Regression

Saugat Gyawali

09/13/2022

Linear Regression

It is used to predict the value of a variable based on the value of another variable. These sort of models are simple and provide and easy to interpret mathematical formula so that we can generate prediction.

How does Linear Regression work?

As the name suggests, linear regression shows the relationship between dependent variable(y) and independent variable(x) in linear format. Let us assume, linear function as Y = b + wX. While training the model, we are given: x - input training data(univariate), y: labels to data(supervised learning), w = coefficient of x and b is y-intercept. Once we find the good w and b we can get the best line for x and y. With that we can find the value of y using the value of x.

It is very important to update w and b values, to find the best fit line by minimizing the error between predicted value and true value(y).

Cost function(mean squared error):

$$J = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Gradient descent: In order to update w and b we can reduce cost function and achieve the best fit line. The idea is to start with random w and b and then iterately updating the values, reaching minimum cost.

$$\hat{b} = \bar{y} - \hat{w}\bar{x} \qquad \hat{w} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Formula to find w

Strength of Linear Regression:

- Implementation is simple using it.

- Works for a good pattern that follows linear.
- Low variance

Weakness of Linear regression

- Underfitting
- Sensitive to outliers
- Assumes that the data is independent.

Source of data set is here

Reading csv file from Kaggle dataset.

```
data <- read.csv("kc_house_data.csv")

dim(data)

## [1] 21613    21
```

Dividing the data into train and test data.

We divide the data in 80:20 ratio meaning, 80 percentage is for training and 20% of data is for testing purpose.

```
set.seed(1234)
i <- sample(1:nrow(data), nrow(data) * 0.80, replace=FALSE)
train <- data[i,]
test <- data[-i,]
```

