Skill Up AirBnB Project – Data Analysis
Vishal Maheshwari

Business goals

Imagine you are a data analyst at Airbnb. Your role is to analyze property listings to help the company better understand its offerings and enhance its services. Your focus is on the following business goals:

Pricing optimization: Identify trends in neighborhood pricing and room types to help Airbnb and hosts adjust rates for maximum profitability.

Service improvement: Identify factors such as cancellation policy, reviews, and cleaning fees that affect host performance.

Steps

Step 1
Briefly describe the problem this project is addressing. Use your own words to summarize the issue being solved.

- The main issues to address for this project are Pricing optimization and service improvement by identifying factors that can affect the two main issues and discovering if there are in inconsistent pricing, unclear impact of service factors and analyze the patterns found to identify pricing trends to guide hosts toward competitive pricing while remaining profitable and to highlight service related factors that improve guest experience and host performance.

Step 2
State the main outcome you aim to achieve through this project.

- The goal of this project is to provide data-driven insights that help Airbnb optimize pricing across neighborhoods and room types, while identifying key service factors that improve host performance and guest satisfaction.

Step 3
Describe the dataset, its source, and the type of data it contains.

- The dataset for this project is sourced from Kaggle and provided as a CSV file. It contains detailed information about Airbnb property listings, including both numerical and categorical data. The key columns include:
  - **id, name, host_id, host_name** – Unique identifiers and host details
  - **host_identity_verified** – Indicates if the host's identity is verified
  - **neighbourhood group, neighbourhood, lat, long** – Location details
  - **country, country code** – Geographic information
  - **instant_bookable, cancellation_policy** – Booking and policy attributes
  - **room type, construction year** – Property characteristics
  - **price, service fee, cleaning fee** – Pricing-related fields
  - **minimum nights, availability_365** – Availability and stay requirements
  - **number of reviews, last review, reviews per month, review rate number** – Guest feedback metrics
  - **calculated host listings count** – Number of listings per host
  - **house_rules, license** – Additional property details

This dataset combines structured data (numeric values like price, availability) and categorical data (room type, cancellation policy), making it suitable for statistical analysis, visualization, and predictive modeling.

Step 4
Explain who is affected by this problem and why the solution matters.

- This problem impacts Airbnb hosts, guests, and the company itself. Hosts often struggle to set competitive prices and choose service policies that attract bookings while maintaining profitability. Guests, on the other hand, face inconsistent pricing and varying service standards, which can lead to dissatisfaction and reduced trust in the platform. For Airbnb, these inefficiencies affect overall revenue, customer retention, and brand reputation. Solving this problem matters because data-driven insights can help hosts make informed decisions, improve guest experiences, and enable Airbnb to strengthen its market position through optimized pricing and enhanced service quality.

Step 5
List the tools, libraries, and technologies you plan to use.

Operating system

- Windows

Software

- Python: Version 3.7
- Microsoft Excel

IDE/Environment

- Visual Studio (VS) Code with Python extension

Libraries

- Pandas (for data manipulation)
- Matplotlib (for plotting and visualization)
- Seaborn (for statistical data visualization)

Tasks

Task 1: Loading the dataset

Step 1
Read the CSV file and load it into a panda DataFrame.

Code Used:

```
combinedtaskscode.py > ...
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# --- Task 1: Load dataset ---
df = pd.read_csv(r"D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Open_Data.csv")

```

Step 2
Display the first five rows of the dataset to understand the data format.

Code Used:

```
# Display first 5 rows and data types
print(df.head())
```

Displayed Output:

```
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis> & C:/Users/ROG/AppData/Local/Programs/Python/Python310/python.exe "d:/Everything APU - Study/Per
sonal/Jobs/Personal Projects/AirBnB dataset/Airbnb_Analysis/task1_load_dataset.py"
d:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis\task1_load_dataset.py:3: DtypeWarning: Columns (25) have mixed types. Specify dtype option on import
 or set low_memory=False.
  df = pd.read_csv("D:\\Everything APU - Study\\Personal\\Jobs\\Personal Projects\\AirBnB dataset\\Airbnb_Open_Data.csv")
        id                                  NAME      host id  ... availability 365                                        house_rules license
0  1001254            Clean & quiet apt home by the park  80014485718  ...            286.0  Clean up and treat the home the way you'd like...     NaN
1  1002102                         Skylit Midtown Castle  52335172823  ...            228.0  Pet friendly but please confirm with me if the...     NaN
2  1002403           THE VILLAGE OF HARLEM....NEW YORK !  78829239556  ...            352.0  I encourage you to use my kitchen, cooking and...     NaN
3  1002755                                           NaN  85098326012  ...            322.0                                                NaN     NaN
4  1003689  Entire Apt: Spacious Studio/Loft by central park  92037596077  ...            289.0  Please no smoking in the house, porch or on th...     NaN
```

Step 3
Display the data types of each column to understand the nature of each variable and identify any potential issues with data types.

