Linear Regression

Vladimir Sokolov

9/26/2022

This notebook explores King County House Sales data from Kaggle.

#Linear regression finds a line of best fit for a target using predictors. Linear regression is a high bias-low variance model which can under fit data but is good for linearly separable data sets. Usage and model are simple and easy to understand which is a strong strength.

Load the kc_housing_data.csv file and change waterfront into a factor.

```
df <- read.csv("kc_house_data.csv")
str(df)
```

```
## 'data.frame':    21613 obs. of  21 variables:
##  $ id           : num  7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
##  $ date         : chr  "20141013T000000" "20141209T000000" "20150225T000000" "20141209T000000" ...
##  $ price        : num  221900 538000 180000 604000 510000 ...
##  $ bedrooms     : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms    : num  1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
##  $ sqft_living  : int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ sqft_lot     : int  5650 7242 10000 5000 8080 101930 6819 9711 7470 6560 ...
##  $ floors       : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ waterfront   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ view         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition    : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ sqft_above   : int  1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
##  $ sqft_basement: int  0 400 0 910 0 1530 0 0 730 0 ...
##  $ yr_built     : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
##  $ yr_renovated : int  0 1991 0 0 0 0 0 0 0 0 ...
##  $ zipcode      : int  98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
##  $ lat          : num  47.5 47.7 47.7 47.5 47.6 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
##  $ sqft_living15: int  1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
##  $ sqft_lot15   : int  5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
```

```
df$waterfront <- factor(df$waterfront)
str(df)
```

```
## 'data.frame':    21613 obs. of  21 variables:
##  $ id           : num  7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
##  $ date         : chr  "20141013T000000" "20141209T000000" "20150225T000000" "20141209T000000" ...
##  $ price        : num  221900 538000 180000 604000 510000 ...
```

```
##  $ bedrooms     : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms    : num  1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
##  $ sqft_living  : int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ sqft_lot     : int  5650 7242 10000 5000 8080 101930 6819 9711 7470 6560 ...
##  $ floors       : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ waterfront   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ view         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition    : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ sqft_above   : int  1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
##  $ sqft_basement: int  0 400 0 910 0 1530 0 0 730 0 ...
##  $ yr_built     : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
##  $ yr_renovated : int  0 1991 0 0 0 0 0 0 0 0 ...
##  $ zipcode      : int  98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
##  $ lat          : num  47.5 47.7 47.7 47.5 47.6 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
##  $ sqft_living15: int  1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
##  $ sqft_lot15   : int  5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
```

Check for null values.

```
sapply(df, function(x) sum(is.na(x)))
```

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

#A. #Divide into 80/20 train/test.

```
set.seed(12345)
i <- sample(1:nrow(df), nrow(df)*.8, replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

#B. & C. #Data Exploration & Graphs

Explore distribution of floors with summary and histogram

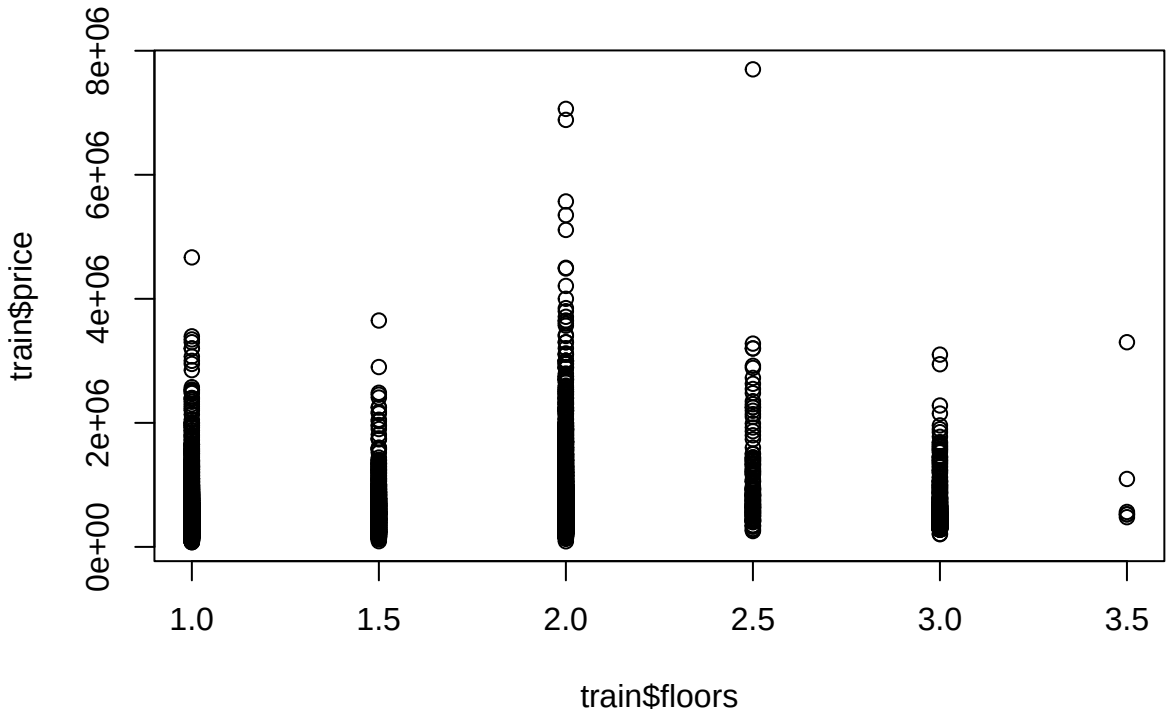
```
summary(train$floors)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   1.500   1.493   2.000   3.500
```

