Final_5205

Yufei Zhao

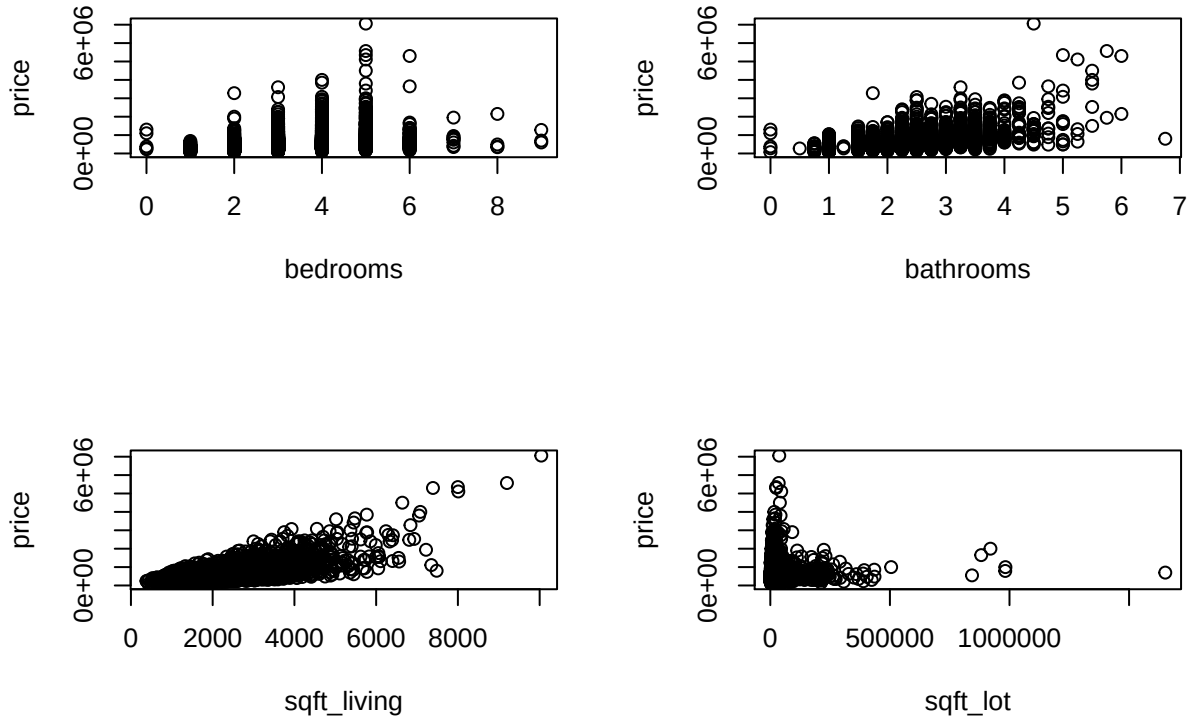
12/3/2016

```
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```

(a)

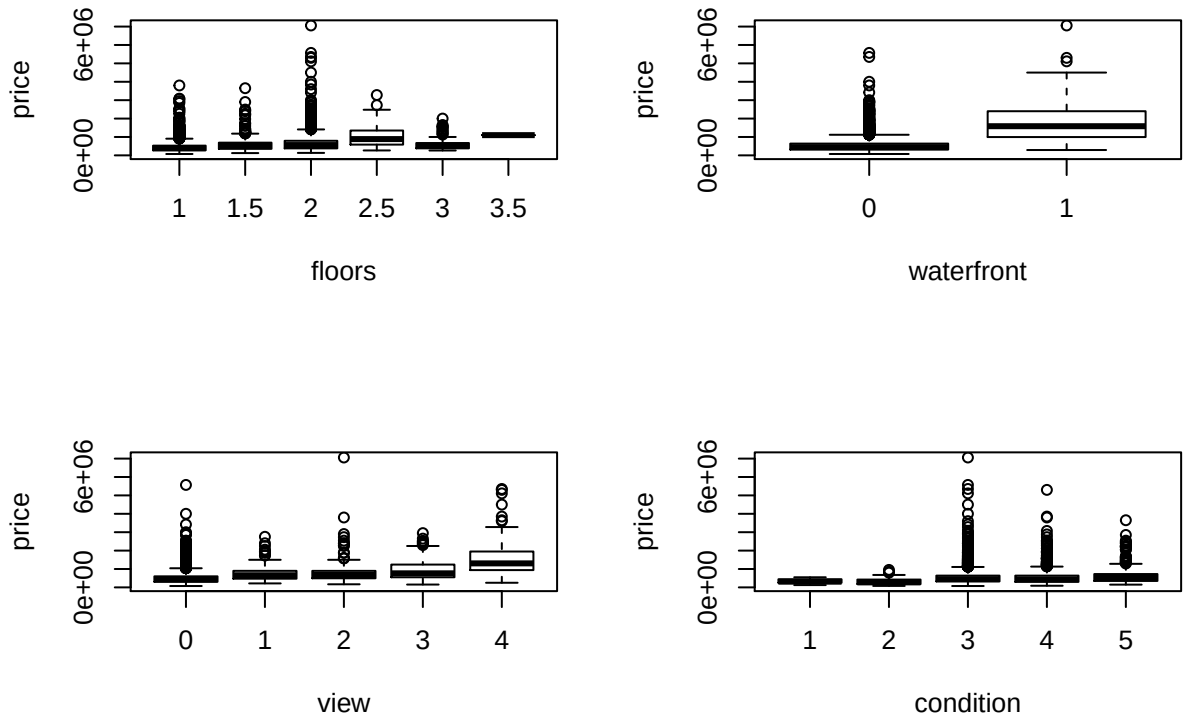
```
train <- read.csv("training.csv", as.is = TRUE, header = TRUE)
test <- read.csv("test.csv", as.is = TRUE, header = TRUE)
```