Code Used:

```
# Display first 5 rows and data types
print(df.head())
print(df.dtypes)

```

Displayed Output:

```
[5 rows x 26 columns]
id                                 int64
NAME                              object
host id                            int64
host_identity_verified            object
host name                         object
neighbourhood group               object
neighbourhood                     object
lat                              float64
long                             float64
country                           object
country code                      object
instant_bookable                  object
cancellation_policy               object
room type                         object
Construction year                float64
price                             object
service fee                       object
minimum nights                   float64
number of reviews                float64
last review                       object
reviews per month                float64
review rate number               float64
calculated host listings count   float64
availability 365                 float64
house_rules                       object
license                           object
dtype: object
```

Task 2: Removing unnecessary fields

Objective: In this task, you will clean the dataset by removing irrelevant or unnecessary fields that could impact the accuracy of the analysis. You can use a tool of your choice to perform this task.

Step 1
Remove the following unwanted columns from the dataset: host id, id, country, and country code. Be sure to include screenshots of the dataset before and after eliminating these columns.

Before Removal:

```
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis> & C:/Users/ROG/AppData/Local/Programs/Python/Python310/python.exe "
sonal/Jobs/Personal Projects/AirBnB dataset/Airbnb_Analysis/task1_load_dataset.py"
d:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis\task1_load_dataset.py:3: DtypeWarning: Columns (25) have mixed types. S
 or set low_memory=False.
  df = pd.read_csv("D:\\Everything APU - Study\\Personal\\Jobs\\Personal Projects\\AirBnB dataset\\Airbnb_Open_Data.csv")
        id                                  NAME      host id  ... availability 365                                        house_rules license
0  1001254            Clean & quiet apt home by the park  80014485718  ...            286.0  Clean up and treat the home the way you'd like...     NaN
1  1002102                         Skylit Midtown Castle  52335172823  ...            228.0  Pet friendly but please confirm with me if the...     NaN
2  1002403           THE VILLAGE OF HARLEM....NEW YORK !  78829239556  ...            352.0  I encourage you to use my kitchen, cooking and...     NaN
3  1002755                                           NaN  85098326012  ...            322.0                                                NaN     NaN
4  1003689  Entire Apt: Spacious Studio/Loft by central park  92037596077  ...            289.0  Please no smoking in the house, porch or on th...     NaN
```

After Removal:

```
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis> & C:/Users/ROG/AppData/Local/Programs/Python/Python310/python.exe "d:/Everything APU - Study/Per
sonal/Jobs/Personal Projects/AirBnB dataset/Airbnb_Analysis/task2_removingfields.py"
d:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb Analysis\task2 removingfields.py:4: DtypeWarning: Columns (25) have mixed types. Specify dtype option on impo
rt or set low_memory=False.
  df = pd.read_csv("D:\\Everything APU - Study\\Personal\\Jobs\\Personal Projects\\AirBnB dataset\\Airbnb_Open_Data.csv")
                                              NAME host_identity_verified host name  ... availability 365                                        house_rules  license
0               Clean & quiet apt home by the park            unconfirmed  Madaline  ...            286.0  Clean up and treat the home the way you'd like...      NaN
1                            Skylit Midtown Castle               verified     Jenna  ...            228.0  Pet friendly but please confirm with me if the...      NaN
2              THE VILLAGE OF HARLEM....NEW YORK !                    NaN     Elise  ...            352.0  I encourage you to use my kitchen, cooking and...      NaN
3                                              NaN            unconfirmed     Garry  ...            322.0                                                NaN      NaN
4  Entire Apt: Spacious Studio/Loft by central park               verified    Lyndon  ...            289.0  Please no smoking in the house, porch or on th...      NaN

[5 rows x 22 columns]
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis>
```

Code Used:

```python
# --- Task 2: Remove unwanted columns safely ---
df.drop(columns=['host id', 'id', 'country', 'country code'], errors='ignore', inplace=True)

# Display first 5 rows after removal
print("\nAfter removing unwanted columns:")
print(df.head())
```

Step 2
Document the reason for excluding these columns from your data analysis in your project documentation. This will demonstrate your ability to distinguish between relevant and irrelevant data.

