Group Members

Yong Quan Zhi, Tommy (A0195353Y / E0384984@u.nus.edu)

Ritesh Munjal (A0195304H / E0384935@u.nus.edu)

CA 1 Project Report

Problem solving using pattern recognition

Contents

[1 Introduction 2](#_Toc18936287)

[2 Tools and Techniques 3](#_Toc18936288)

[2.1 Tools 3](#_Toc18936289)

[2.2 Techniques 3](#_Toc18936290)

[2.2.1 Regression 3](#_Toc18936291)

[2.2.2 Classification 3](#_Toc18936292)

[3 Model Design Process 4](#_Toc18936293)

[3.1 Regression 4](#_Toc18936294)

[3.2 Classification 5](#_Toc18936295)

[4 Model Performance 6](#_Toc18936296)

[4.1 Regression 6](#_Toc18936297)

[4.2 Classification 10](#_Toc18936298)

[5 Project Findings 10](#_Toc18936299)

[5.1 Regression 10](#_Toc18936300)

[5.2 Classification 18](#_Toc18936301)

[6 Summary 18](#_Toc18936302)

[References 19](#_Toc18936303)

# 1 Introduction

With advances in artificial intelligence over the last 20 years, researchers have been exploring the possibilities of how much computers can replicate human behavior or execute complex tasks that perhaps most humans are unable to do. With pattern recognition, it is possible to enable a computer to learn without specifically programming it to execute specific tasks. Examples of such tasks include, but are not limited to, self-driving vehicles, giving recommendations on videos(Netflix, youtube), fraud detection, predicting consumer trends and etc.

Drawing inspiration from our previous project in our first semester at NUS ISS, our group has decided to go back to the home rental market and consider how landlords decide a price on renting out their homes.

With over 6 million listings world wide and an annual revenue of $3 billion, which is expected to grow by 250% in 2020, we chose AirBnb as a case study and sourced the internet for datasets on AirBnb listings. The following project report will detail our approach to the project and state our findings.



*AirBnb image, <* [*https://www.wsaw.com/fox/content/news/Stevens-Point-homeowners-can-rent-homes-on-AirBnB-and-VRBO-512899481.html*](https://www.wsaw.com/fox/content/news/Stevens-Point-homeowners-can-rent-homes-on-AirBnB-and-VRBO-512899481.html)*>.*

# 2 Tools and Techniques

## 2.1 Tools

The following items below are the tools and libraries used to implement this project.

|  |
| --- |
| Tools |
| Anaconda |
| Spyder |
| Libraries |
| SKLearn |
| Pandas |
| Numpy |
| Matplotlib |
| Math |
| statsmodel |

## 2.2 Techniques

During the course of the project, we decided to use both classification and regression techniques to not only predict home rental prices, but also to see what insights can be drawn from the data with relation to pricing.

### 2.2.1 Regression

As the problem statement is to be able to predict the daily lease price of a home/room, regression techniques are used as the output will be a type of continuous data. As such, it is also important to choose a good variety of models that are linear, non-linear and of branching types and analyze them to choose the best model. Below are the types of machine learning models that will be used for our regression problem.

Machine Learning Models

1. Multiple Linear Regression Model
2. Polynomial Regression Model
3. Decision Tree Regression Model
4. Random Forest Regression Model

### 2.2.2 Classification

# 3 Model Design Process

The first step in designing our machine learning model is to understand the problem statement that we would like to tackle for this project. In our first semester, we created an intelligent home rental recommendation system which was able to recommend users what home listings they could rent based on their responses from a list of predetermined questions. For this CA1 project, we wanted to see if we can explore the idea of being able to predict the price of a home/room so that owners will be able to list their accommodations on the site and the system will be able to provide a recommendation on pricing.

With our problem statement, we searched the web for an appropriate dataset of at least 30 features and at least 3000 samples. The search led us to Kaggle.com and the chosen dataset was for Airbnb which has about 94 features and over 30000 samples. This was more than enough for the purposes of our project and would pose the right amount of challenges for us to process the data and try to see if we can build a feasible model with it.

