CZ4032 - Data Analytics & Mining

Predicting Prices for Airbnb Listings in Singapore

Group 17
Edgar Tok Jun Jie
Janaki H Nair
Sean Yap Hock Keong
Dodda Sharon Olivia
Wayne Lim Wei En

1. Abstract

This report summarizes our methods and results of predicting the price range of an Airbnb listing in Singapore based on their location and house type. We used the Singapore Airbnb dataset from Kaggle. We used four different models namely - decision tree classifier, random forest regression, K-nearest neighbour and support vector machine (SVM).

2. Problem description

2.1. Motivation

Airbnb is an online platform for home-sharing, where owners can offer their house for short-term accommodation, mainly to tourists. The owners can decide and list their own pricing per day. Pricing is important in an Airbnb listing as it is one of the main factors that tourists consider when looking for accommodation.

Although Airbnb recommends owners to search for comparable listings in their neighborhood to decide, they do not offer any official pricing tool or strategy. However, this is not an efficient way to decide the pricing as it is tedious and inaccurate.

2.2. Problem definition

Since pricing is a crucial factor, we have decided to come up with a strategy that recommends a price to an owner based on their location and other features. In this project, we are working specifically with the Singapore dataset.

Even though Airbnb is illegal in Singapore, we chose this dataset as we are more familiar with the locations and neighbourhoods in Singapore. Hence, we would know the regions that are generally more expensive due to its location and the landmarks in the neighbourhood, for example, Sentosa. Taking into consideration from other listings in the same neighborhood, we aim to help the host boost Singapore hosts’ revenue by recommending a price for their listings based on their features comparing with the listings near them.

2.3. Related work

A study [1] was done by Laura Lewis addressing the same issue. After performing some data analysis on the London Airbnb dataset, she built a machine learning model (XGBoost model) to predict the prices. She implemented the XGBRegressor algorithm and evaluated the model using the mean squared error (MSE) metric. She then built a neural network model, using relu activation function for the hidden layers and linear activation function for the output layer. The mean squared error (MSE) metric was used to evaluate both models. According to the MSE values, it was concluded that the machine learning model performed better.

3. Approach

3.1. Methodology

3.1.1. Airbnb Dataset

We chose to work with Singapore Airbnb dataset from Kaggle, which contains about 7907 samples (*listings.csv*). The dataset contains information such as host name, house location and type. Refer to Table A.1 in Appendix A for the dataset attributes/columns and its details.

Before the analysis and exploration of the data, one of the most important steps is to validate our data. That is to check if there are any missing records as well as features, as it will greatly affect our analysis and training of our models due to insufficient data.

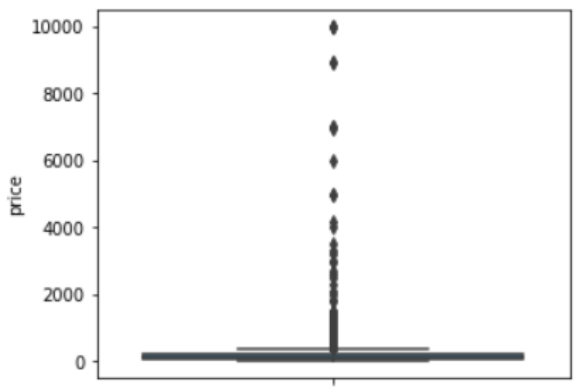
```
id                                   0
name                                 2
host_id                              0
host_name                            0
neighbourhood_group                  0
neighbourhood                        0
latitude                             0
longitude                            0
room_type                            0
price                                0
minimum_nights                       0
number_of_reviews                    0
last_review                       2758
reviews_per_month                 2758
calculated_host_listings_count       0
availability_365                     0
dtype: int64
```

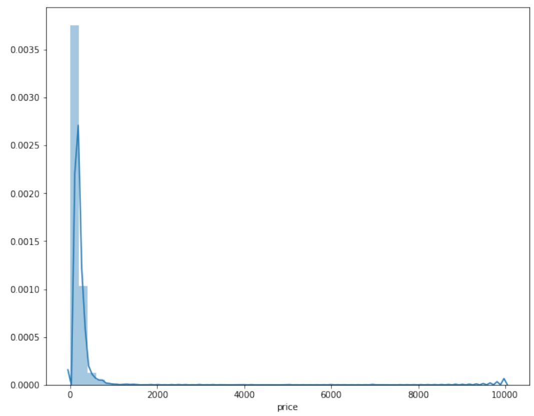
The columns last_review and reviews_per_month contained about 2758 null values (35% of the whole dataset), hence we dropped these two columns.

As you can, there are also two records that have missing names. Therefore, we decided to remove those two records of data before moving on to exploring our dataset.

