# **Housing Market - Time Series Analysis &**

# **Price Prediction using Deep Learning**

**By: Sam Vuong, Raymond (shanhua) Huang, Kyle Murphy, Carmon Ho**

**Abstract:**

The goal of this project is to analyze the trend of Melbourne Australia’s housing market. With the housing market on the rise, it began to see a trend of “cool off”. The success of this project is to determine the cause/s of the housing market that has “cool off”.

Our application on ShinyApps can be accessed at: <https://skvuong.shinyapps.io/project/>

Our application code can be found in GitHub at: <https://github.com/skvuong/housingMarketAnalysis>

# **1. Introduction and Discussion**

Every family needs to have a home to be living in, and not everyone can afford to buy a home. There are a lot of variables for a family to be able to secure the home they want and one of the major factors is price. With this project we will examine Melbourne’s Housing Market on determining factors on why prices have seen decline in January 2016.

We will employ time series technique to analyze when the housing market has declined. We will also use the Neural Network Deep Learning technique to model and predict the house price based on other factors.

## **1.1 Dataset**

We are using the “**Melbourne Housing Market**” from Kaggle website.

The dataset can be found at: <https://www.kaggle.com/anthonypino/melbourne-housing-market>.

The dataset has 34,857 rows and 21 columns.

The column names are:

1. suburb
2. address
3. rooms
4. type
5. price
6. method
7. sellerG
8. date
9. distance
10. postcode
11. bedroom2
12. bathroom
13. car
14. landsize
15. buidlingarea
16. yearbuilt
17. councilarea
18. latitude
19. longitude
20. regionname
21. propertycount

## 

## **1.2 Ethical ML Framework**

In order to conduct our research in an ethically responsible manner we have considered a number of important factors. We have removed all variables that can be linked to personal information of the participants in the dataset. This will prevent the results from discriminating against people based on factors like gender, religion, and region.

The dataset that we are using includes information that is private and personal, like the seller information, as well as house address, postal code and latitude and longitude. We have decided not to have any code to view or print out these information from our dataset in order to protect the privacy of the participants.

## **1.3 Assumptions**

For this dataset we would assume that all submitters live and own a home in Melbourne. Submitters can have different ranges in family size, from being single, married, divorced, widowed, etc, with children, no children, parents, siblings. Submitters would be in a relationship type (couple, married, common-law, etc.)

# **2. Data Preparation**

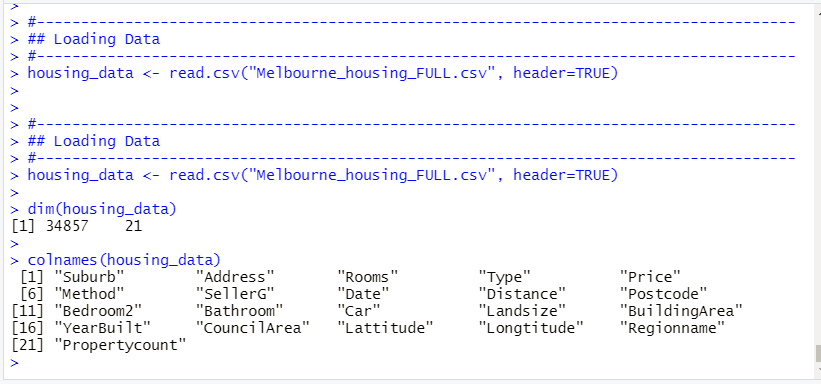
## **2.1 Loading R Libraries**

The following R libraries are used by our application:

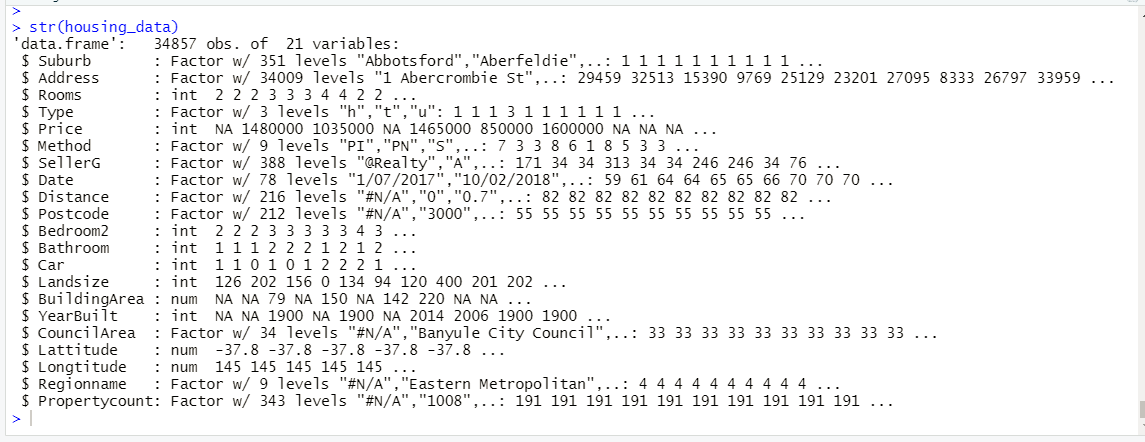
* library(dplyr) #Data manipulation
* library(fastDummies) #Dummies for modeling
* library(corrplot) #Visualizations
* library(ggplot2) #Visualizations
* library(keras) #Deep Learning
* library(magrittr) #Forward-pipe operator
* library(mlbench) #Machine Learning benchmark
* library(neuralnet) #Neural Network
* library(RColorBrewer) #Color palettes
* library(tidyr) #Data manipulation

## **2.2 Loading Data**

We read input data files, check the number of rows and columns and column names..



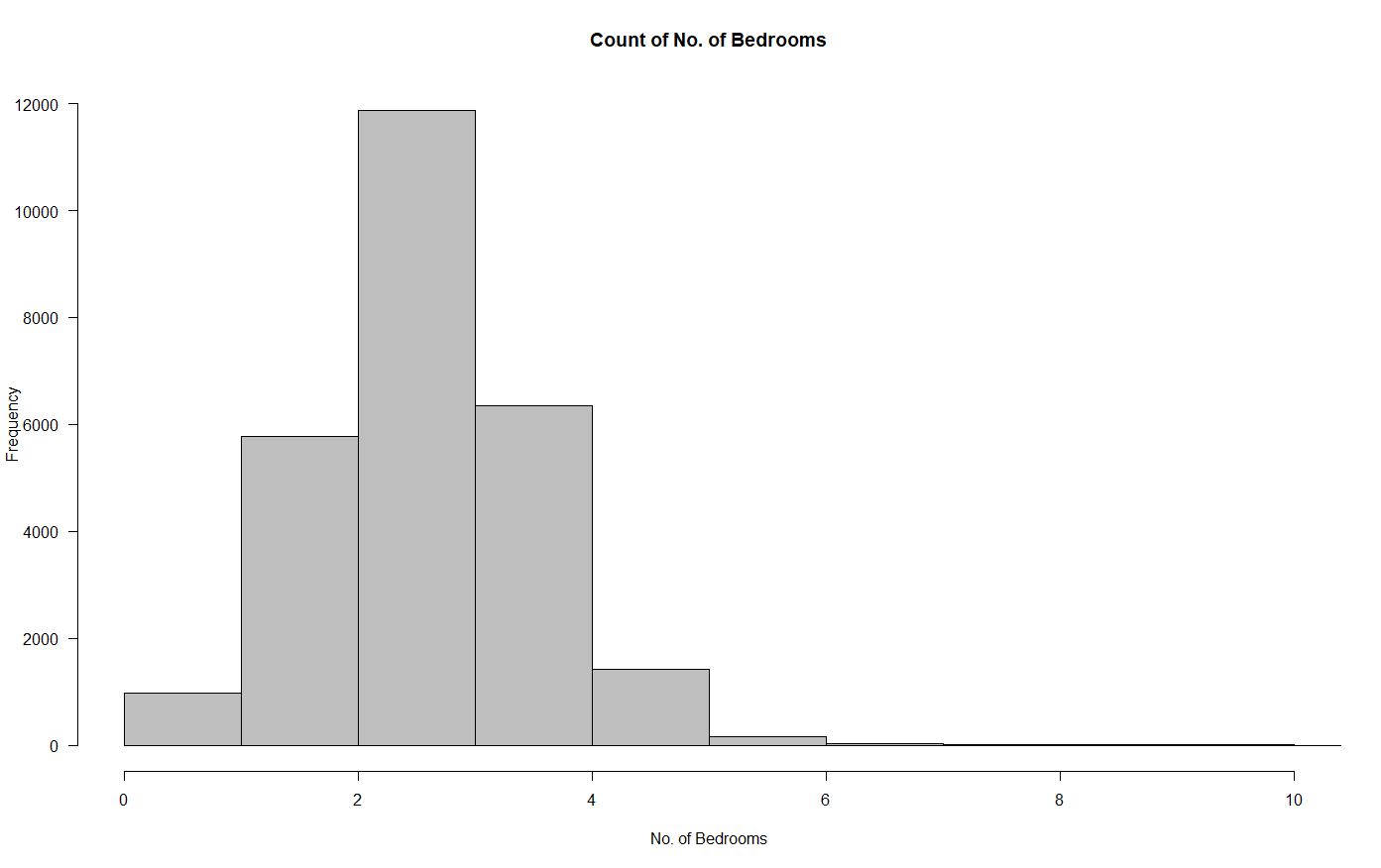
We take a look at the dataframe structure and some data:



