamed: 0                                      0
id                                              0
listing_url                                     0
scrape_id                                       0
last_scraped                                    0
                                               ..
calculated_host_listings_count_shared_rooms     0
reviews_per_month                               0
mark                                            0
city                                            0
price+                                          0
Length: 79, dtype: int64
```

Drop Unnecessary Columns

In [93]:
```
#Dropping identification purpose, url related, non-informative, redundant and unnec
df.drop(columns=['Unnamed: 0', 'listing_url', 'scrape_id', 'source', 'calendar_last
                 'host_picture_url','license','host_url','host_name','host_thumbnai
                 'bathrooms_text','minimum_minimum_nights','maximum_minimum_nights'
                 'minimum_nights_avg_ntm','maximum_nights_avg_ntm','calendar_update
```

In [94]:
```
#df.drop(columns=['host-about','neighbourhood_group_cleansed'], inplace=True, error
```

In [95]:
```
#df.drop(['host_response_time','host_response_rate','host_acceptance_rate','neighbo
```

In [96]:
```
#looking at result above, dropping columns with large number of missing values as i
#and columns with free text data like host_about

#df.drop(['neighborhood_overview','neighbourhood','host_about', 'bathrooms'], axis=
```

In [97]:
```
df.shape
```

Out[97]: (474954, 59)

Check Datatypes

In [98]:
```
#change date column to datetime
df['host_since'] = pd.to_datetime(df['host_since'], errors='coerce')
df['first_review'] = pd.to_datetime(df['first_review'], errors='coerce')
df['last_review'] = pd.to_datetime(df['last_review'], errors='coerce')
df['last_scraped'] = pd.to_datetime(df['last_scraped'], errors='coerce')
```

In [99]:
```
# Convertion for question 4.
date_col=['host_since','first_review','last_review']
txt_to_numeric=['host_response_time','host_verifications']
unique_response=set([i for i in df['host_response_time']])
print(unique_response)
mapping1={'within an hour':1,'within a day':12,'within a few hours':5,'a few days o
unique_verifications=set([i for i in df['host_verifications']])
print(unique_verifications)
mapping2={"['email', 'phone', 'photographer', 'work_email']":4,"['email', 'work_ema
          "['email']":1,'[]':'coerce',"['work_email']":1,"['email', 'phone']":2,"['
txt_to_numeric={'host_response_time':mapping1,'host_verifications':mapping2}
numeric_col=['host_listings_count',
             'bathrooms','bedrooms','review_scores_rating','review_scores_accuracy'
             'review_scores_checkin','review_scores_communication','review_scores_l
per_col=['host_response_rate','host_acceptance_rate']
bool_col=['host_is_superhost','host_has_profile_pic','host_identity_verified','has_
df[date_col]=df[date_col].apply(pd.to_datetime,format='%Y-%m-%d',errors='coerce')
for col,mapping in txt_to_numeric.items():
    df[col]=df[col].map(mapping).apply(pd.to_numeric,errors='coerce')
df[numeric_col]=df[numeric_col].apply(pd.to_numeric,errors='coerce')
df['host_response_rate']=df['host_response_rate'].astype(str).str.rstrip('%').apply
df['host_acceptance_rate']=df['host_acceptance_rate'].astype(str).str.rstrip('%').a
df[bool_col]=df[bool_col].replace({'t':True,'f':False}).astype(bool)
```

```
{'within an hour', 'within a day', 'a few days or more', 'within a few hours', '$0.0
0'}
{"['phone']", '[]', "['email', 'phone', 'photographer', 'work_email']", "['email']",
"['work_email']", "['phone', 'work_email']", '$0.00', "['email', 'work_email']", "
['email', 'phone', 'work_email']", "['email', 'phone']"}
```

In [100…
```
print(df.dtypes)  # Check the data types of the columns
```

```
id                                                int64
last_scraped                                datetime64[ns]
name                                               object
description                                        object
neighborhood_overview                              object
host_id                                             int64
host_since                                  datetime64[ns]
host_location                                      object
host_about                                         object
host_response_time                                float64
host_response_rate                                float64
host_acceptance_rate                              float64
host_is_superhost                                    bool
host_neighbourhood                                 object
host_listings_count                               float64
host_total_listings_count                          object
host_verifications                                float64
host_has_profile_pic                                 bool
host_identity_verified                               bool
neighbourhood                                      object
neighbourhood_cleansed                             object
neighbourhood_group_cleansed                       object
latitude                                          float64
longitude                                         float64
property_type                                      object
room_type                                          object
accommodates                                        int64
bathrooms                                         float64
bedrooms                                          float64
beds                                               object
amenities                                          object
price                                              object
minimum_nights                                      int64
maximum_nights                                      int64
has_availability                                     bool
availability_30                                     int64
availability_60                                     int64
availability_90                                     int64
availability_365                                    int64
number_of_reviews                                   int64
number_of_reviews_ltm                               int64
number_of_reviews_l30d                              int64
first_review                                datetime64[ns]
last_review                                 datetime64[ns]
review_scores_rating                              float64
review_scores_accuracy                            float64
review_scores_cleanliness                         float64
review_scores_checkin                             float64
review_scores_communication                       float64
review_scores_location                            float64
review_scores_value                               float64
instant_bookable                                     bool
calculated_host_listings_count                      int64
calculated_host_listings_count_entire_homes         int64
calculated_host_listings_count_private_rooms        int64
calculated_host_listings_count_shared_rooms         int64
```

```
reviews_per_month                                 float64
city                                               object
price+                                            float64
dtype: object
```

In [101…

```
df.isnull().sum()
```

Out[101…

id	0
last_scraped	0
name	0
description	0
neighborhood_overview	0
host_id	0
host_since	34
host_location	0
host_about	0
host_response_time	110597
host_response_rate	110597
host_acceptance_rate	88593
host_is_superhost	0
host_neighbourhood	0
host_listings_count	34
host_total_listings_count	0
host_verifications	271
host_has_profile_pic	0
host_identity_verified	0
neighbourhood	0
neighbourhood_cleansed	0
neighbourhood_group_cleansed	0
latitude	0
longitude	0
property_type	0
room_type	0
accommodates	0
bathrooms	221302
bedrooms	126453
beds	0
amenities	0
price	0
minimum_nights	0
maximum_nights	0
has_availability	0
availability_30	0
availability_60	0
availability_90	0
availability_365	0
number_of_reviews	0
number_of_reviews_ltm	0
number_of_reviews_l30d	0
first_review	97461
last_review	97461
review_scores_rating	97357
review_scores_accuracy	97707
review_scores_cleanliness	97696
review_scores_checkin	97723
review_scores_communication	97707
review_scores_location	97735
review_scores_value	97725
instant_bookable	0
calculated_host_listings_count	0
calculated_host_listings_count_entire_homes	0
calculated_host_listings_count_private_rooms	0
calculated_host_listings_count_shared_rooms	0

```
reviews_per_month                               97461
city                                                0
price+                                              0
dtype: int64
```

Replace placeholders with null

There are no null values; however, placeholder values like `$0.00` can be observed in a few columns, and the data types need to be fixed. We chose to replace placehokders with NaN values and impute with the mean, median, or a default value as part of Data Imputation.

In [102…
```
# Replace any $0.00 with NaN
columns_to_clean=['neighborhood_overview', 'host_since', 'host_location','host_resp
                  'host_acceptance_rate', 'host_is_superhost', 'host_listings_count
                  'host_has_profile_pic', 'host_identity_verified','neighbourhood',
                 'price','first_review','last_review','review_scores_rating','revie
                  'review_scores_cleanliness','review_scores_checkin','review_score
                  'review_scores_value','reviews_per_month']

placeholders = ['$0.00', 'N/A', 0]

# Replace placeholders with NaN
df[columns_to_clean] = df[columns_to_clean].replace(placeholders, np.nan)
```

In [103…
```
#Convert numerical column values to float64 datatype and remove $ symbol

columns_to_change_datatype= ['host_response_time','host_response_rate',
                  'host_acceptance_rate', 'host_listings_count','host_total_listing
                 'price','minimum_nights', 'maximum_nights','availability_30', 'ava
                    'number_of_reviews', 'number_of_reviews_ltm', 'number_of_review
                             'review_scores_rating','review_scores_accuracy',
                  'review_scores_cleanliness','review_scores_checkin','review_score
                  'review_scores_value','reviews_per_month','price+','calculated_ho
                             'calculated_host_listings_count_private_rooms', 'calcu

#df[columns_to_change_datatype] = df[columns_to_change_datatype].replace({'\$': '',

df[columns_to_change_datatype] = df[columns_to_change_datatype].replace({'\\$': '',
df[columns_to_change_datatype] = df[columns_to_change_datatype].apply(pd.to_numeric
```

Data Imputation

Data imputation is the process of replacing missing values in a dataset with substituted values. Purpose: To fill in missing values so that the dataset can be used for through analysis and training - testing method.

Various techniques can be used to impute data, including: Mean/Median Imputation: Replacing missing values with the mean or median of the column. Mode Imputation: For categorical data, replacing missing values with the most frequent value (mode). Forward/Backward Fill: Using the previous or next values in a time series context.

In [104…
```
df['price'] = df['price'].fillna(df['price'].median())
df['host_listings_count'] = df['host_listings_count'].fillna(df['host_listings_coun
df['host_total_listings_count'] = df['host_total_listings_count'].fillna(df['host_t
df['reviews_per_month'] = df['reviews_per_month'].fillna(df['reviews_per_month'].me
df['bedrooms'] = df['bedrooms'].fillna(df['bedrooms'].median())
df['beds'] = df['beds'].fillna(df['beds'].median())
```

In [105…
```
df['host_is_superhost'] = df['host_is_superhost'].fillna(df['host_is_superhost'].mo
df['host_identity_verified'] = df['host_identity_verified'].fillna(df['host_identit
```

In [106…
```
df['first_review'] = df['first_review'].ffill()
df['host_since'] = df['host_since'].ffill()
df['last_review'] = df['last_review'].ffill()
df['review_scores_rating'] = df['review_scores_rating'].ffill()
df['review_scores_accuracy'] = df['review_scores_accuracy'].ffill()
df['review_scores_cleanliness'] = df['review_scores_cleanliness'].ffill()
df['review_scores_checkin'] = df['review_scores_checkin'].ffill()
df['review_scores_communication'] = df['review_scores_communication'].ffill()
df['review_scores_location'] = df['review_scores_location'].ffill()
df['review_scores_value'] = df['review_scores_value'].ffill()
df['host_has_profile_pic'] = df['host_has_profile_pic'].ffill()


df['host_location'] = df['host_location'].fillna(df['host_location'].mode()[0])
```

Filtering Toronto & Ottawa Dataset

In [107…
```
TnO_df=df[(df['city']=='Toronto')|(df['city']=='Ottawa')]
```

Filtering Toronto Dataset

In [108…
```
Toronto_df= df[df['city']=='Toronto']
```

In [109…
```
Toronto_df.shape
```

Out[109… (248523, 59)

In [110…
```
Toronto_df.isnull().sum()
```

Out[110…

id	0
last_scraped	0
name	0
description	0
neighborhood_overview	115169
host_id	0
host_since	0
host_location	0
host_about	0
host_response_time	75601
host_response_rate	78647
host_acceptance_rate	71561
host_is_superhost	0
host_neighbourhood	0
host_listings_count	0
host_total_listings_count	0
host_verifications	203
host_has_profile_pic	0
host_identity_verified	0
neighbourhood	115159
neighbourhood_cleansed	0
neighbourhood_group_cleansed	0
latitude	0
longitude	0
property_type	0
room_type	0
accommodates	0
bathrooms	122269
bedrooms	0
beds	0
amenities	0
price	0
minimum_nights	0
maximum_nights	0
has_availability	0
availability_30	0
availability_60	0
availability_90	0
availability_365	0
number_of_reviews	0
number_of_reviews_ltm	0
number_of_reviews_l30d	0
first_review	0
last_review	0
review_scores_rating	0
review_scores_accuracy	0
review_scores_cleanliness	0
review_scores_checkin	0
review_scores_communication	0
review_scores_location	0
review_scores_value	0
instant_bookable	0
calculated_host_listings_count	0
calculated_host_listings_count_entire_homes	0
calculated_host_listings_count_private_rooms	0
calculated_host_listings_count_shared_rooms	0

```
reviews_per_month                                0
city                                             0
price+                                           0
dtype: int64
```

In [111…

```
Toronto_df= df[df['city']=='Toronto']
Toronto_df.isnull().sum()
```

Out[111…

id	0
last_scraped	0
name	0
description	0
neighborhood_overview	115169
host_id	0
host_since	0
host_location	0
host_about	0
host_response_time	75601
host_response_rate	78647
host_acceptance_rate	71561
host_is_superhost	0
host_neighbourhood	0
host_listings_count	0
host_total_listings_count	0
host_verifications	203
host_has_profile_pic	0
host_identity_verified	0
neighbourhood	115159
neighbourhood_cleansed	0
neighbourhood_group_cleansed	0
latitude	0
longitude	0
property_type	0
room_type	0
accommodates	0
bathrooms	122269
bedrooms	0
beds	0
amenities	0
price	0
minimum_nights	0
maximum_nights	0
has_availability	0
availability_30	0
availability_60	0
availability_90	0
availability_365	0
number_of_reviews	0
number_of_reviews_ltm	0
number_of_reviews_l30d	0
first_review	0
last_review	0
review_scores_rating	0
review_scores_accuracy	0
review_scores_cleanliness	0
review_scores_checkin	0
review_scores_communication	0
review_scores_location	0
review_scores_value	0
instant_bookable	0
calculated_host_listings_count	0
calculated_host_listings_count_entire_homes	0
calculated_host_listings_count_private_rooms	0
calculated_host_listings_count_shared_rooms	0

```
reviews_per_month                                   0
city                                                0
price+                                              0
dtype: int64
```

In [112…
```
df['neighbourhood'] = df['neighbourhood'].ffill()
```

In [113…
```
Toronto_df= df[df['city']=='Toronto']
Toronto_df.isnull().sum()
```

Out[113…

id	0
last_scraped	0
name	0
description	0
neighborhood_overview	115169
host_id	0
host_since	0
host_location	0
host_about	0
host_response_time	75601
host_response_rate	78647
host_acceptance_rate	71561
host_is_superhost	0
host_neighbourhood	0
host_listings_count	0
host_total_listings_count	0
host_verifications	203
host_has_profile_pic	0
host_identity_verified	0
neighbourhood	0
neighbourhood_cleansed	0
neighbourhood_group_cleansed	0
latitude	0
longitude	0
property_type	0
room_type	0
accommodates	0
bathrooms	122269
bedrooms	0
beds	0
amenities	0
price	0
minimum_nights	0
maximum_nights	0
has_availability	0
availability_30	0
availability_60	0
availability_90	0
availability_365	0
number_of_reviews	0
number_of_reviews_ltm	0
number_of_reviews_l30d	0
first_review	0
last_review	0
review_scores_rating	0
review_scores_accuracy	0
review_scores_cleanliness	0
review_scores_checkin	0
review_scores_communication	0
review_scores_location	0
review_scores_value	0
instant_bookable	0
calculated_host_listings_count	0
calculated_host_listings_count_entire_homes	0
calculated_host_listings_count_private_rooms	0
calculated_host_listings_count_shared_rooms	0

```
reviews_per_month                                 0
city                                              0
price+                                            0
dtype: int64
```

Reading, Cleaning, Filtering for Toronto 2020 and 2021 datasets

In [114…
```
df2 = pd.read_csv('listings 2021 toronto.csv')
df2.head()
df2.shape
df2.isnull().sum()
missing_values = df2['price'].isnull().sum()
print(missing_values)
```

```
0
```

In [115…
```
df2[columns_to_clean] = df[columns_to_clean].replace(placeholders, np.nan)
print(df2['price'])
```

```
0         50.0
1        197.0
2        236.0
3         40.0
4        160.0
         ...
15079    140.0
15080     96.0
15081    159.0
15082    125.0
15083    262.0
Name: price, Length: 15084, dtype: float64
```

In [116…
```
df3 = pd.read_csv('listings_sep_09_2020.csv')
df3.head()
df3.shape
df3.isnull().sum()
missing_values2 = df3['price'].isnull().sum()
print(missing_values)
```

```
0
```

In [117…
```
df3[columns_to_clean] = df3[columns_to_clean].replace(placeholders, np.nan)
print(df3['price'])
df3['price'] = df3['price'].replace({r'\$': '', r',': ''}, regex=True).astype(float
print(df3['price'])
```

```
0        $469.00
1         $99.00
2         $66.00
3         $70.00
4        $135.00
          ...
19338     $99.00
19339     $37.00
19340     $29.00
19341    $111.00
19342    $111.00
Name: price, Length: 19343, dtype: object
0        469.0
1         99.0
2         66.0
3         70.0
4        135.0
         ...
19338     99.0
19339     37.0
19340     29.0
19341    111.0
19342    111.0
Name: price, Length: 19343, dtype: float64
```

6. Exploratory Data Analysis

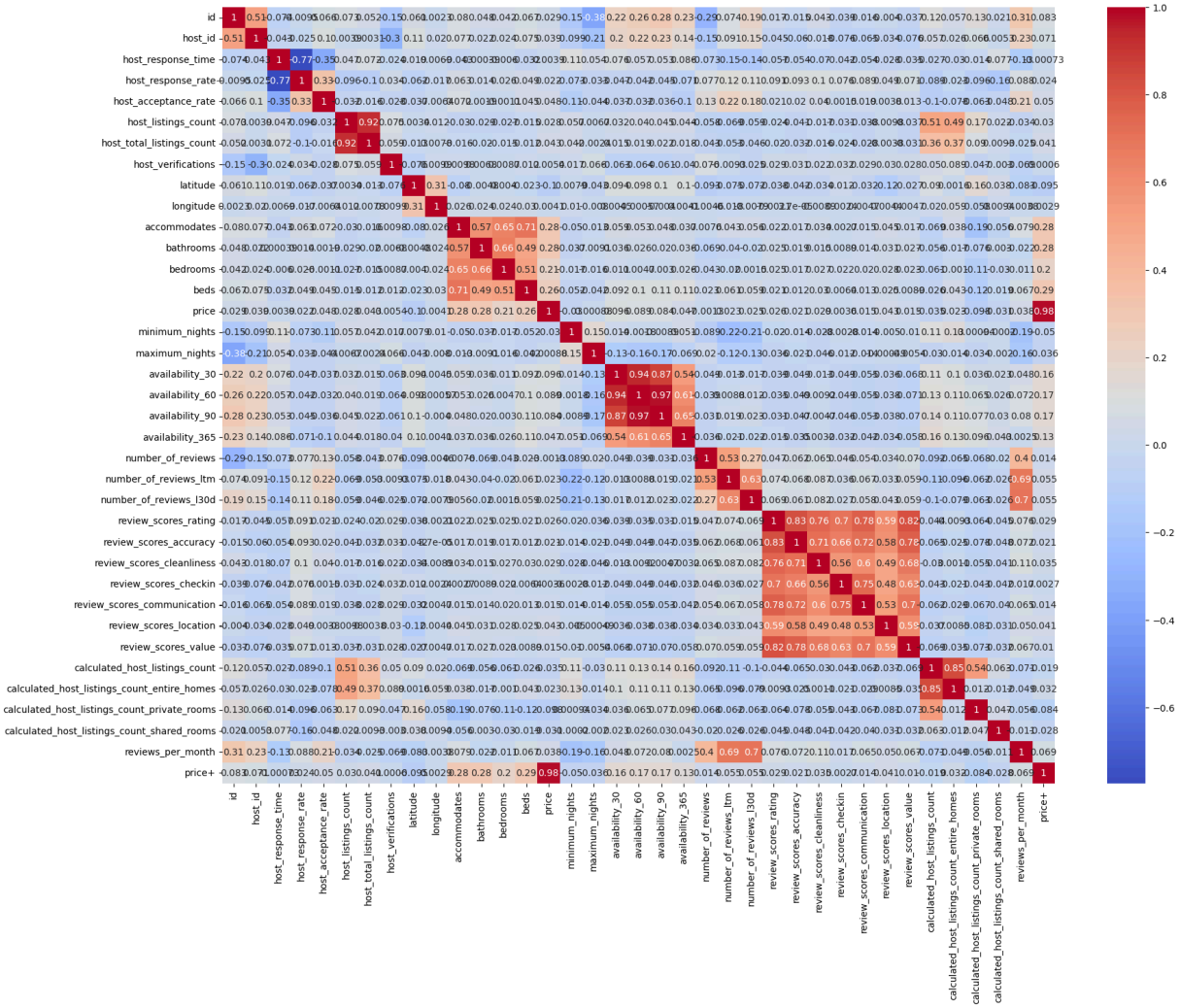
Exploratory Data Analysis (EDA) is a critical step in the data analysis process that involves examining and visualizing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. The goal of EDA is to understand the underlying patterns in the data, identify anomalies, and generate insights that can inform further analysis or model building.

In [118…

```
#correlation_matrix = Toronto_df.corr()
#plt.figure(figsize=(20,15))
#sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm")
#plt.show()
```

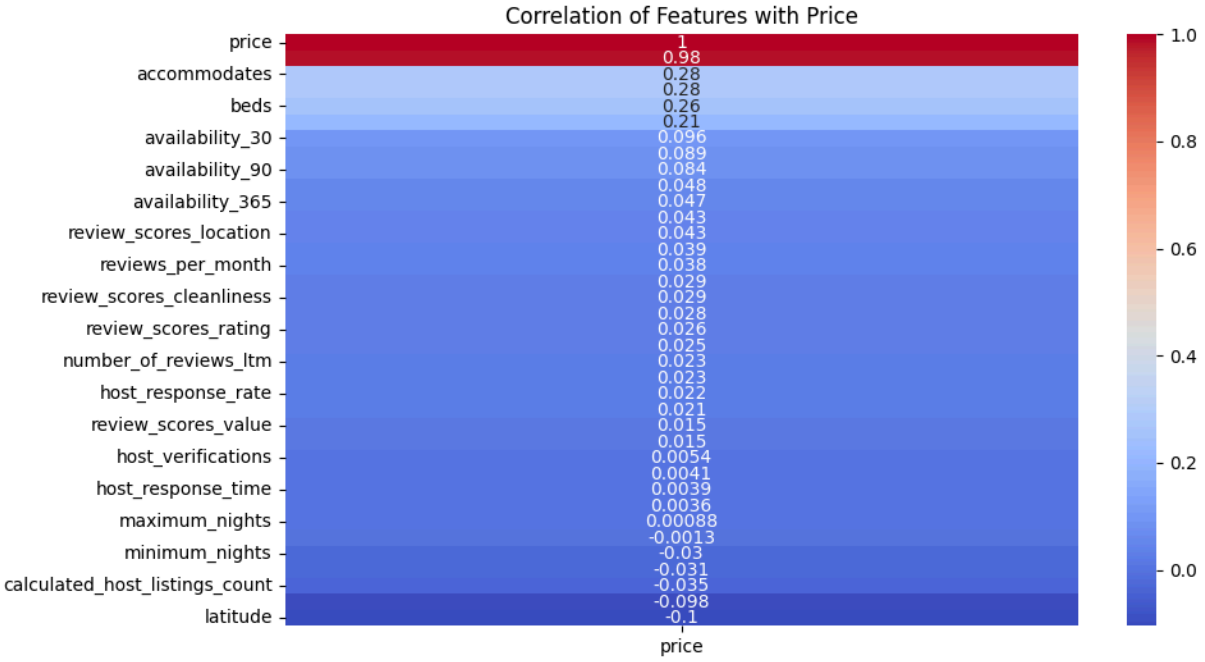
In [119…

```
numeric_df = Toronto_df.select_dtypes(include=['number'])


correlation_matrix = numeric_df.corr()