The following subsections will dive into our classification and regression approaches and how we went about designing the models.

## 3.1 Regression



The first step was to take a look at the dataset and have an idea of what kind of columns were present and how data is represented for each column. As we would need to try and reduce the number of columns to reduce model complexity, columns that were obviously irrelevant to the purposes of the model were noted down so that they could be programmatically removed in our python script.

Next, we proceeded to handle missing data. Through inspection, about a third of the 30000 samples had columns with missing data. It was possible to use data imputation to deal with the missing data, either by replacing it with the mode or mean value of the affected column. In the end, we opted to drop the rows with missing data. This was to ensure the replacing of the mean or mode of the data would not affect the model greatly and to see what could be achieved without the samples with missing data.

Before we proceeded to assign the target and independent variables, we shuffled the data as it was arranged by region and this could have affected our model’s performance as the data could have be split improperly and given the training process an inaccurate spread of samples. This issued was realized only after training had already begun and will be elaborated more later.

Once the target and independent variables had already been assigned, columns were checked to see which were categorical and should be encoded with one hot encoding(get\_dummies method was used to achieve this). After that, the ordinal features were then encoded with labelencoder.

Now that we had columns of features that were of numerical value it was time to see if we could further trim down the number of columns in order to reduce model complexity and improve performance. The p-value of each column with respect to the target variable was used to eliminate undesired columns. The method used for this was backward elimination and the statsmodel library was used to facilitate this process. Initially, the significance level was set at 0.05, however, as there were still too many columns left it was further reduced to 0.000000001. When testing between 0.05 and 0.000000001, the improvements offered was quite minimal to not that significant. Thus, the significance level was set back to 0.05.

The last bit was to scale the features and split them into training and test data. The splitting was set at 9:1 for training to test set ration. This ratio was also determined after much testing was done by increasing and decreasing them, 10% given to the test data was the most optimal amount and further decreasing this amount did not give much improvement.

The four machine learning regression models used were multiple linear, polynomial, decision tree and random forest. The model performance section will show case the performance of each model with the project finding section elaborating more on the results.

## 3.2 Classification

# 4 Model Performance

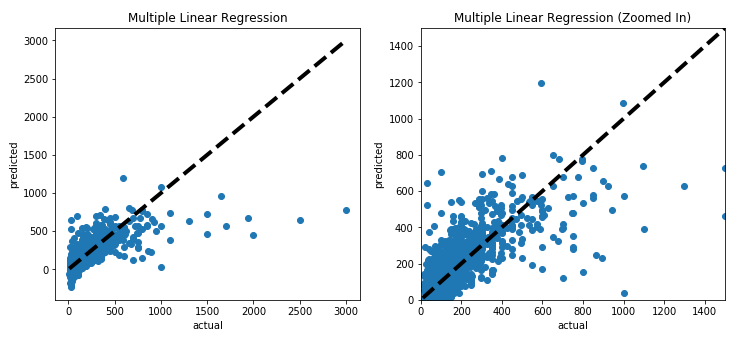
## 4.1 Regression

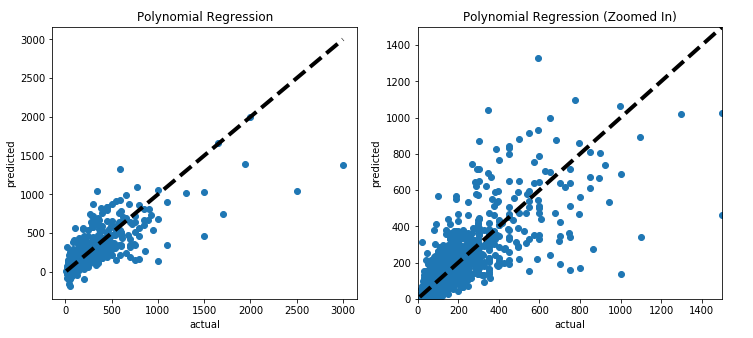
This section will display the model’s performance summary, scatter and distribution plots between predicted and actual Y-values. An analysis for them will be provided in the project findings section(*Section 5.1*) below.