```
par(mfrow = c(2, 2))
plot(price ~ bedrooms, train)
plot(price ~ bathrooms, train)
plot(price ~ sqft_living, train)
plot(price ~ sqft_lot, train)
```



Based on above 2 by 2 plot matrix, there are positive linear relationships between price and bedrooms, price and bathrooms, price and sqft_living. Hence we should include bedrooms, bathrooms, and sqft_living in our model.

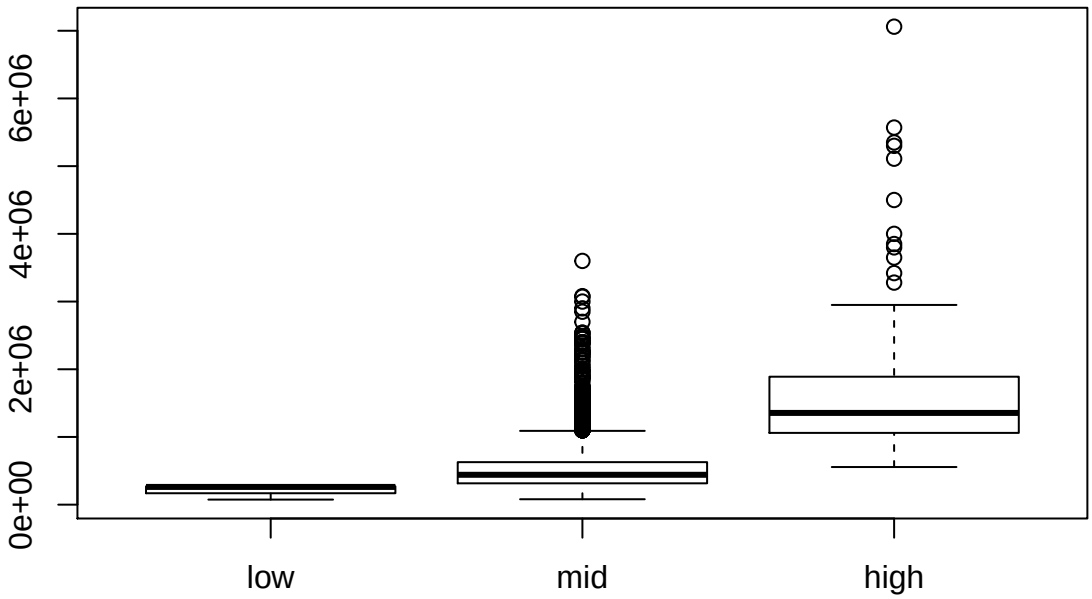
```
par(mfrow = c(2, 2))
boxplot(price ~ floors, train, xlab = "floors", ylab = "price")
boxplot(price ~ waterfront, train, xlab = "waterfront", ylab = "price")
boxplot(price ~ view, train, xlab = "view", ylab = "price")
boxplot(price ~ condition, train, xlab = "condition", ylab = "price")
```



Based on above 2 by 2 plot matrix, we can see there is a big price difference between whether the house has waterfront or not. For the rest of 3 plots, there is no obvious trending or pattern that shows us should include these variables: floors, view, condition. In a word, I prefer to only add waterfront into my model.

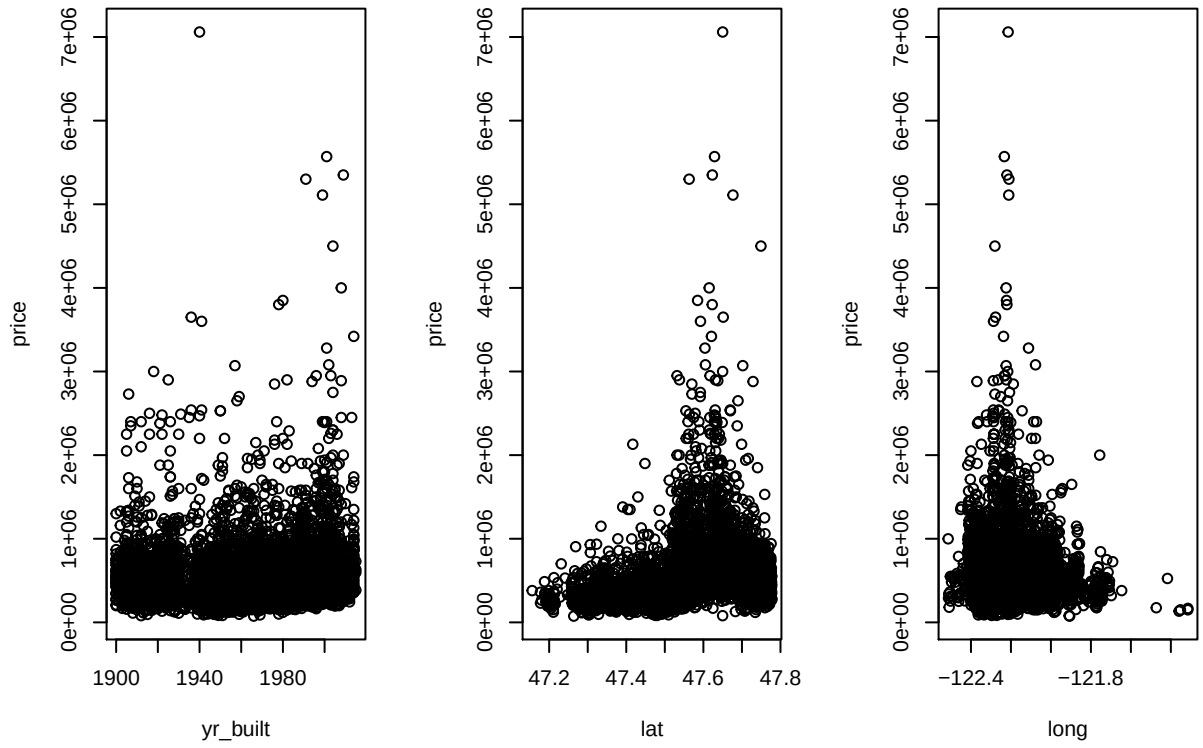
Grade:

```
f <- function(e) {
    if (e <= 3) {
        return("1")
    } else if (e >= 11 & e <= 13) {
        return("3")
    } else {
        return("2")
    }
}
train$Grade <- sapply(train$grade, f)
boxplot(price ~ Grade, train, names = c("low", "mid", "high"))
```



I try to categorize the grade variable into three levels: low:1-3, middle:4-10, high:11-13 based on description from readme.txt. In that boxplot, we can see there is outstanding trend shows us price goes up as the level goes up. Hence I think I should include this variable into my model.

