NYC Airbnb Price Analysis

SI 618 Project 2 Report

Yixin Zheng (yixinzh@umich.edu)

Motivation and Summary

Owing to the personal experience in Airbnb in New York, I am quite curious about the elements that contribute to Airbnb price. My goal is to analysis the factors contributing to prices. The interesting fields are the time host starts, host response time, host response rate, host accuracy rate, whether host is super host, host verifications, whether host has profile pictures, whether host identify is verified, neighborhood group, property type, room type, accommodates, bathroom, bedrooms, beds, bed type, amenities, square feet, review score rating, cancellation. Before dive into the fields, I'll drop the rows with missing values in certain fields.

The dataset, Airbnb Detailed Listings Data for New York City (2015), is the same as project1. In project1, I explored the relationship between price per night and neighborhood group, unemployment as well as income per capita. In project2, there are more classifier and the price per monthly and price weekly are also considered as outcome variable.

Datasets

1. Airbnb Detailed Listings Data for New York City

Source:

http://data.insideairbnb.com/united-states/ny/new-york-city/2015-12-02/data/listings.csv.gz

The dataset is a csv file covered the period in 2015.

It is downloaded from Inside Airbnb site. There are 34376 records of listing

neighbou	room_typ	price
Bronx	Private ro	$60.00
Bronx	Entire hor	$179.00
Bronx	Private ro	$49.00
Bronx	Entire hor	$300.00
Bronx	Entire hor	$200.00
Bronx	Entire hor	$88.00
Bronx	Entire hor	$95.00

It contains field such as price, accommodates, bathrooms, beds, number_of_reviews, reviews_per_month, review_scores_rating, review_scores_accuracy, review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_scores_location, review_scores_value, neighbourhood_group_cleansed, property_type, room_type, bed_type, amenities.

2. New York City census tracts

Source:

https://www.kaggle.com/muonneutrino/new-york-city-census-data?select=nyc_census_tracts.csv

The dataset is a csv file covered the period in 2015.

This file contains a selection of census data taken from the ACS DP03 and DP05 tables. There are 2167 records in the dataset.

It contains fields such as total population, racial/ethnic demographic information, employment and commuting characteristics, income per cap and more are so on. I chose to use borough and IncomePerCap.

Borough	TotalPop	IncomePerCap	Employed
Bronx	7703	2440	0
Bronx	5403	22180	2308
Bronx	5915	27700	2675
Bronx	5879	17526	2120
Bronx	2591	17986	1083
Bronx	8516	12023	2508
Bronx	4774	9781	1191

Manipulation & Analysis

1. What's the relationship between price and census factors?

Importing and cleaning:

Most work was in colab and a small part was done in R. To use the pyspark in colab, I first installed the related modules, and then import the models and csv files.

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
!wget -q https://downloads.apache.org/spark/spark-3.0.1/spark-3.0.1-bin-hadoop3.2.tgz
!tar xf spark-3.0.1-bin-hadoop3.2.tgz
!pip install -q findspark
```

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.0.1-bin-hadoop3.2"
```

```
import findspark
findspark.init("spark-3.0.1-bin-hadoop3.2")
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local[*]").getOrCreate()
```

The csv files were read in pyspark and pandas. I select the field that I'm interested in.

```
sc = SparkContext.getOrCreate()
sqlContext=SQLContext(sc)
df_airbnb=sqlContext.read.option("multiline", "true").option("quote", '"').option("escape", "\\").option("escape", '"').csv('data/listings.csv', header=True)
df_airbnb.registerTempTable('airbnb')
df_airbnb_p=sqlContext.sql("select neighbourhood_group_cleansed, room_type,price from airbnb where neighbourhood_group_cleansed is not null and price is not null and room_type is not null")
df_airbnb_p_rdd=df_airbnb_p.rdd.map(lambda x: (x[0].strip(),x[1],x[2][1:].strip()))
```

I remove the dollar sign before the number, the comma in the number, dropping the null value, joining the datasets, changed the data type from object or string to float or integer. Part of the work is done by Sparksql, RDD, Spark map, part is done in pandas. The field amenities are split by comma to a list, and then calculate the length of it to count it as a numerical variable

amenities
{TV,"Cable TV",Internet,"Wireless Internet","A...

amenities
14
14

Joining

To explore the relationship between Airbnb price and census factors, I should join the price from Airbnb listing dataset with the census factors. The Airbnb listing data set has the field longitude and latitude, while census tracts has the census tract code. The code can be generated in R.

But the running speed is quite slow: it spent half seconds to generate each record. So I filter the Airbnb listing dataset to the neighborhood group in Manhattan, room type as entire home/apartment, accommodates as 2. By filtering the data, the records reduced from 34376 to 4388.

I imported the files in R and add the census_code field, writed it out and input it in colab.

````
```{r}
test<-listings %>%
  filter(neighbourhood_group_cleansed=='Manhattan', room_type=='Entire home/apt',accommodates==2)

test%>%
  nrow()