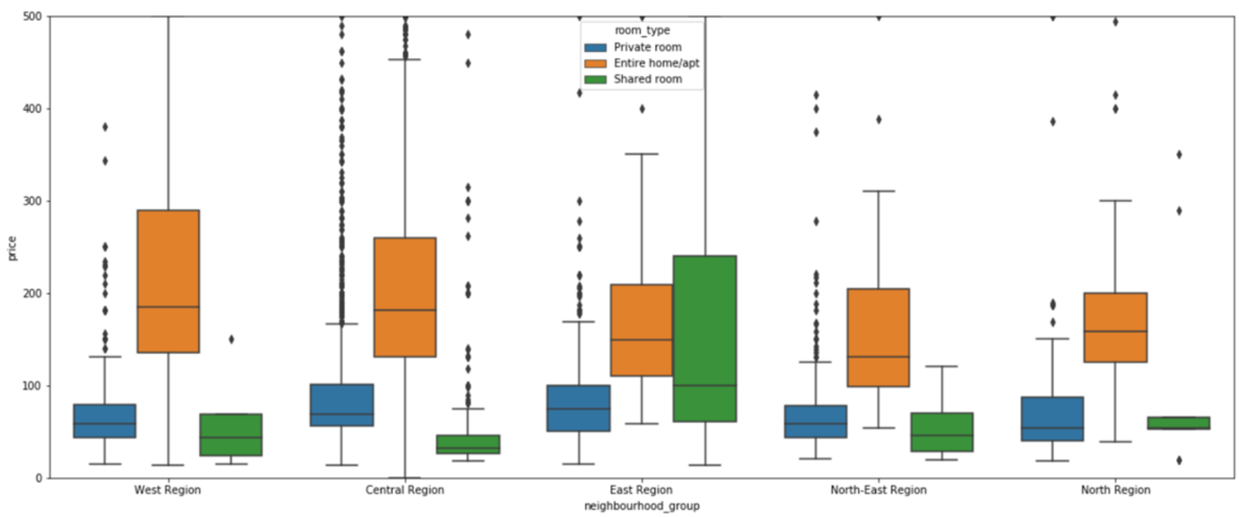
3.1.2. Data Exploration

A simple histogram and boxplot on the price is first plotted to understand the range of price values that are in the dataset. From the figure below, we can see that there are many outliers in the dataset, including listing prices of $10,000 per night. Airbnb are chosen by most vacationers over hotels are due to its competitive prices, therefore these outliers are considered noises.





Since different regions or room type will have a different price range, we decided to look deeper into the price range for specific regions and room type. As seen in the figure below, the most expensive region is “Central Region”. Thus, this dataset has proven that listings that are near a country’s landmark and touristy areas are truly more expensive.



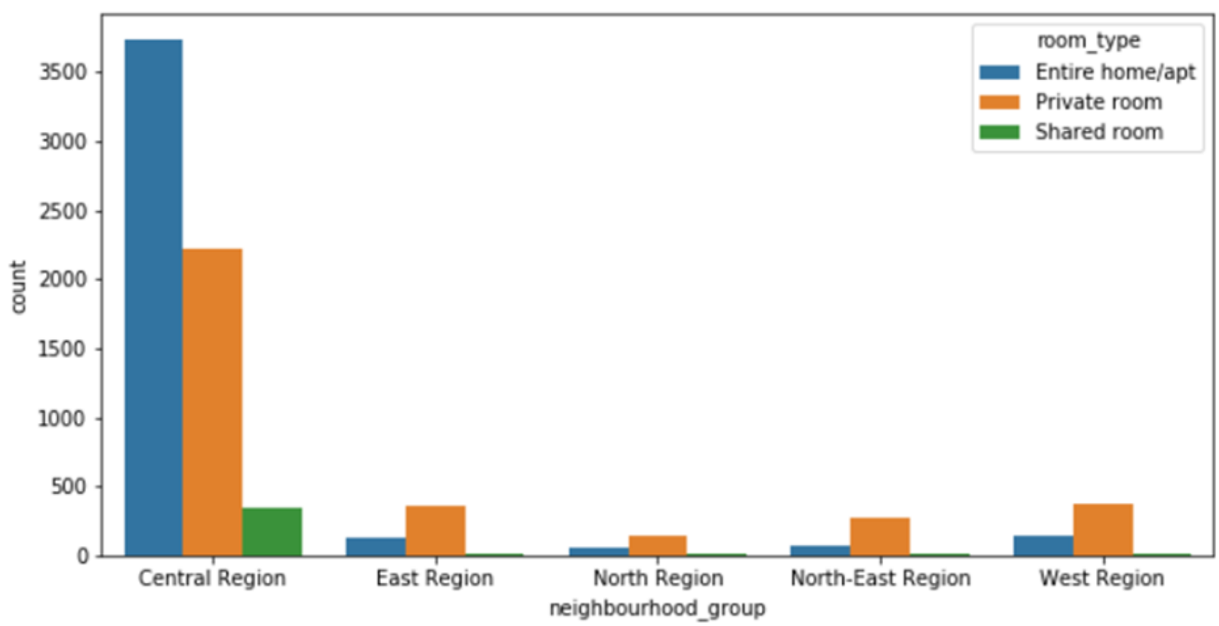
To our surprise, prices on shared rooms are very high. For example, the “East Region”, where the “Shared Room” are as expensive as the “Entire Home/Apt” category. Assumptions were made that it was due to the distance from the airport. However, after looking further into the data, there were only 9 out of the 7097 records that were in “North Region” and “Shared Room” category. Without enough data, we cannot come to a conclusion on why the listings in “Shared

Room" and "East Region" category.

			count	mean	std	min	25%	50%	75%	max
	0	Entire home/apt	3738.0	224.581862	273.367042	0.0	131.00	181.0	260.00	8900.0
central	1	Private room	2223.0	114.380117	325.754109	14.0	56.00	69.0	100.50	10000.0
	2	Shared room	348.0	59.198276	153.239724	18.0	26.00	32.0	46.00	2500.0
	0	Entire home/apt	146.0	334.178082	986.426380	14.0	136.00	185.5	290.00	10000.0
west	1	Private room	378.0	117.825397	627.107711	15.0	44.00	58.0	79.00	10000.0
	2	Shared room	16.0	106.125000	240.729689	15.0	23.50	43.5	69.00	1000.0

From the graphs, it is also obvious to us that there is a huge difference in the price between those that are in the "Entire Home/Apt" and "Private Room" categories. Prices do differ greatly due to this category, Room_type, thus we have decided to choose this as the most important feature for all our models.

