KC_House Final project Submission

Vidyasagar Kummarikunta

11/20/2020

Final Project Data Set: Housing Sales

Section 1:

. I am interested in identifying the patterns in the dataset to address the factors that affect the prices of houses. . The model built can eventually predict the house prices
. The dataset contains house sale prices for King County, USA which includes Seattle and the dataset is obtained from the Kaggle.
. Original source is - https://www.kaggle.com/harlfoxem/housesalesprediction
. The data is present for over a period of 1 year from May 2014 to May 2015.
. This dataset is good for evaluating simple regression models.
. I have observed that there are no missing values in the dataset.
. The packages that I am going to mostly use are 'dplyr' and 'ggplot' along with the basic R utilities
. The dataset has approximately 21613 records 21 different variables. So we can use around 20 different variables as a predictors to predict the house prices which are listed below in the analysis.

Section 2:

. My major goal is to identify the predictive variables for the house pricing.
. I am thinking to use a multivariate regression algorithm to predict the house pricing based on the available data. . As the first step in the process I imported the data and performed the required
cleaning. After cleaning, in my second step I looked for missing values, and if the variables are in right format in terms of datatype. Once the data is in the right format, third step is to visualize data, so I performed extensive visual analysis to understand the critical factors influencing the house prices. Additionally, I am also interested in understanding the overall information specific to each variable like out of all the houses sold, how many of them are 3 bedroom and how many are 2 bedrooms.

Using the data, I will address the below research questions:
– Build a machine learning model to predict the house prices in King County, USA
–Narrow down on factors that predominantly influences the house price
–Understand the distribution of different variables given in the dataset
–In addition to individual factors, understand the effect of combination of factors that influence the house prices
–Understand the correlation between the variables by building correlation matrix – Descriptive statistics related to individual variables
–In addition to predictive model, I am also interested in exploring the classification algorithms to see if I can bucket the observations

. I will be building a flexible and scalable model so that we can feed a new set of data to the model and use it in other areas too.
. For visualization, I am using scatterplots, histograms and box plots.
. Currently, I am learning and exploring machine learning algorithms from online resources. I will see if I can incorporate them for my analysis.

Section 3:

. The code for importing the data is shown below. The dataset is checked for missing values and the dataset does not conatin any missing values. . As I am performing analysis using dplyr package I the imported data is converted
into "tibble" . For few variables I included basic visualization. . Some transformations of the data fid done for performing the analysis

---

```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
```

Loading housing data set

I choose the house price data set for my final project

```
data <- read.csv("kc_house_data.csv", header = TRUE)

head(data)

##           id            date   price bedrooms bathrooms sqft_living
sqft_lot
## 1 7129300520 20141013T000000  221900        3      1.00        1180
5650
## 2 6414100192 20141209T000000  538000        3      2.25        2570
7242
## 3 5631500400 20150225T000000  180000        2      1.00         770
10000
## 4 2487200875 20141209T000000  604000        4      3.00        1960
5000
## 5 1954400510 20150218T000000  510000        3      2.00        1680
8080
## 6 7237550310 20140512T000000 1225000        4      4.50        5420
101930
##   floors waterfront view condition grade sqft_above sqft_basement yr_built
## 1      1          0    0         3     7       1180             0     1955
## 2      2          0    0         3     7       2170           400     1951
## 3      1          0    0         3     6        770             0     1933
## 4      1          0    0         5     7       1050           910     1965
## 5      1          0    0         3     8       1680             0     1987
## 6      1          0    0         3    11       3890          1530     2001
##   yr_renovated zipcode     lat     long sqft_living15 sqft_lot15
## 1            0   98178 47.5112 -122.257          1340       5650
## 2         1991   98125 47.7210 -122.319          1690       7639
## 3            0   98028 47.7379 -122.233          2720       8062
## 4            0   98136 47.5208 -122.393          1360       5000
## 5            0   98074 47.6168 -122.045          1800       7503
## 6            0   98053 47.6561 -122.005          4760     101930
```

Converting the data frame into tibble

```
mydata <- as_tibble(data)
```

Final look at the tibble that will be analyzed

- There are variables like zipcode, year built etc that needs to be converted into proper datatype. In this case they needs to categorical variables.
- In the next steps I am going to handle the missing values and then before going deep into the analysis I will be doing the necessary data type conversions

```
data <- na.omit(data)
glimpse(data)

## Rows: 21,613
## Columns: 21
## $ id            <dbl> 7129300520, 6414100192, 5631500400, 2487200875,
19544...
## $ date          <chr> "20141013T000000", "20141209T000000",
```

```
"20150225T00000...
## $ price         <dbl> 221900, 538000, 180000, 604000, 510000, 1225000,
2575...
## $ bedrooms      <int> 3, 3, 2, 4, 3, 4, 3, 3, 3, 3, 3, 2, 3, 3, 5, 4, 3,
4,...
## $ bathrooms     <dbl> 1.00, 2.25, 1.00, 3.00, 2.00, 4.50, 2.25, 1.50,
1.00,...
## $ sqft_living   <int> 1180, 2570, 770, 1960, 1680, 5420, 1715, 1060, 1780,
...
## $ sqft_lot      <int> 5650, 7242, 10000, 5000, 8080, 101930, 6819, 9711,
74...
## $ floors        <dbl> 1.0, 2.0, 1.0, 1.0, 1.0, 1.0, 2.0, 1.0, 1.0, 2.0,
1.0...
## $ waterfront    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,...
## $ view          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0,
0,...
## $ condition     <int> 3, 3, 3, 5, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, 3,
4,...
## $ grade         <int> 7, 7, 6, 7, 8, 11, 7, 7, 7, 7, 8, 7, 7, 7, 7, 9, 7,
7...
## $ sqft_above    <int> 1180, 2170, 770, 1050, 1680, 3890, 1715, 1060, 1050,
...
## $ sqft_basement <int> 0, 400, 0, 910, 0, 1530, 0, 0, 730, 0, 1700, 300, 0,
...
## $ yr_built      <int> 1955, 1951, 1933, 1965, 1987, 2001, 1995, 1963,
1960,...
## $ yr_renovated  <int> 0, 1991, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,...
## $ zipcode       <int> 98178, 98125, 98028, 98136, 98074, 98053, 98003,
9819...
## $ lat           <dbl> 47.5112, 47.7210, 47.7379, 47.5208, 47.6168,
47.6561,...
## $ long          <dbl> -122.257, -122.319, -122.233, -122.393, -122.045, -
12...
## $ sqft_living15 <int> 1340, 1690, 2720, 1360, 1800, 4760, 2238, 1650,
1780,...
## $ sqft_lot15    <int> 5650, 7639, 8062, 5000, 7503, 101930, 6819, 9711,
811...
```

Checking for missing values

Changed the code to get a boolean value indicating the presence of the missing values. Since the output is 'FALSE' it shows there are no missing values in the dataset

```
any(is.na(data))

## [1] FALSE
```

```r
colSums(is.na(data))
```

```
##            id          date         price      bedrooms     bathrooms 
##             0             0             0             0             0 
##   sqft_living      sqft_lot        floors    waterfront          view 
##             0             0             0             0             0 
##     condition         grade    sqft_above sqft_basement      yr_built 
##             0             0             0             0             0 
##  yr_renovated       zipcode           lat          long sqft_living15 
##             0             0             0             0             0 
##    sqft_lot15 
##             0
```

