From Data to Decisions: Exploring Airbnb Market Dynamics in New York City

1. Introduction:

The Airbnb NYC dataset provides comprehensive information about Airbnb listings in New York City for 2019, covering variables such as neighbourhoods, property types, pricing, availability, host details, and customer reviews. This rich dataset is valuable for analysing market trends, customer preferences, and short-term rental dynamics in one of the world's most competitive markets. It aids Airbnb in data-driven decision-making to optimise pricing, identify high-performing neighbourhoods, and improve resource allocation.

By analysing customer demand trends and host behaviour, Airbnb can enhance customer experiences, refine marketing efforts, and uncover growth opportunities in underutilised areas.

1.1 Objective:

Utilising the Airbnb NYC dataset and machine learning models to predict growth trends in Airbnb listings and average pricing for 2019.

2. Methodology:

2.1. Methodology Overview

This study used a systematic approach, combining data cleaning, exploratory data analysis (EDA), and predictive modelling. Python was the primary tool with libraries for data manipulation, visualisation, and machine learning.

2.2 Data Collection and Preprocessing

The dataset was sourced from a publicly available Airbnb repository. Key variables included room_type, price, latitude, longitude, and number_of_reviews.

The preprocessing phase focused on data cleaning to address inconsistencies and missing entries. Null values were imputed, missing categorical data was replaced or flagged, and outliers and faulty data were removed. These steps were applied to all numerical columns.

2.3 Exploratory Data Analysis (EDA)

EDA uncovered patterns and relationships within the dataset. Statistical summaries provided an overview of key metrics, while visualisations such as histograms, scatterplots, and box plots highlighted trends and anomalies. The distributions of numerical variables, including price and availability, were examined in detail, while scatterplots of latitude and longitude revealed spatial trends. Additionally, categorical analysis explored room type distributions, and host behaviour. Likewise, all numerical columns were cleansed of outliers and faulty data.

2.4 Analytical and Modelling Techniques

Key analytical methods included outlier detection with box plots, trend analysis using correlation coefficients, and neighbourhood grouping for average prices. These techniques helped identify premium and affordable areas while providing insights into market dynamics. The 2019 dataset, with the highest number of properties, contained both active and inactive listings. Historical data (2011–2018) on inactive properties guided a polynomial model, optimised with RMSE, to predict inactive properties in 2019. Subtracting these predictions from total listings refined the dataset. Feature engineering used Scikit-learn's PolynomialFeatures, and performance was evaluated using MAE, MSE, and R².

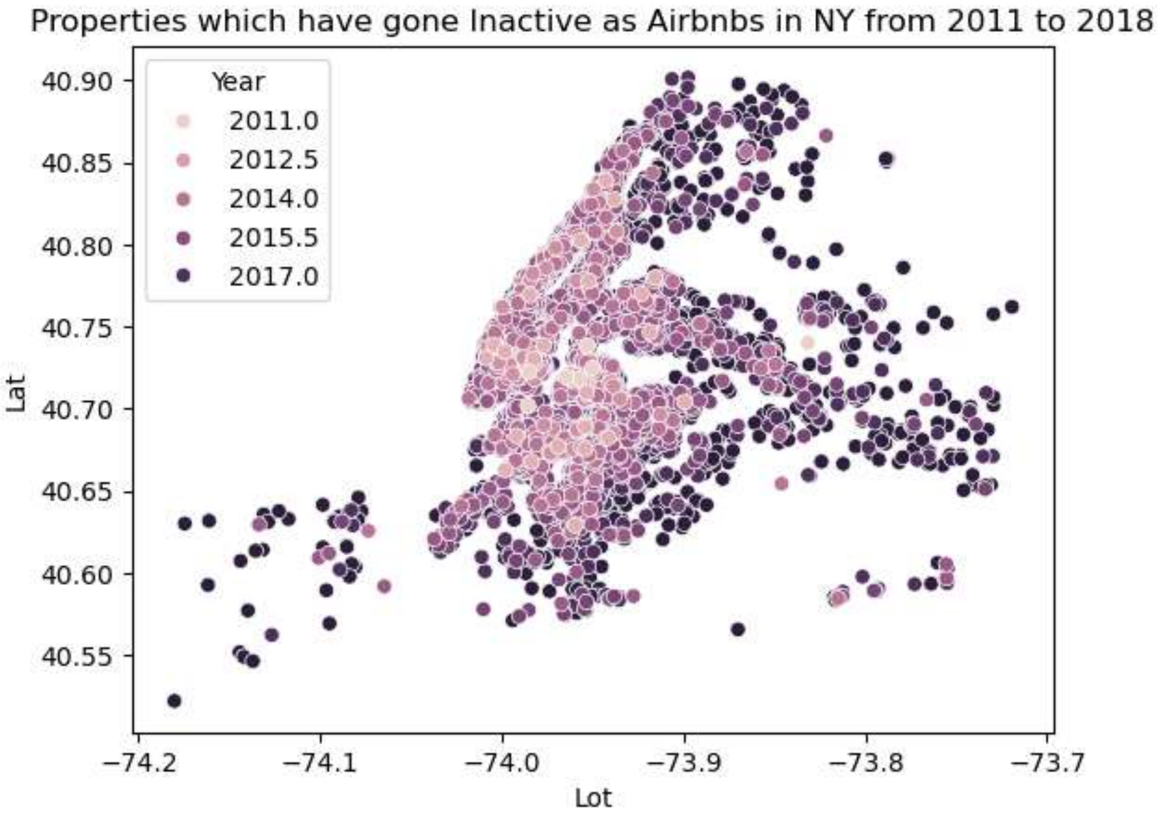


Figure 1: The properties that have become inactive as Airbnbs from 2011-2018 in New York.

2.5 Tools and Technologies

In addition to Python, Pandas was used for data manipulation, while NumPy facilitated numerical calculations. Visualisations were created with Seaborn and Matplotlib, and Scikit-learn was employed for modelling tasks. All analysis was conducted within the Jupyter Notebook environment, ensuring an interactive and iterative workflow.

3. Findings:

3.1 Property Types Reflect Neighborhood Trends Across NYC

Our data analysis aligns with Sarkar et al.'s (2020) research on the spatial and socio-economic factors influencing Airbnb hosting in NYC. Northern Brooklyn and Manhattan, as moderately to highly dense areas, stand out as key hubs of activity, marked by high review counts and property types that attract both tourists and locals.

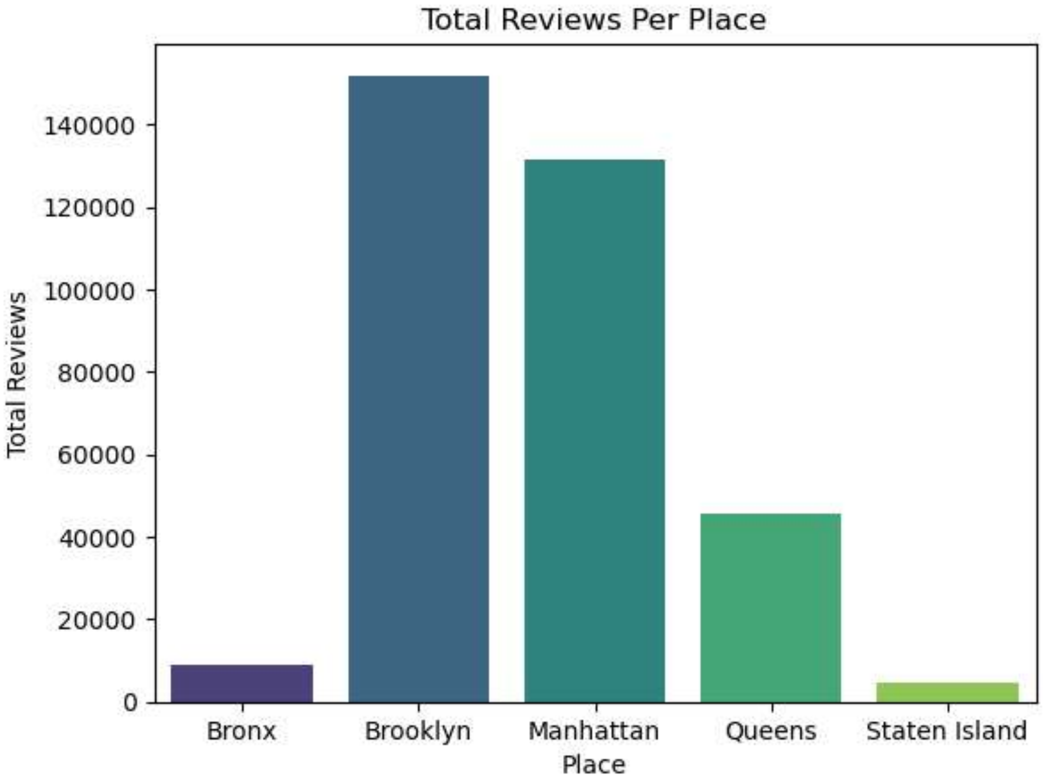


Figure 2: The total number of Airbnb reviews for each location.

