Design Proposal Report

Airbnb Business Analysis Using A Data Science Approach

1. Executive Summary

Airbnb specialises in providing short and long-term stays and experiences to holidaymakers and business travellers globally. In the last decade, Airbnb has experienced a massive upsurge in bookings and listings globally and its popularity amongst clients has grown, making it the preferred marketplace for homestays. The growth of Airbnb has inevitably generated massive data which if explored and analysed could be essential in providing business and customer insights and trends. The following report explores internal Airbnb New York City data to further help this growth (KeyData, 2024).

2. Introduction

Airbnb's marketing team has hired RTH Consultants to provide a comprehensive analysis to the proposed problem: '*What are customers in New York City who book premium listings on Airbnb looking for?*'. This context is important for Airbnb to continue to be the leader in the B&B market. Understanding these preferences helps Airbnb tailor its premium services, attract high-value guests, and stay competitive in the luxury travel market. In reviewing the internal

reservation dataset (Why Always Me, 2019), 1/5th (or 21.45%) of their customers tend to book premium listings.

3. Business Understanding

a. Business Objectives

This report discusses a data science-backed analysis to help Airbnb analyse their internal reservation dataset to identify the patterns that drive the premium bookings on the platform.

b. Expected Business Impact

Airbnb wants to increase their revenue and continue being the leader in the bed-and-breakfast (B&B) service market, especially due to the increase in competition from other B&B's and hotels in New York City.

c. Dataset

AirBnB has provided the dataset for New York City for this analysis (Why Always Me, 2019).

4. Methodology

a. Exploratory Data Analysis (EDA)

To best review this analytical question, EDA was first performed. Python was utilised as the main coding framework, within which certain libraries were imported to assist with the EDA tasks, as well as visualisation, preprocessing, and clustering.

The dataset was first explored by tabular and statistical visualisation. Thereafter, the preprocessing of data was performed by inquiring about missing values in the dataset. The rows containing the missing values (about 20.6%) were perceived not to impact the analysis itself and therefore were not removed. Next, preprocessing was conducted by dropping the columns that would not be utilised in the analysis.

