MTH 404 R Project

You Liang

2023-04-11

This is just a outline and necessary steps and code of the R project. Students should add more explanations. For example, you should add more descriptions about the data visualization. More importantly, you need to write out the final regression and explain some coefficients as we did in the class.

DATA

We collected the data from "kaggle datasets" named as "KC_Housesales_Data". The link of the data: https://www.kaggle.com/swathiachath/kc-housesales-data

Online property companies offer valuations of houses using machine learning techniques. The aim of this report is to predict the house sales in King County, Washington State, USA using Multiple Linear Regression (MLR). The dataset consisted of historic data of houses sold between May 2014 to May 2015.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(lubridate)
```

```
## Loading required package: timechange
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(readr)
library(caTools)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(leaps)
```

```
mydata = read_csv("kc_house_data.csv")
```

```
## Rows: 21597 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): date
## dbl (20): id, price, bedrooms, bathrooms, sqft_living, sqft_lot, floors, wat...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(mydata)
```

```
## # A tibble: 6 x 21
##           id date     price bedro~1 bathr~2 sqft_~3 sqft_~4 floors water~5  view
##        <dbl> <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl> <dbl>
## 1 7129300520 10/13/~ 2.22e5       3    1       1180    5650      1       0     0
## 2 6414100192 12/9/2~ 5.38e5       3    2.25    2570    7242      2       0     0
## 3 5631500400 2/25/2~ 1.8 e5       2    1        770   10000      1       0     0
## 4 2487200875 12/9/2~ 6.04e5       4    3       1960    5000      1       0     0
## 5 1954400510 2/18/2~ 5.1 e5       3    2       1680    8080      1       0     0
## 6 7237550310 5/12/2~ 1.23e6       4    4.5     5420  101930      1       0     0
## # ... with 11 more variables: condition <dbl>, grade <dbl>, sqft_above <dbl>,
## #   sqft_basement <dbl>, yr_built <dbl>, yr_renovated <dbl>, zipcode <dbl>,
## #   lat <dbl>, long <dbl>, sqft_living15 <dbl>, sqft_lot15 <dbl>, and
## #   abbreviated variable names 1: bedrooms, 2: bathrooms, 3: sqft_living,
## #   4: sqft_lot, 5: waterfront
```

```
str(mydata)
```

```
## spc_tbl_ [21,597 x 21] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id           : num [1:21597] 7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
##  $ date         : chr [1:21597] "10/13/2014" "12/9/2014" "2/25/2015" "12/9/2014" ...
##  $ price        : num [1:21597] 221900 538000 180000 604000 510000 ...
##  $ bedrooms     : num [1:21597] 3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms    : num [1:21597] 1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
##  $ sqft_living  : num [1:21597] 1180 2570 770 1960 1680 ...
##  $ sqft_lot     : num [1:21597] 5650 7242 10000 5000 8080 ...
##  $ floors       : num [1:21597] 1 2 1 1 1 1 2 1 1 2 ...
##  $ waterfront   : num [1:21597] 0 0 0 0 0 0 0 0 0 0 ...
##  $ view         : num [1:21597] 0 0 0 0 0 0 0 0 0 0 ...
##  $ condition    : num [1:21597] 3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : num [1:21597] 7 7 6 7 8 11 7 7 7 7 ...
##  $ sqft_above   : num [1:21597] 1180 2170 770 1050 1680 ...
```

```
##  $ sqft_basement: num [1:21597] 0 400 0 910 0 1530 0 0 730 0 ...
##  $ yr_built     : num [1:21597] 1955 1951 1933 1965 1987 ...
##  $ yr_renovated : num [1:21597] 0 1991 0 0 0 ...
##  $ zipcode      : num [1:21597] 98178 98125 98028 98136 98074 ...
##  $ lat          : num [1:21597] 47.5 47.7 47.7 47.5 47.6 ...
##  $ long         : num [1:21597] -122 -122 -122 -122 -122 ...
##  $ sqft_living15: num [1:21597] 1340 1690 2720 1360 1800 ...
##  $ sqft_lot15   : num [1:21597] 5650 7639 8062 5000 7503 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_double(),
##   ..   date = col_character(),
##   ..   price = col_double(),
##   ..   bedrooms = col_double(),
##   ..   bathrooms = col_double(),
##   ..   sqft_living = col_double(),
##   ..   sqft_lot = col_double(),
##   ..   floors = col_double(),
##   ..   waterfront = col_double(),
##   ..   view = col_double(),
##   ..   condition = col_double(),
##   ..   grade = col_double(),
##   ..   sqft_above = col_double(),
##   ..   sqft_basement = col_double(),
##   ..   yr_built = col_double(),
##   ..   yr_renovated = col_double(),
##   ..   zipcode = col_double(),
##   ..   lat = col_double(),
##   ..   long = col_double(),
##   ..   sqft_living15 = col_double(),
##   ..   sqft_lot15 = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
summary(mydata)
```

```
##        id                 date               price            bedrooms
##  Min.   :1.000e+06   Length:21597       Min.   :  78000   Min.   : 1.000
##  1st Qu.:2.123e+09   Class :character   1st Qu.: 322000   1st Qu.: 3.000
##  Median :3.905e+09   Mode  :character   Median : 450000   Median : 3.000
##  Mean   :4.580e+09                      Mean   : 540297   Mean   : 3.373
##  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.   :0.500   Min.   :  370   Min.   :    520   Min.   :1.000
##  1st Qu.:1.750   1st Qu.: 1430   1st Qu.:   5040   1st Qu.:1.000
##  Median :2.250   Median : 1910   Median :   7618   Median :1.500
##  Mean   :2.116   Mean   : 2080   Mean   :  15099   Mean   :1.494
##  3rd Qu.:2.500   3rd Qu.: 2550   3rd Qu.:  10685   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.   :1.00   Min.   : 3.000
##  1st Qu.:0.000000   1st Qu.:0.0000   1st Qu.:3.00   1st Qu.: 7.000
##  Median :0.000000   Median :0.0000   Median :3.00   Median : 7.000
##  Mean   :0.007547   Mean   :0.2343   Mean   :3.41   Mean   : 7.658
```

```
##  3rd Qu.:0.000000   3rd Qu.:0.0000   3rd Qu.:4.00   3rd Qu.: 8.000
##  Max.   :1.000000   Max.   :4.0000   Max.   :5.00   Max.   :13.000
##    sqft_above   sqft_basement       yr_built     yr_renovated
##  Min.   : 370   Min.   :   0.0   Min.   :1900   Min.   :   0.00
##  1st Qu.:1190   1st Qu.:   0.0   1st Qu.:1951   1st Qu.:   0.00
##  Median :1560   Median :   0.0   Median :1975   Median :   0.00
##  Mean   :1789   Mean   : 291.7   Mean   :1971   Mean   :  84.46
##  3rd Qu.:2210   3rd Qu.: 560.0   3rd Qu.:1997   3rd Qu.:   0.00
##  Max.   :9410   Max.   :4820.0   Max.   :2015   Max.   :2015.00
##     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   : 12758
##  3rd Qu.: 10083
##  Max.   :871200
```

The data contains 21 different independent variables like bedrooms, sqft_living, view, grade, etc and the dependent variable is price. The data contains 21597 observations.

```
NA_values=data.frame(no_of_na_values=colSums(is.na(mydata)))
head(NA_values,21)
```

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

We see that there are no missing values in this data.

EXPLORATORY DATA ANALYSIS ON THE TRAIN DATA

Now we modify the data a litle and add two new columns for our better understanding. Price might depend on the age of the house and also the number of times it has been renovated. So we try to extact the age and the number of times a particular house has been renovated from our train data. This step is optional.

