Assignment_08_Housing Data

Karthikeyan Chellamuthu

13/02/2022

```
library(ggplot2)
library(readxl)
library(QuantPsyc)
```

```
## Loading required package: boot
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'QuantPsyc'
```

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

```
library(fitdistrplus)
```

```
## Loading required package: survival
```

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

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

```
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

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

```
library("plyr")
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library(magrittr)
library(ggpubr)
```

```
##
## Attaching package: 'ggpubr'
```

```
## The following object is masked from 'package:plyr':
##
##     mutate
```

```
library(ggm)


setwd("D:/BU/DSC 520 T302-2221 winter 2021-22/GIT-Hub/dsc520-master/data")


# Import "week-7-housing.xlsx" for analysis and create a data frame
Housing <- read_excel("week-7-housing.xlsx")
```

```
## # A tibble: 6 x 24
##   Sale_Date           Sale_Price sale_reason sale_instrument sale_warning
##   <dttm>                   <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00     698000           1               3 <NA>
## 2 2006-01-03 00:00:00     649990           1               3 <NA>
## 3 2006-01-03 00:00:00     572500           1               3 <NA>
## 4 2006-01-03 00:00:00     420000           1               3 <NA>
## 5 2006-01-03 00:00:00     369900           1               3 15
## 6 2006-01-03 00:00:00     184667           1              15 18 51
## # ... with 19 more variables: site_type <chr>, addr_full <chr>, zip5 <dbl>,
## #   ctyname <chr>, postalctyn <chr>, lon <dbl>, lat <dbl>,
## #   building_grade <dbl>, square_feet_total_living <dbl>, bedrooms <dbl>,
## #   bath_full_count <dbl>, bath_half_count <dbl>, batch_3qtr_count <dbl>,
## #   year_built <dbl>, year_renovated <dbl>, current_zoning <chr>,
## #   sq_ft_lot <dbl>, prop_type <chr>, present_use <dbl>
```

###7.1.b 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, Bedrooms, and Bath Full Count as predictors.

##7.1.c 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?

```
##
## Call:
## lm(formula = Sale_Price ~ sq_ft_lot, data = Housing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2016064  -194842   -63293    91565  3735109
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.418e+05  3.800e+03  168.90   <2e-16 ***
## sq_ft_lot   8.510e-01  6.217e-02   13.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 401500 on 12863 degrees of freedom
## Multiple R-squared:  0.01435,    Adjusted R-squared:  0.01428
## F-statistic: 187.3 on 1 and 12863 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = Sale_Price ~ bedrooms + bath_full_count, data = Housing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3566590  -157368   -55794    67256  3891256
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       148987      14890   10.01   <2e-16 ***
## bedrooms           70566       4048   17.43   <2e-16 ***
## bath_full_count   148058       5450   27.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 383200 on 12862 degrees of freedom
## Multiple R-squared:  0.1023, Adjusted R-squared:  0.1022
## F-statistic: 733.2 on 2 and 12862 DF,  p-value: < 2.2e-16
```

###7.1.d 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?

```
## sq_ft_lot
## 0.1198122
```

```
##        bedrooms bath_full_count
##       0.1528878       0.2382795
```

```
## [1] 404381.1
```

```
## [1] 56933.29
```

```
## [1] 0.8761273
```

```
## [1] 0.6507965
```

###7.1.e Calculate the confidence intervals for the parameters in your model and explain what the results indicate.

```
##                    2.5 %       97.5 %
## (Intercept) 6.343730e+05 6.492698e+05
## sq_ft_lot   7.291208e-01 9.728641e-01
```

```
##                     2.5 %    97.5 %
## (Intercept)     119800.94 178173.83
## bedrooms         62630.97  78501.33
## bath_full_count 137375.51 158740.80
```

###7.1.f 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.