# **3. Data Exploration and Visualization**

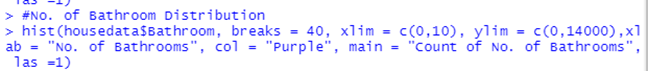
**1. Number of Bedrooms Per Home**

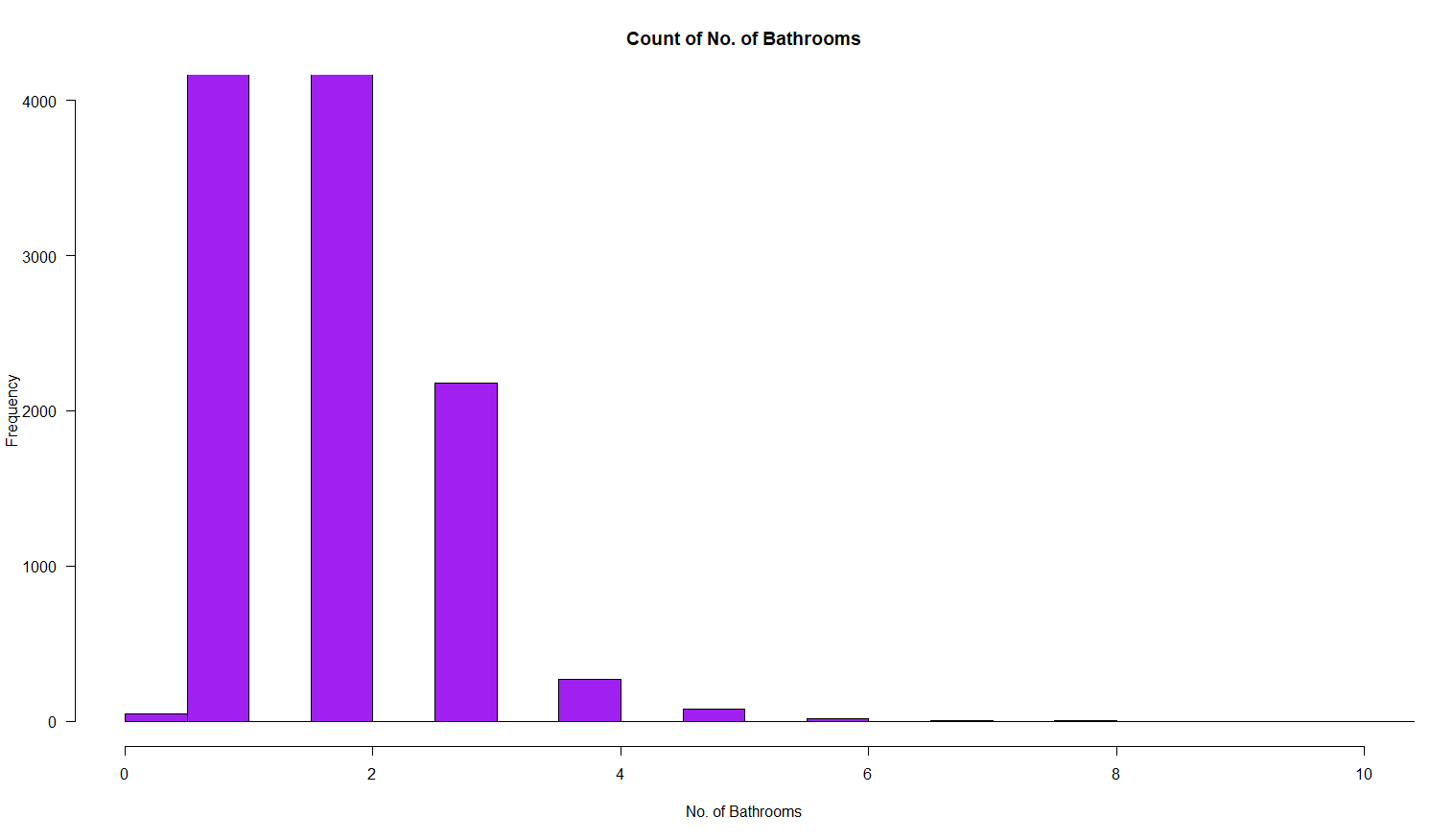
****

****

Melbourne homes have an average of 3 bedrooms. 25% of homes have less than 3 bedrooms, 45% of homes have 3 bedrooms, and 30% of homes have 4 or more,up to a maximum of 30 bedrooms.

**2. Number of Bathrooms Per Home**

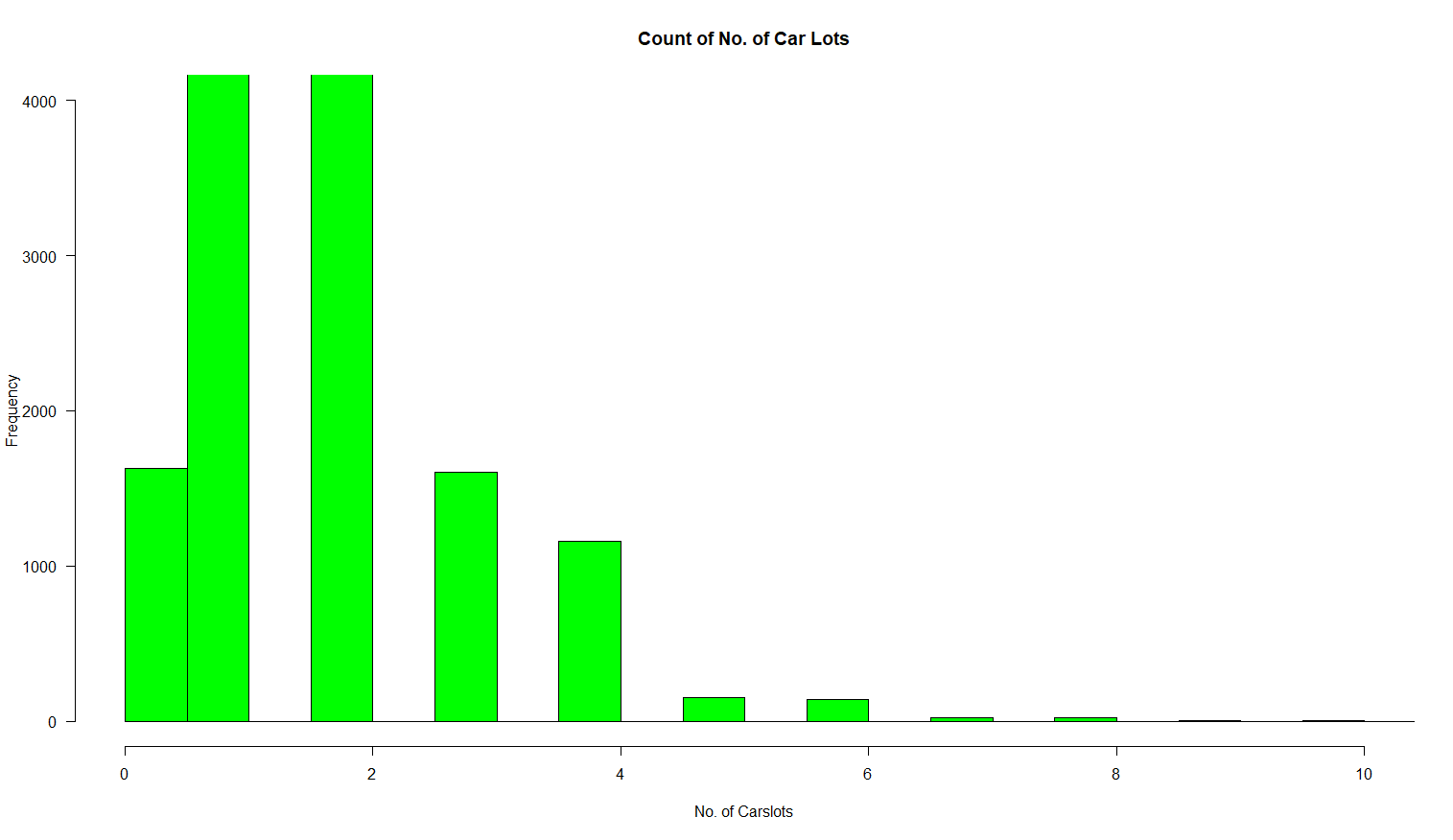
****

****

Melbourne homes have an average of 2 bathrooms per home. 90% of homes have between 1 and 2 bathrooms and 10% of homes have 3 or more bathrooms, up to a maximum of 12.

**3. Number of Car lots Per Home**

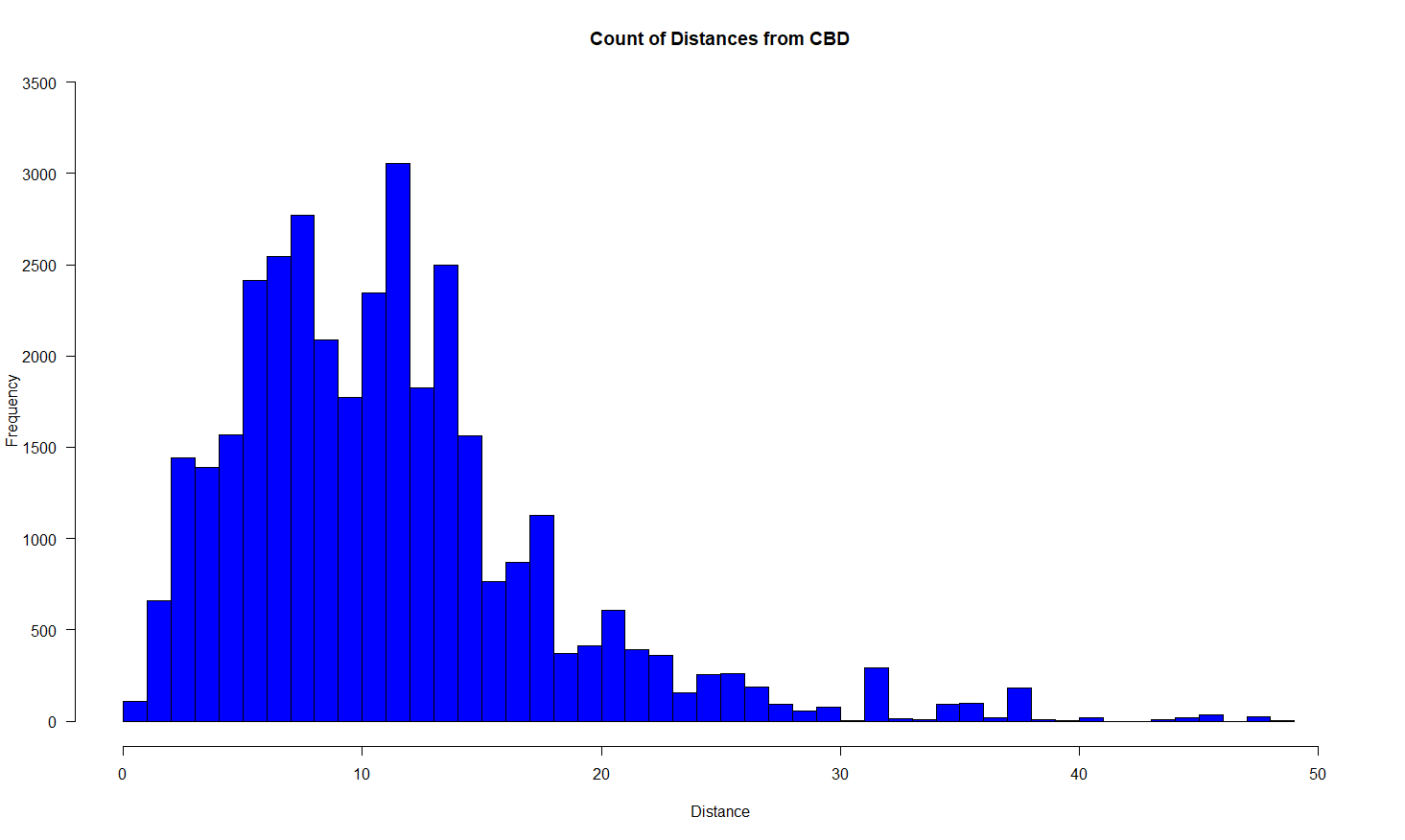
****

****

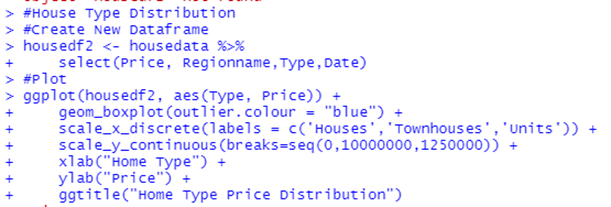
Melbourne homes have an average of 2 car lots per home. 82% of homes have between 1 and 2 car lots, 6% of homes have 0 car lots, and 12% of homes have 3 or more car lots, up to a maximum of 26.

