"Housing Price Property Sales in Canberra"

"Khaled M Elshamy"

"6/15/2020"

I. Project Overview

This document describes the data and methods used to develop and apply a machine learning algorithm to predict Canberra Real Estate Sales 2006-2019. The dataset used on the development and testing of the machine learning algorithm to predict PRICE is titled **Canberra Real Estate Sales 2006-2019** and comes from **Kaggle** internet website,"https://www.kaggle.com/htagholdings/canberra-real-estate-sales-20062019" a popular website used in hosting and making datasets publicly available. A machine learning algorithm, using techniques both linear and non-linear, will be developed and tested to predict Price. The overall accuracy in terms of root mean square (RMSE) value will be used as the metric to measure the overall effectiveness of the algorithm.

II. Introduction

We use this data and other data sources to forecast property market trends in Canberra. You can see some of the reports here: https://www.htag.com.au/act. The data was acquired via a web crawler.

The **Canberra Real Estate Sales 2006-2019** file have 43,178 rows and 11 columns. There are 7 columns where:

datesold is an integer, represents Date of Sale.

price is an integer, identifies Sale Price.

suburb is character, represents Name of the suburb.

parking is an integer, Number of parking spaces.

bathrooms is integer, Number of bathrooms .

bedrooms is integer, Number of bedrooms .

propertyType is character, represents Type of property. i.e. house or unit

```
############################################################
# Preparing the data and Create train and test datasets files
############################################################
##https://www.kaggle.com/htagholdings/canberra-real-estate-sales-20062019
library(tidyverse)
library(dplyr)
library(caret)
library(tidyverse)
library(caret)
library(readr)
housing_price <- read_csv("property_sales_canberra.csv")

##Convert records to numeric or character
```

```
h_price_conv <- as.data.frame(housing_price) %>% mutate(DATESOLD = datesold,
                                                        PRICE = as.numeric(price),
                                                        PRICE_M =as.numeric(price)/1000000,
                                                        SUBURB = as.character(suburb),
                                                        POSTCODE = as.numeric(postcode),
                                                        PARKING = as.numeric(parking),
                                                        BATHROOMS = as.numeric(bathrooms),
                                                        BEDROOMS = as.numeric(bedrooms),
                                                        PROPERTYTYPE = as.character(propertyType),
                                                        SUBURBID = suburbid)

housing_price_sales <- h_price_conv
housing_price_sales[housing_price_sales == "N/A"] <- NA
h_price <- na.omit(housing_price_sales)


set.seed(1)
test_index <- createDataPartition(y = h_price$PRICE_M, times = 1, p = 0.1, list = FALSE)
train_set <- h_price[-test_index,]
temp <- h_price[test_index,]
# Make sure suburb and postcode in test set are also in train set
test_set <- temp %>%
  semi_join(train_set, by = "SUBURB") %>%
  semi_join(train_set, by = "POSTCODE")
# Add rows removed from test set back into train set
removed <- anti_join(temp, test_set)
```

```
train_set <- rbind(train_set, removed)
```

III. Methods and Analysis

The method and analysis used to develop the machine learning algorithm for predicting Price follow the methods found in Rafael Irizarry"s Harvard EDX class titled **Machine Learning**. This online class teaches basic principles and techniques of machine learning. Of interest, from the course, the techniques and applications of regression used in the Movielens example are followed closely and applied here to this project.

A. Exploratory Data Analysis

A1. *Data Extraction*

The **Canberra Real Estate Sales 2006-2019** was downloaded from the Kaggle website using the following link:

"https://www.kaggle.com/htagholdings/canberra-real-estate-sales-20062019"

Downloading at Kaggle requires an account which is available and free upon sign-up at the web site. This downloaded document is provided in a compressed, zip file format that requires extraction before usage in a spreadsheet or database application. For this project, the file was downloaded and extracted as a csv file. The data analysis and work were performed on a computer using Rstudio, a computer platform application that uses the R computer language. The data analysis, machine learning code, and report write-up were all completed on Rstudio.

Following Rafael Irizarry's **Data Science Program** class, the data file was then partitioned into two datasets, (1) a training set and (2) a test set. The training set is used to model the data and is partitioned from the

original dataset and contains roughly 90 percent of the original data. The test set is used to validate the model and contains the remainder or about 10 percent of the overall data.

A2. *Data Cleaning*

Data in its raw state generally has some defects or deficiencies such as format or errors that require correction. This task involves some type of data cleaning or curating the data to make it useful for the application of this project.

Initial review showed that there is missing, blank, or non-recorded observations as well as data recorded as N/As in the dataset. Because a complete set of data is an essential requirement for machine learning application, missing data and N/As were removed from the dataset. The original dataset contained 43,178 observations. Because of the cleaning and removal of non-recorded observations (blanks) and recorded missing observations (N/As), the curated data was left with only 41,777 observations.