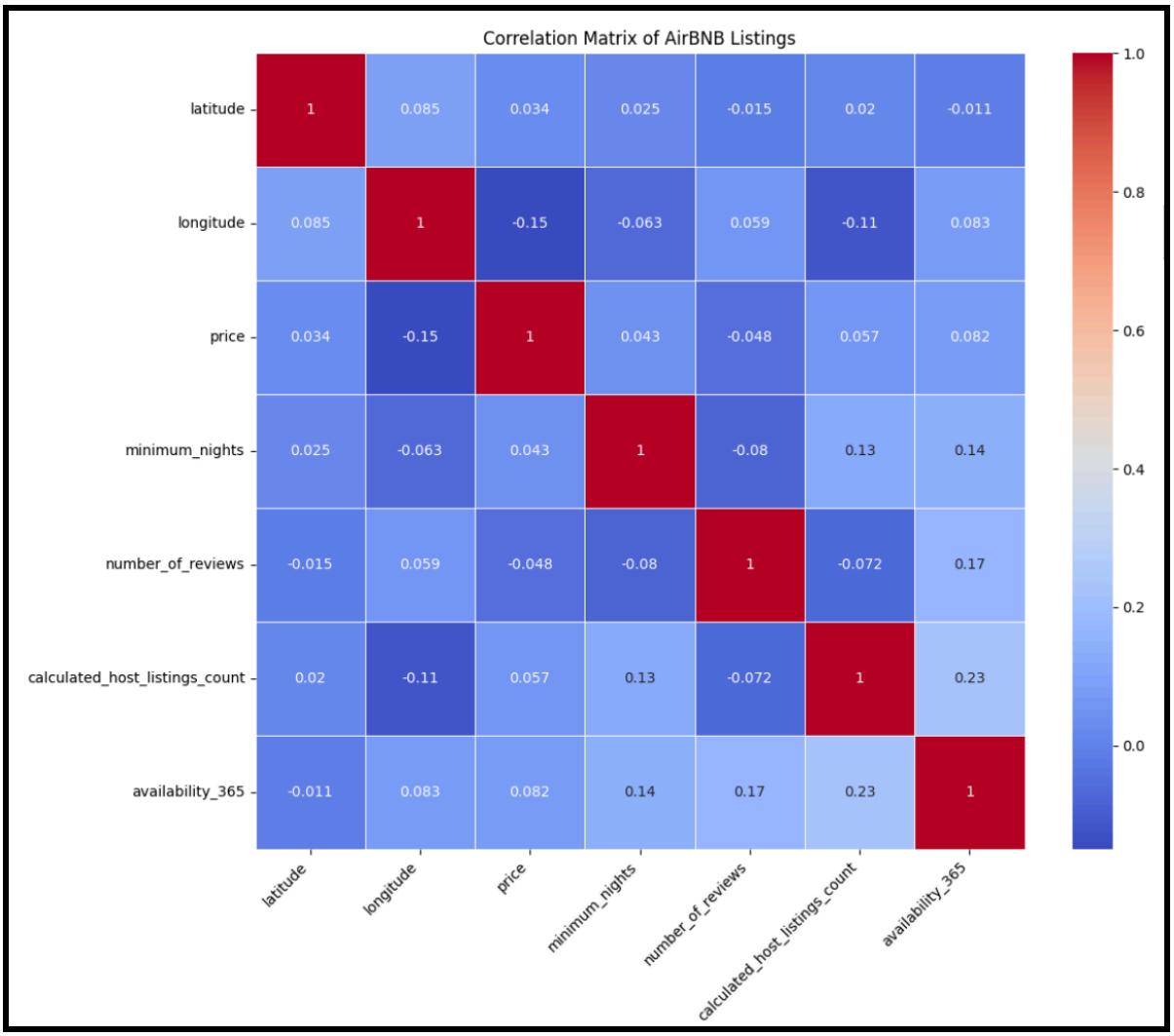

Figure 1: Correlation Matrix Heatmap

A correlation matrix heatmap was created to understand if any of the variables (i.e. columns) have any correlation between them, which they did not (Wagavkar, 2021).