Moreover, the popularity of property types is closely tied to the unique characteristics and appeal of different neighbourhoods. Entire homes are prevalent in family-oriented areas like Brooklyn and highly tourist-friendly Manhattan. Private rooms dominate in culturally vibrant and budget-conscious neighbourhoods like northern Brooklyn and Upper

Manhattan. Shared rooms, while less common, are typically found in Upper Manhattan, where affordability plays a significant role.

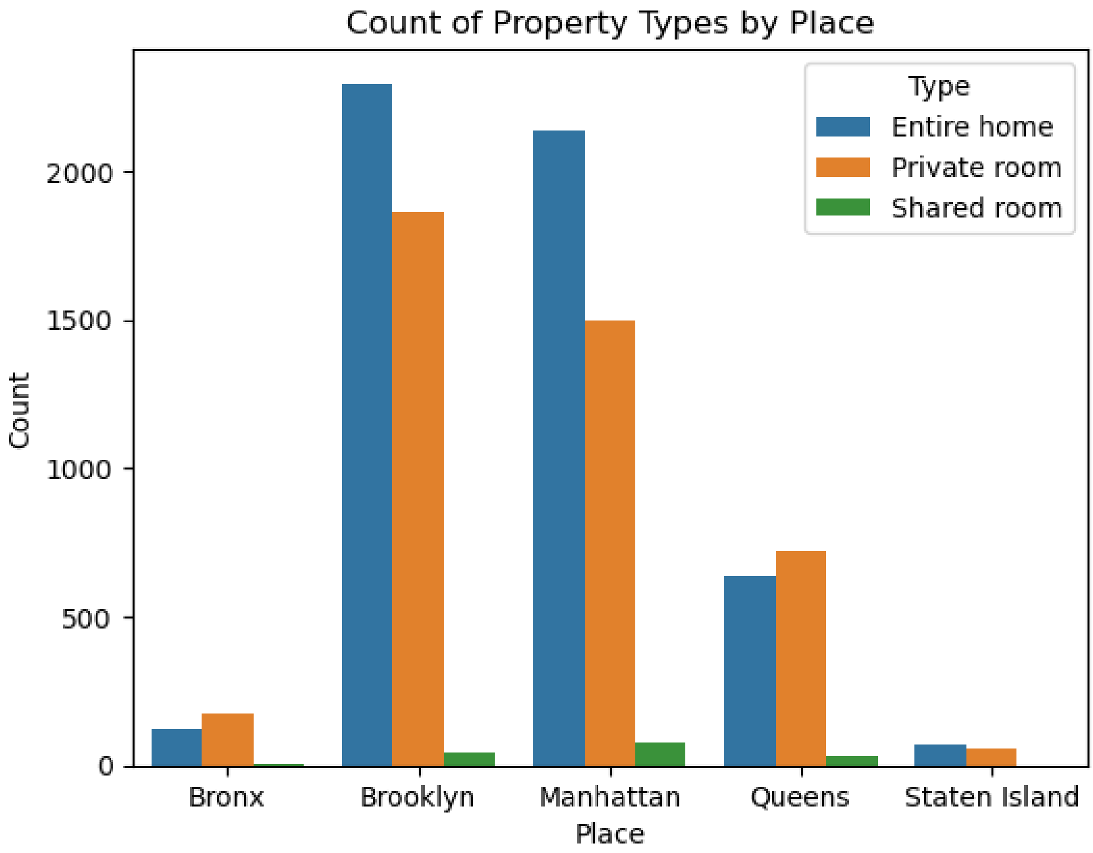


Figure 3: The distribution of Airbnb property types by location.

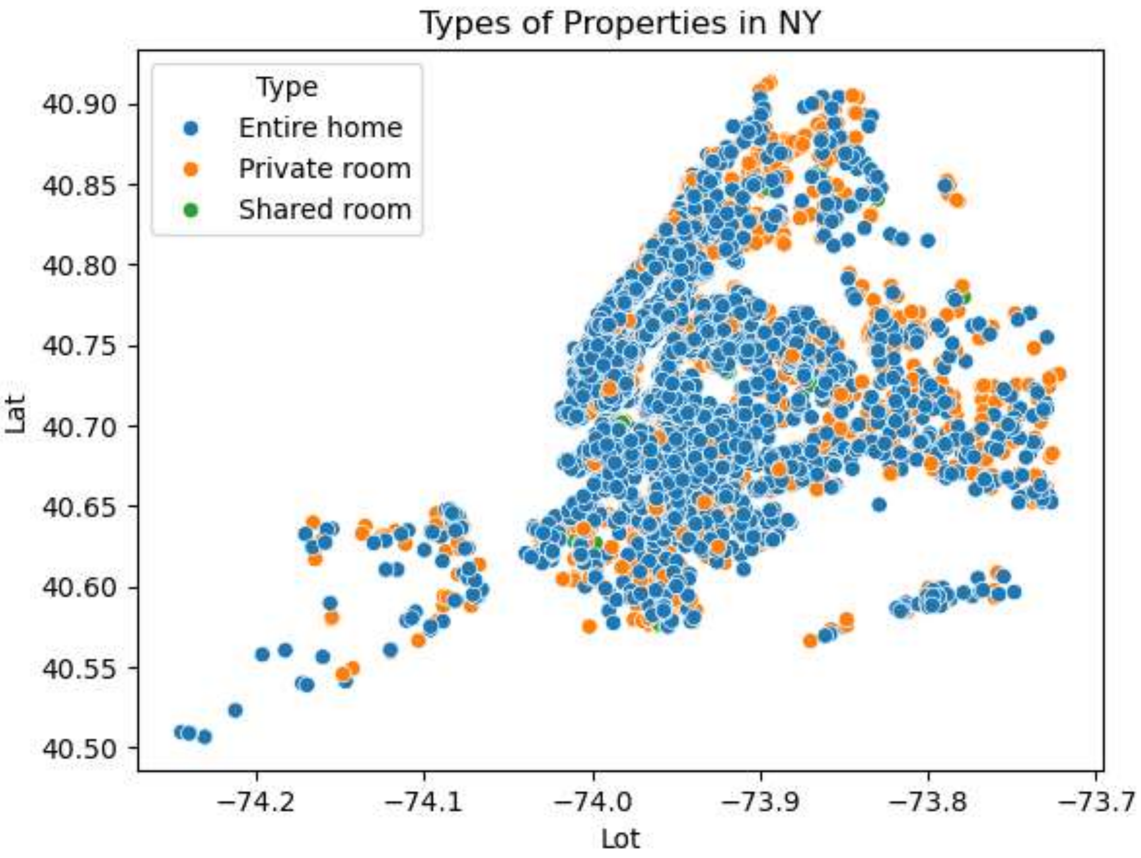


Figure 4: The spatial distribution of Airbnb property types in 2019 across NYC.

3.2 Price Trends by Neighbourhood

Neighborhood	Place	Price (dollars)
Grymes Hill	Staten Island	300.00
Castleton Corners	Staten Island	299.00
Mill Basin	Brooklyn	299.00
Neponsit	Queens	237.00
Tribeca	Manhattan	220.27
NoHo	Manhattan	214.75
Greenwich Village	Manhattan	203.84
Flatiron District	Manhattan	197.78
West Village	Manhattan	195.41
Breezy Point	Queens	195.00

Table 1: Predicted list of the top 10 most expensive neighbourhoods in New York in 2019.