Descriptive statistics

Updated: Performed descriptive statistics on the whole datset

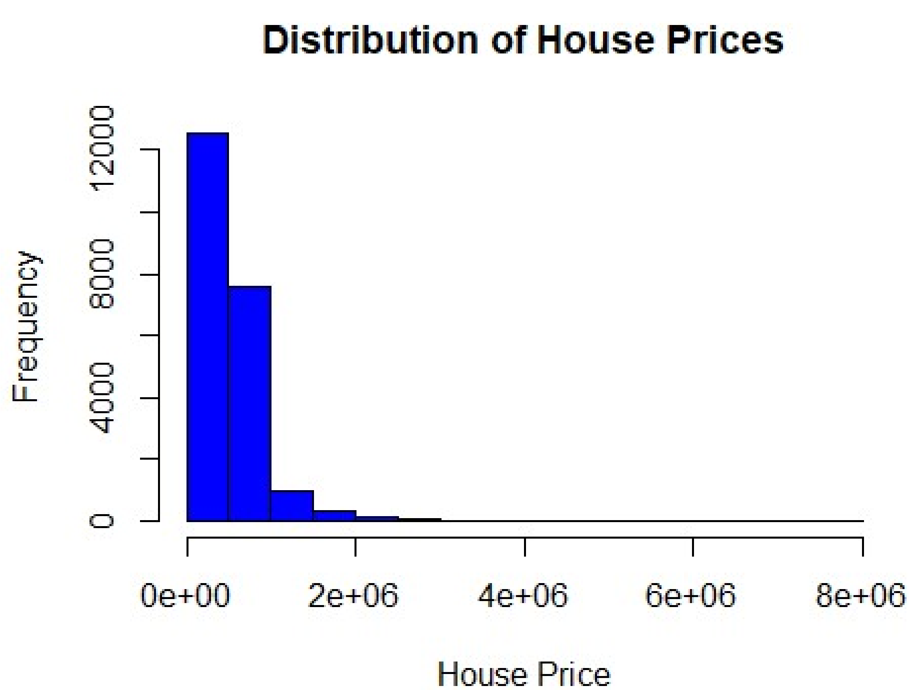
```r
#summary(data[3])
summary(data)
```

```
##        id                date               price            bedrooms     
##  Min.   :1.000e+06   Length:21613       Min.   :  75000   Min.   : 0.000  
##  1st Qu.:2.123e+09   Class :character   1st Qu.: 321950   1st Qu.: 3.000  
##  Median :3.905e+09   Mode  :character   Median : 450000   Median : 3.000  
##  Mean   :4.580e+09                      Mean   : 540088   Mean   : 3.371  
##  3rd Qu.:7.309e+09                      3rd Qu.: 645000   3rd Qu.: 4.000  
##  Max.   :9.900e+09                      Max.   :7700000   Max.   :33.000  
##    bathrooms      sqft_living       sqft_lot           floors     
##  Min.   :0.000   Min.   :  290   Min.   :    520   Min.   :1.000  
##  1st Qu.:1.750   1st Qu.: 1427   1st Qu.:   5040   1st Qu.:1.000  
##  Median :2.250   Median : 1910   Median :   7618   Median :1.500  
##  Mean   :2.115   Mean   : 2080   Mean   :  15107   Mean   :1.494  
##  3rd Qu.:2.500   3rd Qu.: 2550   3rd Qu.:  10688   3rd Qu.:2.000  
##  Max.   :8.000   Max.   :13540   Max.   :1651359   Max.   :3.500  
##    waterfront            view          condition         grade       
##  Min.   :0.000000   Min.   :0.0000   Min.   :1.000   Min.   : 1.000  
##  1st Qu.:0.000000   1st Qu.:0.0000   1st Qu.:3.000   1st Qu.: 7.000  
##  Median :0.000000   Median :0.0000   Median :3.000   Median : 7.000  
##  Mean   :0.007542   Mean   :0.2343   Mean   :3.409   Mean   : 7.657  
##  3rd Qu.:0.000000   3rd Qu.:0.0000   3rd Qu.:4.000   3rd Qu.: 8.000  
##  Max.   :1.000000   Max.   :4.0000   Max.   :5.000   Max.   :13.000  
##    sqft_above   sqft_basement       yr_built     yr_renovated   
##  Min.   : 290   Min.   :   0.0   Min.   :1900   Min.   :   0.0  
##  1st Qu.:1190   1st Qu.:   0.0   1st Qu.:1951   1st Qu.:   0.0  
##  Median :1560   Median :   0.0   Median :1975   Median :   0.0  
##  Mean   :1788   Mean   : 291.5   Mean   :1971   Mean   :  84.4  
##  3rd Qu.:2210   3rd Qu.: 560.0   3rd Qu.:1997   3rd Qu.:   0.0  
##  Max.   :9410   Max.   :4820.0   Max.   :2015   Max.   :2015.0  
##     zipcode           lat             long        sqft_living15 
```

```
##  Min.   :98001   Min.   :47.16   Min.   :-122.5   Min.   : 399
##  1st Qu.:98033   1st Qu.:47.47   1st Qu.:-122.3   1st Qu.:1490
##  Median :98065   Median :47.57   Median :-122.2   Median :1840
##  Mean   :98078   Mean   :47.56   Mean   :-122.2   Mean   :1987
##  3rd Qu.:98118   3rd Qu.:47.68   3rd Qu.:-122.1   3rd Qu.:2360
##  Max.   :98199   Max.   :47.78   Max.   :-121.3   Max.   :6210
##    sqft_lot15
##  Min.   :   651
##  1st Qu.:  5100
##  Median :  7620
##  Mean   : 12768
##  3rd Qu.: 10083
##  Max.   :871200
```

Basic Visualization of House Price

```
hist(data$price, col = "blue", xlab = "House Price",  main = "Distribution of
House Prices")
```



Quantitatively understanding the distribution

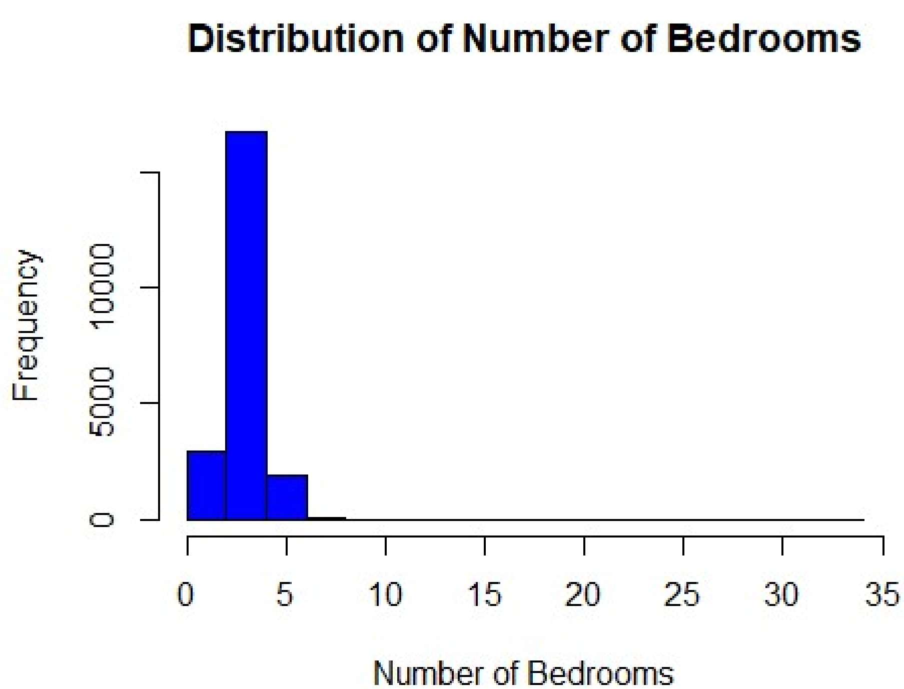
Understanding whether the house prices are normally distributed or is there any skewness. Given the positive values for both skewness and kurtosis, together they are telling
us that there is fat tailing towards the right side

```r
library(moments)
skewness(data$price)

## [1] 4.02379

kurtosis(data$price)

## [1] 37.57726
```

Basic Visualization of Number of bedrooms

Between the visualization by histogram and the table below clearly shows the distribution of the total number of houses by bed rooms. Added the table command to get the exact number which are otherwise
harder to read from the histogram.

```r
hist(data$bedrooms, col = "blue", xlab = "Number of Bedrooms",
     main = "Distribution of Number of Bedrooms")
```



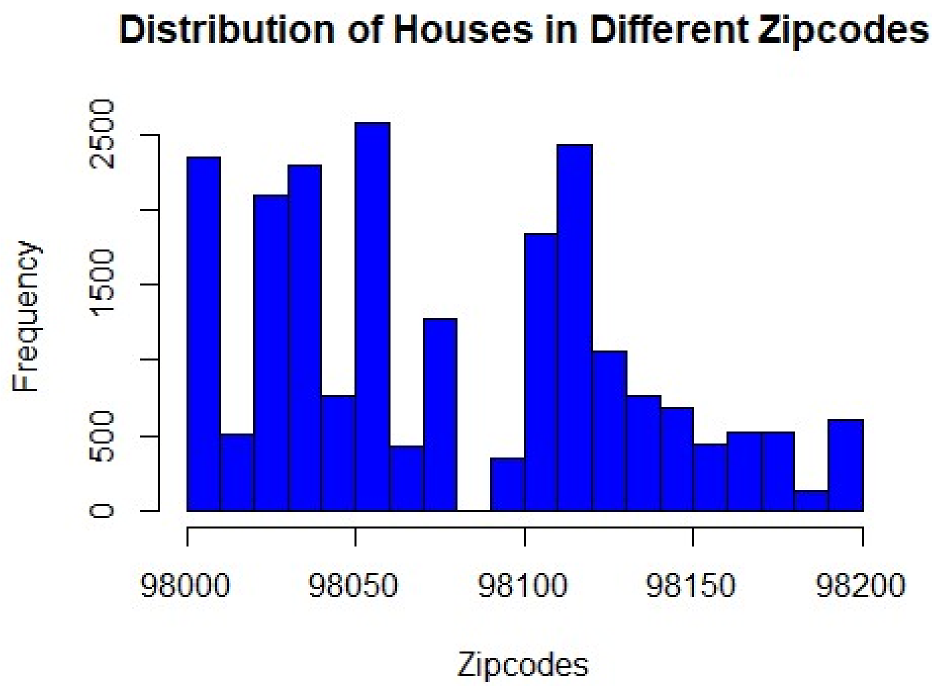

```r
table(data$bedrooms)

##
##    0    1    2    3    4    5    6    7    8    9   10   11   33
##   13  199 2760 9824 6882 1601  272   38   13    6    3    1    1
```

Basic Visualization of Number of Houses from each Zipcode

Between the visualization by histogram and the table below clearly shows the distribution of the overall zipcodes in the data

Added the table command to get the exact number which are otherwise harder to read from the histogram.

```
hist(data$zipcode, col = "blue", xlab = "Zipcodes",
     main = "Distribution of Houses in Different Zipcodes")
```




```
max(table(data$zipcode))

## [1] 602
```

Uncover New Information:

So far we have looked at the data, cleaned the data in terms of missing values, assigning the right data type for the variables etc. In order to uncover any information we need to look at the data by looking at the
relationships both visually and quantitatively. As a first step, I will perform some scatterplot and boxplot visualizations to
get a feel for the data Followed by that will do multivariate regression to undestand the

factors influencing
the house prices

- Handling the conversion of continous variables to factorial variables

```
data$bedrooms <- as.factor(data$bedrooms)
data$floors <- as.factor(data$floors)
data$yr_built <- as.factor(data$yr_built)
data$yr_renovated <- as.factor(data$yr_renovated)
data$grade <- as.factor(data$grade)
data$condition <- as.factor(data$condition)
data$zipcode <- as.factor(data$zipcode)
data$view <- as.factor(data$view)
data$waterfront <- as.factor(data$waterfront)
data$bathrooms <- as.factor(data$bathrooms)
```

View of the final version

```
glimpse(data)

## Rows: 21,613
## Columns: 21
## $ id            <dbl> 7129300520, 6414100192, 5631500400, 2487200875,
19544...
## $ date          <chr> "20141013T000000", "20141209T000000",
"20150225T00000...
## $ price         <dbl> 221900, 538000, 180000, 604000, 510000, 1225000,
2575...
## $ bedrooms      <fct> 3, 3, 2, 4, 3, 4, 3, 3, 3, 3, 3, 2, 3, 3, 5, 4, 3,
4,...
## $ bathrooms     <fct> 1, 2.25, 1, 3, 2, 4.5, 2.25, 1.5, 1, 2.5, 2.5, 1, 1,
...
## $ sqft_living   <int> 1180, 2570, 770, 1960, 1680, 5420, 1715, 1060, 1780,
...
## $ sqft_lot      <int> 5650, 7242, 10000, 5000, 8080, 101930, 6819, 9711,
74...
## $ floors        <fct> 1, 2, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1.5, 1, 1.5, 2,
2...
## $ waterfront    <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,...
## $ view          <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0,
0,...
## $ condition     <fct> 3, 3, 3, 5, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, 3,
4,...
## $ grade         <fct> 7, 7, 6, 7, 8, 11, 7, 7, 7, 7, 8, 7, 7, 7, 7, 9, 7,
7...
## $ sqft_above    <int> 1180, 2170, 770, 1050, 1680, 3890, 1715, 1060, 1050,
...
## $ sqft_basement <int> 0, 400, 0, 910, 0, 1530, 0, 0, 730, 0, 1700, 300, 0,
...
## $ yr_built      <fct> 1955, 1951, 1933, 1965, 1987, 2001, 1995, 1963,
```

```
1960,...
## $ yr_renovated  <fct> 0, 1991, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,...
## $ zipcode       <fct> 98178, 98125, 98028, 98136, 98074, 98053, 98003,
9819...
## $ lat           <dbl> 47.5112, 47.7210, 47.7379, 47.5208, 47.6168,
47.6561,...
## $ long          <dbl> -122.257, -122.319, -122.233, -122.393, -122.045, -
12...
## $ sqft_living15 <int> 1340, 1690, 2720, 1360, 1800, 4760, 2238, 1650,
1780,...
## $ sqft_lot15    <int> 5650, 7639, 8062, 5000, 7503, 101930, 6819, 9711,
811...

data$price <- as.integer(data$price)
data$price <- as.integer((data$price/1000))
```

Below are the few ways to look at the data to uncover some of the information

Plotting the data to numerically and visually uncover the information

Uncovering the relationships numerically

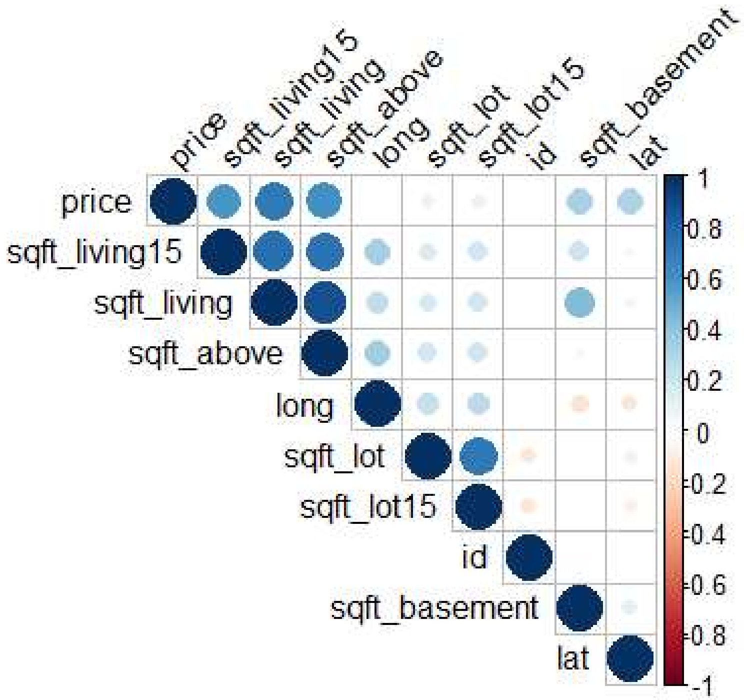
- As we can clearly see after filtering the continous variables, there is a very clear correlation between the sqft_living, sqft_above, squareft_living15, sqft_basement are correlated. Interestingly, sqft_lot does not have a bigger influence

```
nums <- Filter(is.numeric, data)
res <- cor(nums, method = "pearson", use = "complete.obs")
res

##                         id       price sqft_living    sqft_lot
sqft_above
## id             1.000000000 -0.01676686 -0.01225777 -0.13210870 -
0.0108421341
## price         -0.016766856  1.00000000  0.70201168  0.08967666
0.6055265590
## sqft_living   -0.012257765  0.70201168  1.00000000  0.17282566
0.8765965987
## sqft_lot      -0.132108702  0.08967666  0.17282566  1.00000000
0.1835122809
## sqft_above    -0.010842134  0.60552656  0.87659660  0.18351228
1.0000000000
## sqft_basement -0.005151125  0.32384373  0.43504297  0.01528620 -
0.0519433068
## lat           -0.001890932  0.30705846  0.05252946 -0.08568279 -
0.0008164986
## long           0.020798586  0.02159939  0.24022330  0.22952086
```

```
0.3438030175
## sqft_living15 -0.002901004  0.58535305  0.75642026  0.14460817
0.7318702924
## sqft_lot15    -0.138797866  0.08246195  0.18328555  0.71855675
0.1940498619
##                sqft_basement           lat        long sqft_living15
sqft_lot15
## id              -0.005151125 -0.0018909324  0.02079859  -0.002901004 -
0.13879787
## price            0.323843735  0.3070584593  0.02159939   0.585353050
0.08246195
## sqft_living      0.435042974  0.0525294622  0.24022330   0.756420259
0.18328555
## sqft_lot         0.015286202 -0.0856827882  0.22952086   0.144608174
0.71855675
## sqft_above      -0.051943307 -0.0008164986  0.34380302   0.731870292
0.19404986
## sqft_basement    1.000000000  0.1105379580 -0.14476477   0.200354983
0.01727618
## lat              0.110537958  1.0000000000 -0.13551178   0.048857932 -
0.08641881
## long            -0.144764774 -0.1355117836  1.00000000   0.334604984
0.25445129
## sqft_living15    0.200354983  0.0488579321  0.33460498   1.000000000
0.18319175
## sqft_lot15       0.017276181 -0.0864188072  0.25445129   0.183191749
1.00000000
```

