Analysis of Factors Affecting Airbnbs' Price and Popularity in New York

Liuyixin Shao

Section AC

Summary of research questions:

1, Is there any correlation between the average price of Airbnbs by neighborhood and the average property value of housing in those neighborhoods?

> I found that the overall average Airbnb price in a neighborhood is proportional to the average property sales of that neighborhood. The more expensive the neighborhood, the higher the average Airbnb price in the neighborhood will be. However, the generally Airbnb prices of neighborhoods in Staten Island and on the southern coast of Queens are set relatively high compared to their actual property worth.

2, Where are the most popular Airbnbs in New York and their neighborhoods?

> I found that the most popular Airbnbs are located in three main areas-the Lower Manhattan which is closed to Washington Square Park, the northwest corner of Queens which is near the LaGuardia Airport, and the lower central Queens area around the John F. Kennedy International Airport. Popular Airbnbs outside of these areas are very scattered.

3, How can I train a proper model to predict the price for an Airbnb in New York City by its location (neighborhood group), listing space type, required minimum night, reviews per month, and average property value?

> I finally came up with an XGBRegressor model to predict the price and an XGBClassifier model to predict the range of the price. The XGBRegressor has n_estimators (the number of trees in the forest of the model) equals 1500, learning_rate (the step size at each iteration) equals 0.0013, and max_depth equals 5. For my testing dataset, the regressor model's mean squared error is 82494.84, the mean absolute error is 57.33, and the testing score is 0.0412. The XGBClassifier has n_estimators equals 1000, learning_rate equals 0.005, and max_depth equals 6. The testing accuracy according to the testing dataset of the classifier is 0.4856.

4, What are the most popular words used by hosts on Airbnbs to name their housings?

I found that the top 10 most popular words (frequency from large to small) used by hosts in New York to describe their housings are: “room”, “bedroom”, “private”, “apartment”, “cozy”, “brooklyn”, “apt”, “east”, “spacious”, and “studio”.

Motivation:

With the summer program halfway over, my vacation trip is on the agenda. This summer, eight of my friends and I are planning a vacation to Florida. Booking accommodation on Airbnb became our first choice as opposed to booking hotels, as it was more affordable. During the booking process, we did a lot of discussions. I found the surroundings, the number of positive reviews, prices, and even the availability of supermarkets nearby were all taken into our consideration. Of course, the location, type, worth values, and length of tenancy are also the main factors that cause the difference in the price of each Airbnb. After a few days of deciding, finally, we booked a very nice house on the app near Disney.

I think that just like us, other Airbnb clients consider many factors when looking for the right place to stay during their trips. Hosts also price their properties for a variety of reasons. This analysis of factors affecting Airbnbs’ price and popularity takes Airbnbs in New York City as an example (because New York has the most Airbnbs in the US), but its results have universal applicability. It will allow customers to better understand the basis of each Airbnb’s pricing, and thus better choose the best property for them. Similarly, this analysis will also allow hosts to make better pricing, so they can get more customers and make their homes more popular.

Datasets:

1, New York City Airbnb Open Data

This is the main dataset for my analysis. I found it on Kaggle. This public dataset is part of Airbnb, which includes all needed information to find out more about hosts, geographical availability, and necessary metrics to make predictions and conclude.

This dataset contains 16 columns and 48895 rows, documenting the 5 neighborhood groups and 221 neighborhoods in New York City for Airbnbs in 2019.

- The dataset on Kaggle (click on the download button to download the CSV file for the data): https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data
- The original source can be found on: http://insideairbnb.com/

2, Summaries of Neighborhood and Citywide Annual Sales sorted by borough and neighborhood

This is a group of databases that I will use for the analysis. I found it on the website of the NYC Department of Finance. They are the summarized version of the minimum, maximum, mean, and median prices of property sales in each neighborhood. Here, I will mainly focus on the 5 datasets of Neighborhood Sales Data from 2005 through 2021.

- The datasets on the NYC Department of Finance (here, I downloaded the MS Excel version): https://www1.nyc.gov/site/finance/taxes/property-annualized-sales-update.page
    - Manhattan: https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww1.nyc.gov%2Fassets%2Ffinance%2Fdownloads%2Fpdf%2Frolling_sales%2Fneighborhood_sales%2F2021%2F2021_manhattan_sales_prices.xlsx&wdOrigin=BROWSELINK
    - Bronx:

      https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww1.nyc.gov%2Fassets%2Ffinance%2Fdownloads%2Fpdf%2Frolling_sales%2Fneighborhood_sales%2F2021%2F2021_bronx_sales_prices.xlsx&wdOrigin=BROWSELINK
    - Brooklyn:

      https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww1.nyc.gov%2Fassets%2Ffinance%2Fdownloads%2Fpdf%2Frolling_sales%2Fneighborhood_sales%2F2021%2F2021_brooklyn_sales_prices.xlsx&wdOrigin=BROWSELINK
    - Queens:

      https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww1.nyc.gov%2Fassets%2Ffinance%2Fdownloads%2Fpdf%2Frolling_sales%2Fneighborhood_sales%2F2021%2F2021_queens_sales_prices.xlsx&wdOrigin=BROWSELINK
    - Staten Island:

      https://view.officeapps.live.com/op/view.aspx?src=https%3A%2F%2Fwww1.nyc.gov%2Fassets%2Ffinance%2Fdownloads%2Fpdf%2Frolling_sales%2Fneighborhood_sales%2F2021%2F2021_staten_island_sales_prices.xlsx&wdOrigin=BROWSELINK

3, nyc-ny-neighborhood.geojson

This is the dataset for graphing the neighborhood map of New York City. I found it on GitHub. The author of the dataset is yocontra. The dataset contains the geometry information of the boundaries of neighborhood tabulation areas in New York City.

This dataset contains 4 columns and 282 rows, documenting the geometrical shape of the 282 neighborhoods in New York City.

- The dataset on GitHub (click download zip to download the geojson file for the data):

https://gist.github.com/yocontra/43adf8ac0c75bec02c0d7bccd8269075

Method:

Data cleaning:

Before the formal analysis begins, I need to first clean my datasets, merge or concatenate several datasets together, and do some initial computations:

- Mutate each NYC Department of Finance dataset from excel format to CSV format.
- Constructing float_to_int and delete_comma methods to change the data we need to the correct format.
- Combine the rows in each dataset into one dataset with sales data from the 5 neighborhood groups using pd.concat().
- Data cleaning, drop the NaN.
- Construct the pandas DataFrame with the average sales prices in the neighborhood each Airbnb locates (which I call nyc_df in the later step description):
    - Calculate the average sales price for each neighborhood (ignore building type) in the combined dataset.
        - Calculate the total sales price of each building type, and save it in a new column
        - Group the combined dataset by neighborhoods and calculate the total sales price for each neighborhood, save the result in a dataset.
        - Group the combined dataset by neighborhoods and calculate the total number of sales for each neighborhood, save the result in a dataset.
        - Combine the two datasets from groupby() together and calculate the average sales price for each neighborhood (ignore building type), save the result as a new column.
    - Search any difference in neighborhood names between the combined dataset and the New York City Airbnb Open Data.
    - (If there is,) replace the name in the combined dataset with the name in the New York City Airbnb Open Data, if both names represent the same neighborhood.
    - Merge the one dataset with sales data and the New York City Airbnb Open Data together by neighborhood and select useful columns.
- Construct the GeoDataframe with Airbnb coordinates and the average sales prices in the neighborhood each Airbnb locates (which I call geo_nyc_df in the later step description):

    - Change the latitude and longitude from the pandas DataFrame into geometry points.
    - Construct the GeoDataFrame with geometry equaling to point coordinates.
- Construct the GeoDataFrame groups by neighborhoods and contains the mean Airbnb price, the mean property sale price, and the relative price in each neighborhood in New York City (which I call geo_group_df in the later step description):
    - Calculate the mean Airbnb price and the mean property sale price by neighborhood.
    - Calculate the relative price by dividing the average Airbnb price by the average property sale price, then multiplying by 100000.
    - Merge the grouped dataset with the geojson dataset with geometrical shapes of neighborhoods in New York City.
    - Construct the GeoDataFrame with geometry equaling to neighborhoods' shapes.
- Construct the data frame used for building the classifier model:
    - Split the filtered data frame into 6 ranges: '0-50', '50-100', '100-150', '150-200', '200-300', and '300+'.
    - Replace the price column in the filtered data frame with the price range column that I split in the previous step.

Research question 1:

Is there any correlation between the average price of Airbnbs by neighborhood and the average property value of housing in those neighborhoods?

To answer the question, I ended up:

- Use geopandas to generate 1 plot with 2 subplots showing the distribution of average Airbnb price in New York City by neighborhood and the distribution of average property sale price in New York City by neighborhood:
    - Create a subplot with nrow equals to 2.
    - For the distribution of average Airbnb prices in New York City by neighborhood:
        - Use the geojson data frame to plot the entire New York City as the background.
        - Use the geo_group_df to plot the average Airbnb price in each neighborhood.
        - Name the plot Distribution of Average Airbnb Price in NYC by Neighborhood.
    - For the distribution of average property sale price in New York City by neighborhood:
        - Use the geojson data frame to plot the entire New York City as the background.

        - Use the geo_group_df to plot the average property sale price in each neighborhood.
        - Name the plot Distribution of Average Property Sale Price in NYC by Neighborhood.
    - Save the plot as ./q1_mean.png.
- Use geopandas to generate 1 plot showing the distribution of the relative price in each neighborhood in New York City:
    - Create a subplot:
        - Use the geojson dataframe to plot the entire New York City as the background.
        - Use the geo_group_df to plot the calculated relative price in each neighborhood.
        - Name the plot Relative Average Airbnb Price to Property Sales Price in NYC by neighborhood (times 100000).
    - Save the plot as ./q1_relative.png.

Research question 2:

Where are the most popular Airbnbs in New York and their neighborhoods?

To answer the question, I ended up:

- Use sort_values to sort the geo_nyc_df by the number of reviews.
- Use nlargest to slice the sorted data and get the Airbnbs with the top 20 reviews.
- Use sjoin to merge the sliced dataset and the geojson data frame together and leave the intersections of their geometry as a new geoDataFrame called pop_nb
- Use geopandas to generate 1 plot showing the distribution of the Airbnbs with the top 20 total reviews:
    - Create a subplot:
        - Use the geojson data frame to plot the entire New York City as the background.
        - Use the pop_nb to plot the intersection neighborhoods in purple.
        - Use the sliced top 20 reviews dataset to plot the coordinate points as the exact location of the top 20 most popular Airbnbs.
        - Name the plot Distribution of the Top 20 Most Popular Airbnbs.
    - Save the plot as ./q2_pop.png.
- Use plotly to generate a table showing the name, the neighborhood group, the neighborhood, the room type, the Airbnb price, and the number of reviews of the top 20 Airbnbs:
    - Use the go.Figure and go.Table method:

        - Give proper label names to form the table in the header as ‘Name’, ‘Neighbourhood Group’, ‘Neighbourhood’, ‘room type’, ‘Airbnb price’, and ‘Number of Reviews’.
        - Give corresponding columns to represent the data in cells.
        - Name the table The 20 Most Popular Airbnbs Stats.

