```r
library(readr)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```r
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v stringr   1.5.1
## v forcats   1.0.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```r
library(boot)
```

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

```r
library(glmnet)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
```

```
##     expand, pack, unpack
##
## Loaded glmnet 4.1-8
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'lattice'
##
## The following object is masked from 'package:boot':
##
##     melanoma
##
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

Dataset & Cleaning

```
# Read in dataset
seattle_housing <- read.csv("house_sales.csv")

# Check for NA's
na_count <- sum(is.na(seattle_housing))
print(paste("Number of missing variables:", na_count))
```

```
## [1] "Number of missing variables: 0"
```

```r
# Create house_age variable
date <- as.numeric(format(Sys.Date(), "%Y"))  # Get current year
seattle_housing$house_age <- date - seattle_housing$yr_built

# Create "renovated" variable
seattle_housing$renovated <- ifelse(seattle_housing$yr_renovated == 0, 0, 1)
```

```r
# Remove unneccesary variables by creating data subset
seattle_clean <- seattle_housing %>%
  select(price, bedrooms, bathrooms, sqft_living, sqft_lot, floors, waterfront, view, condition, grade,

# Data type conversions
seattle_clean$waterfront <- as.factor(seattle_clean$waterfront)

seattle_clean$view <- as.factor(seattle_clean$view)

seattle_clean$condition <- as.factor(seattle_clean$condition)

seattle_clean$grade <- as.factor(seattle_clean$grade)

seattle_clean$renovated <- as.factor(seattle_clean$renovated)
```

```r
# Remove outlier house with 33 bedrooms

seattle_clean <- seattle_clean[-15871, ]
```

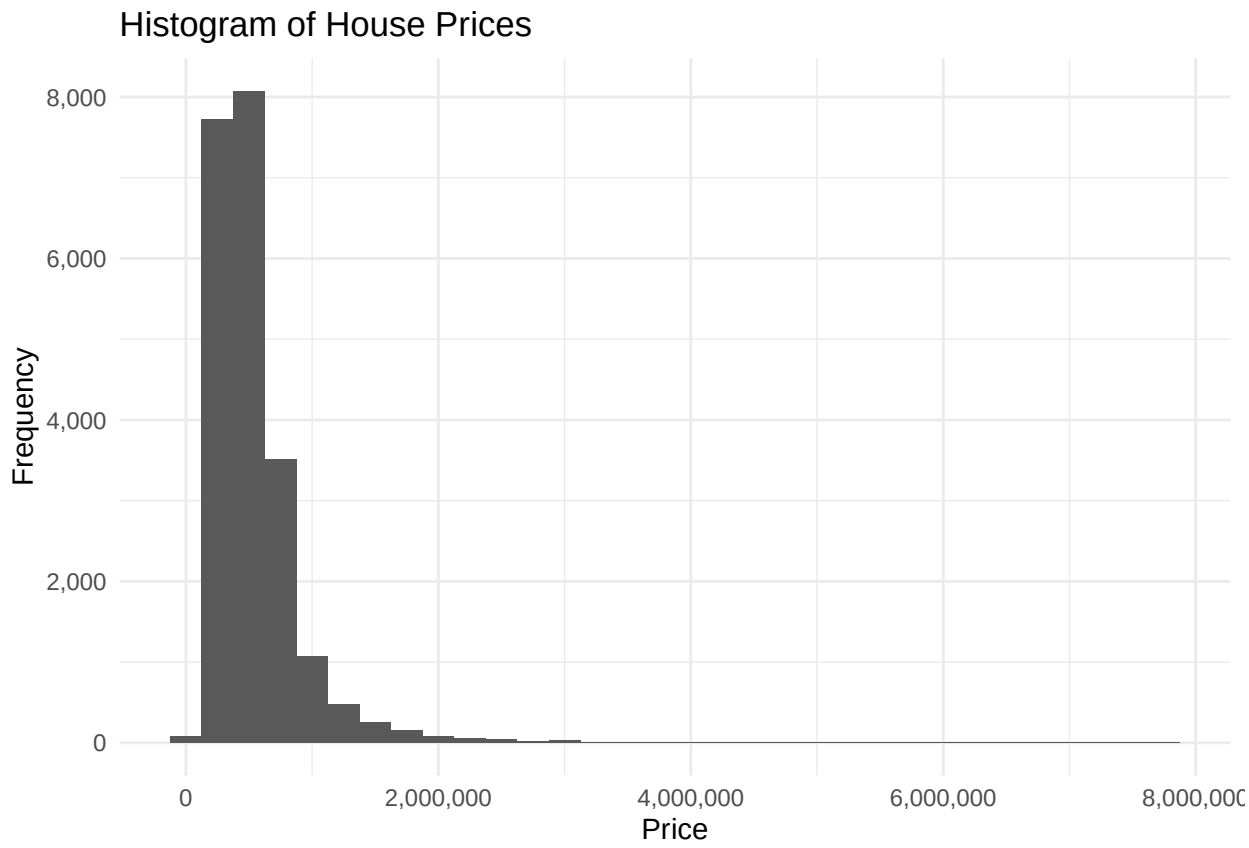
```r
glimpse(seattle_clean)
```

```
## Rows: 21,612
## Columns: 12
## $ price       <int> 221900, 538000, 180000, 604000, 510000, 1225000, 257500, 2~
## $ bedrooms    <int> 3, 3, 2, 4, 3, 4, 3, 3, 3, 3, 3, 2, 3, 3, 5, 4, 3, 4, 2, 3~
## $ bathrooms   <dbl> 1.00, 2.25, 1.00, 3.00, 2.00, 4.50, 2.25, 1.50, 1.00, 2.50~
## $ 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, 6~
## $ floors      <dbl> 1.0, 2.0, 1.0, 1.0, 1.0, 1.0, 2.0, 1.0, 1.0, 2.0, 1.0, 1.0~
## $ waterfront  <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ view        <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0~
## $ condition   <fct> 3, 3, 3, 5, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, 3, 4, 4, 4~
## $ grade       <fct> 7, 7, 6, 7, 8, 11, 7, 7, 7, 7, 8, 7, 7, 7, 7, 9, 7, 7, 7, ~
## $ house_age   <dbl> 70, 74, 92, 60, 38, 24, 30, 62, 65, 22, 60, 83, 98, 48, 12~
## $ renovated   <fct> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

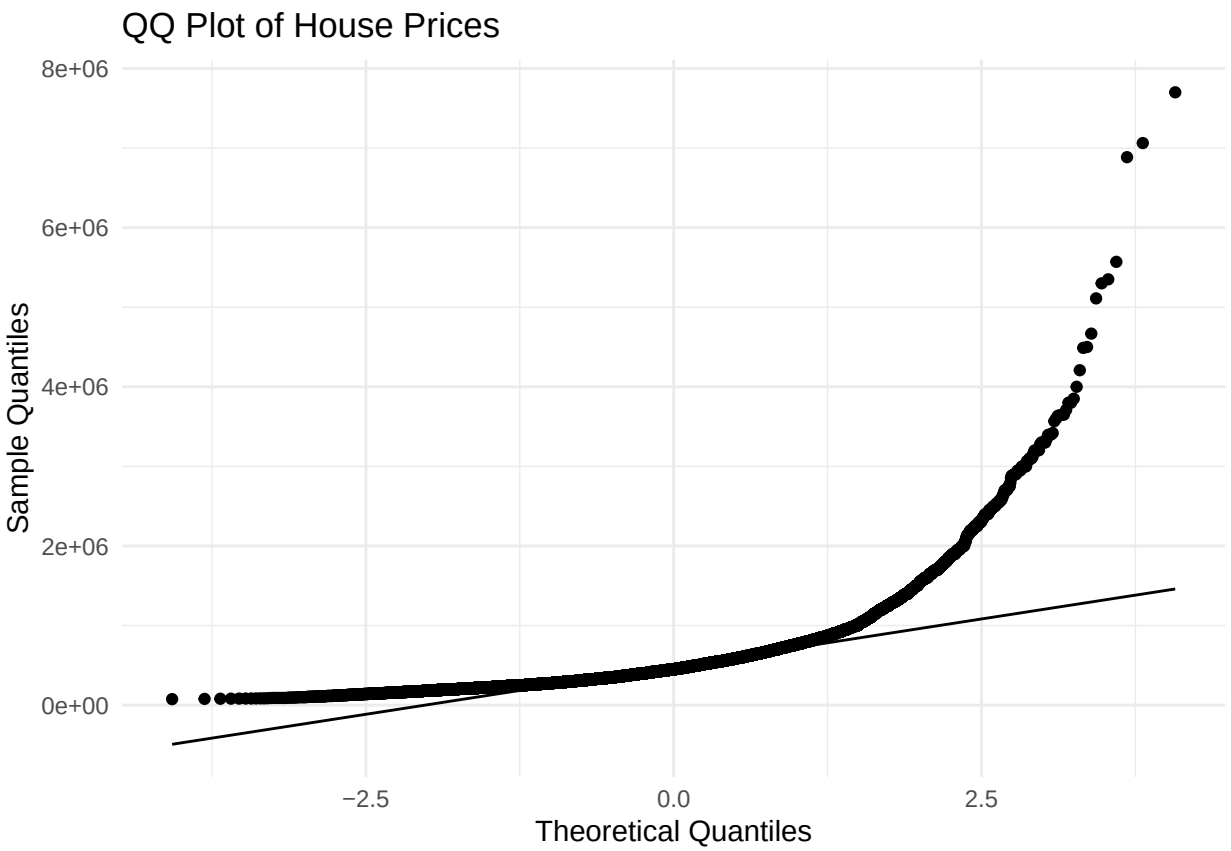
Distribution of House Prices

```r
ggplot(seattle_clean, aes(x = price)) +
  geom_histogram(binwidth = 250000) +
  labs(title = "Histogram of House Prices",
       x = "Price",
       y = "Frequency") +
```

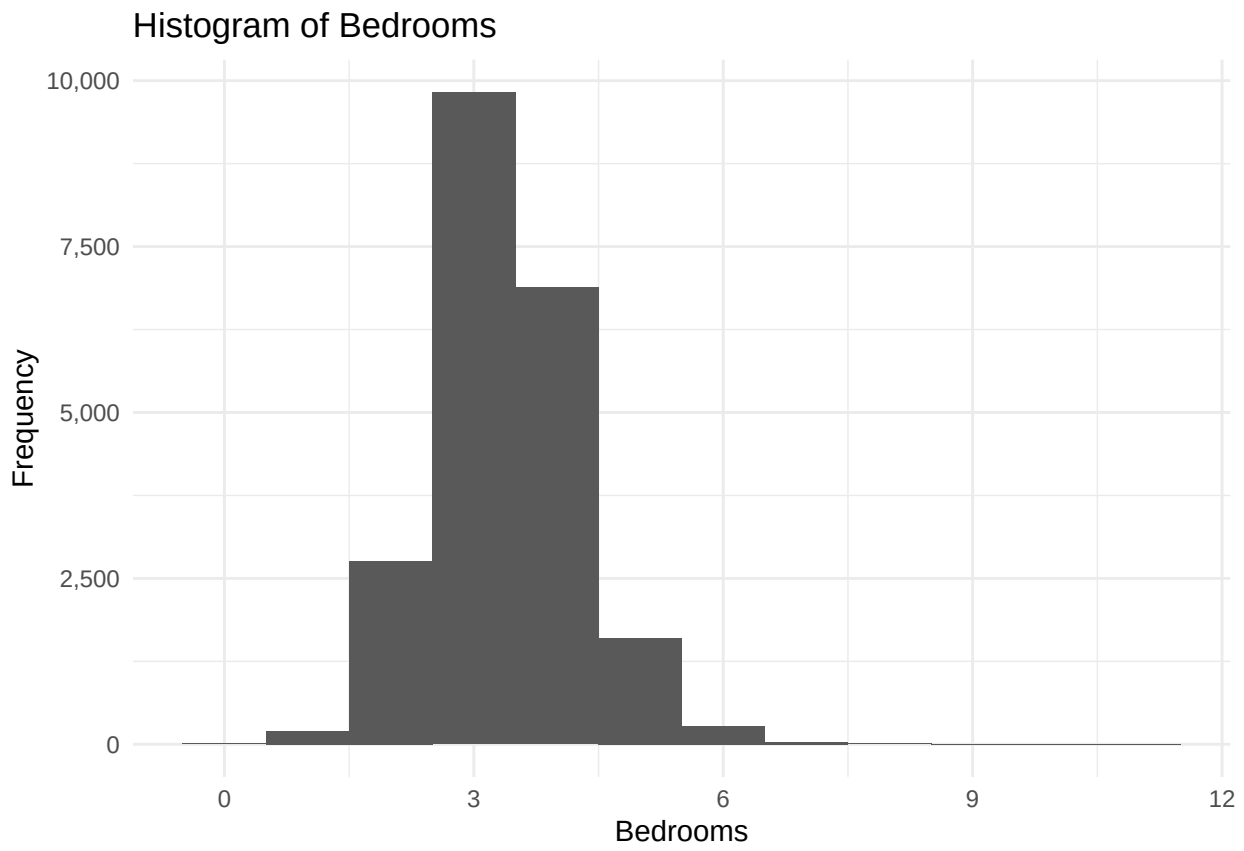
```
  theme_minimal() +
  scale_x_continuous(labels = scales::comma) + scale_y_continuous(labels = scales::comma)
```



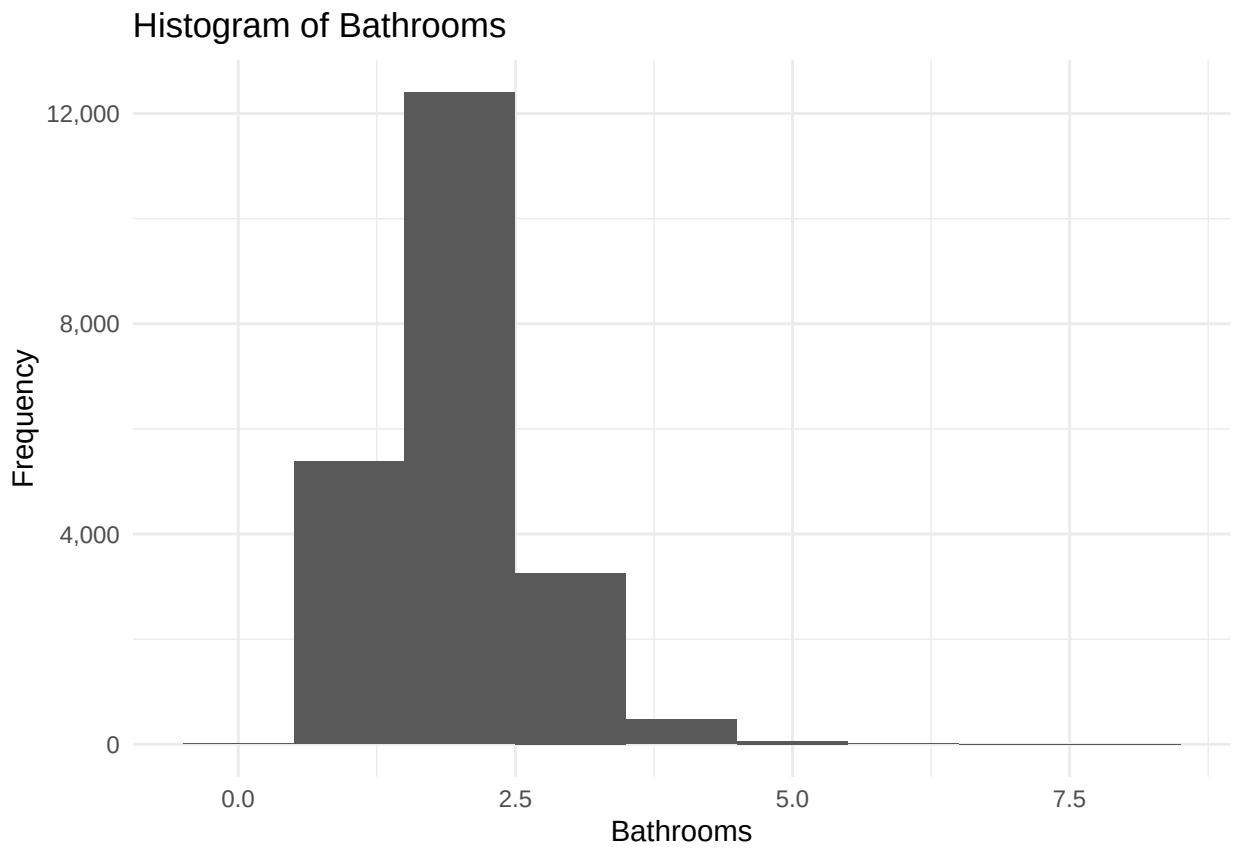

```
ggplot(seattle_clean, aes(sample = price)) +
stat_qq() +
stat_qq_line() +
labs(title = "QQ Plot of House Prices",
x = "Theoretical Quantiles",
y = "Sample Quantiles") +
theme_minimal()
```




```
ggplot(seattle_clean, aes(x = bedrooms)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Histogram of Bedrooms",
       x = "Bedrooms",
       y = "Frequency") +
  theme_minimal() +
  scale_x_continuous(labels = scales::comma) + scale_y_continuous(labels = scales::comma)
```