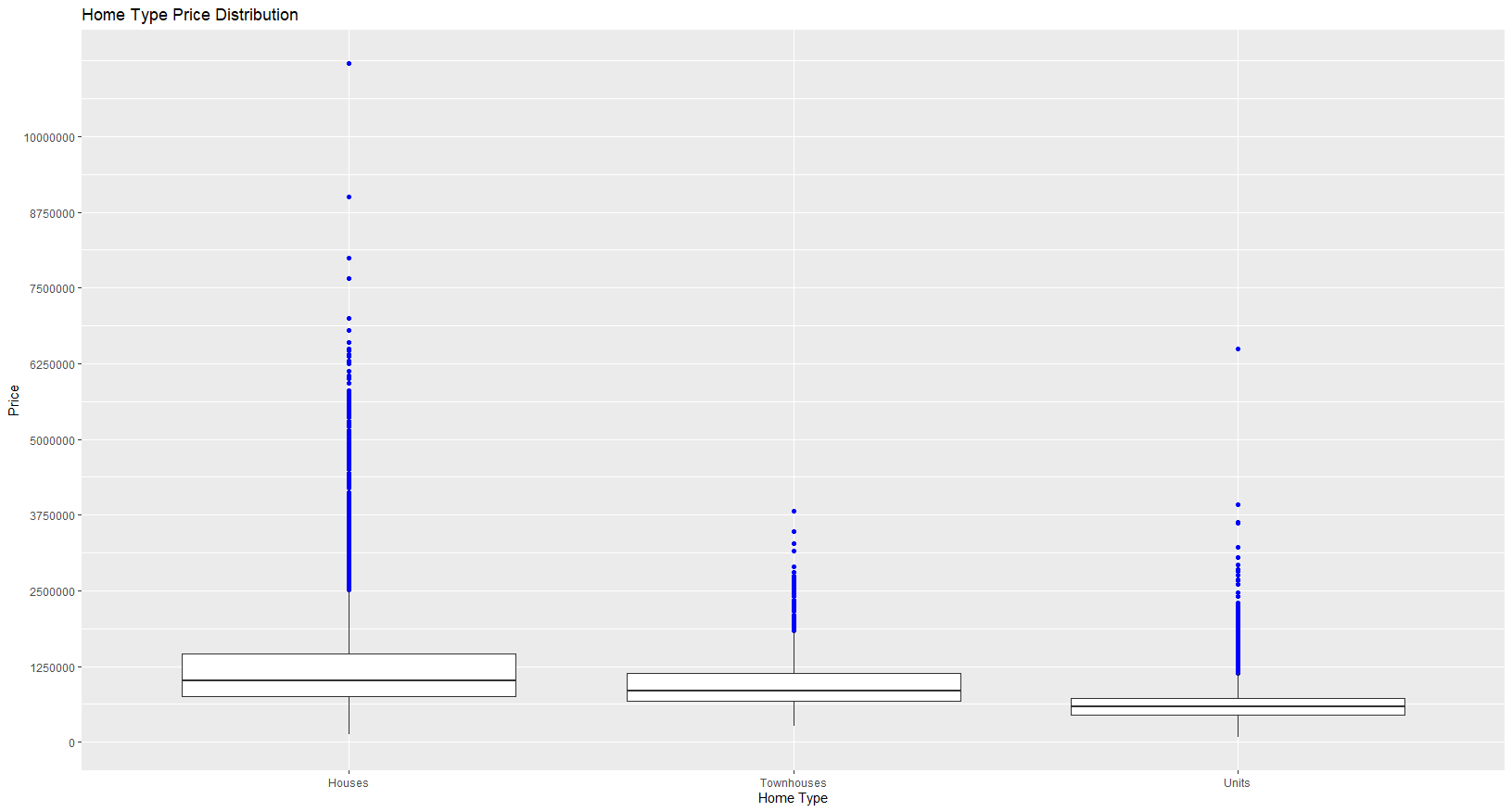
**4. Distance from Central Business District (CBD)**

****

****Melbourne homes are located on average 11 kilometers from Melbourne’s CBD. 15% of homes are within 0 to 5 kilometers from the CBD, 49% of homes are within 5.1 to 12 kilometers, 27% are within 12.1 to 20 kilometers, and 9% of home are at least 20.1 kilometers away from the CBD, up to a maximum of 48 kilometers.

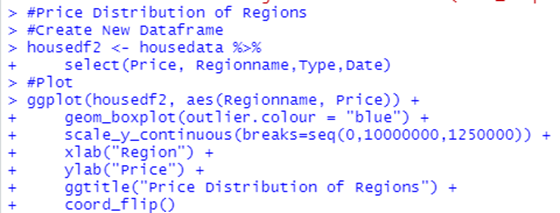
**5. Home Type Price Comparison**

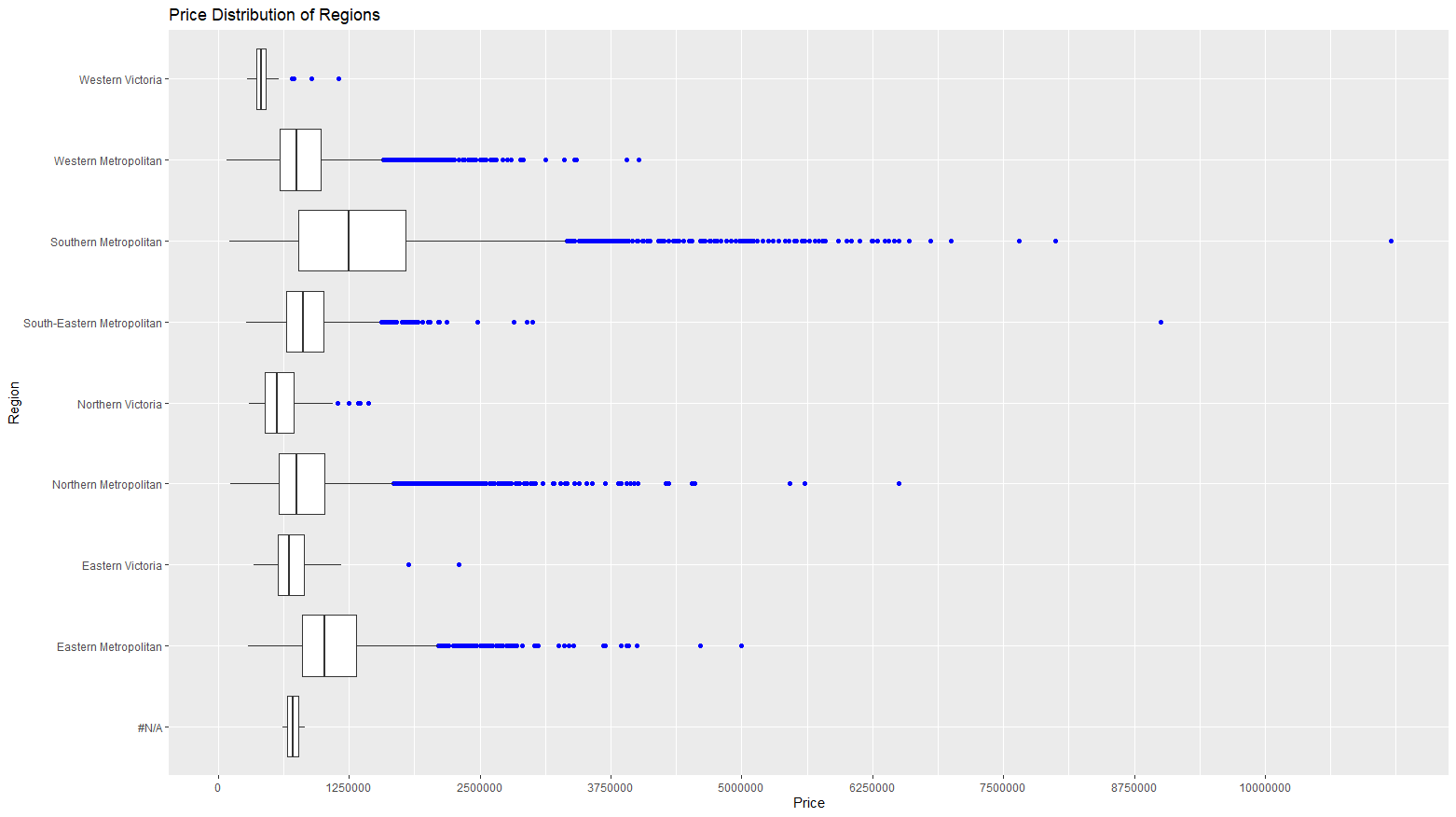
****

****

Based on the boxplot above, houses tend to have the highest mean price compared to townhouses and units. Units are the least expensive option of the three. This visualization also suggests that houses tend to have more price outliers. Some outliers even surpass ten times the average price of a house which appears to be approximately $1.2 million Australian dollars.

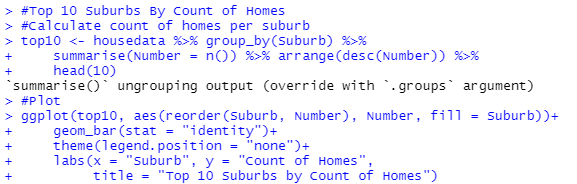
**6. Region Price Comparison**

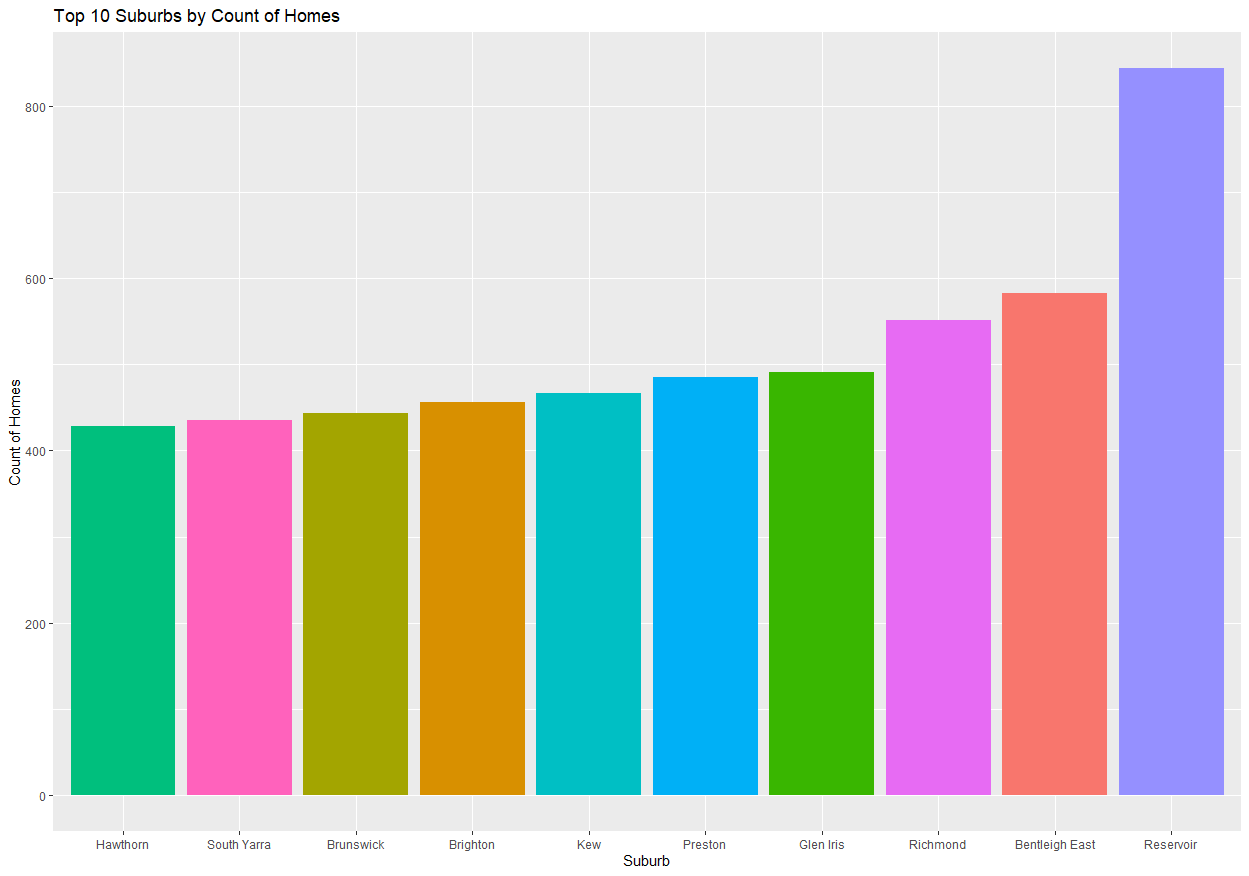
****

****

Based on the Melbourne region boxplots above, Western Victoria appears to have the lowest average price per home: $430,000. Southern Metropolitan appears to have the highest average price per home: $1,400,000. Western Metropolitan, Northern Metropolitan, and South-Eastern Metropolitan appear to have similar price averages with a combined average home price of $850,000.