Research question 3:

How can I train a proper model to predict the price for an Airbnb in New York City by its location (neighborhood group), listing space type, required minimum night, reviews per month, and average property value?

To answer the question, I ended up with one regression model and one classification model:

- Import all necessary machine learning libraries.

For the regression model:

- Clean and separate the data for machine learning:
    - Select the columns neighborhood_group, room_type, minimum_nights, reviews_per_month, the average sale price in this neighborhood, and the price of Airbnb.
    - Remove rows with missing data.
    - Use one-hot encoding to encode neighborhood_group and room_type.
    - Separate the data into features and labels.
        - Features: neighborhood_group, room_type, minimum_nights, reviews_per_month, and the average sale price in this neighborhood
        - Labels: price
    - Split the data into training (70%), validation (15%), and testing (15%) datasets.
    - Set the random state into 1 (we have to make the training and validation datasets fixed in order to compare and pick the best model later).
- Build the DecisionTreeRegressor model and try different max_depths to get the best max_depth for the model:
    - Use a for loop to test different max_depth on the DecisionTreeRegressor model.
    - Fit the model with training datasets.
    - Print the mean squared error, the mean absolute error, and the testing score of the model with different maximum depths using the validation datasets.
- Build the RandomForestRegressor model and try different n_estimators to get the best n_estimators for the model:
    - Set the maximum depth of the model to be 5.
    - Use a for loop to test different n_estimators on the RandomForestRegressor model.

    - Fit the model with training datasets.
    - Print the mean squared error, the mean absolute error, and the testing score of the model with different numbers of trees to be used in the forest using the validation datasets.
- Build the XGBRegressor model and try different learning_rate to get the best learning_rate for the model:
    - Set the maximum depth of the model to be 5 and the n_estimators of the model to be 1,500.
    - Use a for loop to test different learning_rate on the XGBRegressor model.
    - Fit the model with training datasets.
    - Print the mean squared error, the mean absolute error, and the testing score of the model with different tuning parameters using the validation datasets.
- Use plotly to generate a chart with 3 subplots as tables showing the stats of each generated machine learning model and their respective changing hyper-parameters:
    - Create a fig with 3 tables as subplots:
        - Use go.Table to create the Validate Max Depth in DecisionTreeRegressor table.
            - Give proper label names to form the table in the header as “Max Depth”, “Mean Square Error”, “Mean abs Error”, and “Testing Score”.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected max_depth in light green.
        - Use go.Table to create the Validate N Estimators in RandomForestRegressor table:
            - Give proper label names to form the table in the header as “N Estimators”, “Mean Square Error”, “Mean abs Error”, and “Testing Score”.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected n_estimators in light green.
        - Use go.Table to create the Validate Learning Rate in XGBRegressor table:
            - Give proper label names to form the table in the header as “Learning Rate”, “Mean Square Error”, “Mean abs Error”, and “Testing Score”.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected learning_rate in light green.
    - Name the chart Table Comparing the 3 Regressor Models.
- Build the final XGBRegressor model using the chosen value in hyper-parameters:
    - Set the maximum depth of the model to be 5, the n_estimators of the model to be 1,500, and the learning rate to be 0.0013.
    - Fit the model with training datasets.

    - Print the mean squared error, the mean absolute error, and the testing score of the testing datasets.
    - Construct a pandas DataFrame with the actual and predict data in the testing dataset.
- Filter the top 30 rows in the actual and predict dataset.
- Use plotly to generate a chart with 2 subplots as tables showing the stats of the final chosen XGBRegressor model and its actual and prediction performance:
    - Create a fig with 2 tables as subplots:
        - Use go.Table to create the Testing Statistics of the XGBRegressor Model table.
            - Give proper label names to form the table in the header as “Mean Square Error”, “Mean abs Error”, and “Testing Score”.
            - Give corresponding calculated results to the label names in cells.
        - Use go.Table to create the Actual and Predicted Airbnb Price Comparison table (showing the top 30 rows in the testing dataset):
            - Give proper label names to form the table in the header as “Actual Values”, and “Predicted Values”.
            - Give corresponding columns to represent the data in cells.
    - Name the chart Performance of the Selected XGBRegressor Model.

For the classification model:

- Clean and separate the data for machine learning:
    - Use c_filter_data() function to obtain the proper dataset that I used for building the classifier.
    - Use one-hot encoding to encode neighborhood_group and room_type.
    - Separate the data into features and labels.
        - Features: neighborhood_group, room_type, minimum_nights, reviews_per_month, and the average sale price in this neighborhood
        - Labels: price_range
    - Split the data into training (70%), validation (15%), and testing (15%) datasets.
    - Set the random state into 1 (we have to make the training and validation datasets fixed in order to compare and pick the best model later).
- Build the DecisionTreeClassifier model and try different max_depths to get the best max_depth for the model:
    - Use a for loop to test different max_depth on the DecisionTreeClassifier model.
    - Fit the model with training datasets.
    - Print the training and testing accuracy of the model with different maximum depths using the validation datasets.
- Build the RandomForestClassifier model and try different n_estimators to get the best n_estimators for the model:

    - Set the maximum depth of the model to be 6.
    - Use a for loop to test different n_estimators on the RandomForestClassifier model.
    - Fit the model with training datasets.
    - Print the training and testing accuracy of the model with different maximum depths using the validation datasets.
- Build the XGBClassifier model and try different learning_rate to get the best learning_rate for the model:
    - Set the maximum depth of the model to be 6 and the n_estimators of the model to be 1000.
    - Use a for loop to test different learning_rate on the XGBClassifier model.
    - Fit the model with training datasets.
    - Print the training and testing accuracy of the model with different maximum depths using the validation datasets.
- Use plotly to generate a chart with 3 subplots as tables showing the stats of each generated machine learning model and their respective changing hyper-parameters:
    - Create a fig with 3 tables as subplots:
        - Use go.Table to create the Validate Max Depth in DecisionTreeClassifier table.
            - Give proper label names to form the table in the header as “Max Depth”, “Train Accuracy”, and “Test Accuracy”.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected max_depth in light green.
        - Use go.Table to create the Validate N Estimators in RandomForestClassifier table:
            - Give proper label names to form the table in the header as “N Estimators”, “ Train Accuracy”, and “Test Accuracy “.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected n_estimators in light green.
        - Use go.Table to create the Validate Learning Rate in XGBClassifier table:
            - Give proper label names to form the table in the header as “Learning Rate”, “ Train Accuracy”, and “Test Accuracy “.
            - Give corresponding calculated results to the label names in cells.
            - Color the row with selected learning_rate in light green.
    - Name the chart Table Comparing the 3 Classifier Models.
- Build the final XGBClassifier model using the chosen value in hyper-parameters:
    - Set the maximum depth of the model to be 6, the n_estimators of the model to be 1000, and the learning rate to be 0.005.
    - Fit the model with training datasets.
    - Print the training and testing accuracy of the testing datasets.

    - Construct a pandas DataFrame with the actual and predict price range in the testing dataset.
- Filter the top 30 rows in the actual and predict dataset.
- Use plotly to generate a chart with 2 subplots as tables showing the stats of the final chosen XGBClassifier model and its actual and prediction performance:
    - Create a fig with 2 tables as subplots:
        - Use go.Table to create the Testing Statistics of the XGBClassifier Model table.
            - Give proper label names to form the table in the header as “Train Accuracy” and “Test Accuracy”.
            - Give corresponding calculated results to the label names in cells.
        - Use go.Table to create the Actual and Predicted Airbnb Range Comparison table (showing the top 30 rows in the testing dataset):
            - Give proper label names to form the table in the header as “Actual Range”, and “Predicted Range”.
            - Give corresponding columns to represent the data in cells.
    - Name the chart Performance of the Selected XGBClassifier Model.

Research question 4:

What are the most popular words used by hosts on Airbnbs to name their housings?

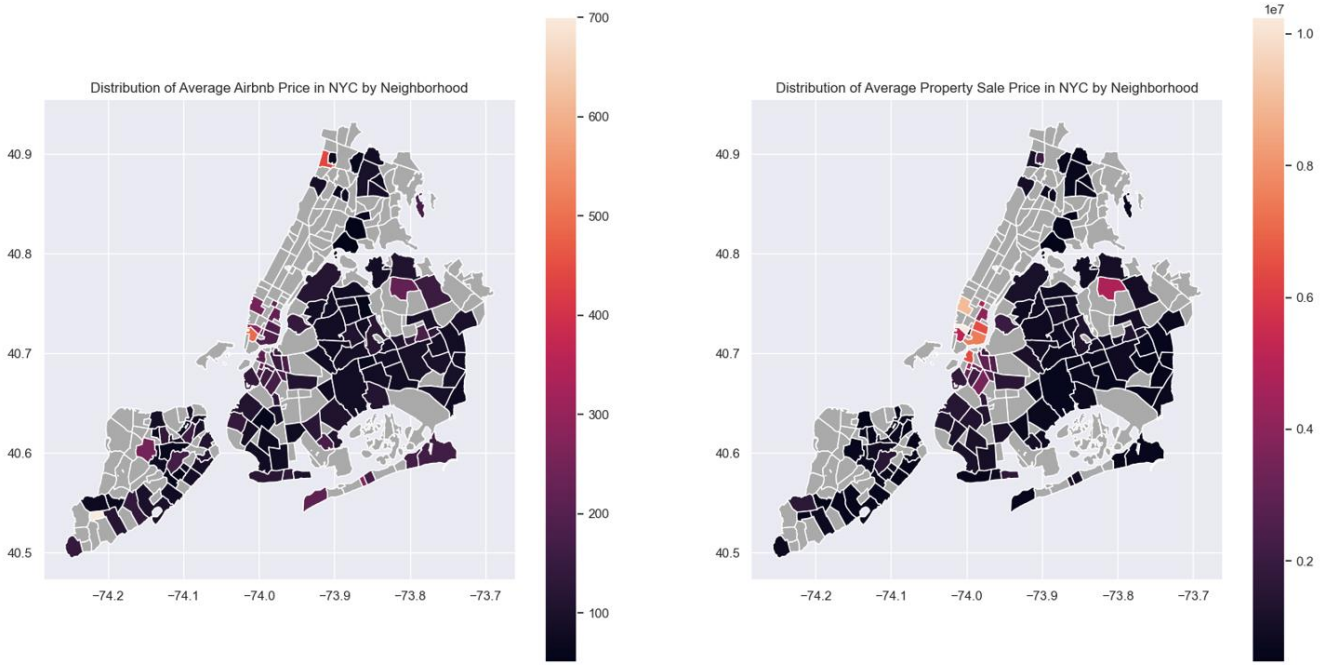
To answer the question, I ended up:

- Normalize all Airbnb names in the column “name” into lower case and concatenate each name with space as the separator.
- Use word_tokenize() to tokenize the words by splitting the normalized strings.
- Convert the result of the tokenized words into a frequency data frame and slice the data frame into the top 20 most frequent words.
- Print the top 20 most frequent words and their corresponding frequency.
- Manually exclude the rows with words such as “the” or “a” and reduce the rows to the top 10.
- Plots a bar chart containing the top 10 most frequently words used as names of Airbnbs in New York City:
    - Set the bar into blue.
    - Rotate the label words 45 degrees.
    - Title the chart The Top 10 Most Frequent Words Used in names of New York Airbnbs.
    - Saves the plot as ./q4_frq.png.