A3. *Data Exploration*

Having a curated data file, the data from the file is partitioned into two sets of data: a training set and a test set. The training set is used for model development while the test set is used for tesing the accuracy of the model. In this section data exploration is performed on the training set.
after Convertig records to numeric or character : The first exploration looks at the features and observations in the data. There are 41,777 observations and 21 features in the dataset. They are listed below.

```
#create summary table
h_price_summary <- data.frame(number_of_rows = nrow(h_price),
                          number_of_column = ncol(h_price),
                          number_of_suburb = n_distinct(h_price$SUBURB),
                          number_of_postcode = n_distinct(h_price$POSTCODE),
                          average_price = round(mean(h_price$PRICE),2),
                          number_of_parking = n_distinct(h_price$PARKING),
                          number_of_BATHROOMS = n_distinct(h_price$BATHROOMS),
                          number_of_bedrooms = n_distinct(h_price$BEDROOMS),
                          number_of_propertytype = n_distinct(h_price$PROPERTYTYPE),
                          the_first_sell_date = min(h_price$DATESOLD),
                          the_last_sell_date = max(h_price$DATESOLD))

knitr::kable(h_price_summary[,1:5],caption = "Summary of House Price set (part 1)")
```

Table 1: Summary of House Price set (part 1)

number_of_rows	number_of_column	number_of_suburb	number_of_postcode	average_price
41777	21	107	32	604188.8

```
knitr::kable(h_price_summary[,6:9],caption = "Summary of House Price set (part 2)")
```

Table 2: Summary of House Price set (part 2)

number_of_parking	number_of_BATHROOMS	number_of_bedrooms	number_of_propertytype
18	9	13	2

```
knitr::kable(h_price_summary[,10:11],caption = "Summary of House Price set (part 3)")
```

Table 3: Summary of House Price set (part 3)

the_first_sell_date	the_last_sell_date
1/01/2016	9/12/2018

```
dim(train_set)
```

```
[1] 37597    21
```

A summary of the data layout is given below. The observations contain both numerical and character values. Additionally, the year is given as a data value.

```
str(train_set[,12:21])
```

```
'data.frame':   37597 obs. of  10 variables:
 $ DATESOLD    : chr  "9/06/2000" "21/09/2005" "22/11/2005" "17/01/2006" ...
 $ PRICE       : num  223000 276000 350000 270000 272500 ...
 $ PRICE_M     : num  0.223 0.276 0.35 0.27 0.273 ...
 $ SUBURB      : chr  "Nicholls" "Isabella Plains" "Theodore" "Gordon" ...
 $ POSTCODE    : num  2913 2905 2905 2906 2905 ...
 $ PARKING     : num  2 1 2 1 1 1 0 1 1 2 ...
 $ BATHROOMS   : num  2 1 2 1 1 1 2 1 2 2 ...
 $ BEDROOMS    : num  4 3 3 3 3 3 3 3 3 4 ...
 $ PROPERTYTYPE: chr  "house" "house" "house" "house" ...
 $ SUBURBID    : chr  "ACT708" "ACT612" "ACT610" "ACT616" ...
```

To help visualize the data, the first ten rows of the data are given below.

First Ten Rows of Dataset :

```
train_set1  <-  train_set %>%
  select(12:18)
head(train_set1, n=10) %>%
  knitr::kable()
```

	DATESOLD	PRICE	PRICE_M	SUBURB	POSTCODE	PARKING	BATHROOMS
1	9/06/2000	223000	0.2230	Nicholls	2913	2	2
4	21/09/2005	276000	0.2760	Isabella Plains	2905	1	1
6	22/11/2005	350000	0.3500	Theodore	2905	2	2
11	17/01/2006	270000	0.2700	Gordon	2906	1	1
12	7/02/2006	272500	0.2725	Chisholm	2905	1	1
13	17/02/2006	280500	0.2805	Gordon	2906	1	1
16	1/03/2006	255000	0.2550	Conder	2906	0	2
17	31/03/2006	250000	0.2500	Theodore	2905	1	1
18	2/05/2006	250000	0.2500	Conder	2906	1	2
19	2/05/2006	420000	0.4200	Conder	2906	2	2

```
train_set2  <-  train_set %>%
  select(19:21)
head(train_set2, n=10) %>%
  knitr::kable()
```

	BEDROOMS	PROPERTYTYPE	SUBURBID
1	4	house	ACT708
4	3	house	ACT612
6	3	house	ACT610
11	3	house	ACT616
12	3	house	ACT609
13	3	house	ACT616
16	3	house	ACT613
17	3	house	ACT610
18	3	house	ACT613
19	4	house	ACT613

Statistical analysis was reviewed on the data. The table below presents a five number summary of the Price data.

Five Number Summary for Price

```
train_set3 <- train_set %>%
  select(PRICE_M)
summary_price <- summary(train_set3)
knitr::kable(summary_price,caption = "Summary of Price ")
```

Table 6: Summary of Price

PRICE_M
Min. :0.0500
1st Qu.:0.4300
Median :0.5390
Mean :0.6049
3rd Qu.:0.6775
Max. :8.0000