Reason for Removing Columns:

- id and host id are unique identifiers that do not provide analytical value for pricing or service improvement.
- country and country code are redundant because the analysis focuses on neighborhood-level trends, making country-level data unnecessary.

Task 3: Handling missing values and duplicate records

Objective: In this task, you will handle missing values and duplicate records to improve the quality and reliability of the dataset before analysis. Use Python to complete this task.

Step 1

Identify missing values in the dataframe and display their count in ascending order. If any values are missing, impute them according to the data type of the columns.

Displayed Output Missing Values Per Coloumn:

```
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis> & C:/Users/ROG/AppData/Local/Programs/Python/Python310/python.exe "d:/Everything APU - Study/Per
sonal/Jobs/Personal Projects/AirBnB dataset/Airbnb_Analysis/task3_cleaning.py"
d:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis\task3_cleaning.py:4: DtypeWarning: Columns (25) have mixed types. Specify dtype option on import or
set low_memory=False.
  df = pd.read_csv(r"D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Open_Data.csv")
Missing values per column:
room type                          0
long                               8
lat                                8
neighbourhood                     16
neighbourhood group               29
cancellation_policy               76
instant_bookable                 105
number of reviews                183
Construction year                214
price                            247
NAME                             250
service fee                      273
host_identity_verified           289
calculated host listings count   319
review rate number               326
host name                        406
minimum nights                   409
availability 365                 448
reviews per month              15879
last review                    15893
house_rules                    52131
license                       102597
dtype: int64
d:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis\task3_cleaning.py:23: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
```

Displayed Output After Imputation:

```
After imputation:
NAME                              0
host_identity_verified            0
host name                         0
neighbourhood group               0
neighbourhood                     0
lat                               0
long                              0
instant_bookable                  0
cancellation_policy               0
room type                         0
Construction year                 0
price                             0
service fee                       0
minimum nights                    0
number of reviews                 0
last review                       0
reviews per month                 0
review rate number                0
calculated host listings count    0
availability 365                  0
house_rules                       0
license                           0
dtype: int64
PS D:\Everything APU - Study\Personal\Jobs\Personal Projects\AirBnB dataset\Airbnb_Analysis>
```

Code Used:

```python
# --- Task 3: Handle missing values ---
print("\nMissing values per column:")
print(df.isnull().sum().sort_values(ascending=True))

# Impute missing values
for col in df.columns:
    if df[col].isnull().sum() > 0:
        if df[col].dtype in ['float64', 'int64']:
            df[col].fillna(df[col].median(), inplace=True)
        else:
            df[col].fillna('Unknown', inplace=True)

# Verify no missing values remain
print("\nAfter imputation:")
print(df.isnull().sum())
```

Step 2

Check whether there are any duplicate values in the dataframe and if present, remove them.

Displayed Output:

```
Duplicates before: 3444, after removal: 0
Total records after cleaning: 99155
```

Code Used:

```python
# Remove duplicates
duplicates_before = df.duplicated().sum()
df.drop_duplicates(inplace=True)
duplicates_after = df.duplicated().sum()
print(f"\nDuplicates before: {duplicates_before}, after: {duplicates_after}")
print(f"Total records after cleaning: {len(df)}")
```

Step 3

Display the total number of records in the dataframe before and after removing the duplicates to confirm that data cleaning has been effectively performed.

Displayed Output of total records after cleaning:

```
Duplicates before: 3444, after removal: 0
Total records after cleaning: 99155
```

Code Used:

```python
# Remove duplicates
duplicates_before = df.duplicated().sum()
df.drop_duplicates(inplace=True)
duplicates_after = df.duplicated().sum()
print(f"\nDuplicates before: {duplicates_before}, after: {duplicates_after}")
print(f"Total records after cleaning: {len(df)}")
```

Task 4: Transforming data

Objective: In this task, you will standardize and transform the data to ensure consistency in your analysis. You can use a tool of your choice to perform this task.

Step 1
Rename the column ***availability 365*** to ***days_booked*** to make the column name more intuitive and business-friendly.

Renamed Coloumn List:

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

Code Used:

```
# --- Task 4: Rename and standardize columns ---
df.rename(columns={'availability 365': 'days_booked'}, inplace=True)
df.columns = df.columns.str.lower().str.replace(' ', '_')
print("\nColumns after standardization:")
print(df.columns)
```

Step 2
Convert all column names to lowercase and replace the spaces in the column names with an underscore symbol.