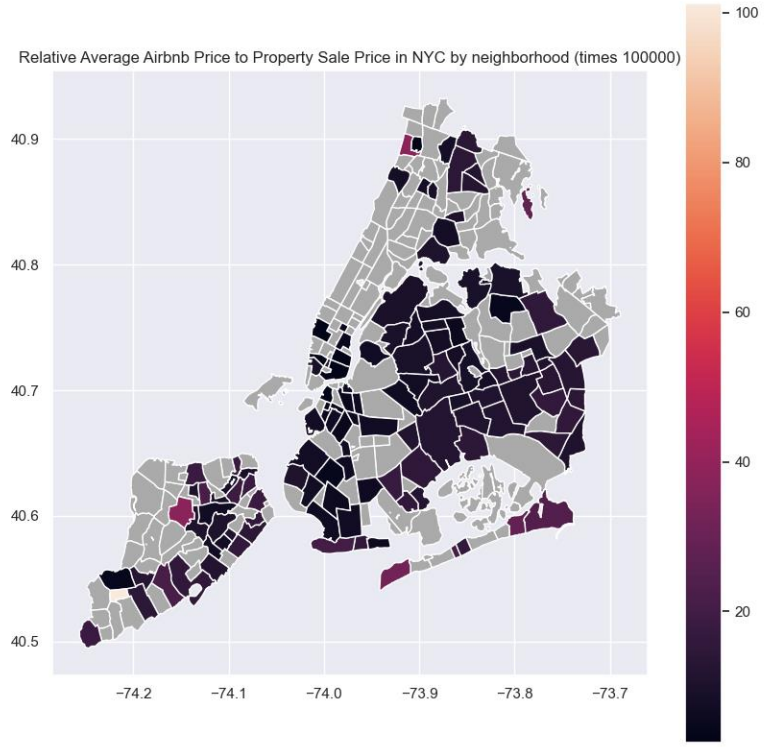
Results

Research Question 1:

To study the correlation between the average Airbnb price and the average property sale price in New York City by neighborhood, I finally came up with two charts. The first chart evals the distribution of the average Airbnb price and the average property sale price in each neighborhood separately (q1_mean.png). The second chart presents the distribution of the relative price in each neighborhood in New York City, which is calculated by dividing the average Airbnb price by the average property sale price, then multiplied by 100000 (q1_relative.png).



According to the first chart, from the plot “Distribution of Average Airbnb Price in NYC by Neighborhood”, we can see that Airbnbs in Lower Manhattan, Staten Island, the upper Brooklyn, and the lower Queens areas adjacent to the Atlantic Ocean are generally in high price. Airbnbs that locate in the north-central region in Manhattan, the central and lower Brooklyn, and the central Queens neighborhoods have very low average prices. From the plot “Distribution of Average Property Sale Price in NYC by Neighborhood”, we learn that housing prices in lower Manhattan are very high. Property sales prices are also high in the northern Brooklyn neighborhood and a neighborhood on top of Queens. However, the prices of real estate in neighborhoods in Staten Island and the lower Queens areas adjacent to the Atlantic Ocean aren’t very high.



According to the second chart, we can see the color of the relative prices in most neighborhoods in Manhattan, Brooklyn, and Queens are very dark, which means that most Airbnb pricings are proportional to the price of properties in their neighborhoods. In other words, the higher the price of the property in this neighborhood, the higher its Airbnb price. Especially in the Lower Manhattan and upper Brooklyn areas, their Airbnb prices are very reasonable (or even, we can say their Airbnb prices are cheap) because the property sales prices in these neighborhoods are already high but the colors of the relative price shown in the chart are dark. However, neighborhoods in Staten Island and the lower Queens areas adjacent to the Atlantic Ocean cannot be applied to this conclusion because Airbnb pricing in these areas is relatively high.

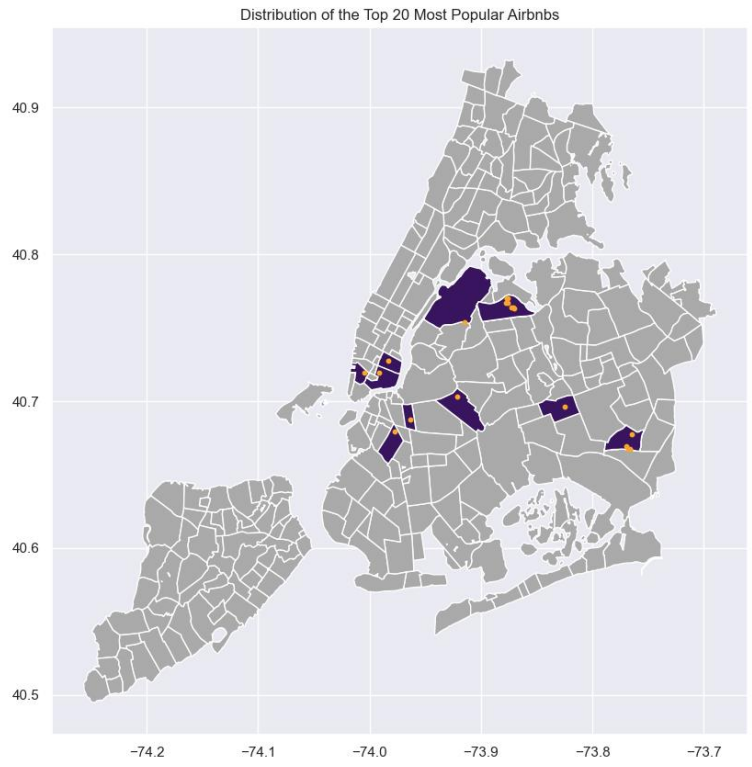
If we compare the Airbnbs price distribution chart and the relative price chart with the satellite map of New York City, we can have a better understanding of the high relative price in these two regions. According to the satellite map, there are many nature reserves and parks on Staten Island, such as Freshkills Park, Bloomingdale Park, and Clay Pit Ponds State Park. These higher-priced Airbnb neighborhoods on Staten Island are located around these parks. Similarly, the satellite map tells us that big national recreation areas and wildlife refuges are also located near the lower Queens areas adjacent to the Atlantic Ocean. The Jamaica Bay Wildlife Refuge, the Gateway National Recreation Area, the Bayswater Point State Park, the Fort Tilden Beach, and the Breezy Point Tip can all be good examples to prove the natural landscapes in the lower Queens neighborhoods. As we all know, natural landscapes are precious and expensive in New York City. Therefore, it’s reasonable to see the neighborhoods in Staten Island and the lower

Queens areas adjacent to the Atlantic Ocean have relatively high Airbnb prices compared to their actual property values.

In conclusion, the average price of Airbnbs in New York City by neighborhood is proportional to the average property sales price in those neighborhoods. The two exceptions to this conclusion are neighborhoods in Staten Island and lower Queens areas. The prices of Airbnbs are higher there because those areas have more natural sceneries.

Research Question 2:

To study the locations of the top 20 most popular Airbnbs in New York City, I finally came up with one chart and one table. The chart evals the distribution of the Airbnbs with the top 20 largest number of reviews (q2_pop.png). The table presents the key information of these top 20 most popular Airbnbs (q2_pop_table.png). (Note: to save the plot generated from plotly, click the camera label on the top right of the popping website, and rename the plot as q2_pop_table.png.)



According to the “Distribution of the Top 20 Most Popular Airbnbs” chart, we can see that the most popular Airbnbs are concentrated in three main areas-the lower Manhattan region, the

northwest corner of Queens, and the lower central Queens region. Some other popular Airbnbs that are not in this range are more loosely distributed.

The 20 Most Popular Airbnbs Stats

Name	Neighbourhood Group	Neighbourhood	room type	Airbnb price	Number of Reviews
Room near JFK Queen Bed	Queens	JAMAICA	Private room	47	629
Room Near JFK Twin Beds	Queens	JAMAICA	Private room	47	576
Steps away from Laguardia airport	Queens	EAST ELMHURST	Private room	46	543
Manhattan Lux Loft.Like.Love.Lots.Look !	Manhattan	LOWER EAST SIDE	Private room	99	540
Cozy Room Family Home LGA Airport NO CLEANING FEE	Queens	EAST ELMHURST	Private room	48	510
Private brownstone studio Brooklyn	Brooklyn	PARK SLOPE	Entire home/apt	160	488
LG Private Room/Family Friendly	Brooklyn	BUSHWICK	Private room	60	480
Bright LARGE BED near Manhattan	Queens	EAST ELMHURST	Private room	65	466
Only Steps away from LaGuardia arpt	Queens	EAST ELMHURST	Private room	45	459
yahmanscrashpads	Queens	JAMAICA	Shared room	39	454
Cozy Room in Lively East Village	Manhattan	EAST VILLAGE	Private room	72	451
Room steps away from LaGuardia airport	Queens	EAST ELMHURST	Private room	45	448
TriBeCa 2500 Sq Ft w/ Priv Elevator	Manhattan	TRIBECA	Entire home/apt	575	447
Safe cute near subway& Manhattan NY NY retro style	Queens	ASTORIA	Entire home/apt	99	441
Cute Tiny Room Family Home by LGA NO CLEANING FEE	Queens	EAST ELMHURST	Private room	48	436
THE PRIVACY DEN ~ 5 MINUTES TO JFK	Queens	SPRINGFIELD GARDENS	Entire home/apt	49	434
Comfy Room Family Home LGA Airport NO CLEANING FEE	Queens	EAST ELMHURST	Private room	54	430
Room with En Suite Bathroom & Deck	Brooklyn	CLINTON HILL	Private room	76	426
Bright LARGE 2 BEDS! near Manhattan	Queens	EAST ELMHURST	Private room	65	426
JFK 10 & LGA 15 MINUTES A/C PRIVATE BEDROOM	Queens	RICHMOND HILL	Private room	50	424

Comparing “The 20 Most Popular Airbnbs Stats” with the satellite map of New York City, I found that location is a big reason for the popularity of Airbnbs in these areas. Room type and the Airbnb price could also be factors for popularity. Let’s analyze the reasons for the popularity of Airbnbs in these neighborhoods one by one.