# Plot the heatmap
plt.figure(figsize=(20, 15))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm")
plt.show()
```



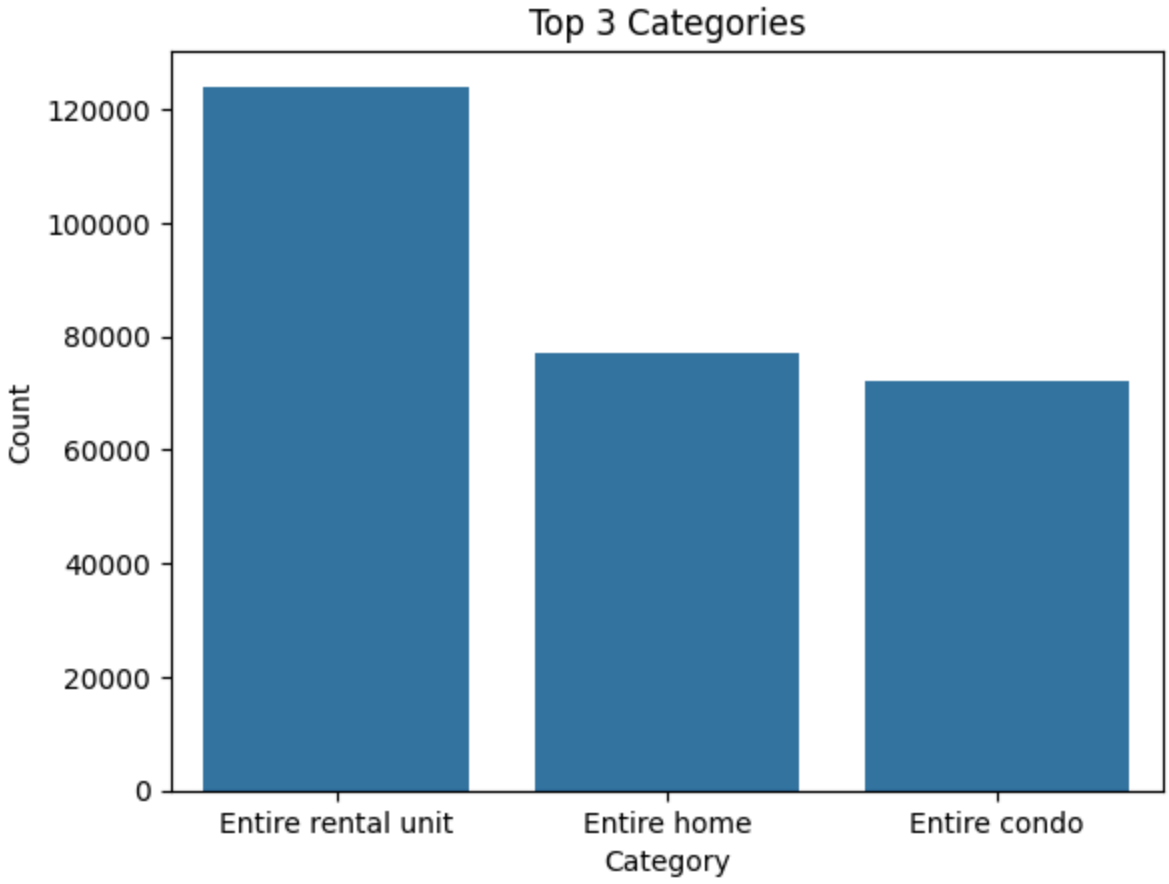
In [120…
```
#corr_matrix = Toronto_df.corr()


#plt.figure(figsize=(10, 6))
#sns.heatmap(corr_matrix[['price']].sort_values(by='price', ascending=False), annot
#plt.title('Correlation of Features with Price')
#plt.show()
```

In [121…
```
numeric_df = Toronto_df.select_dtypes(include=['number'])
corr_matrix = numeric_df.corr()
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix[['price']].sort_values(by='price', ascending=False), annot=
plt.title('Correlation of Features with Price')
plt.show()
```



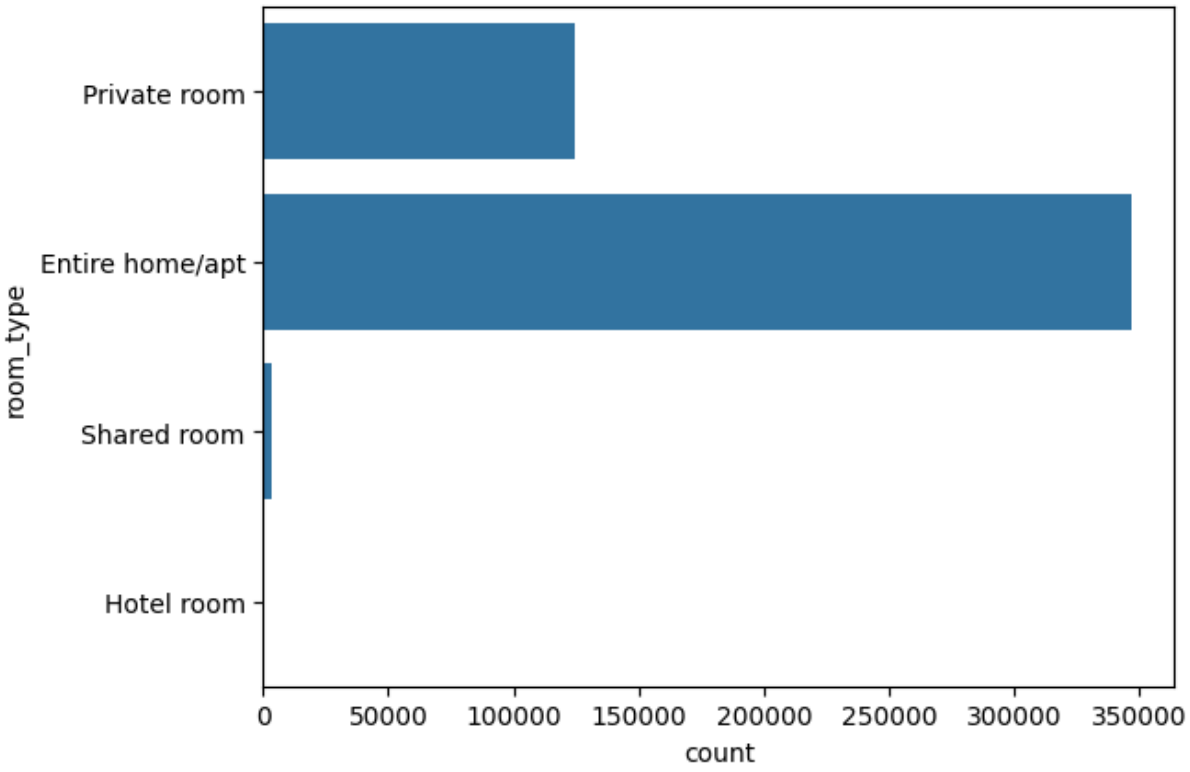
In [122…

```
category_counts = df['property_type'].value_counts().nlargest(3)
sns.barplot(x=category_counts.index, y=category_counts.values)
plt.xlabel("Category")
plt.ylabel("Count")
plt.title("Top 3 Categories")
plt.show()
```

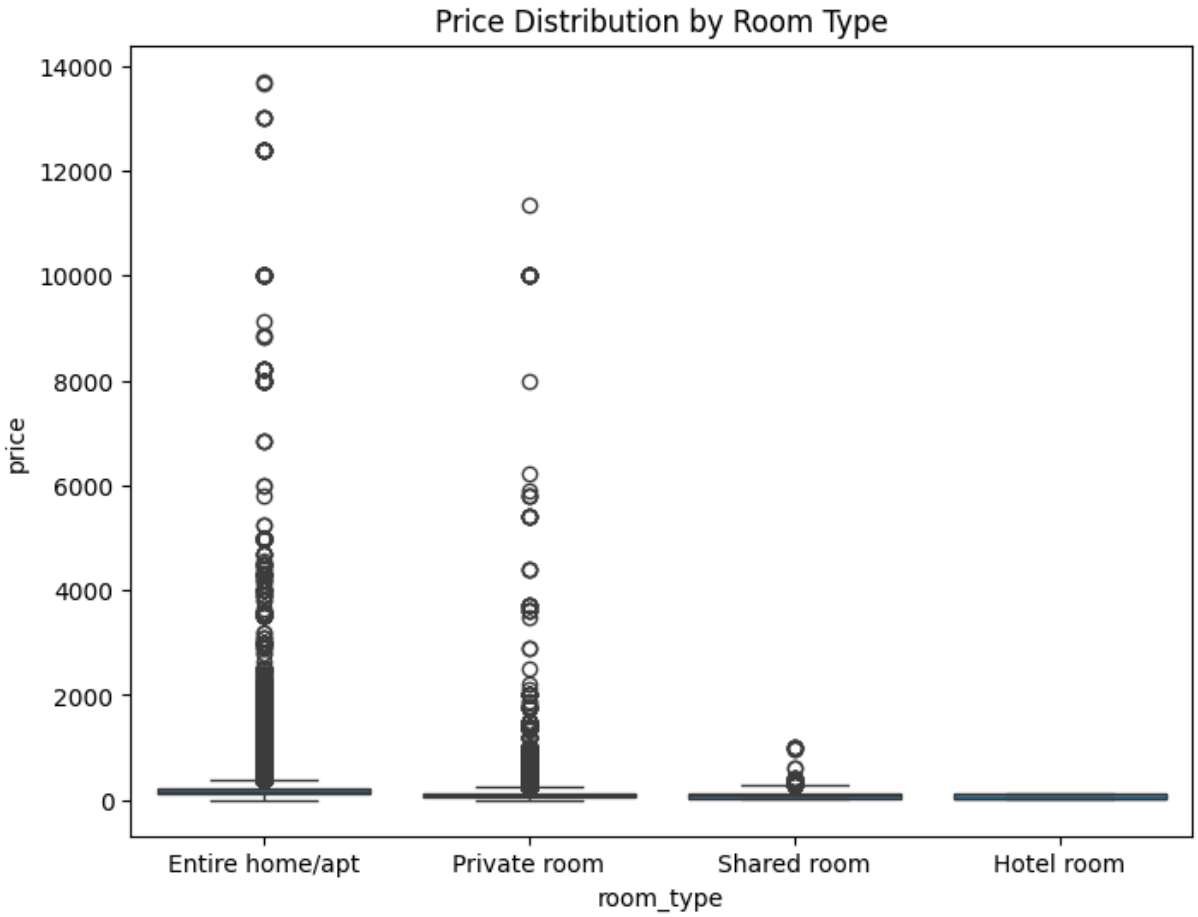


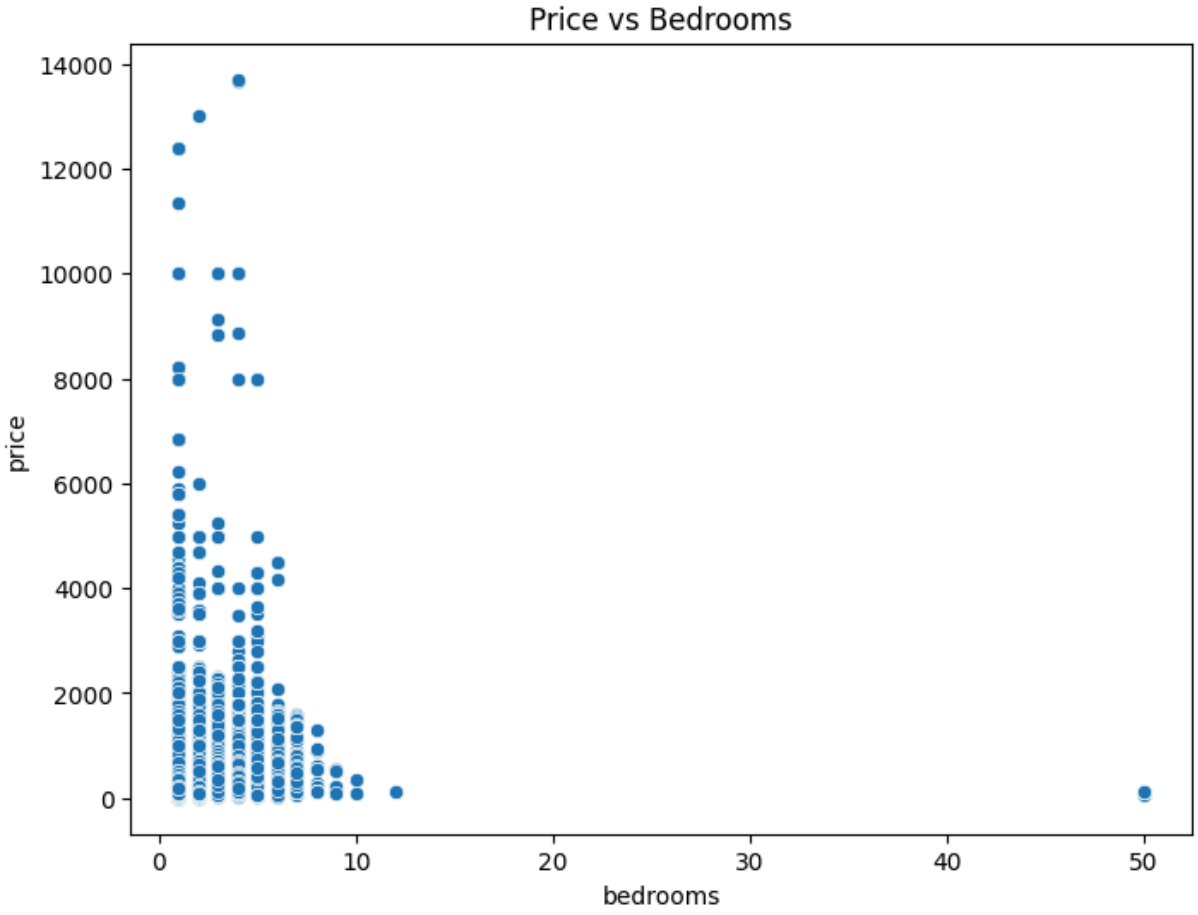
In [123…
```
sns.countplot(df['room_type'])
```

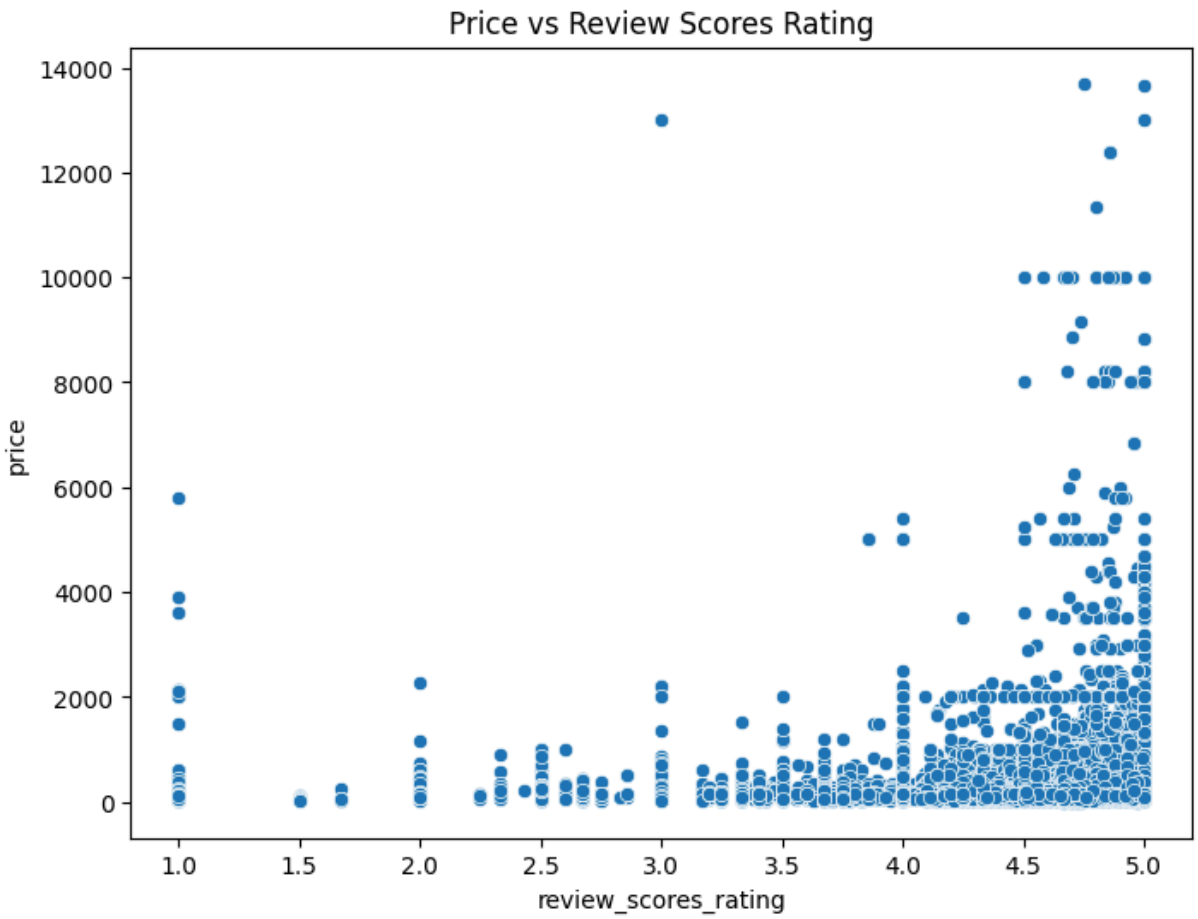
Out[123… <Axes: xlabel='count', ylabel='room_type'>



In [124…
```
# Room Type vs Price
plt.figure(figsize=(8, 6))
sns.boxplot(x='room_type', y='price', data=Toronto_df)
plt.title('Price Distribution by Room Type')
plt.show()

# Bedrooms vs Price
plt.figure(figsize=(8, 6))
sns.scatterplot(x='bedrooms', y='price', data=Toronto_df)
plt.title('Price vs Bedrooms')
plt.show()

# Review Scores Rating vs Price
plt.figure(figsize=(8, 6))
sns.scatterplot(x='review_scores_rating', y='price', data=Toronto_df)
plt.title('Price vs Review Scores Rating')
plt.show()
```


Price Distribution by Room Type
price
14000
12000
10000
8000
6000
4000
2000
0
Entire home/apt
Private room
Shared room
Hotel room
room_type


Price vs Bedrooms
price
14000
12000
10000
8000
6000
4000
2000
0
0
10
20
30
40
50
bedrooms

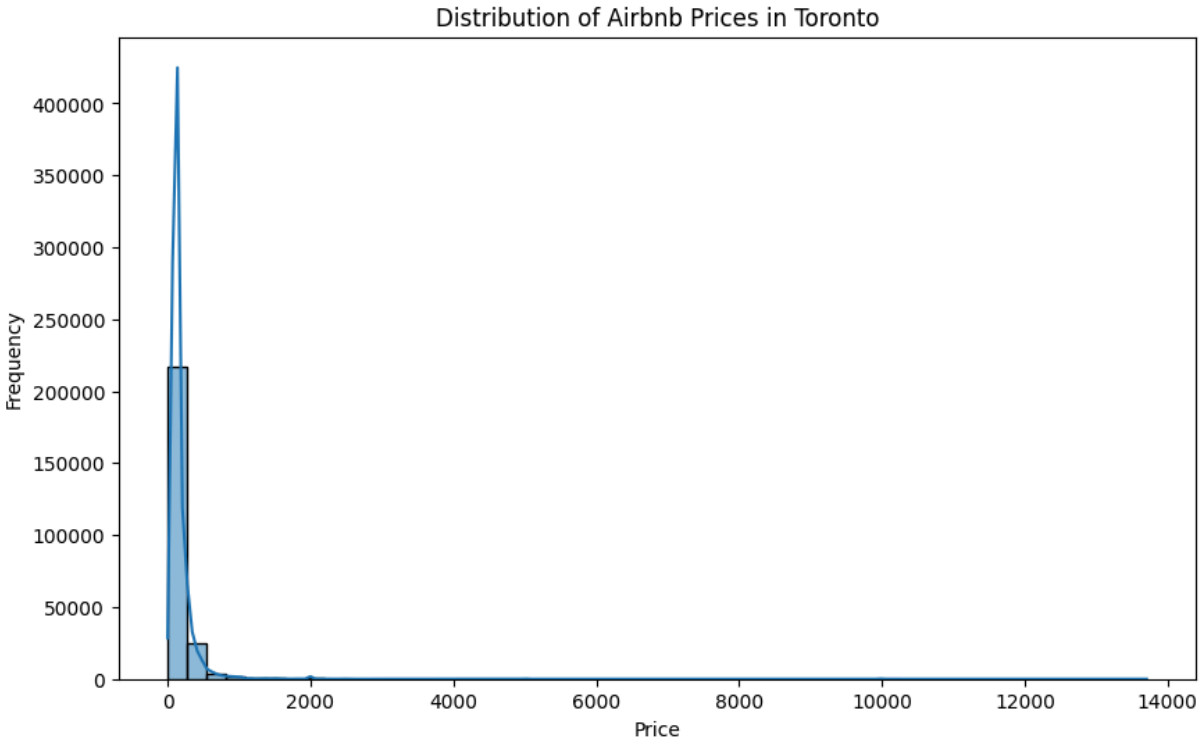


In [125…
```
Toronto_df['price'].skew()
```

Out[125…
```
np.float64(21.028222999712863)
```

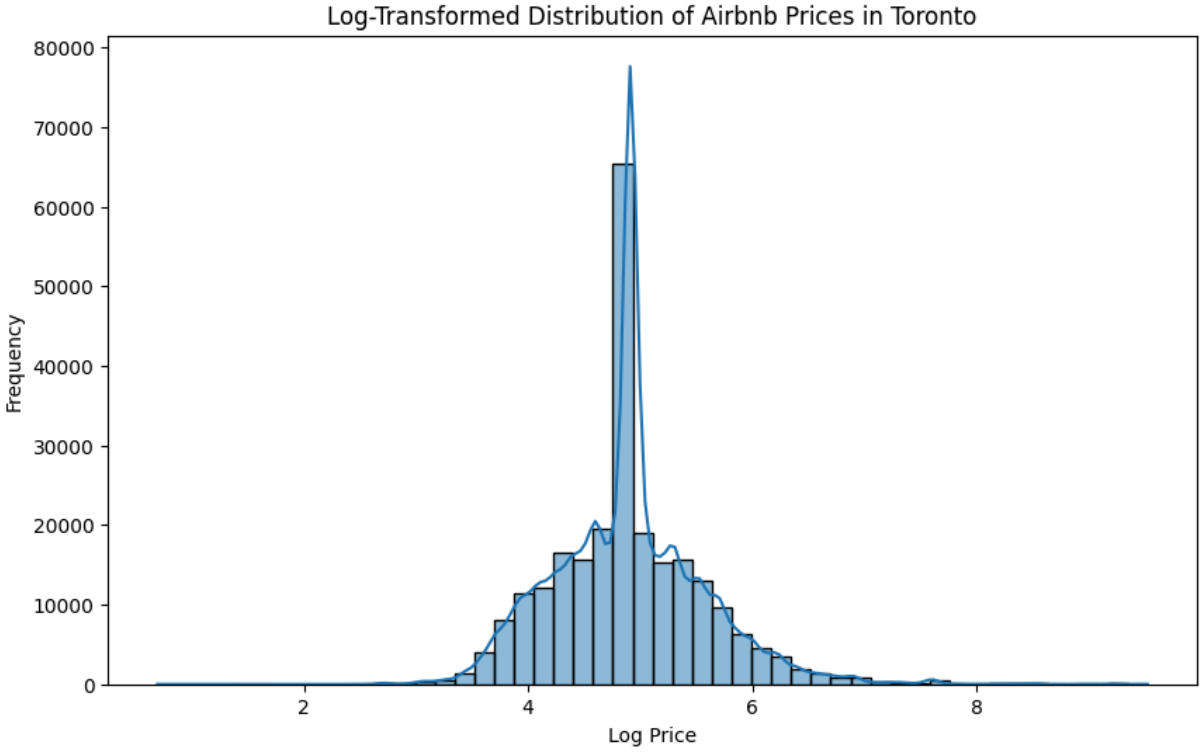
Data Transformation- Normalization

In [126…
```
# Plotting a histogram with KDE for price
plt.figure(figsize=(10,6))
sns.histplot(Toronto_df['price'].dropna(), kde=True, bins=50)
plt.title('Distribution of Airbnb Prices in Toronto')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```



In [127…

```
Toronto_df['log_price'] = np.log(Toronto_df['price'].dropna() + 1)

# Plot the log-transformed prices
plt.figure(figsize=(10, 6))
sns.histplot(Toronto_df['log_price'], kde=True, bins=50)
plt.title('Log-Transformed Distribution of Airbnb Prices in Toronto')
plt.xlabel('Log Price')
plt.ylabel('Frequency')
plt.show()
```



7. Visualizations and Findings

7.1. Guiding Question 1

What are the characteristics of Toronto's Airbnb market compared with other Canadian cities?

