**Airbnb Data Analysis**

*Group 3, University of Essex Online*

Background

Airbnb's business model is connecting people who want to rent their home or room with people who want to get accommodation (Folger, 2022). Airbnb is partnering with the community where Airbnb basically doesn’t have its own accommodation asset (Fox, 2020). An Airbnb dataset will be analyzed to achieve this goal to strengthen its position and competitiveness in the market, especially in New York City.

Business Analytics

To strengthen the competitiveness in the market, the analysis will start with what kind of room type has the highest number in each location. The price average and range need to be analyzed to understand the market price in each area. And which method can be used for the price prediction? Those factors are required to make the strategic decision on the partnership, such as choosing room type and the price range in partnering marketing or commission structure.

Data Analysis

The Airbnb dataset is taken from Kaggle (Dgmonov, 2019) which contains information such as hostname, location, area, coordinates, room type, price, minimum stays, number of reviews, and availability in a year. The data analysis methodology is divided into four sections, explanatory data analysis (EDA), data pre-processing, statistical analysis, and machine learning methods.

## Explanatory Data Analysis

Explanatory data analysis (EDA) is the initial stage to make a hypothesis based on the dataset, and pattern, and explore for any anomalies, using the statistical method and visual representation of the dataset (Patil, 2018).

Chart, pie chart

Description automatically generated Chart, pie chart

Description automatically generated

Figure 1. Host Distribution based on location and room type (Appendix 2)

The figure above shows that Manhattan has the highest number of hosts, and entire home or apartment is the highest type that Airbnb offers.

Chart, bar chart

Description automatically generated

Figure 2. Distribution of room type on each location (Appendix 3)

In Manhattan which has the highest host number, the entire home/apartment is the highest proportion in the location. But unlike Manhattan, private rooms become the highest number of hosts in the other area.



Figure 3. Comparison between room type and price in the neighborhood group (Appendix 4)

The highest price is at the level of 10,000 USD for the private room and entire home/apartment. This price is available in Manhattan, but for a shared room, Queens has the highest range of accommodation prices. Overall, Manhattan has the highest price among other places.



Figure 4. Distribution of host based on latitude and longitude (Appendix 5)

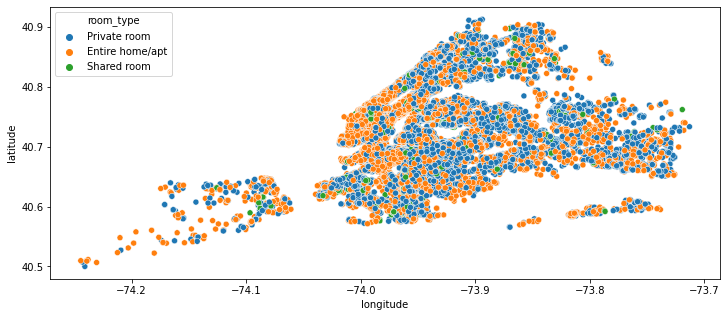


Figure 5. Distribution of room type based on latitude and longitude (Appendix 5)

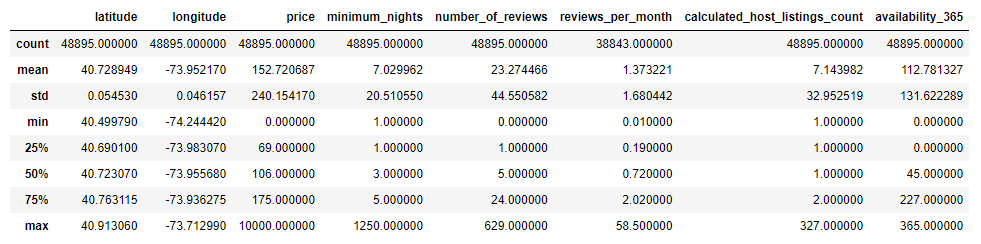


Table 1. Central tendency and variability (Appendix 6)

The table above shows basic statistical analysis including mean, standard deviation, minimum, maximum, and percentage quartile. The below boxplot shows the price range for each area.