Firstly, the two very important airports in New York City-the LaGuardia Airport and the John F. Kenedy International Airport-are located in the northwest corner of Queens and the lower central Queens region separately. According to the table, we learned that most Airbnbs in these two regions are private rooms with affordable prices, around 50 dollars per night. Therefore, I conclude that Airbnbs located in the northwest corner of Queens and the lower central Queens regions are popular because of 1) location close to the airport 2) cheap prices and 3) private rooms.

Secondly, for the region in lower Manhattan, all the three popular Airbnbs are close to Washington Square Park and New York University. Because of the special location of Manhattan, Airbnb prices here are generally relatively high. Names of these popular Airbnbs also have the word describing luxury in them. However, compared to the average hotel price in Manhattan, which is 130 dollars according to Budget Your Trip, Airbnb prices are still lower.

What's more, because Manhattan is a business district, many people are coming and going who need accommodation. Therefore, I conclude that Airbnbs located in the lower Manhattan region are popular because of 1) it's in Manhattan 2) relative low prices compare to hotels 3) their location close to Washington Square Park and New York University.

Thirdly, for the other four popular Airbnbs that are scattered in distribution, two of them in Brooklyn are actually very close to the Barclays Center, which could be favored by NBA fans. The remaining two Airbnbs are around Highland Park and Forest Park. Their beautiful natural scenery could be the reason for their popularity.

Research Question 3:

To train the machine learning model to predict the price of an Airbnb using the neighborhood group, listing space type, required minimum night reviews per month, and average property value in its neighborhood, I first came up with an XGBRegressor model (with hyper-parameters n_estimators=1500, learning_rate= 0.0013, and max_depth=5) to predict the exact Airbnb price. I came up with the model by building 3 different regression models and testing different hyper-parameters using the validation datasets. However, the testing score of my chosen XGBRegression model is 0.0412, which means my model is very inaccurate.

In this part, I spent most of my efforts testing and comparing 3 regression models (the DecisionTreeRegressor, the RandomForestRegressor, and the XGBRegressor), trying to figure out the most accurate model, the most influential hyperparameter, and the most optimized setting for the machine learning model that can give back the highest prediction accuracy.

After the validation testing, I find out that max_depth, n_estimators, and learning_rate are the three most influential parameters regarding our model's prediction accuracy. XGBRegressor models contain the overall smallest max absolute error compared with DecisionTreeRegressor models and RandomForestRegressor models. Eventually, I decided to choose the XGBRegressor model and set out the model as max_depth=5, n_estimators=1500, and learning_rate=0.0013, which according to multiple validation processes has the highest prediction accuracy.

(Note: the tables below are all generated from plotly. To save the plot generated from plotly, click the camera label on the top right of the popping website, and rename the plots as q3_chosen.png, q3_compare_table.png, q3_chosen_c.png, and q3_compare_table_c.png.)

Table Comparing the 3 Regression Models

Validate Max Depth in DecisionTreeRegressor

Max Depth	Mean Squared Error	Mean Absolute Error	Testing Score
2	18276.07	58.21	0.1587
3	17390.1	55.54	0.1995
4	17274.7	54.8	0.2048
5	17228.92	54.42	0.2069
6	55483.89	58.64	-1.5541

Validate N Estimators in RandomForestRegressor

N Estimators	Mean Squared Error	Mean Absolute Error	Testing Score
100	20757.49	55.17	0.0445
1000	19264.79	54.89	0.1132
1500	19105.08	54.88	0.1205
1700	19148.33	54.91	0.1185
2000	19125.55	54.9	0.1196

Validate Learning Rate in XGBRegressor

Learning Rate	Mean Squared Error	Mean Absolute Error	Testing Score
0.003	18927.76	54.74	0.1287
0.002	17771.83	52.54	0.1819
0.0017	17718.76	51.37	0.1844
0.0015	17727.16	50.54	0.184
0.0013	17757.21	49.73	0.1826
0.001	18064.79	49.52	0.1684

Then, I used the previously prepared testing data to train this XGBRegressor model, and printed out the statistics of my model:

Performance of the Selected XGBRegressor Model

Testing Statistics of the XGBRegressor Model

Mean Squared Error	Mean Absolute Error	Testing Score
82494.84	57.33	0.0412

As we can see from the table above, my regression model didn't perform well. The mean squared error of my model is 82494.84, the mean absolute error is 57.33, and the testing score is 0.0412.

The predicted Airbnb prices generated from my model and the actual testing price are shown in the table below:

Actual and Predicted Airbnb Price Comparison

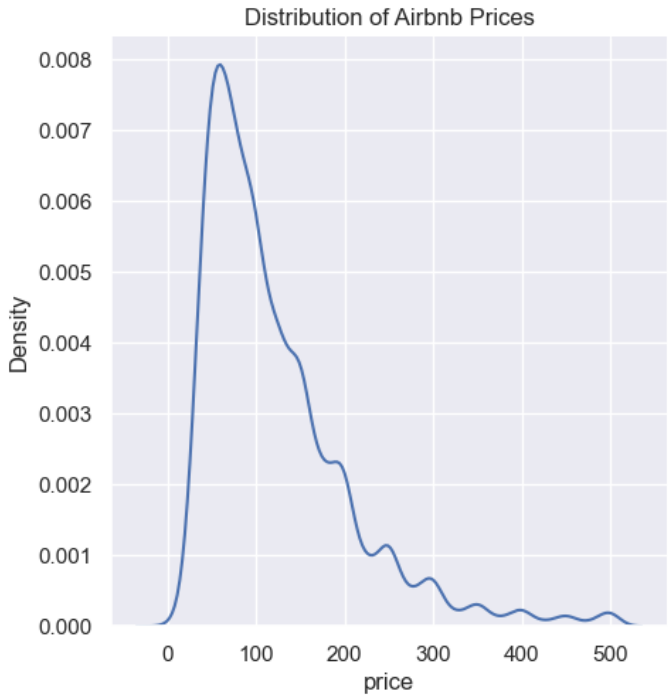
Actual Values	Predicted Values
20	49.094600677490234
200	57.22642517089844
75	105.68795013427734
162	239.6197509765625
155	98.500244140625
80	132.13792419433594
80	126.341552734375
70	53.8489990234375
520	196.4916534423828
56	56.243011474609375
24	50.87822341918945
60	54.26905059814453
60	52.10039138793945
105	132.15069580078125
179	171.0960693359375
120	143.57577514648438
80	94.65580749511719
123	94.65580749511719
175	161.1199188232422
150	115.04634857177734
248	120.30726623535156
68	106.98897552490234
95	78.39208984375
30	57.50191879272461
139	117.51952362060547
89	161.1199188232422
38	57.5450553894043
109	119.77407836914062
228	212.00608825683594
55	57.18102264404297

When I looked closer at the first thirty Airbnb prices in the testing dataset predicted by this regression model and their real prices, I found that some Airbnb price predictions are very accurate, while others are very different from the real ones. This proves that the prices of Airbnbs may be related to many other factors. It was at this point that I realized the complexity of setting prices for Airbnb and that I shouldn't expect regression models to accurately predict the price of every Airbnb. For example, in the same location, an Airbnb with a beautiful garden might cost $20 more than an Airbnb with only a lawn, and an Airbnb on the lower floor of the same building might be $40 cheaper than an Airbnb on the top floor. There are too many types of personalized information like these examples for me to include them all in the model and analyze

them. Instead, it is more accurate to classify Airbnb prices by range and to build the model as a classifier.

Therefore, here, I'd like to conclude that my final XGBRegressor model for Airbnb price prediction isn't very good, but it's also not terrible.

Now, let's move to the classifier models I build.



According to the plot of the distribution of Airbnb prices as well as the mean absolute error I got from previous regression models, I ended up splitting the Airbnb price into 6 ranges: 0-50, 50-100, 100-150, 150-200, 200-300, 300+ (for each range, the number to the left of the range is included and the number to the right is not included).

After splitting the prices by range, I repeated the steps similar to those in building regressor models and compared the accuracy score of the models built from the DecisionTreeClassifier, the RandomForestClassifier, and the XGBClassifier with different hyper-parameter values.

Table Comparing the 3 Classifier Models

Validate Max Depth in DecisionTreeClassifier

Max Depth	Training Accuracy	Testing Accuracy
2	0.4409	0.4327
5	0.4809	0.4707
6	0.4949	0.4772
7	0.5049	0.4745
8	0.5203	0.473

Validate N Estimators in RandomForestClassifier

N Estimators	Training Accuracy	Testing Accuracy
100	0.4953	0.4768
500	0.4949	0.4749
1000	0.4952	0.4734
1500	0.4956	0.4753
2000	0.4961	0.4757

Validate Learning Rate in XGBClassifier

Learning Rate	Training Accuracy	Testing Accuracy
0.01	0.5667	0.481
0.008	0.5562	0.486
0.005	0.5417	0.4883
0.003	0.5299	0.4791
0.001	0.5141	0.4707

This time, I find out max_depth and learning_rate are the two most influential parameters, while n_estimators isn't. Based on the validation test, eventually, I chose the XGBClassifier model and set out the model as max_depth=6, n_estimators=1000, and learning_rate=0.005. The testing accuracy of my chosen XGBClassifier is 0.4856, which is fairly high.

Performance of the Selected XGBClassifier Model

Testing Statistics of the XGBClassifier Model

Train Accuracy	Test Accuracy
0.5417	0.4856

The predicted Airbnb price range generated from my model and the actual testing price range are shown in the table below. From this table, we can see that nearly half of the Airbnb price ranges were accurately predicted. In the case of failed predictions, half of the ranges were not more than one level apart (ex: the actual range is 50-100 but the prediction range is 100-150). Therefore, I concluded that using a classifier to predict the Airbnb price range is better than using a regressor to predict the actual Airbnb price. Plus, the XGBClassifier I came up with performs well.