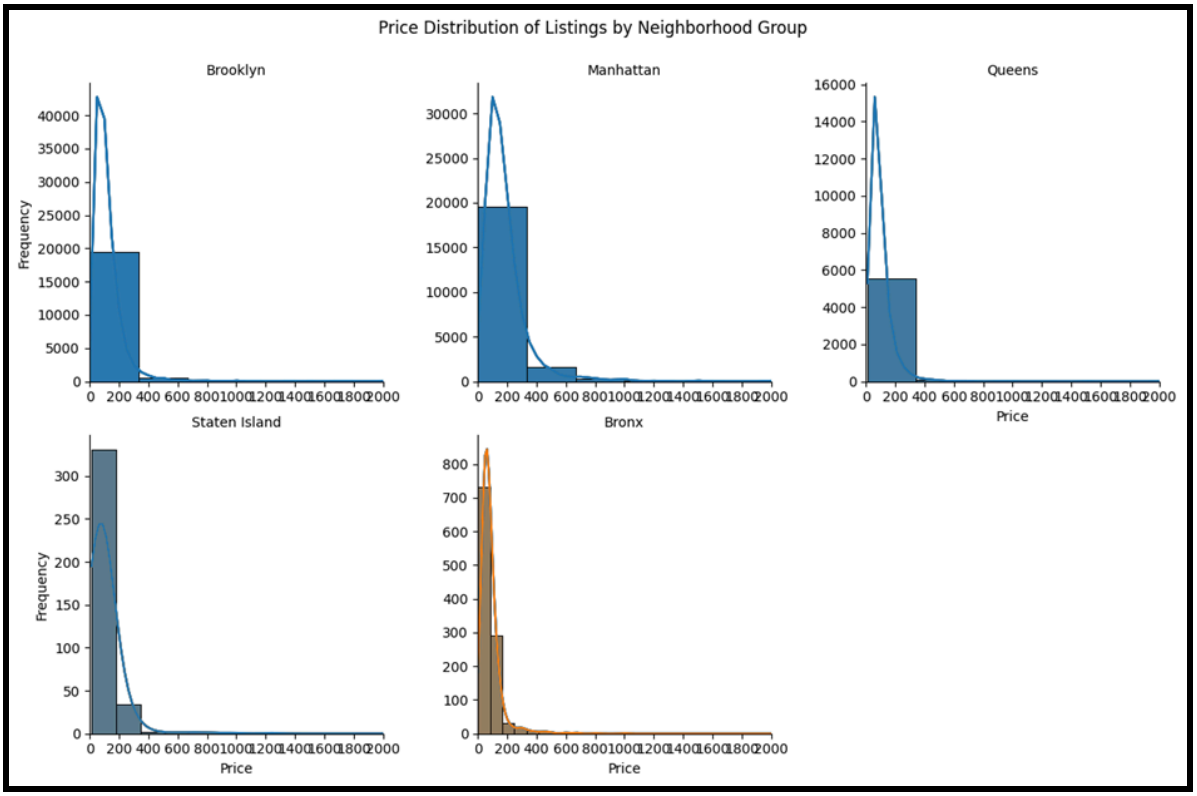

Figure 2: Histograms

Histograms of the price for each neighbourhood group were plotted to see the distribution of prices as well as to find out which neighbourhood group has the biggest spenders (Plapinger, 2022).

Manhattan and Brooklyn both have the highest frequency of listings and the widest range of prices, including significant high-priced outliers. Queens has a moderate frequency of listings with generally lower prices compared to Manhattan and Brooklyn. Staten Island and the Bronx both have lower frequencies of listings and more affordable prices, with limited outliers in the higher price ranges.

b. Clustering and Unsupervised Machine Learning

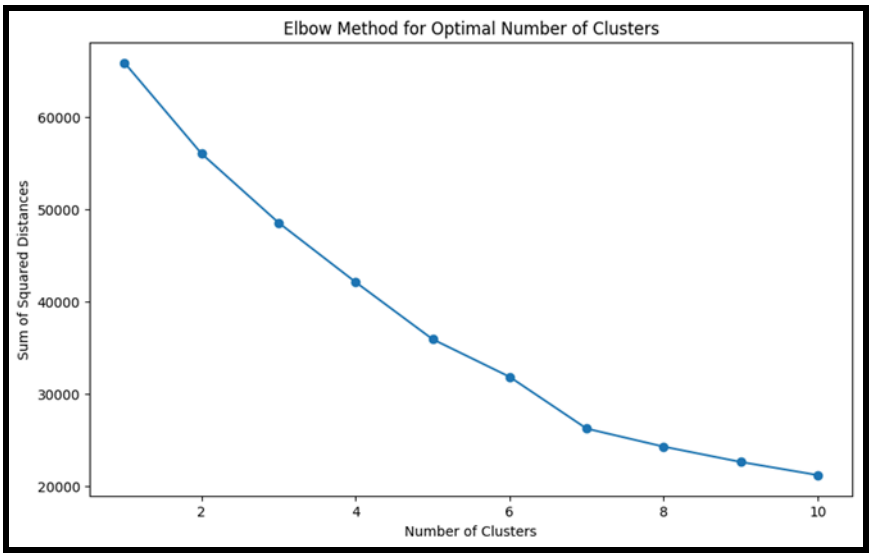

Figure 3: Elbow Method

To get a more detailed insight into the preferences of Airbnb customers who are willing to spend more, the focus was centered on the top 25% of spenders from Manhattan and Brooklyn. Null values were removed before clustering was conducted. Using K-means clustering, four distinct clusters of customers were identified. The optimal number of clusters was determined using the elbow method (Rushirajsinh, 2023).

