DSC520_Week8_9_AssignmentPart3_Guruprasad_VelikaduKrishnamoorthy

Guruprasad Velikadu Krishnamoorthy

2023-02-12

```r
# Calling the Libraries used
library(readxl, quietly = TRUE)
library(dplyr, quietly = TRUE)
```

```
##
## 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
```

```r
library(lubridate, quietly = TRUE)
```

```
##
## Attaching package: 'lubridate'

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

```r
library(magrittr, quietly = TRUE)
library(olsrr, quietly = TRUE)
```

```
##
## Attaching package: 'olsrr'

## The following object is masked from 'package:datasets':
##
##     rivers
```

```r
library(QuantPsyc, quietly = TRUE)
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:magrittr':
##
##     set_names

##
## Attaching package: 'MASS'

## The following object is masked from 'package:olsrr':
##
##     cement

## The following object is masked from 'package:dplyr':
##
##     select

##
## Attaching package: 'QuantPsyc'

## The following object is masked from 'package:base':
##
##     norm
```

```r
library(relaimpo, quietly = TRUE)
```

```
##
## Attaching package: 'survival'

## The following object is masked from 'package:boot':
##
##     aml

##
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':
##
##     dotchart

## This is the global version of package relaimpo.

## If you are a non-US user, a version with the interesting additional metric pmvd is available

## from Ulrike Groempings web site at prof.beuth-hochschule.de/groemping.
```

```r
library(car, quietly = TRUE)
```

```
##
## Attaching package: 'car'

## The following object is masked from 'package:purrr':
##
##     some

## The following object is masked from 'package:boot':
##
##     logit

## The following object is masked from 'package:dplyr':
##
##     recode
```

Assignment Part-3 (Housing Dataset)

i. Explain any transformations or modifications you made to the dataset

```r
# loading the housing dataset
excel_path <- "data/week-6-housing.xlsx"
housing_data_df_all <- read_excel(excel_path)
# Examining the structure and summary
str(housing_data_df_all)
```

```
## tibble [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE 
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```r
summary(housing_data_df_all)
```

```
##    Sale Date                       Sale Price       sale_reason
##  Min.   :2006-01-03 00:00:00.00   Min.   :    698   Min.   : 0.00
##  1st Qu.:2008-07-07 00:00:00.00   1st Qu.: 460000   1st Qu.: 1.00
##  Median :2011-11-17 00:00:00.00   Median : 593000   Median : 1.00
##  Mean   :2011-07-28 15:07:32.48   Mean   : 660738   Mean   : 1.55
##  3rd Qu.:2014-06-05 00:00:00.00   3rd Qu.: 750000   3rd Qu.: 1.00
##  Max.   :2016-12-16 00:00:00.00   Max.   :4400000   Max.   :19.00
##  sale_instrument  sale_warning         sitetype           addr_full
##  Min.   : 0.000   Length:12865       Length:12865       Length:12865
##  1st Qu.: 3.000   Class :character   Class :character   Class :character
##  Median : 3.000   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 3.678
##  3rd Qu.: 3.000
##  Max.   :27.000
##       zip5         ctyname           postalctyn             lon
##  Min.   :98052   Length:12865       Length:12865       Min.   :-122.2
##  1st Qu.:98052   Class :character   Class :character   1st Qu.:-122.1
##  Median :98052   Mode  :character   Mode  :character   Median :-122.1
##  Mean   :98053                                         Mean   :-122.1
##  3rd Qu.:98053                                         3rd Qu.:-122.0
##  Max.   :98074                                         Max.   :-121.9
##       lat        building_grade  square_feet_total_living    bedrooms
##  Min.   :47.46   Min.   : 2.00   Min.   :  240            Min.   : 0.000
##  1st Qu.:47.67   1st Qu.: 8.00   1st Qu.: 1820            1st Qu.: 3.000
##  Median :47.69   Median : 8.00   Median : 2420            Median : 4.000
##  Mean   :47.68   Mean   : 8.24   Mean   : 2540            Mean   : 3.479
##  3rd Qu.:47.70   3rd Qu.: 9.00   3rd Qu.: 3110            3rd Qu.: 4.000
##  Max.   :47.73   Max.   :13.00   Max.   :13540            Max.   :11.000
##  bath_full_count  bath_half_count  bath_3qtr_count   year_built
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.000   Min.   :1900
##  1st Qu.: 1.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:1979
##  Median : 2.000   Median :1.0000   Median :0.000   Median :1998
##  Mean   : 1.798   Mean   :0.6134   Mean   :0.494   Mean   :1993
##  3rd Qu.: 2.000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:2007
##  Max.   :23.000   Max.   :8.0000   Max.   :8.000   Max.   :2016
##  year_renovated    current_zoning       sq_ft_lot         prop_type
##  Min.   :   0.00   Length:12865       Min.   :    785   Length:12865
##  1st Qu.:   0.00   Class :character   1st Qu.:   5355   Class :character
##  Median :   0.00   Mode  :character   Median :   7965   Mode  :character
##  Mean   :  26.24                      Mean   :  22229
##  3rd Qu.:   0.00                      3rd Qu.:  12632
##  Max.   :2016.00                      Max.   :1631322
##   present_use
##  Min.   :  0.000
##  1st Qu.:  2.000
##  Median :  2.000
##  Mean   :  6.598
##  3rd Qu.:  2.000
##  Max.   :300.000
```

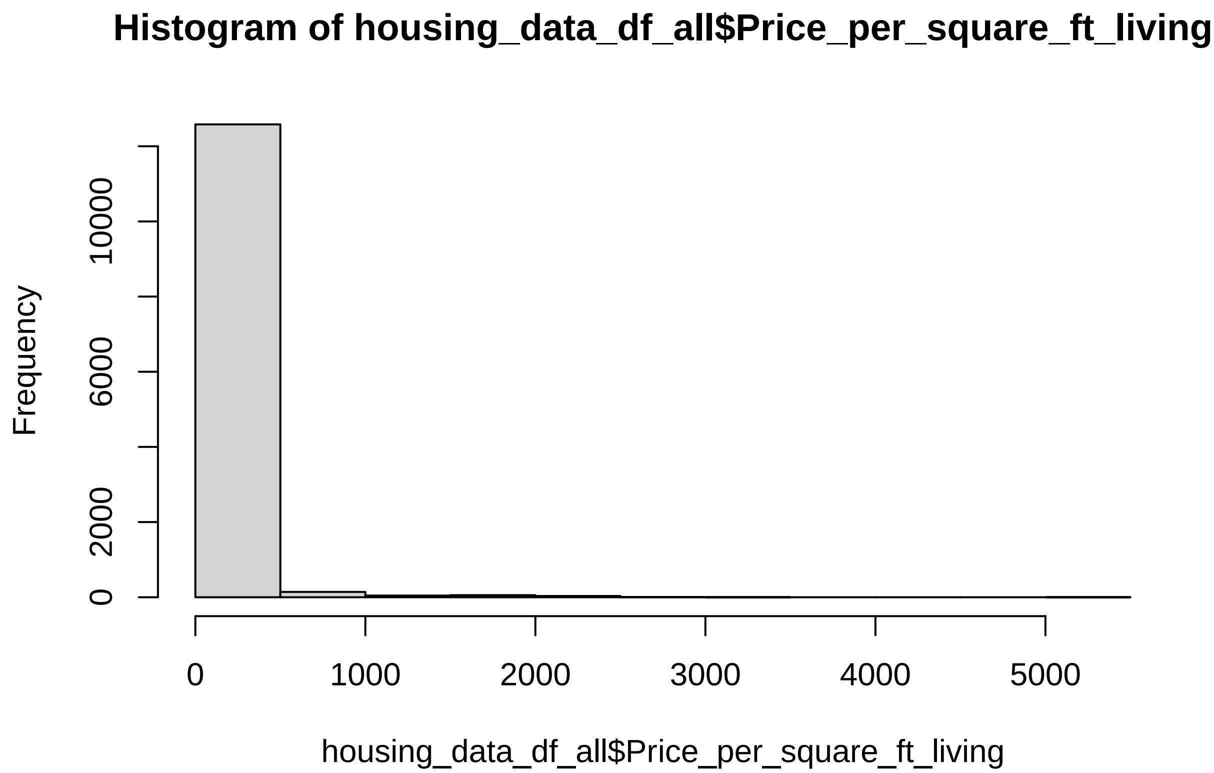
```r
nrow(housing_data_df_all)
```

```
## [1] 12865
```

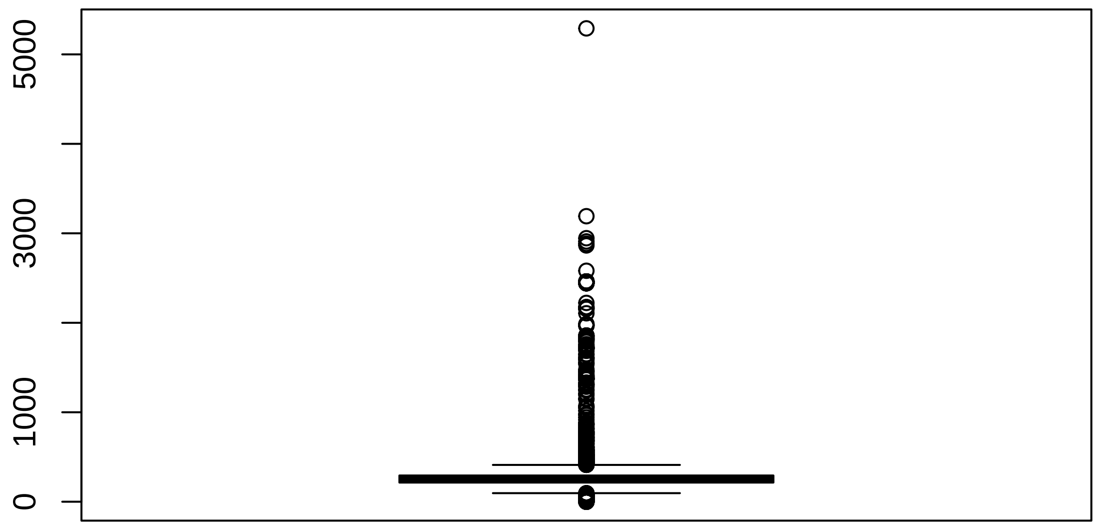
```r
# As some of the columns can be factors, converting the columns as factors

housing_data_df_all$sitetype <- as.factor(housing_data_df_all$sitetype)
housing_data_df_all$zip5 <- as.factor(housing_data_df_all$zip5)
housing_data_df_all$postalctyn <- as.factor(housing_data_df_all$postalctyn)
housing_data_df_all$current_zoning <- as.factor(housing_data_df_all$current_zoning)
housing_data_df_all$prop_type <- as.factor(housing_data_df_all$prop_type)
# Renaming columns for easy usage
housing_data_df_all <- housing_data_df_all %>%
    rename(Sale_Date = "Sale Date")
housing_data_df_all <- housing_data_df_all %>%
    rename(Sale_Price = "Sale Price")
# Transforming and creating new_columns. 2 new columns for Price per square foot are
# calculated
housing_data_df_all$Price_per_square_ft_living <- with(housing_data_df_all, Sale_Price/square_feet_total
housing_data_df_all$Price_per_square_ft_lot <- with(housing_data_df_all, Sale_Price/sq_ft_lot)
nrow(housing_data_df_all)
```