```
hist(train$floors)
```

Graph exploring impact of floors on house price

```
plot(train$floors, train$price)
```

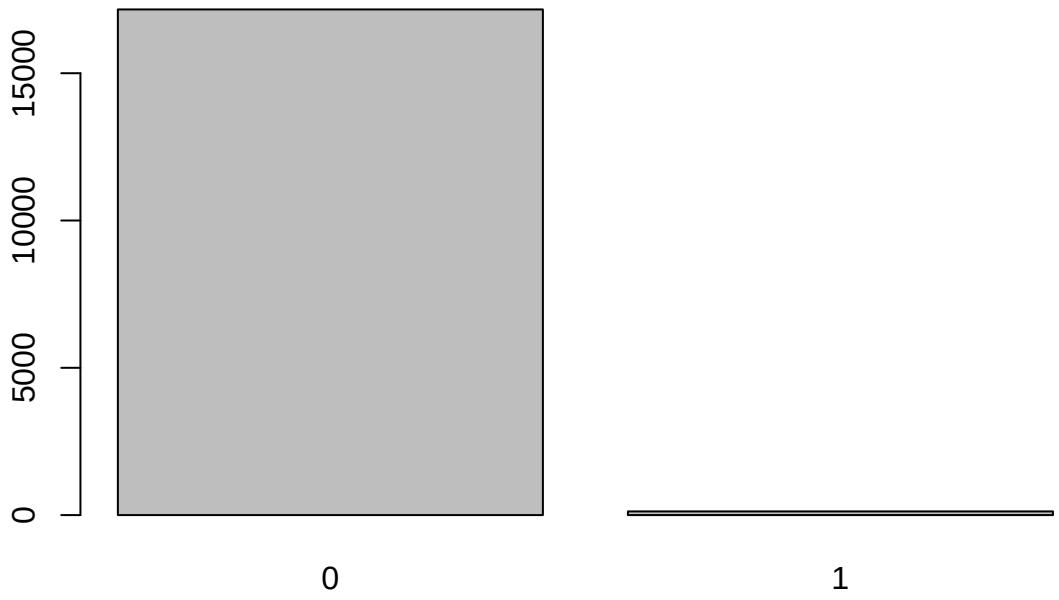


Explore distribution of waterfront with summary and barplot