Table 1 shows that Staten Island has some of the highest-priced properties in NYC, though it's generally not more expensive than Manhattan (Bentley, 2024). With a mix of affluent and modest areas, the borough offers suburban charm alongside urban amenities. According to Bentley (2024), neighbourhoods like Greenwich Village, SoHo, and Sutton Place are among the priciest for Airbnb rentals. A 2019 dataset reveals Manhattan has the highest average nightly rate at $180, followed by Brooklyn ($121), Queens ($96), and Staten Island ($90) (NYC_AirBNB_Data, 2021; Sudhakar, 2020).

Staten Island's higher prices in the dataset may be due to sampling bias, with luxury or unique listings inflating averages. Seasonal pricing, misclassification, or data cleaning errors could also distort results, potentially underrepresenting Manhattan's true average. Real-world data consistently shows Manhattan as the most expensive borough, with an average nightly rate of $180 (Bentley, 2024; NYC_AirBNB_Data, 2021). Figure 4 supports this trend.

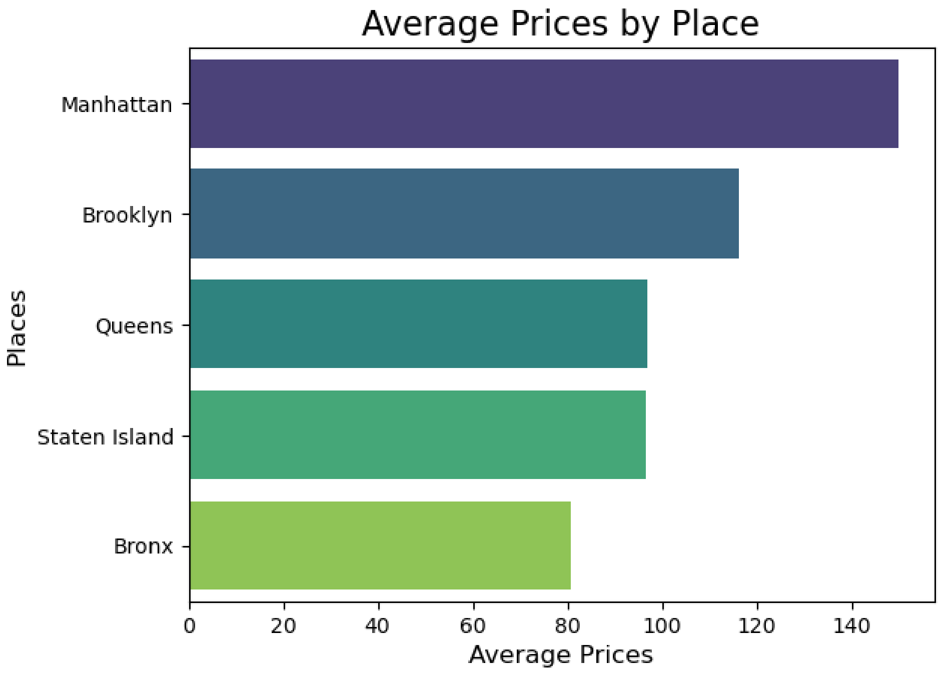


Figure 5: The average predicted price of Airbnb listings in the 5 boroughs of New York in 2019.

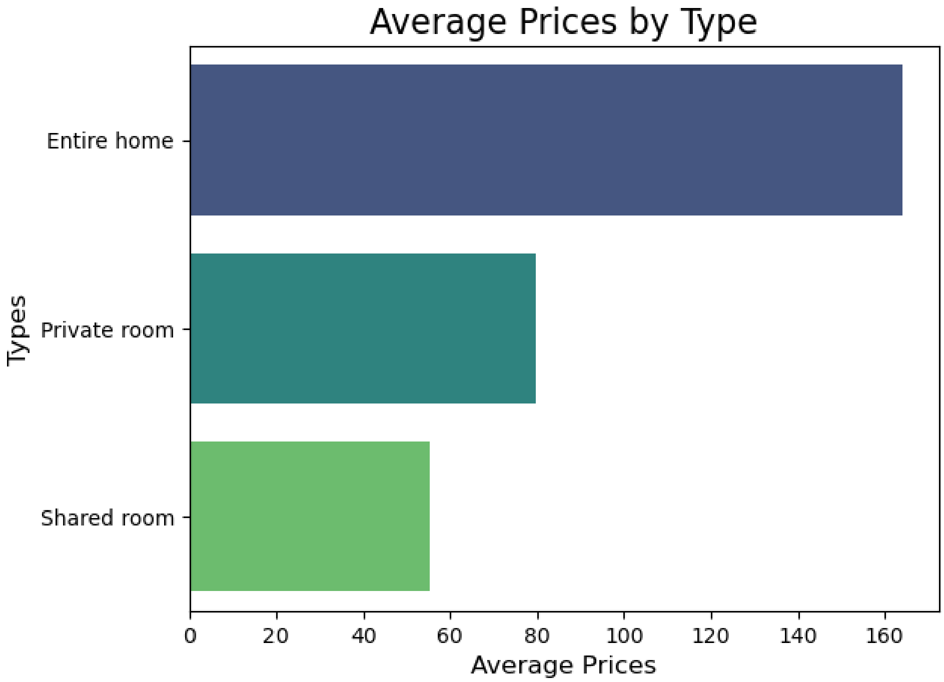


Figure 6: Average predicted price for accommodation on Airbnb in 2019

Figure 6 shows that entire homes on Airbnb are typically more expensive than private or shared rooms, reflecting their full privacy and amenities. Entire homes are ideal for families or groups, while private rooms, with shared common areas, attract solo travellers and couples. Shared rooms are the most affordable, catering to budget-conscious guests. These trends align with market demand, with entire homes more common in tourist areas and private rooms in urban settings (Airbnb, 2024).

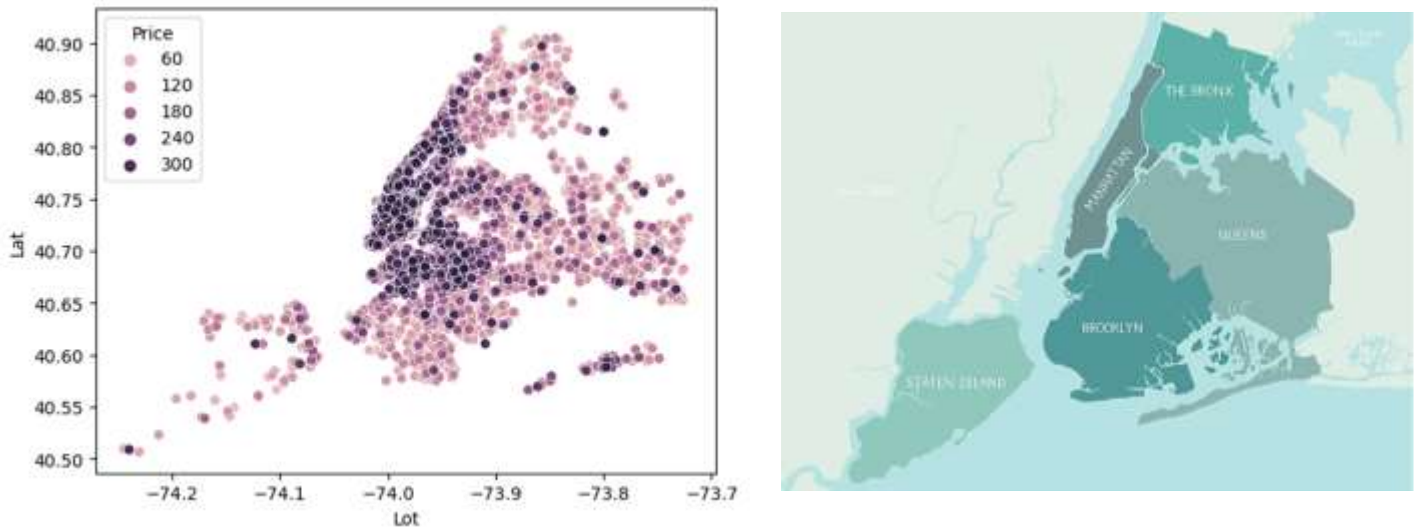


Figure 7: On the left is a scatterplot of New York depicting the predicted Airbnb price listings in 2019. On the right is a map of New York (Stock photo, 2009) highlighting its 5 Boroughs.