```
## [1] 12865
```

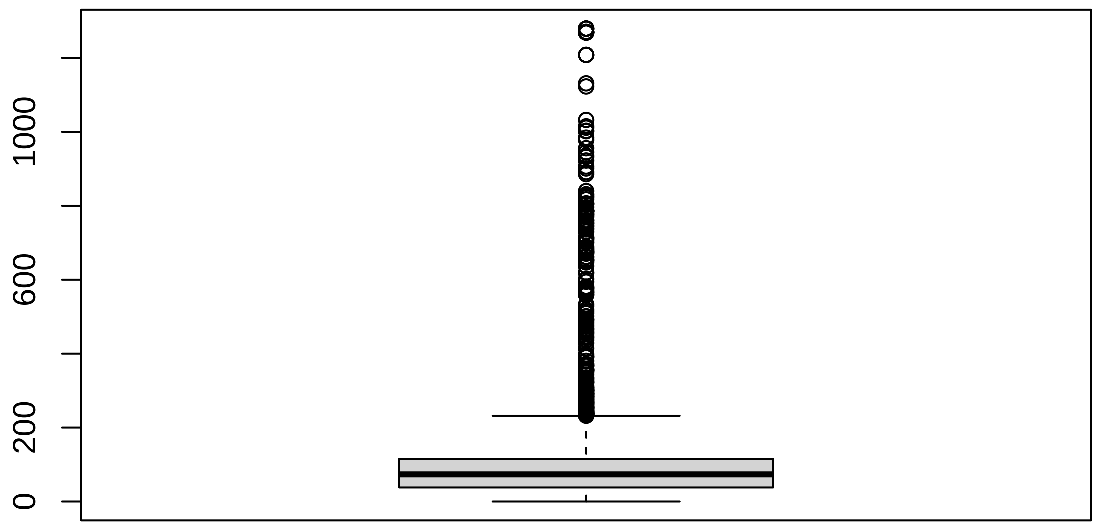
```r
# Identifying the outliers and cleaning the dataset.
hist(housing_data_df_all$Price_per_square_ft_living)
```



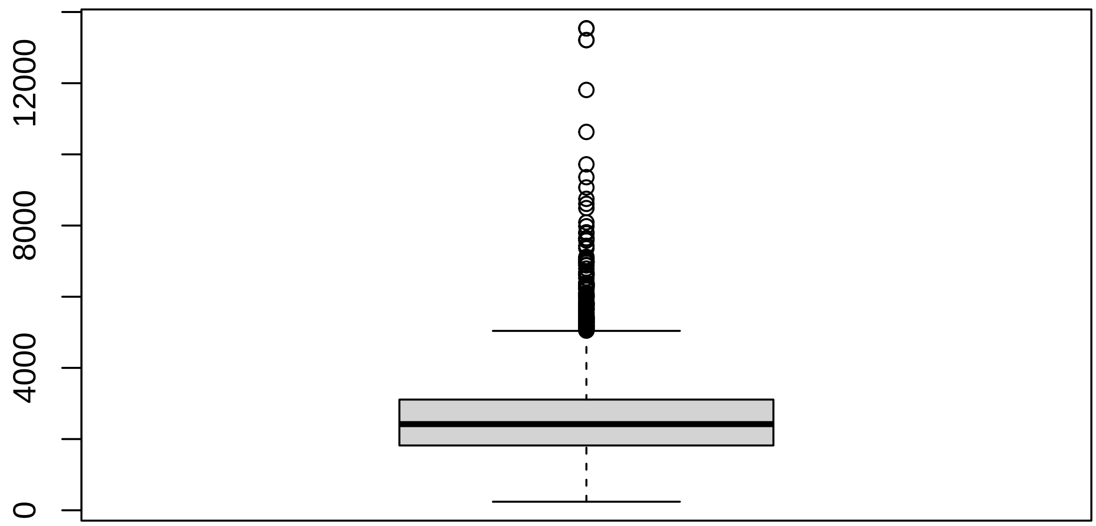

```r
# Box plot is used to identify the outliers in the dataset. They were used on the newly
# created PricePerSqFt fields
boxplot(housing_data_df_all$Price_per_square_ft_living)
```



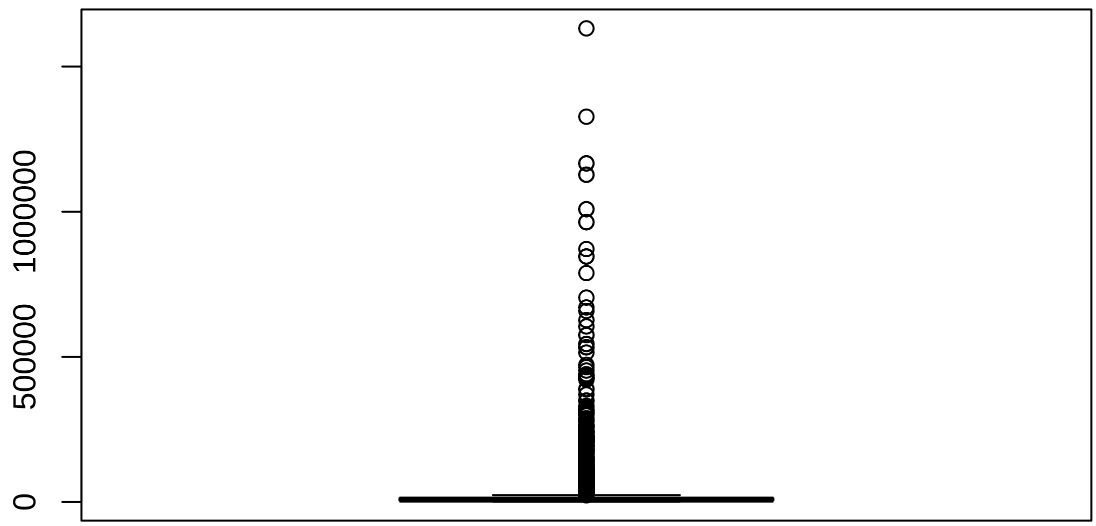

```
boxplot(housing_data_df_all$Price_per_square_ft_lot)
```




```
# Box plot is also used on the columns that can be used as predictors
boxplot(housing_data_df_all$square_feet_total_living)
```




```r
boxplot(housing_data_df_all$sq_ft_lot)
```




```
# Identifying the Outliers using boxplot.stats function

housing_data_df_all <- housing_data_df_all[!housing_data_df_all$Price_per_square_ft_living %in%
    boxplot.stats(housing_data_df_all$Price_per_square_ft_living)$out, ]

housing_data_df_all <- housing_data_df_all[!housing_data_df_all$Price_per_square_ft_lot %in%
    boxplot.stats(housing_data_df_all$Price_per_square_ft_lot)$out, ]


housing_data_df_all <- housing_data_df_all[!housing_data_df_all$square_feet_total_living %in%
    boxplot.stats(housing_data_df_all$square_feet_total_living)$out, ]

housing_data_df_all <- housing_data_df_all[!housing_data_df_all$sq_ft_lot %in% boxplot.stats(housing_da
    ]

# Filtering and cleansing the data based on the results from box plot to have a
# reasonable range of data from the dataset


housing_data_df_all <- housing_data_df_all %>%
    filter(zip5 %in% c("98052", "98053"))
housing_data_df_all <- housing_data_df_all[housing_data_df_all$Price_per_square_ft_living >=
    150 & housing_data_df_all$Price_per_square_ft_living <= 300, ]
housing_data_df_all <- housing_data_df_all[housing_data_df_all$Sale_Price <= 1e+06 & housing_data_df_all
    50000, ]
housing_data_df_all <- housing_data_df_all[housing_data_df_all$square_feet_total_living >=
```

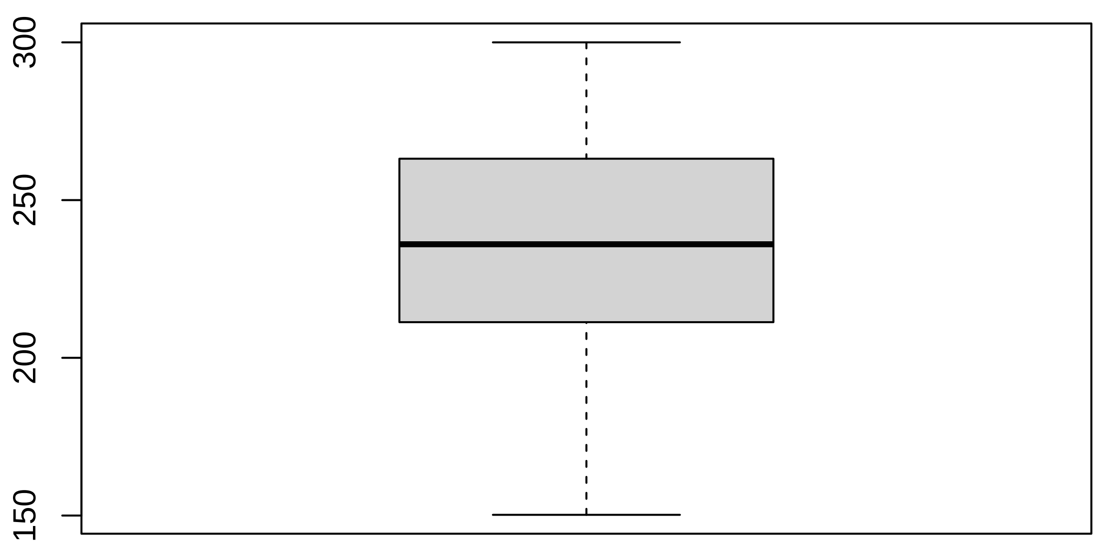
```
    0 & housing_data_df_all$square_feet_total_living <= 4400, ]
housing_data_df_all <- housing_data_df_all[housing_data_df_all$Price_per_square_ft_lot >= 30 &
    housing_data_df_all$Price_per_square_ft_lot <= 210, ]
housing_data_df_all <- housing_data_df_all[housing_data_df_all$sq_ft_lot >= 0 & housing_data_df_all$sq_
    14500, ]
housing_data_df_all <- housing_data_df_all[housing_data_df_all$bath_full_count <= 10, ]
nrow(housing_data_df_all)
```

```
## [1] 7076
```

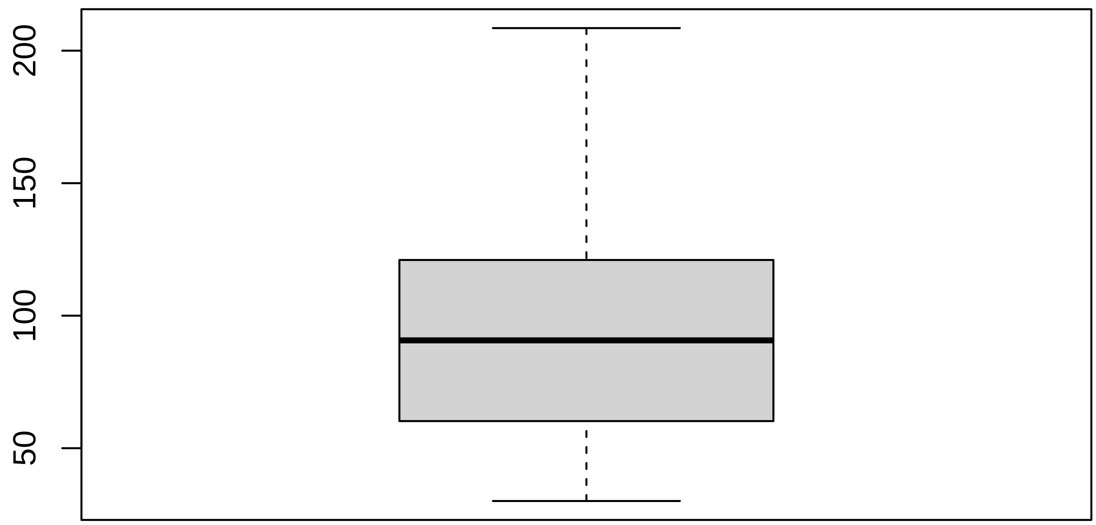
```
# creating a sample dataset from the dataset that was cleansed.
set.seed(40)  # Use seed 40
# created new Dataframe of sample size 3000 for further use
housing_data_df <- housing_data_df_all[sample(nrow(housing_data_df_all), size = 3000), ]
nrow(housing_data_df)
```