Some of the Data Exploration using the training data

```
names(train)

##  [1] "id"            "date"          "price"         "bedrooms"
##  [5] "bathrooms"     "sqft_living"   "sqft_lot"      "floors"
##  [9] "waterfront"    "view"          "condition"     "grade"
## [13] "sqft_above"    "sqft_basement" "yr_built"      "yr_renovated"
## [17] "zipcode"       "lat"           "long"          "sqft_living15"
## [21] "sqft_lot15"

dim(train)

## [1] 17290    21

summary(train)

##        id                 date               price            bedrooms
##  Min.   :1.000e+06   Length:17290       Min.   :  75000   Min.   : 0.000
```

```
##  1st Qu.:2.116e+09   Class :character   1st Qu.: 320900   1st Qu.: 3.000
##  Median :3.902e+09   Mode  :character   Median : 450000   Median : 3.000
##  Mean   :4.564e+09                      Mean   : 541038   Mean   : 3.371
##  3rd Qu.:7.300e+09                      3rd Qu.: 645000   3rd Qu.: 4.000
##  Max.   :9.900e+09                      Max.   :6885000   Max.   :11.000
##    bathrooms      sqft_living       sqft_lot           floors
##  Min.   :0.000   Min.   :  290   Min.   :    520   Min.   :1.000
##  1st Qu.:1.750   1st Qu.: 1430   1st Qu.:   5034   1st Qu.:1.000
##  Median :2.250   Median : 1910   Median :   7616   Median :1.500
##  Mean   :2.117   Mean   : 2082   Mean   :  15175   Mean   :1.497
##  3rd Qu.:2.500   3rd Qu.: 2550   3rd Qu.:  10686   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   :0.007808   Mean   :0.2403   Mean   :3.409   Mean   : 7.655
##  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.   :   0.00
##  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   :1790   Mean   : 292.2   Mean   :1971   Mean   :  85.29
##  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.:1486
##  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.:2370
##  Max.   :98199   Max.   :47.78   Max.   :-121.3   Max.   :6210
##    sqft_lot15
##  Min.   :   659
##  1st Qu.:  5100
##  Median :  7620
##  Mean   : 12807
##  3rd Qu.: 10087
##  Max.   :871200

str(train)

## 'data.frame':    17290 obs. of  21 variables:
##  $ id           : num  7.00e+09 3.89e+09 1.04e+09 8.66e+09 7.94e+09 ...
##  $ date         : chr  "20140715T000000" "20150304T000000" "20150312T00000
0" "20150330T000000" ...
##  $ price        : num  600000 606000 660000 537000 975000 ...
##  $ bedrooms     : int  3 3 3 4 3 3 4 4 3 3 ...
##  $ bathrooms    : num  1 2 3.5 2.5 2.5 1.5 2.5 1.5 2.25 1.5 ...
```

```
##  $ sqft_living  : int  940 1980 2740 1990 2530 1210 2320 1840 1560 2290 ..
.
##  $ sqft_lot     : int  19000 7680 3785 2660 7000 10588 9264 7076 35026 960
0 ...
##  $ floors       : num  1 1.5 2 2 2.5 1 2 1.5 1 1 ...
##  $ waterfront   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ view         : int  0 0 0 0 4 0 0 0 0 0 ...
##  $ condition    : int  3 4 3 3 3 4 3 3 3 4 ...
##  $ grade        : int  6 6 9 8 9 7 8 7 7 7 ...
##  $ sqft_above   : int  940 1070 2190 1990 2530 1210 2320 1840 1290 2290 ..
.
##  $ sqft_basement: int  0 910 550 0 0 0 0 0 270 0 ...
##  $ yr_built     : int  1945 1911 2001 2012 1915 1958 1994 1957 1985 1967 .
..
##  $ yr_renovated : int  0 0 0 0 1999 0 0 0 0 0 ...
##  $ zipcode      : int  98004 98033 98034 98034 98136 98002 98188 98106 980
92 98042 ...
##  $ lat          : num  47.6 47.7 47.7 47.7 47.5 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
##  $ sqft_living15: int  2280 1330 2060 1990 2380 1408 2320 1510 1660 1310 .
..
##  $ sqft_lot15   : int  19000 8704 3457 2665 7000 10588 9129 7320 35160 960
0 ...

head(train)

##              id            date  price bedrooms bathrooms sqft_living sqft
_lot
## 7452 7000100635 20140715T000000 600000        3       1.0         940    1
9000
## 8016 3886903155 20150304T000000 606000        3       2.0        1980
7680
## 7162 1036450170 20150312T000000 660000        3       3.5        2740
3785
## 8086 8663240180 20150330T000000 537000        4       2.5        1990
2660
## 9196 7935000625 20150409T000000 975000        3       2.5        2530
7000
## 623  9500900135 20141021T000000 200000        3       1.5        1210    1
0588
##      floors waterfront view condition grade sqft_above sqft_basement yr_bu
ilt
## 7452    1.0          0    0         3     6        940             0     1
945
## 8016    1.5          0    0         4     6       1070           910     1
911
## 7162    2.0          0    0         3     9       2190           550     2
001
## 8086    2.0          0    0         3     8       1990             0     2
012
```

```
## 9196    2.5          0    4         3     9       2530            0     1
915
## 623     1.0          0    0         4     7       1210            0     1
958
##      yr_renovated zipcode     lat     long sqft_living15 sqft_lot15
## 7452            0   98004 47.5828 -122.190          2280      19000
## 8016            0   98033 47.6839 -122.195          1330       8704
## 7162            0   98034 47.7195 -122.182          2060       3457
## 8086            0   98034 47.7320 -122.178          1990       2665
## 9196         1999   98136 47.5465 -122.398          2380       7000
## 623             0   98002 47.2876 -122.212          1408      10588

tail(train)

##               id            date   price bedrooms bathrooms sqft_living
## 6345  7276100020 20150414T000000  505000        4      1.00        1480
## 17565 8127700210 20150427T000000  600000        2      1.75        1560
## 8500  1722059021 20141217T000000  336500        3      2.00        1830
## 1830  7101100055 20150303T000000  753000        3      1.75        2360
## 657   3760500116 20141120T000000 3070000        3      2.50        3930
## 15486 2873000920 20150331T000000  257000        3      1.75        1430
##       sqft_lot floors waterfront view condition grade sqft_above sqft_base
ment
## 6345     12675    1.5          0    0         4     7       1480
0
## 17565     3200    1.0          0    0         5     7        880
680
## 8500     12891    1.0          0    0         3     7       1830
0
## 1830      8290    1.0          0    0         4     7       1180
1180
## 657      55867    1.0          1    4         4     8       2330
1600
## 15486     7210    1.0          0    0         3     7       1430
0
##       yr_built yr_renovated zipcode     lat     long sqft_living15 sqft_lo
t15
## 6345      1929            0   98133 47.7630 -122.342          1820       7
995
## 17565     1946            0   98199 47.6419 -122.394          2060       4
940
## 8500      1994            0   98031 47.3924 -122.192          2320       8
709
## 1830      1950            0   98115 47.6738 -122.281          1880       7
670
## 657       1957            0   98034 47.7022 -122.224          2730      26
324
## 15486     1975            0   98031 47.4189 -122.168          1220       7
777
```