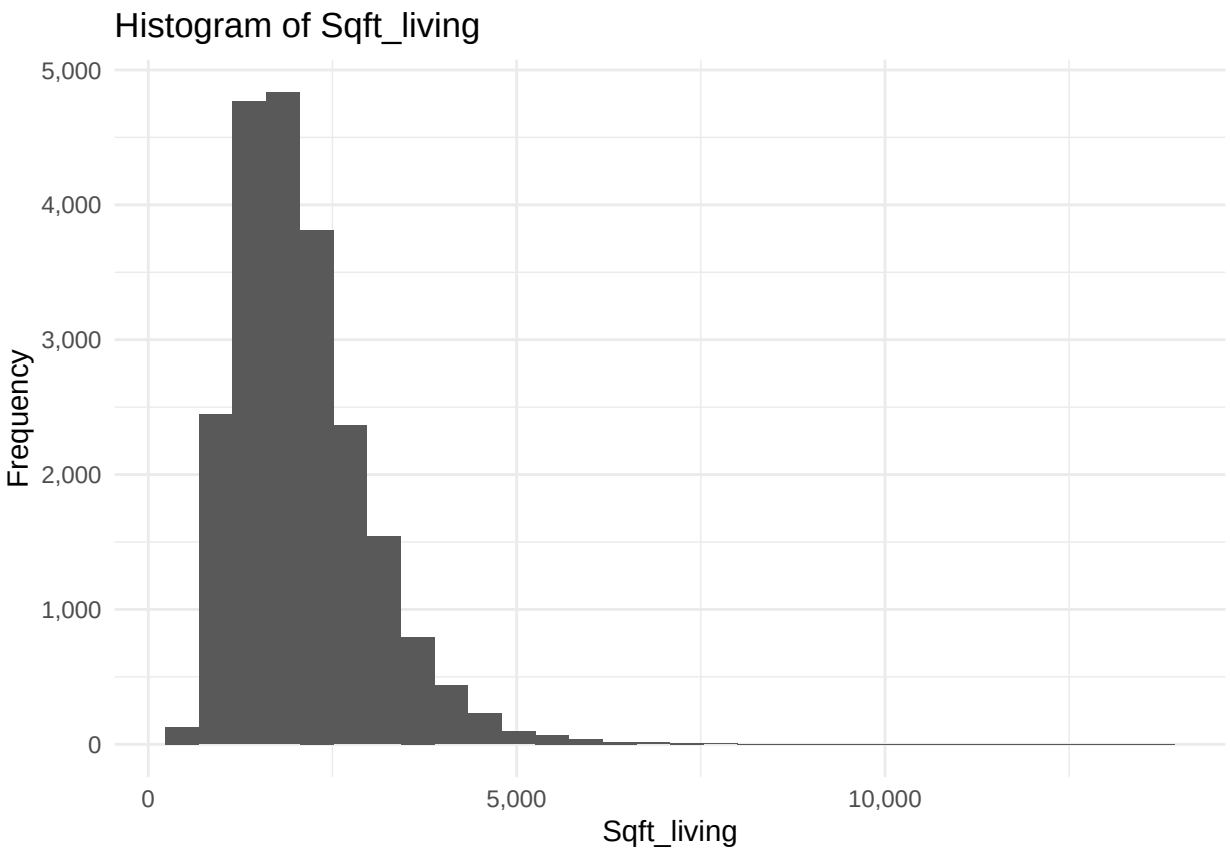



```
ggplot(seattle_clean, aes(x = bathrooms)) +
  geom_histogram(binwidth = 1) +
  labs(title = "Histogram of Bathrooms",
       x = "Bathrooms",
       y = "Frequency") +
  theme_minimal() +
  scale_x_continuous(labels = scales::comma) + scale_y_continuous(labels = scales::comma)
```




```
ggplot(seattle_clean, aes(x = sqft_living)) +
  geom_histogram() +
  labs(title = "Histogram of Sqft_living",
       x = "Sqft_living",
       y = "Frequency") +
  theme_minimal() +
  scale_x_continuous(labels = scales::comma) + scale_y_continuous(labels = scales::comma)
```

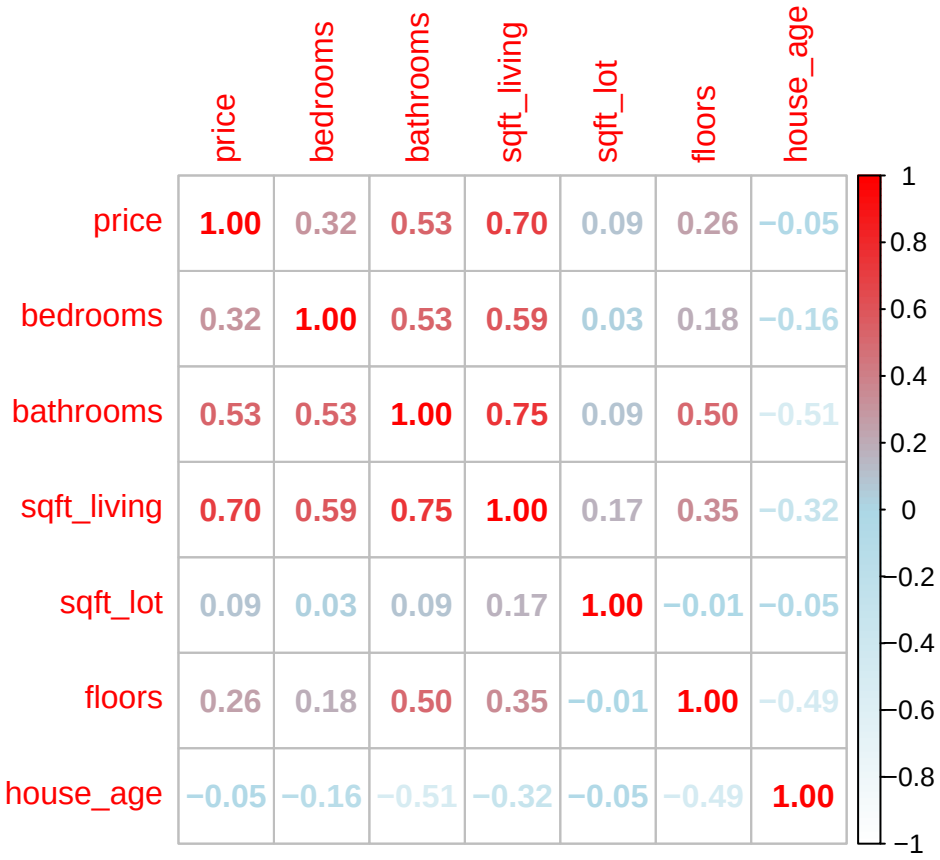
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



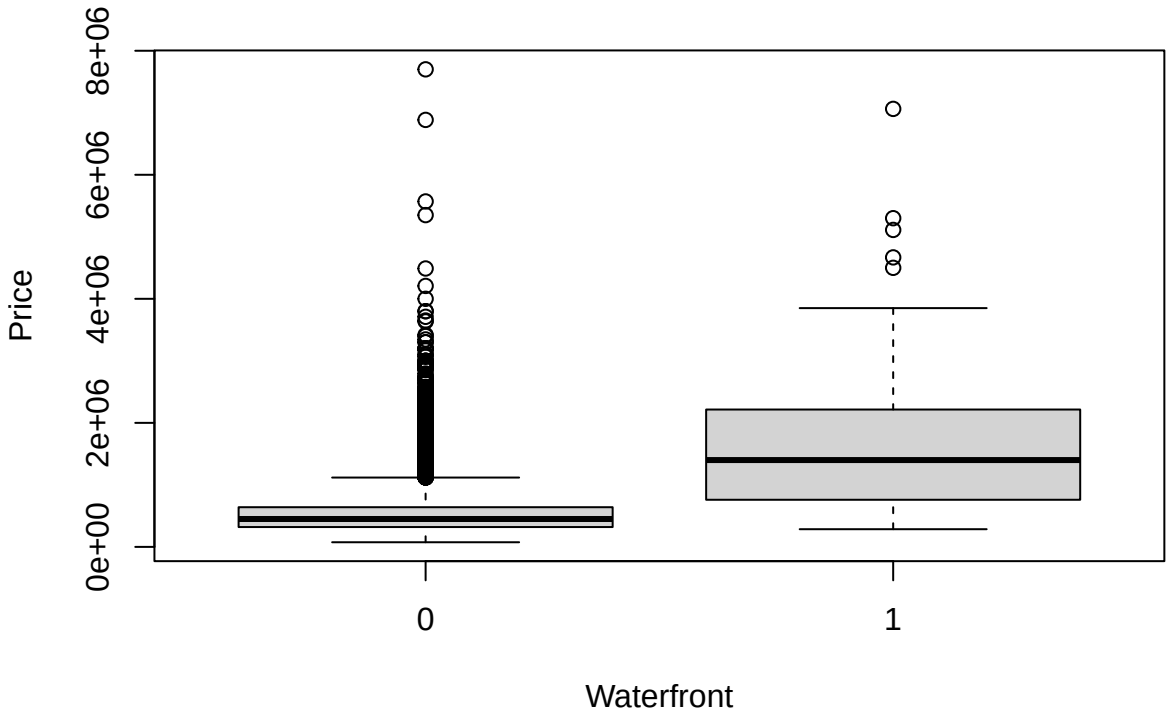

```
ggplot(seattle_clean, aes(sample = sqft_living)) +
stat_qq() +
stat_qq_line() +
labs(title = "QQ Plot of Sqft_living",
x = "Theoretical Quantiles",
y = "Sample Quantiles") +
theme_minimal()
```


```
seattle_quant <- seattle_clean %>%
  select(price, bedrooms, bathrooms, sqft_living, sqft_lot, floors, house_age)
```

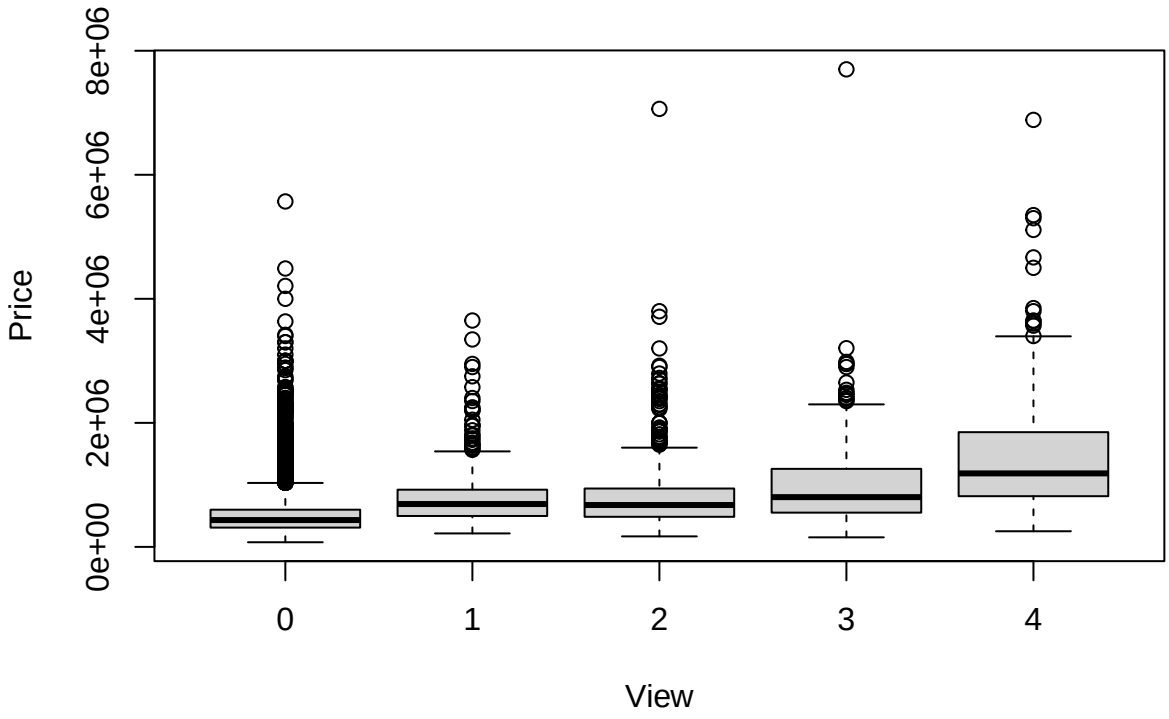
```
seattle.cor <- cor(seattle_quant)

corrplot(seattle.cor, order = "original", method = "number", col = colorRampPalette(c("white","lightblu
```



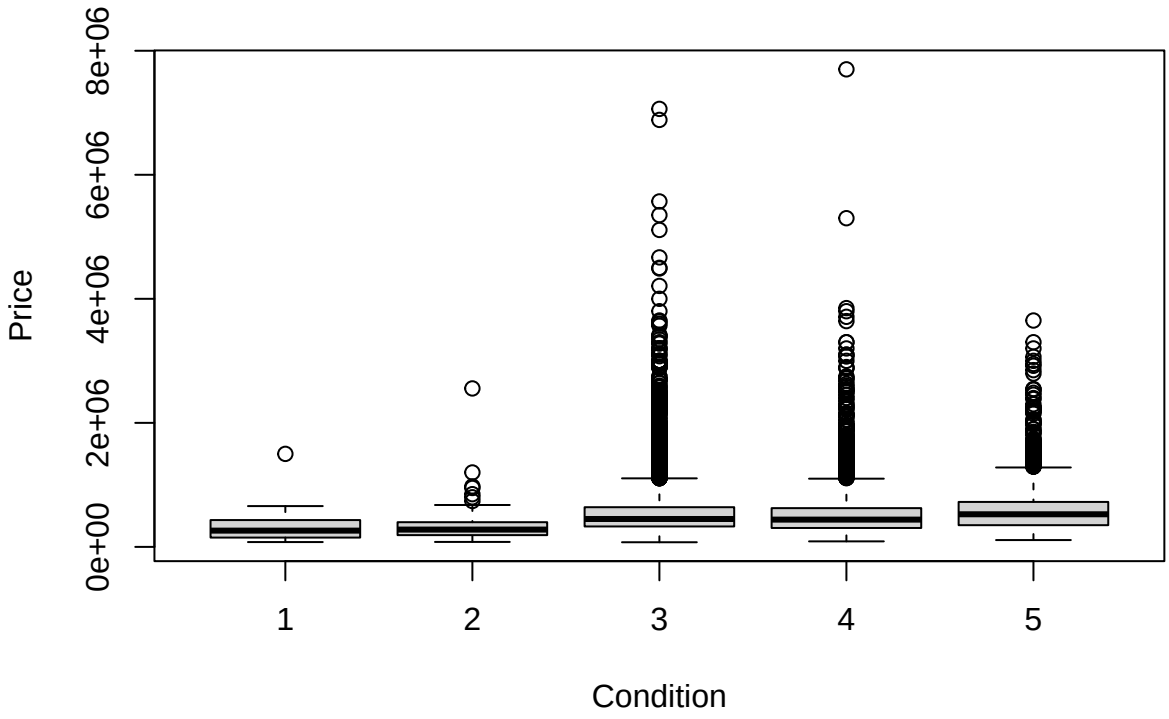

```
boxplot(seattle_clean$price ~ seattle_clean$waterfront, ylab = "Price", xlab = "Waterfront")
```



