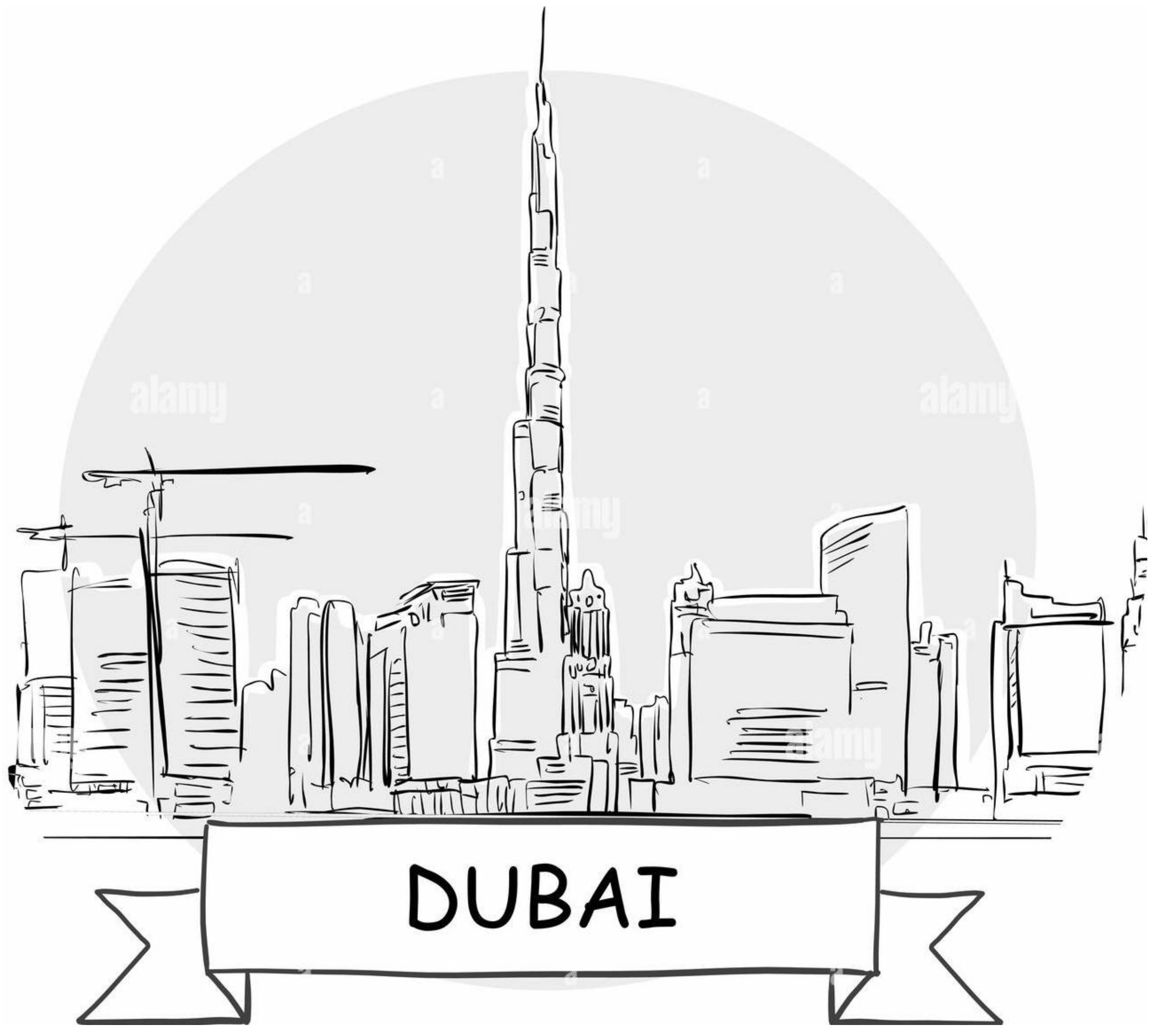
Building a House Price Prediction Model in the City of Dubai

PSTAT 131 - Machine Learning Final Project

Jana Hindiyeh

2022-12-12



Contents

1 Introduction

The aim of this project is to build a machine learning regression model that can predict the price of an apartment in the city of Dubai, United Arab Emirates. I will be using a dataset from kaggle, and testing it in multiple different ways so that it can yield the most accurate price prediction model.

1.1 Why is this model relevant?

Real estate is one of the most important assets that an individual can own, if not the most important. Whether as an investment or as a home or as a place of work, everybody in the world has a connection to, or an interest in real estate. Many different people are interested in many different aspects of this asset, but the most important is of course, its price. That is why I believe that this model is extremely useful and relevant, it could provide me and many others with deep and great insights into the trends behind the real estate market (in Dubai) on a micro level. This model will focus precisely on the sales price of apartments in Dubai based on a host of 37 different predictor variables. This us because this is the type of property that I find most interesting to me and most relevant to younger individuals (since I probably won't be buying a house for a while).

Having lived in Dubai all my life, and worked at a real-tech firm in Dubai, I saw the process of appraising real estate values and thought that it was slightly outdated. A lot of the work was done manually and led to long, tedious tasks that could be simplified using a model like the one I intend to build in this project. I also find the real estate market of Dubai very interesting, since it tends to be a place with a high turnover of residents (especially in apartments), I thought it would be interesting to find out the trends that drive the market.

1.2 Dubai Real Estate Market

There are a few notes I think are helpful to understand about the Dubai Real Estate market. 'New Dubai' or the newer parts of Dubai tend to have a high number of apartments, and is where the majority of younger individuals live. Dubai has a population of 3.5 million people as of April 2022, 3.2 million of those being non-Emaratis ('Emarati' is the term used to define the local population of the United Arab Emirates) or expats. Additionally, around 58.50% of the population is concentrated in the 25-44 age group. What is important to note here is that there is a relatively high turnover of residents that live in Dubai (similar to that of New York City, for example), people that come to Dubai for work and then leave after a few years. Therefore, there is a separation of the demographics that live in apartments and those that live in villas or townhouses. This is why below you will see the map of the properties that are included in this dataset are largely localised in a few areas of the city. These are the areas that surround the main commercial/ work spaces in Dubai, and the rest of the areas are largely populated by houses/ villas/ and townhouses.

1.3 Project Roadmap

Now that a beter understanding of the project and the Dubai real estate market has been established, I want to run through the process in which I will choose the most effective model to predict prices in Dubai.

- Load the necessary packages and the dataset into R.
- Explore the characteristics of, the variables in, and the validity of the data.
- Clean the data and prepare in for use in the model (mutating it accordingly).
- Explore the data, looking at relationships between characters, possible correlations, outliers, etc.
- Model building process. This process will be delved into deeper at the onset of that chapter however, the models that I will be testing are:
  - Ridge Regression

  - Lasso Regression
  - Decision Tree
  - Boosted Tree Model
  - SVM Model
- Analysis stage: Analyze models and select best one.
- Test model on Testing set
- Conclusion

1.3.0.1 Loading Packages

Firstly, I need to load all the R packages that I need throughout this process.

1.4 Data Introduction

1.4.1 Data source

I found the data on 'kaggle', a website that has a collection of data sets and all kinds of coding projects accross different coding languages. The link to the kaggle website is attached here https://www.kaggle.com/datasets/dataregress/dubai-properties-dataset. The code was sourced in December 2020, therefore, I recognize the limitation of the data regarding the fact that it is not up to date. As a result, its credibility in estimating the property prices as of 2022 may differ slightly.

Firstly, lets read in the data. I downloaded the data as a csv file onto my computer, which is part of the project folder uploaded onto github for your viewing.

```r
house <- read.csv("~/Desktop/PSTAT 131 Final/properties_data.csv")

head(house)
```

Let's look at the dimensions of the data

```r
dim(house)
```

```
## [1] 1900   38
```

The dataset contains **1905** properties and was sourced from a web scrape of the real estate portal. The data contains **38 columns** that consist of characteristics of the properies. Within the data there are a mix of boolean (TRUE/FALSE) variables, integer variables, and character variables, all of which will be used to analyze our data.

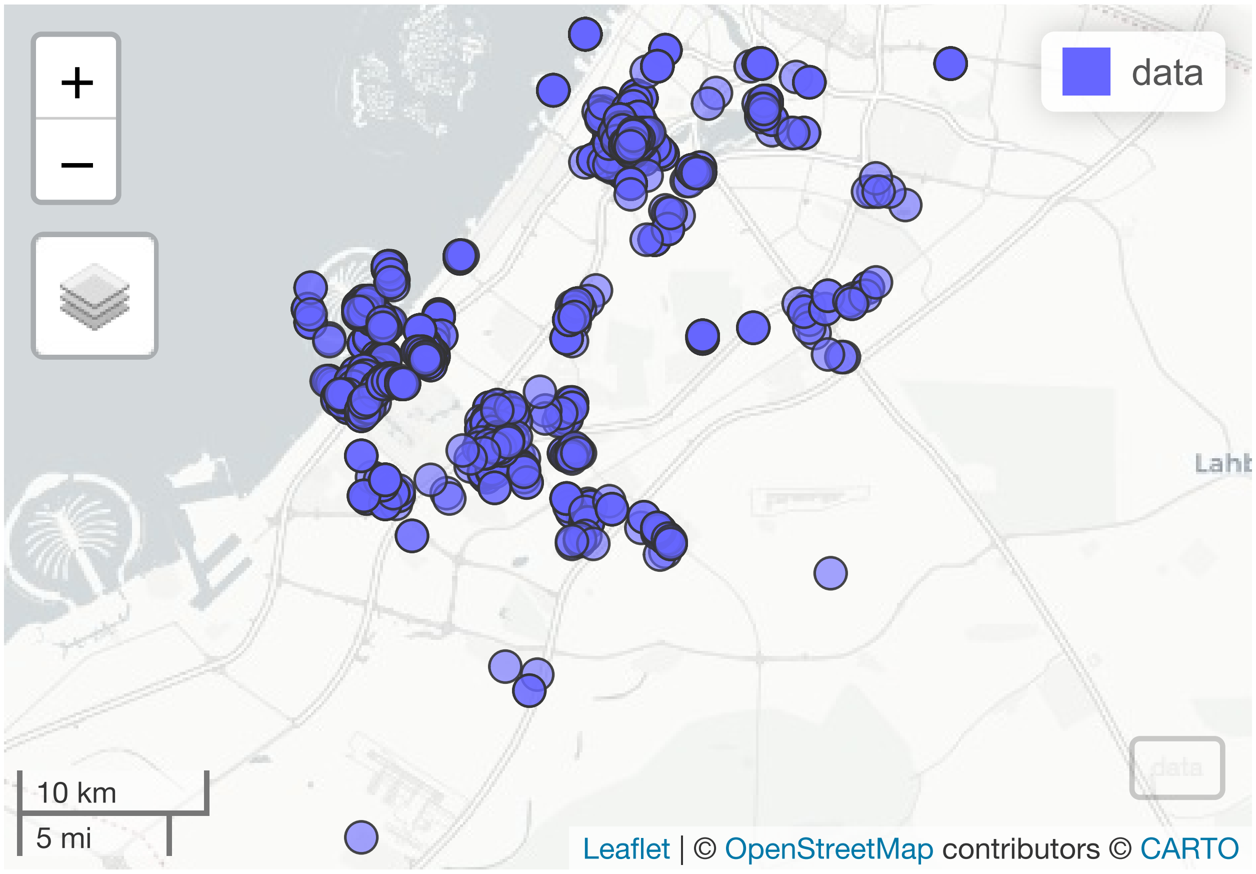
1.4.2 Variables of the data set:

- *id:* Numerical ID for each property.
- *neighborhood:* Neighborhood the apartment is located within.
- *latitude:* Latitude coordinates for the apartment's location
- *longitude*: Longitude coordinates for the apartment's location
- *price:* Property price as of December 2021
- *size_in_sqft:* The size of each apartment in sqft
- *price_per_sq:* Price/ size_in_sqft
- *no_of_bedrooms:* Number of bathrooms in the property
- *no_of_bathrooms*: Number of bedrooms in the property
- *quality:* The relative quality of the apartment (consists of three variables: Low, Medium, High)

- The following are *True or False* observations (e.g., if the property has a maid's room or not)
  - maid_room
  - unfurnished
  - balcony
  - barbecue_area
  - built_in_wardrobe
  - central_ac
  - childrens_play_area
  - childrens_pool
  - ...

Note: a full copy of the codebook is available in my zipped files.

Lets also explore the neighborhoods that the data covers bvy reate a map of the data points



2 Data Cleaning

While the data set that was downloaded was already relatively tidy, a few different cleaning steps are helful before moving on to the next step of the project:

1. Clean the data using clean_names

```
house <- clean_names(house)
```

Clean_names just returns variable names that are all consistently lowercase and contain no spaces. The data already had tidy names, however I thought it could not harm to clean the data a little extra.

2. Check for missing values

```
anyNA(house)
```

```
## [1] FALSE
```

This shows us that there are no missing values in the dataset which is super helpful and reduces the number of steps we need to take to prepare the data for the models.