```
## Analysis of Variance Table
## 
## Model 1: Sale_Price ~ bedrooms + bath_full_count
## Model 2: Sale_Price ~ sq_ft_lot
##   Res.Df        RSS Df   Sum of Sq      F    Pr(>F)    
## 1  12862 1.8883e+15                                    
## 2  12863 2.0734e+15 -1 -1.8509e+14 1260.8 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

###7.1.g. Perform casewise diagnostics to identify outliers and/or influential cases, storing each functions output in a dataframe assigned to a unique variable name.

```
## tibble [12,865 x 32] (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 ...
##  $ site_type               : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 
174TH AVE NE" "3303 178TH AVE NE" ...
##  $ 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 27
60 ...
##  $ 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 ...
##  $ batch_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 ...
##  $ resid                   : Named num [1:12865] -29368 -77378 -6810 -88744 -138844 ...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ stand.resid             : Named num [1:12865] -0.0767 -0.202 -0.0178 -0.2316 -0.3624 
...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ stud.resid              : Named num [1:12865] -0.0766 -0.202 -0.0178 -0.2316 -0.3624 
...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ cooks.dist              : Named num [1:12865] 2.09e-07 1.45e-06 2.90e-08 3.53e-06 8.64e
-06 ...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ dfbeta                  : num [1:12865, 1:3] 3.614 9.521 -0.556 -34.401 -53.822 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:12865] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:3] "(Intercept)" "bedrooms" "bath_full_count"
##  $ dffit                   : Named num [1:12865] -0.000792 -0.002086 -0.000295 -0.003254 -
0.005091 ...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ leverage                : Named num [1:12865] 0.000107 0.000107 0.000275 0.000197 0.000
197 ...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
##  $ cov.ratio               : Named num [1:12865] 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:12865] "1" "2" "3" "4" ...
```

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

###7.1.i. Use the appropriate function to show the sum of large residuals.

```
## [1] 334
```

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

```
## # A tibble: 10 x 11
##    Sale_Price  zip5 building_grade square_feet_total_l~ bedrooms bath_full_count
##         <dbl> <dbl>          <dbl>                <dbl>    <dbl>           <dbl>
##  1    1900000 98053             11                 6610        4               3
##  2    1390000 98053              6                  660        0               1
##  3    1588359 98053              9                 3360        2               2
##  4    1450000 98052              8                 3480        3               2
##  5    1450000 98052              6                  900        2               1
##  6     270000 98053             11                 5060        4              23
##  7    2500000 98053             11                 6310        4               2
##  8    2169000 98053             12                 5080        4               3
##  9    1534000 98052             10                 3320        4               1
## 10     555000 98052             12                 6380        6               6
## # ... with 5 more variables: bath_half_count <dbl>, batch_3qtr_count <dbl>,
## #   year_built <dbl>, sq_ft_lot <dbl>, stand.resid <dbl>
```

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

```
## # A tibble: 1 x 3
##   cooks.dist leverage cov.ratio
##        <dbl>    <dbl>     <dbl>
## 1       3.14   0.0900      1.07
```

There is just one case which has cooks distance greater than 1. Value greater than 1 may be influencing the model.

Leverage is calculated using `(k + 1/n) = (2+1/12865) = 0.0002` . We will look for values either twice as large as this (0.0004) or three times as large (0.0006). Cases with large leverage values will not necessarily have a large influence on the regression coefficients as they are measure on the outcome variables rather than the predictors. + 181 cases > 0.0002. + 67 cases > 0.0004, 2 times average. + 44 cases > 0.0006. 3 times average.

Covariance upper limit is calculated by formula `1 + 3 times Average Leverage = 1 + 3*0.0002 = 1.0006` and Covariance lower limit is calculated by formula `1 - 3 times Average Leverage = 1 - 3*0.0004 = 0.9994` . In our model, 287 cases which are falling outside these limits which may be a little cause for alarm.