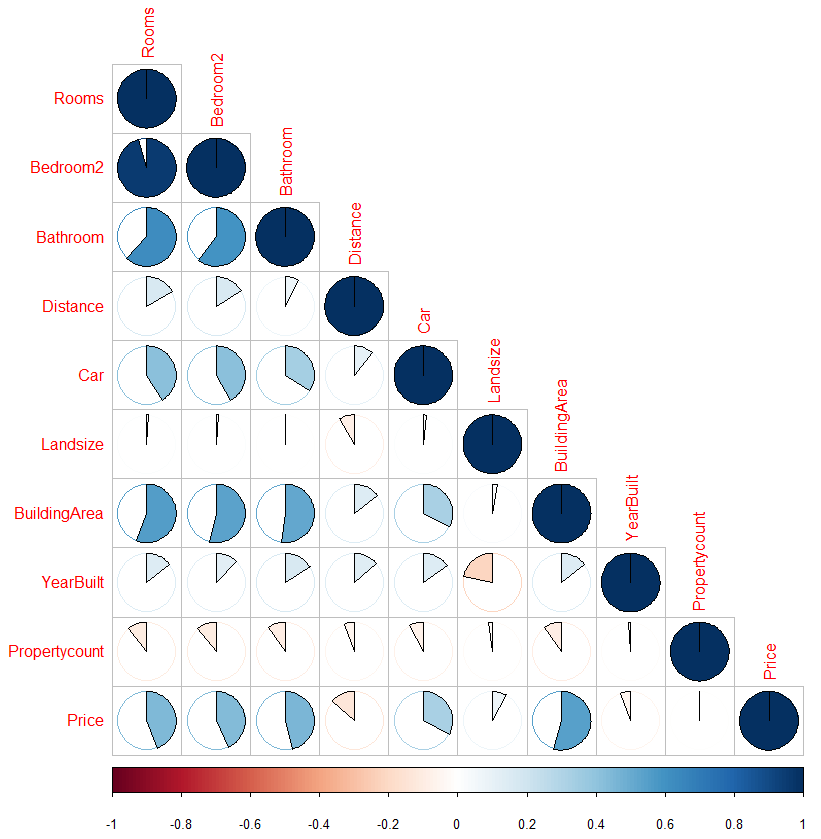
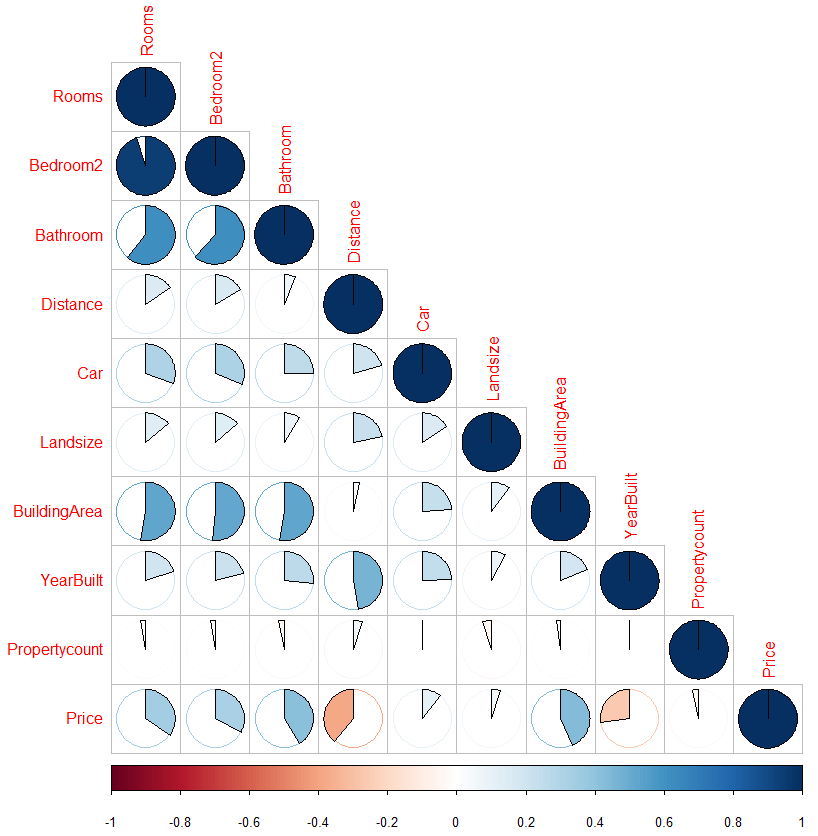
**7. Count of Homes Per Suburb**

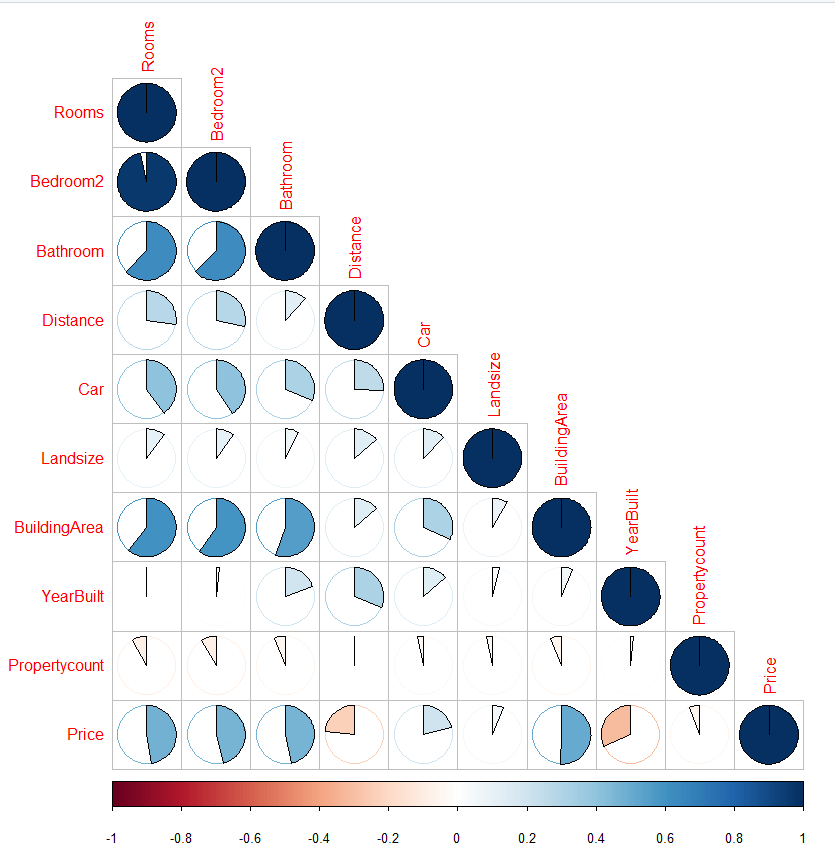
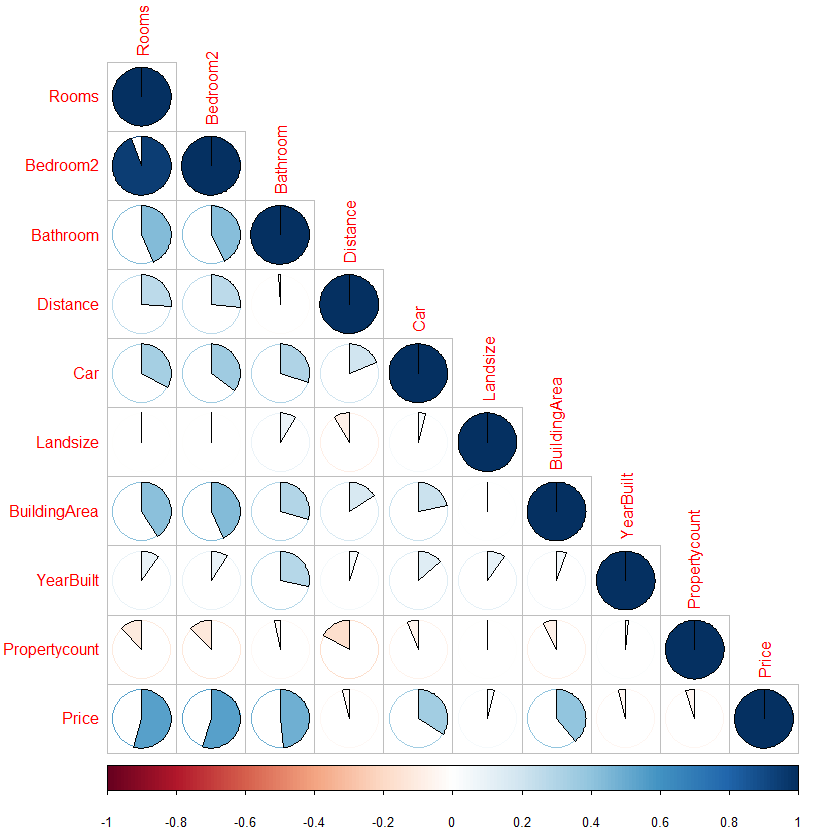
****

****

There are a total of 349 suburbs in this dataset. Based on the visualization above, suburb ‘Reservoir’ appears to have the most homes for sale, approximately 2.42% of all homes. Reservoir is located in the Northern Metropolitan region, which appears to have the closest average price per home to the combined average price of all homes in Melbourne.