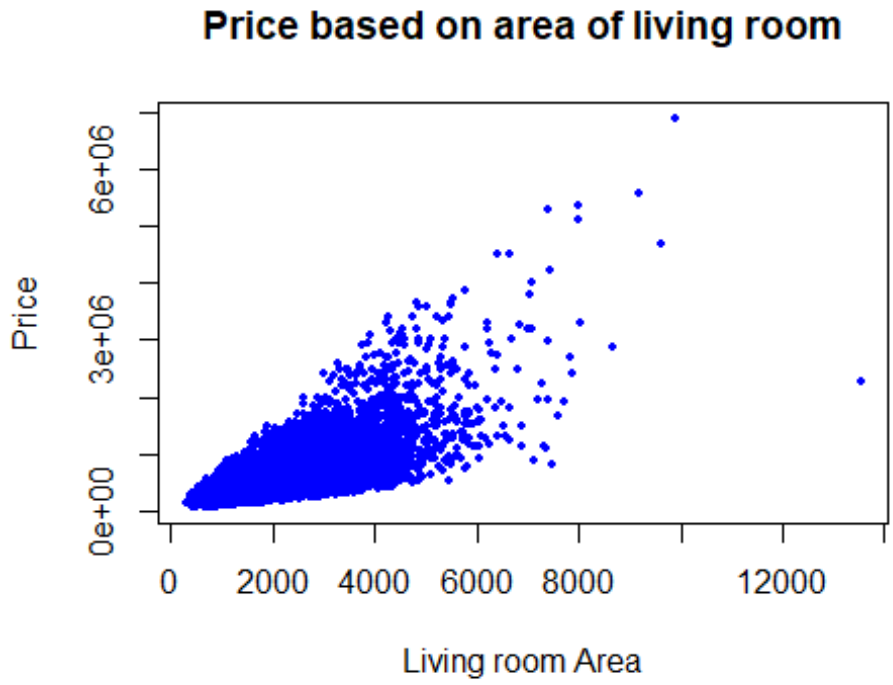
```
sum(is.na(train))
```

```
## [1] 0
```

Some informative graphs

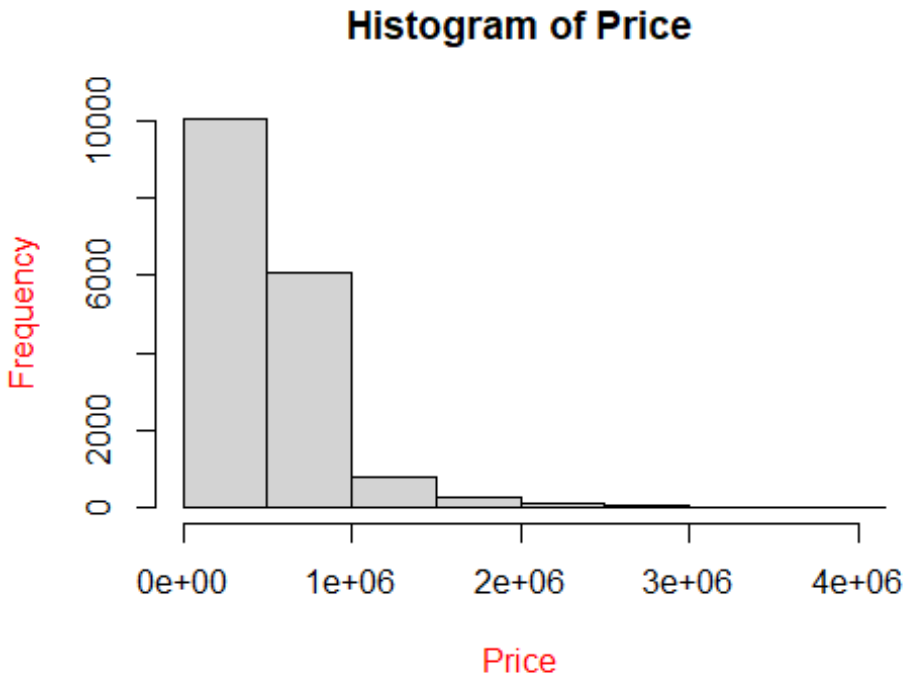
Price vs Area of living room

```
plot(train$sqft_living, train$price, pch = 16, col="blue", cex=0.5, main="Pri
ce based on area of living room", xlab="Living room Area", ylab="Price")
```