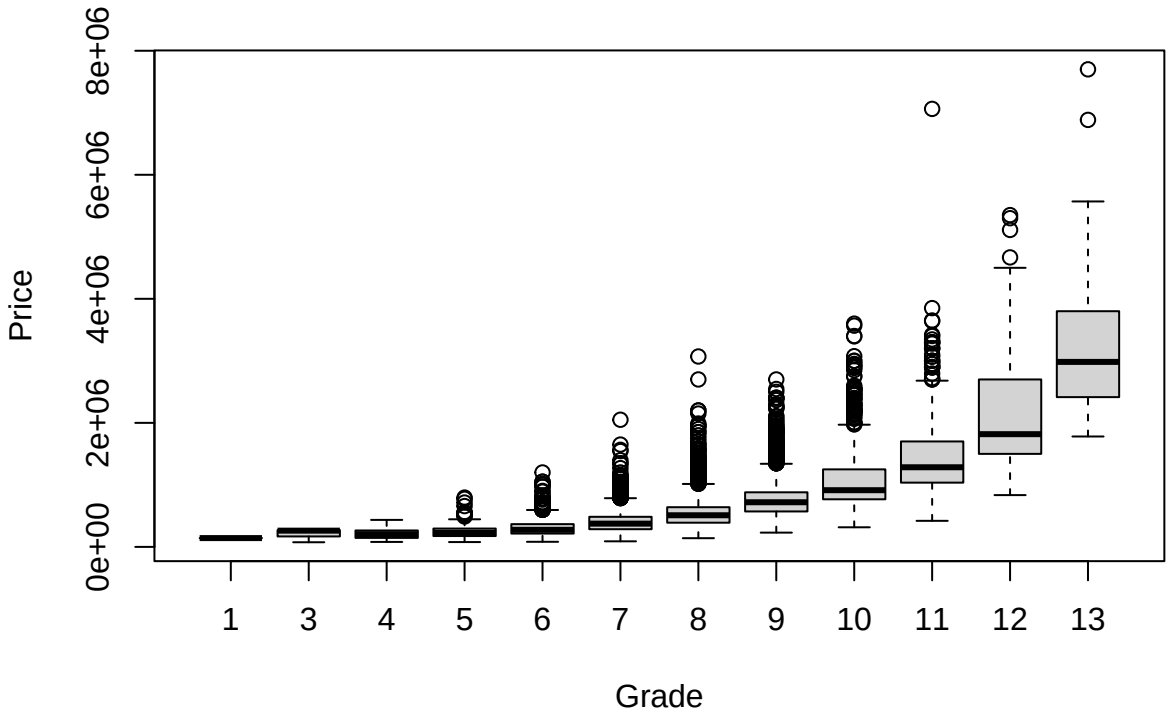

```
boxplot(seattle_clean$price ~ seattle_clean$view, ylab = "Price", xlab = "View")
```




```
boxplot(seattle_clean$price ~ seattle_clean$condition, ylab = "Price", xlab = "Condition")
```




```
boxplot(seattle_clean$price ~ seattle_clean$grade, ylab = "Price", xlab = "Grade")
```



Initial OLS Model

Specification: Predictors Selected w/ Business Knowledge

```
ols.initial <- lm(price ~ ., data = seattle_clean)

summary(ols.initial)
```

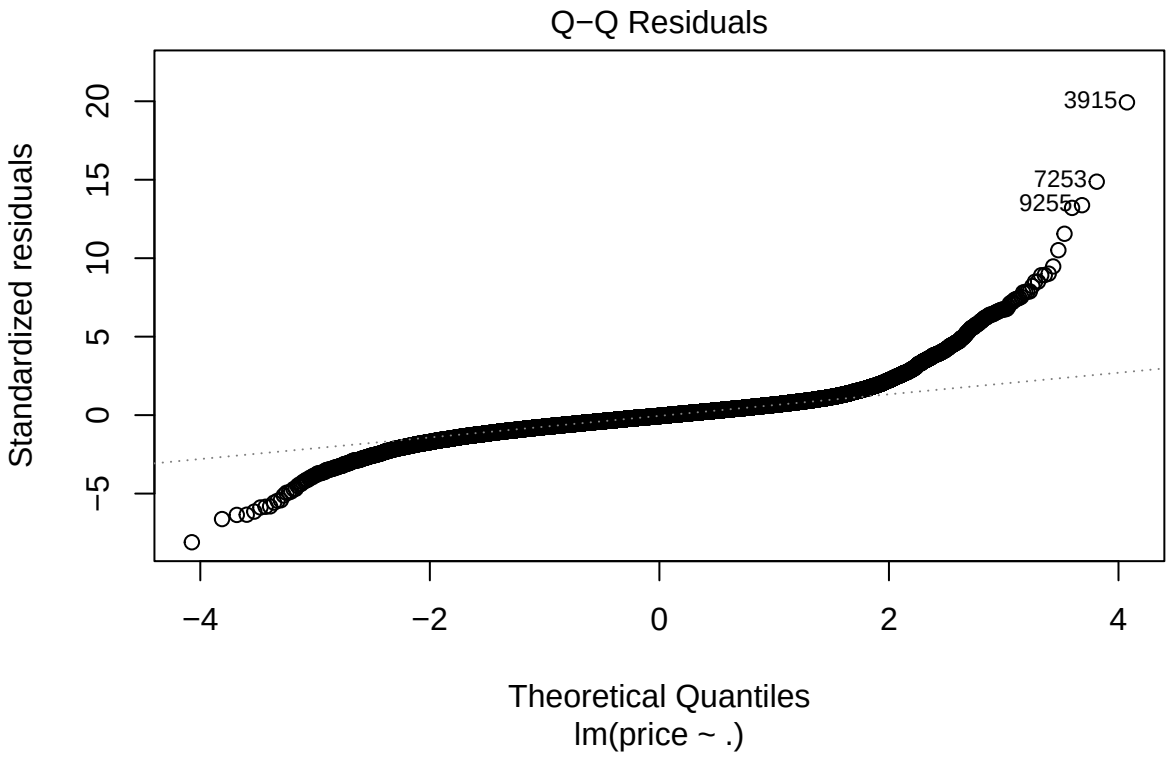
```
##
## Call:
## lm(formula = price ~ ., data = seattle_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1666192  -106071   -11105    86038  4092466
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.228e+05  2.074e+05  -0.592  0.55373
## bedrooms    -2.673e+04  2.071e+03 -12.907  < 2e-16 ***
## bathrooms    5.115e+04  3.333e+03  15.345  < 2e-16 ***
## sqft_living  1.375e+02  3.289e+00  41.802  < 2e-16 ***
## sqft_lot    -2.662e-01  3.500e-02  -7.607 2.92e-14 ***
## floors       3.174e+04  3.332e+03   9.524  < 2e-16 ***
```

```
## waterfront1  4.888e+05  2.030e+04  24.076  < 2e-16 ***
## view1        1.228e+05  1.157e+04  10.612  < 2e-16 ***
## view2        5.669e+04  7.003e+03   8.095 6.02e-16 ***
## view3        1.097e+05  9.572e+03  11.461  < 2e-16 ***
## view4        2.595e+05  1.480e+04  17.534  < 2e-16 ***
## condition2   1.395e+04  4.175e+04   0.334  0.73832
## condition3   3.902e+04  3.887e+04   1.004  0.31546
## condition4   5.761e+04  3.887e+04   1.482  0.13834
## condition5   9.794e+04  3.909e+04   2.506  0.01223 *
## grade3      -6.987e+04  2.426e+05  -0.288  0.77330
## grade4      -1.138e+05  2.142e+05  -0.531  0.59518
## grade5      -1.390e+05  2.111e+05  -0.658  0.51046
## grade6      -8.761e+04  2.110e+05  -0.415  0.67796
## grade7      -5.454e+03  2.110e+05  -0.026  0.97938
## grade8       8.312e+04  2.110e+05   0.394  0.69369
## grade9       2.249e+05  2.111e+05   1.066  0.28666
## grade10      4.063e+05  2.112e+05   1.924  0.05441 .
## grade11      6.720e+05  2.115e+05   3.178  0.00148 **
## grade12      1.128e+06  2.125e+05   5.309 1.11e-07 ***
## grade13      2.323e+06  2.194e+05  10.589  < 2e-16 ***
## house_age    3.206e+03  6.910e+01  46.400  < 2e-16 ***
## renovated1   3.339e+04  7.531e+03   4.435 9.27e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 207300 on 21584 degrees of freedom
## Multiple R-squared:  0.6815, Adjusted R-squared:  0.6811
## F-statistic:  1711 on 27 and 21584 DF,  p-value: < 2.2e-16
```

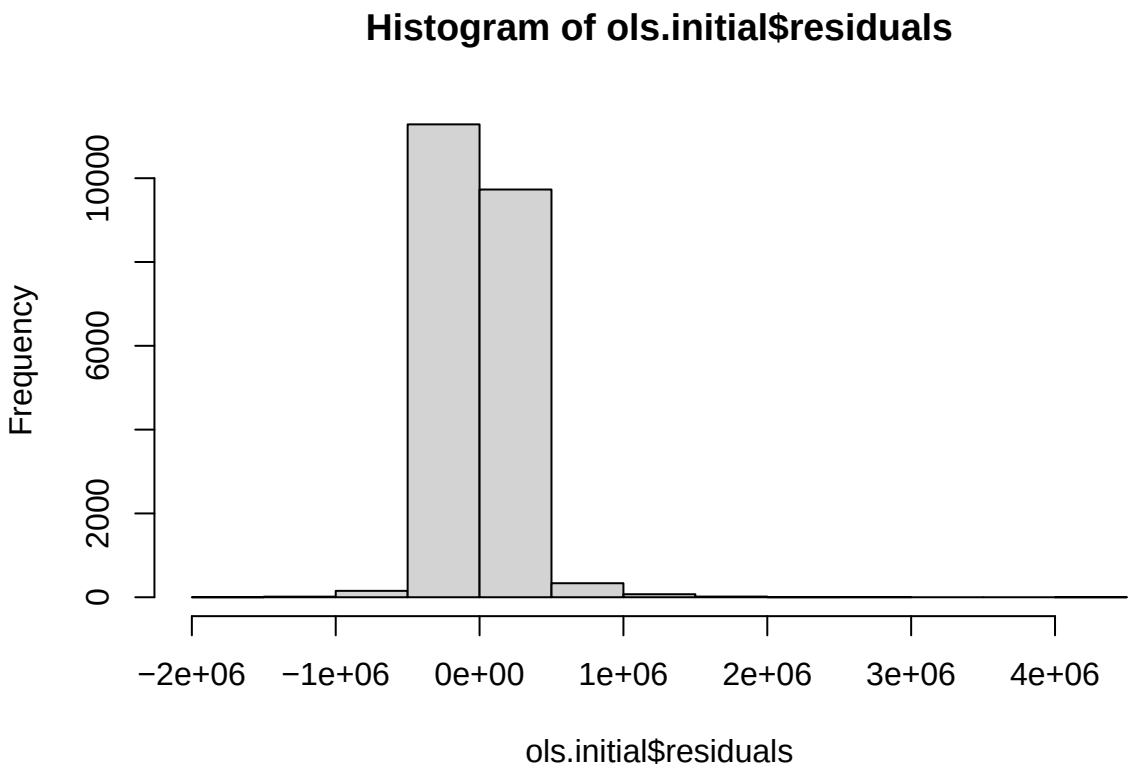
Assumption 2: Errors are normally distributed (EN)

```
plot(ols.initial, which = 2)
```

```
## Warning: not plotting observations with leverage one:
##   19452
```




```r
hist(ols.initial$residuals)
```



Assumption 3: predictors are independent (XI)

```
library(olsrr)
```

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

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

```
ci <- ols_eigen_cindex(ols.initial)

ci[, 1:2]
```

```
##      Eigenvalue Condition Index
## 1  7.7684444991        1.000000
## 2  1.6653316875        2.159816
## 3  1.4040076701        2.352244
## 4  1.1306270185        2.621243
## 5  1.0699063567        2.694599
## 6  1.0347657417        2.739971
```

```
## 7  1.0284540926        2.748366
## 8  1.0167496598        2.764140
## 9  1.0114596616        2.771359
## 10 1.0016743974        2.784862
## 11 1.0006039015        2.786352
## 12 0.9977247908        2.790369
## 13 0.9903920430        2.800680
## 14 0.9790429481        2.816866
## 15 0.9612395560        2.842832
## 16 0.9480912902        2.862477
## 17 0.9382624979        2.877431
## 18 0.8669965664        2.993356
## 19 0.8040422852        3.108333
## 20 0.6807494782        3.378107
## 21 0.3909375175        4.457726
## 22 0.1394318809        7.464247
## 23 0.0798997586        9.860395
## 24 0.0377044023       14.353936
## 25 0.0295603184       16.211100
## 26 0.0229958194       18.379877
## 27 0.0008812382       93.890223
## 28 0.0000229226      582.150263
```

```
ci[nrow(ci), 1:2]
```

```
##     Eigenvalue Condition Index
## 28 2.29226e-05        582.1503
```

```
ols_vif_tol(ols.initial)
```

```
##      Variables    Tolerance         VIF
## 1     bedrooms 0.5625119493    1.777740
## 2    bathrooms 0.3017486401    3.314017
## 3  sqft_living 0.2179667347    4.587856
## 4     sqft_lot 0.9462875535    1.056761
## 5       floors 0.6141699651    1.628214
## 6  waterfront1 0.6444514769    1.551707
## 7        view1 0.9819188812    1.018414
## 8        view2 0.9523622856    1.050021
## 9        view3 0.9419484240    1.061629
## 10       view4 0.6240908542    1.602331
## 11  condition2 0.1444849574    6.921136
## 12  condition3 0.0057810797  172.978068
## 13  condition4 0.0067945264  147.177292
## 14  condition5 0.0179607075   55.677094
## 15      grade3 0.2435104177    4.106600
## 16      grade4 0.0323403339   30.921140
## 17      grade5 0.0040289821  248.201647
## 18      grade6 0.0005230450 1911.881240
## 19      grade7 0.0001839145 5437.308734
## 20      grade8 0.0002211164 4522.503989
## 21      grade9 0.0004195913 2383.271388
## 22     grade10 0.0008967707 1115.112285
```