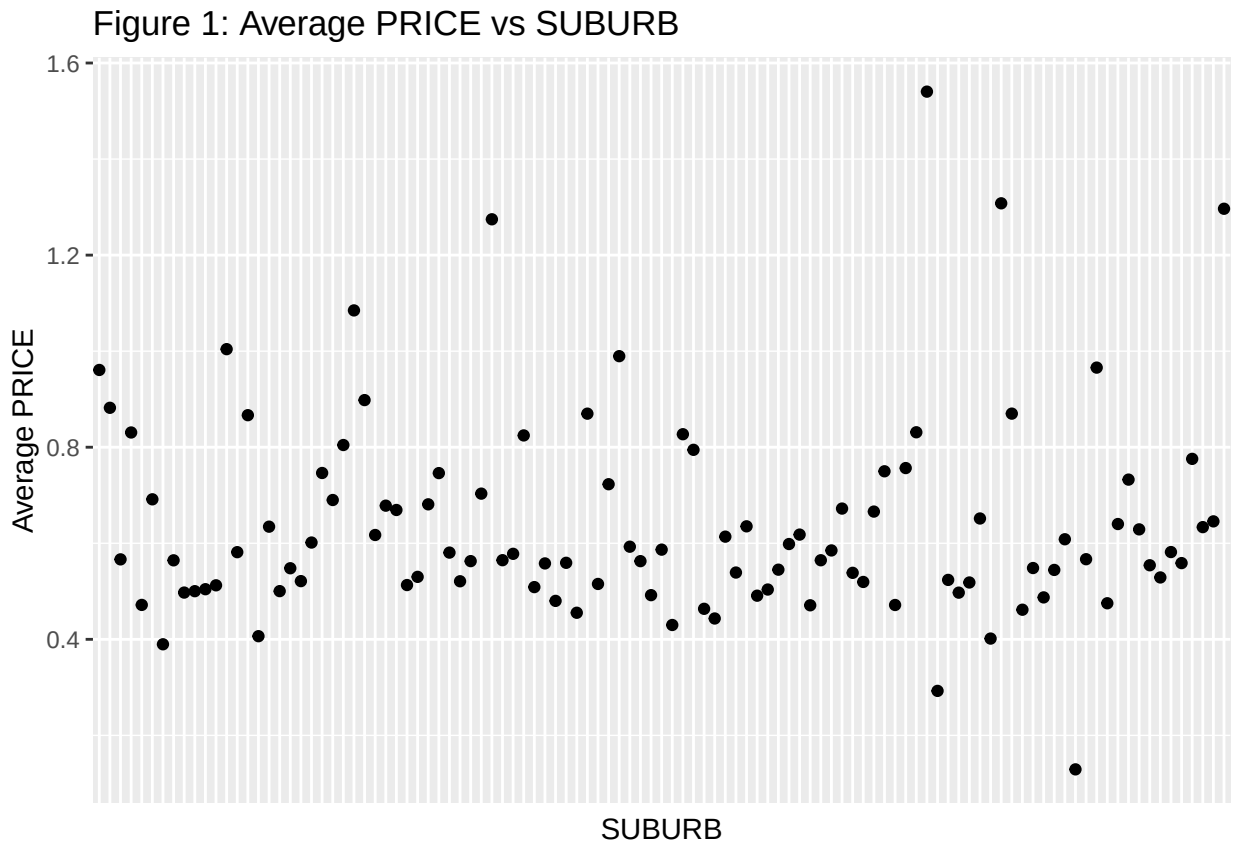
A4. *Data Visualization*

The data was plotted to see different trends in the data.

Figure 1 shows a scatter plot of Price average versus SUBURB. From this figure, we see that SUBURB does have an impact on Price. The miscelaneous SUBURB is the rate highest in terms of Price while the strategy SUBURB has the lowest Price overall. Also, the average Price for each SUBURB ranges roughly from 0.4 to 1.

```
#plot by SUBURB
c <- h_price %>%
  mutate(n=1) %>%
  group_by(SUBURB) %>%
  summarize(average=mean(PRICE_M))

ggplot(c, aes(x=SUBURB, average)) +
  geom_point()+
  labs(y = "Average PRICE")+
  ggtitle("Figure 1: Average PRICE vs SUBURB") +
 theme(axis.text.x=element_blank(),
       axis.ticks.x=element_blank())
```