3. Turn all True/False Statements into factor variables:

```
house <- house %>%
  mutate(maid_room = factor(maid_room),
         unfurnished = factor(unfurnished),
         balcony = factor(balcony),
         barbecue_area = factor(barbecue_area),
         built_in_wardrobes = factor(built_in_wardrobes),
         central_ac = factor(central_ac),
         childrens_play_area = factor(childrens_play_area),
         childrens_pool = factor(childrens_pool),
         concierge = factor(concierge),
         covered_parking = factor(covered_parking),
         kitchen_appliances = factor(kitchen_appliances),
         lobby_in_building = factor(lobby_in_building),
         maid_service = factor(maid_service),
         networked = factor(networked),
         pets_allowed = factor(pets_allowed),
         private_garden = factor(private_garden),
         private_gym = factor(private_gym),
         private_jacuzzi = factor(private_jacuzzi),
         private_pool = factor(private_pool),
         security = factor(security),
         shared_gym = factor(shared_gym),
         shared_pool = factor(shared_pool),
         shared_spa = factor(shared_spa),
         study = factor(study),
         vastu_compliant = factor(vastu_compliant),
         view_of_landmark = factor(view_of_landmark),
         view_of_water = factor(view_of_water),
         walk_in_closet = factor(walk_in_closet))
```

After transforming all the boolean variables into factor variables, I also decided to mutate the character variables 'neighborhood' and 'quality' into factor variables too.

```r
house <- house %>%
  mutate(neighborhood = factor(neighborhood),
         quality = factor(quality, levels = c("Low", "Medium", "High", "Ultra")))
```

That ends the introduction section, next I am going to delve into the exploratory data analysis section.

Finally I have also decided to remove the id variable from the data since I don't believe it adds anything to the analysis, and has no real value to the analysis.

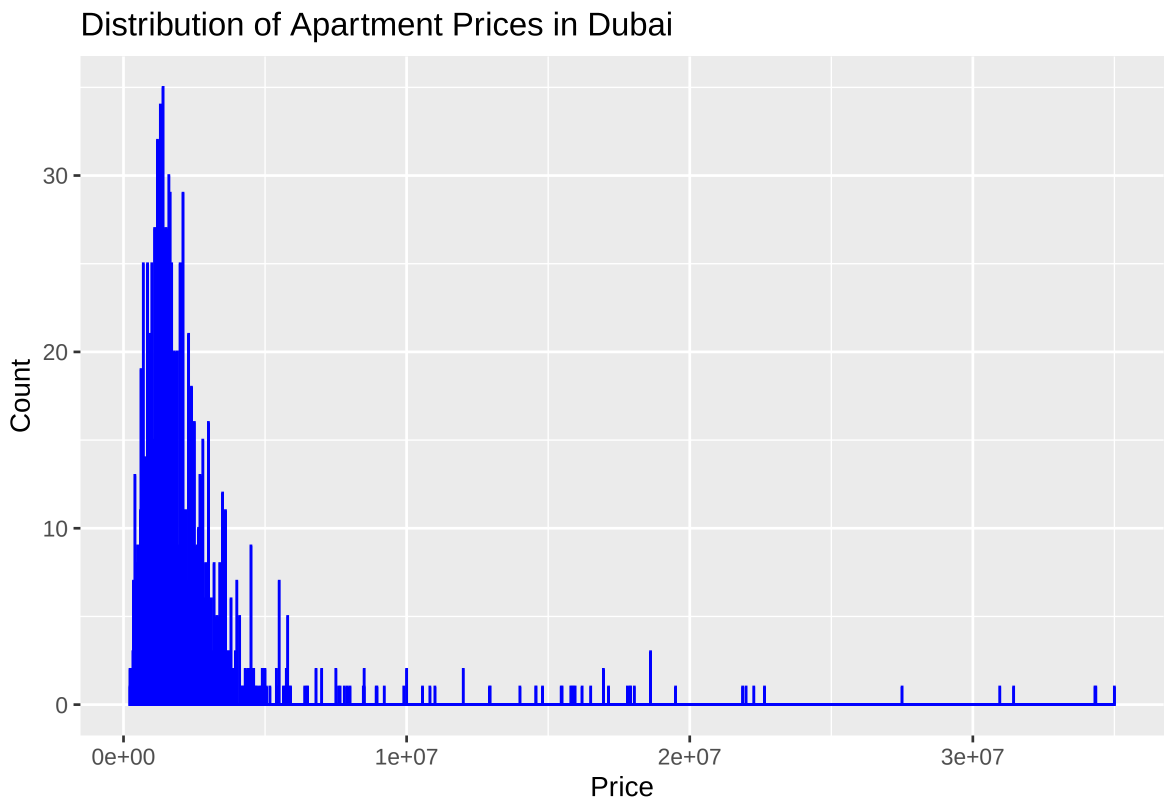
```r
house <- house %>% select(-id)
```

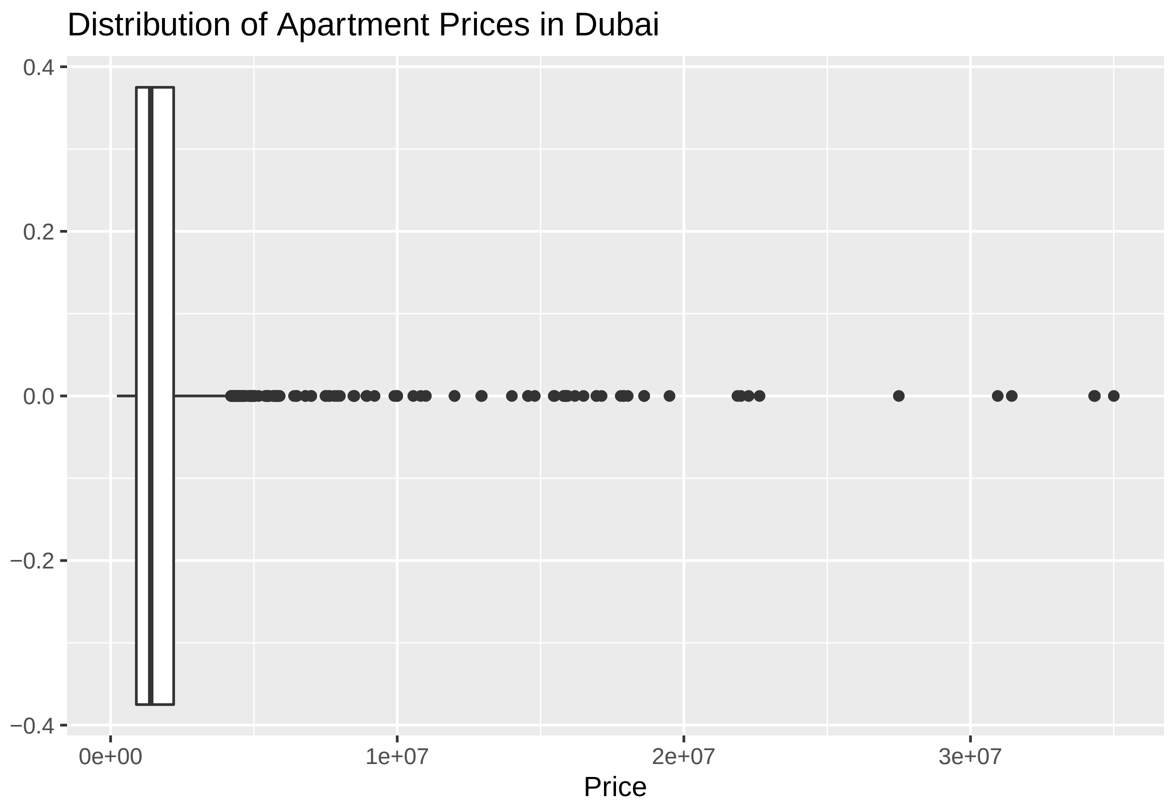
That ends the introduction section, next I am going to delve into the exploratory data analysis section.

3 Exploratory Data Analysis

The exploratory data analysis section will be based on the whole dataset, since I will be splitting the data set in the next section, when I am looking to start building my models. This is because, I want visibility on the trends of the whole dataset in order to make informed decisions about which models to use.

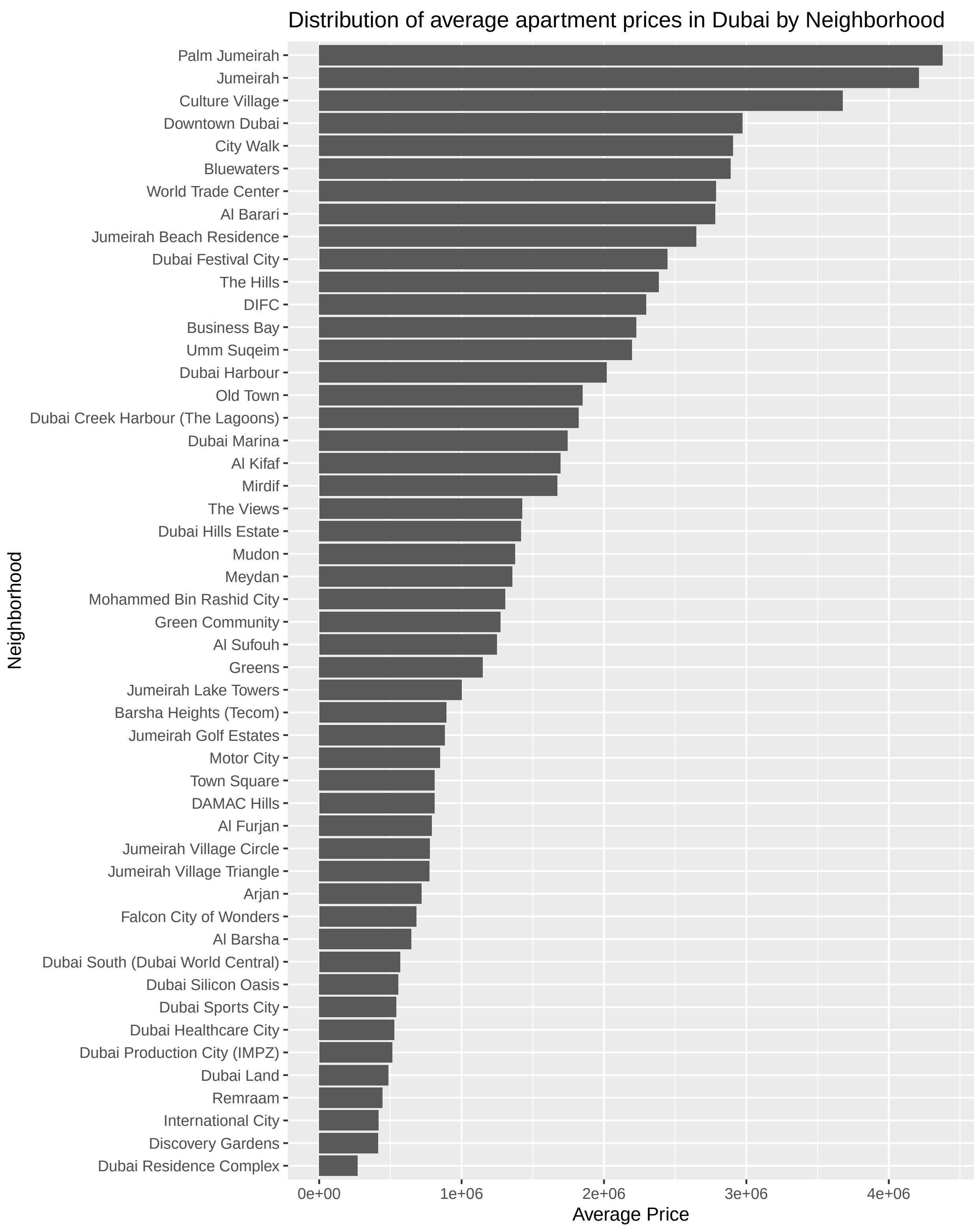
1. Price distribution Create a histogram and a boxplot of the prices so that we can get a better idea regarding the distribution of prices in Dubai

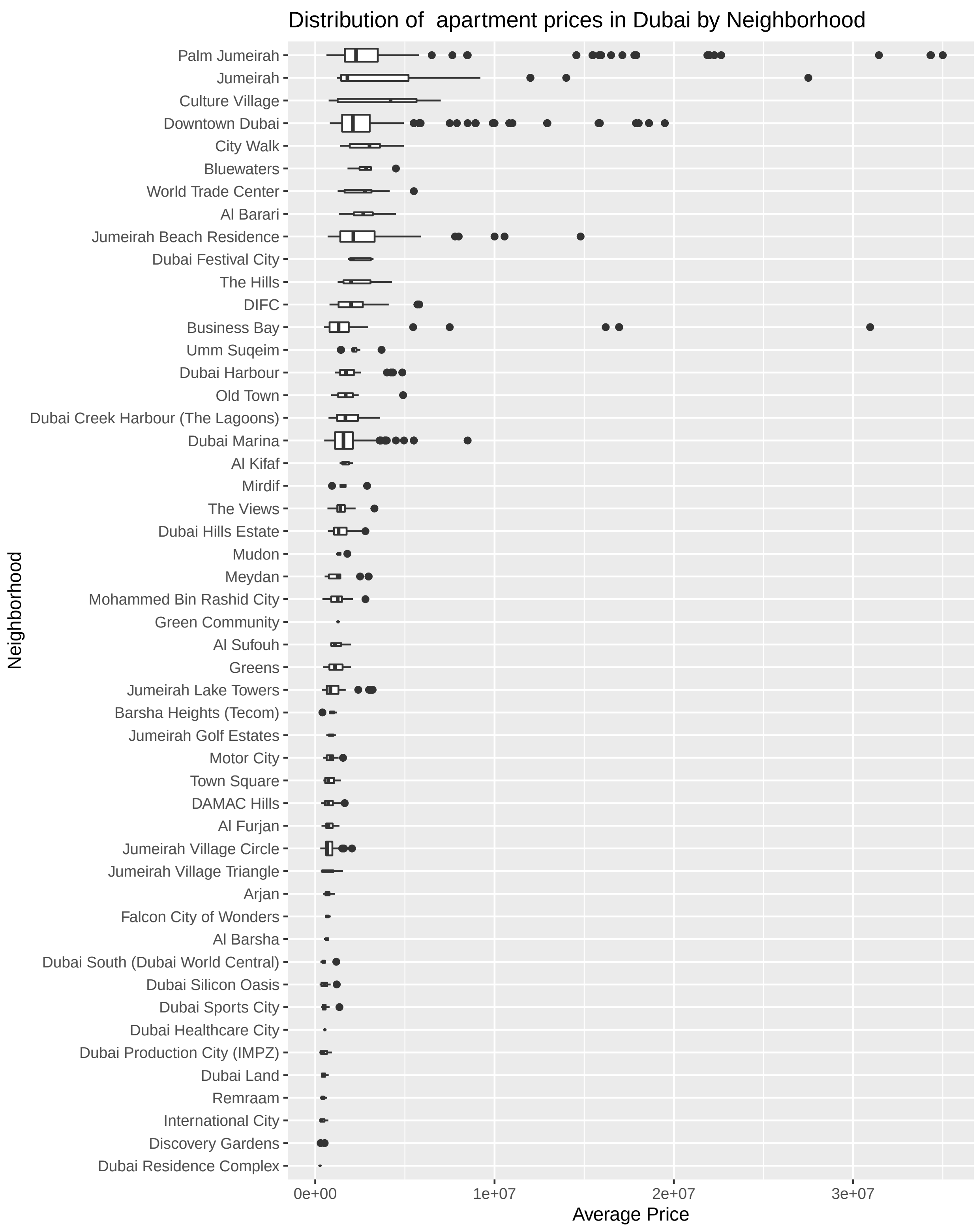




This plot shows us that the prices of apartments in Dubai somewhat follow a normal distribution curve, with a number of outliers (signifying extremely expensive luxury properties). This is to be expected in a city like Dubai that is known for its opulence and wealth. However, I don't believe that these properties are accurately representative of the whole real estate market and thus will be dealt with later in this section.