Actual and Predicted Airbnb Range Comparison

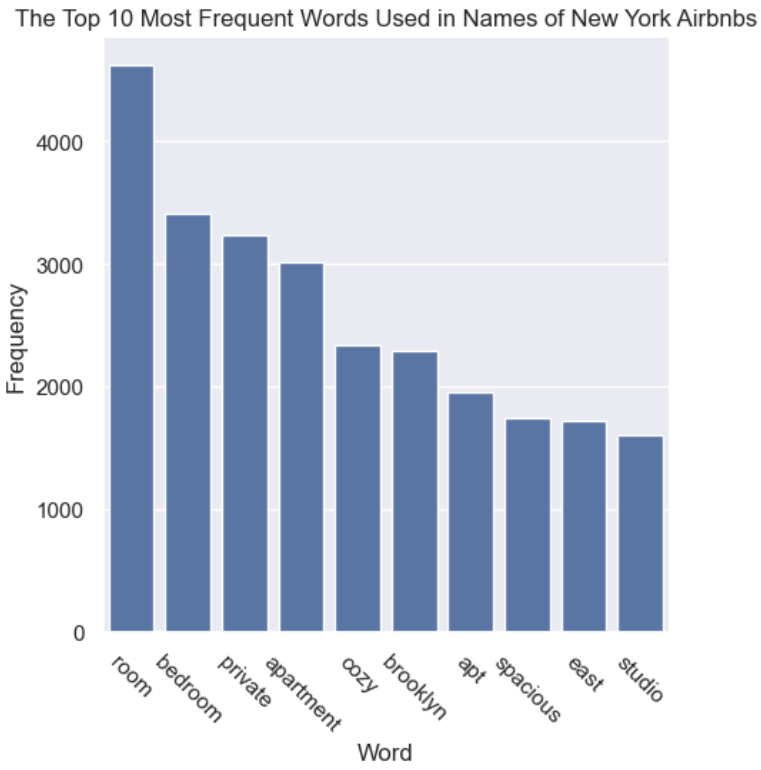
Actual Range	Predicted Range
0-50	0-50
200-300	50-100
50-100	50-100
150-200	200-300
150-200	50-100
50-100	150-200
50-100	100-150
50-100	50-100
300+	150-200
50-100	50-100
0-50	0-50
50-100	50-100
50-100	50-100
100-150	100-150
150-200	100-150
100-150	150-200
50-100	50-100
100-150	50-100
150-200	100-150
150-200	100-150
200-300	100-150
50-100	50-100
50-100	50-100
0-50	50-100
100-150	100-150
50-100	100-150
0-50	0-50
100-150	50-100
200-300	300+
50-100	50-100

In conclusion, many personalized factors affect the price of Airbnb, such as room orientation, decoration, or even having a twin-size bed or a king-size bed. These factors are too trivial to count. Thus, using the location (neighborhood group), listing space type, required minimum night, reviews per month, and average property value isn't enough to train an accurate regressor model to predict Airbnb prices. Plus, we shouldn't expect to use these data to predict the price of an Airbnb down to the penny. Conversely, we can build classification models to predict the price range of an Airbnb. Classification models predict results more accurately and provide more meaningful information than regression models. Therefore, my chosen XGBClassifier model

with max_depth=6, n_estimators=1000, and learning_rate=0.005 is a proper model to predict the price range for an Airbnb in New York City by its location, listing space type, required minimum night, reviews per month, and average property value of its neighborhood.

Research Question 4:

To study the top 10 most popular words used by hosts in the names of Airbnbs, I finally came up with a bar chart. The chart shows the ten most common words and their corresponding frequency (q4_frq.png).



According to the chart, we can divide the most common 10 words into three main categories- room type descriptions, adjectives that describe Airbnbs, and Airbnbs' locations.

Among them, "room" is the most frequent word used by the hosts to name their Airbnb. It appears over 4000 times in 21875 Airbnb names. "Bedroom", "apartment", "apt", and "studio" are also words that are among the top 10 most popular words in names and used in room-type descriptions. The reason for the frequent occurrence of these words is simple. After all, every host has to inform their customers about the type of house they own when they rent their property as an Airbnb. The easiest and most intuitive way to communicate this information is to put it in the Airbnb name. Therefore, words describing the type of house are widely used by hosts when naming their Airbnbs.

Three adjectives that describe Airbnbs are also very popular in Airbnb namings, which are “private”, “cozy”, and “spacious”. These three words are the most important to customers in their search for suitable accommodations because when people are finding for a proper Airbnb, they always take the Airbnb type, the Airbnb condition, and the Airbnb size into consideration. Psychologically, these three words will also create favorable impressions to the customers when they are choosing their Airbnbs. Even though customers didn’t go to their Airbnbs before, the word “private” can give people a sense of security, the word “cozy” can make people feel warm and welcoming, and the word “spacious” can provide people a sense of freedom. Therefore, positive adjectives used to describe Airbnbs are very popular with hosts.

Lastly, words describing locations of the Airbnb are also frequently used in Airbnb naming. “Brooklyn” and “east” are the two most popular words in this category. This result is also reasonable because hosts would also like to convey the location information of their Airbnbs firstly to their customers, especially for hosts who own properties in nice locations. Therefore, because of the need for names to convey location information, words describing locations appear frequently in Airbnb naming.

Impact and Limitations

As the title of the project says, this is an analysis of the factors that impact Airbnb prices and popularity. I have divided the audience for this analysis into two main categories-Airbnb owners and clients who are looking for accommodations.

For hosts, from the results of the analysis of the first and second research questions, they can learn that Airbnb’s location is a major factor affecting its price and popularity. Through the study of popularity in research question 2, price and popularity are connected as the property value affects the price of an Airbnb, and the price of an Airbnb affects its popularity. At this time, how to set an Airbnb price that not only makes the host satisfied but also low enough to be attractive to customers is very important. Here comes the final classification machine learning model. The model can give the host a rough reference when setting their rental price so that there is no overpricing or underpricing. What’s more, by knowing the overall price of Airbnbs in and around the neighborhood where his/her property is located from prediction and visualization, the host can better understand his/her competitors’ pricing, thus making aggressive price adjustments to increase the popularity. Sometimes in the same location, a 20-dollar difference has a big impact when customers are making their final choice. Ultimately, by analyzing the Airbnb names, hosts can optimize the name while ensuring that the information of their Airbnbs is delivered in full. They can also make their Airbnb stand out by avoiding using words that are commonly found in Airbnb naming, such as “cozy” and “spacious”. The results of this analysis are generalizable. Not only for homeowners in New York but also hosts in other areas, this analysis can be used to set Airbnb prices and names in a more reasonable way to achieve maximum profitability.

For consumers, this analysis allows them to know Airbnb pricing factors, so as to avoid price deceiving. Many consumers use Airbnb to book travel accommodations. They may not have traveled to their Airbnb locations before, much less know about the local rates and the surrounding areas. With the classifier model, customers can get a general idea of the overall Airbnb price range for their destination in advance. Although the prediction of this model is not perfect, the approximate range of prices it predicts is informative. For example, if the Airbnb price in an area is predicted to be between 50-100 dollars, they shouldn't expect their final housing spend to be $800 per night. In the same way, this model can also provide valid information for people who have a limited budget and are making travel plans. This application of my machine learning model has universal applicability in this way.

In particular, in New York City, this project can also tell customers neighborhoods with the most economical Airbnbs. It helps hosts in the city to explain the high Airbnb price in some regions. The relative price distribution map in research question 1 conveys the reasonableness of Airbnb prices in each neighborhood. Of course, Airbnb is not the cheaper the better. It is reasonable for Airbnb prices to vary by region and by type. For instance, all other things being equal, an Airbnb in eastern Brooklyn might cost $50 a night, while one in the heart of Manhattan might cost $500. However, by seeing the relative Airbnb prices to their actual worth, maybe the Airbnb in the middle of Manhattan is relatively cheaper since the property is worth millions of dollars. Comparatively, the actual value of that Airbnb in Brooklyn is only one percent of it. At this point, although the Airbnb in Manhattan has a higher price, it is the more economical one for clients and we shouldn't expect these two Airbnbs are at the same price. Therefore, this analysis will not only help customers find great value for their money with Airbnbs in New York City but will also help hosts in some specific regions explain the reasonableness of their pricing.

Now, let's talk about the limitations of my analysis. According to the map generated from research questions 1 and 2, we can see some parts of the maps are in gray, meaning that these neighborhoods lack the necessary information for my analysis. Therefore, one of the limitations in my analysis is that some results from the analysis are one-sided, which are limited to Airbnbs in New York City that only contain full data. Plus, while the factors that influence Airbnb prices and popularity are universal, the specific conclusions drawn for New York, such as that Airbnbs located in southern and northern Queens near the two airports are more popular, are limited and specific to the 100+ neighborhoods with comprehensive information in New York. What's more, on the map we can see large swaths of Airbnbs located in mid-Manhattan and Staten Island contain missing data. There may be more popular or economically priced Airbnbs in these locations, but my limited datasets have led to these Airbnbs being overlooked in the analysis. Therefore, the full conclusions from this analysis are for the 100+ neighborhoods in New York only, and they can only be used as a reference when analyzing Airbnbs in other areas.

Secondly, as stated in the analysis of the machine learning model, there are far more factors affecting Airbnb prices than location (neighborhood group), listing space type, required minimum night, reviews per month, and average property value. More personal factors are taken

when hosts are setting the prices for their Airbnbs, such as housing appearance, bed size, air conditioner, floor level, and maybe even noise level in the nearby areas. We cannot take every factor into account. Moreover, in my analysis, I used the average property value to predict the single Airbnb price. Here I default to the same price in the same neighborhood. In reality, even in the same condominium, the final price will be different due to the different room types, orientations, floors, and decorations. Different apartments in the same location may have different prices depending on the age of the building, condition of the elevator, air conditioning, etc. These two reasons eventually led to a large deviation of the predicted price from the real price in the model I built. Even though the final prediction is the price range and the testing accuracy reaches over 48%, there are still many inaccurate predictions. Therefore, the analysis could only provide a general idea about the price. The predicted results are for reference only.

Challenge Goals

When I started to write the analysis, I realized that my project was more complex than I thought and met more challenge goals. In the end, my final project met three challenge goals:

1, Multiple Datasets

The first challenge goal I met in my final project is multiple datasets. For my analysis, I used 7 original databases and generated about 20 new ones by merging and exporting, each with its own role in coding. I also transformed the 5 databases of Neighborhood Sales Data from 2005 through 2021 from the original xlsx file format to CSV file format before merging. I believe the multiple datasets I used in my final project analysis met this challenge goal.