Coloumns after standardization:

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

Code Used:

```
# --- Task 4b: Clean financial columns safely ---
if 'price' in df.columns:
    df['price'] = df['price'].replace('[\$,]', '', regex=True)
    df['price'] = pd.to_numeric(df['price'], errors='coerce')

if 'service_fee' in df.columns:
    df['service_fee'] = df['service_fee'].replace('[\$,]', '', regex=True)
    df['service_fee'] = pd.to_numeric(df['service_fee'], errors='coerce')

```

Step 3
Remove the dollar sign and comma from the columns ***price*** and ***service_fee***. If necessary,

convert these two columns to the appropriate data type. This is to ensure that financial data can be analyzed correctly.

Cleaned financial coloumns:

```
First 5 rows after cleaning financial columns:
   price  service_fee
0  966.0        193.0
1  142.0         28.0
2  620.0        124.0
3  368.0         74.0
4  204.0         41.0
```

Code Used:

```
# --- Task 4b: Clean financial columns safely ---
if 'price' in df.columns:
    df['price'] = df['price'].replace('[\$,]', '', regex=True)
    df['price'] = pd.to_numeric(df['price'], errors='coerce')

if 'service_fee' in df.columns:
    df['service_fee'] = df['service_fee'].replace('[\$,]', '', regex=True)
    df['service_fee'] = pd.to_numeric(df['service_fee'], errors='coerce')

```

Task 5: Exploring data

Objective: In this task, you will perform exploratory data analysis to identify meaningful trends that can answer business questions. You can use a tool of your choice to perform this task.

Step 1
Display the count of various room types available in the dataset.

Count of room types:

```
Room type counts:
room_type
Entire home/apt    51995
Private room       44895
Shared room         2150
Hotel room           115
Name: count, dtype: int64
```

Code Used:

```
# --- Task 5: Analysis ---
# Room type counts
print("\nRoom type counts:")
print(df['room_type'].value_counts())
```

Step 2
Display the room type with the strictest cancellation policy. In your project documentation, describe the risk factors for both hosts and guests.

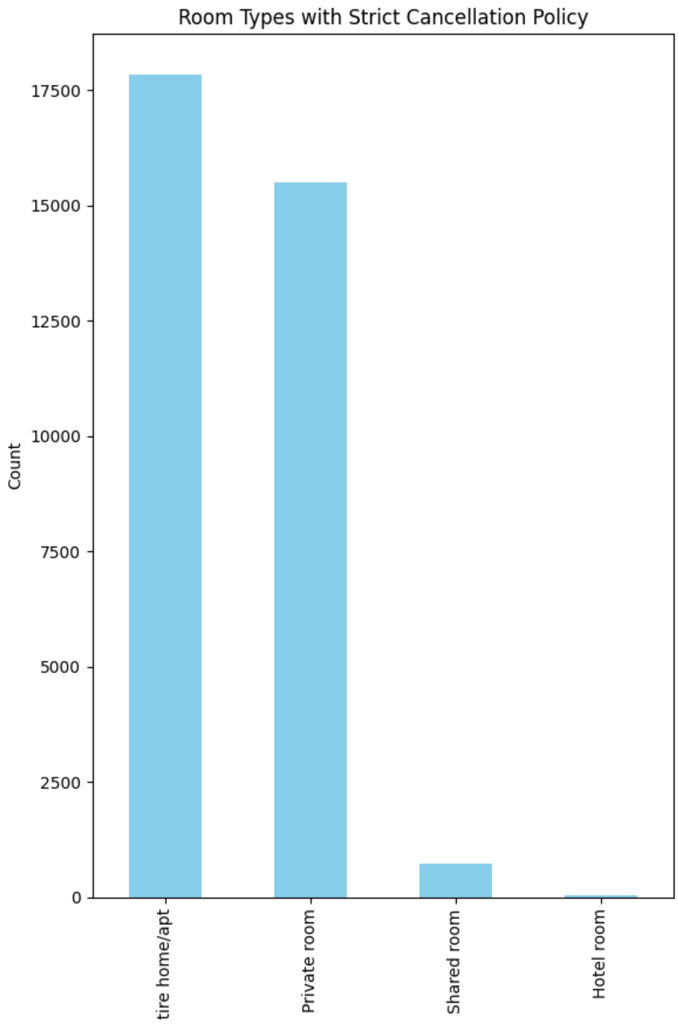
Room types with strictest policy:

```
Room types under strict cancellation policy:
room_type
Entire home/apt    17240
Private room       14937
Shared room          718
Hotel room            34
Name: count, dtype: int64

Room type with most strict policies: Entire home/apt
```

Code Used:

```
# Strict cancellation policy analysis
strict_df = df[df['cancellation_policy'] == 'strict']
room_type_counts = strict_df['room_type'].value_counts()
print("\nRoom types under strict cancellation policy:")
print(room_type_counts)
print("\nRoom type with most strict policies:", room_type_counts.idxmax())
```

Graph Generated:



Risk factors for both hosts and guests:

Hosts:

- Pros: Reduces last-minute cancellations, ensures revenue stability.
- Cons: May discourage bookings, especially for short stays.

