IST 5520 Milestone 2: Data Analysis I

AirBnB Dataset ¶

Student: Ronald Adomako

Student: Idris Dobbs

Student: Narendra Chigili

Student: Nikhil Srirama Sai

Instruction 1: Cleanse and visualize data

Introduction

AirBnB is a PaaS for the short term rental market. Users use the platform to list residences for short term rentals. We noticed that for big cities, such as New York City, there are many host and some may want to use AirBnB for lucrative means. Given location and characteristics of a property, a new host would want to know whether he or she is charging the optimal amount to rent the space to lodgers.

We noticed that from the New York City dataset for AirBnB, the categorical variable for location was too coarse. The descriptor says the location is categorized into five boroughs. For a big city such as New York City, there are a lot of insights missing from a business perspective because neighborhoods vary drastically in property amenities even within a single borough. Moreso, the geo-coordinates are too fine for business purposes. To handle this we implement a zip code converter to categorize properties based on their location - Feature Selection.

- What are the largest determinants / predictors of AirBnB rental prices?
- How can we optimize rental revenue based on rental location and other characteristics?
- What price should be charged based on rental location/ characteristics?

We are opperating under the assumption that NYC AirBnB prices has reached a *steady-state*: i.e. the market has been active for long enough in NYC and there are enough data points (observations) in NYC that the **mean** is meaningful.

We want to know whether a host is charging an optimal price. To do this we group the observations by neigborhood and then take the average price. Hosts who charge at or above this price are considered optimal in their respective neighborhood while hosts who charge below the average price for their neighborhood are sub-optimal. Consider the case where all hosts charge the same price within a neighborhood, then the mean is the mode is the median- uniform data, no variance. All the hosts in this neighborhood would be optimal.

Consider the case where one of those hosts charge below what would have been the average, then only that host is sub-optimal while the rest are optimal. Conversely, if one host charge above the rest while everyone else charges the same, then that one host would be optimal while the rest are suboptimal.

Along with the *steady-state* assumption, by grouping the data by neighborhoods we assume that on average homes and amenities are similar by neighborhood. The geo-coordinates are too fine a scale and the boroughs are too coarse a scale. A meso-scale would be by zipcode, which we would expect to have higher precision of similarities between host, or by neighborhood. For a dataset with 39881 observations, transforming geo-coordinates prove to be computationally expensive (22 hours on standard household computer). We chose the next best meso-scale: what AirBnB features as "neighbourhood".

From a business perspective, we want to know what percent of hosts per neighborhood are charging an optimal price, and aggregating this data the percent hosts charge an optimal price in NYC overall. We see that using the neighborhood grouping allows us to compare on a common scale for all hosts. We don't have hosts income, so we wouldn't be able to measure profit. Likewise, revenue wouldn't be a fair scale because hosts with more units will outperform host with smaller units just by volume. A nieghborhood comparison allows a better metric to assess price per room, where we expect reasonably small variance per neighborhood. Furthermore, comparing by percentage is normalizes our comparison in general.

Data Source and Collection

We chose the AirBnB dataset for New York City (NYC). We want to build a model that indicates whether hosts are charging an optimum amount for their rental.

http://insideairbnb.com/new-york-city/ (http://insideairbnb.com/new-york-city/)

data dictionary (https://docs.google.com/spreadsheets/d/1iWCNJcSutYqpULSQHlNyGInUvHg2BoUGoNRIGa usp=sharing)

http://insideairbnb.com/get-the-data (http://insideairbnb.com/get-the-data)

http://data.insideairbnb.com/united-states/ny/new-york-city/2022-09-07/visualisations/listings.csv (http://data.insideairbnb.com/united-states/ny/new-york-city/2022-09-07/visualisations/listings.csv)

In [123]:
```
import pandas as pd
import matplotlib.pyplot as plt
import sidetable as stb
import numpy as np
import seaborn as sns

%matplotlib inline
```

In [130]:
```
data = pd.read_csv('data_dictionary.csv')
#pd.set_option('display.max_rows',1000)
#data
```

In [131]:
```
data_dict = pd.DataFrame(df2.columns, columns=['Features'])
data_dict['Description']=None
```

In [132]:
```
# aux = data[data['Field']=='id']['Description']
# print(aux.values)
# aux
```