The raw data has mixed data types. Many numeric data are stored in the data as strings. And there are thousands-place as comma,"
0.00 " $as the null value. We processed these data into float type for analysis convenience. W$
10000 or
$0 per night. For some listings, the minimum nights per booking are over 1000 nights which is$
$. Thus we filtered our data and only remained the listings that are less than 365 days on mini$
0 and less than $10000 per night price. Tried to keep the data within a reasonable and truthful range.

In [128…
```
temp=df[['id','city','neighbourhood_group_cleansed','minimum_nights','maximum_night
```

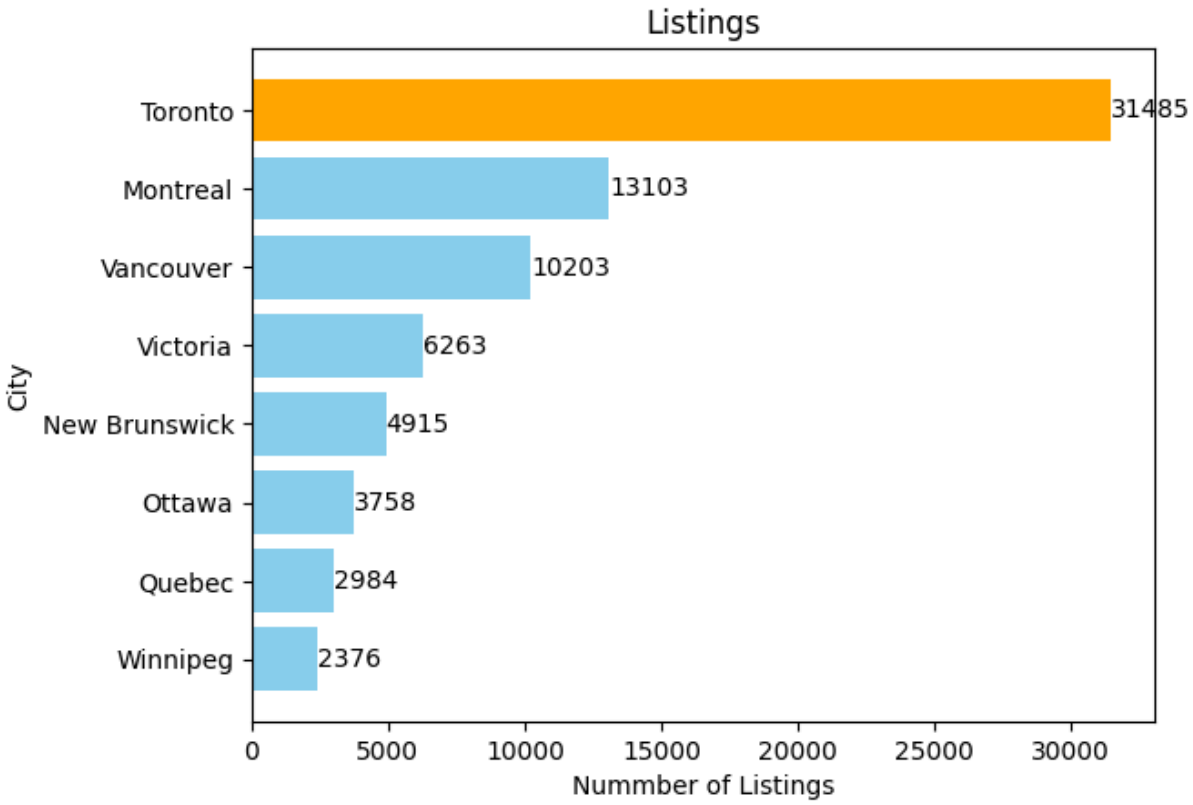
In [129…
```
for i in list(temp.columns)[8:]:
    if type(temp[i][0])!=str:
        temp.loc[temp[i]=='$0.00',i]=0.00
    else:
        temp[i]=temp[i].apply(lambda x: float(x.replace('$','').replace(',','')))
temp=temp[(temp['price']!=0)&(temp['minimum_nights']<=365)&(temp['price']<10000)&(t
temp.drop_duplicates('id',inplace=True)
```

Price and Listings

Listings

In [130…
```
list_com=temp.groupby('city')['id'].nunique().sort_values().reset_index()
colors = ['skyblue'] * len(list_com['id'])
colors[-1] = 'orange'

plt.barh(list_com['city'], list_com['id'],color=colors)


plt.title('Listings')
plt.xlabel('Nummber of Listings')
plt.ylabel('City')

for index, value in enumerate(list_com['id']):
```

```
    plt.text(value + 1, index, str(value), va='center')
plt.show()
```



First of all, Toronto has the highest number of listings in our data,there is a dramatic gap in the number of listings compared to other cities. This makes Toronto a huge market of Airbnb. For the customers, Toronto offers more choices.

Prices

In [131…

```
##price comparing
price_com=temp.groupby('city')['price'].mean().sort_values().reset_index()
price_com['price']=price_com['price'].round(2)
colors = ['skyblue'] * len(price_com['price'])
colors[-3] = 'orange'

plt.barh(price_com['city'], price_com['price'],color=colors)

plt.title('Listing Price ')
plt.xlabel('Average Price')
plt.ylabel('City')

for index, value in enumerate(price_com['price']):
    plt.text(value + 1, index, str(value), va='center')

plt.show()

##price comparing
price_com=temp.groupby('city')['price'].median().sort_values().reset_index()
```

```
price_com['price']=price_com['price'].round(2)
colors = ['skyblue'] * len(price_com['price'])
colors[-5] = 'orange'

plt.barh(price_com['city'], price_com['price'],color=colors)

plt.title('Listing Price')
plt.xlabel('Median Price')
plt.ylabel('City')

for index, value in enumerate(price_com['price']):
    plt.text(value + 1, index, str(value), va='center')
```

In [132…
```
print("The average price of all the cities is {}".format(round(temp['price'].mean()
print("The median price of all the cities is {}".format(temp['price'].median()))
```

```
The average price of all the cities is 179.12
The median price of all the cities is 135.0
```

In terms of price, Toronto's huge market doesn't breed exorbitant prices. Both average price and median price of Toronto are at average levels.

In [133…
```
print("The average price of all the cities except Toronto is {}".format(round(temp[
print("The median price of all the cities except Toronto is {}".format(temp[temp['c
```

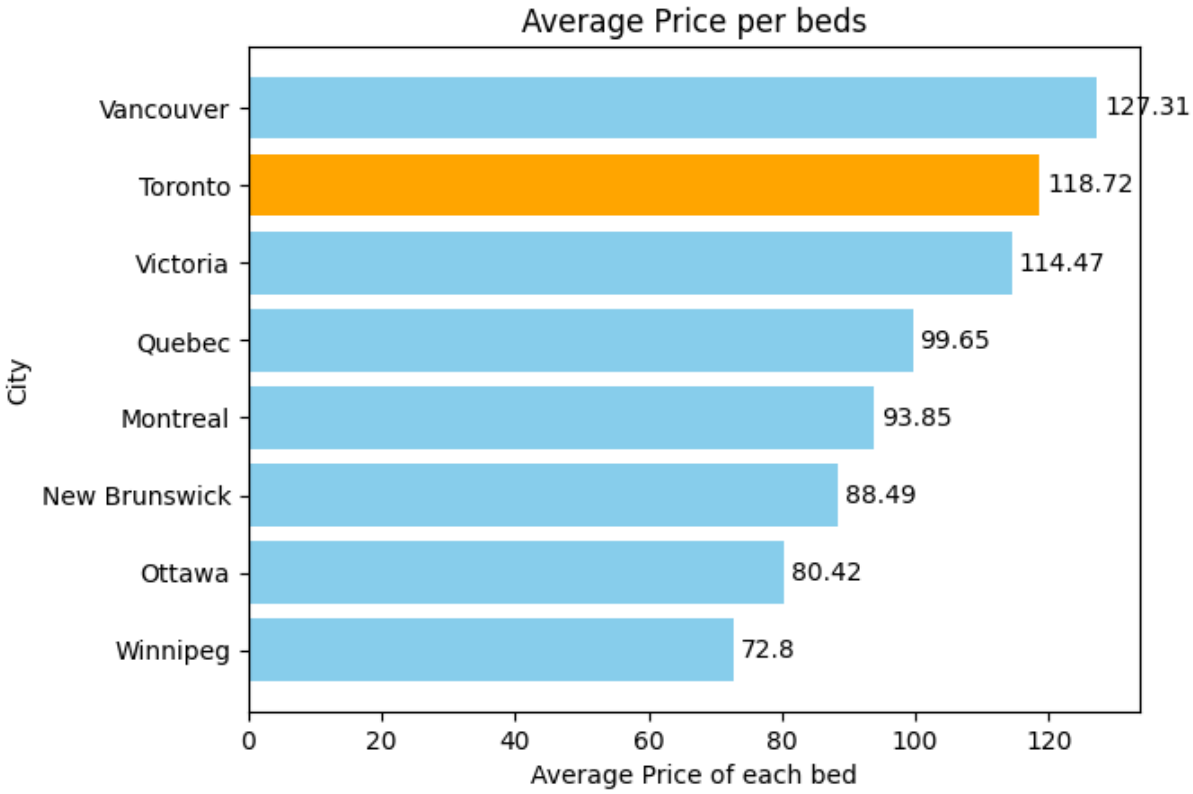
```
The average price of all the cities except Toronto is 178.28
The median price of all the cities except Toronto is 135.0
```

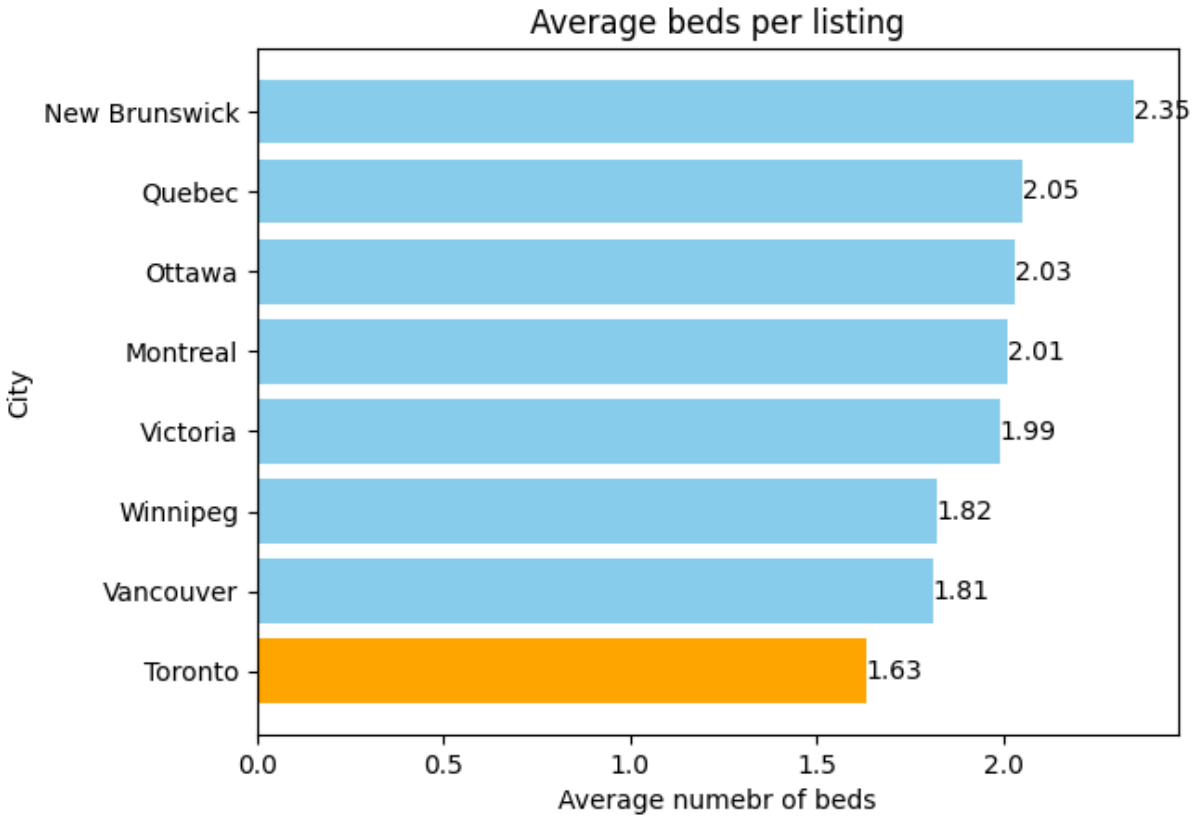
Of course, Toronto's large market share will affect the overall price. But when we take Toronto out and calculate the average and median market prices, we won't find much of a difference. It can be seen that the average price of listings in Toronto is indeed at the average level of the overall market.

Price of bed

In [134…
```
##Average price per bed
beds_p=temp[temp['beds']!=0]
beds_p['beds_rate']=beds_p['price']/beds_p['beds']
beds_p.groupby('city')['beds_rate'].mean()
beds_com=round(beds_p.groupby('city')['beds_rate'].mean().sort_values(),2).reset_in
colors = ['skyblue'] * len(beds_com['beds_rate'])
```

```
colors[-2] = 'orange'
plt.barh(beds_com['city'], beds_com['beds_rate'],color=colors)
plt.title('Average Price per beds')
plt.xlabel('Average Price of each bed')
plt.ylabel('City')
for index, value in enumerate(beds_com['beds_rate']):
    plt.text(value + 1, index, str(value), va='center')
plt.show()
## Average bed per listing
beds_c=round(beds_p.groupby('city')['beds'].mean().sort_values(),2).reset_index()
colors = ['skyblue'] * len(beds_c['beds'])
colors[0] = 'orange'
plt.barh(beds_c['city'], beds_c['beds'],color=colors)
plt.title('Average beds per listing')
plt.xlabel('Average numebr of beds')
plt.ylabel('City')
for index, value in enumerate(beds_c['beds']):
    plt.text(value , index, str(value), va='center')
plt.show()
```





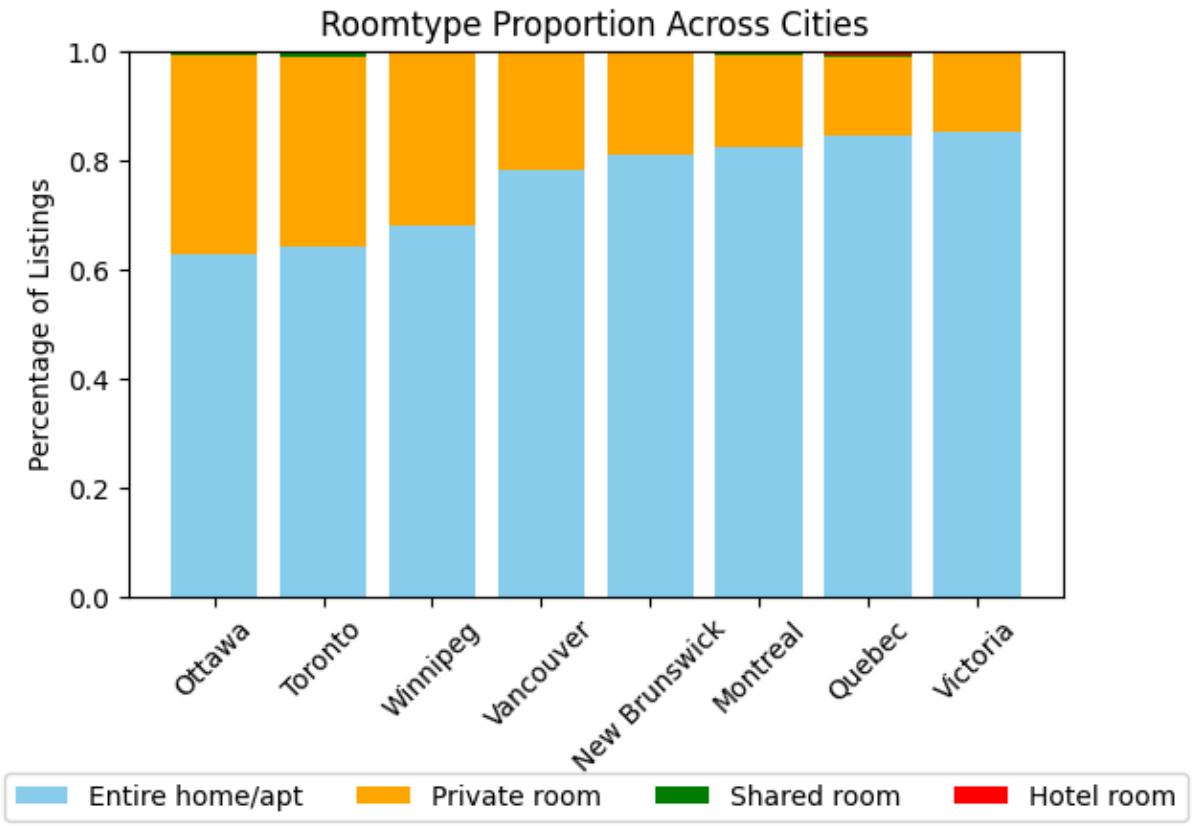
But facilities are also very important factors that affect the listing price, such as parking spaces, network, landscape, etc. In these facilities, we believe that capacity would be an important priority for users.We divide the number of beds offered in the listing by the price to get the price of each bed. We believe this metric would provide a different and accurate information. From the plot, we genurated, we can see the Average price per bed of Toronto is pretty high, which is not the same as the previous result of average price of the listings. We also found that Toronto has the lowest average number of beds per listing.

RoomType

In [135…

```
##Room type
roomtype=temp.groupby(['room_type','city'])['id'].nunique().reset_index()
##Stacked histogram for the Roomtype Proportion.
df_roomtype=roomtype.pivot_table(index='city', columns='room_type', values='id').re

df_roomtype['Total'] = df_roomtype[['Entire home/apt', 'Hotel room', 'Private room'

df_roomtype['Entire home/apt'] = df_roomtype['Entire home/apt'] / df_roomtype['Tota
df_roomtype['Hotel room'] = df_roomtype['Hotel room'] / df_roomtype['Total']
df_roomtype['Private room'] = df_roomtype['Private room'] / df_roomtype['Total']
df_roomtype['Shared room'] = df_roomtype['Shared room'] / df_roomtype['Total']

df_sorted =df_roomtype.sort_values('Entire home/apt')

citys = np.arange(len(df_sorted['city']))
```

```
plt.bar(citys , df_sorted['Entire home/apt'],label='Entire home/apt', color='skyblu
plt.bar(citys , df_sorted['Private room'],bottom=df_sorted['Entire home/apt'], labe
plt.bar(citys , df_sorted['Shared room'],bottom=df_sorted['Entire home/apt'] + df_s
plt.bar(citys, df_sorted['Hotel room'],bottom=df_sorted['Entire home/apt'] + df_sor

plt.ylabel('Percentage of Listings')
plt.title('Roomtype Proportion Across Cities')
plt.xticks(citys, df_sorted['city'], rotation=45)

plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.3), ncol=4)


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

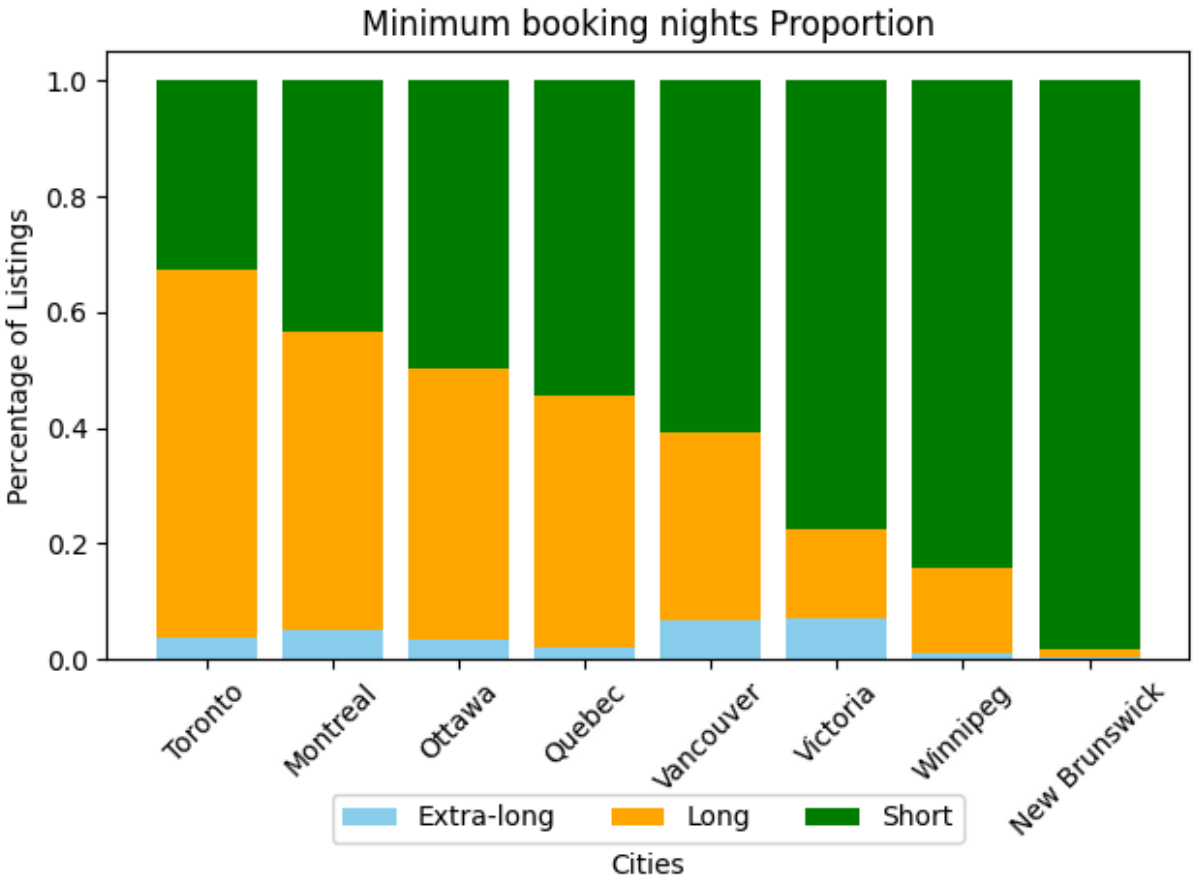


The data of roomtype also can support the point above, comparing to other cities, Entire home/apt room type occupied only around 60% of all listings in Toronto, only higher than Ottawa. Other room types which are more likely to have smaller capacity, the private room, shared room, Hotel room, have a higher proportion than other cities.

In [136…