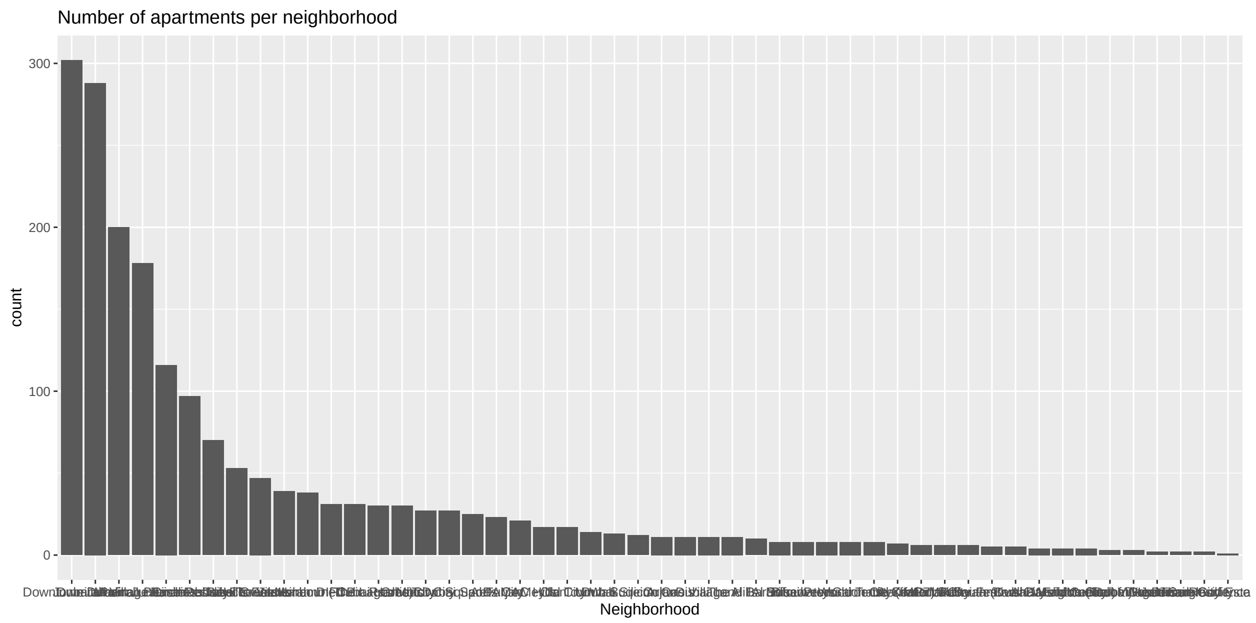
Next, I would also like to Look at the average price per neighborhood. Knowing Dubai, I hypothesize that the most expensive areas would be Palm Jumeirah, Jumeirah, and Downtown Dubai


Distribution of average apartment prices in Dubai by Neighborhood
Palm Jumeirah
Jumeirah
Culture Village
Downtown Dubai
City Walk
Bluewaters
World Trade Center
Al Barari
Jumeirah Beach Residence
Dubai Festival City
The Hills
DIFC
Business Bay
Umm Suqeim
Dubai Harbour
Old Town
Dubai Creek Harbour (The Lagoons)
Dubai Marina
Al Kifaf
Mirdif
The Views
Dubai Hills Estate
Mudon
Meydan
Mohammed Bin Rashid City
Green Community
Al Sufouh
Greens
Jumeirah Lake Towers
Barsha Heights (Tecom)
Jumeirah Golf Estates
Motor City
Town Square
DAMAC Hills
Al Furjan
Jumeirah Village Circle
Jumeirah Village Triangle
Arjan
Falcon City of Wonders
Al Barsha
Dubai South (Dubai World Central)
Dubai Silicon Oasis
Dubai Sports City
Dubai Healthcare City
Dubai Production City (IMPZ)
Dubai Land
Remraam
International City
Discovery Gardens
Dubai Residence Complex
Neighborhood
0e+00
1e+06
2e+06
3e+06
4e+06
Average Price

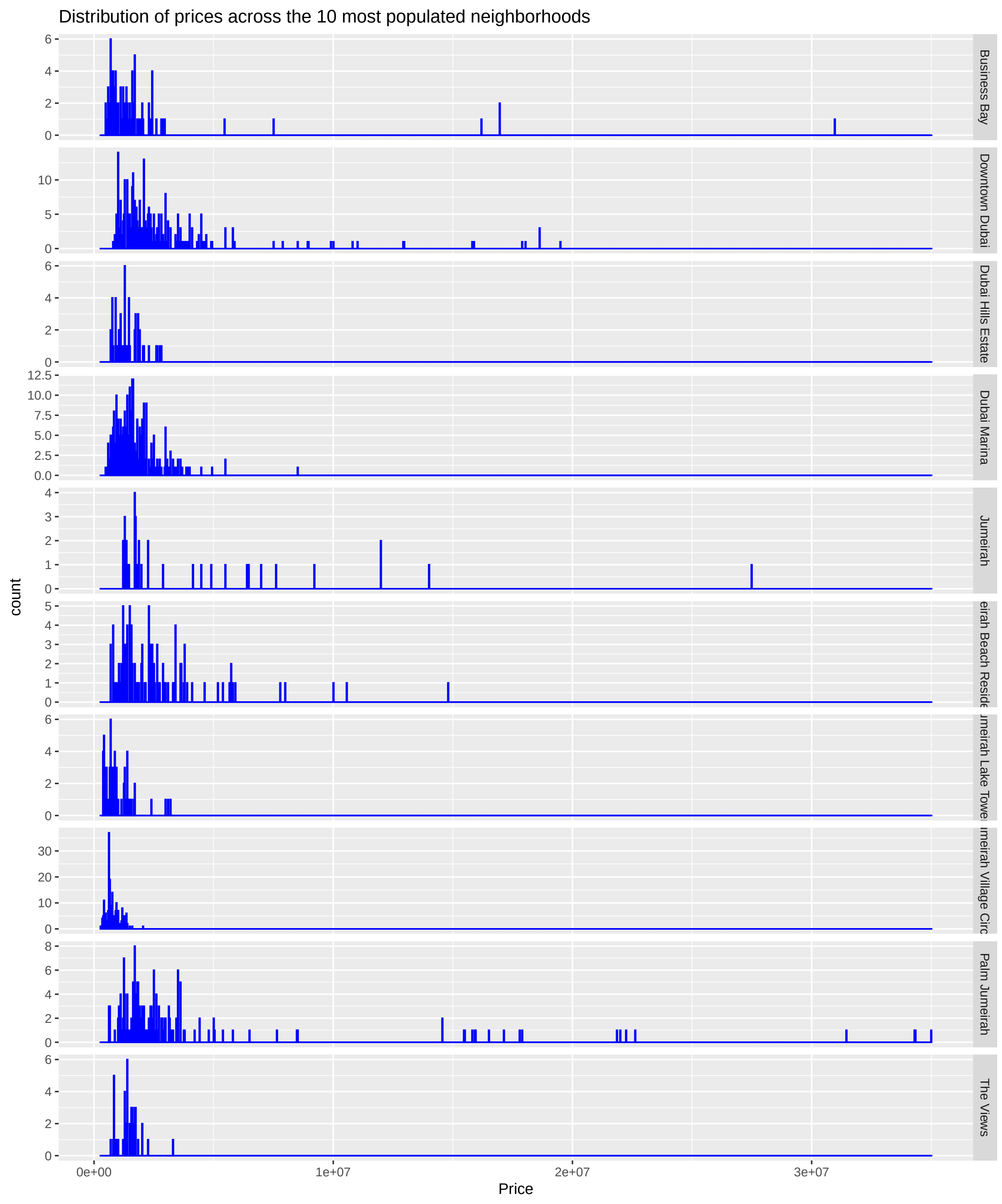


The bar chart confirms my hypothesis, and shows us that there is a large variability between the average price of the most expensive neighborhood and the least expensive neighborhood. Additionally, we can see from the boxplot that Palm Jumeirah, Downtown Dubai, and Business Bay have a number of 'expensive' outliers. This is accurate with my view of the city in that these areas have a few 'staple' or 'ultra expensive/ luxury' buildings like the Burj Khalifa or Atlantas that would be a lot more expensive than the average apartment in the neighborhood.

I want to look at the neighborhoods with the largest number of houses and then assess any trends in the most populated areas



We can see that the top 10 neighborhoods hold a much larger percentage of the market than the rest of the neighborhoods. Therefore, I am going to look at the distribution of prices among those top 10 neighborhoods.



This shows us that the neighborhoods with the greatest deal of variation in their prices are Palm Jumeirah and Downtown Dubai. In order to be able to better analyse the data and create a better model, I have decided to remove the outliers from the data and then assess the effect that that would have on the data.

Remove outliers

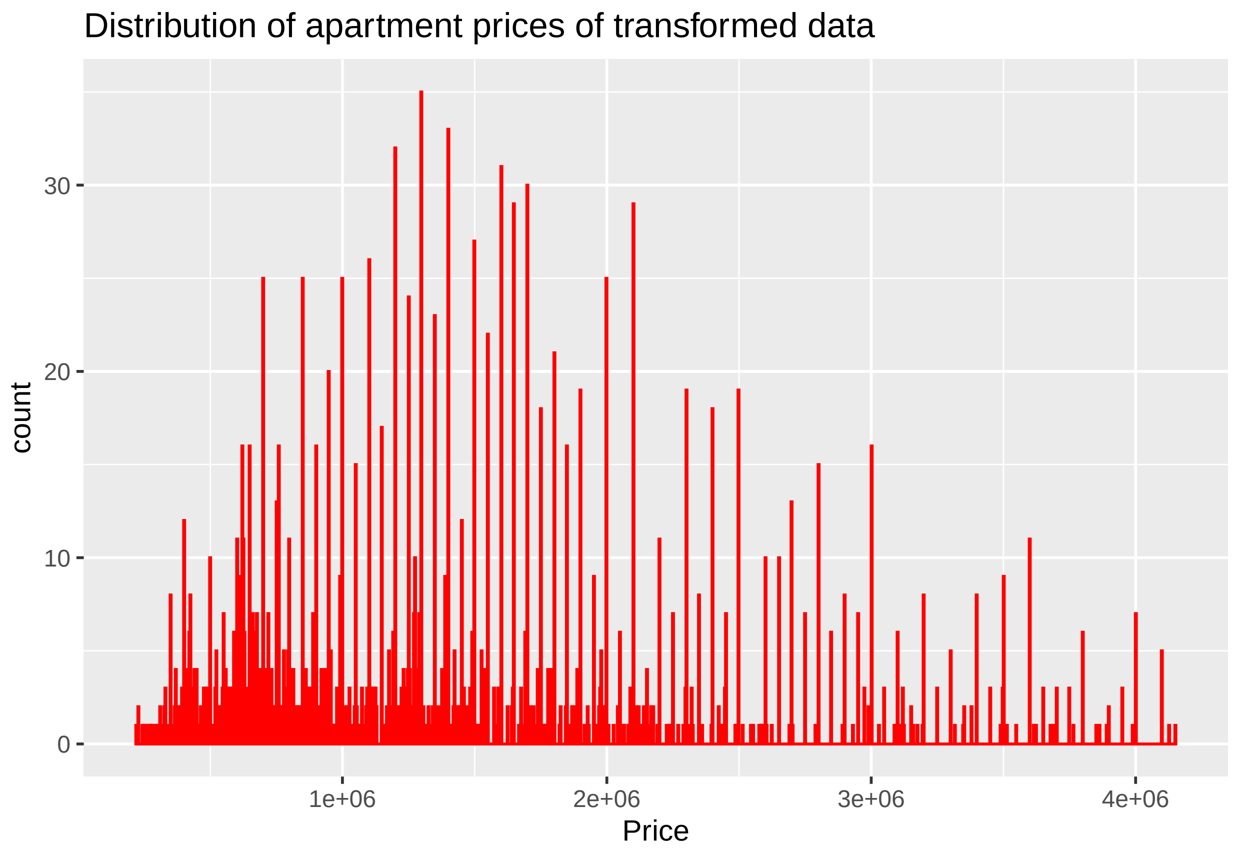
```
outliers <- boxplot(house$price, plot = FALSE)$out
```

Here we see that there were around 130 outliers that were removed from the dataset, indicating the removal of the ultra luxury apartments from our models. This is favorable for me because the aim of this study is to analyze these properties for individuals that are looking for affordable or at least average housing.