Visualizing the correlation plots

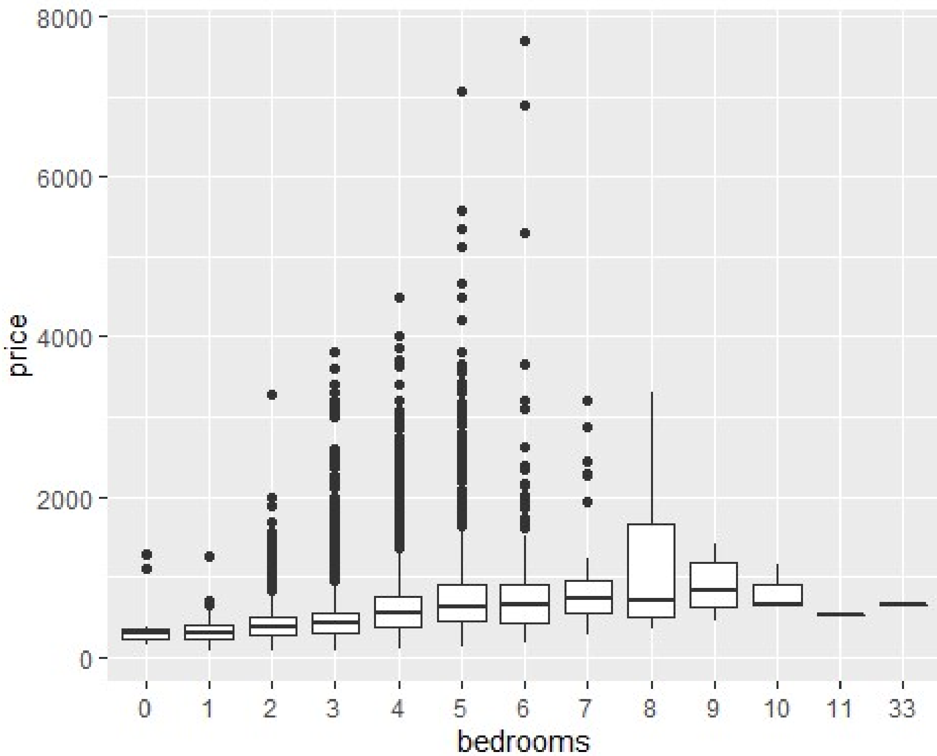
```r
library(corrplot)

## corrplot 0.84 loaded

corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



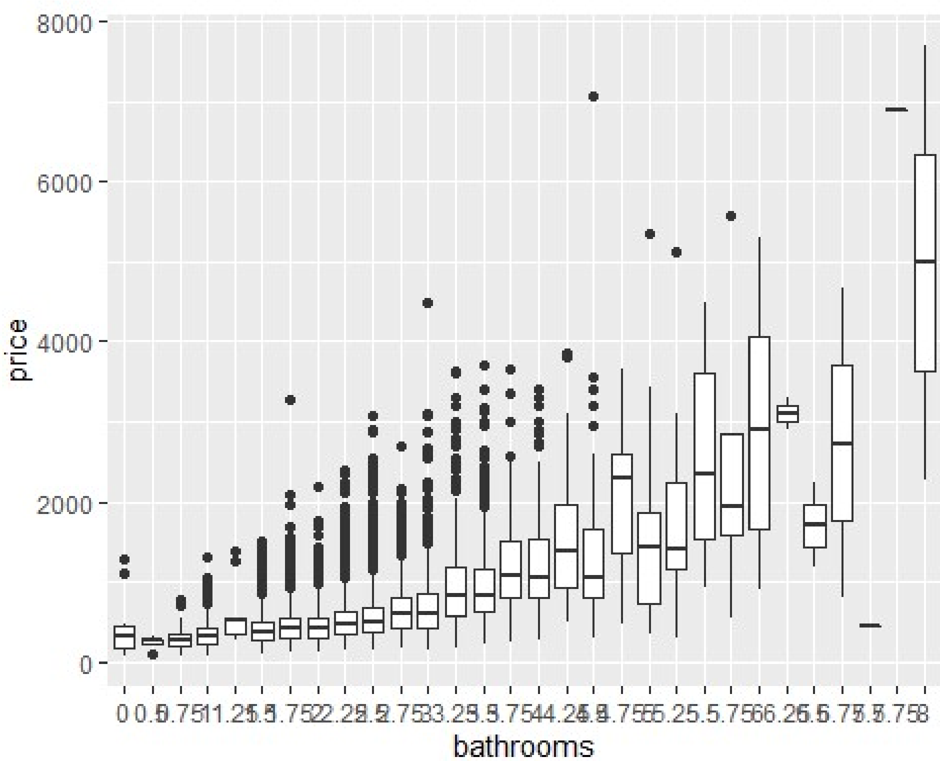
Visualizing the relationship trends that effect the price

Price versus Number of bedrooms

```
ggplot(data = data, aes(x = bedrooms, y = price)) + geom_boxplot()
```

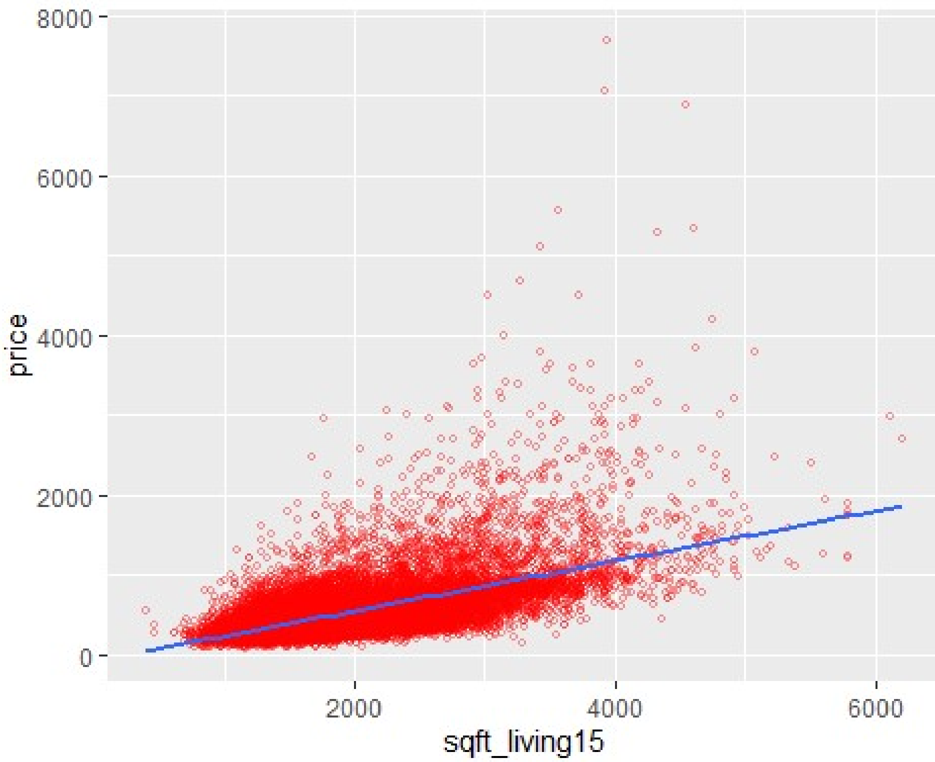



```
ggplot(data = data, aes(x = bathrooms, y = price)) + geom_boxplot()
```



Sqft_living versus Price

```
p1 <- ggplot(data = data, aes(x = sqft_living15 , y = price)) +
geom_point(position = "jitter", size = 1, shape = 1, alpha = 0.4, col =
"red") + geom_smooth(method = "lm", se = FALSE)
p1

## `geom_smooth()` using formula 'y ~ x'
```

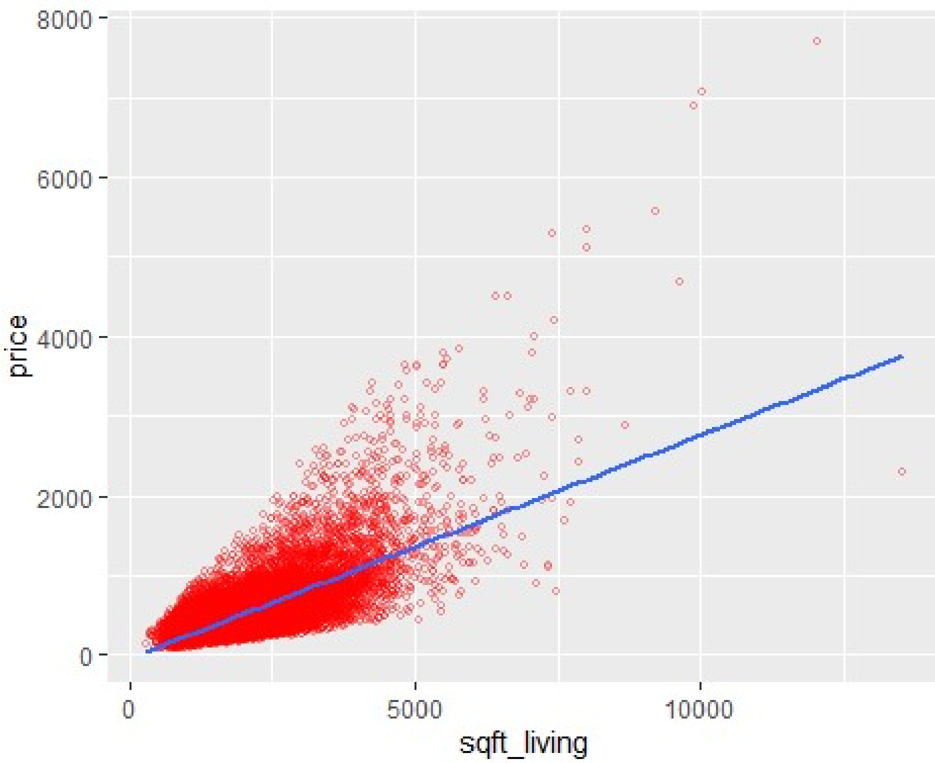



```
summary(p1)

## data: id, date, price, bedrooms, bathrooms, sqft_living, sqft_lot,
##   floors, waterfront, view, condition, grade, sqft_above,
##   sqft_basement, yr_built, yr_renovated, zipcode, lat, long,
##   sqft_living15, sqft_lot15 [21613x21]
## mapping:  x = ~sqft_living15, y = ~price
## faceting: <ggproto object: Class FacetNull, Facet, gg>
##     compute_layout: function
##     draw_back: function
##     draw_front: function
##     draw_labels: function
##     draw_panels: function
##     finish_data: function
##     init_scales: function
##     map_data: function
##     params: list
##     setup_data: function
##     setup_params: function
##     shrink: TRUE
```

```
##     train_scales: function
##     vars: function
##     super:  <ggproto object: Class FacetNull, Facet, gg>
## -----------------------------------
## geom_point: na.rm = FALSE, size = 1, shape = 1, alpha = 0.4, colour = red
## stat_identity: na.rm = FALSE
## position_jitter
##
## geom_smooth: na.rm = FALSE, orientation = NA, se = FALSE, flipped_aes =
FALSE
## stat_smooth: na.rm = FALSE, orientation = NA, se = FALSE, method = lm
## position_identity
```

Sqft_living versus Price