Chart, bar chart, box and whisker chart

Description automatically generated

Figure 6. Boxplot availability in a year (Appendix 7)

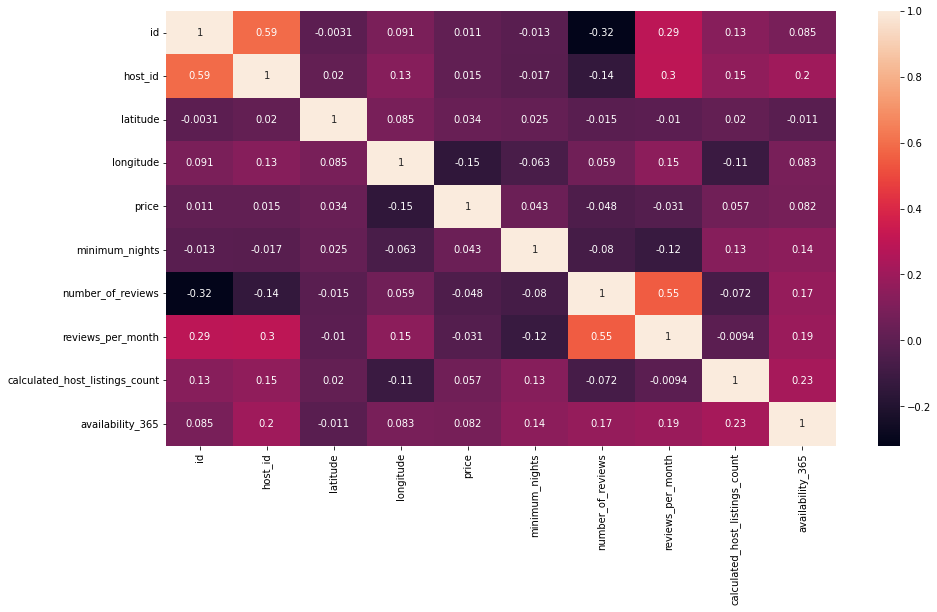


Figure 7. Heatmap correlation matrix (Appendix 8)

The heat-map above shows no strong or positive correlation between variables. Since there is no strong relationship between variables, the calculation would happen independently.

## Data Pre-processing

At this stage, data will be cleaned if any input error, a data type error, a format error, or any outlier where makes the analysis not accurate (Bell, 2020: 26-33). For the analysis, id, host id, hostname, name, latitude, longitude, and last review, are dropped from the dataset (Appendix 10).

The data cleansing step includes checking the null value where there are 10,052 records for reviews per month are null (Appendix 9). The value is replaced with zero indicating that there is no review for the null value. Figure 3 and Table 1 show the discrepancy in price, where the outlier should be cleaned for more accurate analysis. Based on the boxplot in figure 6, the outlier is removed where the price range was set between 0 USD and 350 USD.