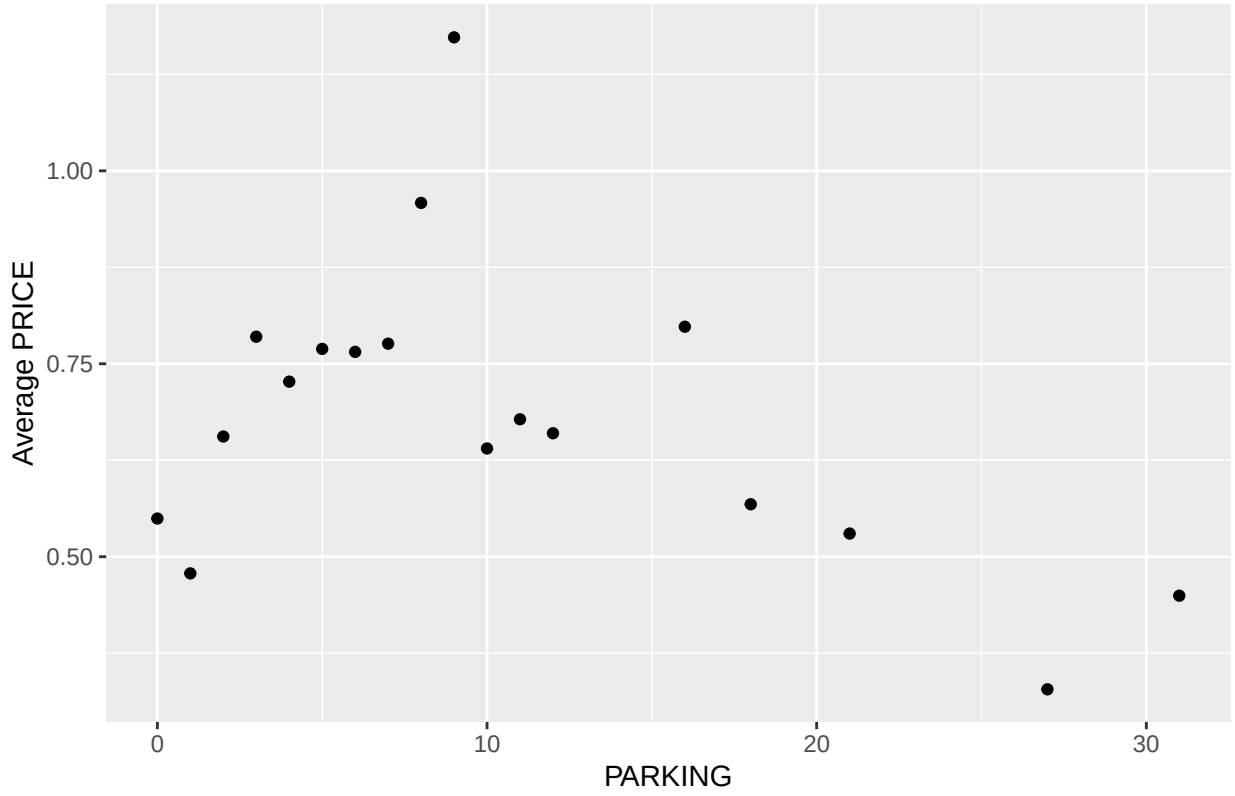


The PARKING effect on Price was also reviewed to see any corresponding trends.

```
c <- h_price %>%
  mutate(n=1) %>%
  group_by(PARKING) %>%
  summarize(average=mean(PRICE_M))

ggplot(c, aes(PARKING, average))+
  geom_point()+
  ggtitle("Figure 2: Average PRICE vs PARKING")+
  labs(y = "Average PRICE")#+
```

Figure 2: Average PRICE vs PARKING



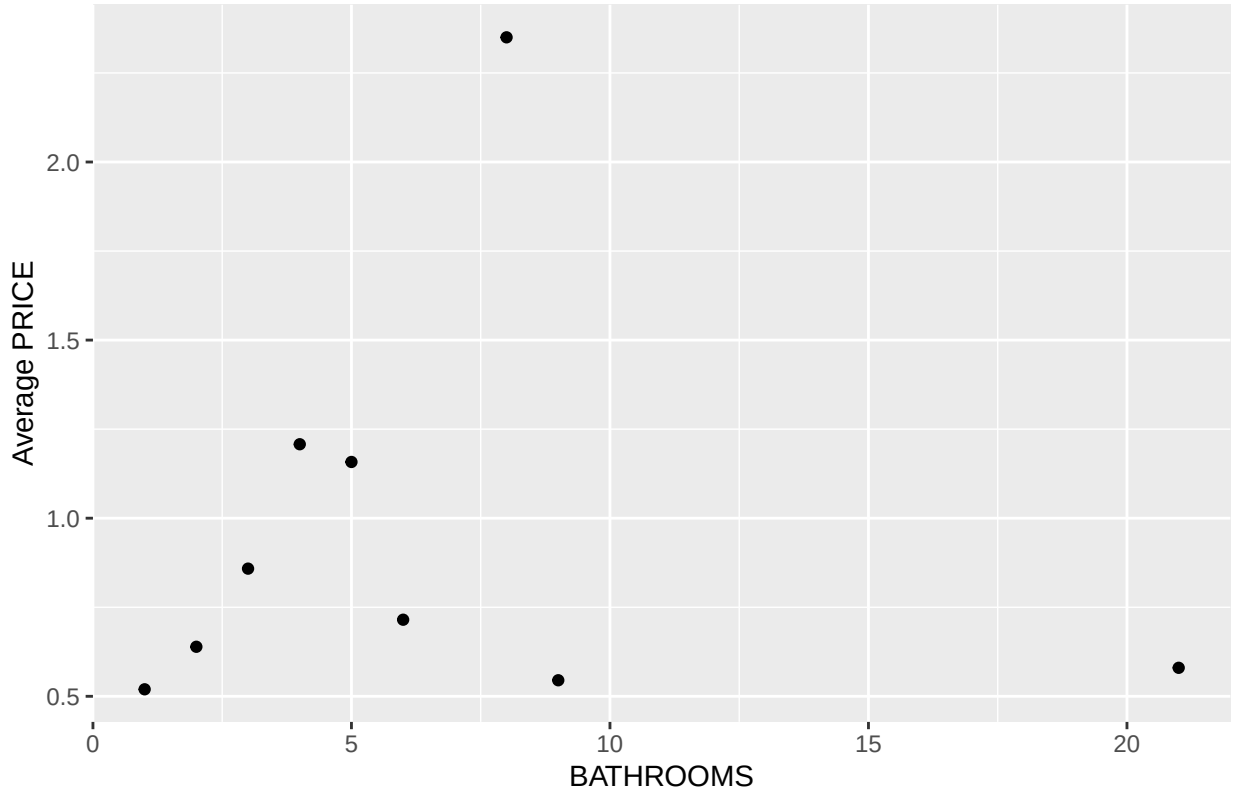

```
 # theme(axis.text.x=element_blank(),
 #       axis.ticks.x=element_blank())
```

An analysis of the number BATHROOMS was release compared to Price. From Figure 3, the number of BATHROOMS shows price to be at the lowest point when one bathroom only , Also, the average Price increase when the number of BATHROOMS increase as shown in the figure.

```
c <- h_price %>%
  mutate(n=1) %>%
  group_by(BATHROOMS) %>%
  summarize(average=mean(PRICE_M))
d <- c %>%
  mutate(year=as.numeric(BATHROOMS))

ggplot(d, aes(BATHROOMS, average))+
  geom_point() +
  ggtitle("Figure 3: Average PRICE vs BATHROOMS")+
  labs(y = "Average PRICE", x="BATHROOMS")
```

