Housing Data

Amelia Farrell

October 18th 2021

Background: Real estate transactions recorded from 1964 to 2016

a.i. Transformations

If you recall, we already took a closer look at this data back on week 4 and 4 (Exercise 4.2 and 5.2). We identified some data that would impact any predictions we hope to make. These include the "homes" with 0 bathrooms and 0 bedrooms. We would like to exclude these from this excerise as well. We would only like to look at homes that are move in ready and not cabins, plots of land, or simply have missing data. In order to drop these from our data set we can use the subset function to remove any line items with more than 0 bathrooms and bedrooms. Lets first re-check that this data exists in our data set (after transforming it from a list to a dataframe) (note that we are using the has_element function within the purrr package to check this data)

```
housing <- dfhousing <- data.frame(housing)
0 %in% dfhousing$bedrooms
```

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

```
0 %in% dfhousing$bath_full_count
```

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

As we can see above, the data does include line items with 0 bathrooms and 0 bedrooms.

We can exude these using the subset function and check that they have been removed.

```
dfhousing2 <- subset(dfhousing, bedrooms!= 0 & bath_full_count!= 0)
0 %in% dfhousing2$bedrooms
```

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

```
0 %in% dfhousing2$bath_full_count
```

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

Next we like to add price per square foot. Note that this will not include the square footage of the lot, but is still an important peice of information when considering a home. We will create a price per square foot variable and add it to the housing data frame below.

```
piceperfoot <- (dfhousing2$Sale.Price/dfhousing2$square_feet_total_living)
cbind(dfhousing2, piceperfoot)
```

b.i. Transformations explained

Lets summarize what we did above; - Create a dataframe to hold our housing data set. This is will allow us to set restrictions such as, not using the same name for two variable, keeping all elements as vectors, and ensuring at all columns are named. - Checking for line items with 0 bathrooms and 0 bedrooms. - Removing line items with 0 bathrooms and 0 bedrooms (reason for doing so explained above). - Creating price per square foot variable and adding it to the data frame.

b.ii. Create two variables (Linear Regression)

We will first fit a linear model using the `Square Foot of Lot` variable as the predictor and `Sale Price` as the outcome.

```
lotSF_lm <- lm(Sale.Price ~ sq_ft_lot, data = dfhousing2)
```

Then fit a linear model with a couple more predictors. Adding `year renovated` (this may impact the price more than the year it was built since renovations/remolding can greatly impact home value), `Square Feet Living` (the total square footage of the home is correlated to the sale price), `Full Bath Count` (the number of full bathrooms is also correlated to home price but not necessarily correlated to total square feet, making it a great additional predictor) as additional the predictors to `Sale Price`.

```
lotSF_lm2 <- lm(Sale.Price ~ sq_ft_lot + year_renovated + square_feet_total_living + bath_full_count, da
```

b.iii. Execute a summary() function

Lets now compare our two models for predicting home sale price with the summary function below.

```
##
## Call:
## lm(formula = Sale.Price ~ sq_ft_lot, data = dfhousing2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2046056  -194710   -63503    91200  3735135
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.417e+05  3.807e+03  168.58   <2e-16 ***
## sq_ft_lot   8.694e-01  6.277e-02   13.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 401600 on 12809 degrees of freedom
## Multiple R-squared:  0.01476,    Adjusted R-squared:  0.01468
## F-statistic: 191.9 on 1 and 12809 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = Sale.Price ~ sq_ft_lot + year_renovated + square_feet_total_living +
##     bath_full_count, data = dfhousing2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1925674  -119387   -39623    44816  3780658
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.458e+05  1.032e+04  14.122  < 2e-16 ***
## sq_ft_lot                  1.328e-01  5.803e-02   2.289   0.0221 *
## year_renovated             1.717e+01  1.404e+01   1.223   0.2212
## square_feet_total_living 1.702e+02  3.890e+00  43.768  < 2e-16 ***
## bath_full_count           4.382e+04  5.820e+03   7.529 5.44e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 359100 on 12806 degrees of freedom
## Multiple R-squared:  0.2124, Adjusted R-squared:  0.2122
## F-statistic: 863.5 on 4 and 12806 DF,  p-value: < 2.2e-16
```

We can locate the R2 and Adjusted R2 statistics is the above summary. Our first linear model with one independent variable (lotSF_lm) has a R2 of 0.01476 and a Adjusted R2 of 0.01468 Our second linear model with 4 independent variables (lotSF_lm2) has a R2 of 0.2124 and a Adjusted R2 of 0.2122 Based off the R2 results we can conclude that lotSF_lm2 (model with more predictors) is a better fit than the model with only one predictor/input variable. However, from the summary we can also see the p vaule of the individual predictors. The independent variable `year_renovated` has a p-vaule of 0.2212 making the least significant of all the other independent variables.

b.iv. Standardized Betas

Taking a closer look at the multiple regression model output (lotSF_lm2) we can make more assumptions regarding the independent variables based off their Standardized Betas (Std. Error). Upon looking at the Standardized Betas above, we can conclude that `Full Bath Count` is the worst at predicting the `Sale Price`. To help better explain this assumption lets reiterate what the Standardized Betas mean. Std. Error (Standardized Beta) is the estimate of the standard deviation of the coefficient. Based off this definition, a Std. Error of 5.820e+03 would mean that that variable `Full Bath Count` has a high uncertainty when it comes to coming making estimates that come close to the mean. From this We can conclude that `Full Bath Count` would likely not help us in making accurate predictions on `Sale Price`.

b.v. Confidence Intervals

However, lets not make assumptions too quickly. Lets look at the confidence intervals for each of our input variables.

```
##                                2.5 %       97.5 %
## (Intercept)               1.255206e+05 1.659801e+05
## sq_ft_lot                 1.907789e-02 2.465562e-01
## year_renovated           -1.034231e+01 4.468380e+01
## square_feet_total_living  1.626171e+02 1.778655e+02
## bath_full_count           3.241393e+04 5.523111e+04
```

As we can see above, `year_renovated` has the widest gap between 2.5% and 97.5% confidentence intervels. Meaning that it is the least confident in predicting where on the slope a new data point would lie. This may be due to the fact that we have many 0 vaules for this variable. Do to it's lack of signifgance and confidnec. It should be removed from the model. The remaining variables have less variabilty and will remain within the model.

b.vi. Analysis of variance

Since we uncovered the fact that `year_renovated` could be harming our model, we will be generating new model below.

```
lotSF_lm3 <- lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count, data = dfhousing2)
```

Next we will perform an analysis of variance between all three of our models. First comparing Model 1 and 2 then Model 2 and 3.

```
anova(lotSF_lm, lotSF_lm2, test = "F")
```

```
## Analysis of Variance Table
##
## Model 1: Sale.Price ~ sq_ft_lot
## Model 2: Sale.Price ~ sq_ft_lot + year_renovated + square_feet_total_living +
##     bath_full_count
##   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
## 1  12809 2.0654e+15
## 2  12806 1.6511e+15  3 4.1439e+14 1071.4 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lotSF_lm2, lotSF_lm3, test = "F")
```

```
## Analysis of Variance Table
##
## Model 1: Sale.Price ~ sq_ft_lot + year_renovated + square_feet_total_living +
##     bath_full_count
## Model 2: Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count
##   Res.Df        RSS Df   Sum of Sq      F Pr(>F)
## 1  12806 1.6511e+15
## 2  12807 1.6512e+15 -1 -1.9294e+11 1.4965 0.2212
```

Good thing we ran an ANOVA on all three of our new models! Looking at the output we can see that the difference in Model 1 (with one input variable) and Model 2 was significant (p value less than 0.05). However, our newest model 3 (which removed `year_renovated`) didn't actually change our model much (p value of 0.2212). Although, we know that `year_renovated` has it's issues so we will continue to use Model 3.

b.vii. Casewise diagnostics

Next we want to perform some case wise diagnostics to identify outlines and/or influential cases. We will do this by using some of the great diagnostic functions readily available in R. Storing each function's output in the main data frame.

```
diagnostics <- data.frame(residuals<-resid(lotSF_lm3))
diagnostics$standardized.residuals<-rstandard(lotSF_lm3)
diagnostics$studentized.residuals<-rstudent(lotSF_lm3)
diagnostics$cooks.distance<-cooks.distance(lotSF_lm3)
diagnostics$dfbeta<-dfbeta(lotSF_lm3)
diagnostics$dffit<-dffits(lotSF_lm3)
diagnostics$leverage<-hatvalues(lotSF_lm3)
diagnostics$covariance.ratios<-covratio(lotSF_lm3)
```

After creating our new data frame, lets take a peak at the data and save it as a csv.

```
write.table(diagnostics, "Casewise diagnostics.csv", sep = ",", row.names
= FALSE)
head(diagnostics,3)
```

```
##   residuals....resid.lotSF_lm3. standardized.residuals studentized.residuals
## 1                     -15079.16            -0.04199672           -0.04199508
## 2                     -74868.28            -0.20851485           -0.20850706
## 3                     -90149.05            -0.25109839           -0.25108921
##   cooks.distance dfbeta.(Intercept) dfbeta.sq_ft_lot
## 1   4.198458e-08       6.389672e-02     7.139457e-06
## 2   1.083747e-06       9.671587e-01     3.973133e-05
## 3   4.740549e-06      -2.210341e+01     5.520325e-05
##   dfbeta.square_feet_total_living dfbeta.bath_full_count         dffit
## 1                   -3.026966e-04          -3.482488e-01 -0.0004097868
## 2                   -2.151176e-03          -1.228884e+00 -0.0020819860
## 3                   -9.645816e-03           2.124416e+01 -0.0043544012
##       leverage covariance.ratios
## 1 9.520896e-05          1.000407
## 2 9.969440e-05          1.000399
## 3 3.006562e-04          1.000594
```

b.viii. Standardized residuals

Now that we know we have what we need in our diagnostics data frame, lets take a closer look at the standardized residuals that fell above or below +-2. We can do this by creating a new varable that indicates whether or not the residual is above or below +-2 (TURE/FALSE).

```
diagnostics$High.residuals <- diagnostics$standardized.residuals > 2 | diagnostics$standardized.residual
names(diagnostics)
```

```
## [1] "residuals....resid.lotSF_lm3." "standardized.residuals"
## [3] "studentized.residuals"         "cooks.distance"
## [5] "dfbeta"                        "dffit"
## [7] "leverage"                      "covariance.ratios"
## [9] "High.residuals"
```

b.ix. Sum of large residuals

Next we will sum up the total TRUE values to get a count of the standardized residuals that fall outside +-2.

```
sum(diagnostics$High.residuals)
```

```
## [1] 316
```

```
nrow(dfhousing2)
```

```
## [1] 12811
```

```
(sum(diagnostics$High.residuals)/nrow(dfhousing2))*100
```

```
## [1] 2.46663
```

By looking at this output, we can conclude that 316 of our values have higher than normal residuals (or 2.4% of our data set). Are any of these residuals extreme? Lets take a quick peak at the max and min residuals.

```
max(diagnostics$standardized.residuals)
```

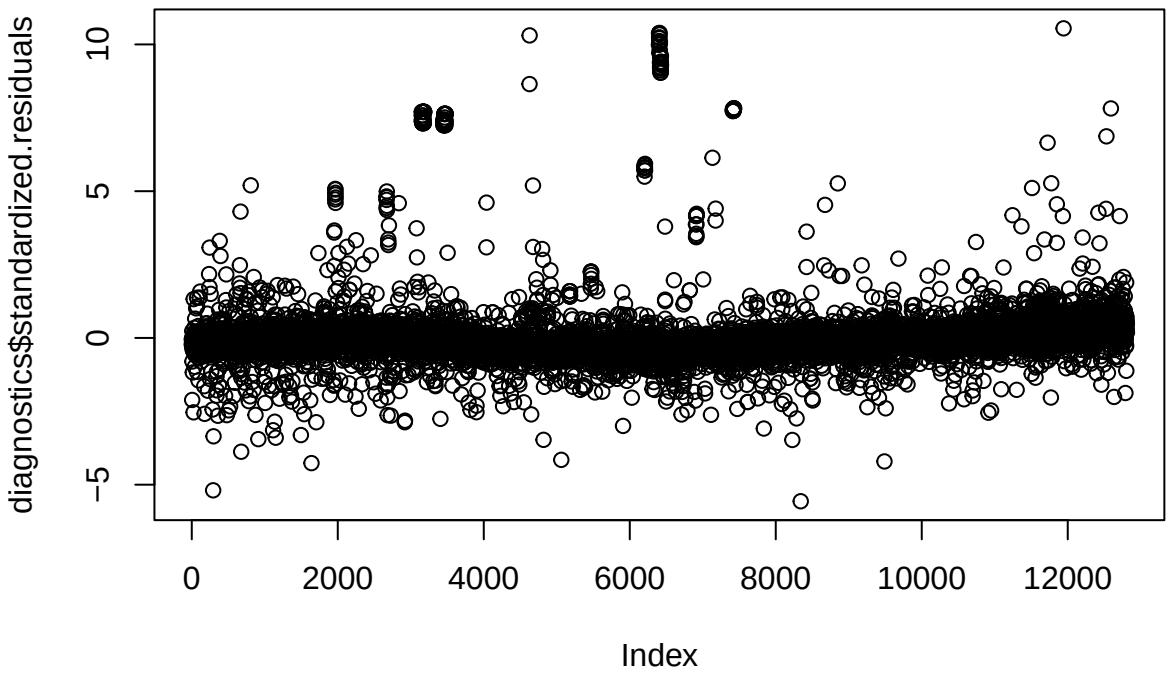
```
## [1] 10.54791
```

```
min(diagnostics$standardized.residuals)
```

```
## [1] -5.56353
```

Wow. 10.4 is certainly high and should be looked into further. How does this comapre to the others? We can quickly visualize it with a scatter plot below,

```
plot(diagnostics$standardized.residuals)
```



We can see the 316 values that fall outside the +-2 norm. As well as the extremes sitting over 5.

b.x. Variables with large residuals

Now how can we look at only the cases where the residual is high? Lets first add our High.residual variable back to the original data frame.

```
dfhousing2_res <- cbind(dfhousing2, diagnostics[,9, drop=FALSE])
```

Now we can filter for TRUE in the combined data frame in order to take a look at the line items with high residuals.

```
High.residuals.Only <- subset(dfhousing2_res, High.residuals!= "FALSE")
head(High.residuals.Only,3)
```

```
##     Sale.Date Sale.Price sale_reason sale_instrument sale_warning sitetype
## 6    1/3/2006     184667           1              15        18 51       R1
## 25  1/11/2006     265000           1               3                    R1
## 178  3/3/2006     390000           1               3                    R1
##                           addr_full  zip5 ctyname postalctyn       lon      lat
## 6                   8101 229TH DR NE 98053           REDMOND -122.0341 47.67545
## 25            25149 NE PATTERSON WAY 98053           REDMOND -122.0032 47.65814
## 178 13414 WOODINVILLE REDMOND RD NE 98052           REDMOND -122.1343 47.72058
##     building_grade square_feet_total_living bedrooms bath_full_count
## 6                7                     4160        4               2
```

```
## 25              10                     4920        4               4
## 178             11                     5800        5               4
##     bath_half_count bath_3qtr_count year_built year_renovated current_zoning
## 6                 1               1       2005              0          URPSO
## 25                1               0       2007              0            RA5
## 178               1               0       2008              0        RA2.5SO
##     sq_ft_lot prop_type present_use High.residuals
## 6        7280         R           2           TRUE
## 25     112650         R           2           TRUE
## 178     63162         R           2           TRUE
```

b.xii. Leverage, cooks distance, and covariance rations

We had calculated Leverage, cooks distance, and covariance rations above in addition to the standardized residuals. Lets take a closer look at these diagnostics for the observations with high standardized residuals. Lets start by adding all of our diagnostic measures to the original data frame.

```
dfhousing2_resAll <- cbind(dfhousing2, diagnostics)
```

We can then filter on he observations with the high residuals and sort by the highest cook's distance.

```
High.residualsAll.Only <- subset(dfhousing2_resAll, High.residuals!= "FALSE")
High.residualsAll.CookSort <- High.residualsAll.Only[order(-High.residualsAll.Only$cooks.distance),]
head(High.residualsAll.CookSort,3)
```

```
##      Sale.Date Sale.Price sale_reason sale_instrument sale_warning sitetype
## 4649  3/2/2010    4400000           1               3        35 45       R1
## 295  3/28/2006     270000           1               3                    R1
## 8377 6/13/2013      14000           1              26                    R1
##              addr_full  zip5 ctyname postalctyn       lon      lat
## 4649 12053 154TH PL NE 98052           REDMOND -122.1345 47.70950
## 295   5806 249TH CT NE 98053           REDMOND -122.0053 47.65706
## 8377  20210 NE 85TH ST 98053           REDMOND -122.0724 47.68034
##      building_grade square_feet_total_living bedrooms bath_full_count
## 4649              6                     2410        3               1
## 295              11                     5060        4              23
## 8377             12                     8750        5               2
##      bath_half_count bath_3qtr_count year_built year_renovated current_zoning
## 4649               0               1       1935              0          A10SO
## 295                1               0       2016              0            RA5
## 8377               2               3       1996              0            RA5
##      sq_ft_lot prop_type present_use residuals....resid.lotSF_lm3.
## 4649   1327090         R           2                       3618303
## 295      89734         R           0                      -1758874
## 8377   1631322         R           2                      -1933222
##      standardized.residuals studentized.residuals cooks.distance
## 4649              10.307652             10.350272      1.2310026
## 295               -5.190932             -5.196199      0.8286914
## 8377              -5.563530             -5.570048      0.5248853
##      dfbeta.(Intercept) dfbeta.sq_ft_lot dfbeta.square_feet_total_living
## 4649       1.753517e+03     1.282253e-01                   -1.857697e+00
## 295        1.065964e+04    -9.873832e-03                    3.338860e+00
```

```
## 8377       1.061709e+03    -8.117206e-02                   -6.682638e-02
##      dfbeta.bath_full_count     dffit   leverage covariance.ratios
## 4649           2.458680e+02  2.228186 0.04429194          1.012368
## 295           -1.056688e+04 -1.822497 0.10954086          1.113943
## 8377           4.048972e+02 -1.450677 0.06352158          1.057875
##      High.residuals
## 4649           TRUE
## 295            TRUE
## 8377           TRUE
```

Based on the observations above, we can see that the extreme standardized residual of 10.54791 also has a cooks distance above 1. Meaning that it has a large overall influence on the Model. However, this observation does not have a significantly high leverage (0.044). This does not mean that this observation is not influential. Low leverage may just indicate that there are other observations near by. We may also want to look at how this observation is effecting our model. We can do that by looking at the covariance ratio. The observation in question (line 4649 from our data set excluding 0 beds and 0 baths) has a covariance ratio of 1.012. This shows us that the observation is positively impacting the outcome variable (positively correlated). After reviewing all of the Casewise diagnostics for this observation (4649) we can determine to remove it from the model. Lets remove this observation and create a new model below.

```
dfhousing3 <- dfhousing2[-c(4649),]
lotSF_lm4 <- lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count, data = dfhousing3)
```

b.xiii. Assumption of independence and state

Next we want to test whether or not our input variables are interdependent of one another, aka assumption of independence. WE can do this using the Durbin Watson test. Luckily the Car package in R comes with this great durbinWatsonTest function so we can easily test our model below.

```
durbinWatsonTest(lotSF_lm4)
```

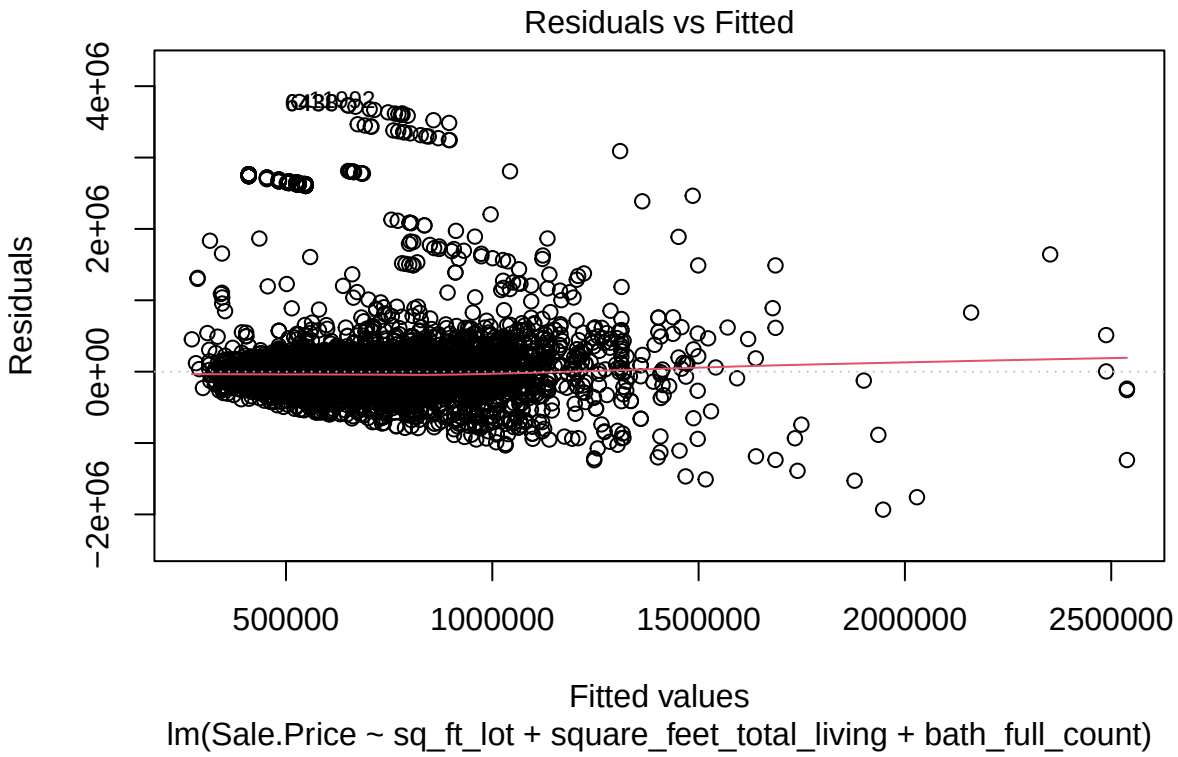
```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.7280234     0.5439502       0
##  Alternative hypothesis: rho != 0
```

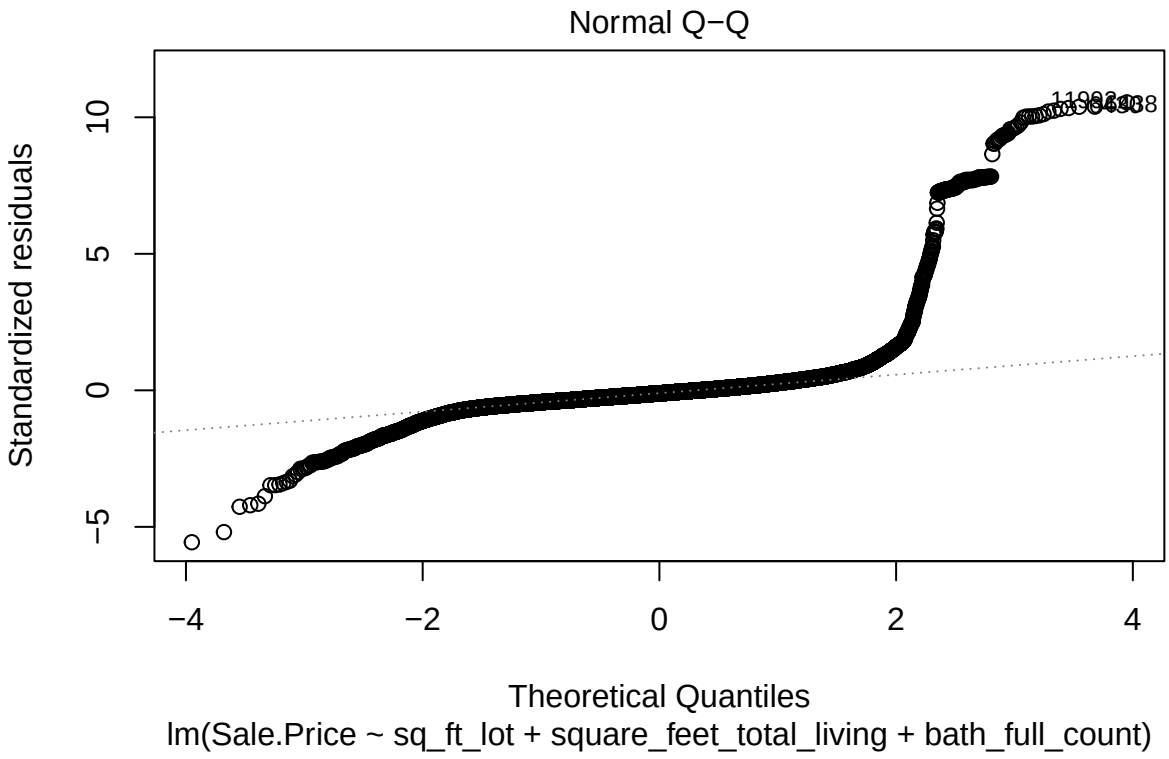
Based off the output from the Durbin Watson test, with a test statistic of 0.544 and p value less than 0.05. We can reject the null and conclude that the residuals in our regression model are autocorrelated. Therefore, the condition is met.

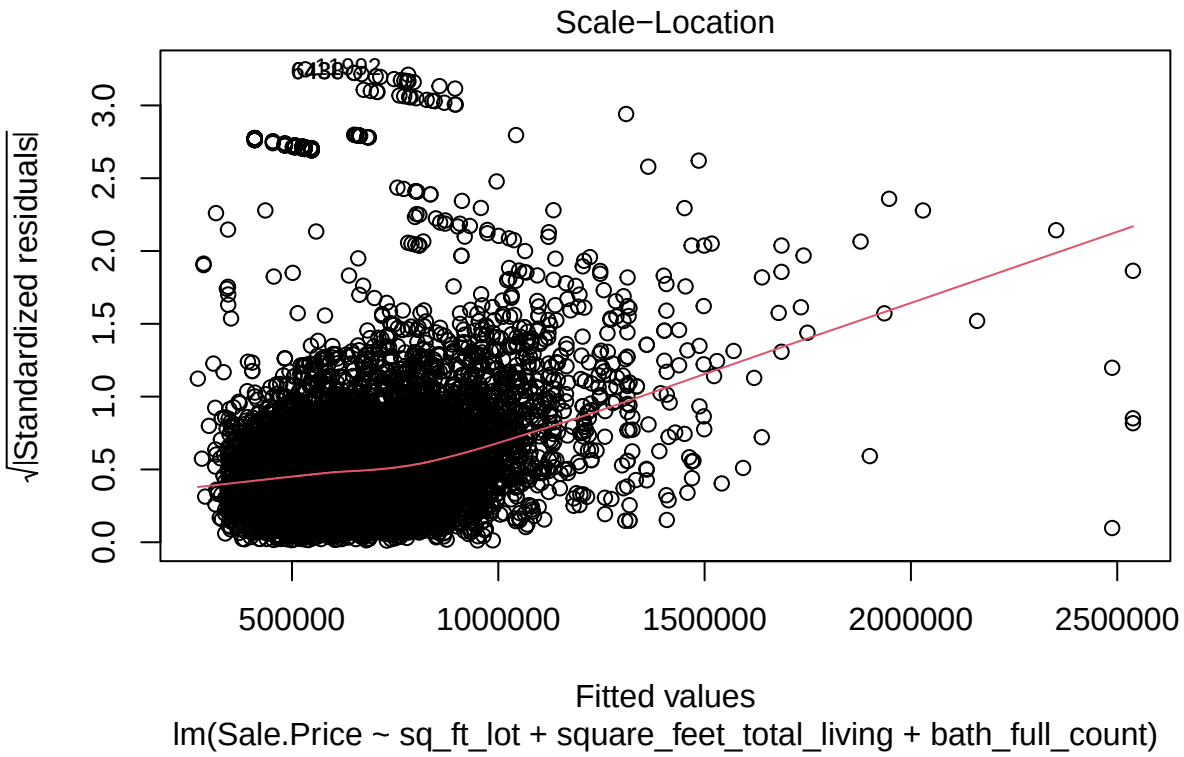
b.xiv. Assumptions related to the residuals using the plot() and hist()

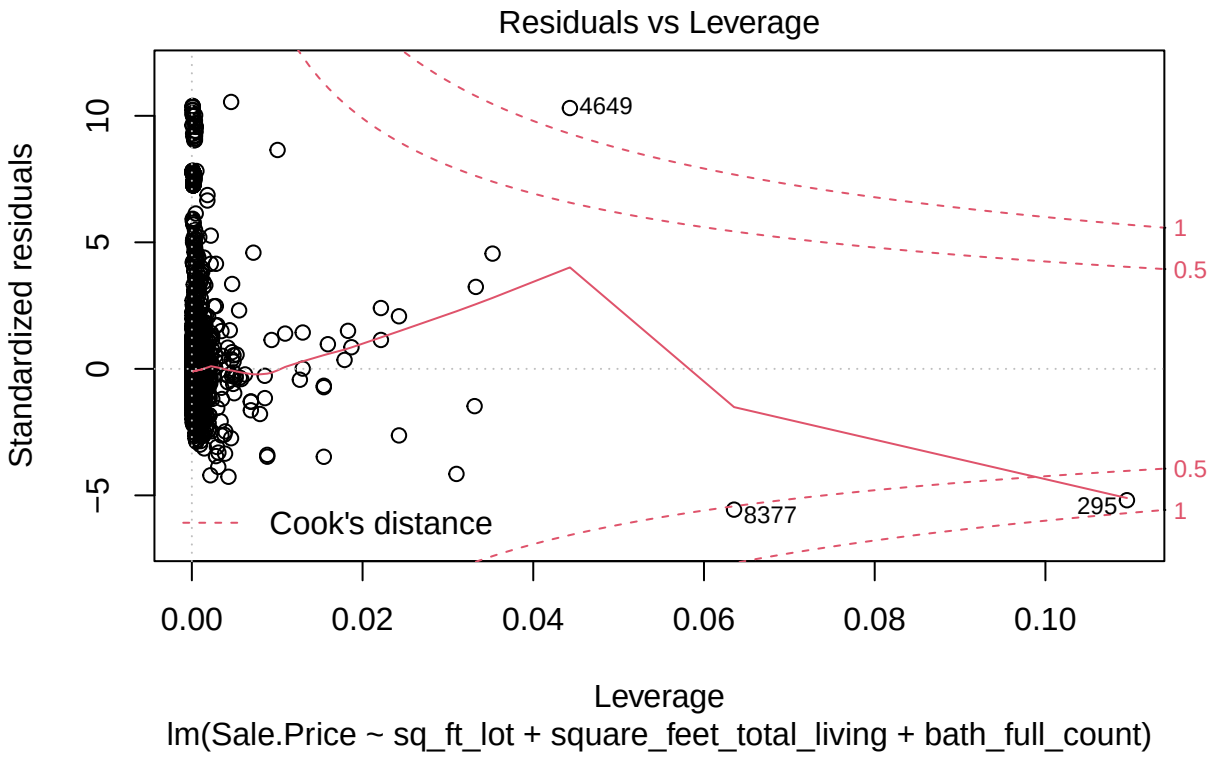
In order to check the assumptions related to the residuals, we need to plot our diagnostics data. First lets remove lets plot the model to visualize the diagnostics (note that this is the model where we removed the one observation with the highest standardized residuals)

```
plot(lotSF_lm4)
```


Residuals vs Fitted
Residuals
4e+06
2e+06
0e+00
−2e+06
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count)


Normal Q−Q
Standardized residuals
10
5
0
−5
−4
−2
0
2
4
Theoretical Quantiles
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count)

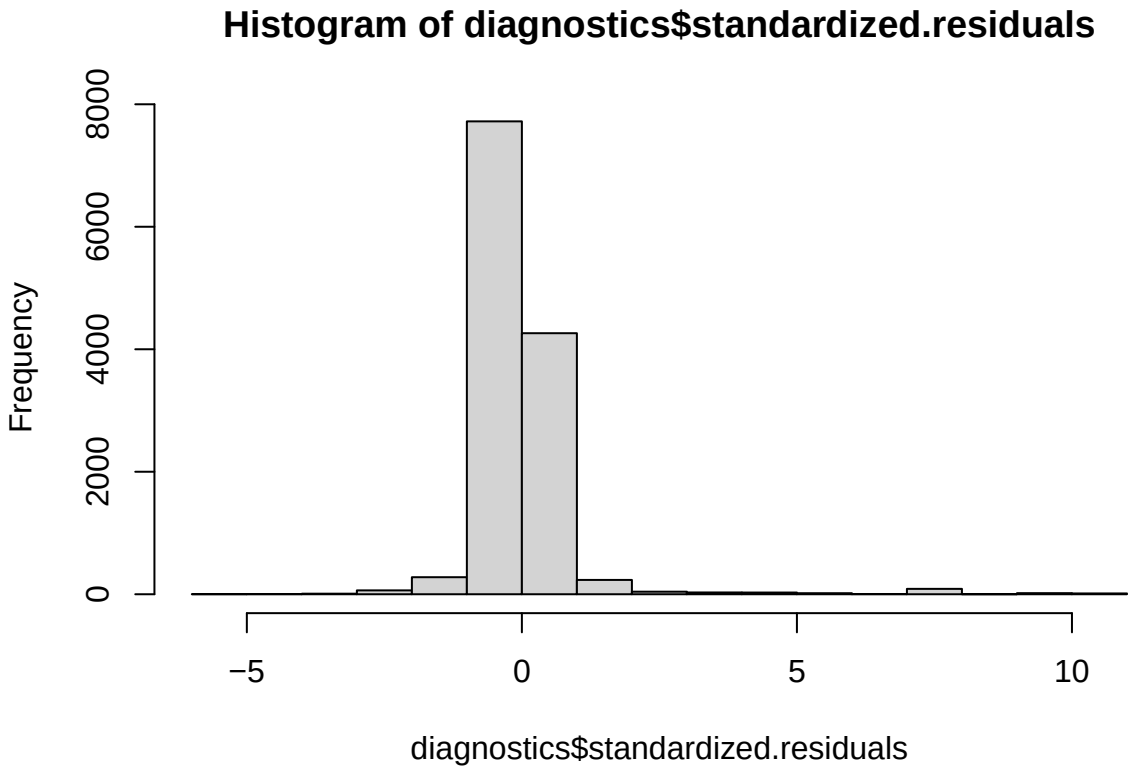

Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
3.0
500000
1000000
1500000
2000000
2500000
Fitted values
lm(Sale.Price ~ sq_ft_lot + square_feet_total_living + bath_full_count)



So far these plots are not indicating a good model. . . i.e. the Q-Q plot should be much more evenly distributed and we should not see so many outlires in our Residuals vs fitted plot.

Lets look at the same data in the form of a histogram.

```
hist(diagnostics$standardized.residuals)
```



As we can see above, our standardized residuals are not every distributed even in the slightest. While we may have a large number around -2 to 2, we can see the over 300 observations that fall outside this range any where from -4 to 10.

b.xv. Is this regression model unbiased?

Is this regression model unbiased? YES. Based off the number of issues we found in the diagnostics above, we can say with certainty that out model is bias. This model cannot be used to make any assumptions about the entire population. A lot more work is needed to be done to understand the data better and build a much better model.

References

Field, A., J. Miles, and Z. Field. 2012. Discovering Statistics Using R. SAGE Publications. https://books.google.com/books?id=wd2K2zC3swIC.

Lander, J. P. 2014. R for Everyone: Advanced Analytics and Graphics. Addison-Wesley Data and Analytics Series. Addison-Wesley. https://books.google.com/books?id=3eBVAgAAQBAJ.