2, Machine Learning

The second challenge goal I met is machine learning. In research question 3, I achieve this challenge goal by testing the validation datasets on 3 different regression models (DecisionTreeRegressor, RandomForestRegressor, and XGBRegressor) and 3 different classification models (DecisionTreeClassifier, RandomForestRegressor, and XGBClassifier), determining the best value for the hyper-parameter in these models. Since my project involved using multiple machine learning models and going more in-depth with it than I have in class, I believe my project meets the machine learning challenge goal.

3, New Library

To achieve the complexity in my final project, I also use 3 new libraries-plotly for graphing consolidated tables with color, xgboost for building the more accurate XGBRegressor and XGBClassifier in machine learning, and nltk for normalizing words and analyzing their frequencies appear in Airbnb names. All of the libraries need time to learn. Since my project involved learning and using three Python libraries not covered in class, I believe my project also

meets the new library goal.

Work Plan Evaluation

For this final project, my planned tasks were:

1, Clean and merge datasets:

Estimate time: 8 hours

Actual time required: 10 hours

In the beginning, the five databases downloaded from the NYC Department of Finance were in xlsx format. I spent an hour and a half searching the web and got how to convert them from xlsx format to CSV format. Then, it took me 8 hours to merge all my 7 datasets together and check. During this time, I also completed some data transformations, such as converting columns of data from string format to integer as they should be. Checking the differences in "neighbourhood" columns in different datasets before merging also took me a long time to figure out. The method of finding groupby() and then getting a database instead of a series is also time-consuming. Finally, I spent about 30 minutes constructing the c_filtered_data() function for building the classification model in research question 3.

2, Generate the maps of the "Distribution of Average Property Price in NYC by Neighborhood" and the "Relative Average Airbnb Price to Property Sale Price NYC by neighborhood (times 100000)" with geopandas for research question 1:

Estimate time: 3 hours

Actual time required: 2 hours

Since we at that time just finished geopandas and subplots in class, the map plotting part of my project was quite smooth for me. I did spend some time changing the colors of the legend into a monochrome gradient in my maps. However, in the end, I realized the default colors are the best and the clearest. Therefore, I changed the color back to default.

3, Generate the maps of the "Distribution of the Top 20 Most Popular Airbnbs" with geopandas and build a table showing the information of the Top 20 Airbnbs with plotly for research question 2:

Estimate time: 4 hours

Actual time required: 8 hours

When generating this map, I ran into a warning about the Coordinate Reference System that I cannot figure out by myself. I spent a long time working on it and finally, I solved the warning with Wen's help. It took me a total of 4 hours to deal with this warning. Then, I spent around 2

hours learning plotly from its official documents and built a table with plotly. It also took me 2 hours to figure out how to save the figure generated from plotly into png file. However, there is no very good way to save a picture from plotly without a web page appearing (I have confirmed this thought with Wen). Therefore, I have to save the generated table manually by clicking the camera icon on the top right of the popping web page.

4, Learn extra machine learning models, construct the XGBRegressor model, and plot tables for visualization for research question 3:

Estimate time: 15 hours

Actual time required: 25 hours

When I chose machine learning as my challenge goal, I knew I would be spending a lot of time on it. But in the end, it took me much longer than I expected to build and select the proper model. Luckily, I didn't have any other classes during the summer. Learning and building regression models took me a total of 4 full days. Processing the data and learning to build validation datasets only took me one hour. First, I want to predict the exact Airbnb price, so I need to build a regression model. Since we talked about DecisionTreeRegressor in class, I first built the model for my data. But the testing score of my validation datasets for the DecisionTreeRegressor is too low that I have to find another model to fit the data. Therefore, I spent almost 6 hours studying different regression models online and looking for the more suitable ones, resulting in picking the RandomForestRegressor and XGBRegressor. At this time our class just happened to talk about the neural network. I had a deeper understanding of the process inside machine learning and decided to use a for loop to test different hyper-parameters in my models. Testing various hyper-parameters took me a long time (over 8 hours, I think) because 1) I didn't know which hyper-parameters has the biggest impact on my model. 2) running machine learning models already took some time. After testing all key hyper-parameters in 3 different regression models and printing the results from the validation datasets out, I constructed a combined table from plotly showing the results of the models, which took me 2 hours to figure out. Then, I fitted my chosen model with the training dataset, tested the model, and constructed another combined table representing statistics from the final regression model and the actual & prediction value comparison. This process took me another 3 hours. During this time, it took me about 4 hours to tweak each model's code, set the random state, reduce the code redundancy, and print out each result. However, as we can see in the previous sections, the testing score of my final picked XGBRegressor model is very small.

5, Learn extra machine learning models, construct the XGBClassifier model, and plot tables for visualization for research question 3:

Estimate time: 5 hours

Actual time required: 7 hours

I was very happy on Wednesday when I knew that the deadline for final deliverable was pushed back. Because the regression model I had spent a long time building did not turn out well, I realized that I could not predict the price accurately. The extra three days gave me time to try to build a classification machine learning model to predict the price range of Airbnbs. Since many of the steps in building the classifier are similar to the regressor, I saved a lot of time learning and familiarizing myself with the models. I ended up spending roughly 7 hours building, testing and comparing three different classification models. I repeated the method of making plotly tables and generated the final comparison table. My efforts were not in vain, as the test accuracy of the generated XGBClassifier model was nearly 50%.

6, Learn extra natural language processing library called NLTK and plot the "The Top 10 Most Frequent Words Used in Names of York Airbnbs" chart:

Estimate time: 5 hours

Actual time required: 4 hours

After one hour of searching, I had no trouble finding a way to use nltk and find the n most frequently occurring words. As a result, this part of the code was written very smoothly. After that, I spent another hour filtering the dataset, manually finding the 10 most frequent words, and making a bar plot. This part took me roughly 4 hours in total.

7, Testing my code:

Estimate time: 5 hours

Actual time required: 10 hours

Because my project produced a lot of graphs and constructed machine learning models, I could not use assert equals for testing. I tried to find a lot of testing methods and asked Wen and my teaching assistant for advice for a long time (about 2 hours for the preparation). Eventually, at the suggestion of Wen and James, I ended up testing the databases I used to draw the diagrams for the first and second research questions. I wanted to use excel for database testing at first. However, I have never learned excel. After spending two hours without any progress, I gave up using excel to compare database identities and chose to use RStudio, which I have mastered very well. After that, I spent two hours testing the datasets for the first two questions. I spent another 3 hours writing the machine learning part of the test. Finally, it took me an hour and a half to test the whole process of the fourth research question.

8, Write the readme instruction:

Estimate time: 2 hours

Actual time required: 2 hours

I had learned markdown formatting in other classes before, so I didn't need to spend time learning it. Since all the code instructions I recorded while coding, in this section I just need to create a readme file, organize my language, and put what I recorded in it. This part ended up taking me two hours.

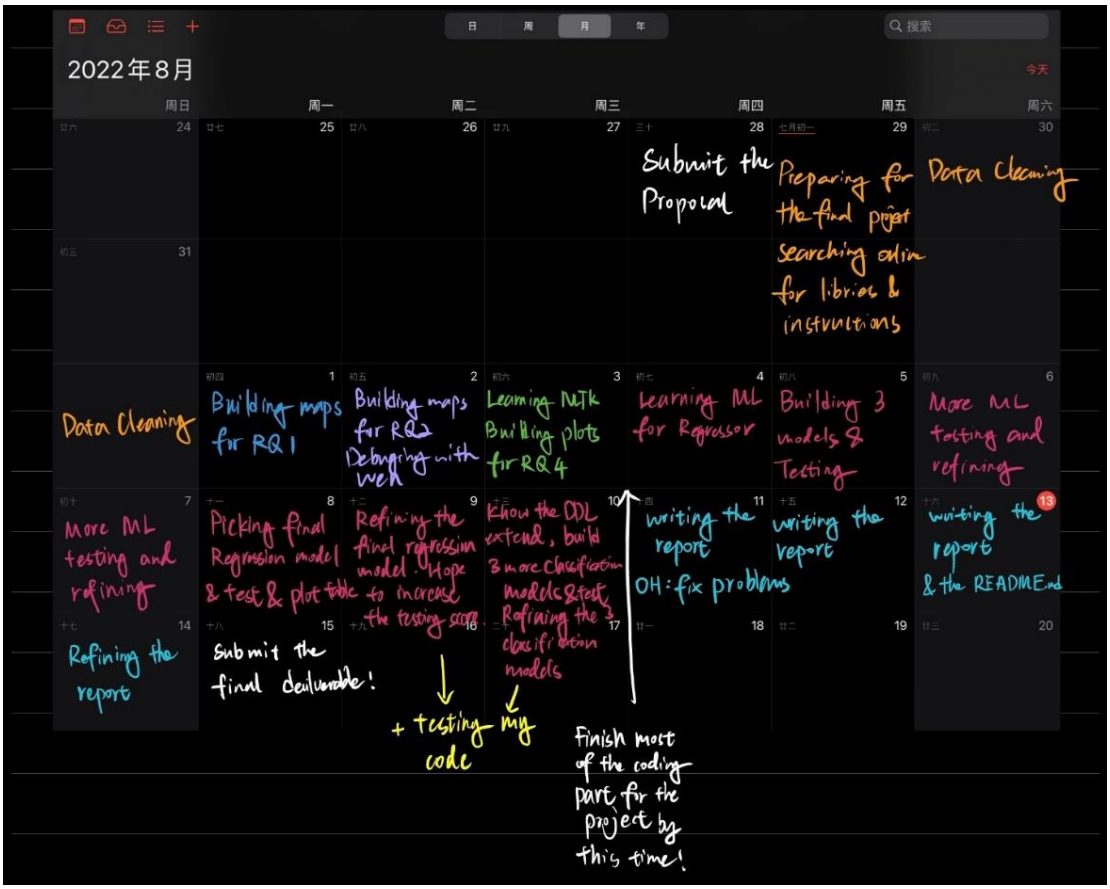
9, Write the report:

Estimate time: 10 hours

Actual time required: 20 hours

Writing the report is a long process for me because I write too much code, and almost every line of code requires me to analyze. That's why I've been working hard every day on my report. Plus, English is not my native language, so I am very slow in writing this report. I can't clearly estimate how long it took me to write this report because I finished it in stitches. Before Thursday, I was writing code while making some notes for the report. After Thursday, I finished all the code parts of the project and focused more on writing the results, limitations, and evaluation of the report. This part took me about 10 hours of time over two days. I also spent 2 hours revising the issues of my proposal. In the end, I also spent some time revising the grammar and upgrading my report.

Finally, I created a schedule to show my daily tasks and progress more clearly.