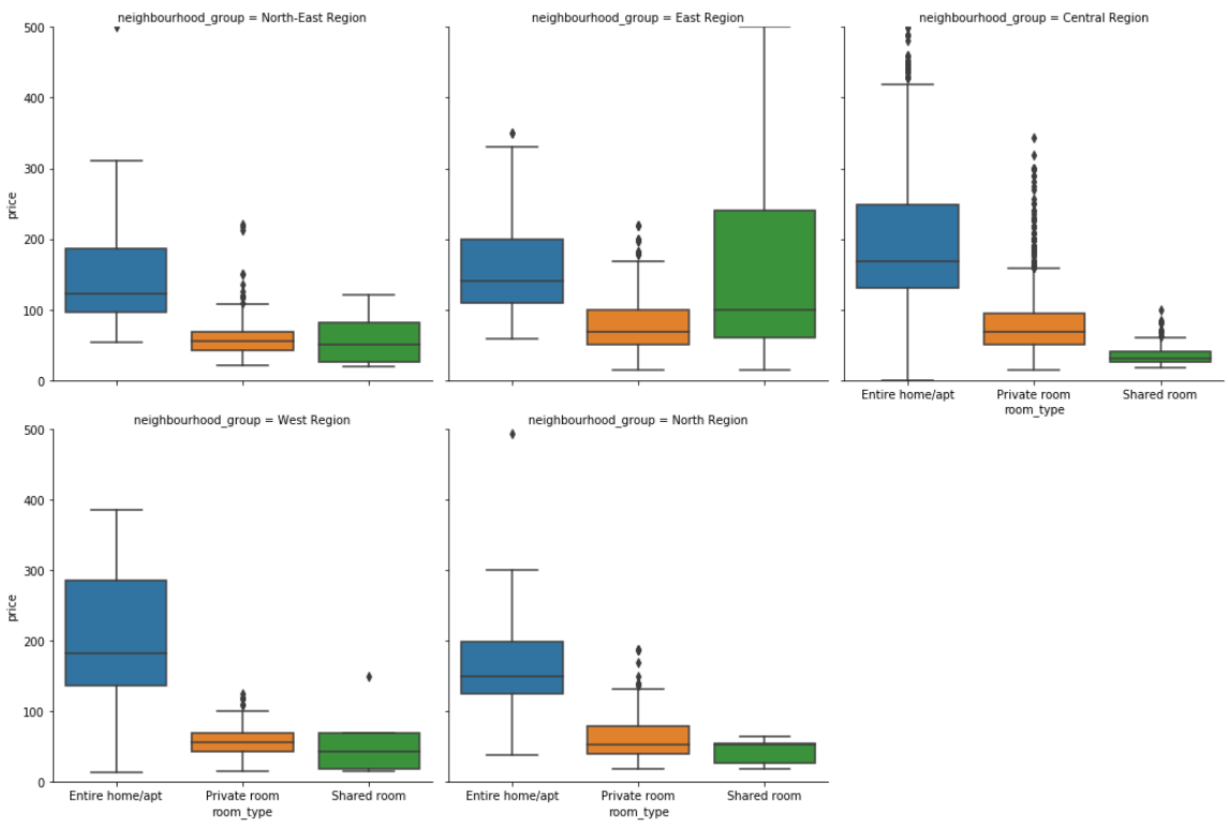
After we have seen the price, we went to compare the number of listings in each region with the different room types. Strangely, we found another interesting fact, where there were actually more listings in the "Private Room" compared to the "Entire Home/Apt" in the "East Region", "North Region", "North-east Region" and "West Region".



We also used google heatmaps to see the prices on singapore's map. On the uncleaned dataset, the listings can hardly be seen. After cleaning the dataset, more areas have started to show up (turning green and red) and we can see that the expensive listings are mostly at the Central area. The Snapshots can be found at Appendix A.

3.1.3. Data Cleaning

Another interesting fact is that the “Entire Home/Apt” category in “West Region” are higher than the “Central Region”. From the figure as seen above, ignoring the max, the 25th, 50th and 75th percentile on the prices in the “West Region” and “Entire Home/Apt” are higher than those in “Central Region”. Interestingly, this shouldn’t be the case as there are not as many landmarks in the west than in the central. However, we do know that there are top schools near the west, like NTU and NUS. As we can see, the maximum prices are at $10,000, which is impossible and not any usual vacationers or exchange students would pay per night. With that in mind, we decided to look clean up and data by removing the outliers that are above the upper bound, using the formula as follows, 75th Quartile + (1.5 * Interquartile Range).



3.2. Algorithms

We chose four main algorithms to predict the prices of the AirBnB listing. Decision Tree Classifier, K-Nearest Neighbour, Simple Vector Machine and Random Forest Tree Regressor.

The first algorithm that was chosen is the Decision Tree Classifier. As discovered, the price of the listing is strongly based on its location such as the neighbourhoods and also the room type. Decision tree is very useful in this area as it can predict accurately based on the neighbourhood and the room type. Instead of predicting the exact values of the listing price, a new class of price range is created with a $50 interval.

	neighbourhood	latitude	longitude	room_type	price	nbh	type	range
0	Bukit Panjang	1.37321	103.77407	Private room	99	1	1	50-99
1	Bukit Panjang	1.34891	103.77137	Private room	69	1	1	50-99
2	Bukit Panjang	1.36895	103.76637	Private room	69	1	1	50-99
3	Bukit Panjang	1.38397	103.77035	Private room	69	1	1	50-99
4	Bukit Panjang	1.34844	103.77219	Private room	65	1	1	50-99
...	...	...	...	...	...	...	...	...
7374	Mandai	1.42562	103.82559	Private room	44	41	1	0-49
7375	Mandai	1.42571	103.82591	Private room	40	41	1	0-49
7376	Western Water Catchment	1.34364	103.68676	Private room	54	43	1	50-99
7377	Western Water Catchment	1.38781	103.74100	Private room	40	43	1	0-49
7378	Western Water Catchment	1.35424	103.68162	Private room	31	43	1	0-49

The next model that was chosen is the K-NN, as the price can be easily predicted based on its location, latitude and longitude, which could be more accurate than using the neighbourhood. An assumption was made that listings within an area should be around the same price range, and therefore could be grouped and classified as one.

