kamalova / NYC-Airbnb-Recommendation-Engine-NLP Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

NYC-Airbnb-Recommendation-Engine-NLP / notebooks / Listings_EDA.ipynb

kamalova Created using Colaboratory

History

1 contributor

4.8 MB

Open in Colab

Airbnb Recommendation Engine for NYC through Sentiment Analysis

Author: Nurgul Kurbanali kyzy

Table of Contents

1. Business Case

About Airbnb: *You can host anything, anywhere, so guests can enjoy everything, everywhere.*

Nowadays the demand for short and long-term temporary accommodation is increasing thanks to easing travel conditions. This demand positively affects the number of online platforms that allow you to make reservations before traveling. **Airbnb** is one such platform, which allows travelers to make accommodation reservations based on the fact that the host leases all or part of his or her home to the traveler.

Customer reviews play an important role in the customer's decision to purchase a product or use a service. Customer preferences and opinions are affected by other customers' reviews online, on blogs or over social networking platforms

The main goal of this work is to combine both recommendation system and sentiment analysis in order to recommend the most accurate listings for users based on their preferences in **New York City**. Since both domains suffer from the lack of labeled data, to overcome that, this project detects the opinions polarity score using **NLTK VADER** (Valence Aware Dictionary and Sentiment Reasoner) Lexicon.

We'll therefore split our approaches into following sections:

- Exploring available AirBnb listings in NYC

- Exploring available AirBnb listings in NYC
- Measuring polarity/sentiment scores along with vader_lexicon. This polarity

measurement adapts to *pos, neu, neg*, and compound. By simply taking the compound from these values, a new feature was created on the data.

- Building a recommendation engine with Collaborative Filtering to predict sentiment score for all reviewer-listing pairs and making personalised recommendations for each user based on their ranked preferences.

2. Aim of this Notebook

This Notebook covers steps starting from loading listing datasets and merging them together. Further implemented basic EDA that covers data understanding, preparation and exploration. With the help of data visualization I will try to uncover some basic statistical patterns within the dataset. Eventually, notebook gives some fundemental statistical informations about the **Airbnb** listings within the **New York City** during 2022.

💻 Project Notebook was run in **Google Colab**

3. Data Understanding

The dataset is obtained from Inside Airbnb. It is is a mission driven project that provides data and advocacy about Airbnb's impact on residential communities. For the purpose of this project we downloaded the most recent quarterly datasets between *December, 2021 - September, 2022* which includes information and metrics for listings in **New York City**. Dataset includes 153199 entries and 75 columns in total that have been adjusted and decreased eventually after applying some data preprocessing. Let's discover further in detail.

3.1. Importing Required Libraries

In [1]:

```python
import numpy as np
import pandas as pd
pd.set_option('display.max_colwidth', None)

# Data visualization
import seaborn as sns
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
```

```
%matplotlib inline
# Seaborn's beautiful styling
import seaborn as sns
sns.set_style('whitegrid')
# Text Preprocessing
import string
string.punctuation
import re

from wordcloud import WordCloud, STOPWORDS
# to get rid of the warnings
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
# Remove sample_data file in Colab
%rm -rf sample_data/
```

3.2. Data Load

In [3]:

```
# Loading datasets to Colab
list_dec = pd.read_csv('/content/listings_dec21.csv', compression='gzip', on_bad_lines='skip',
                       low_memory=False);
list_march = pd.read_csv('/content/listings_march.csv', compression='gzip', on_bad_lines='skip',
                       low_memory=False);
list_jun = pd.read_csv('/content/listings_jun.csv', compression='gzip', on_bad_lines='skip',
                       low_memory=False);
list_sep = pd.read_csv('/content/listings_sep.csv', compression='gzip', on_bad_lines='skip',
                       low_memory=False);
```

In [4]:

```
# Display dimensionality of the DataFrames
print(list_march.shape, list_jun.shape, list_sep.shape, list_dec.shape)
```

```
(37631, 74) (37410, 74) (39881, 75) (38277, 74)
```

In [5]:

```
# Concatinate loaded Dataframes together
df_listings = pd.concat([list_march,list_jun, list_sep,list_dec])
```

In [6]:

```
# Print first 5 rows of DataFrame
df_listings.head()
df_listings.head()
```

Out[6]:

	id	listing_url	scrape_id	last_scraped	name	
0	2595	https://www.airbnb.com/rooms/2595	20220305031505	2022-03-05	Skylit Midtown Castle	STUNNING SKYLIT STU - Gorgeous pyramid skylight with amazin seating area with natural zafu cushions, mo
1	5121	https://www.airbnb.com/rooms/5121	20220305031505	2022-03-05	BlissArtsSpace!	Thank you all for your support. I've traveled a One room available for rent in a 2 bedroc $900permonthforoneperson.Utilitiesnotinc$ per night short term. If you are a couple please
2	5136	https://www.airbnb.com/rooms/5136	20220305031505	2022-03-05	Spacious Brooklyn Duplex, Patio + Garden	We welcome you to stay in our lovely 2 br c Sleeps 4 We are locatec Note: This is our home, we live here with ou

3	5178	https://www.airbnb.com/rooms/5178	20220305031505	2022-03-05	Large Furnished Room Near B'way	Th
4	5203	https://www.airbnb.com/rooms/5203	20220305031505	2022-03-30	Cozy Clean Guest Room - Family Apt	Our best guests are seeking a safe, clean, spare and aren't afraid of a friendly two year old golde war Your guest room is comfortable and clean. It is bathroom is shared and immediately across the

5 rows × 75 columns

In [7]:

```
#  Print information about a DataFrame
df_listings.info()
```

```
Int64Index: 153199 entries, 0 to 38276
Data columns (total 75 columns):
 #   Column                                        Non-Null Count   Dtype
---  ------                                        --------------   -----
 0   id                                            153199 non-null  int64
 1   listing_url                                   153199 non-null  object
 2   scrape_id                                     153199 non-null  int64
 3   last_scraped                                  153199 non-null  object
 4   name                                          153145 non-null  object
 5   description                                   149268 non-null  object
```

```
6   neighborhood_overview                         90954 non-null   object 
7   picture_url                                   153199 non-null  object 
8   host_id                                       153199 non-null  int64  
9   host_url                                      153199 non-null  object 
10  host_name                                     152962 non-null  object 
11  host_since                                    152962 non-null  object 
12  host_location                                 145829 non-null  object 
13  host_about                                    87041 non-null   object 
14  host_response_time                            94929 non-null   object 
15  host_response_rate                            94929 non-null   object 
16  host_acceptance_rate                          100648 non-null  object 
17  host_is_superhost                             152983 non-null  object 
18  host_thumbnail_url                            152962 non-null  object 
19  host_picture_url                              152962 non-null  object 
20  host_neighbourhood                            122416 non-null  object 
21  host_listings_count                           152962 non-null  float64
22  host_total_listings_count                     152962 non-null  float64
23  host_verifications                            153199 non-null  object 
24  host_has_profile_pic                          152962 non-null  object 
25  host_identity_verified                        152962 non-null  object 
26  neighbourhood                                 90958 non-null   object 
27  neighbourhood_cleansed                        153199 non-null  object 
28  neighbourhood_group_cleansed                  153199 non-null  object 
29  latitude                                      153199 non-null  float64
30  longitude                                     153199 non-null  float64
31  property_type                                 153199 non-null  object 
32  room_type                                     153199 non-null  object 
33  accommodates                                  153199 non-null  int64  
34  bathrooms                                     0 non-null       float64
35  bathrooms_text                                152831 non-null  object 
36  bedrooms                                      137989 non-null  float64
37  beds                                          147990 non-null  float64
38  amenities                                     153199 non-null  object 
39  price                                         153199 non-null  object 
40  minimum_nights                                153199 non-null  int64  
41  maximum_nights                                153199 non-null  int64  
42  minimum_minimum_nights                        153134 non-null  float64
43  maximum_minimum_nights                        153134 non-null  float64
44  minimum_maximum_nights                        153134 non-null  float64
45  maximum_maximum_nights                        153134 non-null  float64
46  minimum_nights_avg_ntm                        153134 non-null  float64
47  maximum_nights_avg_ntm                        153134 non-null  float64
48  calendar_updated                              0 non-null       float64
49  has_availability                              153199 non-null  object 
50  availability_30                               153199 non-null  int64  
51  availability_60                               153199 non-null  int64  
```

```
 52  availability_90                             153199 non-null  int64
 53  availability_365                            153199 non-null  int64
 54  calendar_last_scraped                       153199 non-null  object
 55  number_of_reviews                           153199 non-null  int64
 56  number_of_reviews_ltm                       153199 non-null  int64
 57  number_of_reviews_l30d                      153199 non-null  int64
 58  first_review                                118410 non-null  object
 59  last_review                                 118410 non-null  object
 60  review_scores_rating                        118410 non-null  float64
 61  review_scores_accuracy                      116347 non-null  float64
 62  review_scores_cleanliness                   116388 non-null  float64
 63  review_scores_checkin                       116327 non-null  float64
 64  review_scores_communication                 116365 non-null  float64
 65  review_scores_location                      116315 non-null  float64
 66  review_scores_value                         116313 non-null  float64
 67  license                                     11 non-null      object
 68  instant_bookable                            153199 non-null  object
 69  calculated_host_listings_count              153199 non-null  int64
 70  calculated_host_listings_count_entire_homes  153199 non-null  int64
 71  calculated_host_listings_count_private_rooms 153199 non-null  int64
 72  calculated_host_listings_count_shared_rooms  153199 non-null  int64
 73  reviews_per_month                           118410 non-null  float64
 74  source                                      39881 non-null   object
dtypes: float64(22), int64(17), object(36)
memory usage: 88.8+ MB
```

In [8]:

```
# Drop unnecessary columns
df_listings = df_listings.drop(columns=['scrape_id','listing_url','last_scraped','source','license',
                                     'calendar_last_scraped','last_review','first_review',
                          'number_of_reviews_ltm','number_of_reviews_l30d',
                          'minimum_minimum_nights','maximum_minimum_nights',
                          'minimum_maximum_nights','maximum_maximum_nights',
                          'minimum_nights_avg_ntm','maximum_nights_avg_ntm','host_id','host_since','host_url',
                          'host_listings_count','host_thumbnail_url','host_picture_url','host_verifications','
                          'host_has_profile_pic', 'host_identity_verified','host_neighbourhood','bathrooms_tex
                          'calendar_updated','bedrooms'])
```

In [9]:

```
# Check for dimensionality
df_listings.shape
```

Out[9]: (153199, 43)

In [10]:

```
# Print columns of DataFrame
df_listings.columns
```

Out[10]:

```
Index(['id', 'name', 'description', 'neighborhood_overview', 'picture_url',
       'host_name', 'host_about', 'host_response_time', 'host_response_rate',
       'host_acceptance_rate', 'host_is_superhost',
       'host_total_listings_count', 'neighbourhood', 'neighbourhood_cleansed',
       'neighbourhood_group_cleansed', 'latitude', 'longitude',
       'property_type', 'room_type', 'accommodates', 'beds', 'amenities',
       'price', 'minimum_nights', 'maximum_nights', 'has_availability',
       'availability_30', 'availability_60', 'availability_90',
       'availability_365', 'number_of_reviews', 'review_scores_rating',
       'review_scores_accuracy', 'review_scores_cleanliness',
       'review_scores_checkin', 'review_scores_communication',
       'review_scores_location', 'review_scores_value', 'instant_bookable',
       'calculated_host_listings_count',
       'calculated_host_listings_count_entire_homes',
       'calculated_host_listings_count_private_rooms',
       'calculated_host_listings_count_shared_rooms'],
      dtype='object')
```

In [11]:

```
# Count Null values in each column
df_listings.isna().sum()
```

Out[11]:

```
id                                                  0
name                                               54
description                                      3931
neighborhood_overview                           62245
picture_url                                         0
host_name                                         237
host_about                                      66158
host_response_time                              58270
host_response_rate                              58270
host_acceptance_rate                            52551
host_is_superhost                                 216
host_total_listings_count                         237
neighbourhood                                   62241
neighbourhood_cleansed                              0
neighbourhood_group_cleansed                        0
latitude                                            0
longitude                                           0
property_type                                       0
room_type                                           0
```

```
accommodates                                       0
beds                                            5209
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
review_scores_rating                           34789
review_scores_accuracy                         36852
review_scores_cleanliness                      36811
review_scores_checkin                          36872
review_scores_communication                    36834
review_scores_location                         36884
review_scores_value                            36886
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
dtype: int64
```

In [12]:

```
# Drop Null values
df_listings.dropna(subset=['name', 'description', 'neighborhood_overview',
       'host_name', 'host_about', 'host_response_time', 'host_response_rate',
       'host_acceptance_rate', 'host_is_superhost',
       'host_total_listings_count', 'neighbourhood',
        'beds','review_scores_rating','review_scores_accuracy', 'review_scores_cleanliness',
       'review_scores_checkin', 'review_scores_communication','review_scores_location', 'review_scores_value']
```

In [13]:

```
# Print information about a DataFrame
df_listings.info()
```

```
Int64Index: 31538 entries, 0 to 37873
Data columns (total 43 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   id                                            31538 non-null  int64
 1   name                                          31538 non-null  object
 1   name                                         31538 non-null  object
 2   description                                  31538 non-null  object
 3   neighborhood_overview                        31538 non-null  object
 4   picture_url                                  31538 non-null  object
 5   host_name                                    31538 non-null  object
 6   host_about                                   31538 non-null  object
 7   host_response_time                           31538 non-null  object
 8   host_response_rate                           31538 non-null  object
 9   host_acceptance_rate                         31538 non-null  object
 10  host_is_superhost                            31538 non-null  object
 11  host_total_listings_count                    31538 non-null  float64
 12  neighbourhood                                31538 non-null  object
 13  neighbourhood_cleansed                       31538 non-null  object
 14  neighbourhood_group_cleansed                 31538 non-null  object
 15  latitude                                     31538 non-null  float64
 16  longitude                                    31538 non-null  float64
 17  property_type                                31538 non-null  object
 18  room_type                                    31538 non-null  object
 19  accommodates                                 31538 non-null  int64
 20  beds                                         31538 non-null  float64
 21  amenities                                    31538 non-null  object
 22  price                                        31538 non-null  object
 23  minimum_nights                               31538 non-null  int64
 24  maximum_nights                               31538 non-null  int64
 25  has_availability                             31538 non-null  object
 26  availability_30                              31538 non-null  int64
 27  availability_60                              31538 non-null  int64
 28  availability_90                              31538 non-null  int64
 29  availability_365                             31538 non-null  int64
 30  number_of_reviews                            31538 non-null  int64
 31  review_scores_rating                         31538 non-null  float64
 32  review_scores_accuracy                       31538 non-null  float64
 33  review_scores_cleanliness                    31538 non-null  float64
 34  review_scores_checkin                        31538 non-null  float64
 35  review_scores_communication                  31538 non-null  float64
 36  review_scores_location                       31538 non-null  float64
 37  review_scores_value                          31538 non-null  float64
 38  instant_bookable                             31538 non-null  object
 39  calculated_host_listings_count               31538 non-null  int64
 40  calculated_host_listings_count_entire_homes  31538 non-null  int64
 41  calculated_host_listings_count_private_rooms 31538 non-null  int64
 42  calculated_host_listings_count_shared_rooms  31538 non-null  int64
dtypes: float64(11), int64(13), object(19)
memory usage: 10.6+ MB
```

In [14]:

```
# Check for dimentionality
```

```
df_listings.shape
```

Out[14]:
```
(31538, 43)
```

3.4. Exploratory Data Analysis (EDA)

EDA Host Type

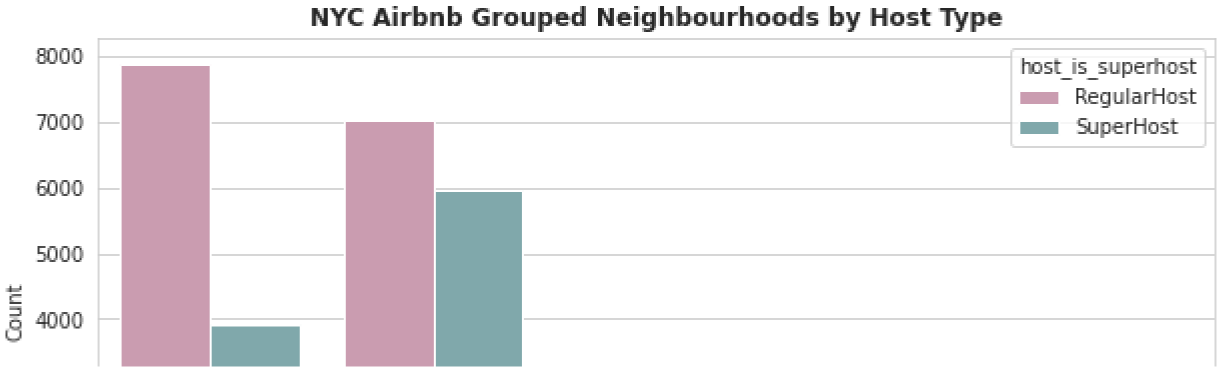
In [ ]:
```
# Count unique values
df_listings.host_is_superhost.value_counts()
```

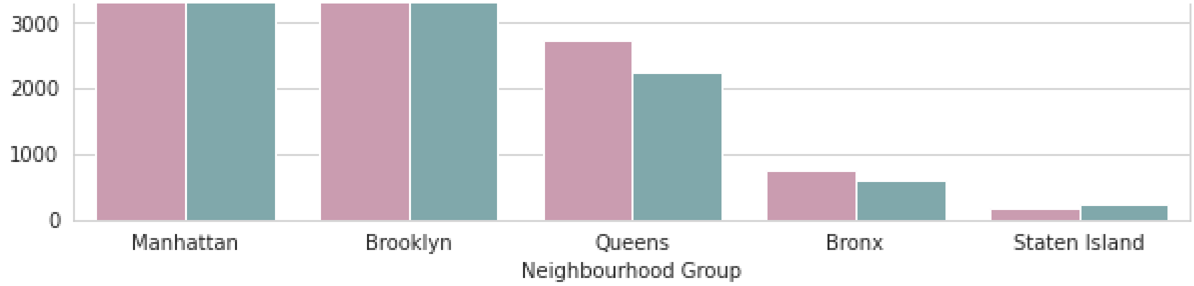
Out[ ]:
```
f    18568
t    12970
Name: host_is_superhost, dtype: int64
```

In [ ]:
```
# Renaming  values within host_is_superhost column
df_listings['host_is_superhost'].replace('t','SuperHost',inplace = True)
df_listings['host_is_superhost'].replace('f','RegularHost',inplace =True)
```