```
date_sale=mdy(mydata$date)
mydata$sale_date_year=as.integer(year(date_sale))
mydata$age=mydata$sale_date_year-mydata$yr_built

mydata$reno=ifelse(mydata$yr_renovated==0,0,1)
mydata$reno=as.factor(mydata$reno)
```

Training and test data

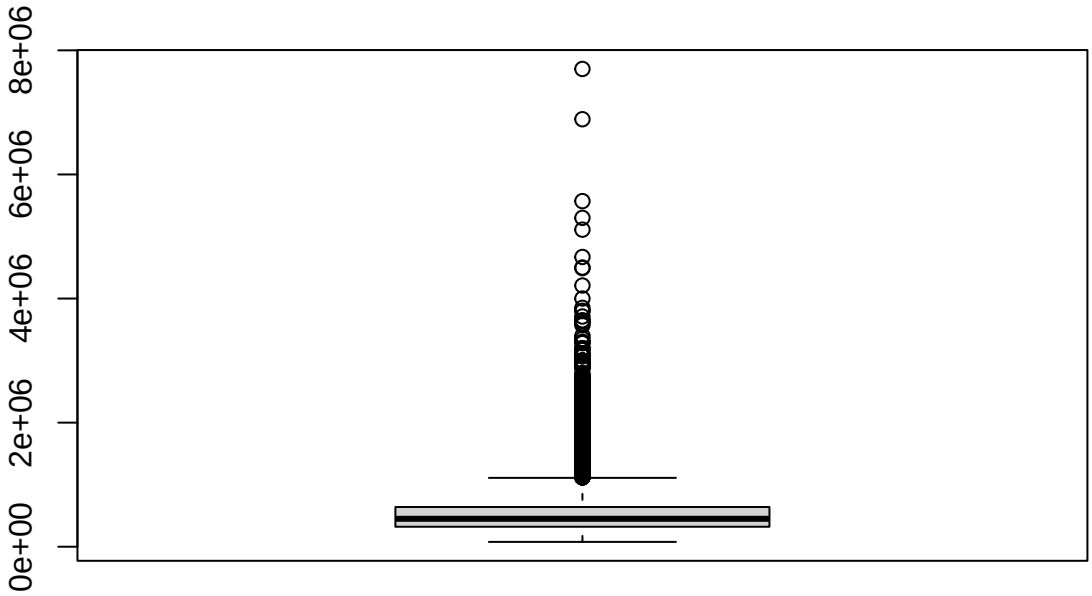
We divide the data to be the training set (80%) and test set (20%). If we have already the training and test set provided, then we do not need to add this step.

```
set.seed(123)   #  set seed to be your student number
n <- nrow(mydata)
ntest <- trunc(0.2*n)
testid <- sample (1:n, ntest)
train_data <- mydata[-testid , ]
test_data <- mydata[testid , ]
## train_data <- read_csv("train.csv")
## test_data <- read_csv("test.csv")
```

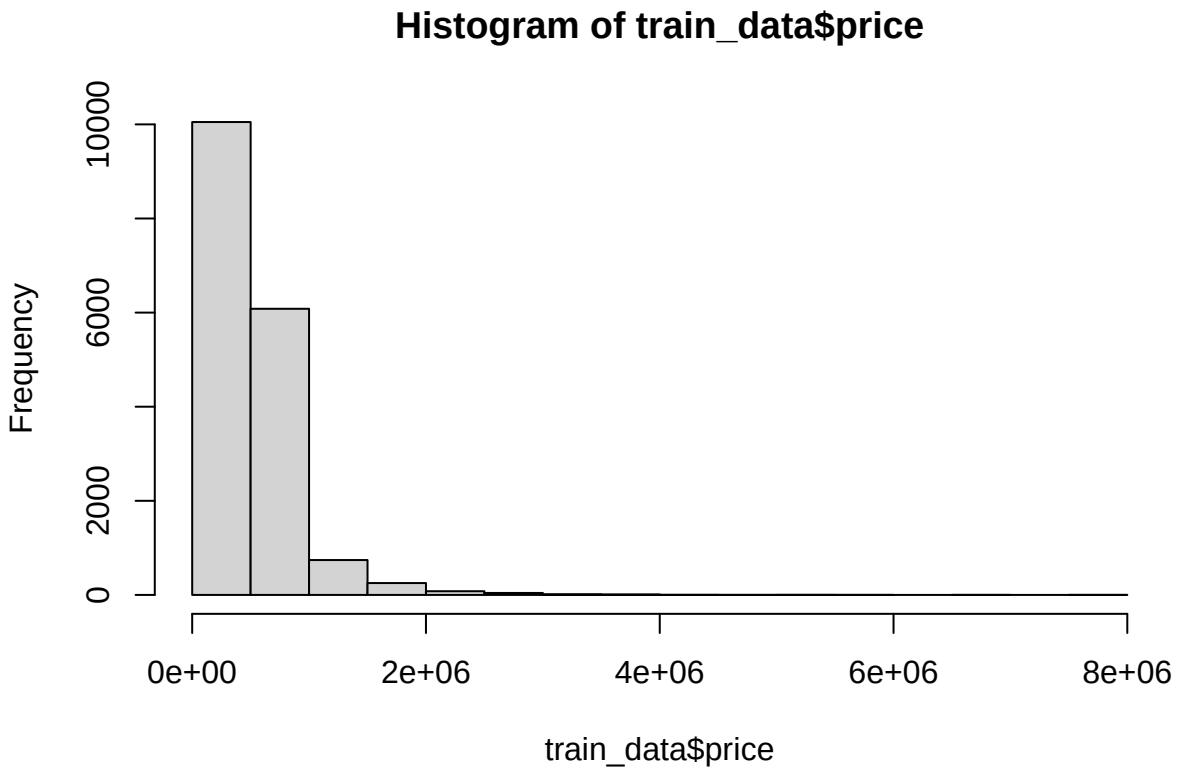
Check the response variable

The price is skewed to the right with several very high prices.

```
boxplot(train_data$price)
```




```
hist (train_data$price)
```



Determining the association between variables.

We take out the correlation plot (corrplot) to understand the association of the dependent variable (price) with the independent variables.

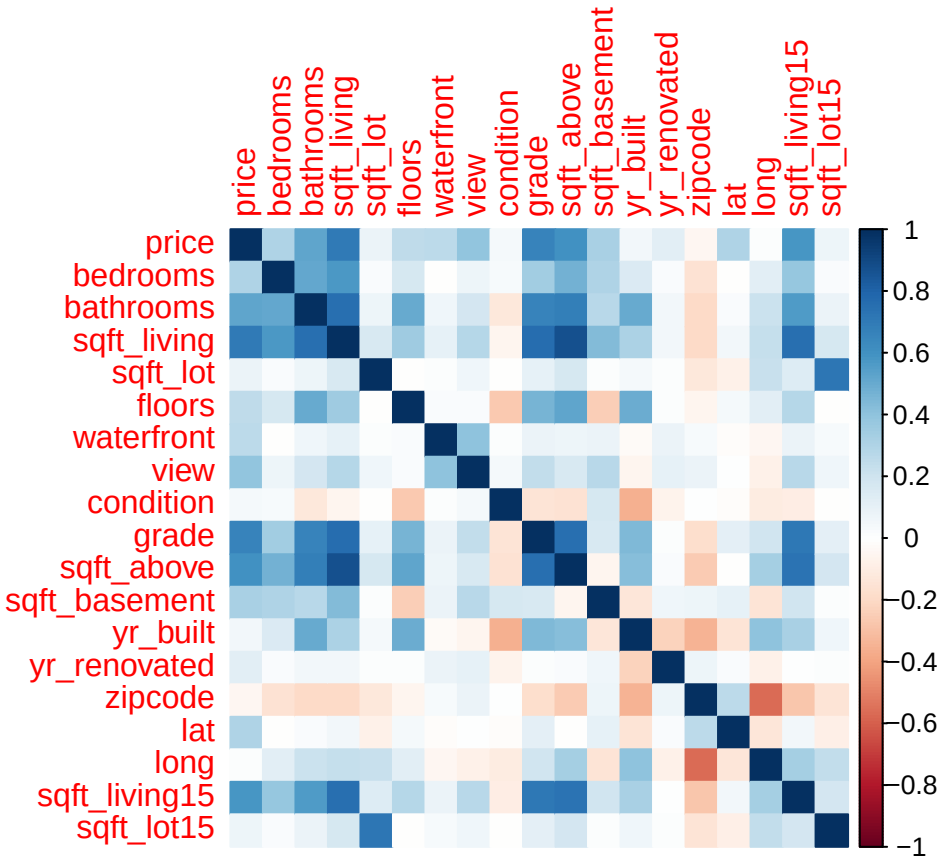
```
cor_data=data.frame(train_data[,3:21])
correlation=cor(cor_data)
correlation
```