In [133]:
```
# aux = data[data['Field']=='id'][['Description']]
# print(aux.values)
# print(aux.values[0])
# print(aux.values[0][0])
# aux.values
```

Dimensional Analysis

The data dictionary for New York City AirBnB dataset consists of 75 columns (variables or features) and 39881 observations. Based on our research question and several Kaggle challenges (https://www.kaggle.com/search?q=airbnb-listing-in-nyc) (www.kaggle.com/search?q=airbnb-listing-in-nyc (http://www.kaggle.com/search?q=airbnb-listing-in-nyc)) we reduced our dimensions to the following **18** features for preliminary analysis.

In [134]:
```
#pd.set_option?
```

In [140]:
```
for i,feature in enumerate(df2.columns):
    #print(feature)
    #print(type(feature))
    val = data[data['Field']==feature]['Description'].values
    try:
        data_dict.loc[i,'Description'] = val[0]
    except: #IndexError
        data_dict.loc[i,'Description'] = None

    if feature == 'neighbourhood':
        data_dict.loc[i,'Description'] = \
            'Neighborhood equivalent for zip code group'
    if feature == 'neighbourhood_group':
        data_dict.loc[i,'Description'] = \
            'Borough'

pd.set_option('display.max_colwidth', 100)
data_dict

```

Out[140]:

	Features	Description
0	id	Airbnb's unique identifier for the listing
1	name	Name of the listing
2	host_id	Airbnb's unique identifier for the host/user
3	host_name	Name of the host. Usually just the first name(s).
4	neighbourhood_group	Borough
5	neighbourhood	Neighborhood equivalent for zip code group
6	latitude	Uses the World Geodetic System (WGS84) projection for latitude and longitude.
7	longitude	Uses the World Geodetic System (WGS84) projection for latitude and longitude.
8	room_type	[Entire home/apt|Private room|Shared room|Hotel]
9	price	daily price in local currency
10	minimum_nights	minimum number of night stay for the listing (calendar rules may be different)

11	number_of_reviews	The number of reviews the listing has
12	last_review	The date of the last/newest review
13	reviews_per_month	The number of reviews the listing has over the lifetime of the listing
14	calculated_host_listings_count	The number of listings the host has in the current scrape, in the city/region geography.
15	availability_365	avaliability_x. The availability of the listing x days in the future as determined by the calend...
16	number_of_reviews_ltm	The number of reviews the listing has (in the last 12 months)
17	license	The licence/permit/registration number

In [144]:
```
csv_URL = "http://data.insideairbnb.com/united-states/ny/new-york-
df = pd.read_csv(csv_URL)
```

Data Manipulation

In [147]:
```
df['neighbourhood_group'].unique()
```

Out[147]:
```
array(['Brooklyn', 'Queens', 'Bronx', 'Manhattan', 'Staten Island'],
      dtype=object)
```

In [148]:
```
#Create a data frame grouping by neighborhood for average price
hood_price_obj = df[['neighbourhood','price']].groupby('neighbourh
df_mean_price = hood_price_obj.mean()
df_mean_price[['price']] = df_mean_price[['price']].round(2)
df_mean_price
```

Out[148]:

neighbourhood	price
Allerton	118.78
Arden Heights	113.86
Arrochar	132.06
Arverne	230.26
Astoria	109.01
Bath Beach	117.19
Battery Park City	230.90
Bay Ridge	121.55
Bay Terrace	151.25
Baychester	111.28

In [149]:
```
df_mean_price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 244 entries, Allerton to Woodside
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   price   244 non-null    float64
dtypes: float64(1)
memory usage: 3.8+ KB
```

We have reduced 39881 into 244 rows of manageable data!