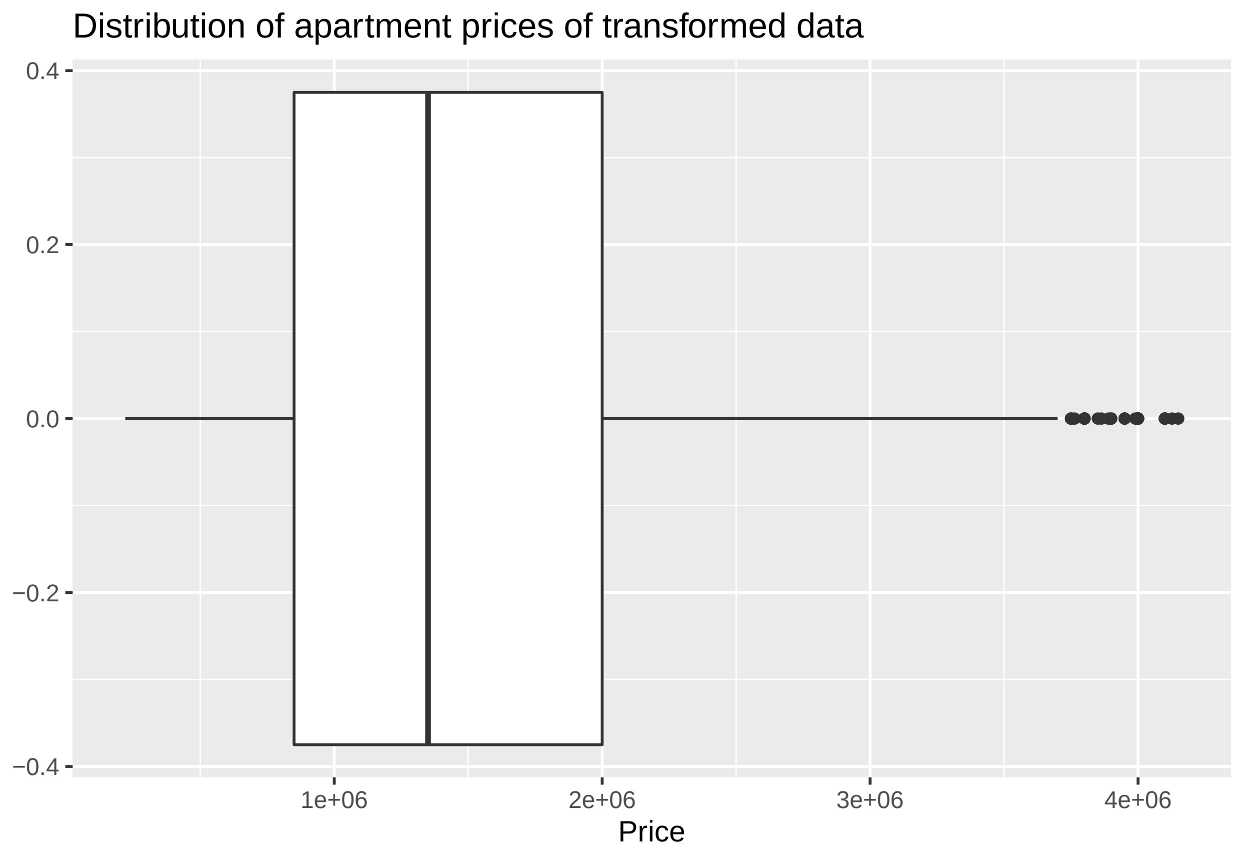
I will transform the data here to account for the removal of the outliers identified above. and then plot a histogram and boxplot of the updated data set.

```
house_trans <- house[-which(house$price %in% outliers),]
```

```
par(mar = c(4, 4, .1, .1))
ggplot(house_trans, aes(x = price)) + geom_histogram(bins = 1000, color =  "red", fill = "red") + labs(
```

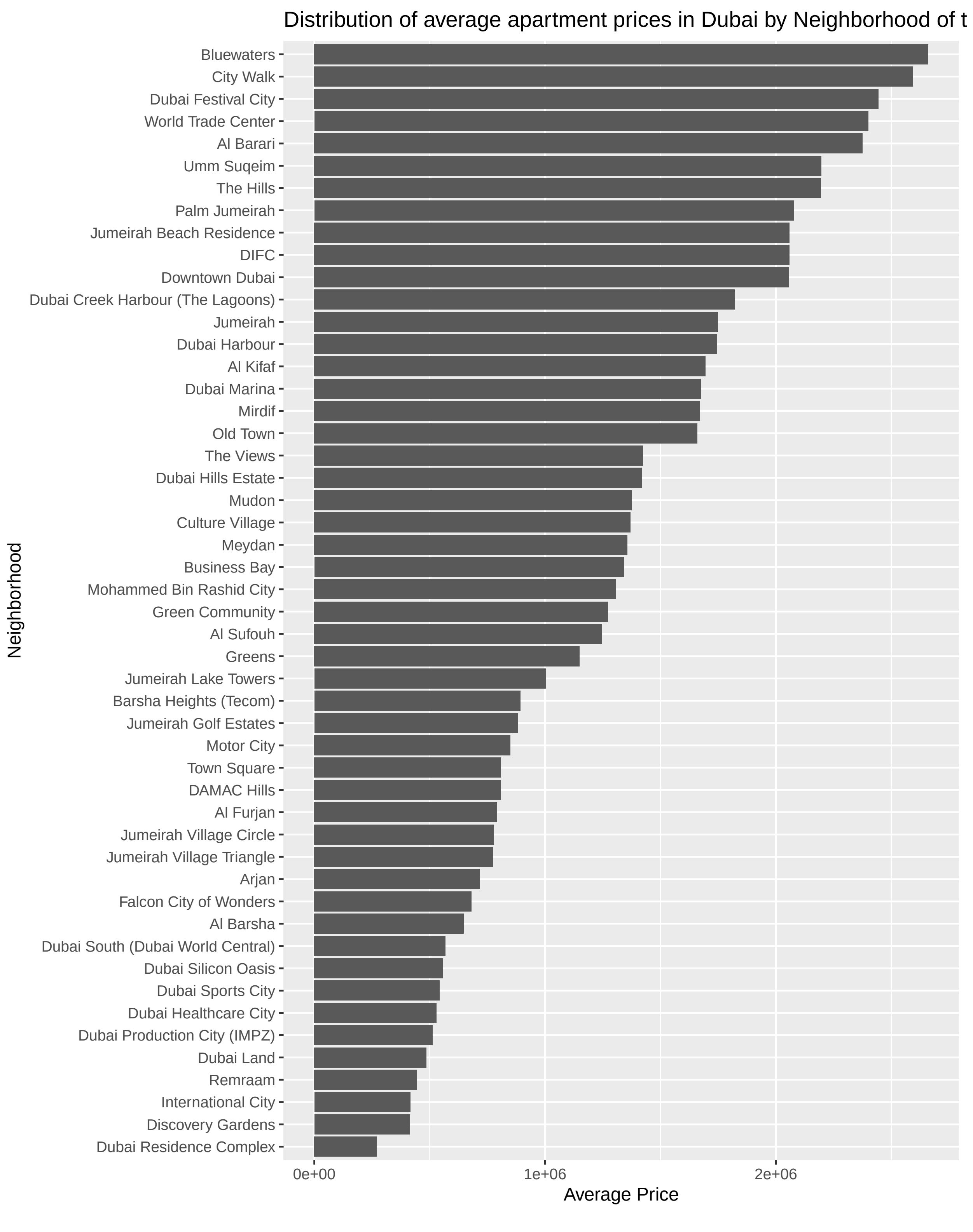


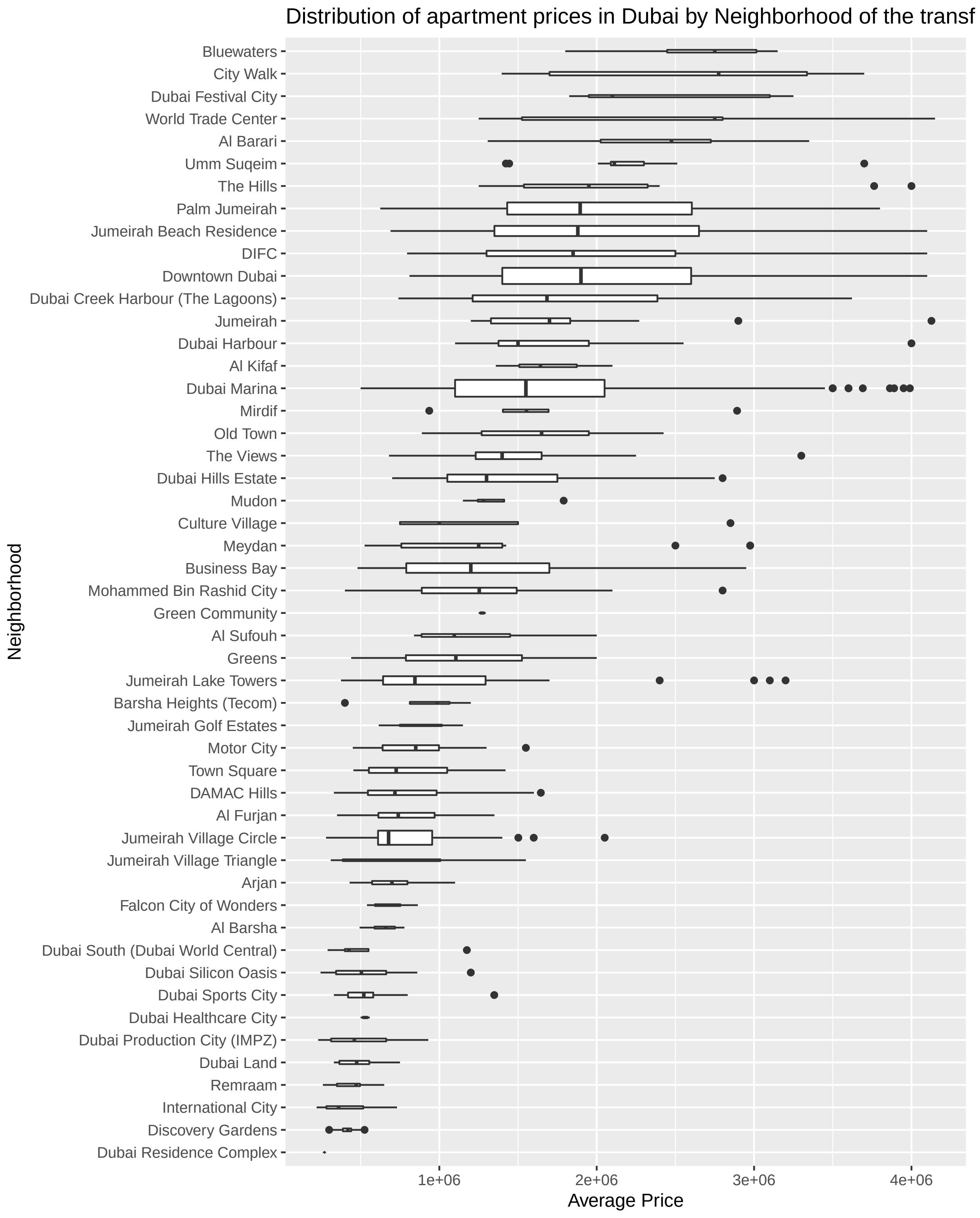
```
ggplot(house_trans, aes(x=price)) + geom_boxplot() + labs( title = "Distribution of apartment prices of
```



From looking at the two plots above, I believe that they look like a much better distribution of the prices and distributions that I think will be conducive to the model building and prediction aspect of this project.