Histogram of Price

```
Price <- train$price
hist(Price, col.lab="red", xlim=c(0e+00, 4e+06))
```



***

Comparison of correlation between different parameters

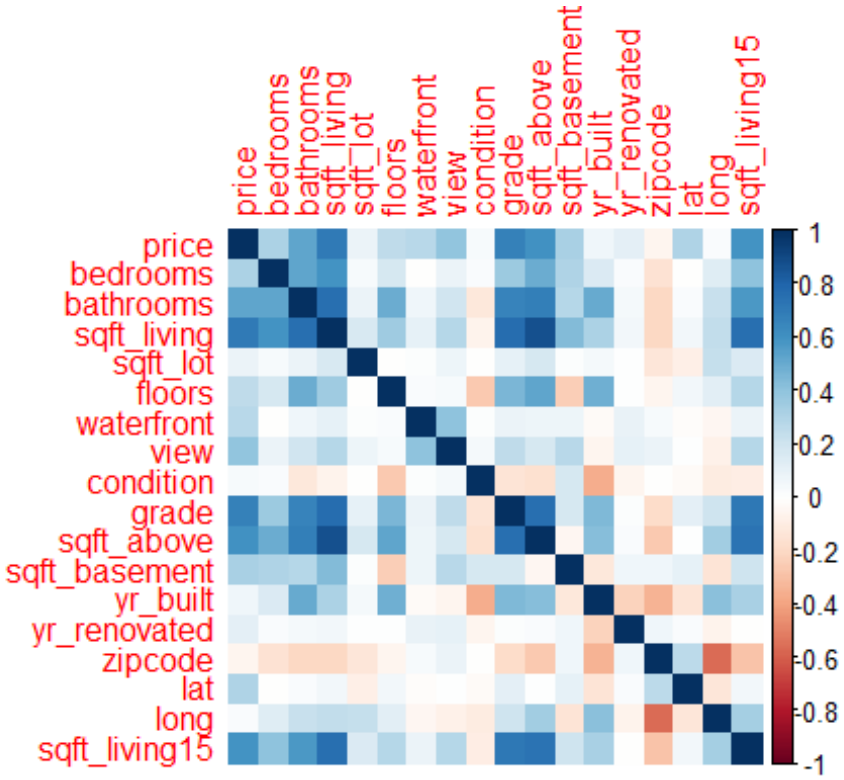
```
#install.packages("corrplot")
library(corrplot)

## corrplot 0.92 loaded

trainData <- train[, 3:20]

M <- cor(trainData)

corrplot(M, method="color")
```



Building a simple linear model

```
lm1 <- lm(price~sqft_above, data=train)
summary(lm1)

##
## Call:
## lm(formula = price ~ sqft_above, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -890409 -165563  -41915  108900 4445909
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 60514.233   5274.673   11.47   <2e-16 ***
## sqft_above    268.462      2.673  100.42   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 291800 on 17288 degrees of freedom
## Multiple R-squared:  0.3684, Adjusted R-squared:  0.3684
## F-statistic: 1.008e+04 on 1 and 17288 DF,  p-value: < 2.2e-16
```

Explanation

Estimates

This means that when a single unit change in x or predictor, changes in Y or target. For example, when finding the linear model of single predictor(sqft_above), this means that 1 sqft_above changes dollars of 268.462

Standard Error

The standard error is estimated error while calculating the coefficients. This is because different sample may have different coefficients. Also, it is a residual standard error divided by the square root of the sum of square of that predictor.

t-value

t-value is the ratio of estimates and standard error. When we have greater t-value, we can go against the null hypothesis, meaning it can have a significance difference.

p-value

Smaller the p-value then they are agains the null hypothesis, which shows they are important. Also the *** shows they are significantly important.