In [ ]:
```
# Plot host type by NYC Neighbourhoods
ax = sns.countplot(df_listings['neighbourhood_group_cleansed'], hue=df_listings.host_is_superhost, palette=['#
fig = plt.gcf()
fig.set_size_inches(10,5)
ax.set_xlabel('Neighbourhood Group')
ax.set_ylabel('Count');
plt.title('NYC Airbnb Grouped Neighbourhoods by Host Type',fontweight="bold")
```

Out[ ]:
```
Text(0.5, 1.0, 'NYC Airbnb Grouped Neighbourhoods by Host Type')
```





Majority of super hosts are from the *Brooklyn* while *Queens, Bronx and Staten Island* have nearly an equal amount of host types

In [ ]:

```
# Remove trailing characters  and change data type into float
df_listings['host_response_rate'] = df_listings['host_response_rate'].str.rstrip('%').astype('float') / 100.0
df_listings['host_acceptance_rate'] = df_listings['host_acceptance_rate'].str.rstrip('%').astype('float') / 10
```

In [ ]:

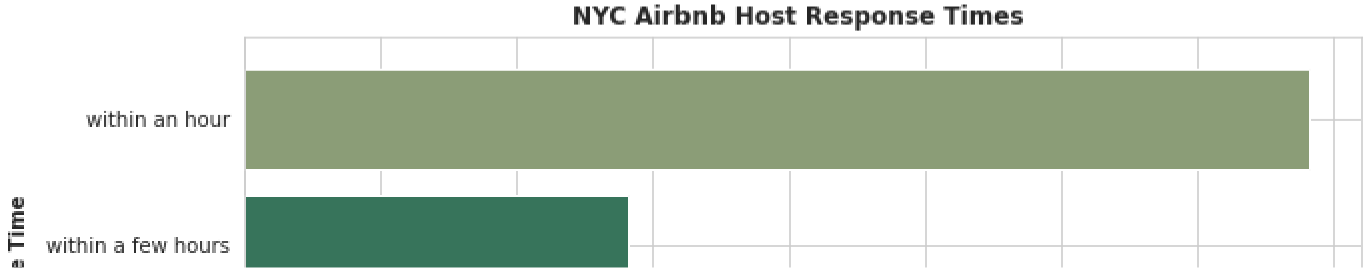
```
df_listings.host_response_time.value_counts()
```

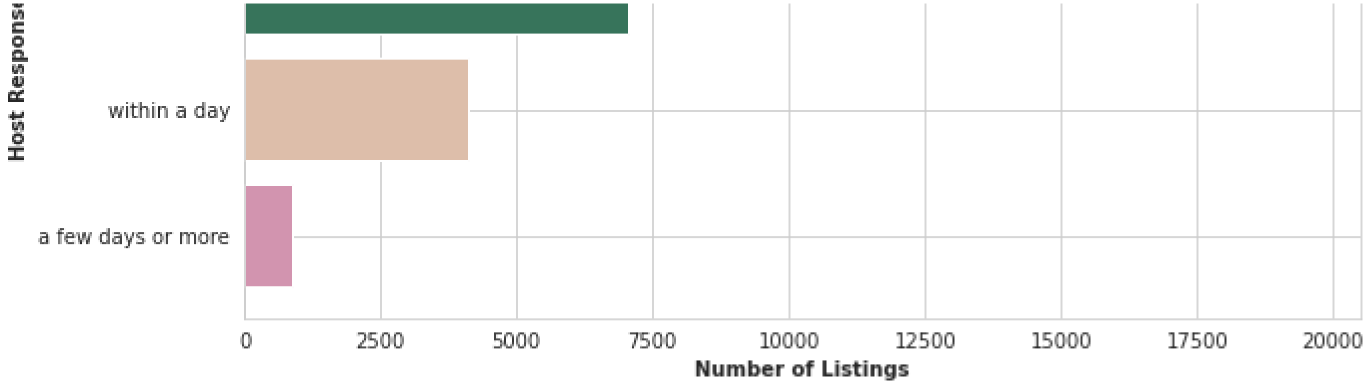
Out[ ]:

```
within an hour        18825
within a few hours     6941
within a day           4028
a few days or more      855
Name: host_response_time, dtype: int64
```

In [ ]:

```
# Plot Host Response Times Frequencies
feq = df_listings['host_response_time'].value_counts().sort_index()
feq.plot.barh(figsize=(10,5), width=0.8, rot=0, color=['#D294AF', '#DDBEAA','#37745B','#8B9D77'])
plt.title('NYC Airbnb Host Response Times ', fontweight="bold")
plt.xlabel('Number of Listings', fontweight="bold")
plt.ylabel('Host Response Time', fontweight="bold")
plt.show()
```

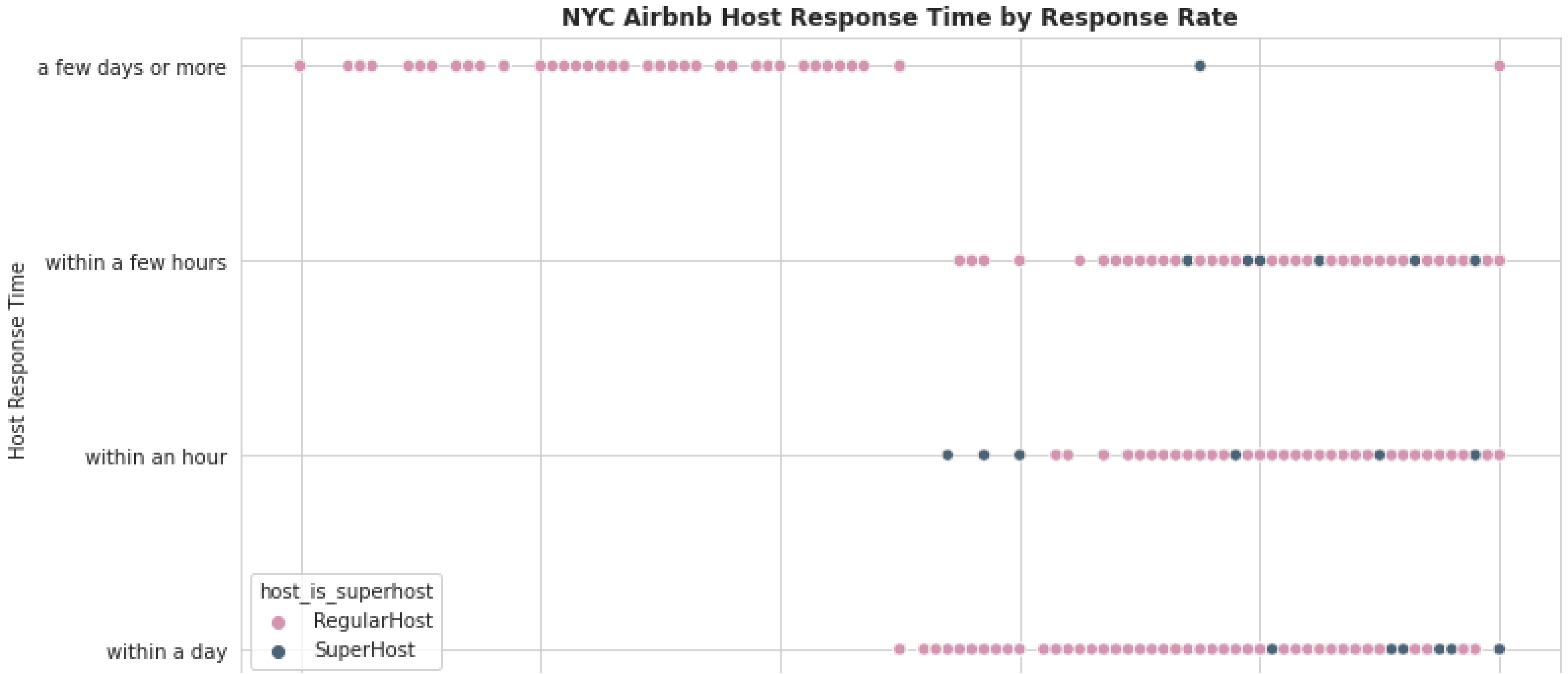




Most hosts respons within an hour up to the few hours. Let's further compare with response rate

In [ ]:

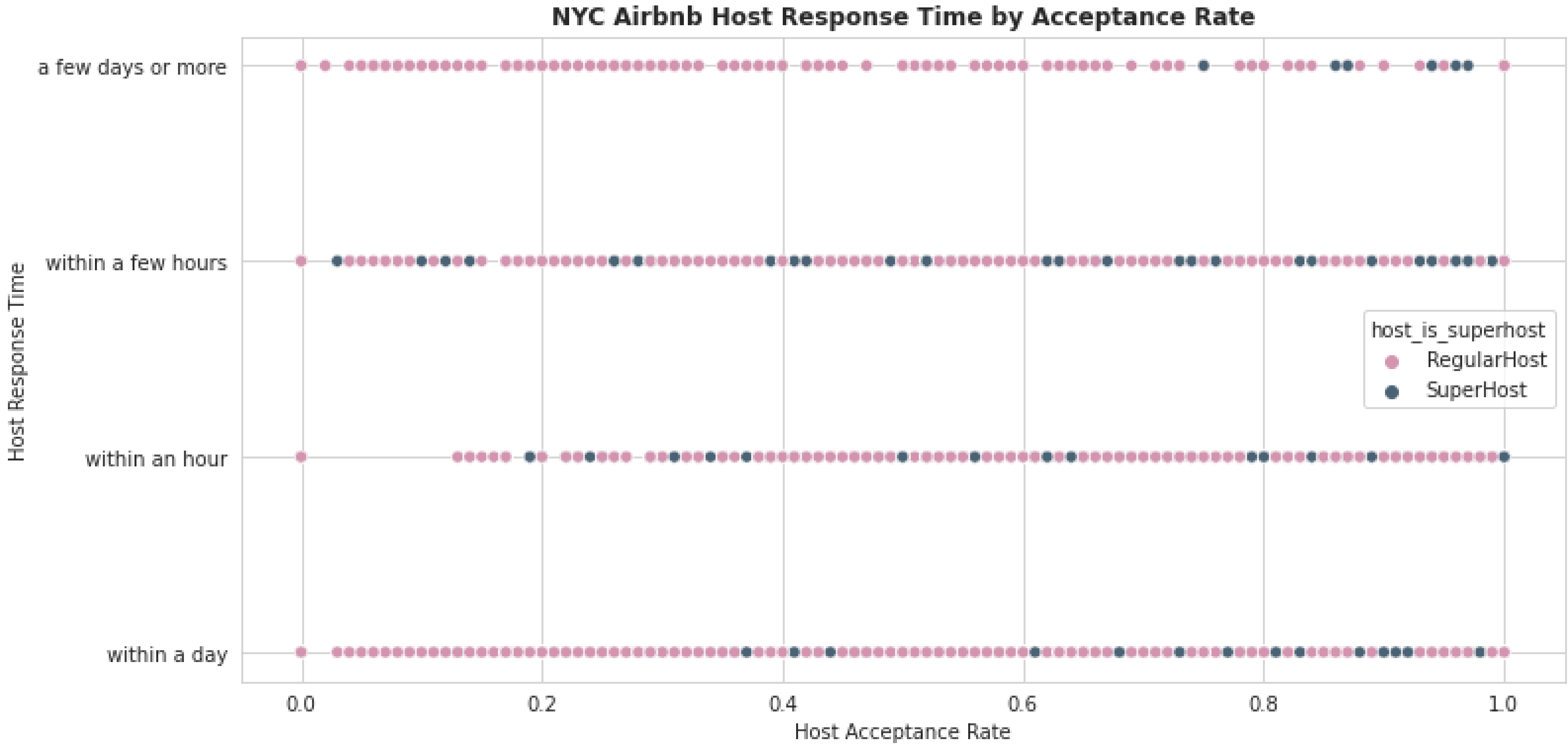
```
# Plot  host_response_time by their response_rate
plt.figure(figsize=(12,6))
sns.scatterplot(df_listings.host_response_rate,df_listings.host_response_time,hue=df_listings.host_is_superhos
plt.ioff()
plt.title('NYC Airbnb Host Response Time by Response Rate',fontweight="bold")
plt.ylabel('Host Response Time')
plt.xlabel('Host Response Rate');
```



The hosts that have responded within a few days or more have been received lower ratings up to 0.45%. from the plot we can see that if hosts can respond within a few hours up to maximum within a da there is higher chance to get better ratings. The majority of the super hosts also fall in this gap which proofs their responsibility.

In [ ]:

```
# Plot host_respons_time by  thier acceptance_rate
plt.figure(figsize=(12,6))
sns.scatterplot(df_listings.host_acceptance_rate,df_listings.host_response_time,hue=df_listings.host_is_superh
plt.ioff()
plt.title('NYC Airbnb Host Response Time by Acceptance Rate',fontweight="bold")
plt.ylabel('Host Response Time')
plt.xlabel('Host Acceptance Rate');
```



In [ ]:

```
#  Get summary of the dataframe
df_listings.info()
```

```
Int64Index: 31538 entries, 0 to 37873
Data columns (total 43 columns):
 #   Column                                          Non-Null Count  Dtype
---  ------                                          --------------  -----
 0   id                                              31538 non-null  int64
 1   name                                            31538 non-null  object
 2   description                                     31538 non-null  object
 3   neighborhood_overview                           31538 non-null  object
 4   picture_url                                     31538 non-null  object
 5   host_name                                       31538 non-null  object
 6   host_about                                      31538 non-null  object
 7   host_response_time                              31538 non-null  object
 8   host_response_rate                              31538 non-null  float64
 9   host_acceptance_rate                            31538 non-null  float64
 10  host_is_superhost                               31538 non-null  object
 11  host_total_listings_count                       31538 non-null  float64
 12  neighbourhood                                   31538 non-null  object
 13  neighbourhood_cleansed                          31538 non-null  object
 14  neighbourhood_group_cleansed                    31538 non-null  object
 15  latitude                                        31538 non-null  float64
 16  longitude                                       31538 non-null  float64
 17  property_type                                   31538 non-null  object
 18  room_type                                       31538 non-null  object
 19  accommodates                                    31538 non-null  int64
 20  beds                                            31538 non-null  float64
 21  amenities                                       31538 non-null  object
 22  price                                           31538 non-null  object
 23  minimum_nights                                  31538 non-null  int64
 24  maximum_nights                                  31538 non-null  int64
 25  has_availability                                31538 non-null  object
 26  availability_30                                 31538 non-null  int64
 27  availability_60                                 31538 non-null  int64
 28  availability_90                                 31538 non-null  int64
 29  availability_365                                31538 non-null  int64
 30  number_of_reviews                               31538 non-null  int64
 31  review_scores_rating                            31538 non-null  float64
 32  review_scores_accuracy                          31538 non-null  float64
 33  review_scores_cleanliness                       31538 non-null  float64
 34  review_scores_checkin                           31538 non-null  float64
 35  review_scores_communication                     31538 non-null  float64
 36  review_scores_location                          31538 non-null  float64
 37  review_scores_value                             31538 non-null  float64
 38  instant_bookable                                31538 non-null  object
 39  calculated host listings count                  31538 non-null  int64
 40  calculated_host_listings_count_entire_homes   31538 non-null  int64
 41  calculated_host_listings_count_private_rooms  31538 non-null  int64
 42  calculated_host_listings_count_shared_rooms   31538 non-null  int64
dtypes: float64(13), int64(13), object(17)
memory usage: 11.6+ MB
```

In [ ]:

```
# Check data type of price column
df_listings.price.dtype
```

Out[ ]:

```
dtype('O')
```

In [15]:

```
# Change price column type into float
df_listings['price'] = df_listings['price'].str.replace('$','')
df_listings['price'] = df_listings['price'].str.replace(',','').astype('float64')
```

In [16]:

```
# Drop null values in column abou_host
df_listings.dropna(subset=['host_about'], inplace=True)
```

In [ ]:

```
# Preview sample values
df_listings['host_about']
```

Out[ ]:

```
0
A New Yorker since 2000! My passion is creating beautiful, unique spaces where unforgettable memories are made.
It's my pleasure to host people from around the world and meet new faces. Welcome travelers! \r\n\r\nI am a Sou
nd Therapy Practitioner and Kundalini Yoga & Meditation teacher. I work with energy and sound for relaxation an
d healing, using Symphonic gong, singing bowls, tuning forks, drums, voice and other instruments.
3
I used to work for a financial industry but now I work at a Japanese food market as an assistant manager.
5        Hello, \r\nI will be welcoming and helpful,  while respecting your privacy.  I know a lot about NY & B
rooklyn and love my neighborhood.  I'm especially interested in arts and music. \r\nI speak and understand seve
ral languages.  I work at home a lot,  on my main floor, and do prefer guests who are busy themselves, and casu
al,  low-key, trusting and flexible people.  \r\n It's an old house with quirks, (not a hotel!)  in a fantastic
and quiet location.\r\nIncluded: Laundry,  excellent coffee & breakfast foods, nice linens, big garden & BBQ,
fans, air conditioners.  \r\nSome use of kitchen can be worked out.\r\n
8
Capturing the Steinbeck side of life in its Fillini moment.\r\nHome is a special place, it is a live-in work of
art... A great experience I hope all to enjoy...
9
I have lived in the same apartment in Brooklyn for more than 10 years and I love it. I also love to travel, and
have been to Brazil, Peru, Costa Rica, Mexico, Germany, Italy, France as well as all over the US and Canada. I
```

```
am in my early 40s, curious, responsible, and organized.\r\n\r\nFalo muito bem português. Mon français est comm
e ci comme ça. Mi español es también más o menos.

...
37474
Hi - my name is Henry, i'm born in Europe, easy to live with and looking forward to meeting you. Don't hesitate
if you have any question about my place or the city!
37579
I work as a freelance photographer and run an arts non-profit, Slideluck.  I am busy, but social, respectful, c
lean, often out at night, cook frequently and travel a lot.
37676
I’m a traveler and entrepreneur!\nWith a love for sports and crypto currency. \n\nI love hosting and meeting di
fferent people and connecting with my guest.\n\nShoot me a message with what you’re thinking about at one of my
properties and we can make something work!\n\nWe own a concierge company for nightlife and restaurants and exot
ic cars. We are your one stop shop for everything nyc! Lived here for 25 years
37854
We are delighted to accommodate you during your stay. We are passionate about providing the finest possible ser
vice, and we are providing accommodations within a very residential setting - whether for vacation, business or
extended stay.\n
37873
Welcoming travellers to my home in New York. I love this city and everything it has to offer. Sharing my passio
n for home decor, balancing beauty and functionality. It’s all about the NYC experience:)\n
Name: host_about, Length: 31538, dtype: object
```

In [17]:

```python
# Drop rows that contains word 'hidden'
df_listings =  df_listings[df_listings["host_about"].str.contains("hidden")==False]
```