```
## [1] 3000
```

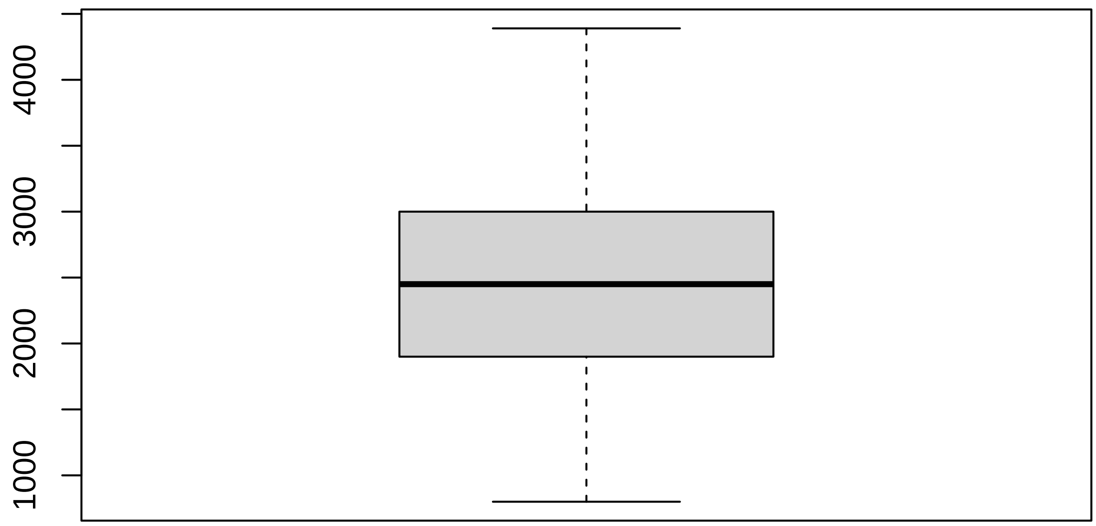
```
# Examine the results after cleansing the data and sampleing the data.  Box plot is used
# to identify the outliers in the dataset. They were used on the newly created
# PricePerSqFt fields
boxplot(housing_data_df$Price_per_square_ft_living)
```



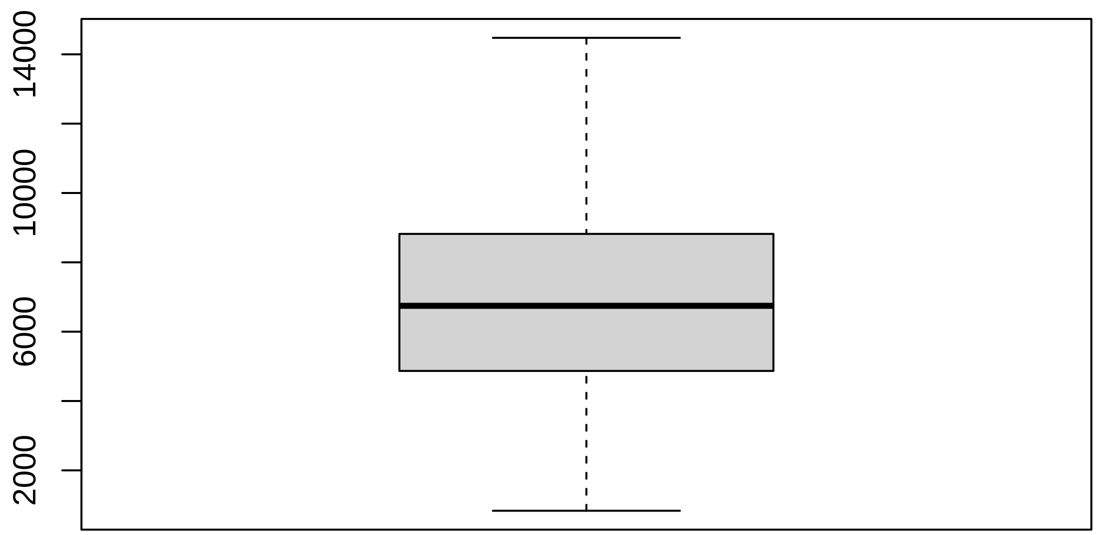
```r
boxplot(housing_data_df$Price_per_square_ft_lot)
```




```r
# Box plot is also used on the columns that can be used as predictors
boxplot(housing_data_df$square_feet_total_living)
```




```r
boxplot(housing_data_df$sq_ft_lot)
```



ii. Create two variables; one that will contain the variables Sale Price and Square Foot of Lot (same variables used from previous assignment on simple regression) and one that will contain Sale Price and several additional predictors of your choice. Explain the basis for your additional predictor selections

```r
# creating the first model with only sq_ft_lot as Predictor
housing_lm1 <- lm(Sale_Price ~ sq_ft_lot, data = housing_data_df, na.action = na.omit)
summary(housing_lm1)
```

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot, data = housing_data_df,
##     na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -386971 -127059  -11295  115385  434134
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.181e+05  8.308e+03  62.364  < 2e-16 ***
## sq_ft_lot   8.431e+00  1.110e+00   7.597 4.03e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 160900 on 2998 degrees of freedom
```

```
## Multiple R-squared:  0.01889,    Adjusted R-squared:  0.01856
## F-statistic: 57.71 on 1 and 2998 DF,  p-value: 4.028e-14
```

```r
# Creating a Regression model with possible 7 fields that can be a good predictor and
# testing the results using olsrr package Assumption: The assumption with building grade
# is, the higher the number better the grade
housing_lm3 <- lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + bedrooms +
    bath_full_count + bath_half_count + year_built, data = housing_data_df, na.action = na.omit)
# creating all models using ols_step_all_possible function and plotting the results
all.mod <- ols_step_all_possible(model = housing_lm3)
head(all.mod, n = 5)
```

```
##   Index N               Predictors  R-Square Adj. R-Square Mallow's Cp
## 2     1 1 square_feet_total_living 0.7461908     0.7461061    481.7821
## 3     2 1           building_grade 0.4587207     0.4585402   4420.7965
## 4     3 1                 bedrooms 0.2161972     0.2159358   7743.9376
## 5     4 1          bath_full_count 0.1882362     0.1879654   8127.0694
## 7     5 1               year_built 0.1761399     0.1758650   8292.8176
```

```r
# plot(all.mod) Finding the best set of predictors and plotting the results
best.mod <- ols_step_best_subset(model = housing_lm3)
best.mod
```

```
##                                                  Best Subsets Regression
## --------------------------------------------------------------------------------------------------------
## Model Index    Predictors
## --------------------------------------------------------------------------------------------------------
##      1         square_feet_total_living
##      2         square_feet_total_living building_grade
##      3         square_feet_total_living building_grade year_built
##      4         sq_ft_lot square_feet_total_living building_grade year_built
##      5         sq_ft_lot square_feet_total_living building_grade bath_half_count year_built
##      6         sq_ft_lot square_feet_total_living building_grade bath_full_count bath_half_count yea
##      7         sq_ft_lot square_feet_total_living building_grade bedrooms bath_full_count bath_half_
## --------------------------------------------------------------------------------------------------------
##
##                                                                    Subsets Regression Summary
## --------------------------------------------------------------------------------------------------------
##                        Adj.        Pred
## Model      R-Square    R-Square    R-Square      C(p)         AIC           SBIC           SBC
## --------------------------------------------------------------------------------------------------------
##   1          0.7462      0.7461      0.7458    481.7821    76393.3936    67879.1751    76411.4127    2
##   2          0.7742      0.7740      0.7737    100.3828    76044.9747    67531.1586    76069.0002    1
##   3          0.7773      0.7770      0.7766     60.0932    76005.6922    67491.9239    76035.7241    1
##   4          0.7806      0.7803      0.7798     16.6201    75962.6589    67449.0057    75998.6971    1
##   5          0.7816      0.7812      0.7807      4.5504    75950.5873    67436.9859    75992.6318    1
##   6          0.7816      0.7812      0.7806      6.0024    75952.0379    67438.4441    76000.0888    1
##   7          0.7816      0.7811      0.7804      8.0000    75954.0354    67440.4470    76008.0927    1
## --------------------------------------------------------------------------------------------------------
## AIC: Akaike Information Criteria
##  SBIC: Sawa's Bayesian Information Criteria
##  SBC: Schwarz Bayesian Criteria
##  MSEP: Estimated error of prediction, assuming multivariate normality
```

```
##  FPE: Final Prediction Error
##  HSP: Hocking's Sp
##  APC: Amemiya Prediction Criteria
```

```r
# plot(best.mod) Confirming the results by executing forward, backward & stepwise methods
ols_step_forward_p(model = housing_lm3, details = FALSE)
```

```
##
##                                        Selection Summary
## -----------------------------------------------------------------------------------------------
##         Variable                                   Adj.
## Step            Entered                R-Square    R-Square     C(p)         AIC           RMSE
## -----------------------------------------------------------------------------------------------
##    1    square_feet_total_living         0.7742      0.7740    100.3828    76044.9747    77218.2288
##    2    building_grade                   0.7773      0.7770     60.0932    76005.6922    76701.5565
##    3    year_built                       0.7806      0.7803     16.6201    75962.6589    76140.7319
##    4    sq_ft_lot                        0.7816      0.7812      4.5504    75950.5873    75975.0554
##    5    bath_half_count                      NA          NA          NA            NA            NA
## -----------------------------------------------------------------------------------------------
```

```r
ols_step_backward_p(model = housing_lm3, details = FALSE)
```

```
##
##
##                                  Elimination Summary
## ------------------------------------------------------------------------------------
##         Variable                            Adj.
## Step        Removed        R-Square    R-Square     C(p)        AIC           RMSE
## ------------------------------------------------------------------------------------
##    1    bedrooms             0.7816      0.7812    6.0024    75952.0379    75980.7887
##    2    bath_full_count      0.7816      0.7812    4.5504    75950.5873    75975.0554
## ------------------------------------------------------------------------------------
```

```r
ols_step_both_p(model = housing_lm3, details = FALSE)
```

```
##
##                                          Stepwise Selection Summary
## ----------------------------------------------------------------------------------------------------
##                                         Added/                  Adj.
## Step             Variable               Removed     R-Square    R-Square     C(p)         AIC
## ----------------------------------------------------------------------------------------------------
##    1    square_feet_total_living      addition        0.774       0.774    100.3830    76044.9747    772
##    2         building_grade           addition        0.777       0.777     60.0930    76005.6922    767
##    3           year_built             addition        0.781       0.780     16.6200    75962.6589    761
##    4           sq_ft_lot              addition        0.782       0.781      4.5500    75950.5873    759
## ----------------------------------------------------------------------------------------------------
```

```r
# Solution: The results of the best.mod indicates that the values of R2 changes very
# little after adding 5 Predictors. Also results of Stepwise selection indicates using 4
# predictors: sq_ft_lot square_feet_total_living building_grade year_built.  creating
# final model based on the above results
housing_lm5 <- lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + year_built,
    data = housing_data_df, na.action = na.omit)
```

iii. Execute a summary() function on two variables defined in the previous step to compare the model results. What are the R2 and Adjusted R2 statistics? Explain what these results tell you about the overall model. Did the inclusion of the additional predictors help explain any large variations found in Sale Price?

```r
# Comparing the results between first and final model created
summary(housing_lm1)
```