Figure 3: Average PRICE vs BATHROOMS

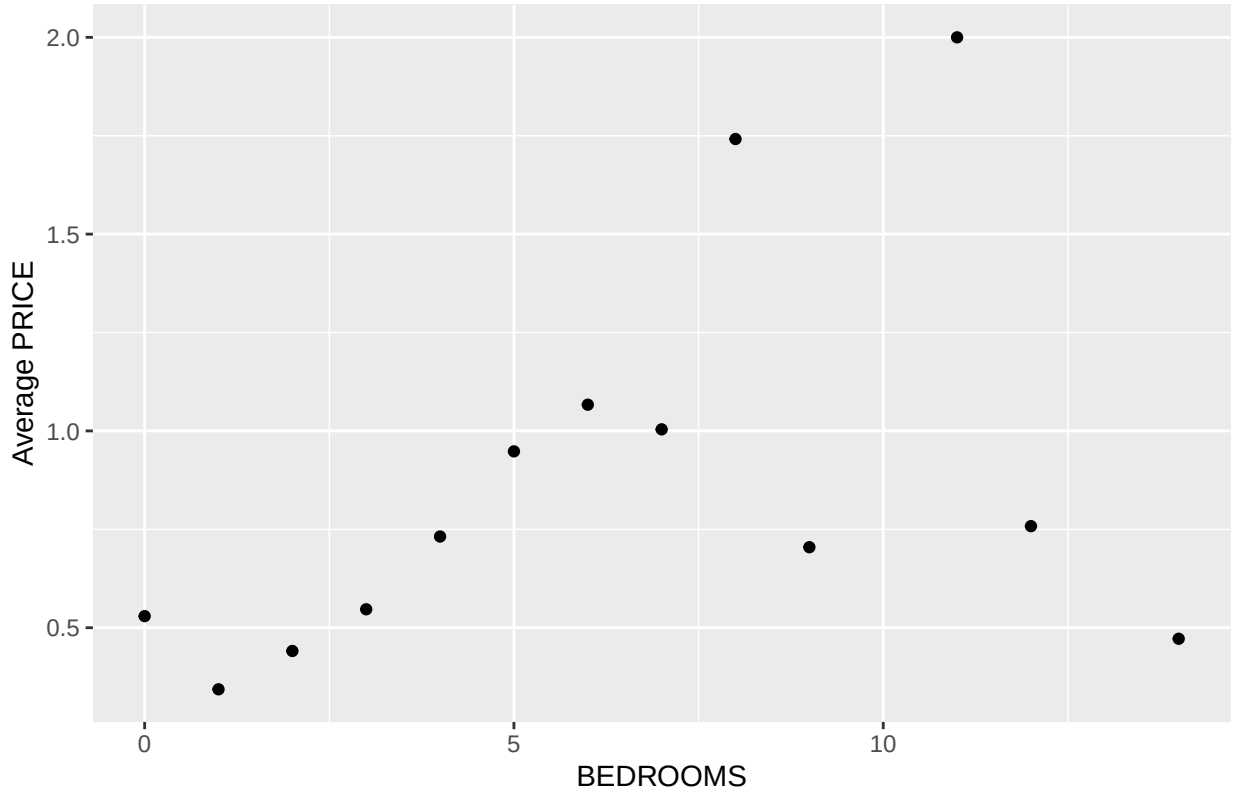


An analysis of the number BEDROOMS was release compared to Price. From Figure 4, the number of BEDROOMS shows price to be at the lowest point when one bedroom only , Also, the average Price increase when the number of BEDROOMS increase as shown in the figure.

```
c <- h_price %>%
  mutate(n=1) %>%
  group_by(BEDROOMS) %>%
  summarize(average=mean(PRICE_M))
d <- c %>%
  mutate(year=as.numeric(BEDROOMS))

ggplot(d, aes(BEDROOMS, average))+
  geom_point() +
  ggtitle("Figure 4: Average PRICE vs BEDROOMS")+
  labs(y = "Average PRICE", x="BEDROOMS")
```

Figure 4: Average PRICE vs BEDROOMS



An analysis of the PROPERTYTYPE was release compared to Price. From Figure 5, the PROPERTYTYPE shows price to be at the lowest point for unit . Also, the average Price at the lowest point for House There is the PROPERTYTYPE trend noticeable in the figure.

```
#plot by PROPERTYTYPE
c <- h_price %>%
  mutate(n=1) %>%
  group_by(PROPERTYTYPE ) %>%
  summarize(average=mean(PRICE_M))

ggplot(c, aes(x=PROPERTYTYPE , average)) +
  geom_point()+
  labs(y = "Average PRICE")+
  ggtitle("Figure 5: Average PRICE vs PROPERTYTYPE ")
```