In [ ]:

```python
df_listings['host_about']
```

Out[ ]:

```
0
A New Yorker since 2000! My passion is creating beautiful, unique spaces where unforgettable memories are made.
It's my pleasure to host people from around the world and meet new faces. Welcome travelers! \r\n\r\nI am a Sou
nd Therapy Practitioner and Kundalini Yoga & Meditation teacher. I work with energy and sound for relaxation an
d healing, using Symphonic gong, singing bowls, tuning forks, drums, voice and other instruments.
3
I used to work for a financial industry but now I work at a Japanese food market as an assistant manager.
5         Hello, \r\nI will be welcoming and helpful,  while respecting your privacy.  I know a lot about NY & B
rooklyn and love my neighborhood.  I'm especially interested in arts and music. \r\nI speak and understand seve
ral languages.  I work at home a lot,  on my main floor, and do prefer guests who are busy themselves, and casu
al,  low-key, trusting and flexible people.  \r\n It's an old house with quirks, (not a hotel!)  in a fantastic
and quiet location.\r\nIncluded: Laundry,  excellent coffee & breakfast foods, nice linens, big garden & BBQ,
fans, air conditioners.  \r\nSome use of kitchen can be worked out.\r\n
8
Capturing the Steinbeck side of life in its Fillini moment.\r\nHome is a special place, it is a live in work of
Capturing the Steinbeck side of life in its Fillini moment.\r\nHome is a special place, it is a live-in work of
art... A great experience I hope all to enjoy...
9
I have lived in the same apartment in Brooklyn for more than 10 years and I love it. I also love to travel, and
have been to Brazil, Peru, Costa Rica, Mexico, Germany, Italy, France as well as all over the US and Canada. I
am in my early 40s, curious, responsible, and organized.\r\n\r\nFalo muito bem português. Mon français est comm
e ci comme ça. Mi español es también más o menos.

...
37474
Hi - my name is Henry, i'm born in Europe, easy to live with and looking forward to meeting you. Don't hesitate
if you have any question about my place or the city!
37579
I work as a freelance photographer and run an arts non-profit, Slideluck.  I am busy, but social, respectful, c
lean, often out at night, cook frequently and travel a lot.
37676
I’m a traveler and entrepreneur!\nWith a love for sports and crypto currency. \n\nI love hosting and meeting di
fferent people and connecting with my guest.\n\nShoot me a message with what you’re thinking about at one of my
properties and we can make something work!\n\nWe own a concierge company for nightlife and restaurants and exot
ic cars. We are your one stop shop for everything nyc! Lived here for 25 years
37854
We are delighted to accommodate you during your stay. We are passionate about providing the finest possible ser
vice, and we are providing accommodations within a very residential setting - whether for vacation, business or
extended stay.\n
37873
Welcoming travellers to my home in New York. I love this city and everything it has to offer. Sharing my passio
n for home decor, balancing beauty and functionality. It’s all about the NYC experience:)\n
Name: host_about, Length: 30649, dtype: object
```

In [ ]:

```python
# Create a function to preprocess text column
def clean_text(text):
    '''
    input- 'text' to be preprocessed
    output- converts input 'text' to lowercase,remove square brackets,links,punctuation
    and words containing numbers. Removes common accent characters and returns clean text.
    '''
    text= re.sub('[0-9\n]',' ',text)
    text = text.lower()
    text = re.sub('',"", text)
    text = re.sub(r'[^a-zA-Z0-9]', ' ', text)
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('\w*\d\w*', ' ', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    #text = re.sub('[^a-zA-Z]', '', str(text))
    return text
```

In [ ]:

```
# Apply fucntion into text column host_about
df_listings['host_about'] = df_listings['host_about'].apply(lambda x: clean_text(x))
df_listings['host_about']
```

Out[ ]:

```
0
a new yorker since       my passion is creating beautiful  unique spaces where unforgettable memories are made 
it s my pleasure to host people from around the world and meet new faces  welcome travelers      i am a sound t
herapy practitioner and kundalini yoga   meditation teacher  i work with energy and sound for relaxation and he
aling  using symphonic gong  singing bowls  tuning forks  drums  voice and other instruments
3
i used to work for a financial industry but now i work at a japanese food market as an assistant manager
5        hello    i will be welcoming and helpful   while respecting your privacy   i know a lot about ny   bro
oklyn and love my neighborhood   i m especially interested in arts and music    i speak and understand several 
languages   i work at home a lot   on my main floor   and do prefer guests who are busy themselves  and casual 
low key  trusting and flexible people      it s an old house with quirks   not a hotel    in a fantastic and qu
iet location   included  laundry   excellent coffee   breakfast foods  nice linens  big garden   bbq   fans  ai
r conditioners     some use of kitchen can be worked out
8
capturing the steinbeck side of life in its fillini moment   home is a special place  it is a live in work of a
rt    a great experience i hope all to enjoy
9
i have lived in the same apartment in brooklyn for more than    years and i love it  i also love to travel  and
have been to brazil  peru  costa rica  mexico  germany  italy  france as well as all over the us and canada  i
am in my early   s  curious responsible  and organized     falo muito bem portugu s  mon fran ais est comme ci
comme  a  mi espa ol es tambi n m s o menos

...
37474
hi   my name is henry  i m born in europe  easy to live with and looking forward to meeting you  don t hesitate
if you have any question about my place or the city
37579
i work as a freelance photographer and run an arts non profit  slideluck   i am busy  but social  respectful  c
lean  often out at night  cook frequently and travel a lot
37676
i m a traveler and entrepreneur  with a love for sports and crypto currency    i love hosting and meeting diffe
rent people and connecting with my guest   shoot me a message with what you re thinking about at one of my prop
erties and we can make something work   we own a concierge company for nightlife and restaurants and exotic car
s  we are your one stop shop for everything nyc  lived here for    years
37854
we are delighted to accommodate you during your stay  we are passionate about providing the finest possible ser
vice  and we are providing accommodations within a very residential setting   whether for vacation  business or
extended stay
37873
```

```
welcoming travellers to my home in new york  i love this city and everything it has to offer  sharing my passio
n for home decor  balancing beauty and functionality  it s all about the nyc experience
Name: host_about, Length: 30649, dtype: object
```

In [ ]:

```
# Create runction to remove single characters within the text
def single_char(text):
  text = re.sub('(\\b[A-Za-z] \\b|\\b [A-Za-z]\\b)', '',text)
  return text;
```

In [ ]:

```
# Apply function to remove any single characters in the text
df_listings['host_about'] = df_listings['host_about'].apply(lambda x: single_char(x))
df_listings['host_about']
```

Out[ ]:

```
0
new yorker since       my passion is creating beautiful  unique spaces where unforgettable memories are made  i
t my pleasure to host people from around the world and meet new faces  welcome travelers      am sound therapy
practitioner and kundalini yoga   meditation teacher  work with energy and sound for relaxation and healing  us
ing symphonic gong  singing bowls  tuning forks  drums  voice and other instruments
3
used to work for financial industry but now work at japanese food market as an assistant manager
5        hello    will be welcoming and helpful   while respecting your privacy   know lot about ny   brooklyn
and love my neighborhood   especially interested in arts and music    speak and understand several languages
work at home lot   on my main floor  and do prefer guests who are busy themselves  and casual   low key  trusti
ng and flexible people      it an old house with quirks   not hotel    in fantastic and quiet location   includ
ed  laundry  excellent coffee  breakfast foods  nice linens  big garden   bbq   fans  air conditioners     so
me use of kitchen can be worked out
8
capturing the steinbeck side of life in its fillini moment   home is special place  it is live in work of art
great experience hope all to enjoy
9
have lived in the same apartment in brooklyn for more than    years and love it  also love to travel  and have
been to brazil  peru  costa rica  mexico  germany  italy  france as well as all over the us and canada  am in m
y early    s  curious  responsible  and organized     falo muito bem portugu  mon fran ais est comme ci comme  a
mi espa ol es tambi menos

...
37474
hi   my name is henry  born in europe  easy to live with and looking forward to meeting you  don hesitate if yo
u have any question about my place or the city
37579
work as freelance photographer and run an arts non profit  slideluck   am busy  but social  respectful  clean
often out at night  cook frequently and travel lot
37676
```

```
traveler and entrepreneur  with love for sports and crypto currency    love hosting and meeting different peopl
e and connecting with my guest   shoot me message with what you re thinking about at one of my properties and w
e can make something work   we own concierge company for nightlife and restaurants and exotic cars  we are your
one stop shop for everything nyc  lived here for    years
37854
we are delighted to accommodate you during your stay  we are passionate about providing the finest possible ser
vice  and we are providing accommodations within very residential setting   whether for vacation  business or e
xtended stay
37873
welcoming travellers to my home in new york  love this city and everything it has to offer  sharing my passion
for home decor  balancing beauty and functionality  it all about the nyc experience
Name: host_about, Length: 30649, dtype: object
```

In [ ]:

```
# Count an unique values
df_listings.host_is_superhost.value_counts()
```

Out[ ]:

```
RegularHost    18076
SuperHost      12573
Name: host_is_superhost, dtype: int64
```

In [ ]:

```
!pip install nltk
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: nltk in /usr/local/lib/python3.8/dist-packages (3.7)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from nltk) (4.64.1)
Requirement already satisfied: joblib in /usr/local/lib/python3.8/dist-packages (from nltk) (1.2.0)
Requirement already satisfied: click in /usr/local/lib/python3.8/dist-packages (from nltk) (7.1.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.8/dist-packages (from nltk) (2022.6.2)
```

In [ ]:

```
#  Import nltk related libraries
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [ ]:

```
# Set stop words
stop_words = set(stopwords.words("english"))
```

In [ ]:

```
# Create seperate DataFrame for super/regular hosts
superhost = df_listings[df_listings['host_is_superhost'].str.contains('SuperHost') == True]# superhost_about_dat
superhost = df_listings[df_listings['host_is_superhost'].str.contains('SuperHost')==True]# superhost about dat
regulhost = df_listings[df_listings['host_is_superhost'].str.contains('RegularHost')==True] # regularhost abou
```

In [ ]:

```
superhost.host_is_superhost
```

Out[ ]:

```
5         SuperHost
8         SuperHost
10        SuperHost
12        SuperHost
17        SuperHost
            ...
37287     SuperHost
37294     SuperHost
37306     SuperHost
37311     SuperHost
37454     SuperHost
Name: host_is_superhost, Length: 12573, dtype: object
```

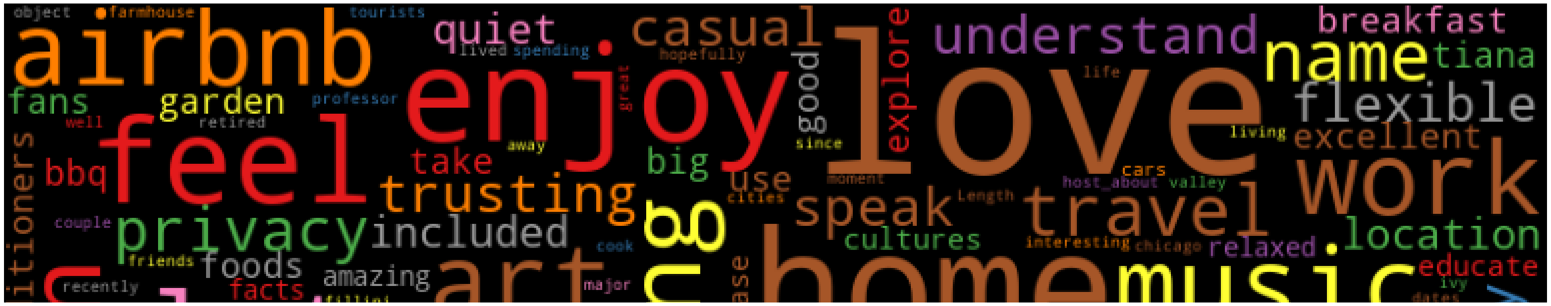
In [ ]:

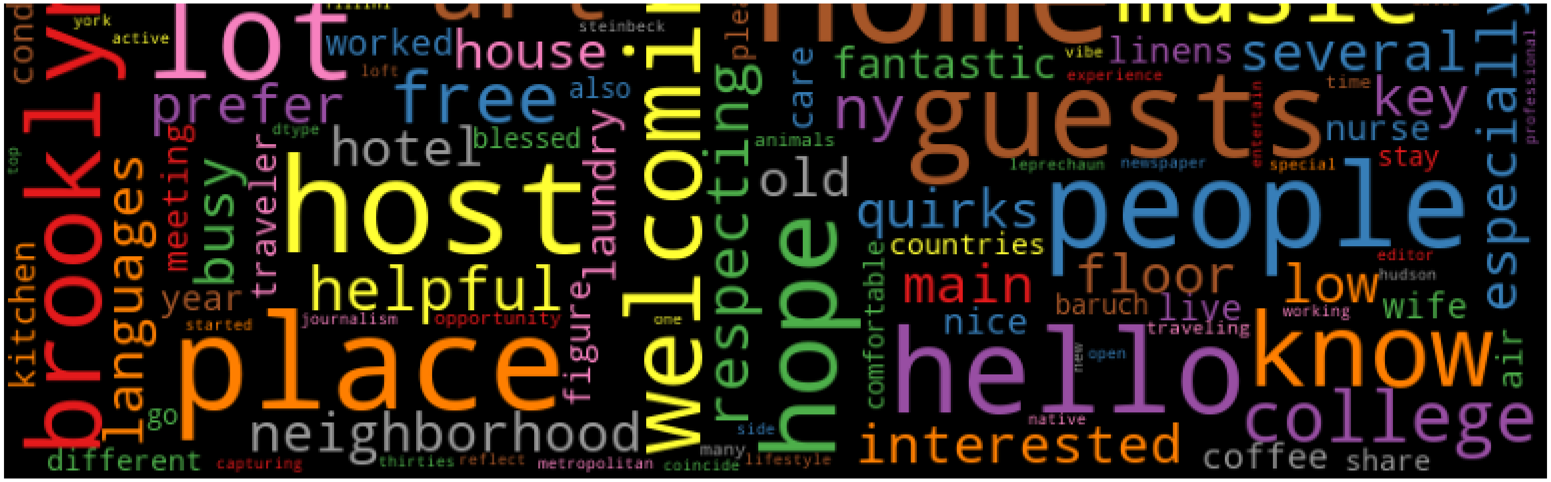
```
# Assign word cloud
wordcloud = WordCloud(background_color='black', stopwords = stop_words,max_words = 500,
                    max_font_size = 100, random_state = 42, width=800, height=400,colormap='Set1')
```

In [ ]:

```
# Plot word cloud (frequent words) about super host
wordcloud.generate(str(superhost['host_about']))
plt.figure(figsize=(12,6))
plt.imshow(wordcloud);
plt.title(f"Most Frequent Words About Super Hosts", fontdict={'size': 20,
                                                    'verticalalignment': 'bottom'})
plt.axis('off');
plt.tight_layout()
```

Most Frequent Words About Super Hosts

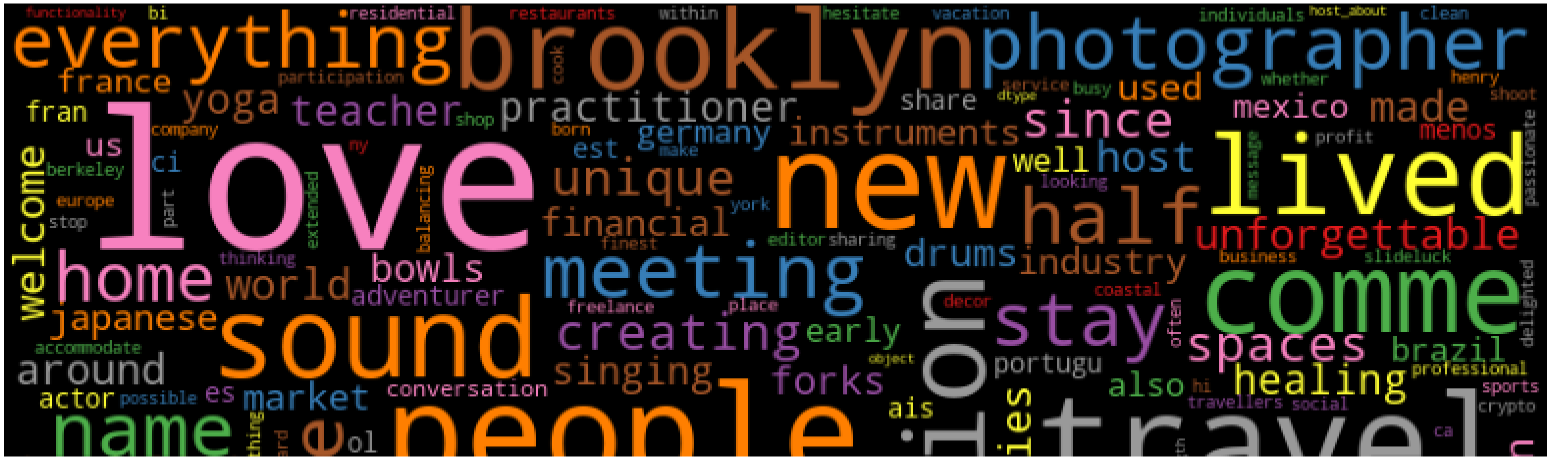


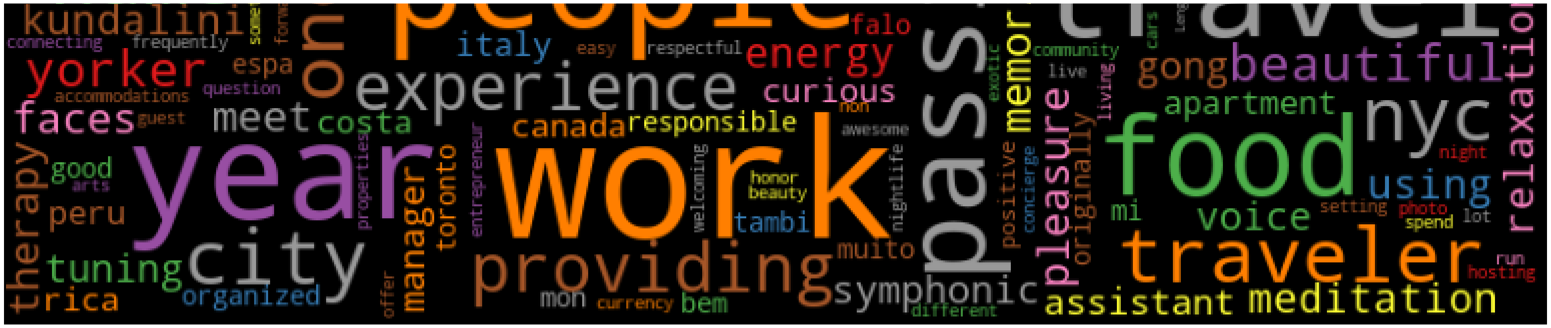


Based on the above word cloud we can say that super hosts are makie emphesases on being welcoming, helpful. *Enjoy, Love, Feel, Excellent* are also the main characteristics.