Figure 7 shows a concentration of properties around $300 per night in Manhattan, northern Brooklyn, and northwestern Queens. Despite Forest Hills being the priciest area in Queens (Keeling, 2024), the abundance of high-priced listings in the northwestern region is due to its proximity to Manhattan, offering affordable alternatives. Astoria and Long Island City are popular for their vibrant atmospheres, transit access, and proximity to LaGuardia Airport, making them attractive to travellers and renters (McNeil, 2023).

4. Conclusion and Recommendations

The 2019 predictions for Airbnb active listings in New York City, generated through machine learning, revealed key trends across boroughs. Entire homes are expected to remain the most expensive option, dominating tourist-heavy areas like Manhattan and Brooklyn due to the demand for privacy and amenities. Private rooms are predicted to be prevalent in budget-friendly, culturally rich neighbourhoods, while shared rooms will likely attract cost-conscious travellers. The results highlighted a decrease in predicted inactive properties and a shift in market dynamics, emphasising growth in certain boroughs.

Recommendations:

1. **Dynamic Pricing**: Adjust pricing models based on borough-specific trends and property types.
2. **Targeted Marketing**: Focus promotional efforts in growth areas such as northern Brooklyn and Queens.
3. **Data Management**: Continuously update datasets to reflect seasonal changes, outliers, and inactive listings.
4. **Customer Experience**: Enhance features in private rooms to attract solo travellers and couples.

References

Airbnb (2024). *Airbnb categories - Airbnb Help Center*. [online] Airbnb. Available at: https://www.airbnb.com/help/article/3374.

Bentley, A. (2024). *12 Most Expensive Neighborhoods in New York City to Rent in 2024*. [online] Apartment Living Tips - Apartment Tips from ApartmentGuide.com. Available at: https://www.apartmentguide.com/blog/most-expensive-neighborhoods-new-york-ny/ [Accessed 27 Nov. 2024].

Keeling, A. (2024). *The 8 Best Neighborhoods to Live in Queens - Neighbor Blog*. [online] Neighbor Blog. Available at: https://www.neighbor.com/storage-blog/best-neighborhoods-in-queens/ [Accessed 27 Nov. 2024].

McNeil, F.C. (2023). *Airbnb Statistics [2023]: User & Market Growth Data*. [online] Positionly. Available at: https://positionly.com/blog/stats/airbnb-statistics-2023-user-market-growth-data/.

NYC_AirBNB_Data (2021). *GitHub - sevesilvestre/NYC_AirBNB_Data: AIrBNB Dataset Analysis in NYC through Python*. [online] GitHub. Available at: https://github.com/sevesilvestre/NYC_AirBNB_Data?tab=readme-ov-file [Accessed 27 Nov. 2024].

Sarkar, A., Gupta, R., Zhang, Z., & Mukherjee, B. (2020) *Spatial and socioeconomic analysis of host participation in the sharing economy: Airbnb in New York City*. *Information Technology & People*, 33(3), pp. 983–1009. Available at: https://essex.primo.exlibrisgroup.com/permalink/44UOES_INST/o3t9un/cdi_crossref_primary_10_1108_ITP_10_2018_0481.

Stock photo (2009). [online] Istockphoto.com. Available at: https://www.istockphoto.com/illustrations/nyc-borough-map [Accessed 27 Nov. 2024].

Sudhakar, S. (2020). *Analyzing New York City Airbnb Data*. [online] Amazonaws.com. Available at: https://rstudio-pubs-static.s3.amazonaws.com/612101_344e3c29505349a488e9ac2e0fcda856.html [Accessed 27 Nov. 2024].

Appendices:

Appendix A:

<u>Explanation:</u>

In our analysis, we initially aimed to determine the number of properties added over time. However, the last_review variable only indicated how many properties became "inactive" each year, showing the number of listings removed from Airbnb annually. This revealed that every year, some properties stop being active on the platform.

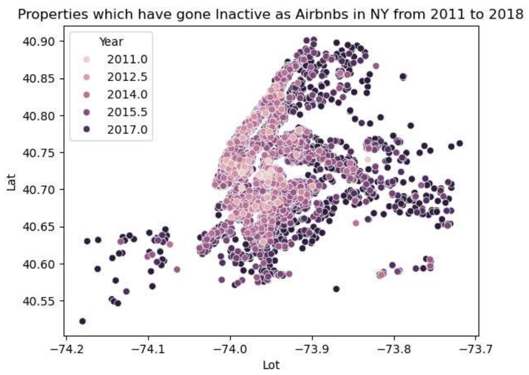
The 2019 dataset includes both active and inactive listings, but we cannot yet determine how many properties will become inactive. Based on historical trends, we anticipate this will occur. We wanted to estimate the number of properties that will become inactive in 2019, enabling us to focus on active listings.

To do this, we grouped properties by their last review year (2011–2018) to understand the number of properties that became inactive annually. We excluded 2019 from this analysis, as we aim to predict inactivity for that year. This will allow us to filter the dataset and focus only on active listings.

We will use a polynomial model to estimate the number of inactive properties in 2019, ensuring an optimal model fit by selecting the right degree using RMSE as an evaluation metric (see appendix B). This approach will result in a more reliable, focused dataset.

```
#Our target is to find the ACTIVE and most Reliable properties
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

#Reading our dataset file
df = pd.read_csv("AB_NYC_2019.csv")

#Converting the last_review column to a Datetime format
```

```
df['last_review'] = pd.to_datetime(df['last_review'])
df['last_review'] = df['last_review'].dt.year

#Replacing a value in a column
df['room_type'] = df['room_type'].replace('Entire home/apt', 'Entire home')

#Renaming columns
df = df.rename(columns={"neighbourhood_group": "Place", "neighbourhood": "Hood",
"latitude": "Lat", "longitude": "Lot", "room_type": "Type", "price":
"Price",'last_review':'Year'})

#Dropping rows with N/A values
df = df.dropna()


#Resetting our index number series everytime after changes
df = df.reset_index(drop=True)

#Finding how many rows and columns our dataset has
df.shape

(38821, 16)

#How to filter out a dataset
df_2018=df[df['Year'] < 2019]
df_2018.shape

(13620, 16)


#Depicting the expansion of inactive properties over the years
sns.scatterplot(df_2018, x = 'Lot', y = 'Lat', hue = 'Year')
```

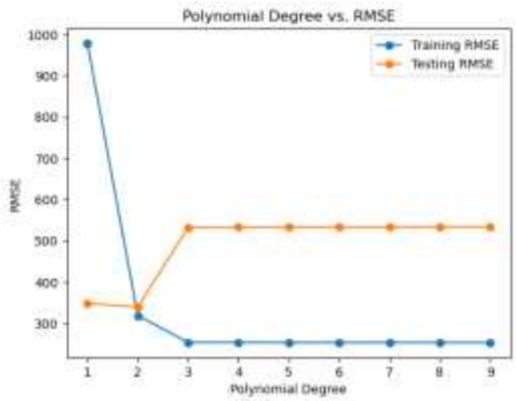


```
#Counting every Place for each Year
df_ml = df_2018.groupby('Year').agg(
    Count_Places=('Place', 'count')
).reset_index()


df_ml = df_ml[['Year', 'Count_Places']]
df_ml
```

```
       Year     Count_Places
 0  2011.0            7
 1  2012.0           25
 2  2013.0           48
 3  2014.0          199
 4  2015.0         1388
 5  2016.0         2703
 6  2017.0         3203
 7  2018.0         6047
```

```
#Applying Polynomial Regression
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split

#Preparing our data
X = df_ml[['Year']]
y = df_ml[['Count_Places']]

#Splitting data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

#Storing our results
degrees = np.arange(1, 10)
test_scores = []
train_scores = []

for degree in degrees:

    #Transforming the data to include polynomial features
    poly = PolynomialFeatures(degree=degree)
    X_poly_train = poly.fit_transform(X_train)
    X_poly_test = poly.transform(X_test)

    #Fitting the polynomial regression model
    model = LinearRegression()
    model.fit(X_poly_train, y_train)

    #Evaluating the model
    predictions_train = model.predict(X_poly_train)
    predictions_test = model.predict(X_poly_test)

    #Calculating RMSE
    train_rmse = np.sqrt(mean_squared_error(y_train, predictions_train))
    test_rmse = np.sqrt(mean_squared_error(y_test, predictions_test))

    train_scores.append(train_rmse)
    test_scores.append(test_rmse)

#Plotting RMSE scores for training and testing
```

```
plt.plot(degrees, train_scores, label='Training RMSE', marker='o')
plt.plot(degrees, test_scores, label='Testing RMSE', marker='o')
plt.title('Polynomial Degree vs. RMSE')
plt.xlabel('Polynomial Degree')
plt.ylabel('RMSE')
plt.legend()
plt.show()


#Choosing the best degree based on RMSE scores
best_degree = degrees[np.argmin(test_scores)]
print(f'The best polynomial degree is: {best_degree}')
```



```
The best polynomial degree is: 2
```

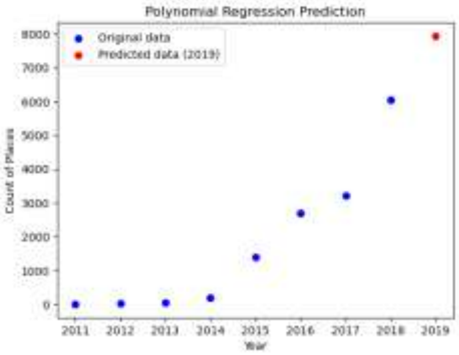
Explanation:

A degree of 2 indicates a quadratic model, which fits the data well without overcomplicating it.