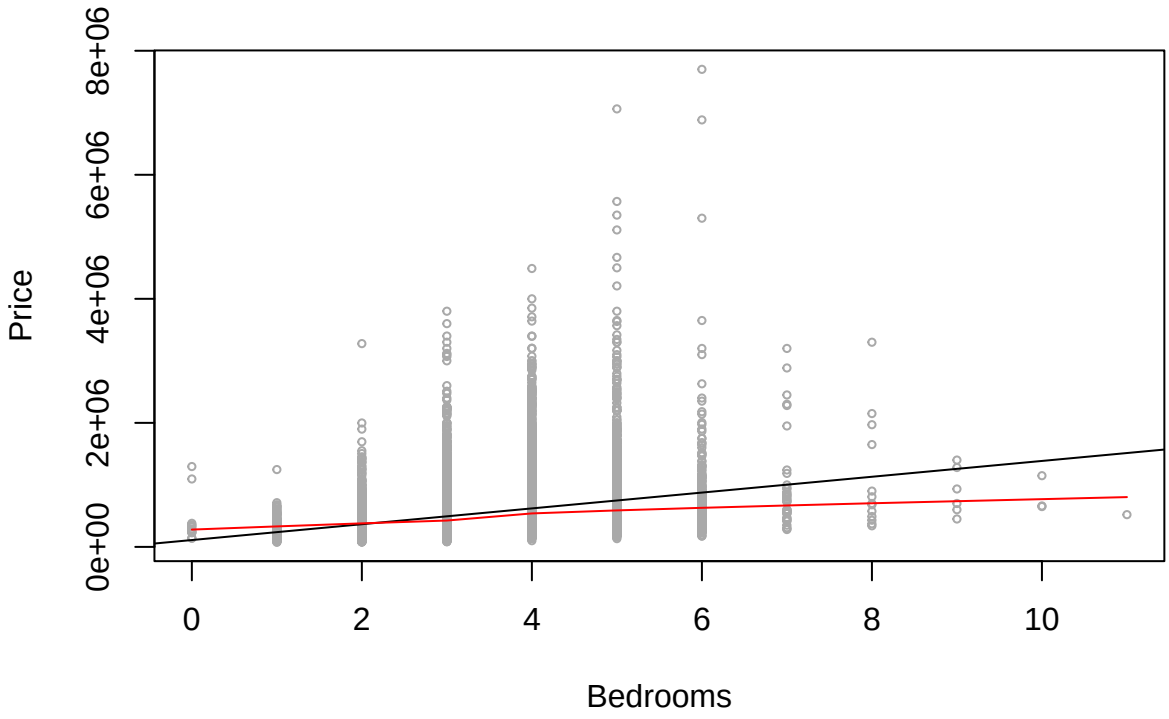
```
## 23     grade11 0.0024544849  407.417464
## 24     grade12 0.0106223951   94.140727
## 25     grade13 0.0687244184   14.550869
## 26   house_age 0.4828066594    2.071222
## 27  renovated1 0.8658139635    1.154983
```

```
vif(ols.initial)
```

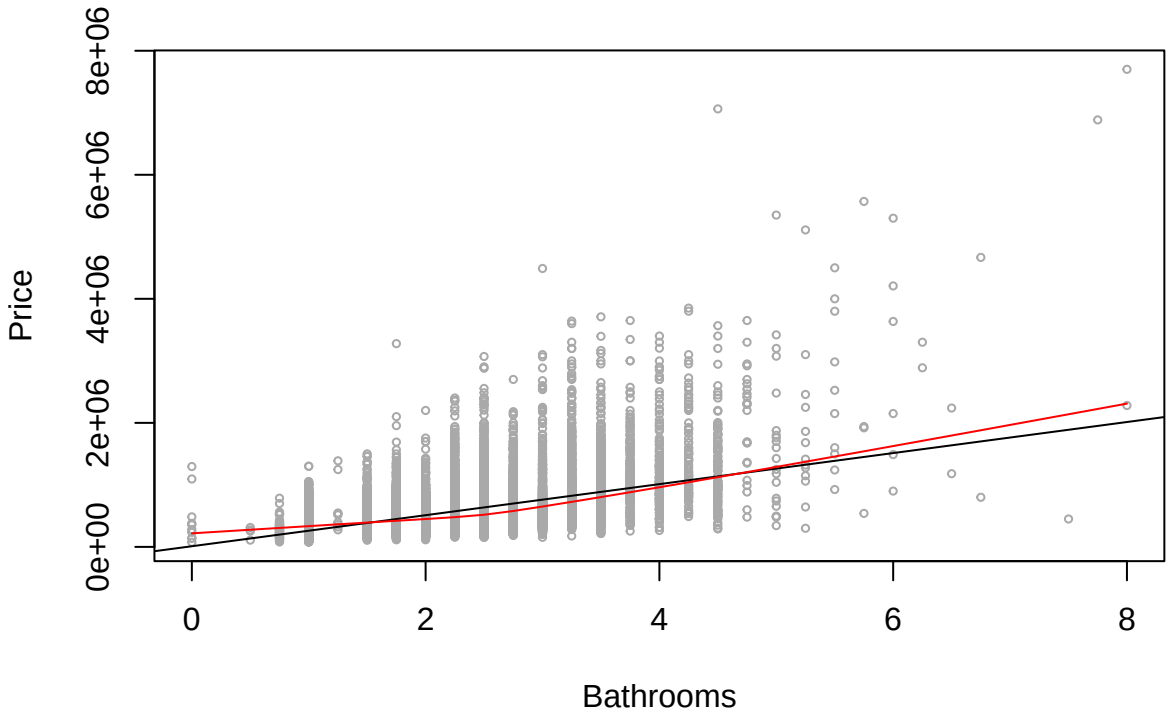
```
##                 GVIF Df GVIF^(1/(2*Df))
## bedrooms    1.777740  1        1.333319
## bathrooms   3.314017  1        1.820444
## sqft_living 4.587856  1        2.141928
## sqft_lot    1.056761  1        1.027989
## floors      1.628214  1        1.276015
## waterfront  1.551707  1        1.245675
## view        1.765777  4        1.073660
## condition   1.389062  4        1.041934
## grade       3.839419 11        1.063059
## house_age   2.071222  1        1.439174
## renovated   1.154983  1        1.074701
```

Assumption 4: the outcome variable is linearly related to the predictors (LI)

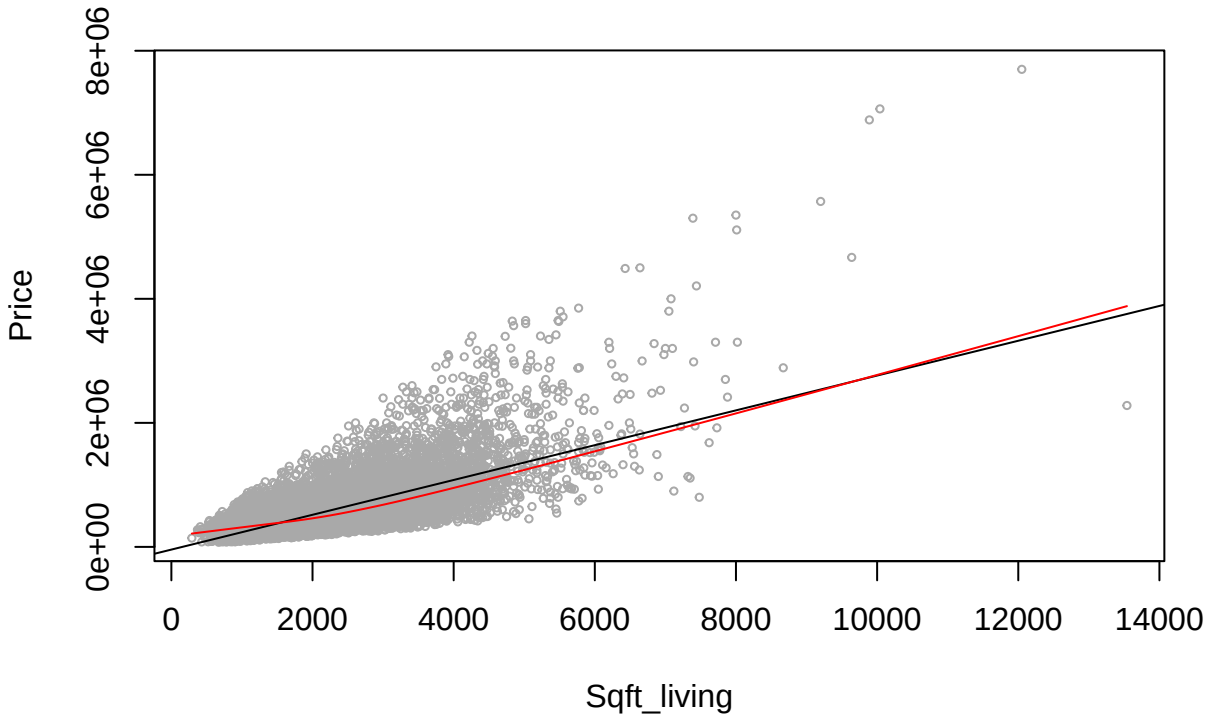
```
bed.fit <- lm(price ~ bedrooms, data = seattle_clean)

plot(seattle_clean$bedrooms, seattle_clean$price,
     xlab = "Bedrooms",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(bed.fit)
lines(lowess(seattle_clean$bedrooms, seattle_clean$price),
      col = "red")
```



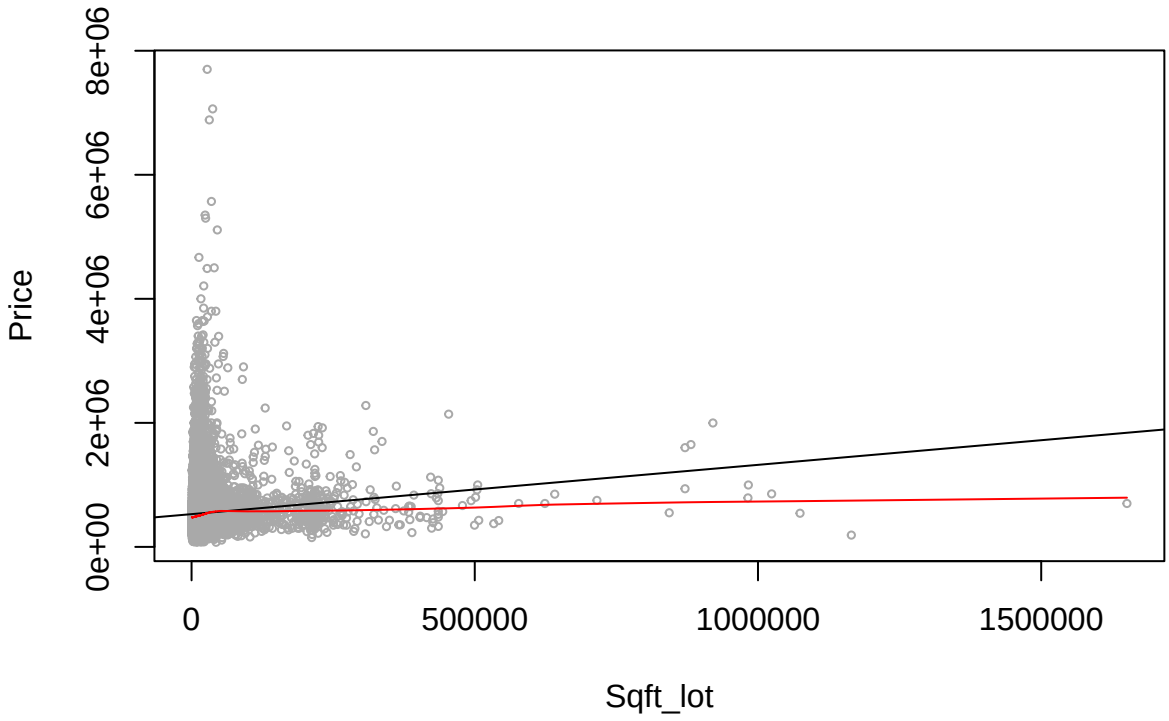

```
bath.fit <- lm(price ~ bathrooms, data = seattle_clean)

plot(seattle_clean$bathrooms, seattle_clean$price,
     xlab = "Bathrooms",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(bath.fit)
lines(lowess(seattle_clean$bathrooms, seattle_clean$price),
      col = "red")
```



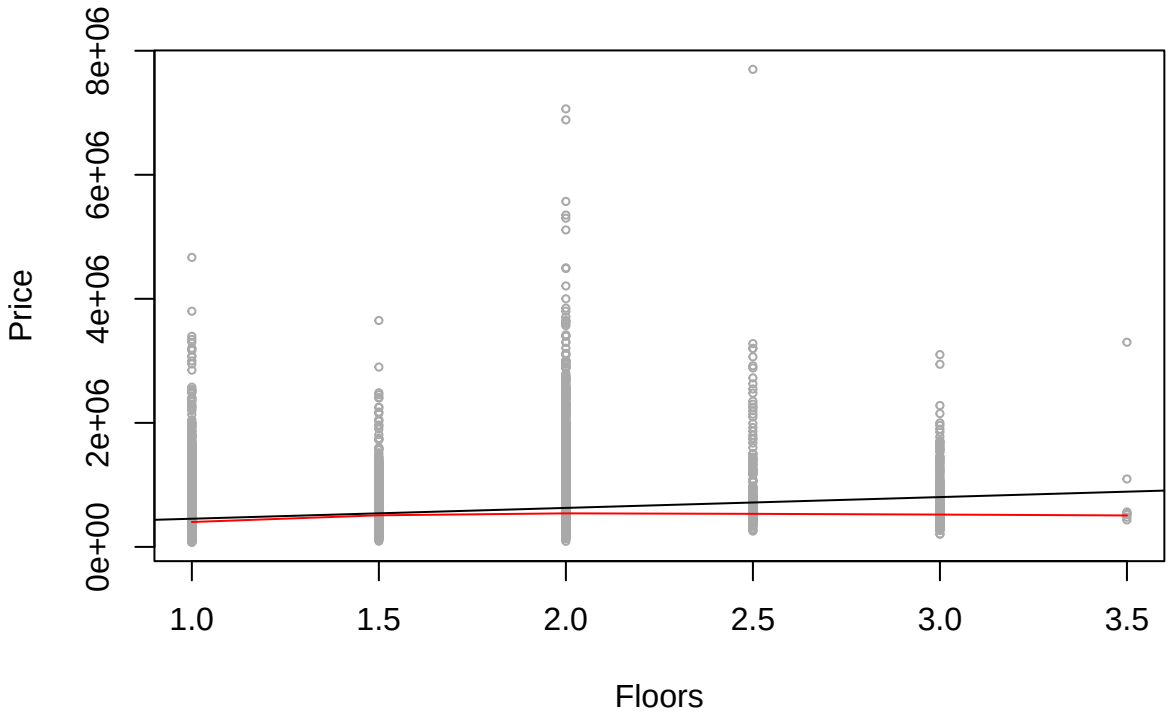

```
living.fit <- lm(price ~ sqft_living, data = seattle_clean)

plot(seattle_clean$sqft_living, seattle_clean$price,
     xlab = "Sqft_living",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(living.fit)
lines(lowess(seattle_clean$sqft_living, seattle_clean$price),
      col = "red")
```