In [151]:
```
df.stb.freq(['neighbourhood'])
```

Out[151]:

	neighbourhood	count	percent	cumulative_count	cumulative_percent
0	Bedford-Stuyvesant	2779	6.968230	2779	6.968230
1	Williamsburg	2456	6.158321	5235	13.126551
2	Harlem	1878	4.709009	7113	17.835561
3	Midtown	1701	4.265189	8814	22.100750
4	Bushwick	1657	4.154861	10471	26.255610
5	Upper West Side	1630	4.087159	12101	30.342770
6	Hell's Kitchen	1578	3.956771	13679	34.299541
7	Upper East Side	1379	3.457787	15058	37.757328
8	Crown Heights	1197	3.001429	16255	40.758757
9	East Village	1057	2.650385	17312	43.409142
10	Chelsea	901	2.259221	18213	45.668363

In [152]:
```
#df.stb.freq(['neighbourhood']).describe()
```

In [154]:
```
#df = df2.sort_values('neighbourhood')
df_hood = df.stb.freq(['neighbourhood'])
df_hood = df_hood.loc[:,'neighbourhood':'percent']
df_hood = df_hood.sort_values('neighbourhood')
```

In [155]:
```
df_hood.reset_index(inplace=True)
```

In [156]:
```
df_mean = df_mean_price[['price']].reset_index()
df_mean.head()
```

Out[156]:

	neighbourhood	price
0	Allerton	118.78
1	Arden Heights	113.86
2	Arrochar	132.06
3	Arverne	230.26
4	Astoria	109.01

In [73]:
```
df_hood['price'] = df_mean['price']
df_hood.rename(columns={'index':'pop_rank'}, inplace=True)
df_hood.head()
```

Out[73]:

	pop_rank	neighbourhood	count	percent	price
0	107	Allerton	45	0.112836	118.78
1	204	Arden Heights	7	0.017552	113.86
2	158	Arrochar	17	0.042627	132.06
3	64	Arverne	110	0.275821	230.26
4	14	Astoria	686	1.720117	109.01

Data Summarization and Visualization

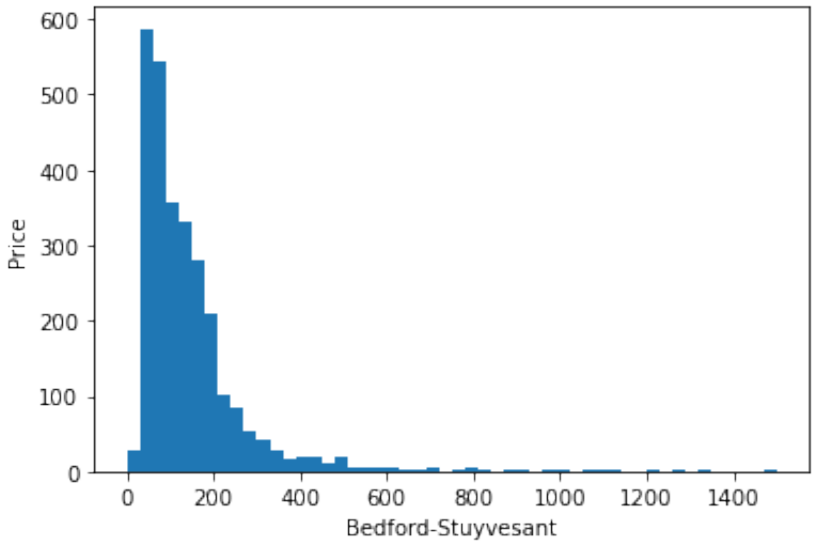
Inspect distribution of top three most populous neighborhoods

In [181]:
```
df2_s1 = df.groupby('neighbourhood').get_group('Bedford-Stuyvesant
df2_s2 = df.groupby('neighbourhood').get_group('Williamsburg')
df2_s3 = df.groupby('neighbourhood').get_group('Harlem')
```