lat = test$latitude
lon = test$longitude
test$census_code <-mapply(call_geolocator_latlon, lat = lat, lon = lon)

```
````

````
```{r}
test$census_code
```
````

```
 [1] "360050516002005" "360050516002005" "360050516002002" "360050516003012" "360050516002004" "360050516001006"
 [7] "360050516002012" "360050516002003" "360050516000002" "360050516003001" "360050342002007" "360050332011003"
[13] "360050332011000" "360050332022000" "360050348004002" "360050332011003" "360050342002006" "360050332012000"
[19] "360050326001001" "360050348003001" "360050344001000" "360050332024000" "360050340002000" "360050344001002"
[25] "360050344001002" "360050224031000" "360810135001004" "360810317001010" "360810113002000" "360810113001001"
[31] "360810117001000" "360810113003002" "360810121001001" "360810105003001" "360810113001001" "360810135001003"
[37] "360810111001001" "360810103004000" "360810317001006" "360810111002003" "360810113001001" "360810113002001"
[43] "360810123011009" "360810117002001" "360810105003002" "360810117003000" "360810113002000" "360810115001000"
[49] "360810117004001" "360810111001000" "360810097004001" "360810121002000" "360810137001014" "360810111002002"
[55] "360810103003001" "360810097004000" "360810125001002" "360810101002006" "360810115002002" "360810125002002"
[61] "360810097002001" "360810117003000" "360810121001000" "360810111001002" "360810317001004" "360810119002000"
[67] "360810117003002" "360810113003001" "360810113001001" "360810105004000" "360810117002000" "360810097004001"
[73] "360810111002006" "360810117003001" "360810137001000" "360810111002001" "360810117003002" "360810125001001"
[79] "360810107011000" "360810113002001" "360810095001000" "360810119002001" "360810117003004" "360810095001001"
[85] "360810097002003" "360810117001000" "360810117001000" "360810115001002" "360810105004000" "360810115002001"
[91] "360810113001001" "360810113003000" "360810117004000" "360810117002000" "360810113002000" "360810115002002"
[97] "360810123011006" "360810111002003" "360810097003000" "360810111001002"
```

````
```{r}
write.csv(test,'listing_census_code.csv')
```
````

The code generated has 15 digits, while the one census tracts dataset has is 11 digits. After importing into the colab, I used the pandas to convert the number to string and do the slicing to make them fit. Joined the census tract data to the newly imported Airbnb listing data. Then I had a dataset has the Airbnb price and related census factors, which can conduct the first question.

```
listing_census_code=pd.read_csv('data/listing_census_code.csv')
listing_census_code=listing_census_code[["price", "accommodates","bathrooms","beds",
    "number_of_reviews","reviews_per_month","review_scores_rating","review_scores_accuracy","review_scores_cleanliness","review_scores_checkin","review_scores_communication","review_scores_location","review_sco
    "neighbourhood_group_cleansed","property_type","room_type","bed_type","amenities",'census_code']].dropna()
listing_census_code=listing_census_code.apply(lambda x: (float(x[0][1:].strip().replace(",","")), x[1],x[2],x[3],x[4],x[5],x[6],x[7],x[8],x[9],x[10],x[11],x[12],'_'.join(x[13].split()).strip(),'_'.join(x[14].re
```

```
df_Dlisting_census=pd.merge(listing_census_code,df_census, left_on='census_code',right_on='CensusTract' )
df_listing_census.head(1)
```

After selecting the interesting field and drop null value, I got the table with price and census fields. Converting the object to integer or float.

price	TotalPop	Men	Women	Hispanic	White	Black	Native	Asian	Citizen	Income	IncomeErr	IncomePerCap	IncomePerCapErr	Poverty	ChildPoverty	Professional	Service	Office	Construction	Production
189	6887	3379	3508	13.3	58.6	1.3	0.0	22.7	4316	152692.0	29447.0	105958.0	17267.0	3.8	3.1	77.1	6.9	13.5	1.8	0.6

```
for i in df_price_census.columns:
  try:
    df_price_census[i]=df_price_census[i].astype(int)
  except:
    df_price_census[i]=df_price_census[i].astype(float)
```

```
df_price_census.dtypes
```

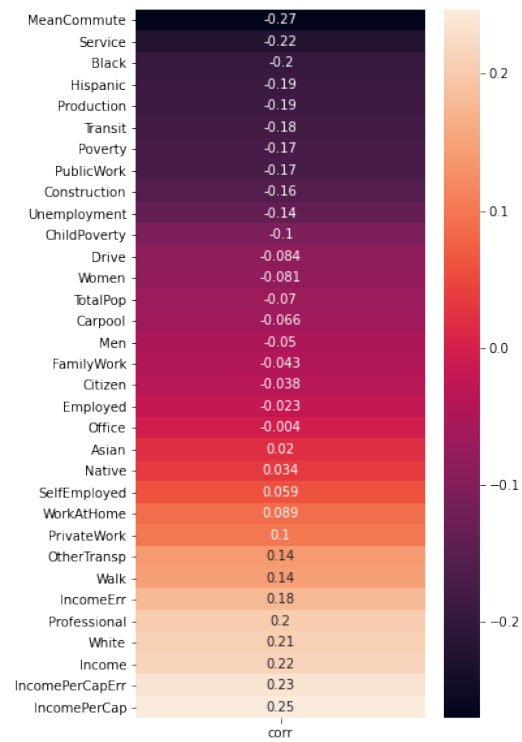
Analysis

Did a heatmap to visualize the correlation coefficient of the price and other variables. I thought the employed will be an important factor affecting the safety to affect the price, but from the heatmap it's not an important factor.

```
price               int64
TotalPop            int64
Men                 int64
Women               int64
Hispanic          float64
White             float64
Black             float64
Native            float64
Asian             float64
Citizen             int64
Income            float64
IncomeErr         float64
IncomePerCap      float64
IncomePerCapErr   float64
Poverty           float64
ChildPoverty      float64
Professional      float64
Service           float64
Office            float64
Construction      float64
Production        float64
Drive             float64
Carpool           float64
Transit           float64
Walk              float64
OtherTransp       float64
WorkAtHome        float64
MeanCommute       float64
Employed            int64
PrivateWork       float64
PublicWork        float64
```

```
df_var=df_price_census.columns[1:].to_frame().rename({0:'variable'},axis=1)
df_corr=df_var.copy()
corr_lst=[]
for i in list(df_price_census.columns[1:]):
    corr, _ = pearsonr(df_price_census['price'], df_price_census[i])
    corr_lst.append(corr)
df_corr['corr']=corr_lst
df_corr=df_corr.loc[:,['corr']].sort_values(by='corr')
```

```
fig, ax = plt.subplots(figsize=(5,10))
ax=sns.heatmap(df_corr,annot=True)
```



Challenge

The most challenging part in first step is to join the dataset. The two dataset doesn't have the same field. I have to generate the census tract code in R and import that file into colab again. Generating the code is time-consuming, I can hardly run it at first; but I decided to narrow down the scope, filter it by certain conditions and got the data.