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot, data = housing_data_df,
##     na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -386971 -127059  -11295  115385  434134
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.181e+05  8.308e+03  62.364  < 2e-16 ***
## sq_ft_lot   8.431e+00  1.110e+00   7.597 4.03e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 160900 on 2998 degrees of freedom
## Multiple R-squared:  0.01889,    Adjusted R-squared:  0.01856
## F-statistic: 57.71 on 1 and 2998 DF,  p-value: 4.028e-14
```

```r
summary(housing_lm5)
```

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot + square_feet_total_living +
##     building_grade + year_built, data = housing_data_df, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -286513  -53569    -130   54780  234662
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.620e+06  2.611e+05 -10.037  < 2e-16 ***
## sq_ft_lot                 4.950e+00  7.355e-01   6.730 2.02e-11 ***
## square_feet_total_living  1.583e+02  2.776e+00  57.041  < 2e-16 ***
## building_grade            3.684e+04  2.266e+03  16.263  < 2e-16 ***
## year_built                1.239e+03  1.326e+02   9.343  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 76140 on 2995 degrees of freedom
## Multiple R-squared:  0.7806, Adjusted R-squared:  0.7803
## F-statistic:  2664 on 4 and 2995 DF,  p-value: < 2.2e-16
```

```r
impacts = calc.relimp(housing_lm5, type = "lmg")
impacts
```

```
## Response variable: Sale_Price
## Total response variance: 26385836362
## Analysis based on 3000 observations
##
## 4 Regressors:
## sq_ft_lot square_feet_total_living building_grade year_built
## Proportion of variance explained by model: 78.06%
## Metrics are not normalized (rela=FALSE).
##
## Relative importance metrics:
##
##                                 lmg
## sq_ft_lot                0.02750090
## square_feet_total_living 0.46349878
## building_grade           0.20321323
## year_built               0.08636337
##
## Average coefficients for different model sizes:
##
##                                   1X         2Xs          3Xs          4Xs
## sq_ft_lot                8.431351e+00    12.37567     8.784936     4.950338
## square_feet_total_living 1.998261e+02   186.31775   170.550275   158.336704
## building_grade           1.333563e+05 97934.18725 59625.999320 36844.010662
## year_built               4.239071e+03  3351.29202  2165.084548  1239.170871
```

```r
# The above relimp command explains the impact of each predictor in the final R2 metric.
# The results indicate, sq_dt_lot contributes 2.75% and square_feet_total_living
# contributes 46.34%, building_grade contributes 20.3% and year_built contributes 8.63%
# of the variance in the Sales price. It all sums up to the total R2 contribution of
# 78.06%
```

```r
# Solution: The multiple R2 metrics has improved from 0.01889 for 1 predictor to 0.7806
# with 4 predictors used in the final mode. This tells us the variable sq_ft_lot accounts
# for 1.89% variation in the Home sale price. Whereas the 4 new predictors in the final
# model can account for 78.06% of variation in the Home sale price which is a significant
# Improvement in the results.

# The Adjusted R2 indicates the shrinkage also known as the loss of predictive power of
# the model. The Adjusted R2 tells us how much variance in the Sales Price can be
# accounted for if the model was derived from the original dataset from which the data
# was sampled. The adjusted R2 for the final model (0.7806) is pretty close to the
# Multiple R2(0.7803) which indicates the model can be a good predictor for any other
# samples derived from the Housing dataset.

# The p-value in the summary of final model is significantly less than zero. So it
# indicates that the Null hypotheses of no model exists can be rejected and the
# Regression model housing_lm5 can be accepted as a good predictor for Sale Price. The
# inclusion of additional predictors explain large variation in the sales price. The
# Final Equation look like below :
```

```r
# Sale_Price$ = $(-2.620e+06) + (4.950e+00 * sq_ft_lot) + (1.583e+02 *
# square_feet_total_living) + (3.684e+04 * building_grade) + (1.239e+03 * year_built)
```

iv. Considering the parameters of the multiple regression model you have created. What are the standardized betas for each parameter and what do the values indicate?

```r
lm.beta(housing_lm5)
```

```
##                sq_ft_lot square_feet_total_living           building_grade
##                0.0806898                0.6844700                0.1871231
##               year_built
##                0.1226844
```

```r
# Calculating the Standard deviation for each predictor and the impact it has on the
# Sales Price
sd(housing_data_df$sq_ft_lot)
```

```
## [1] 2647.703
```

```r
sd(housing_data_df$Sale_Price) * 0.0806898
```

```
## [1] 13107.02
```

```r
sd(housing_data_df$square_feet_total_living)
```

```
## [1] 702.1958
```

```r
sd(housing_data_df$Sale_Price) * 0.68447
```

```
## [1] 111183.4
```

```r
sd(housing_data_df$building_grade)
```

```
## [1] 0.8249849
```

```r
sd(housing_data_df$Sale_Price) * 0.1871231
```

```
## [1] 30395.75
```

```r
sd(housing_data_df$year_built)
```

```
## [1] 16.08213
```

```r
sd(housing_data_df$Sale_Price) * 0.1226844
```

```
## [1] 19928.51
```

```
# Solution: The standardized beta indicates the measure if the standard deviation of the
# Predictor changes by one standard Deviation, how many standard deviations it will
# change in the Outcome variable.

# sq_ft_lot: If the sq_ft_lot changes by 1 std deviation , the Sales price will crease by
# 0.0806898 std deviation. To put it in numbers, if the sq_ft_lot increases by 2647.703
# sq.ft, the Sales price increases by (162437.2 * 0.0806898) $13,107.02

# square_feet_total_living : If the square_feet_total_living changes by 1 std deviation,
# the Sales price will increase by 0.6844700 std deviation. To put it in numbers, if the
# square_feet_total_living increases by 702.1958 sq.ft, the Sales price increases by
# (162437.2 * 0.6844700) $111,183.4

# building_grade : If the building_grade changes by 1 std deviation, the Sales price will
# increase by 0.18745339 std deviation. To put it in numbers, if the building_grade
# increases by 0.1871231, the Sales price increases by (162437.2 * 0.18745339) $30,395.75

# year_built: If the year_built changes by 1 std deviation, the Sales price will increase
# by 0.1226844 std deviation. To put it in numbers, if the year_built increases by
# 16.08213 years, the Sales price increases by (162437.2 * 0.1226844) $19,928.51
```

v. Calculate the confidence intervals for the parameters in your model and explain what the results indicate.

```
confint(housing_lm5)
```

```
##                                  2.5 %        97.5 %
## (Intercept)              -3.132323e+06 -2.108475e+06
## sq_ft_lot                 3.508150e+00  6.392526e+00
## square_feet_total_living  1.528940e+02  1.637794e+02
## building_grade            3.240179e+04  4.128624e+04
## year_built                9.791004e+02  1.499241e+03
```

```
# Explanation: The results indicates that if we were to take 100 samples from the housing
# dataset and calculated the confidence intervals, 95% of the confidence intervals would
# contain the true value of the regression coefficients. All the Predictors have positive
# value which indicates the direction of relationship which is positive .Also none of the
# predictors have coefficients crossing zero.  The 2 predictors square_feet_total_living
# and building grade have tight confidence interval which indicates their estimates are
# likely to be truly representative of the final model. The other predictors sq_ft_lot
# and year_built have fairly larger CI that indicates they are of lesser impact and
# lesser representative of the final model.
```

vi. Assess the improvement of the new model compared to your original model (simple regression model) by testing whether this change is significant by performing an analysis of variance

```
anova(housing_lm1, housing_lm5)
```

```
## Analysis of Variance Table
##
```

```
## Model 1: Sale_Price ~ sq_ft_lot
## Model 2: Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade +
##     year_built
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
## 1   2998 7.7637e+13
## 2   2995 1.7363e+13  3 6.0273e+13 3465.5 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# The results of anova which compares hierarchical models and the results have a F value
# of 3465.5 with a p value significantly smaller than 0. These results indicate the Model
# with 4 predictors(housing_lm5) have significant improvement compared to the simple
# regression model housing_lm1.
```

vii. Perform casewise diagnostics to identify outliers and/or influential cases, storing each function's output in a dataframe assigned to a unique variable name.

```
outliers_inflCases_df <- housing_data_df[, c("Sale_Price", "sq_ft_lot", "square_feet_total_living",
    "building_grade", "year_built")]
# Outliers
outliers_inflCases_df$residuals <- resid(housing_lm5)
outliers_inflCases_df$standardized.residuals <- rstandard(housing_lm5)
outliers_inflCases_df$studentized.residuals <- rstudent(housing_lm5)
# Influential Cases
outliers_inflCases_df$cooks.distance <- cooks.distance(housing_lm5)
outliers_inflCases_df$leverage <- hatvalues(housing_lm5)
outliers_inflCases_df$covariance.ratios <- covratio(housing_lm5)
outliers_inflCases_df$dfbeta <- dfbeta(housing_lm5)
outliers_inflCases_df$dffits <- dffits(housing_lm5)
str(outliers_inflCases_df)
```

```
## tibble [3,000 x 13] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Price              : num [1:3000] 530000 606329 460000 680290 675000 ...
##  $ sq_ft_lot               : num [1:3000] 5732 5895 5848 7563 9402 ...
##  $ square_feet_total_living: num [1:3000] 2650 2740 2620 2530 2790 2030 3360 1470 1640 3370 ...
##  $ building_grade          : num [1:3000] 8 9 7 8 9 7 9 8 7 9 ...
##  $ year_built              : num [1:3000] 2011 2012 2004 2013 1988 ...
##  $ residuals               : Named num [1:3000] -84293 -61104 -104599 73455 12029 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ standardized.residuals  : Named num [1:3000] -1.108 -0.803 -1.375 0.965 0.158 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ studentized.residuals   : Named num [1:3000] -1.108 -0.803 -1.375 0.965 0.158 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ cooks.distance          : Named num [1:3000] 1.98e-04 1.24e-04 7.12e-04 2.96e-04 5.17e-06 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ leverage                : Named num [1:3000] 0.000806 0.00096 0.00188 0.001586 0.001033 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ covariance.ratios       : Named num [1:3000] 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
##  $ dfbeta                  : num [1:3000, 1:5] 4909 2508 4520 -8703 283 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3000] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:5] "(Intercept)" "sq_ft_lot" "square_feet_total_living" "building_grade" ...
```

```
##  $ dffits                  : Named num [1:3000] -0.03146 -0.02488 -0.05968 0.03848 0.00508 ...
##   ..- attr(*, "names")= chr [1:3000] "1" "2" "3" "4" ...
```

```r
# other way of identifying the outliers and/or influential cases is by using
# influence.measures function:
infl_measures <- influence.measures(housing_lm5)
cook.d <- (influence.measures(housing_lm5)$infmat[, "cook.d"])
```