```
##                    price     bedrooms   bathrooms sqft_living     sqft_lot
## price          1.00000000  0.306302787  0.52490594  0.70212024  0.084347903
## bedrooms       0.30630279  1.000000000  0.51055538  0.57297715  0.027650923
## bathrooms      0.52490594  0.510555381  1.00000000  0.75228007  0.078103071
## sqft_living    0.70212024  0.572977152  0.75228007  1.00000000  0.164906173
## sqft_lot       0.08434790  0.027650923  0.07810307  0.16490617  1.000000000
## floors         0.25927538  0.176161143  0.50595510  0.35375593 -0.009305346
## waterfront     0.26663606 -0.004203099  0.06734178  0.10573791  0.015507153
## view           0.39696983  0.075091326  0.18287640  0.28006469  0.064215017
## condition      0.04048509  0.033251516 -0.12226088 -0.05312872 -0.007590424
## grade          0.66792578  0.349968608  0.66195085  0.76301647  0.105263034
## sqft_above     0.60303192  0.472491302  0.68450153  0.87558637  0.176088535
## sqft_basement  0.32656512  0.303097464  0.27829769  0.43394077  0.012438925
## yr_built       0.05344571  0.151088523  0.50561628  0.31695206  0.046040871
## yr_renovated   0.12164813  0.022015283  0.05298045  0.05732197  0.011144552
## zipcode       -0.04374666 -0.151558128 -0.19896966 -0.19508520 -0.127713005
## lat            0.30873338 -0.008250136  0.02226417  0.05095703 -0.077745920
## long           0.01728054  0.127670792  0.21866126  0.23541131  0.221027374
## sqft_living15  0.58219439  0.389774460  0.56319978  0.75614768  0.144348312
## sqft_lot15     0.07999305  0.027023608  0.08123401  0.17995276  0.723738906
##                     floors   waterfront         view    condition       grade
```

```
## price          0.259275382  0.266636063  0.396969830  0.040485087  0.66792578
## bedrooms       0.176161143 -0.004203099  0.075091326  0.033251516  0.34996861
## bathrooms      0.505955100  0.067341777  0.182876397 -0.122260876  0.66195085
## sqft_living    0.353755928  0.105737911  0.280064688 -0.053128717  0.76301647
## sqft_lot      -0.009305346  0.015507153  0.064215017 -0.007590424  0.10526303
## floors         1.000000000  0.027303391  0.025500557 -0.261858784  0.46109093
## waterfront     0.027303391  1.000000000  0.402380080  0.018982642  0.08353448
## view           0.025500557  0.402380080  1.000000000  0.047676331  0.24901143
## condition     -0.261858784  0.018982642  0.047676331  1.000000000 -0.14467247
## grade          0.461090934  0.083534480  0.249011425 -0.144672472  1.00000000
## sqft_above     0.524749062  0.073949968  0.161674571 -0.153976780  0.75612677
## sqft_basement -0.247489596  0.080635204  0.277350701  0.177361466  0.16690372
## yr_built       0.490071230 -0.020644085 -0.053508083 -0.359024804  0.44646652
## yr_renovated   0.010800710  0.082118838  0.101567236 -0.060711155  0.01807452
## zipcode       -0.055487353  0.032590023  0.088109529  0.001374973 -0.17871974
## lat            0.049293971 -0.015880113  0.005464341 -0.016686007  0.11233900
## long           0.122612845 -0.040056713 -0.078155059 -0.105145980  0.19486912
## sqft_living15  0.280676336  0.082875200  0.273640870 -0.090018490  0.71035915
## sqft_lot15    -0.009790261  0.033659660  0.067732862 -0.002310117  0.11475100
##                 sqft_above sqft_basement    yr_built yr_renovated      zipcode
## price          0.603031921    0.32656512  0.05344571  0.121648129 -0.043746665
## bedrooms       0.472491302    0.30309746  0.15108852  0.022015283 -0.151558128
## bathrooms      0.684501529    0.27829769  0.50561628  0.052980448 -0.198969657
## sqft_living    0.875586372    0.43394077  0.31695206  0.057321970 -0.195085203
## sqft_lot       0.176088535    0.01243892  0.04604087  0.011144552 -0.127713005
## floors         0.524749062   -0.24748960  0.49007123  0.010800710 -0.055487353
## waterfront     0.073949968    0.08063520 -0.02064408  0.082118838  0.032590023
## view           0.161674571    0.27735070 -0.05350808  0.101567236  0.088109529
## condition     -0.153976780    0.17736147 -0.35902480 -0.060711155  0.001374973
## grade          0.756126769    0.16690372  0.44646652  0.018074519 -0.178719740
## sqft_above     1.000000000   -0.05525757  0.42291169  0.027129689 -0.256762357
## sqft_basement -0.055257565    1.00000000 -0.13362861  0.067883952  0.075643975
## yr_built       0.422911688   -0.13362861  1.00000000 -0.221916297 -0.345835356
## yr_renovated   0.027129689    0.06788395 -0.22191630  1.000000000  0.072187982
## zipcode       -0.256762357    0.07564398 -0.34583536  0.072187982  1.000000000
## lat           -0.001088706    0.10735697 -0.14822757  0.025793897  0.268750571
## long           0.337882978   -0.14358651  0.40938151 -0.071829779 -0.566075312
## sqft_living15  0.731947188    0.19780264  0.32304513  0.000108915 -0.277278790
## sqft_lot15     0.189917516    0.01774777  0.06720285  0.014171453 -0.145160008
##                        lat         long sqft_living15   sqft_lot15
## price          0.308733385  0.01728054   0.582194391  0.079993046
## bedrooms      -0.008250136  0.12767079   0.389774460  0.027023608
## bathrooms      0.022264171  0.21866126   0.563199785  0.081234009
## sqft_living    0.050957025  0.23541131   0.756147679  0.179952762
## sqft_lot      -0.077745920  0.22102737   0.144348312  0.723738906
## floors         0.049293971  0.12261284   0.280676336 -0.009790261
## waterfront    -0.015880113 -0.04005671   0.082875200  0.033659660
## view           0.005464341 -0.07815506   0.273640870  0.067732862
## condition     -0.016686007 -0.10514598  -0.090018490 -0.002310117
## grade          0.112339001  0.19486912   0.710359154  0.114750998
## sqft_above    -0.001088706  0.33788298   0.731947188  0.189917516
## sqft_basement  0.107356972 -0.14358651   0.197802645  0.017747769
## yr_built      -0.148227566  0.40938151   0.323045128  0.067202845
## yr_renovated   0.025793897 -0.07182978   0.000108915  0.014171453
```

```
## zipcode        0.268750571 -0.56607531  -0.277278790 -0.145160008
## lat            1.000000000 -0.13316643   0.050721208 -0.080010083
## long          -0.133166426  1.00000000   0.332957537  0.248039625
## sqft_living15  0.050721208  0.33295754   1.000000000  0.180965468
## sqft_lot15    -0.080010083  0.24803963   0.180965468  1.000000000
```