```
ggplot(data = data, aes(x = sqft_living , y = price)) + geom_point(position =
"jitter", size = 1, shape = 1, alpha = 0.4, col = "red") + geom_smooth(method
= "lm", se = FALSE)

## `geom_smooth()` using formula 'y ~ x'
```

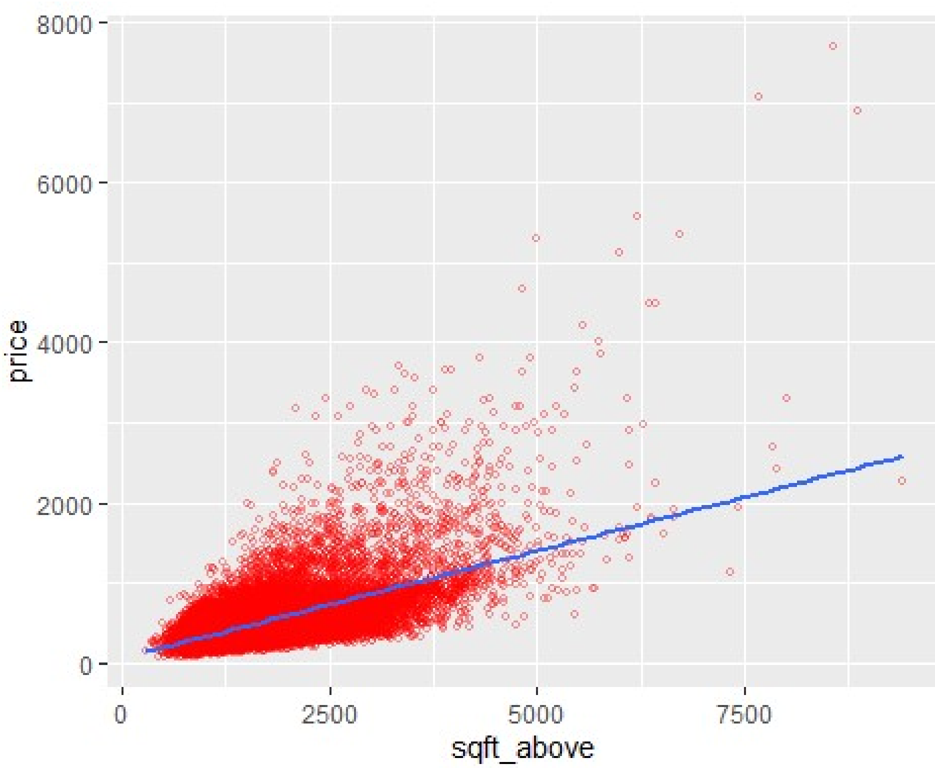



```
lm(formula = price ~ sqft_living, data = data) %>% summary()

##
## Call:
## lm(formula = price ~ sqft_living, data = data)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -1476.0  -147.5   -24.1   106.3  4362.1
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -43.737341   4.403125  -9.933   <2e-16 ***
## sqft_living   0.280633   0.001937 144.911   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 261.5 on 21611 degrees of freedom
## Multiple R-squared:  0.4928, Adjusted R-squared:  0.4928
## F-statistic: 2.1e+04 on 1 and 21611 DF,  p-value: < 2.2e-16
```

Sqft_above versus Price

```
ggplot(data = data, aes(x = sqft_above , y = price)) + geom_point(position =
"jitter", size = 1, shape = 1, alpha = 0.4, col = "red") + geom_smooth(method
= "lm", se = FALSE)

## `geom_smooth()` using formula 'y ~ x'
```

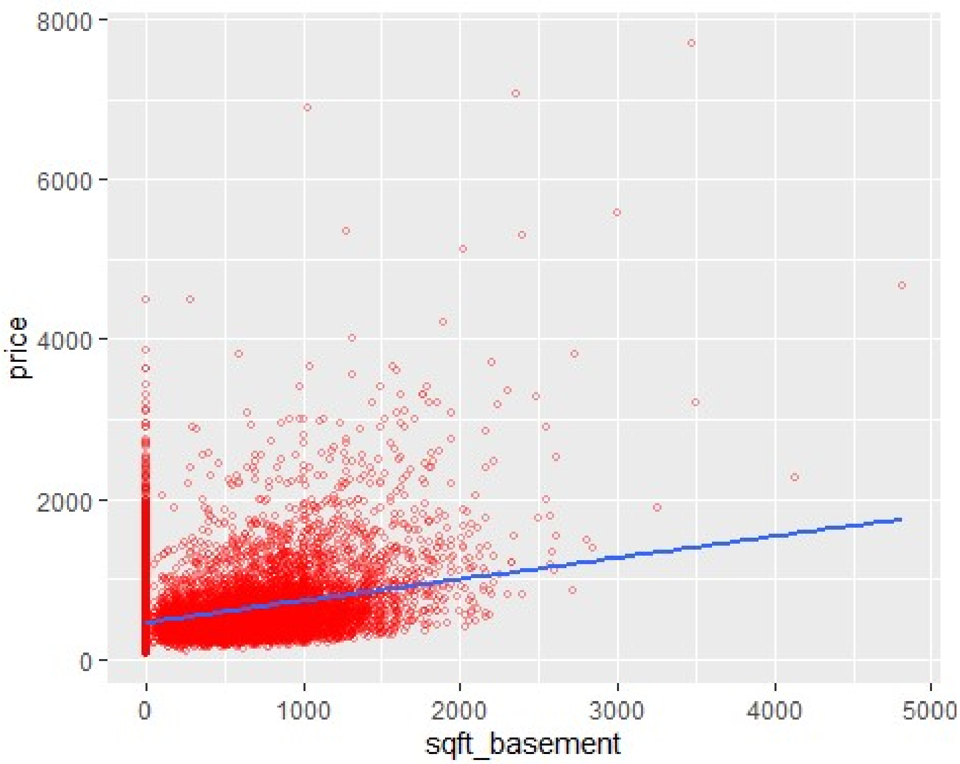



```
lm(formula = price ~ sqft_above, data = data) %>% summary()

##
## Call:
## lm(formula = price ~ sqft_above, data = data)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max 
## -913.0 -165.7  -41.4  109.3 5339.4 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  59.8163     4.7303   12.64   <2e-16 ***
## sqft_above    0.2685     0.0024  111.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 292.2 on 21611 degrees of freedom
## Multiple R-squared:  0.3667, Adjusted R-squared:  0.3666 
## F-statistic: 1.251e+04 on 1 and 21611 DF,  p-value: < 2.2e-16
```

Sqft_basement versus Price

```
ggplot(data = data, aes(x = sqft_basement , y = price)) + geom_point(position
= "jitter", size = 1, shape = 1, alpha = 0.4, col = "red") +
geom_smooth(method = "lm", se = FALSE)

## `geom_smooth()` using formula 'y ~ x'
```



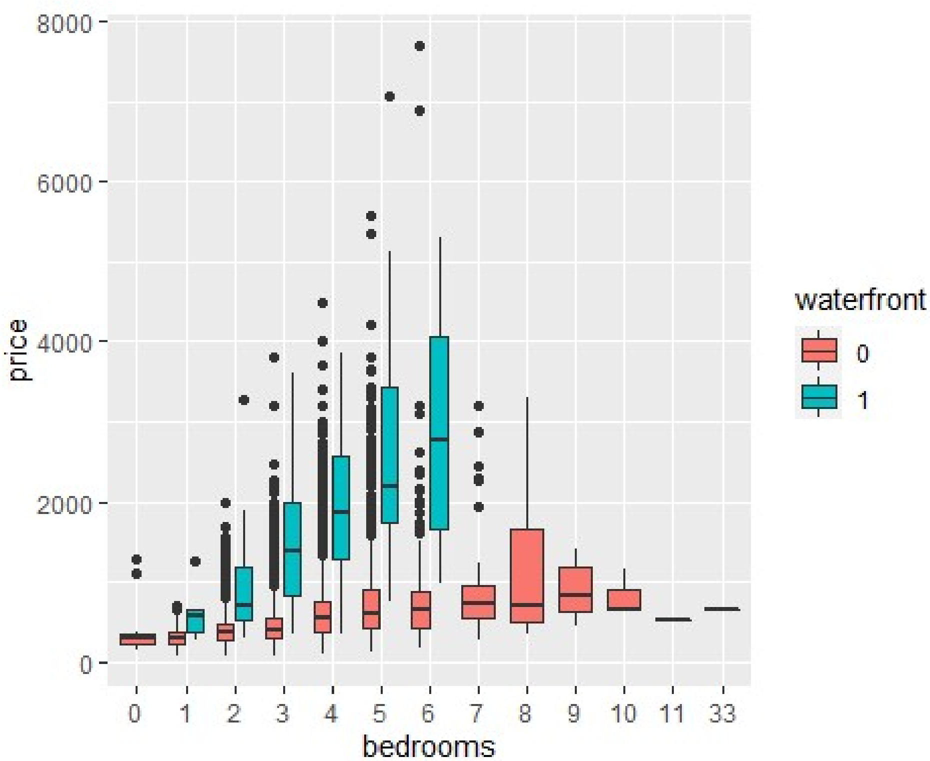

```
lm(formula = price ~ sqft_basement, data = data) %>% summary()