viii. Calculate the standardized residuals using the appropriate command, specifying those that are +-2, storing the results of large residuals in a variable you create.

```r
# For any normally distributes sample we expect 95% of z-scores to lie between -1.96 and
# +1.96. These have been rounded off to 2. So we expect 95% of the Standardized residuals
# to lie with in the range of +/-2. The standardized residuals are calculated as a
# measure of residuals divided by the std.deviation.
outliers_inflCases_df$large.residual <- outliers_inflCases_df$standardized.residuals > 2 |
    outliers_inflCases_df$standardized.residuals < -2
```

ix. Use the appropriate function to show the sum of large residuals.

```r
# creating variables to validate if 95% of the cases lie within the
# standardized.residuals interval of -2 and 2
sum_outliers_95 <- sum(outliers_inflCases_df$large.residual)
sum_outliers_95
```

```
## [1] 115
```

```r
sum_outliers_95_percent <- (sum_outliers_95/nrow(housing_data_df) * 100)
# The results indicates only 3.83 % of cases are outside the range of -2 and 2
# standardized.residuals and is within acceptable range of 5%. SO the model is a good
# representation of the data.
sum_outliers_95_percent
```

```
## [1] 3.833333
```

```r
# creating variables to validate if 99% of the cases lie within the
# standardized.residuals interval of -2.58 and 2.58
sum_outliers_99 <- sum(outliers_inflCases_df$standardized.residuals > 2.58 | outliers_inflCases_df$stan
    -2.58)
sum_outliers_99
```

```
## [1] 20
```

```r
sum_outliers_99_percent <- (sum_outliers_99/nrow(housing_data_df) * 100)
# The results indicates only 0.667 % of cases are outside the range of -2.58 and 2.58
# standardized.residuals and is within acceptable range of 1%. So the model is a good
# representation of the data.
sum_outliers_99_percent
```

```
## [1] 0.6666667
```

```
# creating variables to validate if 99.9% of the cases lie within the
# standardized.residuals interval of -3.29 and 3.29
sum_outliers_99.9 <- sum(outliers_inflCases_df$standardized.residuals > 3.29 | outliers_inflCases_df$sta
    -3.29)
sum_outliers_99.9
```

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

```
sum_outliers_99.9_percent <- (sum_outliers_99.9/nrow(housing_data_df) * 100)
# The results indicates only 0.033 % of cases are outside the range of -3.29 and 3.29
# standardized.residuals and is within acceptable range of 0.1%. So the model is a good
# representation of the data.
sum_outliers_99.9_percent
```

```
## [1] 0.03333333
```

x. Which specific variables have large residuals (only cases that evaluate as TRUE)?

```
# Restricting the rows that has outliers_inflCases_df$large.residual =TRUE and creating a
# new tibble.
large_residuals_df <- outliers_inflCases_df[outliers_inflCases_df$large.residual, c("Sale_Price",
    "sq_ft_lot", "square_feet_total_living", "building_grade", "year_built", "standardized.residuals",
    "cooks.distance", "leverage", "covariance.ratios")]
nrow(large_residuals_df)
```

```
## [1] 115
```

```
large_residuals_df
```

```
## # A tibble: 115 x 9
##    Sale_Price sq_ft_lot square~1 build~2 year_~3 stand~4 cooks~5 lever~6 covar~7
##         <dbl>     <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1     997737     11892     3535       9    2006    2.40 3.32e-3 2.89e-3   0.995
##  2     545000     10871     3340       8    1985   -2.26 1.95e-3 1.91e-3   0.995
##  3     720001      4295     2620       7    2001    2.19 1.89e-3 1.96e-3   0.996
##  4     510000      7200     3300       8    1980   -2.32 2.70e-3 2.51e-3   0.995
##  5     369000      7850     2300       8    1984   -2.19 4.91e-4 5.10e-4   0.994
##  6     722000      6152     2510       7    2001    2.33 1.76e-3 1.62e-3   0.994
##  7     420000      9800     2320       9    1979   -2.09 1.55e-3 1.76e-3   0.996
##  8     998990      5775     3530       9    2015    2.67 1.84e-3 1.29e-3   0.991
##  9     989900      6323     3430       9    2015    2.73 1.71e-3 1.15e-3   0.990
## 10     684000      7867     3990       9    2005   -2.40 2.27e-3 1.97e-3   0.994
## # ... with 105 more rows, and abbreviated variable names
## #   1: square_feet_total_living, 2: building_grade, 3: year_built,
## #   4: standardized.residuals, 5: cooks.distance, 6: leverage,
## #   7: covariance.ratios
```

xi. Investigate further by calculating the leverage, cooks distance, and covariance rations. Comment on all cases that are problematics.

```r
# The cook distance, leverage and covariance ratios were calculated in the previous
# questions.
large_residuals_df[, c("cooks.distance", "leverage", "covariance.ratios")]
```

```
## # A tibble: 115 x 3
##    cooks.distance leverage covariance.ratios
##             <dbl>    <dbl>             <dbl>
##  1       0.00332  0.00289              0.995
##  2       0.00195  0.00191              0.995
##  3       0.00189  0.00196              0.996
##  4       0.00270  0.00251              0.995
##  5       0.000491 0.000510             0.994
##  6       0.00176  0.00162              0.994
##  7       0.00155  0.00176              0.996
##  8       0.00184  0.00129              0.991
##  9       0.00171  0.00115              0.990
## 10       0.00227  0.00197              0.994
## # ... with 105 more rows
```

```r
# The below command checks how many rows have cooks distance greater than 1 and selects
# the required columns.
large_residuals_df[large_residuals_df$cooks.distance >= 1, c("cooks.distance", "leverage",
    "covariance.ratios")]
```

```
## # A tibble: 0 x 3
## # ... with 3 variables: cooks.distance <dbl>, leverage <dbl>,
## #   covariance.ratios <dbl>
```

```r
# Results of cooks distance shows none of them have cook's distance greater than 1 and
# hence none of the cases is having an undue influence on the model.
```

```r
# Average Leverage can be calculated by (k+1/n), where k is the number of predictors and
# n is the total number of cases.
avg_leverage <- (4 + 1)/nrow(housing_data_df)
avg_leverage
```

```
## [1] 0.001666667
```

```r
times3_avg_leverage <- 3 * avg_leverage
times3_avg_leverage
```

```
## [1] 0.005
```

```r
# Validating the number of cases with leverage greater than 3 times the avg.leverage
large_residuals_df[large_residuals_df$leverage > times3_avg_leverage, c("Sale_Price", "sq_ft_lot",
    "square_feet_total_living", "building_grade", "year_built", "cooks.distance", "leverage",
    "covariance.ratios")]
```

```
## # A tibble: 3 x 8
##   Sale_Price sq_ft_lot square_feet_tot~1 build~2 year_~3 cooks~4 lever~5 covar~6
```

```
##         <dbl>     <dbl>                 <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1      550000      8203                  2220       6    1932 0.00957 0.0102    1.00
## 2      610000     10393                  3930       8    1965 0.00595 0.00569   0.999
## 3      500000      9452                  3300       8    1955 0.00534 0.00554   0.999
## # ... with abbreviated variable names 1: square_feet_total_living,
## #   2: building_grade, 3: year_built, 4: cooks.distance, 5: leverage,
## #   6: covariance.ratios
```

```r
# The Upper limit of covariance ratios can be calculated as 1+ 3 times the average
# leverage and lower limit is 1- 3 times the average leverage.
cvr_lower <- 1 - times3_avg_leverage
cvr_lower
```

```
## [1] 0.995
```

```r
cvr_upper <- 1 + times3_avg_leverage
cvr_upper
```

```
## [1] 1.005
```

```r
# Validating the number of cases with covariance ratios above the upper range and below
# the lower range. Results indicate all impacted cases(62 cases) have cov.ratio below the
# lower range.
large_residuals_df[large_residuals_df$covariance.ratios > cvr_upper, c("Sale_Price", "sq_ft_lot",
    "square_feet_total_living", "building_grade", "year_built", "cooks.distance", "leverage",
    "covariance.ratios")]
```

```
## # A tibble: 0 x 8
## # ... with 8 variables: Sale_Price <dbl>, sq_ft_lot <dbl>,
## #   square_feet_total_living <dbl>, building_grade <dbl>, year_built <dbl>,
## #   cooks.distance <dbl>, leverage <dbl>, covariance.ratios <dbl>
```

```r
large_residuals_df[large_residuals_df$covariance.ratios < cvr_lower, c("Sale_Price", "sq_ft_lot",
    "square_feet_total_living", "building_grade", "year_built", "cooks.distance", "leverage",
    "covariance.ratios")]
```

```
## # A tibble: 62 x 8
##    Sale_Price sq_ft_lot square_feet_to~1 build~2 year_~3 cooks~4 lever~5 covar~6
##         <dbl>     <dbl>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1     997737     11892             3535       9    2006 3.32e-3 2.89e-3   0.995
##  2     369000      7850             2300       8    1984 4.91e-4 5.10e-4   0.994
##  3     722000      6152             2510       7    2001 1.76e-3 1.62e-3   0.994
##  4     998990      5775             3530       9    2015 1.84e-3 1.29e-3   0.991
##  5     989900      6323             3430       9    2015 1.71e-3 1.15e-3   0.990
##  6     684000      7867             3990       9    2005 2.27e-3 1.97e-3   0.994
##  7     699950     13492             3820      10    1997 5.15e-3 3.92e-3   0.995
##  8     975000      6254             3590       8    2012 3.18e-3 2.12e-3   0.991
##  9     619500      9787             3530       9    2006 1.87e-3 1.58e-3   0.993
## 10     293504      3073             1900       8    2007 1.49e-3 1.27e-3   0.993
## # ... with 52 more rows, and abbreviated variable names
## #   1: square_feet_total_living, 2: building_grade, 3: year_built,
## #   4: cooks.distance, 5: leverage, 6: covariance.ratios
```

```r
# Examining the summary of the cov.ratios for those impacted cases indicate that the
# minimum is 0.9804 which is 0.01 less than the acceptable value of 0.995. The mean is
# 0.9928 which is pretty close to the acceptable lower range. Also all the impacted cases
# have cook's distance lesser than 1, so there is probably little cause for alarm.
summary(large_residuals_df[large_residuals_df$covariance.ratios < cvr_lower, c("covariance.ratios")])
```

```
##  covariance.ratios
##  Min.   :0.9804
##  1st Qu.:0.9914
##  Median :0.9938
##  Mean   :0.9928
##  3rd Qu.:0.9945
##  Max.   :0.9950
```

xii. Perform the necessary calculations to assess the assumption of independence and state if the condition is met or not.

```r
durbinWatsonTest(housing_lm5)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1     -0.01013318      2.019835   0.604
##  Alternative hypothesis: rho != 0
```

```r
# The results of durbinWatsonTest for model housing_lm5 shows that the Statistic is close
# to 2 which indicates better values and also the value of p is 0.61 which means not
# remotely significant.
```

xiii. Perform the necessary calculations to assess the assumption of no multicollinearity and state if the condition is met or not.

```r
# multicollinearity exists if two or more predictors in a regression model have a
# correlation between them. The results of vif are all well below 10 and there is no
# reason for concern and it suggests the multicollinearity does not exiist in the model.
vif(housing_lm5)
```

```
##                sq_ft_lot square_feet_total_living           building_grade
##                 1.961901                 1.965371                 1.807125
##               year_built
##                 2.353771
```

```r
# 1/ vif(housing_lm5) indicates the tolerance and as all tolerance is more than 0.1, the
# values are satisfactory.
1/vif(housing_lm5)
```