**8. Price Correlation Matrix**

****

****

**\*Top Left Chart: Correlation Matrix of ‘House’ type homes**

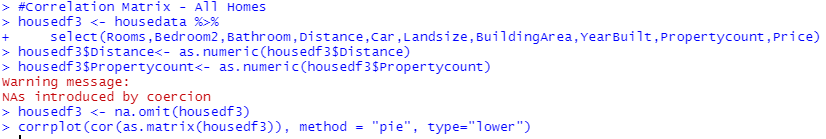
**\*Top Right Chart: Correlation Matrix of ‘Townhouse’ type homes**

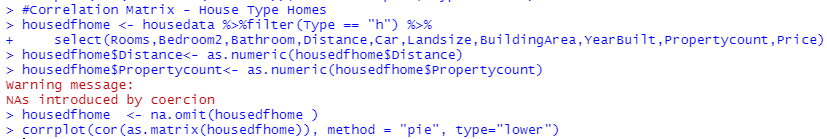
**\*Bottom Left Chart: Correlation Matrix of ‘Unit’ type homes**

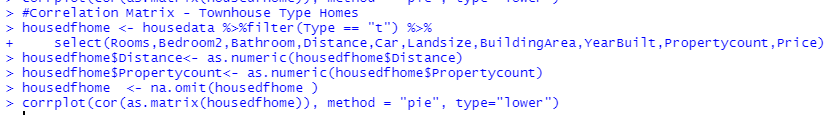
**\*Bottom Right Chart: Correlation Matrix of all type homes**

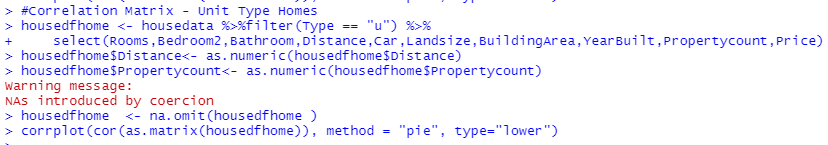
Based on the visualizations above, the price of a Melbourne home is positively correlated with the number of rooms, number of bedrooms, number of bathrooms, number of car lots, land size, and building area. The positive correlation of number of rooms, number of bedrooms, and number of bathrooms tends to be stronger for ‘unit’ type homes compared to ‘house’ and ‘townhouse’ type homes. This would make sense given that these are their main selling features as lot size is more constrained.

The price of a Melbourne home tends to be negatively correlated with the year it was built, and the distance from Melbourne’s central business district. This negative correlation tends to be stronger for ‘house’ type homes. This would make sense given that it costs more in transportation and heating to live farther away from the central business district. It also costs more to maintain an older home as there may be a higher occurrence of unexpected maintenance costs.









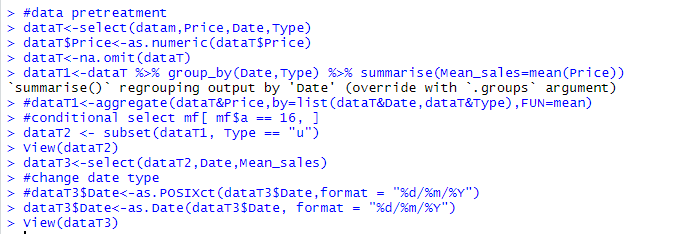
# 

# **4. Time Series**

## **4.1 Data Transformation**

In order to perform time series, the data need to be transformed into time series format. In the section, the average price per date per house type was selected for further analysis after the null value was removed from the data.

The codes for data transformation were presented below and the data for times series will be the average price per date for condo.



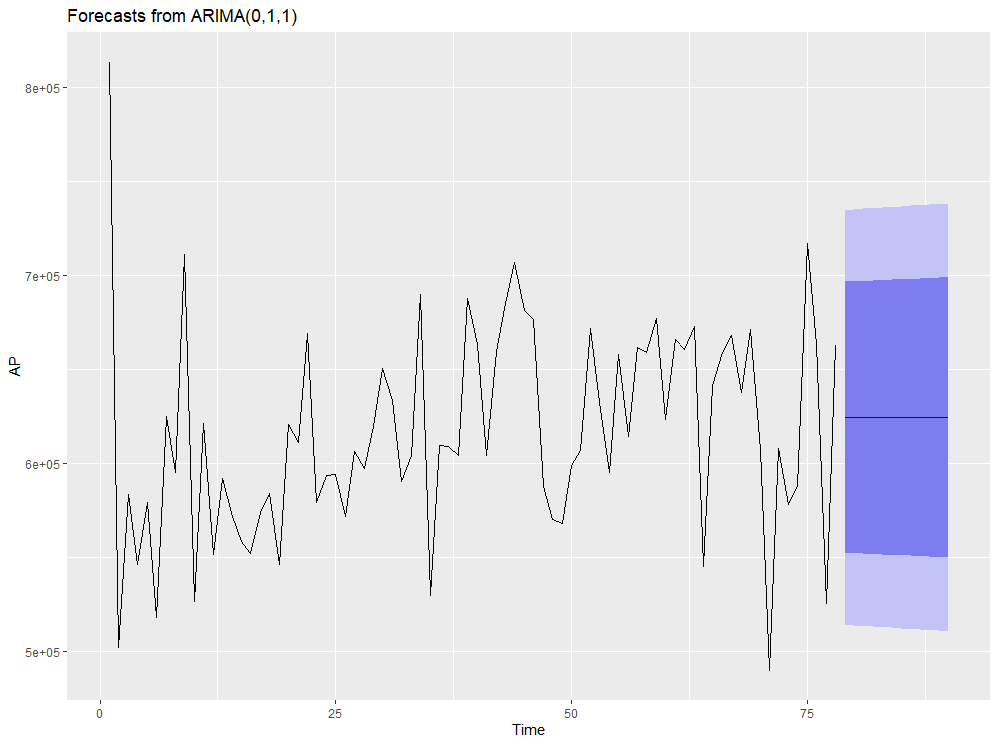
Then the data was changed to TS (time series format).



## **4.2 Time Series Modeling**

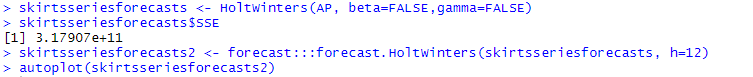
Next, the time series modeling will be divided into 3 parts. In the first part, a model called arima was used for forecasting. There are no seasonal factors here.

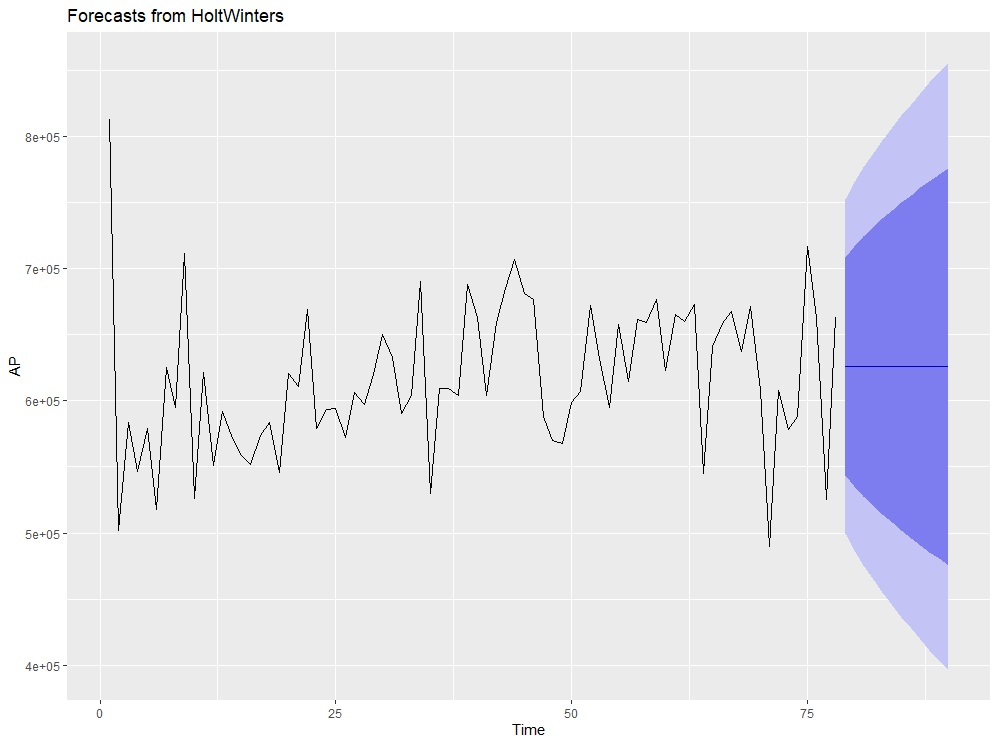




From this model, The next year sales seems to be level off while most price ranges fall between 0.6 to 0.7 millions.

Second, Exponential Smoothing modeling was carried out.

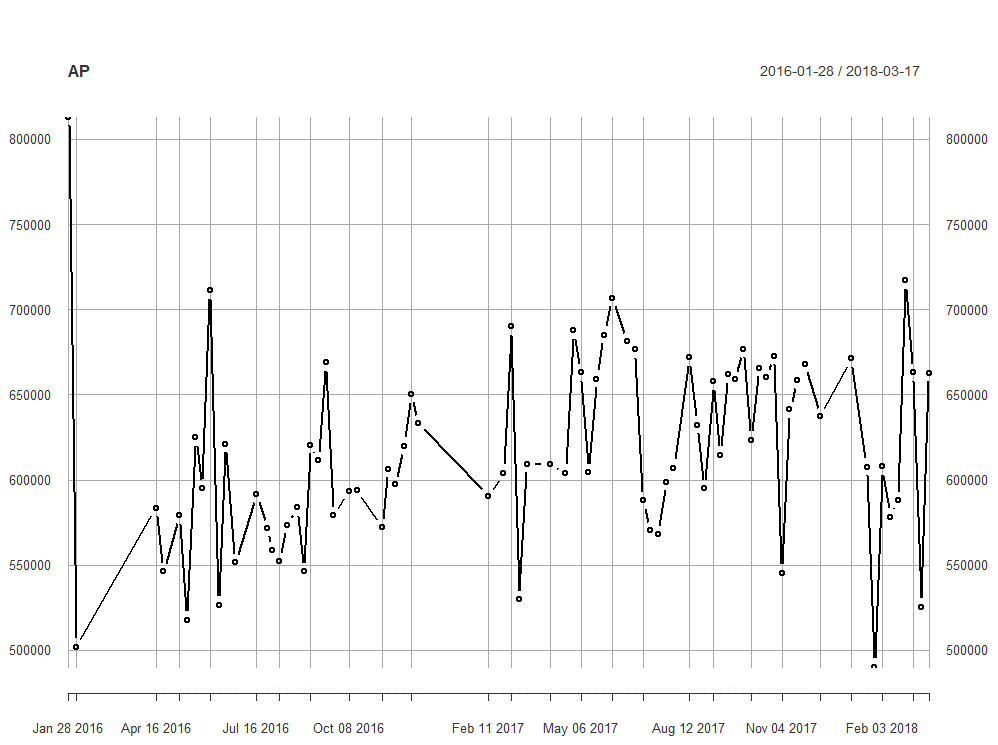




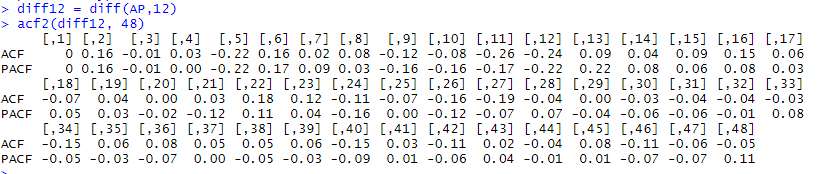
From this second model, the result turns out to be similar but with better MAE which will be explained later. In general, it leveled off again.