## 
## Call:
## lm(formula = price ~ sqft_basement, data = data)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -603.0 -197.6   -77.2  103.4 6303.4
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.616e+02  2.829e+00  163.16   <2e-16 ***
## sqft_basement 2.687e-01  5.339e-03   50.32   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 347.4 on 21611 degrees of freedom
## Multiple R-squared:  0.1049, Adjusted R-squared:  0.1048
## F-statistic:  2532 on 1 and 21611 DF,  p-value: < 2.2e-16
```

Uncovering the information by combination of variables

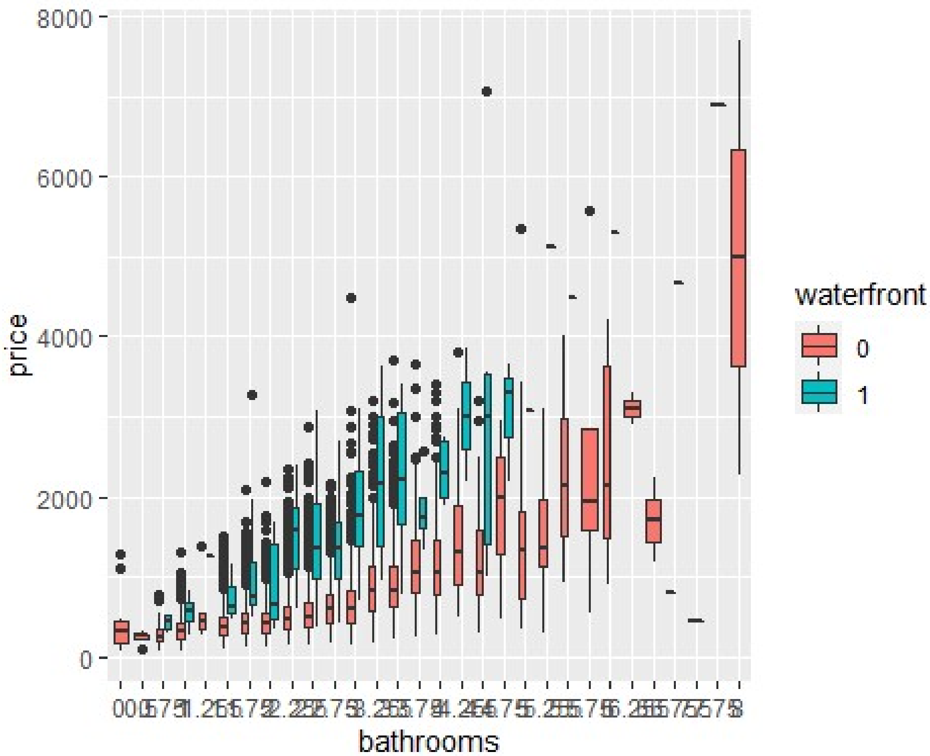
Adding a third variable like waterfront. As we can see in addition to number of bedrooms there is a strong interaction between bedrooms and waterfront with and without waterfront view. As long as there is waterfront view the house prices are higher

```
ggplot(data = data, aes(x = bedrooms, y = price, fill = waterfront)) +
geom_boxplot()
```



- Simalarly having bathrooms has the same effect.i.e. number of bathrooms and waterfront together has influence on the house price

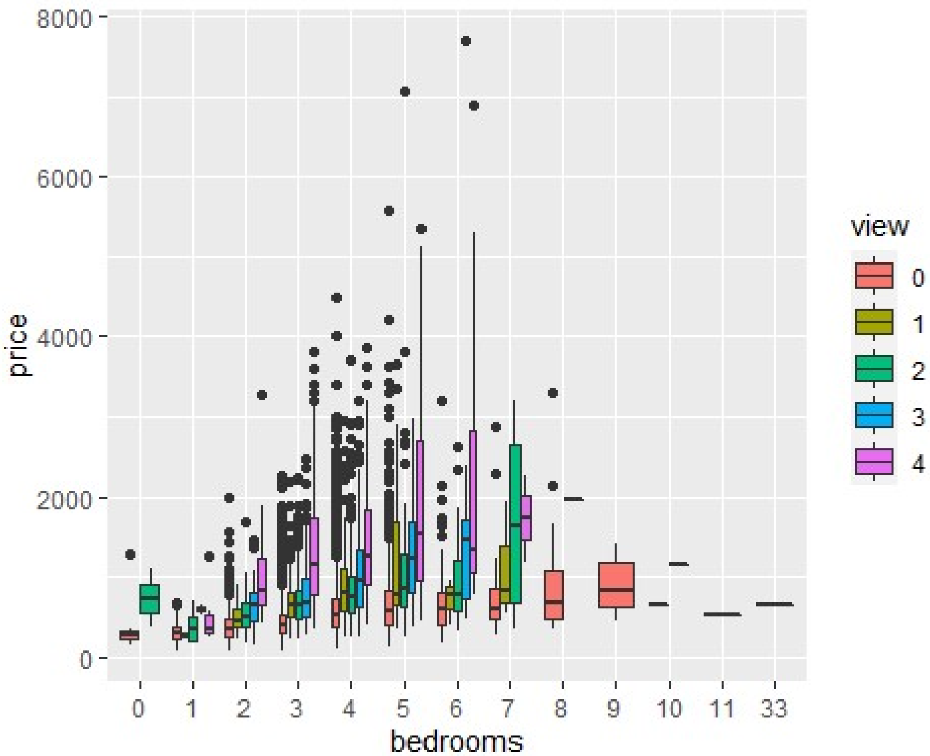
```
ggplot(data = data, aes(x = bathrooms, y = price, fill = waterfront)) +
geom_boxplot()
```



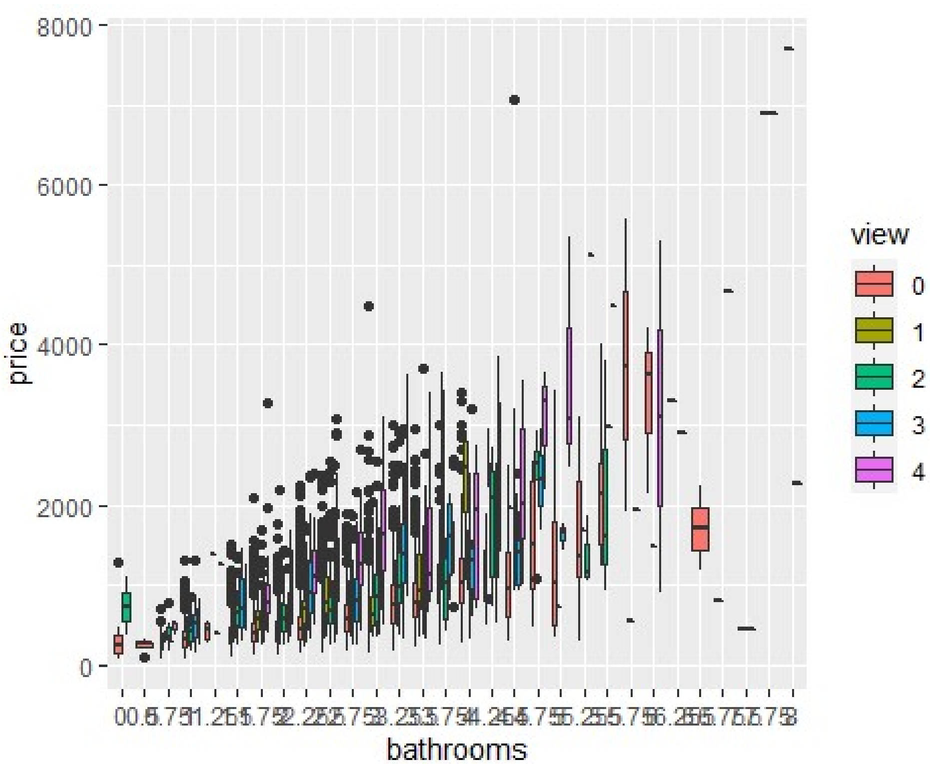
- Similarly looking at the effect of a view in determining the price of the house depending on the number of times the house has been viewed

- In general, as the number of view increases the house prices seems to be increasing

```
ggplot(data = data, aes(x = bedrooms, y = price, fill = view)) +
geom_boxplot()
```

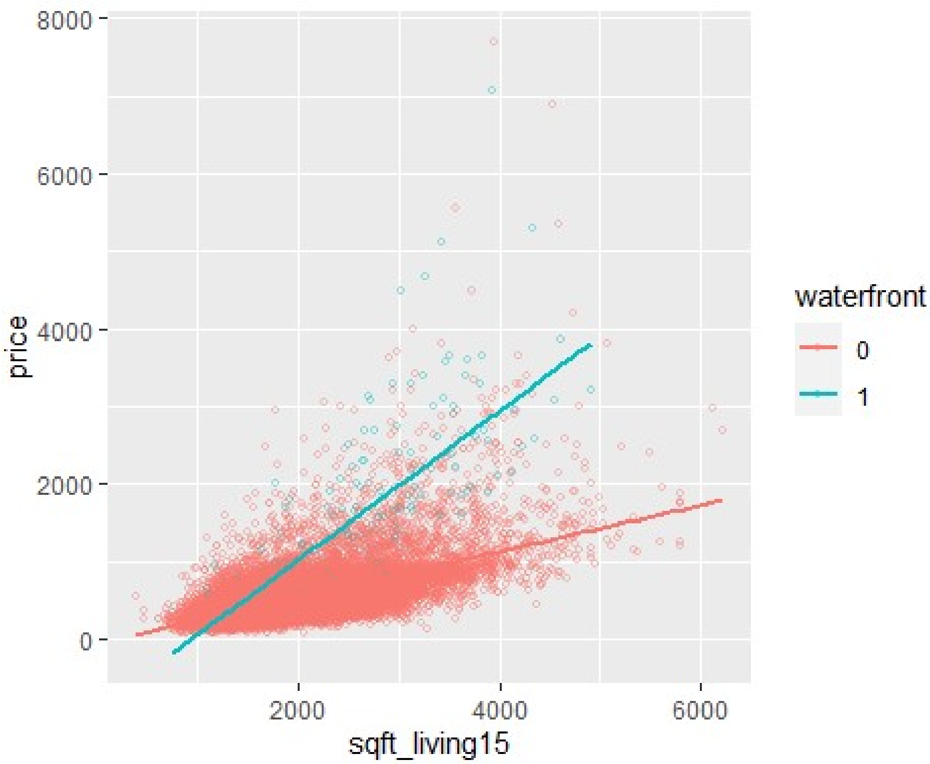