Simple Vector Machine differs from the other classification algorithms in the way that it chooses the decision boundary that maximizes the distance from the nearest data points of all the classes. An SVM doesn't merely find a decision boundary; it finds the most optimal decision boundary, that is why it was chosen to try the training and testing of the datasets.

The reason Random Forest Regressor algorithm was chosen at last is due to the fact that it is robust to overfitting. Random forest uses a machine learning methodology that aggregates imperfect decision trees. When the different predictions of the trees are averaged, the imperfections get minimized. This is called bagging. Moreover, as compared to linear regressors, random forest do not require scaling of features or complex feature engineering.

4. Implementations

To create the different models, we have decided to use Jupyter Lab as the main platform to program and display the models/results.

Jupyter Lab is used as it is versatile, shareable and performs data visualization in the same environment.

The main programming language will be in python, and we utilized many different libraries:

a) Pandas - used to read in the dataset from a CSV file.
b) Matplotlib - used to plot graphs
c) Seaborn - used for statistical data visualization
d) Gmaps - used to display heatmaps and scatter plot of the prices and neighbourhoods.
e) Sklearn - to use the KNeighborsClassifier library for prediction, the train_test_split library to split the data into training data and test data, and the metrics library to calculate the accuracy score of the model.

5. Experimental results and analysis

5.1. Experimental Setup

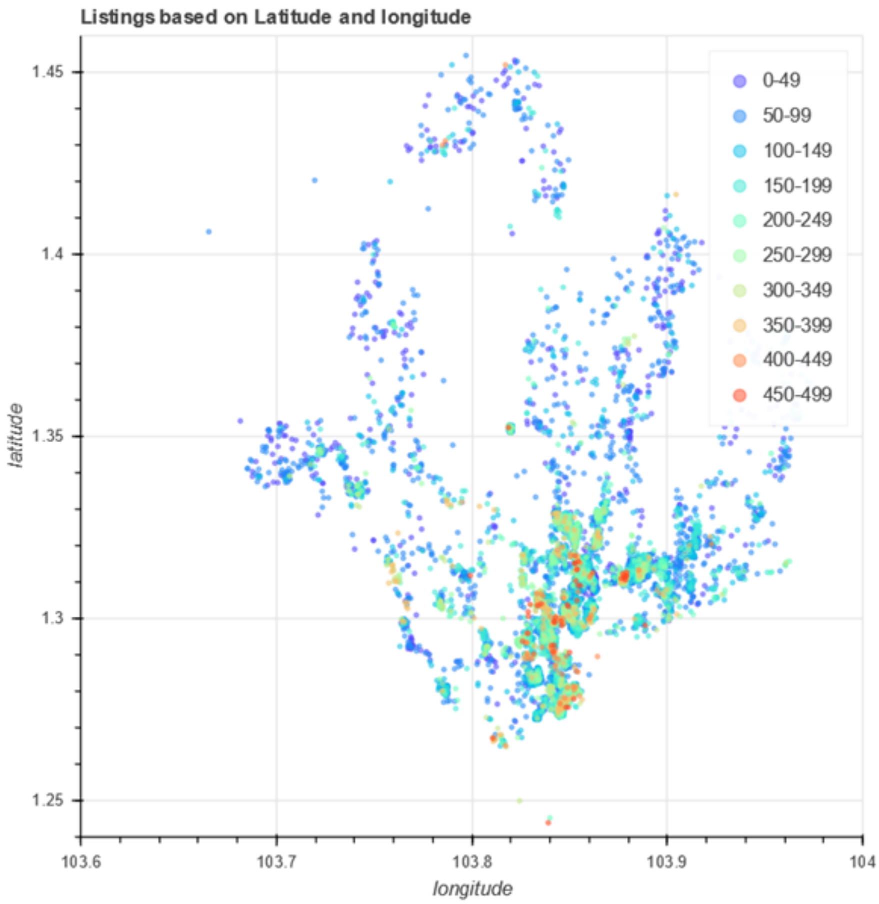
Before we begin working on the models, a new column ‘Price range’ is generated based on the listing’s original price. Each price range has an interval of $50 (e.g. a $78 listing has a price range $50-$99) as we would like to give a closer estimate to the original price(rather than using $100 intervals).

5.1.1. Decision Tree Classifier

Decision Tree Classifier is also considered one of the most commonly used supervised learning methods with decent accuracy. However, one has to use the correct features where has the highest correlation with the price. Therefore, neighbourhood and room type are the two features used mainly for this model. It is also experimented that by adding more features that aren’t important would actually bring down the accuracy as well.

5.1.2. K-NN

Before we start performing K-NN, we would like to see the location of the listing based on its latitude and longitude in a graph. We used a Bokeh plot to visualize this:



Graph K-1 with price range classes.

When using K-NN, we need to consider the issue on the breaking of ties. Breaking of ties happens when there are equal amounts of class nodes and the classifier is unable to decide which class to pick as the prediction. Our solution to combat this is to constantly adjust the K values to obtain the highest possible accuracy.

In other words, our K-NN is performed by setting the value of K starting from 2 and increasing it by 1 each time until the maximum K value(that we set), 40, is reached. Out of all the iterations, the program will note the highest accuracy along with its associated K value. That K value will be the best K value that is possible for the dataset.

5.1.3. Simple Vector Machine (SVM)

In the case of SVM algorithm, there are 4 different types of kernels to choose from, namely Linear, Gaussian, Sigmoid and Polynomial. Out of the 4 kernels, only Polynomial was not able to be executed. With the 3 different kernels remaining, the columns 'neighbourhood', ‘room_type’ and ‘price range’ are being trained and tested on.