When the degree is larger than 2, the training RMSE becomes lower than the testing RMSE, indicating overfitting. This means that the model has learned the noise in the training data rather than the underlying pattern.

```
#Fitting the final model with the best degree
poly = PolynomialFeatures(degree=best_degree)
X_poly_full = poly.fit_transform(X)

model = LinearRegression()
model.fit(X_poly_full, y)


#Predictions and error metrics
predictions_train = model.predict(X_poly_full)
metrics = {
    'Mean Absolute Error': mean_absolute_error(y, predictions_train),
    'Mean Squared Error': mean_squared_error(y, predictions_train),
    'Root Mean Squared Error': mean_squared_error(y, predictions_train,
squared=False),
    'R Square Score': r2_score(y, predictions_train)
}

for metric, value in metrics.items():
    print(f'{metric}: {value:.8f}')




#Predicting for the year 2019
year_2019 = pd.DataFrame({'Year': [2019]})
year_2019_poly = poly.transform(year_2019)
predictions_2019 = model.predict(year_2019_poly)

predicted_df = pd.DataFrame(predictions_2019, columns=['Count_Places'])
predicted_df['Year'] = 2019

#Adding the prediction to the original DataFrame
df_ml = pd.concat([df_ml, predicted_df], ignore_index=True)

#Plotting original and predicted data
plt.scatter(X, y, color='blue', label='Original data')
plt.scatter(predicted_df['Year'], predicted_df['Count_Places'], color='red',
label='Predicted data (2019)')
plt.title('Polynomial Regression Prediction')
plt.xlabel('Year')
plt.ylabel('Count of Places')
plt.legend()
plt.show()

print(df_ml)
```

```
Mean Absolute Error: 255.40178567
Mean Squared Error: 97763.31844181
Root Mean Squared Error: 312.67126258
R Square Score: 0.97623694
```



```
    Year  Count_Places
0  2011.0     7.000000
1  2012.0    25.000000
2  2013.0    48.000000
3  2014.0   199.000000
4  2015.0  1388.000000
5  2016.0  2703.000000
6  2017.0  3203.000000
7  2018.0  6047.000000
8  2019.0  7935.749999
```

```
#Machine Learning predicted that we will have 7936 inactive properties in the year of 2019.
#So, we have to remove the exact number of rows in the year of 2019 RANDOMLY,
#in order to keep only the active ones.

#Removing the Predicted number of inactive properties Randomly
df_2019 = df[df['Year'] == 2019]

if len(df_2019) >= 7936:
    predicted_values_remove = df_2019.sample(n=7936, random_state=1)
    df_2019 = df_2019.drop(predicted_values_remove.index)
else:
    print("Not enough rows to remove 7936 entries.")

df_2019 = df_2019.dropna()
df_2019 = df_2019.sort_values(by=['Year'], ascending=False)
df_2019 = df_2019.reset_index(drop=True)
df_2019.shape
```

```
(17265, 16)
```

```
#Applying all the appropriate filters (We review all the histograms and the distribution of those numeric variables)

#We want properties that are available at least one day of the year
df_2019 = df_2019[df_2019['minimum_nights'] < 365]

#We want properties that works and have at least 1 review
df_2019 = df_2019[df_2019['number_of_reviews'] > 0]

#The same as above. We want active properties
df_2019 = df_2019[df_2019['reviews_per_month'] > 0]

#We want available and active properties. We do not want properties at pause
```

```
df_2019 = df_2019[df_2019['availability_365'] > 0]

#Properties that are available more than 365 days are faulty data
df_2019 = df_2019[df_2019['availability_365'] < 365]

#Properties with zero value price do not exist. Those are also faulty data
df_2019 = df_2019[df_2019['Price'] > 0]

#We do not want rows with N/A numbers
df_2019 = df_2019.dropna()

#That is how we reset our indexing numbers after every change in our data
df_2019 = df_2019.reset_index(drop=True)

#The shape gives as the number of rows and columns in a table
df_2019.shape
```

```
(14492, 16)
```

```
#Removing all the outliers from numerical column variables

Q1 = df_2019['minimum_nights'].quantile(0.25)
Q3 = df_2019['minimum_nights'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_2019 = df_2019[(df_2019['minimum_nights'] >= lower_bound) &
(df_2019['minimum_nights'] <= upper_bound)]

Q1 = df_2019['reviews_per_month'].quantile(0.25)
Q3 = df_2019['reviews_per_month'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
```

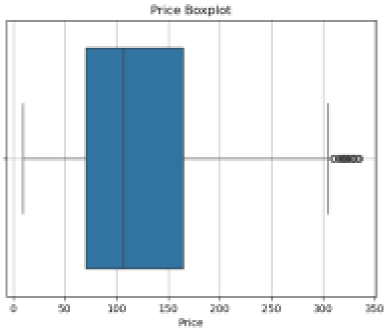
```
upper_bound = Q3 + 1.5 * IQR
df_2019 = df_2019[(df_2019['reviews_per_month'] >= lower_bound) &
(df_2019['reviews_per_month'] <= upper_bound)]

Q1 = df_2019['calculated_host_listings_count'].quantile(0.25)
Q3 = df_2019['calculated_host_listings_count'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_2019 = df_2019[(df_2019['calculated_host_listings_count'] >= lower_bound) &
(df_2019['calculated_host_listings_count'] <= upper_bound)]

Q1 = df_2019['number_of_reviews'].quantile(0.25)
Q3 = df_2019['number_of_reviews'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_2019 = df_2019[(df_2019['number_of_reviews'] >= lower_bound) &
(df_2019['number_of_reviews'] <= upper_bound)]

Q1 = df_2019['Price'].quantile(0.25)
Q3 = df_2019['Price'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df_2019 = df_2019[(df_2019['Price'] >= lower_bound) & (df_2019['Price'] <=
upper_bound)]

df_2019 = df_2019.reset_index(drop=True)

df_2019.shape
```

```
(9733, 16)
```

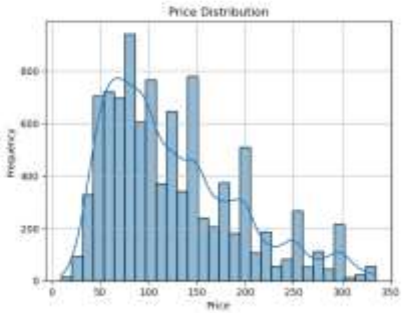
```
#How to count a categorical variable
df_2019['Place'].value_counts()
```

```
Place
Brooklyn       4204
Manhattan       3710
Queens          1390
Bronx           306
Staten Island     123
```

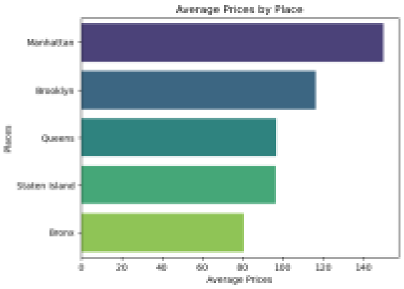
```
#How to check out for outliers in a numerical variable
sns.boxplot(x=df_2019['Price'])
plt.title('Price Boxplot')
plt.xlabel('Price')
plt.grid()
plt.show()
```