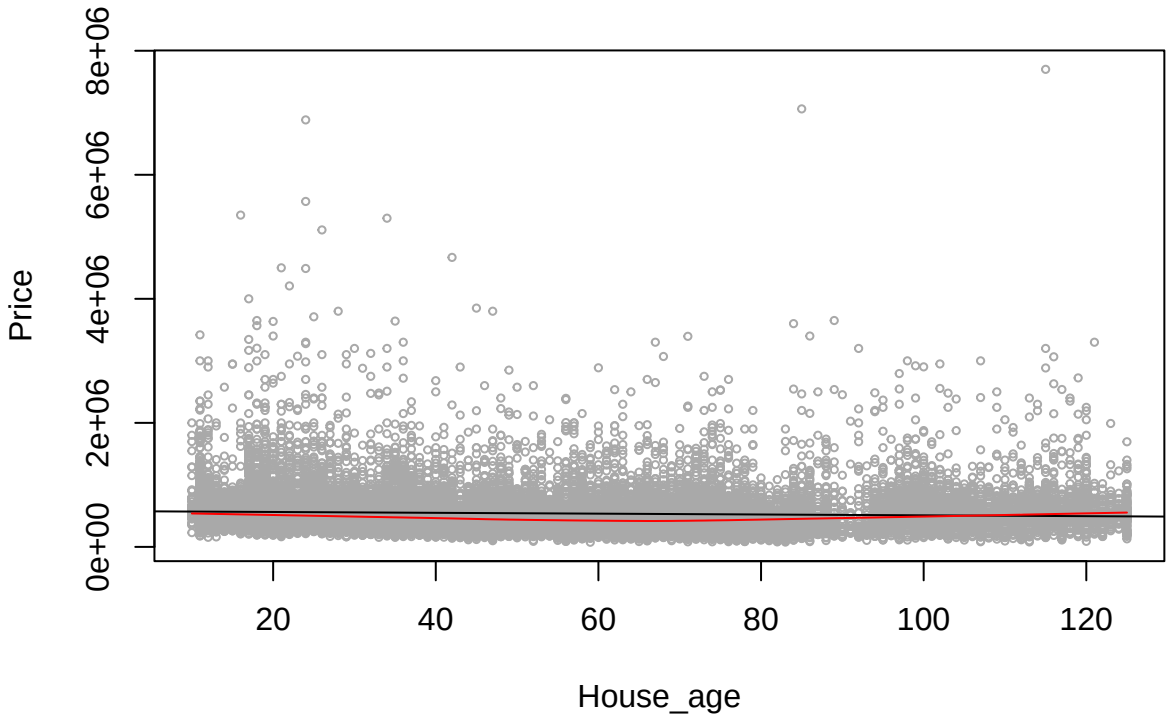

```
lot.fit <- lm(price ~ sqft_lot, data = seattle_clean)

plot(seattle_clean$sqft_lot, seattle_clean$price,
     xlab = "Sqft_lot",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(lot.fit)
lines(lowess(seattle_clean$sqft_lot, seattle_clean$price),
      col = "red")
```




```
floors.fit <- lm(price ~ floors, data = seattle_clean)

plot(seattle_clean$floors, seattle_clean$price,
     xlab = "Floors",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(floors.fit)
lines(lowess(seattle_clean$floors, seattle_clean$price),
      col = "red")
```

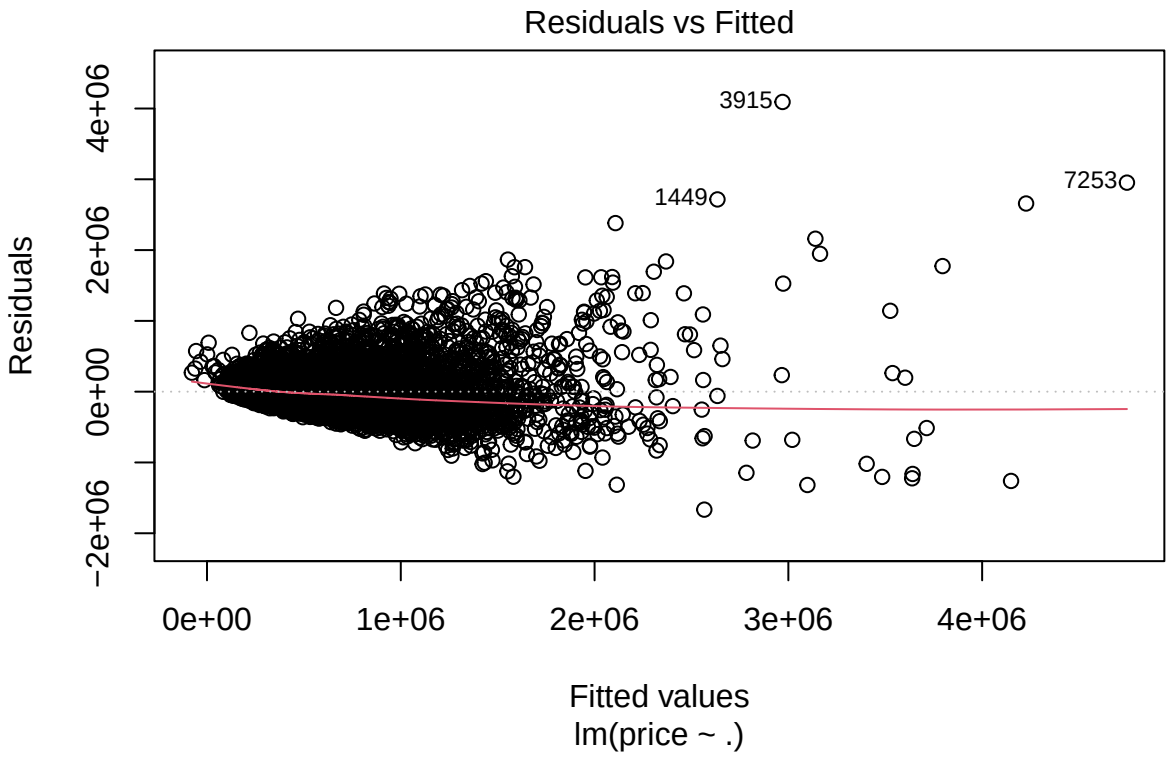



```
age.fit <- lm(price ~ house_age, data = seattle_clean)

plot(seattle_clean$house_age, seattle_clean$price,
     xlab = "House_age",
     ylab = "Price",
     cex = .5,
     col = "darkgrey")
abline(age.fit)
lines(lowess(seattle_clean$house_age, seattle_clean$price),
      col = "red")
```

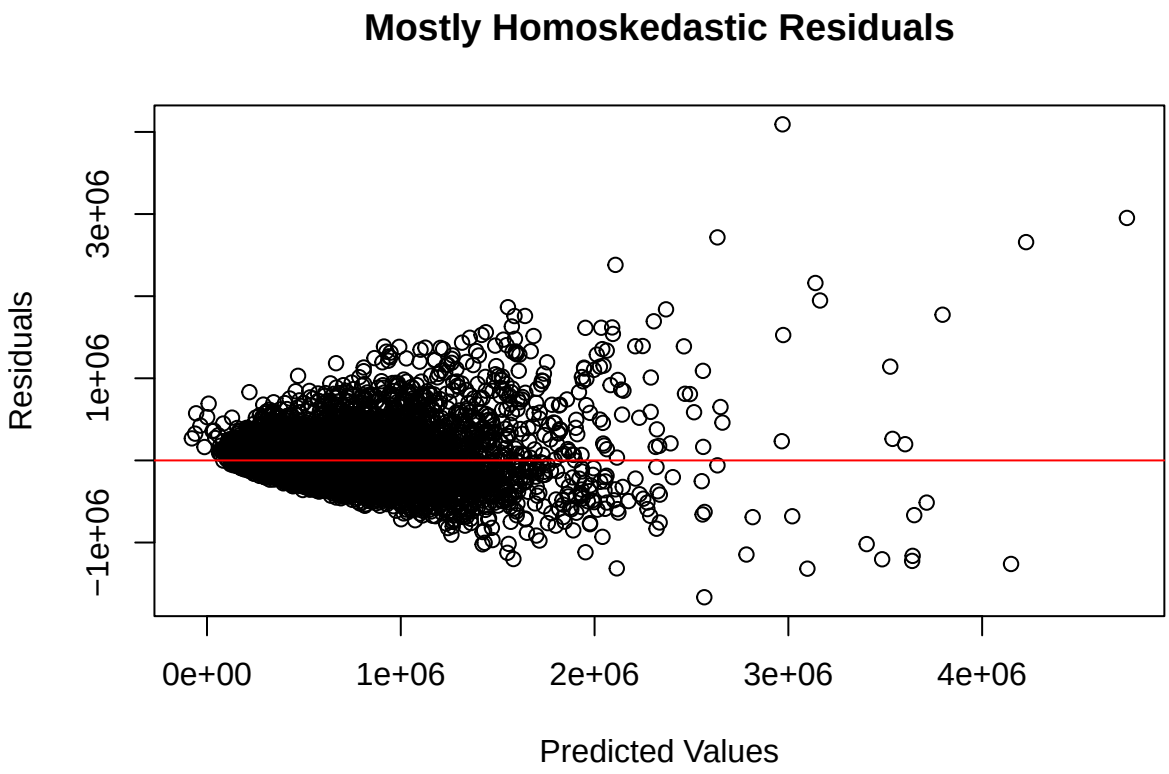


Assumption 8: constant error variance (EV)

```
plot(ols.initial, which = 1)
```




```
plot(ols.initial$residuals ~ ols.initial$fitted.values,
     main = "Mostly Homoskedastic Residuals",
     xlab = "Predicted Values",
     ylab = "Residuals")
abline(h = 0, col = "red")
```



Model 1 (ols.full)

```
ols.full <-  lm(log(price) ~ ., data = seattle_clean)

summary(ols.full)
```

```
##
## Call:
## lm(formula = log(price) ~ ., data = seattle_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.74142 -0.20584  0.01525  0.20877  1.44000
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.137e+01  3.112e-01  36.554  < 2e-16 ***
## bedrooms    -3.309e-02  3.107e-03 -10.651  < 2e-16 ***
## bathrooms    8.040e-02  5.001e-03  16.076  < 2e-16 ***
## sqft_living  1.849e-04  4.934e-06  37.471  < 2e-16 ***
## sqft_lot    -3.047e-08  5.251e-08  -0.580 0.561800
## floors       7.717e-02  5.000e-03  15.435  < 2e-16 ***
## waterfront1  3.147e-01  3.046e-02  10.330  < 2e-16 ***
```

```
## view1        1.801e-01  1.736e-02  10.373  < 2e-16 ***
## view2        9.667e-02  1.051e-02   9.200  < 2e-16 ***
## view3        1.274e-01  1.436e-02   8.872  < 2e-16 ***
## view4        2.517e-01  2.221e-02  11.331  < 2e-16 ***
## condition2  -1.390e-02  6.264e-02  -0.222 0.824410
## condition3   1.488e-01  5.831e-02   2.551 0.010739 *
## condition4   1.675e-01  5.832e-02   2.872 0.004085 **
## condition5   2.339e-01  5.864e-02   3.989 6.66e-05 ***
## grade3      -5.094e-02  3.639e-01  -0.140 0.888687
## grade4      -5.666e-02  3.214e-01  -0.176 0.860065
## grade5      -1.025e-02  3.168e-01  -0.032 0.974197
## grade6       2.077e-01  3.166e-01   0.656 0.511848
## grade7       4.869e-01  3.166e-01   1.538 0.124030
## grade8       7.226e-01  3.166e-01   2.282 0.022494 *
## grade9       9.556e-01  3.167e-01   3.017 0.002555 **
## grade10      1.115e+00  3.169e-01   3.518 0.000435 ***
## grade11      1.231e+00  3.172e-01   3.881 0.000104 ***
## grade12      1.318e+00  3.188e-01   4.136 3.55e-05 ***
## grade13      1.338e+00  3.292e-01   4.066 4.81e-05 ***
## house_age    5.785e-03  1.037e-04  55.802  < 2e-16 ***
## renovated1  -1.049e-03  1.130e-02  -0.093 0.926044
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.311 on 21584 degrees of freedom
## Multiple R-squared:  0.6517, Adjusted R-squared:  0.6512
## F-statistic:  1496 on 27 and 21584 DF,  p-value: < 2.2e-16
```

ols.full 10FCV MSE

```
set.seed(1)

train_control <- trainControl(method = "cv", number = 10)

ols.full.10f <- train(log(price) ~ ., data = seattle_clean, method = "lm", trControl = train_control)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from rank-deficient fit;
## attr(*, "non-estim") has doubtful cases
```

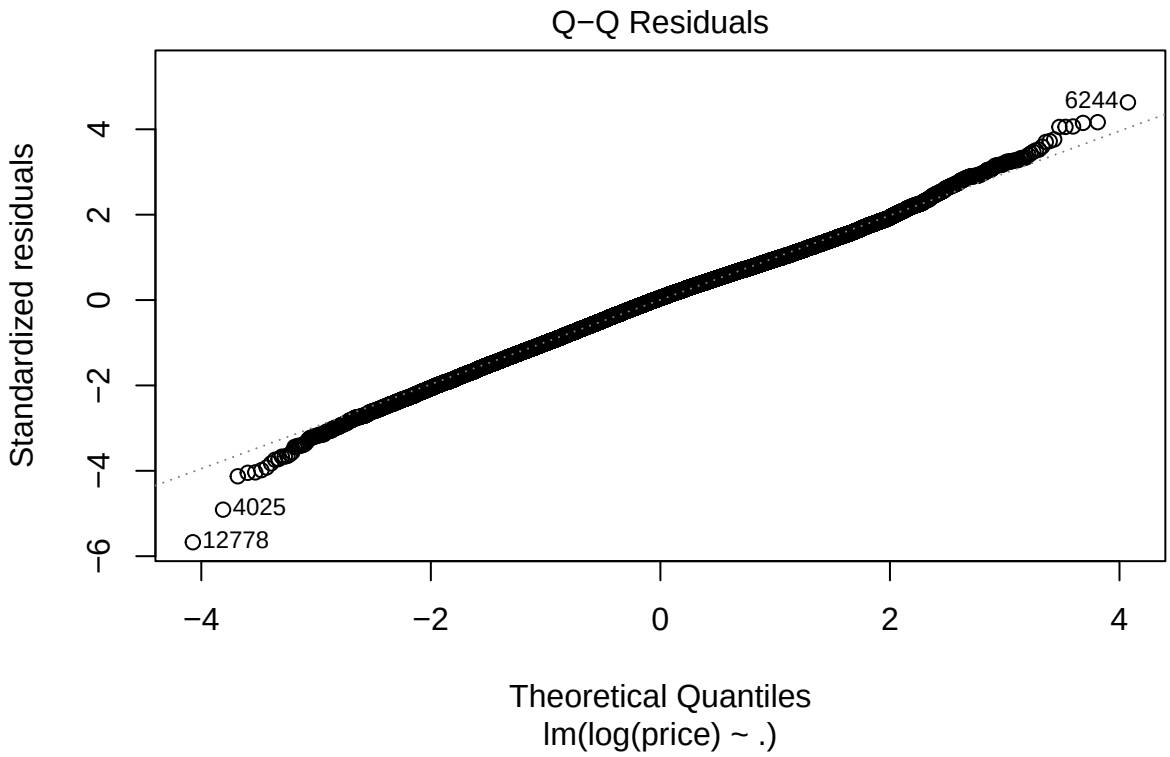
```
cbind("MSE" = ols.full.10f$results$RMSE^2)
```

```
##             MSE
## [1,] 0.09716908
```

Assumption 2: Errors are normally distributed (EN)