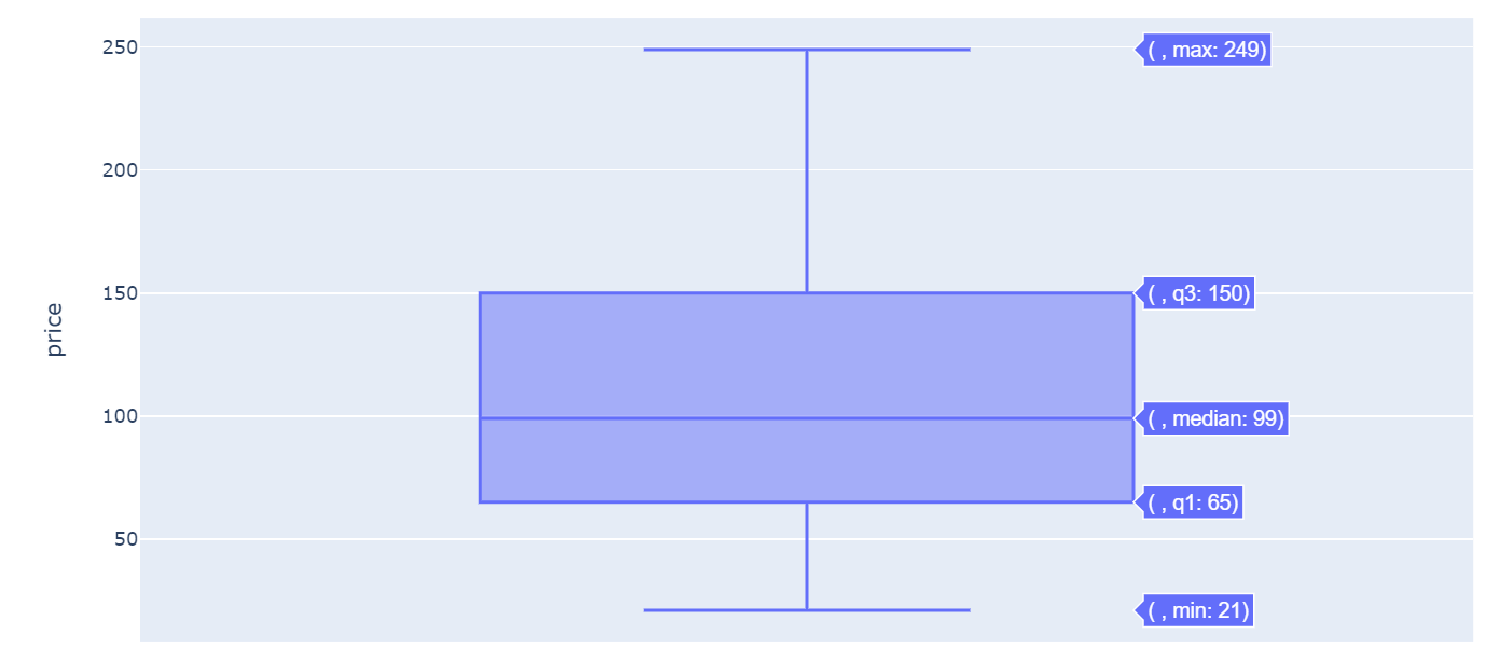


Figure 8. Boxplot price after outlier cleansing (Appendix 10)

After the data cleansing, the minimum price is 21 USD**,** and the maximum price is 249 USD.

## Multilinear Regression Method

In this section, the price is estimated using the multilinear regression method where the price is set as the dependent variable, while other information is set as independent variables. A relationship between variables where one variable is dependent on the change of another variable is called linear regression (Bruce, et al, 2020: 141). When the independent variable is more than one, it is called multiple linear regression (Appendix 12).

Table

Description automatically generated

Table 2. Multilinear Prediction (Appendix 11)

## The table result above uses the OLS method or Ordinary Least Square where this method minimizes the line best fit from the error (Bronshtein, 2017). The result of the R-squared calculation is 0.46 which is relatively low accuracy but still moderate. R-squared normally ranges between 0 and 1 (Bruce, et al, 2020: 154). The OLS result shows the constants and coefficients for the multilinear calculation for the prediction calculation. This is related to the heatmap shown in figure 7, that there is not much correlation between variables where it would impact the accuracy.

## K-Nearest Neighbor Clustering

K-Nearest Neighbor (KNN) Clustering is one of the unsupervised machine learning classifications. It classified unlabeled data and grouped those unlabeled data into a class (Srivastava, 2020). Based on the test using the Airbnb dataset, the KNN method shows a relatively high accuracy of 0.91 and an AUC score of 0.91.

Table

Description automatically generated

Table 3 Classification Result (Appendix 13)

The accurate prediction, AUC, or area under the ROC curve, will be close to one, and less accurate if the AUC value is close to zero (Google, ND). The test classification is using availability per 365 days as the classifier.

Conclusion

Based on the data analysis, Manhattan has the highest host number in the region. Entire homes/apartments also have the highest distribution. The price range within New York City has the range between 21-249 USD. This price range could be the consideration and suggestion to the newly joining Airbnb partners. Manhattan is also one of the regions which have the highest price.

To predict the price, multilinear regression is used to analyze the price with moderate accuracy. For classification, the KNN method is used. KNN test shows that it has a high accuracy to classify the dataset. There are many methods that can be used for prediction and classification. For further improvement, other methods can be used and compare the accuracy result to determine which one is better.

References

Bell, J. (2020) *Machine learning: Hands-on for developers and technical professionals*. Indianapolis: Wiley.