Looking at the change in distributions in the average price by neighborhoods

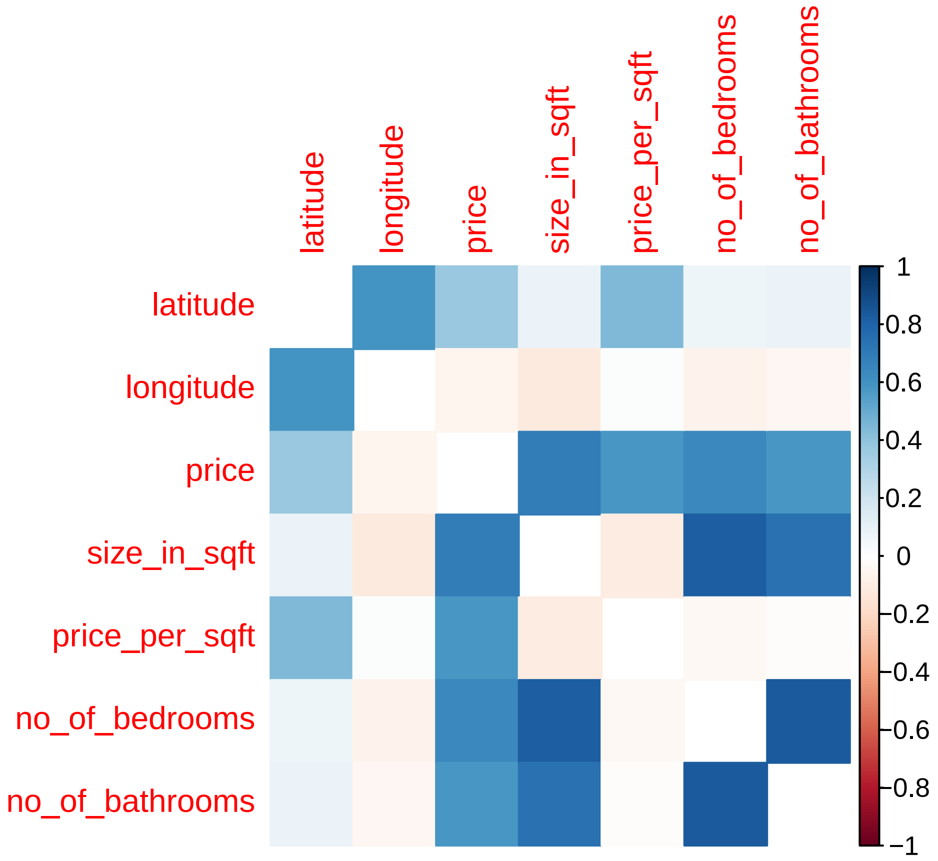

Distribution of average apartment prices in Dubai by Neighborhood of t
Bluewaters
City Walk
Dubai Festival City
World Trade Center
Al Barari
Umm Suqeim
The Hills
Palm Jumeirah
Jumeirah Beach Residence
DIFC
Downtown Dubai
Dubai Creek Harbour (The Lagoons)
Jumeirah
Dubai Harbour
Al Kifaf
Dubai Marina
Mirdif
Old Town
The Views
Dubai Hills Estate
Mudon
Culture Village
Meydan
Business Bay
Mohammed Bin Rashid City
Green Community
Al Sufouh
Greens
Jumeirah Lake Towers
Barsha Heights (Tecom)
Jumeirah Golf Estates
Motor City
Town Square
DAMAC Hills
Al Furjan
Jumeirah Village Circle
Jumeirah Village Triangle
Arjan
Falcon City of Wonders
Al Barsha
Dubai South (Dubai World Central)
Dubai Silicon Oasis
Dubai Sports City
Dubai Healthcare City
Dubai Production City (IMPZ)
Dubai Land
Remraam
International City
Discovery Gardens
Dubai Residence Complex
Neighborhood
0e+00
1e+06
2e+06
Average Price



The top 5 neighborhoods changed after we removed our outliers. This shows that previously, Palm Jumeriah, Downtown Dubai, Jumeirah etc. had average lower prices across the neighborhood however had a few properties that really dragged the average price of the area up. The new top 5 costing neighborhoods are Bluewaters, CIty Walk, Dubai Festival City, World Trade Center, and Al Barari. Again this intuitively makes sense, since these areas tend to be on average more expensive since they are in 'hotspots' of the city - where a lot of the malls and tourists reside.

2. Correlation matrix

Bellow I have created a correlation matrix of all the numeric variables in the data.

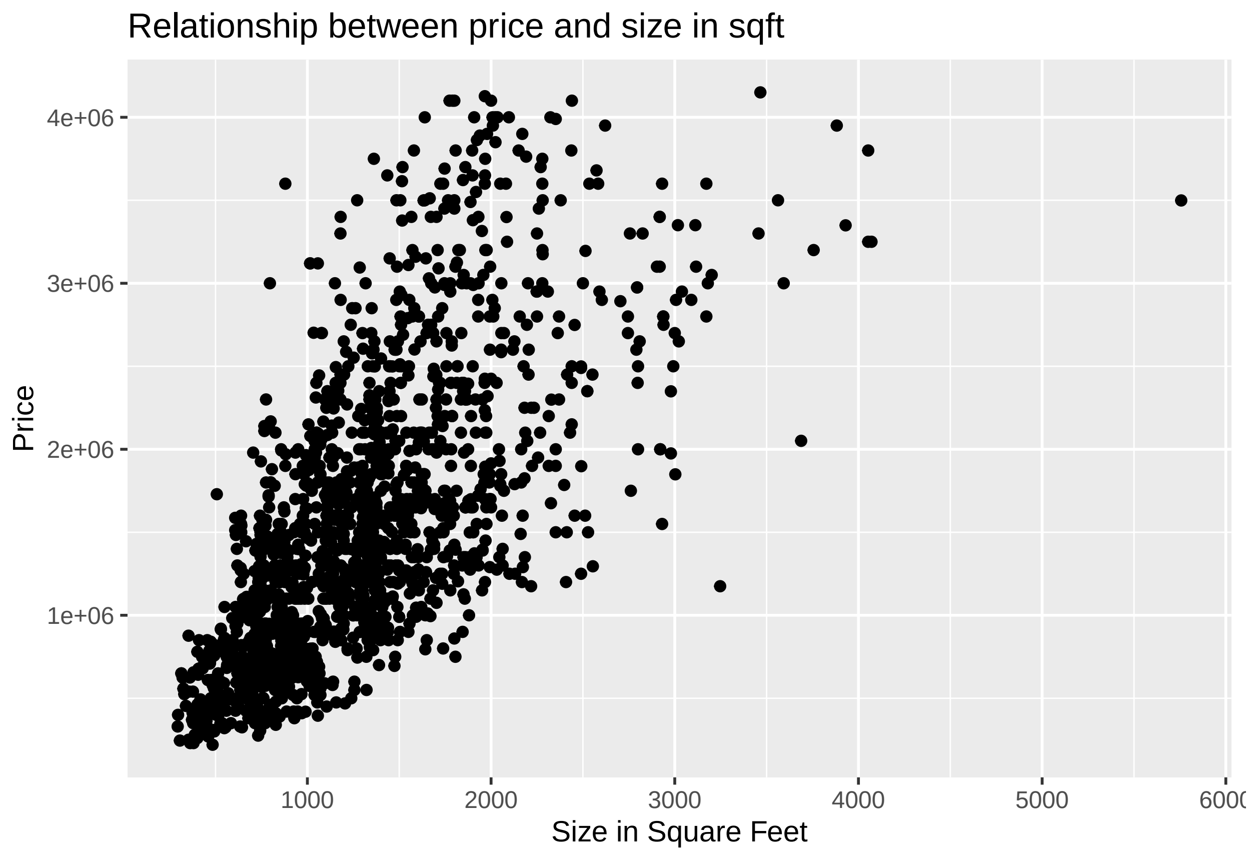


The above correlation matrix shows us some pretty obvious correlations and some that I may not have expected. I have decided to list below the observations that I made from this correlation matrix:

1. The number of bathrooms in an apartment is strongly positively correlated with the number of bedrooms.
2. Number of bathrooms is also positively correlated with the size_in sqft and the price of the apartment.
3. Interestingly the number of bathrooms is not correlated at all with the price per sqft.
4. The second and third observation are the same for the number of bedroom.
5. Interestingly price has a very slight positive correlation with the latitude coordinate of the property. I predict that this is because those properties are in areas that are closer to the beach.

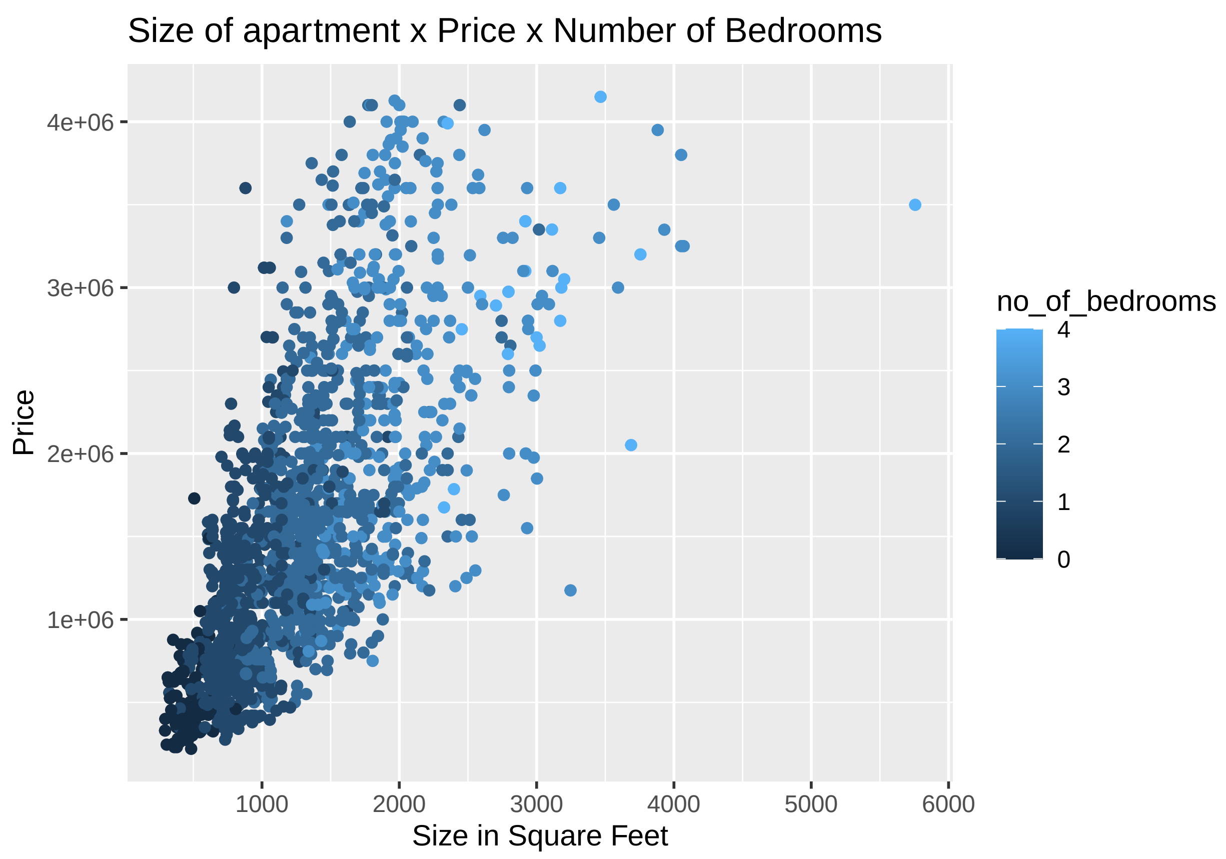
I have decided to look at the relationship between price and a few more variables below:

Firstly I think it is super interesting to look at the relationship between price and size. I would think that the larger the apartment, the higher the price, but I would be interested to see what relationship is displayed.

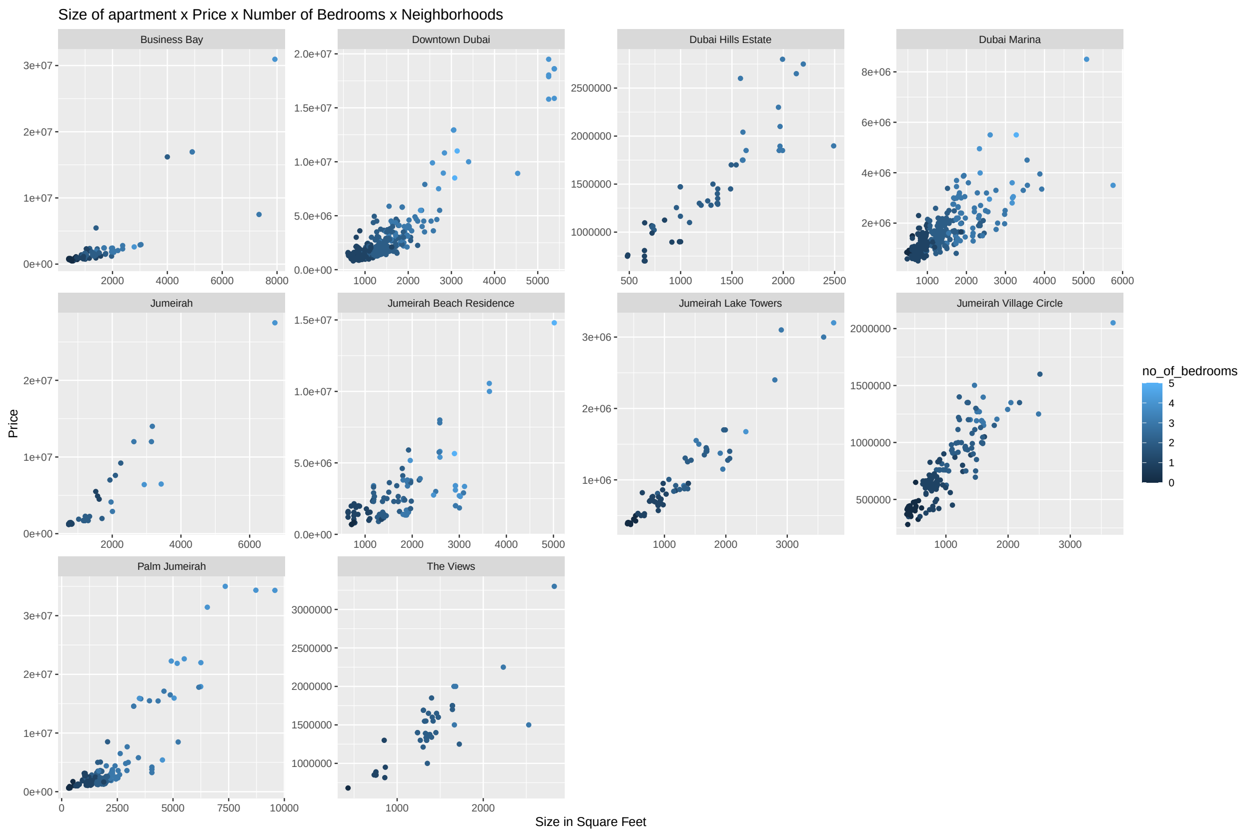


The plot affirms the hypothesis I had, indicating that the larger the apartment, the higher the price. However, another observation is that the apartments tend to be a lot more clustered towards the smaller apartments than the others. (There is more variability in price with the larger apartments.)