In [182]:
```
df_top3= pd.concat([df2_s1,df2_s2,df2_s3])
```

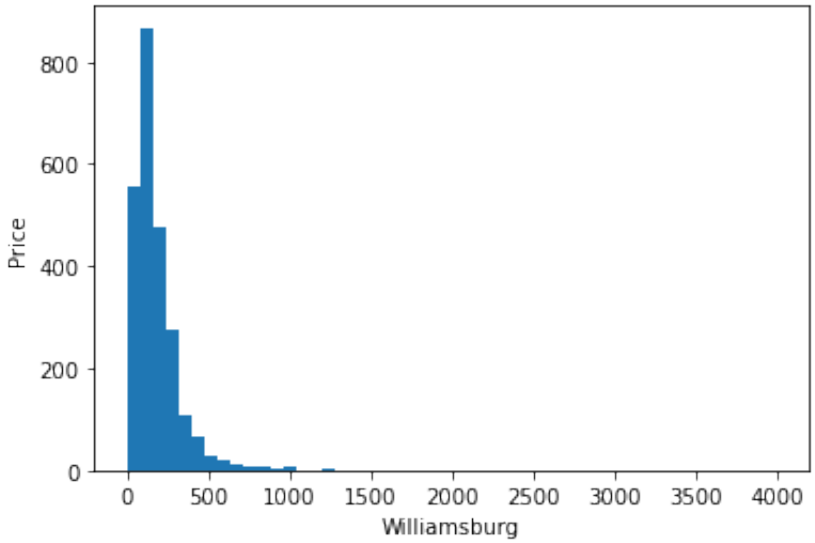
In [191]:

```
plt.hist(df2_s1.price, bins=50)
plt.xlabel('Bedford-Stuyvesant')
plt.ylabel("Price");
```

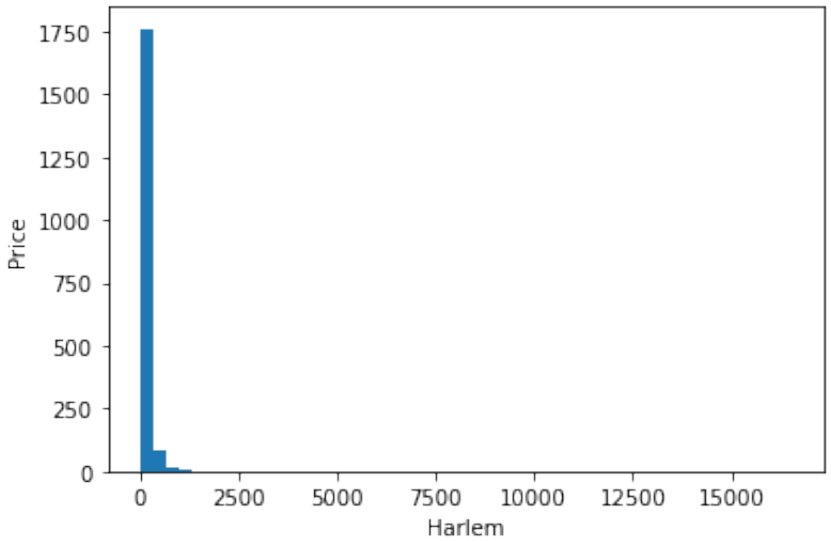


In [193]:

```
plt.hist(df2_s2.price, bins=50)
plt.xlabel('Williamsburg')
plt.ylabel("Price");
```



In [199]:
```
plt.hist(df2_s3.price,bins = 50)
plt.xlabel('Harlem')
plt.ylabel("Price");
```



Interpreation

The aim was to get a normally distributed curve for our assumptions in obtaining a mean price per night. Without correcting for outliers, the data seems to approach a logarithmic or skewed normally distributed curve, in which case taking the median or mode for the optimum price would be most appropriate.

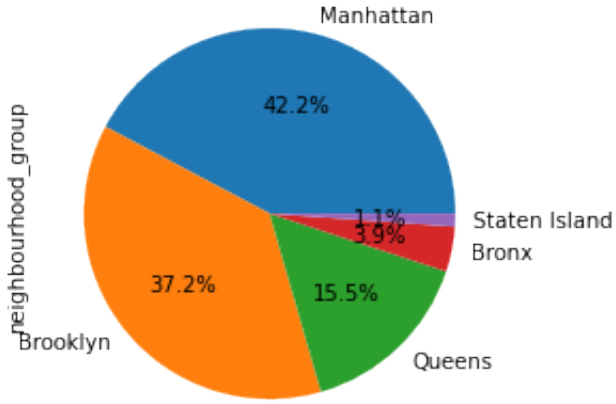
Proportion of AirBNB Listings by Borough and Room Type in NYC