Bronshtein, A. (2017) *Simple and multiple linear regression in python*, *Toward Data Science*. Available at: https://towardsdatascience.com/simple-and-multiple-linear-regression-in-python-c928425168f9 (Accessed: October 20, 2022).

Bruce, P.C., Bruce, A. and Gedeck, P. (2020) *Practical statistics for data scientists: 50+ essential concepts using r and python*. Sebastopol, CA: O'Reilly.

Dgomonov. (2019) *New York City airbnb open data*, *Kaggle*. Available at: https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data (Accessed: October 1, 2022).

Folger, J. (2022) *How Airbnb Works*. [online] Investopedia. Available at: <https://www.investopedia.com/articles/personal-finance/032814/pros-and-cons-using-airbnb.asp> [Accessed 15 October 2022].

Fox, G. (2020) *Airbnb business model - how Airbnb makes money?* *Gary Fox*. Available at: https://www.garyfox.co/airbnb-business-model/ [Accessed: October 28, 2022].

Google*. (ND) Classification: Roc curve and AUC*. Google. Available at: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc#:~:text=AUC%20represents%20the%20probability%20that,has%20an%20AUC%20of%201.0. (Accessed: October 30, 2022).

Patil, P. (2018) *What is Explanatory Data Analysis?* [online] Medium. Available at: <https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15> [Accessed 16 October 2022].

Srivastava, T. (2020) *K nearest neighbor: Knn algorithm: KNN in Python & R*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/ (Accessed: October 30, 2022).

Appendices

# Appendix 1

The libraries used for the data processing such as pandas, seaborn, matplotlib, numpy, and sklearn.

dt **=** pd**.**read\_csv('AB\_NYC\_2019.csv')

dt**.**head()

dt**.**info()

The structure of the data including the data type executed using info() method.

Table

Description automatically generated

# Appendix 2

The below code displays the pie chart to get the proportion of neighborhood groups.

fig **=** plt**.**figure(figsize**=**(5,5), dpi**=**80)

dt['neighbourhood\_group']**.**value\_counts()**.**plot(kind**=**'pie', autopct**=**'%1.0f%%', startangle**=**360, fontsize**=**13)

The following codes display the pie chart to get the proportion of room type.

fig **=** plt**.**figure(figsize**=**(5,5), dpi**=**80)

dt['room\_type']**.**value\_counts()**.**plot(kind**=**'pie', autopct**=**'%1.0f%%', startangle**=**360, fontsize**=**13)

# Appendix 3

Stack bar chart group by the neighborhood group for each room type generated by the below codes.

df **=** (dt**.**groupby('neighbourhood\_group')['room\_type']

**.**value\_counts(normalize**=True**)

**.**mul(100)

**.**round(2)

**.**unstack())

fig, ax **=** plt**.**subplots(figsize **=** (12,6))

df**.**plot(kind**=** "bar",

stacked **=** **True**,

ax **=** ax,

width **=** 0.3,

edgecolor **=** "black")

This part of code will display the label of the bar.

**for** x **in** ax**.**containers:

labels **=** [str(round(v**.**get\_height(), 2)) **+** "%" **if** v**.**get\_height() **>** 0 **else** '' **for** y **in** x]

ax**.**bar\_label(x,

label\_type**=**'center',

labels **=** labels,

size **=** 14)

**for** i **in** ["top", "right"]:

ax**.**spines[i]**.**set\_visible(**False**)

*# Adding tick and axes labels*

ax**.**tick\_params(labelsize **=** 14, labelrotation **=** 0)

ax**.**set\_ylabel("Percentage", size **=** 14)

ax**.**set\_xlabel("Neighbourhood Group", size **=** 14)

*# Fixing legend position*

ax**.**legend\_**.**set\_bbox\_to\_anchor([0.99, 0.8])

# Appendix 4

Below is the scatterplot code to compare the price with the room type within the neighborhood group.

plt**.**figure(figsize**=**(10,10))

sb**.**scatterplot(x**=**"room\_type",

y**=**"price",

hue**=**"neighbourhood\_group",

size**=**"neighbourhood\_group",

sizes**=**(50, 150),

palette**=**"Dark2", data**=**dt)

plt**.**xlabel("Room Type", size**=**10)

plt**.**ylabel("Price", size**=**10)

plt**.**title("Room Type vs Price in the Neighbourhood Group",size**=**15, weight**=**'bold')