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

```
## Loading required package: carData
```

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

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

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.6416135     0.7167711       0
##  Alternative hypothesis: rho != 0
```

###7.1.m Perform the necessary calculations to assess the assumption of no multicollinearity and state if the condition is met or not. + VIF:

```
##       bedrooms bath_full_count
##       1.102269        1.102269
```

- Tolerance:

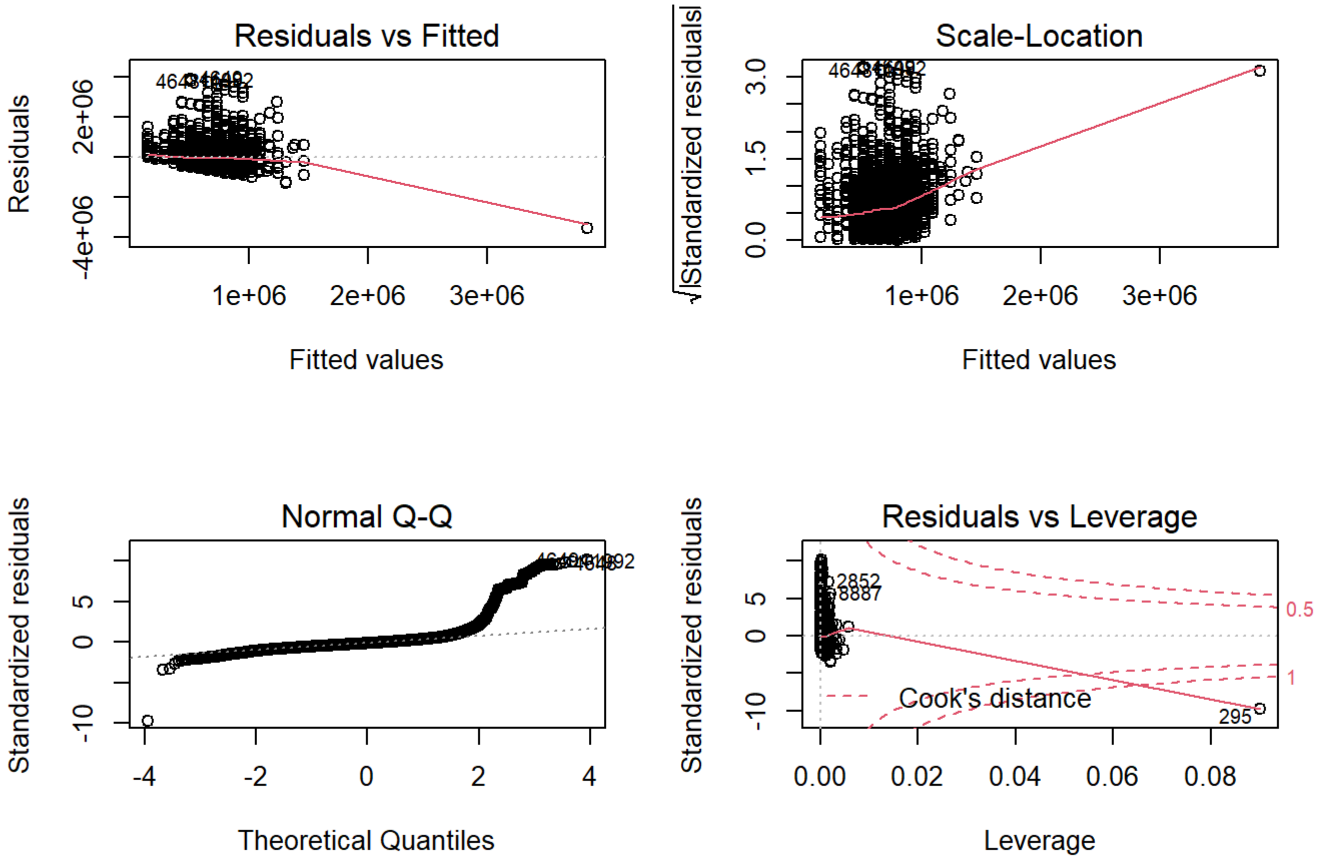
```
##       bedrooms bath_full_count
##      0.9072195       0.9072195
```

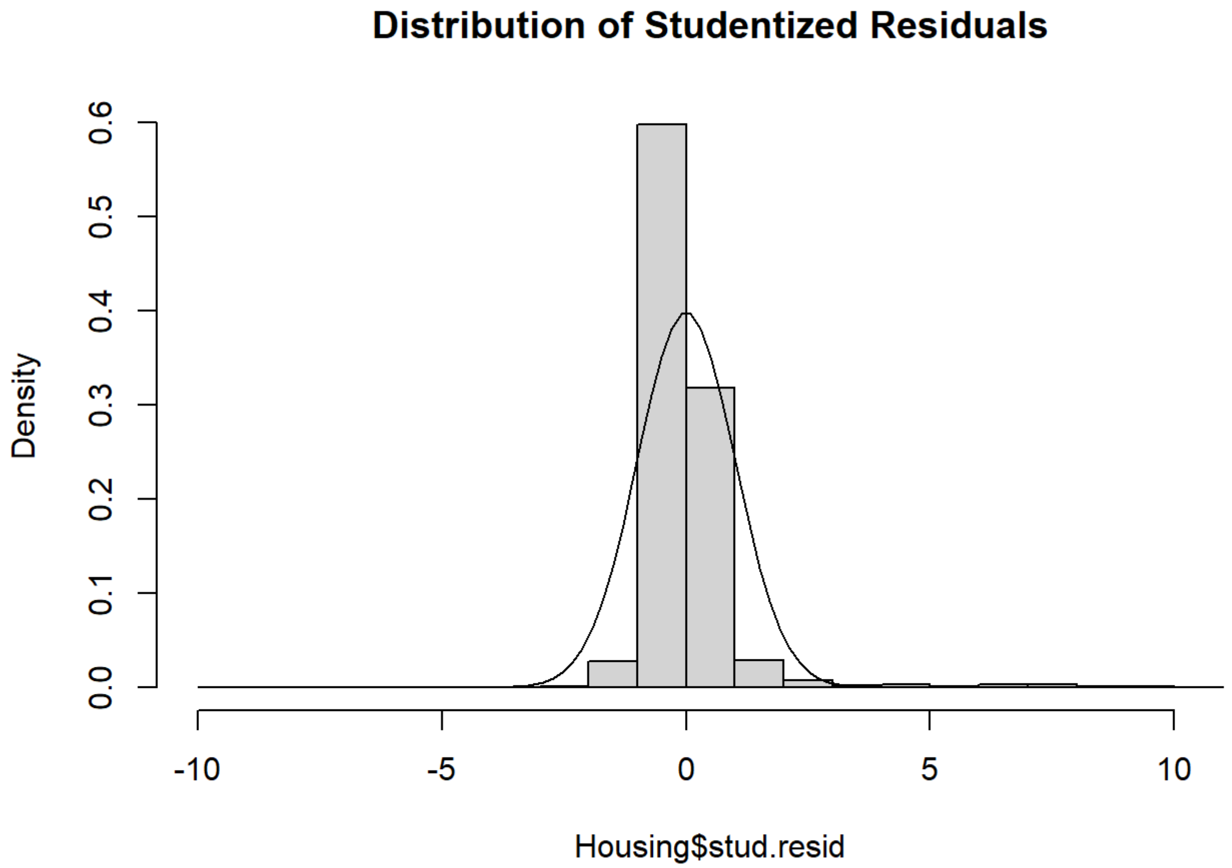
- Mean VIF:

```
## [1] 1.102269
```

VIF values are below 10 and tolerance statistics are above 0.2. Hence, there is no collinearity within the data.

###7.1.n 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.





Residual vs Fitted graph: Residuals in the model shows random distribution, which is the assumption of linearity. Hence it can be said that our model heteroscedasticity, randomness and linearity have been met. Plot: Residuals in our plot deviate from normality and the dots are very distant from the line, which indicates deviation from normality. Hist functions: By looking at the distribution, deviation from normality is at extremes. The bell curve shows normal distribution for most of the data, but due to extremes non-normality can be assumed.

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

```
## [1] 1.102269
```

From above, mean of VIF is close to 1. Hence model is unbiased.