```
par(mfrow = c(1, 3))
plot(price ~ yr_built, train)
plot(price ~ lat, train)
plot(price ~ long, train)
```

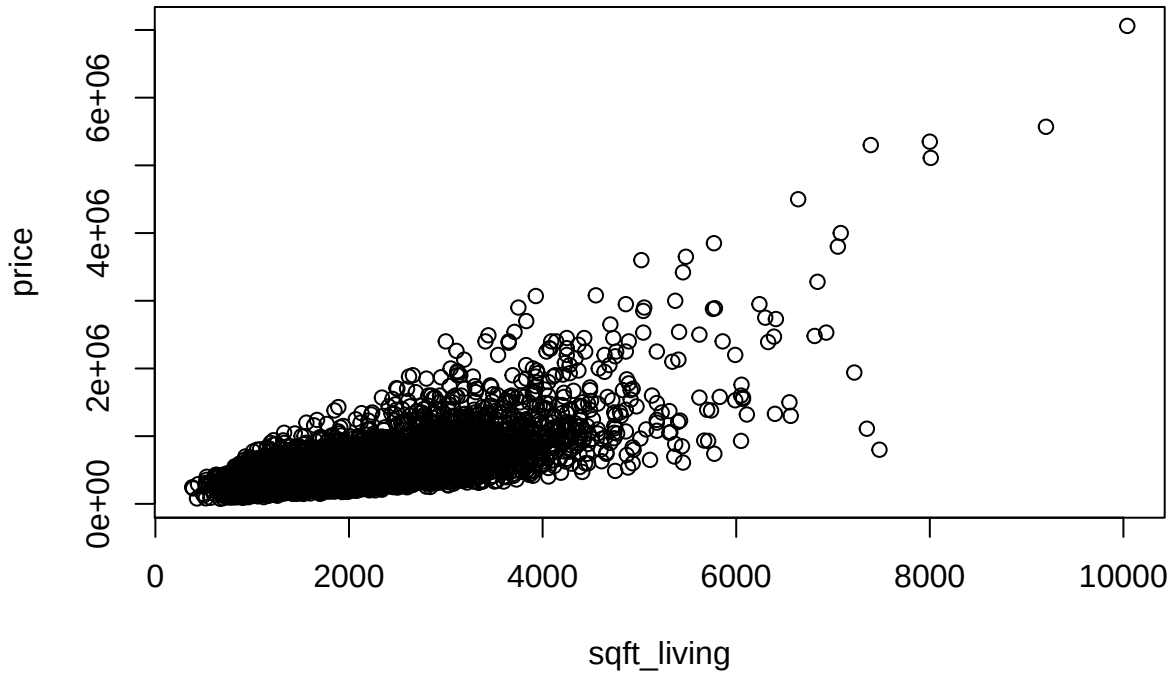


For these three scatterplots, I do not see any good and obvious linear relationships in all of them. Hence, I do not decide to add them into my model.

Overall, I only pick bedrooms, bathrooms, sqft_living, waterfront, and grade as my predictors.

(b)

```
plot(price ~ sqft_living, train)
```



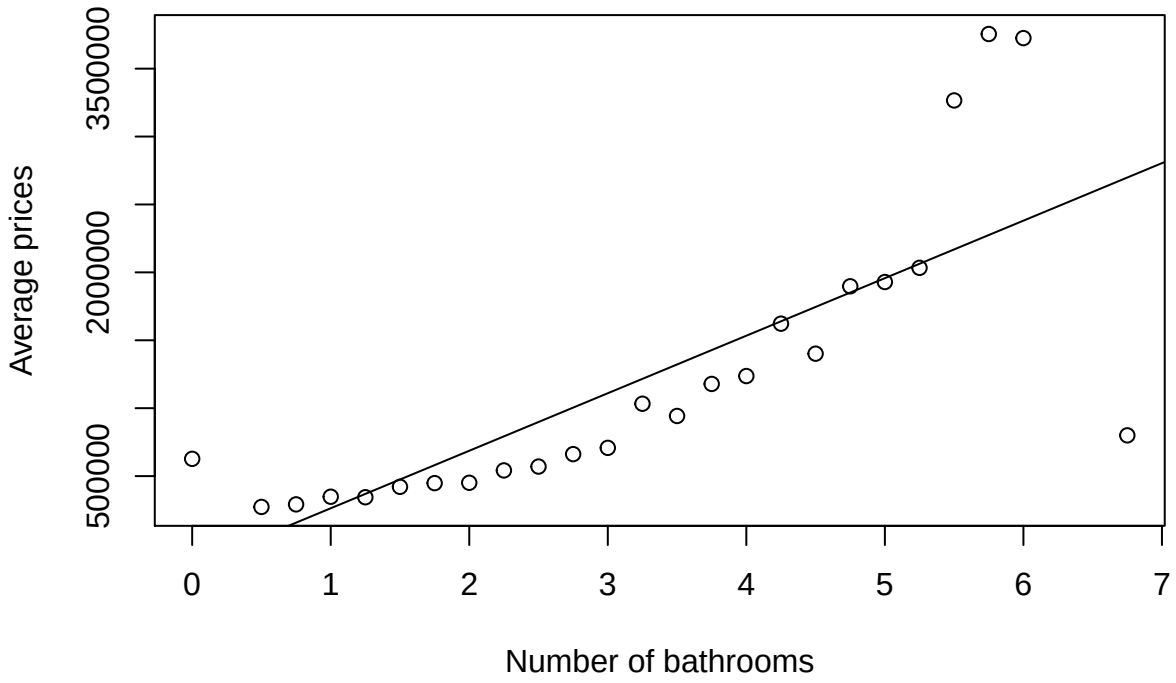
The simple linear model does not catch the some of the pattern displayed in scatterplot. In addition, if we fit a linear model based on that, the normality of resdiuals cannot meet either. Also, since only one predictors are involved within the model, hence it has a high bias. We need to either add more predictors or add some synthetic predictors.

(c)

```
(avg <- tapply(train$price, train$bathrooms, mean))
```

```
##         0       0.5      0.75         1      1.25       1.5      1.75
##  627000.0  273000.0  291852.0  348101.1  344166.7  420897.4  448190.7
##         2      2.25       2.5      2.75         3      3.25       3.5
##  450882.8  541800.5  570163.9  661709.9  707950.7 1032751.4  942780.4
##      3.75         4      4.25       4.5      4.75         5      5.25
## 1178598.9 1236557.5 1622204.3 1401403.8 1897490.0 1929000.0 2033375.0
##       5.5      5.75         6      6.75
## 3266000.0 3755000.0 3725000.0  800000.0
```