```
##define the short term, long term, extra-long term booking requirements
temp.loc[temp['minimum_nights']<10,'term']='short'
temp.loc[(temp['minimum_nights']>=10)&(temp['minimum_nights']<50),'term']='long'
temp.loc[temp['minimum_nights']>=50,'term']='extra-long'

rent_term=temp.groupby(['city','term'])['id'].nunique().reset_index()
df_term=rent_term.pivot_table(index='city', columns='term', values='id').reset_inde

df_term['Total'] = df_term[['extra-long', 'long', 'short']].sum(axis=1)
```

```
df_term['extra-long'] = df_term['extra-long'] / df_term['Total']
df_term['long'] = df_term['long'] / df_term['Total']
df_term['short'] = df_term['short'] / df_term['Total']
df_term_sorted = df_term.sort_values(by='short')
citys = np.arange(len(df_term_sorted['city']))
plt.bar(citys, df_term_sorted['extra-long'],label='Extra-long', color='skyblue')
plt.bar(citys, df_term_sorted['long'], bottom=df_term_sorted['extra-long'], label='
plt.bar(citys, df_term_sorted['short'], bottom=df_term_sorted['extra-long'] + df_te
plt.xlabel('Cities')
plt.ylabel('Percentage of Listings')
plt.title('Minimum booking nights Proportion')
plt.xticks(citys, df_term_sorted['city'], rotation=45)
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.2), ncol=3)
plt.tight_layout()
plt.show()
```



In Airbnb, the host is able to limit the minimum nights for each booking. This is directly related to the user's travel plans and will also impact on the host's profit. We roughly seperate the minimum booking nights data into three groups, "Short" represent less than 10 nights ,"Long" represent above and euqal 10 nights and less than 50 nights,"Extra-long" represent above and equal 50 nights.
We got Toronto has the largest propotion of Long term booking limitation, Which means the customers in Toronto usually need to have more nights for each booking.

Overall, above data is a relatively clear reflection of the price situation of Airbnb Toronto. Although Toronto's property prices are only average compared to other cities, the capacity

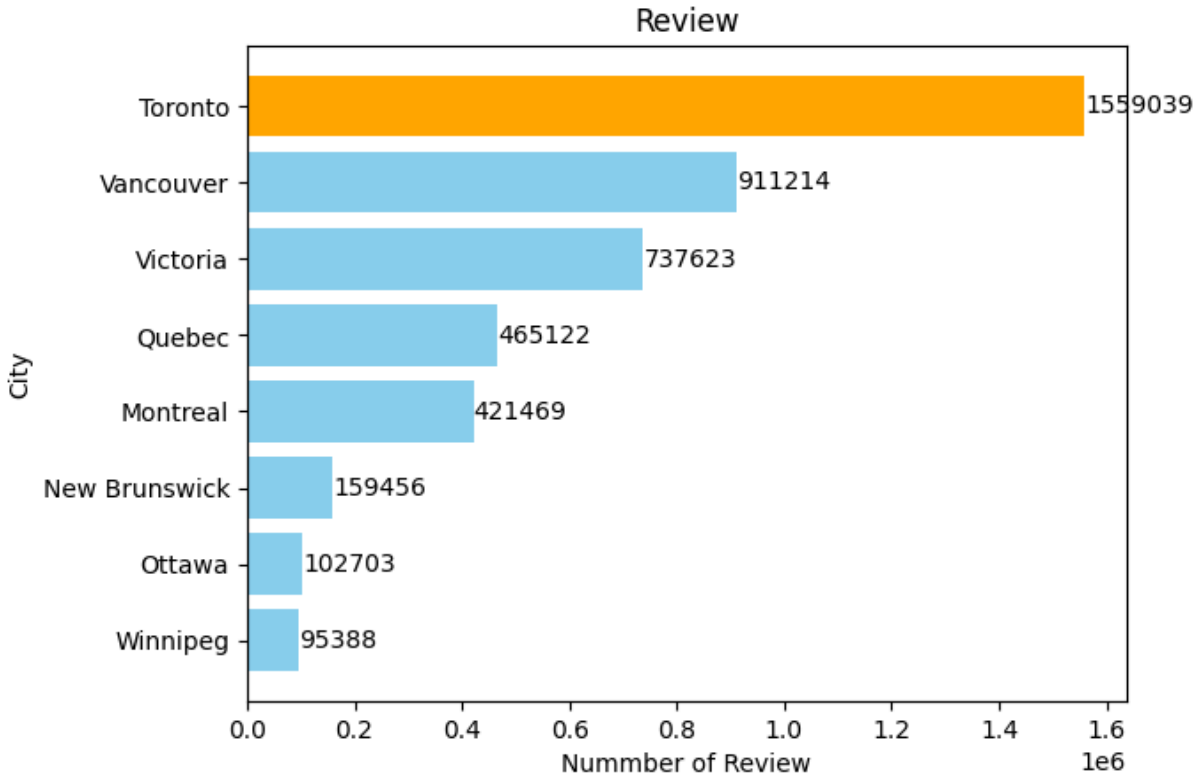
of the listings is lower than in other cities. Thus the price of each bed in Toronto is relativly higher. And the minimum nights requirement for each booking is higher than all other cities. All of these terms will make the cost of booking in Toronto higher than the price they set. But when the customer's needs perfectly fits the characteristics, which has one or two people and need to stay around one month, Toronto is more cost-effective.

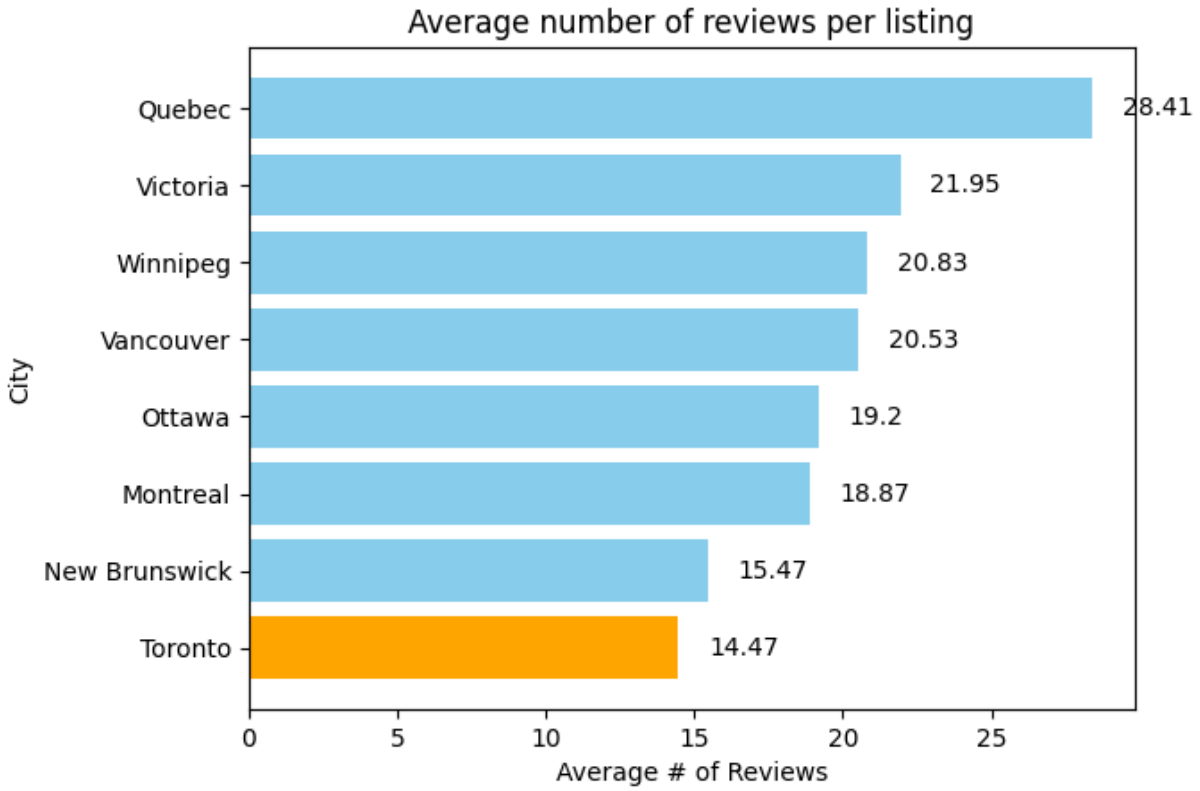
Reviews

We've tried to get the Airbnb's booking data, and expected to discover the vitalities of the market. This type of data is usually confidential. Thus we chose to use the review data as an alternative.

In [137…
```
df_reviews=pd.read_csv('reviews.csv')
##Filtering date to after 2023-06
df_reviews=df_reviews[df_reviews['date']>='2023-06']
```

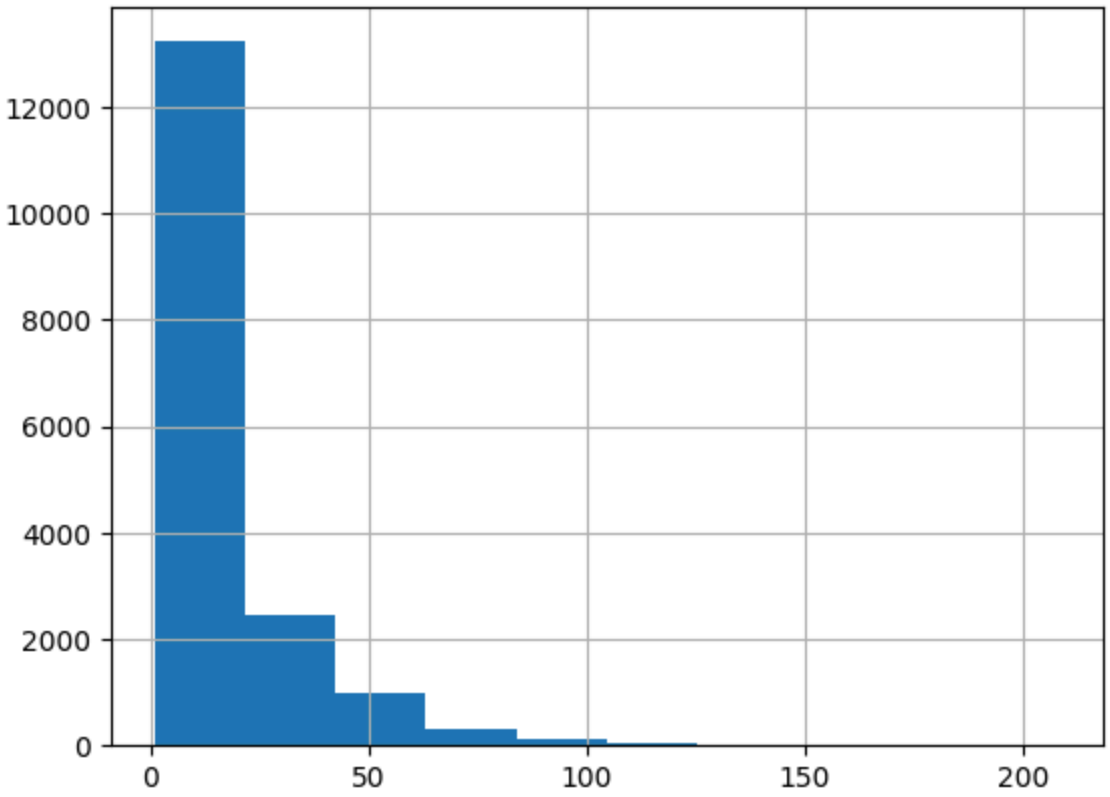
In [138…
```
##Total reviews each city
comments_total=df_reviews.groupby(['city'])['comments'].count().reset_index()
comments_total=comments_total.sort_values('comments')
colors = ['skyblue'] * len(comments_total['comments'])
colors[-1] = 'orange'

plt.barh(comments_total['city'], comments_total['comments'],color=colors)


plt.title('Review')
plt.xlabel('Nummber of Review')
plt.ylabel('City')

for index, value in enumerate(comments_total['comments']):
    plt.text(value + 1, index, str(value), va='center')
plt.show()

##Average reviews per listing
comments=round(df_reviews.groupby(['listing_id','city'])['comments'].nunique().rese
comments=comments.sort_values('comments')
colors = ['skyblue'] * len(comments['comments'])
colors[0] = 'orange'

plt.barh(comments['city'], comments['comments'],color=colors)


plt.title('Average number of reviews per listing')
plt.xlabel('Average # of Reviews')
plt.ylabel('City')

for index, value in enumerate(comments['comments']):
    plt.text(value + 1, index, str(value), va='center')
plt.show()
```





In [139…
```
#distribution of number of reviews for each listings
df_reviews[df_reviews['city']=='Toronto'].groupby('listing_id')['comments'].nunique
```

Out[139… <Axes: >



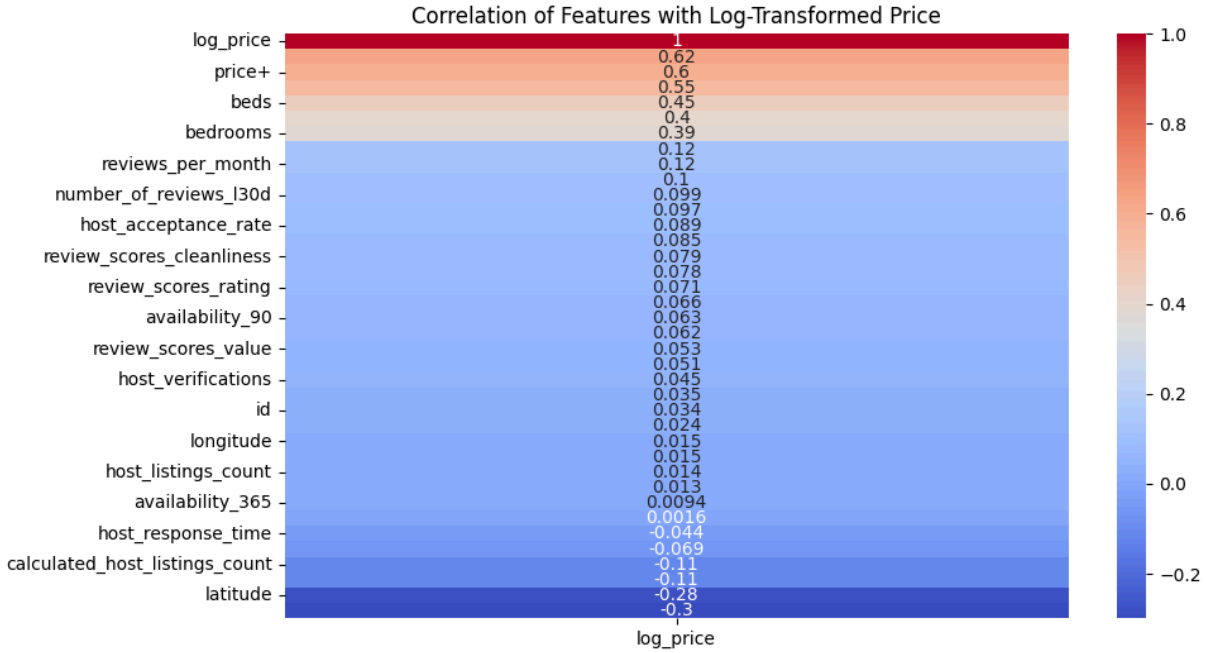
We found that Toronto has a much higher number of reviews than other cities. More reviews also means more bookings. This means that Toronto's market is huge and very active. But when we calculate the average number of reviews per listing, Toronto is the city that has least average number of reviews per listing.The most likely reason for this is that the vast majority of listings have a very small number of reviews, and only some of the most popular listings have a large number of reviews. This uneven distribution means that Toronto's market, while active, is competitive and harsh.

7.2. Guiding Question 2

Which factors have the most significant impact on the pricing of Airbnb listings in Toronto?

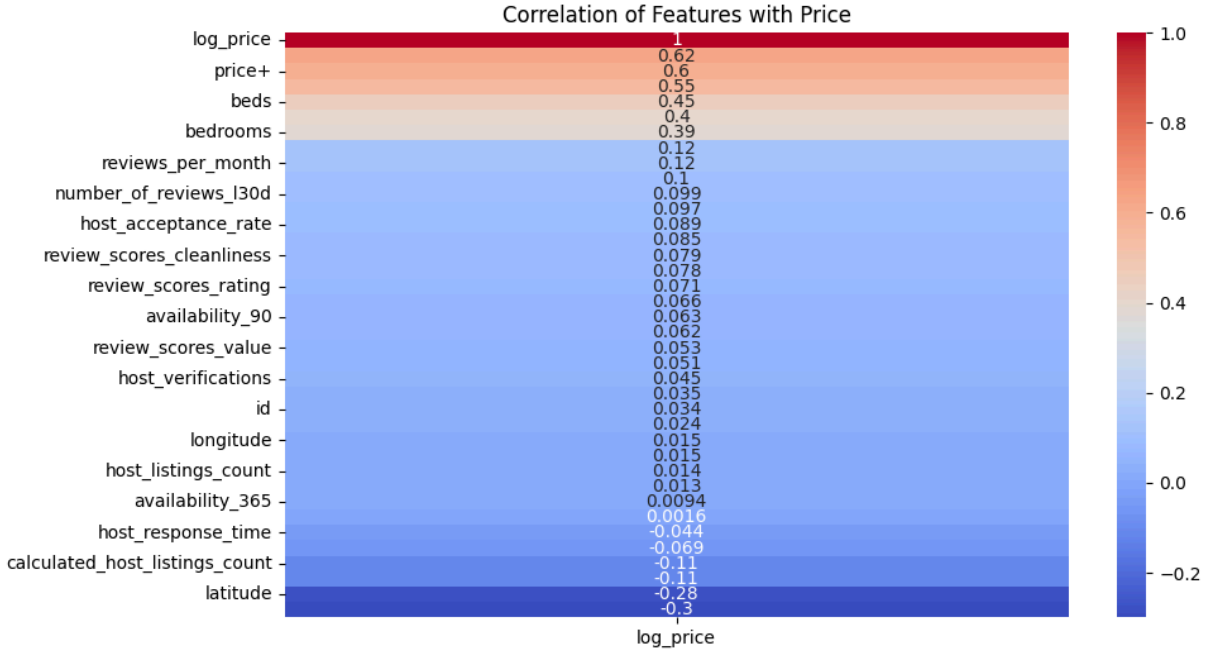
In [140…

```
numeric_df = Toronto_df.select_dtypes(include=['number'])

# Compute the correlation matrix
corr_matrix = numeric_df.corr()

# Visualize correlations with the log_price column
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix[['log_price']].sort_values(by='log_price', ascending=False)
plt.title('Correlation of Features with Log-Transformed Price')
plt.show()
```



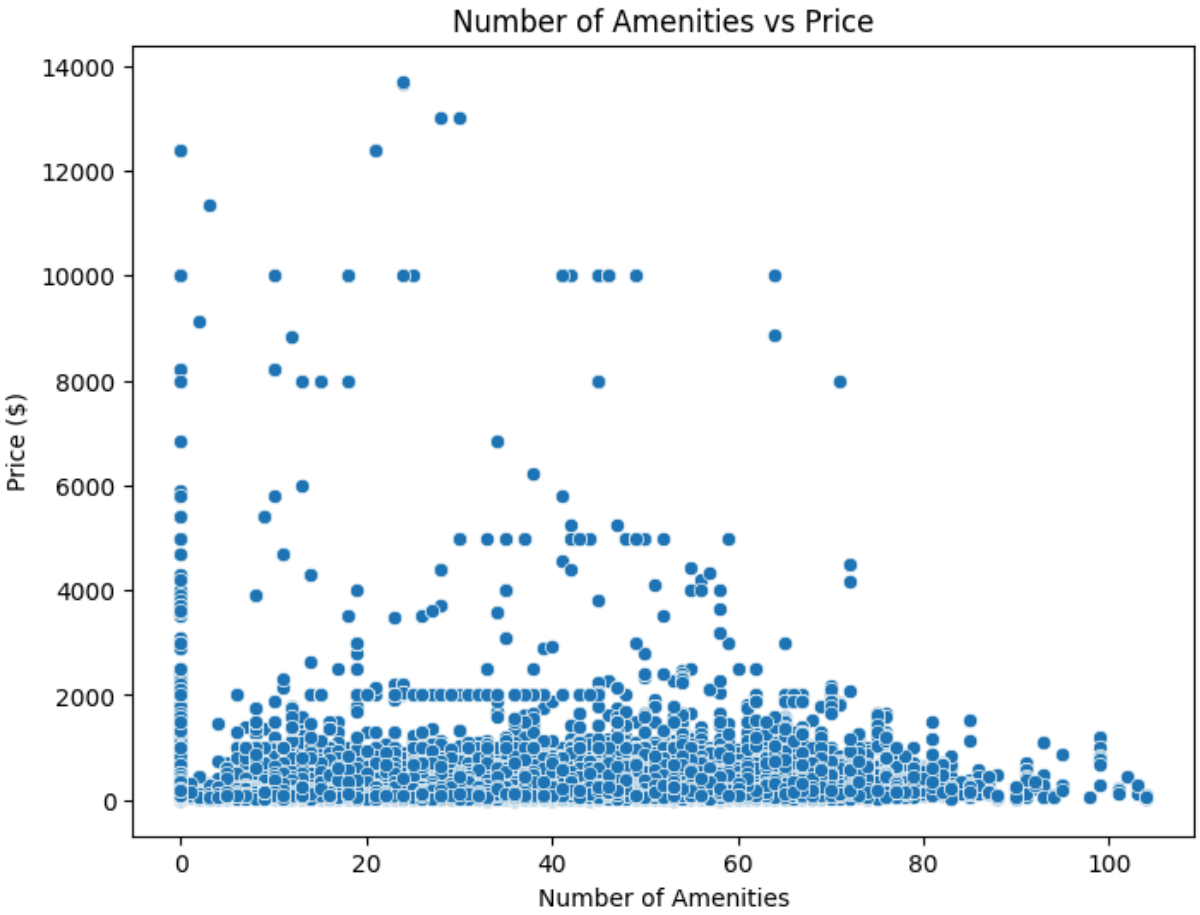
In [141…
```
df['room_type'] = df['room_type'].replace({'Entire home/apt': 0, 'Private room': 1,
```

In [142…
```
corr_matrix = numeric_df.corr()

# Visualize correlations with the price column
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix[['log_price']].sort_values(by='log_price', ascending=False)
plt.title('Correlation of Features with Price')
plt.show()
```



In [143…
```
Toronto_df['num_amenities'] = Toronto_df['amenities'].apply(lambda x: len(eval(x))


plt.figure(figsize=(8, 6))
```

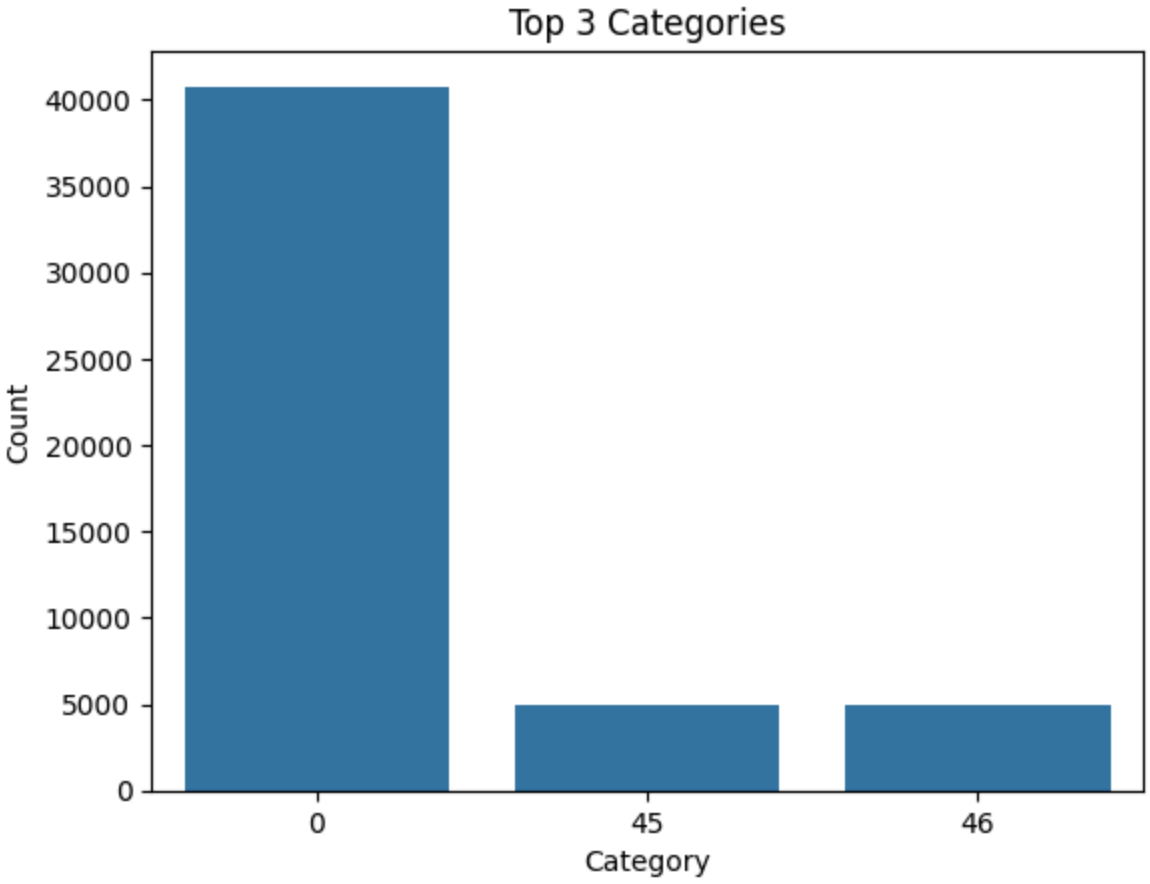
```
sns.scatterplot(x='num_amenities', y='price', data=Toronto_df)
plt.title('Number of Amenities vs Price')
plt.xlabel('Number of Amenities')
plt.ylabel('Price ($)')
plt.show()

correlation = Toronto_df['num_amenities'].corr(Toronto_df['price'])
print(f"Correlation between number of amenities and price: {correlation}")
```