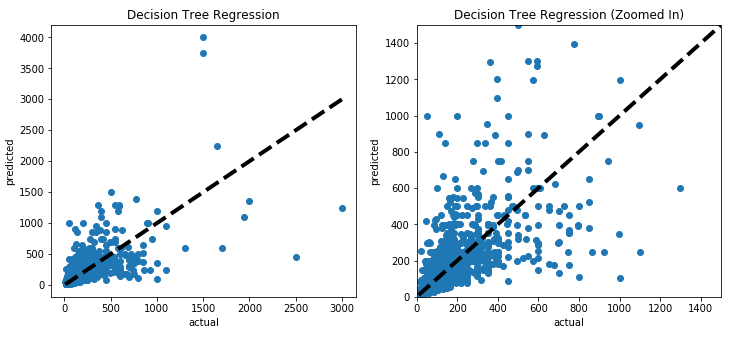
Model Summary (Rounded to nearest 2 decimals)

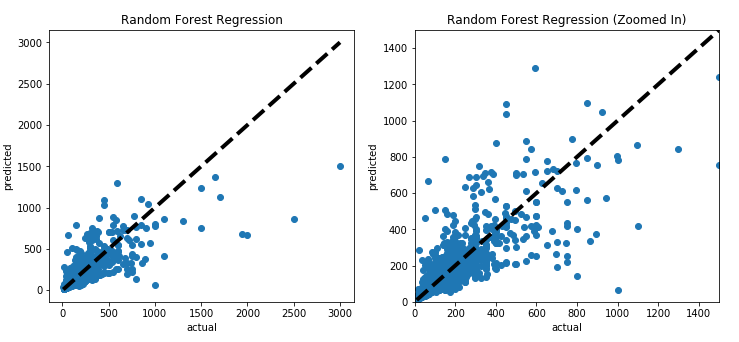
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Multiple Linear | Polynomial | Decision Tree | Random Forest |
| Mean absolute error | ﻿67.65 | ﻿57.11 | ﻿66.61 | ﻿47.79 |
| Root mean sq error (RMSE) | ﻿130.80 | ﻿109.44 | ﻿156.26 | 108.83 |
| Median absolute error | ﻿40.9 | ﻿32.28 | 30.0 | ﻿23.49 |
| R2 score | ﻿0.46 | ﻿0.62 | ﻿0.23 | ﻿0.62 |

Scatter Plots (Predicted vs Actual)

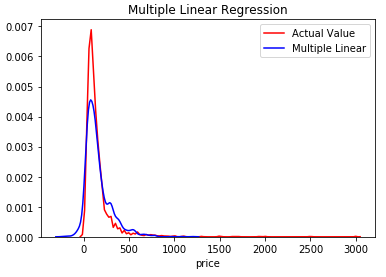


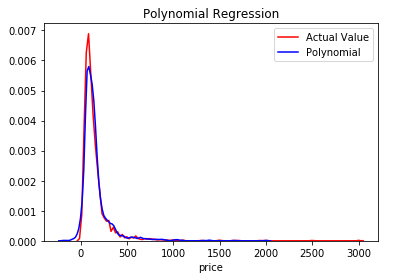


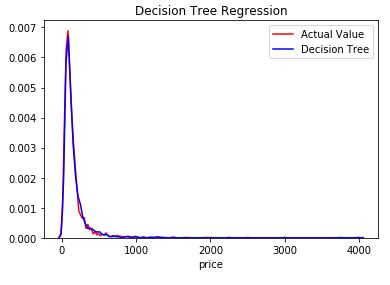


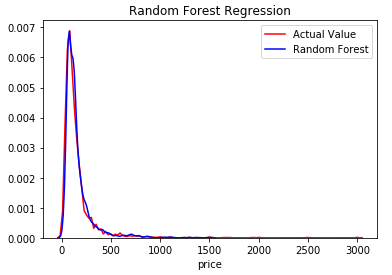


Distribution Plots (Predicted vs Actual)