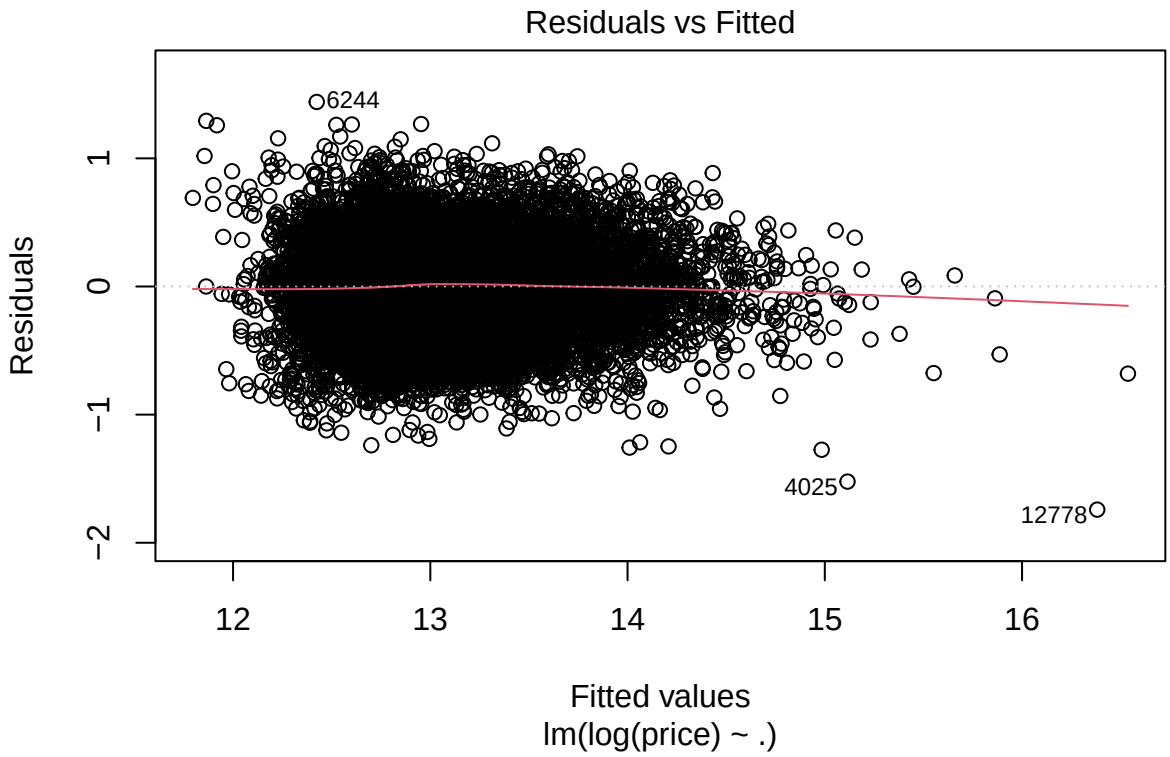
```
plot(ols.full, which = 2)
```

```
## Warning: not plotting observations with leverage one:
##   19452
```



Assumption 8: constant error variance (EV)

```
plot(ols.full, which = 1)
```



Stepwise

```
ols.null <- lm(price ~ 1, data = seattle_clean)

kval <- qchisq(0.05, 1, lower.tail = F)
kval
```

```
## [1] 3.841459
```

```
fit.step.05p <- step(ols.full, scope = list(lower = ols.null, upper = ols.full), direction = "both", tes
```

```
## Start:  AIC=-50398.54
## log(price) ~ bedrooms + bathrooms + sqft_living + sqft_lot +
##     floors + waterfront + view + condition + grade + house_age +
##     renovated
##
##               Df Sum of Sq    RSS    AIC   F value Pr(>F)
## - renovated    1      0.00 2088.2 -50402    0.0086 0.9260
## - sqft_lot     1      0.03 2088.2 -50402    0.3366 0.5618
## <none>                     2088.2 -50399
## - waterfront   1     10.32 2098.5 -50296  106.7100 <2e-16 ***
## - bedrooms     1     10.98 2099.2 -50289  113.4420 <2e-16 ***
## - condition    4     15.74 2103.9 -50252   40.6673 <2e-16 ***
```

```
## - floors       1     23.05 2111.2 -50165  238.2504 <2e-16 ***
## - bathrooms    1     25.00 2113.2 -50145  258.4428 <2e-16 ***
## - view         4     32.60 2120.8 -50079   84.2270 <2e-16 ***
## - sqft_living  1    135.84 2224.0 -49040 1404.0573 <2e-16 ***
## - house_age    1    301.26 2389.5 -47490 3113.8522 <2e-16 ***
## - grade       11    525.44 2613.6 -45590  493.7296 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=-50402.37
## log(price) ~ bedrooms + bathrooms + sqft_living + sqft_lot +
##     floors + waterfront + view + condition + grade + house_age
##
##               Df Sum of Sq    RSS    AIC   F value Pr(>F)
## - sqft_lot     1      0.03 2088.2 -50406    0.3384 0.5608
## <none>                     2088.2 -50402
## + renovated    1      0.00 2088.2 -50399    0.0086 0.9260
## - waterfront   1     10.34 2098.5 -50299  106.8858 <2e-16 ***
## - bedrooms     1     10.99 2099.2 -50293  113.5764 <2e-16 ***
## - condition    4     15.97 2104.2 -50253   41.2711 <2e-16 ***
## - floors       1     23.08 2111.3 -50169  238.5527 <2e-16 ***
## - bathrooms    1     25.38 2113.6 -50145  262.3960 <2e-16 ***
## - view         4     32.61 2120.8 -50083   84.2717 <2e-16 ***
## - sqft_living  1    135.85 2224.1 -49044 1404.2144 <2e-16 ***
## - house_age    1    338.29 2426.5 -47161 3496.7438 <2e-16 ***
## - grade       11    525.62 2613.8 -45592  493.9261 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=-50405.87
## log(price) ~ bedrooms + bathrooms + sqft_living + floors + waterfront +
##     view + condition + grade + house_age
##
##               Df Sum of Sq    RSS    AIC   F value Pr(>F)
## <none>                     2088.2 -50406
## + sqft_lot     1      0.03 2088.2 -50402    0.3384 0.5608
## + renovated    1      0.00 2088.2 -50402    0.0103 0.9190
## - waterfront   1     10.34 2098.6 -50303  106.8396 <2e-16 ***
## - bedrooms     1     10.96 2099.2 -50297  113.2798 <2e-16 ***
## - condition    4     16.01 2104.2 -50256   41.3817 <2e-16 ***
## - floors       1     23.27 2111.5 -50170  240.4942 <2e-16 ***
## - bathrooms    1     25.47 2113.7 -50148  263.2626 <2e-16 ***
## - view         4     32.61 2120.8 -50086   84.2626 <2e-16 ***
## - sqft_living  1    138.11 2226.3 -49026 1427.6456 <2e-16 ***
## - house_age    1    339.57 2427.8 -47154 3510.0799 <2e-16 ***
## - grade       11    526.34 2614.6 -45590  494.6123 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(fit.step.05p)
```

```
##
## Call:
## lm(formula = log(price) ~ bedrooms + bathrooms + sqft_living +
```

```
##     floors + waterfront + view + condition + grade + house_age,
##     data = seattle_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.74661 -0.20576  0.01512  0.20882  1.44002
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.137e+01  3.111e-01  36.555  < 2e-16 ***
## bedrooms    -3.295e-02  3.096e-03 -10.643  < 2e-16 ***
## bathrooms    8.043e-02  4.957e-03  16.225  < 2e-16 ***
## sqft_living  1.845e-04  4.882e-06  37.784  < 2e-16 ***
## floors       7.733e-02  4.986e-03  15.508  < 2e-16 ***
## waterfront1  3.145e-01  3.042e-02  10.336  < 2e-16 ***
## view1        1.803e-01  1.735e-02  10.388  < 2e-16 ***
## view2        9.658e-02  1.051e-02   9.193  < 2e-16 ***
## view3        1.270e-01  1.435e-02   8.855  < 2e-16 ***
## view4        2.518e-01  2.220e-02  11.343  < 2e-16 ***
## condition2  -1.427e-02  6.264e-02  -0.228 0.819748
## condition3   1.489e-01  5.830e-02   2.555 0.010630 *
## condition4   1.677e-01  5.831e-02   2.875 0.004045 **
## condition5   2.342e-01  5.864e-02   3.994 6.52e-05 ***
## grade3      -5.117e-02  3.639e-01  -0.141 0.888175
## grade4      -5.690e-02  3.214e-01  -0.177 0.859462
## grade5      -1.061e-02  3.168e-01  -0.033 0.973280
## grade6       2.076e-01  3.165e-01   0.656 0.511883
## grade7       4.870e-01  3.166e-01   1.538 0.123955
## grade8       7.227e-01  3.166e-01   2.283 0.022466 *
## grade9       9.557e-01  3.167e-01   3.018 0.002551 **
## grade10      1.115e+00  3.169e-01   3.519 0.000435 ***
## grade11      1.231e+00  3.172e-01   3.882 0.000104 ***
## grade12      1.318e+00  3.188e-01   4.136 3.55e-05 ***
## grade13      1.340e+00  3.292e-01   4.070 4.71e-05 ***
## house_age    5.785e-03  9.764e-05  59.246  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.311 on 21586 degrees of freedom
## Multiple R-squared:  0.6517, Adjusted R-squared:  0.6513
## F-statistic:  1615 on 25 and 21586 DF,  p-value: < 2.2e-16
```

Stepwise removed "sqft_lot" and "renovated" variable

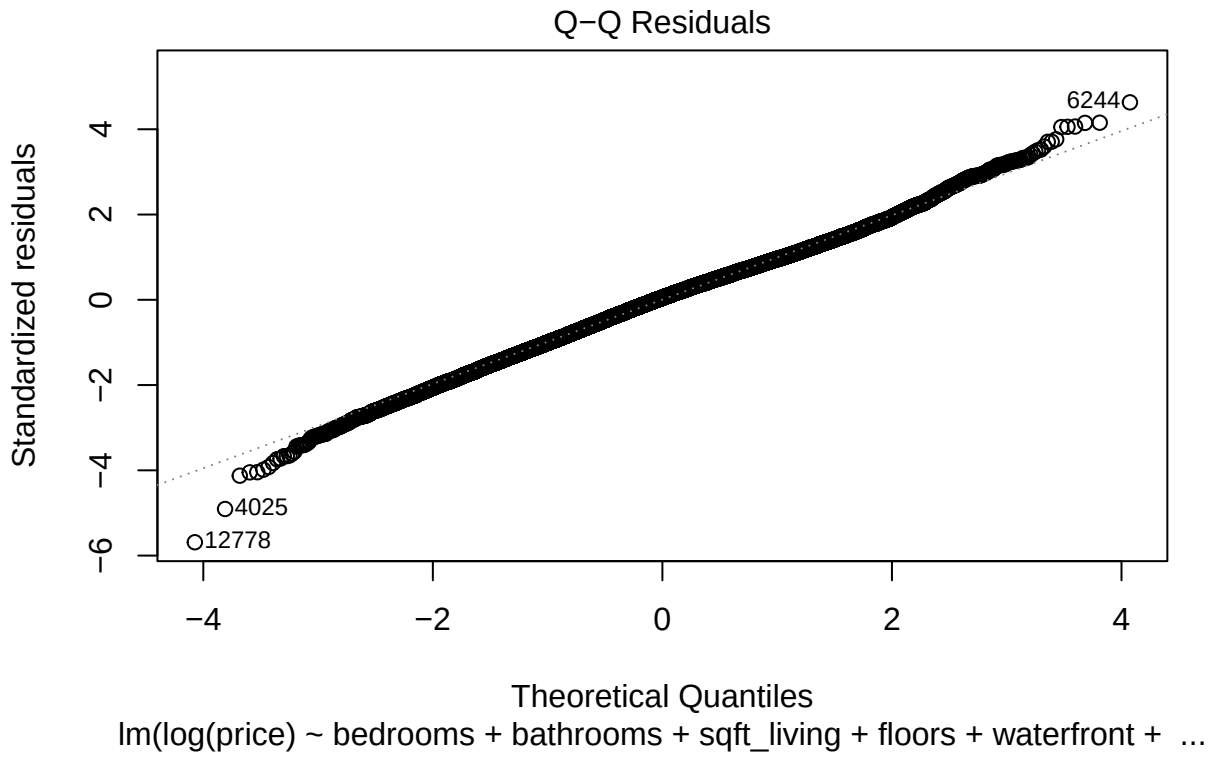
Model 2 (ols.small)

```
ols.small <- lm(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age,
    data = seattle_clean)

summary(ols.small)
```

```
##
## Call:
## lm(formula = log(price) ~ bedrooms + bathrooms + sqft_living +
##     floors + waterfront + view + condition + grade + house_age,
##     data = seattle_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.74661 -0.20576  0.01512  0.20882  1.44002
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.137e+01  3.111e-01  36.555  < 2e-16 ***
## bedrooms    -3.295e-02  3.096e-03 -10.643  < 2e-16 ***
## bathrooms    8.043e-02  4.957e-03  16.225  < 2e-16 ***
## sqft_living  1.845e-04  4.882e-06  37.784  < 2e-16 ***
## floors       7.733e-02  4.986e-03  15.508  < 2e-16 ***
## waterfront1  3.145e-01  3.042e-02  10.336  < 2e-16 ***
## view1        1.803e-01  1.735e-02  10.388  < 2e-16 ***
## view2        9.658e-02  1.051e-02   9.193  < 2e-16 ***
## view3        1.270e-01  1.435e-02   8.855  < 2e-16 ***
## view4        2.518e-01  2.220e-02  11.343  < 2e-16 ***
## condition2  -1.427e-02  6.264e-02  -0.228 0.819748
## condition3   1.489e-01  5.830e-02   2.555 0.010630 *
## condition4   1.677e-01  5.831e-02   2.875 0.004045 **
## condition5   2.342e-01  5.864e-02   3.994 6.52e-05 ***
## grade3      -5.117e-02  3.639e-01  -0.141 0.888175
## grade4      -5.690e-02  3.214e-01  -0.177 0.859462
## grade5      -1.061e-02  3.168e-01  -0.033 0.973280
## grade6       2.076e-01  3.165e-01   0.656 0.511883
## grade7       4.870e-01  3.166e-01   1.538 0.123955
## grade8       7.227e-01  3.166e-01   2.283 0.022466 *
## grade9       9.557e-01  3.167e-01   3.018 0.002551 **
## grade10      1.115e+00  3.169e-01   3.519 0.000435 ***
## grade11      1.231e+00  3.172e-01   3.882 0.000104 ***
## grade12      1.318e+00  3.188e-01   4.136 3.55e-05 ***
## grade13      1.340e+00  3.292e-01   4.070 4.71e-05 ***
## house_age    5.785e-03  9.764e-05  59.246  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.311 on 21586 degrees of freedom
## Multiple R-squared:  0.6517, Adjusted R-squared:  0.6513
## F-statistic:  1615 on 25 and 21586 DF,  p-value: < 2.2e-16
```

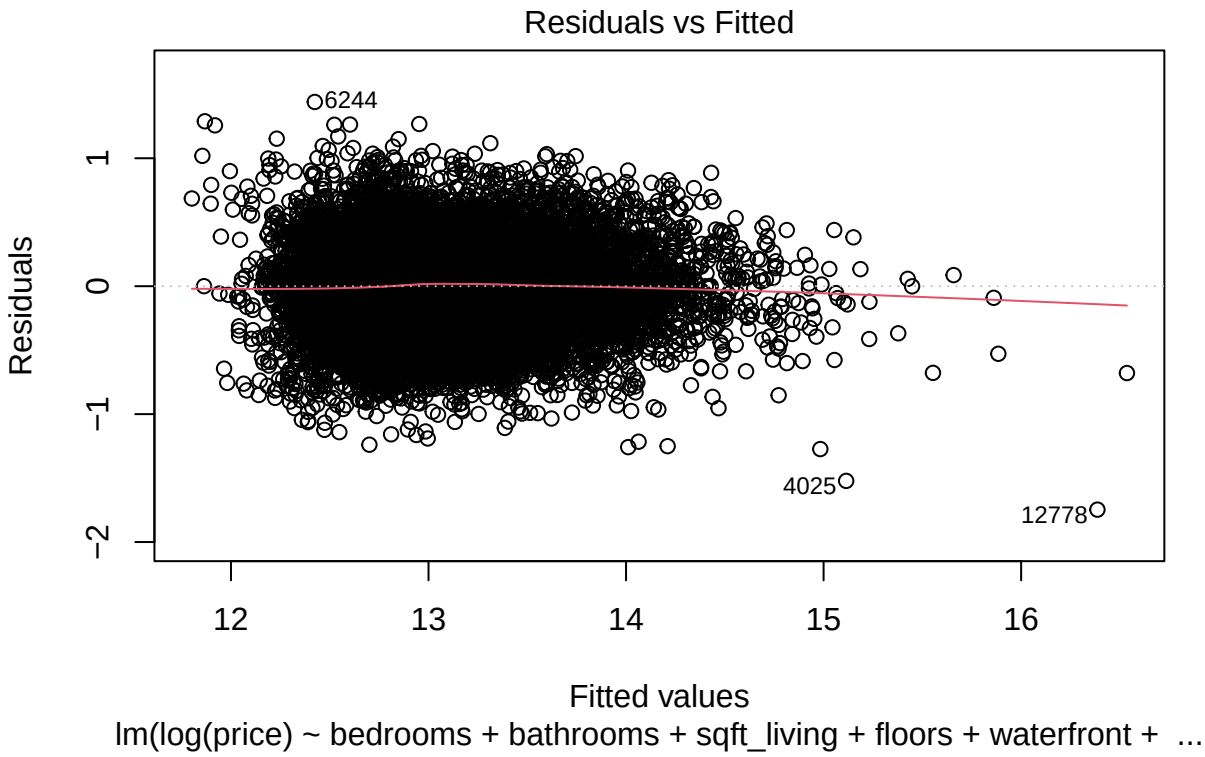
Assumption 2: Errors are normally distributed (EN)

```
plot(ols.small, which = 2)
```