```
Correlation between number of amenities and price: 0.08296634984559231
```

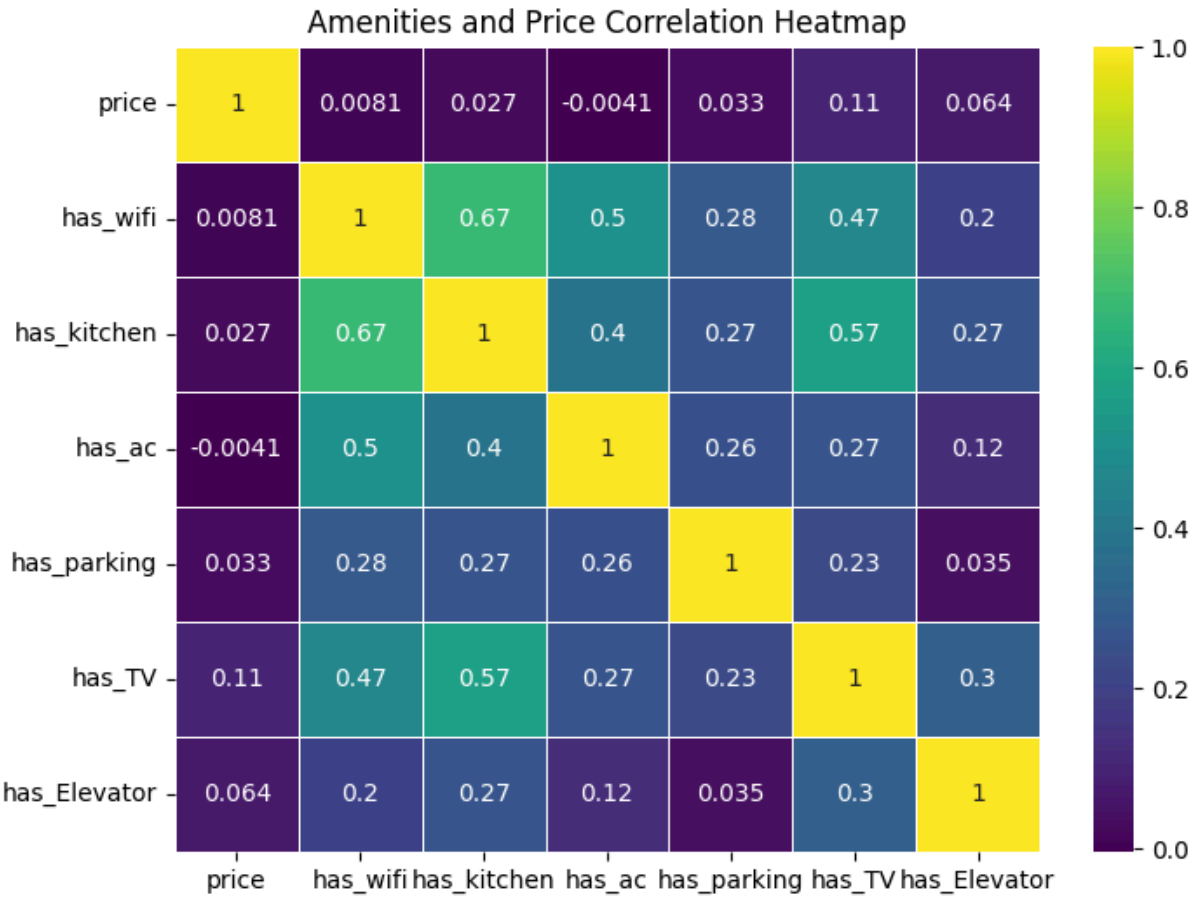
In [144…

```
category_counts = Toronto_df['num_amenities'].value_counts().nlargest(3)
sns.barplot(x=category_counts.index, y=category_counts.values)
plt.xlabel("Category")
plt.ylabel("Count")
plt.title("Top 3 Categories")
plt.show()
```



In [145…

```
Toronto_df['has_kitchen'] = Toronto_df['amenities'].apply(lambda x: 1 if 'Kitchen'
Toronto_df['has_ac'] = Toronto_df['amenities'].apply(lambda x: 1 if 'Air conditioni
Toronto_df['has_parking'] = Toronto_df['amenities'].apply(lambda x: 1 if 'Free park
Toronto_df['has_wifi'] = Toronto_df['amenities'].apply(lambda x: 1 if 'Wifi' in x e
Toronto_df['has_TV'] = Toronto_df['amenities'].apply(lambda x: 1 if 'TV' in x else
Toronto_df['has_Elevator'] = Toronto_df['amenities'].apply(lambda x: 1 if 'Elevator

amenities_data = Toronto_df[['price', 'has_wifi', 'has_kitchen', 'has_ac', 'has_par

plt.figure(figsize=(8, 6))
sns.heatmap(amenities_data.corr(), annot=True, cmap='viridis', linewidths=0.5)
plt.title('Amenities and Price Correlation Heatmap')
plt.show()
```



In this guiding question, we examined the factors affecting the pricing of Airbnb listings in Toronto by visualizing bar charts for room type and property type. The data revealed that entire rental units are the most popular property type, with entire homes being the preferred room type, followed by private rooms as the second most popular choice. A scatter plot between review score ratings and price indicated that higher reviews do not consistently result in higher prices, apart from a few outliers. This suggests that price lacks a linear relationship with the number of reviews, though premium prices may occasionally align with high review scores.

To deepen our analysis, we used a heatmap to assess the correlation between price and other variables, hence accommodations, beds, and bedrooms, showed a modest positive relationship with price. However, these variables did not provide a clear picture of the most influential factor. As a result, we shifted focus to amenities, specifically basic features like WiFi, TV, AC, kitchen, elevator, and parking. Among these, TV emerged as the most impactful amenity affecting pricing.

In conclusion, none of the analyzed variables exhibit a strong correlation with pricing in this dataset. This suggests that external factors such as economic conditions, tourism demand, and the real estate market in specific areas may play a crucial role in shaping Airbnb pricing strategies.

7.3. Guiding Question 3

How does a year-over-year comparison account for changes in the Airbnb market in Toronto?

Gathering and printing variables that will be used for graphs and analysis from each dataset or certain months

August 2021:

In [146…
```
df2['last_scraped'] = pd.to_datetime(df2['last_scraped'])
df2['Month'] = df2['last_scraped'].dt.to_period('M')
average_price = df2['price'].mean()
unique_hosts_2021_toronto = df2.groupby('Month')['host_id'].nunique()
unique_listings_2021_toronto =  df2.groupby('Month')['id'].nunique()
average_minimum_nights_2021 = df2['minimum_nights'].mean()
print(average_minimum_nights_2021)
print(unique_listings_2021_toronto)
print(average_price)
```

```
27.427207637231504
Month
2021-08    15084
Freq: M, Name: id, dtype: int64
181.85965261203924
```

September 2020

In [147…
```
df3['last_scraped'] = pd.to_datetime(df3['last_scraped'])
df3['Month'] = df3['last_scraped'].dt.to_period('M')
unique_hosts_2020_toronto = df3.groupby('Month')['host_id'].nunique()
unique_listings_2020_toronto =  df3.groupby('Month')['id'].nunique()
average_price3 = df3['price'].mean()
average_minimum_nights_2020 = df3['minimum_nights'].mean()
df3['review_scores_rating'] = pd.to_numeric(df3['review_scores_rating'], errors='co


average_review_september_2020 = df3['review_scores_rating'].mean()


print(average_review_september_2020)
print(average_price3)
print(average_minimum_nights_2020)
print(unique_listings_2020_toronto)
```

```
94.30493004663558
141.27811559737376
9.988729773044513
Month
2020-09    19343
Freq: M, Name: id, dtype: int64
```

September 2024

In [148…
```
Toronto_df['Month'] = Toronto_df['last_scraped'].dt.month
september_2024 = Toronto_df[(Toronto_df['last_scraped'].dt.month == 9) & (Toronto_d
unique_hosts_2024_sept = september_2024.groupby('Month')['host_id'].nunique()
unique_listings_2024_sept = september_2024.groupby('Month')['id'].nunique()
average_price_september = september_2024['price'].mean()
average_minimum_september_2024 = september_2024['minimum_nights'].mean()
print(average_price_september)
print(average_minimum_september_2024)
```

```
181.61892325315006
24.41049255441008
```

August 2024

In [149…
```
august_2024 = Toronto_df[(Toronto_df['last_scraped'].dt.month == 8) & (Toronto_df['
unique_hosts_2024_aug = august_2024.groupby('Month')['host_id'].nunique()
unique_listings_2024_aug =  august_2024.groupby('Month')['id'].nunique()
average_price_august = august_2024['price'].mean()
average_minimum_august_2024 = august_2024['minimum_nights'].mean()
print(average_price_august)
print(average_minimum_august_2024)
```

```
185.0800498868308
24.334749872973347
```

October 2023

In [150…
```
october_2023 = Toronto_df[(Toronto_df['last_scraped'].dt.month == 10) & (Toronto_df
unique_hosts_2023_oct = october_2023.groupby('Month')['host_id'].nunique()
unique_listings_2023_oct = october_2023.groupby('Month')['id'].nunique()
average_price_october = october_2023['price'].mean()
average_minimum_october_2023 = october_2023['minimum_nights'].mean()
print(average_price_october)
print(average_minimum_october_2023)
```

```
194.29141374837872
25.397665369649804
```

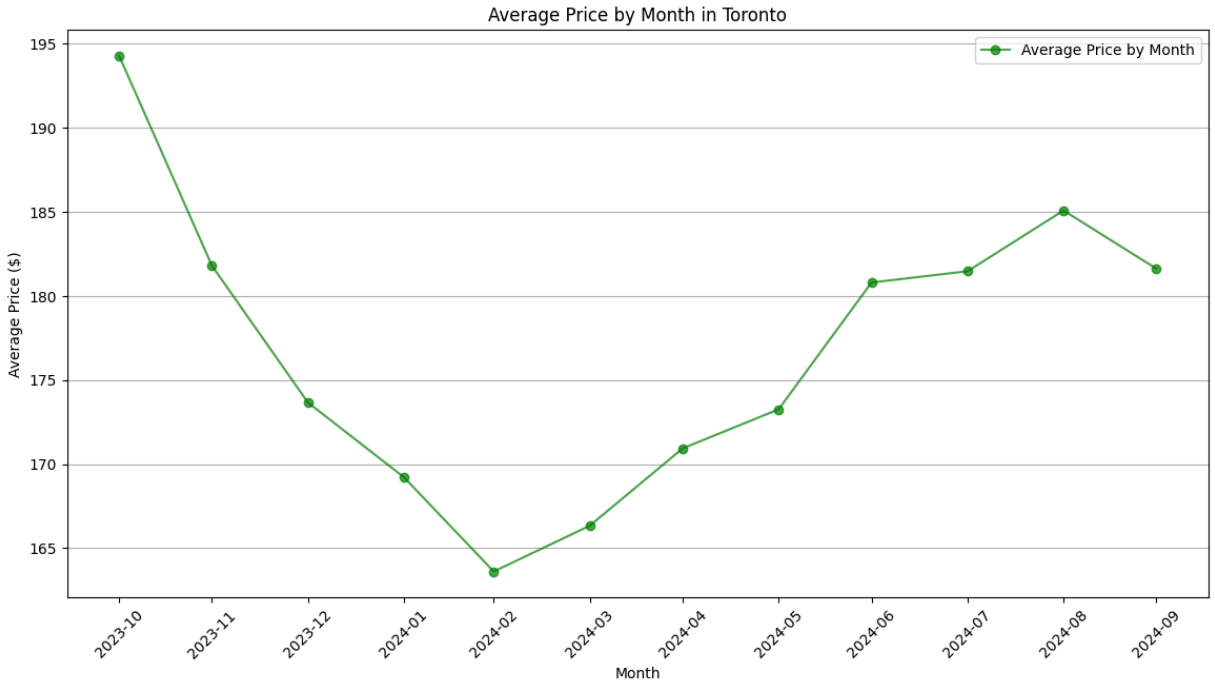
An important thing to note is that we only have data from October 2023 to September 2024 and from September 2020 to August 2021. It would have been nice to have data on 2022 or just all the months in all the years in general, like from October 2023 to September 2024.

Price Analysis

Graphing the average price in Toronto first by month from October 2023 to September 2024, and then in September 2020, August 2021, October 2023, and September 2024

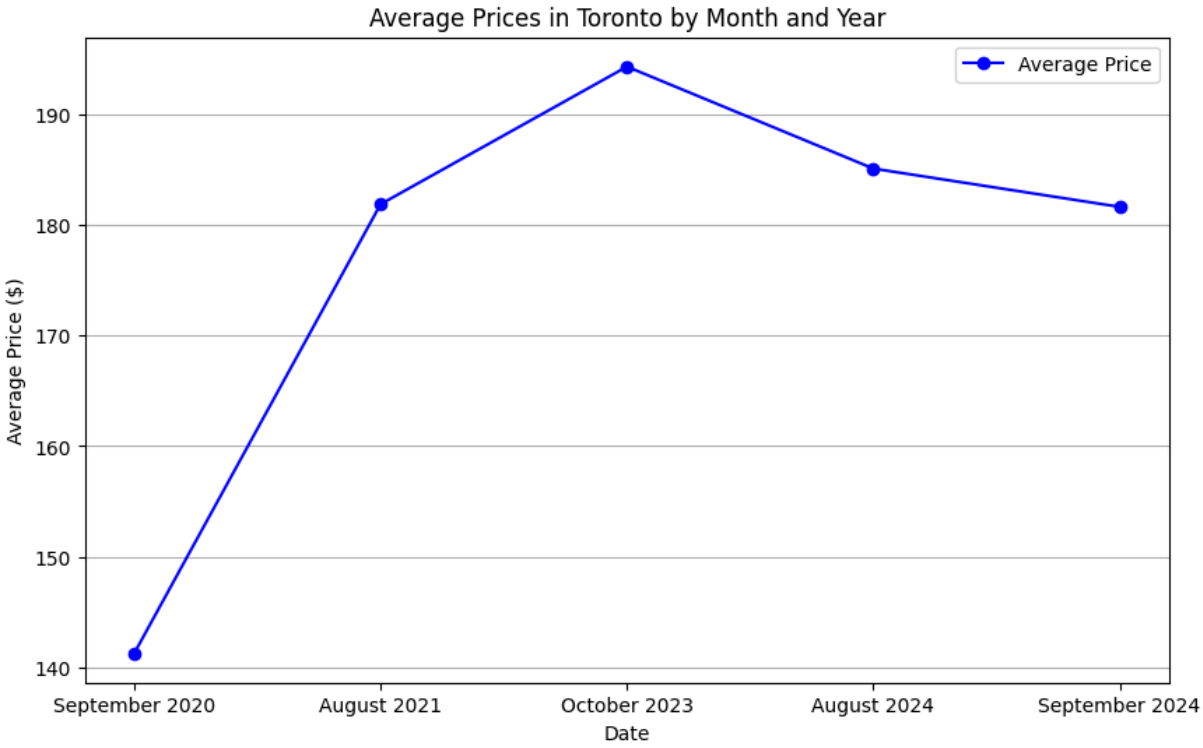
In [151…
```
#print(Toronto_df['last_scraped'])
#print(Toronto_df['id'])
Toronto_df.set_index('last_scraped', drop=False, inplace=True)
monthly_avg_price = Toronto_df['price'].resample('ME').mean()
```

```
plt.figure(figsize=(14, 7))
plt.plot(monthly_avg_price.index, monthly_avg_price, color='green', marker='o', lab
plt.title('Average Price by Month in Toronto')
plt.xlabel('Month')
plt.ylabel('Average Price ($)')
plt.xticks(monthly_avg_price.index, monthly_avg_price.index.strftime('%Y-%m'), rota
plt.legend()
plt.grid(axis='y')
plt.show()
```



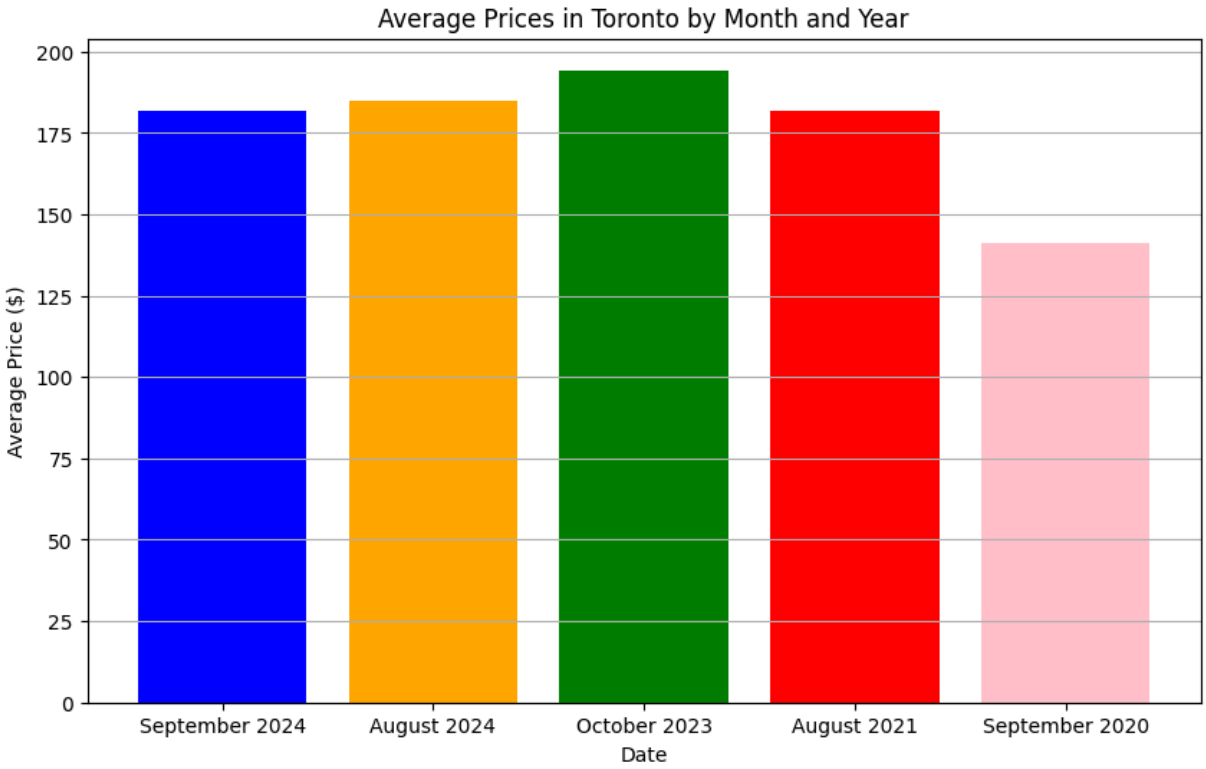
In [152…

```
average_prices = {
    'September 2024': average_price_september,
    'August 2024': average_price_august,
    'October 2023': average_price_october,
    'August 2021': average_price,
    'September 2020': average_price3}

sorted_prices = dict(sorted(average_prices.items(), key=lambda x: pd.to_datetime(x[

plt.figure(figsize=(10, 6))
plt.plot(sorted_prices.keys(), sorted_prices.values(), marker='o', color='blue', la
plt.title('Average Prices in Toronto by Month and Year')
plt.xlabel('Date')
plt.ylabel('Average Price ($)')
plt.grid(axis='y')
plt.legend()

plt.show()
```



In [153…

```
average_prices = {
    'September 2024': average_price_september,
    'August 2024': average_price_august,
    'October 2023': average_price_october,
    'August 2021': average_price,
    'September 2020': average_price3}


plt.figure(figsize=(10, 6))
plt.bar(average_prices.keys(), average_prices.values(), color=['blue', 'orange', 'g

plt.title('Average Prices in Toronto by Month and Year')
plt.xlabel('Date')
plt.ylabel('Average Price ($)')
plt.grid(axis='y')


plt.show()
```



We can see a large spike in average price from September 2020 to August 2021 from a little over $140 to 180$ as shown in the bar and more clearly in the line graph with the higher starting y-axis point. From there we can see the price continue to increase to around $195 in October 2023 with a bit of a drop from there to August and September 2024. When looking at the graph with all the months from October 2023 to September 2024 we see the peak average price was in October 2023. From there there was a steady decline to February 2024 and from there, an increase until August 2024 followed by a decrease to September 2024.

The main takeaway is that price did dramatically increase from 2020 to 2021 and since then it's been pretty steady. It was a bit surprising to see October 2023 as the peak for average prices and how low December and January were being holiday months. It was not as surprising to see the increase in summer months with another drop-off in September. Perhaps Toronto is more desired in the summer than wintertime, whether due to weather, holidays, people moving, or other factors.

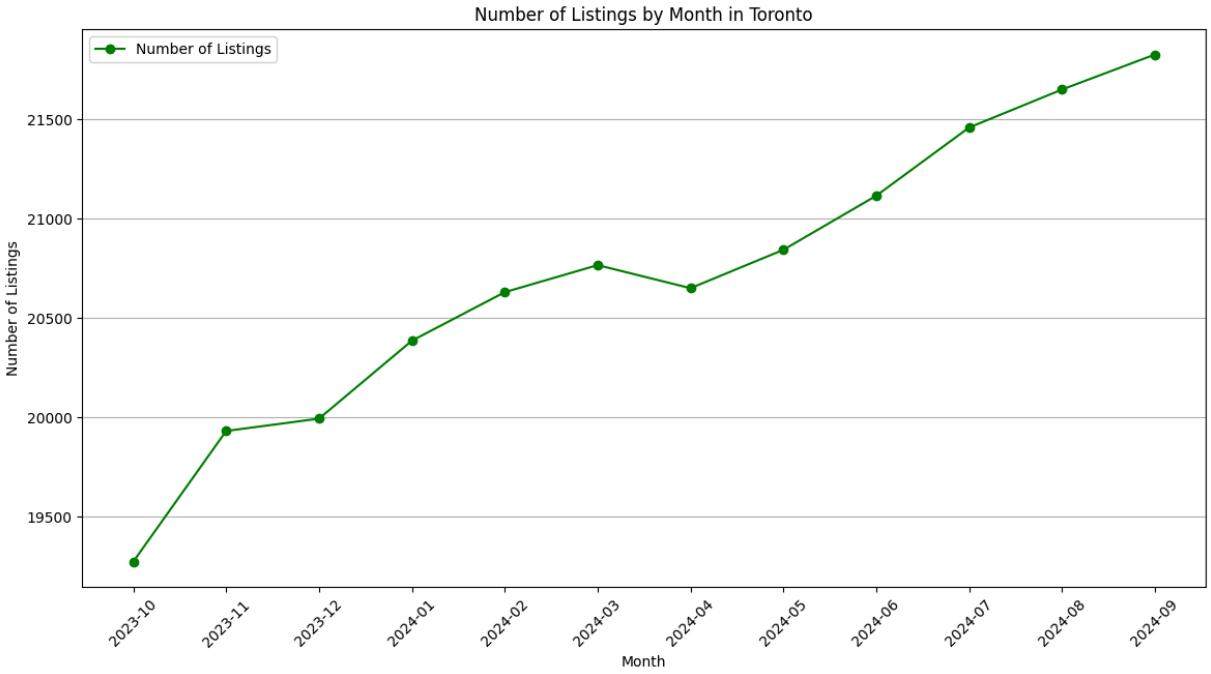
Host and Listings Analysis

Similar to price first we'll plot the number of listings and hosts by month from October 2023 to September 2024, as well as by year for the dates we have in 2020, 2021, and similar times in 2023 and 2024.

In [154…