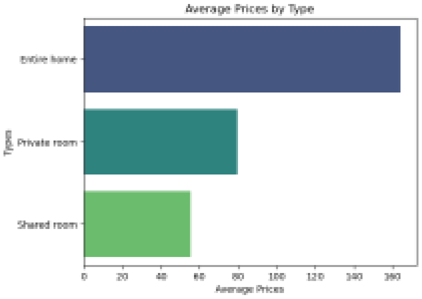
```
#How to view a numerical variable distribution with Histogram
sns.histplot(df_2019['Price'], bins=30, kde=True)
plt.title('Price Distribution')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.grid()
plt.show()
```



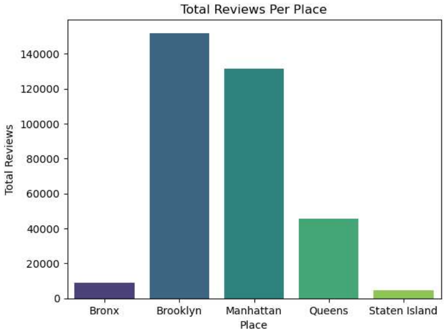
```
#How to find out the average Prices by Place
avg_prices_places = df_2019.groupby('Place')['Price'].mean().reset_index()
avg_prices_places = avg_prices_places.sort_values(by='Price', ascending=False)

ax = sns.barplot(x='Price', y='Place', data=avg_prices_places, palette='viridis')
ax.set_title('Average Prices by Place')
ax.set_xlabel('Average Prices')
ax.set_ylabel('Places')
plt.show()
```



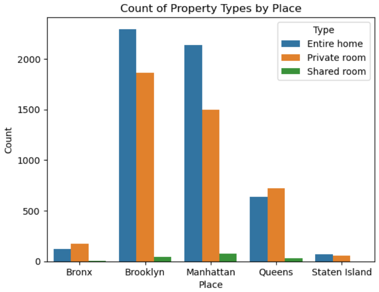
```
#How to find out the average Prices by Type of property
avg_prices_types = df_2019.groupby('Type')['Price'].mean().reset_index()
avg_prices_types = avg_prices_types.sort_values(by='Price', ascending=False)

ax_T = sns.barplot(x='Price', y='Type', data=avg_prices_types, palette='viridis')
ax_T.set_title('Average Prices by Type')
ax_T.set_xlabel('Average Prices')
ax_T.set_ylabel('Types')
plt.show()
```



```
#How to find out the Total Reviews per Place
total_reviews = df_2019.groupby('Place')['number_of_reviews'].sum().reset_index()

sns.barplot(x='Place', y='number_of_reviews', data=total_reviews, palette='viridis')
plt.title('Total Reviews Per Place')
plt.xlabel('Place')
plt.ylabel('Total Reviews')
plt.show()
```



```
#How to find out the Total number of properties by Place
type_counts_place = df_2019.groupby(['Place',
'Type']).size().reset_index(name='Count')

sns.barplot(data=type_counts_place, x='Place', y='Count', hue='Type')
plt.title('Count of Property Types by Place')
plt.ylabel('Count')
plt.xlabel('Place')
plt.legend(title='Type')
plt.show()
```



```
df_2019 = df_2019.reset_index(drop=True)
```

```
df_2019.shape
```

```
(9733, 16)
```

```
#Top 10 hoods with higher reviews
top_review_hoods =
df_2019.groupby(['Hood','Place'])['number_of_reviews'].sum().reset_index()
top_review_hoods = top_review_hoods.sort_values(by='number_of_reviews',
ascending=False)
print(top_review_hoods.head(10))
```

```
                  Hood      Place  number_of_reviews
12   Bedford-Stuyvesant   Brooklyn              33568
90               Harlem  Manhattan              25825
201        Williamsburg   Brooklyn              24913
27             Bushwick   Brooklyn              15299
91       Hell's Kitchen  Manhattan              13265
59          East Harlem  Manhattan              11627
49       Crown Heights   Brooklyn              11181
62         East Village  Manhattan              10814
189     Upper East Side  Manhattan               9265
190     Upper West Side  Manhattan               8246
```

```
#Top 10 hoods with the most properties with their average prices
top_property_hoods = df_2019.groupby(['Hood', 'Place']).agg(count=('Hood', 'size'),
average_price=('Price', 'mean')).reset_index()
top_property_hoods = top_property_hoods.sort_values(by='count', ascending=False)
print(top_property_hoods.head(10))
```

```
                   Hood      Place  count  average_price
12   Bedford-Stuyvesant   Brooklyn    863     108.713789
201         Williamsburg   Brooklyn    772     135.865285
90                Harlem  Manhattan    639     113.215962
27              Bushwick   Brooklyn    453      92.450331
91        Hell's Kitchen  Manhattan    367     174.651226
62          East Village  Manhattan    324     163.194444
49         Crown Heights   Brooklyn    313     117.578275
59           East Harlem  Manhattan    281     129.387900
189      Upper East Side  Manhattan    278     151.377698
190      Upper West Side  Manhattan    242     162.260331
```

```
#Top 10 hoods with the most properties with their Type
top_type_hoods = df_2019.groupby(['Hood', 'Place', 'Type']).size().unstack().fillna(0)
top_type_hoods['Total'] = top_type_hoods[['Entire home', 'Private room', 'Shared
room']].sum(axis=1)
top_type_hoods = top_type_hoods.sort_values(by='Total', ascending=False)
print(top_type_hoods.head(10))
```

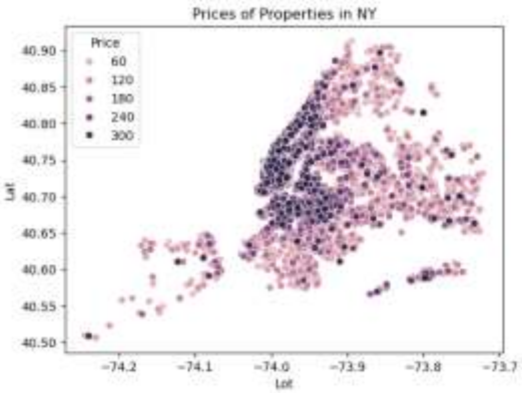
```
Type                          Entire home  Private room  Shared room  Total
Hood               Place
Bedford-Stuyvesant Brooklyn         475.0         379.0          9.0  863.0
Williamsburg       Brooklyn         412.0         353.0          7.0  772.0
Harlem             Manhattan        266.0         359.0         14.0  639.0
Bushwick           Brooklyn         160.0         291.0          2.0  453.0
Hell's Kitchen     Manhattan        234.0         128.0          5.0  367.0
East Village       Manhattan        221.0          99.0          4.0  324.0
Crown Heights      Brooklyn         188.0         120.0          5.0  313.0
```

```
East Harlem       Manhattan       138.0        136.0          7.0  281.0
Upper East Side    Manhattan       180.0         91.0          7.0  278.0
Upper West Side    Manhattan       142.0          95.0          5.0  242.0
```

```
#Top 10 most expensive hoods
expensive_hoods = df_2019.groupby(['Hood','Place'])['Price'].mean().reset_index()
expensive_hoods = expensive_hoods.sort_values(by='Price', ascending=False)
print(expensive_hoods.head(10))
```

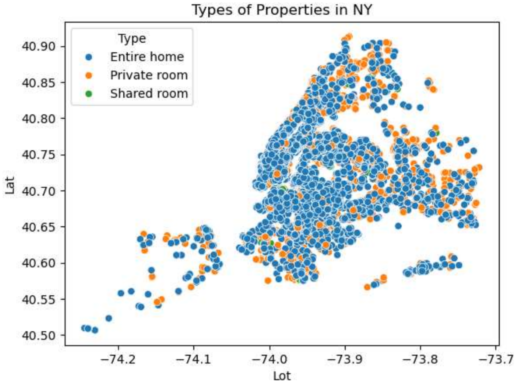
```
                  Hood          Place       Price
89         Grymes Hill  Staten Island  300.000000
32   Castleton Corners  Staten Island  299.000000
123          Mill Basin       Brooklyn  299.000000
133            Neponsit         Queens  237.000000
185             Tribeca      Manhattan  220.266667
137                NoHo      Manhattan  214.750000
88   Greenwich Village      Manhattan  203.838710
72   Flatiron District      Manhattan  197.777778
197        West Village      Manhattan  195.411290
20         Breezy Point         Queens  195.000000
```

```
#Depicting the expansion of property prices
sns.scatterplot(df_2019, x = 'Lot', y = 'Lat', hue = 'Price')
```



```
#Depicting the expansion of property types
```

```
sns.scatterplot(df_2019, x = 'Lot', y = 'Lat', hue = 'Type')
```