```
bath <- as.numeric(names(avg))
plot(avg ~ bath, xlab = "Number of bathrooms", ylab = "Average prices")
m1 <- lm(avg ~ bath)
abline(m1)
```



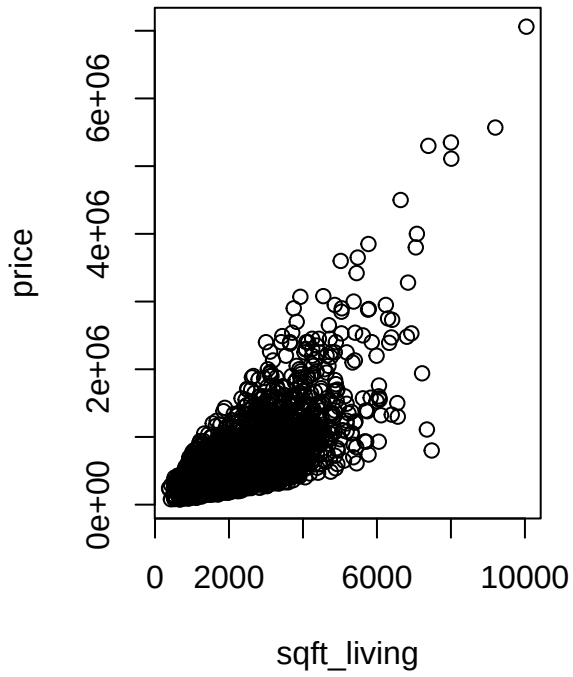

```
summary(m1)
```

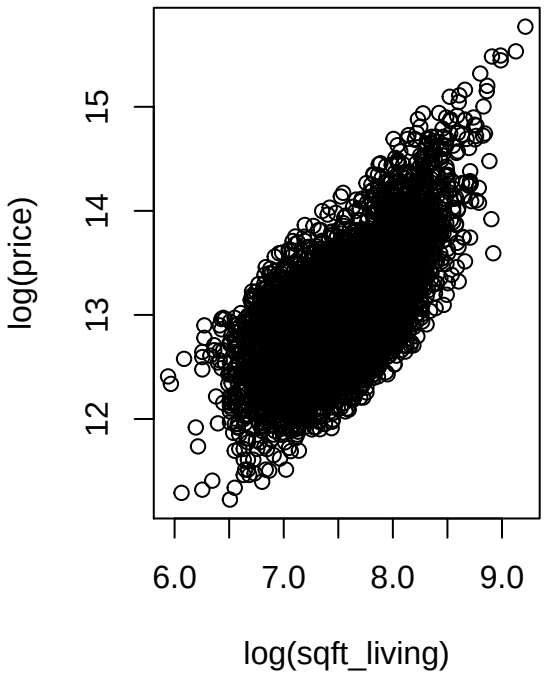
```
##
## Call:
## lm(formula = avg ~ bath)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1898411  -297102   -53896    85080  1480135
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -160525     272889  -0.588    0.562
## bath          423546      72751   5.822 6.23e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 674900 on 23 degrees of freedom
## Multiple R-squared:  0.5957, Adjusted R-squared:  0.5782
## F-statistic: 33.89 on 1 and 23 DF,  p-value: 6.227e-06
```

(d)

(1)

```
par(mfrow = c(1, 2))
plot(price ~ sqft_living, train)
plot(log(price) ~ log(sqft_living), train)
```





In terms of plot based on model 1, we see that there are a lot of points in the far right sides away from most of points. However, in plot based on model 2, we can see that the data points are not as sparse as first plot and the plot shows better data concentration by scaling the response and the predictor, I will pick model 2 since it fits the linearity.

(2) Model 1

```
M1 <- lm(price ~ sqft_living, train)
coef(M1)
```

```
## (Intercept) sqft_living
## -58066.2587    288.0945
```

```
summary(M1)$r.squared
```

```
## [1] 0.4913359
```

The estimate of β_0 is -58066.2587 and the estimate of β_1 is 288.0945 . The R^2 is 0.4913359.

Model 2

```
M2 <- lm(log(price) ~ log(sqft_living), train)
(C2 <- coef(M2))
```

```
##      (Intercept) log(sqft_living)
##        6.6529609        0.8464942
```

```
summary(M2)$r.squared
```

```
## [1] 0.4531333
```

The estimate of β_0 is 6.6529609 and the estimate of β_1 is 0.8464942 . The R^2 is 0.4531333.

(3) My strategy is to use the estimatted coefficients to fit the test data set in order to see how well the models predict. The detailed process are displayed below codes. $(1/n) * \sum(y_i - \hat{\beta}_0 - \hat{\beta}_1 * x_i)^2$

for model 1

```
fitted <- -58066.2587 + 288.0945 * test$sqft_living
mean((test$price - fitted)^2)
```

```
## [1] 68440821693
```

for model 2

```
fitted <- exp(6.6529609 + 0.8464942 * log(test$sqft_living))
mean((test$price - fitted)^2)
```

```
## [1] 77067316693
```

Since the error in sample prediction from model 1 is smaller than model 2, therefore Model 1 is better than Model 2.

(e)

```
best <- function(train, test, train_pred, test_pred) {
    m <- lm(log(price) ~ train_pred, train)
    c <- coef(m)
    fitted <- c[1] + c[2] * test_pred
    sum((log(test$price) - fitted)^2)
}
```

Bedrooms

```
best(train, test, train$bedrooms, test$bedrooms)
```

```
## [1] 1565.702
```

Bathrooms

```
best(train, test, train$bathrooms, test$bathrooms)
```

```
## [1] 1204.354
```

log(sqft_living)

```
best(train, test, log(train$sqft_living), log(test$sqft_living))
```

```
## [1] 935.1893
```

log(sqft_lot)