```r
ggplot(data = data, aes(x = bathrooms, y = price, fill = view)) +
geom_boxplot()
```



- Here we are looking at the effect of having waterfront on the house prices. As we can see the slope of the line is more steeper indicating the influence of having a waterfront along with sqft of living from 2015

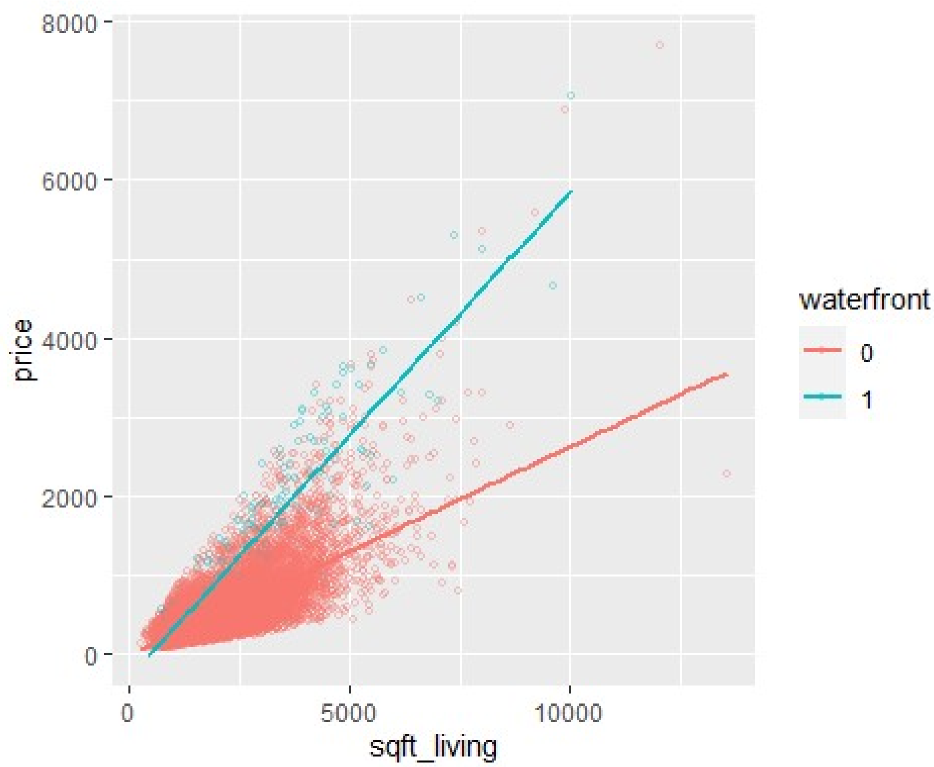
```
ggplot(data = data, aes(x = sqft_living15 , y = price, col = waterfront)) +
geom_point(position = "jitter", size = 1, shape = 1, alpha = 0.4) +
geom_smooth(method = "lm", se = FALSE)

## `geom_smooth()` using formula 'y ~ x'
```



Sqft_living versus Price

- Here we are looking at the effect of having waterfront on the house prices. As we can see the slope of the line is more steeper indicating the influence of having a waterfront along with sqft of living

```
ggplot(data = data, aes(x = sqft_living , y = price, col = waterfront)) +
geom_point(position = "jitter", size = 1, shape = 1, alpha = 0.4) +
geom_smooth(method = "lm", se = FALSE)

## `geom_smooth()` using formula 'y ~ x'
```



Slicing and Dicing, Learn Packages

- In the current data set ther is no need for actually to slice and dice the data. Currently I am interested in only understanding the factors effecting the house price. At this point dataset is very clean does not need any kind of slicing or dicing. Its a sinle data set and does not require any joing of data. Coming to learning packages, at this I am good at “dplyr”, “ggplot” are enough for this project.

Summarizing the data

- The key question is what are the factors influencing the house prices

From the visualizations and linear fits it is evident that:

- Sqft_living, sqft_above are the top two factors with highest correlation to the price as seen from the correlation matrix

- Coming to the categorical factors as we saw in the box plots, price has a clear correlation to the number of bedrooms and bathrooms. In addition, while we control other factors, number of views and waterfront has a positive impact on the house prices

- However, all the above conclusions are by looking at the each variables individually and hecne in the following code, I am going build a linear predictive model to dtermine the price of the house

```r
lm1 <- lm(data$price ~ sqft_living + sqft_lot + sqft_basement + waterfront +
view + grade + zipcode + condition + sqft_living:waterfront +
sqft_living:view + sqft_lot:waterfront, data = data)
summary(lm1)
```

```
##
## Call:
## lm(formula = data$price ~ sqft_living + sqft_lot + sqft_basement +
##     waterfront + view + grade + zipcode + condition +
sqft_living:waterfront +
##     sqft_living:view + sqft_lot:waterfront, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1484.71   -58.88     0.07    52.90  2770.65
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -7.027e+01  1.449e+02  -0.485 0.627802
## sqft_living                1.544e-01  2.123e-03  72.708  < 2e-16 ***
## sqft_lot                   2.359e-04  2.607e-05   9.051  < 2e-16 ***
## sqft_basement             -4.310e-02  2.894e-03 -14.893  < 2e-16 ***
## waterfront1               -3.409e+02  3.219e+01 -10.591  < 2e-16 ***
## view1                     -7.743e+01  2.147e+01  -3.606 0.000311 ***
## view2                      2.318e+01  1.239e+01   1.870 0.061449 .
## view3                     -3.033e+01  1.795e+01  -1.689 0.091160 .
## view4                      2.306e+02  2.427e+01   9.502  < 2e-16 ***
## grade3                     5.931e+01  1.692e+02   0.350 0.725974
## grade4                     8.192e+00  1.495e+02   0.055 0.956293
## grade5                    -3.418e+01  1.474e+02  -0.232 0.816603
## grade6                    -4.476e+01  1.473e+02  -0.304 0.761184
## grade7                    -4.034e+01  1.473e+02  -0.274 0.784216
## grade8                    -1.431e+01  1.473e+02  -0.097 0.922637
## grade9                     6.437e+01  1.474e+02   0.437 0.662312
## grade10                    1.888e+02  1.475e+02   1.280 0.200619
## grade11                    3.558e+02  1.477e+02   2.409 0.015987 *
## grade12                    6.766e+02  1.484e+02   4.558 5.19e-06 ***
## grade13                    1.868e+03  1.535e+02  12.168  < 2e-16 ***
## zipcode98002               2.466e+00  1.274e+01   0.194 0.846448
## zipcode98003              -1.106e+00  1.147e+01  -0.096 0.923182
## zipcode98004               7.840e+02  1.120e+01  69.987  < 2e-16 ***
## zipcode98005               3.343e+02  1.354e+01  24.687  < 2e-16 ***
## zipcode98006               2.658e+02  1.012e+01  26.256  < 2e-16 ***
## zipcode98007               2.602e+02  1.432e+01  18.167  < 2e-16 ***
## zipcode98008               2.587e+02  1.146e+01  22.571  < 2e-16 ***
## zipcode98010               6.288e+01  1.630e+01   3.857 0.000115 ***
## zipcode98011               1.475e+02  1.280e+01  11.527  < 2e-16 ***
## zipcode98014               9.520e+01  1.514e+01   6.286 3.33e-10 ***
## zipcode98019               9.927e+01  1.292e+01   7.681 1.64e-14 ***
## zipcode98022              -3.373e+00  1.225e+01  -0.275 0.783093
```