2. What's the relationship between price and other variables in Airbnb listings?

Data manipulation

Reading the data in spark sql. And then register it as a table

```
sc = SparkContext.getOrCreate()
sqlContext=SQLContext(sc)
df_airbnb=sqlContext.read.option("multiline", "true").option("quote", '"').option("escape", "\\").option("escape", '"').csv('data/listings.csv', header=True)
df_airbnb.registerTempTable('airbnb')
```

```
df_airbnb.show()
```

```
-----+--------------------+-----------+-------+------------+-------------+----------------+------------+---------------+------------+--------------+--------------+
_type|           amenities|square_feet|  price|weekly_price|monthly_price|security_deposit|cleaning_fee|guests_included|extra_people|minimum_nights|maximum_nights|
-----+--------------------+-----------+-------+------------+-------------+----------------+------------+---------------+------------+--------------+--------------+
l Bed|{"Cable TV","Wire...|       null| $60.00|     $385.00|         null|            null|        null|              1|      $10.00|             1|          1125|
l Bed|{TV,"Cable TV",In...|       null|$179.00|   $1,100.00|    $4,000.00|            null|     $125.00|              1|      $30.00|             3|          1125|
l Bed|{Internet,"Wirele...|       null| $49.00|     $350.00|    $1,200.00|            null|        null|              2|      $10.00|             1|          1125|
l Bed|{TV,"Cable TV",In...|       null|$300.00|        null|         null|         $800.00|     $100.00|              4|      $25.00|             7|            90|
l Bed|{"Cable TV",Inter...|       null|$200.00|   $1,200.00|    $3,500.00|            null|        null|              1|       $0.00|             1|          1125|
l Bed|{TV,Internet,"Wir...|       null| $88.00|     $536.00|    $1,800.00|            null|      $40.00|              1|       $5.00|             3|           330|
l Bed|{TV,"Wireless Int...|       null| $95.00|     $550.00|         null|         $500.00|      $30.00|              1|       $0.00|             1|          1125|
l Bed|                  {}|       null| $75.00|     $300.00|         null|            null|        null|              1|       $0.00|             1|          1125|
l Bed|{TV,"Cable TV",In...|       null| $65.00|     $275.00|    $1,000.00|            null|        null|              1|      $10.00|             1|          1125|
l Bed|{TV,"Wireless Int...|       null|$125.00|     $775.00|         null|            null|      $75.00|              1|       $0.00|             3|            21|
```

Selecting the interesting field by Spark sql.

```
] df_airbnb_1=sqlContext.sql("""select price, accommodates,bathrooms,beds,
    number_of_reviews,reviews_per_month,review_scores_rating,review_scores_accuracy,review_scores_cleanliness,review_scores_checkin,review_scores_communication,rev
    neighbourhood_group_cleansed,property_type,room_type,bed_type,amenities,latitude,longitude from airbnb""")
```

```
] df_airbnb_1.show()
```

```
+-------+------------+---------+----+-----------------+-----------------+--------------------+----------------------+-------------------------+----------------------
|  price|accommodates|bathrooms|beds|number_of_reviews|reviews_per_month|review_scores_rating|review_scores_accuracy|review_scores_cleanliness|review_scores_checki
+-------+------------+---------+----+-----------------+-----------------+--------------------+----------------------+-------------------------+----------------------
| $60.00|           2|      1.0|   2|               41|             2.39|                  99|                    10|                       10|                    1
|$179.00|           5|      2.5|   2|                1|             0.56|                 100|                    10|                       10|                    1
| $49.00|           4|      1.0|   1|               36|             5.63|                  97|                    10|                       10|                    1
|$300.00|           4|      3.0|   3|                0|             null|                null|                  null|                     null|                  nul
|$200.00|           2|      1.5|   1|                0|             null|                null|                  null|                     null|                  nul
| $88.00|           4|      1.0|   2|               35|             1.18|                  96|                    10|                       10|                    1
| $95.00|           2|      1.0|   1|                0|             null|                null|                  null|                     null|                  nul
| $75.00|           1|     null|null|                1|             0.38|                 100|                    10|                       10|                    1
| $65.00|           2|      1.0|   1|               12|             0.93|                  92|                    10|                        9|                    1
|$125.00|           3|      1.0|   1|                7|             1.38|                  91|                    10|                       10|                    1
| $55.00|           2|      1.0|   2|               35|             3.54|                  91|                     9|                        9|                    1
| $49.00|           3|      1.0|   1|               55|             1.46|                  89|                     9|                        9|
| $85.00|           2|      1.0|   1|                0|             null|                null|                  null|                     null|                  nul
|$120.00|           4|      1.0|   1|                1|             0.51|                null|                  null|                     null|                  nul
| $50.00|           2|      1.0|   1|               17|             0.39|                  89|                    10|                        9|                    1
| $35.00|           1|      1.0|   1|                0|             null|                null|                  null|                     null|                  nul
```

