CS 396: Project Proposal

Multi-dimensional Analysis On New York City's Airbnb Data

Jiwon Choi

Proposed Problems

The era of post-COVID is coming, and many people started to plan their trip ahead. In order to provide a meaningful data to both suppliers (Airbnb hosts) and demanders (Airbnb users), this multi-dimensional analysis on Airbnb data is needed. What users may take a look at before making a final decision is not only the price and but also the reviews that other users have written before, and the description provided by the host. In case of hosts, they need to manually search what other hosts' are doing in order to competitively compare themselves to one another. With those needs, some type of more intuitive, concise, and informative data visualization is needed, which can be easily understood and less time-consuming.

In order to tackle this purpose, further analysis on the followings are needed:

- Sentiment analysis per each county in New York City, to suggest users which area to consider.
- Word cloud plots so that users and hosts know the summarized keywords.
- Other additional visualizations that analyze the properties/amenities and price of the place.
- Based on the feature that the host has, give a reasonable price estimate for listings.

Dataset

About Dataset

In order to analyze the proposed problem, the dataset provided by Airbnb will be used. They provided the data here (http://insideairbnb.com/get-the-data.html), and they have a subsection for New York City which will be used in this study. Under New York City, there are multiple dataset that Airbnb has provided. Among 7 dataset that they have, this report plan to make a use of 4 of them: `listings_detail.csv`, `listings_summary.csv`, `reviews_details.csv`, and `reviews_summary.csv`. To briefly explain, the detail version contains more of raw and specific data, while summary data has summarized data only. Note that description of each field can be found here (https://docs.google.com/spreadsheets/d/1iWCNJcSutYqpULSQHlNyGInUvHg2BoUGoNRIGa6Szc4/edit#gid=982310896).

1. Listing Data

```
import pandas as pd
data_path = "./data/"
df = pd.read_csv(data_path + "listings.csv")
pd.set_option('display.max_columns', None)
print("Data: listings_detail.csv")
```

```
## Data: listings_detail.csv
```

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 36923 entries, 0 to 36922
## Data columns (total 74 columns):
##  #   Column                                        Non-Null Count  Dtype
## ---  ------                                        --------------  -----
##  0   id                                            36923 non-null  int64
##  1   listing_url                                   36923 non-null  object
```

```
##  2   scrape_id                                  36923 non-null  int64
##  3   last_scraped                               36923 non-null  object
##  4   name                                       36910 non-null  object
##  5   description                                35710 non-null  object
##  6   neighborhood_overview                      22510 non-null  object
##  7   picture_url                                36923 non-null  object
##  8   host_id                                    36923 non-null  int64
##  9   host_url                                   36923 non-null  object
##  10  host_name                                  36812 non-null  object
##  11  host_since                                 36812 non-null  object
##  12  host_location                              36714 non-null  object
##  13  host_about                                 21636 non-null  object
##  14  host_response_time                         21180 non-null  object
##  15  host_response_rate                         21180 non-null  object
##  16  host_acceptance_rate                       21821 non-null  object
##  17  host_is_superhost                          36812 non-null  object
##  18  host_thumbnail_url                         36812 non-null  object
##  19  host_picture_url                           36812 non-null  object
##  20  host_neighbourhood                         29726 non-null  object
##  21  host_listings_count                        36812 non-null  float64
##  22  host_total_listings_count                  36812 non-null  float64
##  23  host_verifications                         36923 non-null  object
##  24  host_has_profile_pic                       36812 non-null  object
##  25  host_identity_verified                     36812 non-null  object
##  26  neighbourhood                              22511 non-null  object
##  27  neighbourhood_cleansed                     36923 non-null  object
##  28  neighbourhood_group_cleansed               36923 non-null  object
##  29  latitude                                   36923 non-null  float64
##  30  longitude                                  36923 non-null  float64
##  31  property_type                              36923 non-null  object
##  32  room_type                                  36923 non-null  object
##  33  accommodates                               36923 non-null  int64
##  34  bathrooms                                  0 non-null      float64
##  35  bathrooms_text                             36818 non-null  object
##  36  bedrooms                                   32987 non-null  float64
##  37  beds                                       36312 non-null  float64
##  38  amenities                                  36923 non-null  object
##  39  price                                      36923 non-null  object
##  40  minimum_nights                             36923 non-null  int64
##  41  maximum_nights                             36923 non-null  int64
##  42  minimum_minimum_nights                     36906 non-null  float64
##  43  maximum_minimum_nights                     36906 non-null  float64
##  44  minimum_maximum_nights                     36906 non-null  float64
##  45  maximum_maximum_nights                     36906 non-null  float64
##  46  minimum_nights_avg_ntm                     36906 non-null  float64
##  47  maximum_nights_avg_ntm                     36906 non-null  float64
##  48  calendar_updated                           0 non-null      float64
##  49  has_availability                           36923 non-null  object
##  50  availability_30                            36923 non-null  int64
##  51  availability_60                            36923 non-null  int64
##  52  availability_90                            36923 non-null  int64
##  53  availability_365                           36923 non-null  int64
##  54  calendar_last_scraped                      36923 non-null  object
##  55  number_of_reviews                          36923 non-null  int64
##  56  number_of_reviews_ltm                      36923 non-null  int64
##  57  number_of_reviews_l30d                     36923 non-null  int64
##  58  first_review                               27627 non-null  object
##  59  last_review                                27627 non-null  object
##  60  review_scores_rating                       27627 non-null  float64
##  61  review_scores_accuracy                     26998 non-null  float64
##  62  review_scores_cleanliness                  27009 non-null  float64
##  63  review_scores_checkin                      26991 non-null  float64
##  64  review_scores_communication                27002 non-null  float64
##  65  review_scores_location                     26987 non-null  float64
##  66  review_scores_value                        26987 non-null  float64
##  67  license                                    0 non-null      float64
##  68  instant_bookable                           36923 non-null  object
##  69  calculated_host_listings_count             36923 non-null  int64
##  70  calculated_host_listings_count_entire_homes  36923 non-null  int64
##  71  calculated_host_listings_count_private_rooms  36923 non-null  int64
##  72  calculated_host_listings_count_shared_rooms  36923 non-null  int64
##  73  reviews_per_month                          27627 non-null  float64
## dtypes: float64(23), int64(17), object(34)
## memory usage: 20.8+ MB
```