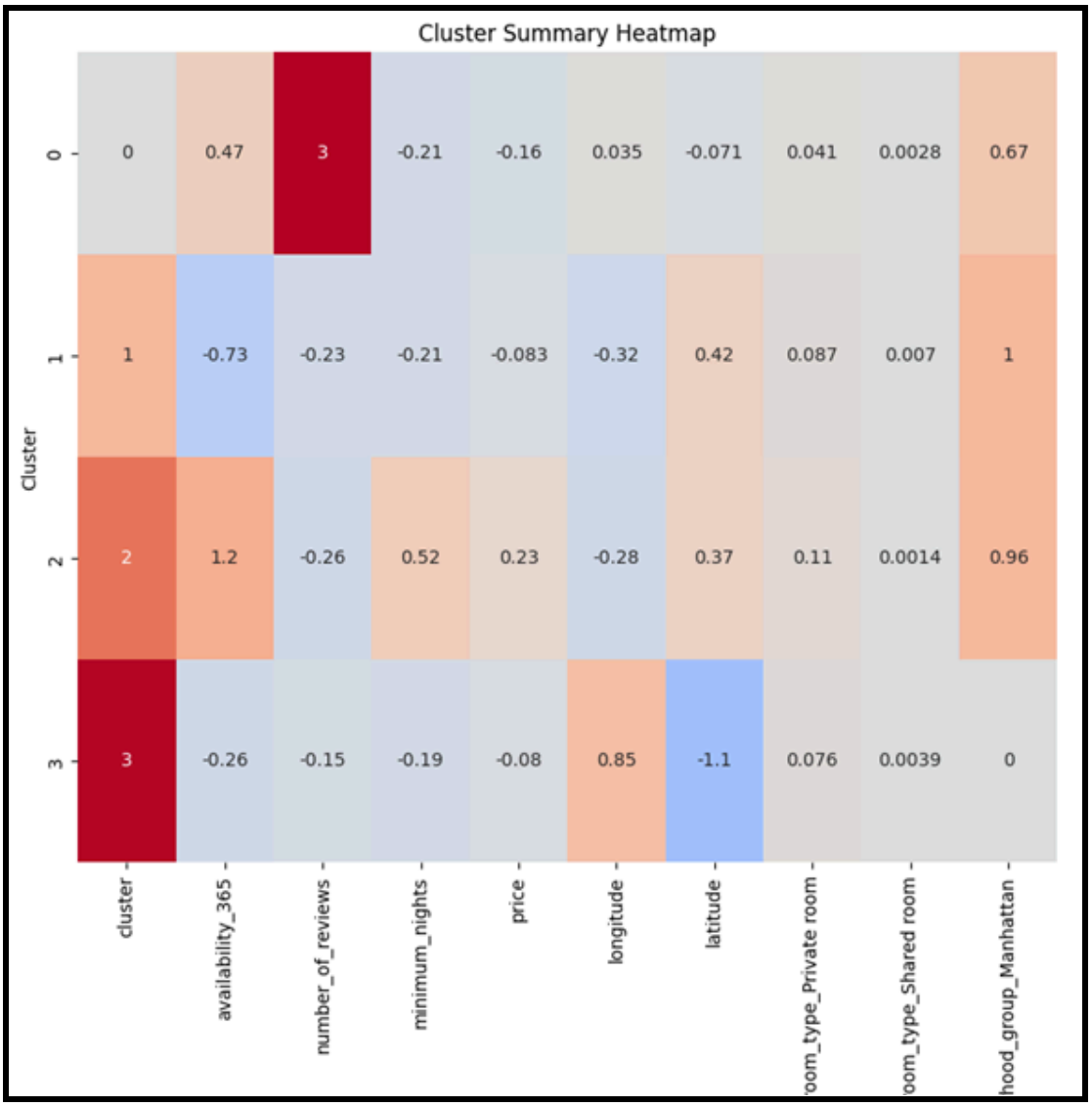

Figure 4: Cluster Summary Heatmap

Cluster 2 represents the high spenders who prioritise comfort and extended accommodations. These luxury travellers prefer entire homes or apartments in Manhattan. They seek higher availability and opt for longer stays, willing to pay premium prices for their stays. This cluster indicates that these customers value space and duration, making them ideal targets for premium, high-end listings.