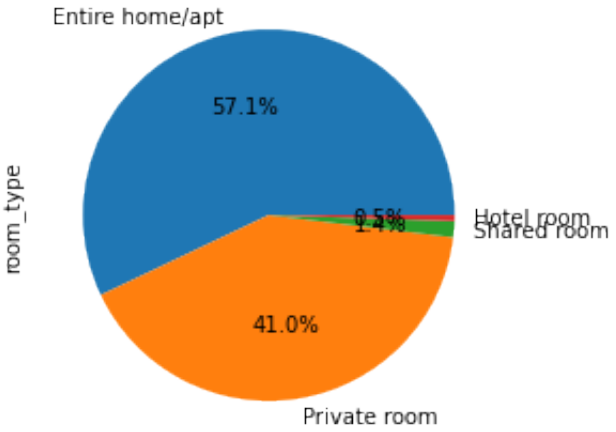
The 1st pie chart shows 95% of the AirBnB listings are in Manhattan, Brooklyn and Queens. Brooklyn and Staten Island make up the remaining listings. Manhattan and Brooklyn alone make up neary 80% of the listings.

The 2nd pie chart shows the listings distributed by room type. Most of the observations consist of entire home / apartments, or private rooms. Hotel and shared rooms are an insignificant proportion of the distribution.

In [166]:

```
#Create a pie chart showing the percentage of listings per bourrou
df1 = df.neighbourhood_group.value_counts()
df1.plot.pie(autopct="%.1f%%")
plt.show()

df2 = df.room_type.value_counts()
df2.plot.pie(autopct="%.1f%%")
plt.show()
```





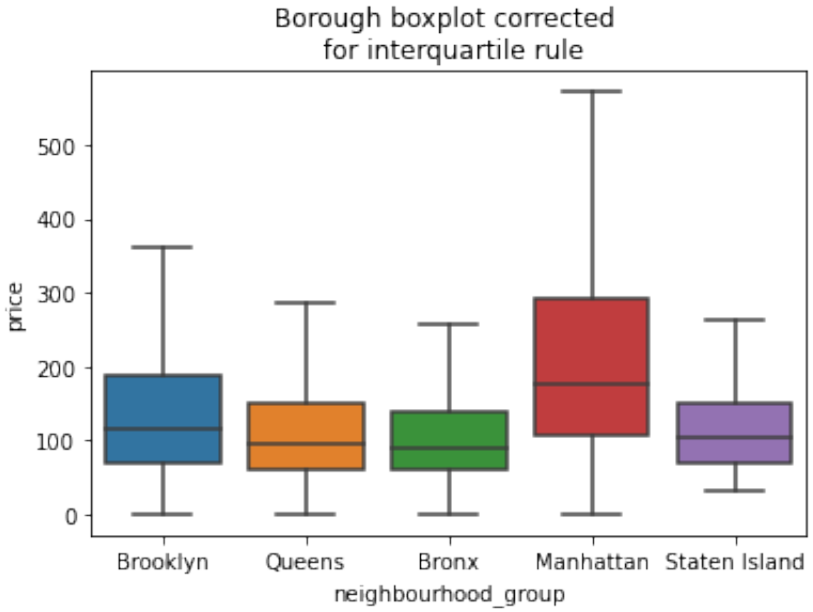
NYC AirBNB Listing Price Statistics by Borough

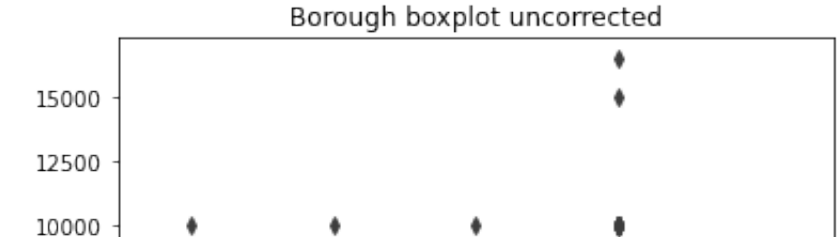
The table and box/whisker plots above provide illustration of the frequency, price statistics, and price distribution by neighborhood group or burrough in New York City. There is wide variability in the observations, with standard deviations consistently higher than the mean for each burrough. Mean prices are consistently higher than the median price, which suggest the prevalance of high values or outliers in the dataset that are pulling up the average. Average prices are highest in Manhattan, Brooklyn, and Queens respectively.