```
df = pd.read_csv(data_path + "listings_summary.csv")
print("Data: listings_summary.csv")
```

```
## Data: listings_summary.csv
```

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 36923 entries, 0 to 36922
## Data columns (total 18 columns):
##  #   Column                          Non-Null Count  Dtype
## ---  ------                          --------------  -----
##  0   id                              36923 non-null  int64
##  1   name                            36910 non-null  object
##  2   host_id                         36923 non-null  int64
##  3   host_name                       36812 non-null  object
##  4   neighbourhood_group             36923 non-null  object
##  5   neighbourhood                   36923 non-null  object
##  6   latitude                        36923 non-null  float64
##  7   longitude                       36923 non-null  float64
##  8   room_type                       36923 non-null  object
##  9   price                           36923 non-null  int64
##  10  minimum_nights                  36923 non-null  int64
##  11  number_of_reviews               36923 non-null  int64
##  12  last_review                     27627 non-null  object
##  13  reviews_per_month               27627 non-null  float64
##  14  calculated_host_listings_count  36923 non-null  int64
##  15  availability_365                36923 non-null  int64
##  16  number_of_reviews_ltm           36923 non-null  int64
##  17  license                         0 non-null      float64
## dtypes: float64(4), int64(8), object(6)
## memory usage: 5.1+ MB
```

2. Review Data

```
import pandas as pd
data_path = "./data/"
df = pd.read_csv(data_path + "reviews.csv")
pd.set_option('display.max_columns', None)
print("Data: reviews_detail.csv")
```

```
## Data: reviews_detail.csv
```

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 848725 entries, 0 to 848724
## Data columns (total 6 columns):
##  #   Column         Non-Null Count   Dtype
## ---  ------         --------------   -----
##  0   listing_id     848725 non-null  int64
##  1   id             848725 non-null  int64
##  2   date           848725 non-null  object
##  3   reviewer_id    848725 non-null  int64
##  4   reviewer_name  848719 non-null  object
##  5   comments       847901 non-null  object
## dtypes: int64(3), object(3)
## memory usage: 38.9+ MB
```

```
df = pd.read_csv(data_path + "reviews_summary.csv")
print("Data: reviews_summary.csv")
```

```
## Data: reviews_summary.csv
```

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 848725 entries, 0 to 848724
## Data columns (total 2 columns):
##  #   Column      Non-Null Count   Dtype
## ---  ------      --------------   -----
##  0   listing_id  848725 non-null  int64
##  1   date        848725 non-null  object
## dtypes: int64(1), object(1)
## memory usage: 13.0+ MB
```

As seen in the `info()`, studying the full dataset will be a viable option. The `Non-Null Count` field has a minimum of 20,000+ for fields that are necessary to study. Even considering data cleaning, it will not go less than 20,000 samples since the fields with `NaN` values are mostly a text data, which has a less probability of duplicate -i.e. host description field.