Residual standard error

This is also called standard deviation, which shows how good the model does at predicting price based on the average.

Multiple R-squared and Adjusted R-squared

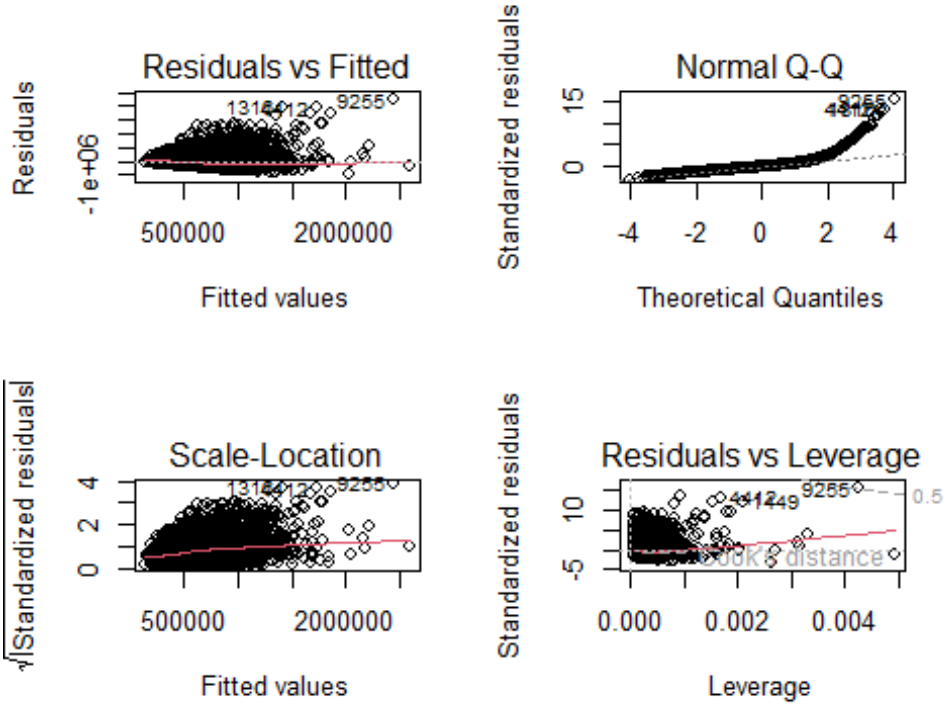
R squared shows to what extent the variance of dependent variable is explained by independent variable. For example, here R-squared is almost 37%, this means 37% of observed variation can be explained by the input model. Adjusted R-squared penalize while adding useless variables.

F-Statistic

It measures the significance of model overall but not with just one variable. For a single predictor, F-value is just a square root of t. And, less the p-value greater the significance as above.

Plot the Residuals

```
par(mfrow=c(2,2))
plot(lm1)
```



Explanation:

Residuals vs Fitted:

Plot shows the residuals have non-linear patterns or not. If we have equally distributed data between horizontal line, this shows that we don't have non-linear relationships. In the figure, the data points is divided by horizontal red line.

Normal Q-Q

If the residuals are normally distributed, we will see a fairly straight diagonal line following the dashed line.

Scale-Location

This shows there is not fairly distributed around the line. Meaning there is not a same variance.

Residuals vs Levarage

This shows the leverage points which are influencing the regression line.Leverage point is a data point with an unusual x-value.