After seeing that plot, I also found myself interested in how the sqft of those properties were distributed in regards to the number of rooms.



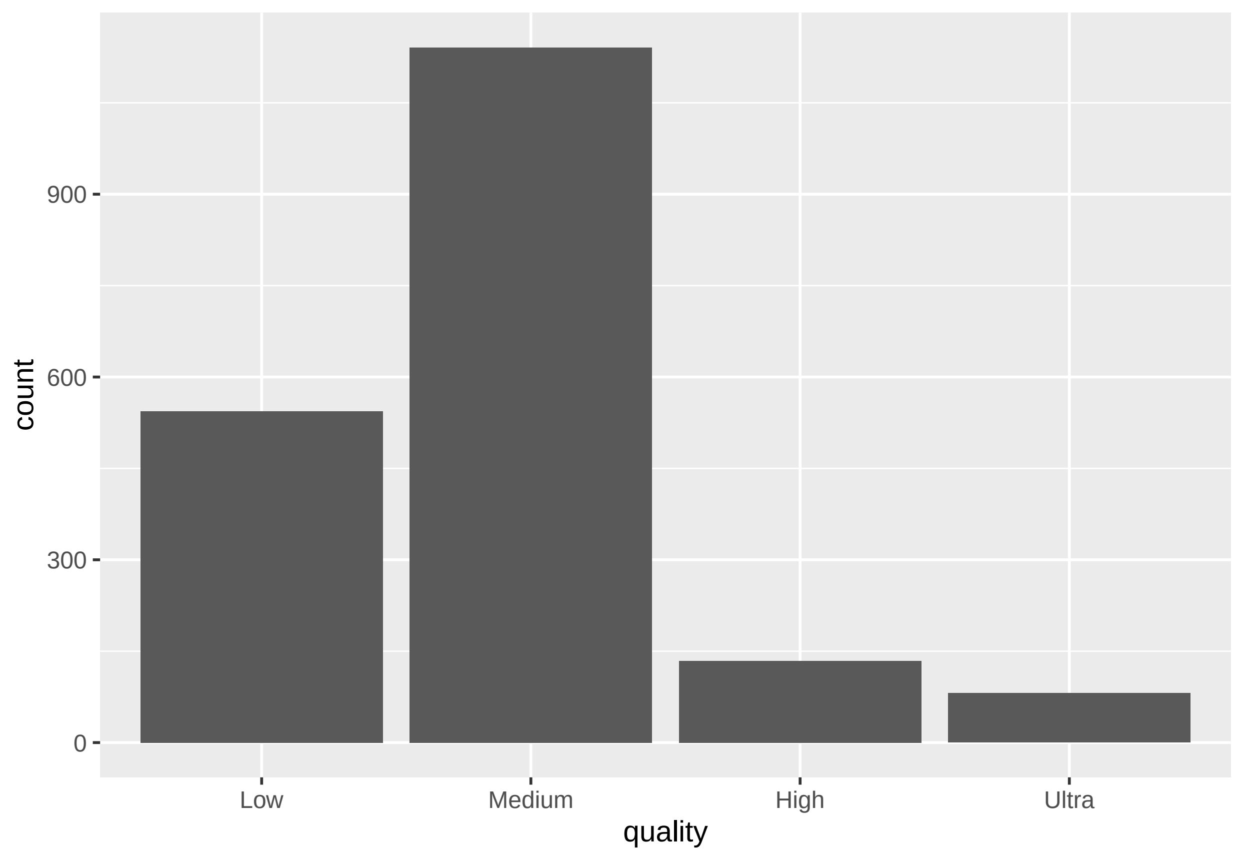
The plot above shows us that there is actually overlap in the different number of rooms in respect to the price and square footage. Therefore, it is not necessary that the larger apartments that are more expensive actually have more rooms. Another thing that I think would be useful to observe is the relationship of these variables with the neighborhoods they belong in. I think that this would definitely effect the distribution of properties. I decided to only look at the ten neighborhoods with the greatest number of apartments to simplify the viewing og the graph.

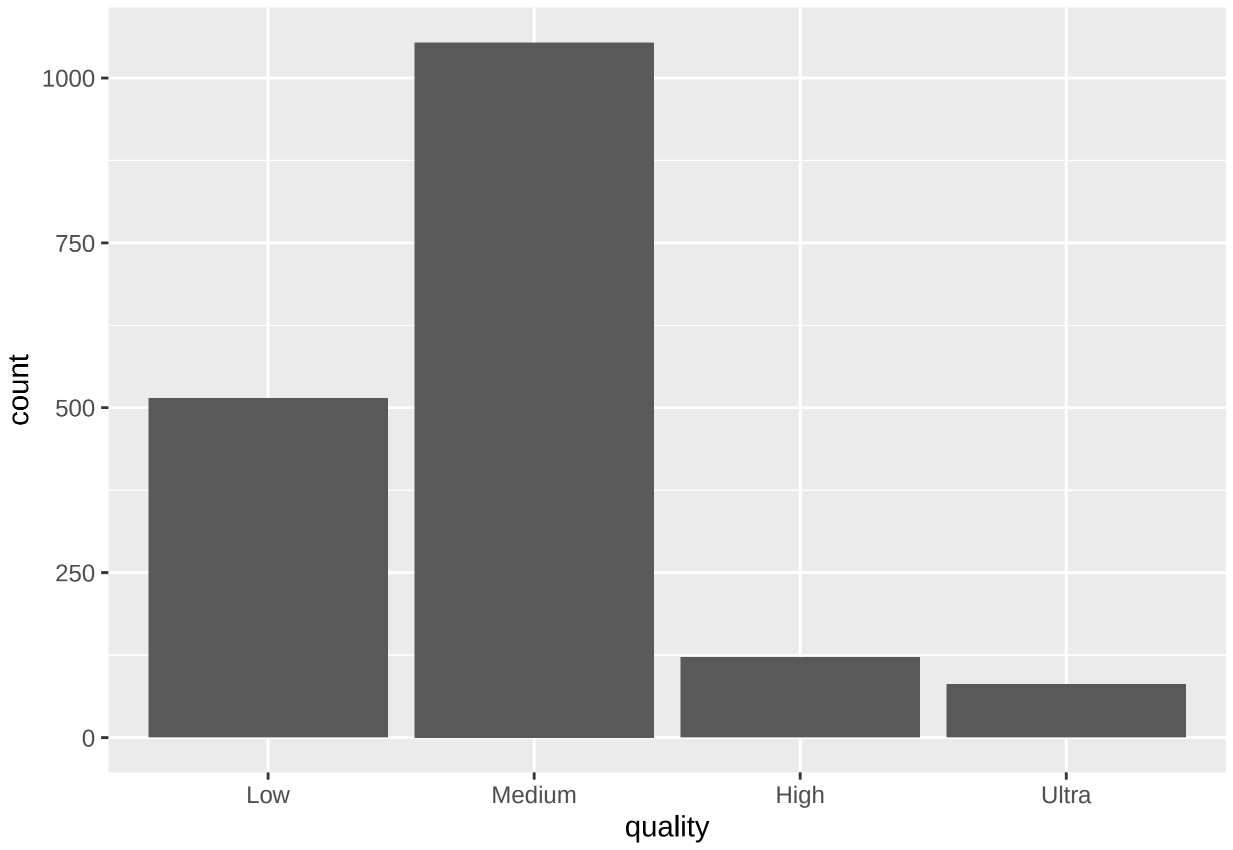


What we can see here is that properties in most of the areas follow the same kind of distribution as the overall plot above in that the larger the property the higher the price, and although there is some overlap between the number of bedrooms in some places, it is still the overall observation that the larger the property the higher the number of rooms.

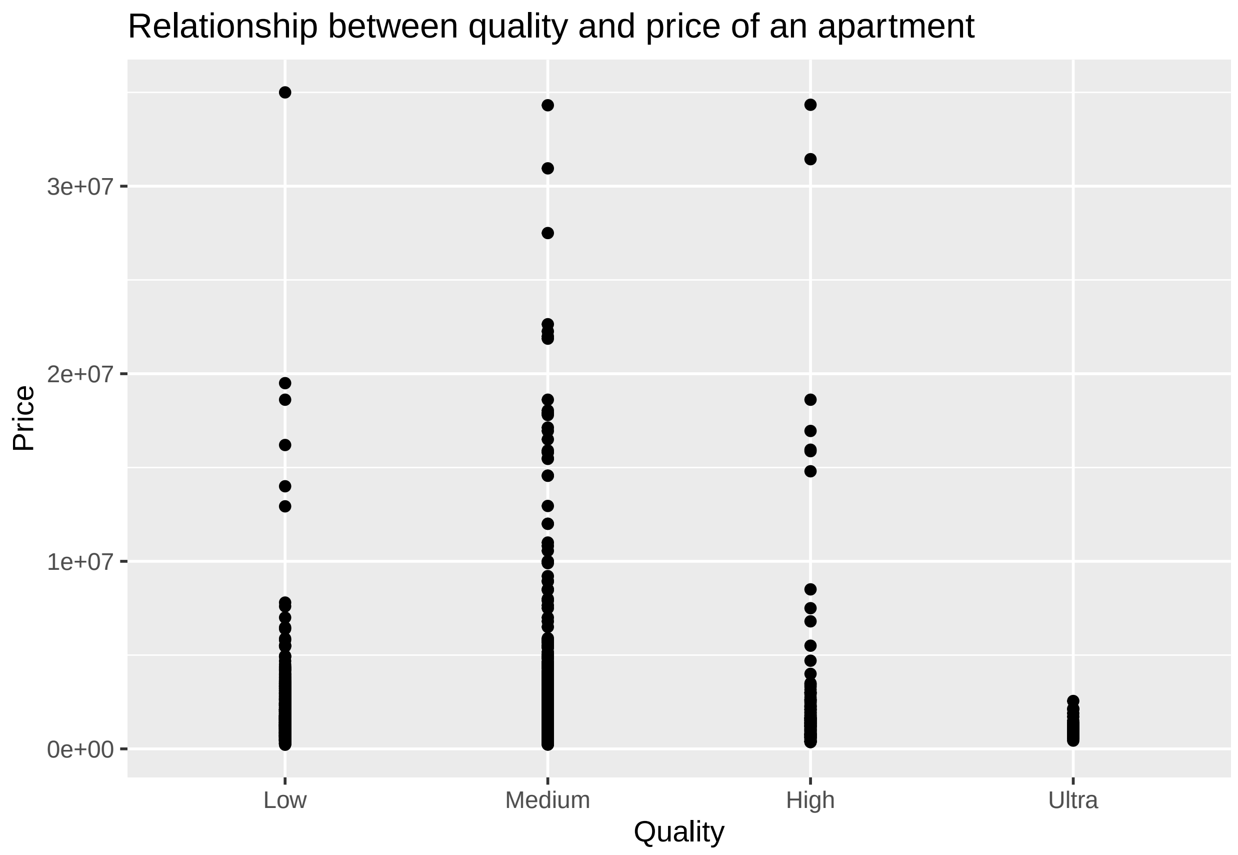
Another variable that I would like to explore is the quality variable. The quality variable is a slightly subjective observation that ranks the apartments into 4 different categories according to their 'quality' or their luxury value: Low, medium, high, and ultra.

Below I am going to graph two bar plots: the distribution of quality with the original data set and then the distribution of quality after the outliers have been taken out. I hypothesize that the number of ultra and high apartments will decrease significantly as they tend to be the more expensive ones.


count
900
600
300
0
Low
Medium
High
Ultra
quality



What I find quite surprising here is that the number of each quality variable decreased, but the overall distribution of ‘quality’ stayed the same a much higher value of Medium, followed by low, then high and finally Ultra. I think to fully understand the distribution of quality I want to look at the relationship between quality and price.



What this plot here shows us, which is very interesting, is that quality does not have that big of an effect on the price. Intuitively, I am not sure if I fully agree with that statement, because I would think that quality is one of the biggest determinants of price. It is still an interesting observation, that I will be sure to look further into.

That concludes the exploratory data stage of this project, I will now move on to the model building section of the project.

4 Data Prep

Prior to building our models I need to prep the data sets by getting them in the format that I want. This is done through stratified data splitting and through K-fold Cross Validation.

4.1 Data Splitting

First of all let us split our data. I have decided to split my data according to a 80/20 split, because I think this will leave the testing set with a decent number of observations. I am aware that my dataset is not the biggest data set, however, I still think that an 80/20 split is a good way to split the data so that the model can learn as much as possible. I have also set a seed of 14235 for the simple reason of consistency and reproducibility of my results, every time I run it. Finally, I have stratified the data on the response variable (price) so that the training data set is as close to the original data set as possible.

```
set.seed(14235)
```

```r
house_split <- initial_split(house_trans, prop = 0.80, strata = price)
house_train <- training(house_split)

house_test <- testing(house_split)
```

Let us check the dimensions of the two data sets:

```r
dim(house_train)
```

```
## [1] 1417   37
```

```r
dim(house_test)
```

```
## [1] 355  37
```

Here, we see that we have 1417 observations in my training data set and 356 in my testing data set.