Figure 5: Average PRICE vs PROPERTYTYPE

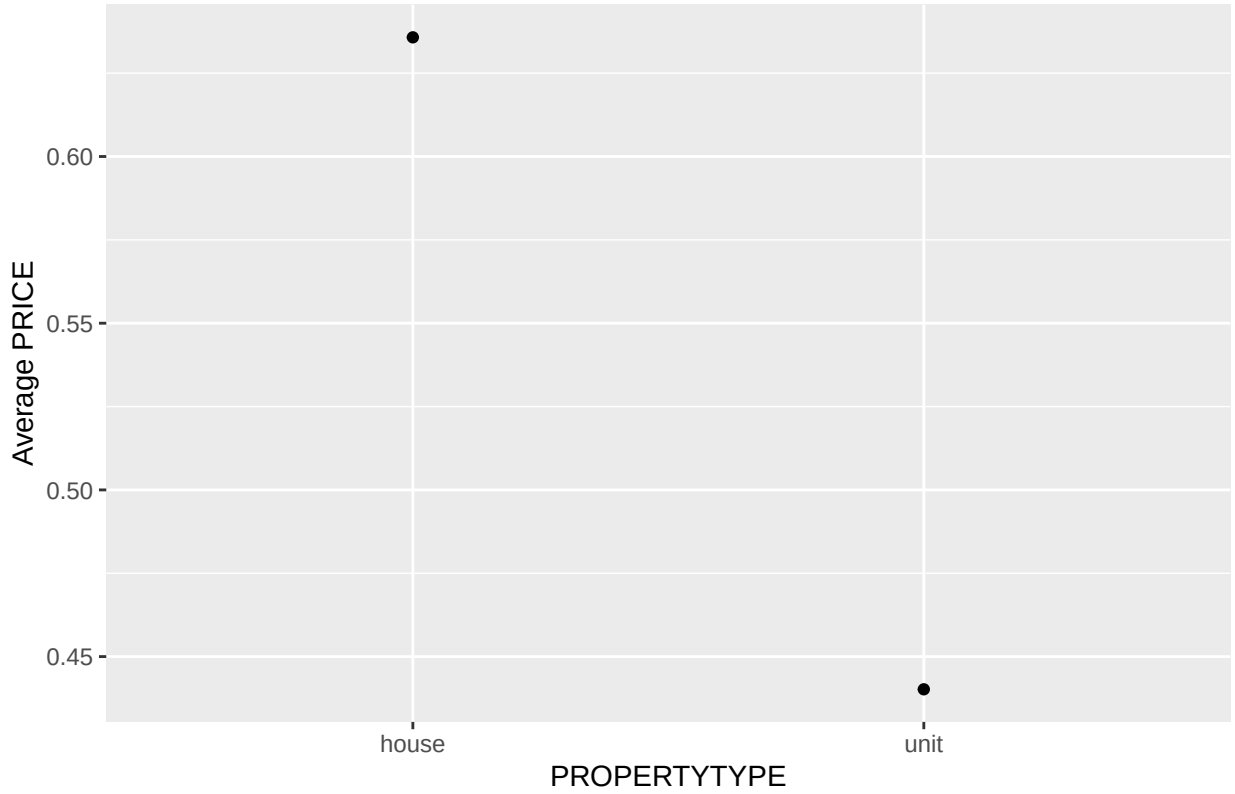
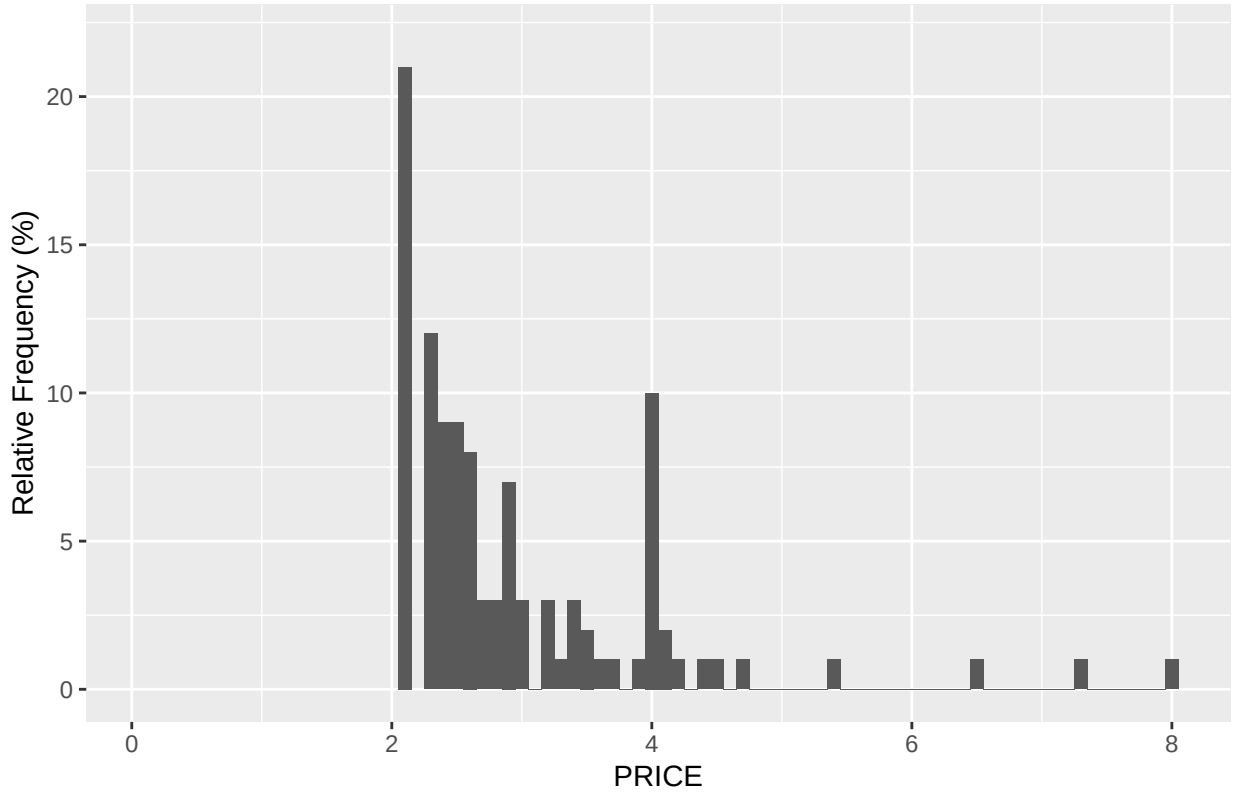