Multiple linear regression

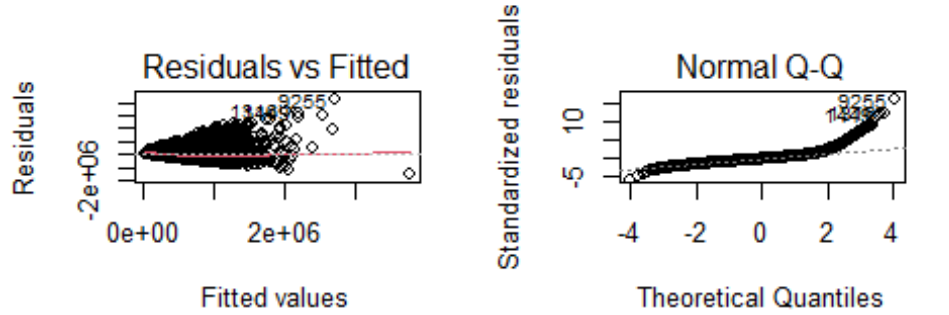
For multiple linear regression, we will be using predictors like: sqft_living, sqft_above, floors with price.

```
lm2 <- lm(price~sqft_living + sqft_above, data=train)
summary(lm2)
```

```
##
## Call:
## lm(formula = price ~ sqft_living + sqft_above, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1483555  -147112   -24599   105375  4178155
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -38051.928   4958.938  -7.673 1.76e-14 ***
## sqft_living    291.929      4.469  65.329  < 2e-16 ***
## sqft_above     -16.059      4.970  -3.231  0.00123 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 261300 on 17287 degrees of freedom
## Multiple R-squared:  0.4935, Adjusted R-squared:  0.4934
## F-statistic:  8420 on 2 and 17287 DF,  p-value: < 2.2e-16
```

Residual plots for multiple linear regression

```
par(mfrow=c(2,2))
plot(lm2)
```



Third Linear regression using different predictors

For this, I am using sqft_living, sqft_lot, sqft_above, yr_built with price.

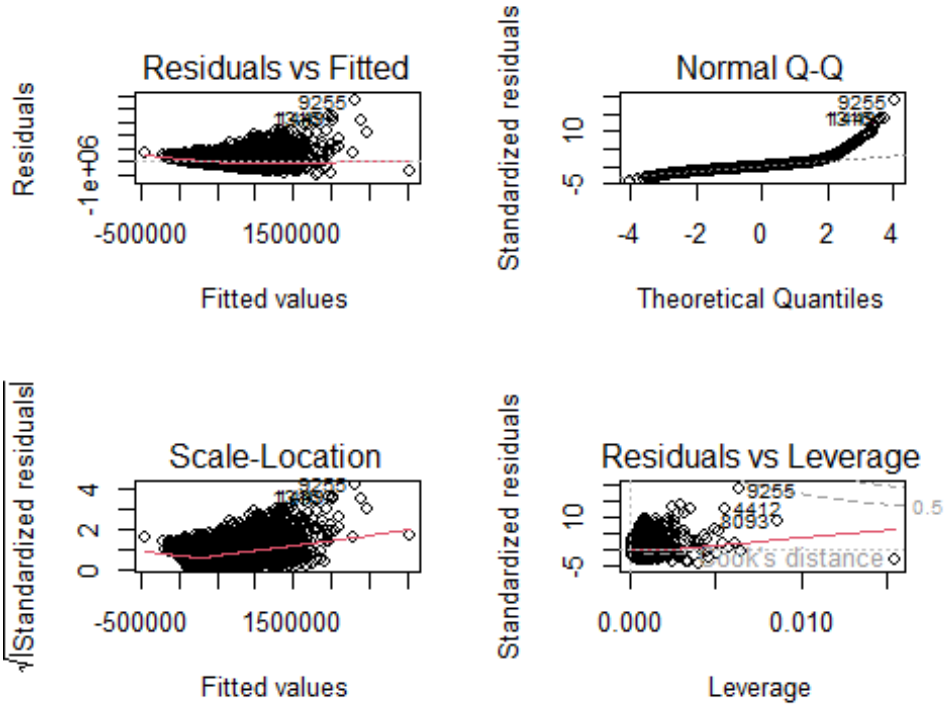
```
lm3 <- lm(price~sqft_living + sqft_above + bathrooms + grade + sqft_living15,
data=train)
summary(lm3)

##
## Call:
## lm(formula = price ~ sqft_living + sqft_above + bathrooms + grade +
##     sqft_living15, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1004866  -135204   -22631    99122  4568033
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -6.527e+05  1.511e+04 -43.191  < 2e-16 ***
## sqft_living    2.369e+02  5.023e+00  47.157  < 2e-16 ***
## sqft_above    -7.980e+01  4.956e+00 -16.101  < 2e-16 ***
## bathrooms     -3.221e+04  3.812e+03  -8.450  < 2e-16 ***
## grade          1.112e+05  2.761e+03  40.262  < 2e-16 ***
## sqft_living15  3.048e+01  4.501e+00   6.773 1.31e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 247500 on 17284 degrees of freedom
## Multiple R-squared:  0.5459, Adjusted R-squared:  0.5458
## F-statistic:  4156 on 5 and 17284 DF,  p-value: < 2.2e-16
```