5.1.4. Random Forest Regressor

In the case of the random forest regressor algorithm, the column ‘price range’ was not required. After balancing the dataset by removing the outliers, we then split the data into the training set and test set in the ratio 8:2. We set the random_state to 42 which means that the results will be the same each time the split is run for results.

We then tuned the hyperparameters for the regressor. The following parameters were used:

n_estimators=100

max_features='auto'

min_samples_split=2

min_samples_leaf=1

bootstrap=True

5.2. Comparison Schemes

Models	Features Used
Decision Tree Classifier	Neighbourhood, Room_Type
K-NN	Latitude,Longitude
SVM	Neighbourhood, Room_Type
Random Forest Regressor	Neighbourhood, Room_type, Longitude, Latitude, Host_id, availability_365

5.3. Results and Analysis

5.3.1.Decision Tree Classifier

The model was trained on clean and unclean dataset, to prove that the accuracy can be increased by performing proper cleaning before training the dataset. As shown in the figure below, it is also proven that by putting in many features will not help in improving the accuracy of the model, instead only selecting the most important features with the highest correlation with the classifier. Since the accuracy is lower with the other features such as latitude, longitude and number of reviews, model was not trained with those features on the clean dataset. With 45.5% accuracy using the two features, neighbourhood and room type on the clean dataset.

Uncleaned

Features	Accuracy
Neighbourhood, Room_type	0.4348
Neighbourhood, Room_type, Latitude, Longitude	0.3801
Neighbourhood, Room_type, Number_of_reviews	0.3940
Neighbourhood, Room_type, Latitude, Longitude, Number_of_reviews	0.3843

Cleaned

Features	Accuracy
Neighbourhood, Room_type	0.4552
Neighbourhood, Room_type, Latitude, Longitude	0.4209

5.3.2. K-NN

Firstly, we tried performing K-NN on both the uncleaned and cleaned dataset to compare the accuracy of each dataset. Naturally, the cleaned dataset has a higher accuracy.

	Uncleaned		**Cleaned**	
Listing Type	**K value**	**Accuracy (%)**	**K value**	**Accuracy (%)**
All Listings	25	38.314176	21	39.943227
Entire home	24	37.609970	17	41.100076
Private room	22	55.734767	24	64.038461
Shared room	7	73.282442	4	83.050847

Table K-1. Uncleaned Dataset.

Now focus on the Cleaned Dataset. From the accuracy score above, performing K-NN on all listings gives an accuracy of 39.943227% which is quite low. This is expected since different types of housing can exist together within a vicinity which causes the algorithm to classify a test data inaccurately. Hence, we tried to explore further by narrowing down the search space by the room_type.

The accuracy for Private room is not that bad, and for shared room is excellent. However, the accuracy for Entire home is still quite low. Hence, the search space is further narrowed down by room_type and **neighbourhood_group.**

Listing Type	K value	Accuracy (%)
West, Entire home	6	57.446808
West, Private room	29	59.504132
West, Shared room	4	66.666666
East, Entire home	4	33.33333
East, Private room	35	60.52631
East, Shared room	2	50
Central, Entire home	11	39.780405
Central, Private room	22	62.130177
Central, Shared room	4	85.849056
North, Entire home	17	33.333333
North, Private room	3	63.043478
North, Shared room	3	66.666666
North-East, Entire home	3	33.333333
North-East, Private room	17	60.465116
North-East, Shared room	2	33.333333

Table K-2.

Overall, when comparing the above results(Table K-2) with the cleaned results before the search space is narrowed down(Table K-1) by neighbourhood_group and room_type, the performance of K-NN is better when narrowing the search space only on room_type. This is because when the search space is narrowed down, there are not enough listing data to split into training and test data. For example, some shared rooms in the east only have 5 listings. Hence, decreasing the accuracy of the model.

Another challenge K-NN face is that the price range interval is too small. This is one of the reasons why the accuracy for predicting ‘Entire home’ is so low because the prices of ‘Entire home’ listings are spread far apart. The prices of these listings usually depend on the size of the house and since the price range interval is only $50, there will be a lot of classes which will decrease K-NN’s accuracy. In other words, if a surrounding area has different types of price range classes, it becomes harder to predict accurately.

From the results and analysis, K-NN is not a good algorithm for predicting the AirBnB listing prices.

5.3.3. Simple Vector Machine (SVM)

SVM was performed on both cleaned and uncleaned datasets, with minimal changes in the accuracy for each kernel, with the cleaned datasets achieving a slightly higher accuracy of 1% better. Results of the SVM can be found on Figure A.3, A.4 and A.5 in Appendix A.

From the results, it can be seen that Gaussian Kernel provides the best accuracy out of the remaining 3 kernels available, however, the accuracy is not as high as Random Forest Regressor.

5.3.4. Random Forest Regressor

Random forest regressor was performed on both the cleaned and uncleaned datasets. It was observed that the cleaned data set produces a much higher accuracy.

Features	Accuracy
room_type, longitude, latitude	36.23%
room_type, longitude, latitude, host_id	43.04%
room_type, longitude, latitude, host_id, availability_365	44.3%

Uncleaned dataset

Features	Accuracy
room_type, longitude, latitude	64.26%
room_type, longitude, latitude, host_id	70.7%
room_type, longitude, latitude, host_id, availability_365	71.44%

Cleaned dataset

Initially, important features like room_type and latitude were taken to train the model and the accuracy produced was quite high. Then, host_id was considered as it gave a good correlation value with the price attribute when the correlation plot was plotted. In the same way, the attribute 'availability_365' was also added which in turn gave much better accuracy results.