## 4.2 Classification

# 5 Project Findings

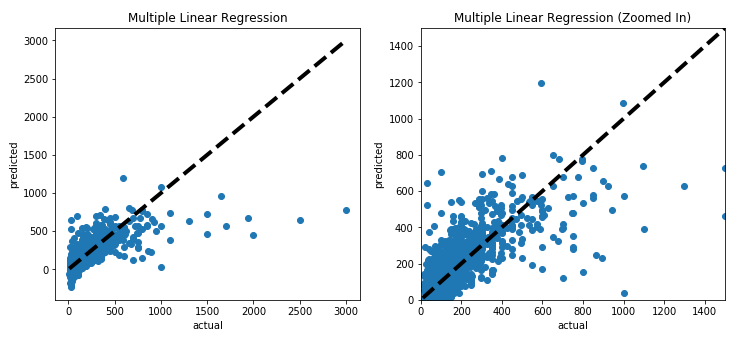
## 5.1 Regression

The main performance metric that will be used for our regression models will be based on mean squared error(MSE). We used root mean squared error(RMSE) in printing out our results so that it is easier for us to read and understand the figures.

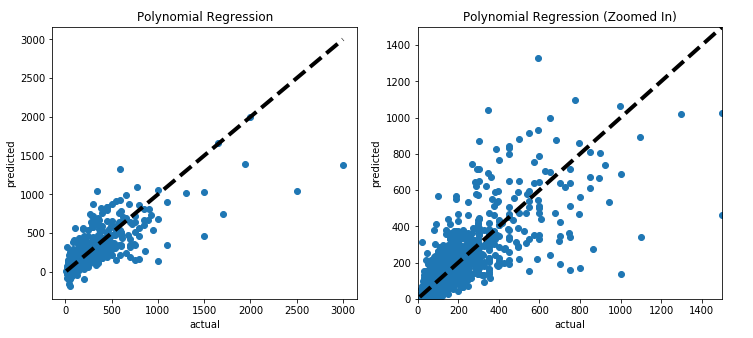
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Multiple Linear | Polynomial | Decision Tree | Random Forest |
| Mean absolute error | ﻿67.65 | ﻿57.11 | ﻿66.61 | ﻿47.79 |
| Root mean sq error (RMSE) | ﻿130.80 | ﻿109.44 | ﻿156.26 | 108.83 |
| Median absolute error | ﻿40.9 | ﻿32.28 | 30.0 | ﻿23.49 |
| R2 score | ﻿0.46 | ﻿0.62 | ﻿0.23 | ﻿0.62 |

From the model summary above, we want to know how far apart the predicted Y-values are from the actual Y-values. We first look at the RMSE, which generally tells us the average of differences for actual and predicted Y-values. Amongst each of the models, our random forest model has the lowest RMSE, followed by polynomial, decision tree then multiple linear. The mean absolute error metric also shows that the random forest model has the lowest score. It is important to note that each time we ran our script, we get different results with some showing the polynomial model performing better while others show the random forest as being better, both these models often are not that far apart from one another in terms of performance. The differences in performance at each run is due to the shuffling of data and certain kinds of data are excluded from the training set. This implies that the original dataset has some data imbalance and perhaps quite a few outliers. Therefore we decided to showcase this example as it is a good representation of the overall performance the machine learning models for all of our runs.

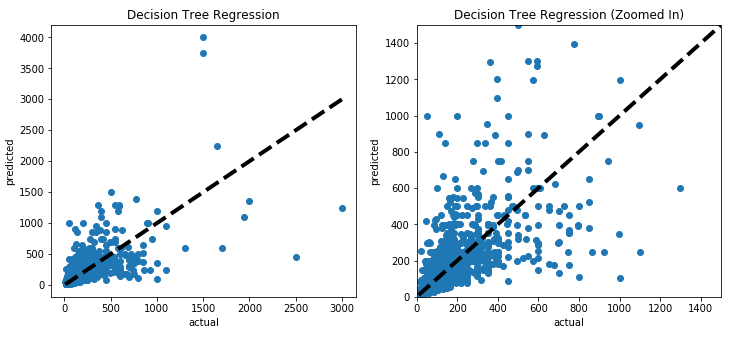
In order to get a better idea of how the data is distributed based on the various machine learning models, we can visualize it by using a scatter plot.