Droping the NA value. Join the value by underscore for categorical variable prediction later. Remove the dollar sign before the price.

```
[327] df_airbnb_1=df_airbnb_1.dropna()
```

```
[328] df_airbnb_1_rdd=df_airbnb_1.rdd.map(lambda x: (x[0][1:].strip().replace(",",""), x[1],x[2],x[3],x[4],x[5],x[6],x[7],x[8],x[9],x[10],x[11],x[12],'_'.join(x[13].split
```

```
df = spark.createDataFrame(df_airbnb_1_rdd).toDF( "price", "accommodates","bathrooms","beds",
    "number_of_reviews","reviews_per_month","review_scores_rating","review_scores_accuracy","review_scores_cleanliness","review_scores_checkin","review_scores_commur
    "neighbourhood_group_cleansed","property_type","room_type","bed_type","amenities")
df.show(10)
```

```
+------+------------+---------+----+-----------------+-----------------+--------------------+----------------------+-------------------------+---------------------+-
| price|accommodates|bathrooms|beds|number_of_reviews|reviews_per_month|review_scores_rating|review_scores_accuracy|review_scores_cleanliness|review_scores_checkin|r
+------+------------+---------+----+-----------------+-----------------+--------------------+----------------------+-------------------------+---------------------+-
| 60.00|           2|      1.0|   2|               41|             2.39|                  99|                    10|                       10|                   10|
|179.00|           5|      2.5|   2|                1|             0.56|                 100|                    10|                       10|                   10|
| 49.00|           4|      1.0|   1|               36|             5.63|                  97|                    10|                       10|                   10|
| 88.00|           4|      1.0|   2|               35|             1.18|                  96|                    10|                       10|                   10|
| 65.00|           2|      1.0|   1|               12|             0.93|                  92|                    10|                        9|                   10|
|125.00|           3|      1.0|   1|                7|             1.38|                  91|                    10|                       10|                   10|
| 55.00|           2|      1.0|   2|               35|             3.54|                  91|                     9|                        9|                   10|
| 49.00|           3|      1.0|   1|               55|             1.46|                  89|                     9|                        9|                    9|
| 50.00|           2|      1.0|   1|               17|             0.39|                  89|                    10|                        9|                   10|
| 55.00|           2|      1.5|   1|               25|             1.45|                  89|                     9|                        9|                   10|
```

Turning it into pandas dataframe. Get dummies for the categorical variables. Turn the fields into string or integer.

```
df_p.head(1)
```

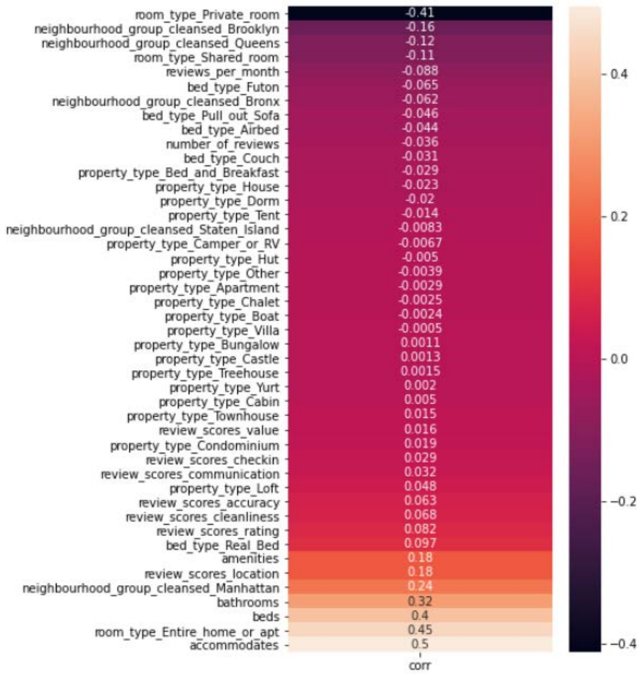
	price	accommodates	bathrooms	beds	number_of_reviews	reviews_per_month	review_scores_rating	review_scores_accuracy	review_scores_cleanliness	review_sco
0	60.00	2	1.0	2	41	2.39	99	10	10	

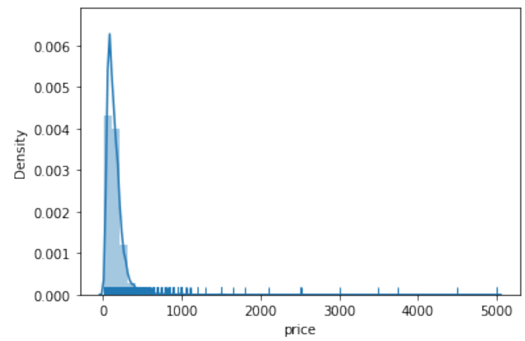
```
df_p[['price','bathrooms','reviews_per_month']]=df_p[['price','bathrooms','reviews_per_month']].astype(float)
df_p[['accommodates','beds','number_of_reviews','review_scores_rating','review_scores_accuracy',"review_scores_cleanliness","review_scores_checkin","review_scores
df_p['price_per_accommodate']=df_p['price']/df_p['accommodates']
var=df_p.columns.to_list()
var=var[-1:]+var[:-1]
df_p=df_p[var]
df_d=pd.get_dummies(data=df_p, columns=["neighbourhood_group_cleansed","property_type","room_type","bed_type"])
df_d.head(10)
```

	price_per_accommodate	price	accommodates	bathrooms	beds	number_of_reviews	reviews_per_month	review_scores_rating	review_scores_accuracy	review_scores_
0	30.000000	60.0	2	1.0	2	41	2.39	99	10	
1	35.800000	179.0	5	2.5	2	1	0.56	100	10	
2	12.250000	49.0	4	1.0	1	36	5.63	97	10	
3	22.000000	88.0	4	1.0	2	35	1.18	96	10	
4	32.500000	65.0	2	1.0	1	12	0.93	92	10	