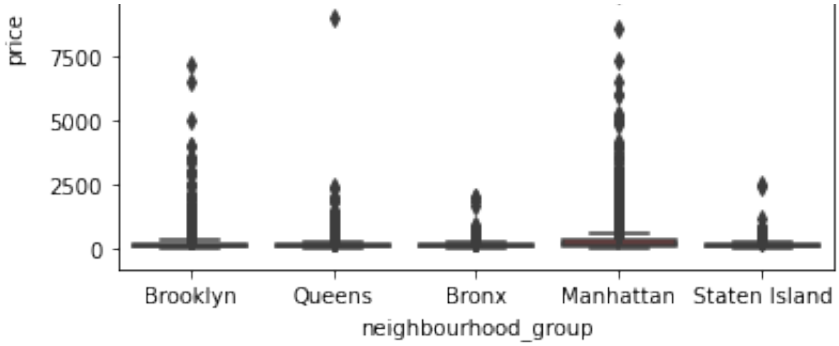
In [171]:

```
#Display the table of price statistics and counts by New York City
display(df.groupby('neighbourhood_group').aggregate({'neighbourhoo
                                    'price':['mean','median','std'

#Plot boxplot with outliers turned off.
sns.boxplot(y = df['price'],
            x = df['neighbourhood_group'],
            showfliers = False)
plt.title('Borough boxplot corrected \n for interquartile rule')
plt.show()

sns.boxplot(y = df['price'], x = df['neighbourhood_group'], showfl
plt.title('Borough boxplot uncorrected')
plt.show()
```

	neighbourhood_group	price				
	count	mean	median	std	min	max
neighbourhood_group						
Bronx	1568	124.737245	90.0	278.572839	0	9994
Brooklyn	14845	157.927114	115.0	209.526092	0	10000
Manhattan	16847	264.933341	175.0	473.171623	0	16500
Queens	6175	131.365506	94.0	213.120396	0	10000
Staten Island	446	143.163677	103.5	194.997315	33	2500







NYC AirBNB Listing Price Statistics by Room Type

The table and box/whisker plots above provide illustration of the frequency, price statistics, and price distribution by room type in New York City. As in the distribution by borough section, there is wide variuability with standard deviations consistently higher than the mean for each burrough (Hotel rooms being the exception). Mean prices are also consistently higher than the median price, which suggest the prevalance of high values or outliers in the dataset that are pulling up the average. Comparisons of the 1st boxplot (corrected for outliers) with the second (not corrected for outliers bear this out.

Major Room Type Categories

These categories encompass 98% of the listings and are likely to be more useful for informing business decisions.

- Entire home/apt: Entire residences make up 57% of the dataset and are the category with the second highest average price. Prices range from $10 to $15K which could be indicative of data errors and/ or outliers.
- Private room: Private rooms constitute 42% of the listings in the data and are the category with the third highest average price. Prices range from $10 to $16.5K which could be indicative of data errors and/ or outliers.

Minor Room Type Categories

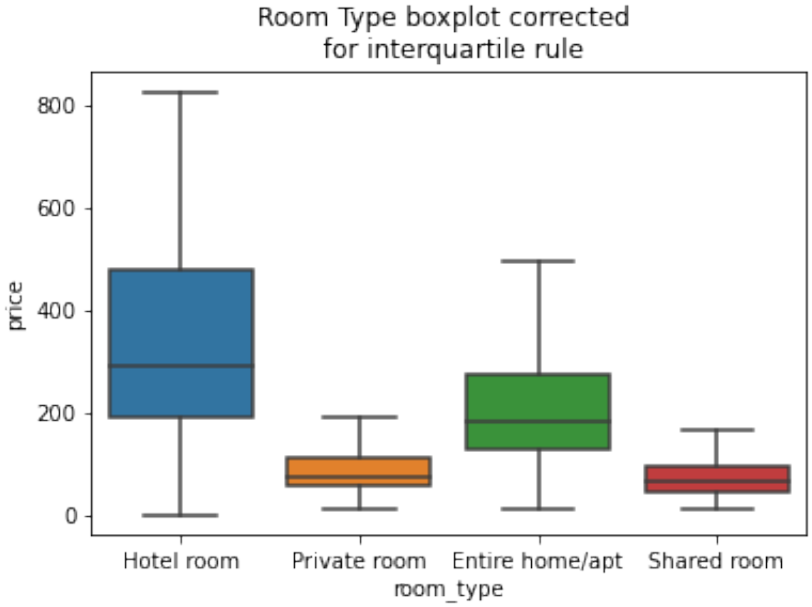
Categories that are a significantly smaller number of the overall listings in the dataset. They are less likely to be useful for informing business decisions.