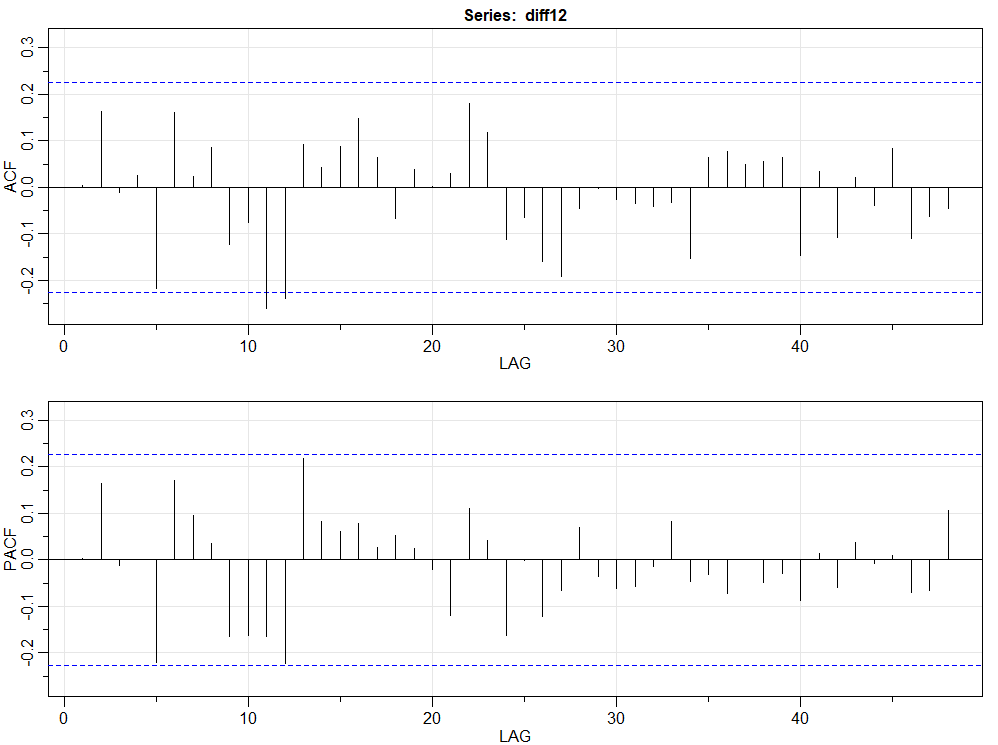
The third model was for seasonal analysis. This model was used as the data has some seasonal changes. To begin with some plots were used to see if any seasonal changes.





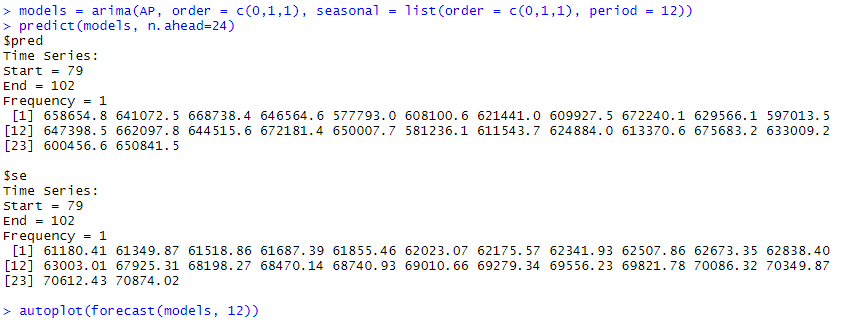
From the plot, further analysis was performed.

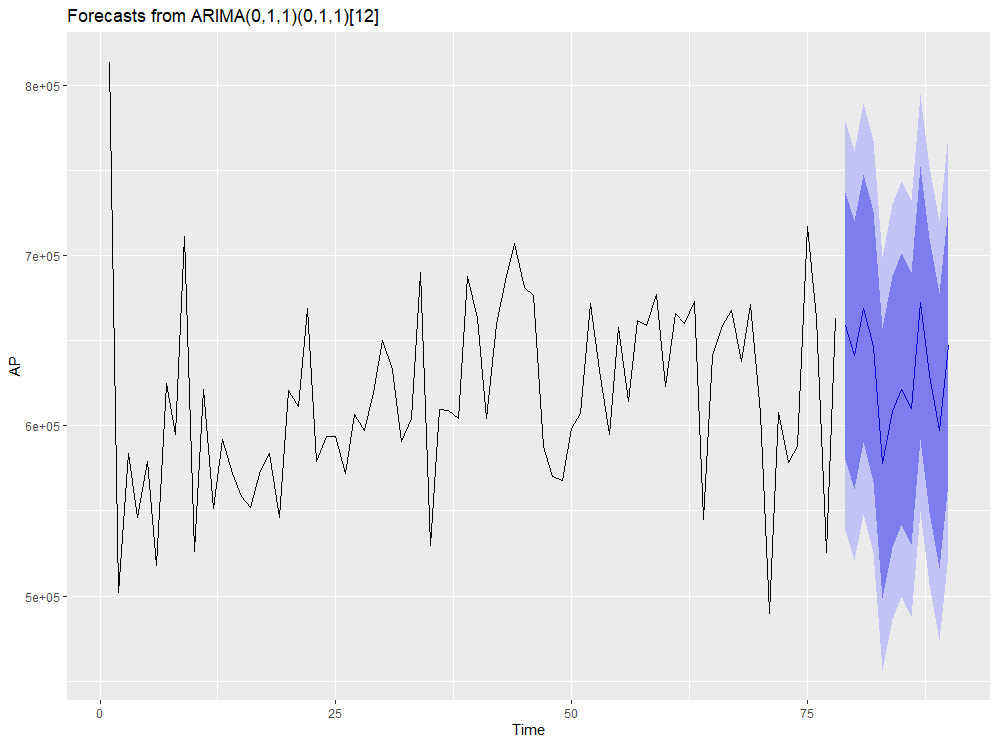




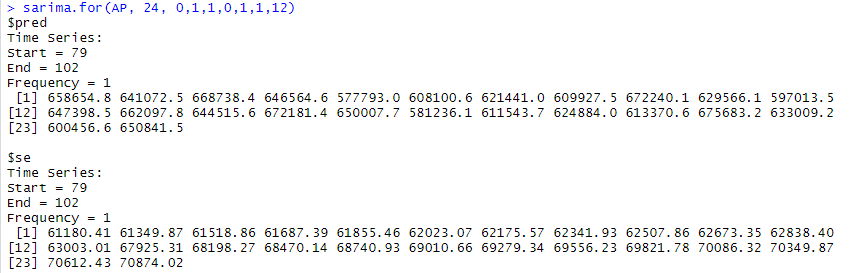
From the plot, some monthly change was captured but no obvious trend was observed. Based on this, ARIMA(1,0,0)×(0,1,1)12 was used to fit this model. 12 here in the model stand for 12 months. The parameter in the first () was the ARIMA model while the second () represent the seasonal mimicking those meanings for the first(). 12 stands for 12 seasonal changes in a year.

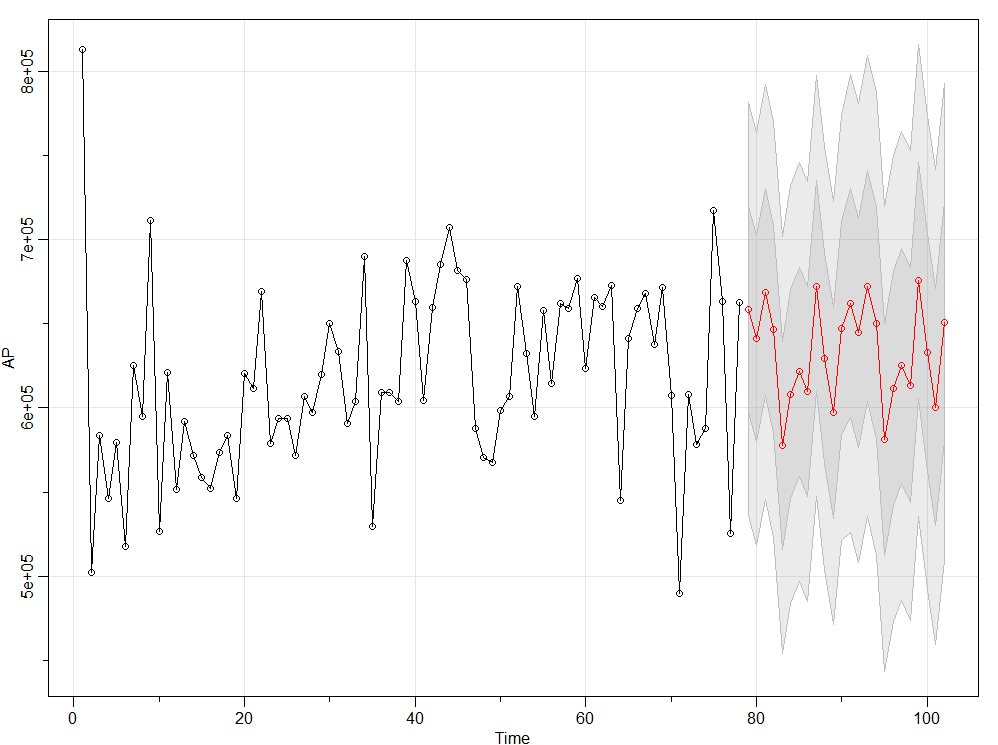
So this in other words, if R native language was used for prediction, the code will be like this.



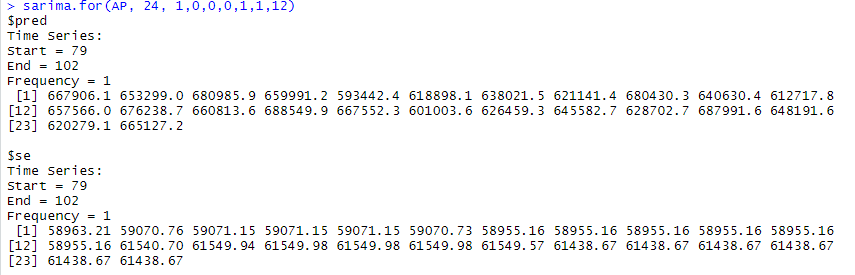


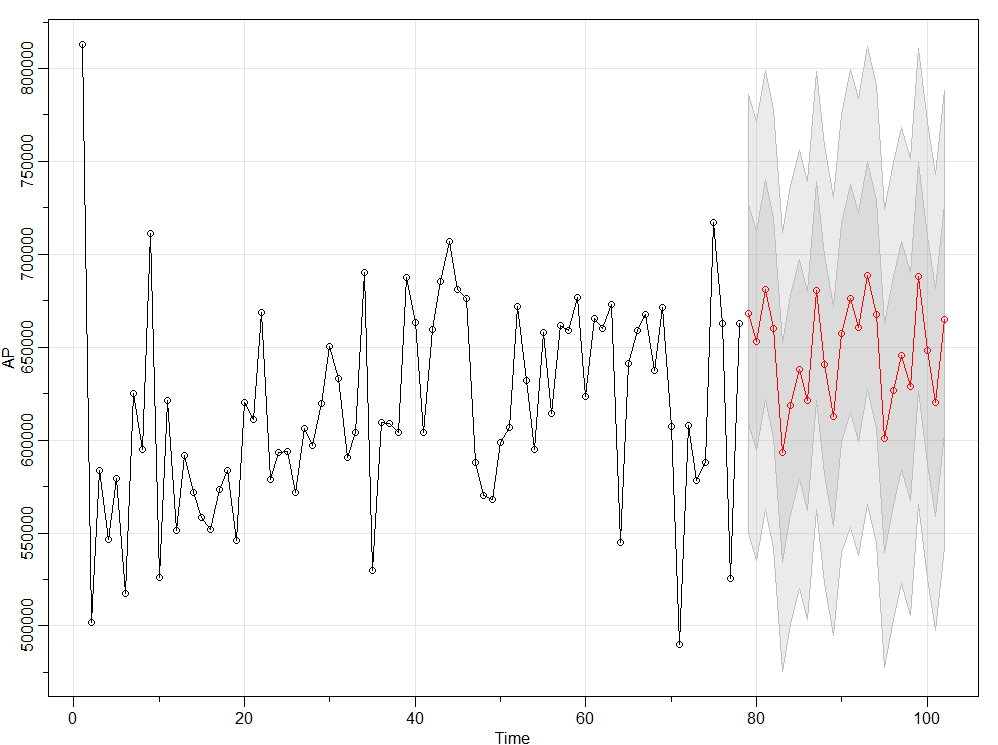
For seasonal arima model, sarima() could be used to plot as well with same meaning.