```
Toronto_df['Month'] = Toronto_df['last_scraped'].dt.to_period('M')

unique_listings_by_month = Toronto_df.groupby('Month')['id'].nunique()

plt.figure(figsize=(14, 7))
```

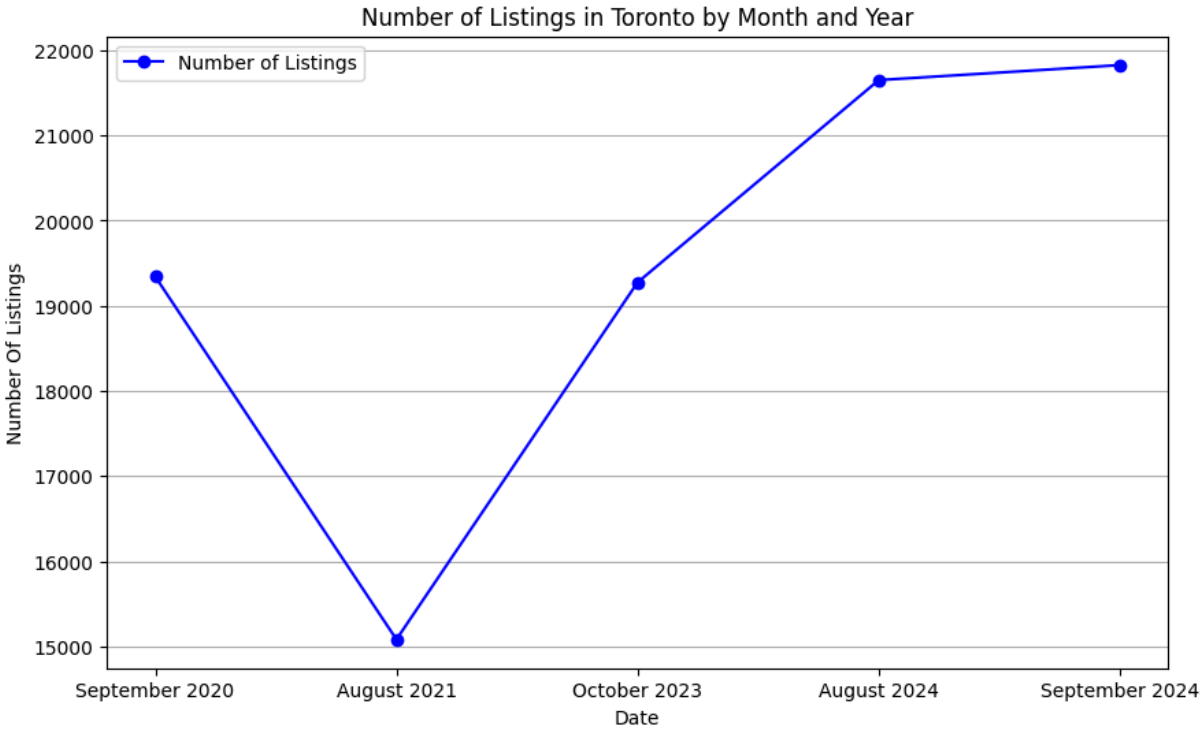
```
plt.plot(unique_listings_by_month.index.astype(str), unique_listings_by_month, colo

plt.title('Number of Listings by Month in Toronto')
plt.xlabel('Month')
plt.ylabel('Number of Listings')
plt.xticks(rotation=45)
plt.legend()
plt.grid(axis='y')
plt.show()
```



We can see a pretty steady increase in the number of listings from October 2023 to September 2024

In [155…

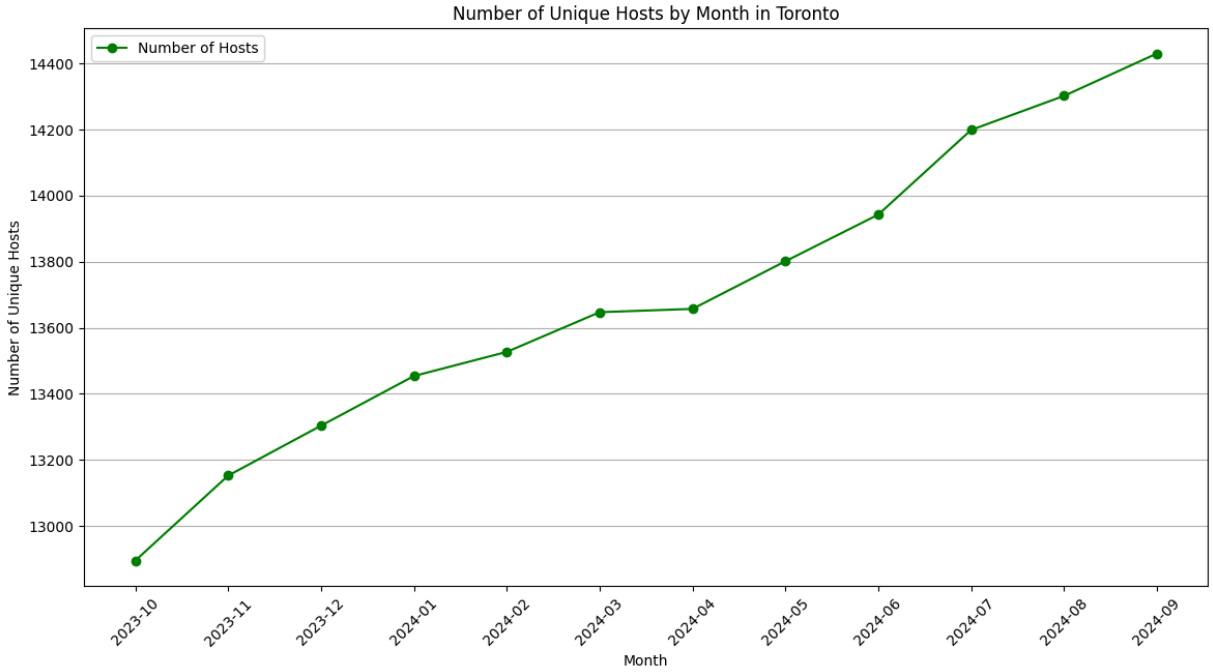
```
average_listings = {
    'September 2024': unique_listings_2024_sept,
    'August 2024': unique_listings_2024_aug,
    'October 2023': unique_listings_2023_oct,
    'August 2021': unique_listings_2021_toronto,
    'September 2020': unique_listings_2020_toronto
}

sorted_listings = dict(sorted(average_listings.items(), key=lambda x: pd.to_datetim

plt.figure(figsize=(10, 6))
plt.plot(sorted_listings.keys(), sorted_listings.values(), marker='o', color='blue'
plt.title('Number of Listings in Toronto by Month and Year')
plt.xlabel('Date')
plt.ylabel('Number Of Listings')
plt.grid(axis='y')
plt.legend()
plt.show()
```



We see a pretty major drop in listings from September 2020 to August 2021, likely due to covid, and from there an increase similar to the previous graph from 2021 to September 2024

In [156…

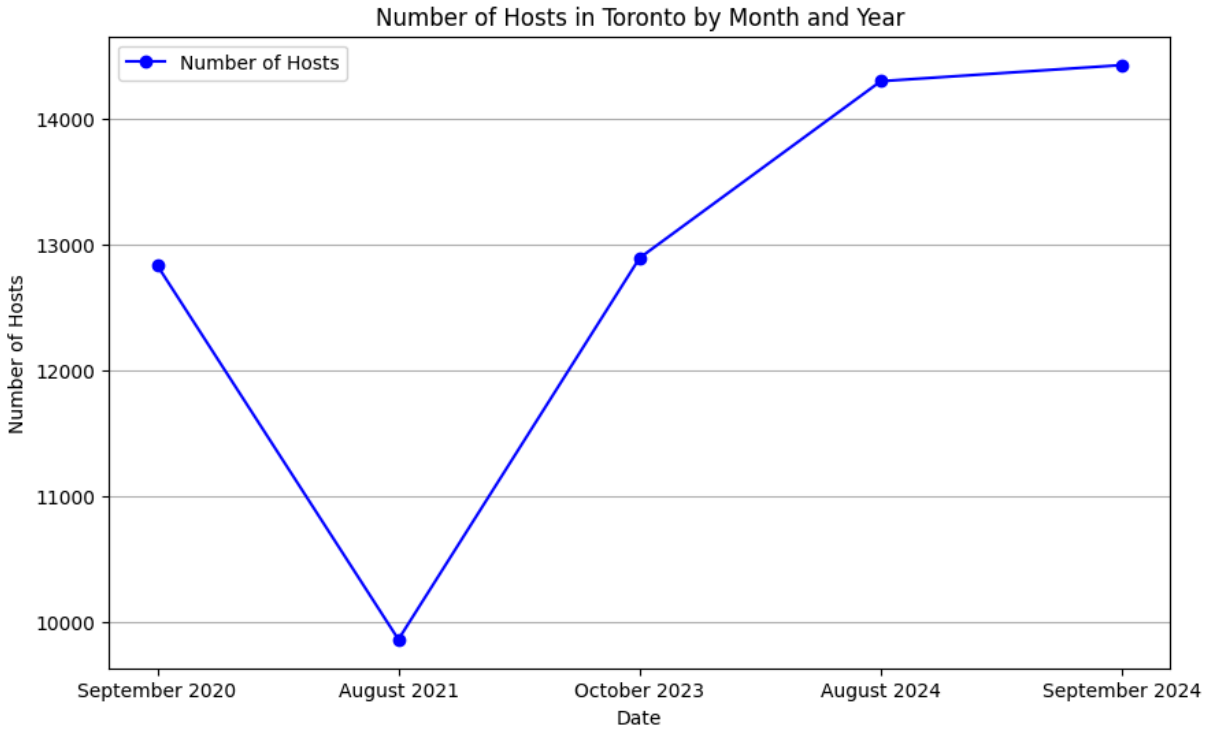
```
unique_hosts_by_month = Toronto_df.groupby('Month')['host_id'].nunique()

plt.figure(figsize=(14, 7))
plt.plot(unique_hosts_by_month.index.astype(str), unique_hosts_by_month, color='gre

plt.title('Number of Unique Hosts by Month in Toronto')
plt.xlabel('Month')
plt.ylabel('Number of Unique Hosts')
plt.xticks(rotation=45)
plt.legend()
plt.grid(axis='y')
plt.show()
```



The number of unique hosts trend from October 2023 to September 2024, is pretty much identical in terms of steady increase as the listing version of this graph, although one thing to notice is that the number of hosts from October 2023 to September 2024 did not increase as much as the difference in listings, suggesting that perhaps many hosts now have multiple listings.

In [157…

```
average_hosts = {
    'September 2024': unique_hosts_2024_sept,
    'August 2024': unique_hosts_2024_aug,
    'October 2023': unique_hosts_2023_oct,
    'August 2021': unique_hosts_2021_toronto,
    'September 2020': unique_hosts_2020_toronto
}

sorted_hosts = dict(sorted(average_hosts.items(), key=lambda x: pd.to_datetime(x[0]

plt.figure(figsize=(10, 6))
plt.plot(sorted_hosts.keys(), sorted_hosts.values(), marker='o', color='blue', labe
plt.title('Number of Hosts in Toronto by Month and Year')
plt.xlabel('Date')
plt.ylabel('Number of Hosts')
plt.grid(axis='y')
plt.legend()

plt.show()
```



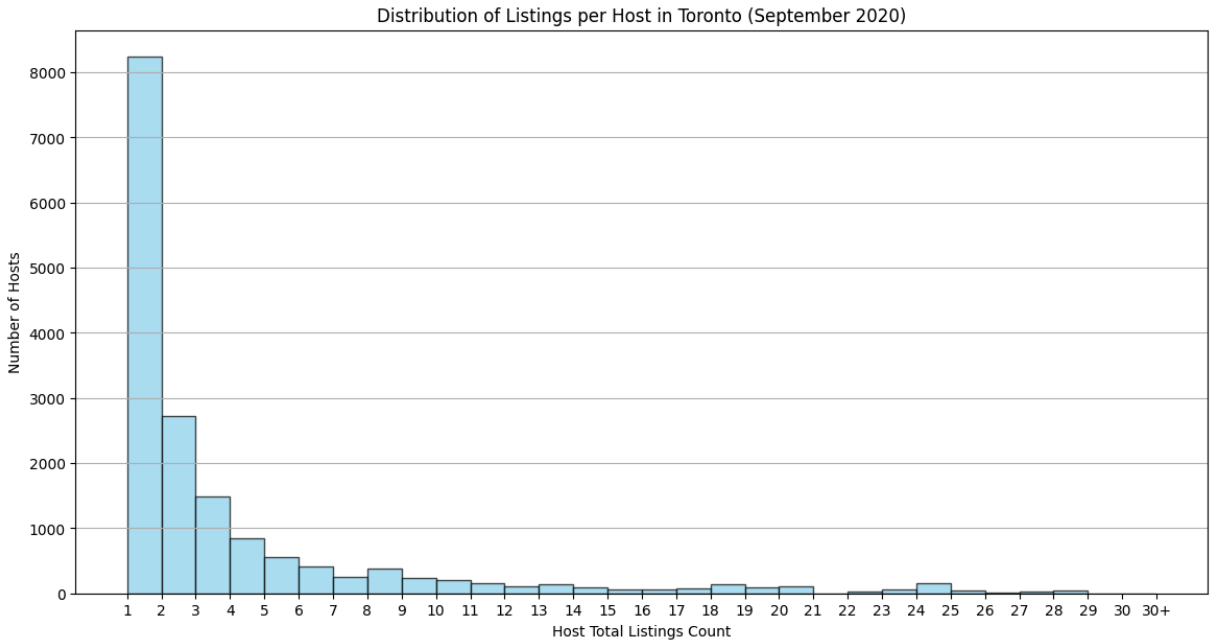
Similar to the listings trend we see a decline from September 2020 to August 2021, likely due to COVID-19, and from there a steady increase until September 2024

Listings per host distribution

To look into the number of listings per host we'll graph a histogram of the distribution of listings per host first in September 2020 and then in September 2024

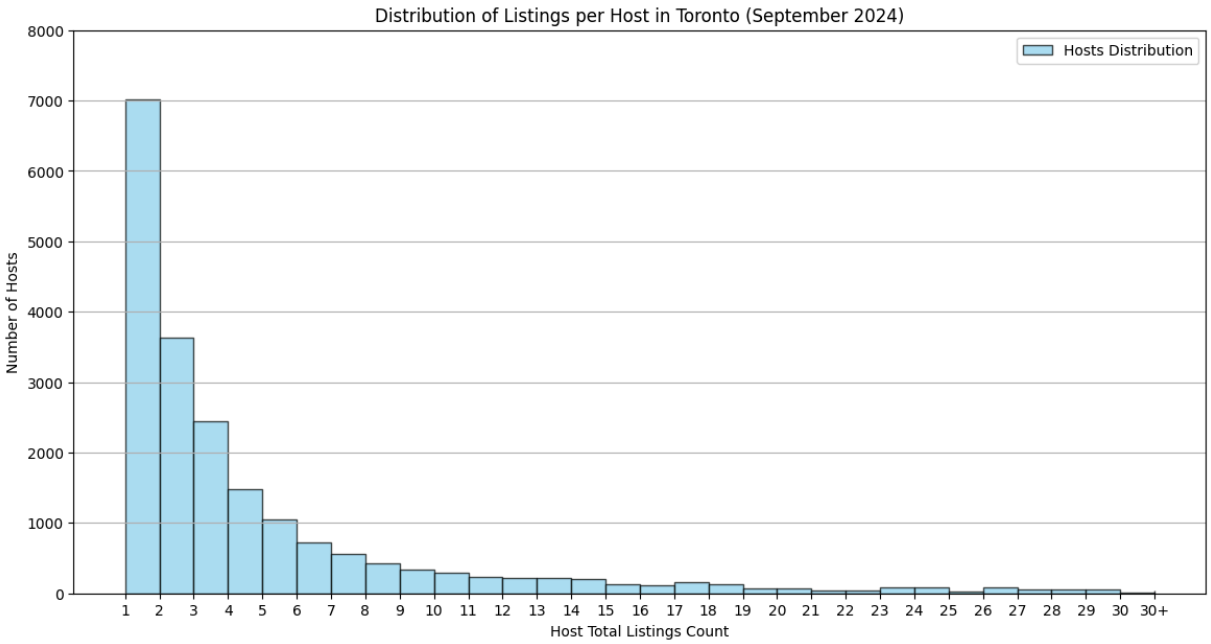
In [158…

```
bins = list(range(1, 32)) + [31]
labels = [str(i) for i in range(1, 31)] + ['30+']
plt.figure(figsize=(14, 7))
plt.hist(
    df3['host_total_listings_count'],
    bins=bins,
    color='skyblue',
    edgecolor='black',
    alpha=0.7
)

plt.title('Distribution of Listings per Host in Toronto (September 2020)')
plt.xlabel('Host Total Listings Count')
plt.ylabel('Number of Hosts')
plt.xticks(ticks=bins[:-1], labels=labels)

plt.grid(axis='y')
plt.show()
```



In [159…

```
bins = list(range(1, 32)) + [31]
labels = [str(i) for i in range(1, 31)] + ['30+']


plt.figure(figsize=(14, 7))
plt.hist(
    september_2024['host_total_listings_count'],
    bins=bins,
    color='skyblue',
    edgecolor='black',
    alpha=0.7,
    label='Hosts Distribution'
)

plt.title('Distribution of Listings per Host in Toronto (September 2024)')
plt.xlabel('Host Total Listings Count')
plt.ylabel('Number of Hosts')
plt.xticks(ticks=bins[:-1], labels=labels)
plt.yticks([0, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000])
plt.grid(axis='y')
plt.legend()
plt.show()
```



From the previous graphs, we know that the overall number of hosts and listings has increased from September 2020 to September 2024, but from these distributions, we can see that the number of hosts with just 1 listing has decreased by a bit over 1000 from 2020 to 2024. We can see that the number of hosts with listings in the 2-10 range has all increased from 2020 to 2024 confirming that many new listings are from hosts who already have at least one listing. The max amount of listings for a host in both 2020 and 2024 is around 30 and in general, the high 20s for the number of listings per host is still pretty low in both years.

The main takeaway here is that the number of listings and hosts in the Toronto Airbnb market is increasing but the number of listings from hosts with at least one listing is increasing more than new hosts.

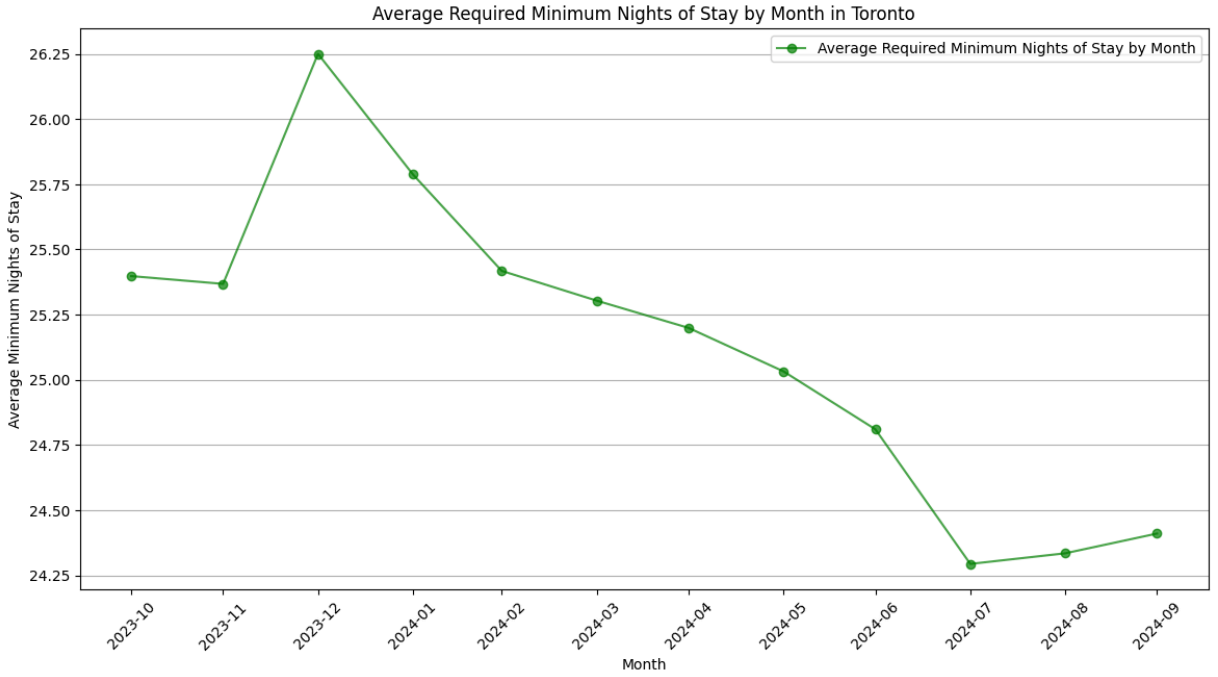
Minimum Nights of Stay Analysis

Next, we're going to look at the minimum required nights of stay, again by first plotting the average between October 2023 to September 2024 and then in the times during 2020, 2021, 2023, and 2024.

In [160…

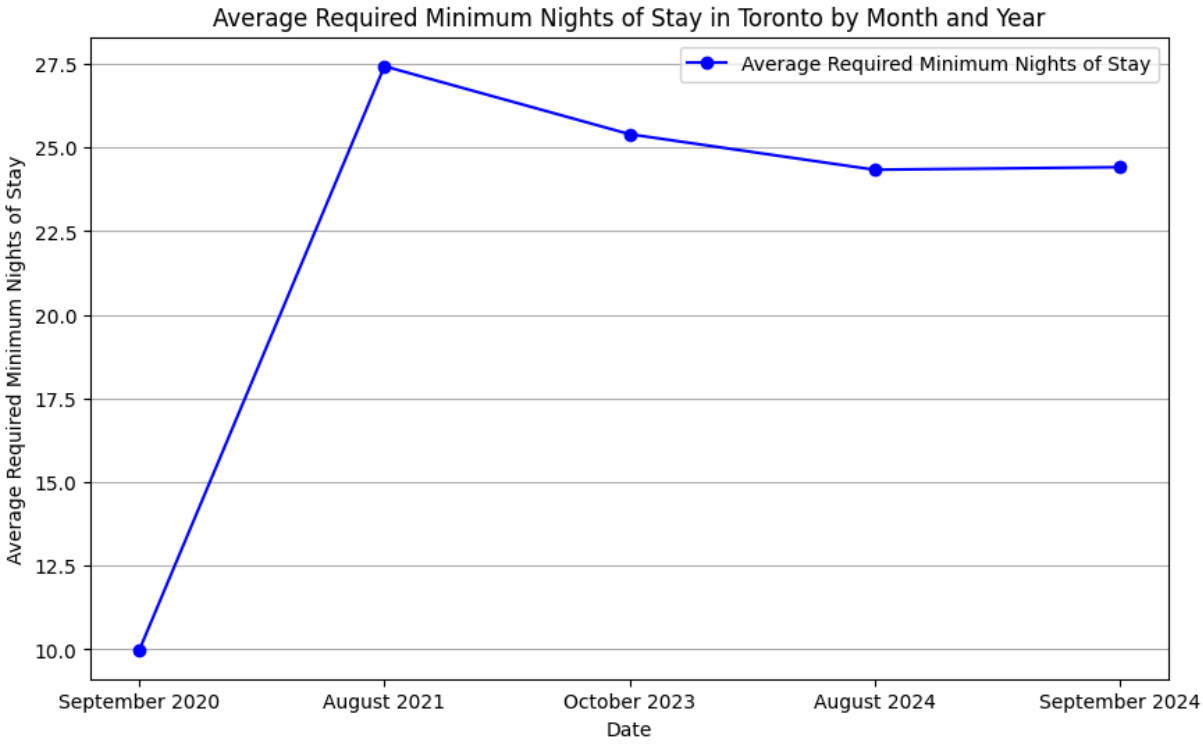
```
Toronto_df = Toronto_df.sort_index()

monthly_avg_price = Toronto_df['minimum_nights'].resample('ME').mean()

plt.figure(figsize=(14, 7))
plt.plot(monthly_avg_price.index, monthly_avg_price, color='green', marker='o', lab

plt.title('Average Required Minimum Nights of Stay by Month in Toronto')
plt.xlabel('Month')
plt.ylabel('Average Minimum Nights of Stay')
plt.xticks(monthly_avg_price.index, monthly_avg_price.index.strftime('%Y-%m'), rota
```