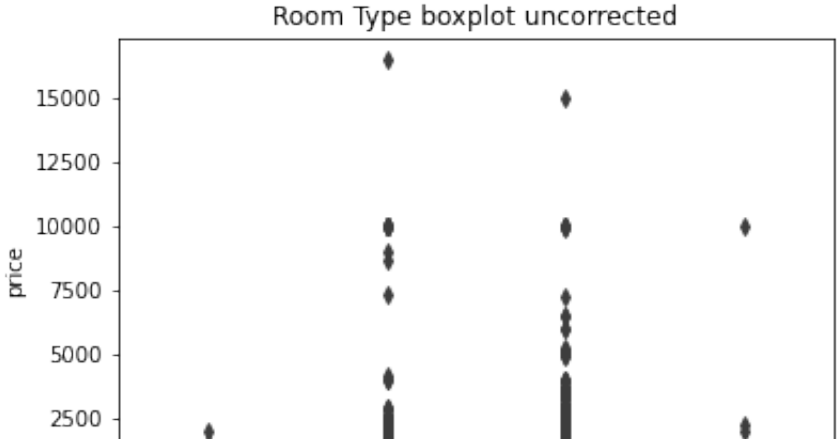
- Shared room: Shared rooms constitute 1.4% of the listings with the lowest average price of all room type categories. Prices range from $10 to $10K which seems to indicate data entry errors and/or outliers.
- Hotel room: Hotel rooms seem to have the highest average price and the least number of observations. This is the only category where the standard deviation is less than the mean, which suggest a more limited number of high priced outliers.

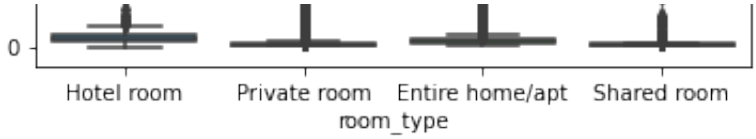
In [172]:

```
#Display the table of price statistics and counts by New York City
display(df.groupby('room_type').aggregate({'room_type':'count','pr

#Plot boxplot with outliers turned off.
sns.boxplot(y = df['price'], x = df['room_type'], showfliers = Fal
plt.title('Room Type boxplot corrected \n for interquartile rule')
plt.show()
sns.boxplot(y = df['price'], x = df['room_type'], showfliers = Tru
plt.title('Room Type boxplot uncorrected')
plt.show()
```

	room_type	price				
	count	mean	median	std	min	max
room_type						
Entire home/apt	22761	251.546022	180.0	338.044654	10	15000
Hotel room	202	371.648515	291.0	303.482491	0	1998
Private room	16361	122.936495	75.0	356.373737	10	16500
Shared room	557	119.398564	66.0	454.106078	10	10000







AirBNB Listings Statistics by Borough and Room Type in NYC

The listings dataset was aggregated by borough and room type for statistical analysis. The results underscore the need to remove outliers and illogical values from the data.

In [175]:
```
#Show price statistics by neighborhood group and room type
df_borrough = df[[ 'neighbourhood_group', 'room_type','price']]
display(df_borrough.groupby(['neighbourhood_group', 'room_type']).
    'mean', 'median','std', 'min', 'max'}))
```