Testing

Research Question 1:

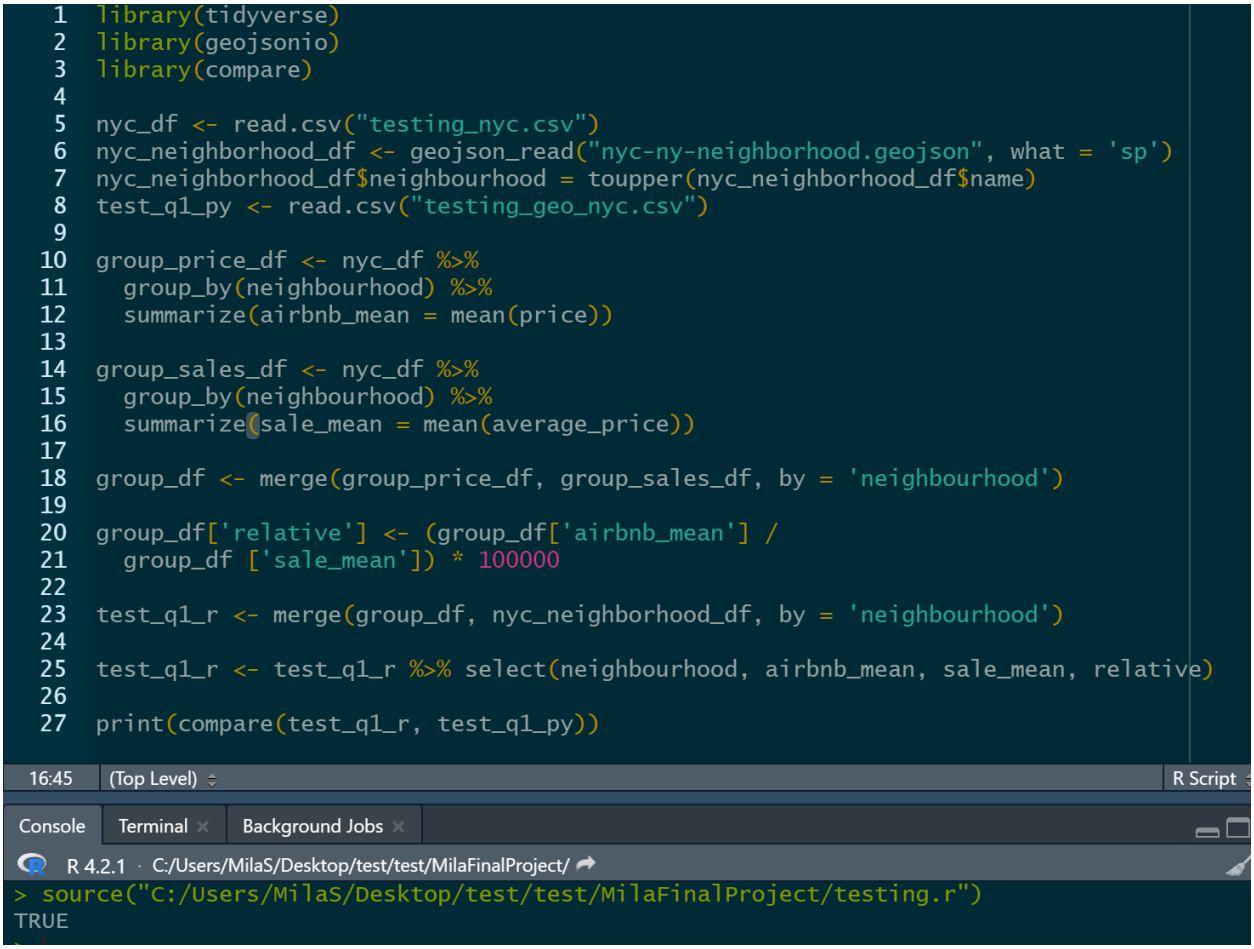
Since testing charts generated from geoDataFrame aren't an easy process, I decide to test the dataset I use for graphing. For research question 1, I used RStudio to test the calculated average Airbnb price, average property sales price, and relative price by neighborhood.

Firstly, I rebuilt a grouped data frame by neighborhood in RStudio and calculated the average Airbnb prices, the average property values in each neighborhood, and the relative price using the nyc_df. I ended up with:

- Export the nyc_df into a CSV file in my laptop using the to_csv() method.
- Import the nyc_df into RStudio.
- Group the data by neighborhood and summarize the average Airbnb prices.
- Group the data by neighborhood and summarize average property values in each neighborhood.
- Merge the two grouped datasets together.
- In the merged dataset, create a new column to calculate the relative price for each neighborhood.
- Merge the dataset with the nyc_neighborhood_df (the data frame that contains the geometry of each neighborhood) and select the columns back to the calculated average Airbnb price, average property sales price, and relative price.
- Name it test_q1_r.

Secondly, I compared the calculated average Airbnb price, average property sales price, and relative price by neighborhood in my merged dataset in RStudio with those I used for building the chart. I ended up with:

- Filter the geo_group_df I used for plotting in Visual Studio Code into four columns- neighborhood, average Airbnb price, average property sales price, and relative price
- Export the filtered data frame into a CSV file in my laptop using the to_csv() method.
- Import the dataframe into RStudio, name it test_q1_py.
- Use compare() to compare test_q1_r and test_q1_py.



As we can see in the picture, the result of compare() is TRUE, meaning that the two datasets are the same in value. Therefore, with RStudio, I proved that the dataset I used for plotting q1_mean.png and q1_relative.png was correct.

Research Question 2:

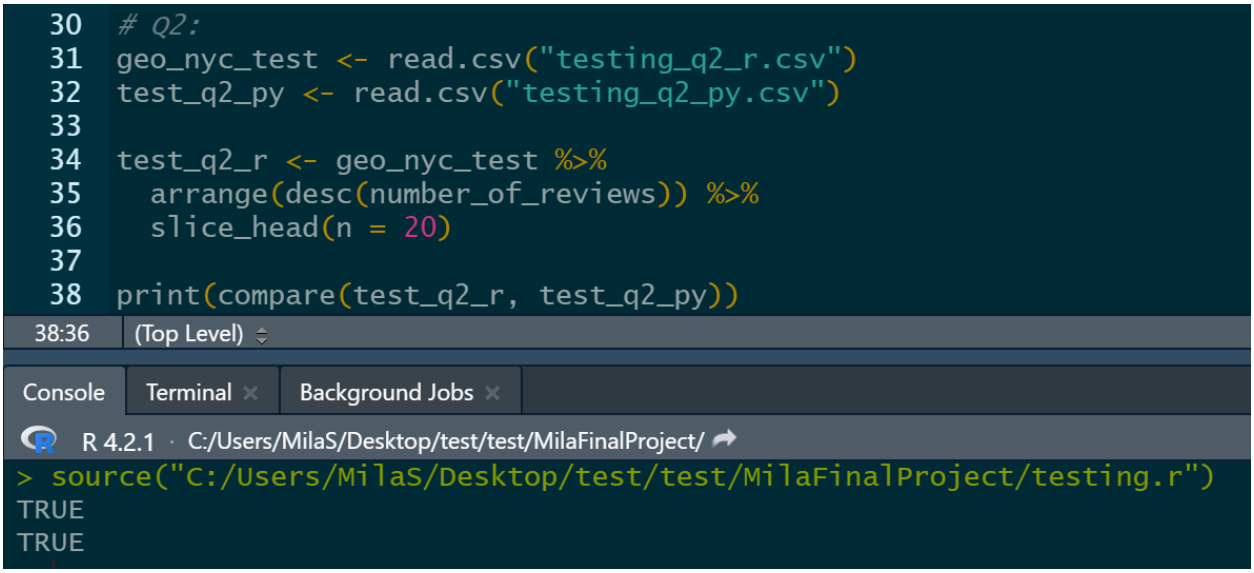
Since testing charts generated from geoDataFrame aren't an easy process, I decide to test the dataset I use for graphing. For research question 2, I used RStudio to test the top 20 Airbnbs with the most reviews.

Firstly, I rebuilt a top 20 most popular Airbnbs data frame in RStudio by arranging the filtered geo_nyc_df that I used for plotting. I ended up with:

- Export the filtered geo_nyc_df into a CSV file in my laptop using the to_csv() method.
- Import the data frame into RStudio.
- Use arrange() to arrange the number_of_reviews in descending order and slice the data frame into the top 20.
- Name it test_q2_r.

Secondly, I compared the top 20 most popular Airbnbs I calculated in RStudio with those I used for building the chart. I ended up with:

- Filter the geo_nyc_20 I used for plotting in Visual Studio Code into 2 columns- neighbourhood and number_of_reviews.
- Export the filtered dataframe into a CSV file on my laptop using the to_csv() method.
- Import the dataframe into RStudio, name it test_q2_py.
- Use compare() to compare test_q2_r and test_q2_py.



As we can see in the picture, the result of compare() is TRUE (the first TRUE is for the testing in research question 1 and the second TRUE is for this test), meaning that the two datasets are the same in value. Therefore, with RStudio, I proved that the dataset I used for plotting q2_pop.png and q2_pop_table.png was also correct.

Research Question 3:

(Note: I have verified my testing method for this question with my mentor-James Cao.)

For the machine learning portion of the project, most of the testing parts are already done when trying different hyper-parameters and building the machine learning models. Coming up with an XGBRegressor model with hyper-parameters n_estimators equals 1500, learning_rate equals 0.0013, and max_depth equals 5, I ended up with:

Table Comparing the 3 Regression Models

Validate Max Depth in DecisionTreeRegressor

Max Depth	Mean Squared Error	Mean Absolute Error	Testing Score
2	18276.07	58.21	0.1587
3	17390.1	55.54	0.1995
4	17274.7	54.8	0.2048
5	17228.92	54.42	0.2069
6	55483.89	58.64	-1.5541

Validate N Estimators in RandomForestRegressor

N Estimators	Mean Squared Error	Mean Absolute Error	Testing Score
100	20757.49	55.17	0.0445
1000	19264.79	54.89	0.1132
1500	19105.08	54.88	0.1205
1700	19148.33	54.91	0.1185
2000	19125.55	54.9	0.1196

Validate Learning Rate in XGBRegressor

Learning Rate	Mean Squared Error	Mean Absolute Error	Testing Score
0.003	18927.76	54.74	0.1287
0.002	17771.83	52.54	0.1819
0.0017	17718.76	51.37	0.1844
0.0015	17727.16	50.54	0.184
0.0013	17757.21	49.73	0.1826
0.001	18064.79	49.52	0.1684

Firstly, I wanted to build a DecisionTreeRegressor for the prediction. I made the hyper-parameter max_depth changeable so that I wish to find a better model with proper max_depth. However, according to the “Table Comparing the 3 Machine Learning Models”, all the models generated from the DecisionTreeRegressor aren’t very good in performance. I found that when max_depth equals 5, I got the best DecisionTreeRegressor model. But the mean absolute error of the model is still very large and the testing score of the model isn’t very high. Therefore, I decided to try some other machine learning model in order to come up with a more accurate mode.