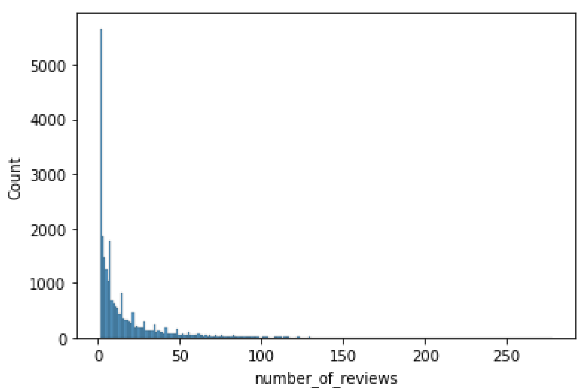
Analysis

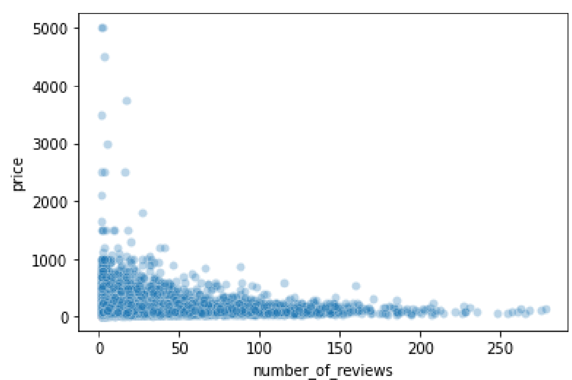
First I did the heatmap of correlation coefficient of the price and other variables. In the correlation coefficient heatmap, we can identify that the room type, neighborhood group, bathrooms, beds, accommodates have great impact on price. the number of reviews it has the lower the price it has while the higher the score it has the higher price it has. Plotting the distribution of price we can see it has many out liers.



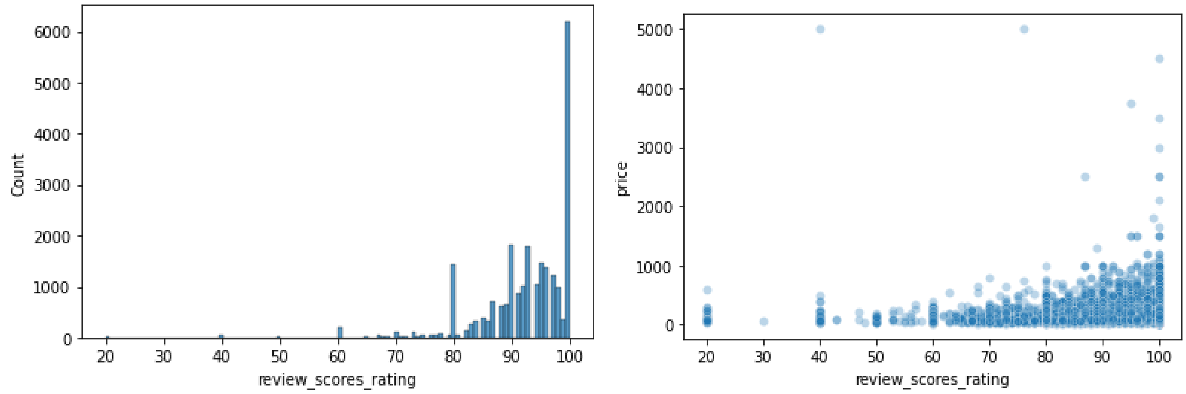


Higher the accommodates, bathrooms, bed has quite positive linear relationship with price. Beds has the extreme outliers that can be removed later. Neighborhoods groups and room types are two important categorical variables for price.