Residual model for an improved model

```
par(mfrow=c(2,2))
plot(lm3)
```



Adding interaction effects

I have added more interaction effects between sqft_living and sqft_above. Also between sqft_above and bathrooms.

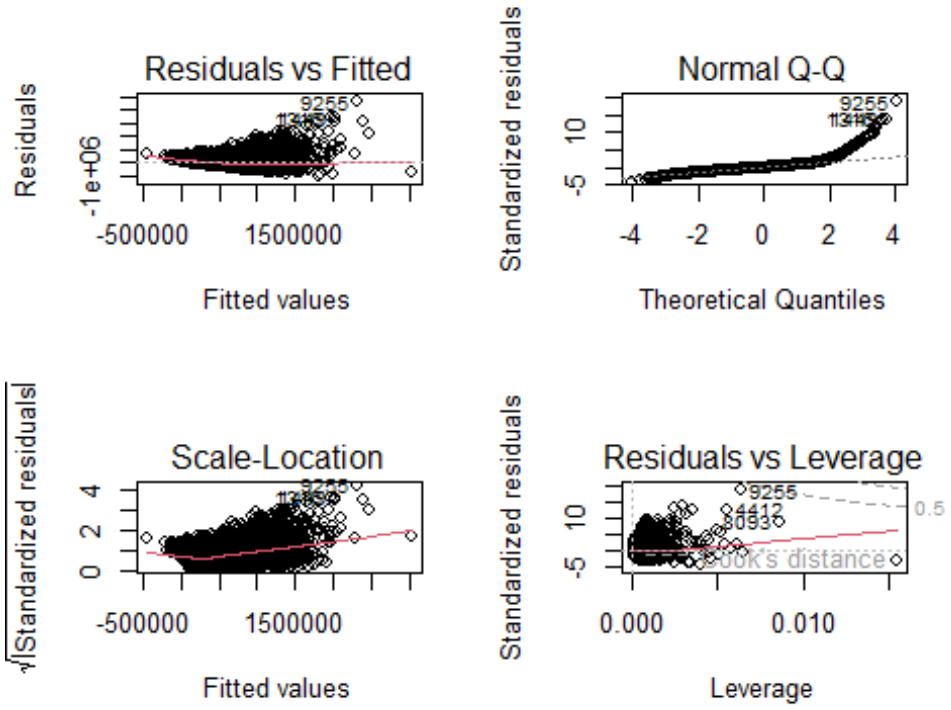
```
lm4 <- lm(price~sqft_living + sqft_above + sqft_living * sqft_above + grade +
bathrooms + sqft_above * bathrooms + sqft_living15, data = train)
summary(lm4)
```

```
##
## Call:
## lm(formula = price ~ sqft_living + sqft_above + sqft_living *
##     sqft_above + grade + bathrooms + sqft_above * bathrooms +
##     sqft_living15, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4266421  -124125   -24749    88222  2600506
```

```
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -3.728e+05  1.607e+04 -23.202   <2e-16 ***
## sqft_living                1.540e+02  7.112e+00  21.654   <2e-16 ***
## sqft_above                -2.813e+02  7.035e+00 -39.980   <2e-16 ***
## grade                      1.171e+05  2.639e+03  44.378   <2e-16 ***
## bathrooms                 -9.921e+04  7.822e+03 -12.683   <2e-16 ***
## sqft_living15              5.290e+01  4.323e+00  12.236   <2e-16 ***
## sqft_living:sqft_above     2.411e-02  2.470e-03   9.763   <2e-16 ***
## sqft_above:bathrooms       4.703e+01  3.666e+00  12.828   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 235700 on 17282 degrees of freedom
## Multiple R-squared:  0.588,  Adjusted R-squared:  0.5879
## F-statistic:  3524 on 7 and 17282 DF,  p-value: < 2.2e-16
```