```
par(mfrow=c(1, 1))
corrplot(correlation,method="color")
```



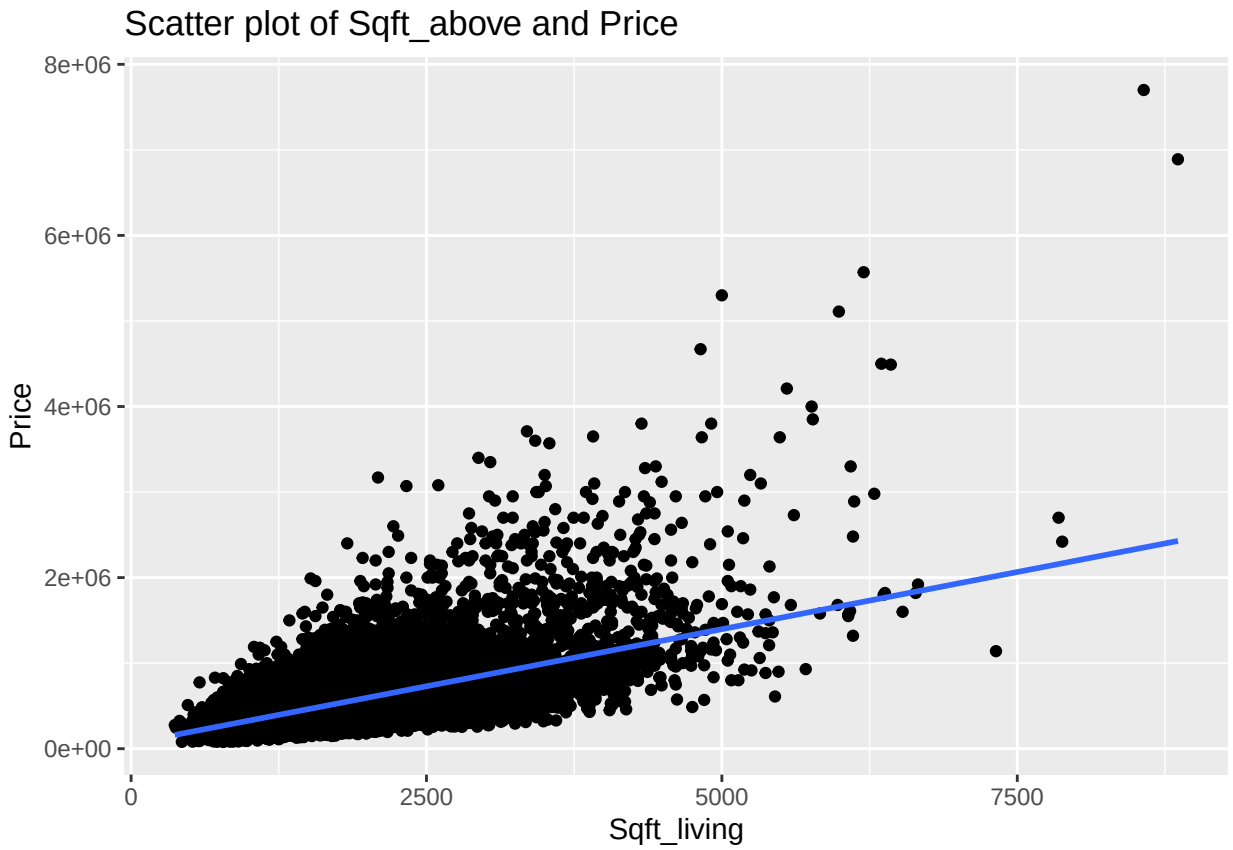
Price is strongly positively correlated with bathroom, Sqft_living, grade, sqft_above, sqft_living15. We use scatterplot and boxplot to visualize the relationship between price and some predictors.

```
ggplot(data = train_data, aes(x = sqft_living, y = price)) +
  geom_jitter() +  geom_smooth(method = "lm", se = FALSE)+labs(title="Scatter plot of Sqft_living and P
```

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

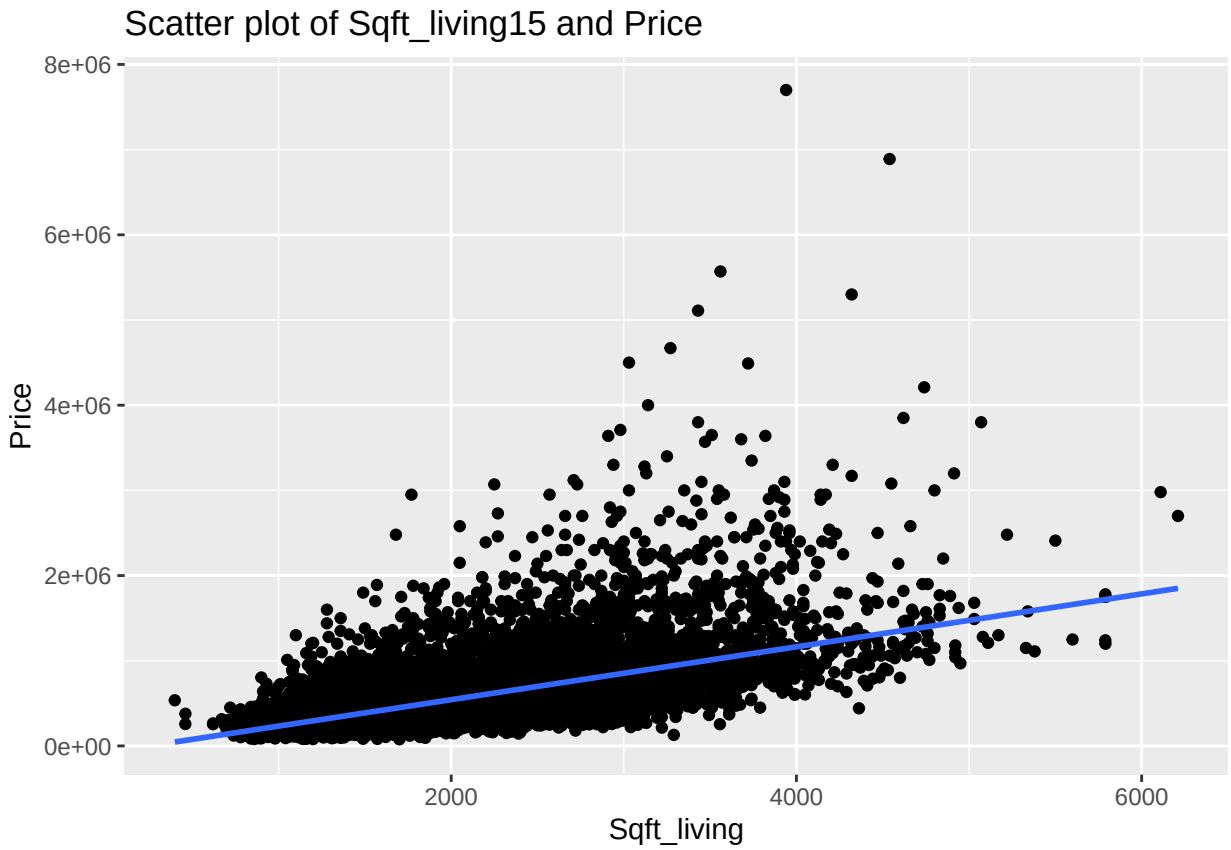