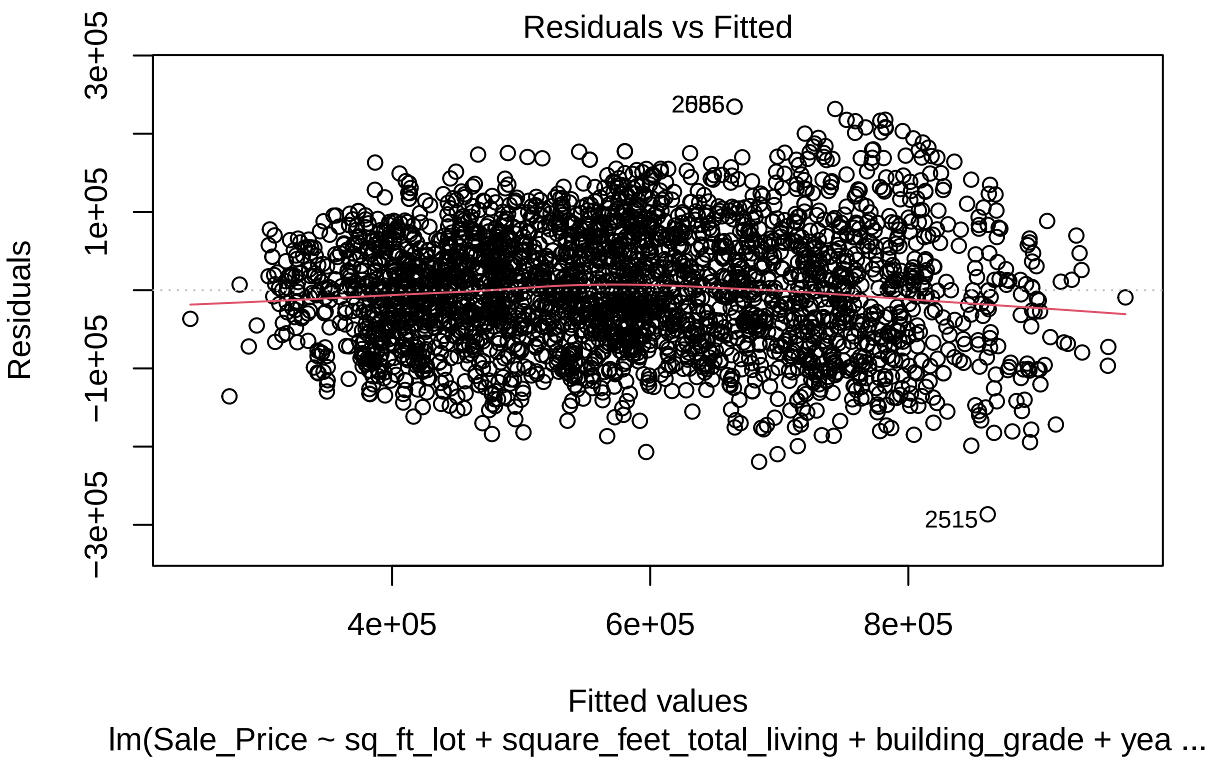
```
##                sq_ft_lot square_feet_total_living           building_grade
##                0.5097096                0.5088097                0.5533652
##               year_built
##                0.4248501
```

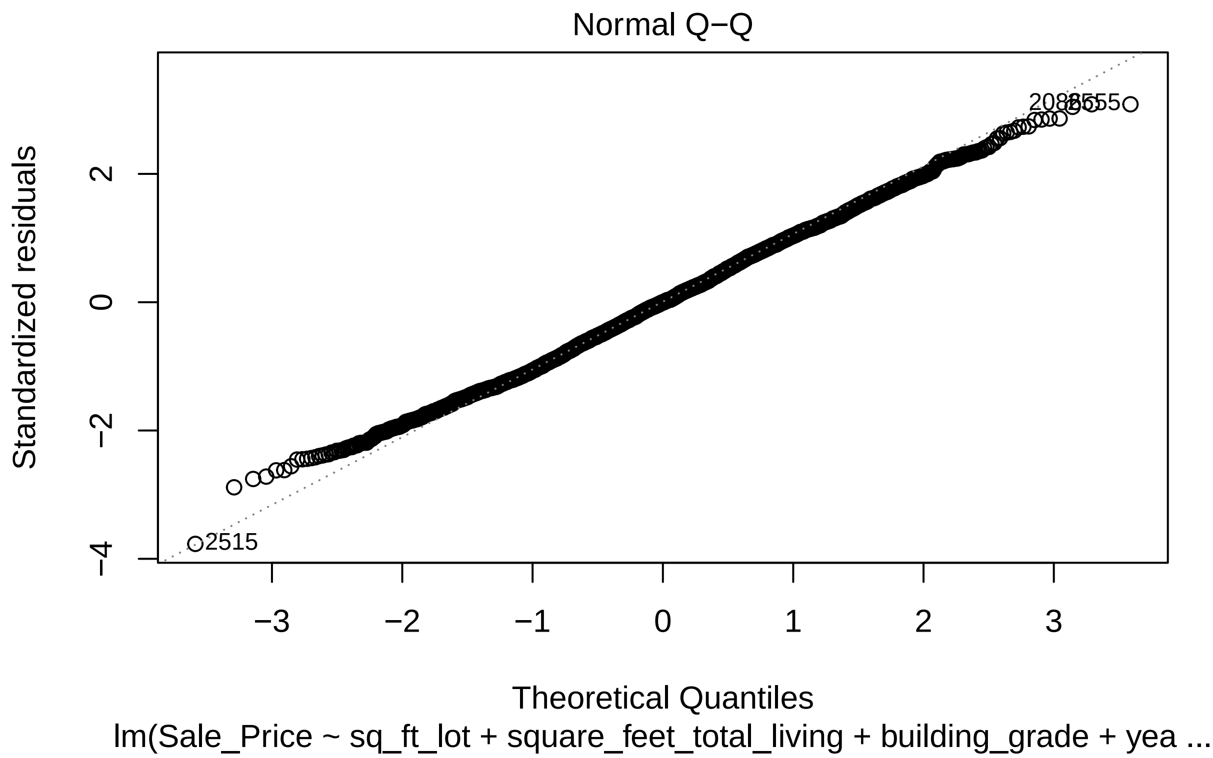
```r
# The Average of vif is higher than 1 but not significantly higher, so the model is not
# too biased.
mean(vif(housing_lm5))
```

```
## [1] 2.022042
```

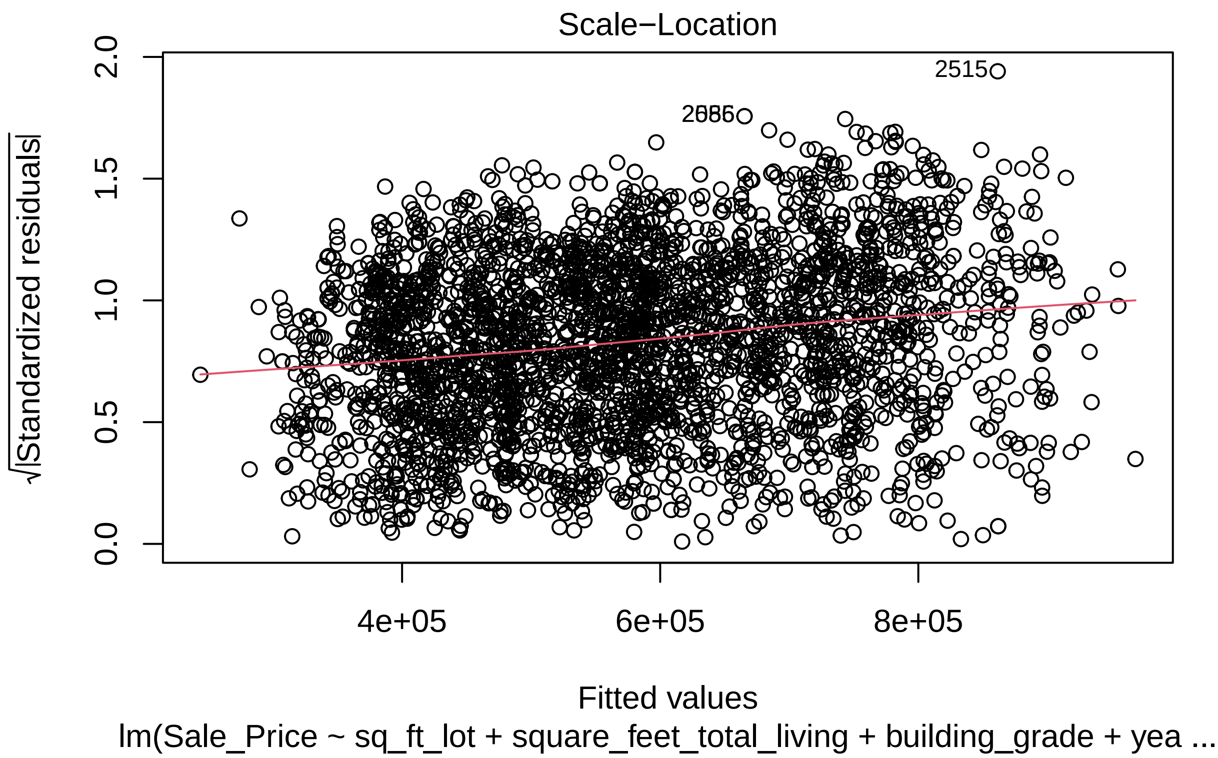
xiv. Visually check the assumptions related to the residuals using the plot() and hist() functions. Summarize what each graph is informing you of and if any anomalies are present.

```r
plot(housing_lm5)
```