Appendix B:

Predicting Inactive Properties for 2019 Using Trend Analysis

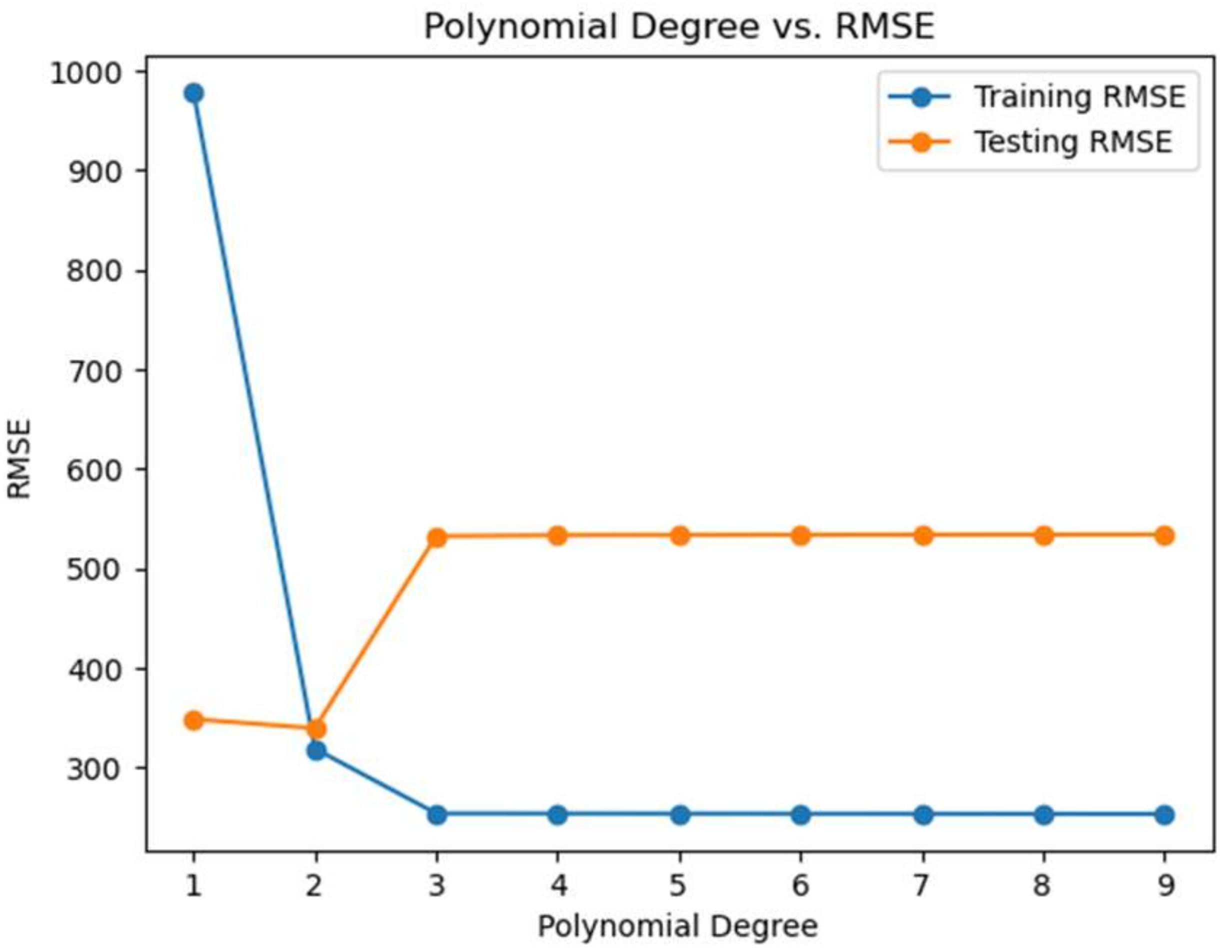
Using the trend graph and regression analysis, we extended our analysis to predict the number of inactive properties in 2019.

1. **Regression Model**:

We applied a **second-degree polynomial regression model**. The model was chosen due to its ability to capture the non-linear trends observed in the data over time and fits the data well without overcomplicating it.

1. **Model Evaluation**:

RMSE (Root Mean Squared Error) is a commonly used metric for evaluating the performance of a regression model. It measures the average magnitude of the prediction errors, giving more weight to larger errors due to the squaring step. It is particularly useful for understanding how well a regression model's predictions align with the observed data.



By analysing the plot, it is evident that the training and testing RMSE converge at the 2nd polynomial degree, indicating that this is the optimal choice.

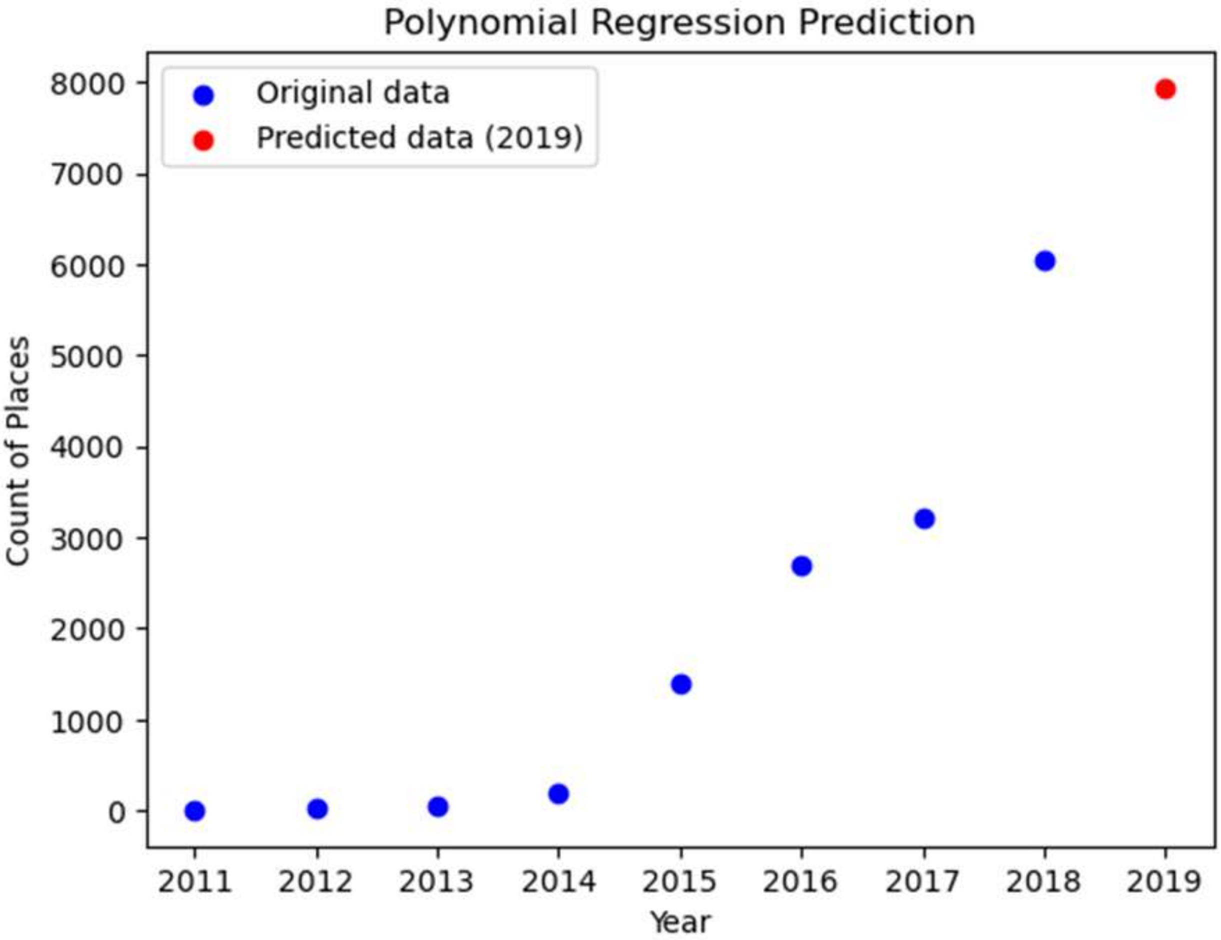
When the degree is larger than 2, the training RMSE becomes lower than the testing RMSE, indicating overfitting. This means that the model has learned the noise in the training data rather than the underlying pattern.