```
ggplot(data = train_data, aes(x = sqft_above, y = price)) +
  geom_jitter() +  geom_smooth(method = "lm", se = FALSE)+labs(title="Scatter plot of Sqft_above and Pri
```

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

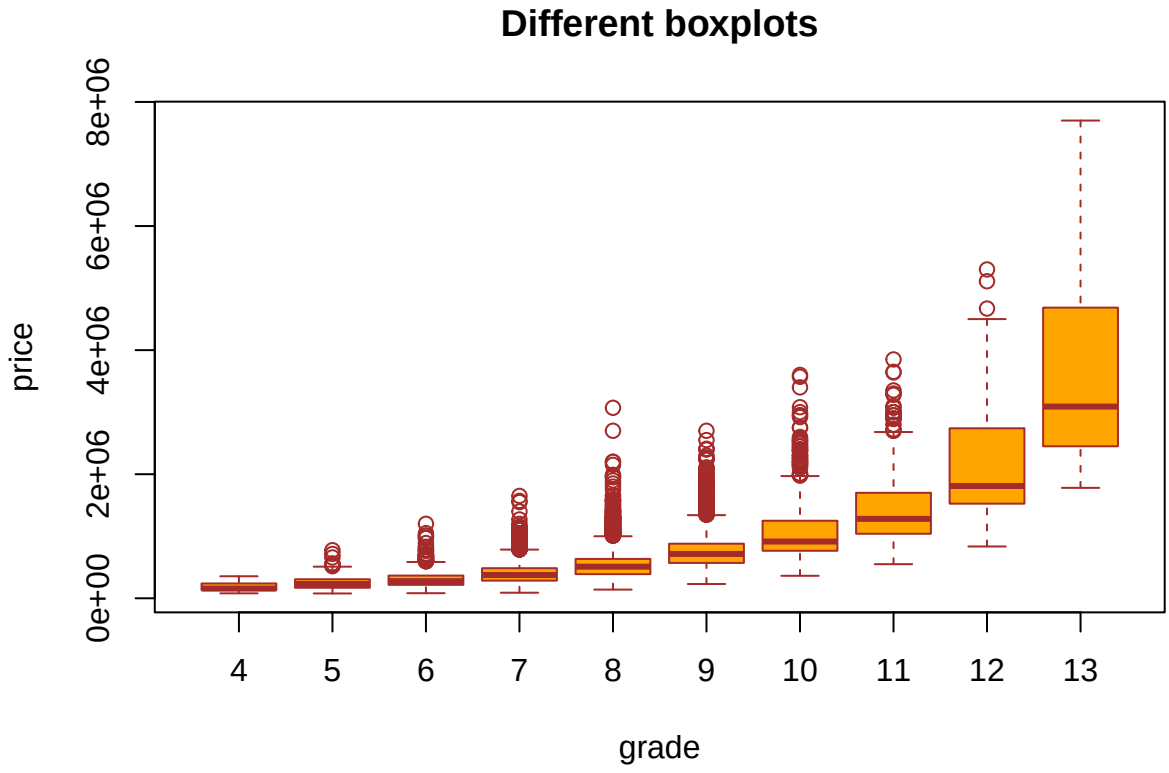



```
ggplot(data = train_data, aes(x = sqft_living15, y = price)) +
  geom_jitter() +  geom_smooth(method = "lm", se = FALSE)+labs(title="Scatter plot of Sqft_living15 and
```

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



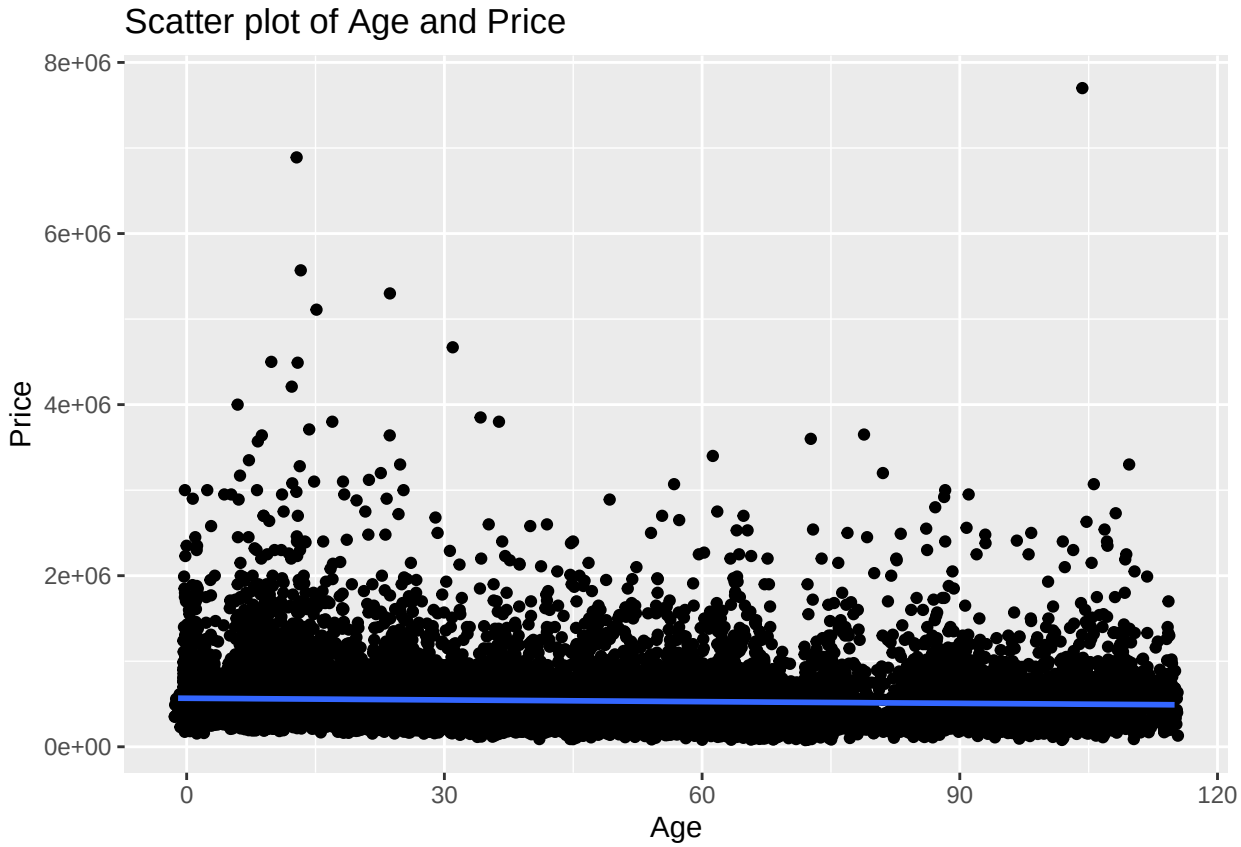

```
boxplot(price~grade,data=train_data,main="Different boxplots", xlab="grade",ylab="price",col="orange",b
```



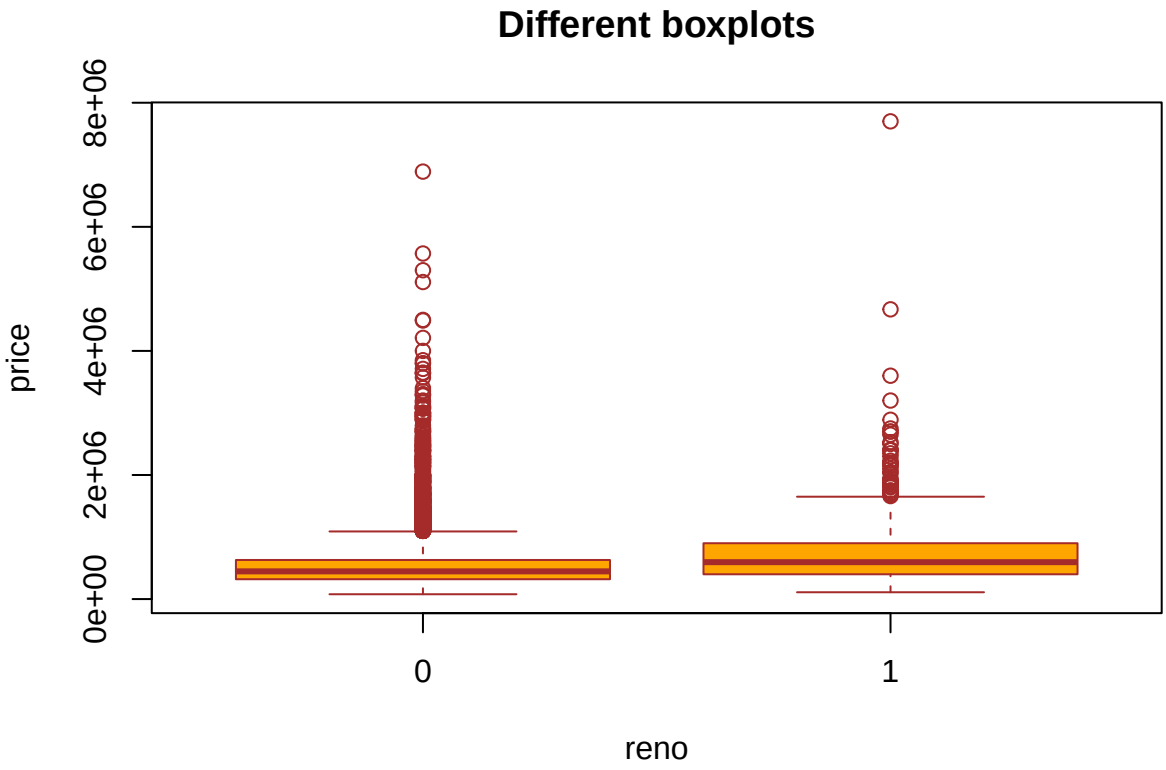
Check the new added variables:

```
ggplot(data = train_data, aes(x = age, y = price)) +
  geom_jitter() +  geom_smooth(method = "lm", se = FALSE)+labs(title="Scatter plot of Age and Price", x=
```

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




```
boxplot(price~reno,data=train_data,main="Different boxplots", xlab="reno",ylab="price",col="orange",bor
```



Removing outlier could be optional

We see that we have a significantly large number of outliers.

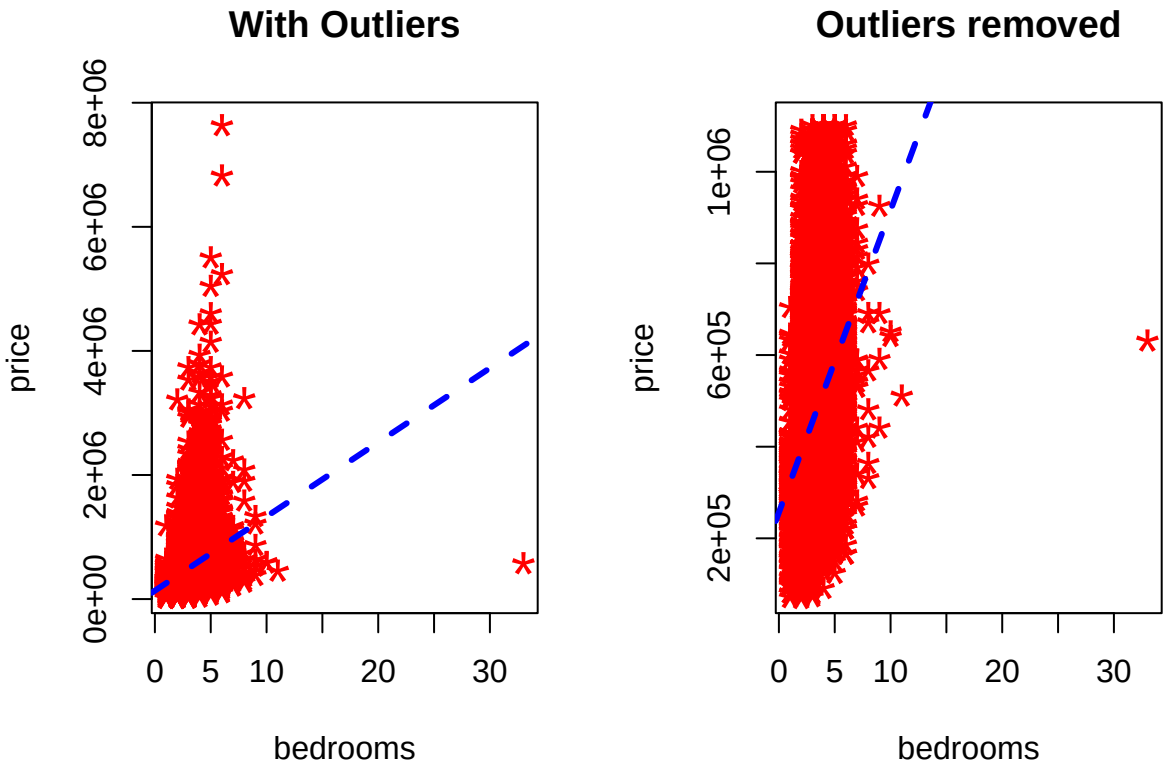
Treating or altering the outlier/extreme values in genuine observations is not a standard operating procedure. However, it is essential to understand their impact on our predictive models.

To better understand the implications of outliers better, we should compare the fit of a simple linear regression model on the dataset with and without outliers. For this we first extract outliers from the data and then obtain the data without the outliers.

```
outliers=boxplot(train_data$price,plot=FALSE)$out
outliers_data=train_data[which(train_data$price %in% outliers),]
train_data1= train_data[-which(train_data$price %in% outliers),]
```

Now plot Now we plot the data with and without outliers.

```
par(mfrow=c(1, 2))
plot(train_data$bedrooms, train_data$price, main="With Outliers", xlab="bedrooms", ylab="price", pch="*
abline(lm(price ~ bedrooms, data=train_data), col="blue", lwd=3, lty=2)
# Plot of original data without outliers. Note the change of slope.
plot(train_data1$bedrooms, train_data1$price, main="Outliers removed", xlab="bedrooms", ylab="price", p
abline(lm(price ~bedrooms, data=train_data1), col="blue", lwd=3, lty=2)
```



MODELING

We first use the entire data.

```
train_data.m <- train_data[, -c(1, 2, 15, 16, 17, 22)] %>% mutate(waterfront=as.factor(waterfront), viev
str (train_data.m)
```

```
## tibble [17,278 x 18] (S3: tbl_df/tbl/data.frame)
##  $ price        : num [1:17278] 221900 538000 180000 604000 1230000 ...
##  $ bedrooms     : num [1:17278] 3 3 2 4 4 3 3 3 3 3 ...
##  $ bathrooms    : num [1:17278] 1 2.25 1 3 4.5 2.25 1.5 1 2.5 2.5 ...
##  $ sqft_living  : num [1:17278] 1180 2570 770 1960 5420 ...
##  $ sqft_lot     : num [1:17278] 5650 7242 10000 5000 101930 ...
##  $ floors       : num [1:17278] 1 2 1 1 1 2 1 1 2 1 ...
##  $ waterfront   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ view         : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ condition    : Factor w/ 5 levels "1","2","3","4",..: 3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : num [1:17278] 7 7 6 7 11 7 7 7 7 8 ...
##  $ sqft_above   : num [1:17278] 1180 2170 770 1050 3890 ...
##  $ sqft_basement: num [1:17278] 0 400 0 910 1530 0 0 730 0 1700 ...
##  $ lat          : num [1:17278] 47.5 47.7 47.7 47.5 47.7 ...
##  $ long         : num [1:17278] -122 -122 -122 -122 -122 ...
##  $ sqft_living15: num [1:17278] 1340 1690 2720 1360 4760 ...
##  $ sqft_lot15   : num [1:17278] 5650 7639 8062 5000 101930 ...
##  $ age          : num [1:17278] 59 63 82 49 13 19 52 55 12 50 ...
##  $ reno         : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
```

```
ncol(train_data.m)
```

```
## [1] 18
```

```r
model.full <- lm (formula = price ~ ., data = train_data.m)
summary(model.full)
```