4.2 K-Fold Cross Validation

I am also going to use cross validation to help with the issue of imbalanced data.

```r
house_folds <- vfold_cv(house_train, v = 10, repeats = 5)
house_folds
```

Because building models take so much computng time, I decided to save the results to an RDA file once I had the model I wanted so I could go back and load it later with no time commitment.

```r
save(house_folds, house_train, house_test, file = "~/Desktop/PSTAT 131 Final/house-Modeling-Setup.rda")
```

5 Model Building

Now it is time for the actual model building. Before I delve into the specifics of how I am going to build each model, I wanted to give an overview of the process that I am going to be taken while building the models. While this is the most important section, it is the most tediious, which is why I saved all of the models that I ran in this section in a file and just called on them when running. The reason behind this is that each model took up to hours to run, which I could not afford every time I needed to run or fit the model. In the sections below, I have kept my code in for two of the models to show what kind of work I was doing and commented what each code does, but have not run each section every time, however for the rest I have just loaded the models below (for the sake of not repeating myself).

1. Building and running each model

a. Setting up the recipe
b. Setting up the model specification (setting the mode as regression)
c. Coding the workflow (Adding the model soec and the recipe)
d. Setting up the tuning grid with the respective parameters for each model
e. Run the model

2. Plotting the autoplots of each model
3. Analyzing the models (to pick the best model)
4. Selecting the best model
5. Testing the best model on the testing data set

a. Fit the model to the testing data set

I am going to load in the models below:

```
load("/Users/janahindiyeh/Desktop/PSTAT 131 Final/models.rda")
load("/Users/janahindiyeh/Desktop/PSTAT 131 Final/models1.rda")
load("/Users/janahindiyeh/Desktop/PSTAT 131 Final/models2.rda")
```

5.1 Recipe

Since each model is going to use the same model, I have created one recipe for all the models to use. This is essentially like a mini manual that each model will use and apply differently according to the purpose of each given model. I have used all the variables in the data (except id) because I believe that they all have a purpose in the model building section. After specifying the recipe itself, I have added *step_novel* which will assign a previously unseen factor level to a new value, then I used *step_dummy* to convert any character and factor variables into one or more numeric binary values. NExt, i used *step_zv* that removes all variables that have only a single value (or zero variance), and finally I used *step_normalize* to standardize (center and scale) the data so that it follows a stanard normal distribution curve (mean = 0 and standard deviation = 1).

```
house_recipe <-
  recipe(formula = price ~., data = house_train) %>%
  step_novel(all_nominal_predictors()) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors()) %>%
  step_normalize(all_predictors())


summary(house_recipe)
```

You can see above the summary of our recipe. This includes the variable name, the type of variable it is, the role that it plays (are all predictors escept for price which is the outcome variable), and the source (they are all original).

5.2 Ridge Regression

Firstly, we will set up model specification, using hyperparametric tuning to find the ridge model that performs the best.

```
ridge_spec <-
  linear_reg(penalty = tune(), mixture = 0) %>%
  set_mode("regression") %>%
  set_engine("glmnet")
```

Next, I will create the associated workflow object.

```
ridge_wf <- workflow() %>%
  add_recipe(house_recipe) %>%
  add_model(ridge_spec)
```

I am also going to create a grid of the values of penalty I am going to be using. I have decided to range the penalty values from -5 to 5 with 50 levels

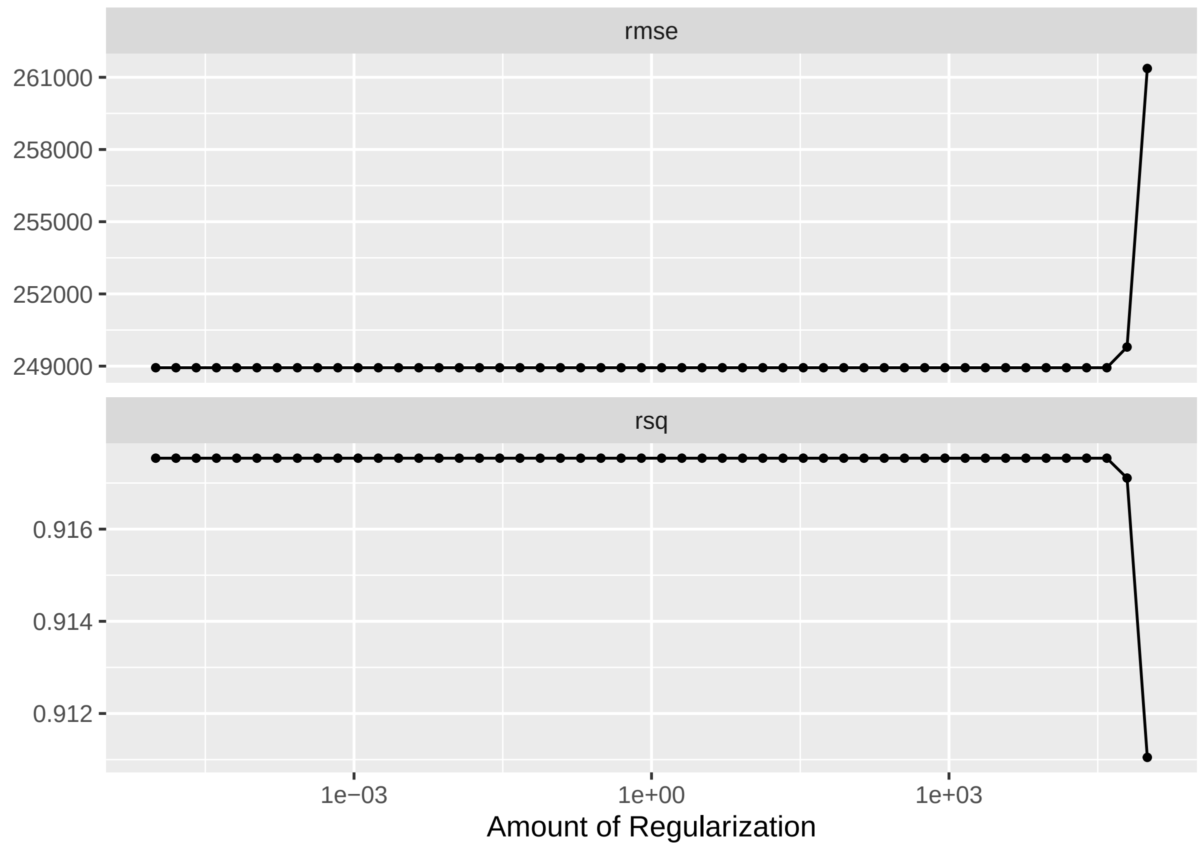
```
ridge_grid <- grid_regular(penalty(range = c(-5, 5)), levels = 50)
```

Finally, I can now fit the ridge regression model using tune_grid

```
tune_ridge <- tune_grid(
  ridge_wf,
  resamples = house_folds,
  grid = ridge_grid
)
```

Finally, I can plot the autoplot of the ridge regression

```
autoplot(tune_ridge)
```

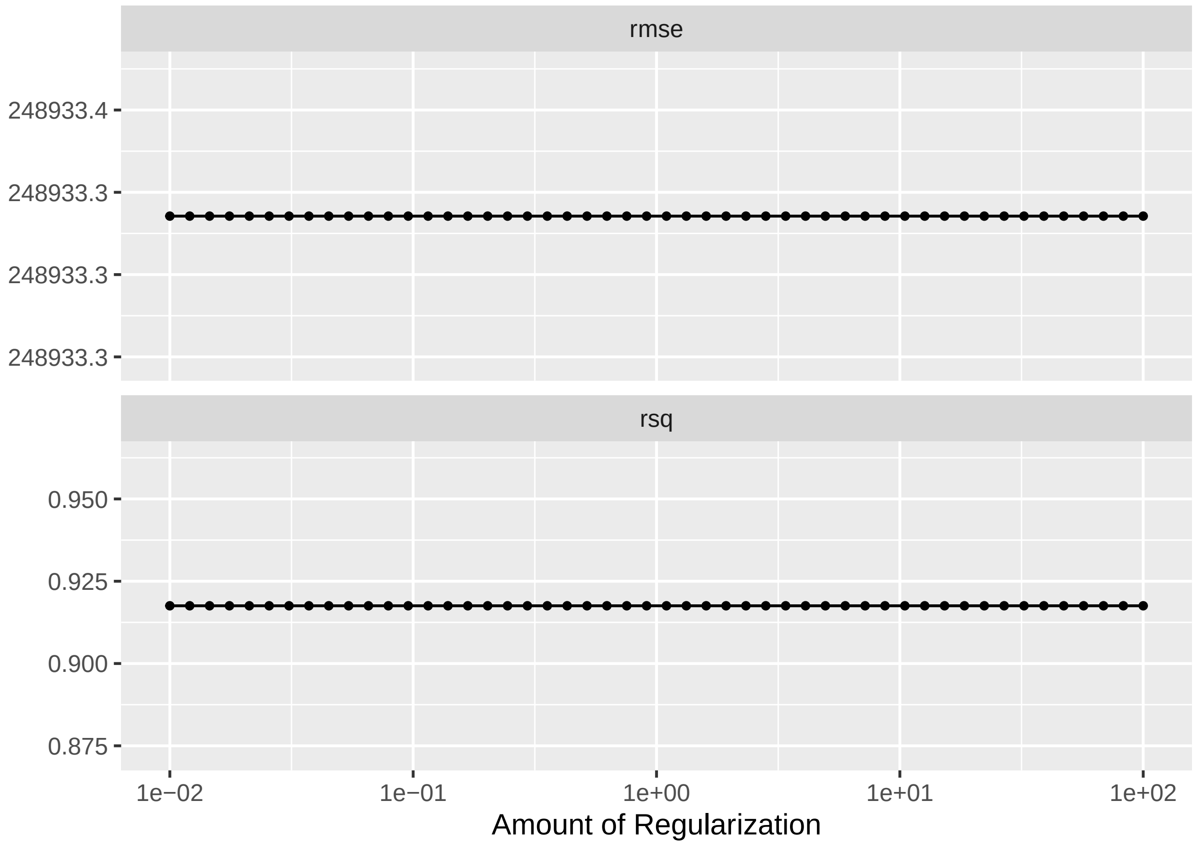


The autoplot here shows pretty linear values of RMSE and RSQ accross all levels of regularization but then a spoke towards the end of RMSE and a fall at the end of RSQ.

5.3 Lasso Regression

Here I am plotting the Lasso Regression autoplot

```
autoplot(tune_lasso)
```

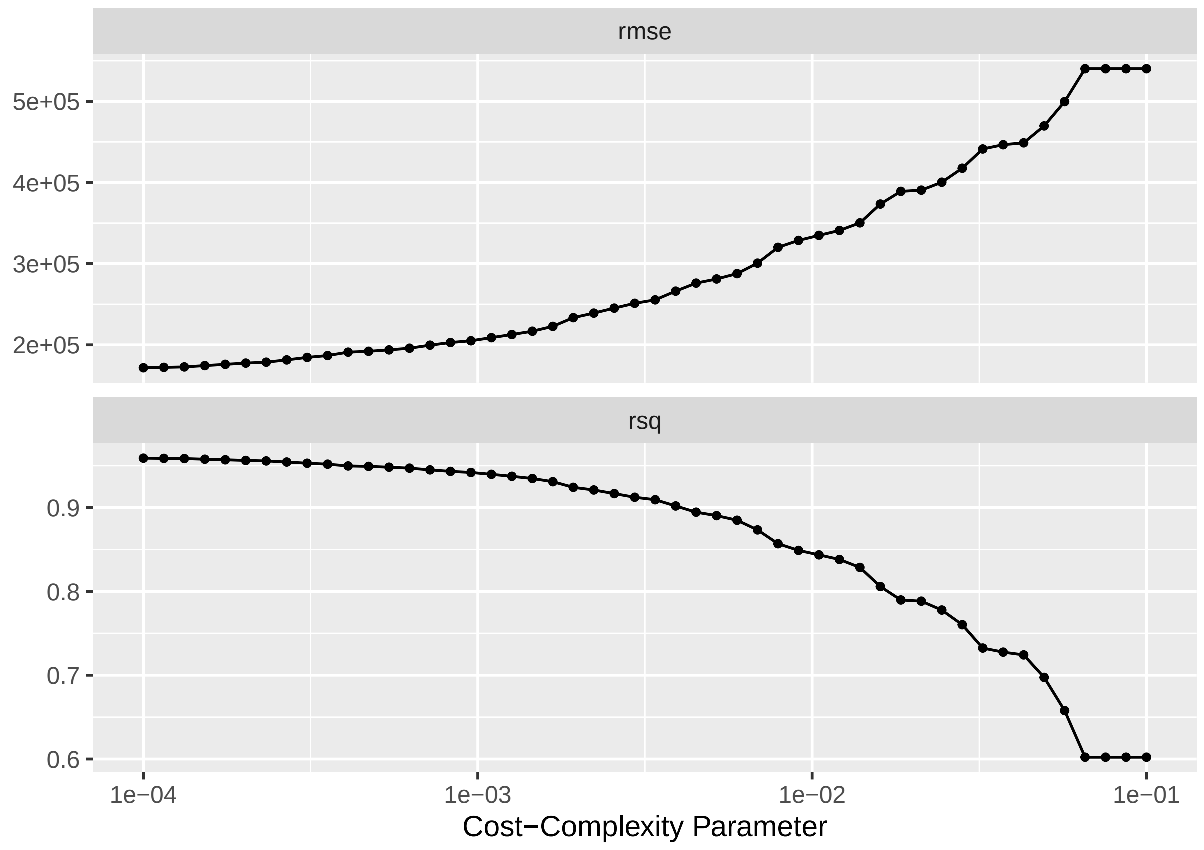


Similar to the ridge regression model the RMSE and RSQ models remain linear accross all amounts of regularization, which I find very odd.