Guests:

- Pros: None (except assurance host is serious).
- Cons: High financial risk if plans change; less flexibility.

Step 3

Display the average price per neighborhood group. In your project documentation, mention the most expensive neighborhood for rentals. This will help Airbnb understand pricing patterns and identify premium areas.

The most expensive neighbourhood group is Queens with an average price of $630.21.

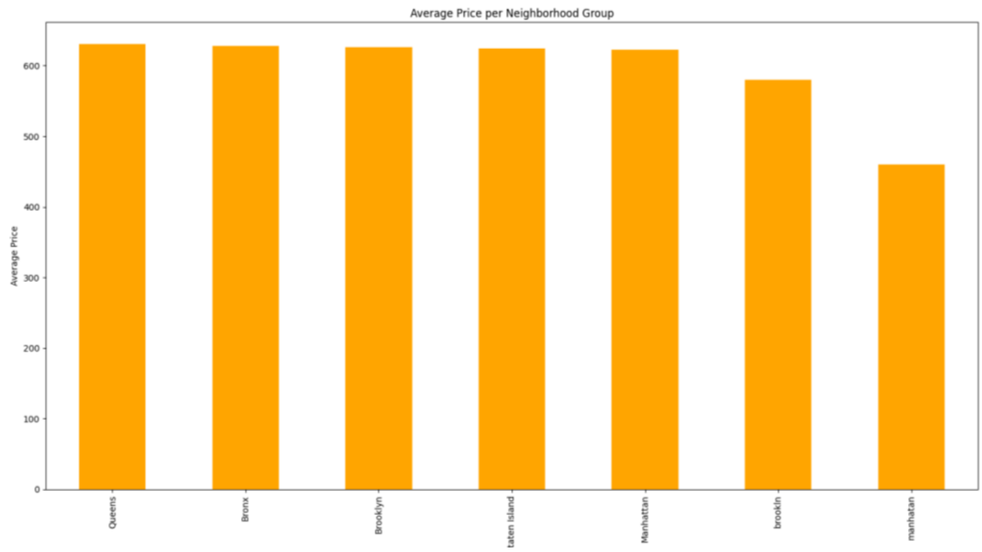
Avg price per neighborhood:

```
Average price per neighborhood group:
neighbourhood_group
Unknown          657.206897
Queens           629.700623
Bronx            626.610092
Brooklyn         626.444673
Staten Island    626.427174
Manhattan        622.714633
brookln          580.000000
manhatan         460.000000
Name: price, dtype: float64

Most expensive neighborhood group: Unknown with an average price of $657.21
```

Code Used:

```python
# Average price per neighborhood group
avg_price_by_group = df.groupby('neighbourhood_group')['price'].mean().sort_values(ascending=False)
print("\nAverage price per neighborhood group:")
print(avg_price_by_group)
print(f"\nMost expensive neighborhood group: {avg_price_by_group.idxmax()} with an average price of ${avg_price_by_group.max():.2f}")

# Visualization for average price
avg_price_by_group.plot(kind='bar', title='Average Price per Neighborhood Group', color='orange')
plt.xlabel('Neighborhood Group')
plt.ylabel('Average Price')
plt.show()
```

Graph Generated:



Task 6: Visualizing data, part 1

Objective: In this task, you will create visualizations to effectively communicate key findings and trends. This task focuses on visualizing price distribution and location/type-based comparisons.