# Appendix 5

The below codes show the distribution of room type and the neighborhood group based on the coordinates.

plt**.**figure(figsize**=**(12,5))

sb**.**scatterplot(x **=** dt['longitude'], y **=** dt['latitude'], hue **=** dt['neighbourhood\_group'])

plt**.**show()

plt**.**figure(figsize**=**(12,5))

sb**.**scatterplot(x **=** dt['longitude'], y **=** dt['latitude'], hue **=** dt['room\_type'])

plt**.**show()

# Appendix 6

Statistical information such as mean, standard deviation, minimum and maximum, or quartile, in python may use describe() method.

x **=** dt**.**iloc[:,4:16]

x**.**describe()

# Appendix 7

The availability for a year is presented as the boxplot to display the minimum, maximum, and quartile for each neighborhood group.

plt**.**figure(figsize**=**(10,6))

ax **=** sb**.**boxplot(data**=**dt, x**=**'neighbourhood\_group',y**=**'availability\_365')

# Appendix 8

Heatmap usually used to display the correlation matrix between variables. The code in python to achieve this is as follows.

corr\_matrix **=** dt**.**corr()

plt**.**figure(figsize**=**(14,8))

sb**.**heatmap(corr\_matrix, annot**=True**)

# Appendix 9

The first step of fata cleaning process is to remove fields which not included in the analysis.

dt**.**drop(['id','host\_id','host\_name','name','latitude','longitude','last\_review'], axis**=**1, inplace**=True**)

dt**.**head()

Graphical user interface, application

Description automatically generated

The next step is to check whether any null value

dt**.**isna()**.**sum()

And the following are the result after check if any null value.

Text

Description automatically generated

The null value is found on the reviews\_per\_month data. To normalize the data, the null value is replaced with zero with the assumption that the null value doesn’t have any review.

dt['reviews\_per\_month'] **=** dt['reviews\_per\_month']**.**fillna(0)

# Appendix 10

Outlier data should be removed to make sure the analysis is accurate. To check the outlier for the price, we analyzed the box plot distribution for the price using the below code.

fig **=** px**.**box(dt, y**=**"price")

fig**.**update\_layout(

autosize **=** **False**,

width **=** 400,

height **=** 400

)

fig**.**show()

From the box plot, we can use the price range between 0 – 350 USD. Removed the outlier by using the range using the below code.

dt **=** dt[dt["price"]**<**350]

dt **=** dt[dt["price"]**>**0]

# Appendix 11

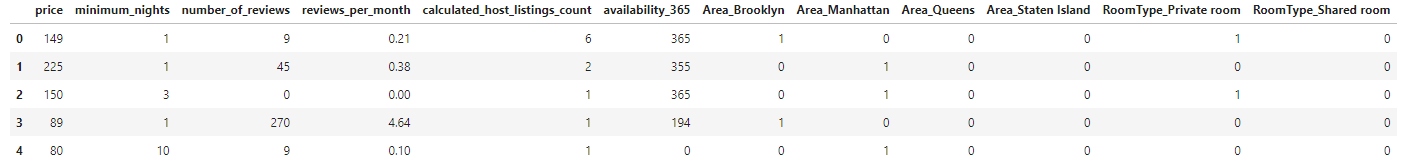
The following codes are used to get the multilinear regression.

dt**.**fillna({'reviews\_per\_month':0}, inplace**=True**)

dt **=** pd**.**get\_dummies(dt, columns**=**['neighbourhood\_group',"room\_type"], prefix **=** ['Area',"RoomType"],drop\_first**=True**)

dt**.**drop(["neighbourhood"], axis**=**1, inplace**=True**)

dt**.**head()



# set X as independent variable

X **=** dt**.**drop(['price'],axis**=**1)

# y: price set as dependent variable

y **=** dt['price']**.**values**.**reshape(**-**1,1)

stdc\_X **=** StandardScaler()

stdc\_y **=** StandardScaler()

X **=** stdc\_X**.**fit\_transform(X)

y **=** stdc\_X**.**fit\_transform(y)

Import library train\_split\_split from sklearn.model\_selection to analyze further.