```
plt.legend()
plt.grid(axis='y')
plt.show()
```



In [161…

```
average_prices = {
    'September 2024': average_minimum_september_2024,
    'August 2024': average_minimum_august_2024,
    'October 2023': average_minimum_october_2023,
    'August 2021': average_minimum_nights_2021,
    'September 2020': average_minimum_nights_2020
}

sorted_prices = dict(sorted(average_prices.items(), key=lambda x: pd.to_datetime(x[

plt.figure(figsize=(10, 6))
plt.plot(sorted_prices.keys(), sorted_prices.values(), marker='o', color='blue',lab

plt.title('Average Required Minimum Nights of Stay in Toronto by Month and Year')
plt.xlabel('Date')
plt.ylabel('Average Required Minimum Nights of Stay')
plt.grid(axis='y')
plt.legend()

plt.show()
```



From the first graph, we can see a spike from November to December in average required minimum nights of stay and from there, a decrease until July followed by a slight increase in August and September 2024. But the changes were all within 1 night so nothing major.

The second graph shows a much more interesting observation as we see a large spike from September 2020 to August 2021 with the average minimum required nights of stay going from 10 to 27.5. From there there was a slight decrease but pretty stable around 25 in 2023 and 2024.

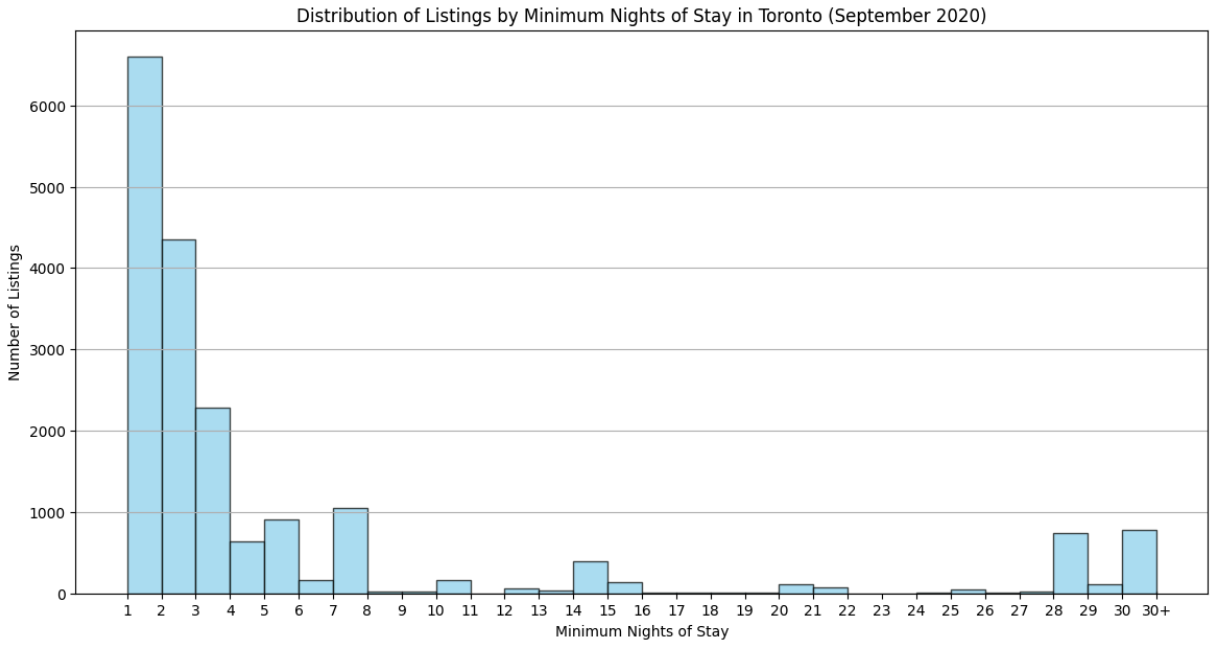
Distribution of Listings by Required Minimum Nights of Stay

To look further into this spike from 2020 to 2020 we'll graph the distribution of listings by required minimum night of stay with a histogram, first in September 2020 and then in September 2024

In [162…

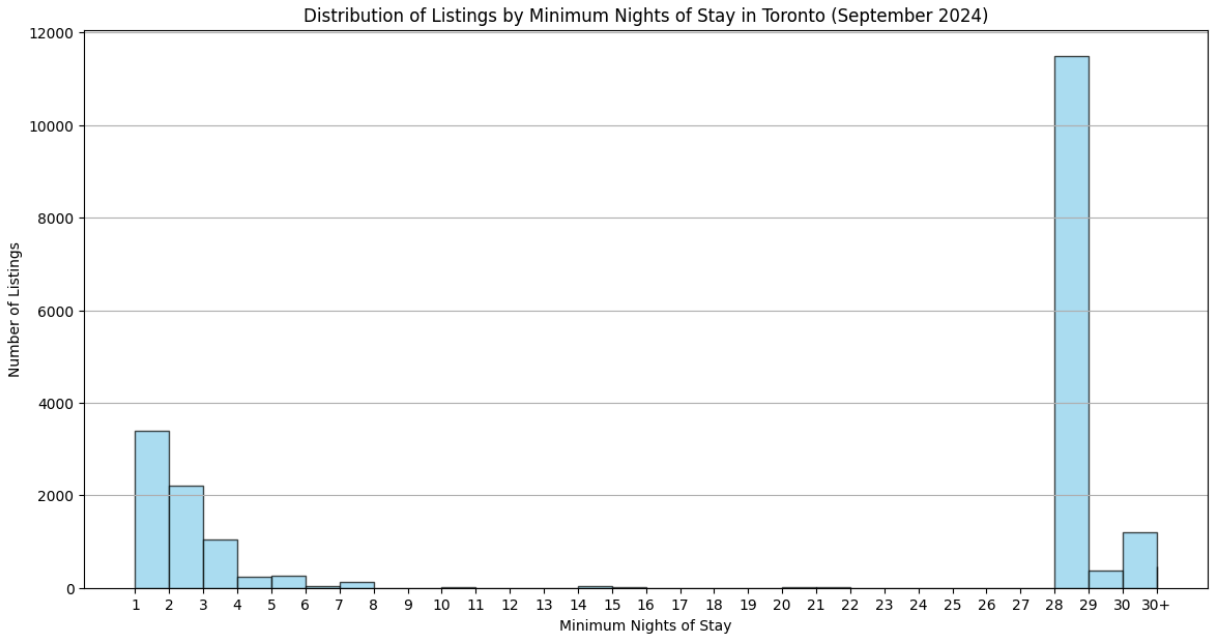
```
bins = list(range(1, 32)) + [31]
labels = [str(i) for i in range(1, 31)] + ['30+']
plt.figure(figsize=(14, 7))

plt.hist(
    df3['minimum_nights'],
    bins=bins,
    color='skyblue',
    edgecolor='black',
    alpha=0.7
)

plt.title('Distribution of Listings by Minimum Nights of Stay in Toronto (September
plt.xlabel('Minimum Nights of Stay')
```

```
plt.ylabel('Number of Listings')

plt.xticks(ticks=bins[:-1], labels=labels)
plt.grid(axis='y')
plt.show()
```



In [163…

```
bins = list(range(1, 32)) + [31]
labels = [str(i) for i in range(1, 31)] + ['30+']

plt.figure(figsize=(14, 7))
plt.hist(
    september_2024['minimum_nights'],
    bins=bins,
    color='skyblue',
    edgecolor='black',
    alpha=0.7
)

plt.title('Distribution of Listings by Minimum Nights of Stay in Toronto (September
plt.xlabel('Minimum Nights of Stay')
plt.ylabel('Number of Listings')

plt.xticks(ticks=bins[:-1], labels=labels)
plt.grid(axis='y')
plt.show()
```



We can see a very interesting change between September 2020 and September 2024 here in that the majority of the listings are now at 28 required minimum nights of stay in 2024 rather than 1 day in 2020 which even decreased from about 7000 listings in 2020 to 3000 in 2024 and 1 required minimum night of stay despite the overall increase in listings from 2020 to 2024.

Because of this, I decided to do some searching on short-term rentals in Toronto specifically in 2021 since that is when the initial spike is first seen from our earlier line graph. I found that a short-term rental is a rental for less than 28 days and that in 2021 a bylaw was introduced on this requiring registration and stricter regulations for these short-term rentals [5][6]. It is likely due to this that we now see the switch to a large number of listings with a minimum required nights of stay of 28 to try and avoid these regulations. Interestingly enough I also found another article much more recently just a month or two ago on new regulations and restrictions being introduced for rentals so perhaps we'll see even more changes in listing trends [7].

Geographic Analysis

The code for the maps will save locally wherever this is ran as htmls otherwise it drastically increases the size of the jpynb and slows everything down, the maps are also exported to the submitted html file where they can be seen

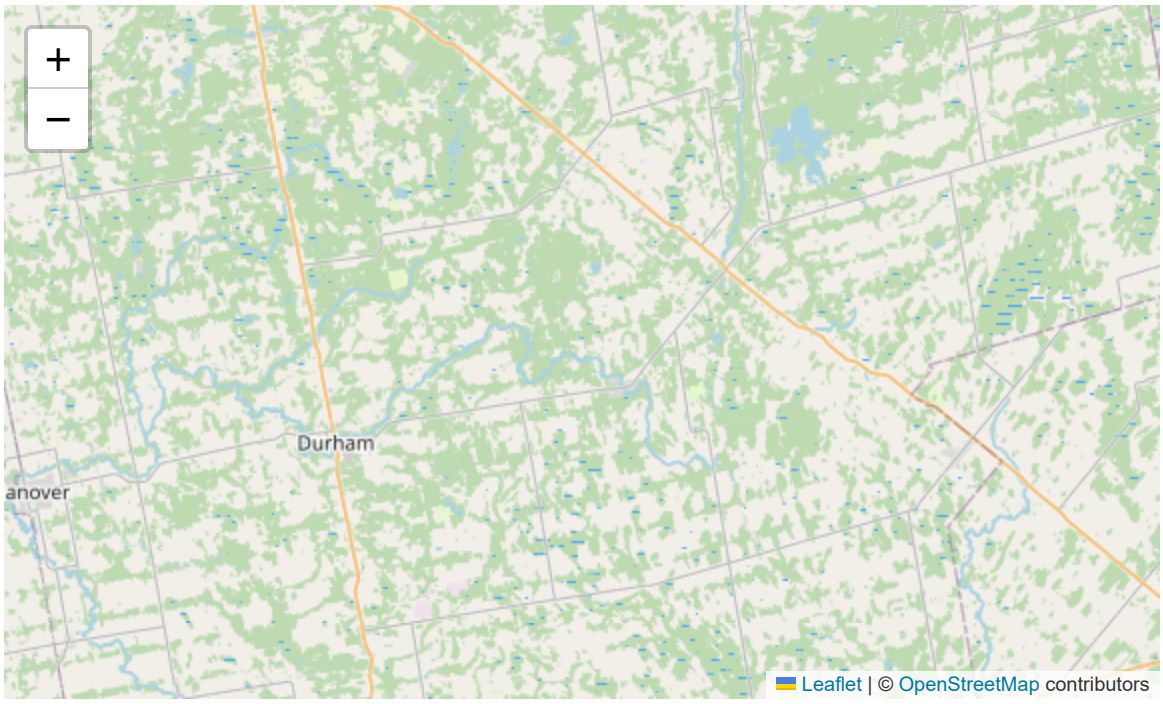
We're going to make 3 heatmaps of the number of listings in Toronto in September 2020, August 2021, and September 2024 to see if there's anything interesting.

In [174…

```
#september 2020
import folium
from folium.plugins import MarkerCluster, HeatMap
m = folium.Map(location=[df3['latitude'].mean(), df3['longitude'].mean()], zoom_sta
```

```
marker_cluster = MarkerCluster().add_to(m)
heat_data = [[row['latitude'], row['longitude']] for index, row in df3.iterrows()]
for idx, row in df3.iterrows():
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        popup=f"ID: {row['id']}"
    ).add_to(marker_cluster)
HeatMap(heat_data, radius=15).add_to(m)


m.save('toronto_map2020.html')
m
```

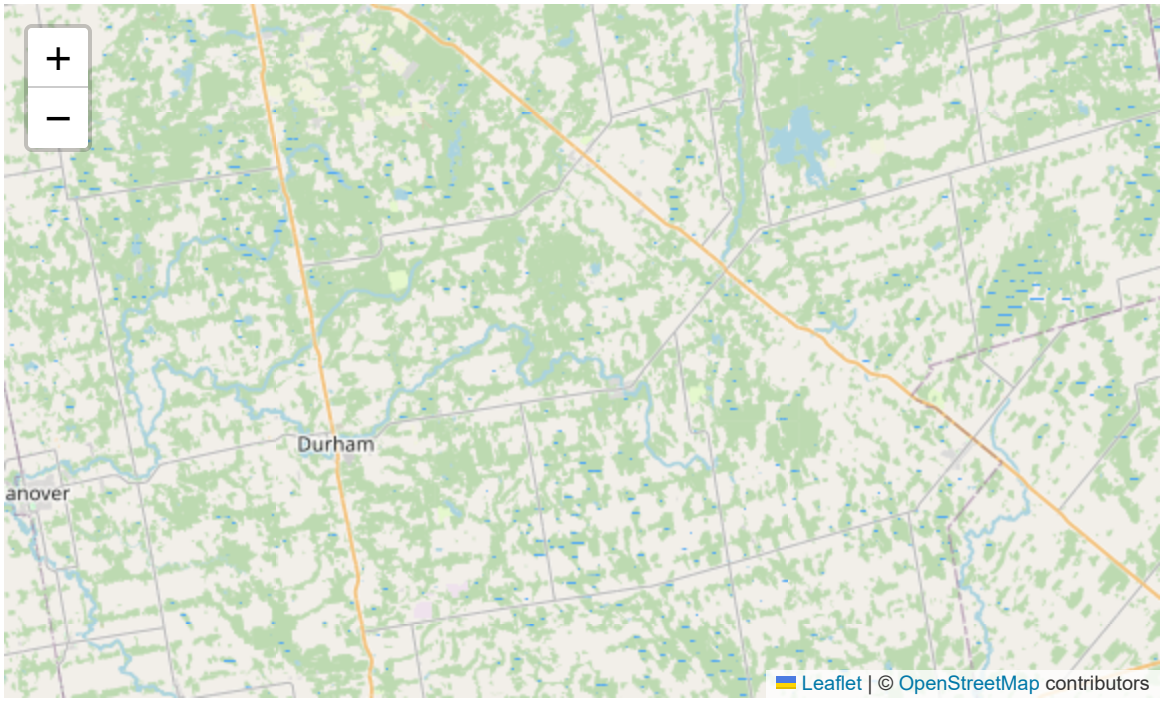
Out[174…



In [175…

```
#august 2021
m1 = folium.Map(location=[df2['latitude'].mean(), df2['longitude'].mean()], zoom_st
marker_cluster = MarkerCluster().add_to(m1)
heat_data = [[row['latitude'], row['longitude']] for index, row in df2.iterrows()]
for idx, row in df2.iterrows():
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        popup=f"ID: {row['id']}"
    ).add_to(marker_cluster)
HeatMap(heat_data, radius=15).add_to(m1)


m1.save('toronto_map2021.html')
m1
```

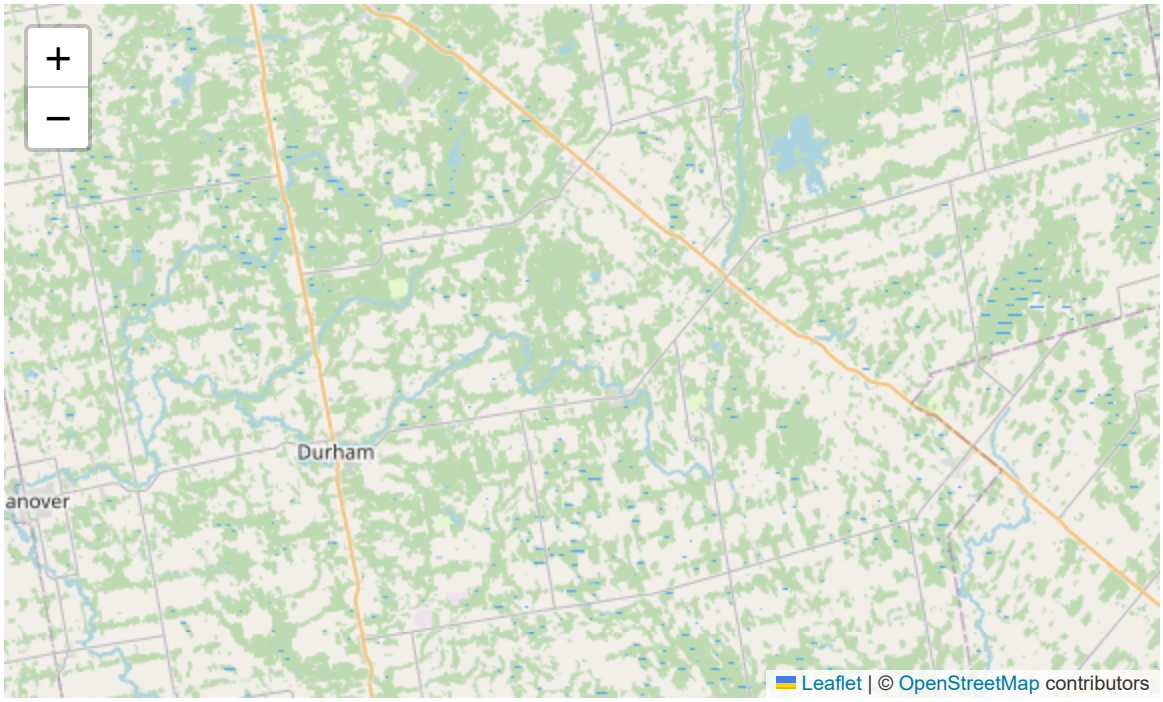
Out[175…



In [176…

```
#september 2024
september_2024 = Toronto_df[(Toronto_df['last_scraped'].dt.month == 9) & (Toronto_d
m2 = folium.Map(location=[september_2024['latitude'].mean(), september_2024['longit
marker_cluster = MarkerCluster().add_to(m2)
heat_data = [[row['latitude'], row['longitude']] for index, row in september_2024.i
for idx, row in september_2024.iterrows():
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        popup=f"ID: {row['id']}"
    ).add_to(marker_cluster)
HeatMap(heat_data, radius=15).add_to(m2)


m2.save('toronto_map2024.html')
m2
```

Out[176...



When looking at the 3 maps first off we see the the total number of listings decrease from September 2020 to August 2021 to then increasing in September 2024 as we also know from our previous analysis. Not surprisingly the center in the South and heart of Toronto has The most listings by far in all 3 maps. One interesting thing when we zoom out to about 6 clusters in the September 2020 and August 2021 map is that first the clusters are quite similar, and second the majority of the decrease from COVID between 2020 to 2021 seems to have mostly affected listings in the east and south of Toronto clusters with the north and west not decreasing as much.

Next, when we look at the most recent September 2024 we see that the zoomout at around that level of the previous graphs now has 7 clusters instead of 6 with the noticeable one being the new large cluster right west of the major cluster with the majority of the listings in the South. We also see continued growth from 2020 to 2024 in the east, west, and southwest of Toronto. Lastly, the northeast cluster also increased a lot while the nearby cluster in the north present in the 2020 and 2021 maps has moved a bit more west in the 2024 graph.

Summary

The average price in the Airbnb market increased drastically between September 2020 and August 2021, since then it has increased a bit and then remained mostly steady. The market is growing with the number of listings and hosts growing although many of these listings are from hosts who already have at least one listing as multihosts keep increasing.

After the introduction of bylaws and short-term rental regulations in 2021 the market switched to a more long-term rental one with the majority of rentals being listed for a minimum required nights of stay of 28, it will be interesting to see how the market continues to evolve as more regulations and restrictions are applied to try and help the housing crisis.

When looking at a geographic map we can see that the decrease in listings from September 2020 to August 2021 during COVID mostly affected the east and south of Toronto and not the north and west as much. In 2024 in comparison, we've seen increased growth in every area but in particular in the east, west, and southwest of Toronto which were lower areas before.

I improved the axes on the graph for the distribution for the number of listings per host as the numbers were quite close making it hard to see the difference compared to now having the same scale which was a major feedback. I didn't for the required minimum nights of stay as the numbers are quite different and the main takeaway is the major switch from 1 minimum night of stay to 28 rather than the exact difference in number for 1 minimum night of stay which is still reasonably clear, unlike the other graph.

One thing I'd like to look into is if there is a better way than the simple line graphs to show change over time. For future work, it would be nice if I had data from 2022 or even all of 2020 and 2021 so I could compare months between the years in more detail to see if there were any specific trends. I also would've liked to do more neighborhood analysis of certain areas of Toronto, likely with a choropleth map showing price but I was having some trouble finding and getting the geojson to work properly with it in time. If we could also find data on the number of bookings or occupancy we could try to see the profit of certain areas, hosts, or even listings over time which would be interesting. Finally, it was mentioned that an increase in bylaws, restrictions, and regulations on a variety of rental things led to changes like the switch to longer-term rentals. One thing these restrictions are doing is increasing the requirement and checks on licensing. We do have a license column and I'm not sure how accurate the data in it is from being from a web scraper but briefly looking through it, I noticed that the number of rows with a license did seem to increase from 2020 to 2021 to 2024 so it would be interesting to look into that in more detail and if these regulations are working to make more rentals official and licensed.

7.4. Guiding Question 4

What aspects of the Airbnb platform affect hosts' reviews and performance, do top rated hosts have anything in common, are there any differences between hosts with few listings vs many?

We calculated the average scores of review and the average count of listings for Toronto and Ottawa separately and used as the threshold of each of them to group the data into 'high rated listings' vs. 'low rated listings' and 'hosts have many listings' vs. 'hosts have few listings'. Then explored what they do/do not have in common.

In [167…

```
city_review_mean={}
city_listing_mean={}
review_col=['review_scores_rating','reviews_per_month']
```

```
listing_col=['host_listings_count','price+']
unique_cities=TnO_df['city'].unique()
for i in unique_cities:
    city_review_mean[i]=TnO_df[TnO_df['city']==i][review_col].mean()
    city_listing_mean[i]=TnO_df[TnO_df['city']==i][listing_col].mean()
print(city_review_mean)
print(city_listing_mean)

review_threshold={}
for city in city_review_mean:
    review_threshold[city]=city_review_mean[city]
def review_category(x,review_threshold):
    city=x['city']
    threshold=review_threshold[city]
    if x['review_scores_rating']>threshold['review_scores_rating']:
        return 'high'
    else:
        return 'low'
TnO_df['review_category']=TnO_df.apply(lambda x: review_category(x,review_threshold

listing_threshold={}
for city in city_listing_mean:
    listing_threshold[city]=city_listing_mean[city]
def listing_category(x,listing_threshold):
    city=x['city']
    threshold=listing_threshold[city]
    if x['host_listings_count']>threshold['host_listings_count']:
        return 'many'
    else:
        return 'few'
TnO_df['listing_category']=TnO_df.apply(lambda x: listing_category(x,listing_thresh