Clusters 0, 1, and 3 encompass customers who spend more than the average, but are not necessarily the highest spenders. Cluster 0 consists of travellers who value highly-reviewed and popular listings in Manhattan. They prefer shorter stays and predominantly choose private rooms, emphasising review quality and value for money.

Cluster 1 includes cost-effective travellers looking for short stays in high-demand listings, also in Manhattan, and primarily opting for entire homes or apartments. Cluster 3 represents budget-conscious travellers in Brooklyn who seek affordable entire homes or apartments, balancing cost and comfort with moderately popular listings and shorter stays.

These insights can help tailor marketing strategies and improve listing offerings to meet the specific needs of high-spending customers, enhancing their overall Airbnb experience.

5. Marketing Insights

RTH Consultants would suggest the following marketing strategies based on the above predictions. Based on Cluster 2, Airbnb can highlight the luxury and exclusivity of entire homes in Manhattan. Focus on comfort, extended stays, and premium amenities. Use personalised offers, loyalty programs, and testimonials from previous high-spending guests.

Additionally, regarding Cluster 0, Airbnb could emphasise the exceptional reviews and high popularity of private rooms in Manhattan. They can showcase quality and value for

the money. Furthermore, they can use targeted ads featuring guest satisfaction and top-rated hosts, with discounts for shorter stays.

Moreover, Cluster 1 indicates that they should promote the affordability and convenience of entire homes in high-demand Manhattan areas, and highlight flexible booking options, last-minute deals, and short-stay promotions (while also emphasising the cost-quality balance).

Cluster 3 highlights budget-conscious spenders in Brooklyn. Airbnb should emphasise the affordability and comfort of entire homes in this area, showcasing value-for-money and unique neighbourhood experiences. They can promote discounts for extended stays, and special offers for budget travellers to attract these efficient spenders (S source, 2023).

6. Conclusion

This report discusses the analysis conducted by RTH consultants to understand what motivates customers to make premium reservations (if money is not a constraint). Utilising the dataset provided by Airbnb, four clusters of types of customers were produced to reflect what motivates a certain subset of people to make reservations. Cluster 2 represents the high spenders who prioritise comfort, extended accommodations, high availability, space and duration of the booking. This is the target audience for this analysis and can be further incentivized using comfort, extended stays, and premium amenities including offering the entire home for the booking. Other identified clusters also provide some interesting insights should Airbnb choose to target

those clusters in a varied manner moving forward. Such data science-targeted marketing can aid in revenue increase, furthering Airbnb's main goal herein.

7. References

KeyData (2024) *Exploring airbnb's surge: Key Growth Statistics and what they mean for property managers: Key Data*, *RSS*. Available at: https://www.keydatadashboard.com/en-gb/blog/exploring-airbnbs-surge-key-growth-statistics-and-what-they-mean-for-property-managers (Accessed: 09 June 2024).

Plapinger, T. (2022) *Visualising your exploratory data analysis*, *Medium*. Available from: https://towardsdatascience.com/visualizing-your-exploratory-data-analysis-d2d6c2e3b30e [Accessed 09 June 2024].

Rushirajsinh, Z. (2023) The Elbow Method: Finding The Optimal Number Of Clusters, Medium. Available from: https://medium.com/@zalarushirajsinh07/the-elbow-method-finding-the-optimal-number-of-clusters-d297f5aeb189 [Accessed 09 June 2024].

S source (2023) What is sales promotion: Sales Promotion in marketing, Medium. Available from: https://medium.com/@ssource.blr/what-is-sales-promotion-sales-promotion-in-marketing-a78c4a71ebd5#:~:text=It%20is%20typically%20a%20short,%2C%20sweepstakes%2C%20and%20free%20samples [Accessed 09 June 2024].