```
summary(train$waterfront)
```

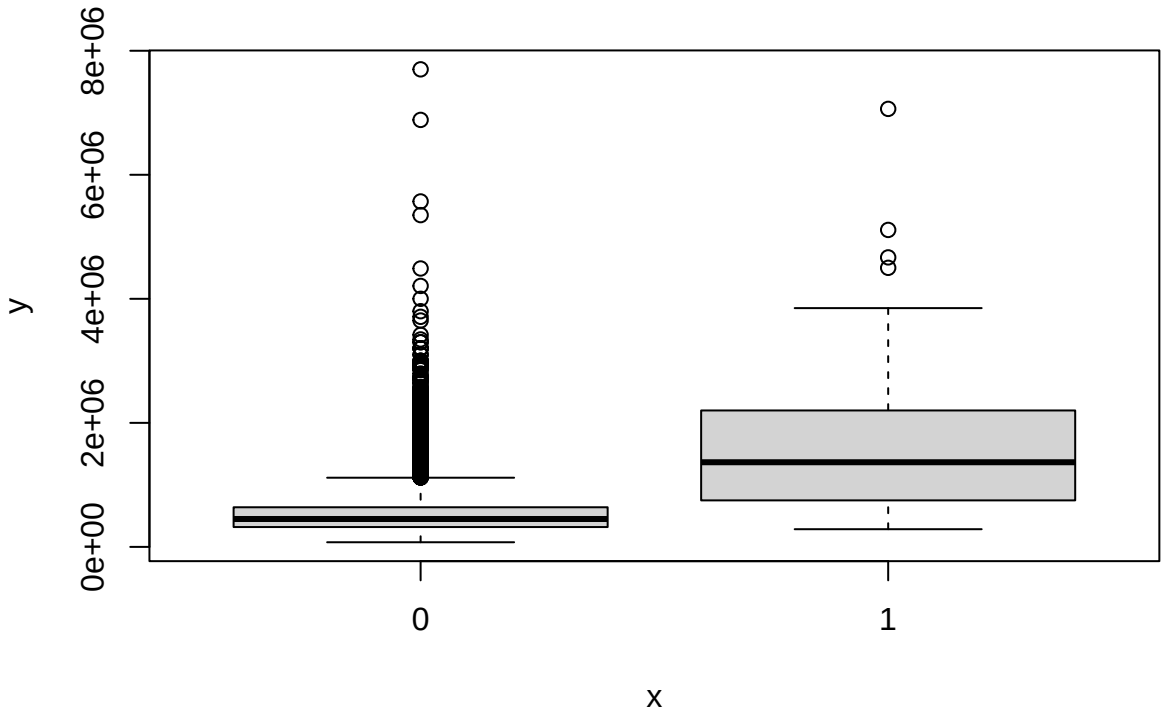
```
##     0     1
## 17165   125
```

```
counts <- table(train$waterfront)
barplot(counts)
```



Graph exploring impact of waterfront placement on house price

```
plot(train$waterfront, train$price)
```

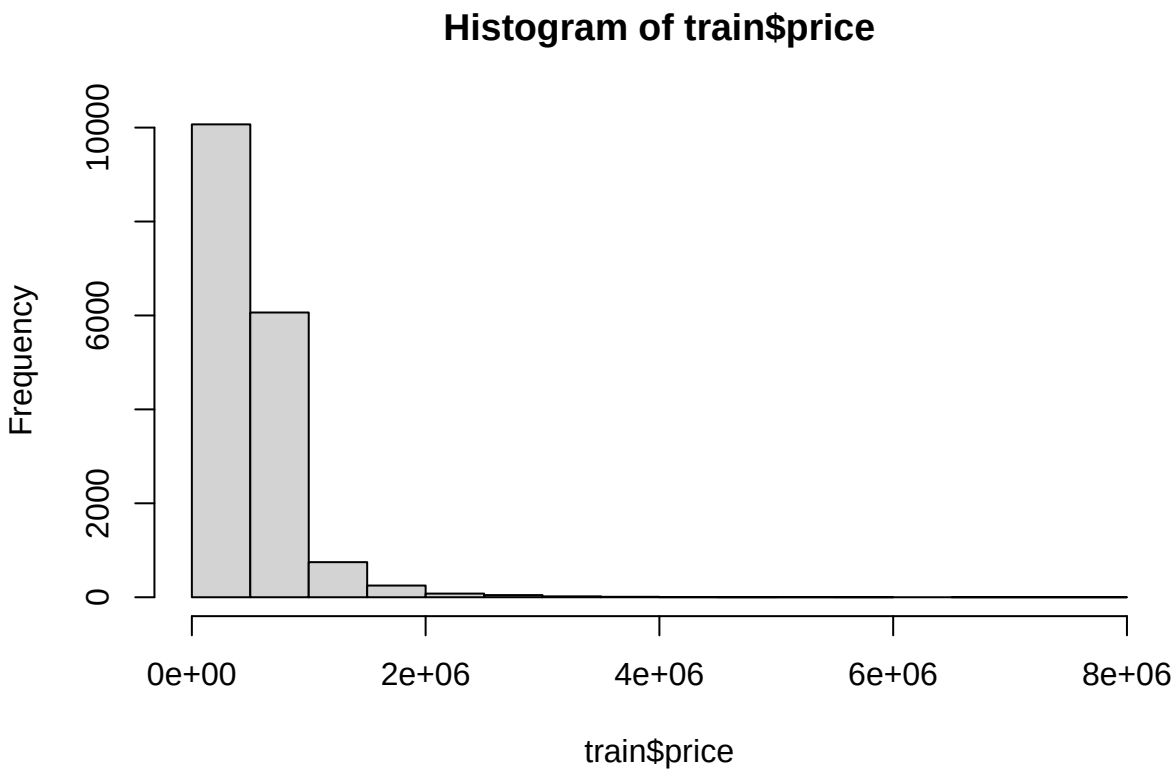


Explore the distribution of prices with summary and histogram