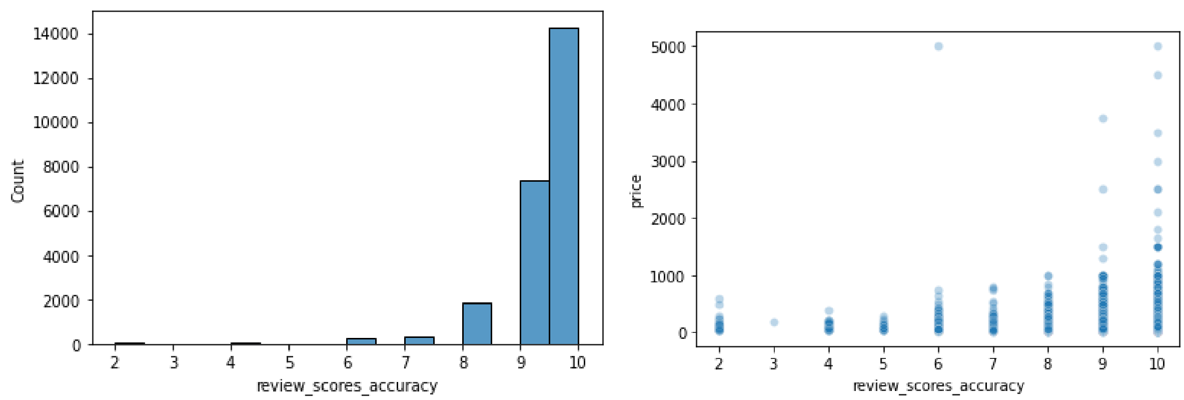




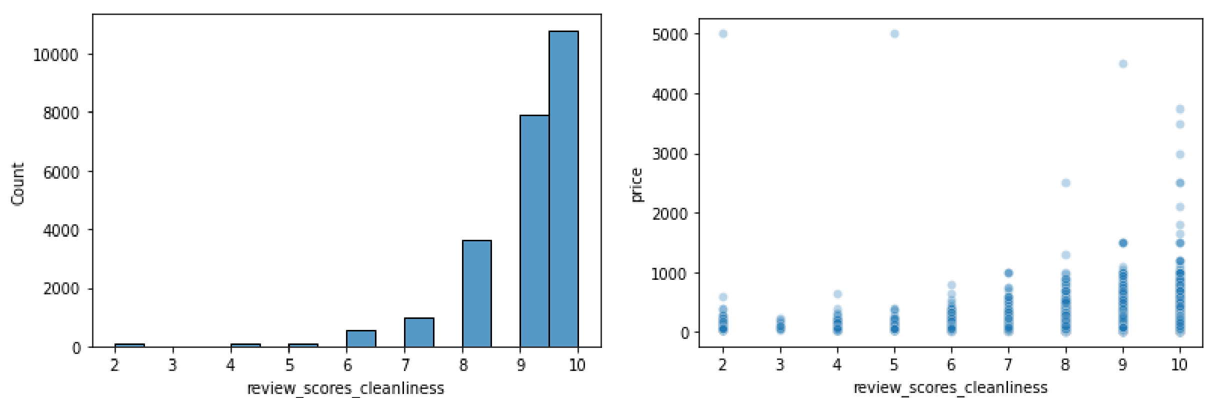
Number of Reviews



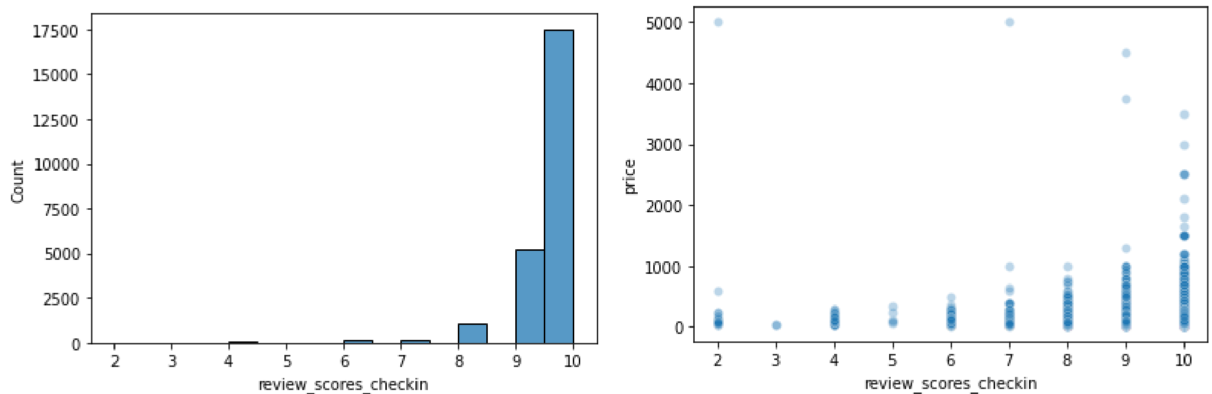
Review Scores Rating



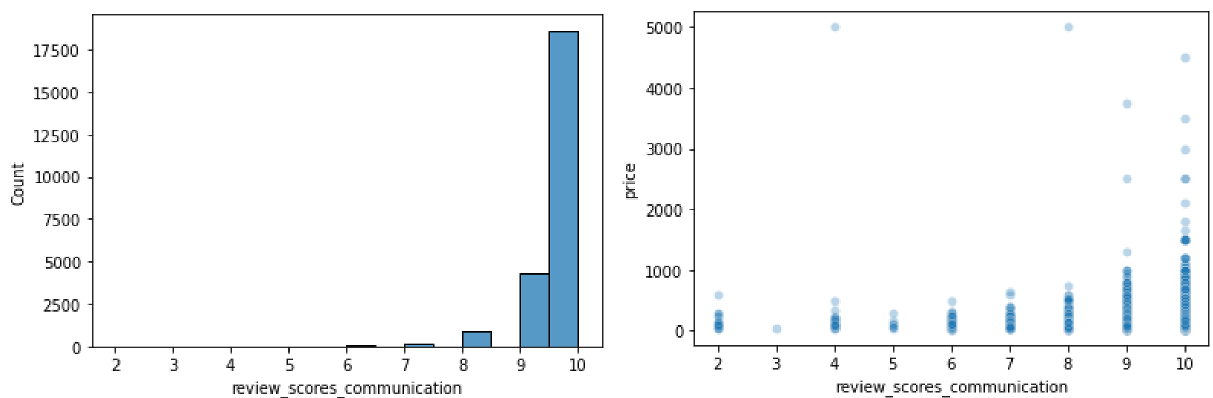
Review Scores Accuracy



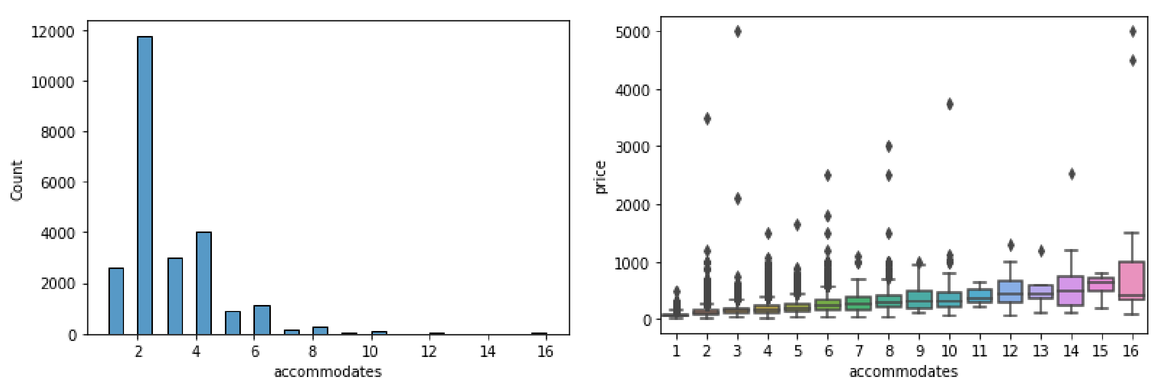
Review Scores Cleanliness

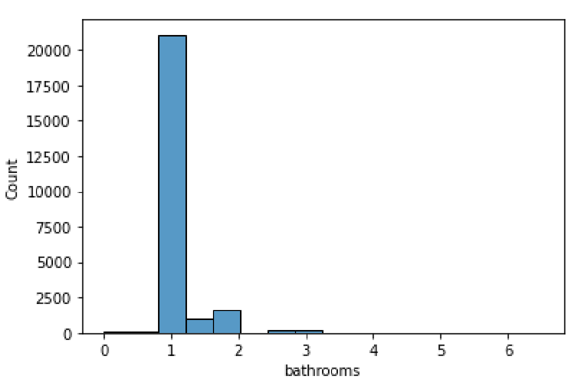


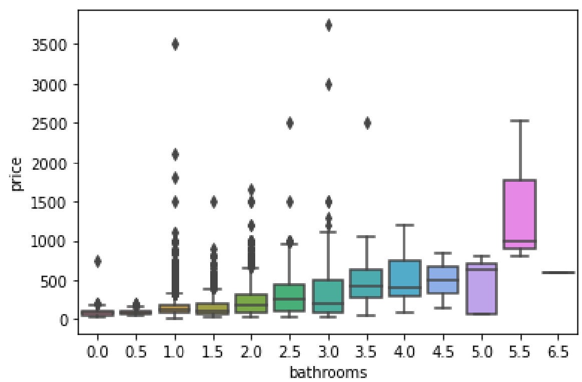
Review Scores Checkin

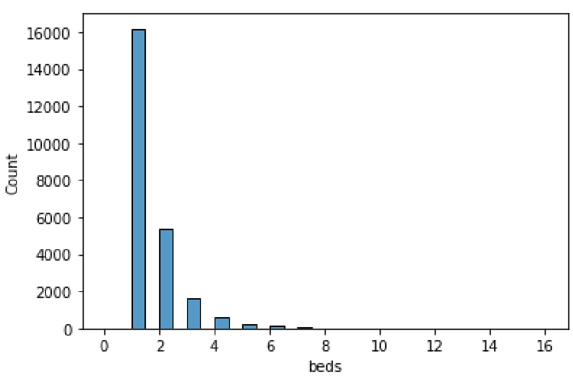


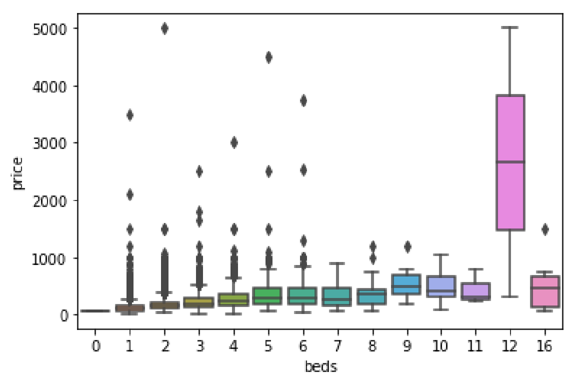
Review Scores Communication

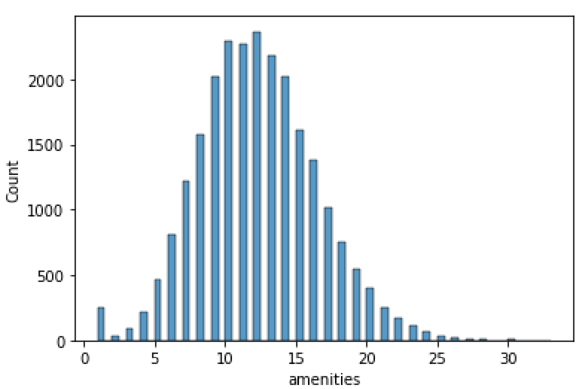


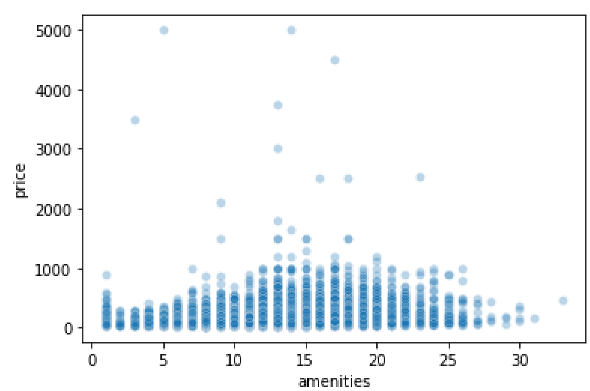

Count
bathrooms

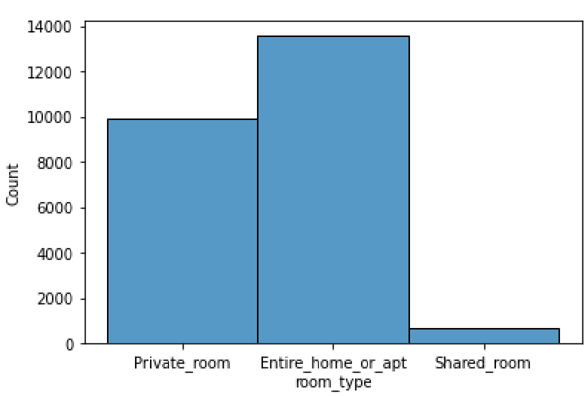

price
bathrooms

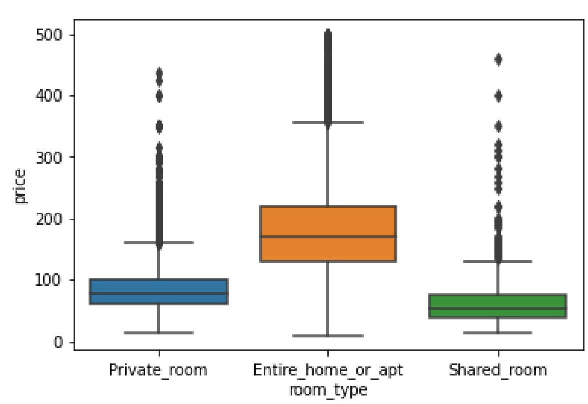

Count
beds

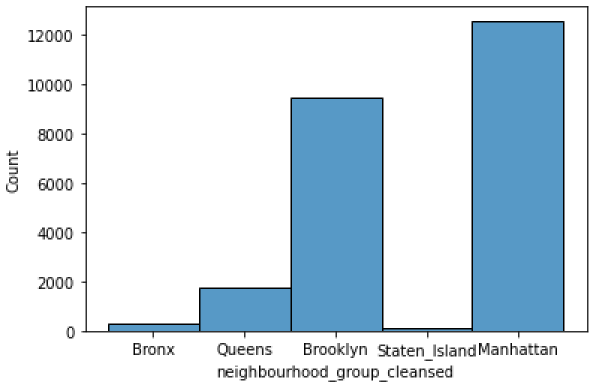

price
beds

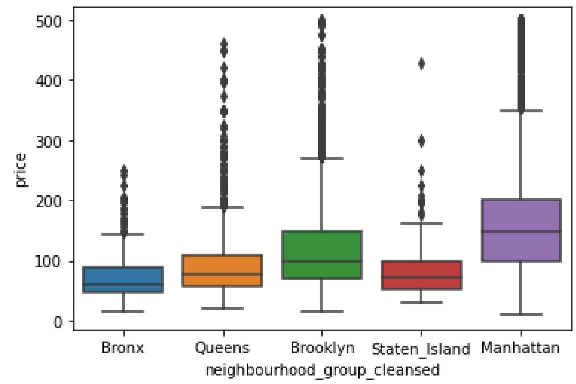