5.4 Decision Tree

Here I am plotting the decision tree autoplot

```
autoplot(tune_tree)
```

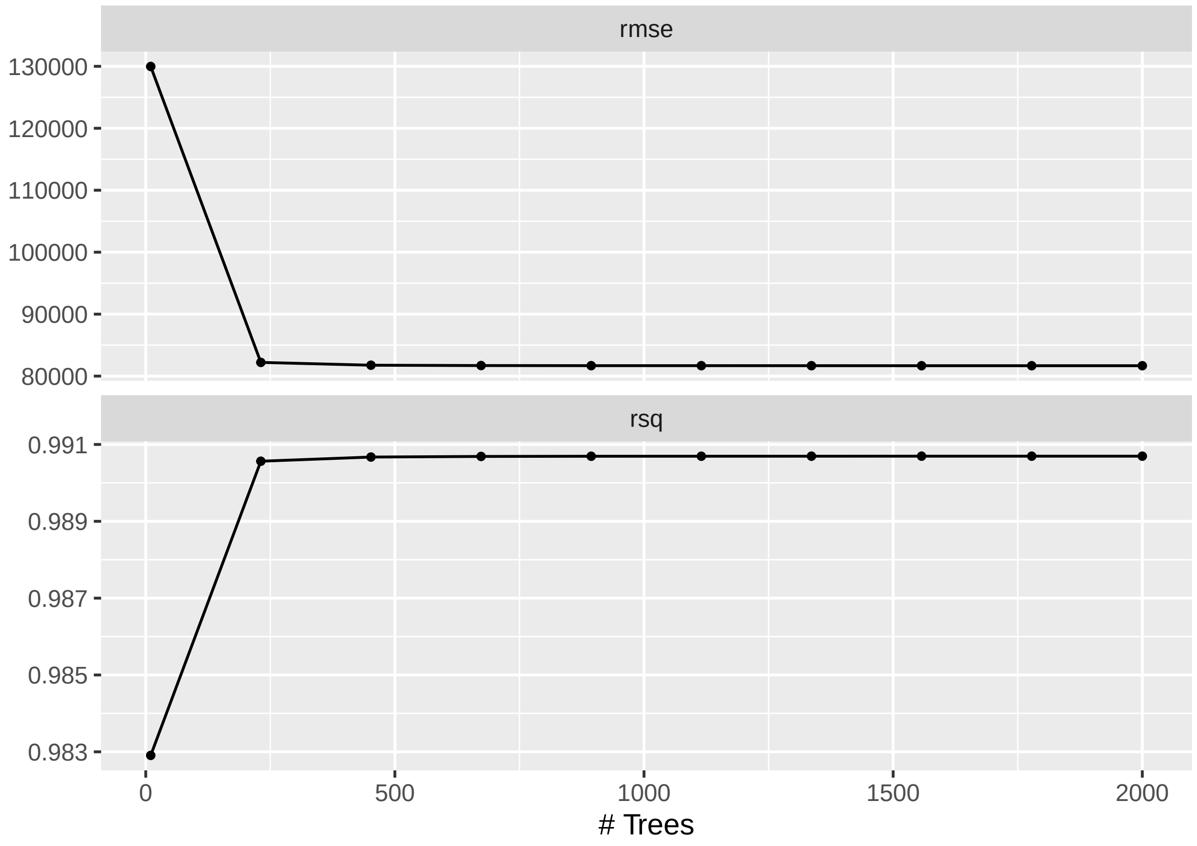


The autoplot for the Decision tree shows some variation. The RMSE metric increases as a steady rate and then platues towards the highest end of the Cost-Complexity Parameter, whereas the exact opposite occurs with the RSQ metric.

5.5 Boosted Tree

Here I am plotting the decision tree autoplot

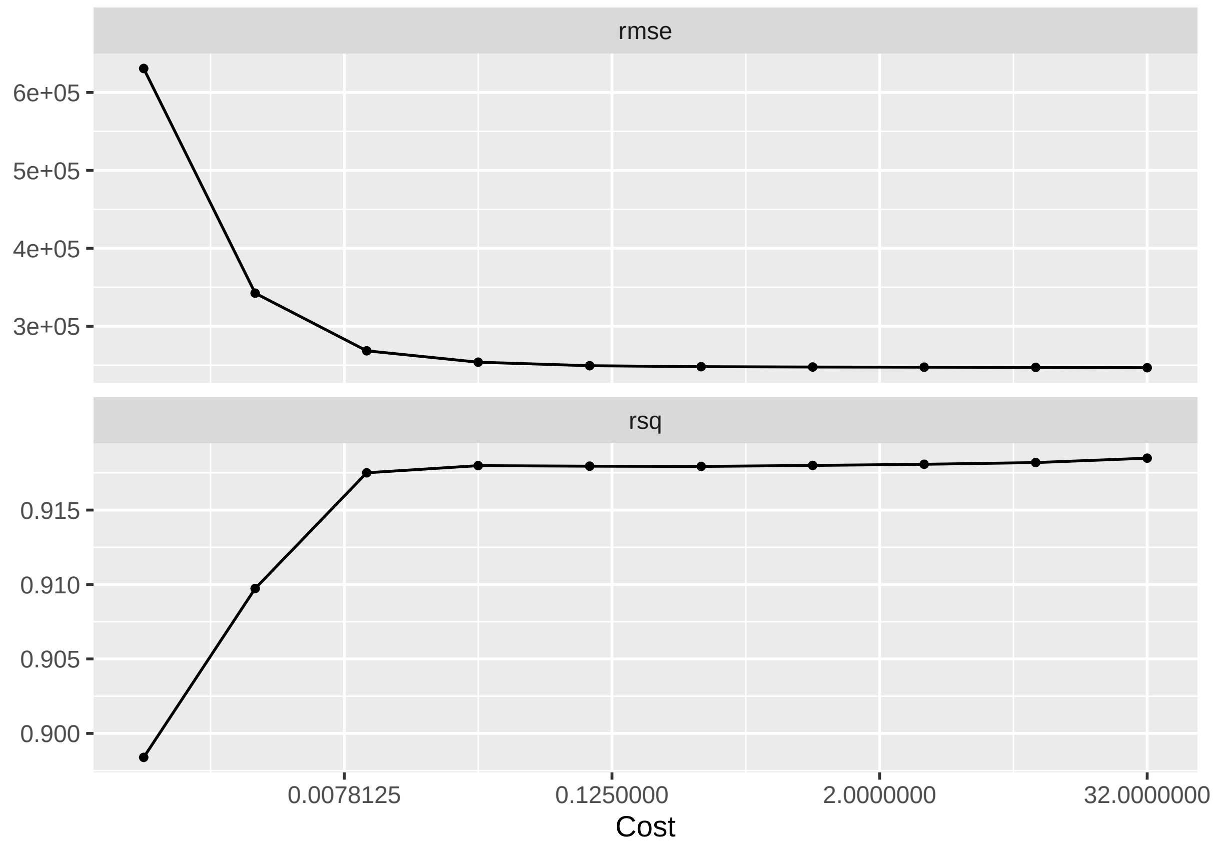
```
autoplot(tune_boosted)
```



The boosted tree autoplot shows a sharp decrease from 0 to 250 trees of the RMSE metric and a sharp increase from 0 to 250 of the RSQ metric. After that both plateu and become almost linear.

5.6 Support Vector Machines

```
autoplot(tune_svm)
```



Finally, the SVM autoplots are again parallel for the two metrics. RMSE decreases gradually from a cost of 0 to a cost of ~0.0078 and the platues pretty quickly to a straight line after that. The opposite happening for the RSQ metric.

Now that we are done model biulding we can move on to the next section, which is finding out which model is the best (which model we are going to select).

6 Model Analysis

Now that we have built all of out models, it is time to test them to see which model gives us the best result, and which one we should use to test our testing data.

6.1 Metrics

First of all, I want to touch quickly on the two metrics I have been using in the model building and in the autoplots that were just displayed.

1. **RSME: Root Mean Square Error**

$$RSME = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$$

Root Mean Square Error is the standard deviation of the prediction errors or the residuals. Residuals are a measure of how far from the regression line data points are. Therefore, it tells you how concentrated the data is around the line of best fit.

2. **RSQ: R- Squared**

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(\hat{x}_i - x_i)^2}{\sum_{i=1}^{N}(x_i - \bar{x}_i)^2}$$

RSQ is a measure of fit that calculates the relationship between variables. It measures how much variation of the outcome variable is explained by the predictor variables in the model, essentially measuring the accuracy of the model.

6.2 Model Selection

First of al I am going to run show_best() on all the models so that we can get the model that has the optimal RMSE from each model.

```r
ridge_best <- select_best(tune_ridge, metric = "rmse")
lasso_best <- select_best(tune_lasso, metric = "rmse")
tree_best <- select_best(tune_tree, metric = "rmse")
boosted_best <- select_best(tune_boosted, metric = "rmse")
svm_best <- select_best(tune_svm, metric = "rmse")
```

I will then finalize each of the models to upfate the final recipe by replacing tune() with the specific model chosen for each model above.

```r
ridge_final <- finalize_workflow(ridge_wf, ridge_best)
lasso_final <- finalize_workflow(lasso_wf, lasso_best)
tree_final <- finalize_workflow(tree_wf, tree_best)
boosted_final <- finalize_workflow(boosted_wf, boosted_best)
svm_final <- finalize_workflow(svm_linear_wf, svm_best)
```

Next I need to fit all the models again by using the whole training data set.

```r
ridge_fit <- fit(ridge_final, data = house_train)
lasso_fit <- fit(lasso_final, data = house_train)
tree_fit <- fit(tree_final, data = house_train)
boosted_fit <- fit(boosted_final, data = house_train)
svm_fit <-fit(svm_final, data = house_train)
```

Finally, I can evaluate the performance of each of the models on the trainng dataset so that I can finally choose the final model.

```r
ridge <- augment(ridge_fit, new_data = house_train) %>%
  rmse(truth = price, estimate = .pred)
lasso <- augment(lasso_fit, new_data = house_train) %>%
  rmse(truth = price, estimate = .pred)
tree <- augment(tree_fit, new_data = house_train) %>%
  rmse(truth = price, estimate = .pred)
boosted <- augment(boosted_fit, new_data = house_train) %>%
  rmse(truth = price, estimate = .pred)
svm <- augment(svm_fit, new_data = house_train) %>%
  rmse(truth = price, estimate = .pred)

names <- c("Ridge", "Lasso", "Decision Tree", "Boosted Tree", "SVM")
RSMEs <- c(ridge$.estimate, lasso$.estimate, tree$.estimate, boosted$.estimate, svm$.estimate)

results <- tibble(Model = names, RSME = RSMEs)
results
```

Very clearly, and by far we see that the boosted_tree model is the best model in this project. It has a RSME value of 29.84617, whereas the rest of the RSME values range well above 100,000. A large value of RSME is indicative of the fact that the models are failing to account for important features underlying the data. This makes sense to me becuase after exploring the data and delving deeped into it I found that I just had insufficient data to allow the model to really learn from itself.

However, a RSME value of ~30 is pretty good, especially since we are trying to predict a price of a house. I think that when the prices of houses are minimum ranging in the hundreds of thousands to the millions, an estimate that is around 30 is not bad.

Now that we have selected our model we can continue to test the model further.

6.2.1 Model testing

Here is a show of the best boosted_tree models.

```r
show_best(tune_boosted)
```

And the model that I have finally picked is

```r
boosted_best
```

Therefore, the characteristics of our chosen model is that it ascribes tp the boosted tree model, it has 2000 trees, a mean of 81676.01 and a standard error of 2813.979 (The lowest STDV out of all of the other models). With respect to the other models in this project, this is by far the best, however It still makes me quetion the validity of the model on this data in general.

NExt I am going to evaluate the performance of the model on the testing set

```
boosted_test <- augment(boosted_fit, new_data = house_test) %>%
  rmse(truth = price, estimate = .pred)
boosted_test
```

I will also calculate the overall RSQ (which is the area under the curve) which will also give us an indication into how accurate our model is

```
#Calculate the overall ROC AUC
augment(boosted_fit, new_data = house_test) %>%
  rsq(truth = price, .pred)
```

The metric above signifies a R-squared value of 0.9928326 This indicates that around 99% of the variability observed in the price of the apartments is explained by the regression model. This therefore tells us that the fitted model I have coded above fit the data extremely well. Thus our model may indeed predict with the highest level of accuracy any new observations.

6.3 Conclusion

To conclude, after running through the project, doing my research, data analysis, exploration, model building, testing and analysis, the best model to predict apartment prices in Dubai is a Boosted tree model, and specifically Preprocessor1_Model10, and it was the best model BY FAR. It gave me an R-squared value of 0.9928 which signifies the utmost validity and accuracy. Despite this, I belive that this analysis was not perfect.

For potential improvements, I think that trying out a greater number of models like the random forest or neural networks models woukd have been super helpful and insightful, I just did not have the computing capacity since seach model was taking hours to run. I attempted (for the longest time) to run the random forest model, but to no avail. Additionally, another imrpovement could be to try and get ahold of more accurate and more up to date data, and also to more data. I think that if I were able to supply my models with more information to learn from my models would have performed a lot better,

Overall, this model project has really improved my knowledge of the real estate market and has been a great opportunity to build my experience with R and machine learning in general.