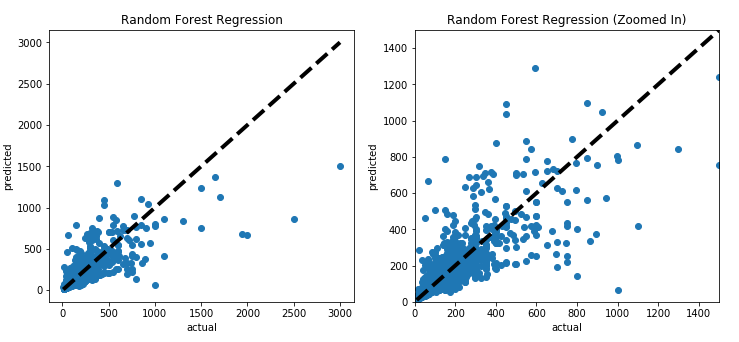
In the multiple linear scatterplot, we can see that the samples are mostly tightly clustered in the 0 to 400 range along the X-axis. As X gets higher, the data points seem to begin deviate further away from the line which represents the machine learning model. This shows that the dataset does not necessarily have a linear pattern, but most of the data conform to a linear trend and which is shown by the cluster at the 0 – 400 range.



For the polynomial scatterplot, majority of the samples are seen clustered at the 0 – 400 range as well but they seem to gravitate closer to the line. Moreover, as X gets bigger, the line seems to centralize itself better to the distribution pattern of the data points. This would help explain why the RMSE score is lower than the multiple linear model.

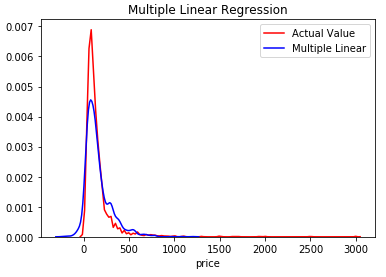


Over to the decision tree scatter plot, you can see that the data clusters quite closely to the line at the 0 – 300 range on the X-axis, however as X gets bigger, the data begins to spread out further away from the line. Comparing between the above 2 models, the decision tree model seems to perform better at earlier values but start to become more imprecise for higher values. This is why it has a higher RMSE than the polynomial model.

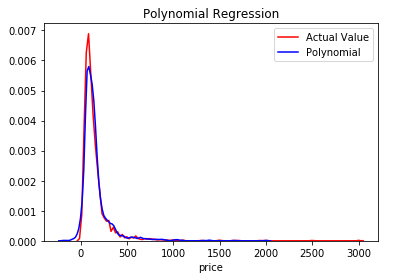


Finally, the random forest model is our best performing model as it has the lowest RMSE score. From the scatterplot you can see that the data points are clustering quite close to the line. Even as X gets bigger, the model does better than the decision tree to fit the data. Although, it begins to go further away from the line as X gets bigger, it is still closer than any of the previous 3 models. For the hyperparamter tuning of the random forest model, we used grid search to discover the most optimal values for the hyperparameters. This enabled use to obtain the most optimal random forest than just test every value one by one.

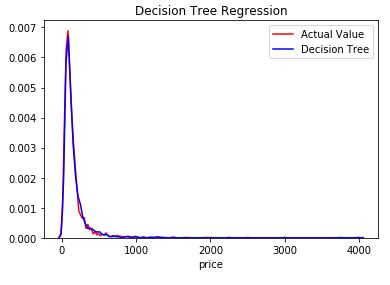
Finally, distribution plots will be used to better visualize how closely the data between predicted and actual Y-values map to one another.



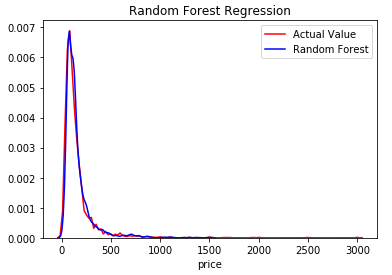
For our multiple linear model, its does not seem to fit to the data very well. It predicts negative values and seems to predict values in the $100 range quite inaccurately. Past the $250 mark, as expected with a linear model, it does not fit to the data very well.