Assumption 8: constant error variance (EV)

```
plot(ols.small, which = 1)
```



ols.small 10FCV MSE

```
set.seed(1)

train_control <- trainControl(method = "cv", number = 10)

ols.small.10f <- train(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, data = seattle_clean, method = "lm", tr
```

```
## Warning in predict.lm(modelFit, newdata): prediction from rank-deficient fit;
## attr(*, "non-estim") has doubtful cases
```

```
cbind("MSE" = ols.small.10f$results$RMSE^2)
```

```
##            MSE
## [1,] 0.0971286
```

Model 3 (WLS.full)

```r
wts.full <- 1 / lm(abs(ols.full$residuals) ~ ols.full$fitted.values)$fitted.values^2
```

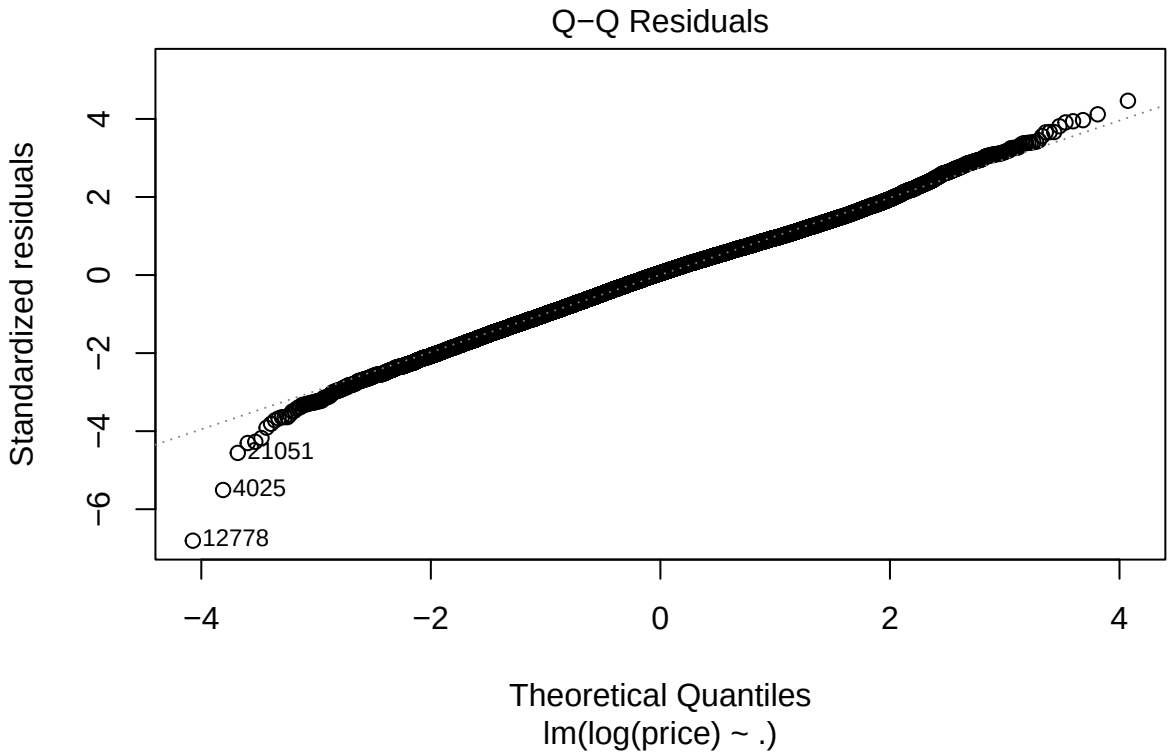
```r
wls.full <- lm(log(price) ~ ., data = seattle_clean, weights = wts.full)

summary(wls.full)
```

```
##
## Call:
## lm(formula = log(price) ~ ., data = seattle_clean, weights = wts.full)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4504 -0.8392  0.0625  0.8466  5.6431
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.138e+01  3.322e-01  34.257  < 2e-16 ***
## bedrooms    -3.168e-02  3.090e-03 -10.251  < 2e-16 ***
## bathrooms    7.969e-02  4.957e-03  16.077  < 2e-16 ***
## sqft_living  1.815e-04  4.837e-06  37.521  < 2e-16 ***
## sqft_lot    -6.167e-08  5.178e-08  -1.191 0.233628
## floors       7.536e-02  4.982e-03  15.126  < 2e-16 ***
## waterfront1  3.208e-01  2.880e-02  11.139  < 2e-16 ***
## view1        1.802e-01  1.697e-02  10.621  < 2e-16 ***
## view2        9.527e-02  1.030e-02   9.252  < 2e-16 ***
## view3        1.268e-01  1.391e-02   9.116  < 2e-16 ***
## view4        2.464e-01  2.115e-02  11.648  < 2e-16 ***
## condition2  -1.442e-02  6.453e-02  -0.223 0.823163
## condition3   1.468e-01  6.007e-02   2.443 0.014555 *
## condition4   1.663e-01  6.007e-02   2.769 0.005625 **
## condition5   2.341e-01  6.038e-02   3.877 0.000106 ***
## grade3      -4.990e-02  3.871e-01  -0.129 0.897424
## grade4      -5.775e-02  3.426e-01  -0.169 0.866139
## grade5      -9.113e-03  3.378e-01  -0.027 0.978478
## grade6       2.097e-01  3.376e-01   0.621 0.534399
## grade7       4.906e-01  3.376e-01   1.453 0.146144
## grade8       7.278e-01  3.376e-01   2.156 0.031116 *
## grade9       9.633e-01  3.377e-01   2.853 0.004342 **
## grade10      1.125e+00  3.378e-01   3.331 0.000866 ***
## grade11      1.243e+00  3.382e-01   3.676 0.000237 ***
## grade12      1.326e+00  3.394e-01   3.907 9.37e-05 ***
## grade13      1.342e+00  3.471e-01   3.865 0.000111 ***
## house_age    5.753e-03  1.030e-04  55.842  < 2e-16 ***
## renovated1   2.149e-04  1.116e-02   0.019 0.984634
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.264 on 21584 degrees of freedom
## Multiple R-squared:  0.6625, Adjusted R-squared:  0.6621
## F-statistic:  1569 on 27 and 21584 DF,  p-value: < 2.2e-16
```

Assumption 2: Errors are normally distributed (EN)

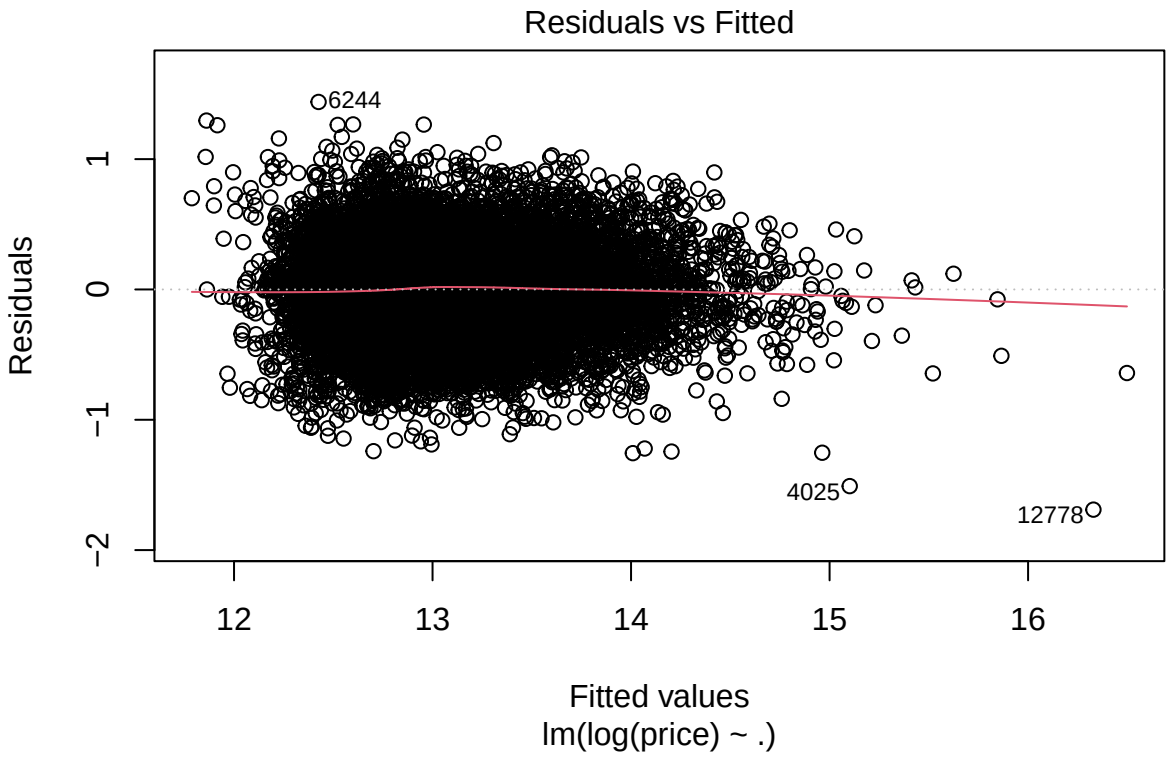
```
plot(wls.full, which = 2)
```

```
## Warning: not plotting observations with leverage one:
##   19452
```



Assumption 8: constant error variance (EV)

```
plot(wls.full, which = 1)
```



wls.full 10FCV MSE

```
set.seed(1)

train_control <- trainControl(method = "cv", number = 10)

wls.full.10f <- train(log(price) ~ ., data = seattle_clean, method = "lm", trControl = train_control, w
```

```
## Warning in predict.lm(modelFit, newdata): prediction from rank-deficient fit;
## attr(*, "non-estim") has doubtful cases
```

```
print(wls.full.10f$results$RMSE^2)
```

```
## [1] 0.09717929
```

Model 4 (WLS.small)

```
wts.small <- 1 / lm(abs(ols.small$residuals) ~ ols.small$fitted.values)$fitted.values^2
```

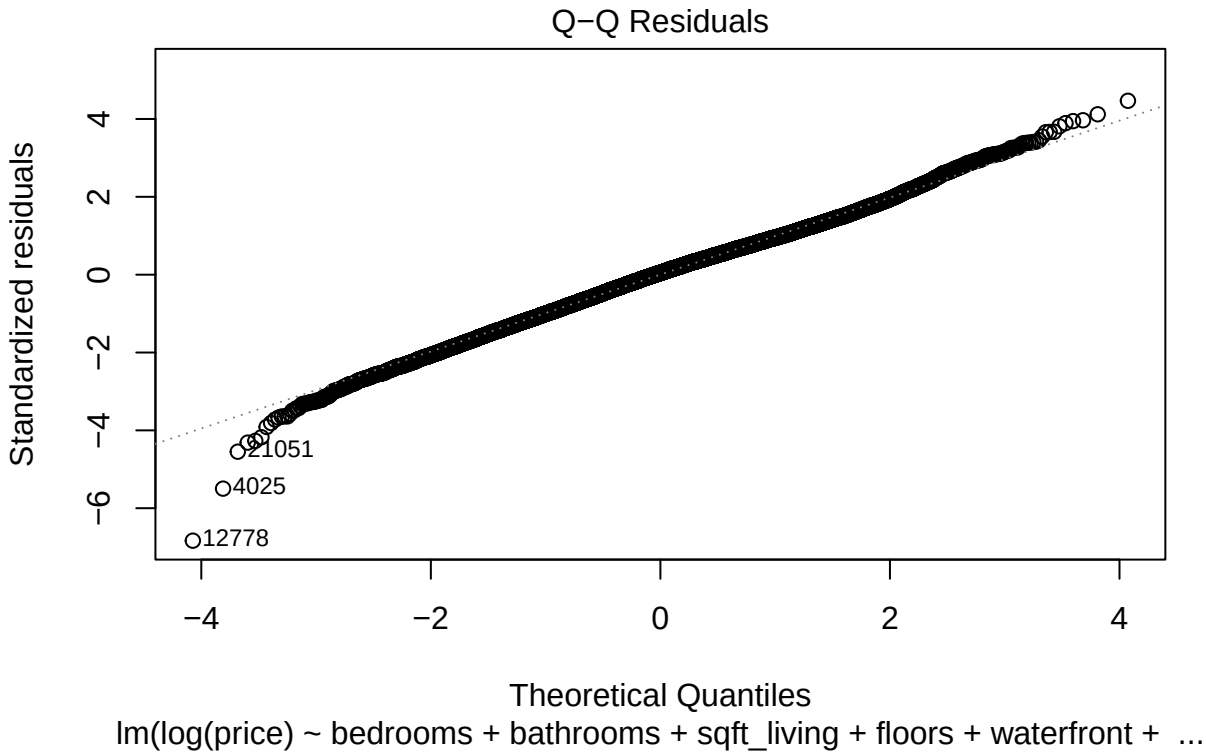
```r
wls.small <- lm(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, data = seattle_clean, weights = wts.sma

summary(wls.small)
```

```
##
## Call:
## lm(formula = log(price) ~ bedrooms + bathrooms + sqft_living +
##     floors + waterfront + view + condition + grade + house_age,
##     data = seattle_clean, weights = wts.small)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4914 -0.8389  0.0617  0.8456  5.6449
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.138e+01  3.320e-01  34.275  < 2e-16 ***
## bedrooms    -3.142e-02  3.079e-03 -10.206  < 2e-16 ***
## bathrooms    7.988e-02  4.912e-03  16.262  < 2e-16 ***
## sqft_living  1.807e-04  4.785e-06  37.757  < 2e-16 ***
## floors       7.573e-02  4.969e-03  15.241  < 2e-16 ***
## waterfront1  3.207e-01  2.877e-02  11.144  < 2e-16 ***
## view1        1.807e-01  1.696e-02  10.651  < 2e-16 ***
## view2        9.513e-02  1.030e-02   9.238  < 2e-16 ***
## view3        1.262e-01  1.391e-02   9.076  < 2e-16 ***
## view4        2.468e-01  2.115e-02  11.669  < 2e-16 ***
## condition2  -1.511e-02  6.451e-02  -0.234 0.814857
## condition3   1.473e-01  6.004e-02   2.454 0.014126 *
## condition4   1.667e-01  6.005e-02   2.777 0.005494 **
## condition5   2.346e-01  6.036e-02   3.887 0.000102 ***
## grade3      -5.062e-02  3.869e-01  -0.131 0.895889
## grade4      -5.849e-02  3.424e-01  -0.171 0.864336
## grade5      -1.018e-02  3.376e-01  -0.030 0.975948
## grade6       2.094e-01  3.373e-01   0.621 0.534791
## grade7       4.905e-01  3.374e-01   1.454 0.145966
## grade8       7.278e-01  3.374e-01   2.157 0.031008 *
## grade9       9.633e-01  3.375e-01   2.854 0.004318 **
## grade10      1.125e+00  3.376e-01   3.333 0.000860 ***
## grade11      1.243e+00  3.379e-01   3.678 0.000235 ***
## grade12      1.326e+00  3.392e-01   3.909 9.31e-05 ***
## grade13      1.344e+00  3.469e-01   3.875 0.000107 ***
## house_age    5.760e-03  9.678e-05  59.517  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.264 on 21586 degrees of freedom
## Multiple R-squared:  0.6623, Adjusted R-squared:  0.6619
## F-statistic:  1694 on 25 and 21586 DF,  p-value: < 2.2e-16
```