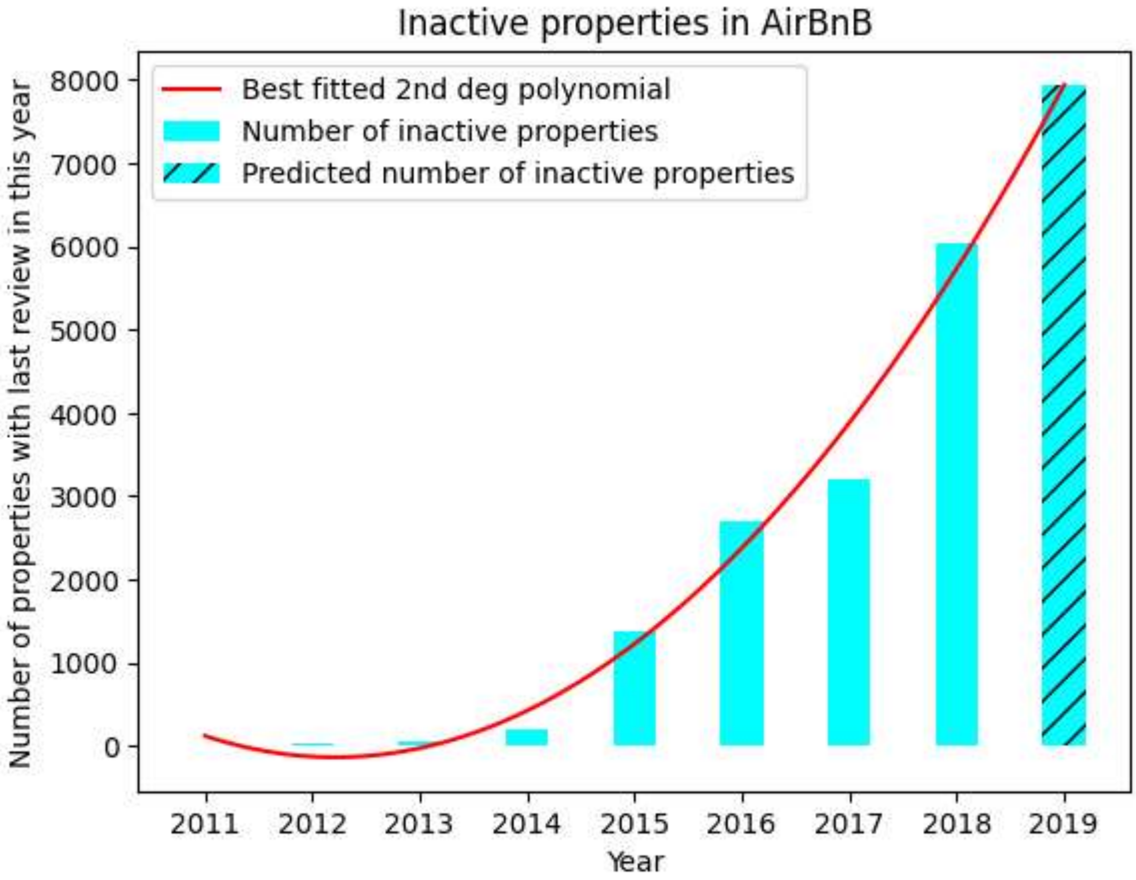
The calculated **R^2** (R-squared or the coefficient of determination: a statistical measure that indicates how well a regression model fits the observed data. R2 ranges between 0 and 1; 1 indicates perfect fit) value for the model was very close to 1, indicating a strong fit and that the polynomial curve effectively represents the data used to construct it. This high R^2 value gave us confidence in the reliability of the predictions.

R Square Score: 0.97623694

3. Predictions:

Using the fitted model, we predicted the number of properties that would become inactive by the end of 2019. **7935.749999**





Machine Learning predicted that there will be 7936 inactive properties in 2019. To reflect this prediction, we randomly removed the exact number of rows corresponding to inactive properties from the 2019 dataset, leaving only the active properties. The updated dataset now contains 17265 rows (for a clearer understanding of the data cleaning process, you can refer to Panagiotis's code, which provides a detailed and structured approach to this task)

With this cleaned data, we can analyse the location and price distributions for active properties in 2019.

Bibliography

Airbnb (2020). *How to set a pricing strategy - Resource Center*. [online] Airbnb. Available at: https://www.airbnb.com/resources/hosting-homes/a/how-to-set-a-pricing-strategy-15.

Airbnb. “New Survey Finds Strict Short-Term Rental Rules Deter Visitors to NYC.” Airbnb Newsroom, 8 Dec. 2023, news.airbnb.com/new-survey-finds-strict-short-term-rental-rules-deter-visitors-to-nyc/.

Airbnb.com (no date) *Fast facts*. Available at: https://news.airbnb.com/about-us. Accessed: 10th November 2024.

Nascimento, J.L. & Mazali, R. (2023) *Technological innovations and preexisting markets: The interaction between Airbnb and New York’s hotel and housing markets*. *Regional Science Policy & Practice*, 15(2), pp. 256–287. Available at: https://essex.primo.exlibrisgroup.com/permalink/44UOES_INST/o3t9un/cdi_crossref_primary_10_1111_rsp3_12584.

AirDNA. “AirDNA: 2024 Outlook Report.” Www.airdna.co, 2024, www.airdna.co/outlook-report.

Battaglia, Evelyn. “An Insider’s Guide to Finding a Short-Term, Furnished Apartment Rental in NYC.” Brick Underground, 26 Dec. 2023, www.brickunderground.com/rent/guide-to-finding-short-term-furnished-apartment-rental-NYC. Accessed 11 Nov. 2024.

Godoy, Janik. “New York City Visitor Statistics and Tourism Figures 2022.” Family Destinations Guide, 30 Mar. 2022, familydestinationsguide.com/new-york-city-visitor-statistics-and-tourism-figures/. Accessed Nov. 11, 2024.

Hostfully (2024). *7 Strategies To Optimize Your Airbnb Pricing & How To Set Your Rates*. [online] Hostfully. Available at: https://www.hostfully.com/blog/airbnb-pricing-strategies/.

Hoyt, A. (2021). *Why Have Airbnb, VRBO Prices Suddenly Skyrocketed?* [online] HowStuffWorks. Available at: https://money.howstuffworks.com/airbnb-vrbo-prices-skyrocketed-news.htm.

McKinsey & Company. “Accelerating the Recovery of NYC’s Travel Sector | McKinsey.” Www.mckinsey.com, 21 Dec. 2021, www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/new-york-a-concrete-jungle-where-dreams-are-still-made.

Mwigeka, Samwel. (2022). What Factors drives the Airbnb Listing’s Prices?. International Business & Economics Studies. 4. p26. 10.22158/ibes.v4n1p26.

New York City Tourism and Conventions. “Annual Report 2023–24 | New York City Tourism + Conventions.” Corporate.nyctourism.com, Jan. 2024, corporate.nyctourism.com/annual-report/2024.

Office of the New York State Comptroller. “The Tourism Industry in New York City | Office of the New York State Comptroller.” Www.osc.ny.gov, Apr. 2021, www.osc.ny.gov/reports/osdc/tourism-industry-new-york-city. Accessed 11 Nov. 2024.

Stabrowski, F. (2022) *Political organizing and narrative framing in the sharing economy: Airbnb host clubs in New York City*. *City*, 26(1), pp. 142–159. Available at: https://essex.primo.exlibrisgroup.com/permalink/44UOES_INST/o3t9un/cdi_crossref_primary_10_1080_13604813_2021_2018853.

Xie, K.L., Kwok, L., & Wu, J. (2020) *Are neighbors friends or foes? Assessing Airbnb listings’ agglomeration effect in New York City*. *Cornell Hospitality Quarterly*, 61(2), pp. 128–141. Available at: https://essex.primo.exlibrisgroup.com/permalink/44UOES_INST/o3t9un/cdi_crossref_primary_10_1177_1938965519890578.