```
##
## Call:
## lm(formula = price ~ ., data = train_data.m)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1241633   -99268    -9611    77078  4371509
##
## Coefficients: (1 not defined because of singularities)
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -4.073e+07  1.692e+06 -24.079  < 2e-16 ***
## bedrooms      -3.293e+04  2.086e+03 -15.782  < 2e-16 ***
## bathrooms      4.396e+04  3.615e+03  12.161  < 2e-16 ***
## sqft_living    1.482e+02  4.900e+00  30.253  < 2e-16 ***
## sqft_lot       1.490e-01  5.434e-02   2.743   0.0061 **
## floors         4.912e+03  3.992e+03   1.230   0.2185
## waterfront1    4.950e+05  2.214e+04  22.356  < 2e-16 ***
## view1          8.923e+04  1.275e+04   6.996 2.73e-12 ***
## view2          6.063e+04  7.710e+03   7.864 3.93e-15 ***
## view3          1.366e+05  1.039e+04  13.151  < 2e-16 ***
## view4          3.157e+05  1.630e+04  19.364  < 2e-16 ***
## condition2     3.913e+04  4.336e+04   0.903   0.3668
## condition3     1.441e+04  4.019e+04   0.359   0.7200
## condition4     4.813e+04  4.019e+04   1.198   0.2310
## condition5     8.152e+04  4.044e+04   2.016   0.0438 *
## grade          9.877e+04  2.406e+03  41.055  < 2e-16 ***
## sqft_above     2.677e+01  4.855e+00   5.513 3.57e-08 ***
## sqft_basement         NA         NA      NA       NA
## lat            5.611e+05  1.161e+04  48.326  < 2e-16 ***
## long          -1.087e+05  1.323e+04  -8.217 2.24e-16 ***
## sqft_living15  2.498e+01  3.819e+00   6.540 6.30e-11 ***
## sqft_lot15    -4.113e-01  8.247e-02  -4.987 6.20e-07 ***
## age            2.470e+03  8.045e+01  30.708  < 2e-16 ***
## reno1          3.306e+04  8.131e+03   4.066 4.81e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 199600 on 17255 degrees of freedom
## Multiple R-squared:  0.6996, Adjusted R-squared:  0.6992
## F-statistic:  1827 on 22 and 17255 DF,  p-value: < 2.2e-16
```

```r
models <- regsubsets(price~., data = train_data.m, nvmax = 23)
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 1 linear dependencies found
```

```
## Reordering variables and trying again:
```

```r
summary(models)
```

```
## Subset selection object
## Call: regsubsets.formula(price ~ ., data = train_data.m, nvmax = 23)
## 23 Variables  (and intercept)
```

```
##               Forced in Forced out
## bedrooms          FALSE      FALSE
## bathrooms         FALSE      FALSE
## sqft_living       FALSE      FALSE
## sqft_lot          FALSE      FALSE
## floors            FALSE      FALSE
## waterfront1       FALSE      FALSE
## view1             FALSE      FALSE
## view2             FALSE      FALSE
## view3             FALSE      FALSE
## view4             FALSE      FALSE
## condition2        FALSE      FALSE
## condition3        FALSE      FALSE
## condition4        FALSE      FALSE
## condition5        FALSE      FALSE
## grade             FALSE      FALSE
## sqft_above        FALSE      FALSE
## lat               FALSE      FALSE
## long              FALSE      FALSE
## sqft_living15     FALSE      FALSE
## sqft_lot15        FALSE      FALSE
## age               FALSE      FALSE
## reno1             FALSE      FALSE
## sqft_basement     FALSE      FALSE
## 1 subsets of each size up to 22
## Selection Algorithm: exhaustive
##           bedrooms bathrooms sqft_living sqft_lot floors waterfront1 view1
## 1  ( 1 )  " "      " "       "*"         " "      " "    " "         " "
## 2  ( 1 )  " "      " "       "*"         " "      " "    " "         " "
## 3  ( 1 )  " "      " "       "*"         " "      " "    "*"         " "
## 4  ( 1 )  " "      " "       "*"         " "      " "    " "         " "
## 5  ( 1 )  " "      " "       "*"         " "      " "    "*"         " "
## 6  ( 1 )  " "      " "       "*"         " "      " "    "*"         " "
## 7  ( 1 )  "*"      " "       "*"         " "      " "    "*"         " "
## 8  ( 1 )  "*"      "*"       "*"         " "      " "    "*"         " "
## 9  ( 1 )  "*"      "*"       "*"         " "      " "    "*"         " "
## 10  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         " "
## 11  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         " "
## 12  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         "*"
## 13  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         "*"
## 14  ( 1 ) "*"      "*"       " "         " "      " "    "*"         "*"
## 15  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         "*"
## 16  ( 1 ) "*"      "*"       " "         " "      " "    "*"         "*"
## 17  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         "*"
## 18  ( 1 ) "*"      "*"       "*"         " "      " "    "*"         "*"
## 19  ( 1 ) "*"      "*"       "*"         "*"      " "    "*"         "*"
## 20  ( 1 ) "*"      "*"       "*"         "*"      " "    "*"         "*"
## 21  ( 1 ) "*"      "*"       "*"         "*"      "*"    "*"         "*"
## 22  ( 1 ) "*"      "*"       "*"         "*"      "*"    "*"         "*"
##           view2 view3 view4 condition2 condition3 condition4 condition5 grade
## 1  ( 1 )  " "   " "   " "   " "        " "        " "        " "        " "
## 2  ( 1 )  " "   " "   " "   " "        " "        " "        " "        " "
## 3  ( 1 )  " "   " "   " "   " "        " "        " "        " "        " "
## 4  ( 1 )  " "   " "   " "   " "        " "        " "        " "        "*"
```

```
## 5  ( 1 )  " "   " "   " "   " "        " "        " "        " "        "*"
## 6  ( 1 )  " "   " "   "*"   " "        " "        " "        " "        "*"
## 7  ( 1 )  " "   " "   "*"   " "        " "        " "        " "        "*"
## 8  ( 1 )  " "   " "   "*"   " "        " "        " "        " "        "*"
## 9  ( 1 )  " "   "*"   "*"   " "        " "        " "        " "        "*"
## 10  ( 1 ) " "   "*"   "*"   " "        "*"        " "        " "        "*"
## 11  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        " "        "*"
## 12  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        " "        "*"
## 13  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        " "        "*"
## 14  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        " "        "*"
## 15  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        " "        "*"
## 16  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        "*"        "*"
## 17  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        "*"        "*"
## 18  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        "*"        "*"
## 19  ( 1 ) "*"   "*"   "*"   " "        "*"        " "        "*"        "*"
## 20  ( 1 ) "*"   "*"   "*"   "*"        " "        "*"        "*"        "*"
## 21  ( 1 ) "*"   "*"   "*"   "*"        " "        "*"        "*"        "*"
## 22  ( 1 ) "*"   "*"   "*"   "*"        "*"        "*"        "*"        "*"
##           sqft_above sqft_basement lat long sqft_living15 sqft_lot15 age reno1
## 1  ( 1 )  " "        " "           " " " "  " "           " "        " " " "
## 2  ( 1 )  " "        " "           "*" " "  " "           " "        " " " "
## 3  ( 1 )  " "        " "           "*" " "  " "           " "        " " " "
## 4  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 5  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 6  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 7  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 8  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 9  ( 1 )  " "        " "           "*" " "  " "           " "        "*" " "
## 10  ( 1 ) " "        " "           "*" " "  " "           " "        "*" " "
## 11  ( 1 ) " "        " "           "*" " "  " "           " "        "*" " "
## 12  ( 1 ) " "        " "           "*" " "  " "           " "        "*" " "
## 13  ( 1 ) " "        " "           "*" "*"  " "           " "        "*" " "
## 14  ( 1 ) "*"        "*"           "*" "*"  " "           " "        "*" " "
## 15  ( 1 ) " "        "*"           "*" "*"  "*"           " "        "*" " "
## 16  ( 1 ) "*"        "*"           "*" "*"  "*"           " "        "*" " "
## 17  ( 1 ) "*"        " "           "*" "*"  "*"           "*"        "*" " "
## 18  ( 1 ) "*"        " "           "*" "*"  "*"           "*"        "*" "*"
## 19  ( 1 ) "*"        " "           "*" "*"  "*"           "*"        "*" "*"
## 20  ( 1 ) "*"        " "           "*" "*"  "*"           "*"        "*" "*"
## 21  ( 1 ) " "        "*"           "*" "*"  "*"           "*"        "*" "*"
## 22  ( 1 ) "*"        " "           "*" "*"  "*"           "*"        "*" "*"
```