Figure 6 below shows a histogram of Price relative frequency. Most of the House have Price within roughly 2 to 10 million units range. The largest Price is over 10 milion. There is a large gap between 4 to 10 million units range, making over 4 millions units point an obvious outlier. It was found that only small number of House fit this points.

```
train_set %>%
  ggplot(aes(PRICE_M)) +
  geom_histogram(binwidth = 0.1) +
  scale_x_continuous() +
  scale_y_continuous() +
  xlab("PRICE") +
  ylab("Relative Frequency (%)") +
  ggtitle("Figure 6: Global PRICE Histogram")+
  ylim(0, 22)
```

Figure 6: Global PRICE Histogram



B. Data Modeling

To build an algorithm based on observations of features, various methods will be utilized to predict Price based on features. In this project, we will only focused on the following features: (1) SUBURB, (2) PARKING, (3) BATHROOMS, (4) BEDROOMS ,(5) PROPERTYTYPE . The models use include just the average, linear regression, and prediction based on all 5 features effects.

B1. *Modeling Using the Global Sales Average*

This model takes Price and used an average value of the total Price as the predictor. The predictor is then tested on the test set to determine the RMSE. This will serve as the baseline for comparison with RMSEs from other models.

mu=sum(Price/number Price)

y_hat = mu

where:

y_hat = predicted Price for any input

mu = average of total Price

B2. *Linear Model Price Average by SUBURB*

The linear regression model uses simple linear regression to create a prediction line based on the relationship between the SUBURB and Price.

y_hat = b_o + b_1 * x

where:

y_hat = predicted Price for a given SUBURB

x = SUBURB

b_o = y-intercept for regression line

b_1 = slope of regression line

B3. *Model Based on SUBURB and PARKING Effect*

To build an algorithm based on observations of features, an equation was developed and modeled to predict price. This equation correlates features to Price and takes account errors and biases in the features.

We begin with a baseline model (see B1 above) where we assume an average value of Price and calculate the root mean square error (RMSE) value on it. This simple method will establish a baseline RMSE value that uses the same predictor (average Price) for all titles in our dataset.

To improve this model, we add the SUBURB and PARKING classes biases. The modeling equation is provided as:

y_s,p = mu + b_s + b_p + e_s,p

where:

y_s,p= predicted Price for all houses, for SUBURB (s) for PARKING (p)

mu = "true" Price of all Houses

b_SUBURB = SUBURB effect for Houses (s)

b_PARKING = PARKING effect for Houses (p)

e_s,p= residual (errors)

IV Results

The results are presented in this section. For each model, an RMSE value was calculated based on predicted value for the test set.

Result for Model B1

The average model was first used to established an RMSE baseline. In this model, the average Price was used as a predictor (in this case a constant value) to estimate the RMSE value against the test set values. An RMSE of 0.3165149 was calculated based on this average value, see Table 1 below.

The value of mu is this:

```
mu <- mean(train_set$PRICE_M)
#define mu (average) of PRICE_M column in train_set
#dataset PRICE_M (in millions of units)
mu
```

```
[1] 0.6048581
```

The value of the rmse for this model is this:

```
naive_rmse <- RMSE(train_set$PRICE_M, mu)
```

```
      #As we go along, we will be comparing different approaches.
      #Let's start by creating a results table with this naive approach
```

```
rmse_results <- tibble(method = "Just the Average", RMSE = naive_rmse)
rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	0.3165149

Result for Model B2

The linear model was first used to established a second RMSE. In this model, a linear regression was used to obtain an RMSE value on Price. An RMSE of 0.2477835 was calculated based on this average value which is an improvement over model B1.

```
#-------------------------------------------------------------------------

fit_lm <- lm(PRICE_M ~ SUBURB, data=train_set)
lm_predict <- predict(fit_lm, test_set)
rmse_lm <- RMSE(test_set$PRICE_M, lm_predict)

rmse_results <- bind_rows(rmse_results,
                          tibble(method="Linear Model",
                                 RMSE = rmse_lm))  # putting data into a table
rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	0.3165149
Linear Model	0.2477835

Result for Model B3

Lastly, the model based on the SUBURB and PARKING effect was used to calculate an RMSE value. In this model, both SUBURB and PARKING effects were modeled as biases into the equation to obtain an RMSE value on PRICE. The RMSE value for this model is 0.2319912 which is which is an improvement over model B2.

```
SUBURB_avgs <- train_set %>%  #incorporating SUBURB effect into equation,
                              #b_i is for SUBURB
  group_by(SUBURB) %>%
  summarize(b_SUBURB = mean(PRICE_M - mu))

predicted_PRICE <- mu +
  test_set %>%  #testing predicted PRICE data to test_set dataset
  left_join(SUBURB_avgs, by = 'SUBURB') %>%
  pull(b_SUBURB)

# model_1_rmse <- RMSE(predicted_PRICE, test_set$PRICE_M)  #this test SUBURB effect

PARKING_avgs <- train_set %>%
  left_join(SUBURB_avgs, by='SUBURB') %>%
  group_by(PARKING) %>%
  summarize(b_PARKING = mean(PRICE_M - mu - b_SUBURB))

predicted_PRICE_PARKING <- test_set %>%   #putting both SUBURB and PARKING into model
```

```
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  mutate(pred = mu +b_PARKING + b_SUBURB) %>%
  #na.omit() %>%
  pull(pred)

model_2_rmse <- RMSE(predicted_PRICE_PARKING, test_set$PRICE_M)
#calculate residual mean square error for the two effects
rmse_results <- bind_rows(rmse_results,
                          tibble(method="PARKING + SUBURB Effects Model",
                                 RMSE = model_2_rmse))  # putting data into a table
```