```
## zipcode98023             -2.248e+01  9.955e+00  -2.258 0.023948 *  
## zipcode98024              1.626e+02  1.793e+01   9.067  < 2e-16 ***
## zipcode98027              1.727e+02  1.043e+01  16.557  < 2e-16 ***
## zipcode98028              1.404e+02  1.143e+01  12.284  < 2e-16 ***
## zipcode98029              2.250e+02  1.112e+01  20.243  < 2e-16 ***
## zipcode98030              9.718e+00  1.175e+01   0.827 0.408154    
## zipcode98031              1.863e+01  1.153e+01   1.616 0.106137    
## zipcode98032              2.114e+00  1.494e+01   0.141 0.887480    
## zipcode98033              3.664e+02  1.030e+01  35.572  < 2e-16 ***
## zipcode98034              2.048e+02  9.769e+00  20.966  < 2e-16 ***
## zipcode98038              3.850e+01  9.626e+00   4.000 6.37e-05 ***
## zipcode98039              1.274e+03  2.196e+01  58.014  < 2e-16 ***
## zipcode98040              5.038e+02  1.167e+01  43.174  < 2e-16 ***
## zipcode98042              7.363e+00  9.753e+00   0.755 0.450304    
## zipcode98045              9.573e+01  1.233e+01   7.763 8.65e-15 ***
## zipcode98052              2.516e+02  9.717e+00  25.890  < 2e-16 ***
## zipcode98053              2.243e+02  1.049e+01  21.372  < 2e-16 ***
## zipcode98055              4.816e+01  1.160e+01   4.152 3.31e-05 ***
## zipcode98056              8.924e+01  1.044e+01   8.547  < 2e-16 ***
## zipcode98058              3.898e+01  1.015e+01   3.842 0.000123 ***
## zipcode98059              9.054e+01  1.011e+01   8.952  < 2e-16 ***
## zipcode98065              1.015e+02  1.119e+01   9.068  < 2e-16 ***
## zipcode98070              9.376e+01  1.587e+01   5.907 3.54e-09 ***
## zipcode98072              1.760e+02  1.157e+01  15.209  < 2e-16 ***
## zipcode98074              1.931e+02  1.032e+01  18.708  < 2e-16 ***
## zipcode98075              1.847e+02  1.090e+01  16.947  < 2e-16 ***
## zipcode98077              1.386e+02  1.286e+01  10.776  < 2e-16 ***
## zipcode98092             -2.285e+01  1.081e+01  -2.114 0.034512 *  
## zipcode98102              4.953e+02  1.603e+01  30.896  < 2e-16 ***
## zipcode98103              3.427e+02  9.611e+00  35.655  < 2e-16 ***
## zipcode98105              4.861e+02  1.220e+01  39.840  < 2e-16 ***
## zipcode98106              1.254e+02  1.097e+01  11.423  < 2e-16 ***
## zipcode98107              3.470e+02  1.167e+01  29.741  < 2e-16 ***
## zipcode98108              1.242e+02  1.301e+01   9.549  < 2e-16 ***
## zipcode98109              5.239e+02  1.578e+01  33.196  < 2e-16 ***
## zipcode98112              6.555e+02  1.169e+01  56.070  < 2e-16 ***
## zipcode98115              3.461e+02  9.669e+00  35.793  < 2e-16 ***
## zipcode98116              3.032e+02  1.104e+01  27.465  < 2e-16 ***
## zipcode98117              3.260e+02  9.768e+00  33.373  < 2e-16 ***
## zipcode98118              1.679e+02  9.963e+00  16.857  < 2e-16 ***
## zipcode98119              5.055e+02  1.311e+01  38.565  < 2e-16 ***
## zipcode98122              3.571e+02  1.140e+01  31.332  < 2e-16 ***
## zipcode98125              2.089e+02  1.040e+01  20.090  < 2e-16 ***
## zipcode98126              1.970e+02  1.083e+01  18.195  < 2e-16 ***
## zipcode98133              1.637e+02  9.977e+00  16.406  < 2e-16 ***
## zipcode98136              2.661e+02  1.175e+01  22.656  < 2e-16 ***
## zipcode98144              2.824e+02  1.089e+01  25.936  < 2e-16 ***
## zipcode98146              1.081e+02  1.142e+01   9.459  < 2e-16 ***
## zipcode98148              7.638e+01  2.053e+01   3.720 0.000199 ***
## zipcode98155              1.476e+02  1.020e+01  14.474  < 2e-16 ***
```

```
## zipcode98166             7.121e+01  1.184e+01   6.015 1.82e-09 ***
## zipcode98168             5.362e+01  1.167e+01   4.594 4.36e-06 ***
## zipcode98177             2.331e+02  1.186e+01  19.658  < 2e-16 ***
## zipcode98178             4.669e+01  1.175e+01   3.973 7.13e-05 ***
## zipcode98188             3.633e+01  1.448e+01   2.508 0.012133 *
## zipcode98198             1.214e+01  1.149e+01   1.057 0.290651
## zipcode98199             4.035e+02  1.118e+01  36.090  < 2e-16 ***
## condition2               5.110e+01  2.906e+01   1.758 0.078710 .
## condition3               6.818e+01  2.704e+01   2.521 0.011699 *
## condition4               9.783e+01  2.707e+01   3.614 0.000302 ***
## condition5               1.462e+02  2.722e+01   5.372 7.88e-08 ***
## sqft_living:waterfront1  3.005e-01  8.769e-03  34.272  < 2e-16 ***
## sqft_living:view1        7.128e-02  7.786e-03   9.155  < 2e-16 ***
## sqft_living:view2        2.432e-02  4.391e-03   5.538 3.10e-08 ***
## sqft_living:view3        7.071e-02  5.624e-03  12.572  < 2e-16 ***
## sqft_living:view4        2.469e-02  6.695e-03   3.688 0.000227 ***
## sqft_lot:waterfront1    -8.505e-04  2.676e-04  -3.178 0.001484 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 143.8 on 21514 degrees of freedom
## Multiple R-squared:  0.8472, Adjusted R-squared:  0.8465
## F-statistic:  1218 on 98 and 21514 DF,  p-value: < 2.2e-16
```

- The above model has a good R square value and Adjusted R square Value.
- ~ 85% of the variability in price is explained by the sqft_living, sqft_basement,have waterfront or not, and the number of views the house has since listed, zipcode, condition and grade of the house

Tables and Plots And Things to Learn

. Expand my knowledge on logistic regression model to predict the probability of price of house . Scatterrplots and Box plots were used to explore and uncover the data. . While trying to plot bivariate analysis the figures are very crowded, trying to find a way to fix this issue . I am not comfortable for changing the price values from 1000's to k's to plot as variable on the y-axis. . I still need to learn how to bin my x-axis values as the x-axis looks very crowded when I try to plot the price against years_built, years_renovated.

Build a machine learning model

Currently I used multivariate linear regression. However, I did not really split the data into train and test to assess the predictions from machine learning model. Similraly I want to explore, if I can apply any other models that I will learn in the rest of the course.

#1

I am working on the dataset which has the house prices of King County, USA. There were ~21000 records and 21 variables of data. The main goal was to identify the variables that are accountable for the prediction of house prices. So, the dataset was downloaded from the kaggles website and cleaned for missing values and then dataset handled to correct the data type of all the independent variables. Latter performed data visualization to see the distribution of variables through scatter plots or histograms to understand the distribution of data within each variable or the relationships between the variables. Finally, to address the prediction variables the cleaned and processed data is fed into multiple linear regression analysis. Based on analysis, by looking at the significant variables in the data the most influencing predictor variables on price were determined.

#2

The problem statement addressed in the analysis is "What are the key factors in predicting a house price in the King County USA?" There are around 21 variables that provides information such as - when the house was sold, number of bedrooms, number of bathrooms, how many floors, does it have a waterfront, what is the house condition, what is the year built and renovated , does it have a basement, square footage of the lot size both interior and exterior. At the end of the analysis the user using the model should be able to feed the relevant factors as input and get an accurate estimation of the house price.

#3

I need to pick most affecting predictors among the 21 variables. To achieve this, I used a multiple linear regression model. It fits well for the purpose as there are multiple independent variables, I need to reduce the number of dimensions I am interested in the prediction of the house price. At first, I included all the variables for the analysis. Further the variables are fine tuned bsed on the outcome of the regression model. I was mainly considering the significance level of the p value associated with each of the predictor variables that were fed into the model. Later in the iterative process I removed one variable at a time and refitted the model. Finally, I also included the interaction terms inorder to identify if any of the factors together influence the outcome of the house price in the model.

#4

The key variables that are key in predicting the price of the house in the King, County, UAS are clearly identified by the model. Out of the 21 variables from the dataset, the model selected zipcode, sqft_basement, sqft_living, have waterfront or not, and the number of views the house has since listed, condition and grade of the house as the key indicators of the house prices. From 21 variables going to 7 varaibles is a great drop in the number of variables. Given the R-Ssquare value of 0.85 and with the selected variables, the model is able explain about 85% of the variability in the house price prediction.

#5

Regarding the effects on the target user, the model has a huge value in terms of predicting the house price. As an end user who is interested in buying a house in the King County, USA now has a pretty good idea of what are the factors that will influence the price of the house. This will help both the seller and buyer. Based on the coefficents generated in the model, the end user now clearly can see by what factor a house price will change if one of the above listed variables are changed by a unit. Given the coefficients from the linear model the model can identify within those factors which are key so that the end users can prioritize depending on the end price.

#6
In my view, I think I could optimize the model furthermore by reducing the number of categorical variables. For example, if I incorporate the categorical variable zipcode into the model as we can see it has too many levels. And I realized this is one of the major issues while dealing with categorical varaibles whenever there are too many levels. The linear model is going to generate coefficients for each of the individual categorical variable and as we know this can get out of control soon. So, I was exploring different options on how to reduce the dimensionality of the data and came across several techniques online and I found clustering is one of the major technique that can be used to group categorical variables, either based on abundance, colinearity, number of missing values or by ranking. These are the techniques I would like to explore and retune my model so that the model is more sophisticated.