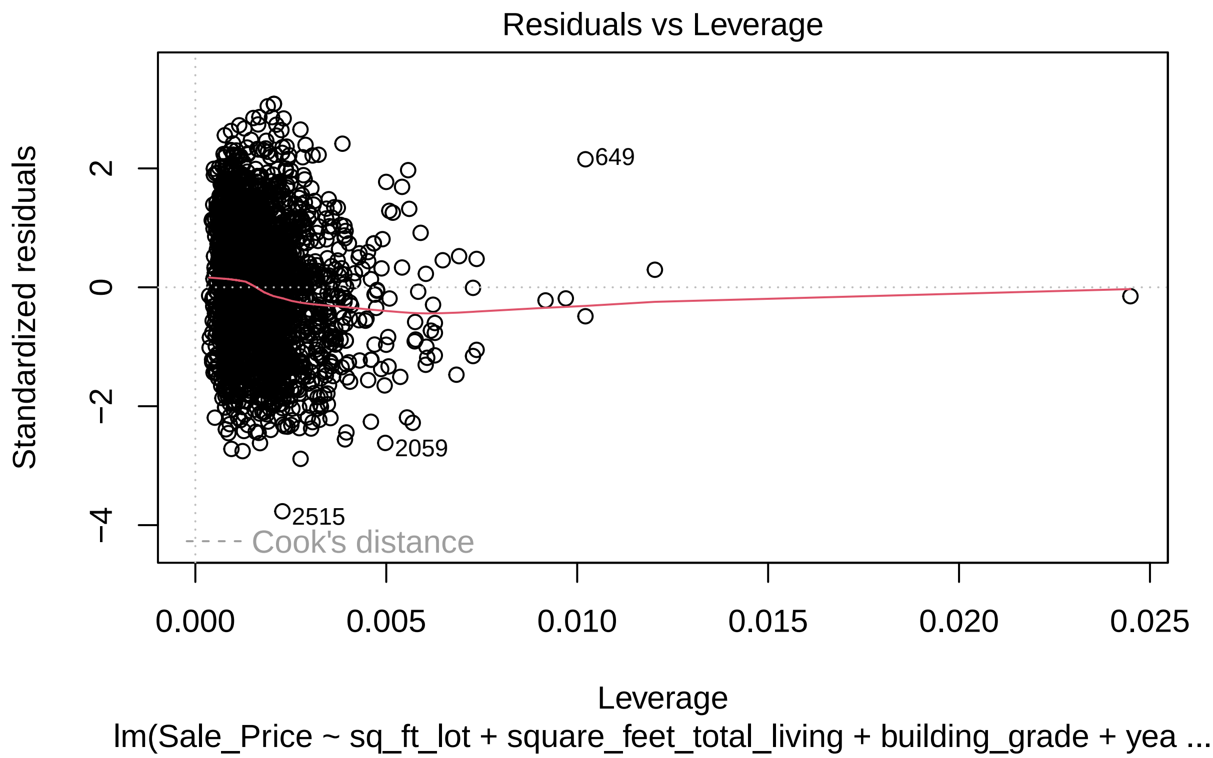




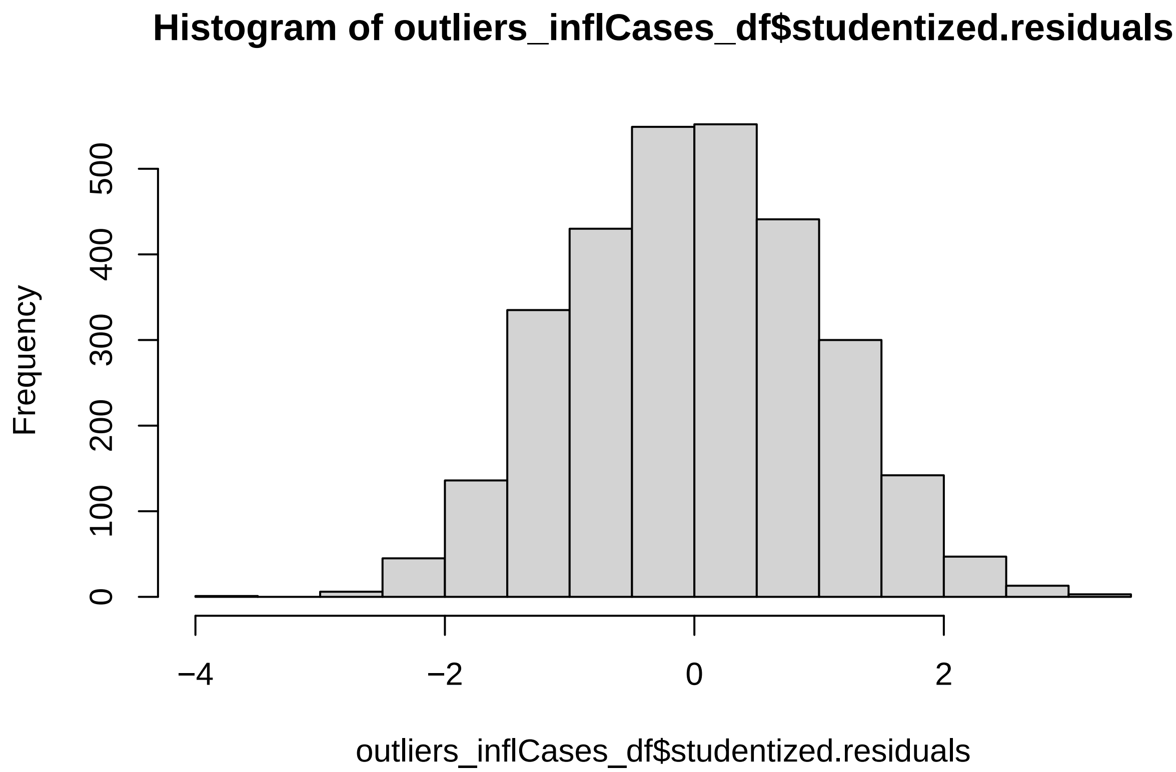
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + yea ...
−4
−2
0
2
−3
−2
−1
0
1
2
3
2515
2086
2555



Scale−Location
2515
2086
0.0
0.5
1.0
1.5
2.0
√|Standardized residuals|
4e+05
6e+05
8e+05
Fitted values
lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + yea ...




```
hist(outliers_inflCases_df$studentized.residuals)
```




```
# Explanation: The first graph of fitted values vs residuals shows a random array of dots
# dispersed around zero and does not funnel out. So this shows that the assumptions for
# linearity and homoscedasicity have been met. There are no curve patterns in this graph
# either. Also we see similar chunk of residuals above and below zero which indicates the
# relationship is linear.

# The second Q-Q plot shows most of the values are between -2 and +2 Standard deviation
# and there are a few outliers and the numbering is shown as case number 2515, etc.

# The third plot SCale-location plot explains the extent of homoscedasicity in the model.
# Though the redline is not relatively horizontal, there is not cluster or pattern in the
# data points and looks like a cloud which indicates homoscedasicity.

# The fourth plot Residuals vs Leverage helps us find Influential data points that can
# have a bigger effect on the linear model. As shown in the plot there are no data points
# that lies outside the cook's distance. SO it indicates there are no influential outlier
# in the model

# The histogram on the studentized residuals shows that the model is almost normally
# distributed with some left skew.
```

xv. Overall, is this regression model unbiased? If an unbiased regression model, what does this tell us about the sample vs. the entire population model?

```r
# Solution: The average Value of VIF indicates how much the model is biased. the average
# for our model comes about `mean(vif(housing_lm5))` which is not significantly greater
# than 1, so our model is not too biased.

# Other ways of finding the effectiveness of our model to other samples from dataset is
# by validating the adjusted R2. If the Adjusted R2 is close to the Multiple R2, it
# indicates there is not much bias, which is true for the model housing_lm5.
summary(housing_lm5)
```

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot + square_feet_total_living +
##     building_grade + year_built, data = housing_data_df, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -286513  -53569    -130   54780  234662
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.620e+06  2.611e+05 -10.037  < 2e-16 ***
## sq_ft_lot                 4.950e+00  7.355e-01   6.730 2.02e-11 ***
## square_feet_total_living  1.583e+02  2.776e+00  57.041  < 2e-16 ***
## building_grade            3.684e+04  2.266e+03  16.263  < 2e-16 ***
## year_built                1.239e+03  1.326e+02   9.343  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 76140 on 2995 degrees of freedom
## Multiple R-squared:  0.7806, Adjusted R-squared:  0.7803
## F-statistic:  2664 on 4 and 2995 DF,  p-value: < 2.2e-16
```

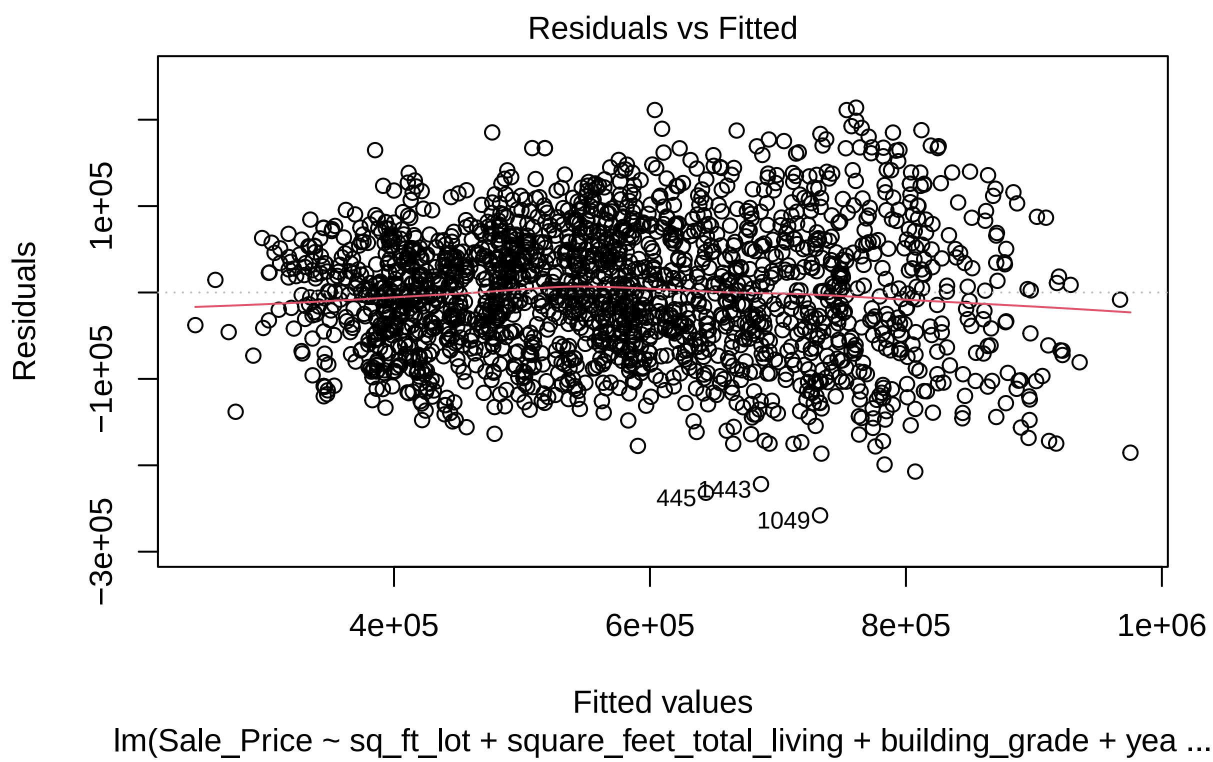
```r
# The Model can be used to predict values and the results can be compared to see the
# effectiveness.
outliers_inflCases_df$predict_saleprice <- predict(housing_lm5, predict_saleprice = outliers_inflCases_(

# Another way of testing the effectiveness/presence of bias in the model is by taking
# different samples from original dataset and testing the results and comparing the
# results.