```
best(train, test, log(train$sqft_lot), log(test$sqft_lot))
```

```
## [1] 1755.586
```

floor

```
best(train, test, train$floors, test$floors)
```

```
## [1] 1670.104
```

waterfront

```
best(train, test, train$waterfront, test$waterfront)
```

```
## [1] 1732.977
```

view

```
best(train, test, train$view, test$view)
```

```
## [1] 1571.371
```

condition

```
best(train, test, train$condition, test$condition)
```

```
## [1] 1781.551
```

grade

```
best(train, test, train$grade, test$grade)
```

```
## [1] 847.0589
```

yr_built

```
best(train, test, train$yr_built, test$yr_built)
```

```
## [1] 1750.316
```

lat

```
best(train, test, train$lat, test$lat)
```

```
## [1] 1439.953
```

long

```
best(train, test, train$long, test$long)
```

```
## [1] 1767.817
```

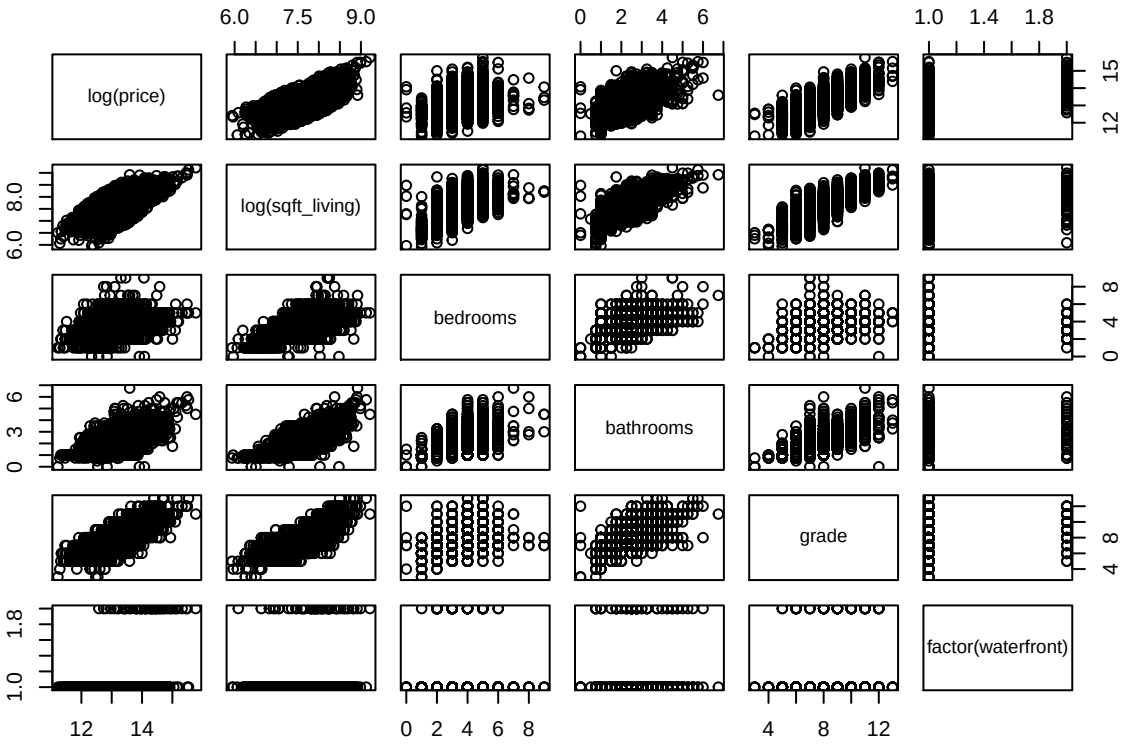
Overall, the SEE from model based on predictor:grade is the smallest one which is 847.0589. Hence the best predictor among them is grade.

(f)

(1) Stratification: since waterfront has two values 0 and 1, then if one row(one onservation) whose waterfront is 0 belong to stratum 0, otherwise, the row belongs to stratum 1.By using this algorithm, the dataframe is automatically separated into 2 stratas. Furthermore, we can take the regression on each of stratas. In terms of stratification method, we have a larger variance and a smaller bias compared with Model 3.

(2)

```
pairs(~log(price) + log(sqft_living) + bedrooms + bathrooms +
    grade + factor(waterfront), train)
```



Based on above matrix of scatterplot, all of predictors: bedrooms, bathrooms, and grade are useful since all of them look like have linear relationships with log(price) We can even make the three-way interaction among bedrooms, bathrooms, and grade in order to imporve the performance of model. Also, we may can use polynomial methods on these useful predictors. The last but not the least, we may factor these three variables.

(3)

```
M3 <- lm(log(price) ~ log(sqft_living) + bedrooms + bathrooms +
    grade + factor(waterfront), train)
(C3 <- coef(M3))
```

```
##         (Intercept)    log(sqft_living)            bedrooms
##         8.048814659         0.483182031        -0.023573975
##           bathrooms               grade factor(waterfront)1
##        -0.008814917         0.188826294         0.742457111
```

```
summary(M3)$r.squared
```

```
## [1] 0.5534403
```

In model 3, the $R^2 = 0.5534403$. In model 2, the R^2 is 0.4531333. Hence, obviously R^2 from model 3 is larger than R^2 from model 2.

(4) Model 2 has a higher bias. Model 3 has a higher variance.

(5) Like method I have used before, basicly I used the train dataset to train the model. Then I apply estimated coefficient from training data to test data. I am trying to see which model predicts test data better.

Model 3

```
fitted <- C3[1] + C3[2] * log(test$sqft_living) + C3[3] * test$bedrooms +
    C3[4] * test$bathrooms + C3[5] * test$grade + C3[6] * test$waterfront
mean((log(test$price) - fitted)^2)
```

```
## [1] 0.1135075
```

Model 2

```
fitted <- C2[1] + C2[2] * log(test$sqft_living)
mean((log(test$price) - fitted)^2)
```

```
## [1] 0.1455095
```

Based above quantative analysis, MSE from Model 3 is samller than the one from Model 2. Therefore, I reach the conclusion that Model 3 makes better predictions than Model 2.

(6)

```
confint(M3, level = 0.9)[-1, ]
```

```
##                               5 %         95 %
## log(sqft_living)     0.44999241  0.516371651
## bedrooms            -0.03370421 -0.013443741
## bathrooms           -0.02373650  0.006106663
## grade                0.17951166  0.198140926
## factor(waterfront)1  0.66489126  0.820022962
```

(7) Based on Bonferroni's approach, the confidence interval for each of β_1 and β_2 is 97.5%.

First way:

```
confint(M3, level = 1 - 0.05/2)[2:3, ]
```

```
##                        1.25 %      98.75 %
## log(sqft_living)  0.43795163  0.528412435
## bedrooms         -0.03737933 -0.009768618
```

Second way:

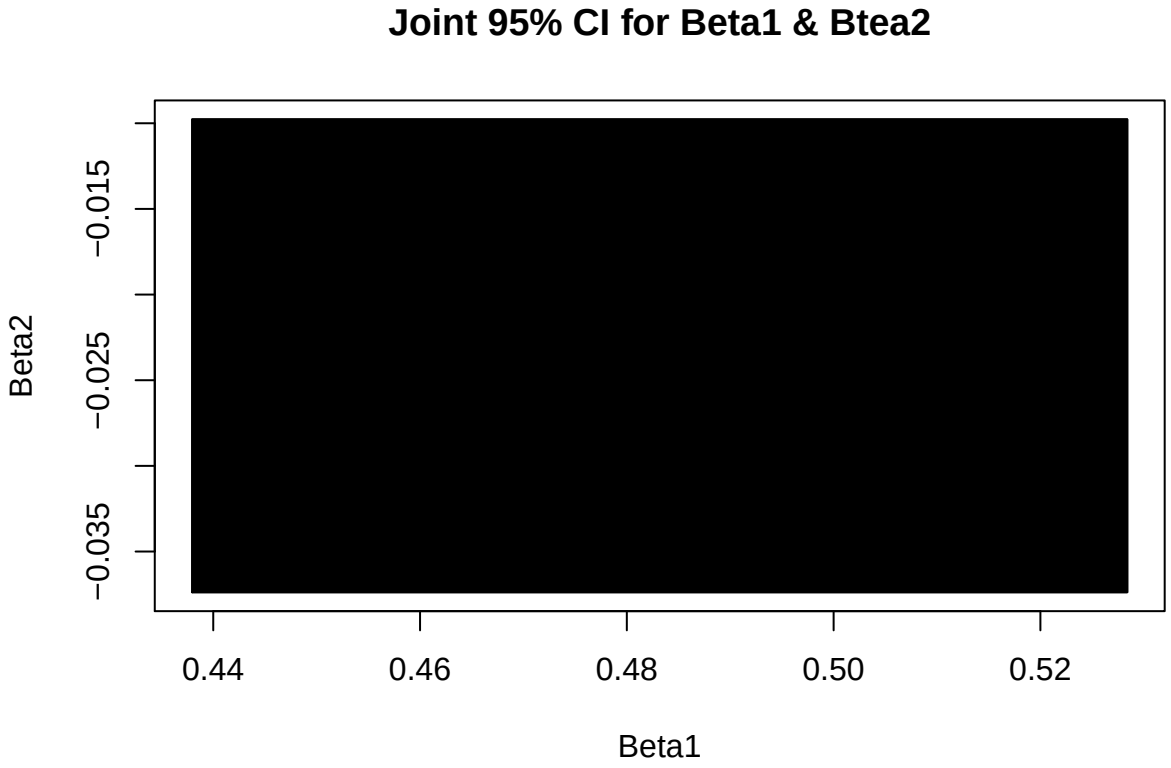
```
cutoff <- qnorm(2.5 * 0.01/2, lower.tail = FALSE)
(CI1 <- 0.483182 + c(-1, 1) * 0.020175 * cutoff)
```