```
summary(train$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   75000  320000  450000  539062  643000 7700000
```

```
hist(train$price)
```



Explore the distribution of sqft_living and sqft_lot with summary and histogram

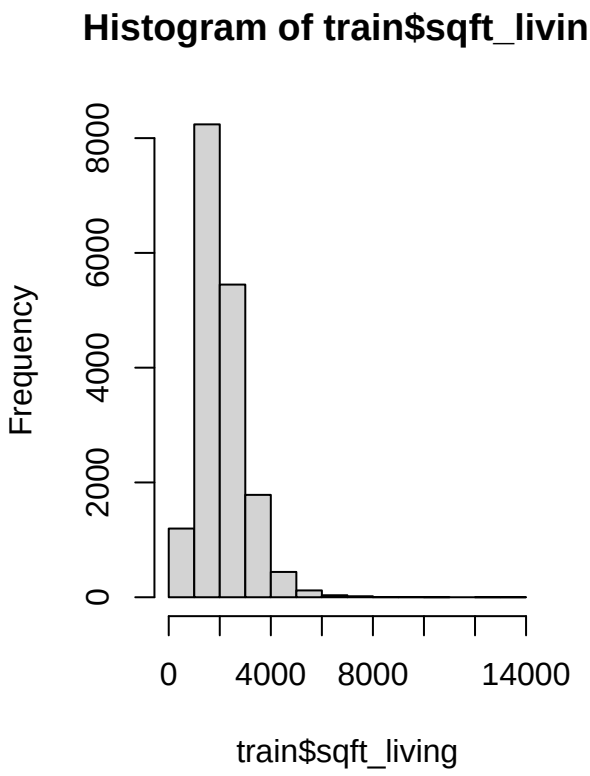
```
summary(train$sqft_living)
```

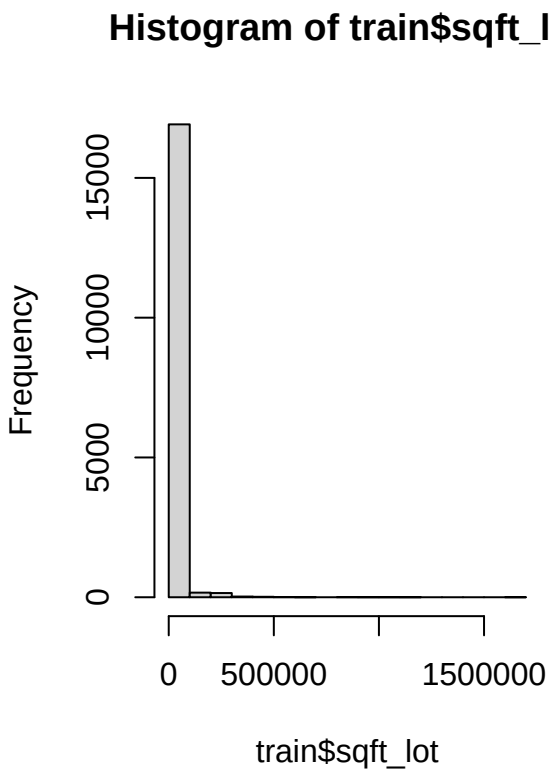
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     290    1428    1900    2075    2530   13540
```

```
summary(train$sqft_lot)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     572    5040    7620   15152   10660 1651359
```

```
par(mfrow=c(1,2))
hist(train$sqft_living)
hist(train$sqft_lot)
```