set.seed(42)
housing_data_df_2 <- housing_data_df_all[sample(nrow(housing_data_df_all), size = 2000), ]
nrow(housing_data_df_2)
```

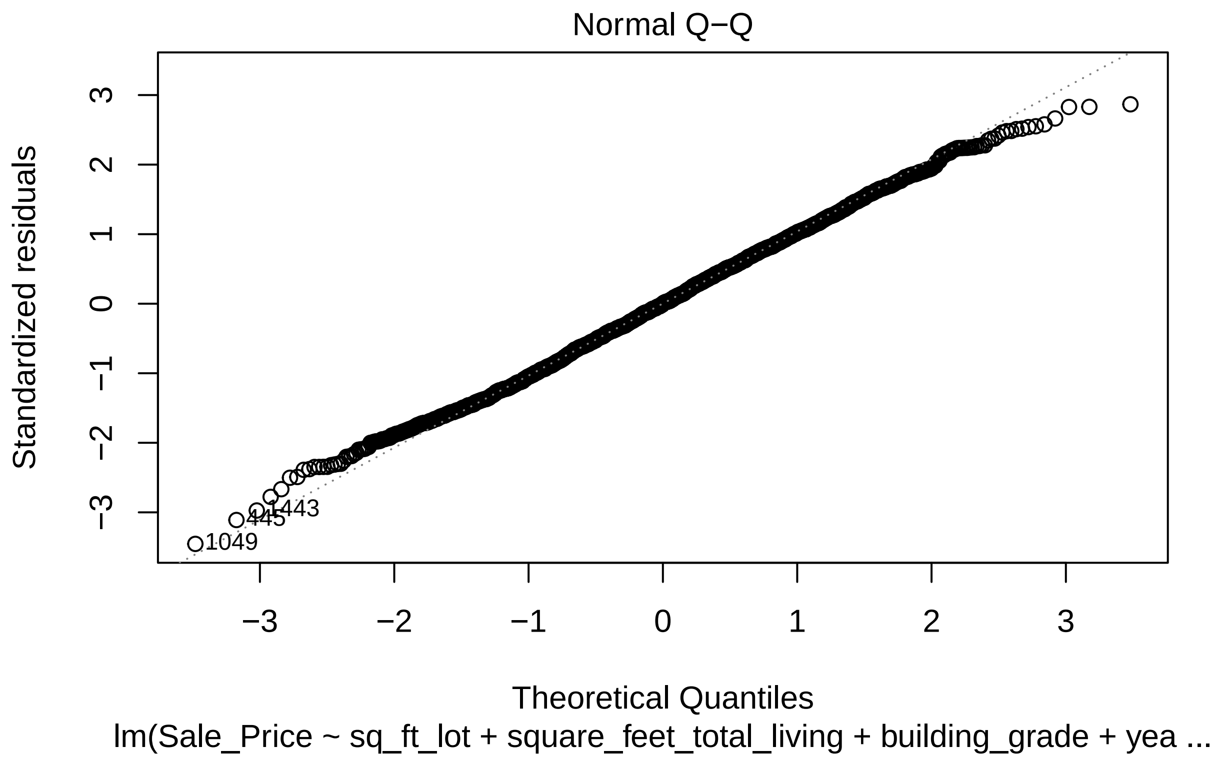
```
## [1] 2000
```

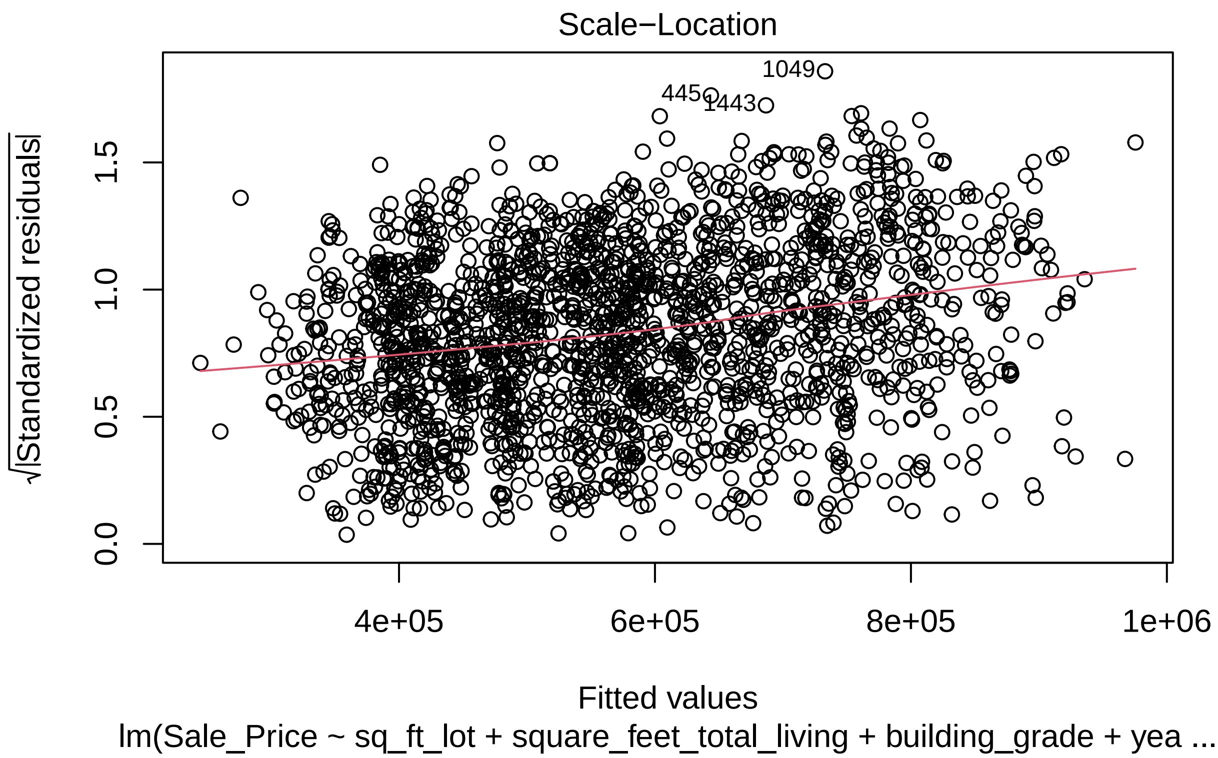
```r
# create a new model from same set of predictors taking different sample set to compare
# the results
housing_lm6 <- lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + year_built,
    data = housing_data_df_2, na.action = na.omit)

summary(housing_lm6)
```

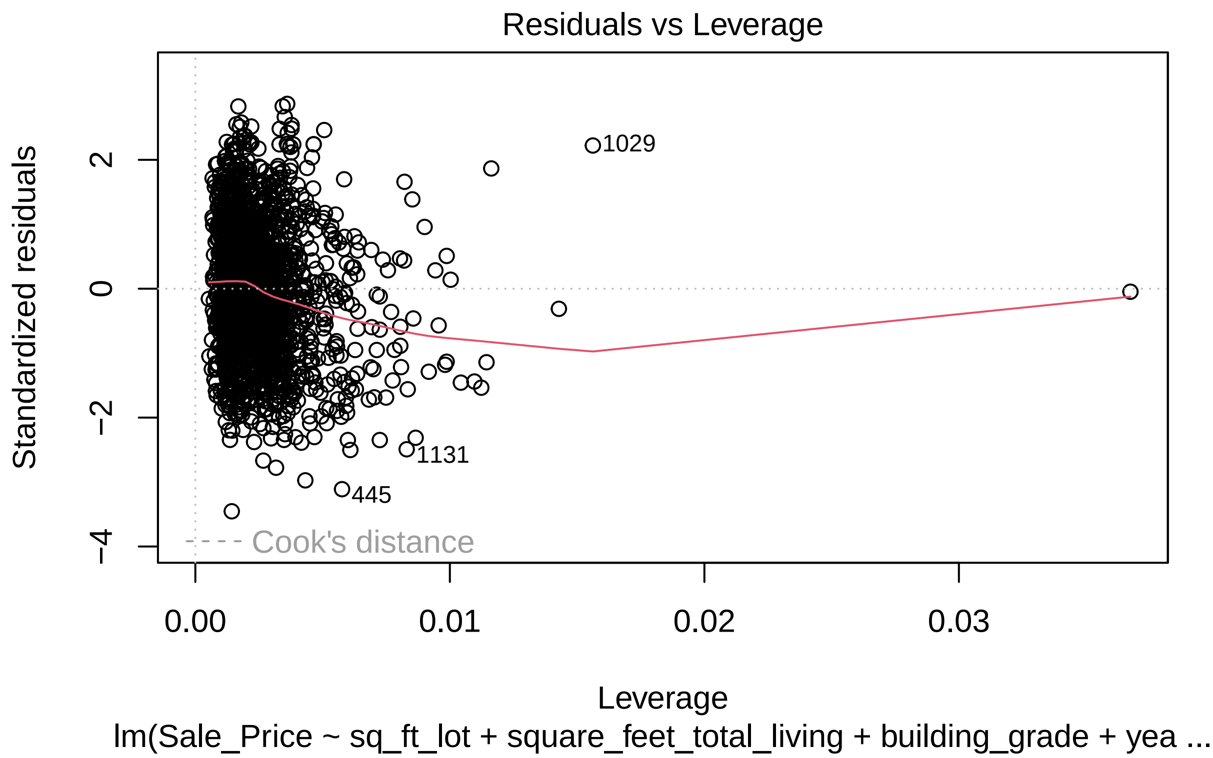
```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot + square_feet_total_living +
##     building_grade + year_built, data = housing_data_df_2, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -257920  -51975    -112   52417  213927
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.820e+06  3.151e+05  -8.950  < 2e-16 ***
## sq_ft_lot                 5.337e+00  8.999e-01   5.930 3.56e-09 ***
## square_feet_total_living  1.581e+02  3.372e+00  46.885  < 2e-16 ***
## building_grade            3.520e+04  2.803e+03  12.559  < 2e-16 ***
## year_built                1.346e+03  1.600e+02   8.411  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 74730 on 1995 degrees of freedom
## Multiple R-squared:  0.7881, Adjusted R-squared:  0.7877
## F-statistic:  1855 on 4 and 1995 DF,  p-value: < 2.2e-16
```

```
plot(housing_lm6)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + yea ...
1049
445
1443
−3
−2
−1
0
1
2
3


Scale−Location
1049
445
1443
√|Standardized residuals|
0.0
0.5
1.0
1.5
4e+05
6e+05
8e+05
1e+06
Fitted values
lm(Sale_Price ~ sq_ft_lot + square_feet_total_living + building_grade + yea ...




```
# The results of summary of the new model created for different sample size indicates
# there is not much difference between R2 and adjusted R2 and is very close to the values
# of original model housing_lm5. Also the plots of housing_lm6 are very similar to the
# plot of housing_lm5. All these results prove that our model is almost unbiased and can
# be effective with any samples and beyond.
```

Session Info

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22621)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
```

```
## [8] base
##
## other attached packages:
##  [1] car_3.1-1        carData_3.0-5    relaimpo_2.2-6   mitools_2.4
##  [5] survey_4.1-1     survival_3.4-0   Matrix_1.5-1     QuantPsyc_1.6
##  [9] MASS_7.3-58.1    purrr_0.3.5      boot_1.3-28      olsrr_0.5.3
## [13] magrittr_2.0.3   lubridate_1.9.0  timechange_0.1.1 dplyr_1.0.10
## [17] readxl_1.4.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0  nortest_1.0-4     xfun_0.34         splines_4.2.2
##  [5] lattice_0.20-45   colorspace_2.0-3  vctrs_0.5.0       generics_0.1.3
##  [9] htmltools_0.5.3   yaml_2.3.6        utf8_1.2.2        rlang_1.0.6
## [13] pillar_1.8.1      withr_2.5.0       glue_1.6.2        DBI_1.1.3
## [17] lifecycle_1.0.3   stringr_1.4.1     munsell_0.5.0     gtable_0.3.1
## [21] cellranger_1.1.0  evaluate_0.18     knitr_1.41        fastmap_1.1.0
## [25] fansi_1.0.3       highr_0.9         Rcpp_1.0.9        corpcor_1.6.10
## [29] scales_1.2.1      formatR_1.12      abind_1.4-5       gridExtra_2.3
## [33] ggplot2_3.4.0     digest_0.6.30     stringi_1.7.8     cli_3.4.1
## [37] tools_4.2.2       goftest_1.2-3     tibble_3.1.8      pkgconfig_2.0.3
## [41] data.table_1.14.4 assertthat_0.2.1  rmarkdown_2.18    rstudioapi_0.14
## [45] R6_2.5.1          compiler_4.2.2
```