The polynomial model appears to fit our test data quite well. Almost matching the predicted values from the $30 - $250 range. From $250 - $400, it does not fit the data very well but after the $400 mark, it tries to conform to the trend of the data and does better than the multiple linear model.

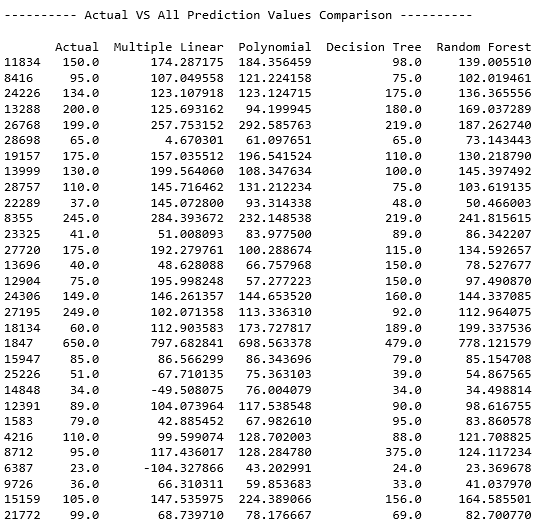


The decision tree model appears to do very well from the $0 - $250 mark, almost mapping exactly with the actual values. From the $250 - $800 mark, it seems to map to the actual values quite inaccurately. This part appears to be more inaccurate than the polynomial model which could explain where the cause of the model having a slightly higher RMSE score, than polynomial model, would come from.



Lastly, our best performing model, the random forest model, appears to map closely to the actual values from start to end. Even though it does not map as well as the decision tree from $200 - $280, it tries to be consistent to the actual values from start to end. This would help explain why this model has a much better RMSE score than the multiple linear and decision tree models.

The figure below will show a portion of all the predicted values of our models versus the actual value. A quick observation will show that the random forest model does well to predict reasonablely close enough to the actual values each time with some values almost matching exactly.



Hence, based on the RMSE score, scatterplot and distribution plots, the random forest model would be the best model for the purposes of our problem statement. Although it would have been more ideal to obtain a lower RMSE score, we would have needed more data and perhaps tried to adopt an alternative approach to dealing with missing data rather than dropping rows entirely. In addition, we will need to pinpoint where exactly the data imbalances come from which affects the performance of the models at each run, with some performances doing drastically worse than other runs.

It is important to note that during the training process there were 2 particular steps taken to help reduce the RMSE score with our existing dataset. Initially, the RMSE was close to 150-200 for all models. We decided to shuffle the data as they may have been sorted in a particular order which affected the splitting of data. Some vital data which gave a fuller picture of the dataset could have been divided into test data completely. In addition, we reduced the percentage of test data to be split so that more data could be given for training, initially test data was given a 20% amount. With these 2 steps, the general RMSE score across all models fell by about 20 – 45%.

## 5.2 Classification

# 6 Summary

As mentioned earlier, for our regression models we had differing performances for each model whenever we ran our script. The general 2 best performances were from our polynomial and random forest models. On some runs, even though the RMSE would be slightly lower for the polynomial than the random forest model, the polynomial model would give some negative value predictions while the random forest would always give a positive value and an amount that was a decent enough prediction most of the time. Given these circumstances, the random forest model would be a better and more consistent model to choose for our regression problem.

By utilizing this model, we can provide a way to give a recommendation of pricing whenever a user wants to lease their property on any form of home rental website. This also helps to expand upon our home rental recommendation system which we built in semester one.

# References

1. <https://ipropertymanagement.com/airbnb-statistics/#targetText=The%20target%20market%20for%20Airbnb,guest%20arrivals%20at%20Airbnb%20listings.>
2. <https://www.sas.com/en_sg/insights/analytics/machine-learning.html>
3. <https://towardsdatascience.com/machine-learning-general-process-8f1b510bd8af>
4. <https://www.youtube.com/watch?v=wpQiEHYkBys>