#D. #Linear Regression Model Impact of sqft_living on price Explanation: Residuals shows a summary of the residuals of the model. Coefficients shows each variable and the intercept. The estimated weight is the coefficient in the model produced(the number you are looking for), the standard error is how far it deviates from the sample(relatively smaller is better), the t value is how significant the variable is (bigger is better), and the p value is the probability of obtaining the t value (smaller is better). R marks good p values with stars and in this case marked both the intercept and sqft_living. At the bottom of the summary is information about how the model fits the training data. The standard error(smaller is better), the r-squared(closer to 1 is better) with an adjusted stat for multiple regression which calculates how much of the variation is predicted by the model, and F-statistic which determines if the predictors are significant (>1 is good). This model has an F-statistic >1 and low p value so it is confident that sqft_living effects price. At a R squared value of 0.5032 it does not capture enough predictors to reasonably estimate price using the formula produced by the model (price = -53497.702 + 285.572 * sqft_living).

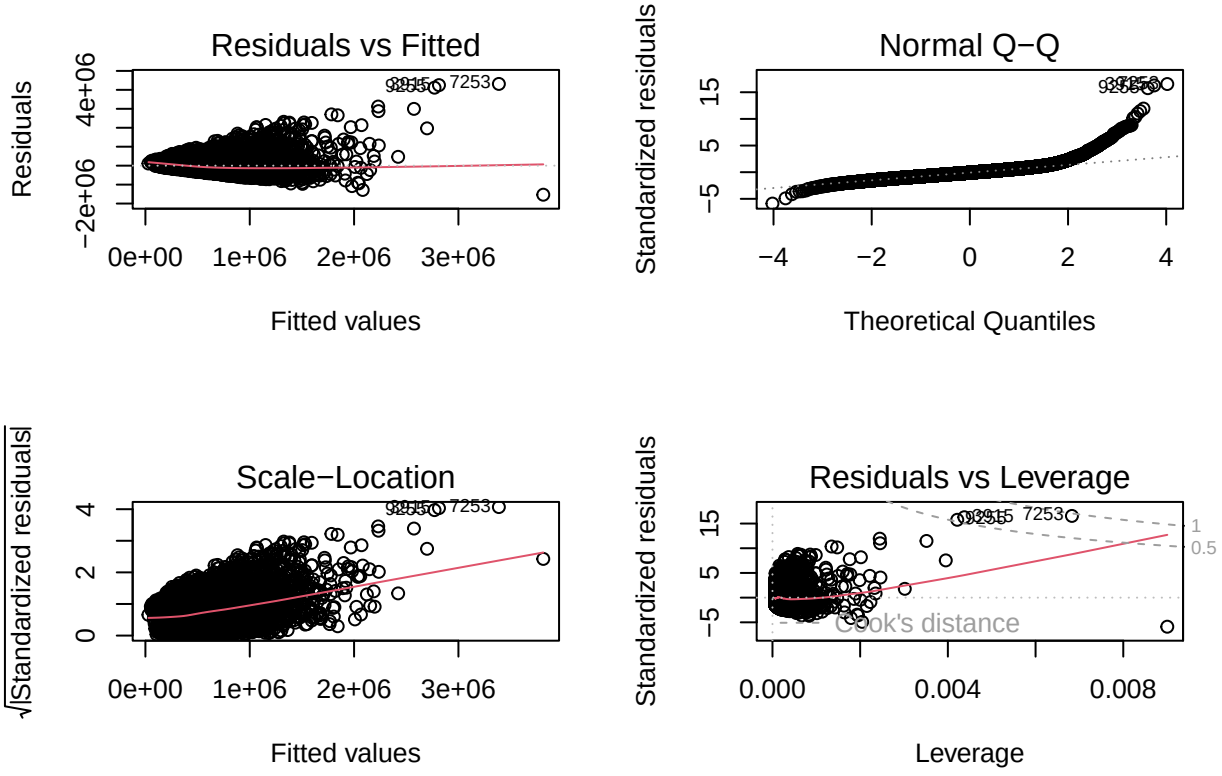
```
lm1 <- lm(price~sqft_living, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = price ~ sqft_living, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1533144  -147651   -23447   107384  4312358
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -53497.702   4899.842  -10.92   <2e-16 ***
## sqft_living    285.572      2.158  132.33   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 261600 on 17288 degrees of freedom
## Multiple R-squared:  0.5032, Adjusted R-squared:  0.5032
## F-statistic: 1.751e+04 on 1 and 17288 DF,  p-value: < 2.2e-16
```