Data Cleaning, Management, and EDA

1. Data Cleaning & Management

- Drop unavailable data like `bathrooms`, `calendar_updated`, and `license`, to perform `dropna()` better.
- Provided county description is already cleaned version in the column `neighbourhood_cleansed`. There is no duplicate values, or different format with a same meaning. Same for the `room_type` field.
- In `bathrooms_text` field, there are 2 samples which indicate that they have 10 and 10.5 baths. This seems to be an outlier and unrealistic, so manual check was done. It turned out that they rent the entire townhouse and apartment which sounds about to be right. They also indicated a similar number of bedrooms.
- The `price` entry has 36,923 data in total, but 36 samples have a price of $0.0. This seems to be faulty, so dropped them.
- The field of reviews: `review_scores_rating`, `review_scores_accuracy`, `review_scores_cleanliness`, `review_scores_checkin`, `review_scores_communication`, `review_scores_location`, and `review_scores_value` got some blank field, so dropped them. They remain to be over 25,000+ samples even excluding them.
- In the `description` field in `listings.csv` and the `comments` field in `reviews.csv`, if input includes newline it shows `<br>`, so we need to replace this into `""`.
- In `reviews.csv`, `comments` field where it has reviews are not in English. In this study, the text analysis is limited to English, and below is the refining process. This gives a result of 769,559, which is an accpetable amount of sample data.

```
from polyglot.detect import Detector
import pandas as pd
data_path = "./data/"
df = pd.read_csv(data_path + "reviews.csv")
comments = df["comments"]
eng_count = 0
for c in comments:
  if type(c) is float: continue
  if ("<br>" in c): c.replace("<br>", "")
  try:
    c.encode('utf-8')
    lang = Detector(c, quiet=True)
    if (lang.language.name == "English"): eng_count += 1
  except: continue
print (eng_count)
```

2. EDA

A. Non-Graphical EDA

Non-graphical EDA provides a basic information on review rates.

```
import pandas as pd
data_path = "./data/"
df = pd.read_csv(data_path + "listings.csv")
desc = df["review_scores_rating"].describe()
desc
```

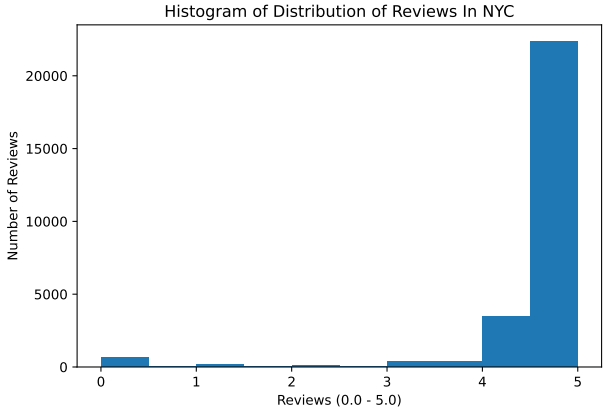
```
## count    27627.000000
## mean         4.578315
## std          0.854467
## min          0.000000
## 25%          4.570000
## 50%          4.820000
## 75%          5.000000
## max          5.000000
## Name: review_scores_rating, dtype: float64
```

B. Graphical EDA

Graphical EDA includes a histogram of reviews, a bar plot of average reviews per county, and a scatter plot that has reviews and price information.

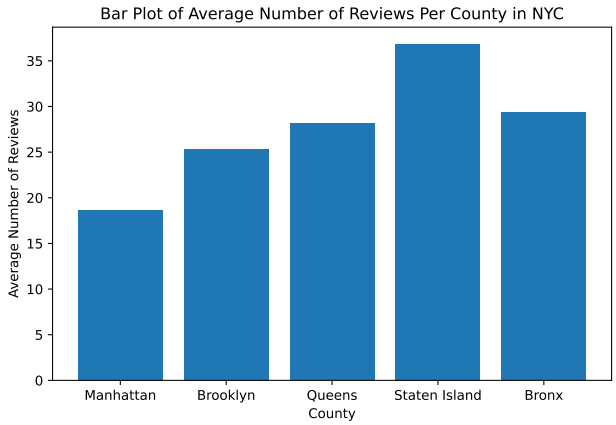
```
import pandas as pd
import matplotlib.pyplot as plt
data_path = "./data/"
df = pd.read_csv(data_path + "listings.csv")

plt.hist(df["review_scores_rating"])
```

```
plt.title("Histogram of Distribution of Reviews In NYC")
plt.xlabel("Reviews (0.0 - 5.0)")
plt.ylabel("Number of Reviews")
plt.tight_layout()
plt.show()
```