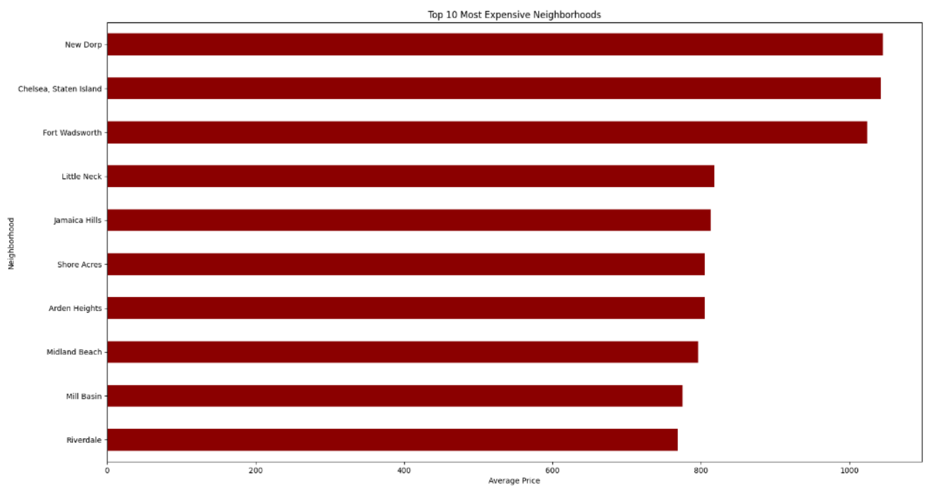
Step 1

Create a horizontal bar chart to display the top 10 most expensive neighborhoods in the dataset. Create another chart with the 10 least expensive neighborhoods in the dataset.

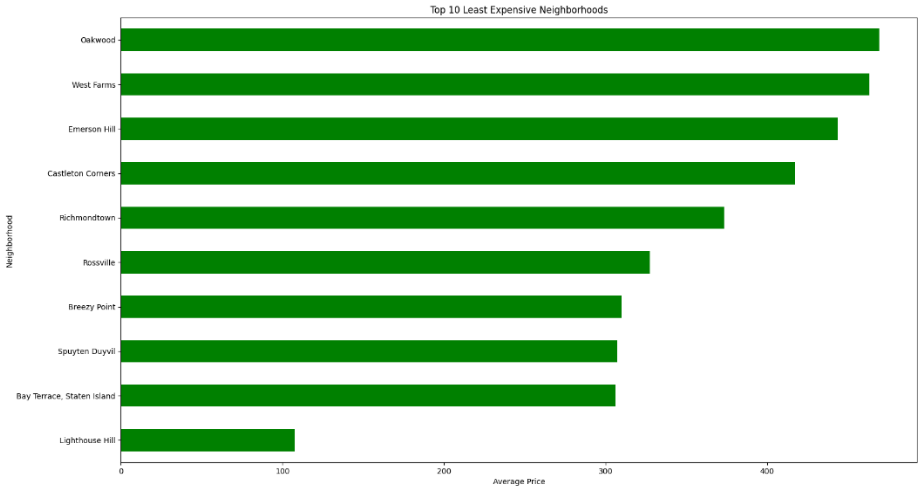
Code used to identify top 10 most expensive neighborhoods:

```
# ------------------------------
# Step 1: Top 10 Most and Least Expensive Neighborhoods
# Group by neighborhood and calculate average price
avg_price_by_neighborhood = df.groupby('neighbourhood')['price'].mean().sort_values(ascending=False)

# Top 10 most expensive neighborhoods
top10_expensive = avg_price_by_neighborhood.head(10)

# Top 10 least expensive neighborhoods
bottom10_expensive = avg_price_by_neighborhood.tail(10)

# Plot Top 10 Most Expensive Neighborhoods
plt.figure(figsize=(10, 6))
top10_expensive.plot(kind='barh', color='darkred')
plt.title('Top 10 Most Expensive Neighborhoods')
plt.xlabel('Average Price')
plt.ylabel('Neighborhood')
plt.gca().invert_yaxis()  # Highest price at top
plt.tight_layout()
plt.show()

# Plot Top 10 Least Expensive Neighborhoods
plt.figure(figsize=(10, 6))
bottom10_expensive.plot(kind='barh', color='green')
plt.title('Top 10 Least Expensive Neighborhoods')
plt.xlabel('Average Price')
plt.ylabel('Neighborhood')
plt.gca().invert_yaxis()  # Lowest price at top
plt.tight_layout()
plt.show()
```

Top 10 most expensive neighborhoods:



Top 10 least expensive neighborhoods:



Step 2
Create a box and whisker chart that showcases the price distribution of all listings split by room type to understand how prices vary within different accommodation categories.

Code Used:

```
# ---------------------------------
# Step 2: Box and Whisker Plot for Price Distribution by Room Type
plt.figure(figsize=(10, 6))
sns.boxplot(x='room_type', y='price', data=df)
plt.title('Price Distribution by Room Type')
plt.xlabel('Room Type')
plt.ylabel('Price')
plt.tight_layout()
plt.show()
```

Price distribution box and whisker chart:

Task 7: Visualizing data, part 2

Objective: In this task, you will create visualizations that highlight relationships between different factors, like cleaning fees and room prices. This task explores trends and correlations.