Then, I built a RandomForestRegressor, a model which combines several decision trees. I set up the max_depth of the model into 5 and made the hyper-parameter n_estimators changeable. I

hope I can get a better performance model. However, the RandomForestRegressor models perform worse than the DecisionTreeRegressor models. I found that when n_estimators equals 1500, the RandomForestRegressor model performs the best. Therefore, I need to keep going for another model.

Finally, I came up with an XGBRegressor model. This is a popular Extreme Gradient Boosting Regressor. I set up the max_depth into 5 and the n_estimators equals 1500. Then I made the learning_rate changeable for better models. In the end, I decide to use the XGBRegressor model with learning_rate equaling to 0.0013. Even though the testing score of the model is still low, the maximum absolute error of the validation datasets was reduced to below 50, which is huge progress for the model.

Performance of the Selected XGBRegressor Model

Testing Statistics of the XGBRegressor Model

Mean Squared Error	Mean Absolute Error	Testing Score
82494.84	57.33	0.0412

Therefore, I found out the optimum hyperparameter setting for my model is: max_depth=5, n_estimators=1500, learning_rate=0.0013.

The mean squared error of my model is 82494.84, the mean absolute error is 57.33, and the testing score is 0.0412. The result predicted values of Airbnb prices and the actual prices in the testing datasets are shown below:

Actual and Predicted Airbnb Price Comparison

Actual Values	Predicted Values
20	49.094600677490234
200	57.22642517089844
75	105.68795013427734
162	239.6197509765625
155	98.500244140625
80	132.13792419433594
80	126.341552734375
70	53.8489990234375
520	196.4916534423828
56	56.243011474609375
24	50.87822341918945
60	54.26905059814453
60	52.10039138793945
105	132.15069580078125
179	171.0960693359375
120	143.57577514648438
80	94.65580749511719
123	94.65580749511719
175	161.1199188232422
150	115.04634857177734
248	120.30726623535156
68	106.98897552490234
95	78.39208984375
30	57.50191879272461
139	117.51952362060547
89	161.1199188232422
38	57.5450553894043
109	119.77407836914062
228	212.00608825683594
55	57.18102264404297

Coming up with an XGBClassifier model with hyper-parameters n_estimators equals 1000, learning_rate equals 0.005, and max_depth equals 6, I ended up with:

Table Comparing the 3 Classifier Models

Validate Max Depth in DecisionTreeClassifier

Max Depth	Training Accuracy	Testing Accuracy
2	0.4409	0.4327
5	0.4809	0.4707
6	0.4949	0.4772
7	0.5049	0.4745
8	0.5203	0.473

Validate N Estimators in RandomForestClassifier

N Estimators	Training Accuracy	Testing Accuracy
100	0.4953	0.4768
500	0.4949	0.4749
1000	0.4952	0.4734
1500	0.4956	0.4753
2000	0.4961	0.4757

Validate Learning Rate in XGBClassifier

Learning Rate	Training Accuracy	Testing Accuracy
0.01	0.5667	0.481
0.008	0.5562	0.486
0.005	0.5417	0.4883
0.003	0.5299	0.4791
0.001	0.5141	0.4707

Firstly, I wanted to build a DecisionTreeClassifier for the prediction. I made the hyper-parameter max_depth changeable so that I wish to find a better model with proper max_depth. This time, my classifier models perform better with all of their testing accuracy above 0.4. I found that when max_depth equals 6, I got the best DecisionTreeClassifier model. Since the testing accuracy of the classifier is so much higher than the regressor, I'd like to try some other classifiers to see if I can improve the testing accuracy to 0.5.

Then, I constructed a RandomForestClassifier, a model which combines several decision trees. I set up the max_depth of the model into 6 and made the hyper-parameter n_estimators changeable. I hope I can get a better performance model. However, the performance of the RandomForestClassifier is similar to that of the DecisionTreeClassifier. Plus, the varying

n_estimators didn’t affect the testing accuracy much. Therefore, I decide to pick the n_estimators as 1000 and keep going to find a more accurate model.

Finally, I came up with an XGBClassifier model. I set up the max_depth into 6 and the n_estimators equals 1000. Then I made the learning_rate changeable for better models. In the end, I decide to use the XGBClassifier model with learning_rate equaling to 0.005, resulting in the testing accuracy in my validation dataset boosting into over 0.48.

Performance of the Selected XGBClassifier Model

Testing Statistics of the XGBClassifier Model

Train Accuracy	Test Accuracy
0.5417	0.4856

Finally, I found out the optimum hyperparameter setting for my model is: max_depth=6, n_estimators=1000, learning_rate=0.005.

The training accuracy of my chosen model is 0.5417 and the testing accuracy is 0.4856, which evals the classification model is ok. The result predicted Airbnb price range and the actual price range in the testing datasets is shown below:

Actual and Predicted Airbnb Range Comparison

Actual Range	Predicted Range
0-50	0-50
200-300	50-100
50-100	50-100
150-200	200-300
150-200	50-100
50-100	150-200
50-100	100-150
50-100	50-100
300+	150-200
50-100	50-100
0-50	0-50
50-100	50-100
50-100	50-100
100-150	100-150
150-200	100-150
100-150	150-200
50-100	50-100
100-150	50-100
150-200	100-150
150-200	100-150
200-300	100-150
50-100	50-100
50-100	50-100
0-50	50-100
100-150	100-150
50-100	100-150
0-50	0-50
100-150	50-100
200-300	300+
50-100	50-100

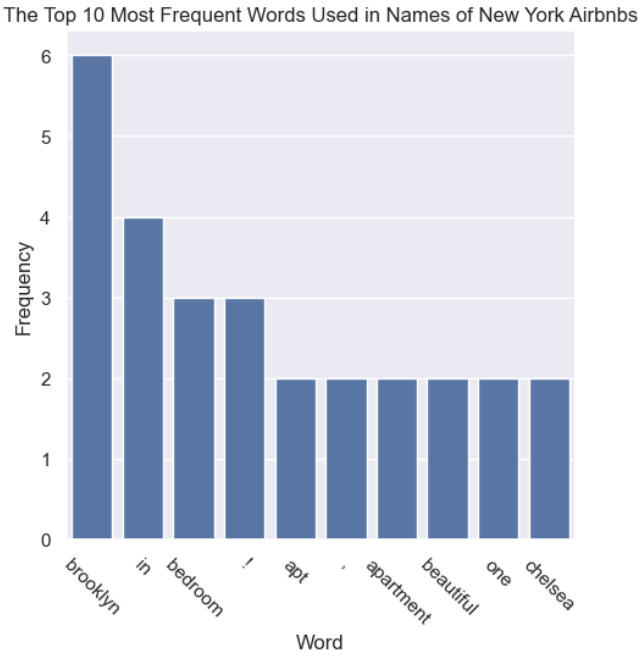
Research Question 4:

For research question 4, I decided to test with a smaller dataset using Python in Visual Studio Code. I counted the frequencies of words that appear in names in my testing dataset manually. Then I called the test mode of the q4_count() function and generated a graph to test if the top 10 most frequent words matched with the result I counted. I ended up with:

- Randomly select 10 rows in the nyc_df to form a small testing dataset and filter the column in the testing dataset to only name:

```
                                                    name
7                                    Lovely Brooklyn Apt
9            Vintage Chic Haven in Kensington, Brooklyn
48                           Brooklyn Basement Apartment
148    Cozy tiny bedroom w/large living-room & kitchen.
2000                       Beautiful One Bedroom Chelsea
2022                Modern Flat in the Heart of Chelsea
248                    SUPER BOWL Brooklyn Duplex Apt!!
448              Sunny apartment in Brooklyn brownstone
548   Hip Modern Brooklyn Studio, Minutes to Manhattan!
600            Sunny One Bedroom in Beautiful Brownstone
```

- According to the printed-out names in my testing dataset, my manually counted result for the top 10 words in it is: “brooklyn” (6 times), “in” (4 times), “bedroom” (3 times), “!” (3 times), “apt” (2 times), “,” (2 times), “apartment” (2 times), “beautiful” (2 times), “one” (2 times), and “chelsea” (2 times).
- Test the 10-rows dataset in the function q4_count() in testing mode (with test=True) and name the generated testing plot as ./q4_frq_test.png.



According to the generated testing plot, both the words and the word frequencies in the graph match exactly with my manual calculations. Therefore, with a smaller data frame, I proved that the plot-generating function I used in question was correct.

Collaboration

In the course of learning new libraries, I consulted many online resources.

Specific instances are listed below:

- Figuring out how to convert excel to CSV format: https://www.exceldemy.com/convert-excel-to-csv-format/#:~:text=1%20Convert%20Excel%20to%20CSV%20Format%20Through%20Save,of%20a%20single%20worksheet%20to%20CSV%20format.%20
- Figuring out how to concatenate pandas datasets: https://pandas.pydata.org/docs/reference/api/pandas.concat.html
- Figuring out how to groupby() and return a DataFrame: https://stackoverflow.com/questions/37984736/pandas-return-a-dataframe-after-groupby
- Figuring out how to solve the coordinate reference system problem: https://geopandas.org/en/stable/docs/reference/api/geopandas.GeoDataFrame.set_crs.html
- Figuring out how to build a table in plotly: https://plotly.com/python/table/
- Figuring out how to build multiple tables in one chart using plotly: https://plotly.com/python/table-subplots/
- Figuring out the average hotel price in Manhattan: https://www.budgetyourtrip.com/hotels/united-states-of-america/manhattan-4274994#:~:text=After%20analyzing%2012%20hotels%20in%20Manhattan%2C%20we%20found,hotel%27s%20amenities%2C%20available%20dates%2C%20and%20the%20general%20neighborhood.
- Figuring out how to split the datasets into training, testing, and validation datasets: https://www.kaggle.com/discussions/getting-started/143685
- Figuring out the definition of RandomForestRegressor and the way to build the model: https://towardsdatascience.com/random-forest-regression-5f605132d19d
- Figuring out how to build a XGBRegressor model: https://www.datatechnotes.com/2019/06/regression-example-with-xgbregressor-in.html
- Figuring out how to build a XGBClassifier model: https://www.kaggle.com/code/raafaq/xgbclassifier
- Figuring out how to count the word frequency with nltk: https://codereview.stackexchange.com/questions/249329/finding-the-most-frequent-words-in-pandas-dataframe