#E. #Residuals of linear model 1 Eplanation: Residuals vs Fitted determines if linear regression is appropriate depending on whether the line is horizontal or not. Normal Q-Q determines if the residuals are normally distributed depending on whether the points follow the diagonal line. Scale-Location determines if the residuals have the same variance depending on whether the line is horizontal or not. Residuals vs Leverage determines if there are outliers or leverage points depending on whether they fall outside the area. This model has a fairly horizontal residuals vs fitted indicating a linear model is a good fit. Normal Q-Q deviates significantly at the tail end indicating that not all of the residuals are normally distributed. The scale location is far from horizontal indicating that the residuals do not all have the same variance. The residuals vs leverage has three points that are outside the normal area indicating that there are some observations with unusual values.

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

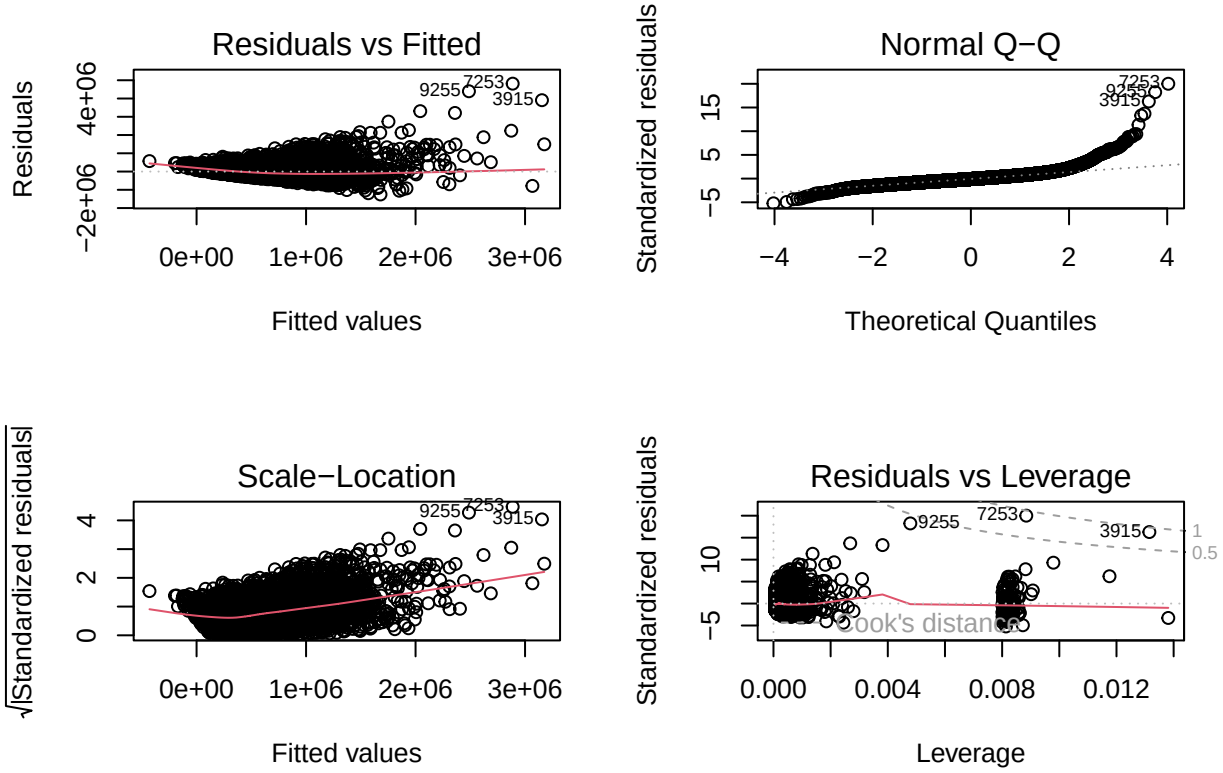


#F. #Multiple Linear Regression Model Impact of sqft_living, grade, and waterfront on price

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

```
##
## Call:
## lm(formula = price ~ sqft_living + grade + waterfront, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1254577  -133356   -21977   101761  4813143
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -579123.8    14382.0  -40.27   <2e-16 ***
## sqft_living     184.9        3.1   59.66   <2e-16 ***
## grade         95207.2     2424.9   39.26   <2e-16 ***
## waterfront1  829482.0    21809.3   38.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 241700 on 17286 degrees of freedom
## Multiple R-squared:  0.5761, Adjusted R-squared:  0.576
## F-statistic:  7830 on 3 and 17286 DF,  p-value: < 2.2e-16
```

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



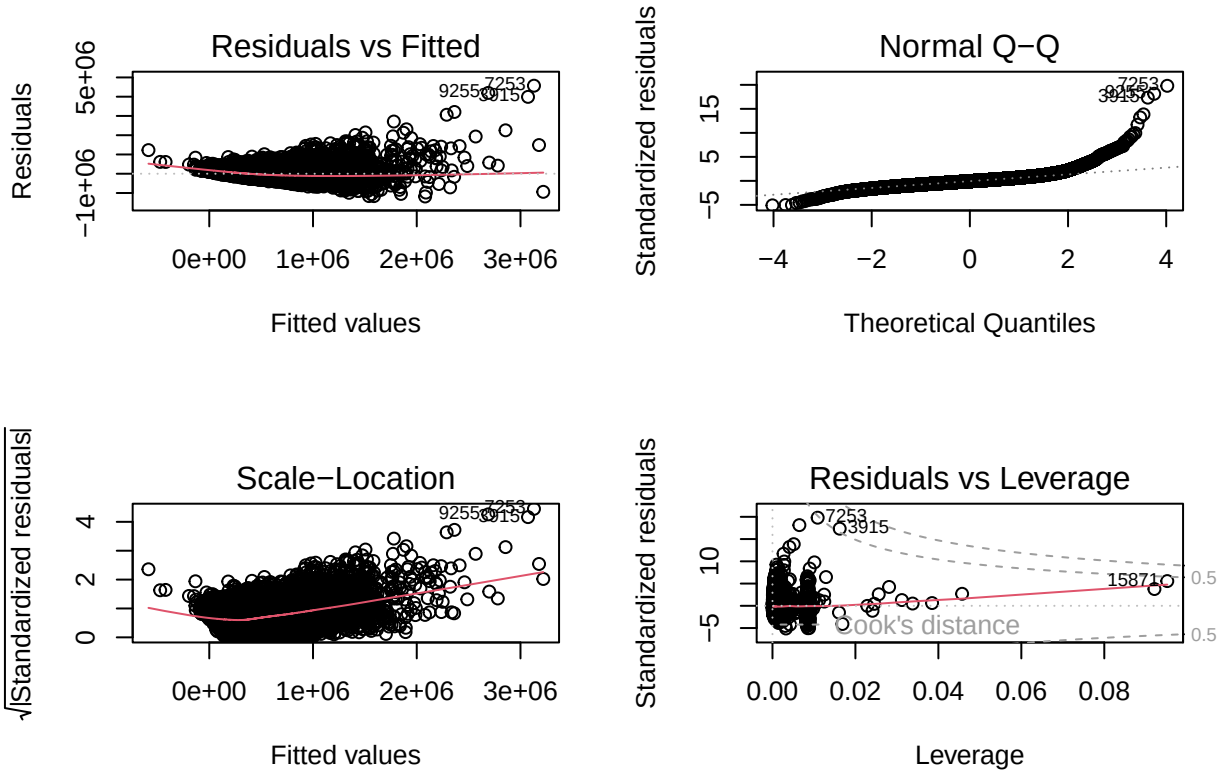
#G. #Third Model Impact of bedrooms, bathrooms, sqft_living, sqft_lot, waterfront, view, condition, and grade on price

```
lm3 <- lm(price~bedrooms+bathrooms+sqft_living+sqft_lot+waterfront+view+condition+grade, data=train)
summary(lm3)
```

```
##
## Call:
## lm(formula = price ~ bedrooms + bathrooms + sqft_living + sqft_lot +
##     waterfront + view + condition + grade, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1180858  -125117   -16532    95532  4569765
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.805e+05  1.931e+04 -35.239  < 2e-16 ***
## bedrooms    -3.545e+04  2.404e+03 -14.745  < 2e-16 ***
## bathrooms   -1.850e+04  3.655e+03  -5.062 4.20e-07 ***
## sqft_living  2.083e+02  3.842e+00  54.216  < 2e-16 ***
## sqft_lot    -3.395e-01  4.264e-02  -7.961 1.81e-15 ***
## waterfront1  5.776e+05  2.276e+04  25.378  < 2e-16 ***
## view         6.337e+04  2.631e+03  24.082  < 2e-16 ***
## condition    5.749e+04  2.782e+03  20.665  < 2e-16 ***
## grade        9.619e+04  2.452e+03  39.230  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 232100 on 17281 degrees of freedom
## Multiple R-squared:  0.6092, Adjusted R-squared:  0.609
## F-statistic:  3367 on 8 and 17281 DF,  p-value: < 2.2e-16
```

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



#H. #Comparison: lm1 had r^2 of .5032 lm2 had r^2 of .576 lm3 had r^2 of .609 Residuals vs fitted and Normal Q-Q were similar for all three. Scale location was a diagonal line in lm1 but changed to a checkmark shape in lm2 and lm2. Residuals vs leverage identified the same points for lm1 and lm2 but in lm3 one was different. I think lm3 is better because it has the highest r^2 value.

#I. #Correlation and MSE lm3 was better than lm2 which was better than lm1. These results probably happened because adding more predictors helped predict more of the variation so the correlation increased. The correlation could have most likely been improved with the exclusion or modification of the outlier values.

```
pred1 <- predict(lm1, newdata=test)
cor1 <- cor(pred1, test$price)
mse1 <- mean((pred1-test$price)^2)
rmse1 <- sqrt(mse1)

print(paste('1correlation: ' , cor1))
```

```
## [1] "1correlation:  0.670702141614655"
```

```
print(paste('1mse: ' , mse1))
```

```
## [1] "1mse:  68181667579.4331"
```

```
print(paste('1rmse: ' , rmse1))
```

```
## [1] "1rmse:  261116.195551776"
```

```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$price)
mse2 <- mean((pred2-test$price)^2)
rmse2 <- sqrt(mse2)

print(paste('2correlation: ' , cor2))
```

```
## [1] "2correlation:  0.745252865580616"
```

```
print(paste('2mse: ' , mse2))
```

```
## [1] "2mse:  55098233965.1365"
```

```
print(paste('2rmse: ' , rmse2))
```

```
## [1] "2rmse:  234730.130075234"
```

```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$price)
mse3 <- mean((pred3-test$price)^2)
rmse3 <- sqrt(mse3)

print(paste('3correlation: ' , cor3))
```

```
## [1] "3correlation:  0.76513836420218"
```

```
print(paste('3mse: ' , mse3))
```

```
## [1] "3mse:  51344722032.014"
```

```
print(paste('3rmse: ' , rmse3))
```

```
## [1] "3rmse:  226593.73784819"
```