Xval\_train, Xval\_test, yval\_train, yval\_test **=** train\_test\_split(X, y, test\_size**=**0.30, random\_state**=**42)

print(Xval\_train**.**shape)

print(yval\_train**.**shape)

print(Xval\_test**.**shape)

print(yval\_test**.**shape)

linear\_reg **=** LinearRegression()

The next step is fitting training data from the model and get intercept and coefficient value.

linear\_reg**.**fit(Xval\_train, yval\_train)

print("intercept is: ",linear\_reg**.**intercept\_)

print("coefficients are: ",linear\_reg**.**coef\_)

yval\_pred **=** linear\_reg**.**predict(Xval\_test)

After get the intercept and coefficient, display the information for R square value, MAE, MSE, RMSE with the print statement where the value imported from the function as below.

r2\_score(yval\_test,yval\_pred) #to get R Square value

metrics**.**mean\_absolute\_error(yval\_test, yval\_pred) #to get MAE

metrics**.**mean\_squared\_error(yval\_test, yval\_pred) #to get MSE

np**.**sqrt(metrics**.**mean\_squared\_error(yval\_test, yval\_pred)) #to get RMSE

Import library cross\_val\_score from sklearn.model\_selection.

my\_pipeline **=** Pipeline(steps**=**[('model', LinearRegression())])

*# Sklearn calculates as negative scores, hence muliply by -1*

scor1 **=** 1 **\*** cross\_val\_score(my\_pipeline,

X,y,

cv**=**10,

scoring**=**'r2')

scor2 **=** **-**1 **\*** cross\_val\_score(my\_pipeline, X, y,

cv**=**10,

scoring**=**'neg\_mean\_absolute\_error')

scor3 **=** **-**1 **\*** cross\_val\_score(my\_pipeline, X, y,

cv**=**10,

scoring**=**'neg\_root\_mean\_squared\_error')

print("R squared score:\n", scor1)

print("Avg R squared score):",scor1**.**mean())

print("RMSE scores:\n", scor3)

print("Avg RMSE score:",scor3**.**mean())

x2 **=** sm**.**add\_constant(X)

est1 **=** sm**.**OLS(y, x2)

*#OLS is Ordinary Least Squares*

*#est.TAB*

est2 **=** est1**.**fit()

print(est2**.**summary())

# Appendix 12

Multiple linear regression is noted in the below equation.

|  |  |
| --- | --- |
|  | (1) |

Y is an independent variable while x is the dependent variable and b is the coefficient.

# Appendix 13

This section of code is the function to get the ROC. The first step, import classification\_report from library sklearn.metrics.

*# Get classification result*

**def** get\_classification\_results():

confusion\_mat **=** confusion\_matrix(y\_true**=**y\_test, y\_pred**=**y\_pred)

print('Conf matrix:\n', confusion\_mat)

labels **=** ['Positive', 'Negative']

figs **=** plt**.**figure()

a\_x **=** figs**.**add\_subplot(111)

c\_ax **=** a\_x**.**matshow(confusion\_mat, cmap**=**plt**.**cm**.**Blues)

figs**.**colorbar(cax)

a\_x**.**set\_xticklabels([''] **+** labels)

a\_x**.**set\_yticklabels([''] **+** labels)

plt**.**xlabel('Pred.')

plt**.**ylabel('Exp.')

plt**.**show()

print("Accuracy", metrics**.**accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

auc\_score **=** roc\_auc\_score(y\_test, y\_pred)

print("AUC: ")

print(auc\_score)

Initiating the library to the KNN classifier will be the first step of the code. Then calculate the confusion matrix to compare the true positive results and negative positive result.

classifier **=** KNeighborsClassifier()

Import train\_test\_split from the library sklearn.model\_selection.

Xval\_train, Xval\_test, yval\_train, yval\_test **=** train\_test\_split(X, y, test\_size**=**0.30, random\_state**=**42)

classifier**.**fit(Xval\_train,yval\_train)

yval\_pred **=** classifier**.**predict(Xval\_test)

To get the confusion matrix, import confusion\_matrix from sklearn.metrics library.

cm **=** confusion\_matrix(yval\_test,yval\_pred)

get\_classification\_results()