```
res.sum <- summary(models)
data.frame(
  Adj.R2 = which.max(res.sum$adjr2),
  CP = which.min(res.sum$cp),
  BIC = which.min(res.sum$bic)
)
```

```
##   Adj.R2 CP BIC
## 1     21 19  18
```

```
# id: model id
# object: regsubsets object
# data: data used to fit regsubsets
```

```r
# outcome: outcome variable
get_model_formula <- function(id, object, outcome){
  # get models data
  models <- summary(object)$which[id,-1]
  # Get outcome variable
  #form <- as.formula(object$call[[2]])
  #outcome <- all.vars(form)[1]
  # Get model predictors
  predictors <- names(which(models == TRUE))
  predictors <- paste(predictors, collapse = "+")
  # Build model formula
  as.formula(paste0(outcome, "~", predictors))
}

get_model_formula(21, models, "Price")
```

```
## Price ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors +
##     waterfront1 + view1 + view2 + view3 + view4 + condition2 +
##     condition4 + condition5 + grade + sqft_basement + lat + long +
##     sqft_living15 + sqft_lot15 + age + reno1
## <environment: 0x7fc4c7ce1f58>
```

```r
model1 <- lm (price ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condi
summary(model1)
```

```
##
## Call:
## lm(formula = price ~ bedrooms + bathrooms + sqft_living + sqft_lot +
##     floors + waterfront + view + condition + grade + sqft_basement +
##     lat + long + sqft_living15 + sqft_lot15 + age + reno, data = train_data.m)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1241633   -99268    -9611    77078  4371509
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -4.073e+07  1.692e+06 -24.079  < 2e-16 ***
## bedrooms      -3.293e+04  2.086e+03 -15.782  < 2e-16 ***
## bathrooms      4.396e+04  3.615e+03  12.161  < 2e-16 ***
## sqft_living    1.750e+02  4.110e+00  42.580  < 2e-16 ***
## sqft_lot       1.490e-01  5.434e-02   2.743   0.0061 **
## floors         4.912e+03  3.992e+03   1.230   0.2185
## waterfront1    4.950e+05  2.214e+04  22.356  < 2e-16 ***
## view1          8.923e+04  1.275e+04   6.996 2.73e-12 ***
## view2          6.063e+04  7.710e+03   7.864 3.93e-15 ***
## view3          1.366e+05  1.039e+04  13.151  < 2e-16 ***
## view4          3.157e+05  1.630e+04  19.364  < 2e-16 ***
## condition2     3.913e+04  4.336e+04   0.903   0.3668
## condition3     1.441e+04  4.019e+04   0.359   0.7200
## condition4     4.813e+04  4.019e+04   1.198   0.2310
## condition5     8.152e+04  4.044e+04   2.016   0.0438 *
## grade          9.877e+04  2.406e+03  41.055  < 2e-16 ***
## sqft_basement -2.677e+01  4.855e+00  -5.513 3.57e-08 ***
```

```
## lat            5.611e+05  1.161e+04  48.326  < 2e-16 ***
## long          -1.087e+05  1.323e+04  -8.217 2.24e-16 ***
## sqft_living15  2.498e+01  3.819e+00   6.540 6.30e-11 ***
## sqft_lot15    -4.113e-01  8.247e-02  -4.987 6.20e-07 ***
## age            2.470e+03  8.045e+01  30.708  < 2e-16 ***
## reno1          3.306e+04  8.131e+03   4.066 4.81e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 199600 on 17255 degrees of freedom
## Multiple R-squared:  0.6996, Adjusted R-squared:  0.6992
## F-statistic:  1827 on 22 and 17255 DF,  p-value: < 2.2e-16
```

```
model2 <- lm (price ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condi
summary(model2)
```

```
##
## Call:
## lm(formula = price ~ bedrooms + bathrooms + sqft_living + sqft_lot +
##     floors + waterfront + view + condition + grade + sqft_basement +
##     lat + long + sqft_living15 + sqft_lot15 + age + reno + bathrooms *
##     grade + grade * sqft_living15 + grade * sqft_lot15 + lat *
##     long, data = train_data.m)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1994817   -91128    -8349    74394  2856228
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)          5.921e+09  4.617e+08  12.826  < 2e-16 ***
## bedrooms            -1.319e+04  1.993e+03  -6.620 3.69e-11 ***
## bathrooms           -4.659e+05  1.313e+04 -35.490  < 2e-16 ***
## sqft_living          1.201e+02  4.037e+00  29.750  < 2e-16 ***
## sqft_lot             1.169e-01  5.087e-02   2.299 0.021545 *
## floors               1.974e+04  3.786e+03   5.214 1.87e-07 ***
## waterfront1          5.106e+05  2.072e+04  24.644  < 2e-16 ***
## view1                9.427e+04  1.191e+04   7.913 2.66e-15 ***
## view2                5.359e+04  7.208e+03   7.434 1.10e-13 ***
## view3                1.169e+05  9.721e+03  12.028  < 2e-16 ***
## view4                2.619e+05  1.526e+04  17.156  < 2e-16 ***
## condition2           7.996e+04  4.050e+04   1.974 0.048345 *
## condition3           9.007e+04  3.756e+04   2.398 0.016483 *
## condition4           1.343e+05  3.756e+04   3.575 0.000352 ***
## condition5           1.708e+05  3.782e+04   4.517 6.31e-06 ***
## grade               -3.450e+04  4.064e+03  -8.491  < 2e-16 ***
## sqft_basement        1.363e+00  4.611e+00   0.295 0.767627
## lat                 -1.248e+08  9.711e+06 -12.852  < 2e-16 ***
## long                 4.867e+07  3.778e+06  12.884  < 2e-16 ***
## sqft_living15        8.078e+01  1.570e+01   5.145 2.70e-07 ***
## sqft_lot15           1.295e+00  2.786e-01   4.649 3.35e-06 ***
## age                  1.806e+03  7.670e+01  23.551  < 2e-16 ***
## reno1                5.433e+04  7.613e+03   7.137 9.90e-13 ***
## bathrooms:grade      6.554e+04  1.615e+03  40.576  < 2e-16 ***
## grade:sqft_living15 -4.863e+00  1.829e+00  -2.658 0.007863 **
```

```
## grade:sqft_lot15    -1.995e-01  3.235e-02  -6.166 7.15e-10 ***
## lat:long            -1.026e+06  7.946e+04 -12.910  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 186300 on 17251 degrees of freedom
## Multiple R-squared:  0.7384, Adjusted R-squared:  0.738
## F-statistic:  1873 on 26 and 17251 DF,  p-value: < 2.2e-16
```

PREDICTION ON THE TEST DATA

```
test_data.m <- test_data[, -c(1, 2, 15, 16, 17, 22)] %>% mutate(waterfront=as.factor(waterfront), view=a
```

```
pred_test=predict(newdata=test_data.m,model2)

tally_table_1=data.frame(actual=test_data.m$price, predicted=pred_test)

mean(abs(test_data.m$price - pred_test))
```

```
## [1] 122197.1
```

Compare with one-layer forward neural network

```
x <- model.matrix(price ~ . - 1, data = train_data.m) %>% scale ()
y <- train_data.m$price
```

```
library(keras)
modnn <- keras_model_sequential () %>%
  layer_dense(units = 100, activation = "relu",
              input_shape = ncol(x)) %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 1)
```

```
## Loaded Tensorflow version 2.7.0
```

```
modnn %>% compile(loss = "mse",
                  optimizer = optimizer_rmsprop (),
                  metrics = list("mean_absolute_error")
)
```

```
x.test <- model.matrix(price ~ . - 1, data = test_data.m) %>% scale ()
y.test <- test_data.m$price

history <- modnn %>% fit(
  x.test, y.test, epochs = 1000, batch_size = 32,
  validation_data = list(x.test, y.test)
)

npred <- predict(modnn , x.test)
mean(abs(y.test - npred))
```

```
## [1] 128736.9
```

Neural network has similar test error than the multiple linear regression. But if we have time, we can tune the parameters to have a better neural network.