6. Discussion of pros and cons

Model	Pros	Cons
Decision Tree Classifier	- Requires little data preprocessing - Does not require normalization or scaling of data - Model is not affected by missing values in data	- Expensive as time taken and complexity is higher - Unstable as a small change in data can cause a large change in the structure
K-nearest Neighbour	- Simple model with no training required. - Easy to implement. - Constantly evolving, adapts to new data input quickly.	- Very sensitive to outliers - Cannot deal with missing value problem - Curse of Dimensionality

Support Vector Machine	- Accurate in high dimensional spaces - Memory efficient	- Not suitable for large datasets
Random Forest Regression	- Can solve both types of problems: classification and regression - Provides a feature importance estimate	- High computational cost - Predictions are slower

7. Conclusions

7.1. Summary of Project Achievements

To summarize, out of the four algorithms used, the Random Forest Regressor is the best model for predicting the price of an AirBnB listing. The dataset we worked on is infested with a multitude of issues:

a) About 5900 listings out of the total 7900 listings are located in the Central
b) Some room types in a neighbourhood only have a few listings
c) Missing metrics - Harder to predict for ‘Entire home’ room_type as the prices not only depend on location but on the size of the house.

These issues affected the models greatly, but Random Forest Regressor proved to be resilient against the problems as it provides a feature importance estimate, hence it selects the good features on its own. Furthermore, it does not require feature scaling.

7.2. Directions for Improvements

For future improvements, we would like to introduce a new feature ‘Distance to MRT’ or ‘Distance to Shopping malls’. It is a widely known fact that the closer a housing is to an MRT or Shopping mall, the housing would be slightly pricier compared to those that do not have easy access to them.

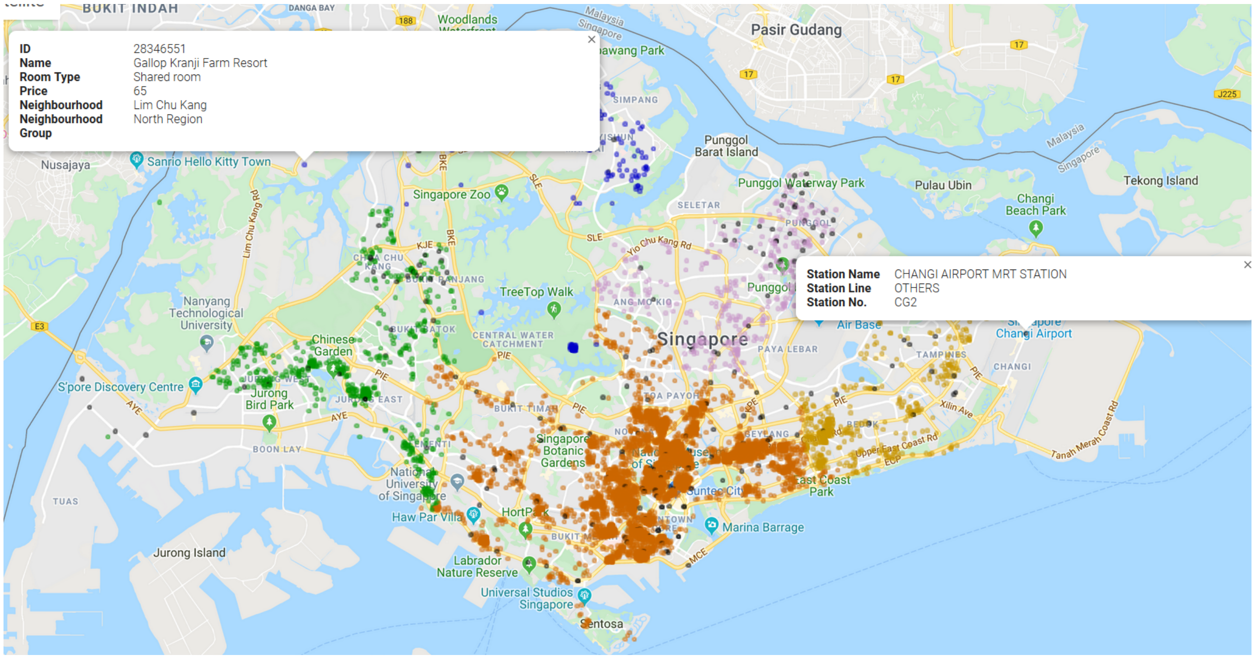


Figure 7.2. Gmaps Scatter Plot showing AirBnB Listings and MRT/LRT.

Another improvement that can be done to improve the prediction of the pricing includes the usage of the ‘name’ column, by picking up commonly used words and training together with the dataset to produce a better accuracy. An example is, we can pick up words such as “mrt”, “near”, “bedroom” etc, and convert it into a point system to use as part of the training of datasets. Refer to the screenshot below:

Feature Engineering - Point System

name	wordCount	mrt	room	near	bedroom	apartment	apt	city	studio	private	spacious	total_pts
Common room in Condo	4	0	1	0	0	0	0	0	0	0	0	1
Master Bedroom in Bishan	4	0	0	0	1	0	0	0	0	0	0	1
Room for 2 persons	4	0	1	0	0	0	0	0	0	0	0	1
Master Room for rent near Bishan MRT	7	1	1	1	0	0	0	0	0	0	0	3
Common Room BISHAN MRT AirCon+Wifi+CanCook	5	1	1	0	0	0	0	0	0	0	0	2
One single bedroom for rent, 3 months minimum	9	0	1	0	0	0	0	0	0	0	0	1
Queen size bed+ParkView@Bishan	3	0	0	0	0	0	0	0	0	0	0	0

8. References

[1] L. Lewis, “Predicting Airbnb prices with machine learning and deep learning”, 22 May 2019. [online] Available: https://towardsdatascience.com/predicting-airbnb-prices-with-machine-learning-and-deep-learning-f46d44afb8a6

[2] J. Devlin, “Python Machine Learning Tutorial: Predicting Airbnb Prices”, 10 July 2019.