TnO_df.head()
```

```
{'Ottawa': review_scores_rating    4.767168
reviews_per_month       1.900795
dtype: float64, 'Toronto': review_scores_rating    4.776990
reviews_per_month       1.423179
dtype: float64}
{'Ottawa': host_listings_count     14.288426
price+                 133.723925
dtype: float64, 'Toronto': host_listings_count      7.892933
price+                 151.202404
dtype: float64}
```

Out[167…

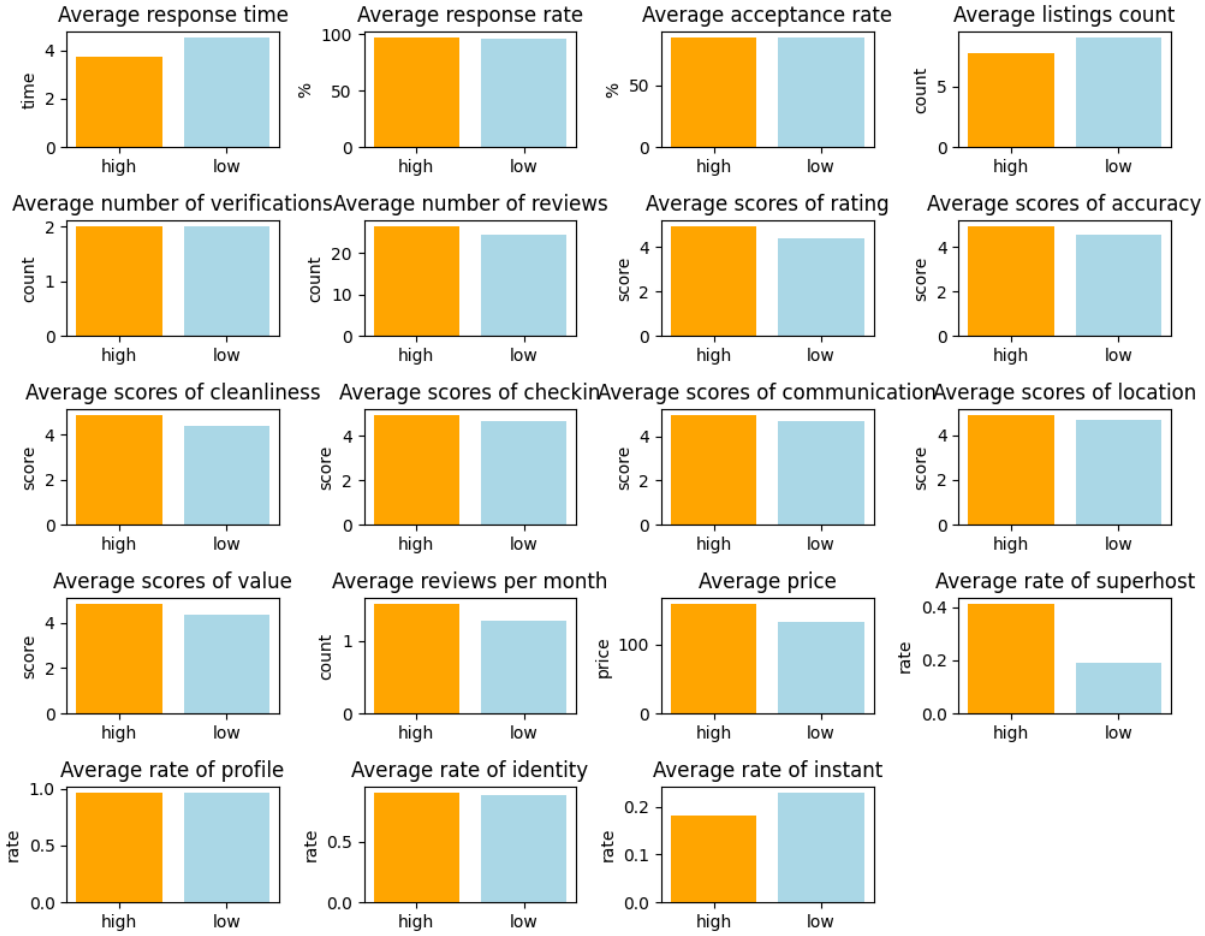
	id	last_scraped	name	description	neighborhood_overview	host_id	hos
18835	34220	2023-12-18	Rental unit in Ottawa · 1 bedroom · 1 bed · 1 …	$0.00	NaN	147438	20
18836	261065	2023-12-18	Rental unit in Ottawa · ★4.96 · 1 bedroom · 1 …	$0.00	Westboro Village is a century-old community, a…	1369632	20
18837	290033	2023-12-18	Home in North Gower · ★5.0 · 2 bedrooms · 2 be…	$0.00	Most folks ask about Strathmere - we are less …	415201	20
18838	336692	2023-12-18	Rental unit in Ottawa · ★5.0 · 1 bedroom · 1 b…	$0.00	Walking distance to: -Parliament -Do…	1712975	20
18839	490182	2023-12-18	Home in Ottawa · ★5.0 · 1 bedroom · 1 bed · 1 …	$0.00	The worlds longest skating rink, a UNESCO heri…	2401179	20

5 rows × 61 columns

In [168…

```
grouped_df=TnO_df.groupby('review_category').agg({'host_response_time':'mean','host
                                                  'host_acceptance_rate':'mean'
                                                  'host_verifications':'mean','
                                                  'review_scores_rating':'mean'
                                                  'review_scores_cleanliness':'
                                                  'review_scores_communication'
                                                  'review_scores_value':'mean',
                                                  'host_is_superhost':'mean','h
                                                  'host_identity_verified':'mea
fig,ax=plt.subplots(5,4,figsize=(10,8))
ax[0,0].bar(grouped_df['review_category'],grouped_df['host_response_time'],color=['
ax[0,0].set_title('Average response time')
ax[0,0].set_ylabel('time')
ax[0,1].bar(grouped_df['review_category'],grouped_df['host_response_rate'],color=['
ax[0,1].set_title('Average response rate')
```

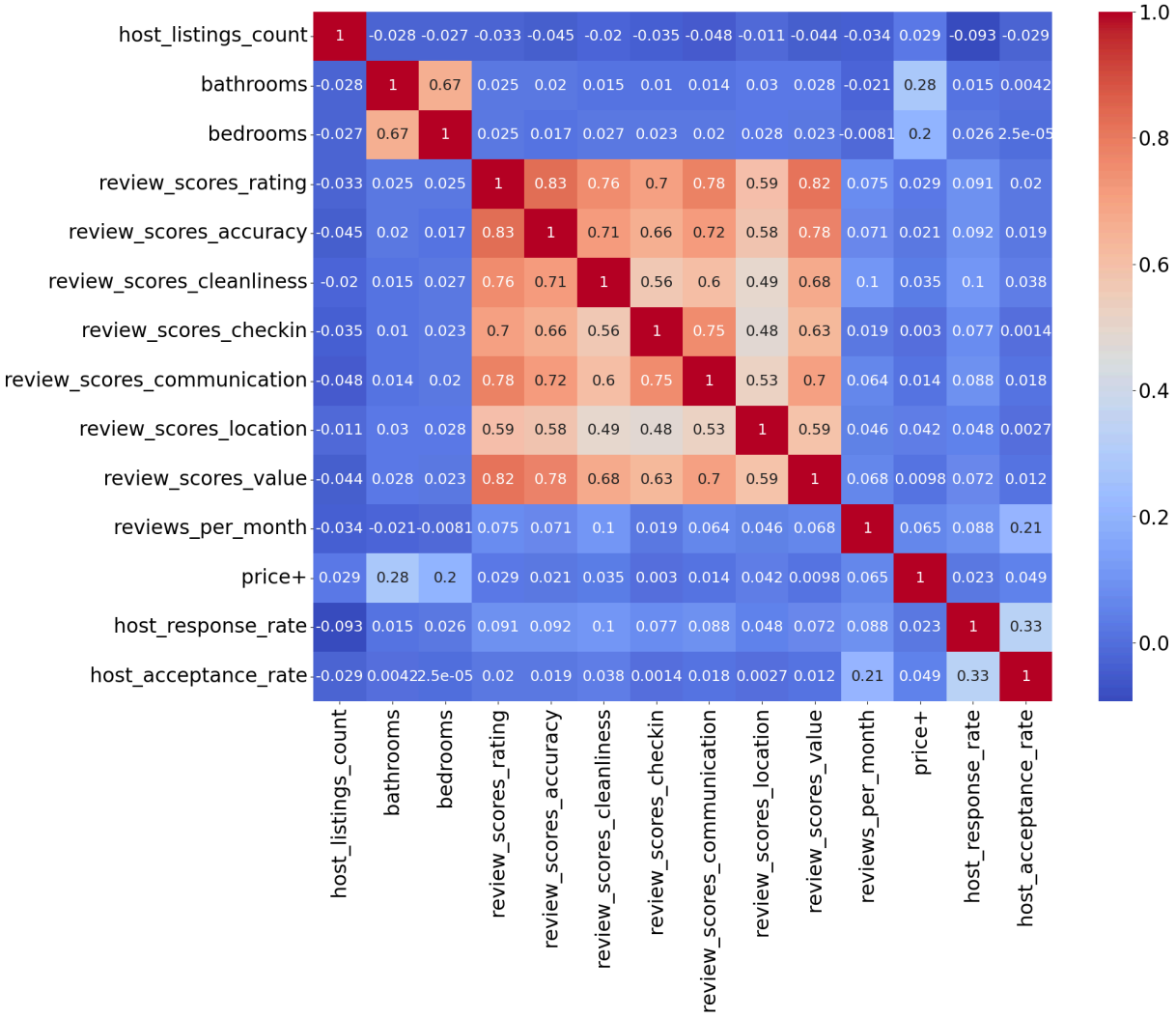
```
ax[0,1].set_ylabel('%')
ax[0,2].bar(grouped_df['review_category'],grouped_df['host_acceptance_rate'],color=
ax[0,2].set_title('Average acceptance rate')
ax[0,2].set_ylabel('%')
ax[0,3].bar(grouped_df['review_category'],grouped_df['host_listings_count'],color=[
ax[0,3].set_title('Average listings count')
ax[0,3].set_ylabel('count')
ax[1,0].bar(grouped_df['review_category'],grouped_df['host_verifications'],color=['
ax[1,0].set_title('Average number of verifications')
ax[1,0].set_ylabel('count')
ax[1,1].bar(grouped_df['review_category'],grouped_df['number_of_reviews'],color=['o
ax[1,1].set_title('Average number of reviews')
ax[1,1].set_ylabel('count')
ax[1,2].bar(grouped_df['review_category'],grouped_df['review_scores_rating'],color=
ax[1,2].set_title('Average scores of rating')
ax[1,2].set_ylabel('score')
ax[1,3].bar(grouped_df['review_category'],grouped_df['review_scores_accuracy'],colo
ax[1,3].set_title('Average scores of accuracy')
ax[1,3].set_ylabel('score')
ax[2,0].bar(grouped_df['review_category'],grouped_df['review_scores_cleanliness'],c
ax[2,0].set_title('Average scores of cleanliness')
ax[2,0].set_ylabel('score')
ax[2,1].bar(grouped_df['review_category'],grouped_df['review_scores_checkin'],color
ax[2,1].set_title('Average scores of checkin')
ax[2,1].set_ylabel('score')
ax[2,2].bar(grouped_df['review_category'],grouped_df['review_scores_communication']
ax[2,2].set_title('Average scores of communication')
ax[2,2].set_ylabel('score')
ax[2,3].bar(grouped_df['review_category'],grouped_df['review_scores_location'],colo
ax[2,3].set_title('Average scores of location')
ax[2,3].set_ylabel('score')
ax[3,0].bar(grouped_df['review_category'],grouped_df['review_scores_value'],color=[
ax[3,0].set_title('Average scores of value')
ax[3,0].set_ylabel('score')
ax[3,1].bar(grouped_df['review_category'],grouped_df['reviews_per_month'],color=['o
ax[3,1].set_title('Average reviews per month')
ax[3,1].set_ylabel('count')
ax[3,2].bar(grouped_df['review_category'],grouped_df['price+'],color=['orange','lig
ax[3,2].set_title('Average price')
ax[3,2].set_ylabel('price')
ax[3,3].bar(grouped_df['review_category'],grouped_df['host_is_superhost'],color=['o
ax[3,3].set_title('Average rate of superhost')
ax[3,3].set_ylabel('rate')
ax[4,0].bar(grouped_df['review_category'],grouped_df['host_has_profile_pic'],color=
ax[4,0].set_title('Average rate of profile')
ax[4,0].set_ylabel('rate')
ax[4,1].bar(grouped_df['review_category'],grouped_df['host_identity_verified'],colo
ax[4,1].set_title('Average rate of identity')
ax[4,1].set_ylabel('rate')
ax[4,2].bar(grouped_df['review_category'],grouped_df['instant_bookable'],color=['or
ax[4,2].set_title('Average rate of instant')
ax[4,2].set_ylabel('rate')
fig.delaxes(ax[4,3])
plt.tight_layout()
plt.show()
```



As can be seen from the bar plots above, 'response time', 'listings count', 'price', 'is/is not superhost' and 'is/is not bookable instantly' generally impact on the rate scores.

In [169…

```
corr_col=numeric_col+per_col
correlation_matrix=TnO_df[corr_col].corr()
plt.figure(figsize=(20,15))
heatmap=sns.heatmap(correlation_matrix,annot=True,cmap="coolwarm",annot_kws={"size"
plt.xticks(fontsize=24)
plt.yticks(fontsize=24)
colorbar=heatmap.collections[0].colorbar
colorbar.ax.tick_params(labelsize=24)
plt.show()
```

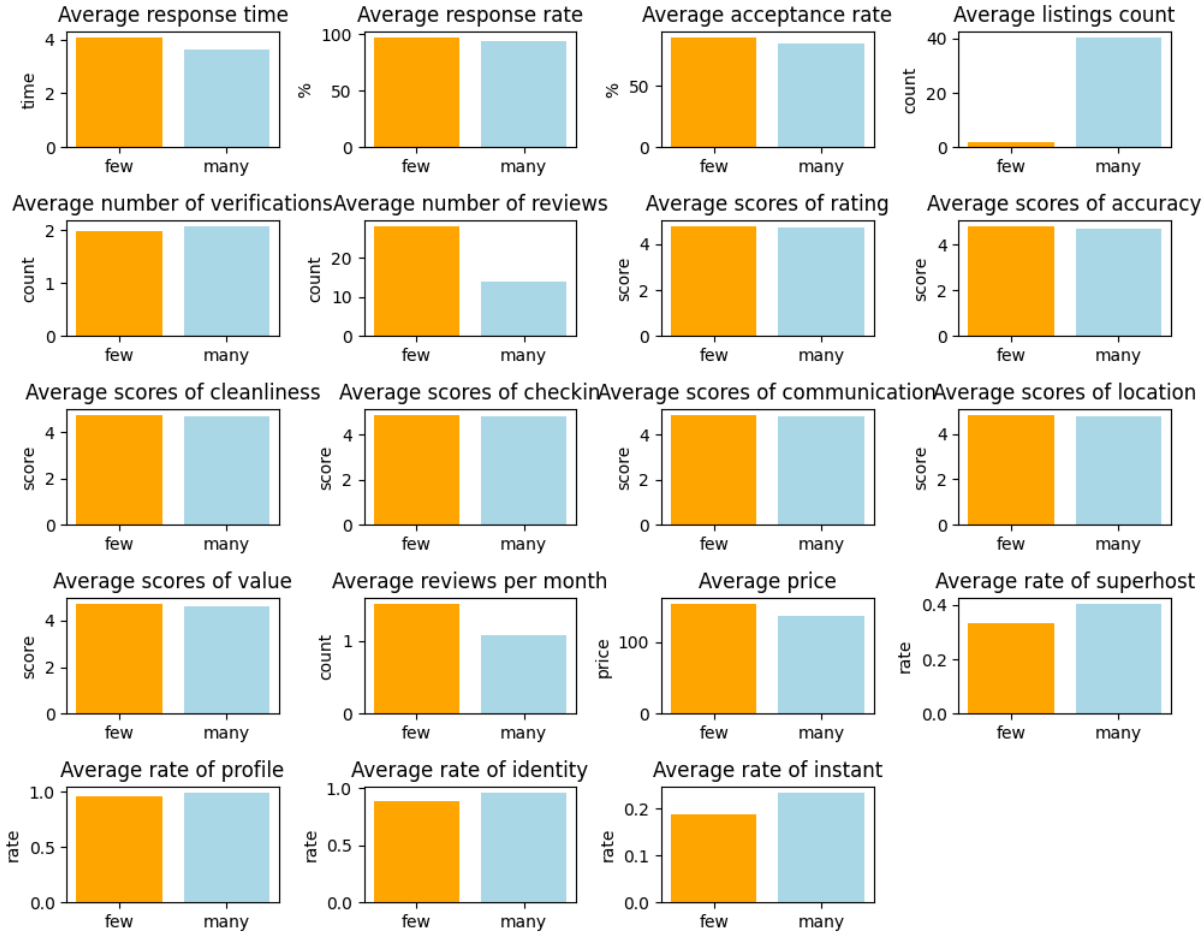


The heatmap of correlation about reviews shows that aspects that have stronger correlation are accuracy>price>communication>cleanliness>check in>location.

In [170…

```
classified_df=TnO_df.groupby('listing_category').agg({'host_response_time':'mean','
                                                      'host_acceptance_rate':'mean'
                                                      'host_verifications':'mean','
                                                      'review_scores_rating':'mean'
                                                      'review_scores_cleanliness':'
                                                      'review_scores_communication'
                                                      'review_scores_value':'mean',
                                                      'host_is_superhost':'mean','h
                                                      'host_identity_verified':'mea
fig,ax=plt.subplots(5,4,figsize=(10,8))
ax[0,0].bar(classified_df['listing_category'],classified_df['host_response_time'],c
ax[0,0].set_title('Average response time')
ax[0,0].set_ylabel('time')
ax[0,1].bar(classified_df['listing_category'],classified_df['host_response_rate'],c
ax[0,1].set_title('Average response rate')
ax[0,1].set_ylabel('%')
ax[0,2].bar(classified_df['listing_category'],classified_df['host_acceptance_rate']
ax[0,2].set_title('Average acceptance rate')
ax[0,2].set_ylabel('%')
ax[0,3].bar(classified_df['listing_category'],classified_df['host_listings_count'],
```

```
ax[0,3].set_title('Average listings count')
ax[0,3].set_ylabel('count')
ax[1,0].bar(classified_df['listing_category'],classified_df['host_verifications'],c
ax[1,0].set_title('Average number of verifications')
ax[1,0].set_ylabel('count')
ax[1,1].bar(classified_df['listing_category'],classified_df['number_of_reviews'],co
ax[1,1].set_title('Average number of reviews')
ax[1,1].set_ylabel('count')
ax[1,2].bar(classified_df['listing_category'],classified_df['review_scores_rating']
ax[1,2].set_title('Average scores of rating')
ax[1,2].set_ylabel('score')
ax[1,3].bar(classified_df['listing_category'],classified_df['review_scores_accuracy
ax[1,3].set_title('Average scores of accuracy')
ax[1,3].set_ylabel('score')
ax[2,0].bar(classified_df['listing_category'],classified_df['review_scores_cleanlin
ax[2,0].set_title('Average scores of cleanliness')
ax[2,0].set_ylabel('score')
ax[2,1].bar(classified_df['listing_category'],classified_df['review_scores_checkin'
ax[2,1].set_title('Average scores of checkin')
ax[2,1].set_ylabel('score')
ax[2,2].bar(classified_df['listing_category'],classified_df['review_scores_communic
ax[2,2].set_title('Average scores of communication')
ax[2,2].set_ylabel('score')
ax[2,3].bar(classified_df['listing_category'],classified_df['review_scores_location
ax[2,3].set_title('Average scores of location')
ax[2,3].set_ylabel('score')
ax[3,0].bar(classified_df['listing_category'],classified_df['review_scores_value'],
ax[3,0].set_title('Average scores of value')
ax[3,0].set_ylabel('score')
ax[3,1].bar(classified_df['listing_category'],classified_df['reviews_per_month'],co
ax[3,1].set_title('Average reviews per month')
ax[3,1].set_ylabel('count')
ax[3,2].bar(classified_df['listing_category'],classified_df['price+'],color=['orang
ax[3,2].set_title('Average price')
ax[3,2].set_ylabel('price')
ax[3,3].bar(classified_df['listing_category'],classified_df['host_is_superhost'],co
ax[3,3].set_title('Average rate of superhost')
ax[3,3].set_ylabel('rate')
ax[4,0].bar(classified_df['listing_category'],classified_df['host_has_profile_pic']
ax[4,0].set_title('Average rate of profile')
ax[4,0].set_ylabel('rate')
ax[4,1].bar(classified_df['listing_category'],classified_df['host_identity_verified
ax[4,1].set_title('Average rate of identity')
ax[4,1].set_ylabel('rate')
ax[4,2].bar(classified_df['listing_category'],classified_df['instant_bookable'],col
ax[4,2].set_title('Average rate of instant')
ax[4,2].set_ylabel('rate')
fig.delaxes(ax[4,3])
plt.tight_layout()
plt.show()
```



Hosts who have few listings and those who have many listings commonly have a significant difference on their 'response time', 'price', 'is/is not superhost', 'is/is not identified' and 'is/is not bookable instantly'. Listings with higher review scores and more reviews tend to attract higher prices hosts should also limit their effort into more effective aspects

8. Conclusion

In summary, the data suggests that while property prices in Toronto are average compared to other cities, Airbnb listings exhibit a lower capacity, leading to higher per-bed costs. Additionally, Toronto's stricter minimum night requirements contribute to bookings not being as low-cost as they may appear. However, for customers seeking longer stays (around one month) for one or two people, Toronto can offer better value. Despite having more total reviews than other cities, Toronto has the lowest average reviews per listing, highlighting a highly competitive market where only a few listings attract most of the attention.

Our analysis revealed that entire rental units are the most popular property type on Airbnb, while review ratings appear to have little effect on pricing. Although amenities like TV showed some influence on prices, there was no strong correlation between pricing and the other variables analysed. External factors, such as economic trends and tourism, likely play a significant role in shaping Airbnb pricing strategies.

We also discovered that the Airbnb market is constantly growing from 2020 to now with hosts and listings increasing, although we are also seeing a large increase in multihosts. Prices spiked from 2020 to 2021 and since have been relatively stable. The market has also shifted from a short-term to a long-term rental market with the majority of hosts listing their rentals for a minimum required 28 nights of stay to try and avoid short-term rental regulations. The feedback we received helped us improve things like clarity on graphs as well as gather other ideas for future steps and things to add.

Listings with higher review scores and more reviews tend to command higher prices, indicating that hosts should prioritize enhancing key aspects of their properties to maximize their appeal and pricing potential.

Recommendations and Future Work

In conclusion, we conducted a comprehensive data analysis and discovered valuable insights. However, to further enhance the analysis and precision of price predictions for Airbnb listings in Toronto, we advise putting in place a machine learning strategy in future that makes use of a variety of features, such as property attributes, location data, seasonal trends, and external factors such as local events and economic conditions. Future research should concentrate on:

1. Data Enrichment: Adding more datasets to give a more complete picture of the variables affecting prices, such as tourism data and local economic indicators.
2. Feature Engineering: Investigating additional variables and the relationships between existing features to uncover patterns that could enhance the model's predictive capabilities.
3. Data Modelling: Experimenting with various machine learning algorithms (e.g., regression models, decision trees, or ensemble methods) to identify which effectively reflects the intricate pricing dynamics of Toronto's Airbnb market.
4. Model Evaluation: Evaluating and improving the model using new data and performance feedback to ensure it adjusts to evolving market conditions. By following these recommendations, we seek to create a strong price prediction model that helps hosts and potential investors make informed pricing choices and boosts their competitiveness in the Toronto Airbnb market.

9. Bibliography

[1]. The Future of Airbnb: Navigating Regulatory Challenges and Embracing New Trends. (2024). Kavout.com. https://www.kavout.com/market-lens/the-future-of-airbnb-navigating-regulatory-challenges-and-embracing-new-trends

[2]. Sherwood, H. (2019, May 5). How Airbnb took over the world. The Guardian. https://www.theguardian.com/technology/2019/may/05/airbnb-homelessness-renting-housing-accommodation-social-policy-cities-travel-leisure

[3]. Get the Data. (n.d.). Inside Airbnb. Retrieved September 23, 2024, from https://insideairbnb.com/get-the-data/

[4]. Data Policies. (n.d.). Inside Airbnb. Retrieved September 23, 2024, from https://insideairbnb.com/data-policies/

[5]. Toronto, C. O. (2024, June 17). Short-Term rentals. City of Toronto. https://www.toronto.ca/community-people/housing-shelter/short-term-rentals/

[6]. Burman, D. (2021, January 28). Toronto begins laying charges for short-term rental non-compliance. CityNews Toronto. https://toronto.citynews.ca/2021/01/28/toronto-begins-laying-charges-for-short-term-rental-non-compliance/

[7]. K, G., & K, G. (2024, August 14). Toronto announces updates to Short-Term Rental Bylaw - Ontario Landlord help. Ontario Landlord Help - Ontario Landlord Help. https://olhgroup.ca/toronto-announces-updates-to-short-term-rental-bylaw/