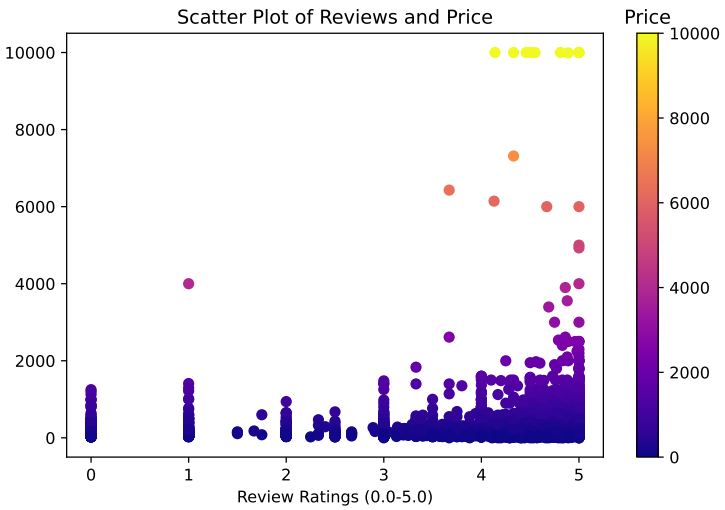

```
county = df["neighbourhood_group_cleansed"].unique()
avg_review = []
for c in county:
  avg = ((df[df["neighbourhood_group_cleansed"] == c])["number_of_reviews"]).mean()
  avg_review.append(avg)
plt.bar(county, avg_review)
```

```
plt.title("Bar Plot of Average Number of Reviews Per County in NYC")
plt.xlabel("County")
plt.ylabel("Average Number of Reviews")
plt.tight_layout()
plt.show()
```




```
import pandas as pd
import matplotlib.pyplot as plt
data_path = "./data/"
df = pd.read_csv(data_path + "listings.csv")
df['price'] = df["price"].str.replace('$', '')
```

```
df['price'] = df["price"].str.replace(',', '')
df["price"] = pd.to_numeric(df["price"])
scatter_df = df[['price', 'review_scores_rating']]
drop_na = scatter_df.dropna()
plt.scatter(drop_na["review_scores_rating"], drop_na["price"], c=drop_na['price'], cmap='plasma')
plt.title("Scatter Plot of Reviews and Price")
plt.xlabel("Review Ratings (0.0-5.0)")
plt.ylabel("Price (USD)")
clb = plt.colorbar()
clb.ax.set_title('Price')
plt.show()
```



Interesting Findings

- Non-grpahical EDA shows that people tend to satisify with their Airbnb rent.
- The visual histogram of distribution of ratings shows that people tend to either give high rates to their housing, or just poor rates. People barely give a scores of 2-3.

- People wrote many reviews after visiting the places in Staten Island, as seen in bar plot.
- According to the scatter plot, even if the prices are high, people give higher reviews. This also indicates that they got satisfactory housing/amenities.

Dataset Validation

Review dataset was validated in the former section. Even without non-English reviews, there are 769,559 samples available for sentiment and word cloud analysis. In case of listing information dataset, their column subset are greater than 20,000 samples as we briefly discussed in EDA. Note that the data that this study will be mainly focused on is reviews, price, and amenities.

Anticipated Method & Outcome

Problem Study Plan

- General visualization will be further added to aid general understandings of the lists and reviews. Except for general reviews, there are other review data available, such as check-in experience, how realistic it is, cleanliness, communication, location, and values, so these will also be used to plot such start plots.
- Amenities list will later be converted so that it may reflect the word cloud plot. Host information and review text will be used to plot the word cloud as well.
- Sentiment analysis will be done based on customers' reviews.
- Price data will be further analyzed because it seems to have an outlier right now (values higher than $1,000), so manual validation is further needed to examine them.
- Machine learning model will be built based on sentiment analysis, feature/amenities, county, and other viable options available in the dataset, to predict a right price for a room for new hosts.

Potential Difficulties

- As seen in the price-integrated scatter plot, the price data had to be converted since it is in text format. In case of `bathrooms` field, this is also in text format like "shared bathroom", "private bathroom", "2 baths", and etc, so converting them in a consistent manner is needed.
- In case of a machine learning model, the researcher does not have enough experience on predicting a certain number based on feature, since the main research area is machine learning with imaging data (computer vision).

Anticipated Outcomes

The final paper will include all EDA visualizations that may aid customers and hosts of Airbnb. The price estimator machine learing model with a higher accuracy is also expected.