the model based on the SUBURB ,PARKING and Bathrooms effect was used to calculate an RMSE value. In this model, The RMSE value for this model is 0.2266636 which is which is more an improvement .

```
BATHROOMS_avgs <- train_set %>%               # adding BATHROOMS into model
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  group_by(BATHROOMS) %>%
  summarize(b_BATHROOMS = mean(PRICE_M - mu - b_SUBURB - b_PARKING))

predicted_PRICE_PARKING_BATHROOMS <- test_set %>%
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  left_join(BATHROOMS_avgs, by='BATHROOMS') %>%
  mutate(pred = mu +b_PARKING + b_SUBURB + b_BATHROOMS) %>%
  #na.omit() %>%
  pull(pred)



model_3_rmse <- RMSE(predicted_PRICE_PARKING_BATHROOMS, test_set$PRICE_M)
#calculate residual mean square error for the two effects
rmse_results <- bind_rows(rmse_results,
                          tibble(method="PARKING + SUBURB + BATHROOMS Effects Model",
                                 RMSE = model_3_rmse))  # putting data into a table
```

the model based on the SUBURB ,PARKING ,Bathrooms and BEDROOMS effect was used to calculate an RMSE value. In this model,The RMSE value for this model is 0.2155541 which is which is more an improvement .

```
BEDROOMS_avgs <- train_set %>%               # adding BEDROOMS into model
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  left_join(BATHROOMS_avgs, by='BATHROOMS') %>%
  group_by(BEDROOMS) %>%
  summarize(b_BEDROOMS = mean(PRICE_M - mu - b_SUBURB - b_PARKING - b_BATHROOMS))

predicted_PRICE_PARKING_BATHROOMS_BEDROOMS <- test_set %>%
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  left_join(BATHROOMS_avgs, by='BATHROOMS') %>%
  left_join(BEDROOMS_avgs, by='BEDROOMS') %>%
  mutate(pred_BEDROOMS = mu +b_PARKING + b_SUBURB + b_BATHROOMS +b_BEDROOMS) %>%
  #na.omit() %>%
```

```
  pull(pred_BEDROOMS)



model_4_rmse <- RMSE(predicted_PRICE_PARKING_BATHROOMS_BEDROOMS, test_set$PRICE_M)
#calculate residual mean square error for the two effects
rmse_results <- bind_rows(rmse_results,
    tibble(method="PARKING + SUBURB + BATHROOMS + BEDROOMS Effects Model",
                  RMSE = model_4_rmse))  # putting data into a table
```

the model based on the SUBURB ,PARKING ,Bathrooms, BEDROOMS and PROPERTYTYPE effect was used to calculate an RMSE value. In this model,The RMSE value for this model is 0.2154278 which is which is more an improvement .

```
PROPERTYTYPE_avgs <- train_set %>%               # adding PROPERTYTYPE into model
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  left_join(BATHROOMS_avgs, by='BATHROOMS') %>%
  left_join(BEDROOMS_avgs, by='BEDROOMS') %>%
  group_by(PROPERTYTYPE) %>%
  summarize(b_PROPERTYTYPE = mean(PRICE_M - mu - b_SUBURB - b_PARKING - b_BATHROOMS
                                  - b_BEDROOMS))

predicted_PRICE_PARKING_BATHROOMS_BEDROOMS_PROPERTYTYPE <- test_set %>%
  left_join(SUBURB_avgs, by='SUBURB') %>%
  left_join(PARKING_avgs, by='PARKING') %>%
  left_join(BATHROOMS_avgs, by='BATHROOMS') %>%
  left_join(BEDROOMS_avgs, by='BEDROOMS') %>%
  left_join(PROPERTYTYPE_avgs, by='PROPERTYTYPE') %>%
  mutate(pred_PROPERTYTYPE = mu +b_PARKING + b_SUBURB + b_BATHROOMS +b_BEDROOMS +
           b_PROPERTYTYPE) %>%
  #na.omit() %>%
  pull(pred_PROPERTYTYPE)



model_5_rmse <- RMSE(predicted_PRICE_PARKING_BATHROOMS_BEDROOMS_PROPERTYTYPE,
                     test_set$PRICE_M)
#calculate residual mean square error for the two effects
rmse_results <- bind_rows(rmse_results,
 tibble(method="PARKING + SUBURB + BATHROOMS + BEDROOMS + PROPERTYTYPE Effects Model",
   RMSE = model_5_rmse))  # putting data into a table
```

V. Conclusion

In this project, several machine learning algorithms were developed to predict Price of House. A summary of the performance of each algorithm is presented in the following table:

```
rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	0.3165149
Linear Model	0.2477835

method	RMSE
PARKING + SUBURB Effects Model	0.2319912
PARKING + SUBURB + BATHROOMS Effects Model	0.2266636
PARKING + SUBURB + BATHROOMS + BEDROOMS Effects Model	0.2155541
PARKING + SUBURB + BATHROOMS + BEDROOMS + PROPERTYTYPE Effects Model	0.2154278

We started with a baseline model which only consider the Price average as the predictor. Then, we tried to improve upon the prediction with the development of the linear model which showed substantial improvement from the baseline model with an RMSE of 0.3165149 down to 0.2477835. To seek further improvement, we used the PARKING, SUBURB,BATHROOMS,BEDROOMS,PROPERTYTYPE effects model, which showed the most improvement among all models considered, with an RMSE value of 0.2154278.