```
In [ ]: # Plot word cloud (frequent words) about regular host
        wordcloud.generate(str(regulhost['host_about']))
        plt.figure(figsize=(12,6))
        plt.imshow(wordcloud);
        plt.title(f"Most Frequent Words About Regular Hosts", fontdict={'size': 20,
                                                                        'verticalalignment': 'bottom'})
        plt.axis('off');
        plt.tight_layout()
```

Most Frequent Words About Regular Hosts





Regular hosts are expressed with the frequent words such as *Love, Work,Food, Unique and Unforgettable*

In [ ]:

```
#  Get summary of the DataFrame
df_listings.info()
```

```
Int64Index: 30649 entries, 0 to 37873
Data columns (total 43 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   id                                        30649 non-null  int64
 1   name                                      30649 non-null  object
 2   description                               30649 non-null  object
 3   neighborhood_overview                     30649 non-null  object
 4   picture_url                               30649 non-null  object
 5   host_name                                 30649 non-null  object
 6   host_about                                30649 non-null  object
 7   host_response_time                        30649 non-null  object
 8   host_response_rate                        30649 non-null  float64
 9   host_acceptance_rate                      30649 non-null  float64
 10  host_is_superhost                         30649 non-null  object
 11  host_total_listings_count                 30649 non-null  float64
 12  neighbourhood                             30649 non-null  object
 13  neighbourhood_cleansed                    30649 non-null  object
 14  neighbourhood_group_cleansed              30649 non-null  object
 15  latitude                                  30649 non-null  float64
 16  longitude                                 30649 non-null  float64
 17  property_type                             30649 non-null  object
 18  room_type                                 30649 non-null  object
 19  accommodates                              30649 non-null  int64
 20  beds                                      30649 non-null  float64
 21  amenities                                 30649 non-null  object
 22  price                                     30649 non-null  float64
 23  minimum_nights                                30649 non-null  int64
 24  maximum_nights                                30649 non-null  int64
 25  has_availability                              30649 non-null  object
 26  availability_30                               30649 non-null  int64
 27  availability_60                               30649 non-null  int64
 28  availability_90                               30649 non-null  int64
 29  availability_365                              30649 non-null  int64
 30  number_of_reviews                             30649 non-null  int64
 31  review_scores_rating                          30649 non-null  float64
 32  review_scores_accuracy                        30649 non-null  float64
 33  review_scores_cleanliness                     30649 non-null  float64
 34  review_scores_checkin                         30649 non-null  float64
 35  review_scores_communication                   30649 non-null  float64
 36  review_scores_location                        30649 non-null  float64
 37  review_scores_value                           30649 non-null  float64
 38  instant_bookable                              30649 non-null  object
 39  calculated_host_listings_count                30649 non-null  int64
 40  calculated_host_listings_count_entire_homes   30649 non-null  int64
 41  calculated_host_listings_count_private_rooms  30649 non-null  int64
 42  calculated_host_listings_count_shared_rooms   30649 non-null  int64
dtypes: float64(14), int64(13), object(16)
memory usage: 10.3+ MB
```

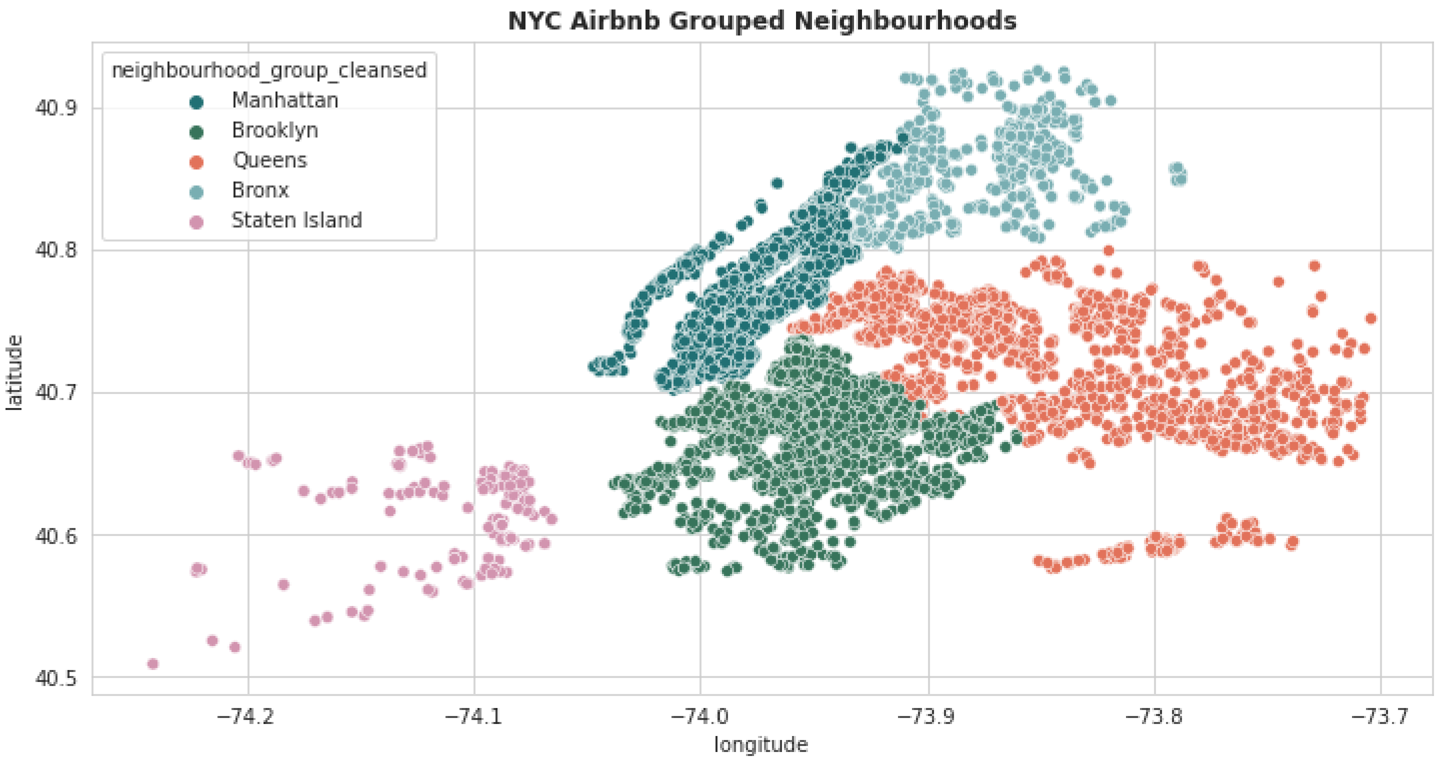
Summary of Host Type EDA

- Majority of super hosts are from the Brooklyn while *Queens, Bronx* and *Staten Island* have nearly an equal amount of host types (super/regular)
- The hosts that have responded within a few days or more have been received lower ratings up to 0.45%. from the plot we can see that if hosts can respond within a few hours up to maximum within a da there is higher chance to get better ratings. The majority of the super hosts also fall in this gap which proofs their responsibility.
- Based on the above word cloud we can say that super hosts are makie emphesases on being welcoming, helpful. *Enjoy, Love, Feel, Excellent* are also the main characteristics.
- Regular hosts are expressed with the frequent words such as *Love, Work,Food, Unique and Unforgettable*

EDA Neighbourhoods

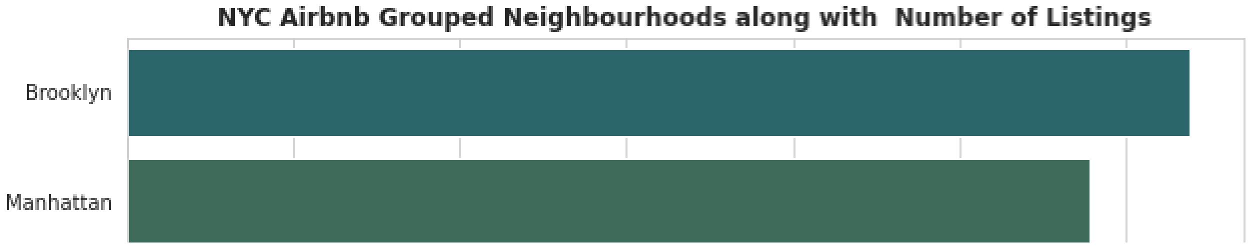
In [18]:

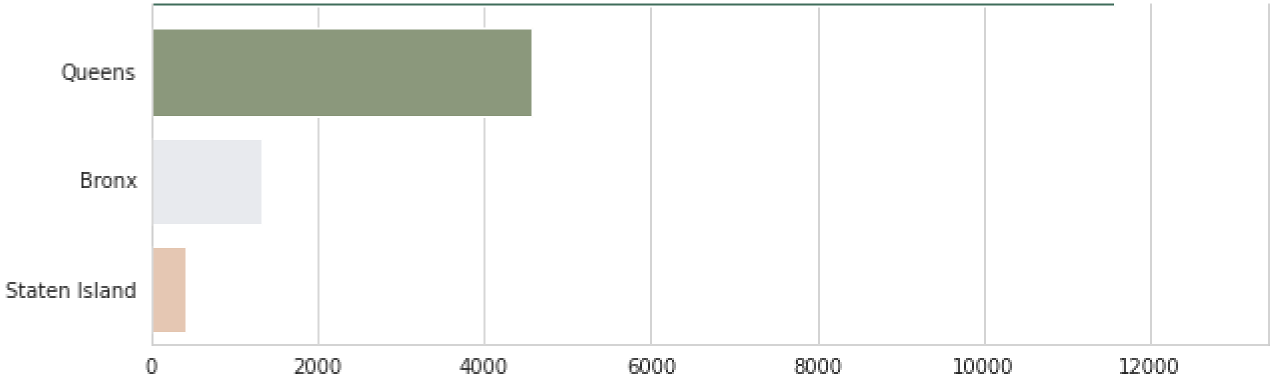
```
# POT distribution map of listings based on NYC neighbourhoods
plt.figure(figsize=(12,6))
sns.scatterplot(df_listings.longitude,df_listings.latitude,hue=df_listings.neighbourhood_group_cleansed,
                palette=['#217074','#37745B','#E2725A',"#79AEB2", '#D294AF'])
plt.ioff()
plt.title('NYC Airbnb Grouped Neighbourhoods',fontweight="bold")
plt.show();
```



In [ ]:

```
# Which grouped neighborhood has the highest number of listings?
plt.figure(figsize=(10,5))
sns.barplot(y = df_listings['neighbourhood_group_cleansed'].value_counts().sort_values(ascending=False).keys()
        x = df_listings['neighbourhood_group_cleansed'].value_counts().sort_values(ascending=False).values,
        orient='h', palette=['#217074','#37745B','#8B9D77','#E7EAEF','#EDC5AB']);
plt.title('NYC Airbnb Grouped Neighbourhoods along with  Number of Listings',fontweight="bold")
ax.set_xlabel('Number of Listings')
ax.set_ylabel('Neighborhoods Group');
```

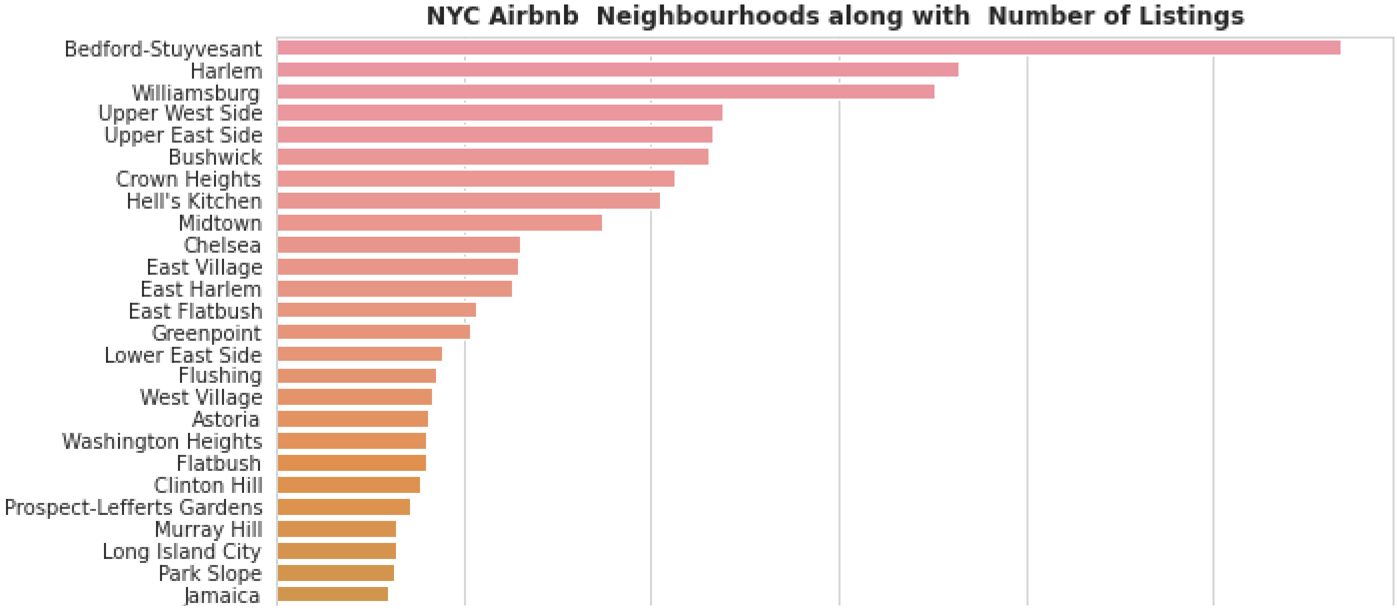


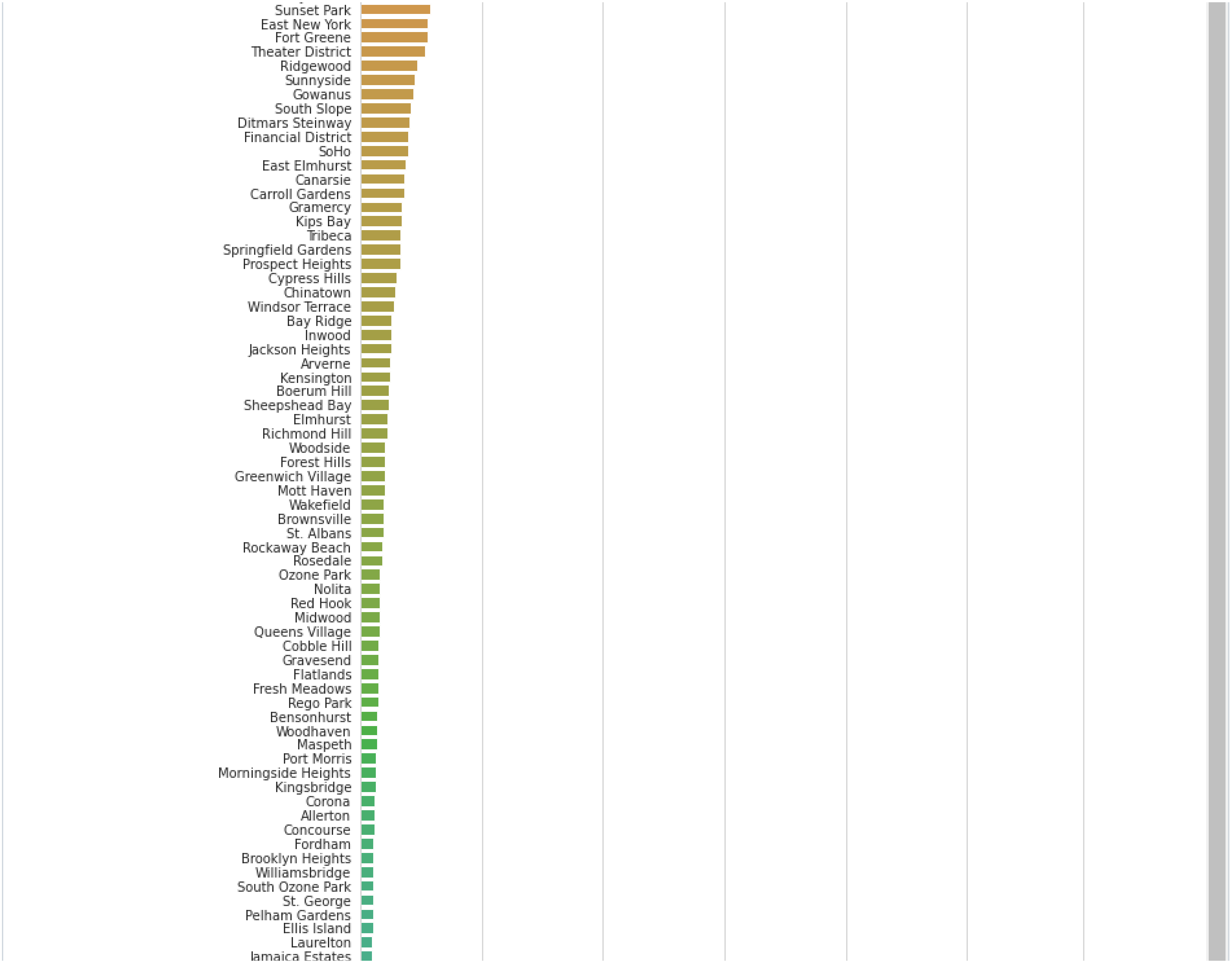


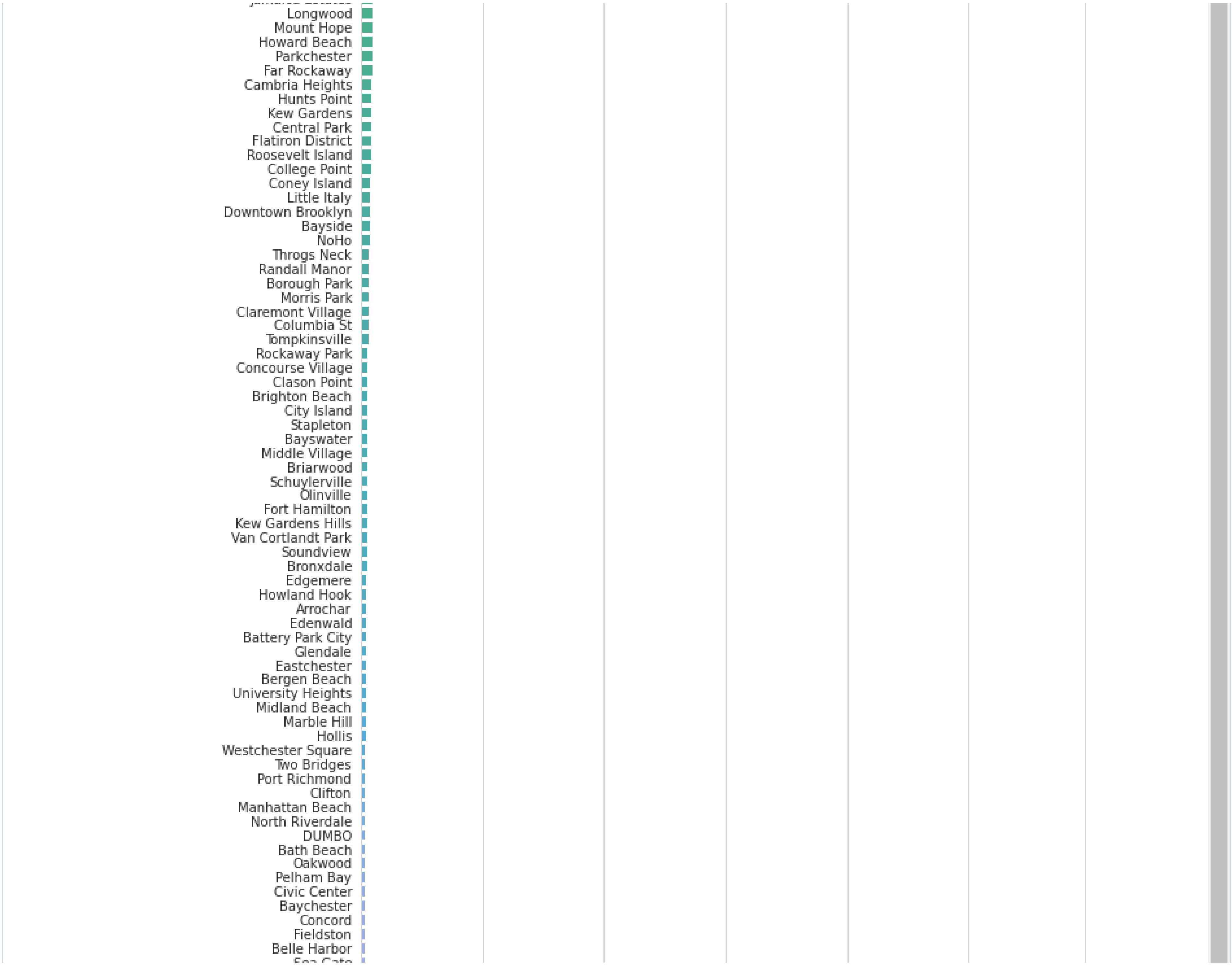
The majority of the listings (11000/14000) are located in *Brooklyn and Manhattan* while *Staten Island* is in the last place with the least amount of the listings.

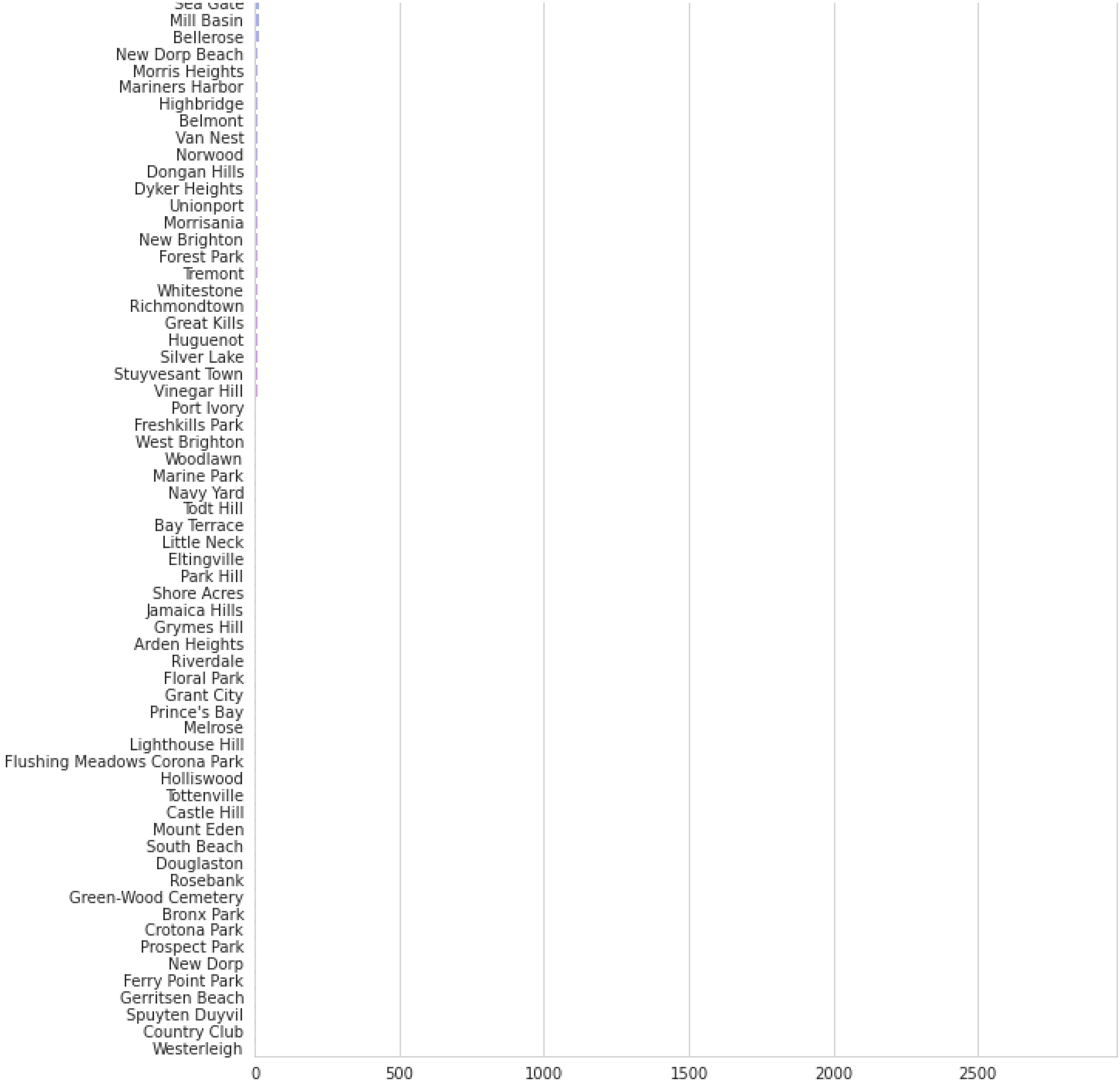
In [ ]:

```
# Which detailed neighborhood has the highest number of listings?
plt.figure(figsize=(10,45))
sns.barplot(y = df_listings['neighbourhood_cleansed'].value_counts().sort_values(ascending=False).keys(),
        x = df_listings['neighbourhood_cleansed'].value_counts().sort_values(ascending=False).values,
        orient='h');
plt.title('NYC Airbnb  Neighbourhoods along with  Number of Listings',fontweight="bold")
ax.set_xlabel('Number of Listings')
ax.set_ylabel('NYC Neighborhoods');
```




Sunset Park
East New York
Fort Greene
Theater District
Ridgewood
Sunnyside
Gowanus
South Slope
Ditmars Steinway
Financial District
SoHo
East Elmhurst
Canarsie
Carroll Gardens
Gramercy
Kips Bay
Tribeca
Springfield Gardens
Prospect Heights
Cypress Hills
Chinatown
Windsor Terrace
Bay Ridge
Inwood
Jackson Heights
Arverne
Kensington
Boerum Hill
Sheepshead Bay
Elmhurst
Richmond Hill
Woodside
Forest Hills
Greenwich Village
Mott Haven
Wakefield
Brownsville
St. Albans
Rockaway Beach
Rosedale
Ozone Park
Nolita
Red Hook
Midwood
Queens Village
Cobble Hill
Gravesend
Flatlands
Fresh Meadows
Rego Park
Bensonhurst
Woodhaven
Maspeth
Port Morris
Morningside Heights
Kingsbridge
Corona
Allerton
Concourse
Fordham
Brooklyn Heights
Williamsbridge
South Ozone Park
St. George
Pelham Gardens
Ellis Island
Laurelton
Jamaica Estates


Longwood
Mount Hope
Howard Beach
Parkchester
Far Rockaway
Cambria Heights
Hunts Point
Kew Gardens
Central Park
Flatiron District
Roosevelt Island
College Point
Coney Island
Little Italy
Downtown Brooklyn
Bayside
NoHo
Throgs Neck
Randall Manor
Borough Park
Morris Park
Claremont Village
Columbia St
Tompkinsville
Rockaway Park
Concourse Village
Clason Point
Brighton Beach
City Island
Stapleton
Bayswater
Middle Village
Briarwood
Schuylerville
Olinville
Fort Hamilton
Kew Gardens Hills
Van Cortlandt Park
Soundview
Bronxdale
Edgemere
Howland Hook
Arrochar
Edenwald
Battery Park City
Glendale
Eastchester
Bergen Beach
University Heights
Midland Beach
Marble Hill
Hollis
Westchester Square
Two Bridges
Port Richmond
Clifton
Manhattan Beach
North Riverdale
DUMBO
Bath Beach
Oakwood
Pelham Bay
Civic Center
Baychester
Concord
Fieldston
Belle Harbor



The top 10 neighbourgood which includes the most of the listings are *Bedford-Stuyvesant, Harlem, Williamsburg, Upper West*

Side, Upper East Side, Bushwick,Crown Heights, Hell's Kitchen, Midtown and Chelsea.

In [ ]:

```
df_listings = df_listings.reset_index()
```

In [ ]:

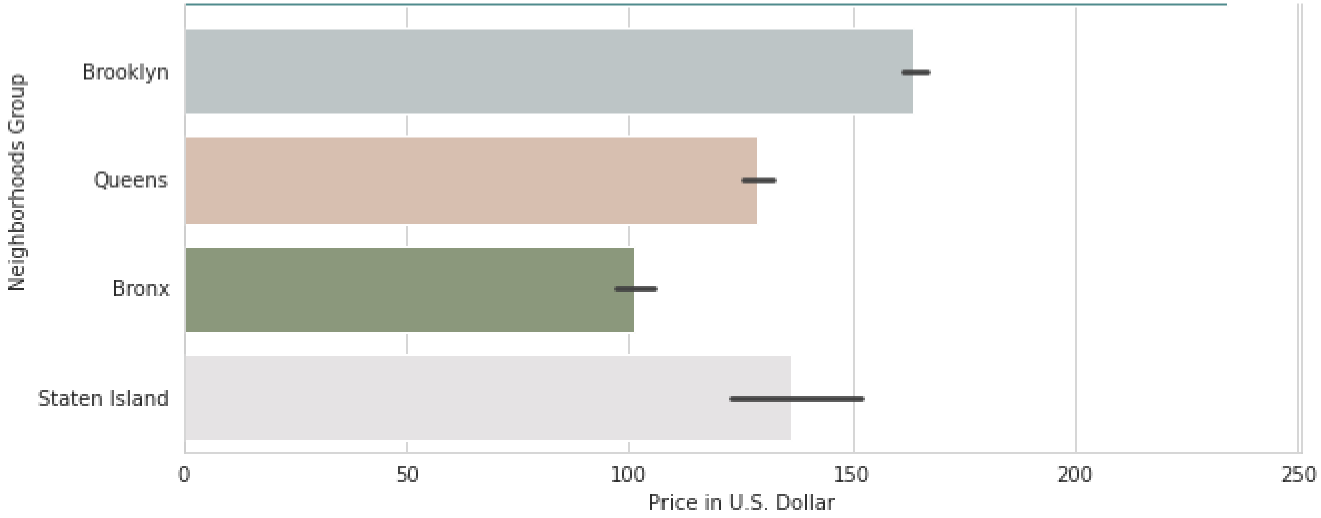
```
# Average price per neighborhood
price_per_neighb = df_listings.groupby(['neighbourhood_group_cleansed'])['price'].mean()
price_per_neighb = price_per_neighb.reset_index()
```

In [49]:

```
# PLot scatter mapbox of price in accordance with location
import plotly.express as px
fig = px.scatter_mapbox(data_frame=df_listings,
                      lat="latitude",
                      lon="longitude",
                      #color="price",
                      hover_data=["price"],
                      hover_name="neighbourhood_group_cleansed",
                      height=400,
                      width=800,
                      opacity=0.08)
                      #size="price",);

fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":1,"l":0,"b":0})
# Distribution of the prices by location
fig.show();
```

In [ ]:

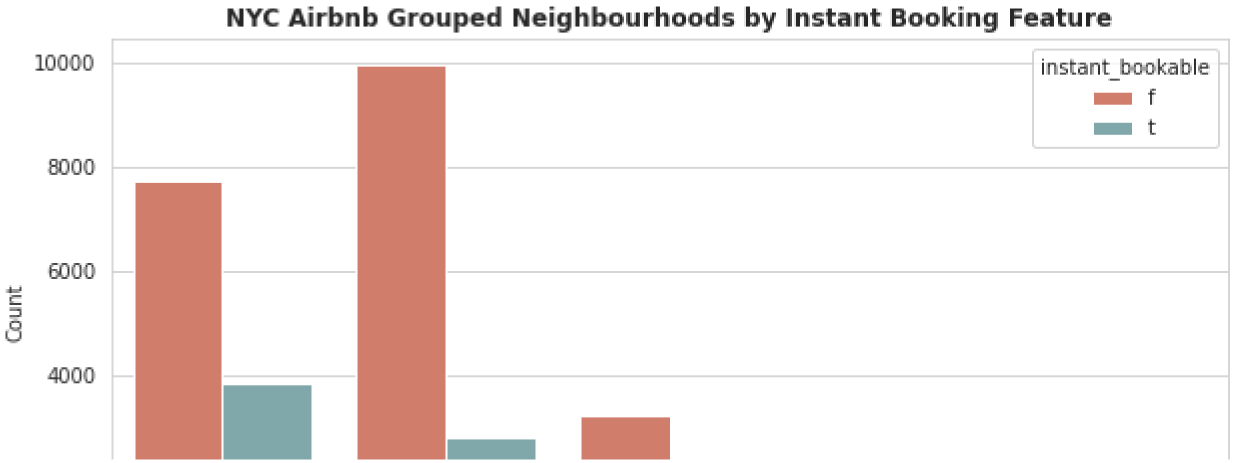
```
# PLot grouped neighborhoods with their average price
plt.figure(figsize=(10,5))
ax = sns.barplot(y = df_listings['neighbourhood_group_cleansed'], x = df_listings['price'],
                 data = price_per_neighb, orient='h', palette=['#469597','#BBC6C8','#DDBEAA','#8B9D77','#E5E3E
plt.title('NYC Airbnb  Grouped Neighbourhoods with Average Price',fontweight="bold")
ax.set_xlabel('Price in U.S. Dollar')
ax.set_ylabel('Neighborhoods Group');
```

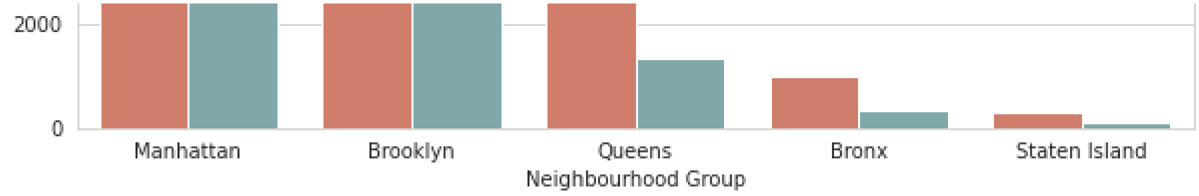


Even though *Brooklyn* includes the more listings *Manhattan* listing prices are more higher. *Staten Island* also showing more expensive listings despite the less amount of listings compare to other neighbourhoods. Average price starts from $100 and above

In [ ]:

```
# Plot grouped neighbourhoods by instant booking type
ax = sns.countplot(df_listings['neighbourhood_group_cleansed'], hue=df_listings.instant_bookable, palette=['#E
fig = plt.gcf()
fig.set_size_inches(10,5)
plt.title('NYC Airbnb Grouped Neighbourhoods by Instant Booking Feature',fontweight="bold")
ax.set_xlabel('Neighbourhood Group')
ax.set_ylabel('Count');
```





Summary of Neighbourhoods EDA

- The majority of the listings (11000/14000) are located in *Brooklyn* and *Manhattan* while *Staten Island* is in the last place with the least amount of the listings.
- The top 10 neighbourgood which includes the most of the listings are: *Bedford-Stuyvesant, Harlem, Williamsburg, Upper West Side, Upper East Side, Bushwick,Crown Heights, Hell's Kitchen, Midtown and Chelsea*.
- Even though Brooklyn includes the more listings Manhattan listing prices are more higher. Staten Island also showing more expensive listings despite the less amount of listings compare to other neighbourhoods. Average price starts from $100 and above

EDA Property Types

In [ ]:

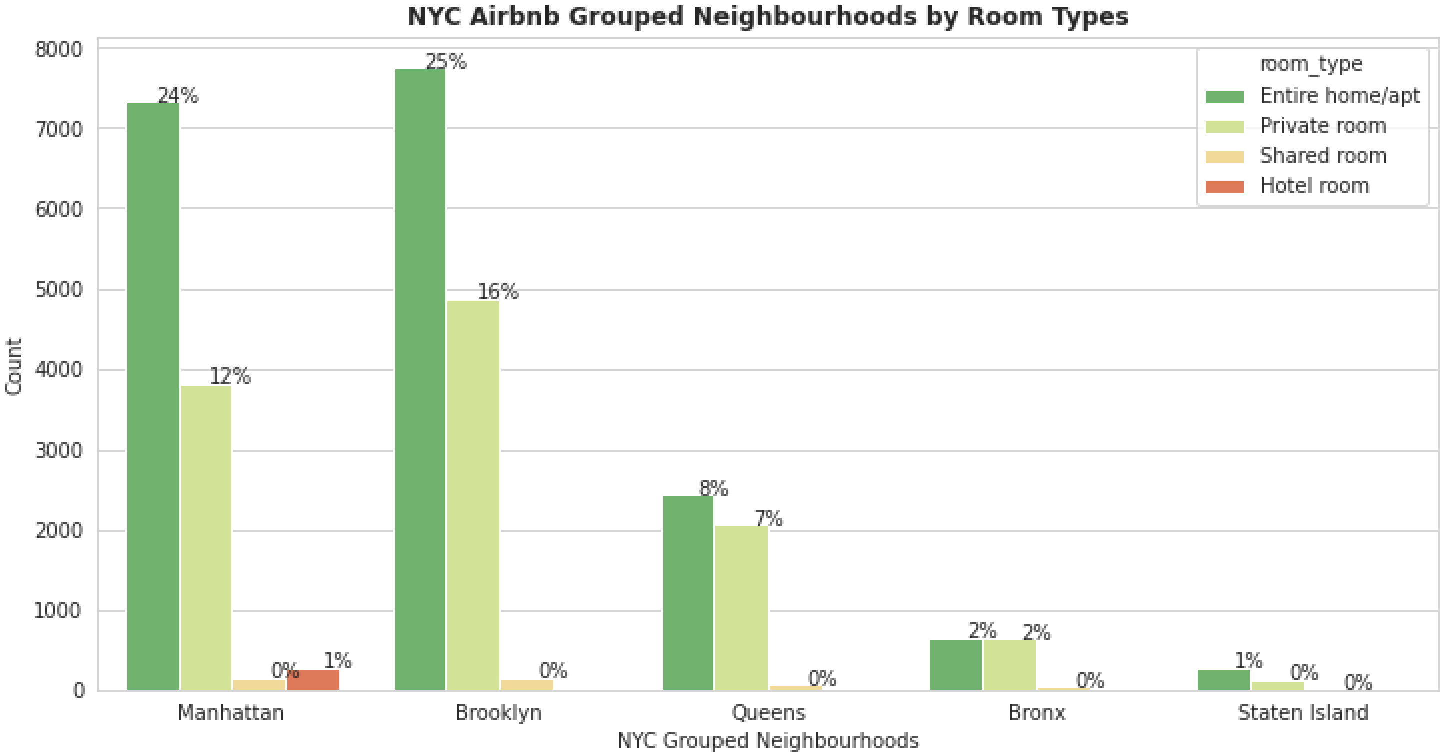
```
# Check for an unique values
df_listings.room_type.unique()
```

Out[ ]:

```
array(['Entire home/apt', 'Private room', 'Shared room', 'Hotel room'],
      dtype=object)
```

In [ ]:

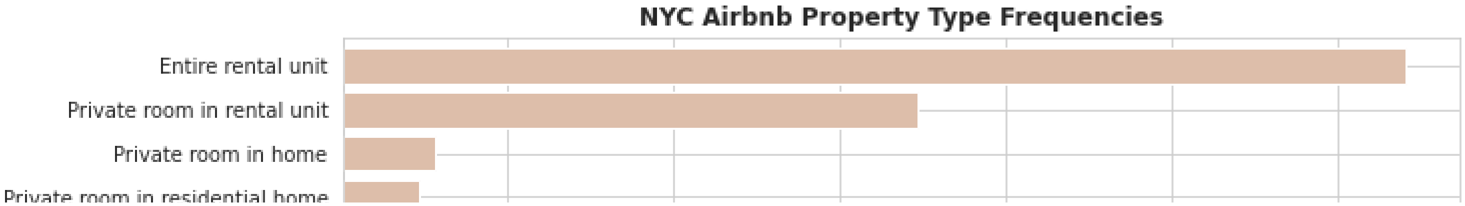
```
# Plot distribution of room types by NYC grouped neighbourhoods
ax = sns.countplot(df_listings['neighbourhood_group_cleansed'], hue=df_listings.room_type, palette='RdYlGn_r')
fig = plt.gcf()
fig.set_size_inches(12,6)
plt.title('NYC Airbnb Grouped Neighbourhoods by Room Types',fontweight="bold")
ax.set_xlabel('NYC Grouped Neighbourhoods')
ax.set_ylabel('Count');
# Display the percentage values on top the each bar
total = float(len(df_listings))
for p in ax.patches:
    percentage = '{:.0f}%'.format(100 * p.get_height()/total)
    x = p.get_x() + p.get_width()
    y = p.get_height()
    ax.annotate(percentage, (x, y),ha='center')
plt.show();
plt.show();
```

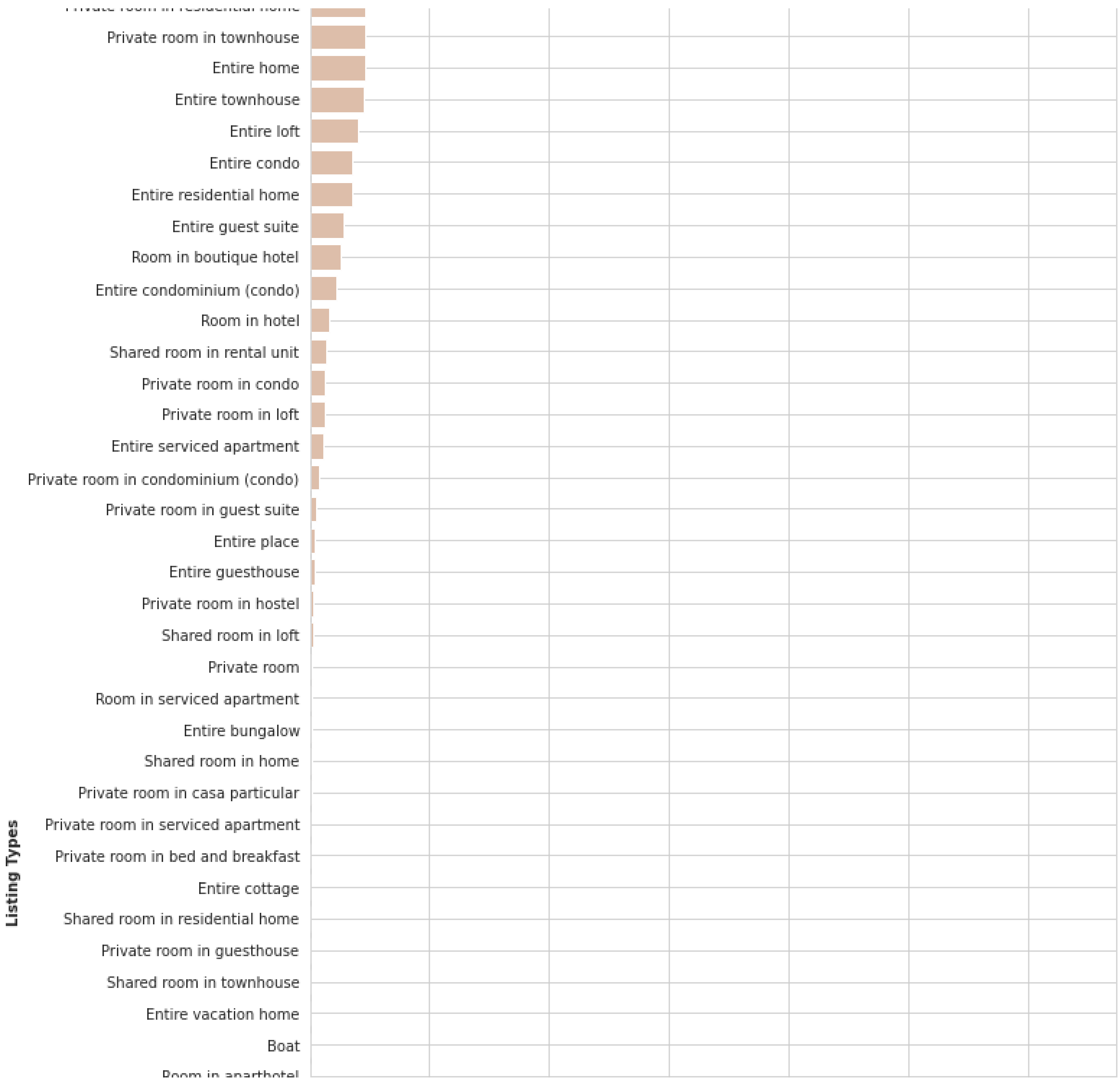


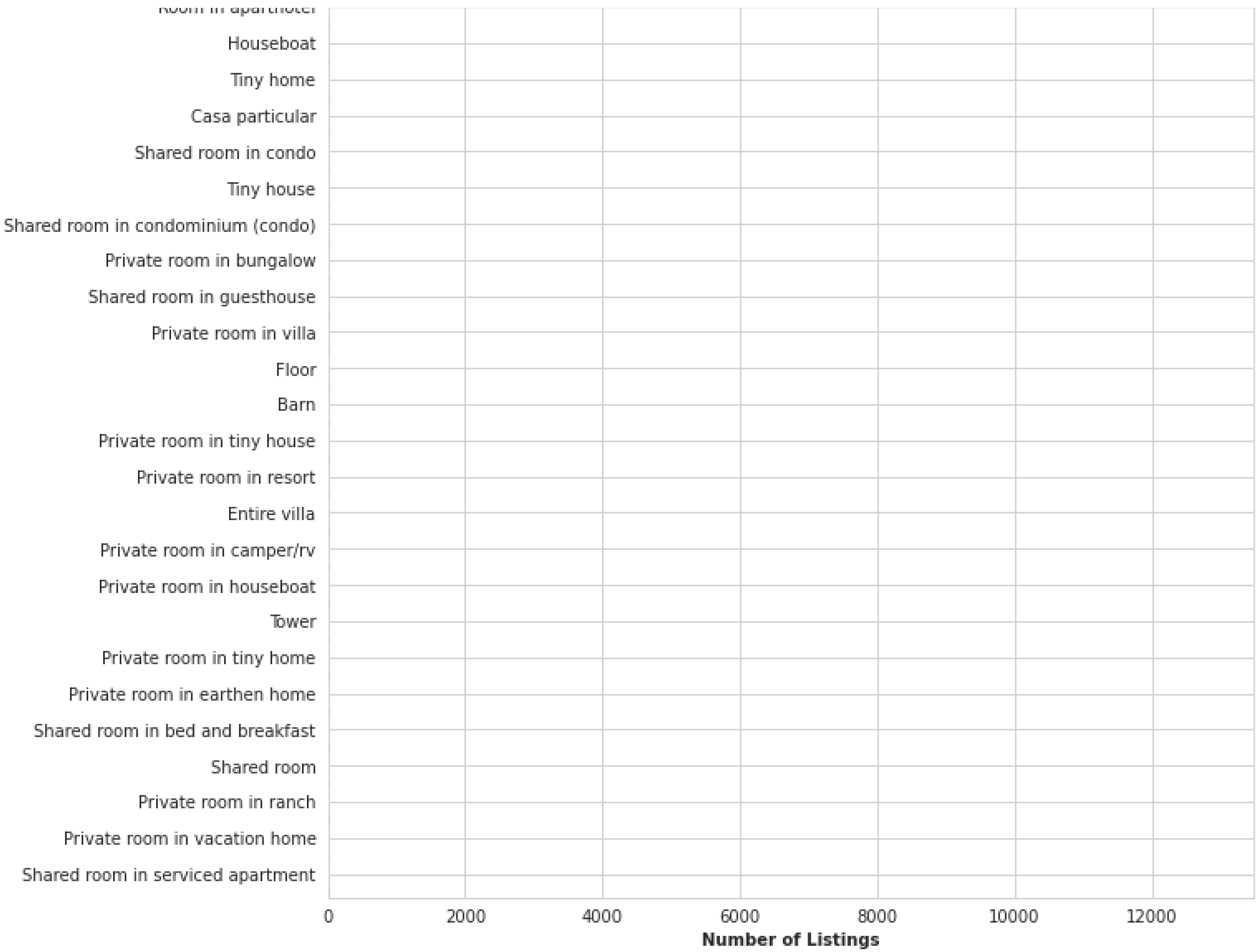
Based on the above analysis we can say that people can find *Entire home/apartment and Private rooms* almost in all NYC major 5 neighbourhoods while only Manhattan includes listings with *Hotel room* type.

In [ ]:

```
# Count and plot property types
freq_ptype = df_listings['property_type'].value_counts().sort_values(ascending=True)
freq_ptype.plot.barh(figsize =(10,25),width=0.8, color='#DDBEAA')
plt.title('NYC Airbnb Property Type Frequencies',fontweight="bold")
plt.xlabel('Number of Listings', fontweight="bold")
plt.ylabel('Listing Types', fontweight="bold");
plt.show();
```

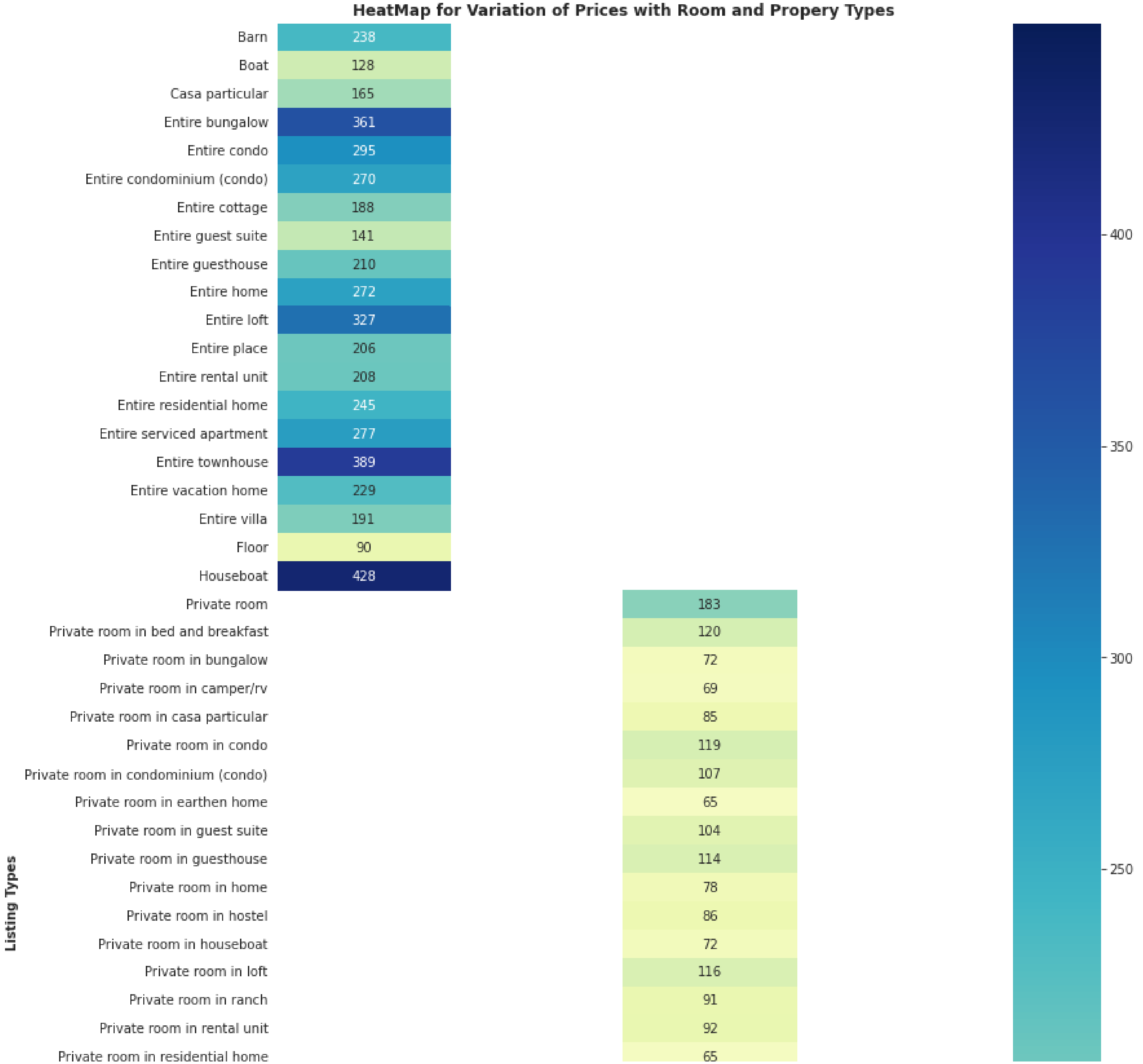


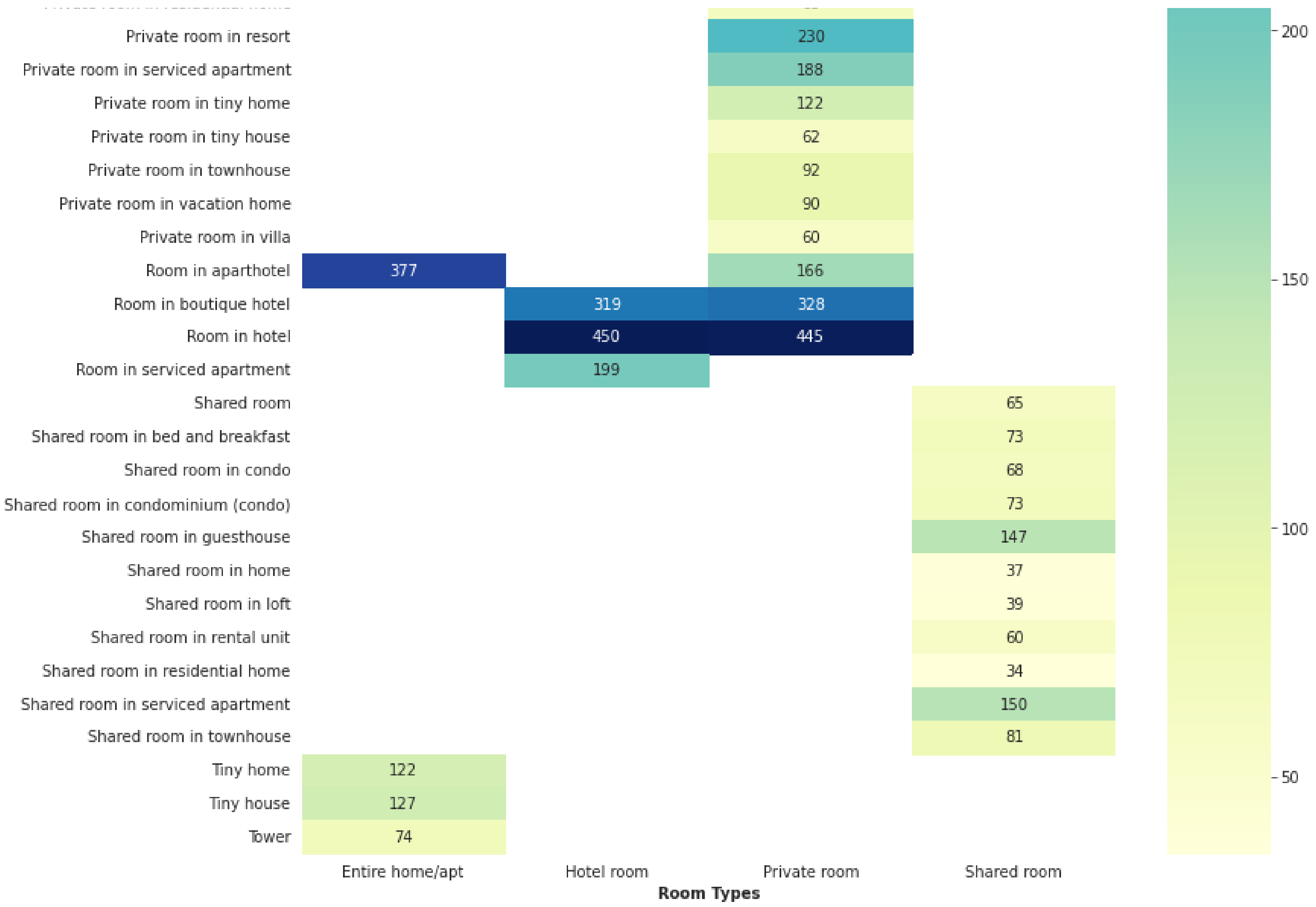

Private room in residential home
Private room in townhouse
Entire home
Entire townhouse
Entire loft
Entire condo
Entire residential home
Entire guest suite
Room in boutique hotel
Entire condominium (condo)
Room in hotel
Shared room in rental unit
Private room in condo
Private room in loft
Entire serviced apartment
Private room in condominium (condo)
Private room in guest suite
Entire place
Entire guesthouse
Private room in hostel
Shared room in loft
Private room
Room in serviced apartment
Entire bungalow
Shared room in home
Private room in casa particular
Private room in serviced apartment
Private room in bed and breakfast
Entire cottage
Shared room in residential home
Private room in guesthouse
Shared room in townhouse
Entire vacation home
Boat
Room in aparthotel
Listing Types



In [ ]:

```
# HeatMap for variation of prices with room and propery types
plt.figure(figsize=(12,25))
sns.heatmap(df_listings.groupby([
        'property_type', 'room_type']).price.mean().unstack(),annot=True, fmt=".0f", cmap="YlGnBu")
plt.title('HeatMap for Variation of Prices with Room and Propery Types',fontweight="bold")
plt.xlabel('Room Types', fontweight="bold")
plt.ylabel('Listing Types', fontweight="bold");
```


HeatMap for Variation of Prices with Room and Propery Types
Listing Types
Barn 238
Boat 128
Casa particular 165
Entire bungalow 361
Entire condo 295
Entire condominium (condo) 270
Entire cottage 188
Entire guest suite 141
Entire guesthouse 210
Entire home 272
Entire loft 327
Entire place 206
Entire rental unit 208
Entire residential home 245
Entire serviced apartment 277
Entire townhouse 389
Entire vacation home 229
Entire villa 191
Floor 90
Houseboat 428
Private room 183
Private room in bed and breakfast 120
Private room in bungalow 72
Private room in camper/rv 69
Private room in casa particular 85
Private room in condo 119
Private room in condominium (condo) 107
Private room in earthen home 65
Private room in guest suite 104
Private room in guesthouse 114
Private room in home 78
Private room in hostel 86
Private room in houseboat 72
Private room in loft 116
Private room in ranch 91
Private room in rental unit 92
Private room in residential home 65
400
350
300
250

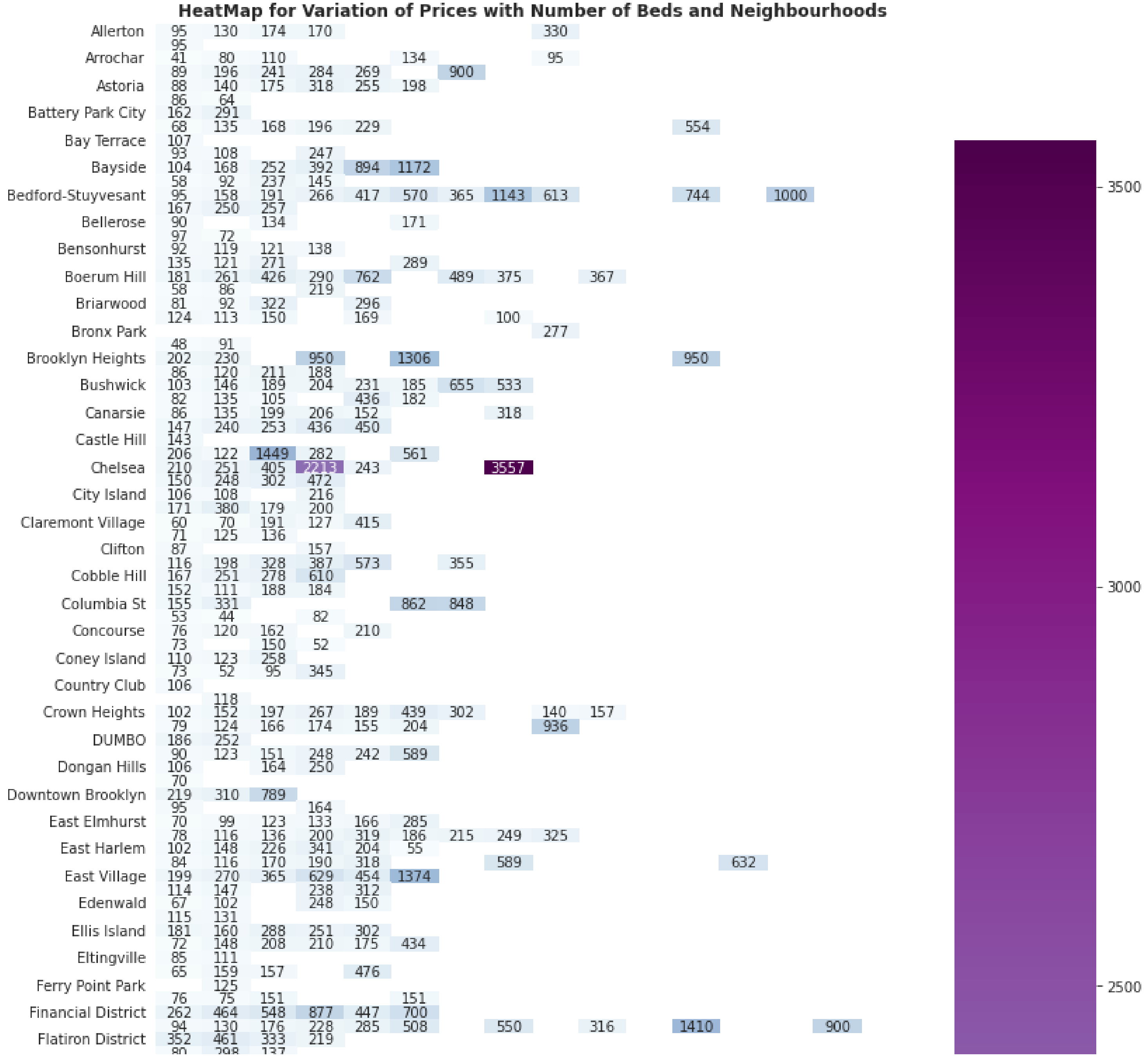


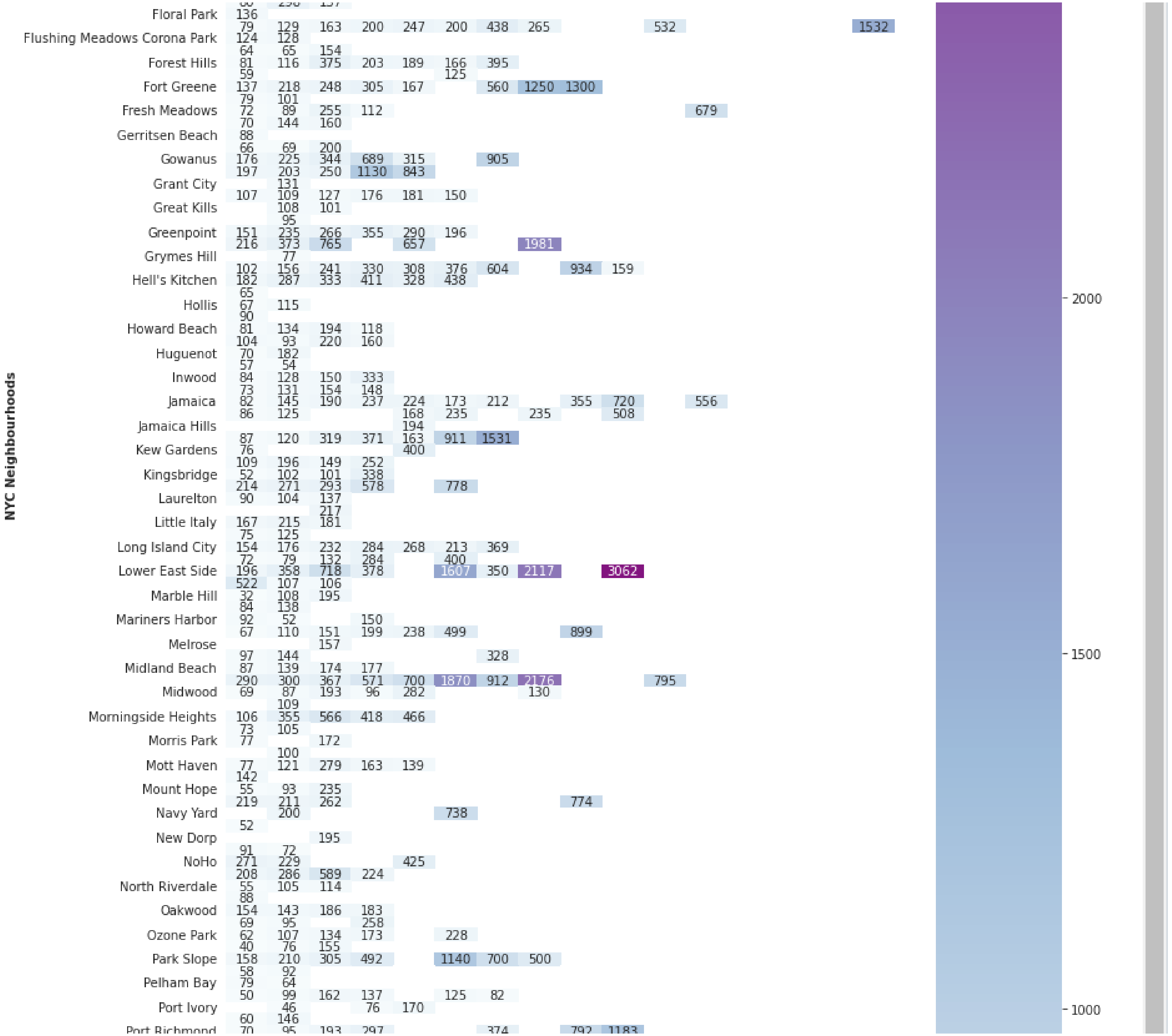
This chart allows us to see all the listings' prices broken down by property_type and roo_type in **NYC**. It can be analyzed that for almost all Property types, prices for Entire Home/Apartment is the maximum. This tells us that Property type and Room type plays a very important role in deciding price of a listing. Lets see how the number of bedrooms available affects the price of a listing

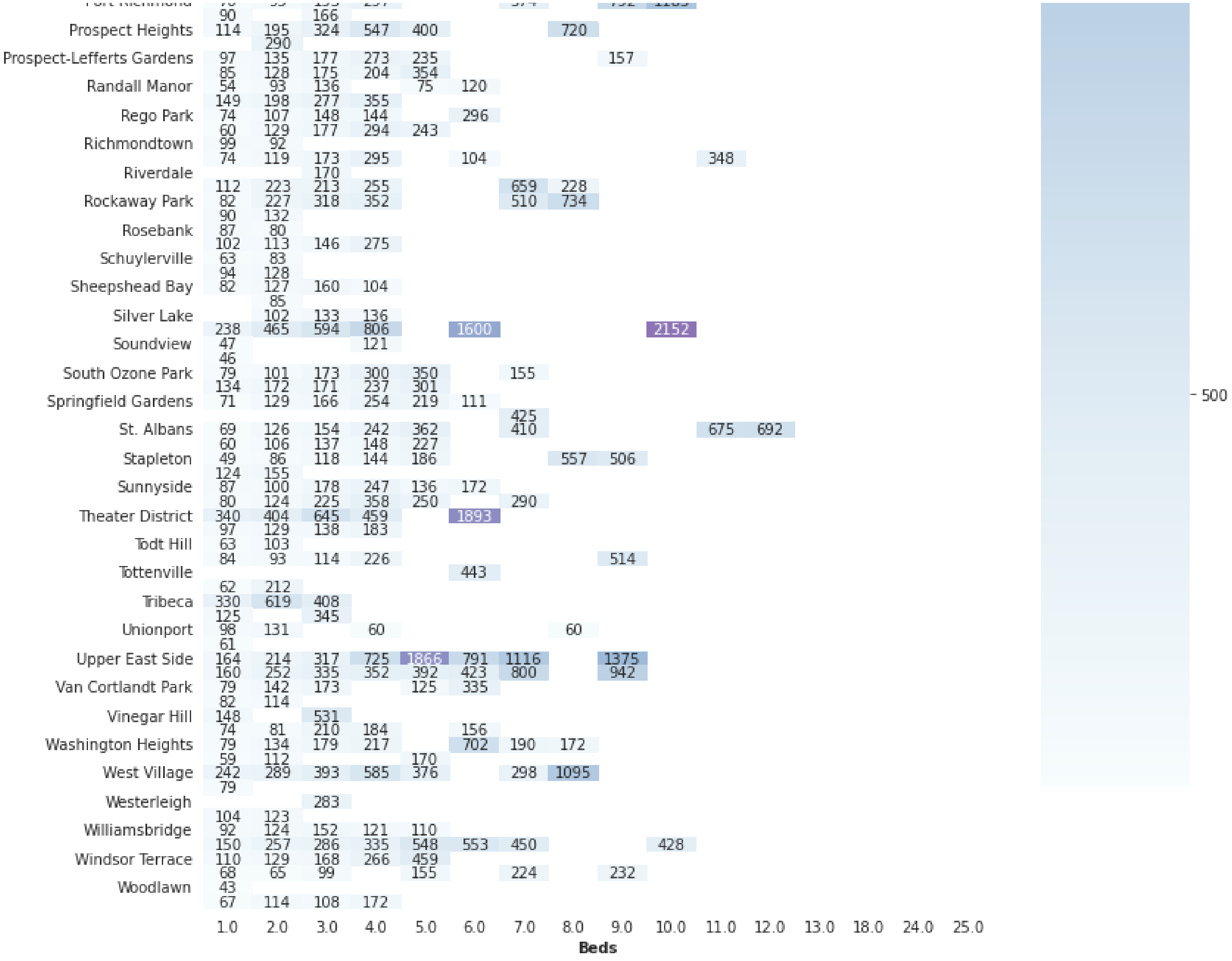
In [ ]:

```
# HeatMap for variation of prices with number of beds for listings and neighbourhoods
plt.figure(figsize=(12,40))
sns.heatmap(df_listings.groupby([
        'neighbourhood_cleansed', 'beds']).price.mean().unstack(),annot=True, fmt=".0f",cmap="BuPu")
```

```
plt.title('HeatMap for Variation of Prices with Number of Beds and Neighbourhoods',fontweight="bold")
plt.xlabel('Beds', fontweight="bold")
plt.ylabel('NYC Neighbourhoods', fontweight="bold");
```




NYC Neighbourhoods
Floral Park
136
79 129 163 200 247 200 438 265 532 1532
Flushing Meadows Corona Park
124 128
64 65 154
Forest Hills
81 116 375 203 189 166 395
59 125
Fort Greene
137 218 248 305 167 560 1250 1300
79 101
Fresh Meadows
72 89 255 112 679
70 144 160
Gerritsen Beach
88
66 69 200
Gowanus
176 225 344 689 315 905
197 203 250 1130 843
Grant City
131
107 109 127 176 181 150
Great Kills
108 101
95
Greenpoint
151 235 266 355 290 196
216 373 765 657 1981
Grymes Hill
77
102 156 241 330 308 376 604 934 159
Hell's Kitchen
182 287 333 411 328 438
65
Hollis
67 115
90
Howard Beach
81 134 194 118
104 93 220 160
Huguenot
70 182
57 54
Inwood
84 128 150 333
73 131 154 148
Jamaica
82 145 190 237 224 173 212 355 720 556
86 125 168 235 235 508
Jamaica Hills
194
87 120 319 371 163 911 1531
Kew Gardens
76 400
109 196 149 252
Kingsbridge
52 102 101 338
214 271 293 578 778
Laurelton
90 104 137
217
Little Italy
167 215 181
75 125
Long Island City
154 176 232 284 268 213 369
72 79 132 284 400
Lower East Side
196 358 718 378 1607 350 2117 3062
522 107 106
Marble Hill
32 108 195
84 138
Mariners Harbor
92 52 150
67 110 151 199 238 499 899
Melrose
157
97 144 328
Midland Beach
87 139 174 177
290 300 367 571 700 1870 912 2176 795
Midwood
69 87 193 96 282 130
109
Morningside Heights
106 355 566 418 466
73 105
Morris Park
77 172
100
Mott Haven
77 121 279 163 139
142
Mount Hope
55 93 235
219 211 262 774
Navy Yard
200 738
52
New Dorp
195
91 72
NoHo
271 229 425
208 286 589 224
North Riverdale
55 105 114
88
Oakwood
154 143 186 183
69 95 258
Ozone Park
62 107 134 173 228
40 76 155
Park Slope
158 210 305 492 1140 700 500
58 92
Pelham Bay
79 64
50 99 162 137 125 82
Port Ivory
46 76 170
60 146
Port Richmond
70 95 193 297 374 792 1183
2000
1500
1000



In [ ]:

```
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
```

```
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
```

Out[ ]:

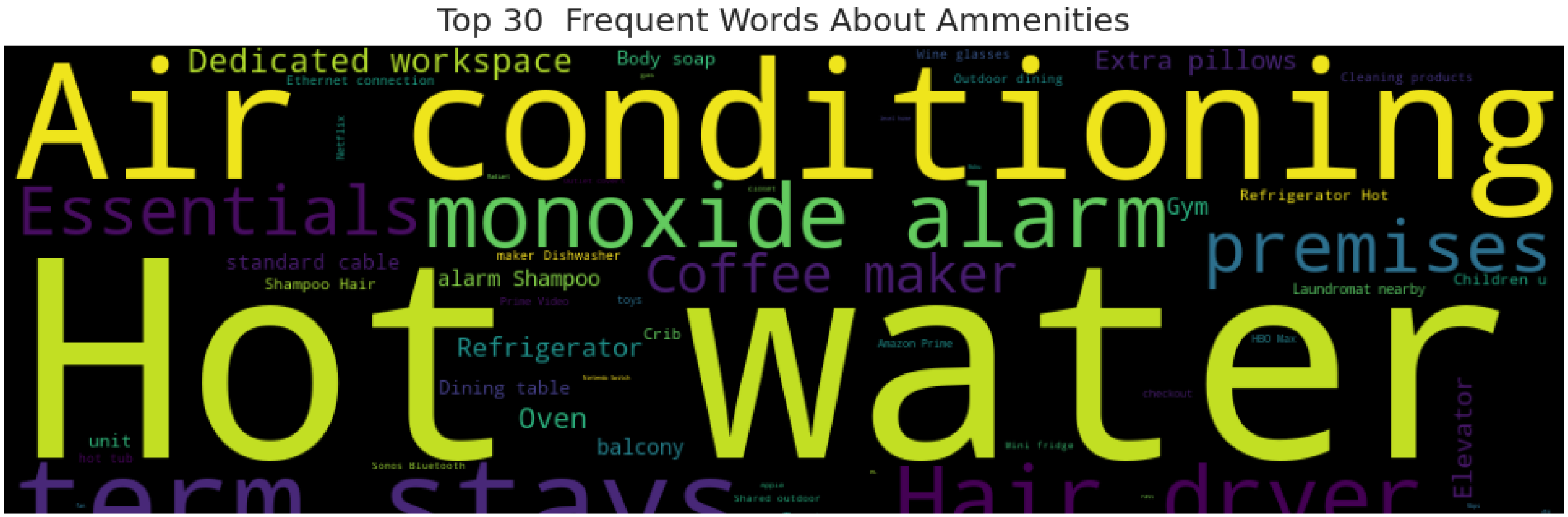
```
True
```

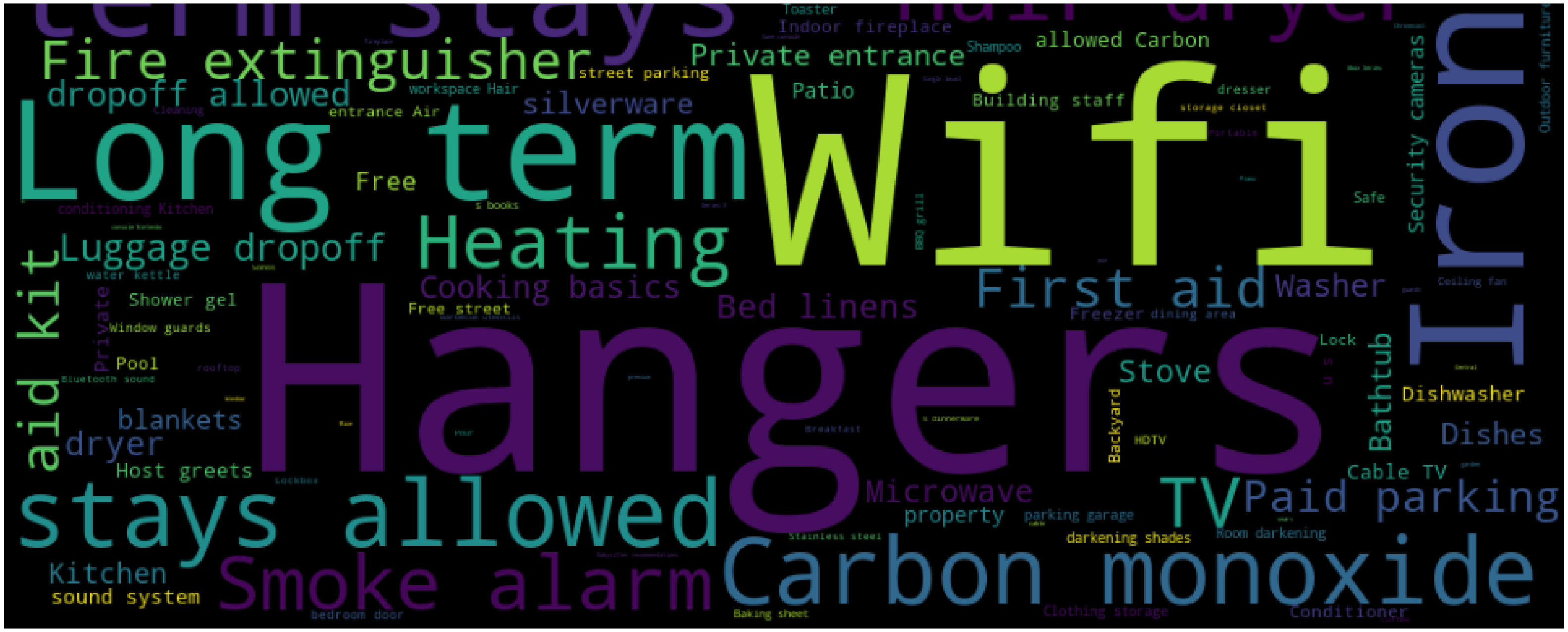
In [ ]:

```
# Analyzing what amenities costs more
amenities = df_listings[['amenities','price','id',]]
amenities_top = amenities.sort_values('price',ascending=[0])
amenities_top = amenities_top.head(30)
allemenities = ''
for index,row in amenities_top.iterrows():
    p = re.sub('[^a-zA-Z]+',' ', row['amenities'])
    allemenities+=p

all_amenities_df=nltk.word_tokenize(allemenities)
filtered_data=[word for word in all_amenities_df if word not in stopwords.words('english')]
wnl = nltk.WordNetLemmatizer()
allemenities_data=[wnl.lemmatize(data) for data in filtered_data]
allemenities_words=' '.join(all_amenities_df)
```

In [ ]:

```
# Plot top 30 ammenities word cloud
wordcloud = WordCloud(width = 1000, height = 700).generate(allemenities_words)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud)
plt.axis("off")
plt.title(f"Top 30  Frequent Words About Ammenities", fontdict={'size': 20,
                                                               'verticalalignment': 'bottom'})
plt.tight_layout() ;
```

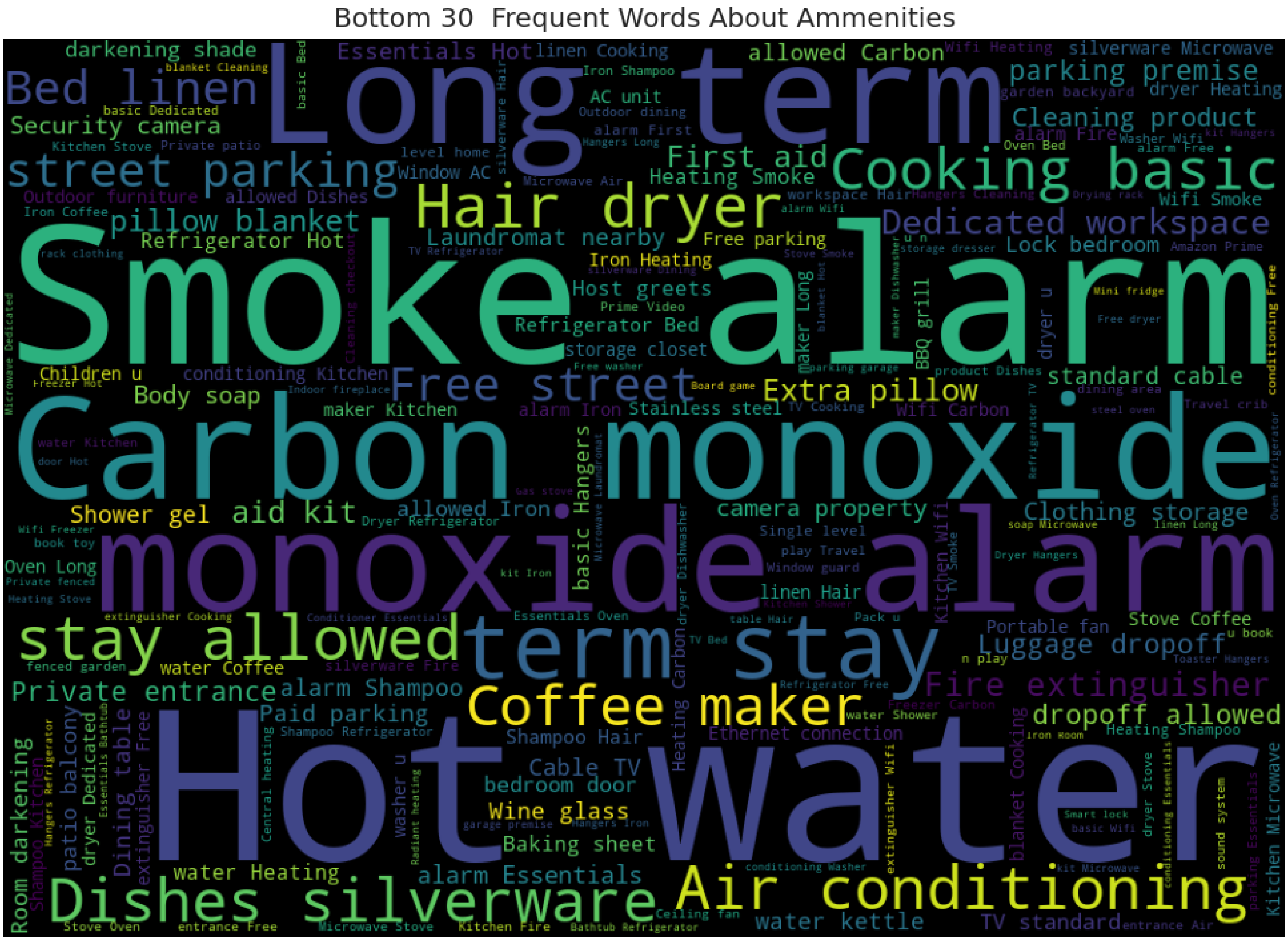




The most frequent words that appear within the ammenities sections are *Air Conditioning, Hot water,Hangers,Wifi, Terms, Hiar Dryer and Heating*

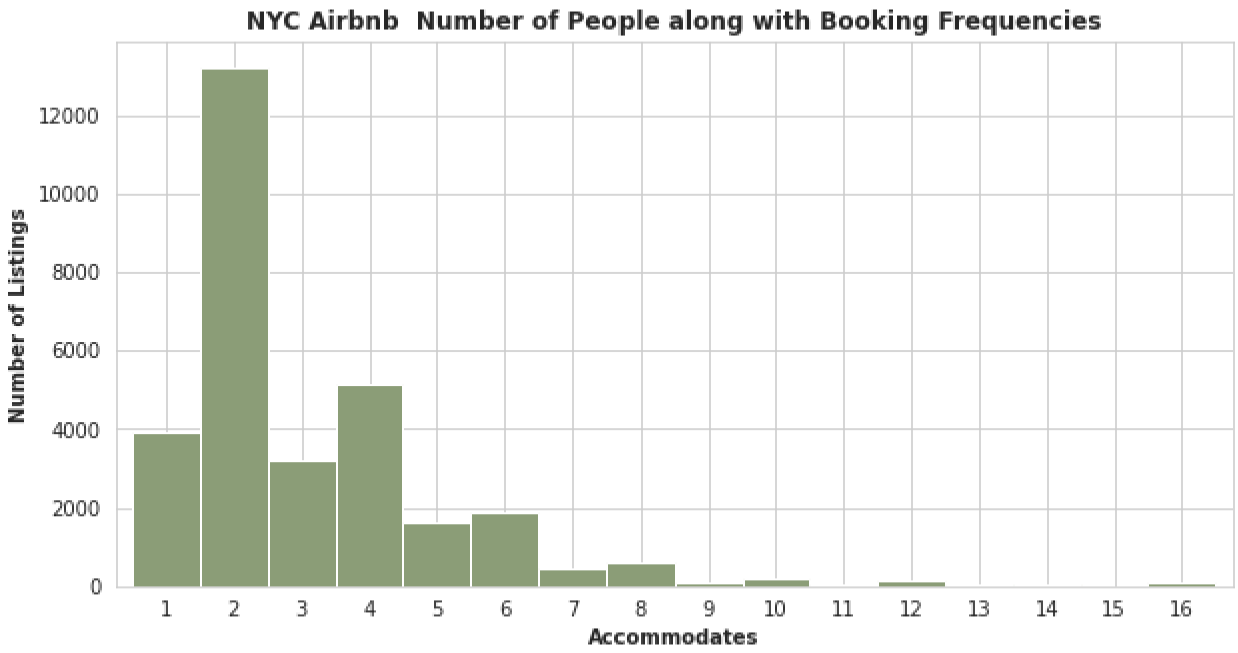
In [ ]:

```
# Plot bottom (rare) 30 ammenities word cloud
amenities_bott = df_listings.sort_values('price',ascending=[1])
amenities_Fbott=amenities_bott.head(30)

allemenities_bott = ''
for index,row in amenities_bott.iterrows():
    p = re.sub('[^a-zA-Z]+',' ', row['amenities'])
    allemenities_bott+=p

allemenities_df_bott=nltk.word_tokenize(allemenities_bott)
filtered_datab=[word for word in allemenities_df_bott if word not in stopwords.words('english')]
wnl = nltk.WordNetLemmatizer()
allemenities_df_bott=[wnl.lemmatize(data) for data in filtered_datab]
allemenities_wordsb=' '.join(allemenities_df_bott)


wordcloud = WordCloud(width = 1000, height = 700).generate(allemenities_wordsb)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud)
plt.axis("off")
plt.title(f"Bottom 30  Frequent Words About Ammenities", fontdict={'size': 20,
                                                                  'verticalalignment': 'bottom'})
plt.tight_layout()
```



In [ ]:

```
# Plot NYC Airbnb  Number of by Booking Freaquency
feq = df_listings['accommodates'].value_counts().sort_index()
feq.plot.bar(figsize=(10,5), width=1, rot=0, color='#8B9D77')
plt.title('NYC Airbnb  Number of People along with Booking Frequencies', fontweight="bold")
plt.ylabel('Number of Listings', fontweight="bold")
plt.xlabel('Accommodates', fontweight="bold")
plt.show()
```



Summary Property Types

- Based on the above analysis we can say that people can find *Entire home/apartment and Private rooms* almost in all NYC major 5 neighbourhoods while only Manhattan includes listings with *Hotel room* type.
- Almost all property types, prices for Entire Home/Apartment is the maximum. This tells us that Property type and Room type plays a very important role in deciding price of a listing.
- The most frequent words that appear within the ammenities sections are: *Air Conditioning, Hot water,Hangers,Wifi, Terms, Hiar Dryer and Heating.*
- Majority of the people make booking for 2 person while 3,4,1 number of people far more less compare to 2 people booking.

4. Conclusion

In [ ]:

```
df_listings.info()
```

```
RangeIndex: 30649 entries, 0 to 30648
Data columns (total 44 columns):
 #   Column                                         Non-Null Count  Dtype  
---  ------                                         --------------  -----  
 0   index                                          30649 non-null  int64  
 1   id                                             30649 non-null  int64  
 2   name                                           30649 non-null  object 
 3   description                                    30649 non-null  object 
 4   neighborhood_overview                          30649 non-null  object 
 5   picture_url                                    30649 non-null  object 
 6   host_name                                      30649 non-null  object 
 7   host_about                                     30649 non-null  object 
 8   host_response_time                             30649 non-null  object 
 9   host_response_rate                             30649 non-null  float64
 10  host_acceptance_rate                           30649 non-null  float64
 11  host_is_superhost                              30649 non-null  object 
 12  host_total_listings_count                      30649 non-null  float64
 13  neighbourhood                                  30649 non-null  object 
 14  neighbourhood_cleansed                         30649 non-null  object 
 15  neighbourhood_group_cleansed                   30649 non-null  object 
 16  latitude                                       30649 non-null  float64
 17  longitude                                      30649 non-null  float64
 18  property_type                                  30649 non-null  object 
 19  room_type                                      30649 non-null  object 
 20  accommodates                                   30649 non-null  int64  
 21  beds                                           30649 non-null  float64
 22  amenities                                      30649 non-null  object 
 23  price                                          30649 non-null  float64
 24  minimum_nights                                 30649 non-null  int64  
 25  maximum_nights                                 30649 non-null  int64  
 26  has_availability                               30649 non-null  object 
 27  availability_30                                30649 non-null  int64  
 28  availability_60                                30649 non-null  int64  
 29  availability_90                                30649 non-null  int64  
 30  availability_365                               30649 non-null  int64  
 31  number_of_reviews                              30649 non-null  int64  
 32  review_scores_rating                           30649 non-null  float64
 33  review_scores_accuracy                         30649 non-null  float64
 34  review_scores_cleanliness                      30649 non-null  float64
 35  review_scores_checkin                          30649 non-null  float64
 36  review_scores_communication                    30649 non-null  float64
 37  review_scores_location                         30649 non-null  float64
 38  review_scores_value                            30649 non-null  float64
 39  instant_bookable                               30649 non-null  object 
 40  calculated_host_listings_count                 30649 non-null  int64  
 40  calculated_host_listings_count               30649 non-null  int64
 41  calculated_host_listings_count_entire_homes   30649 non-null  int64
 42  calculated_host_listings_count_private_rooms  30649 non-null  int64
 43  calculated_host_listings_count_shared_rooms   30649 non-null  int64
dtypes: float64(14), int64(14), object(16)
memory usage: 10.3+ MB
```

4.1. Cleaning Process

- The overall dataset had few null values for within some features. We dropped some of the unnecessary columns.
- Within the text columns have been applied some text preprocessing techniques such: oconverting into lowercase,remove square brackets,links,punctuation and words containing numbers.

4.2. Summary of Exploration

An Exploratory data analysis have been applied based on the following sections:

Host Type

- Majority of super hosts are from the Brooklyn while *Queens, Bronx* and *Staten Island* have nearly an equal amount of host types (super/regular)
- The hosts that have responded within a few days or more have been received lower ratings up to 0.45%. from the plot we can see that if hosts can respond within a few hours up to maximum within a da there is higher chance to get better ratings. The majority of the super hosts also fall in this gap which proofs their responsibility.
- Based on the above word cloud we can say that super hosts are makie emphesases on being welcoming, helpful. *Enjoy, Love, Feel, Excellent* are also the main characteristics.
- Regular hosts are expressed with the frequent words such as *Love, Work,Food, Unique and Unforgettable*

Neighbourhoods

- The majority of the listings (11000/14000) are located in *Brooklyn* and *Manhattan* while *Staten Island* is in the last place with the least amount of the listings.
- The top 10 neighbourgood which includes the most of the listings are: *Bedford-Stuyvesant, Harlem, Williamsburg, Upper West Side, Upper East Side, Bushwick,Crown Heights, Hell's Kitchen, Midtown and Chelsea*.
- Even though Brooklyn includes the more listings Manhattan listing prices are more higher. Staten Island also showing more expensive listings despite the less amount of listings compare to other neighbourhoods. Average price starts from $100 and above