Wagavkar, S. (2021) *Correlation matrix*, *Medium*. Available from: https://medium.com/analytics-vidhya/correlation-matrix-5e764bcee34 [Accessed 09 June 2024].

Why Always Me (2019) AB_NYC_2019 Python - New York City Airbnb Open Data. Available from: https://www.kaggle.com/code/whyalwaysme/ab-nyc-2019 [Accessed 09 June 2024].

8. Appendix (Python Code)

```
#Libraries and Packages

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

df = pd.read_csv('/AB_NYC_2019.csv')

df.head()
df.tail()

# Calculate the total number of rows with at least one missing value
total_rows_with_missing_values = df.isna().any(axis=1).sum()

# Print the total number of rows with missing values
print(f'Total number of rows with at least one missing value: {total_rows_with_missing_values}')

df.isna().sum()

"""From the results, over 10k listings do not have reviews. Will this have a bearing on our ML algorithm if we use last_review and reviews_per_month"""

df.describe()

# Drop the 'id', 'host_id', and 'reviews_per_month' columns
df = df.drop(columns=['id', 'host_id', 'reviews_per_month', 'last_review', 'name'])


# Select only numeric columns for the correlation matrix
numeric_cols = df.select_dtypes(include=[np.number])

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

# Plot the correlation matrix as a heatmap
```

```
plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix of AirBNB Listings')
plt.xticks(rotation=45, ha='right')  # Rotate x labels
plt.yticks(rotation=0)  # Keep y labels horizontal
plt.show()

"""- No Correlation between any of the variables"""

neighborhood_colors = {
    'Brooklyn': 'dodgerblue',
    'Manhattan': 'darkorange',
    'Queens': 'forestgreen',
    'Staten Island': 'mediumvioletred',
    'Bronx': 'gold'
}

# Create the FacetGrid
g = sns.FacetGrid(df, col="neighbourhood_group", col_wrap=3, height=4, sharex=False,
sharey=False)

# Map the histogram plot with custom colors
for neighborhood, color in neighborhood_colors.items():
    g.map_dataframe(sns.histplot, "price", bins=30, kde=True, color=color,
                    hue=df['neighbourhood_group'] == neighborhood)

# Set titles and labels
g.set_titles("{col_name}")
g.set_axis_labels("Price", "Frequency")

# Adjust the x-axis limits and ticks for better readability
for ax in g.axes.flat:
    ax.set_xlim(0, 2000)  # Set the x-axis limit to a more reasonable range
    ax.set_xticks(range(0, 2001, 200))  # Set the ticks at intervals of 200

plt.subplots_adjust(top=0.9)
g.fig.suptitle('Price Distribution of Listings by Neighborhood Group')

plt.show()

# Filter listings for Manhattan, Brooklyn, and Queens
filtered_df = df[df['neighbourhood_group'].isin(['Manhattan', 'Brooklyn', 'Queens'])]

# Select relevant columns for summary statistics
```

```
columns_of_interest = ['neighbourhood_group', 'availability_365', 'number_of_reviews',
'minimum_nights', 'price', 'longitude', 'latitude']

# Compute mean and standard deviation for each neighborhood group
summary_stats = filtered_df[columns_of_interest].groupby('neighbourhood_group').agg(['mean',
'std']).reset_index()

# Flatten the MultiIndex columns
summary_stats.columns = ['_'.join(col).strip() for col in summary_stats.columns.values]

# Rename columns for better readability
summary_stats.rename(columns={
    'neighbourhood_group_': 'Neighborhood Group',
    'availability_365_mean': 'Availability Mean',
    'availability_365_std': 'Availability Std',
    'number_of_reviews_mean': 'Reviews Mean',
    'number_of_reviews_std': 'Reviews Std',
    'minimum_nights_mean': 'Min Nights Mean',
    'minimum_nights_std': 'Min Nights Std',
    'price_mean': 'Price Mean',
    'price_std': 'Price Std',
    'longitude_mean': 'Longitude Mean',
    'longitude_std': 'Longitude Std',