Residual model for an interactive effects:

```
par(mfrow=c(2,2))
plot(lm3)
```



Different predictions of three models

Simple linear regression

```
pred1 <- predict(lm1, newdata=test)
```

```r
cor1 <- cor(pred1, test$price)

mse1 <- mean((pred1-test$price)^2)
rmse1 <- sqrt(mse1)

print(paste('correlation:', cor1))

## [1] "correlation: 0.599891011331259"

print(paste('mse:',mse1))

## [1] "mse: 86209239482.0612"

print(paste('rmse:', rmse1))

## [1] "rmse: 293614.099596837"
```

Multiple linear regression

```r
pred2 <- predict(lm2, newdata=test)

cor2 <- cor(pred2, test$price)

mse2 <- mean((pred2-test$price)^2)
rmse2 <- sqrt(mse2)

print(paste('correlation:', cor2))

## [1] "correlation: 0.701898655803892"

print(paste('mse:',mse2))

## [1] "mse: 68366274164.356"

print(paste('rmse:', rmse2))

## [1] "rmse: 261469.451684812"
```

Adding more predictors

```r
pred3 <- predict(lm3, newdata=test)

cor3 <- cor(pred3, test$price)

mse3 <- mean((pred3-test$price)^2)
rmse3 <- sqrt(mse3)

print(paste('correlation:', cor3))

## [1] "correlation: 0.732840773320671"

print(paste('mse:',mse3))

## [1] "mse: 62376336140.9964"
```

```
print(paste('rmse:', rmse3))
```

```
## [1] "rmse: 249752.549818808"
```

Adding interaction effects

```
pred4 <- predict(lm4, newdata=test)

cor4 <- cor(pred4, test$price)

mse4 <- mean((pred4-test$price)^2)
rmse4 <- sqrt(mse4)

print(paste('correlation:', cor4))
```

```
## [1] "correlation: 0.789144340928874"
```

```
print(paste('mse:',mse4))
```

```
## [1] "mse: 51029093219.0618"
```

```
print(paste('rmse:', rmse4))
```

```
## [1] "rmse: 225896.200098766"
```

Comparison of different models

Comparing first and second model

```
anova(lm1, lm2)
```

```
## Analysis of Variance Table
##
## Model 1: price ~ sqft_above
## Model 2: price ~ sqft_living + sqft_above
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
## 1  17288 1.4720e+15
## 2  17287 1.1806e+15  1 2.9146e+14 4267.9 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we see that RSS is lower for model 2 compared to that of model 1. Furthermore, p-value for model 2 is less. This shows model 2 is better than model 1. Since, we found out model 2 is better, now we can compare with third model.

Comparing second and third model

```
anova(lm2,lm3)
```

```
## Analysis of Variance Table
##
## Model 1: price ~ sqft_living + sqft_above
## Model 2: price ~ sqft_living + sqft_above + bathrooms + grade + sqft_livin
g15
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
```

```
## 1  17287 1.1806e+15
## 2  17284 1.0584e+15  3 1.2219e+14 665.16 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Similarly, here we found out RSS for model 3 is lesser than model 2, and similarly p-value is less. This shows that model 3 is better than model 2. Finally, we can conclude that model 3 is better than other model.

Comparing third and fourth model

```
anova(lm3,lm4)
```

```
## Analysis of Variance Table
##
## Model 1: price ~ sqft_living + sqft_above + bathrooms + grade + sqft_livin
g15
## Model 2: price ~ sqft_living + sqft_above + sqft_living * sqft_above +
##     grade + bathrooms + sqft_above * bathrooms + sqft_living15
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
## 1  17284 1.0584e+15
## 2  17282 9.6013e+14  2 9.8253e+13 884.26 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova function shows that lm4 is better than lm3 because it has low RSS and p-value. Thus, we got better model while adding interaction effect on our model. I tried with polynomial regression too with the model but it was good while using an interaction effect.

Explanation

We got model 4 as a good model. It is a model where we have added interaction effect. This is because there is a synergy between the predictors too. We know that living room area and area of lot have correlation between them, which is shown in correlation map too. This variable are also called confounding variables, meaning that hava a relationship or correlation between other predictors and target variables.