```
## [1] 0.4379617 0.5284023
```

```
(CI2 <- -0.023574 + c(-1, 1) * 0.006158 * cutoff)
```

```
## [1] -0.037376558 -0.009771442
```

```
x <- c(rep(CI1[1], 2), rep(CI1[2], 2), CI1[1])
y <- c(CI2[1], rep(CI2[2], 2), CI2[1], CI2[1])
plot(x, y, type = "l", xlab = "Beta1", ylab = "Beta2", main = "Joint 95% CI for Beta1 & Btea2")
polygon(x, y, col = "black")
```

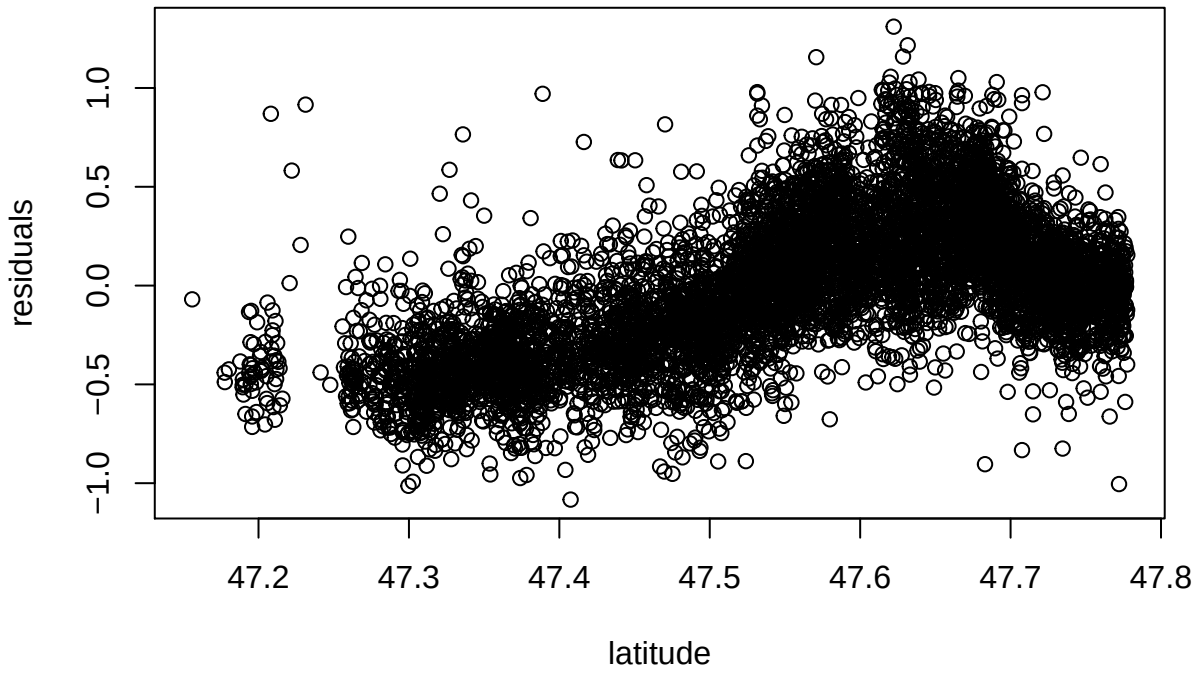


The black area is joint 95% confidence interval for β_1 and β_2

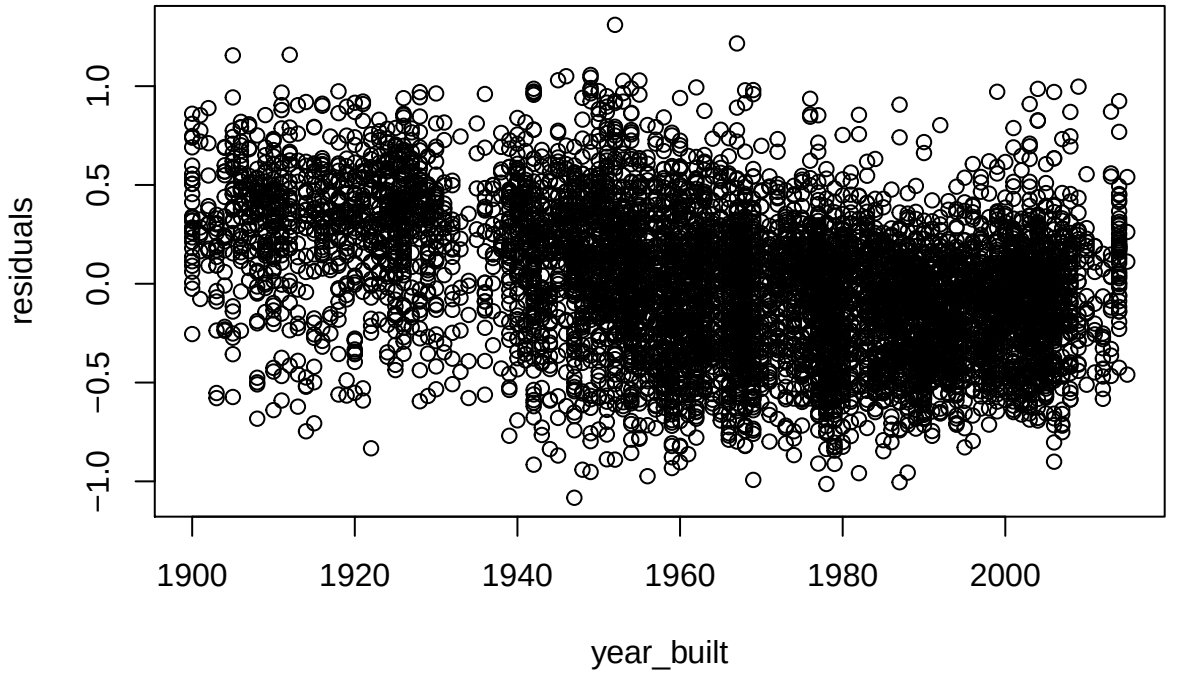
(g) It is a fantastic idea. As we all know, the residual is the amonut that is cannot explained by the current predictors whithin the model. If we plot residual against other predictors and see certain non-random data pattern, it basically means that this new pedictor can help us explain the residual. If we add this new good predictor into our current model, the R^2 will go up and model becomes better than before.

(h) Do several scatterplots to convince myself.

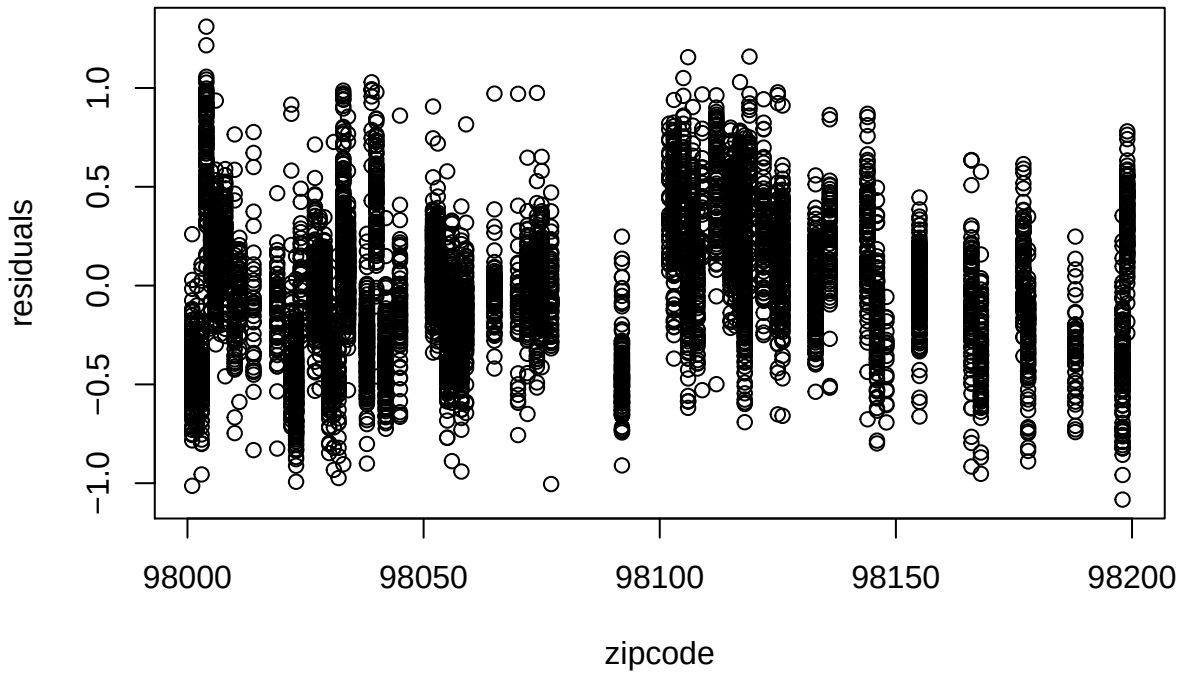
```
residuals <- resid(M3)
plot(residuals ~ train$lat, xlab = "latitude")
```




```
plot(residuals ~ train$yr_built, xlab = "year_built")
```




```
plot(residuals ~ train$zipcode, xlab = "zipcode")
```



(1) My model: log(price)~log(sqft_living) + bedrooms + bathrooms + grade + waterfront + lat + I(lat^2) + yr_built

(2)

```
M4 <- lm(log(price) ~ log(sqft_living) + bedrooms + bathrooms +
    grade + factor(waterfront) + yr_built + lat, train)
(C4 <- coef(M4))
```