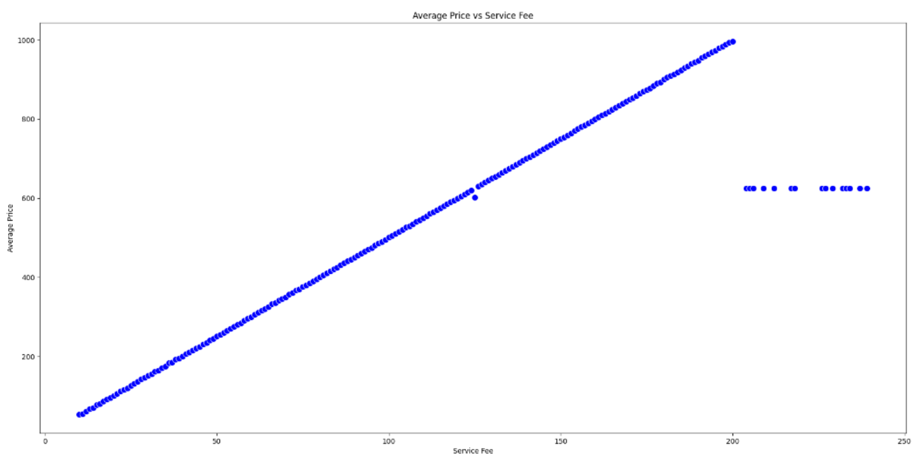
Step 1

Create a scatter plot to illustrate the relationship between the cleaning fee and room price. Note any correlation in your project documentation and comment whether cleaning fees influence pricing strategies.

Code Used:

```python
# --- Task 7: Analysis ---
# Standardize column names
df.columns = df.columns.str.lower().str.replace(' ', '_')

# Convert numeric columns
df['price'] = pd.to_numeric(df['price'], errors='coerce')
fee_column = 'cleaning_fee' if 'cleaning_fee' in df.columns else 'service_fee'
df[fee_column] = pd.to_numeric(df[fee_column], errors='coerce')

# Drop missing values
df = df.dropna(subset=['price', fee_column])

# Remove extreme outliers
df = df[(df['price'] <= 1000) & (df[fee_column] <= 500)]

# Aggregate by fee
agg_df = df.groupby(fee_column)['price'].mean().reset_index()

# ------------------------------
# Scatter plot with aggregated data
plt.figure(figsize=(10, 6))
sns.scatterplot(x=fee_column, y='price', data=agg_df, color='blue', s=80)
plt.title(f'Average Price vs {fee_column.replace("_", " ").title()}')
plt.xlabel(fee_column.replace("_", " ").title())
plt.ylabel('Average Price')
plt.tight_layout()
plt.show()

```

Displayed output:



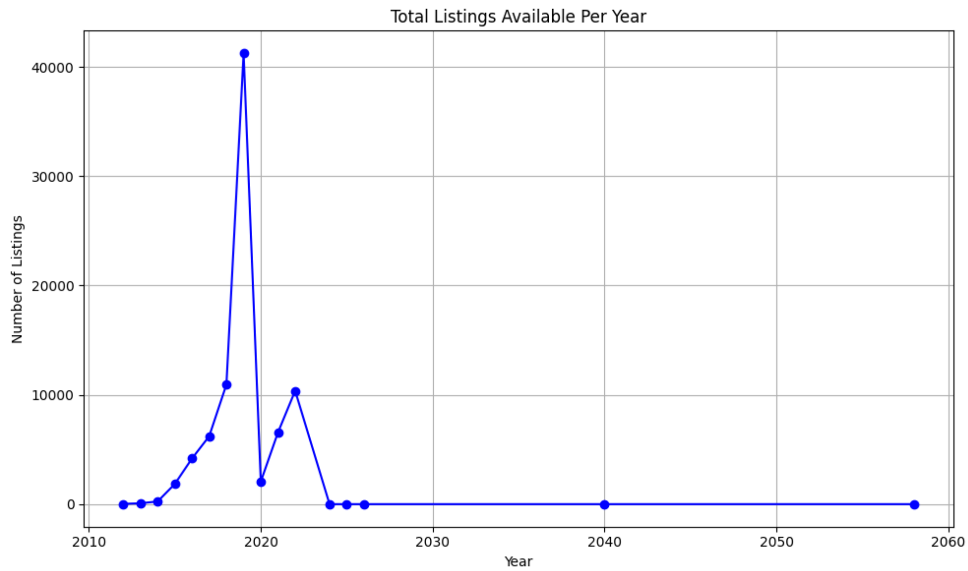
Step 2

Create a line chart to showcase the total amount of listings available per year.