Count
amenities


price
amenities


Count
Private_room
Entire_home_or_apt
Shared_room
room_type


price
Private_room
Entire_home_or_apt
Shared_room
room_type

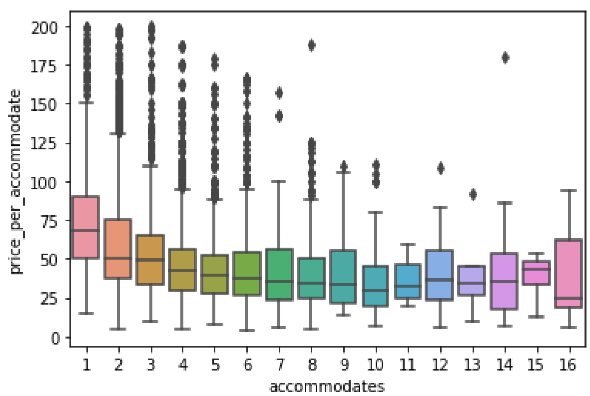


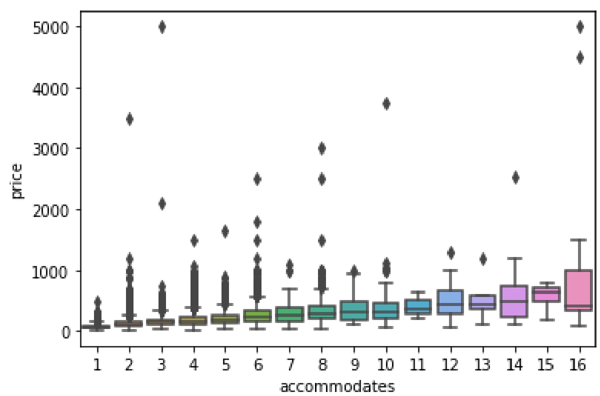


Accommodates, Entire room type, beds, bathrooms, neighborhood in Manhattan, more amenities have the positive correlation coefficient relationship with price; private and shared room type, neighborhood in Brooklyn and Queens, has the negative correlation coefficient relationship with price. Most property type does not affect price much.

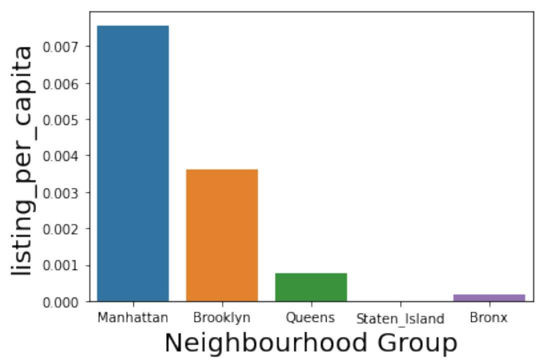
The entire home or apartment takes the highest price. The similar factor behind those variables is that the listing more people is served or designed for, the higher price it has.

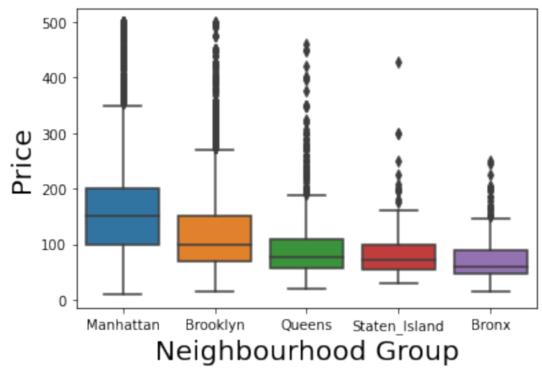
Finding





As the graphs shows, or each house, the more guest it can host, the higher price it is while the lower price per guest need to pay. We might expect that the home for more accommodates is more profitable for hosts and more affordable for guests.





In the previous step, there are higher percentage of listings per capita in Manhattan and Brooklyn, so as that of listing count. Here comes to the question: is the same trend of price per listing and listing number because of more population in Manhattan and Brooklyn? To figure it out, I standardize the listing count via dividing it by population. It remains the same trend: the higher price of listings is, the higher listing per capita is. The conclusion to be draw from this are that people are more willing to become the hosts in the higher price area.