https://www.dataquest.io/blog/machine-learning-tutorial/

9. Appendix A

Column Name	Meaning/ Example
id	Room ID
name	Room name
host_id	Host ID
host_name	Host names
neighbourhood_group	Singapore regions (e.g. west, east, central, etc.)
neighbourhood	Specific place (e.g. Ang Mo Kio, Bishan, etc.)
latitude	Geographic coordinate of north-south position
longitude	Geographic coordinate of east-west position
room_type	Type of room (e.g. private, entire house/apartment, shared room)
price	SGD per night
minimum_nights	Minimum no. of nights guests need to spend
number_of_reviews	No. of reviews written by previous guests
last_review	Date of last review written
reviews_per_month	No. of reviews per month
calculated_host_listings	Total no. of rooms or houses in host catalog in Airbnb
availability_365	Number of days the listing is available for rent

Table A.1. Dataset Columns and Details

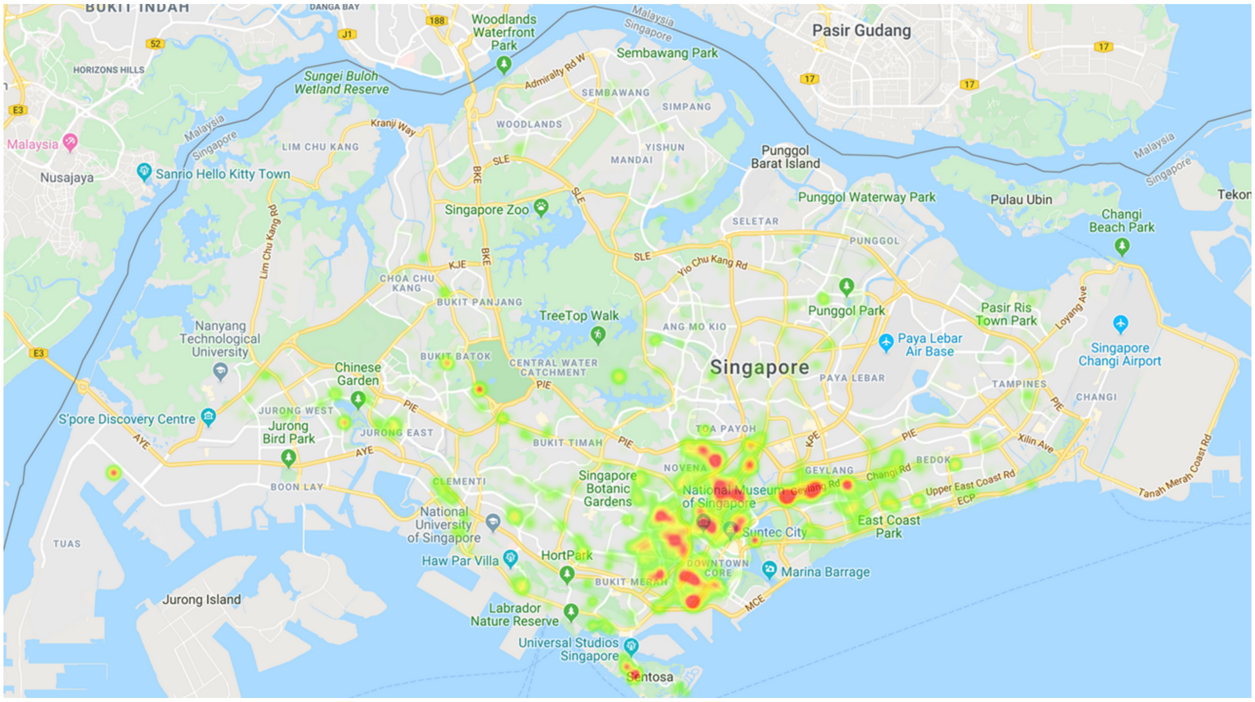


Figure A.1. Google Heatmaps on Uncleaned Dataset

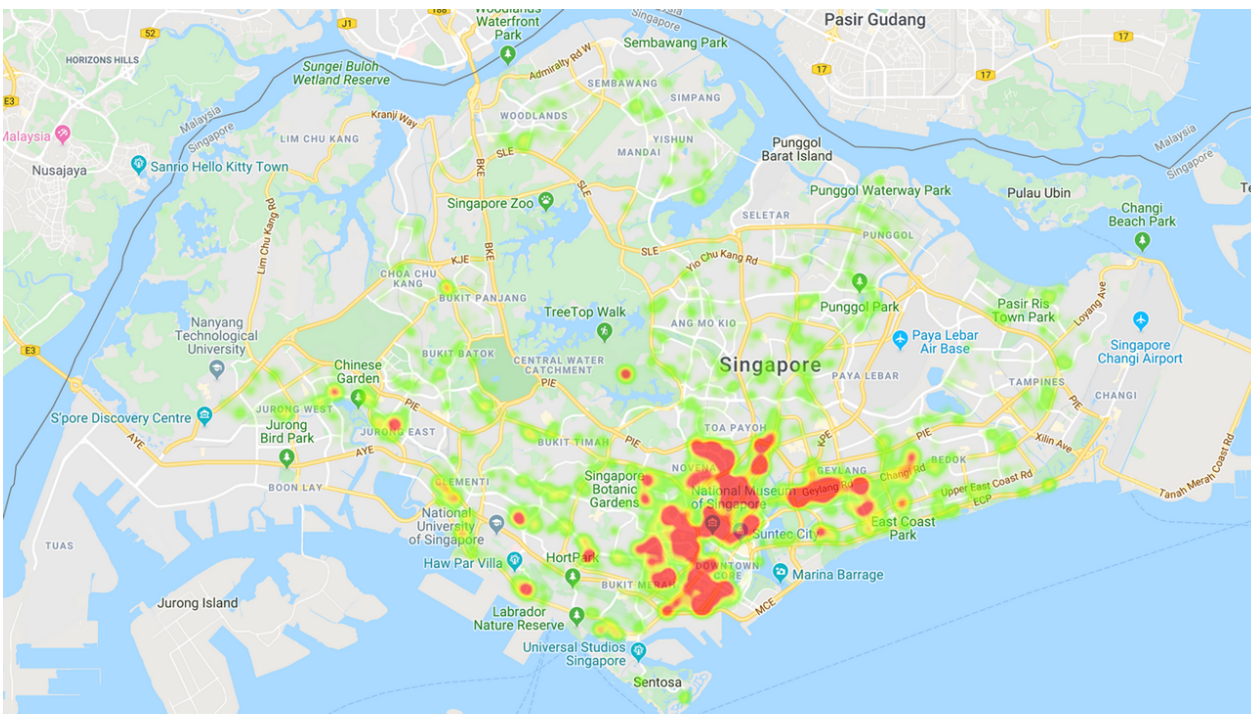


Figure A.2. Google Heatmaps on Cleaned Dataset

```
[[  0   0   0   0   0   6   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  35   4   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 110   7   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 105  11   0   0   0   1   0   0   0   0   0   0]
 [  0   0   0   0   0 160  29   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 245  50   0   0   0   2   0   0   0   0   0   0]
 [  0   0   0   0   0  40 238   0   0   0   9   0   0   0   0   0   0]
 [  0   0   0   0   0   5   3   0   0   0   1   0   0   0   0   0   0]
 [  0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   5   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   1  80   0   0   0  53   0   0   0   0   0   0]
 [  0   0   0   0   0   2   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   9   3   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   8   2   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  21   4   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   6   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   3   1   0   0   0   0   0   0   0   0   0   0]]
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         6
           2       0.00      0.00      0.00        39
           3       0.00      0.00      0.00       117
           4       0.00      0.00      0.00       117
           5       0.00      0.00      0.00       189
           6       0.32      0.82      0.46       297
           7       0.55      0.83      0.66       287
           8       0.00      0.00      0.00         9
           9       0.00      0.00      0.00         3
          10       0.00      0.00      0.00         5
          11       0.80      0.40      0.53       134
          12       0.00      0.00      0.00         2
          13       0.00      0.00      0.00        12
          14       0.00      0.00      0.00        10
          15       0.00      0.00      0.00        25
          16       0.00      0.00      0.00         6
          17       0.00      0.00      0.00         4

    accuracy                           0.42      1262
   macro avg       0.10      0.12      0.10      1262
weighted avg       0.29      0.42      0.32      1262
```

Figure A.3. Linear Kernel

```
[[  0   1   0   0   1   2   2   0   0   0   0   0   0   0   0   0   0]
 [  0   6   0   0  10  26   4   0   0   0   0   0   0   0   0   0   0]
 [  0   3   0   0  23  62   4   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0  31  66  10   0   0   0   5   0   0   0   0   0   0]
 [  0   1   0   0  64 105  32   0   0   0   0   0   0   0   0   0   0]
 [  0   5   0   0  40 178  57   0   0   0   2   0   0   0   0   0   0]
 [  0   0   0   0  19  16 269   0   0   0   5   0   0   0   0   0   0]
 [  0   0   0   0   1   2   5   2   0   0   1   0   0   0   0   0   0]