		price					
		min	max	mean	count	std	median
neighbourhood_group	room_type						
Bronx	Entire home/apt	28	2000	164.569293	736	156.115405	130.0
	Hotel room	0	0	0.000000	1	NaN	0.0
	Private room	11	9994	90.527112	793	356.925302	65.0
	Shared room	10	775	70.447368	38	124.059423	35.0
Brooklyn	Entire home/apt	30	7184	216.452539	8154	228.702022	169.0
	Hotel room	0	529	177.555556	9	194.204860	145.0
	Private room	10	10000	86.907680	6510	157.575675	69.0
	Shared room	15	1000	70.389535	172	84.576582	45.0
Manhattan	Entire home/apt	29	15000	300.645829	10862	420.865138	205.0
	Hotel room	0	1998	392.163934	183	308.268374	307.0
	Private room	10	16500	194.915346	5552	549.384274	100.0
	Shared room	29	10000	175.124000	250	662.012440	82.0
Queens	Entire home/apt	10	10000	191.693713	2736	245.295887	150.0
	Hotel room	0	282	189.888889	9	89.324471	209.0
	Private room	19	9000	83.118776	3334	169.268863	65.0
	Shared room	16	1250	82.093750	96	156.164089	50.0
Staten Island	Entire home/apt	39	2500	180.487179	273	236.437035	129.0
	Private room	33	500	84.412791	172	65.521555	68.0
	Shared room	59	59	59.000000	1	NaN	59.0

Correlation Matrix

A correlation matrix was conducted for the listings data. At this point, most of the factors appear to be weakly correlated with price, which is the primary variable of concern.

In [176]:
```
1 df.corr(method='pearson')
```

Out[176]:

	id	host_id	latitude	longitude	price	minimum_ı
id	1.000000	0.335359	-0.004366	0.079277	0.047596	-0.1
host_id	0.335359	1.000000	0.027405	0.144694	0.039558	-0.1
latitude	-0.004366	0.027405	1.000000	0.048995	0.027974	0.0
longitude	0.079277	0.144694	0.048995	1.000000	-0.123122	-0.0
price	0.047596	0.039558	0.027974	-0.123122	1.000000	-0.0
minimum_nights	-0.119177	-0.147641	0.030504	-0.083799	-0.035304	1.0
number_of_reviews	-0.185724	-0.095511	-0.033748	0.041505	-0.032691	-0.1
reviews_per_month	0.246370	0.271079	-0.037751	0.102208	0.019562	-0.2
calculated_host_listings_count	0.040260	-0.024191	0.033737	-0.071354	0.042761	0.1
availability_365	0.286508	0.247358	-0.018182	0.123042	0.095482	-0.0
number_of_reviews_ltm	-0.080899	0.126596	-0.034825	0.057616	-0.002458	-0.2

Map Visualization of the Distribution of Price by Neighborhood in NYC

This section provides analysis on the distribution of price by neighborhood. The mean price by neighborhood was computed from the data, keyed on a geojson file and incorporated into a folium map. There are several clusters of relatively higher prices, but the most significant one is around the Manhattan area.

Conclusion and TakeAways

- Average listing prices for Manhattan tend to be the highest, followed by Brooklyn, Queens, Staten Island, and the Bronx. Most of the listings are in Manhattan, Brooklyn, and Queens.
- Entire residences/apartments and private rooms comprise ~97% of the listings here. Hotels and shared rooms are less than 2% of the dataset. Hotels tend to have the highest average price followed by residences, private rooms, and lastly shared rooms. Any predictive model underlying algorithms should be differentiated by room type. In terms of business analytics utility, residences and private rooms seem to be the most promising, while hotels and shared rooms may be less useful due to the relatively limited amount of data.
- The variability in the data is significant. The data should be disaggregated by room type, and all values greater than the 3rd quartile + (1.5 * IQR) should be removed. also, all zero values should also be removed.
- It is hypothesized that properties closer to either Manhattan or other significant attractions will have higher list prices ceteris paribus.

Instruction 2: Jupyter Notebook

In [178]:
```
1 !which python
```

```
/opt/anaconda3/envs/MyEnv/bin/python
```

In [179]:
```
1 !which jupyter
```

```
/opt/anaconda3/envs/MyEnv/bin/jupyter
```

In [180]:
```
1 !which python3
```

```
/opt/anaconda3/envs/MyEnv/bin/python3
```

Instruction 3: Github Repository, Handles, and Evaluation

https://github.com/Naren1610/IST5520GrpProj/tree/milestone2
(https://github.com/Naren1610/IST5520GrpProj/tree/milestone2)

IST5520GrpProj

Ronald Adomako, adomakor412

Idris Dobbs, idobbs-2012

Narendra Chigili, Naren1610

Nikhil Srirama Sai, SaiNikhilPalaparthi