From the graph above, the seasonal modeling will be more fitted as it depicts the change and fluctuation as month changes during a year.

## **4.3 Time Series Modeling Evaluation**

The accuracy for the Arima model was presented as follows.



The accuracy for the Exponential Smoothing modeling was presented as follows.



The accuracy for the seasonal modeling was presented as follows.



Among all parameters, MAPE, which stands for Mean absolute percentage error, considers absolute difference between actual values and fitted values as a percentage of actual value and finally calculates the mean of that. In general, the smaller the MAPE, the better the model is. In this case, the Exponential Smoothing modeling was slightly worse compared to the Arima model regarding prediction. But the seasonal Arima model had the best MAPE at 6.67. Thus, a Seasonal Arima model was best suited for this analysis.

# **5. House Price Prediction Model - Deep Learning**

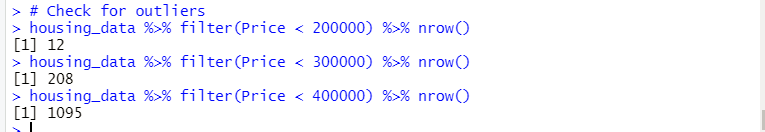
## **5.1 Data Preparation and Features Selection**

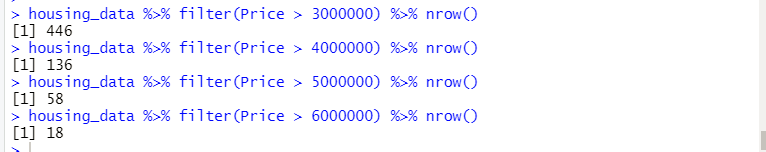
**Step-1 - Data Cleanup for Price attribute**:

First we check for observations with missing Price values and outliers. We found that the dataset has:

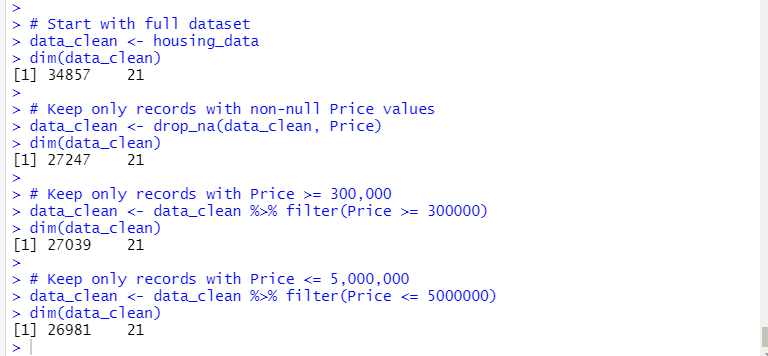
* 7,610 rows (approx. 21.8%) with null value.
* 208 rows (approx. 0.6%) with price values < 300,000.
* 58 rows (approx. 0.2%) with price values > 5,000,000.





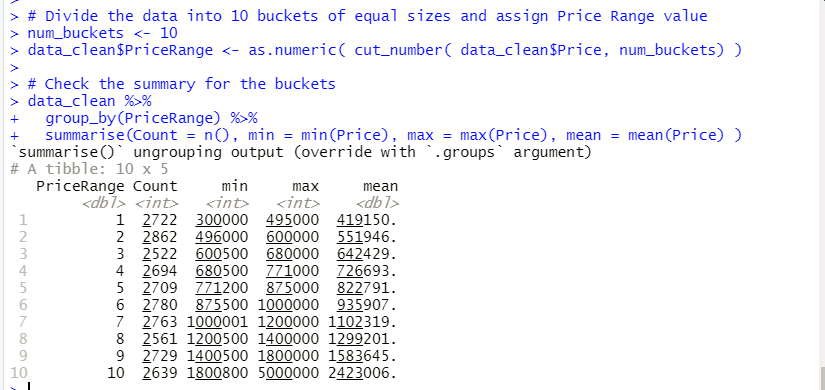


We drop the records with missing price values and those with price values less than 300,000 and more than 5,000,000. After that the dataframe has 26,981 rows (approx. 77.4%) left.



**Step-2 - Divide Data Buckets**:

Next we divide the data into 10 buckets of equal sizes and assign the bucket number for each group to a new variable (column) named **PriceRange**. Each bucket has about 10% of data (between 2,500 and 2,800 rows). The newly created column PriceRange has integer values from 1 to 10.



**Step-3 - Feature Selection**:

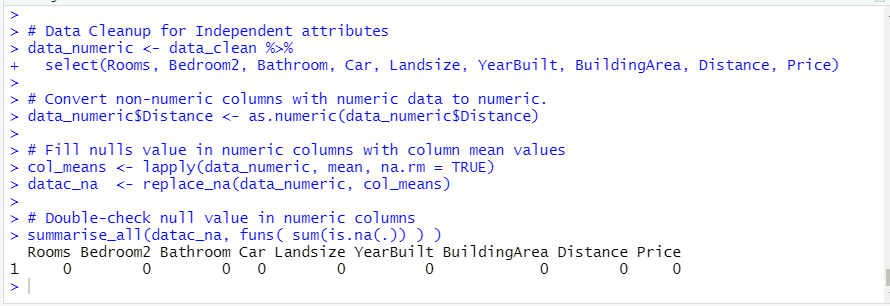
The following attributes were identified as having an impact on (are correlated to) the house price by our Data Analysis in section 3 above:

* Type - Property type - 3 different values: House, Townhouse, Unit.
* Rooms - Number of rooms
* Bedroom - Number of bedrooms
* Bathroom - Number of bathrooms
* Car - Number of car lots
* Landsize - Land size
* YearBuilt - Year property was built
* BuildingArea - Build area
* Distance - Distance from Central Business District (CBD)

**Step-4 - Data Cleanup for Independent attributes**:

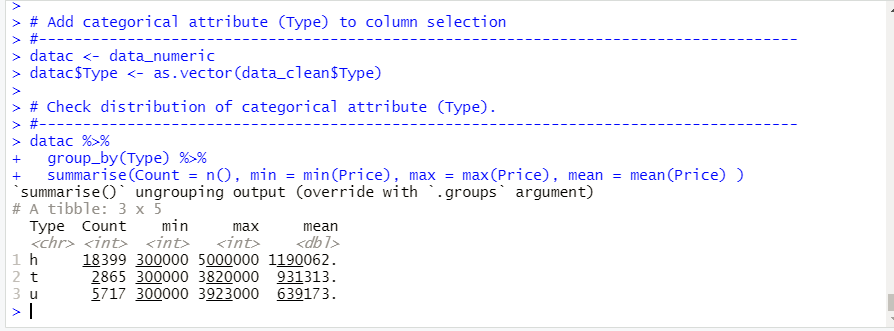
We perform data clean up on the attributes that were identified as having an impact on (are correlated to) the house price by our Data Analysis in section 3 above. We perform the following cleanup:

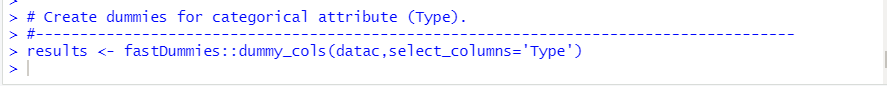
* Convert non-numeric columns with numeric data to numeric.
* Fill null values in numeric columns with column mean values.



**Step-5 - Create dummies for categorical attribute (Type)**:

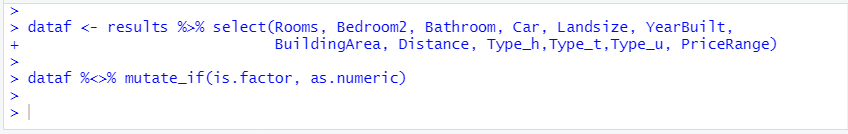
We add categorical attribute (Type) to column selection and create dummies.



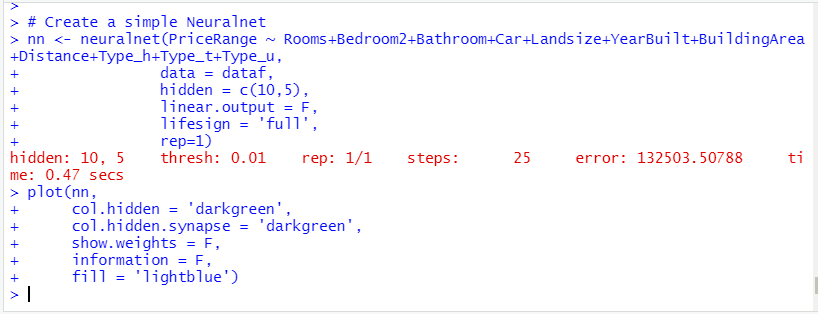


**Step-6 - Prepare Model Data**:

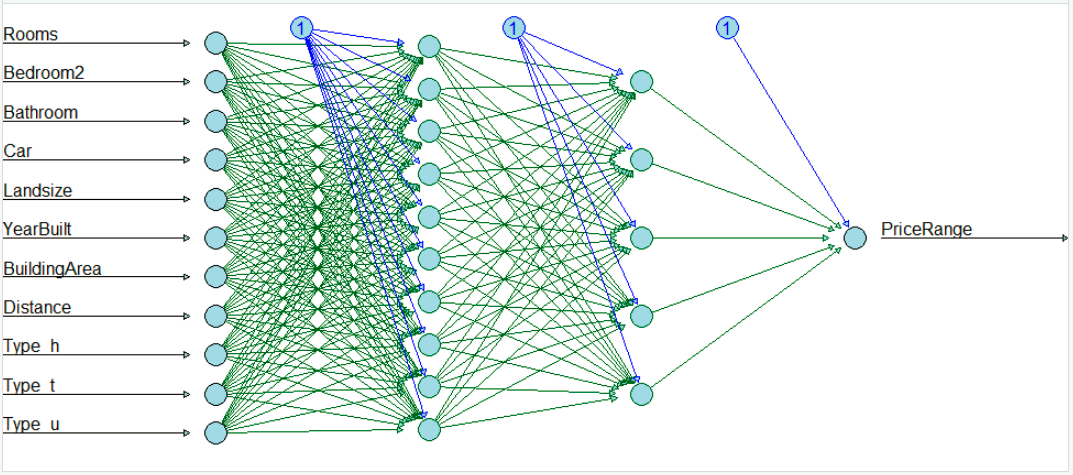
1. Create a dataframe with the selected features:



2. Create a simple NeuralNet with the selected features:



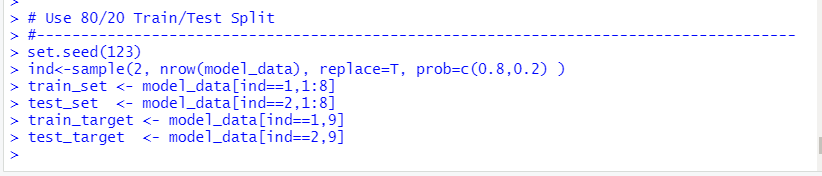
Below is the visualization of the NeuralNet created:



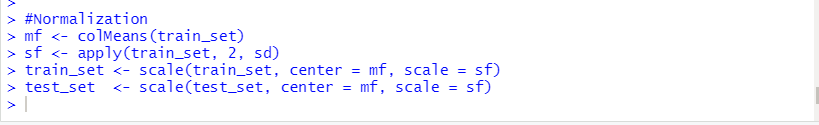
3. Convert model data to matrix:



4. Use 80/20 split for Train/Test sets:

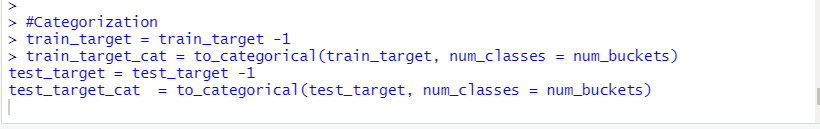


5. Normalize the train and test data:



6. Categorization the target vector.

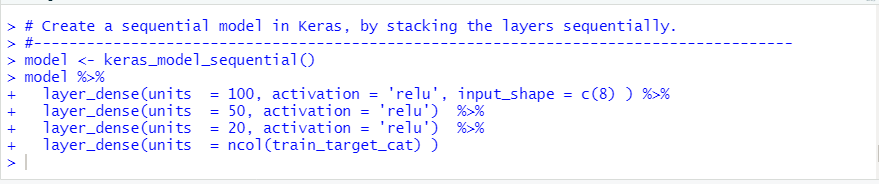
This is called one-hot encoding in Keras (Deep Learning), which means we will have multiple binary-classifications, instead of one classification with multiple possible outcome values.



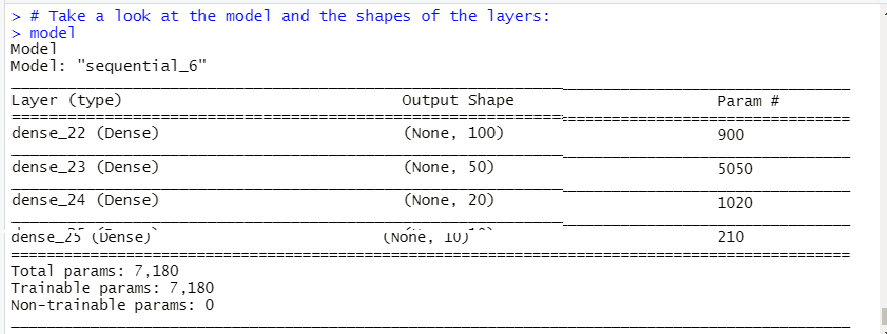
## **5.2 Keras Deep Learning Model**

Create a sequential model in Keras, by stacking the layers sequentially. Our model has

* Three hidden layers, with 100, 50 and 20 neurons.
* Keras builds an implicit input layer using the input\_shape parameter.
* Last layer is the output layer.

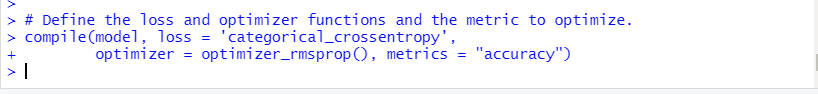


Below are more details on the model:



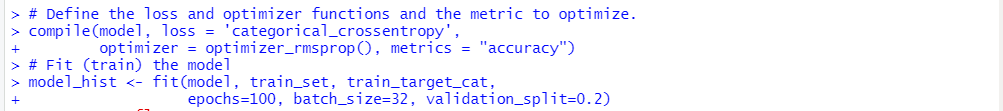
We use the following options with our model:

* Loss function = categorical\_crossentropy
* Optimizer function = optimizer\_rmsprop()
* Accuracy for our metrics

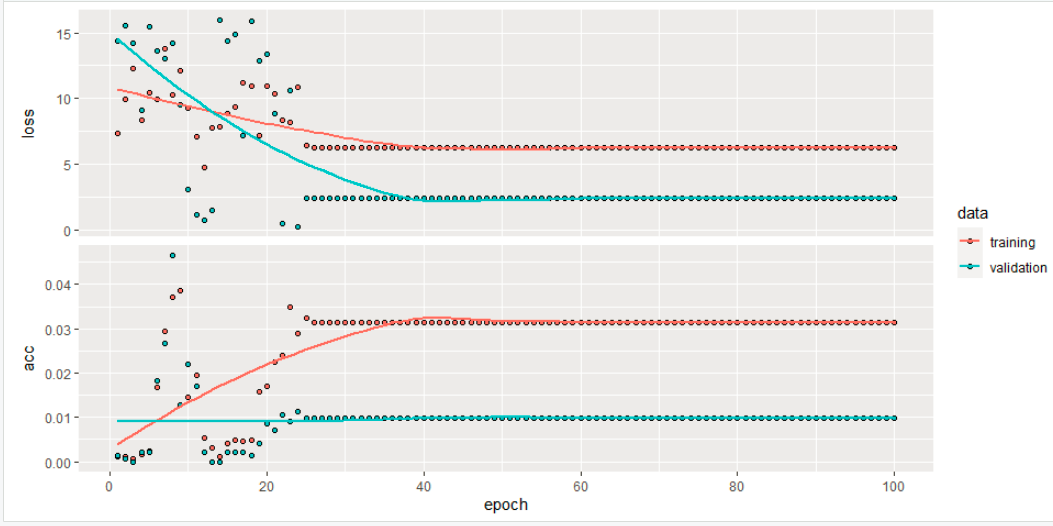
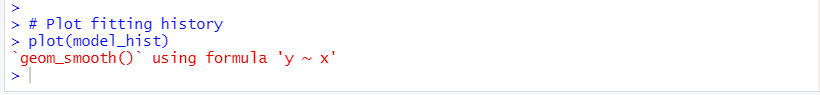


We fit the model with train dataset and with the following options:

* # epochs = 100
* Batch\_size = 32
* Validation\_split=0.2



Below is the chart of our model fitting history:



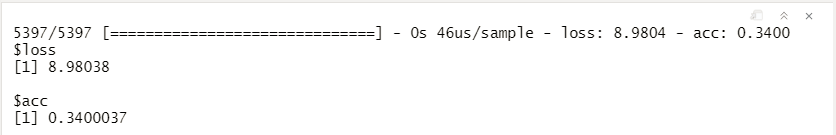
## **5.4 Model Evaluation**

We evaluate our using the testset data. The screenshot below shows that:

The loss rate is 8.98, which is fairly good.

However the accuracy is only about 34%, which is not very good.

We definitely need more time to fine tune the model to improve the performance and accuracy.



# **6. Deployment**

We used the ShinyApps website to deploy our application.

Our application on ShinyApps can be accessed at: <https://skvuong.shinyapps.io/project/>

Our application code can be found in GitHub at: <https://github.com/skvuong/housingMarketAnalysis>

**6.1 ShinyApps Design**

The **user interface** for the ShinyApp presents a drop-down list of options for the users to choose which analysis and visualization they would like to see. The **server code** code receives the user input and prepares the data and returns the information to the user interface. **The results** are then presented to users.

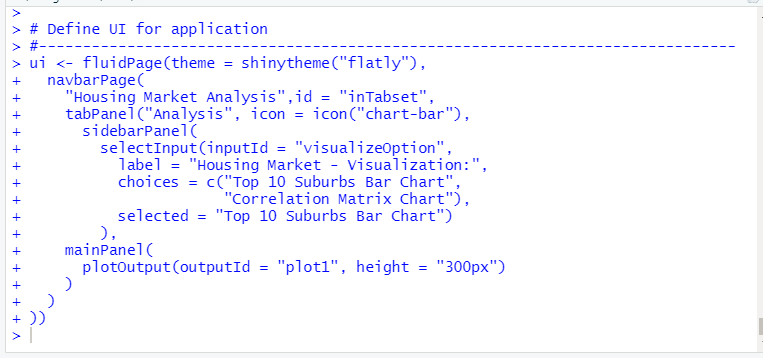
Our ShinyApp presents the following options to the users:

* Top 10 Suburbs Bar Chart
* Correlation Matrix Chart

**6.2 ShinyApps Code**

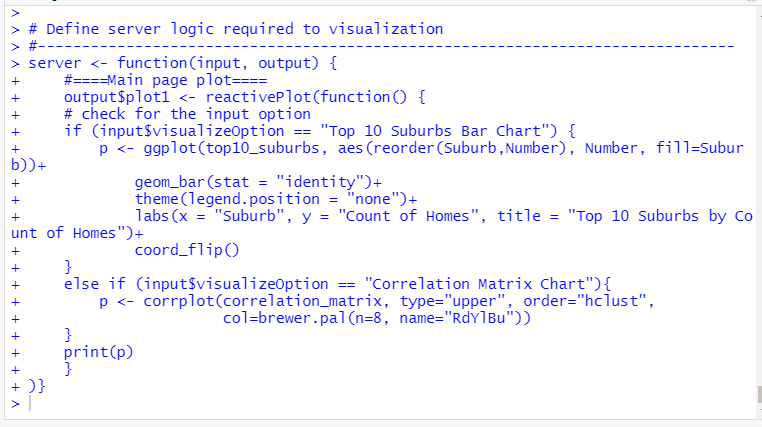
**1. User Interface Code:**

The code for the user interface is as follow:



**2. Server Code:**

The server code is as follows:



# 

# **7. Conclusion**

1. For the data exploration, the most favorable home was found to have 3 bedrooms, 2 bathrooms, 2 car spots and 11 km from the center. Western Victoria appears to have the lowest average price per home at $430,000 while Southern Metropolitan appears to have the highest average price per home.

2. Based on the correlation, the price of a Melbourne home is positively correlated with the number of rooms, number of bedrooms, number of bathrooms, number of car lots, land size, and building area but negative with the year built. The positive correlation of number of rooms, number of bedrooms, and number of bathrooms tends to be stronger for ‘unit’ type homes compared to ‘house’ and ‘townhouse’ type homes.

3. From time series analysis, the Seasonal Arima model was found to be best suited with 6.75 for MPAE which means the model has 93.25% accuracy for prediction. Further analysis by changing the parameters inside the bracket could be performed to further improvement of the model. In terms of business, the price for a condo in Melbourne will go up during the summer season while dropping down during year change.

4. The result from our Price Prediction model using Keras Deep Learning shows that the model can achieve the loss rate of 8.98, which is fairly good. However the model accuracy is only about 34%, which is not very good. We definitely need more time to fine tune the performance of our model to have a better accuracy rate. Further analysis could be carried out to increase either units or layers for the models. What’s more, the effect of region or district could also be introduced into the model to see if accuracy could be increased.

# **8.** **Reference / Citation**

Pino, T. (2016). “*Melbourne Housing Market*.” Melbourne, Victoria: Kaggle, 08 Jun. 2019, https://www.kaggle.com/anthonypino/melbourne-housing-market