Assumption 2: Errors are normally distributed (EN)

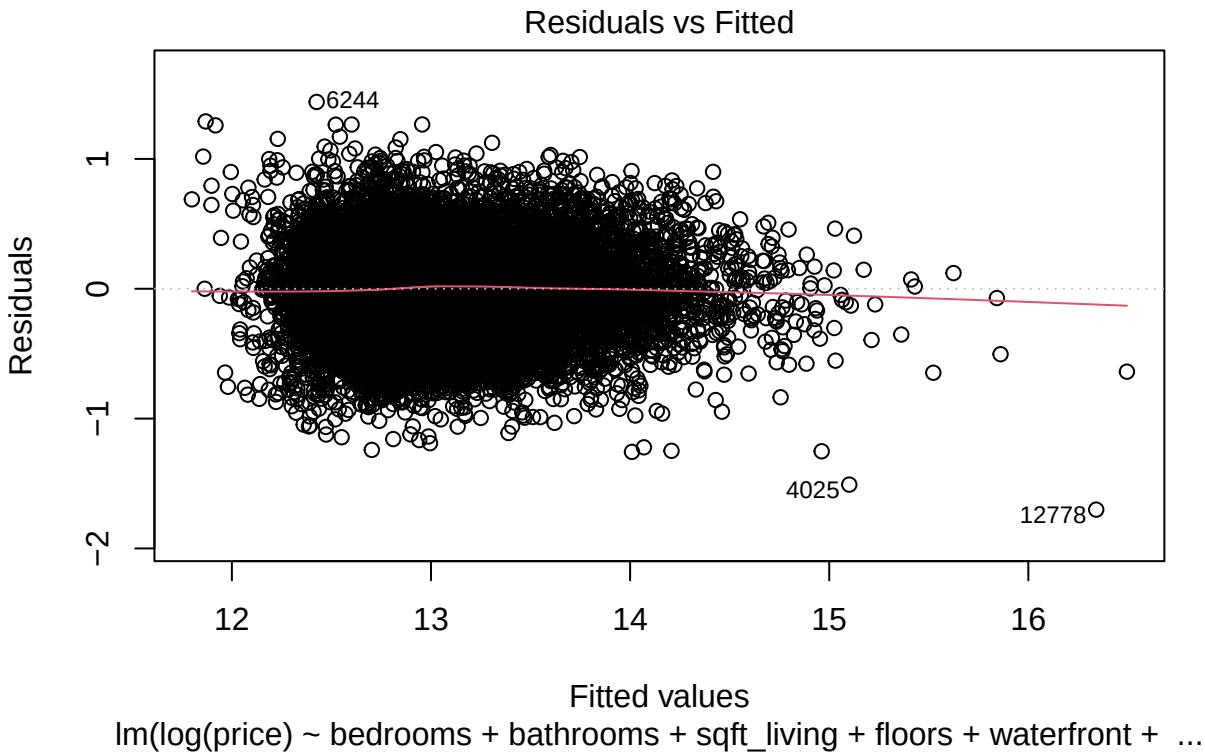
```
plot(wls.small, which = 2)
```

```
## Warning: not plotting observations with leverage one:
##   19452
```



Assumption 8: constant error variance (EV)

```
plot(wls.small, which = 1)
```



wls.small 10FCV MSE

```
set.seed(1)

train_control <- trainControl(method = "cv", number = 10)

wls.small.10f <- train(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, data = seattle_clean, method = "lm", tr
```

```
## Warning in predict.lm(modelFit, newdata): prediction from rank-deficient fit;
## attr(*, "non-estim") has doubtful cases
```

```
print(wls.small.10f$results$RMSE^2)
```

```
## [1] 0.09713529
```

Model 5 (LASSO.full)

```
set.seed(1)
```

```r
x1 <- model.matrix(log(price) ~ ., data = seattle_clean)[, -1]

y1 <- log(seattle_clean$price)

lasso.full <- cv.glmnet(x1, y1, alpha = 1)

lasso.full.best.lambda <- lasso.full$lambda.min

min.cv.lasso.full <- min(lasso.full$cvm)

cbind("Best Lambda" = lasso.full.best.lambda, "Best 10FCV MSE" = min.cv.lasso.full)
```

```
##       Best Lambda Best 10FCV MSE
## [1,] 0.0005438469     0.09711023
```

Model 6 (LASSO.full.wts)

```r
set.seed(1)

x1wts <- model.matrix(log(price) ~ ., weights = wts.full, data = seattle_clean)[, -1]

y1wts <- log(seattle_clean$price)

lasso.full.wts <- cv.glmnet(x1wts, y1wts, weights = wts.full, alpha = 1)

lasso.full.wts.best.lambda <- lasso.full.wts$lambda.min

min.cv.lasso.full.wts <- min(lasso.full.wts$cvm)

cbind("Best Lambda" = lasso.full.wts.best.lambda, "Best 10FCV MSE" = min.cv.lasso.full.wts)
```

```
##      Best Lambda Best 10FCV MSE
## [1,] 0.000558692     0.09708893
```

Model 7 (LASSO.small)

```r
set.seed(1)

x2 <- model.matrix(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, data = seattle_clean)[, -1]

y2 <- log(seattle_clean$price)

lasso.small <- cv.glmnet(x2, y2, alpha = 1)

lasso.small.best.lambda <- lasso.small$lambda.min
```

```
min.cv.lasso.small <- min(lasso.small$cvm)

cbind("Best Lambda" = lasso.small.best.lambda, "Best 10FCV MSE" = min.cv.lasso.small)
```

```
##       Best Lambda Best 10FCV MSE
## [1,] 0.0005438469     0.09709113
```

model 8 (LASSO.small.wts)

```
set.seed(1)

x2wts <- model.matrix(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, weights = wts.small, data = seattle_clea

y2wts <- log(seattle_clean$price)

lasso.small.wts <- cv.glmnet(x2wts, y2wts, weights = wts.small, alpha = 1)

lasso.small.wts.best.lambda <- lasso.small.wts$lambda.min

min.cv.lasso.small.wts <- min(lasso.small.wts$cvm)

cbind("Best Lambda" = lasso.small.wts.best.lambda, "Best 10FCV MSE" = min.cv.lasso.small.wts)
```

```
##       Best Lambda Best 10FCV MSE
## [1,] 0.0005585243     0.09706957
```

Model 9 (RF.full)

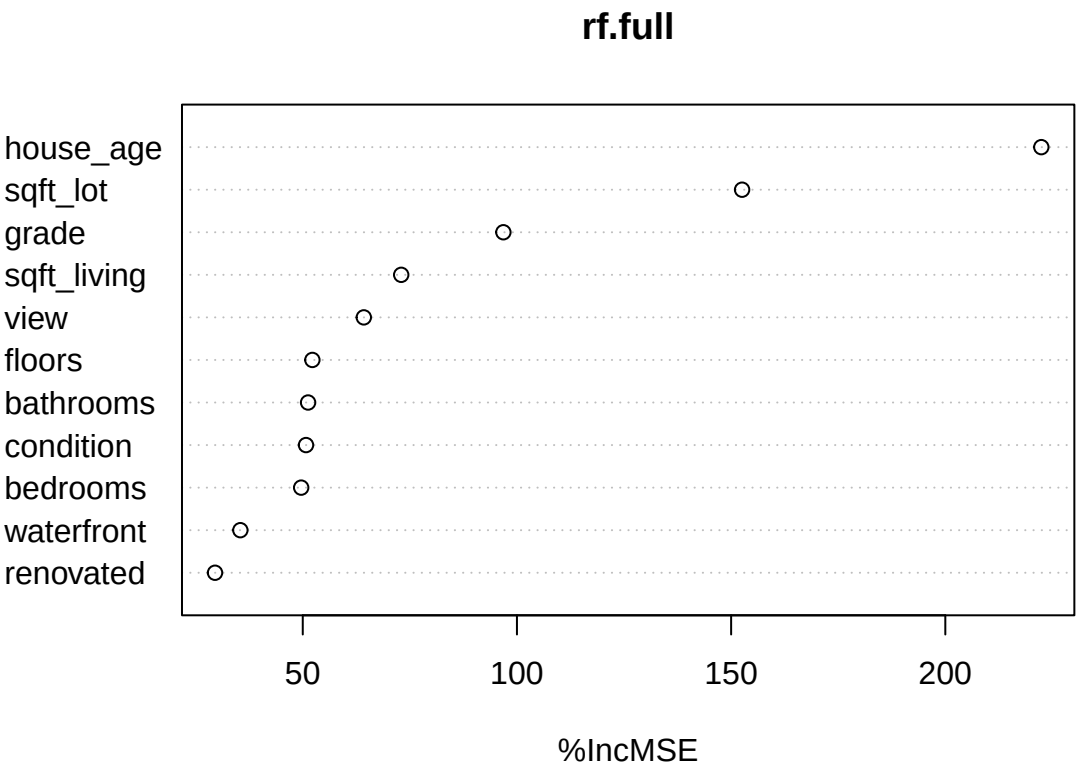
```
set.seed(1)

rf.full <- randomForest(log(price) ~ ., data = seattle_clean, mtry = 4, importance = T)

rf.full
```

```
##
## Call:
##  randomForest(formula = log(price) ~ ., data = seattle_clean,      mtry = 4, importance = T)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 0.08438146
##                     % Var explained: 69.58
```

```r
cbind("RF.full MSE" = mean(rf.full$mse))
```

```
##      RF.full MSE
## [1,]  0.08629973
```

```r
varImpPlot(rf.full, type = 1)
```




```r
importance(rf.full, type = 1)
```

```
##                %IncMSE
## bedrooms     49.63291
## bathrooms    51.24166
## sqft_living  72.98404
## sqft_lot    152.55696
## floors       52.24112
## waterfront   35.42999
## view         64.24916
## condition    50.75915
## grade        96.82785
## house_age   222.40691
## renovated    29.51846
```

Model 10 (RF.small)

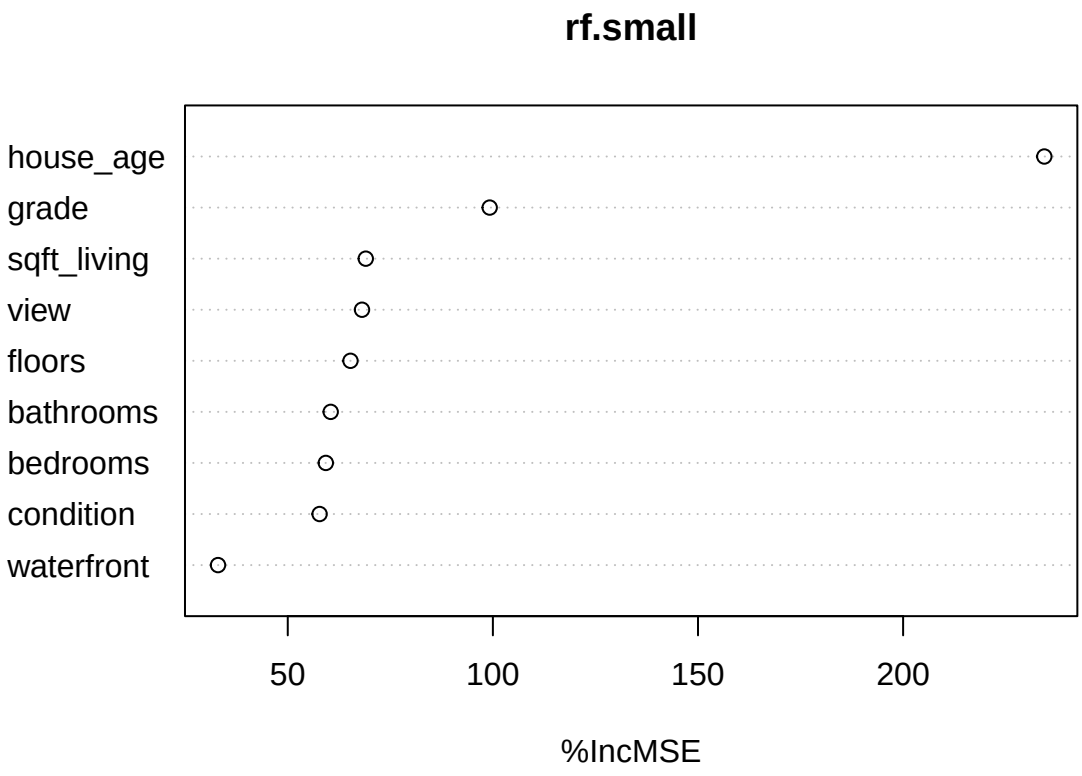
```
set.seed(1)

rf.small <- randomForest(log(price) ~ bedrooms + bathrooms + sqft_living +
    floors + waterfront + view + condition + grade + house_age, data = seattle_clean, mtry = 3, importa

rf.small
```

```
##
## Call:
##  randomForest(formula = log(price) ~ bedrooms + bathrooms + sqft_living +      floors + waterfront +
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.09003505
##                     % Var explained: 67.54
```

```
cbind("RF.small MSE" = mean(rf.small$mse))
```

```
##      RF.small MSE
## [1,]   0.09152555
```

```
varImpPlot(rf.small, type = 1)
```



```r
importance(rf.small, type = 1)
```

```
##               %IncMSE
## bedrooms     59.27591
## bathrooms    60.46377
## sqft_living  69.01131
## floors       65.28760
## waterfront   32.99931
## view         68.09564
## condition    57.75665
## grade        99.21919
## house_age   234.41597
```

Final MSE Results

```r
cbind("OLS.full MSE" = ols.full.10f$results$RMSE^2, "OLS.small MSE" = ols.small.10f$results$RMSE^2, "WLS
```

```
##      OLS.full MSE OLS.small MSE WLS.full MSE WLS.small MSE LASSO.full MSE
## [1,]   0.09716908     0.0971286   0.09717929    0.09713529     0.09711023
##      LASSO.full.wts MSE LASSO.small MSE LASSO.small.wts MSE RF.full MSE
## [1,]         0.09708893      0.09709113          0.09706957  0.08629973
##      RF.small MSE
## [1,]   0.09152555
```

```r
results <- data.frame(
  Method = c("OLS.full", "OLS.small", "WLS.full", "WLS.small", "LASSO.full", "LASSO.full.wts", "LASSO.sm
  MSE = c(ols.full.10f$results$RMSE^2, ols.small.10f$results$RMSE^2, wls.full.10f$results$RMSE^2, wls.sm
)

results$RMSE <- sqrt(results$MSE)

results
```

```
##             Method        MSE      RMSE
## 1         OLS.full 0.09716908 0.3117196
## 2        OLS.small 0.09712860 0.3116546
## 3         WLS.full 0.09717929 0.3117359
## 4        WLS.small 0.09713529 0.3116653
## 5       LASSO.full 0.09711023 0.3116251
## 6   LASSO.full.wts 0.09708893 0.3115910
## 7      LASSO.small 0.09709113 0.3115945
## 8  LASSO.small.wts 0.09706957 0.3115599
## 9          RF.full 0.08629973 0.2937682
## 10        RF.small 0.09152555 0.3025319
```