 [  0   0   0   0   0   2   0   0   0   0   0   0   0   0   0   0   0]
 [  0   2   0   0   0   2   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   1   1  81   0   0   0  52   0   0   0   0   0   0]
 [  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   4   0   0   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0   2   2   5   0   0   0   0   0   0   0   0   0   0]
 [  0   5   0   0  13  13   2   0   0   0   0   0   0   0   0   0   0]
 [  0   2   0   0   1   4   1   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   1   4   0   0   0   0   0   0   0   0   0   0   0]]
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         6
           2       0.23      0.13      0.17        46
           3       0.00      0.00      0.00        92
           4       0.00      0.00      0.00       112
           5       0.31      0.32      0.31       202
           6       0.36      0.63      0.46       282
           7       0.57      0.87      0.69       309
           8       1.00      0.18      0.31        11
           9       0.00      0.00      0.00         2
          10       0.00      0.00      0.00         4
          11       0.80      0.39      0.52       135
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         4
          14       0.00      0.00      0.00        10
          15       0.00      0.00      0.00        33
          16       0.00      0.00      0.00         8
          17       0.00      0.00      0.00         5

    accuracy                           0.45      1262
   macro avg       0.19      0.15      0.14      1262
weighted avg       0.37      0.45      0.39      1262
```

Figure A.4. Gaussian Kernel

```
[[  0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0   0  51   5   0   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0   0  99   5   0   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0   0 114   7   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 185  12   0   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0   0 245  27   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 289   7   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   7   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   4   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   4   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 130   1   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   2   1   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  11   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  13   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0  28   2   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   7   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0]]
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         1
           2       0.25      0.02      0.03        57
           3       0.00      0.00      0.00       105
           4       0.00      0.00      0.00       122
           5       0.00      0.00      0.00       197
           6       0.21      0.90      0.33       273
           7       0.10      0.02      0.04       296
           8       0.00      0.00      0.00         7
           9       0.00      0.00      0.00         4
          10       0.00      0.00      0.00         4
          11       0.00      0.00      0.00       131
          12       0.00      0.00      0.00         3
          13       0.00      0.00      0.00        11
          14       0.00      0.00      0.00        13
          15       0.00      0.00      0.00        30
          16       0.00      0.00      0.00         7
          17       0.00      0.00      0.00         1

    accuracy                           0.20      1262
   macro avg       0.03      0.06      0.02      1262
weighted avg       0.08      0.20      0.08      1262
```

Figure A.5. Sigmoid Kernel