Code used:

```
# ------------------------------
# Line chart for listings per year
if 'last_review' in df.columns:
    df['last_review'] = pd.to_datetime(df['last_review'], errors='coerce')
    df['year'] = df['last_review'].dt.year
elif 'construction_year' in df.columns:
    df['year'] = pd.to_numeric(df['construction_year'], errors='coerce')

listings_per_year = df['year'].value_counts().sort_index()

plt.figure(figsize=(10, 6))
plt.plot(listings_per_year.index, listings_per_year.values, marker='o', color='blue')
plt.title('Total Listings Available Per Year')
plt.xlabel('Year')
plt.ylabel('Number of Listings')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Displayed output:



Task 8: Visualizing data, part 3

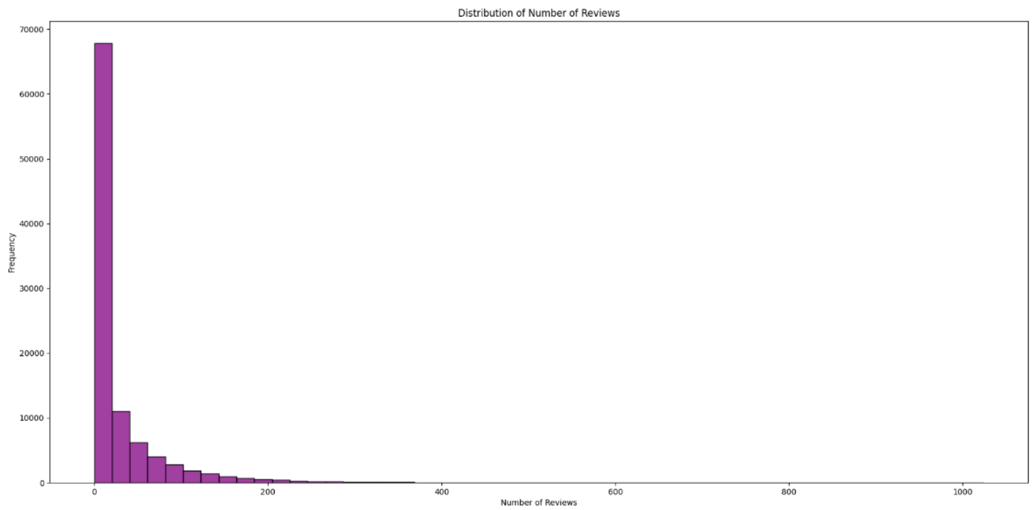
Objective: In this task, you will analyze host performance by comparing Superhosts and regular hosts to identify what makes a successful host. This task focuses on qualitative metrics like reviews and host-specific performance.

Step 1
Create a data visualization of your choice using one of the review columns in isolation or combination with another column.

Code used:

```python
# --- Task 8: Analysis ---
# Standardize column names
df.columns = df.columns.str.lower().str.replace(' ', '_')

# Convert numeric columns
df['price'] = pd.to_numeric(df['price'], errors='coerce')
df['number_of_reviews'] = pd.to_numeric(df['number_of_reviews'], errors='coerce')
df['review_rate_number'] = pd.to_numeric(df['review_rate_number'], errors='coerce')

# Drop rows with missing values in key columns
df = df.dropna(subset=['price', 'number_of_reviews'])

# ------------------------------
# Step 1: Visualization using review column
plt.figure(figsize=(10, 6))
sns.histplot(df['number_of_reviews'], bins=50, color='purple')
plt.title('Distribution of Number of Reviews')
plt.xlabel('Number of Reviews')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()

```

Displayed output:



Step 2

Create a visualization to compare at least two different variables between Superhosts and regular hosts. Identify the performance differences between highly-rated and average hosts and document your findings in your project documentation.

Code used:

```python
# ------------------------------
# Step 2: Compare Superhosts vs Regular Hosts
# Use 'host_identity_verified' as proxy for Superhost status
df['superhost'] = df['host_identity_verified'].apply(lambda x: 'Superhost' if str(x).lower() == 'verified' else 'Regular')

# Group by Superhost status and calculate average price and average reviews
comparison = df.groupby('superhost').agg({'price': 'mean', 'number_of_reviews': 'mean'}).reset_index()

# Melt for plotting
comparison_melted = comparison.melt(id_vars='superhost', value_vars=['price', 'number_of_reviews'],
                                    var_name='Metric', value_name='Average Value')

# Plot comparison
plt.figure(figsize=(10, 6))
sns.barplot(x='Metric', y='Average Value', hue='superhost', data=comparison_melted)
plt.title('Comparison of Superhosts vs Regular Hosts')
plt.xlabel('Metric')
plt.ylabel('Average Value')
plt.tight_layout()
plt.show()
```

Displayed output: