House Price Prediction in the City of Windsor, Canada

Khushboo Jha

2016-12-21

Teacher's Name: Dr. Douglas Jones

Course Period: September 2016 to December 2016 (Fall)

House Prices: AN OVERVIEW

The principal benchmark used to assess property is current value (market value, actual value). Current value is considered to be the most likely selling price a willing buyer would pay and a willing seller would accept in an open market, where both parties are under no pressure either to buy or sell the property.

Research Objective:

To Investigate whether there is influence of factors such as bedrooms , bathrooms ,A/C or size of plot etc. in affecting the price of houses in city of Windsor,Canada.

Question - Problem statement:

When fairly assessed, similar properties located in a similar area will be assessed at a similar rate and it will help to predict price of future property.

What are the important factors that might impact and influence housing price in the City of Windsor, Canada ?

Hypothesis:

-Null Hypothesis H0: None of the predictors explain variance in price. -Alternate Hypothesis H1: Atleast one of the predictors explain variance in price.

Software:

We user R for statistical analysis. Below are the packages used.

```
library("AER")
```

```
## Warning: package 'AER' was built under R version 3.2.5
```

```
## Loading required package: car
```

```
## Loading required package: lmtest
```

```
## Warning: package 'lmtest' was built under R version 3.2.5
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.2.5
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Warning: package 'sandwich' was built under R version 3.2.5
```

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

```
## Warning: package 'survival' was built under R version 3.2.5
```

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

```
## Warning: package 'dplyr' was built under R version 3.2.5
```

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

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

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library("ggplot2")
```

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
library("car")
library("faraway")
```

```
## Warning: package 'faraway' was built under R version 3.2.5
```

```
##
## Attaching package: 'faraway'
```

```
## The following object is masked from 'package:survival':
##
##     rats
```

```
## The following objects are masked from 'package:car':
##
##     logit, vif
```

```
library("DAAG")
```

```
## Warning: package 'DAAG' was built under R version 3.2.5
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:faraway':
##
##     melanoma
```

```
##
```

```
## Attaching package: 'DAAG'
```

```
## The following objects are masked from 'package:faraway':
##
##     orings, ozone, vif
```

```
## The following object is masked from 'package:survival':
##
##     lung
```

```
## The following object is masked from 'package:car':
##
##     vif
```

Data Analysis Procedure

1. Data Cleaning

- We clean data for reliable and accurate analysis.One typical problem with raw data is that it contains miscoded values.Many data sets have missing values coded with a numerical value such as 00, ???9999 etc.Also there will be categorical variable that will be coded with numerical values as well.

2.Data Visualization. - We will do Graphical representations of the distribution of a variable in the data set this very useful to understand what kind of data set it is and necessary in data analysis.

3.Created model with all variables. - We will create a model with all the variables to understand inspection of the relation between the variables.

4.Used attribute Selector to find best attributes. - From the Previous model we can see that there is heterodoxy, in order to remove that we are trying to remove predictor with high correlation among each other to find the best predictor by performing log transformation.

5.Identify and fit suitable models to the data. - In order to the select the best model we perform backward step regression that gives a fit suitable model to the data.

6.Compared models to determine best model. - For model comparison we are performing ANOVA test and cross validation to determine the best model. We compare the previous model with the new model to find out which gives us the best model.

7.For robust analysis - do cross validation. - We perform cross validation to check which models are performing well to give the best accurate model.

Results:

Some major factors: which account for the home's value are

1. Location
2. Air-condition
3. Number of bath rooms
4. Lot size (lot dimensions)

From our analysis we got to know other factors in dataset are equally important. Only those independent variables that are highly correlated are redundant and needs to get removed.

WE ARE ABLE TO FIT A MODEL FOR THIS DATA SET, AND BASED ON OUR ANALYSIS WE WILL BE ALBE TO PREDICT FUTURE PRICE OF THE SIMILAR PROPERTY.

DATA DESCRIPTION

Sales prices of houses sold in the city of Windsor, Canada, during July, August and September, 1987.

Format:

A data frame containing 546 observations on 12 variables.

There are 2 quantifiable/continuous variables:

-Price: Sale price of a house. -Lot size: Lot size of a property in square feet.

There are 10 categorical/qualitative variables:

-Bedrooms: Number of bedrooms. -Bathrooms: Number of full bathrooms. -Stories: Number of stories excluding basement. -driveway Factor: Does the house have a driveway? -recreation Factor: Does the house have a recreational room? -full base Factor: Does the house have a full finished basement? -gas heat Factor: Does the house use gas for hot water heating? -Air con Factor: Is there central air conditioning? -Garage: Number of garage places. -prefer Factor: Is the house located in the preferred neighborhood of the city?

Prediction:

Price

Data and structure of the data

```
data("HousePrices")

data <- HousePrices

#structure of the data
str(data)
```

```
## 'data.frame':    546 obs. of  12 variables:
##  $ price     : num  42000 38500 49500 60500 61000 66000 66000 69000 83800 88500 ...
##  $ lotsize   : num  5850 4000 3060 6650 6360 4160 3880 4160 4800 5500 ...
##  $ bedrooms  : num  3 2 3 3 2 3 3 3 3 3 ...
##  $ bathrooms : num  1 1 1 1 1 1 2 1 1 2 ...
##  $ stories   : num  2 1 1 2 1 1 2 3 1 4 ...
##  $ driveway  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ recreation: Factor w/ 2 levels "no","yes": 1 1 1 2 1 2 1 1 2 2 ...
##  $ fullbase  : Factor w/ 2 levels "no","yes": 2 1 1 1 1 2 2 1 2 1 ...
##  $ gasheat   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ aircon    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 1 1 1 2 ...
##  $ garage    : num  1 0 0 0 0 0 2 0 0 1 ...
##  $ prefer    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

DATA PREPROCESSING

check for NA values: there are no Null values

```
which (is.na(data$price))
```

```
## integer(0)
```

Convert categorical variables into factor variables

Convert bedrooms, bathroom, stories, prefer and garage into factor variables

```
data$bedrooms <- as.factor(data$bedrooms)
#check for the number of levels of the bedroom
levels(data$bedrooms)
```

```
## [1] "1" "2" "3" "4" "5" "6"
```

```
data$bathrooms <- as.factor(data$bathrooms)
#check for the number of levels of the bedroom
levels(data$bathrooms)
```

```
## [1] "1" "2" "3" "4"
```

```
data$stories <- as.factor(data$stories)
#check for the number of levels of the bedroom
levels(data$stories)
```

```
## [1] "1" "2" "3" "4"
```

```
data$prefer <- as.factor(data$prefer)
levels(data$prefer)
```

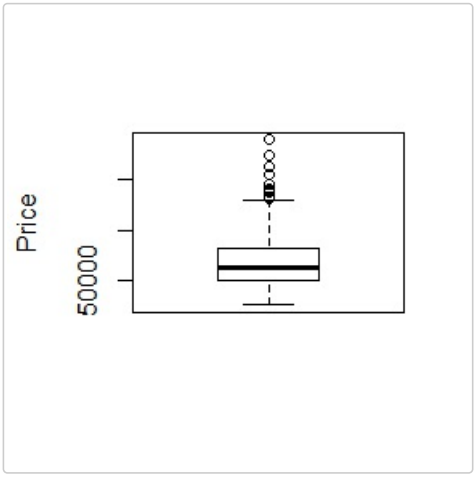
```
## [1] "no"  "yes"
```

```
data$garage <- as.factor(data$garage)
levels(data$garage)
```

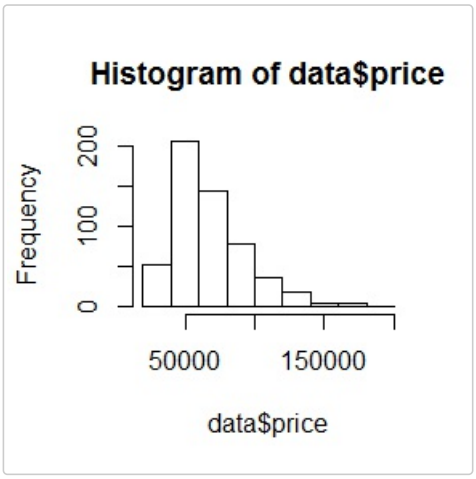
```
## [1] "0" "1" "2" "3"
```

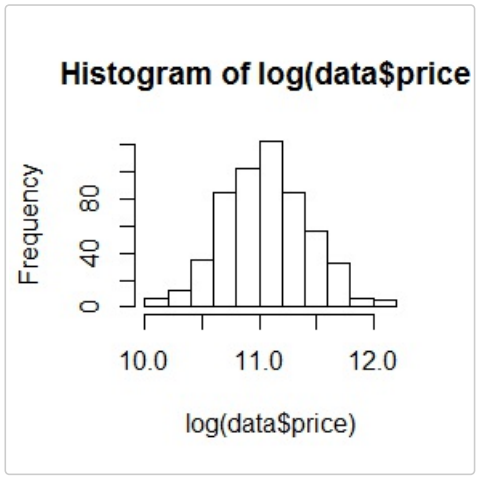
Check for Price outliers using boxplot

```
boxplot(data$price, ylab = "Price")
```



Distribution of Price variable





Checking summary & correlation

-From five number summary we observe mean and median, and find our data is little skewed for price and lotsize -There are no negative numbers -From correlation, we cand find that there is a relation between lot size and price, so lotsize is a good predictor of price

```
summary(data)
```

```
##      price           lotsize      bedrooms  bathrooms  stories  driveway
##  Min.   : 25000   Min.   : 1650   1:  2    1:402     1:227   no : 77
```

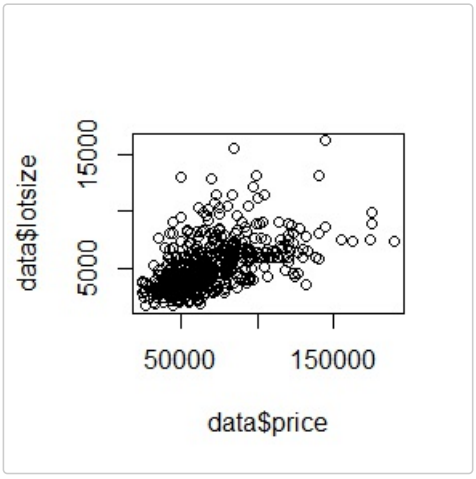
```
##  1st Qu.: 49125   1st Qu.: 3600   2:136    2:133     2:238   yes:469
##  Median : 62000   Median : 4600   3:301    3: 10     3: 40
##  Mean   : 68122   Mean   : 5150   4: 95    4:  1     4: 41
##  3rd Qu.: 82000   3rd Qu.: 6360   5: 10
##  Max.   :190000   Max.   :16200   6:  2
##  recreation fullbase  gasheat   aircon    garage  prefer
##  no :449    no :355   no :521   no :373   0:300   no :418
##  yes: 97    yes:191   yes: 25   yes:173   1:126   yes:128
##                                           2:108
##                                           3: 12
##
##
```

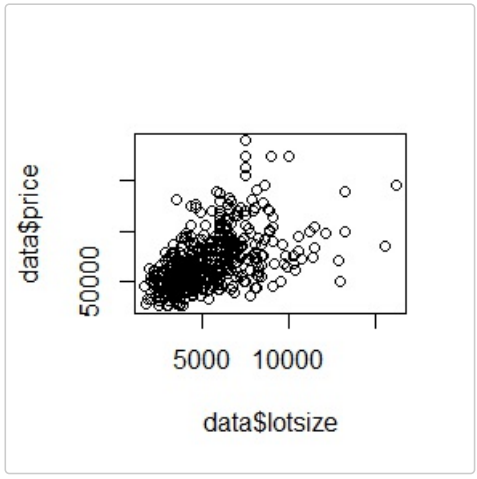
```
cor(data$price,data$lotsize)
```

```
## [1] 0.5357957
```

Price vs lot size plot

```
plot(data$price,data$lotsize)
plot(data$lotsize,data$price)
```





Top five and least five house prices

```
#top 5 prices
top5 <- data %>% arrange(desc(price))

head(top5)
```

```
##    price lotsize bedrooms bathrooms stories driveway recreation fullbase
## 1 190000    7420        4         2       3      yes         no       no
## 2 175000    8960        4         4       4      yes         no       no
## 3 175000    9960        3         2       2      yes         no      yes
## 4 174500    7500        4         2       2      yes         no      yes
## 5 163000    7420        4         1       2      yes        yes      yes
## 6 155000    7500        3         3       1      yes         no      yes
##   gasheat aircon garage prefer
## 1      no    yes      2    yes
## 2      no    yes      3     no
## 3      no     no      2    yes
## 4      no    yes      3    yes
## 5      no    yes      2     no
## 6      no    yes      2    yes
```

```
#Least house prices
tail(top5)
```

```
##     price lotsize bedrooms bathrooms stories driveway recreation fullbase
## 541 26500    2990        2         1       1       no         no       no
## 542 26000    3000        2         1       1      yes         no      yes
## 543 25245    2400        3         1       1       no         no       no
```

```
## 544 25000    3620        2         1       1      yes         no       no
## 545 25000    2910        3         1       1       no         no       no
## 546 25000    3850        3         1       2      yes         no       no
##     gasheat aircon garage prefer
## 541      no     no      1     no
## 542      no     no      2     no
## 543      no     no      0     no
## 544      no     no      0     no
## 545      no     no      0     no
## 546      no     no      0     no
```

Linear model: Price against all predictors

- basic regression -From the five number summary, mean and median, it is observed that our data is somewhat skewed (mean and median of price and lotsize).
- we also observe the important variables using the p- value
- adjusted r-square is around 66.66 % i.e our model is able to explain more than 66% of variance in the price, which is good.
- But we try to get better by using log transform.

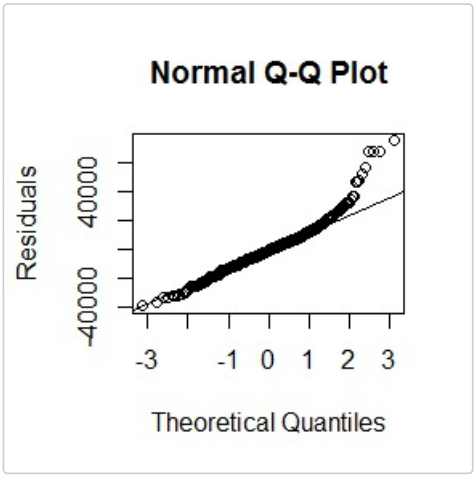
```
model1 <- lm(price ~ .,data=data)

summary(model1)
```

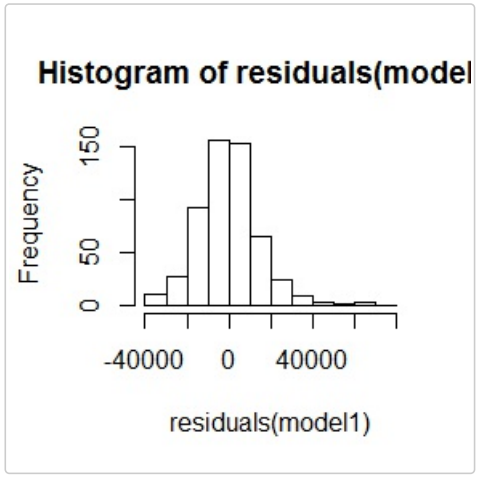
```
##
## Call:
## lm(formula = price ~ ., data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -39281  -9195   -909   7213  75412
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   22507.343  11000.467   2.046 0.041252 *
## lotsize           3.456      0.360   9.600  < 2e-16 ***
## bedrooms2     -2190.663  11016.988  -0.199 0.842462
## bedrooms3      1248.530  11053.813   0.113 0.910113
## bedrooms4      2891.382  11204.557   0.258 0.796466
## bedrooms5      3913.397  12184.006   0.321 0.748193
## bedrooms6     11004.913  15566.674   0.707 0.479910
## bathrooms2    13114.247   1756.304   7.467 3.45e-13 ***
## bathrooms3    29550.820   5157.685   5.729 1.70e-08 ***
## bathrooms4    76993.725  16352.076   4.708 3.20e-06 ***
## stories2       4856.200   1742.991   2.786 0.005527 **
## stories3      12505.681   2980.753   4.195 3.20e-05 ***
## stories4      19953.854   3124.777   6.386 3.77e-10 ***
## drivewayyes    6839.223   2068.830   3.306 0.001012 **
## recreationyes  4298.723   1909.197   2.252 0.024762 *
## fullbaseyes    5779.706   1617.102   3.574 0.000384 ***
## gasheatyes    12574.597   3267.652   3.848 0.000134 ***
## airconyes     12335.351   1580.236   7.806 3.23e-14 ***
## garage1        6065.258   1715.040   3.537 0.000441 ***
## garage2        9477.444   1886.412   5.024 6.95e-07 ***
## garage3        2510.257   4851.950   0.517 0.605116
## preferyes      9270.978   1708.003   5.428 8.73e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15410 on 524 degrees of freedom
## Multiple R-squared:  0.6797, Adjusted R-squared:  0.6668
## F-statistic: 52.95 on 21 and 524 DF,  p-value: < 2.2e-16
```

Normal Q-Q plot, Histogram Plot and Residual plot for basic regression model

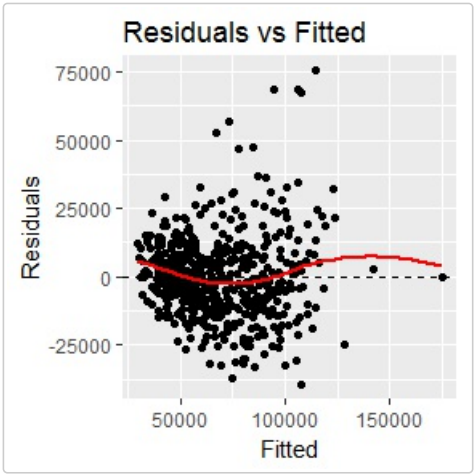
- Normality test: we can find that from the plot that residuals are almost normally distributed, there is heavy tail in the fourth quartile of the QQ plot
- Histogram shows distribution of residuals. It is skewed in the right.
- Residuals vs Fitted Plot: from the graph we can say that the red line is curved - - which indicates NON

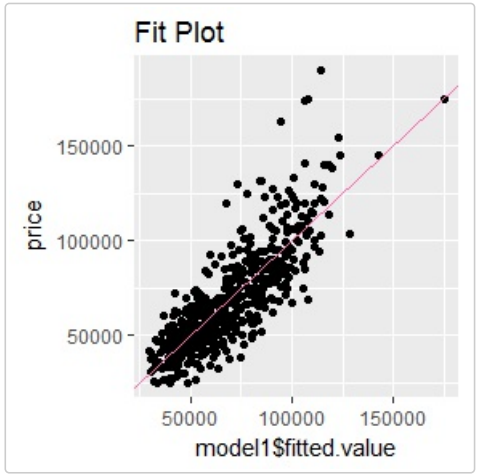
CONSTANT VARIANCE OF ERRORS -Fit plot: the fit plot looks fine but there are still some outliers






```
## `geom_smooth()` using method = 'loess'
```





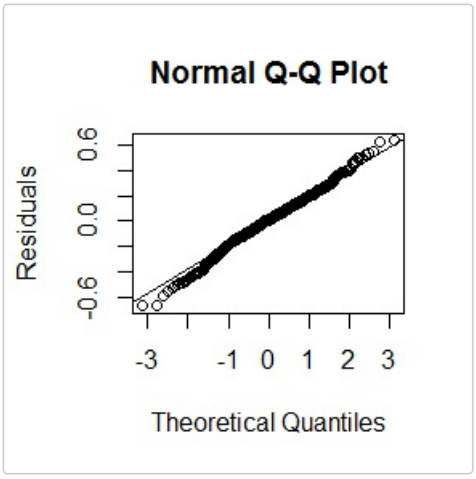
Linear model adjusted: considering the log transformation of the price variable

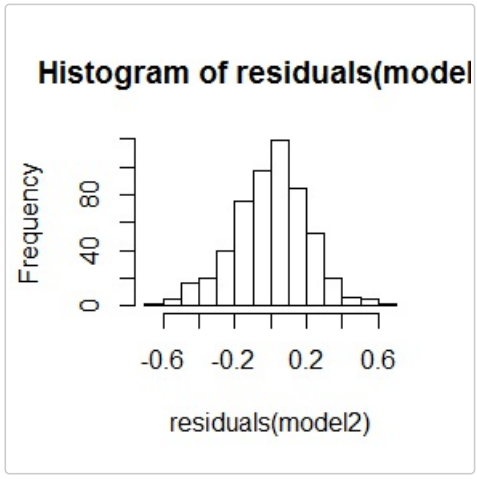
- applying log transformation to make the variable normally distributed
- considering the log tranfsformation of the price variable

```
#Lienar model adjusted
model2 <- lm(log(price) ~ . , data = data)
```

Normal Q-Q & Histogram Plot

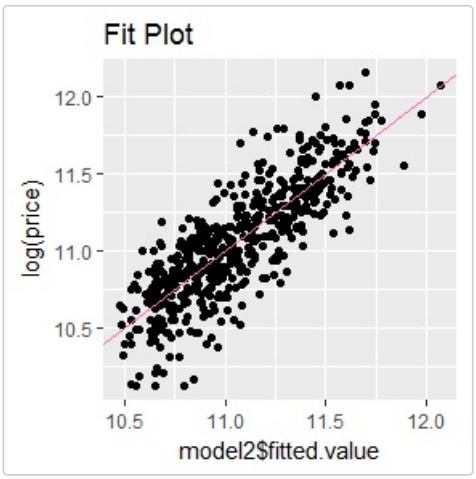
the erros are more normal



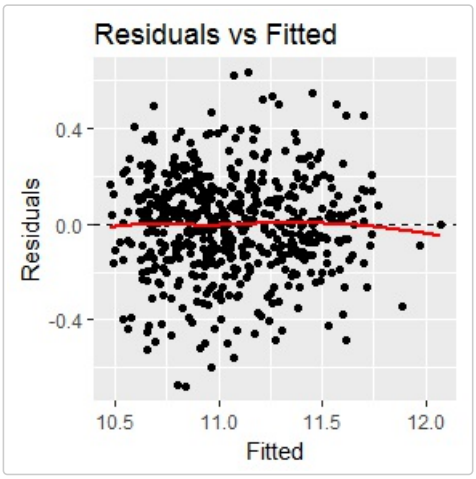


This plot fits the data well, compared to the pervious plot

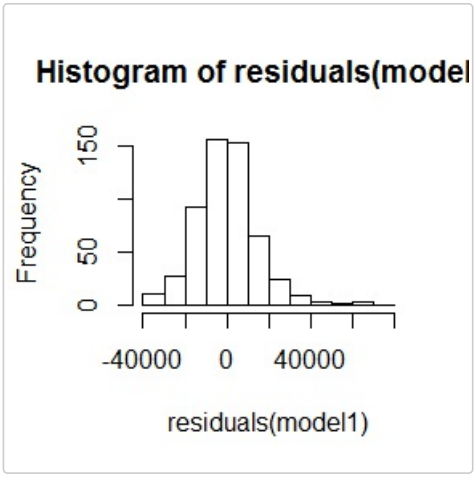
the red line is more linear indicating there is very little non linear relationships between the variables

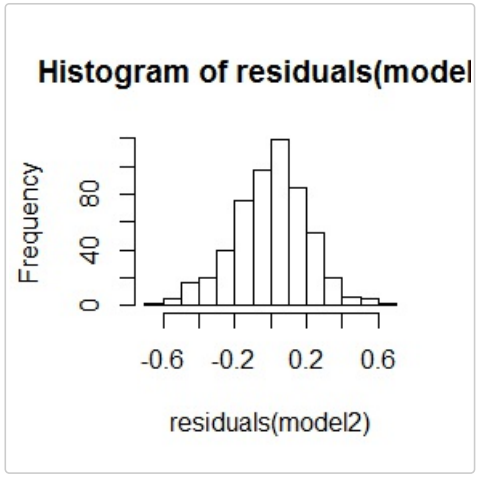



```
## `geom_smooth()` using method = 'loess'
```



Residual histogram comparison for model1 and model2



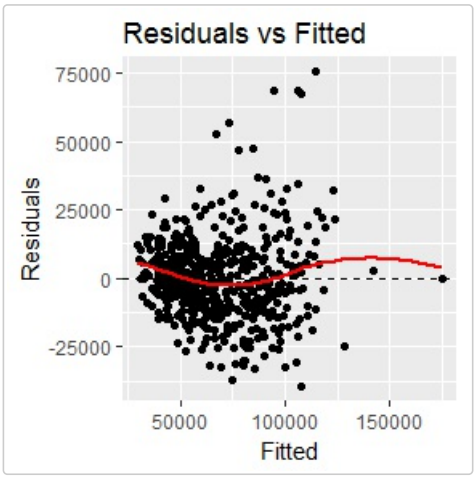


Residuals vs Fitted plot comparison for model1 and model2

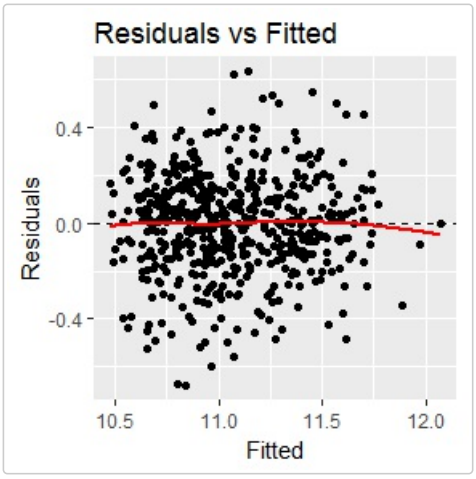
- Basic regression model residual plot compared to
- Residual plot after log transformation: the red line is more linear indicating there is very little non linear

relationships between the variables

```
## `geom_smooth()` using method = 'loess'
```

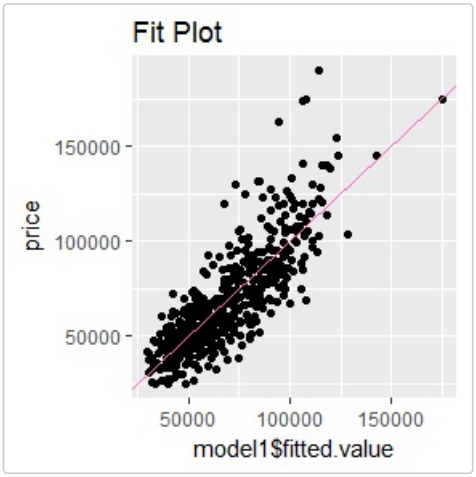


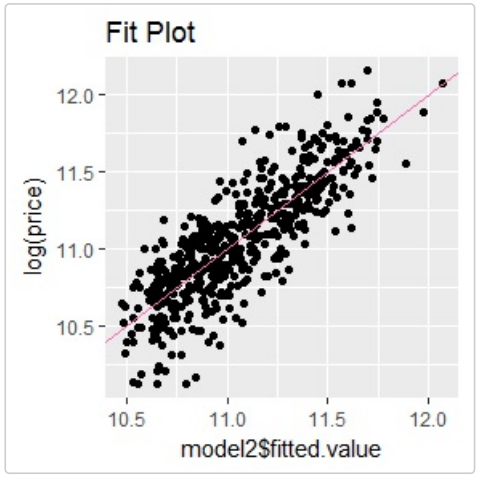

```
## `geom_smooth()` using method = 'loess'
```



Fit plot Comparison for model1 and model2

- Fit plot for basic regression model: the fit plot looks fine but there are still some outliers
- Fit plot after log transformation: this plot fits the data well than compared to the pervious plot





Backward Selection one at a time

- First we will remove the value bedroom because it has high p values
- Null hypothesis H0: bedrooms = 0
- since the value of p is greater than 0.05 we cannot reject the null hypothesis, so we drop bedroom, as it is a redundant variable (high cor relation between bathrooms and bedrooms) as proved by hypothesis testing.

```
g1 <- lm(log(price) ~ . - bedrooms,data=data)

anova(model2,g1)
```

```
## Analysis of Variance Table
##
## Model 1: log(price) ~ lotsize + bedrooms + bathrooms + stories + driveway +
##     recreation + fullbase + gasheat + aircon + garage + prefer
## Model 2: log(price) ~ (lotsize + bedrooms + bathrooms + stories + driveway +
##     recreation + fullbase + gasheat + aircon + garage + prefer) -
##     bedrooms
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    524 23.812
## 2    529 24.276 -5  -0.46407 2.0424 0.07129 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We consider the smaller model

- However, we further perform analysis to come up with better model, so we go with backward selection
- Performing a step wise regression in backward direction: in backward direction first we consider the whole model and then each indivdual model is dropped and the model with least AIC is the final model.
- stepwise regression says that inclusion of bedrooms yeilds the low AIC value

```
step(g1)
```

```
## Start:  AIC=-1665.76
## log(price) ~ (lotsize + bedrooms + bathrooms + stories + driveway +
##     recreation + fullbase + gasheat + aircon + garage + prefer) -
##     bedrooms
##
##              Df Sum of Sq    RSS     AIC
## <none>                     24.276 -1665.8
## - recreation  1    0.3108 24.587 -1660.8
## - gasheat     1    0.6188 24.895 -1654.0
## - driveway    1    0.8493 25.126 -1649.0
## - garage      3    1.2867 25.563 -1643.6
## - fullbase    1    1.1139 25.390 -1643.3
## - prefer      1    1.2698 25.546 -1639.9
## - aircon      1    2.8827 27.159 -1606.5
## - stories     3    3.3527 27.629 -1601.1
## - bathrooms   3    3.6398 27.916 -1595.5
## - lotsize     1    5.0357 29.312 -1564.8
```

```
##
## Call:
## lm(formula = log(price) ~ (lotsize + bedrooms + bathrooms + stories +
##     driveway + recreation + fullbase + gasheat + aircon + garage +
##     prefer) - bedrooms, data = data)
##
## Coefficients:
##   (Intercept)        lotsize     bathrooms2     bathrooms3     bathrooms4
##     1.037e+01      5.159e-05      1.845e-01      3.239e-01      6.259e-01
##      stories2       stories3       stories4    drivewayyes  recreationyes
##     9.911e-02      2.305e-01      2.971e-01      1.217e-01      6.903e-02
##   fullbaseyes     gasheatyes      airconyes        garage1        garage2
##     1.094e-01      1.660e-01      1.736e-01      8.335e-02      1.267e-01
##       garage3      preferyes
##     2.467e-02      1.239e-01
```

```
step(model2, direction='backward', criterion='AIC')
```

```
## Start:  AIC=-1666.3
## log(price) ~ lotsize + bedrooms + bathrooms + stories + driveway +
##     recreation + fullbase + gasheat + aircon + garage + prefer
##
##              Df Sum of Sq    RSS     AIC
## <none>                     23.812 -1666.3
## - bedrooms    5    0.4641 24.276 -1665.8
```

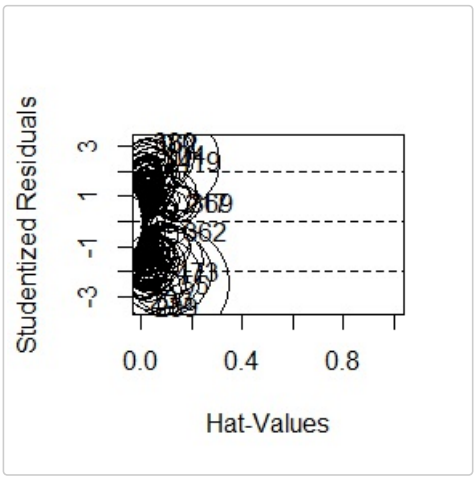
```
## - recreation  1    0.3110 24.123 -1661.2
## - gasheat     1    0.6248 24.437 -1654.2
## - driveway    1    0.9482 24.761 -1647.0
## - fullbase    1    0.9493 24.762 -1647.0
## - garage      3    1.1859 24.998 -1645.8
## - prefer      1    1.1289 24.941 -1643.0
## - stories     3    2.0931 25.905 -1626.3
## - aircon      1    2.7534 26.566 -1608.6
## - bathrooms   3    3.0561 26.868 -1606.4
## - lotsize     1    4.5361 28.348 -1573.1
```

```
##
## Call:
## lm(formula = log(price) ~ lotsize + bedrooms + bathrooms + stories +
##     driveway + recreation + fullbase + gasheat + aircon + garage +
##     prefer, data = data)
##
## Coefficients:
##   (Intercept)        lotsize      bedrooms2      bedrooms3      bedrooms4
##     1.030e+01      4.975e-05      3.961e-02      1.144e-01      1.289e-01
##     bedrooms5      bedrooms6     bathrooms2     bathrooms3     bathrooms4
##     1.271e-01      2.764e-01      1.733e-01      3.148e-01      6.247e-01
##      stories2       stories3       stories4    drivewayyes  recreationyes
##     6.149e-02      1.901e-01      2.629e-01      1.307e-01      6.908e-02
##   fullbaseyes     gasheatyes      airconyes        garage1        garage2
##     1.022e-01      1.676e-01      1.701e-01      8.526e-02      1.179e-01
##       garage3      preferyes
##     7.581e-03      1.177e-01
```

Finding and removing Influential plot

```
# Finding influencial plot for g1 model
influencePlot(g1, id.method=cooks.distance(g1), id.n=4)
```

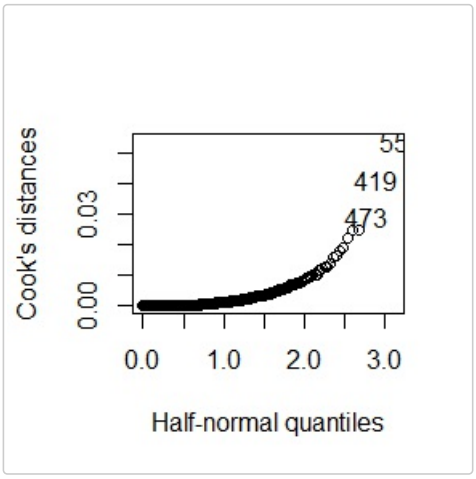
```
## Warning in if (id.method != "identify") {: the condition has length > 1 and
## only the first element will be used
```




```
##        StudRes         Hat       CookD
## 55  -2.5183484 0.128045430 0.054236198
## 104  2.6351784 0.058497945 0.025097916
## 162  3.0531860 0.022850394 0.012624417
## 217  0.7294280 0.145693676 0.005342289
## 233 -3.1204864 0.008808644 0.005007635
## 239 -3.4076079 0.024042151 0.016495551
## 330  3.1624085 0.019977682 0.011791496
## 332        NaN 1.000000000         NaN
## 362 -0.4635797 0.137806583 0.002023533
## 369  0.6985192 0.166096895 0.005722348
## 419  2.3811385 0.111672315 0.041560019
## 473 -2.0200195 0.110193525 0.029552964
```

```
#model without the observation 19 ( fartherst point)

cook <- cooks.distance(g1)
```

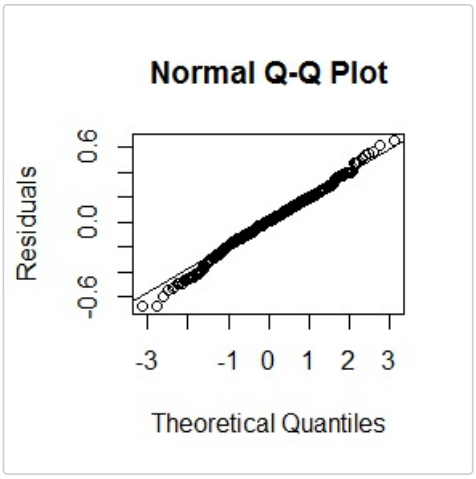
```
halfnorm  (cook,  3, ylab="Cook's distances")
```



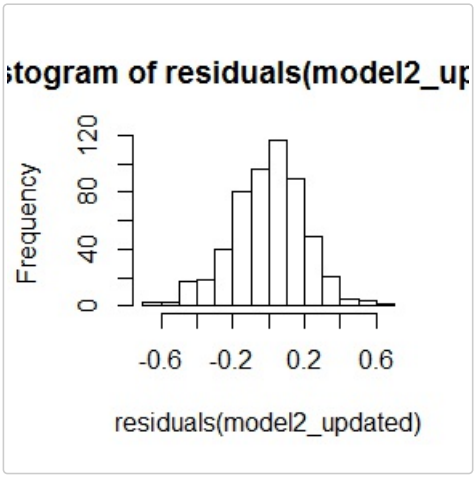

```
#removing the influential points
data_update <- data[-c(55,419),]

model2_updated <- lm(log(price) ~ . , data = data_update)
summary(model2_updated)
```

```
##
## Call:
## lm(formula = log(price) ~ ., data = data_update)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67826 -0.12589  0.00899  0.13151  0.64663
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.031e+01  1.506e-01  68.438  < 2e-16 ***
## lotsize        4.943e-05  4.930e-06  10.028  < 2e-16 ***
## bedrooms2      4.475e-02  1.508e-01   0.297 0.766798
## bedrooms3      1.243e-01  1.513e-01   0.821 0.411896
## bedrooms4      1.295e-01  1.534e-01   0.845 0.398700
## bedrooms5      1.227e-01  1.668e-01   0.735 0.462371
## bedrooms6      2.823e-01  2.131e-01   1.325 0.185851
## bathrooms2     1.719e-01  2.409e-02   7.137 3.21e-12 ***
## bathrooms3     3.761e-01  7.463e-02   5.040 6.43e-07 ***
## bathrooms4     6.818e-01  2.247e-01   3.034 0.002535 **
## stories2       5.976e-02  2.390e-02   2.501 0.012698 *
## stories3       1.899e-01  4.080e-02   4.655 4.12e-06 ***
## stories4       2.643e-01  4.280e-02   6.176 1.33e-09 ***
## drivewayyes    1.245e-01  2.846e-02   4.373 1.48e-05 ***
## recreationyes  8.055e-02  2.636e-02   3.055 0.002362 **
## fullbaseyes    9.500e-02  2.224e-02   4.272 2.30e-05 ***
## gasheatyes     1.647e-01  4.474e-02   3.682 0.000256 ***
## airconyes      1.660e-01  2.168e-02   7.659 9.13e-14 ***
## garage1        8.467e-02  2.349e-02   3.605 0.000342 ***
## garage2        1.172e-01  2.583e-02   4.537 7.08e-06 ***
## garage3       -4.262e-02  6.949e-02  -0.613 0.539906
## preferyes      1.135e-01  2.344e-02   4.840 1.71e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.211 on 522 degrees of freedom
## Multiple R-squared:  0.6866, Adjusted R-squared:  0.674
## F-statistic: 54.45 on 21 and 522 DF,  p-value: < 2.2e-16
```

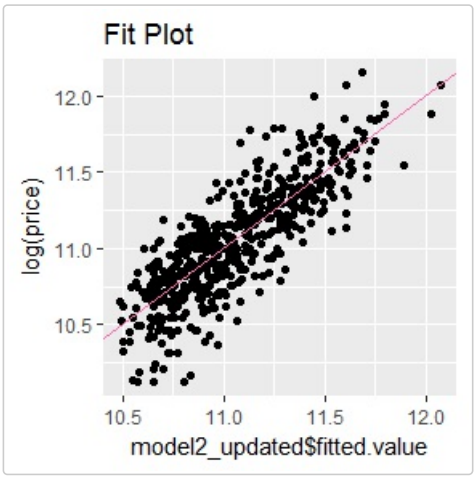
```
qqnorm (residuals(model2_updated), ylab="Residuals")
qqline (residuals(model2_updated))
```




```
hist(residuals(model2_updated))
```

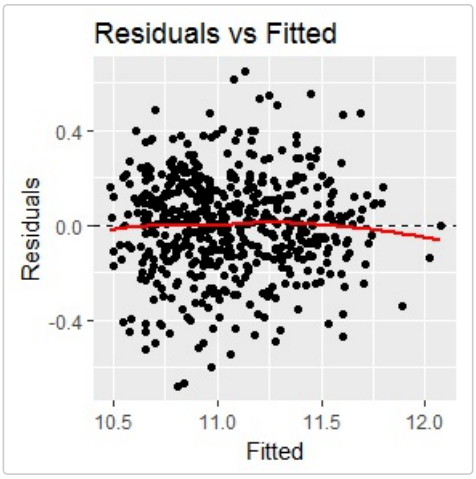



```
#this plot fits the data well compared to the pervious plot
qplot(model2_updated$fitted.value, log(price), data=data_update) +
  geom_abline(intercept = 0, slope = 1, color="hot pink") +
  ggtitle("Fit Plot")
```



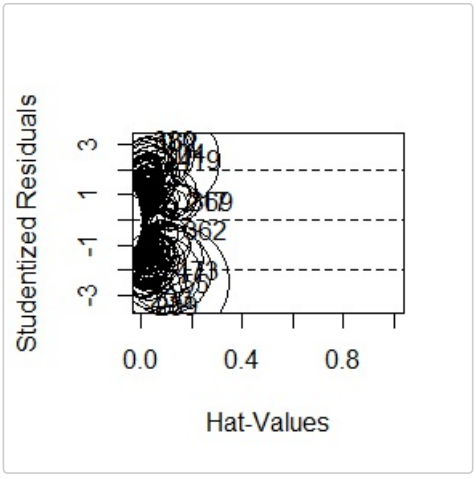

```
#the red line is more linear indicating there is non linear relationships between the variables
mod <- fortify(model2_updated)
p2 <- qplot(.fitted, .resid, data = mod) + geom_hline(yintercept = 0, linetype = "dashed") +
  labs(title = "Residuals vs Fitted", x = "Fitted", y = "Residuals") + geom_smooth(color = "red",
                                                                                   se = F)
p2
```

```
## `geom_smooth()` using method = 'loess'
```



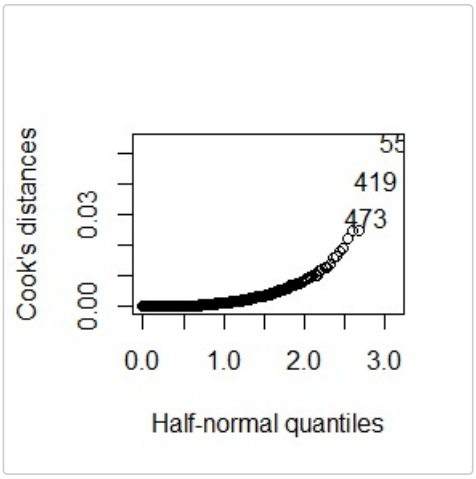

```
# Finding influencial plot for g1 model
influencePlot(g1, id.method=cooks.distance(g1), id.n=4)
```

```
## Warning in if (id.method != "identify") {: the condition has length > 1 and
## only the first element will be used
```




```
##        StudRes         Hat       CookD
## 55  -2.5183484 0.128045430 0.054236198
## 104  2.6351784 0.058497945 0.025097916
## 162  3.0531860 0.022850394 0.012624417
## 217  0.7294280 0.145693676 0.005342289
## 233 -3.1204864 0.008808644 0.005007635
## 239 -3.4076079 0.024042151 0.016495551
## 330  3.1624085 0.019977682 0.011791496
## 332        NaN 1.000000000         NaN
## 362 -0.4635797 0.137806583 0.002023533
## 369  0.6985192 0.166096895 0.005722348
## 419  2.3811385 0.111672315 0.041560019
## 473 -2.0200195 0.110193525 0.029552964
```

```
#model without the observation 19 ( fartherst point)

cook <- cooks.distance(g1)
halfnorm  (cook,  3, ylab="Cook's distances")
```



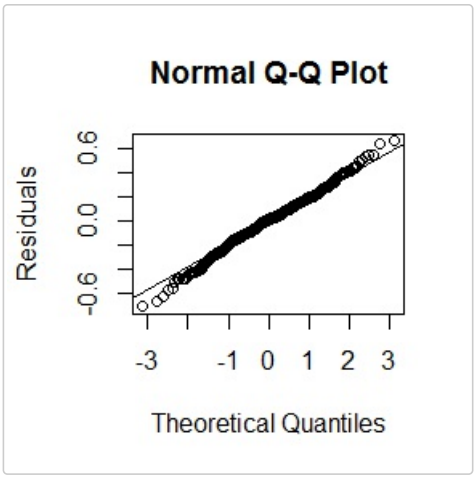

```
#removing the influential points
data_update <- data[-c(55,419),]
```

Developing model on Updated data, i.e. data without influential points

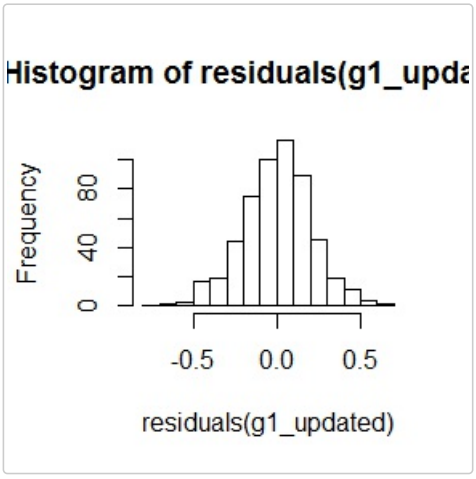
```
g1_updated <- lm(log(price) ~ . - bedrooms,data = data_update)
summary(g1_updated)
```

```
##
## Call:
## lm(formula = log(price) ~ . - bedrooms, data = data_update)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.71074 -0.12383  0.01171  0.13230  0.66416
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.038e+01  3.248e-02 319.560  < 2e-16 ***
## lotsize        5.118e-05  4.882e-06  10.484  < 2e-16 ***
## bathrooms2     1.817e-01  2.347e-02   7.742 5.05e-14 ***
## bathrooms3     3.813e-01  7.348e-02   5.190 3.01e-07 ***
## bathrooms4     6.734e-01  2.251e-01   2.991 0.002909 **
## stories2       9.726e-02  2.045e-02   4.757 2.54e-06 ***
## stories3       2.313e-01  3.870e-02   5.977 4.20e-09 ***
## stories4       2.987e-01  4.140e-02   7.217 1.87e-12 ***
## drivewayyes    1.159e-01  2.813e-02   4.121 4.37e-05 ***
## recreationyes  8.024e-02  2.650e-02   3.028 0.002585 **
## fullbaseyes    1.027e-01  2.209e-02   4.649 4.21e-06 ***
## gasheatyes     1.631e-01  4.480e-02   3.642 0.000297 ***
## airconyes      1.699e-01  2.173e-02   7.819 2.91e-14 ***
## garage1        8.267e-02  2.355e-02   3.510 0.000486 ***
## garage2        1.261e-01  2.577e-02   4.894 1.31e-06 ***
## garage3       -2.232e-02  6.956e-02  -0.321 0.748391
## preferyes      1.207e-01  2.338e-02   5.162 3.46e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2122 on 527 degrees of freedom
## Multiple R-squared:  0.6799, Adjusted R-squared:  0.6702
## F-statistic: 69.96 on 16 and 527 DF,  p-value: < 2.2e-16
```

```
qqnorm (residuals(g1_updated), ylab="Residuals")
qqline (residuals(g1_updated))
```

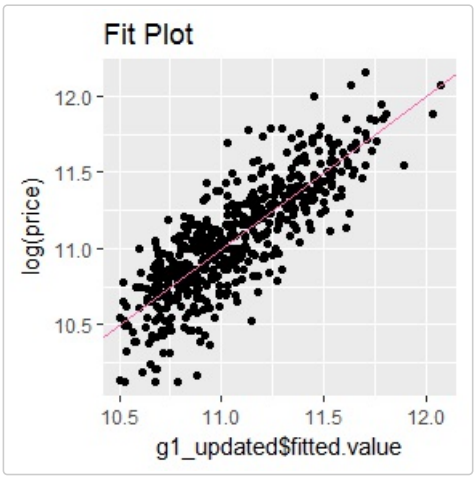



```
hist(residuals(g1_updated))
```

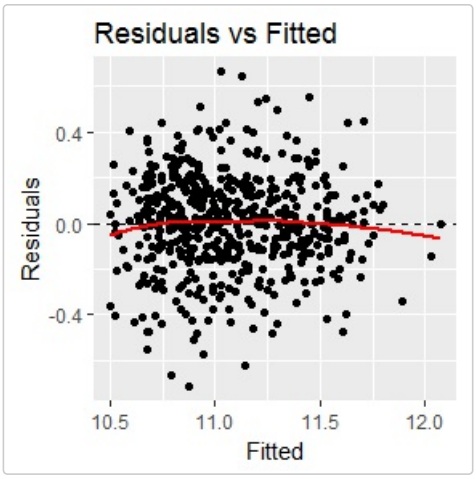


this plot fits the data well compared to the pervious plot

the red line is more linear indicating there is non linear relationships between the variables




```
## `geom_smooth()` using method = 'loess'
```



Collinearity test

```
vif(g1_updated)
```

```
##       lotsize    bathrooms2    bathrooms3    bathrooms4      stories2
##        1.3518        1.2235        1.0613        1.1237        1.2406
##      stories3      stories4   drivewayyes recreationyes   fullbaseyes
##        1.2328        1.4428        1.1491        1.2332        1.3404
##    gasheatyes     airconyes       garage1       garage2       garage3
##        1.0630        1.2335        1.1928        1.2766        1.1582
##     preferyes
##        1.1818
```

```
vif(model2_updated)
```

```
##       lotsize     bedrooms2     bedrooms3     bedrooms4     bedrooms5
##        1.3945       52.1200       69.2340       41.0930        6.1393
##     bedrooms6    bathrooms2    bathrooms3    bathrooms4      stories2
##        2.0328        1.3037        1.1076        1.1327        1.7144
##      stories3      stories4   drivewayyes recreationyes   fullbaseyes
##        1.3861        1.5600        1.1897        1.2346        1.3733
##    gasheatyes     airconyes       garage1       garage2       garage3
##        1.0724        1.2416        1.1997        1.2980        1.1693
##     preferyes
##        1.2018
```

```
coef(model2_updated)
```

```
##   (Intercept)       lotsize     bedrooms2     bedrooms3     bedrooms4
##  1.030608e+01  4.943336e-05  4.474892e-02  1.242777e-01  1.295438e-01
##     bedrooms5     bedrooms6    bathrooms2    bathrooms3    bathrooms4
##  1.227141e-01  2.822780e-01  1.719472e-01  3.761479e-01  6.818326e-01
##      stories2      stories3      stories4   drivewayyes recreationyes
##  5.976303e-02  1.899281e-01  2.643029e-01  1.244523e-01  8.055414e-02
##   fullbaseyes    gasheatyes     airconyes       garage1       garage2
##  9.499672e-02  1.647171e-01  1.660277e-01  8.466889e-02  1.172178e-01
##       garage3     preferyes
## -4.262355e-02  1.134654e-01
```

```
coef(g1_updated)
```

```
##   (Intercept)       lotsize    bathrooms2    bathrooms3    bathrooms4
##  1.037828e+01  5.117801e-05  1.817250e-01  3.813228e-01  6.734381e-01
##      stories2      stories3      stories4   drivewayyes recreationyes
##  9.725749e-02  2.312984e-01  2.987411e-01  1.159372e-01  8.023715e-02
##   fullbaseyes    gasheatyes     airconyes       garage1       garage2
##  1.027214e-01  1.631462e-01  1.699119e-01  8.266775e-02  1.261186e-01
##       garage3     preferyes
## -2.232389e-02  1.206878e-01
```

Robust testing using cross validation

- Cross Validation: perform cross validation for model2 and g1
- In order to perform cross validation, we have created dummy variables for the categorical variables so that k-fold cross validation functions works properly.
- Used [,-1] to remove intercept for each categorical variable else we will get an error in identifying variables in CV
- We can use a different seed for choosing different random folds here we use m=4 folds
- We will repeat the above model comparisons ten times with different seeds

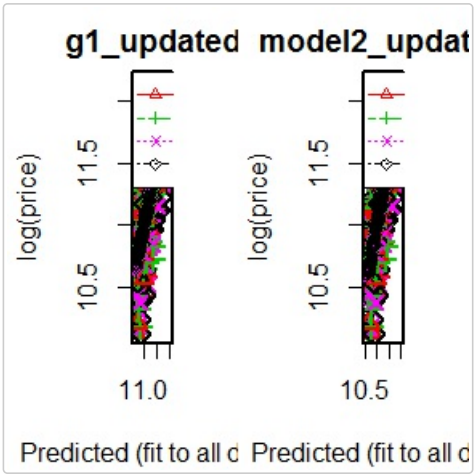
```
dummy_bedrooms <- model.matrix(~bedrooms, data=data_update)[,-1]
dummy_bathrooms <- model.matrix(~bathrooms, data=data_update)[,-1]
dummy_stories <- model.matrix(~stories, data=data_update)[,-1]
dummy_garage <- model.matrix(~garage, data=data_update)[,-1]
data_update<- Filter(is.numeric,data_update)
data_update <- cbind(data_update,dummy_bedrooms,dummy_bathrooms,dummy_stories,dummy_garage)
cvmodel2 <- lm(log(price) ~ . , data = data_update)
cvg1 <- lm(log(price) ~ . - bedrooms2- bedrooms3- bedrooms4- bedrooms5-
bedrooms6,data=data_update,na.action = na.exclude)


# Cross Validation
# We can use a different seed for choosing different random folds
# here we use m=4 folds
# We will repeat the above model comparisons ten times with different seeds

df <- data.frame(mse.cvg1=NULL,
                 mse.cvmodel2=NULL)
for (i in 1:10)
{
  seed <- round(runif(1, min=0, max=100))
  oldpar <- par(mfrow=c(1,2))
  mse.cvg1 <- CVlm(data = data_update,
                   form.lm=cvg1,
                   m=4,
                   seed=seed,
                   printit=F,
                   main = "g1_updated")
  mse.cvmodel2 <- CVlm(data = data_update,
                       form.lm=cvmodel2,
                       m=4,
                       seed=seed,
                       printit=F,
                       main = "model2_updated")
  par(oldpar)

  df.Housemodel <- data.frame (mse.cvg1=attr(mse.cvg1, "ms"),
                               mse.cvmodel2=attr(mse.cvmodel2, "ms"))

  df <- rbind(df,df.Housemodel)

}
```

```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, : prediction from a
rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

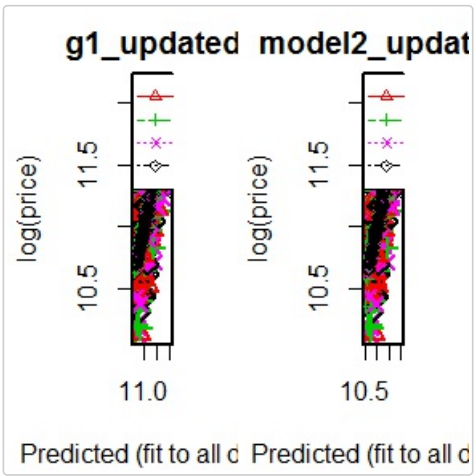
```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```

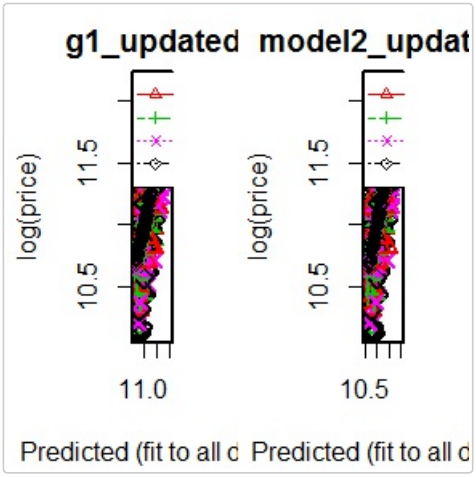



```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

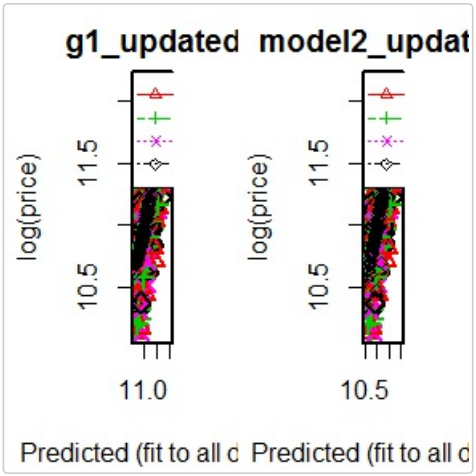
```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading

## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```

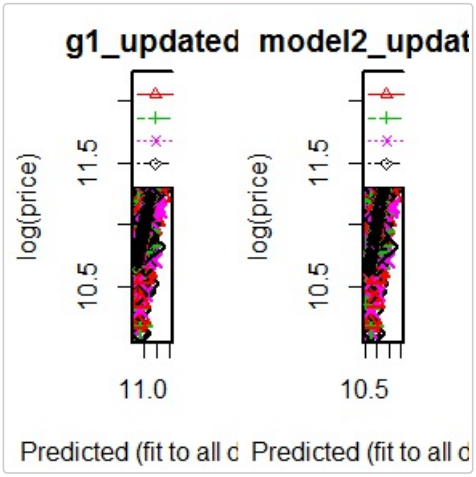



```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
```

```
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

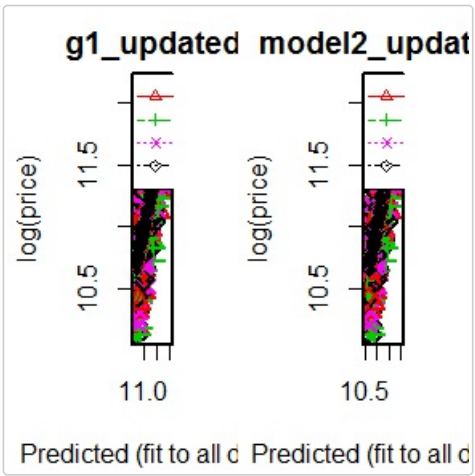
```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```

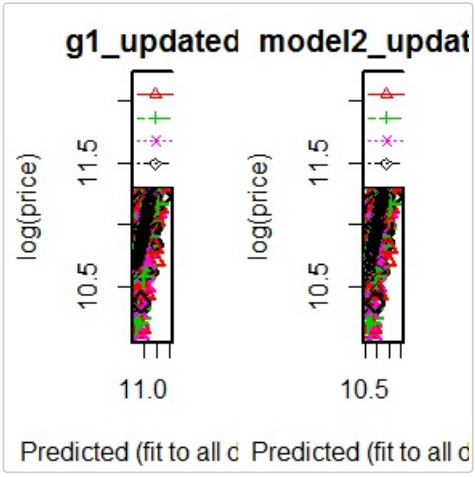



```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

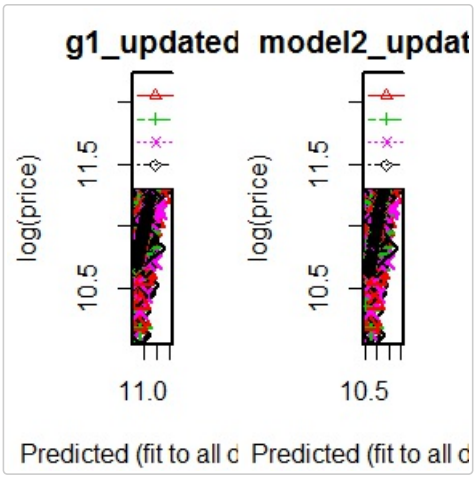
```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

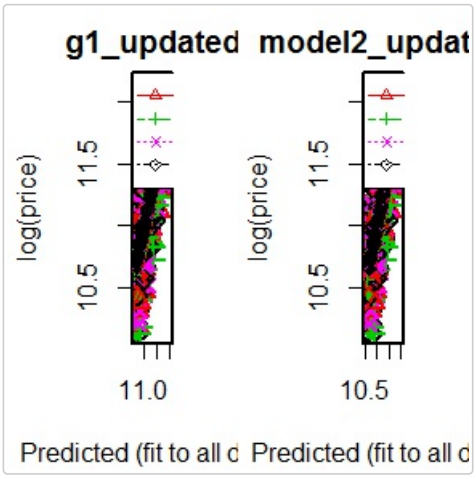
```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

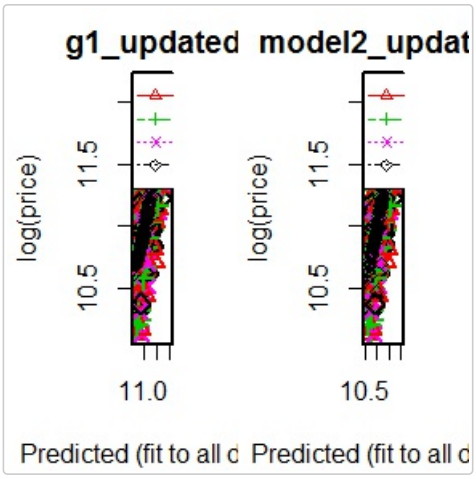
```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate

## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, : prediction from
a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvg1, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading

## Warning in predict.lm(subs.lm, newdata = data[rows.out, ]): prediction from
## a rank-deficient fit may be misleading
```

```
## Warning in CVlm(data = data_update, form.lm = cvmodel2, m = 4, seed = seed, :
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
df
```

```
##      mse.cvg1 mse.cvmodel2
## 1  0.06372113   0.06393612
## 2  0.06474407   0.06350315
## 3  0.06540915   0.06589321
## 4  0.06405570   0.06455871
## 5  0.06370636   0.06432436
## 6  0.06365563   0.06320398
## 7  0.06405570   0.06455871
## 8  0.06370636   0.06432436
## 9  0.06365563   0.06320398
## 10 0.06405570   0.06455871
```

Conclusion:

- From the results we can see that both the models are very close in comparison with mean squared error but due to the collineraity in model2 we go with g1,g1model is best as we get least sum of square error.
- This proves our alternate hypothesis, i.e. at least one variable explains price. In our case except variable bedroom, all other independent variables are able to explain variance in price. This model (g1model) will help to predict future price of property in a particular location.