    'latitude_mean': 'Latitude Mean',
    'latitude_std': 'Latitude Std'
}, inplace=True)

# Style the DataFrame with a single color scheme
styled_stats = summary_stats.style.format({
    "Availability Mean": "{:.2f}", "Availability Std": "{:.2f}",
    "Reviews Mean": "{:.2f}", "Reviews Std": "{:.2f}",
    "Min Nights Mean": "{:.2f}", "Min Nights Std": "{:.2f}",
    "Price Mean": "{:.2f}", "Price Std": "{:.2f}",
    "Longitude Mean": "{:.5f}", "Longitude Std": "{:.5f}",
    "Latitude Mean": "{:.5f}", "Latitude Std": "{:.5f}"
}).background_gradient(cmap='Blues', subset=[
    "Availability Mean", "Availability Std", "Reviews Mean", "Reviews Std",
    "Min Nights Mean", "Min Nights Std", "Price Mean", "Price Std",
    "Longitude Mean", "Longitude Std", "Latitude Mean", "Latitude Std"
])

styled_stats

# Filter listings for Manhattan, Brooklyn as they have the highest average prices
```

```
filtered_df = df[df['neighbourhood_group'].isin(['Manhattan', 'Brooklyn'])]

filtered_df = filtered_df.sort_values(by='price', ascending=False)

# Select the top quartile based on price
top_quartile_threshold = filtered_df['price'].quantile(0.75)
top_quartile_df = filtered_df[filtered_df['price'] >= top_quartile_threshold]

top_quartile_df[columns_of_interest].groupby('neighbourhood_group').agg(['mean',
'std']).reset_index()

top_quartile_df.isna().sum()

top_quartile_df = top_quartile_df.dropna()

# Select relevant columns
columns_of_interest = ['availability_365', 'number_of_reviews', 'minimum_nights', 'price',
'longitude', 'latitude', 'room_type', 'neighbourhood_group']

# Filter the relevant columns
top_quartile_df = top_quartile_df[columns_of_interest]


# Define the preprocessing steps for numerical and categorical data
numerical_features = ['availability_365', 'number_of_reviews', 'minimum_nights', 'price',
'longitude', 'latitude']
categorical_features = ['room_type', 'neighbourhood_group']

numerical_transformer = StandardScaler()

# One-hot encode categorical features
categorical_transformer = OneHotEncoder(drop='first')

# Combine the preprocessing steps
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Apply the transformations
X_preprocessed = preprocessor.fit_transform(top_quartile_df)
```

```
X_preprocessed

sse = []
k_range = range(1, 11)
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_preprocessed)
    sse.append(kmeans.inertia_)

# Plot the results
plt.figure(figsize=(10, 6))
plt.plot(k_range, sse, marker='o')
plt.title('Elbow Method for Optimal Number of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('Sum of Squared Distances')
plt.show()

# Perform K-means clustering with the optimal number of clusters (e.g., k=4)
kmeans = KMeans(n_clusters=4, random_state=42)
clusters = kmeans.fit_predict(X_preprocessed)

# Get feature names after transformation
num_features = numerical_features
cat_features =
preprocessor.named_transformers_['cat'].get_feature_names_out(categorical_features)
feature_names = list(num_features) + list(cat_features)

# Create a DataFrame from the preprocessed data
X_preprocessed_df = pd.DataFrame(X_preprocessed, columns=feature_names)

# Add the cluster labels to the original DataFrame
X_preprocessed_df['cluster'] = clusters

# Group by the cluster labels and calculate the mean for each cluster
cluster_summary = X_preprocessed_df.groupby('cluster').mean().reset_index()

# Display the summary table
print(cluster_summary)


# Plot the heatmap with features on the y-axis and clusters on the x-axis
plt.figure(figsize=(12, 8))
sns.heatmap(cluster_summary, annot=True, cmap='coolwarm', center=0)
plt.title('Cluster Summary Heatmap')
```

```
plt.xlabel('Features')  # X-axis label for clusters
plt.ylabel('Cluster')  # Y-axis label for features
plt.show()
```