```
##         (Intercept)    log(sqft_living)            bedrooms
##       -44.704376429         0.430063127        -0.032191151
##           bathrooms               grade factor(waterfront)1
##         0.088883551         0.195166698         0.689311055
##            yr_built                 lat
##        -0.004746286         1.309303678
```

```
summary(M4)$r.squared
```

```
## [1] 0.7506941
```

The R^2 in model 4 is 0.7506941. The R^2 in model 3 is 0.5534. Hence, obviously R^2 from model 4 has improved a lot from model 3.

(3) Model 3

```
fitted <- C3[1] + C3[2] * log(test$sqft_living) + C3[3] * test$bedrooms +
    C3[4] * test$bathrooms + C3[5] * test$grade + C3[6] * test$waterfront
mean((log(test$price) - fitted)^2)
```

```
## [1] 0.1135075
```

Model 4

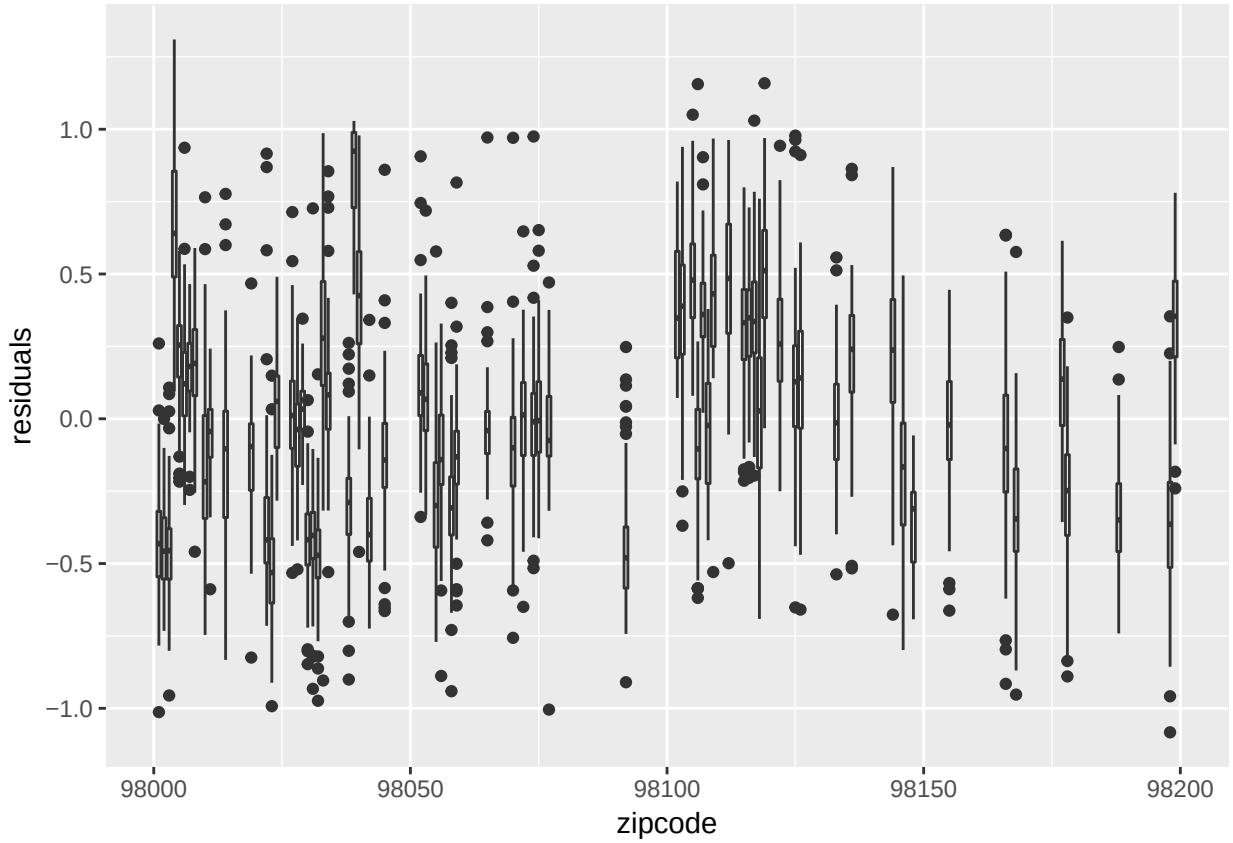
```
fitted <- C4[1] + C4[2] * log(test$sqft_living) + C4[3] * test$bedrooms +
    C4[4] * test$bathrooms + C4[5] * test$grade + C4[6] * test$waterfront +
    C4[7] * test$yr_built + C4[8] * test$lat
mean((log(test$price) - fitted)^2)
```

```
## [1] 0.06818569
```

Based above quantative analysis, MSE from Model 4 is far samller than the one from Model 3. Therefore, I reach the conclusion that Model 4 makes better predictions than Model 3.

(i)

```
library(ggplot2)
ggplot(train, mapping = aes(y = residuals, x = zipcode, group = zipcode)) +
    geom_boxplot()
```



I see the some patterns about residuals against zipcode and I should include zipcode as my new predictors.

```
M5 <- lm(log(price) ~ log(sqft_living) + bedrooms + bathrooms +
    grade + factor(waterfront) + yr_built + lat + factor(zipcode),
    train)
summary(M5)$r.squared
```

```
## [1] 0.8630945
```

The R^2 for Model 5 is 0.8630945

Model 5

```
(val5 <- mean((log(test$price) - predict(M5, test))^2))
```

```
## [1] 0.03686583
```

Model 4

```
mean((log(test$price) - predict(M4, test))^2)
```

```
## [1] 0.06818569
```

Based above quantative analysis, MSE from Model 5 is far samller than the one from Model 4. Therefore, I reach the conclusion that Model 5 makes better predictions than Model 4.

(j)

```
M6 <- lm(log(price) ~ log(sqft_living) + factor(waterfront) +
    bedrooms + bathrooms + grade + yr_built + lat + I(lat^2) +
    long + log(sqft_living15) + log(sqft_lot15) + factor(zipcode) +
    view + condition + yr_renovated + sqft_above + sqft_basement +
    log(sqft_lot) + floors, train)
(val6 <- mean((log(test$price) - predict(M6, test))^2))
```

```
## [1] 0.03112002
```

```
(val6 - val5)/val5
```

```
## [1] -0.1558576
```

Based on above reduction, I have achieved the 15% reduction requirement.